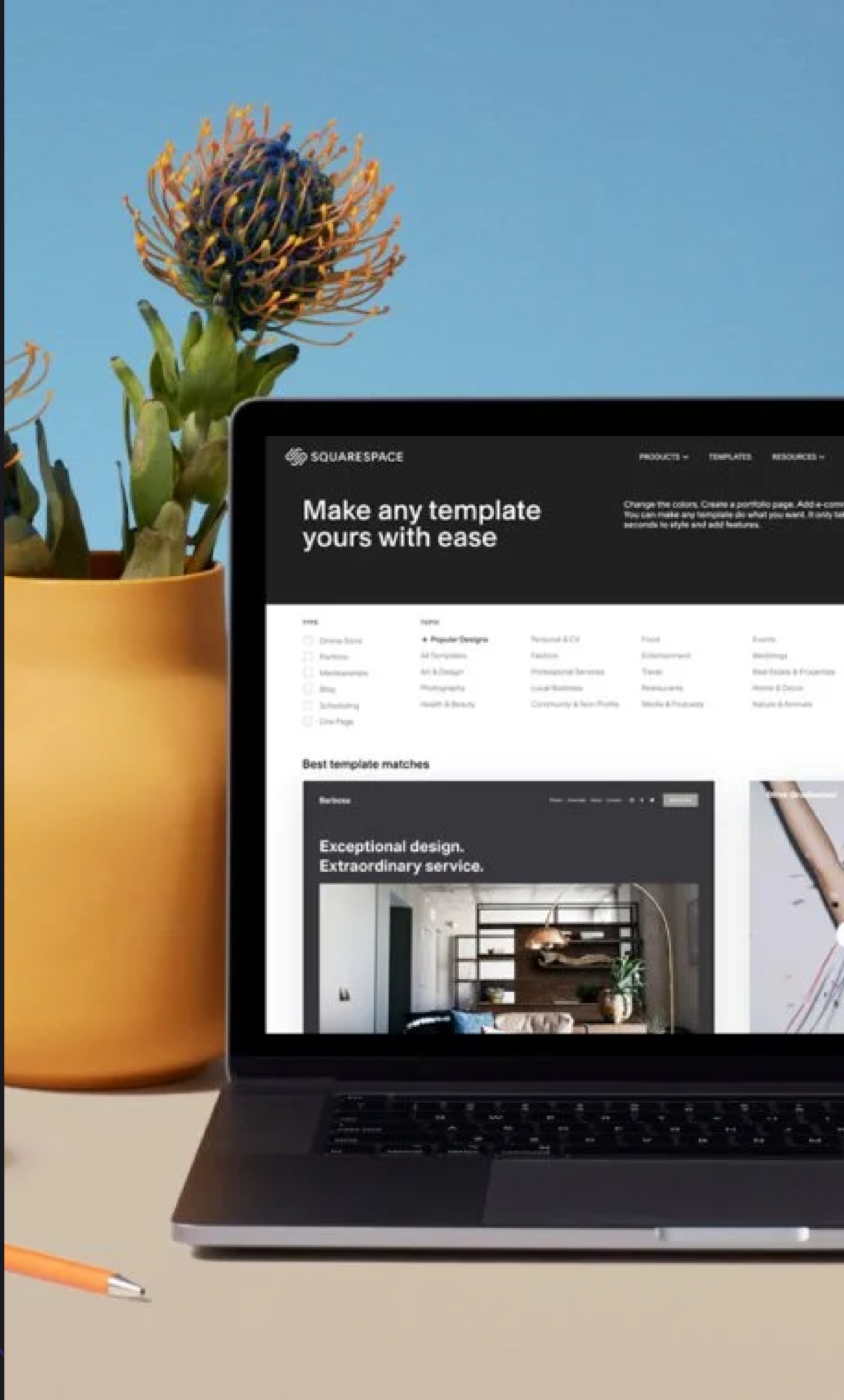
Cockroach Labs

CASE STUDY

Why (and how) Squarespace migrated to CockroachDB to achieve global scale

SQUARESPACE


SQUARESPACE
Make any template yours with ease
Best template matches
Exceptional design. Extraordinary service.

SQUARESPACE

Why (and how) Squarespace migrated to CockroachDB to achieve global scale

INDUSTRY:
Software

CHALLENGES:
A performant platform built for global scale with always-on availability.

