BookMyShow Scalable Platform – Project Report

Objective:

To simulate a BookMyShow-like ticketing system and test performance under high traffic, identifying bottlenecks, optimizing architecture, and ensuring scalability, resilience, and cost efficiency.

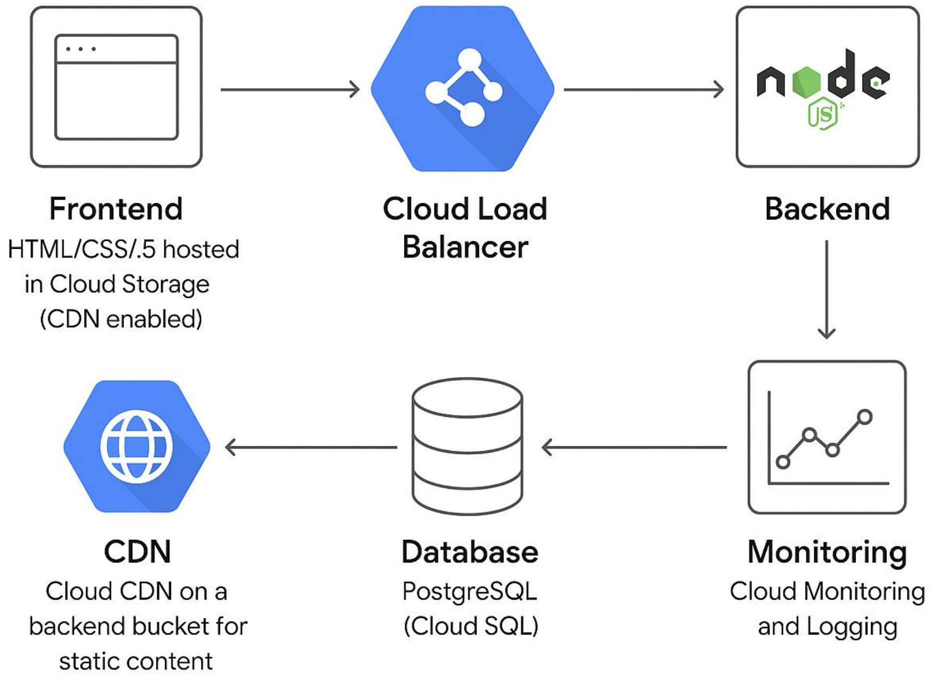
Scope:

- Create a prototype with frontend and backend
- Deploy backend on Cloud Run
- Use Cloud SQL for database
- Perform load testing with k6
- Implement caching and CDN
- Monitor performance and analyze costs

System Architecture

Components:

- Frontend: HTML/CSS/JS hosted in Cloud Storage (CDN enabled)
- Backend: Node.js + Express, deployed on Cloud Run
- Database: PostgreSQL (Cloud SQL)
- CDN: Cloud CDN on a backend bucket for static content
- Cloud Load Balancing: Handled by Cloud Run autoscaling
- Monitoring: Cloud Monitoring and Logging



Deployment & Code

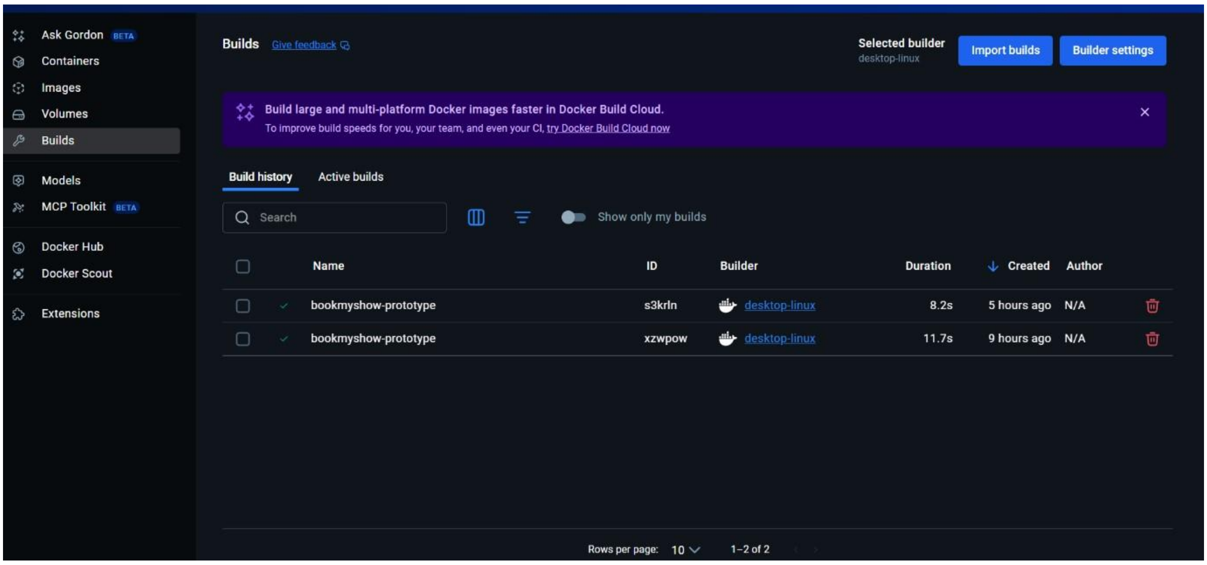
Backend (server.js):

