Flatiron School Catalog
California Version 1.0
Catalog published 11.30.2018
Catalog valid through 11.30.2019

All classes will be held at
1460 Mission Street
San Francisco, CA 94103
888-958-0569
Flatironschool.com

Contents

This catalog shall be made available to prospective students and other interested parties via download at the institution's website. In addition, this catalog shall be provided, upon request, to prospective students and other interested parties via email in .pdf format.

GENERAL INFORMATION

Flatiron School's Mission Statement

To enable the pursuit of a better life through education.

Values that Support the Mission

Make no little plans
We aim high. There's no plan too ambitious. For us, it's not enough to teach people how to code – we want to change their lives. It's not enough to bring this transformative experience to a select few – we hope to use our technology to make education accessible to all, regardless of background. We're building for the whole world.

How? Flatiron School creates an education platform that allows people across the globe to learn to code.

Work together
Innovation happens at the intersection of technology and creativity. At Flatiron School, we study technology and inspire creativity by seeking a diversity of perspective, informed by race, gender, life experience, and passion. We are PhDs and college dropouts; poets and analysts; athletes and activists. And together, we are greater than the sum of our parts.

How? Flatiron School collaborates with other institutions to find new ways to bring diversity to tech.

Pursue Mastery
We are all beginners. At Flatiron School, we learn with intent. We find the minimum bar and greatly exceed it. We see errors as opportunities and believe that grit can bring talent to its knees.

How? Flatiron School developers, instructors, and students never stop learning and challenge themselves to present at tech conferences around the world.

Be scrappy
To match the pace of technology, we move fast. This means embracing change, not running from it, and thriving in ambiguity. We jump across department lines and actively look for problems to solve. Put simply: we get stuff done.

How? Our employees jump at the chance to grow and reshape their roles – marketers have become developers; developers have become instructors.

Radiate positivity
We aim to actively radiate positivity. To improve the world around us in every capacity. To leave every interaction, big or small, better. We find what to love in every pursuit and leverage our passion as a force to motivate others.

How? With passionate instructors and inspiring guest speakers, we make tech a more welcoming space and breed optimism for the future.

History and Vision of Flatiron School
Education used to be the best investment one could make. It can be again.

It's that shared belief that brought together a venture capitalist with a passion for education and a self-taught technologist bent on helping his students reinvent themselves as programmers. In 2012, Adam Enbar and Avi Flombaum partnered to create an alternative to an education industry leaving a wide skills gap in a booming tech world. Alongside a scrappy community of students ready for a new way to learn, they launched Flatiron School – an accelerated programming school that inspired a coding industry.

In 2015, we took our classroom online with Learn.co, bringing the ability to change your career through code to those previously left out of the model – people holding full-time jobs, parents balancing packed schedules, students hundreds of miles from the nearest tech hub.

Flatiron School is financially solvent and possesses the adequate resources to achieve its mission and goals; it also has the financial solvency to launch a new campus in California. We at Flatiron School have intentionally delayed expansion until we had the infrastructure in place and could support a new campus with the high-quality instruction and student services that we are known for providing to our students from our first campus in New York City.

In October 2017, Flatiron School was acquired by WeWork Companies, Inc. WeWork provides workspace, community, and services for entrepreneurs, freelancers, startups, small businesses, and enterprise companies. WeWork is well capitalized, financially solvent, and possesses the adequate resources, mission and goals. WeWork intends to deploy capital and resources to ensure the success of Flatiron School's expansion into California.

Technology has the potential to reshape the way we learn – and we're just getting started.

PROGRAM AND COURSE INFORMATION

Program and Course Offerings
Programs (Full-Time)
- Software Engineering
- Full-Time Online Software Engineering

- Self-Paced Online Software Engineering
- Data Science
- Full-Time Online Data Science
- Self-Paced Online Data Science
- UX/UI Design

Programs (Part-Time)

- Part-Time Online Software Engineering
- Part-Time Online Data Science

Courses (Part-Time)

- Intro to Front End Web Development
- Intro to Data Science

Required Equipment to be Supplied by the Student

Students must have access to a laptop to bring to each class that is running on the most current operating system. For questions about the suitability of technical equipment, please contact: info@flatironschool.com.

Language of Instruction

All instruction will take place in English. Students must have native or bilingual fluency in English in order to succeed in the program. Proficiency in English will be assessed during the application process. Flatiron School does not accept the TOEFL or any other proficiency documentation and does not provide ESL services.

Requirements for Graduation

Software Engineering and Data Science: Students who meet the below requirements will have completed their course of study and will be awarded a Certificate of Completion:

- Attend eighty-five percent (85%) of all class hours
- Pass all assessments
- Complete 95% of all lessons
- Publish seven blog posts
- Pass the final exam
- Return all loaner equipment (if applicable)
- Show a zero balance on their student account

Online Software Engineering and Online Data Science: Students who meet the below requirements will have completed their course of study and will be awarded a Certificate of Completion:

- Pass all assessments
- Return all loaner equipment (if applicable)
- Show a zero balance on their student account

UX/UI Design: Students who meet the below requirements will have completed their course of study and will be awarded a Certificate of Completion:

- Attend eighty-five percent (85%) of all class hours
- Pass all assessments
- Complete 95% of all lessons
- Submit all final project and client deliverables on time
- Return all loaner equipment (if applicable)
- Show a zero balance on their student account

Intro to Data Science and Intro to Front End Web Development: Students who meet the below requirements will have completed their course of study and will be awarded a Certificate of Completion:

- Attend eighty-five percent (85%) of all class hours
- Pass all assessments
- Return all loaner equipment (if applicable)
- Show a zero balance on their student account

PROGRAM: Software Engineering

Educational Objectives and Program Description

The objective of this program is to prepare students for an entry-level career in Software Engineering. Flatiron School provides its students with foundational knowledge to be flexible and versatile web developers who can quickly learn new languages to keep pace with changing technology. No other software is required for this program.

Students develop a foundation of programming fundamentals and conquer the concepts of object-oriented programming. Students work with APIs (Application Programming Interfaces), become proficient in database modeling and ORM (Object Relational Mapping), understand the concept of MVC the (Model View Controller) Framework, and execute application deployment. Labs are taught using industry-standard test-driven development, allowing students to gain real-world programming experience, while giving instructors the ability to check on student progress with ease. After completing each module, students are required to complete a module assessment. This is an open-ended project that students complete and then debrief with instructors to assess their ability to code and explain their code.

Due to the nature of the technology industry this program focuses heavily on teaching students how to learn new skills on their own. Software development moves too quickly for any skill to remain relevant forever, and therefore students are required to work with their classmates to solve problems and use various online tools to discover new coding techniques that may not be covered in class.

Newly enrolled students must complete the Pre-Work before beginning in-class instruction. The purpose of this preparatory work is to provide real examples of the curriculum so that applicants are fully aware of the demands of the program prior to committing to it financially. The Pre-Work is free and available online: http://prework.flatironschool.com/web-development/

Delivery Method

Traditional classroom

Program Outline

Module Sequence	Module Title	Clock Hours
Module 1	Ruby Basics and Object-Oriented Programming	105
Module 2	Intro to Web Development and Data Persistence	105
Module 3	Advanced Web Development with Rails	105
Module 4	Creating Interactive and Performant Front-Ends with JavaScript	105
Module 5	Scalable Front-End Design with React	105
	Total Clock Hours:	**525**

Module Descriptions

Module 1 – Ruby Basics and Object-Oriented Programming
Clock Hours: 105

Students will begin exploring version control using git commands including with cloning, branching, merging, rolling back commits, forking, and submitting pull requests. We will also learn explore the history of software development from Waterfall to Version control and discuss the historical significance of open sourced tools like Linux.

Students will then learn fundamental concepts in programming including REPLS, methods, loops, variables, variable scope, conditionals, blocks and iterators, case statements, arrays, scope, hashes, regular expressions, iterators, enumerables, data structures, nesting, etc. Topics will build in complexity and provide the foundation for the rest of the course.

Students will learn to embrace error messages as clues and gain a fundamental appreciation for failure as the only way to learn and progress. Students gain experience in debugging with various gems and tools designed to track down issues in code.

After gaining a deep understanding of Ruby basics, students will gain experience with Object-oriented Programming and understand how it allows programmers to bundle code and create reusable objects and methods, allowing for increasing complexity in software. Object-oriented programming is the design pattern that most languages use to create software. While this pattern is learned in Ruby, we re-visit object-oriented design in JavaScript later in the class to ensure that students are learning the pattern of object-oriented programming.

Deliverable: A solid foundation in Ruby, continued experience with git, experience reading and understanding unit tests, enhanced problem-solving skills and an ability to encapsulate code in classes and objects.

Prerequisite: Program Pre-Work

Module 2 – Intro to Web Development and Data Persistence
Clock Hours: 105

Before discussing web development, students must first understand data persistence. Saving data in a structured way that maps to object-oriented design is handled through an object relational mapper (ORM). ORM allows programmers to query and manipulate data from a database using an object-paradigm. Students learn to write manipulate data using the Ruby language. Students will gain an appreciation for the structure of a database, how to map out tables, and the difference between the various table relationships. Students will learn how to wireframe database structures, as well as how to link their applications to a database. This unit covers SQL, domain modeling, relational database theory, schema architecture, and the Object Relational Model, include the ACTIVE Record pattern.

Once students are able to save data to databases, we must cover delivering that data to our users over the web. Before showing students Sinatra, students must first understand how the HyperText Transfer Protocol (HTTP) works. We explain HTTP using the Ruby web interface known as Rack. Students will build their own HTTP Servers and learn how the request/response model of the web works. Their servers will listen to HTTP requests and respond with well-formed HTML responses. They will understand the web with few abstractions provided by the tool set.

After obtaining an understanding of HTTP and Rack, students will learn Sinatra. Sinatra is a Domain Specific Language (DSL) written in Ruby for building web applications on top of Rack. This framework provides students with exposure to design patterns in web applications. The topics covered in this unit include architectural patterns such as REST (Representational State Transfer), MVC (Model-View-Controller), HTML Forms, ERB (Embedded Ruby) and template rendering, and application environments.

Deliverable: A Sinatra powered Ruby web application that processes data from HTML forms and communicates with a model layer to communicate with the database.

Prerequisite: Module A

Module 3 – Advanced Web Development with Rails
Clock Hours: 105

Having a foundation in the Ruby language as well as the architecture of the world wide web, students will use Rails to build complex functional web applications from the ground up. They will learn the file structure of Rails, how to set up their own databases, how to draw routes, create Rails forms, gain an understanding of the asset pipeline, and bring it together by integrating their front-end design skills.

Students will also be capable to take on more advanced concepts such as authorization, validation and callbacks. Once students grasp the basic functionality of Rails, they will spend time building out their own Rails applications, moving through the entire process from idea to execution

Deliverable: A multi-model Ruby on Rails powered web application that integrates nested forms and URL routes to a multi-relational data set.

Prerequisite: Module B

Module 4 – Creating Interactive and Performant Front-Ends with JavaScript
Clock Hours: 105

JavaScript (EMCAScript) powers the user experience of the web. Students will learn the basics of the JavaScript syntax, it's functional architecture, along with a few approaches to the object model. They will then learn the Document Object Model (DOM) JavaScript API provided by the browser to dynamically interact with HTML. This unit will focus on jQuery, the most popular JavaScript library, to aid students in learning how to collect user input, manipulate the DOM with animations and injection, and send Asynchronous AJAX requests for a rich user experience. After understanding jQuery students are given an introduction into front-end frameworks such as Backbone and Ember.js. These frameworks allow students to write larger, more organized code-bases.

Deliverable: A Rails application that includes examples of DOM Manipulation, AJAX HTML Loading, and AJAX Form Submissions.

Prerequisite: Module C

Module 5 – Scalable Front-End Design with React
Clock Hours: 105

Building complex front ends with plain JavaScript doesn’t work as applications get more complex. Just as object-oriented design was created to help large teams build incredibly complex back end apps, front end frameworks were created to provide the structure needed to build complex, dynamic front ends. This module covers the React framework by Google. Continuing from our JavaScript module, students build a simple version of React. JavaScript frameworks change quite frequently so understanding the fundamentals of structured JavaScript is crucial to remaining relevant throughout a web developer’s career. Building a simple React clone before using it ensures students understanding of the fundamentals.

After building all of the base components that make up React, students are then introduced to using React throughout their app. First, we discuss how to make reusable user interface components and handle user input and display. After being able to receive data from the user and display it, we then cover how to send that data to the back end using $http and services. Finally, students learn how to handle asynchronous function calls using promises.

Deliverable: A multi-model Ruby on Rails powered backend that serves JSON to a multi-page React front end.

Prerequisite: Module D

PROGRAM: Full-Time Online Software Engineering

Educational Objectives and Program Description

The objective of this program is to prepare students for an entry-level career in Software Engineering. Flatiron School provides its students with foundational knowledge to be flexible and versatile web developers who can quickly learn new languages to keep pace with changing technology. No other software is required for this program.

Students develop a foundation of programming fundamentals and conquer the concepts of object-oriented programming. Students work with APIs (Application Programming Interfaces), become proficient in database modeling and ORM (Object Relational Mapping), understand the concept of MVC the (Model View Controller) Framework, and execute application deployment. Labs are taught using industry-standard test-driven development, allowing students to gain real-world programming experience, while giving instructors the ability to check on student progress with ease. After completing each module, students are required to complete a module assessment. This is an open-ended project that students complete and then debrief with instructors to assess their ability to code and explain their code.

Due to the nature of the technology industry this program focuses heavily on teaching students how to learn new skills on their own. Software development moves too quickly for any skill to remain relevant forever, and therefore students are required to work with their classmates to solve problems and use various online tools to discover new coding techniques that may not be covered in class.

Delivery Method

Online. Over the course of 5 months, students regularly engage with their instructors and fellow students. Students view live lectures delivered by instructor and complete weekly assignments. Weekly assignments are reviewed and returned to the student within 2 days. Students also virtually meet with their instructor weekly to break down tough technical concepts. Students also meet with Educational Coaches ten times throughout the program to help students adopt behaviors that foster the successful and timely completion of the program. Students interact with members of their cohort in weekly lab assignments and study groups. For lab assignments, students are paired to work on lab assignments at the beginning of each week.

The program consists of five modules as outlined below. Each module concludes with a comprehensive portfolio project meant to demonstrate student learning. Instructors virtually meet with each student individually to discuss their portfolio projects and ensure the student is ready to progress to the next module.

Communication and Technical Support

Students have unrestricted access to the program curriculum and online community via the Learn.co platform while enrolled (24 hours per day, 7 days per week). Flatiron's team of Technical Coaches via the Learn.co platform's Ask a Question Feature are also available to provide live support for students Monday-Friday, 9AM-1AM ET, and Saturday-Sunday, 9AM-12AM ET. Flatiron also maintains a collection of help articles, with advice and answers to frequently asked questions from the Flatiron School Team at help.learn.co.

Program Outline

Module Sequence	Module Title	Clock Hours
Online Module 1	Ruby Basics and Object-Oriented Programming	105
Online Module 2	Intro to Web Development and Data Persistence	105
Online Module 3	Advanced Web Development with Rails	105
Online Module 4	Creating Interactive and Performant Front-Ends with JavaScript	105
Online Module 5	Scalable Front-End Design with React	105
	Total Clock Hours:	**525**

Module Descriptions

Online Module 1 – Ruby Basics and Object-Oriented Programming
Clock Hours: 105

Students will begin exploring version control using git commands including with cloning, branching, merging, rolling back commits, forking, and submitting pull requests. We will also learn explore the history of software development from Waterfall to Version control and discuss the historical significance of open sourced tools like Linux.

Students will then learn fundamental concepts in programming including REPLS, methods, loops, variables, variable scope, conditionals, blocks and iterators, case statements, arrays, scope, hashes, regular expressions, iterators, enumerables, data structures, nesting, etc. Topics will build in complexity and provide the foundation for the rest of the course.

Students will learn to embrace error messages as clues and gain a fundamental appreciation for failure as the only way to learn and progress. Students gain experience in debugging with various gems and tools designed to track down issues in code.

After gaining a deep understanding of Ruby basics, students will gain experience with Object-oriented Programming and understand how it allows programmers to bundle code and create reusable objects and methods, allowing for increasing complexity in software. Object-oriented programming is the design pattern that most languages use to create software. While this pattern is learned in Ruby, we re-visit object-oriented design in JavaScript later in the class to ensure that students are learning the pattern of object-oriented programming.

Deliverable: A solid foundation in Ruby, continued experience with git, experience reading and understanding unit tests, enhanced problem-solving skills and an ability to encapsulate code in classes and objects.