SOLUTION:
Migrating from PostgreSQL to CockroachDB for a future-ready foundation.

About Squarespace

Founded in 2003, Squarespace is a design-driven platform that helps entrepreneurs build brands and businesses online. The company's product portfolio helps their users go from idea to a fully fledged business, inspiring anyone to become an entrepreneur.

Millions of people around the globe use Squarespace to build websites and run successful businesses. Their users span many different industries and verticals from brick-and-mortar stores such as restaurants, barber shops, and yoga studios, to service providers like photographers, coaches, and consultants. Their robust suite of products includes websites, domains, marketing tools, and ecommerce, as well as tools for AI visibility, scheduling appointments, and creating and managing social media presence.

Squarespace continues to focus on platform reliability and improving performance for their end users around the world. In order to cater to a global customer base, the engineering team in charge of stateful systems decided to modernize their infrastructure and migrated from PostgreSQL to CockroachDB.

Watch YouTube: Powering Global Scale at Squarespace with CockroachDB ▸

0
DOWNTIME MIGRATION

100
CUTOVERS

20
CLUSTERS PER REGION

PostgreSQL scaling limitations

Squarespace historically used PostgreSQL as their primary relational database. As the company expanded their presence globally, the team realized it was going to need a solution that could scale across regions, handle sudden growth (i.e. workload spikes), and manage giant tables just via vertical scaling.

They were running mostly on-premise in their own private data centers, requiring them to think critically about how to manage disk space, power consumption, and even planning for natural disasters (hurricanes, freezes, power outages). The team knew that moving to the cloud would help alleviate some of these issues, but they were still worried about how much I/O a single machine could handle.

The team was spending a significant amount of time managing PostgreSQL. A lot of auxiliary systems were required to make PostgreSQL work for their use cases and they needed to handle replication, backups, monitoring, and interconnectivity themselves. Upgrading was a daunting process for them, but staying on the same version wasn't feasible.

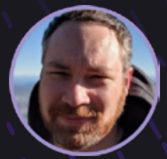
Gavin McQuillan, a Senior Staff Software Engineer, works on the team that is in charge of the stateful systems at Squarespace, especially for the customer-facing processes and services. Along with his team, they wanted to develop a scalable system that was easy to manage, and would allow them to achieve better global reach and provide better responsiveness to users.

This meant switching to a new database that could deliver:

- Built-in horizontal scale
- PostgreSQL compatibility
- Online operations (upgrades, schema changes, etc.)
- High availability with desirable RTO
- Multi-region deployment

Gavin heard about CockroachDB when it was first introduced (and was familiar with the Spanner whitepaper), and thought it was an impressive base to build a relational data system on top of. He followed the distributed database's progress over the past 10+ years and saw that it was maturing really well and had a great team to support it.