Frontend: HTML, CSS, JS hosted on Cloud Storage + served via Cloud CDN.

```
const express = require("express");
const path = require("path");
const { Pool } = require("pg");
const app = express();
const PORT = process.env.PORT || 3000;

// Serve static files
app.use(express.static(path.join(__dirname, "public")));

// Database configuration
const pool = new Pool({
  user: process.env.DB_USER,
  host: process.env.DB_HOST,
  database: process.env.DB_NAME,
  password: process.env.DB_PASS,
  port: 5432,
});

// Test DB connection
pool.connect((err, client, release) => {
  if (err) console.error("Error connecting to DB:", err.stack);
  else console.log("Connected to PostgreSQL DB");
  release();
});

// Example API route
app.get("/api/movies", async (req, res) => {
  try {
    const result = await pool.query("SELECT * FROM movies");
    res.json(result.rows);
  } catch (err) {
    console.error(err);
    res.status(500).json({ error: "Database error" });
  }
});

// Fallback for index.html
app.get("/", (req, res) => {
  res.sendFile(path.join(__dirname, "public", "index.html"));
});

app.listen(PORT, () => console.log(`Server running on port ${PORT}`));
```

Docker:

- Image built locally → pushed to Google Container Registry
- Deployed to Cloud Run with Cloud SQL integration



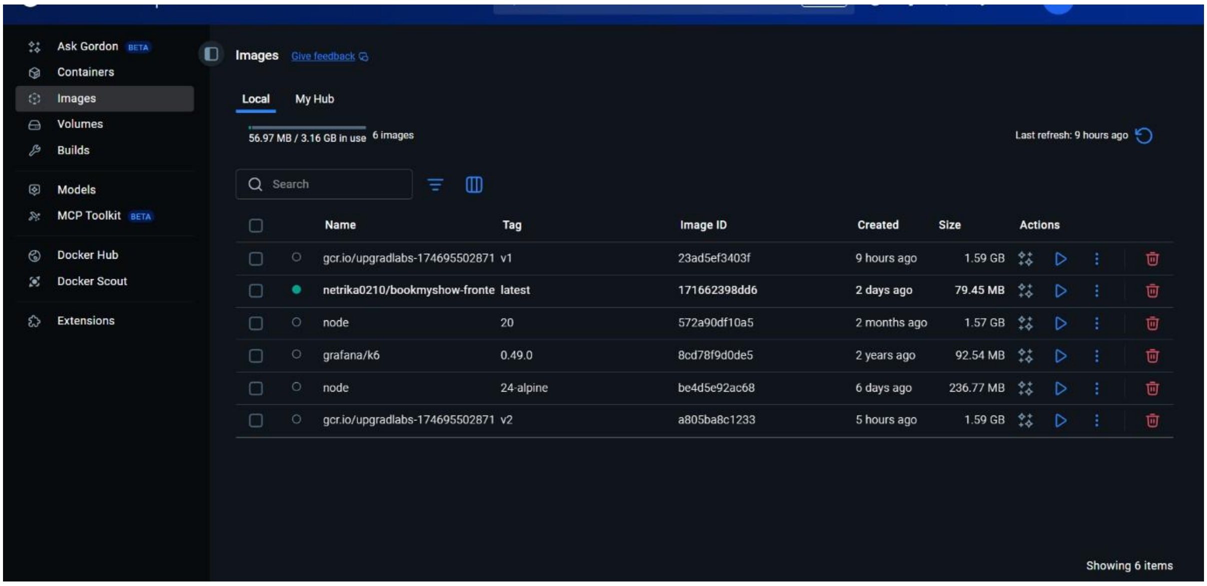
Builds

Selected builder
desktop-linux

Build large and multi-platform Docker images faster in Docker Build Cloud.
To improve build speeds for you, your team, and even your CI, try Docker Build Cloud now

☐		Name	ID	Builder	Duration	Created	Author
☐	✓	bookmyshow-prototype	s3krln	desktop-linux	8.2s	5 hours ago	N/A
☐	✓	bookmyshow-prototype	xzwpow	desktop-linux	11.7s	9 hours ago	N/A

Rows per page: 10 1–2 of 2



Images

56.97 MB / 3.16 GB in use 6 images

Last refresh: 9 hours ago

☐		Name	Tag	Image ID	Created	Size	Actions
☐	○	gcr.io/upgradlabs-174695502871	v1	23ad5ef3403f	9 hours ago	1.59 GB	
☐	●	netrika0210/bookmyshow-fronte	latest	171662398dd6	2 days ago	79.45 MB	
☐	○	node	20	572a90df10a5	2 months ago	1.57 GB	
☐	○	grafana/k6	0.49.0	8cd78f9d0de5	2 years ago	92.54 MB	
☐	○	node	24-alpine	be4d5e92ac68	6 days ago	236.77 MB	
☐	○	gcr.io/upgradlabs-174695502871	v2	a805ba8c1233	5 hours ago	1.59 GB	