Prerequisite: None

Online Module 2 – Intro to Web Development and Data Persistence
Clock Hours: 105

Before discussing web development, students must first understand data persistence. Saving data in a structured way that maps to object-oriented design is handled through an object relational mapper (ORM). ORM allows programmers to query and manipulate data from a database using an object-paradigm. Students learn to write manipulate data using the Ruby language. Students will gain an appreciation for the structure of a database, how to map out tables, and the difference between the various table relationships. Students will learn how to wireframe database structures, as well as how to link their applications to a database. This unit covers SQL, domain modeling, relational database theory, schema architecture, and the Object Relational Model, include the ACTIVE Record pattern.

Once students are able to save data to databases, we must cover delivering that data to our users over the web. Before showing students Sinatra, students must first understand how the HyperText Transfer Protocol (HTTP) works. We explain HTTP using the Ruby web interface known as Rack. Students will build their own HTTP Servers and learn how the request/response model of the web works. Their servers will listen to HTTP requests and respond with well-formed HTML responses. They will understand the web with few abstractions provided by the tool set.

After obtaining an understanding of HTTP and Rack, students will learn Sinatra. Sinatra is a Domain Specific Language (DSL) written in Ruby for building web applications on top of Rack. This framework provides students with exposure to design patterns in web applications. The topics covered in this unit include architectural patterns such as REST (Representational State Transfer), MVC (Model-View-Controller), HTML Forms, ERB (Embedded Ruby) and template rendering, and application environments.

Deliverable: A Sinatra powered Ruby web application that processes data from HTML forms and communicates with a model layer to communicate with the database.

Prerequisite: Online Module 1

Online Module 3 – Advanced Web Development with Rails
Clock Hours: 105

Having a foundation in the Ruby language as well as the architecture of the world wide web, students will use Rails to build complex functional web applications from the ground up. They will learn the file structure of Rails, how to set up their own databases, how to draw routes, create Rails forms, gain an understanding of the asset pipeline, and bring it together by integrating their front-end design skills.

Students will also be capable to take on more advanced concepts such as authorization, validation and callbacks. Once students grasp the basic functionality of Rails, they will spend time building out their own Rails applications, moving through the entire process from idea to execution

Deliverable: A multi-model Ruby on Rails powered web application that integrates nested forms and URL routes to a multi-relational data set.

Prerequisite: Online Module 2

Online Module 4 – Creating Interactive and Performant Front-Ends with JavaScript
Clock Hours: 105

JavaScript (EMCAScript) powers the user experience of the web. Students will learn the basics of the JavaScript syntax, it's functional architecture, along with a few approaches to the object model. They will then learn the Document Object Model (DOM) JavaScript API provided by the browser to dynamically interact with HTML. This unit will focus on jQuery, the most popular JavaScript library, to aid students in learning how to collect user input, manipulate the DOM with animations and injection, and send Asynchronous AJAX requests for a rich user experience. After understanding jQuery students are given an introduction into front-end frameworks such as Backbone and Ember.js. These frameworks allow students to write larger, more organized code-bases.

Deliverable: A Rails application that includes examples of DOM Manipulation, AJAX HTML Loading, and AJAX Form Submissions.

Prerequisite: Online Module 3

Online Module 5 – Scalable Front-End Design with React
Clock Hours: 105

Building complex front ends with plain JavaScript doesn't work as applications get more complex. Just as object-oriented design was created to help large teams build incredibly complex back end apps, front end frameworks were created to provide the structure needed to build complex, dynamic front ends. This module covers the React framework by Google. Continuing from our JavaScript module, students build a simple version of React. JavaScript frameworks change quite frequently so understanding the fundamentals of structured JavaScript is crucial to remaining relevant throughout a web developer's career. Building a simple React clone before using it ensures students understanding of the fundamentals.

After building all of the base components that make up React, students are then introduced to using React throughout their app. First, we discuss how to make reusable user interface components and handle user input and display. After being able to receive data from the user and display it, we then cover how to send that data to the back end using $http and services. Finally, students learn how to handle asynchronous function calls using promises.

Deliverable: A multi-model Ruby on Rails powered backend that serves JSON to a multi-page React front end.

Prerequisite: Online Module 4

PROGRAM: Part-Time Online Software Engineering

Educational Objectives and Program Description

The objective of this program is to prepare students for an entry-level career in Software Engineering. Flatiron School provides its students with foundational knowledge to be flexible and versatile web developers who can quickly learn new languages to keep pace with changing technology. No other software is required for this program.

Students develop a foundation of programming fundamentals and conquer the concepts of object-oriented programming. Students work with APIs (Application Programming Interfaces), become proficient in database modeling and ORM (Object Relational Mapping), understand the concept of MVC the (Model View Controller) Framework, and execute application deployment. Labs are taught using industry-standard test-driven development, allowing students to gain real-world programming experience, while giving instructors the ability to check on student progress with ease. After completing each module, students are required to complete a module assessment. This is an open-ended project that students complete and then debrief with instructors to assess their ability to code and explain their code.

Due to the nature of the technology industry this program focuses heavily on teaching students how to learn new skills on their own. Software development moves too quickly for any skill to remain relevant forever, and therefore students are required to work with their classmates to solve problems and use various online tools to discover new coding techniques that may not be covered in class.

Delivery Method

Online. Over the course of 10 months, students regularly engage with their instructors and fellow students. Students view live lectures delivered by their instructor and complete weekly assignments. Weekly assignments are reviewed and returned to the student within 2 days. Students also virtually meet with their instructor weekly to break down tough technical concepts. Students also meet with Educational Coaches ten times throughout the program to help students adopt behaviors that foster the successful and timely completion of the program. Students interact with members of their cohort in weekly lab assignments and study groups. For lab assignments, students are paired to work on lab assignments at the beginning of each week.

The program consists of five modules as outlined below. Each module concludes with a comprehensive portfolio project meant to demonstrate student learning. Instructors virtually meet with each student individually to discuss their portfolio projects and ensure the student is ready to progress to the next module.

Communication and Technical Support

Students have unrestricted access to the program curriculum and online community via the Learn.co platform while enrolled (24 hours per day, 7 days per week). Flatiron's team of Technical Coaches via the Learn.co platform's Ask a Question Feature are also available to provide live support for students Monday-Friday, 9AM-1AM ET, and Saturday-Sunday, 9AM-12AM ET. Flatiron also maintains a collection of help articles, with advice and answers to frequently asked questions from the Flatiron School Team at help.learn.co.

Program Outline

Module Sequence	Module Title	Clock Hours
Online Module 1	Ruby Basics and Object-Oriented Programming	105
Online Module 2	Intro to Web Development and Data Persistence	105
Online Module 3	Advanced Web Development with Rails	105
Online Module 4	Creating Interactive and Performant Front-Ends with JavaScript	105

Online Module 5	Scalable Front-End Design with React	105
	Total Clock Hours:	**525**

Module Descriptions

Online Module 1 – Ruby Basics and Object-Oriented Programming
Clock Hours: 105

Students will begin exploring version control using git commands including with cloning, branching, merging, rolling back commits, forking, and submitting pull requests. We will also learn explore the history of software development from Waterfall to Version control and discuss the historical significance of open sourced tools like Linux.

Students will then learn fundamental concepts in programming including REPLS, methods, loops, variables, variable scope, conditionals, blocks and iterators, case statements, arrays, scope, hashes, regular expressions, iterators, enumerables, data structures, nesting, etc. Topics will build in complexity and provide the foundation for the rest of the course.

Students will learn to embrace error messages as clues and gain a fundamental appreciation for failure as the only way to learn and progress. Students gain experience in debugging with various gems and tools designed to track down issues in code.

After gaining a deep understanding of Ruby basics, students will gain experience with Object-oriented Programming and understand how it allows programmers to bundle code and create reusable objects and methods, allowing for increasing complexity in software. Object-oriented programming is the design pattern that most languages use to create software. While this pattern is learned in Ruby, we re-visit object-oriented design in JavaScript later in the class to ensure that students are learning the pattern of object-oriented programming.

Deliverable: A solid foundation in Ruby, continued experience with git, experience reading and understanding unit tests, enhanced problem-solving skills and an ability to encapsulate code in classes and objects.

Prerequisite: None

Online Module 2 – Intro to Web Development and Data Persistence
Clock Hours: 105

Before discussing web development, students must first understand data persistence. Saving data in a structured way that maps to object-oriented design is handled through an object relational mapper (ORM). ORM allows programmers to query and manipulate data from a database using an object-paradigm. Students learn to write manipulate data using the Ruby language. Students will gain an appreciation for the structure of a database, how to map out tables, and the difference between the various table relationships. Students will learn how to wireframe database structures, as well as how to link their applications to a database. This unit covers SQL, domain modeling, relational database theory, schema architecture, and the Object Relational Model, include the ACTIVE Record pattern.

Once students are able to save data to databases, we must cover delivering that data to our users over the web. Before showing students Sinatra, students must first understand how the HyperText Transfer Protocol (HTTP) works. We explain HTTP using the Ruby web interface known as Rack. Students will build their own HTTP Servers and learn how the request/response model of the web works. Their servers will listen to HTTP requests and respond with well-formed HTML responses. They will understand the web with few abstractions provided by the tool set.

After obtaining an understanding of HTTP and Rack, students will learn Sinatra. Sinatra is a Domain Specific Language (DSL) written in Ruby for building web applications on top of Rack. This framework provides students with exposure to design patterns in web applications. The topics covered in this unit include architectural patterns such as REST (Representational State Transfer), MVC (Model-View-Controller), HTML Forms, ERB (Embedded Ruby) and template rendering, and application environments.

Deliverable: A Sinatra powered Ruby web application that processes data from HTML forms and communicates with a model layer to communicate with the database.

Prerequisite: Online Module 1

Online Module 3 – Advanced Web Development with Rails
Clock Hours: 105

Having a foundation in the Ruby language as well as the architecture of the world wide web, students will use Rails to build complex functional web applications from the ground up. They will learn the file structure of Rails, how to set up their own databases, how to draw routes, create Rails forms, gain an understanding of the asset pipeline, and bring it together by integrating their front-end design skills.

Students will also be capable to take on more advanced concepts such as authorization, validation and callbacks. Once students grasp the basic functionality of Rails, they will spend time building out their own Rails applications, moving through the entire process from idea to execution

Deliverable: A multi-model Ruby on Rails powered web application that integrates nested forms and URL routes to a multi-relational data set.

Prerequisite: Online Module 2

Online Module 4 – Creating Interactive and Performant Front-Ends with JavaScript
Clock Hours: 105

JavaScript (EMCAScript) powers the user experience of the web. Students will learn the basics of the JavaScript syntax, it's functional architecture, along with a few approaches to the object model. They will then learn the Document Object Model (DOM) JavaScript API provided by the browser to dynamically interact with HTML. This unit will focus on jQuery, the most popular JavaScript library, to aid students in learning how to collect user input, manipulate the DOM with animations and injection, and send Asynchronous AJAX requests for a rich user experience. After understanding jQuery students are given an introduction into front-end frameworks such as Backbone and Ember.js. These frameworks allow students to write larger, more organized code-bases.

Deliverable: A Rails application that includes examples of DOM Manipulation, AJAX HTML Loading, and AJAX Form Submissions.

Prerequisite: Online Module 3

Online Module 5 – Scalable Front-End Design with React
Clock Hours: 105

Building complex front ends with plain JavaScript doesn't work as applications get more complex. Just as object-oriented design was created to help large teams build incredibly complex back end apps, front end frameworks were created to provide the structure needed to build complex, dynamic front ends. This module covers the React framework by Google. Continuing from our JavaScript module, students build a simple version of React. JavaScript frameworks change quite frequently so understanding the fundamentals of structured JavaScript is crucial to remaining relevant throughout a web developer's career. Building a simple React clone before using it ensures students understanding of the fundamentals.

After building all of the base components that make up React, students are then introduced to using React throughout their app. First, we discuss how to make reusable user interface components and handle user input and display. After being able to receive data from the user and display it, we then cover how to send that data to the back end using $http and services. Finally, students learn how to handle asynchronous function calls using promises.

Deliverable: A multi-model Ruby on Rails powered backend that serves JSON to a multi-page React front end.

Prerequisite: Online Module 4

PROGRAM: Self-Paced Online Software Engineering

Educational Objectives and Program Description

The objective of this program is to prepare students for an entry-level career in Software Engineering. Flatiron School provides its students with foundational knowledge to be flexible and versatile web developers who can quickly learn new languages to keep pace with changing technology. No other software is required for this program.

Students develop a foundation of programming fundamentals and conquer the concepts of object-oriented programming. Students work with APIs (Application Programming Interfaces), become proficient in database modeling and ORM (Object Relational Mapping), understand the concept of MVC the (Model View Controller) Framework, and execute application deployment. Labs are taught using industry-standard test-driven development, allowing students to gain real-world programming experience, while giving instructors the ability to check on student progress with ease. After completing each module, students are required to complete a module assessment. This is an open-ended project that students complete and then debrief with instructors to assess their ability to code and explain their code.

Due to the nature of the technology industry this program focuses heavily on teaching students how to learn new skills on their own. Software development moves too quickly for any skill to remain relevant forever, and therefore students are required to work with their classmates to solve problems and use various online tools to discover new coding techniques that may not be covered in class.

Delivery Method

Online. Students have up to 15 months to complete this self-paced program. Throughout the program, students have the option to view and participate in live study group discussions or view recorded study group discussions. Students also meet with an Educational Coach five times throughout the program. Educational Coaches help students set goals for the program and ensure students progress through the program at a steady pace.

The program consists of five modules as outlined below. Each module concludes with a comprehensive portfolio project meant to demonstrate student learning. Instructors virtually meet with each student individually to discuss their portfolio projects and ensure the student is ready to progress to the next module.

Communication and Technical Support

Students have unrestricted access to the program curriculum and online community via the Learn.co platform while enrolled (24 hours per day, 7 days per week). Flatiron's team of Technical Coaches via the Learn.co platform's Ask a Question Feature are also available to provide live support for students Monday-Friday, 9AM-1AM ET, and Saturday-Sunday, 9AM-12AM ET. Flatiron also maintains a collection of help articles, with advice and answers to frequently asked questions from the Flatiron School Team at help.learn.co.

Program Outline

Module Sequence	Module Title	Clock Hours

Online Module 1	Ruby Basics and Object-Oriented Programming	105
Online Module 2	Intro to Web Development and Data Persistence	105
Online Module 3	Advanced Web Development with Rails	105
Online Module 4	Creating Interactive and Performant Front-Ends with JavaScript	105
Online Module 5	Scalable Front-End Design with React	105
	Total Clock Hours:	**525**

Module Descriptions

Online Module 1 – Ruby Basics and Object-Oriented Programming
Clock Hours: 105

Students will begin exploring version control using git commands including with cloning, branching, merging, rolling back commits, forking, and submitting pull requests. We will also learn explore the history of software development from Waterfall to Version control and discuss the historical significance of open sourced tools like Linux.

Students will then learn fundamental concepts in programming including REPLS, methods, loops, variables, variable scope, conditionals, blocks and iterators, case statements, arrays, scope, hashes, regular expressions, iterators, enumerables, data structures, nesting, etc. Topics will build in complexity and provide the foundation for the rest of the course.

Students will learn to embrace error messages as clues and gain a fundamental appreciation for failure as the only way to learn and progress. Students gain experience in debugging with various gems and tools designed to track down issues in code.

After gaining a deep understanding of Ruby basics, students will gain experience with Object-oriented Programming and understand how it allows programmers to bundle code and create reusable objects and methods, allowing for increasing complexity in software. Object-oriented programming is the design pattern that most languages use to create software. While this pattern is learned in Ruby, we re-visit object-oriented design in JavaScript later in the class to ensure that students are learning the pattern of object-oriented programming.

Deliverable: A solid foundation in Ruby, continued experience with git, experience reading and understanding unit tests, enhanced problem-solving skills and an ability to encapsulate code in classes and objects.

Prerequisite: None

Online Module 2 – Intro to Web Development and Data Persistence
Clock Hours: 105

Before discussing web development, students must first understand data persistence. Saving data in a structured way that maps to object-oriented design is handled through an object relational mapper (ORM). ORM allows programmers to query and manipulate data from a database using an object-paradigm. Students learn to write manipulate data using the Ruby language. Students will gain an appreciation for the structure of a database, how to map out tables, and the difference between the various table relationships. Students will learn how to wireframe database structures, as well as how to link their applications to a database. This unit covers SQL, domain modeling, relational database theory, schema architecture, and the Object Relational Model, include the ACTIVE Record pattern.