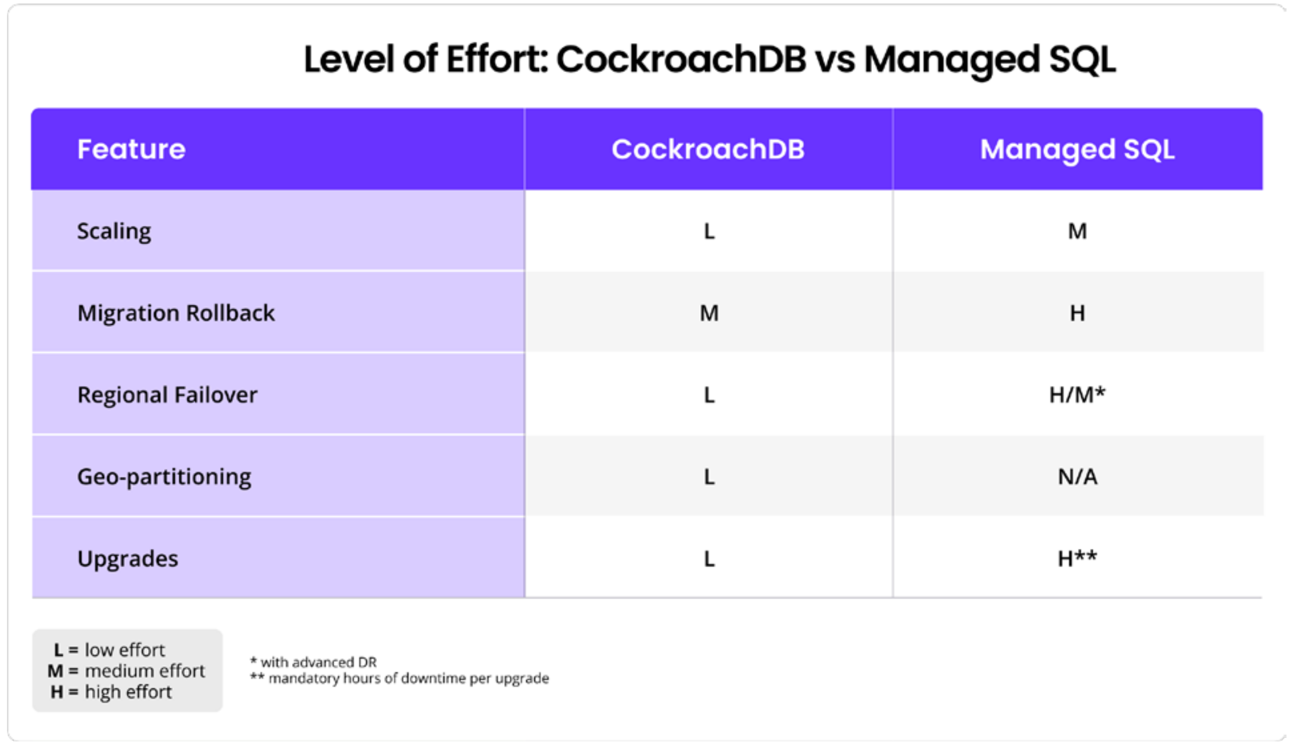
"You have to ask yourself things like how much downtime is acceptable? Will this solution allow me to reach my customers? Strategically, for us, we no longer thought that PostgreSQL was the right fit for global workloads and we needed a database like CockroachDB that could deliver distributed, horizontal scale."



Gavin McQuillan
Sr. Staff Software Engineer, Squarespace

Evaluating solutions

The team decided to move forward with testing CockroachDB, and they also did a side-by-side comparison of a managed version of PostgreSQL. However, some of the challenges they were trying to overcome (i.e. no downtime for maintenance) still wouldn't be possible. Plus, the level of effort it would take to make managed PostgreSQL work for their use case was significantly higher than what it would be with CockroachDB as illustrated below:



Level of Effort: CockroachDB vs Managed SQL

Feature	CockroachDB	Managed SQL
Scaling	L	M
Migration Rollback	M	H
Regional Failover	L	H/M*
Geo-partitioning	L	N/A
Upgrades	L	H**

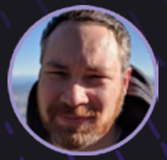
L = low effort
M = medium effort
H = high effort

* with advanced DR
** mandatory hours of downtime per upgrade

Overall, the Squarespace team was impressed with the maturity and durability of CockroachDB, commenting that it's "a database that definitely records your records and doesn't lie to you about it." The Squarespace team also liked that CockroachDB presents itself as a PostgreSQL cluster, but with a lot of topological configurations, so you can keep scaling separate from application logic.

When it came to global performance, they looked closely at what exactly multi-regionality means in CockroachDB, how reads and writes work, and how the system handles the data distribution. Gavin mentioned that one of the critical differences between CockroachDB and many other systems is that CockroachDB has a metadata layer that is broken up into ranges and can be distributed into multiple regions which would be very beneficial to many of their use cases.

"There's a fluidity to CockroachDB that lets you respond dynamically without having to make changes to your application. This is really powerful to us because the teams managing the application are different from those managing the data systems. They have different control points and we don't want to disrupt their workflow."