Showing 6 items

Traffic Simulation

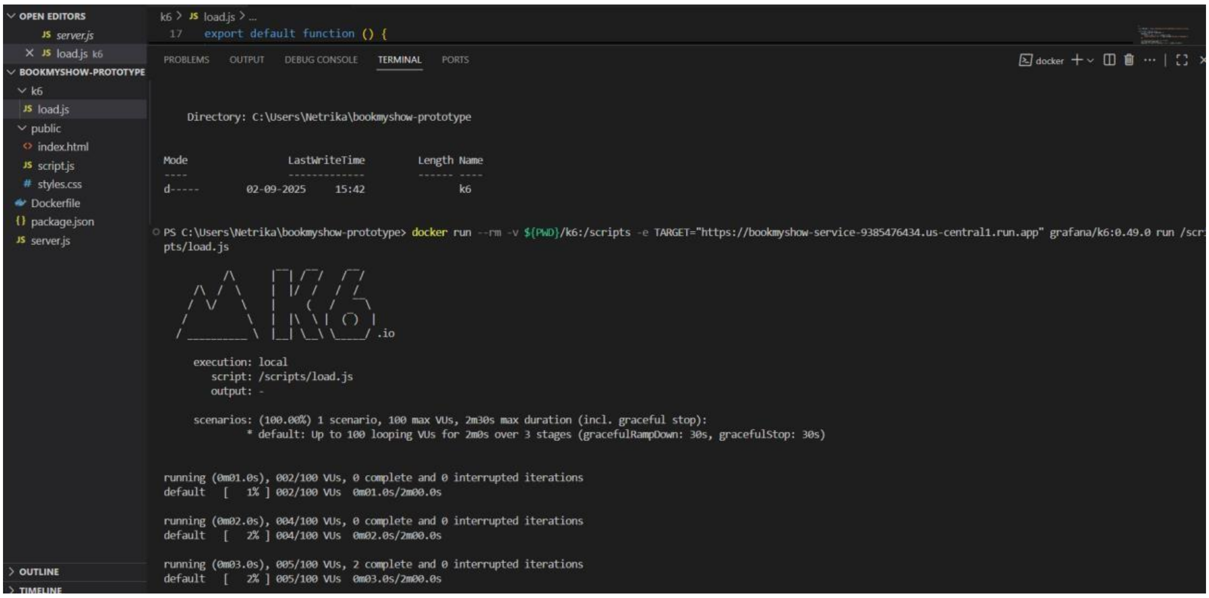
Tool: k6 load testing

Metrics for v2 Deployment:

Metrics	Value
Avg Response Time	766ms
Max Response Time	2.37s
Iterations	1748
VUs	50

Observations:

- Cold starts on Cloud Run may cause initial delays
- Database response time ~765ms, potential for query optimization



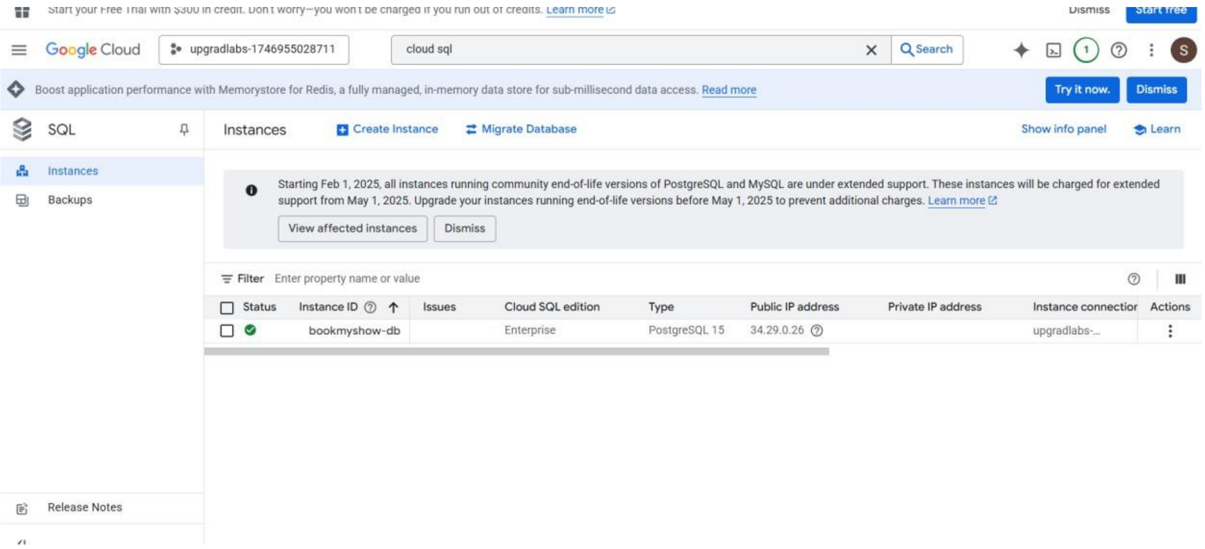
```
export default function () {
```

```
    Directory: C:\Users\Netrika\bookmyshow-prototype


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        02-09-2025     15:42                k6


PS C:\Users\Netrika\bookmyshow-prototype> docker run --rm -v ${PWD}/k6:/scripts -e TARGET="https://bookmyshow-service-9385476434.us-central1.run.app" grafana/k6:0.49.0 run /scr
pts/load.js

          /\      |‾‾| /‾‾/   /‾‾/
     /\  /  \     |  |/  /   /  /
    /  \/    \    |     (   /   ‾‾\
   /          \   |  |\  \ |  (‾)  |
  / __________ \  |__| \__\ \_____/ .io

     execution: local
        script: /scripts/load.js
        output: -

     scenarios: (100.00%) 1 scenario, 100 max VUs, 2m30s max duration (incl. graceful stop):
              * default: Up to 100 looping VUs for 2m0s over 3 stages (gracefulRampDown: 30s, gracefulStop: 30s)


running (0m01.0s), 002/100 VUs, 0 complete and 0 interrupted iterations
default   [   1% ] 002/100 VUs  0m01.0s/2m00.0s

running (0m02.0s), 004/100 VUs, 0 complete and 0 interrupted iterations
default   [   2% ] 004/100 VUs  0m02.0s/2m00.0s

running (0m03.0s), 005/100 VUs, 2 complete and 0 interrupted iterations
default   [   2% ] 005/100 VUs  0m03.0s/2m00.0s
```