Once students are able to save data to databases, we must cover delivering that data to our users over the web. Before showing students Sinatra, students must first understand how the HyperText Transfer Protocol (HTTP) works. We explain HTTP using the Ruby web interface known as Rack. Students will build their own HTTP Servers and learn how the request/response model of the web works. Their servers will listen to HTTP requests and respond with well-formed HTML responses. They will understand the web with few abstractions provided by the tool set.

After obtaining an understanding of HTTP and Rack, students will learn Sinatra. Sinatra is a Domain Specific Language (DSL) written in Ruby for building web applications on top of Rack. This framework provides students with exposure to design patterns in web applications. The topics covered in this unit include architectural patterns such as REST (Representational State Transfer), MVC (Model-View-Controller), HTML Forms, ERB (Embedded Ruby) and template rendering, and application environments.

Deliverable: A Sinatra powered Ruby web application that processes data from HTML forms and communicates with a model layer to communicate with the database.

Prerequisite: Online Module 1

Online Module 3 – Advanced Web Development with Rails
Clock Hours: 105

Having a foundation in the Ruby language as well as the architecture of the world wide web, students will use Rails to build complex functional web applications from the ground up. They will learn the file structure of Rails, how to set up their own databases, how to draw routes, create Rails forms, gain an understanding of the asset pipeline, and bring it together by integrating their front-end design skills.

Students will also be capable to take on more advanced concepts such as authorization, validation and callbacks. Once students grasp the basic functionality of Rails, they will spend time building out their own Rails applications, moving through the entire process from idea to execution

Deliverable: A multi-model Ruby on Rails powered web application that integrates nested forms and URL routes to a multi-relational data set.

Prerequisite: Online Module 2

Online Module 4 – Creating Interactive and Performant Front-Ends with JavaScript
Clock Hours: 105

JavaScript (EMCAScript) powers the user experience of the web. Students will learn the basics of the JavaScript syntax, it's functional architecture, along with a few approaches to the object model. They will then learn the Document Object Model (DOM) JavaScript API provided by the browser to dynamically interact with HTML. This unit will focus on jQuery, the most popular JavaScript library, to aid students in learning how to collect user input, manipulate the DOM with animations and injection, and send Asynchronous AJAX requests for a rich user experience. After understanding jQuery students are given an introduction into front-end frameworks such as Backbone and Ember.js. These frameworks allow students to write larger, more organized code-bases.

Deliverable: A Rails application that includes examples of DOM Manipulation, AJAX HTML Loading, and AJAX Form Submissions.

Prerequisite: Online Module 3

Online Module 5 – Scalable Front-End Design with React
Clock Hours: 105

Building complex front ends with plain JavaScript doesn't work as applications get more complex. Just as object-oriented design was created to help large teams build incredibly complex back end apps, front end frameworks were created to provide the structure needed to build complex, dynamic front ends. This module covers the React framework by Google. Continuing from our JavaScript module, students build a simple version of React. JavaScript frameworks change quite frequently so understanding the fundamentals of structured JavaScript is crucial to remaining relevant throughout a web developer's career. Building a simple React clone before using it ensures students understanding of the fundamentals.

After building all of the base components that make up React, students are then introduced to using React throughout their app. First, we discuss how to make reusable user interface components and handle user input and display. After being able to receive data from the user and display it, we then cover how to send that data to the back end using $http and services. Finally, students learn how to handle asynchronous function calls using promises.

Deliverable: A multi-model Ruby on Rails powered backend that serves JSON to a multi-page React front end.

Prerequisite: Online Module 4

PROGRAM: Data Science

Educational Objectives and Program Description

The objective of this program is to prepare students for an entry-level career as a Data Scientist. Working as a data scientist requires software engineering skills, statistical understanding, and skills for understanding a new domain. More important than teaching any tooling, the program will teach students to gather data, apply statistical analysis to answer questions with data, and make their insights and information actionable. No other software is required for this program.

To ensure students walk away with a foundation of programming fundamentals, statistics, and integration of these insights into the business, we will teach students a variety of skills. Students will learn how to retrieve data through the use of APIs, SQL, and scraping. They will learn how to analyze information through use of probability, frequentist and Bayesian statistics and linear regression. Students will learn how to communicate and integrate this information by learning data visualization libraries like Seaborn and Bokeh, and the web development library Flask.

Due to the nature of the technology industry this program focuses heavily on teaching students how to learn new skills on their own. Software development moves too quickly for any skill to remain relevant forever, and therefore students are required to work with their classmates to solve problems and use various online tools to discover new coding techniques that may not be covered in class.

Newly enrolled students must complete the Pre-Work before beginning in-class instruction. The purpose of this preparatory work is to provide real examples of the curriculum so that applicants are fully aware of the demands of the program prior to committing to it financially. The Pre-Work is free and available online: http://prework.flatironschoo.com/web-development/

Delivery Method

Traditional classroom.

Program Outline

Module	Module Title	Clock Hours
Module 1	Data Exploration and Analysis	105

Module 2	Probability and Statistics for Data Science	105
Module 3	Machine Learning: Regression Optimization and Big Data	105
Module 4	Machine Learning: Classification and Deep Learning	105
Module 5	Data Science Projects	105
	Total Clock Hours:	**525**

Module Descriptions

Module 1 – Data Exploration and Analysis
Clock Hours: 105

Students will gain an overview into skills required of data scientist: data gathering and cleaning, analysis using probability and summary statistics, and presentation of information with visualization libraries. The focus will be on gathering and cleaning data.

To gather and clean data, students will learn fundamental concepts in programming using Python and SQL. Topics will include writing, functions and object orientation, scraping and regular expressions, and SQL.

Students will learn how to go from question requirements to targeted data mining. In doing so, students will learn about using experimental design and issue trees to turn problem requirements into quantifiable pieces of work. From there, students will learn how to use targeted data mining with tools such as Pandas, probability, and summary statistics to answer their questions. Students will also learn about plotting and communicating results with visualization tools like Seaborn.

Deliverable: A solid foundation in cleaning and gathering data with Python, Pandas, and SQL. Practice with going from problem requirements to actionable steps with issue trees and experimental design. Practice with communicating results with Seaborn.

Prerequisite: Program Pre-Work

Module 2 – Probability and Statistics for Data Science
Clock Hours: 105

Now that students have learned how to gather and explore data with Python and SQL students can go deeper into analyzing that information with statistics. Students will learn about learn about Bayesian and Frequentist statistics. Students will learn binary classifications and how to evaluate the accuracy classifier system by using confusion matrices, false positives and false negatives, the true positive rate with ROC curves.

Students will learn about repeated random sampling with Monte Carlo simulations and Markov Process. Then students will revisit experimental design techniques and apply their deeper statistical knowledge to A/B testing a website.

Students will go into regression analysis, learning about linear and logistic regression. In building regression models, students will learn about penalization terms, preventing overfitting through regularization and using cross validation to validate regression model. Finally, students will learn about solving linear regression with gradient descent.

Deliverable: Students will understand the difference between Frequentist and Bayesian, and how to apply each. Students will learn how to build and validate regression models. Students will learn how to best fit a linear regression to sample data using ordinary least squares, and apply gradient descent to ordinary least squares.

Prerequisite: Module A

Module 3 – Machine Learning: Regression Optimization and Big Data
Clock Hours: 105

Students will move into supervised learning, non-parametric algorithms like k-nearest-neighbors and support vector machines. Students will also learn about decision tree learning and how it can be applied to classification and regression tree analysis. Students will learn about various techniques in decision trees such as bagging and boosting to construct more than one decision tree. Students will also learn how to conduct time series analysis using Pandas.

Students will be introduced using threading and multiprocessing to work with big data. In doing so, students will learn about Apache Spark, Apache Spark on AWS.

Deliverable: Students will be able to use disparate large data sets to build classification engines.

Prerequisite: Module B

Module 4 – Machine Learning: Classification and Deep Learning
Clock Hours: 105

Students will move onto unsupervised learning techniques such as clustering techniques like k-means and hierarchical clustering. Students will also learn how to simplify their machine learning models by using non-negative matrix factorization and principle component analysis. Students will learn about how to accommodate for imbalanced data in their machine learning models.

Students will move on to building recommender algorithms using collaborative filtering, matrix decomposition, clustering, and deep learning approaches.

Deliverable: Students will learn about unsupervised techniques, and then learn how to implement supervised or unsupervised learning techniques to build recommender systems.

Prerequisite: Module C

Module 5 – Data Science Projects
Clock Hours: 105

Students will work on and complete a solo final project as an opportunity to review and push deeper into the material covered in the previous modules. Along the way, coursework will emphasize project management techniques in Data Science, as well as presenting and communicating an Data Science project to a non-technical audience.

Deliverable: Guidance in constructing a project that gathers, explores, builds statistical or machine learning models to deliver insights and communicate findings with data visualization and storytelling techniques.

Prerequisite: Module D

PROGRAM: Full-Time Online Data Science

Educational Objectives and Program Description

The objective of this program is to prepare students for an entry-level career as a Data Scientist. Working as a data scientist requires software engineering skills, statistical understanding, and skills for understanding a new domain. More important than teaching any tooling, the program will teach students to gather data, apply statistical analysis to answer questions with data, and make their insights and information actionable. No other software is required for this program.

To ensure students walk away with a foundation of programming fundamentals, statistics, and integration of these insights into the business, we will teach students a variety of skills. Students will learn how to retrieve data through the use of APIs, SQL, and scraping. They will learn how to analyze information through use of probability, frequentist and Bayesian statistics and linear regression. Students will learn how to communicate and integrate this information by learning data visualization libraries like Seaborn and Bokeh, and the web development library Flask.

Due to the nature of the technology industry this program focuses heavily on teaching students how to learn new skills on their own. Software development moves too quickly for any skill to remain relevant forever, and therefore students are required to work with their classmates to solve problems and use various online tools to discover new coding techniques that may not be covered in class.

Delivery Method

Online. Over the course of 5 months, students regularly engage with their instructors and fellow students. Students virtually meet with the instructor weekly to break down tough technical concepts. Students also meet with Educational Coaches ten times throughout the program to help students adopt behaviors that foster the successful and timely completion of the program. Students also interact with members of their cohort in weekly study groups.

The program consists of five modules as outlined below. Each module concludes with a comprehensive portfolio project meant to demonstrate student learning. Instructors virtually meet with each student individually to discuss their portfolio projects and ensure the student is ready to progress to the next module.

Communication and Technical Support

Students have unrestricted access to the program curriculum and online community via the Learn.co platform while enrolled (24 hours per day, 7 days per week). Flatiron's team of Technical Coaches via the Learn.co platform's Ask a Question Feature are also available to provide live support for students Monday-Friday, 9AM-1AM ET, and Saturday-Sunday, 9AM-12AM ET. Flatiron also maintains a collection of help articles, with advice and answers to frequently asked questions from the Flatiron School Team at help.learn.co.

Program Outline

Module	**Module Title**	**Clock Hours**
Online Module 1	Data Exploration and Analysis	105
Online Module 2	Probability and Statistics for Data Science	105
Online Module 3	Machine Learning: Regression Optimization and Big Data	105
Online Module 4	Machine Learning: Classification and Deep Learning	105
Online Module 5	Data Science Projects	105
	Total Clock Hours:	**525**

Module Descriptions

Online Module 1 – Data Exploration and Analysis
Clock Hours: 105

Students will gain an overview into skills required of data scientist: data gathering and cleaning, analysis using probability and summary statistics, and presentation of information with visualization libraries. The focus will be on gathering and cleaning data.

To gather and clean data, students will learn fundamental concepts in programming using Python and SQL. Topics will include writing, functions and object orientation, scraping and regular expressions, and SQL.

Students will learn how to go from question requirements to targeted data mining. In doing so, students will learn about using experimental design and issue trees to turn problem requirements into quantifiable pieces of work. From there, students will learn how to use targeted data mining with tools such as Pandas, probability, and summary statistics to answer their questions. Students will also learn about plotting and communicating results with visualization tools like Seaborn.

Deliverable: A solid foundation in cleaning and gathering data with Python, Pandas, and SQL. Practice with going from problem requirements to actionable steps with issue trees and experimental design. Practice with communicating results with Seaborn.

Prerequisite: None

Online Module 2 – Probability and Statistics for Data Science
Clock Hours: 105

Now that students have learned how to gather and explore data with Python and SQL students can go deeper into analyzing that information with statistics. Students will learn about learn about Bayesian and Frequentist statistics. Students will learn binary classifications and how to evaluate the accuracy classifier system by using confusion matrices, false positives and false negatives, the true positive rate with ROC curves.

Students will learn about repeated random sampling with Monte Carlo simulations and Markov Process. Then students will revisit experimental design techniques and apply their deeper statistical knowledge to A/B testing a website.

Students will go into regression analysis, learning about linear and logistic regression. In building regression models, students will learn about penalization terms, preventing overfitting through regularization and using cross validation to validate regression model. Finally, students will learn about solving linear regression with gradient descent.

Deliverable: Students will understand the difference between Frequentist and Bayesian, and how to apply each. Students will learn how to build and validate regression models. Students will learn how to best fit a linear regression to sample data using ordinary least squares, and apply gradient descent to ordinary least squares.

Prerequisite: Online Module 1

Online Module 3 – Machine Learning: Regression Optimization and Big Data
Clock Hours: 105

Students will move into supervised learning, non-parametric algorithms like k-nearest-neighbors and support vector machines. Students will also learn about decision tree learning and how it can be applied to classification and regression tree analysis. Students will learn about various techniques in decision trees such as bagging and boosting to construct more than one decision tree. Students will also learn how to conduct time series analysis using Pandas.

Students will be introduced using threading and multiprocessing to work with big data. In doing so, students will learn about Apache Spark, Apache Spark on AWS.

Deliverable: Students will be able to use disparate large data sets to build classification engines.

Prerequisite: Online Module 2

Online Module 4 – Machine Learning: Classification and Deep Learning
Clock Hours: 105

Students will move onto unsupervised learning techniques such as clustering techniques like k-means and hierarchical clustering. Students will also learn how to simplify their machine learning models by using non-negative matrix factorization and principle component analysis. Students will learn about how to accommodate for imbalanced data in their machine learning models.

Students will move on to building recommender algorithms using collaborative filtering, matrix decomposition, clustering, and deep learning approaches.

Deliverable: Students will learn about unsupervised techniques, and then learn how to implement supervised or unsupervised learning techniques to build recommender systems.

Prerequisite: Online Module 3

Online Module 5 – Data Science Projects
Clock Hours: 105

Students will work on and complete a solo final project as an opportunity to review and push deeper into the material covered in the previous modules. Along the way, coursework will emphasize project management techniques in data science, as well as presenting and communicating a data science project to a non-technical audience.

Deliverable: Guidance in constructing a project that gathers, explores, builds statistical or machine learning models to deliver insights and communicate findings with data visualization and storytelling techniques.

Prerequisite: Online Module 4

PROGRAM: Part-Time Online Data Science

Educational Objectives and Program Description

The objective of this program is to prepare students for an entry-level career as a Data Scientist. Working as a data scientist requires software engineering skills, statistical understanding, and skills for understanding a new domain. More important than teaching any tooling, the program will teach students to gather data, apply statistical analysis to answer questions with data, and make their insights and information actionable. No other software is required for this program.

To ensure students walk away with a foundation of programming fundamentals, statistics, and integration of these insights into the business, we will teach students a variety of skills. Students will learn how to retrieve data through the use of APIs, SQL, and scraping. They will learn how to analyze information through use of probability, frequentist and Bayesian statistics and linear regression. Students will learn how to communicate and integrate this information by learning data visualization libraries like Seaborn and Bokeh, and the web development library Flask.

Due to the nature of the technology industry this program focuses heavily on teaching students how to learn new skills on their own. Software development moves too quickly for any skill to remain relevant forever, and therefore students are required to work with their classmates to solve problems and use various online tools to discover new coding techniques that may not be covered in class.

Delivery Method

Online. Over the course of 10 months, students regularly engage with their instructors and fellow students. Students virtually meet with the instructor weekly to break down tough technical concepts. Students also meet with Educational Coaches ten times throughout the program to help students adopt behaviors that foster the successful and timely completion of the program. Students also interact with members of their cohort in weekly study groups.

The program consists of five modules as outlined below. Each module concludes with a comprehensive portfolio project meant to demonstrate student learning. Instructors virtually meet with each student individually to discuss their portfolio projects and ensure the student is ready to progress to the next module.