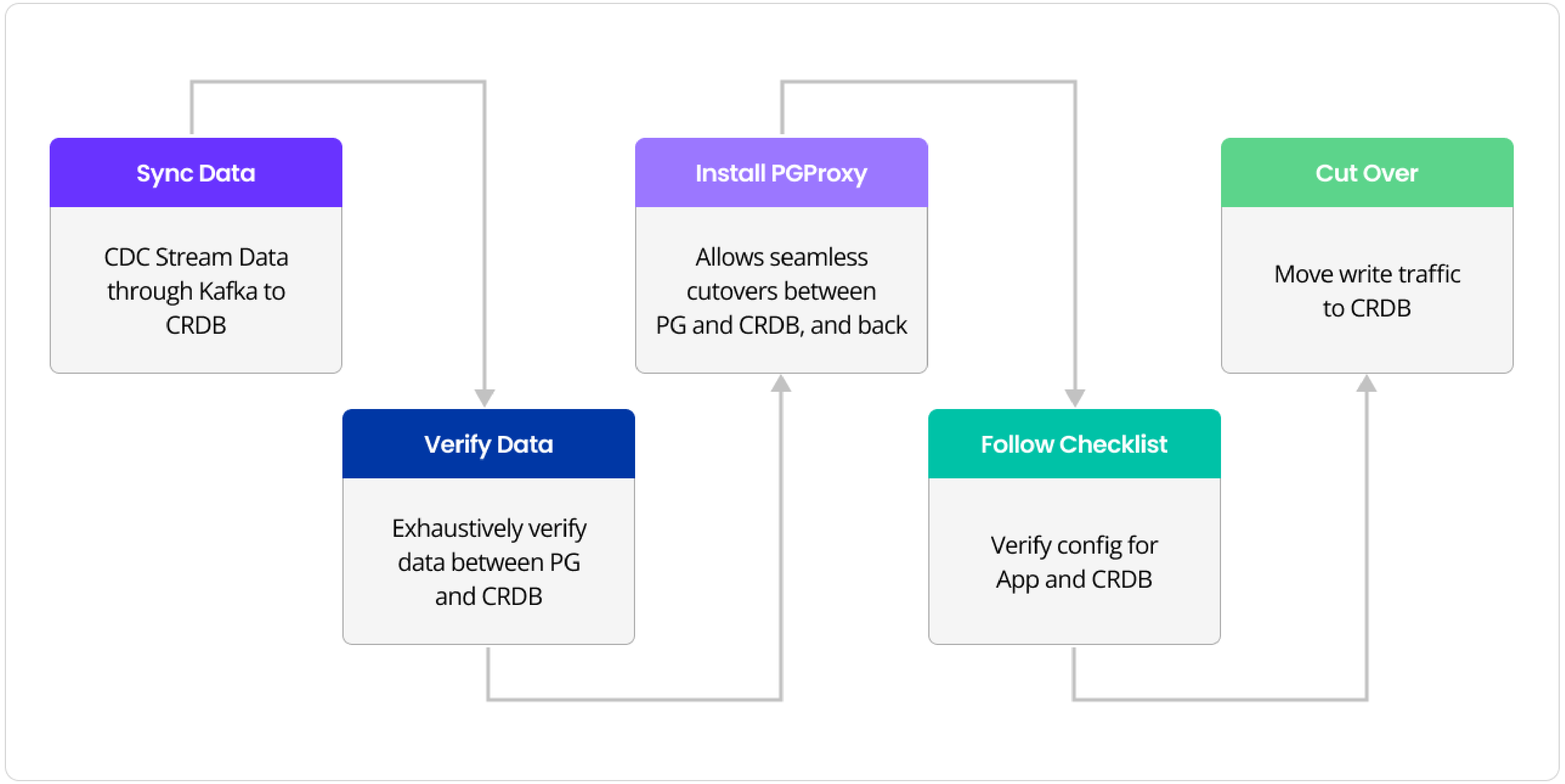


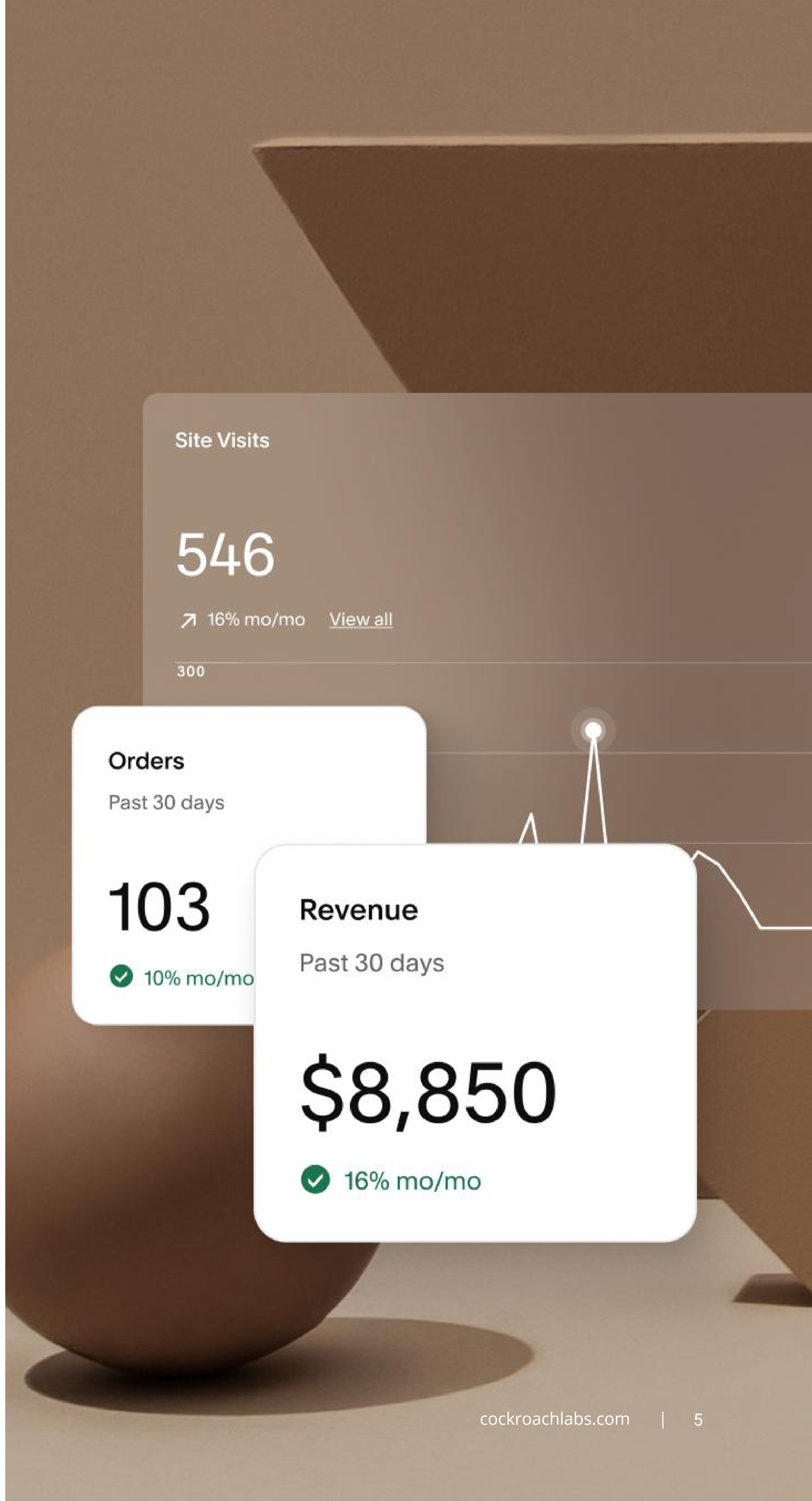
Gavin McQuillan
Sr. Staff Software Engineer, Squarespace

Executing a migration

After load testing CockroachDB, they felt confident in moving forward with the migration. Most of Squarespace's services are microservices written in Java and most were backed by PostgreSQL. It was very important to the team to avoid extensive downtime while migrating services from PostgreSQL to CockroachDB.

The Squarespace infrastructure team worked with Cockroach Labs to ensure they had a solid plan in place before executing the migration. They were focusing on staging and production environments which are the most sensitive to downtime. In total, they migrated 50 clusters across two environments which resulted in around 100 cutovers. Here's a visual example of how it worked:


Sync Data
CDC Stream Data through Kafka to CRDB
Verify Data
Exhaustively verify data between PG and CRDB
Install PGProxy
Allows seamless cutovers between PG and CRDB, and back
Follow Checklist
Verify config for App and CRDB
Cut Over
Move write traffic to CRDB



Phase 1: Sync data
They began with a schema dump, and then via Change Data Capture (using Debezium) they streamed the data out of PostgreSQL. They started with a snapshot, and continued with a real-time replication stream into Kafka. They also configured custom queue processors that take the Debezium data stream from Kafka, and write it into CockroachDB, with heavy batching and retries. For speed, they paused the foreign key constraints, but turned them back on before moving traffic over.