Database Optimisation

Actions Taken / Recommendations:

- Cloud SQL instance with PostgreSQL 15 • Database connection pooling via pg module
- **Future enhancements:**
  - Horizontal read replicas for high availability
  - Backups and point-in-time recovery ○ Query indexing for faster reads

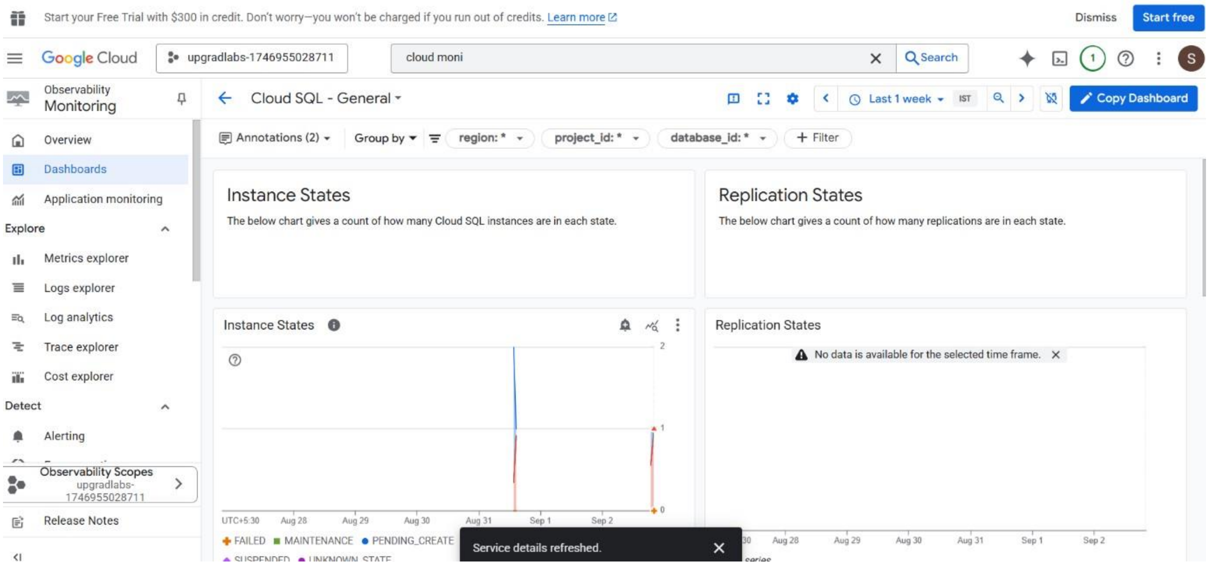


Start your Free Trial with $300 in credit. Don't worry—you won't be charged if you run out of credits. Learn more

Boost application performance with Memorystore for Redis, a fully managed, in-memory data store for sub-millisecond data access. Read more

Starting Feb 1, 2025, all instances running community end-of-life versions of PostgreSQL and MySQL are under extended support. These instances will be charged for extended support from May 1, 2025. Upgrade your instances running end-of-life versions before May 1, 2025 to prevent additional charges. Learn more

□	Status	Instance ID ↑	Issues	Cloud SQL edition	Type	Public IP address	Private IP address	Instance connectior	Actions
□	✓	bookmyshow-db		Enterprise	PostgreSQL 15	34.29.0.26		upgradlabs-...	⋮



Instance States

The below chart gives a count of how many Cloud SQL instances are in each state.