Program Outline

Module	Module Title	Clock Hours

Online Module 1	Data Exploration and Analysis	105
Online Module 2	Probability and Statistics for Data Science	105
Online Module 3	Machine Learning: Regression Optimization and Big Data	105
Online Module 4	Machine Learning: Classification and Deep Learning	105
Online Module 5	Data Science Projects	105
	Total Clock Hours:	**525**

Module Descriptions

Online Module 1 – Data Exploration and Analysis
Clock Hours: 105

Students will gain an overview into skills required of data scientist: data gathering and cleaning, analysis using probability and summary statistics, and presentation of information with visualization libraries. The focus will be on gathering and cleaning data.

To gather and clean data, students will learn fundamental concepts in programming using Python and SQL. Topics will include writing, functions and object orientation, scraping and regular expressions, and SQL.

Students will learn how to go from question requirements to targeted data mining. In doing so, students will learn about using experimental design and issue trees to turn problem requirements into quantifiable pieces of work. From there, students will learn how to use targeted data mining with tools such as Pandas, probability, and summary statistics to answer their questions. Students will also learn about plotting and communicating results with visualization tools like Seaborn.

Deliverable: A solid foundation in cleaning and gathering data with Python, Pandas, and SQL. Practice with going from problem requirements to actionable steps with issue trees and experimental design. Practice with communicating results with Seaborn.

Prerequisite: None

Online Module 2 – Probability and Statistics for Data Science
Clock Hours: 105

Now that students have learned how to gather and explore data with Python and SQL students can go deeper into analyzing that information with statistics. Students will learn about learn about Bayesian and Frequentist statistics. Students will learn binary classifications and how to evaluate the accuracy classifier system by using confusion matrices, false positives and false negatives, the true positive rate with ROC curves.

Students will learn about repeated random sampling with Monte Carlo simulations and Markov Process. Then students will revisit experimental design techniques and apply their deeper statistical knowledge to A/B testing a website.

Students will go into regression analysis, learning about linear and logistic regression. In building regression models, students will learn about penalization terms, preventing overfitting through regularization and using cross validation to validate regression model. Finally, students will learn about solving linear regression with gradient descent.

Deliverable: Students will understand the difference between Frequentist and Bayesian, and how to apply each. Students will learn how to build and validate regression models. Students will learn how to best fit a linear regression to sample data using ordinary least squares, and apply gradient descent to ordinary least squares.

Prerequisite: Online Module 1

Online Module 3 – Machine Learning: Regression Optimization and Big Data
Clock Hours: 105

Students will move into supervised learning, non-parametric algorithms like k-nearest-neighbors and support vector machines. Students will also learn about decision tree learning and how it can be applied to classification and regression tree analysis. Students will learn about various techniques in decision trees such as bagging and boosting to construct more than one decision tree. Students will also learn how to conduct time series analysis using Pandas.

Students will be introduced using threading and multiprocessing to work with big data. In doing so, students will learn about Apache Spark, Apache Spark on AWS.

Deliverable: Students will be able to use disparate large data sets to build classification engines.

Prerequisite: Online Module 2

Online Module 4 – Machine Learning: Classification and Deep Learning
Clock Hours: 105

Students will move onto unsupervised learning techniques such as clustering techniques like k-means and hierarchical clustering. Students will also learn how to simplify their machine learning models by using non-negative matrix factorization and principle component analysis. Students will learn about how to accommodate for imbalanced data in their machine learning models.

Students will move on to building recommender algorithms using collaborative filtering, matrix decomposition, clustering, and deep learning approaches.

Deliverable: Students will learn about unsupervised techniques, and then learn how to implement supervised or unsupervised learning techniques to build recommender systems.

Prerequisite: Online Module 3

Online Module 5 – Data Science Projects
Clock Hours: 105

Students will work on and complete a solo final project as an opportunity to review and push deeper into the material covered in the previous modules. Along the way, coursework will emphasize project management techniques in data science, as well as presenting and communicating a data science project to a non-technical audience.

Deliverable: Guidance in constructing a project that gathers, explores, builds statistical or machine learning models to deliver insights and communicate findings with data visualization and storytelling techniques.

Prerequisite: Online Module 4

PROGRAM: Self-Paced Online Data Science

Educational Objectives and Program Description

The objective of this program is to prepare students for an entry-level career as a Data Scientist. Working as a data scientist requires software engineering skills, statistical understanding, and skills for understanding a new domain. More important than teaching any tooling, the program will teach students to gather data, apply statistical analysis to answer questions with data, and make their insights and information actionable. No other software is required for this program.

To ensure students walk away with a foundation of programming fundamentals, statistics, and integration of these insights into the business, we will teach students a variety of skills. Students will learn how to retrieve data through the use of APIs, SQL, and scraping. They will learn how to analyze information through use of probability, frequentist and Bayesian statistics and linear regression. Students will learn how to communicate and integrate this information by learning data visualization libraries like Seaborn and Bokeh, and the web development library Flask.

Due to the nature of the technology industry this program focuses heavily on teaching students how to learn new skills on their own. Software development moves too quickly for any skill to remain relevant forever, and therefore students are required to work with their classmates to solve problems and use various online tools to discover new coding techniques that may not be covered in class.

Delivery Method

Online. Students have up to 15 months to complete this self-paced program. Throughout the program, students have the option to view and participate in live study group discussions or view recorded study group discussions. Students also meet with an Educational Coach five times throughout the program. Educational Coaches help students set goals for the program and ensure students progress through the program at a steady pace.

The program consists of five modules as outlined below. Each module concludes with a comprehensive portfolio project meant to demonstrate student learning. Instructors virtually meet with each student individually to discuss their portfolio projects and ensure the student is ready to progress to the next module.

Program Outline

Module	Module Title	Clock Hours
Online Module 1	Data Exploration and Analysis	105
Online Module 2	Probability and Statistics for Data Science	105
Online Module 3	Machine Learning: Regression Optimization and Big Data	105
Online Module 4	Machine Learning: Classification and Deep Learning	105
Online Module 5	Data Science Projects	105
	Total Clock Hours:	**525**

Module Descriptions

Online Module 1 – Data Exploration and Analysis
Clock Hours: 105

Students will gain an overview into skills required of data scientist: data gathering and cleaning, analysis using probability and summary statistics, and presentation of information with visualization libraries. The focus will be on gathering and cleaning data.

To gather and clean data, students will learn fundamental concepts in programming using Python and SQL. Topics will include writing, functions and object orientation, scraping and regular expressions, and SQL.

Students will learn how to go from question requirements to targeted data mining. In doing so, students will learn about using experimental design and issue trees to turn problem requirements into quantifiable pieces of work. From there, students will learn how to use targeted data mining with tools such as Pandas, probability, and summary statistics to answer their questions. Students will also learn about plotting and communicating results with visualization tools like Seaborn.

Deliverable: A solid foundation in cleaning and gathering data with Python, Pandas, and SQL. Practice with going from problem requirements to actionable steps with issue trees and experimental design. Practice with communicating results with Seaborn.

Prerequisite: None

Online Module 2 – Probability and Statistics for Data Science
Clock Hours: 105

Now that students have learned how to gather and explore data with Python and SQL students can go deeper into analyzing that information with statistics. Students will learn about learn about Bayesian and Frequentist statistics. Students will learn binary classifications and how to evaluate the accuracy classifier system by using confusion matrices, false positives and false negatives, the true positive rate with ROC curves.

Students will learn about repeated random sampling with Monte Carlo simulations and Markov Process. Then students will revisit experimental design techniques and apply their deeper statistical knowledge to A/B testing a website.

Students will go into regression analysis, learning about linear and logistic regression. In building regression models, students will learn about penalization terms, preventing overfitting through regularization and using cross validation to validate regression model. Finally, students will learn about solving linear regression with gradient descent.

Deliverable: Students will understand the difference between Frequentist and Bayesian, and how to apply each. Students will learn how to build and validate regression models. Students will learn how to best fit a linear regression to sample data using ordinary least squares, and apply gradient descent to ordinary least squares.

Prerequisite: Online Module 1

Online Module 3 – Machine Learning: Regression Optimization and Big Data
Clock Hours: 105

Students will move into supervised learning, non-parametric algorithms like k-nearest-neighbors and support vector machines. Students will also learn about decision tree learning and how it can be applied to classification and regression tree analysis. Students will learn about various techniques in decision trees such as bagging and boosting to construct more than one decision tree. Students will also learn how to conduct time series analysis using Pandas.

Students will be introduced using threading and multiprocessing to work with big data. In doing so, students will learn about Apache Spark, Apache Spark on AWS.

Deliverable: Students will be able to use disparate large data sets to build classification engines.

Prerequisite: Online Module 2

Online Module 4 – Machine Learning: Classification and Deep Learning
Clock Hours: 105

Students will move onto unsupervised learning techniques such as clustering techniques like k-means and hierarchical clustering. Students will also learn how to simplify their machine learning models by using non-negative matrix factorization and principle component analysis. Students will learn about how to accommodate for imbalanced data in their machine learning models.

Students will move on to building recommender algorithms using collaborative filtering, matrix decomposition, clustering, and deep learning approaches.

Deliverable: Students will learn about unsupervised techniques, and then learn how to implement supervised or unsupervised learning techniques to build recommender systems.

Prerequisite: Online Module 3

Online Module 5 – Data Science Projects
Clock Hours: 105

Students will work on and complete a solo final project as an opportunity to review and push deeper into the material covered in the previous modules. Along the way, coursework will emphasize project management techniques in data science, as well as presenting and communicating a data science project to a non-technical audience.

Deliverable: Guidance in constructing a project that gathers, explores, builds statistical or machine learning models to deliver insights and communicate findings with data visualization and storytelling techniques.

Prerequisite: Online Module 4

PROGRAM: UX/UI Design

Educational Objectives and Program Description

The objective of this program is to prepare students for design and technology careers and to prepare designers for a lifetime of learning.

The UX/UI Design program is a 24-week program specializing in the fields of UX and UI design with the primary goal to turn students into hireable candidates for innovative and tech-focused companies. After the first six weeks of the program, students decide whether to complete the UX or UI track.

The UX/UI Design program is a hybrid program, consisting of both online education and in-person immersion. The curriculum is taught in a time span of 24 weeks, the first 12 weeks being an online and remote phase with weekly virtual check-ins. Classes are held four nights per week during the in-person phase, with daytime topic sprints held daily at 10AM sharp.

Throughout the program, students attend presentations by guest speakers, sponsored workshops, and lab sessions. Lab time isn't prescheduled, but is prevalent 24/7. Students schedule their own lab time to complete projects and receive mentorship. While there is no formal grading, students are asked to create portfolio deliverables and to actively document their design process for the purpose of finding a job after graduation.

During the fifth week of the program, each designer has a mandatory performance evaluation with the instructor, who makes a determination of the designer's ability to continue the rest of the program. Students who pass their mandatory performance evaluation continue to the Virtual Phase of the program.

Delivery Method

Hybrid. The first 12 weeks of the program are online and the final 12 weeks of the program are in-person.

Program Outline

Phase	Phase Title	Clock Hours
Phase 1	Design Essentials (online)	90
Phase 2	Virtual Phase (online)	240
Phase 3	Immersion Phase (in-person)	330
Phase 4	Client Phase (in-person)	264
Phase 5	Career Phase (in-person)	198

	Total Clock Hours:	1122

Phase Descriptions

Phase 1 – Design Essentials
Clock Hours: 90

Over this six-week online phase, students will gain an overview of the basics of user experience, design, and research. Students will learn the building blocks of UX, interaction design, UI, and visual design, and then put them all together to form a design process. During the fifth week of Design Essentials, designers will participate in a mandatory performance evaluation with an instructor. The instructor will make a determination of the designer's ability to continue the rest of the program. Students who pass their mandatory performance evaluation continue to Phase 2 – Virtual Phase and select whether to pursue the UX or UI track.

Phase 2 – Virtual Phase
Clock Hours: 240

Throughout this six-week online phase, students apply the skills acquired in Design Essentials to UX or UI case studies and team projects.

Prerequisite: Phase 1

Phase 3 – Immersion Phase
Clock Hours: 330

During this five-week in-person phase, students apply the skills learned in either the UX or UI virtual phases to a real client case study. The Immersion Phase ties together important features of professional practice and team-based design, along with the vital skills of analyzing and synthesizing work. Each of the five weeks of the Immersion Phase is designed as a sprint, so teams know works needs to be completed before the next sprint review and how to budget their time to accomplish all aspects of the sprint.

Prerequisite: Phase 2

Phase 4 – Client Phase
Clock Hours: 264

During this 4-week in-person phase, students work on UX or UI design briefs with live clients. Students typically work in teams of 2-4 students, with each team paired with a single client company or the duration of the phase.

Students in the UX track tackle projects focused on product functionality, starting with user research and delivering low-fidelity wireframes, information architecture, and prototypes. Students in the UI track focus on the visual look and feel of user interfaces, starting with visual brand exploration and delivering high-fidelity screen mockups and comprehensive UI style guides. While they receive direction and support from dedicated creative directors and project managers, the design teams are tasked with taking ownership of their professional client engagements.

Successful completion of the Client Phase involves collaborating effectively with teammates while practically applying the skills, concepts, tools, and methodologies learned in the previous phases of the program.

Prerequisite: Phase 3

Phase 5 – Career Phase
Clock Hours: 198

The purpose of this three-week phase is to provide students with all the tools they need to start their careers as Designers. The Career Phase is a significant shift from the rest of the program; the first twelve weeks are about learning how to design, and the next nine weeks are about learning how to be designers. These three weeks are about learning how to tell the story of their grown as designers through the program. The primary objective is to learn how to communicate effectively with passive and active audiences to achieve professional goals.

Over the course of the Career Phase, students complete three case studies, a resume, a portfolio/site/theme/URL/hosting platform, a personal statement, a sample cover letter, and a values report.

Prerequisite: Phase 4

COURSE: Intro to Front End Web Development

Educational Objectives and Course Description

The objective of this course is to provide the student with introductory knowledge and skills in front end web development. Over 10 weeks, students in this part-time course will code in HTML, CSS, and JavaScript; industry standard tools and services to build and publish responsive websites. No other software is required for this program.

This beginner level course will teach the fundamentals of front end web development and professional workflow using GIT, CLI, HTML5, CSS3, JavaScript, jQuery, and Responsive Design Strategies. This is a flipped curriculum course that combines online learning via watching videos at home with on location code challenges and working on personal projects in the classroom.

Students are expected to have a general knowledge of computers, opening, saving files, etc. Students can expect to commit to a minimum of 9-10 hours per week on this course with approximately 4 hours watching videos and coding exercises at home and another 6 hours in the classroom working on group code challenges and personal site projects. A total of approximately 100 hours is needed to complete this course in ten weeks. All lectures will be online and the majority of class time will be spent in hands-on coding. There is limited pre-work and the student is encouraged to complete it prior to beginning the course.

Delivery Method

Traditional classroom with some additional online components

Course Outline

60 clock hours over 10 weeks. Classes meet two nights per week from 6-9 PM. Students are expected to watch approximately 3-4 hours of online lectures each week, then attend twice-weekly class sessions to implement and practice what they learned in the lectures.

Unit 1 - Web Fundamentals, 6 hours
Unit 2 - Styling Site Content, 6 hours
Unit 3 - Page Layout, 6 hours
Unit 4 - Responsive Design, 6 hours
Unit 5 - Building with Bootstrap, 6 hours
Unit 6 - JavaScript Fundamentals, 6 hours
Unit 7 - JavaScript Application Flow Control, 6 hours
Unit 8 - Object Oriented Programming in JavaScript, 6 hours
Unit 9 - Asynchronous Content & Site Publishing, 6 hours
Unit 10 - Testing & MV Frameworks, 6 hours

Course Unit Descriptions

Unit 1 - Web Fundamentals
This unit will cover how the web works, site planning, wireframes, and HTML fundamentals. Students will begin to create page content and familiarize themselves with common HTML elements.

Topics
- The HTML Programming Environment
- Authoring HTML Page

Unit 2 - Styling Site Content
This unit will cover additional HTML elements such as forms, tables, and iframes as well as an introduction to using CSS for styling content. Students will familiarize themselves with selecting HTML elements and styling them using CSS.

Topics
- Fuller Exploration of the HTML tag universe
- Advanced I/O HTML interfaces, media elements, and semantic elements

Unit 3 - Page Layout

This unit will cover everything you need to know about page layout. Students will learn to build their own grid systems and have full command to position their content anywhere they please.

Topics

- Box Model
- Layout & Positioning

Unit 4 - Responsive Design
This unit will cover how to style boxes, create rollover images using sprites, as well as introduce responsive design. Students will learn to use CSS media queries to alter the sizing and positioning of content on different size devices.