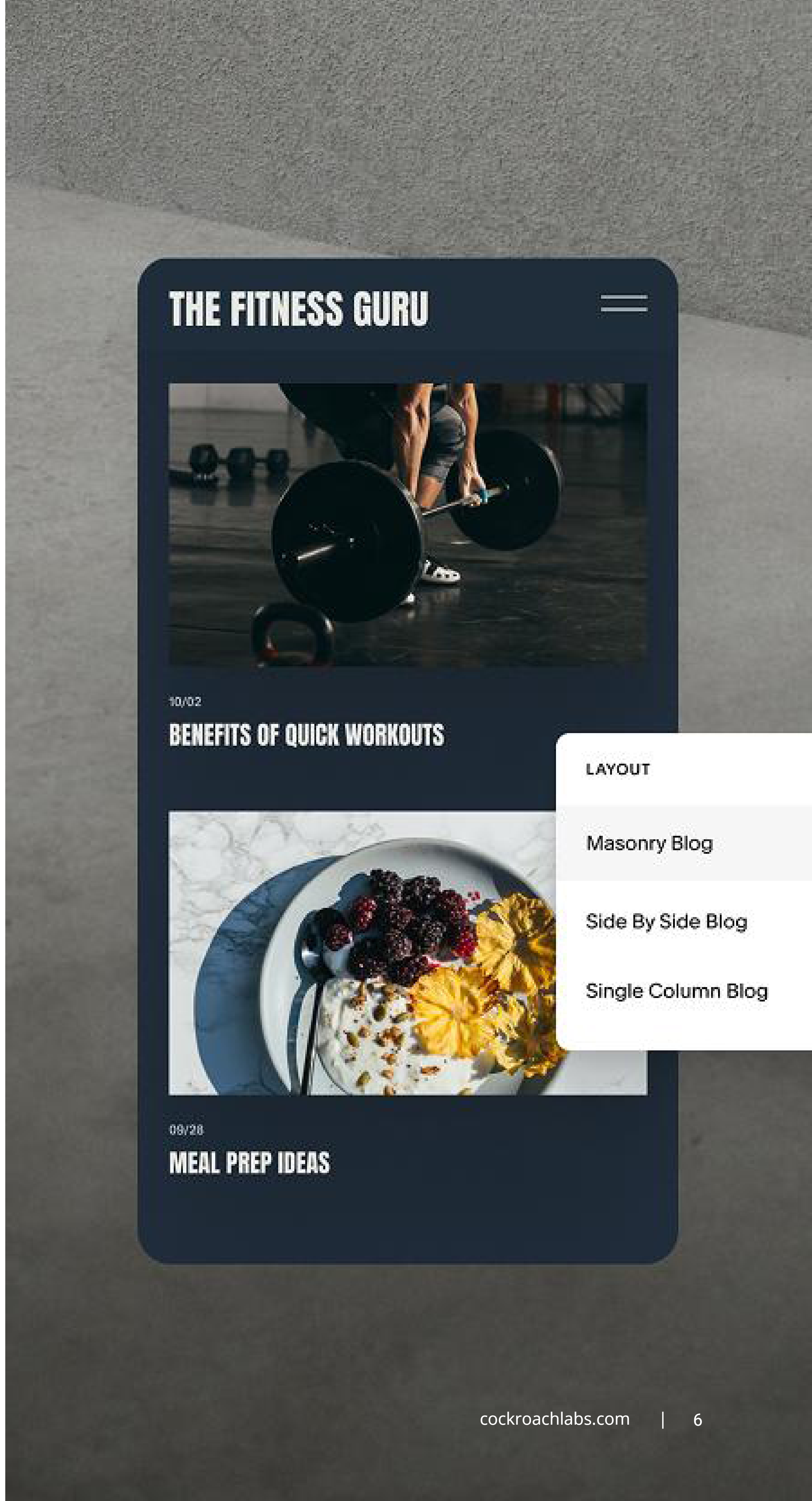
Phase 2: Verify data
Once the CDC streams caught up, they started running data validation to ensure that the data in PostgreSQL and the data in CockroachDB, were exactly the same.