Replication States

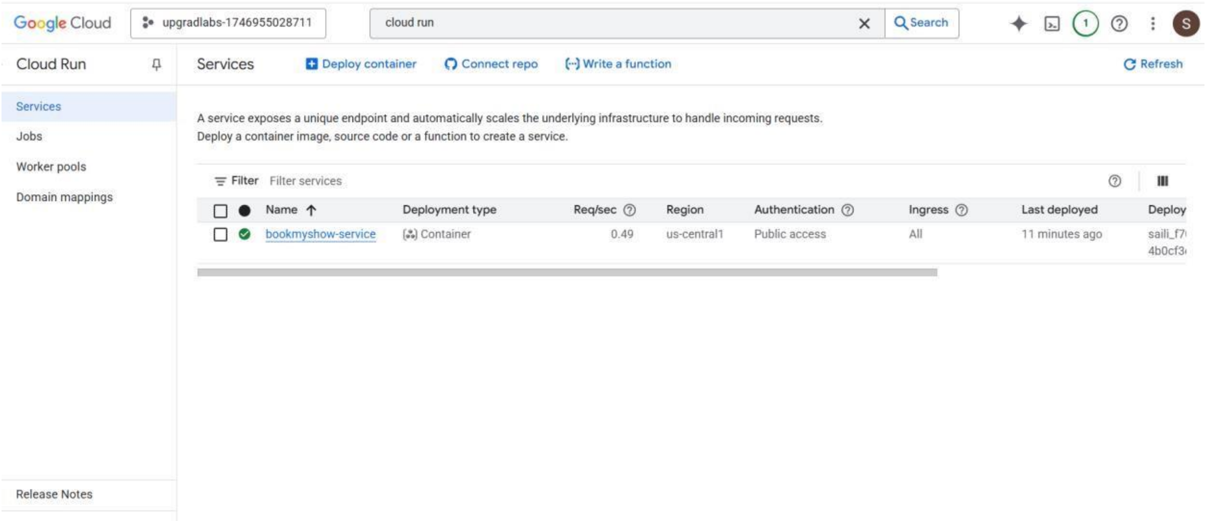
The below chart gives a count of how many replications are in each state.

No data is available for the selected time frame.

Service details refreshed.

Application Resilience

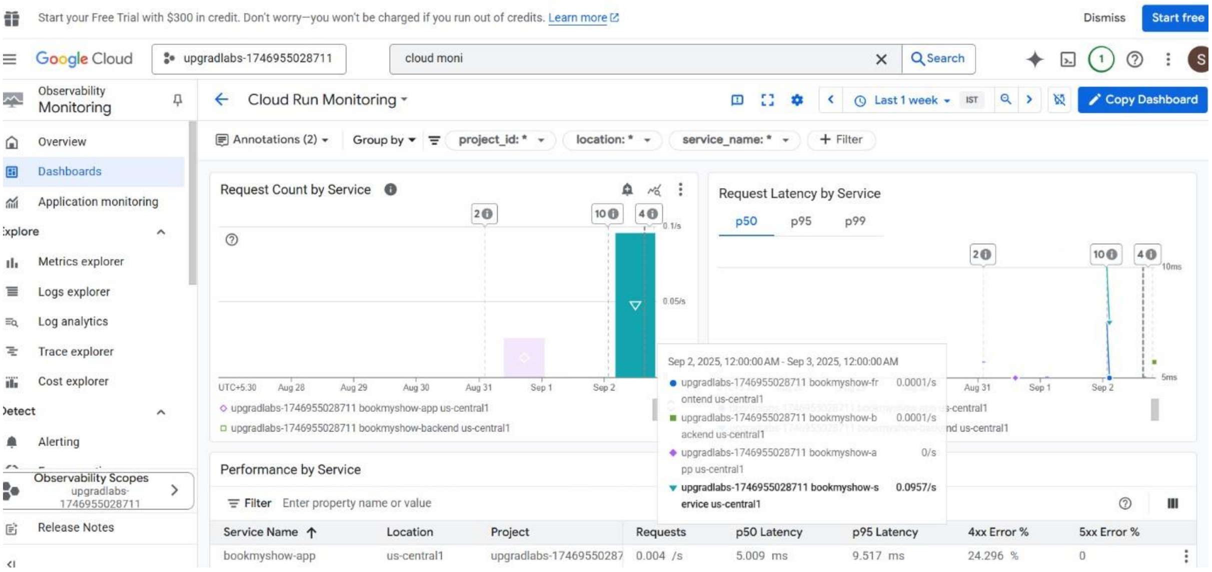
- Static content cached in Cloud Storage + Cloud CDN • Autoscaling enabled on Cloud Run
- Future enhancements:
  - Rate limiting
  - Circuit breakers for backend APIs ○ Multi-region deployment



A service exposes a unique endpoint and automatically scales the underlying infrastructure to handle incoming requests.
Deploy a container image, source code or a function to create a service.

Filter Filter services

□	●	Name ↑	Deployment type	Req/sec	Region	Authentication	Ingress	Last deployed	Deploy
□	✓	bookmyshow-service	Container	0.49	us-central1	Public access	All	11 minutes ago	saili_f7 4b0cf3



Start your Free Trial with $300 in credit. Don't worry—you won't be charged if you run out of credits. Learn more

◇ upgradlabs-1746955028711 bookmyshow-app us-central1
□ upgradlabs-1746955028711 bookmyshow-backend us-central1