Topics

- Box Aesthetics
- Image Sprites
- Responsive Strategies

Unit 5 - Building with Bootstrap
This unit will cover how to build new site projects using the Twitter Bootstrap Framework.

Topics

- Bootstrap Grid Systems
- Theme & Components
- Customization

Unit 6 - JavaScript Fundamentals
This unit will cover JavaScript fundamentals and introduce jQuery. We will explore adding behavior and adding complexity to our user interface.

Topics

- JavaScript Fundamentals
- Document Object Model
- Introduction to jQuery

Unit 7 - JavaScript Application Flow Control
This unit will cover JavaScript flow control by introducing functions and loops. We will also look at using jQuery to gather data and retrieve form input.

Topics

- JavaScript Flow Control
- jQuery Data & Input
- Form Validation
- Regular Expressions

Unit 8 - Object Oriented Programming in JavaScript
This unit will cover object-oriented programming in JavaScript. We will look at creating objects, constructors, and getting objects to talk to one another.

Topics
- Object Literals
- Constructor
- Prototype
- Objects Interactions

Unit 9 - Asynchronous Content & Site Publishing
This unit will cover JSON and AJAX which will allow us to make asynchronous requests. We will also explore APIs, frameworks, and site publishing using FTP and SSH.

Topics
- JSON
- AJAX
- APIs
- Site Publishing

Unit 10 - Testing & MV Frameworks
This unit will cover MVC/MV* patterns and an introduction to Backbone JS.

Topics
- Testing with Jasmine
- Introduction to Backbone
- Models & Collections
- Views
- Router

COURSE Intro to Data Science

Educational Objectives and Course Description

The objective of this course is to provide the student with introductory knowledge and skills in Data Science. No other software is required for this program.

This 10-week part-time program will provide students with the knowledge and skills to collect and retrieve data using Python, and use statistics to analyze that data and make their insight actionable. Students will learn how to retrieve data through the use of APIs, SQL, and scraping. They will learn how to analyze information through use of frequentist statistics and linear regression. Students will learn how to communicate and integrate this information by learning data visualization libraries like Plotly and show process and insights in a Jupyter notebook.

Due to the nature of the technology industry this program focuses heavily on teaching students how to learn new skills on their own. Software development moves too quickly for any skill to remain relevant forever, and therefore students are required to work with their classmates to solve problems and use various online tools to discover new coding techniques that may not be covered in class.

Delivery Method

Traditional classroom.

Course Outline

60 clock hours over 10 weeks. Classes meet two nights per week from 6-9 PM. Students are expected to complete approximately 3-4 hours of homework per week.

Unit 1 – Introduction to Data Science and Programming, 3 clock hours
Unit 2 – Collections, 3 clock hours
Unit 3 – Collections and Visualization, 3 clock hours
Unit 4 – Looping and Iteration, 3 clock hours
Unit 5 – Functions and Scope, 3 clock hours
Unit 6 – Conditionals and Booleans Map and Filter, 3 clock hours
Unit 7 – Working with APIs, 3 clock hours
Unit 8 – Manipulating Data from the Web, 3 clock hours
Unit 9 – Python Projects, 3 clock hours
Unit 10 – Continue Projects/Review Python, 3 clock hours
Unit 11 – Introduction to SQL, 3 clock hours
Unit 12 – Relational Databases, 3 clock hours
Unit 13 – Working with SQL and Python, 3 clock hours
Unit 14 – Summary Statistics, 3 clock hours
Unit 15 – Distributions, 3 clock hours
Unit 16 – Relationships between Variables, 3 clock hours
Unit 17 – Regression, 3 clock hours
Unit 18 – Advanced Regression, 3 clock hours
Unit 19 – Regression Projects, 3 clock hours
Unit 20 – Regression Projects, 3 clock hours

Course Unit Descriptions

Unit 1 - Introduction to Data Science and Programming
This unit will provide an introduction to data science and programming.

Topics

- Strings
- Numbers and Booleans
- Variables

Unit 2 - Collections
This unit will serve as an introduction to collections.

Topics

- Lists
- Dictionaries

Unit 3 - Collections and Visualization
This unit will continue the discussion and instruction on regarding collections and introduce concepts on visualization.

Topics

- Introduction to Plotly
- Advanced List Functions
- Advanced Hash Functions

Unit 4 - Looping and Iteration
This unit will instruct on the subject of looping and iteration.

Topics

- For Loop to Operate on Data
- For Loop to Change Data
- Plotting with For Loop

Unit 5 - Functions and Scope
This unit will present functions and scope to the class.

Topics
Functions without arguments

- Scope
- Functions with arguments

Unit 6 - Conditionals and Booleans Map and Filter
This unit will give class insight into conditionals and booleans.

Topics

- If else
- Looping with conditionals

Unit 7 - Working with APIs
This unit will serve as an introduction to APIs.

Topics

- Working with NYC open data API and plotting data

Unit 8 - Manipulating Data from the Web
This unit will present concepts on web data manipulation.

Topics

- Scraping data with Beautiful Soup library
- Using the requests library to retrieve JSON from the web
- Then manipulating data with previous collections knowledge

Unit 9 - Python Projects
This unit will be used to begin Python projects.

Topics
- Answer question with data by pull data from APIs to display visually

Unit 10 - Continue Projects/Review Python
This unit will be used to continue and review Python and Python projects.

Topics
- Refactor code and identify conclusions of projects

Unit 11 - Introduction to SQL
This unit will introduce students to the basics of SQL and how to use .sql.

Topics
- SQL Introduction
- Installing SQLite
- SQL Basics
    - SELECT
    - INSERT
    - UPDATE
    - CREATE
    - ALTER
- Using .sql files
- SQLite data types

Unit 12 - Relational Databases
This unit will give insight into relational databases.

Topics
- Relational Databases
    - History
    - Relations as related to ruby object relations
- JOIN Statements
    - foreign keys
    - Left Outer Join
    - Left Inner Join

Unit 13 - Working with SQL and Python
This unit will bring together concepts of working with SQL from recent classes with Python from earlier in the course.

Topics

- Sorting and Grouping Data
  - ORDER BY
  - GROUP BY
- Complex Joins
  - Many-to-Many relationships
  - Join tables

Unit 14 - Summary Statistics
This unit will discuss summary statistics.

Topics

- Prob density function
- Identification of central tendency, outliers, variance, standard deviation
- Normal distribution

Unit 15 - Distributions
This unit will present concepts of distributions.

Topics

- Univariate distributions
  - T, Laplace, Gamma
- Joint probability distributions
- Central limit theorem

Unit 16 - Relationships between Variables
This unit will demonstrate relationships between variables.

Topics

- Correlation, Covariance
- Confidence Intervals
- Hypothesis Testing

Unit 17 - Regression
This unit will provide an introduction to regression and lay the foundation for further work and projects in regression.

Topics

- Linear Least Squares
- Intro to regression

Unit 18 - Advanced Regression
This unit continues the study of regression and introduces higher concepts.

Topics

- Linear Regression
- Multiple Regression
- Nonlinear Relationships

Unit 19 - Regression Projects
This unit will take material from previous units on regression to application.

Topics
- Pull data from web and clean data

Unit 20 - Regression Projects
This unit continues taking material from previous units on regression to application.

Topics
- Perform regression analysis, and make prediction given data

APPLICATION AND ADMISSIONS

Admissions Philosophy

Admission to Flatiron School is competitive with only approximately ten percent of applicants being admitted into the Software Engineering program. The decisions are made by Flatiron School employees who are admissions professionals who have no financial incentive to admit a particular applicant. We developed a rigorous applications process which enables Flatiron School to successfully place such a high percentage of graduates in jobs that are directly related to their education; we do the hard work up front in the admissions process so that we are confident that our applicants will continue to have successful outcomes as borne out by Flatiron School's audited job placement data of applicants that have already completed the program.

Admission Requirements

- Be at least 18 years old
- Have a high school diploma or equivalent
- Successfully complete the application process

Application Process

The application process for the **Data Science** and **Software Engineering** programs is designed to determine if the applicant will be able to learn the material within the amount of time allotted. The application process for the **Online Data Science** and **Online Software Engineering** programs is designed to determine if the applicant is motivated to learn and sufficiently disciplined to successfully complete an online program.

All programs require a written application, cultural interview and technical interview:

- *Written application*: All incoming applications are ranked on a scale of 1-5 as listed below:
    - **5 - Top Priority**, will lead to an invitation to the cultural interview if the applicant has an exceptional application which may include evidence of two or three of the following areas:
        - A history of significant accomplishments including a top-tier education, experience working for a prestigious employer, regular career progression and promotions, etc.
        - A desire to learn/passion for coding including completion of online tutorials, quantitative background, some computer science classes, experience in tech-adjacent roles, etc.
        - A compelling personal story that can add a unique perspective to the class composition such as military background, poet, circus performer, pro-athlete, etc.
    - **4 – Priority,** will lead to an invitation to the cultural interview if applicant has a very strong application which may include evidence of at least one of the following areas:
        - A history of significant accomplishments including a top-tier education, experience working for a prestigious employer, regular career progression and promotions, etc.
        - A desire to learn/passion for coding including completion of online tutorials, quantitative background, some computer science classes, experience in tech-adjacent roles, etc.
        - A compelling personal story that can add a unique perspective to the class composition such as military background, poet, circus performer, pro-athlete, etc.
    - **3 – Interview,** the application is not as strong as those receiving higher ratings but the applicant may be invited to the cultural interview; priority will first go to applicants receiving 4's and 5's. Looking for evidence of:
        - High effort application/good personal story
        - Some code background–demonstrated effort to learn
        - Evidence of passion
        - Evidence of grit/stick-to-itiveness/disciplined to learn
    - **2 – Conditional Interview** – The application is not as strong as those above; it may have two of the following:
        - High effort application/good personal story
        - Some code background–demonstrated effort to learn
        - Evidence of passion
        - Evidence of grit/stick-to-itiveness/disciplined to learn
    - **1 – Reject** – Application is poor or indicative of very low-effort; applicant is rejected without an interview.
- *Cultural interview*: All applicants who are interviewed are scored in the following four areas:
    - *Passion*
        - Can demonstrate a general desire to learn, through programming, new experiences etc.;

        - Can express a deep interest in code; has a personal history of interest in technology;
        - Has researched careers in technology, can describe the kind of work they think they would enjoy.
    - *Maturity & Collaboration*
        - Understands and is excited about the, full time program structure;
        - Can express a history of successfully working in groups;
        - Is fully committed to the rigorous program demands and expectations;
        - Congenial and possesses good verbal communication skills;
        - Demonstrates versatility by describing hobbies/interests outside of code
    - *Hirable*
        - Articulate verbal communication skills;
        - Establishes eye contact and displays professionalism;
        - Shows positive energy, enthusiasm and interest in the interview topic;
        - Demonstrates career direction, ability to articulate some sort of specific job goal/vision into the future.
    - *Technical Background*
        - 1: No experience with code
        - 2: Very basic, i.e. started a free track online
        - 3: Enthusiastic beginner--some online coursework, has tried several languages, maybe read some books
        - 4: Advanced beginner--has gone deep in online courses or tutorial, tried to build a website, taken some classes/read some books
        - 5: Has taken a full course, built websites, understands at least one language deeply
    - Passion, maturity and collaboration and hirable are each scored from 1-5 as follows:
        - (1) Did not demonstrate
        - (2) Demonstrated sporadically
        - (3) Average
        - (4) Above Average
        - (5) Outstanding, among the best we've seen
    - Applicants with high scores in each category are invited to complete a technical interview.
- *Technical interview***:** The technical interview is designed to measure an applicant's talent or potential with code and is conducted by a member of our instructional staff. The outcome is a measure of an applicant's current coding ability, but more importantly how well we expect the applicant to learn in the future. Online applicants are required to complete the free assessment which is located at Learn.co and made available to the student upon submission of the written portion of the application.

    - Applicant aptitude can be demonstrated by:
        - The nature of the questions asked by the applicant in the interview;
        - The applicant's responses to questions designed to expand on the applicant's code as written;
        - Challenging the applicant to think through a new code problem and explain their solution;
        - Demonstration of sophisticated coding skill and passion in advance of our programs.
        - Online programs - Upon completion of this assessment, Flatiron School's instructors assess the technical aptitude of the applicant and his/her motivation (based upon factors such as the time it took to complete the online assessment and interview responses). An instructor then conducts the technical interview with the applicant asynchronously.
    - As part of the technical interview, applicants are provided a technical challenge which requires little coding knowledge to complete.
    - A major component of the interview is the review of their work on the technical challenge and is designed to evaluate four areas of each applicant:
        - <u>Ability to Communicate About Logic/Thought Process</u>: We assess whether the individual understands what they have written, if they can communicate about code in a logical way, and how well they explain the steps taken and research conducted in the process.
        - <u>Evidence of Passion</u>: We determine if the applicant worked hard on the challenge and observe their learning process given their experience, and whether the applicant challenged him/herself given their existing knowledge.
        - <u>Ability to Think Quickly/Integrate New Information</u>: We observe whether the applicant has any idea where to start when presented with a new challenge to determine if the applicant is someone who can navigate through unknowns, ask questions and make new connections.
        - <u>Evidence of Problem Solving Skills</u>: This assessment looks to see how well the applicant can break down a problem, independently identify the steps to solve the problem and smoothly move forward from task to task.
        - <u>Online programs only: Motivation and Discipline</u>: The instructor determines if the applicant is mature enough to stay focused to complete the online program in a timely manner.
- *Interviewer Decision***:** At this point, a decision is made to admit or not admit the applicant to Flatiron School's on-campus program.
- *Online Self-Paced programs* - If admitted to the **Self-Paced Online Software Engineering** program **or Self-Paced Online Data-Science** program, the

applicant will be provided an estimate of the time it will take to complete the online program and the estimated program cost based upon the time the applicant used to complete the free assessment.

- *Required Pre-Work for On-Campus Students*: Once an applicant is admitted to the Software Engineering or Data Science program and before on-campus instruction begins, the student must complete the required preparatory work.
  - The purpose of this preparatory work is to provide real examples of the curriculum so that applicants are fully aware of the demands of the program prior to committing to it financially.
  - The Pre-work is free and available online: http://prework.flatironschoo.com/web-development/

The application process for the **UX/UI Design** program involves a written application and a 45-minute, in-depth interview. If an applicant does not meet the minimum viable level of potential for the program, he/she will not be accepted but may be given the chance to reapply in the future. If the applicant shows a minimum viable level of potential for the program, but needs to demonstrate greater understanding of UX or UI design, he/she will be offered the opportunity to complete the self-paced Design Essentials module. If the applicant successfully completes the module and receives a positive recommendation from the DE instructor, he/she will be offered the chance to continue into the next scheduled cohort's Virtual Phase. Highly qualified applicants will be accepted directly into the next cohort and promising applicants may need to take additional steps, including a second interview or completing a design challenge within a set number of business days, before being offered admission.

Acceptance Notification

All successful applicants are notified of their acceptance via email. There is no provisional acceptance at Flatiron School.

As a prospective student, you are encouraged to review this catalog prior to signing an enrollment agreement. You are also encouraged to review the School Performance Fact Sheet, which must be provided to you prior to signing an enrollment agreement.

Application Deadlines & Enrollment Periods

Flatiron School has a rolling admissions process because most new Program cohorts start approximately every six weeks, and Stand Alone Course cohorts are scheduled to begin every month. New cohorts of the UX/UI Design program start approximately every fourteen weeks. Please contact info@flatironschool.com for upcoming start dates.

Applicants are encouraged to submit their application four to six weeks prior to the next scheduled cohort start date; in certain limited circumstances, Flatiron School may accept an application two weeks prior to the next scheduled start date. Students are not permitted to join a cohort once classes have begun.

The Software Engineering, Full-Time Online Software Engineering, Data Science, and Full-Time Online Data Science programs end 15 weeks after their start date. The Part-Time Online Software Engineering and Part-Time Online Data Science end 30 weeks after their start date. The UX/UI Design program ends 24 weeks after the start date (Adjusted for school holidays, if applicable). The Intro to Front End Web Development and Intro to Data Science programs end 10 weeks after their start date (adjusted for school holidays, if applicable). The Self-Paced Online Software Engineering and Self-Paced Online Data Science end when the student has completed all the program coursework, or after 15 months, whichever comes first.

Flatiron School reserves the right to delay or cancel the start of an on-campus program for reasons such as low enrollment.

Credit for Previous Education/Training and Transferring Credits

NOTICE CONCERNING TRANSFERABILITY OF CREDITS AND CREDENTIALS EARNED AT OUR INSTITUTION.