Phase 3: Install PGProxy
Next they moved all the clients for a given service to using the managed proxy. If the HAProxy lives in Kubernetes near applications, it can pivot between sending traffic for a given endpoint to either PostgreSQL or CockroachDB using automation. This helped with achieving their goal of having as atomic of a cutover as possible.

Phase 4: Follow checklist
During this phase the team validated that the data sync is good, and corresponds 1:1 with PostgreSQL continually. They also made sure the applications worked with the new database which meant testing queries, setting up load tests, etc. While a majority (80%) of the applications didn't require any app-level changes, some needed to be refactored to work with a distributed system.

Phase 5: Cut over
Finally they performed the cutover and observed. Usually within a few minutes they had a good sense of how well the system is performing. However, they observe for a few hours before making a call about whether to fall back to PostgreSQL and try again after making adjustments. Once the cutover happens, they set up jobs that read from CockroachDB's changefeed to stream any mutations from CockroachDB into a separate set of topics. They turn this on just after migrating writes to CockroachDB to prevent write cycles. They use this data in case they decide to cut back to PostgreSQL, so they can preserve any writes written to CockroachDB in the meantime.

The process outlined above is for a single cutover which they performed twice for each application that was migrated. For other "lower" environments, they use CockroachDB's MOLT tooling to help assist with a faster migration process offline.



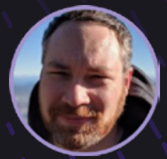
Distributed use cases & multi-region deployment

Following the migration, CockroachDB serves as the backend for many core applications within Squarespace. For example, it supports their CRM product which has multiple workloads such as user accounts, commerce data, and event data. It's also used as the foundational infrastructure for use cases that handle billing or certificate termination.

Across the organization, Squarespace's workloads take different shapes. A lot are very read-heavy, with a few exceptions that are write-only. Most of their OLTP reads are simple and contain one or fewer JOINs, however some systems have hundreds of tables and dozens of JOINs. It's a huge benefit that CockroachDB is still able to process both reads and writes even if an entire region goes down.

Squarespace currently runs across three regions and has around 20 clusters per environment. They are running a hybrid deployment with two GCP regions and one data center. In the long term, they plan to fully migrate to GCP and will add additional regions if demand requires it (i.e. keeping latency low for distributed users).

"Our data management philosophy shifted from managing a bunch of data silos, to thinking about managing data as a series of streams into and out of a system of record. CockroachDB is an ideal hub for bringing consistency to the data itself, and to the format for service-to-service communication with asynchronous data sharing with changefeeds. We've been able to shift our mentality thanks to CockroachDB's feature set."