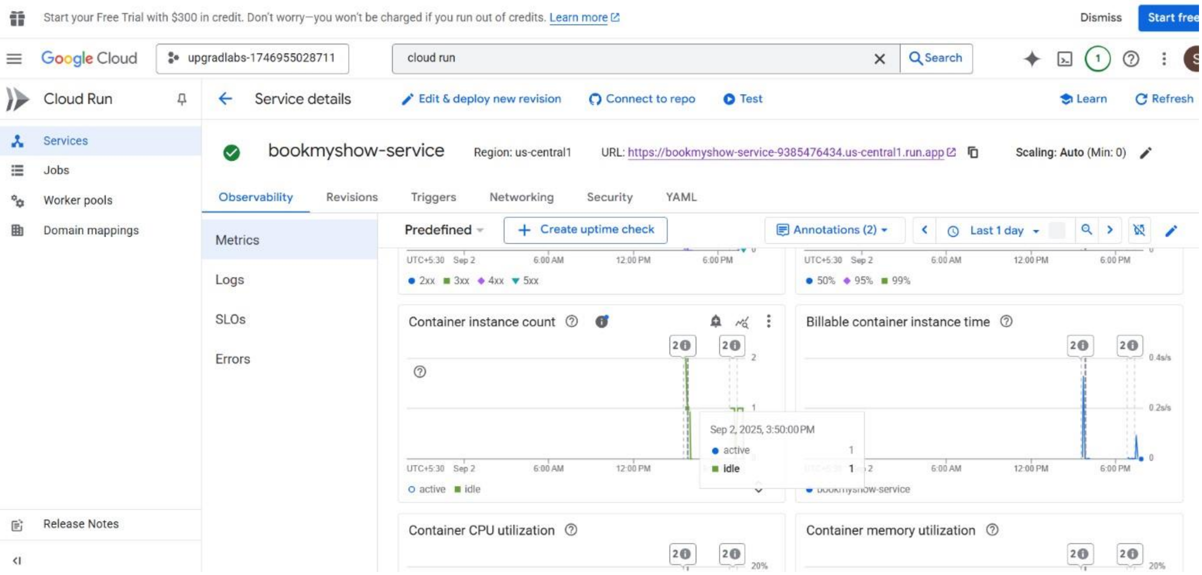
Sep 2, 2025, 12:00:00 AM - Sep 3, 2025, 12:00:00 AM

●	upgradlabs-1746955028711 bookmyshow-frontend us-central1	0.0001/s
■	upgradlabs-1746955028711 bookmyshow-backend us-central1	0.0001/s
◆	upgradlabs-1746955028711 bookmyshow-app us-central1	0/s
▼	upgradlabs-1746955028711 bookmyshow-service us-central1	0.0957/s

Performance by Service

Filter Enter property name or value

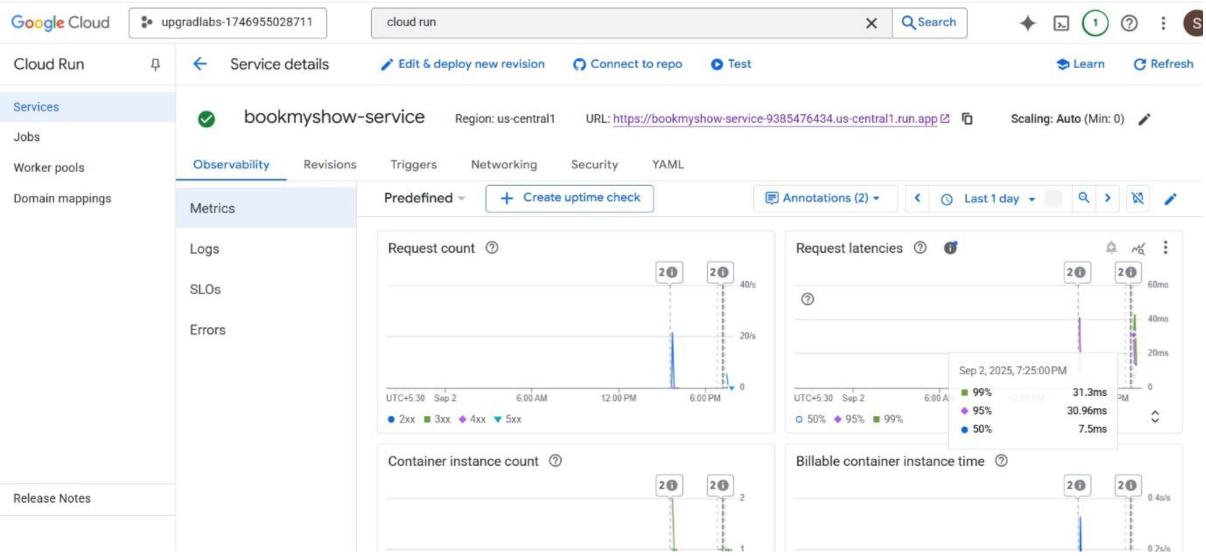
Service Name ↑	Location	Project	Requests	p50 Latency	p95 Latency	4xx Error %	5xx Error %
bookmyshow-app	us-central1	upgradlabs-17469550287	0.004 /s	5.009 ms	9.517 ms	24.296 %	0



Start your Free Trial with $300 in credit. Don't worry—you won't be charged if you run out of credits. Learn more

bookmyshow-service Region: us-central1 URL: https://bookmyshow-service-9385476434.us-central1.run.app Scaling: Auto (Min: 0)