The transferability of credits you earn at Flatiron School is at the complete discretion of an institution to which you may seek to transfer. Acceptance of the certificate of completion you earn in your educational program is also at the complete discretion of the institution to which you may seek to transfer. If the credits or certificate of completion that you earn at this institution are not accepted at the institution to which you seek to transfer, you may be required to repeat some or all of your coursework at that institution. For this reason you should make certain that your attendance at this institution will meet your educational goals. This may include contacting an institution to which you may seek to transfer after attending Flatiron School to determine if your credits or certificate of completion will transfer.

Flatiron School does not award credit for prior experiential learning. Flatiron School does not accept credit earned at other institutions or through challenge examinations or achievement tests. Flatiron School does not accept Ability-to-Benefit students.

Flatiron School does not offer any programs for credit and does not measure its programs in traditional academic "credit hours." As such, Flatiron School does not guarantee the transferability of its coursework to any other institution unless there is a written agreement with that institution. Flatiron School has not entered into an articulation or transfer agreement with any other institutions.

Flatiron School does not provide visa services, nor does it vouch for student status for international students.

Postponement of Start Date

Postponement of a starting date, whether at the request of the school or the student, requires a written agreement signed by the student and the school. The agreement must set forth:

a) Whether the postponement is for the convenience of the school or the student, and;
b) A deadline for the new start date, beyond which the start date will not be postponed.

If the course is not commenced, or the student fails to attend by the new start date set forth in the agreement, the student will be entitled to an appropriate refund of prepaid tuition and fees within 30 days of the deadline in accordance with the school's refund policy.

Students Seeking Reasonable Accommodation

Information pertaining to an applicant's disability is voluntary and confidential. If this information is supplied, it will be used to reasonably attempt to overcome the effects of conditions that limit the participation of qualified disabled students. All inquiries about accommodations should be made to admissions@flatironschool.com. Reasonable accommodations will be made on an individual basis. It is the responsibility of persons with disabilities to seek available assistance and to make their needs known as soon as those needs arise.

SCHOOL POLICIES

Attendance, Tardiness, and Leaves of Absence

Attendance Policies and Procedures for On-Campus Students

Software Engineering and Data Science students are required to be on campus Monday through Friday, 9AM through 6PM. There will be an hour break for lunch each day.

Students enrolled in the Intro Data Science and Intro to Front End Web Development courses are expected to be on campus during class times. There will be one ten-minute break each evening.

The classroom instructor maintains the attendance roster. The attendance roster is kept at the school at all times. In the event that a substitute instructor is used, the substitute must initial the day's attendance. Attendance will be taken approximately ten minutes after class begins.

It is the student's responsibility to make up any work missed due to absences. Students should schedule time with the relevant instructor to ensure they understand the material, requirements, and expectations from assignments.

Tardiness

Students who miss the attendance roll call are considered tardy and are considered unexcused, unless the instructor has granted an excused absence in advance. For any missed class time, including tardiness and excused absences, students must make up their work. Students who do not make up their work within 5 days of their return to class will receive an unexcused absence for the missed time.

Absence Tracking Process

A Flatiron School representative will email any student who is absent a first, second, and third time. In addition, the absences will be recorded on the Absence/Drop Record Form.

A representative from Flatiron School will call any student who is absent a fourth or fifth time to advise the student of possible consequences such as being withdrawn. A school representative will provide the student with written documentation of potential consequences on their next visit to campus, at which time the student must sign the Absence/Drop Record Form acknowledging receipt of the document.

If absent a sixth time, the student will be informed in writing that the module must be repeated and the student will be subject to paying additional tuition for the repeat. The student will not be able to repeat more than one module per program due to absences.

Students who are absent for six consecutive days without notifying the school will be considered withdrawn. The school will contact the student by telephone and in writing and formally advise the student of this status. Results will be recorded on the Absent/Drop Form. If appropriate, the student will be informed that a leave of absence status is available. The student will also be informed that the tuition paid will remain valid (and may be subject to any applicable refund policy), unless there has been a significant financial or educational change to the program, in which case, extra tuition may be charged. The school will then request that the student respond in writing and provide reasons for withdrawal. The results will be recorded in the Absent/Drop Record Form.

An absence is considered excused if the student notifies the instructor before missing class and it is an approved reason. Approved reasons for absences include but are not limited to: Illness, death, or birth in the immediate family, and other valid reasons at the discretion of the lead instructor. If a student is absent for any other reasons, or doesn't give the lead instructor notice, it is considered an unexcused absence. In addition to absences, a tardy (unexcused or excused) where work is not made up, or three tardies (of 30 minutes late or more) will count as one unexcused absence.

Attendance Policy for Online Students

While attendance is not taken for online programs, student progress is recorded automatically via the *Learn.co* platform. A student who fails to complete a lesson within any consecutive 14-day period will be contacted by an instructor via email. If the student does not respond or complete a lesson within the following 10 days, they will be contacted again via email, and possibly by phone.

Students who have not completed any lessons within 30 consecutive days, and have not been responsive to outreach from instructors may be informed in writing that the module must be repeated. **See Enrollment Agreement regarding time allowed for completion.**

Leave of Absence

An on-campus student may request a Leave of Absence at any time by requesting a "Leave of Absence Request Form" from their instructor. The LOA can be issued for a minimum of 7 days and a maximum of 30 days. In extenuating circumstances, the Director can grant an extension up to an additional 30 days. Any student whose LOA is 30 days or greater will need to be tested upon return to determine their retention level and placement back into the program. Students will be evaluated upon return from a Leave of Absence and placed at the appropriate part of the program based upon their understanding of the material at that point in time. If a student fails to return to school on the date listed, the student may be dismissed from the school.

Grading System

Flatiron School uses a pass/fail grading system for all assigned work, blog posts and assessments, including the final assessment. Students can log in to their Learn.co account to see their grades at their convenience. Grades are posted within two days of completion of the work.

Grievance Procedure

Flatiron School strives to address student dissatisfaction as soon as possible. If a student has a complaint or concern and desires to file a formal complaint relating to student conduct, curriculum, teacher conduct, or anything else at Flatiron School, he or she should email complaint@flatironschool.com as soon as possible to schedule an in-person meeting (or via phone for online students). Students should describe the complaint in as much detail as possible and include the words "Formal Complaint" in the subject line of the email. An administrator will follow up within 24 hours to review questions or discuss next steps. Flatiron School will not retaliate against any student who files a grievance or complaint against the school. Nothing regarding this internal Grievance Procedure prevents a student from bringing a complaint with the regulatory agency(ies) listed below (see "Regulatory Notices").

Intellectual Property Ownership

Flatiron School respects intellectual property rights and ownership. These policies provide guidance as to ownership of intellectual property. Flatiron School will provide opportunities for students to create projects or contribute their own writing (blog posts), designs, images, code or other content as part of or in connection with the current modular curriculum. Students are solely responsible for their own Student Content. Flatiron School does not endorse Student Content and has no responsibility or liability for Student Content. Each student represents and warrants that his or her Student Content is original and he or she has the unrestricted right to share such Student Content.

Plagiarism Policy

Students are expected to attend Flatiron School with the utmost transparency and honesty. This means the cheating, plagiarism or any unauthorized assistance is strictly forbidden.

Cheating occurs when a student avails her/himself of an unfair or disallowed advantage which includes but is not limited to:

- Theft of or unauthorized access to an exam, answer key or other graded work from previous course offerings.
- Use of an alternate, stand-in or proxy during an examination.
- Copying from the examination or work of another person or source.
- Submission or use of falsified data.
- Using false statements to obtain additional time or other accommodation.
- Falsification of academic credentials.

Plagiarism is defined as the use of work or concepts contributed by other individuals without proper attribution or citation. Unique ideas or materials taken from another source for either written or oral use must be fully acknowledged in academic work to be graded. Examples of sources expected to be referenced include but are not limited to:

- Text, either written or spoken, quoted directly or paraphrased.
- Graphic elements.
- Passages of music, existing either as sound or as notation.
- Mathematical proofs.
- Scientific data.
- Concepts or material derived from the work, published or unpublished, of another person.

Unauthorized assistance refers to the use of sources of support that have not been specifically authorized in this policy statement or by the course instructor(s) in the completion of academic work to be graded. Such sources of support may include but are not limited to advice or help provided by another individual, published or unpublished written sources, and electronic sources. Examples of unauthorized assistance include but are not limited to:

- Collaboration on any assignment beyond the standards authorized by this policy statement and the course instructor(s).
- Submission of work completed or edited in whole or in part by another person.
- Supplying or communicating unauthorized information or materials, including graded work and answer keys from previous course offerings, in any way to another student.
- Use of unauthorized information or materials, including graded work and answer keys from previous course offerings.
- Use of unauthorized devices.
- Submission for credit of previously completed graded work in a second course without first obtaining permission from the instructor(s) of the second course. In the case of concurrent courses, permission to submit the same work for credit in two courses must be obtained from the instructors of both courses.

Probation, Dismissal and Readmission

Academic Probation
Students who fail to make Satisfactory Academic Progress will be placed on academic probation. The probation period is thirty days, during which the student will receive additional academic support. If, at the end of the 30-day period, the student still is not making satisfactory academic progress, they will be dismissed according to the Termination policy.

Dismissal
Dismissed students will be notified in writing. Students may appeal this decision in to the School Director within 5 business days of receiving the Notice of Dismissal. In the School Director's absence, students may appeal to the VP of Education or VP of Online Education (as applicable). The School Director, VP of Education or VP of Online Education will provide a written response to the appeal within 3 business days. This response will indicate whether the student may be reinstated, and if so, the terms of reinstatement.

Readmission
Students wishing to re-enroll after a period of non-attendance will be required to reapply. Students who were terminated due to unsatisfactory attendance, unsatisfactory academic progress or breach of the Code of Conduct may be re-admitted only at the discretion of Flatiron School. Any increase in tuition will be applied to the student's account.

Progress Standards/Performance Assessment

Assessment Overview and Process
Throughout each program and in the five modules listed below, students will be required to complete assignments designed to assess their mastery of key subject areas in the curriculum.

For the Software Engineering and Online Software Engineering programs, the assessments include, but may not be limited to:
- Ruby & Object-Oriented Programming
- Intro to Web Development and Data Persistence
- Advanced Web Development with Rails
- Creating Interactive and Performant Front-Ends with JavaScript
- Scalable Front-End Design with React

For the Data Science and Online Data Science programs, the assessments include, but may not be limited to:
- Data Exploration and Analysis
- Probability and Statistics for Data Science
- Machine Learning: Regression Optimization and Big Data
- Machine Learning: Machine Learning: Classification and Deep Learning
- Data Science Projects

Assessments will be used primarily to identify weaknesses that need to be addressed through remediation. While the assessments are not numerically graded, they will be used as a benchmark for student progress. Assessments are given to students after they complete each module of the program. Completion of each assessment is necessary for completion of the program. Assessments are designed to test a student's ability to assimilate all that they have learned in the module and to give students a chance to verbalize their technical skills (in preparation for the technical interviews they'll receive from employers upon graduation).

Students complete the assessment individually and schedule a 45-60-minute review session (in person or asynchronously depending upon the program delivery method). The instructor will review the student assessment with the student using a standardized grading rubric.

In addition to writing the code behind their application, students are expected to complete a 30-minute coding demonstration to show how they approach problems. After completing the project, students then record another 5-minute demonstration of how the application works. Finally, students must write a blog post explaining what they wrote and their process for completing the work. All of these various additional requirements give instructors a better picture of a student's technical and communication abilities. Depending on performance on each assessment, students may be assigned additional work and asked to retake the assessment at a later date.

Satisfactory Academic Progress - On-Campus Programs and Courses

The Data Science and Software Engineering programs are broken into five three-week modules. Assessments are scheduled and delivered after completion of the second week of each module. If a student does not complete the assessment satisfactorily, the student will be given remediation materials, as well as individual support from instructors and be invited to re-take the assessment at the end of the third week of that module.

Should the student fail to pass the module on the second attempt, he/she will be given the option to repeat that module in its entirety. If the student fails to complete the module again after having already repeated, the student will be dismissed from the program and may receive a prorated refund of any tuition paid according to the applicable refund policy.

The UX/UI Design program is broken into five modules. During the fifth week of the first module, Design Essentials, designers will participate in a mandatory performance evaluation with an instructor. The instructor will make a determination of the designer's ability to continue the rest of the program. Students who pass their mandatory performance evaluation continue to Phase 2 – Virtual Phase and select whether to pursue the UX or UI track. For the remainder of the phases, students are expected to submit and complete all client deliverables on time. Because the program is treated like a professional experience as much as possible, each student's performance is measured through external feedback from instructors and self-evaluation. When issues are found with participation, teamwork, timeliness, effort, other areas, they are brought to the student as soon as possible with the hope of making improvements. If issues persist, more formal performance meetings take place with the student. Finally, if the instructor continues to see no improvement in the issues that impact the student's work and client deliverables, the student may be dismissed from the program and may receive a prorated refund for any tuition paid according to the applicable refund policy.

Satisfactory Academic Progress – Full-Time and Part-Time Online Programs

The Full-Time Online Data Science program and Full-Time Online Software Engineering program are broken into five four-week modules. Assessments are scheduled and delivered after completion of the fourth week of each module. The Part-Time Online Data Science program and Part-Time Online Software Engineering program are broken into five eight-week modules. Assessments are scheduled and delivered after completion of the seventh week of each module. For both the Full-Time and Part-Time versions of the online programs, if a student does not complete the assessment satisfactorily, the student will be given remediation materials, as well as individual support from instructors. Students enrolled in the Full-Time program and Part-Time program will be invited to re-take the assessment at the end of the fourth week of that module and eighth week of that module, respectively.

Should the student fail to pass the module on the second attempt, he/she will be given the option to repeat that module in its entirety. If the student fails to complete the module again after having already repeated, the student will be dismissed from the program and may receive a prorated refund of any tuition paid according to the applicable refund policy.

Satisfactory Academic Progress – Self-Paced Online Programs

For the Self-Paced Online Data Science and Self-Paced Online Software Engineering programs, students complete the five modules on their own schedule and will be given the opportunity to schedule an assessment with an instructor upon completing the relevant module. If a student does not complete the assessment satisfactorily, the student will be given remediation materials, as well as individual support from instructors and be invited to re-take the assessment at his or her earliest convenience. Should the student fail to pass the module on the second attempt, he/she will be given additional materials and remediation and be invited to re-take the assessment again. Students may re-take assessments as often as necessary until they demonstrate the ability to complete them satisfactorily, but not more than once in any two-week period.

Student Conduct

Code of Conduct

This Code of Conduct applies to all Flatiron School students regardless of which program the student is enrolled in, or which method of delivery is used for the program.

Flatiron School believes our community should be open for everyone. We are committed to providing a friendly, safe, and welcoming environment for all, regardless of gender, sexual orientation, disability, ethnicity, religion, preferred operating system, programming language, or text editor. Whether learning online or on-campus, the idea of learning together as a community is baked into the Flatiron School philosophy via its Learn.co platform… it strives to be the most effective online community for the world to meet, to create, to exchange and to pursue mastery of knowledge.

By being committed, engaged and caring people, Learners will define the tone of the community, and will give shape to the product experience. It is our hope and expectation that Learn will reflect the best in all of us.

We expect all faculty, teachers, employees, mentors, students, guests, friends, basically everyone involved with the school, to help us create a safe and positive environment for everyone. Let's build a place where we can achieve more together than we could ever achieve alone.

We expect everyone to:

- Be considerate, respectful, and collaborative when engaging with one another.
- Refrain from demeaning, discriminatory, or harassing behavior and speech.
- Create a positive impact on everything around them.
- Participate in an authentic and active way.
- To consider it their duty to help lift up Learners who are struggling, to encourage them, to help them along
- To contribute to improving the curriculum for the benefit of the whole world

We think that everyone is born with a little voice inside them. That little voice goes off whenever they are about to do or say something stupid. Most of the time, we don't listen to it. Start listening to that little voice. If you think you are about to do something that has the potential, even the very slightest potential, of offending or hurting someone, just don't do it. It is that simple. But if you need a more explicit list:

Unacceptable behaviors include harassing, abusive, discriminatory, derogatory or violent conduct. Harassment includes: offensive comments related to gender, sexual orientation, race, religion, disability; inappropriate use of sexual images; deliberate intimidation, stalking, or following; threats of violence or violent language directed at another person; harassing photography or recording; sustained disruption of talks or events; inappropriate physical contact, and unwelcome sexual attention.