Gavin McQuillan

Sr. Staff Software Engineer, Squarespace

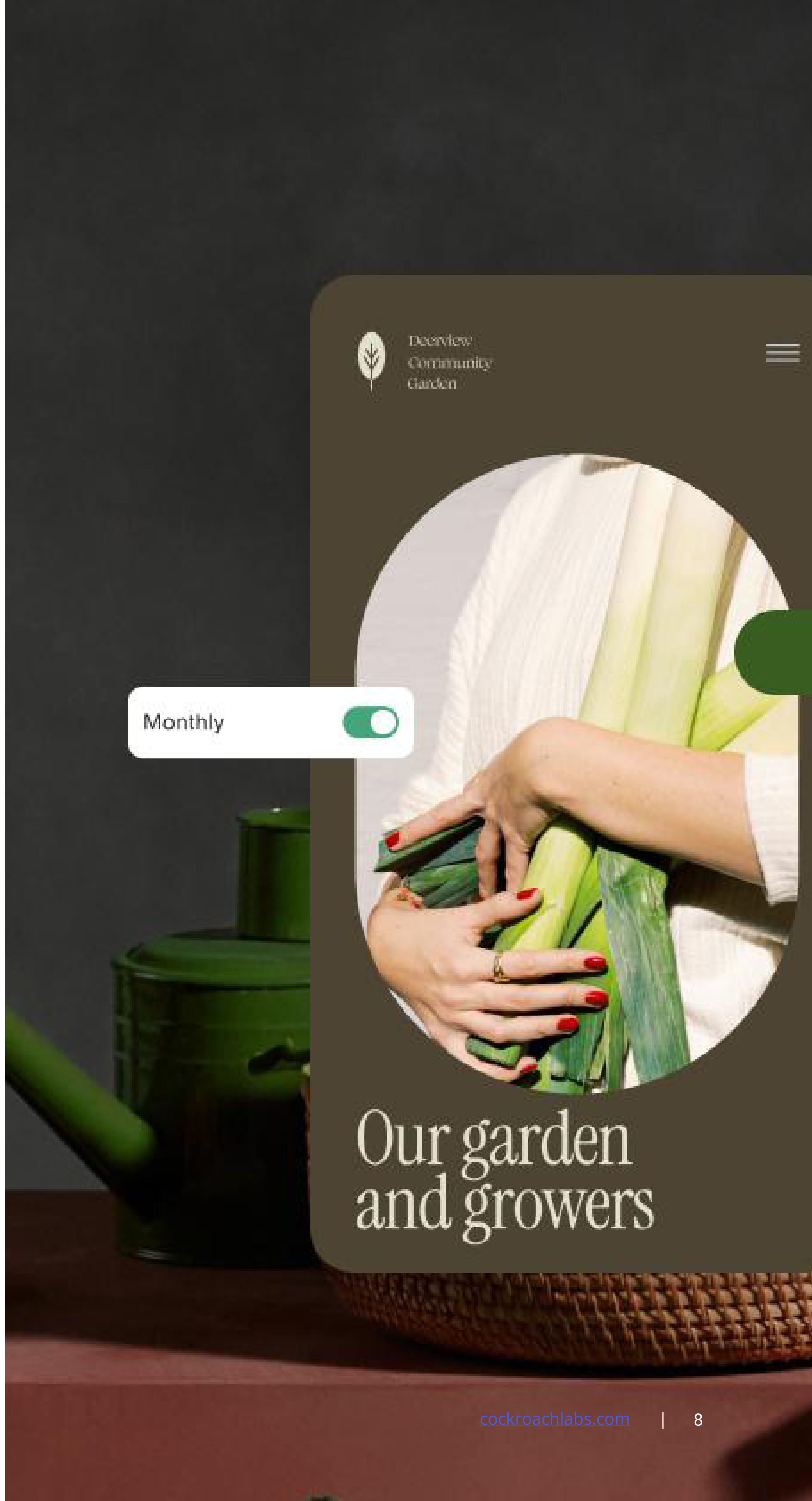
Advice for distributed SQL newcomers

Gavin says that a big challenge they had to overcome was buy-in across different teams.He recommends contextualizing how expensive downtime can be and how a migration to CockroachDB can relieve a ton of technical debt.

He also suggests providing teams with tools to help simulate workload testing before completing the migration. This is for two reasons 1) some applications weren't designed to be distributed systems and required cross team collaboration and 2) oftentimes if there were performance differences, they were able to make acceptable trade offs prior to migrating.

Gavin's been impressed with all the developers who have taken advantage of CockroachDB's tooling right away. For example, they often leverage the DB Console insights to see what's happening with the clusters. He says that the visibility into cluster health, and the specific health of different code paths in their applications, have been "a game changer."

Now that the migration is complete, they've been pleased with CockroachDB's straightforward operations patterns including online upgrades and online schema changes. They complete upgrades quarterly for all clusters without any downtime which helps improve their security posture. There's no more compatibility matrices to follow, and no more mandatory auxiliary tooling for PostgreSQL.



What’s next

Squarespace’s platform backed by CockroachDB has the ability to serve read and write traffic from anywhere in the world. If there is an outage or natural disaster, the system can quickly and automatically pivot without losing a single write.

Now that Squarespace has a really stable base to launch off of, Gavin says they are going to be focused on a lot more self-service workflows to help improve developer efficiency and ramp up company velocity. This includes figuring out how to manage schema changes at scale a little better, improve user management, and deal with re-orgs more gracefully.

Overall, Squarespace is on track to exceed business goals of global reach, horizontal scale, improved recovery time, and reduced complexity with CockroachDB. To learn more about Squarespace’s products and resources, visit their website.

Ready to get started?

Go hands-on with 100% free CockroachDB Serverless. Spin up your first cluster in just a few clicks.

Get Started →

Compare Products →