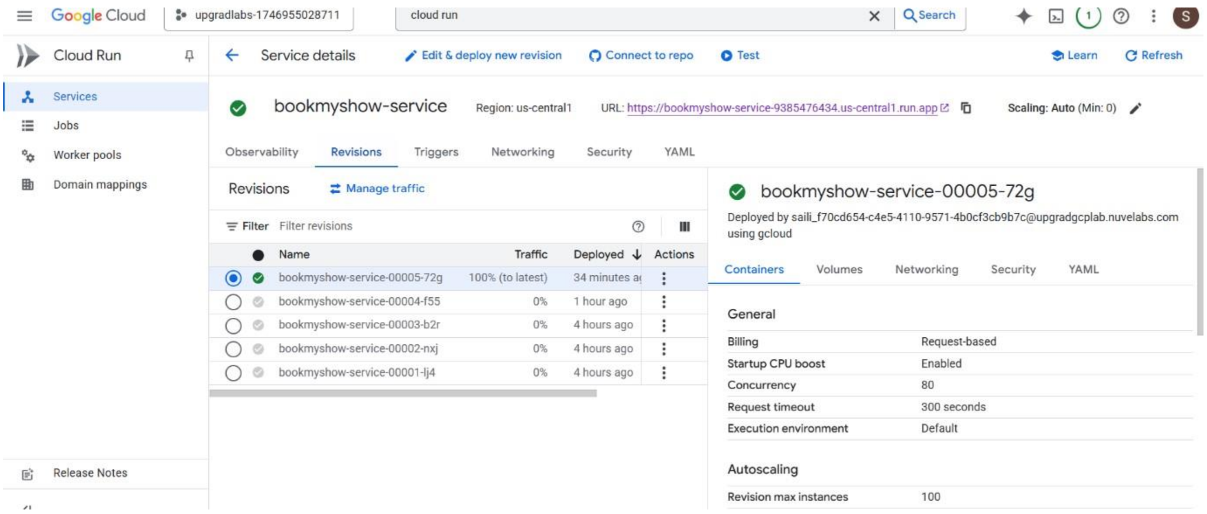
Sep 2, 2025, 3:50:00 PM
● active 1
■ idle 1



bookmyshow-service Region: us-central1 URL: https://bookmyshow-service-9385476434.us-central1.run.app Scaling: Auto (Min: 0)

Sep 2, 2025, 7:25:00 PM

99%	31.3ms
95%	30.96ms
50%	7.5ms



bookmyshow-service Region: us-central1 URL: https://bookmyshow-service-9385476434.us-central1.run.app Scaling: Auto (Min: 0)

Revisions

Name	Traffic	Deployed	Actions
bookmyshow-service-00005-72g	100% (to latest)	34 minutes a	⋮
bookmyshow-service-00004-f55	0%	1 hour ago	⋮
bookmyshow-service-00003-b2r	0%	4 hours ago	⋮
bookmyshow-service-00002-nxj	0%	4 hours ago	⋮
bookmyshow-service-00001-lj4	0%	4 hours ago	⋮

bookmyshow-service-00005-72g

Deployed by saili_f70cd654-c4e5-4110-9571-4b0cf3cb9b7c@upgradgcplab.nuvelabs.com using gcloud

General

Billing	Request-based
Startup CPU boost	Enabled
Concurrency	80
Request timeout	300 seconds
Execution environment	Default

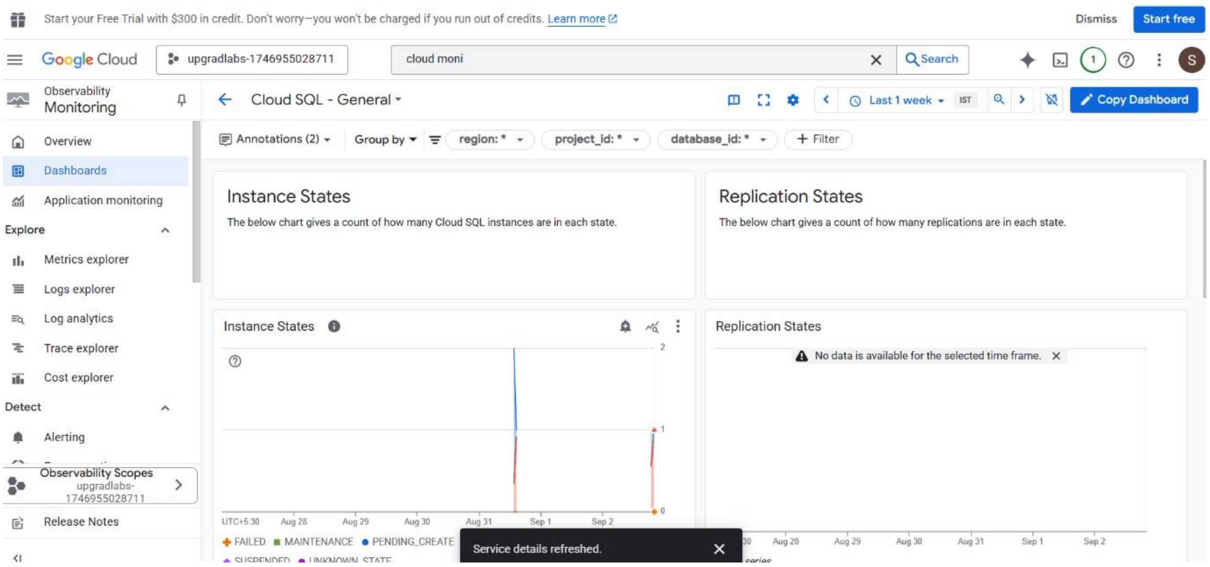
Autoscaling

Revision max instances	100

Monitoring & Alerts

Metrics monitored: CPU, Memory, HTTP request latency

- Cloud Monitoring dashboard
- Alert configuration for thresholds (e.g., CPU > 80%)