Conduct Violations

Students who violate the Code of Conduct will be disciplined as follows:

1st Offense: Verbal Warning
2nd Offense: Written Warning
3rd Offense: Final Written Warning
4th Offense: Dismissal

Serious violations of the Code of Conduct may result in immediate dismissal at the discretion of the School Director.

Notice regarding underage drinking on WeWork premises: Flatiron School is located within the WeWork offices which provides beer on tap during working hours, so long as a certified WeWork employee is present on-site. Consumption of alcohol on the WeWork premises by any Flatiron School student who is under the legal drinking age of 21 years is prohibited and is cause for immediate dismissal.

Reporting Harassment/Abuse of the Code of Conduct on Slack

If you are being harassed, notice that someone else is being harassed, or have any other concerns, please contact Learn administrators immediately (support@learn.co).
Please follow these guidelines for the report:

1. Report within 24 hours of occurrence.
2. Include a screenshot of the conversation or harassment incident.
3. Provide a written description outlining the incident.

Steps Flatiron School will take:

1. Learn administrators will review the report.
2. Learn administrators will discuss incident with all parties involved.
3. Based on an evaluation of the evidence, report, and discussion, Learn administrators will take action as they see fit.

For more information please read our Terms of Service and Code of Conduct

Recordkeeping, Organization, and Maintenance

Maintenance and Security of Records

All student records are maintained exclusively in electronic format. Student transcripts are retained permanently, and other pertinent student records are maintained for a minimum of 5 years from the student's date of completion or withdrawal. Student record data is housed on the secure Flatiron School server, which can produce exact, legible printed copies of stored records.[1] The Flatiron School maintains records in a manner that is secure from damage or loss, with appropriate backups kept. Personnel who are able to operate the relevant systems in order to access all records (and are able to explain the process for doing so) will be on hand during all normal business hours.

Contents of Student Records

All student records are electronic, and include the elements listed below. Student transcripts are retained permanently, and other pertinent student records are maintained for a minimum of 5 years from the student's date of completion or withdrawal.

- Student Contact Information
    - Name

[1] It is Flatiron's internal policy to respond to student record requests within one business day of a student making such request. However, such records are stored in an immediately-retrievable format.

    - Address
    - Email address
    - Telephone number
    - Date of birth
- Admissions Information
    - Proof of High School Diploma or equivalent
    - Proof of age greater than or equal to 18
    - Any other application documents, including
        - basis for admission decision (including any transcripts or other records or prior education, training or experience received),
        - basis for any transfer credits accepted (including, if applicable, to the extent such credits are awarded based on prior experiential learning), and
        - records of any entrance examination or other instrument used to measure academic ability or educational achievement.
- A transcript for each student showing:
    - The certificate granted and the date on which that certificate was granted (if applicable).
    - The courses and units on which the certificate was based (if applicable).
    - The grades earned by the student in each of those courses (if applicable).
    - The courses or other educational programs that were completed, or were attempted but not completed, and the dates of completion or withdrawal.
    - Credit awarded for prior experiential learning, including the course title for which credit was awarded and the amount of credit (if applicable).
    - Credit for courses earned at other institutions (if applicable).
    - Credit based on any examination of academic ability or educational achievement used for admission or college placement purposes (if applicable)
    - The name, address, website address, and telephone number of the institution.
- Academic Information
    - Copy of Enrollment Agreement and any other document(s) signed by student
    - Program start date
    - Program completion date, or date of termination, cancellation or withdrawal
    - Attendance Record (including dates of any leaves of absence)
    - Student Progress Record
- Payment Information
    - Records relating to financial payment and refunds
    - Receipt Form
    - Student Ledger
    - Refund Calculation
- Miscellaneous
    - Record of any student grievance and subsequent action and resolution

  - For all graduates: Information on subsequent employment, salary, and employment start date, or reasons for not seeking employment

Academic and financial records will be stored separately.

Flatiron School will take reasonable steps to protect the privacy of personal information contained in a student's record, as specified in the Flatiron School Privacy Policy found at: https://flatironschool.com/privacy-policy/.

Student Record and Transcript Request Procedure

Students may request a copy of their student record or transcript emailing the request to: info@flatironschool.com. Students should send the email from the email address on file with Flatiron School, and include the name used while enrolled, last session of enrollment, and a phone number Flatiron School to call with any questions.

Transfer Between Flatiron School Locations

Students who transfer from one Flatiron School location to another will not have to repeat any completed modules, provided the student remains in the same program or course. The transfer timing must coincide with the start of the next module at the new location. Please consult with your lead faculty member prior to planning a move.

FACILITIES, EQUIPMENT AND RESOURCES

Description of Campus Facilities and Equipment

Classes are held at Flatiron School, located at 1460 Mission Street, San Francisco, CA 94103. The campus is equipped with high speed internet, classrooms, tables and chairs, conference rooms and lounge areas. There is complimentary coffee available on campus. Students have access to refrigerators and lockers to store their food or belongings while on campus. The school is fully handicapped accessible.

Flatiron's Campus is spacious, with four classroom spaces (one large-sized space and three breakout spaces), one large primary lab area and one breakout lab area, and a pantry. This space is designed for flexible use, and classroom spaces can readily be repurposed for lab purposes, and vice versa. There are a total of 66 desks, in addition to further space available in the pantry area. Flatiron School will utilize classrooms, desks, internet access, whiteboards and other standard teaching materials to provide instruction.

Students must provide their own laptop with an up-to-date operating system. The student's laptop represents the primary mode for students to access and interact with the learning materials and there is no other significant physical equipment necessary in order for students to achieve the educational objectives of each program. The campus is equipped with high speed FiOS Internet Service. While enrolled, students have unrestricted access to Flatiron School's proprietary Learning Management Software via Learn.co. This includes access to all of Flatiron School's curriculum, support materials, and online community. Flatiron's team of Technical Coaches via the Learn.co platform's Ask a Question Feature are also available to provide live support for students

Monday-Friday, 9AM-1AM ET, and Saturday-Sunday, 9AM-12AM ET. Flatiron also maintains a collection of help articles, with advice and answers to frequently asked questions from the Flatiron School Team at help.learn.co.

The privacy policy can be found here: https://learn.co/privacy

Flatiron's programs are designed to prepare students for careers such as web developers (15-1254), web designers (15-1255), software engineers (15-1252), computer/research scientists (15-1220), database administrators (15-1242), data architects (15-1243), data engineers (15-1243), UX designers (15-1255), and UI designers (15-1255).[2]

School Hours, Holidays and Vacation Schedule

Flatiron School is open from 9 AM to 6 PM, Monday through Friday. Enrolled students have access to the common areas of the campus 24/7 except for the following holidays:

- Birthday of Martin Luther King, Jr.
- President's Day
- Memorial Day
- Independence Day (4th of July)
- Labor Day
- Columbus Day
- Veterans Day
- Thanksgiving Day
- All the days between December 24th and January 1st (inclusive)

In the event of unexpected closures due to inclement weather or otherwise, students will be notified via their primary email address on file.

Academic Resources

Faculty Availability and Academic Support

Full-time faculty are available to meet with students on-campus from 9:00AM-6:00PM Monday through Friday, excluding school holidays. Students may schedule an appointment to with faculty during these hours, excluding hours in which the faculty are scheduled to teach.

In addition, live instructional support is available from instructors online from 9 AM – 6 PM EDT Monday through Friday, excluding school holidays. Additional instructional support from technical coaches is available Monday through Friday from 9 AM – 1 AM EDT, and Saturday and Sunday from 12 PM – 12 AM EDT.

[2] These codes represent the applicable U.S. Department of Labor Standard Occupational Classification ("SOC") codes of the occupations for which Flatiron's programs prepare students.

Online Resources and Digital Library

While enrolled, all students will have unrestricted access to a digital library of resources, available 24 hours a day, 7 days a week by visiting Flatiron School's proprietary Learning Management Software via *Learn.co*. This also includes access to all of Flatiron School's curriculum, support materials, and online community. During the enrollment process, students are provided with login credentials sufficient to access the learning resources. All resources included in the Learn.co platform are available to students without additional charge.

Flatiron subject matter experts have ensured that the learning resources available through the Learn.co platform are sufficient to support each educational program, and to enable students to pursue inquiries, searches for information and documentation, and assignments connected with their programs.

Due to the rapidly changing nature of the content of the curriculum for our programs, maintaining a digital library allows us to continually modify and upgrade the available materials so that Flatiron School students have access to the most current information available.

Slack Communication

Customizing Slack and Slack Integrations

We encourage students to make the Learn.co Slack a fun place to work and play as a community. That being said, please keep in mind that this is a community centered on learning.

When creating or adding a Slackbot response, Loading Message, App or Integration be thoughtful in your execution. We discourage adding customizations that detract from the flow of conversation and focus of the community. Think of the needs of the whole community you are creating for - will anyone be offended, discouraged, or annoyed by the customization you are considering creating? If the answer is yes, think of a different addition. Learn.co administrators reserve the right to remove any Slack customizations.

Using @channel

Do not use @channel in the Learn.co Slack. If you would like the attention of everyone on Slack in the moment, please use @here. Using @channel will notify every person in that channel, whether or not they are online at the moment - this is a very disruptive practice and highly frowned upon.

TUITION, FEES AND REFUNDS

Tuition & Fees

Software Engineering, Data Science, UX/UI Design programs

The tuition includes access to all materials needed to complete the program, with the exception of the UX/UI Design program. The total tuition is $17,000, with a deposit due within seven (7) days of notification of acceptance. Depending upon your selected method of payment, the balance is due at the start of the program.

Full-Time Online Software Engineering, Part-Time Online Software Engineering, Full-Time Online Data Science and Part-Time Online Data Science programs

The tuition includes access to all materials needed to complete the program. The total tuition is $15,000, with a deposit due within seven (7) days of notification of acceptance.

Self-Paced Online Software Engineering and Self-Paced Online Data Science programs

The tuition includes access to all materials needed to complete the program. The tuition is $9,600 with a deposit due within seven (7) days of notification of acceptance.

UX/UI Design program

The total tuition is $17,000, with a deposit of at least $1,000 due within seven (7) days of notification of acceptance. In addition to tuition and the deposit, students must pay a $50 fee for software (Sketch) paid directly to the vendor in the first phase of the program. During the fifth week of the program, each designer has a mandatory performance evaluation with the instructor, who makes a determination of the designer's ability to continue to the rest of the program. When approved to move to the next phase of the program, the balance is due the Sunday before the Virtual UX or UI phase start date.

Intro to Front End Web Development and Intro to Data Science Courses

Tuition for part-time courses will be $3,500, due at the start of the course. The tuition includes access to all materials needed to complete the course. Acceptable forms of payment include check, money order, or credit card.

Payment Options

Software Engineering & Data Science payment plan options:

Upfront Plan: Student makes an initial up-front deposit of $1,000 within 7 days of notification of acceptance. Student pays the total tuition amount, less the $1,000 deposit, by the first day of the program. Students choosing this payment option may pay by credit card, debit card, check, money order, or wire transfer.

Third Party Financing: Student makes an initial up-front deposit within 7 days of notification of acceptance. Student seeks to finance his/her tuition through a third party and arranges to pay the remainder of the tuition through a third party.

Full-Time Online Software Engineering & Full-Time Online Data Science payment plan options:

Upfront Plan: Student makes an initial up-front deposit of $500 within 7 days of notification of acceptance. Student pays an installment of $11,500 by the start of the program. After completing ten (10) weeks of the program, Student pays the remaining $3,000. Students choosing this payment option may pay by credit card, debit card, check, money order, or wire transfer.

Installment Plan: Student makes an initial up-front deposit of $500 within 7 days of notification of acceptance. Students may pay the total tuition, less the $500 deposit due within seven (7) days of notification of acceptance, over the course of 10 equal monthly installments. The first payment is reduced by the amount of the deposit. The first payment is due the first day of the program. Students choosing this payment option may pay by credit or debit card only, and will receive an invoice for each monthly payment.

Third Party Financing: Student makes an initial up-front deposit within 7 days of notification of acceptance. Student seeks to finance his/her tuition through a third party and arranges to pay the remainder of the tuition through a third party.

Part-Time Online Software Engineering & Part-Time Online Data Science payment plan options:

Upfront Plan: Student makes an initial up-front deposit of $500 within 7 days of notification of acceptance. Student pays an installment of $5,500 by the start of the program. After completing four months of the program, Student pays an additional $6,000. After completing the fifth month of the program, Student pays the remaining $3,000. Students choosing this payment option may pay by credit card, debit card, check, money order, or wire transfer.

Installment Plan: Student makes an initial up-front deposit of $500 within 7 days of notification of acceptance. Students may pay the total tuition, less the $500 deposit due within seven (7) days of notification of acceptance, over the course of 10 equal monthly installments. The first payment is reduced by the amount of the deposit. The first payment is due the first day of the program. Students choosing this payment option may pay by credit or debit card only, and will receive an invoice for each monthly payment.

Third Party Financing: Student makes an initial up-front deposit within 7 days of notification of acceptance. Student seeks to finance his/her tuition through a third party and arranges to pay the remainder of the tuition through a third party.

Self-Paced Online Software Engineering & Self-Paced Online Data Science payment plan options:

Installment Plan: Student makes an initial up-front deposit of $500 within 7 days of notification of acceptance. Students may pay the total tuition, less the $500 deposit, over the course of 10 equal monthly installments. The first payment is reduced by the amount of the deposit. The first payment is due the first day of the program. Students choosing this payment option may pay by credit or debit card only, and will receive an invoice for each monthly payment.

Third Party Financing: Student makes an initial up-front deposit within 7 days of notification of acceptance. Student seeks to finance his/her tuition through a third party and arranges to pay the remainder of the tuition through a third party.

UX/UI Design payment plan options:

Upfront Plan: Student makes an initial up-front deposit of $1,000 within 7 days of notification of acceptance. Student pays an installment of $8,000 after successfully completing the first 6 weeks of the program. After completing 12 weeks of the program, Student pays the remaining $8,000. Students choosing this payment option may pay by credit card, debit card, check, money order, or wire transfer.

Third Party Financing: Student makes an initial up-front deposit within 7 days of notification of acceptance. Student seeks to finance his/her tuition through a third party and arranges to pay the remainder of the tuition through a third party.

In the event that your access to and use of any of the modules in the program is terminated by Flatiron School for any reason other than your breach of the payment agreement or violation of the Code of Conduct after you have commenced but before you have completed any module of the program, subject to the applicable Refund Policy below, Flatiron School retains the discretion to either: (i) allow your user account to remain active for an additional six (6) months following the date of notice of termination of your user account, or (ii) provide you with a full or partial refund of all tuition or other payments made to Flatiron School for such program, pursuant to the Refund Policy below.

Please note that Flatiron School reserves the right to change pricing terms on a prospective basis with respect to any program at any time. If Flatiron School raises a price with respect to a given program, and you have selected a program requiring periodic payments, or if you have selected a periodic payment option (e.g., the Installment Plan), and commenced a module under that program before the effective date of that price increase, Flatiron School will honor your original price.

Tuition Refund Policy and Liability

STUDENT'S RIGHT TO CANCEL:
Students have the right to cancel the enrollment agreement and obtain a refund of charges, paid through attendance at the first class session, or the seventh day after enrollment, whichever is later.

To cancel, submit written notice to Flatiron School at admissions@flatironschool.com. Cancellation is effective on the date written notice of cancellation is sent. The institution shall make the refund pursuant to section 71750 of the Bureau's Regulations. If Flatiron School sent any physical copies of the first lesson and materials before an effective cancellation notice was received, the school shall make a refund within 45 days after the student's return of the materials.

If the student has received federal student financial aid funds, the student is entitled to a refund of moneys not paid from federal student financial aid program funds.

Furthermore, if the student is eligible for a loan guaranteed by the federal or state government and the student defaults on the loan, both of the following may occur: 1) The federal or state government or a loan guarantee agency may take action against the student, including applying any income tax refund to which the person is entitled to reduce the balance owed on the loan. 2) The student may not be eligible for any other federal student financial aid at another institution or other government assistance until the loan is repaid.

Refund Policy: all programs except UX/UI Design

1. Students not accepted to the school are entitled to a refund of all monies paid. A student who cancels within seven (7) calendar days of signing the enrollment agreement or through attendance at the first class session, whichever is later, is entitled to a full refund of all tuition paid.
2. If a student begins instruction and withdraws or is discontinued for any reason after instruction begins but prior to completion of 61% of the scheduled program, Flatiron School will refund to the student a sum which is the exact pro rata portion of the student's prepaid but unused tuition. Students who have completed 61% or more shall not receive a refund and shall be charged 100% of tuition and fees.
3. The pro rata refund shall be the total amount owed by the student for the portion

of the educational program provided subtracted from the amount paid by the student, calculated as follows: the amount owed equals the charge per hour (total institutional charge divided by the number of hours in the program), multiplied by the number of hours the student completed prior to withdrawal.