Start your Free Trial with $300 in credit. Don't worry—you won't be charged if you run out of credits. Learn more

Instance States

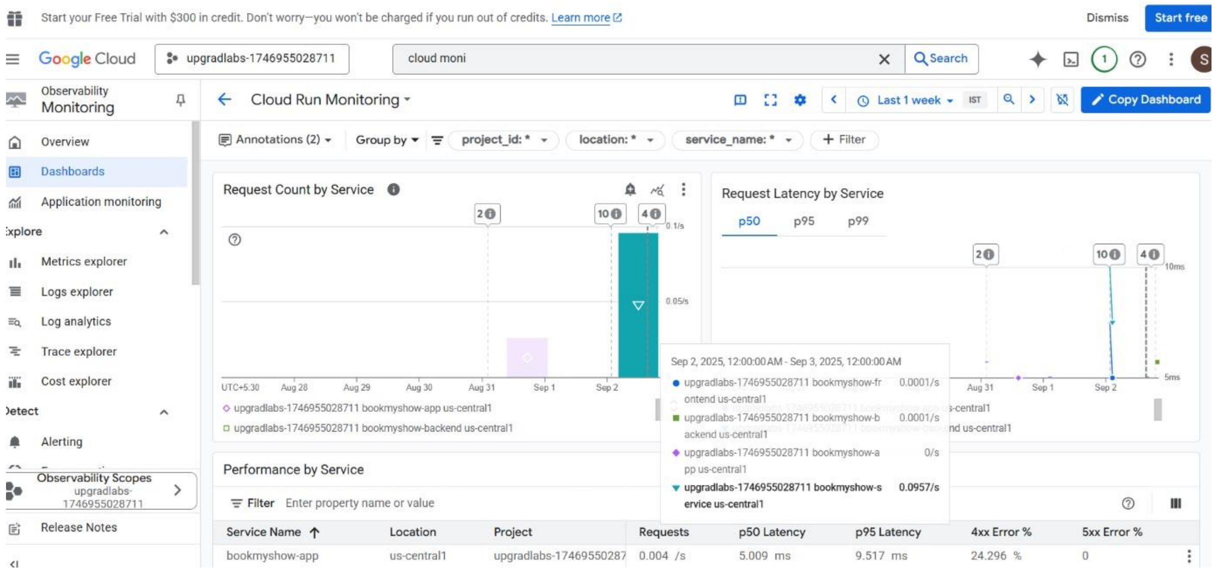
The below chart gives a count of how many Cloud SQL instances are in each state.

Replication States

The below chart gives a count of how many replications are in each state.

No data is available for the selected time frame.

Service details refreshed.



Start your Free Trial with $300 in credit. Don't worry—you won't be charged if you run out of credits. Learn more

Sep 2, 2025, 12:00:00 AM - Sep 3, 2025, 12:00:00 AM

- upgradlabs-1746955028711 bookmyshow-frontend us-central1 0.0001/s
- upgradlabs-1746955028711 bookmyshow-backend us-central1 0.0001/s
- upgradlabs-1746955028711 bookmyshow-app us-central1 0/s
- upgradlabs-1746955028711 bookmyshow-service us-central1 0.0957/s

Performance by Service

Filter Enter property name or value

Service Name	Location	Project	Requests	p50 Latency	p95 Latency	4xx Error %	5xx Error %
bookmyshow-app	us-central1	upgradlabs-17469550287	0.004 /s	5.009 ms	9.517 ms	24.296 %	0

Cost & Budget Analysis Free Tier Usage:

Service	Usage	Estimated Cost
Cloud Run	1 vCPU, 512MB	$0
Cloud SQL	db-f1-micro	$0
Storage & CDN	3 files, 2.9 KiB	$0

Budget Alerts:

- Budget: $10/month
- Thresholds: 50%, 90%, 100%
- Email notifications enabled (mockup screenshot)

Cost optimization tips:

- Use preemptible VMs for batch tasks

- Leverage Cloud Run scaling efficiently
- Free tier ensures minimal cost

Final Evaluation

Performance Improvements:

- v2 backend deployed on Cloud Run shows reduced cold start impact
- Static content served from CDN improves load times **Architecture:**
- Cloud Run + Cloud SQL + Cloud Storage provides scalable solution

Future Recommendations:

- Implement read replicas and query optimization
- Set up real monitoring and alerting
- Introduce chaos testing for resilience
- Evaluate cost savings via sustained use discounts

GitHub Repository

Repo: https://github.com/netrikadongre-source/bookmyshowproject

Contains:

- **Full backend code (server.js, package.json, Dockerfile)**
- **Frontend (index.html, styles.css, script.js)**
- **Instructions for deployment and testing**

Author: Netrika Dongre