4. Students are required to honor their loan obligations directly with the third-party financing providers irrespective of any refunds from Flatiron School.

Refund Policy: UX/UI Design

1. Students not accepted to the school are entitled to a refund of all monies paid. A student who cancels within seven (7) calendar days of signing the enrollment agreement or through attendance at the first class session, whichever is later, is entitled to a full refund of all tuition paid.
2. Although the entire program is 24 weeks in length, Flatiron will apply the below policy to the percentage of the program completed after the student has completed the Phase 1 – Design Essentials in the first six weeks of the program and has paid his/her full tuition (in other words, a student who completes Phase 1 will be considered for refund purposes to have completed 0% of the program, with the calculation of the proportion completed beginning from the start of Phase 2). If a student begins instruction and withdraws or is discontinued for any reason after instruction begins but prior to completion of 61% of the scheduled program, Flatiron School will refund to the student a sum which is the exact pro rata portion of the student's prepaid but unused tuition. Students who have completed 61% or more shall not receive a refund and shall be charged 100% of tuition and fees.
3. The pro rata refund shall be the total amount owed by the student for the portion of the educational program provided subtracted from the amount paid by the student, calculated as follows: the amount owed equals the charge per hour (total institutional charge divided by the number of hours in the program), multiplied by the number of hours the student completed prior to withdrawal.
4. Students are required to honor their loan obligations directly with the third-party financing providers irrespective of any refunds from Flatiron School.

Students are required to honor their loan obligations directly with the third-party financing providers irrespective of any refunds from Flatiron School. If a third party paid for tuition on a student's behalf, the refund will be made to that third party in the amount of the refund due (but in no event greater than what that third party paid to Flatiron School). If there is an excess balance of the refund after payment to that third party, that amount will be refunded to the student.

In case of prolonged illness, accident, death in the family, or other circumstances that make it impractical to complete the program, a refund that is reasonable and fair to both parties shall be made, but in no event will the amount refunded be less than that reflected in the applicable refund policy.

If a student needs to withdraw due to an emergency, such as personal or family illness or national service, they may be re-enrolled into another Flatiron School cohort following approval by the Director of Education. Please contact the Director of Education as soon as possible.

Refund Process: To formally withdraw, a student should contact admissions@flatironschool.com with the following information:

- Student Full Name
- Updated Contact Information
- Program Name
- Start Date
- Reason for Withdrawal.

All tuition refunds will be processed within 45 days of withdrawal. The official date of termination or withdrawal of a student shall be determined in the following manner:

2. The date on which the school receives notice of the student's intention to discontinue the training program; or
3. The date on which the student violates published school policy which provides for termination.
4. Should a student fail to return from an excused leave of absence, the effective date of termination for a student on an extended leave of absence or a leave of absence is the earlier of the date the school determines the student is not returning or the day following the expected return date.

The failure of a student to notify the school in writing of withdrawal may delay refund of any tuition due.

Financial Assistance

Neither Flatiron School nor its programs are accredited by an accrediting agency recognized by the United States Department of Education or any other accrediting agency. As an unaccredited institution, Flatiron School does not participate in any federal or state financial aid programs. If a student obtains a loan to pay for an educational program, the student will have the responsibility to repay the full amount of the loan plus interest, less the amount of any refund, and if the student has received federal student financial aid funds, the student is entitled to a refund of the moneys not paid from federal student financial aid program funds.

STUDENT SERVICES

Career Services

Career Services Philosophy

Flatiron School prides itself on its students' success in finding good paying jobs upon graduation from the Software Engineering Program. Its placement rate for 2016 graduates of the New York City campus is 97% with an average starting salary of $75,997. While placement rates are high, students are not guaranteed a job; they're guaranteed a full tuition refund if they don't get a job. Students who successfully complete one of Flatiron's Money Back Guarantee Qualifying Programs and who fully comply with Flatiron's job search requirements but do not receive a qualifying job offer within 180 days of commencing their job search will be eligible for a full tuition refund. Job search requirements include, but are not limited to, the following typical activities: replying to Flatiron's Career Services Team within a reasonable amount of time; completing mock interviews; and maintaining a personal job-search tracking tool. Please see the Career Coaching and Money-Back Guarantee for Paid Services agreement for more details regarding Flatiron's Money Back Guarantee. *Flatiron's Money Back Guarantee Qualifying Programs are currently the following programs: Software Engineering; Full-Time Online Software Engineering; Part-Time Online Software Engineering; Data Science; Full-Time Online Data Science; Part-Time Online Data Science; and UX/UI Design.*

In Flatiron School's five years offering the Software Engineering program, over 1,000 students have been successfully trained, and we have worked very closely with them as they launched their careers. Throughout this time, we've learned the specific methodology that is necessary for someone to be successful in the job search process. We expect similar results with our Data Science programs.

Flatiron School is unique in that we use audited placement data to ensure that the placement rates we list are accurate. When we launched in 2012, coding were not really an industry. Since then the industry has grown rapidly and has attracted attention from investors, larger educational institutions, etc. Our biggest fear has always been that, similar to what happened in for profit higher education sector, bad actors would enter the industry, grow quickly without regard for quality, and tarnish the reputation of the industry as a whole.

To ensure that our students are receiving the education that they expected and paid for, in 2014 we engaged a third-party CPA to "audit" our job placement statistics. Our hope was that we can force the industry towards more transparency and accountability. Since inventing this new method of outcomes reporting, we've had our data verified for every single graduate of the Software Engineering program. Further, we were invited to the White House in 2015 to help spread this standard as part of President Obama's TechHire initiative.

The Flatiron School career services program supports our mission: ***to enable the pursuit of a better life through education.***

Career & Placement Services Process

Flatiron School recognizes that admitting a high caliber candidate is the first and a very critical step to success. Given the intensity of our programs, we have a robust screening process. While we don't ask for traditional metrics like GPA, SAT score, or even college degree, we do look for a passion for the field, an ability to learn technical concepts, a dedication to securing a job, etc. To this end, all students must submit a lengthy written application and complete two interviews - one cultural and one technical, so that we can determine their ability to be successful in a program.

The second step is that once enrolled, students are expected to do a significant amount of work and ultimately pass five assessments to graduate from a program. The assessments are designed to mirror the types of questions asked in a formal interview setting, so that by the end, we have confidence that if a student can pass the assessment, they have achieved the level of competency to pass formal interviews and gain employment.

Placement Services

The final step is the career services process. Our career services team is broken up into two parts: coaching and business development.

- The coaching team works 1:1 with students through weekly 1:1 meetings, to help them with their job search. This includes resume review, mock interviews, reviewing cover letters and networking emails, coaching students on how to build and leverage their networks, finding networking events for them to attend, etc.
- The business development team works directly with employers to set up interviews for students.

Additionally, all students are expected to adhere to our "career services commitment," which is made up of the things we've seen reliably lead to successful job outcomes. These things include maintaining an active technical blog, continuing to practice coding after their program, being responsive to employer and coach emails within 48 hours, and so on. We've found that almost everyone who has adhered to this disciplined process is successful in finding employment; the coaches check in with students every week to make sure they are following through and to keep them motivated.

MISCELLANEOUS INFORMATION

Curriculum Review Process

Flatiron School uses a continuous loop process improvement method for the systematic review, evaluation, and modification of curricula and programs of study to ensure that its students are receiving the most current and up-to-date education in web development.

The value of an education from Flatiron School is based upon its nimbleness and responsiveness to market demands in the field of web development. This poses certain challenges to stay current, which we Flatiron School considers to be opportunities to offer cutting-edge programs. To provide students with the most current skills that are immediately valued in the marketplace, it requires our faculty to be constantly learning and attentive to even the slightest change in technology.

The Flatiron School curriculum is developed by senior faculty, all of whom have significant experience both teaching, and working as practitioners; they serve as the curriculum team and make the final decisions about curriculum revisions as provided from the following sources as well as their own research and inquiry. Flatiron School's curriculum content is so new and ever-changing that there are no peer institutions or industry benchmarks or other external resources that can reasonably be leveraged or compared. It is incumbent upon faculty to be on the front edge of the most recent technology trends.

To continually increase the efficacy of our courses, and ensure they are aligned to market, we conduct ongoing reviews both internally and with external third parties.
Internally, we evaluate the efficacy of our course materials with input from several sources:

- Career Services: The Career Services team collects ongoing market data from employers and students regarding the change in demand for specific skills and technologies and that information is regularly relayed back to the education team to continuously update the curriculum.
- Alumni Survey: We conduct regular alumni surveys to gain insight into how our curriculum can be improved. We incorporate feedback from both recent alumna about curriculum that would help them enter the job market, as well as seasoned alumna (2+ years out) to better understand how we can more effectively prepare students for long term success.
- Learn.co Data: By analyzing student data provided on the backend of the Learn.co system, we can observe where students struggle with material both within, and across cohorts. In doing so, we can identify areas of improvement along with root causes (i.e., if students across cohorts struggle with an assignment it may be a curriculum issue, whereas if students in one cohort struggle, it is likely an instructional concern). In doing so, we can intervene quickly and measure the impact of those changes.
- Ongoing surveys: For students on-campus and online to glean immediate feedback on the educational experience itself, as well as the instructors and staff.

Externally, our Software Engineering curriculum has been reviewed by the following parties:

- State University of New York: The curriculum was reviewed in submission of a joint application to the U.S. Department of Education, in consideration for the EQUIP program. As part of the review, the leaders of the Computer Science department at Empire State College reviewed our course material and instructional processes, both remotely and during on-site visits to Flatiron School's campus. The curriculum has been approved by SUNY's accreditors and board of trustees to qualify for 12 credits of Applied Learning.

- NYC Tech Talent Pipeline (TTP): Our program was reviewed by the TTP advisory board in preparation for our partnerships with the City of New York. The Advisory Board includes representatives from companies like Spotify, Microsoft, Goldman Sachs, Etsy, Facebook, etc.

Housing

Flatiron School does not have dormitory facilities under its control. Flatiron School has no responsibility to find or assist a student in finding housing.

Students can choose between many housing options at a number of apartment complexes located within a reasonable distance of the San Francisco campus. Rent for studio and 1-bedroom apartments ranges from $2,461 to $3,261 per month based on single occupancy. For students who prefer to live with roommates, individual rent ranges from $1,230.50 to $1,630.50 per month, for nearby 2-bedroom apartments, based on 2 person occupancy. Additional charges for electricity, trash, water, and internet access may apply.

LEADERSHIP AND FACULTY INFORMATION

Flatiron School Leadership

The leadership team of Flatiron School manages the day-to-day operations with input from students as well as the greater technical community.

Adam Enbar, Chief Executive Officer
Peter Barth, GM and Dir. of Campus Launches; Interim San Francisco Campus Director
Avi Flombaum, VP of Product
Kristi Riordan, Chief Operating Officer
James Leslie, Chief Administrative Officer
Joe Burgess, Senior Director of Education
Rebekah Rombom, Senior Director of Student Success
Annette Doskow, Senior Director of Admissions
Gretchen Jacobi, Senior Director of Career Services
Nicole Kroese, Director of Marketing

Flatiron School's Instructors

Flatiron School's instructors are highly qualified to teach in their respective programs and possess significant experience, education and/or training in their respective subject areas. Because of the cutting-edge and constantly evolving nature of Flatiron's programs, there is no single academic degree or qualification that provides the skills and competencies required. All of the senior faculty who develop Flatiron School's curriculum have significant experience both teaching and working as practitioners. Flatiron hires faculty who can demonstrate a mastery of the subject they will be teaching, and an ability to relate complex concepts to students.

The following is a system-wide list of Flatiron School's instructors for its online programs.

- Jacob Eli Thomas
- Dakota Martinez
- Jennifer Hansen
- Amelie Oller
- Erika Hughes
- Z Drake
- Nancy Noyes
- Ian Candy
- Luisa Scavo
- Dustin Anderson
- Jeff Herman
- Enoch Griffith
- Alice Balbuena
- Howard DeVennish
- Matt Cassara
- Rafael Carrasco

Statement of Legal Control

Flatiron School is owned by Flatiron School LLC, a New York limited liability company. In October 2017, Flatiron School LLC was acquired by WeWork Companies, Inc., a Delaware corporation. There are five members of the WeWork Companies, Inc. Board of Directors:

Adam Neumann
Bruce Dunlevie
Ronald D. Fisher
Lew Frankfurt
M. Steven Landman
Mark Schwartz

REGULATORY NOTICES

Any questions a student may have regarding this catalog that have not been satisfactorily answered by the institution may be directed to the Bureau for Private Postsecondary Education at 2535 Capitol Oaks Drive, Suite 400, Sacramento, CA 95833, P.O. Box

980818, West Sacramento, CA 95798-0818, www.bppe.ca.gov, (888) 370-7589 or (916) 431-6959, or by fax (916) 263-1897.

A student or any member of the public may file a complaint about this institution with the Bureau for Private Postsecondary Education by calling (888) 370-7589 or by completing a complaint form, which can be obtained on the bureau's internet Web site www.bppe.ca.gov.

Flatiron School is a private institution that is approved to operate by the Bureau, and is in compliance with state standards as set forth in the California Private Postsecondary Education Act of 2009 and its applicable regulations.

Flatiron School does not have a pending petition in bankruptcy, is not operating as a debtor in possession, has not filed a petition within the preceding five years, and has not had a petition in bankruptcy filed against it within the preceding five years that resulted in reorganization under Chapter 11 of the United States Bankruptcy Code (11 U.S.C. Sec. 1101 et seq.)

Student Tuition Recovery Fund (STRF)

The State of California established the Student Tuition Recovery Fund (STRF) to relieve or mitigate economic loss suffered by a student in an educational program at a qualifying institution, who is or was a California resident while enrolled, or was enrolled in a residency program, if the student enrolled in the institution, prepaid tuition, and suffered an economic loss. Unless relieved of the obligation to do so, you must pay the state-imposed assessment for the STRF, or it must be paid on your behalf, if you are a student in an educational program, who is a California resident, or are enrolled in a residency program, and prepay all or part of your tuition.

It is a California state requirement that a student who pays his or her tuition is required to pay a state-imposed assessment for the Student Tuition Recovery Fund. You are not eligible for protection from the STRF and you are not required to pay the STRF assessment, if you are not a California resident or are not enrolled in a residency program.

It is important that you keep copies of your enrollment agreement, financial aid documents, receipts, or any other information that documents the amount paid to the school. Questions regarding the STRF may be directed to the Bureau for Private Postsecondary Education, 2535 Capitol Oaks Drive, Suite 400, Sacramento, CA 95833, (916) 431-6959 or (888) 370-7589.

To be eligible for STRF, you must be a California resident or are enrolled in a residency program, prepaid tuition, paid or deemed to have paid the STRF assessment, and suffered an economic loss as a result of any of the following:

1. The institution, a location of the institution, or an educational program offered by the institution was closed or discontinued, and you did not choose to participate in

a teach-out plan approved by the Bureau or did not complete a chosen teach-out plan approved by the Bureau.

2. You were enrolled at an institution or a location of the institution within the 120-day period before the closure of the institution or location of the institution, or were enrolled in an educational program within the 120-day period before the program was discontinued.
3. You were enrolled at an institution or a location of the institution more than 120 days before the closure of the institution or location of the institution, in an educational program offered by the institution as to which the Bureau determined there was a significant decline in the quality or value of the program more than 120 days before closure.
4. The institution has been ordered to pay a refund by the Bureau but has failed to do so.
5. The institution has failed to pay or reimburse loan proceeds under a federal student loan program as required by law, or has failed to pay or reimburse proceeds received by the institution in excess of tuition and other costs.
6. You have been awarded restitution, a refund, or other monetary award by an arbitrator or court, based on a violation of this chapter by an institution or representative of an institution, but have been unable to collect the award from the institution.
7. You sought legal counsel that resulted in the cancellation of one or more of your student loans and have an invoice for services rendered and evidence of the cancellation of the student loan or loans.

To qualify for STRF reimbursement, the application must be received within four (4) years from the date of the action or event that made the student eligible for recovery from STRF.

A student whose loan is revived by a loan holder or debt collector after a period of non-collection may, at any time, file a written application for recovery from STRF for the debt that would have otherwise been eligible for recovery. If it has been more than four (4) years since the action or event that made the student eligible, the student must have filed a written application for recovery within the original four (4) year period, unless the period has been extended by another act of law.

However, no claim can be paid to any student without a social security number or a taxpayer identification number.