
Deployment and Operations

Student Names :

Harsh Doshi(92200133002)

Krish Mamtora (92200133022)

Rishit Rathod(92200133027)

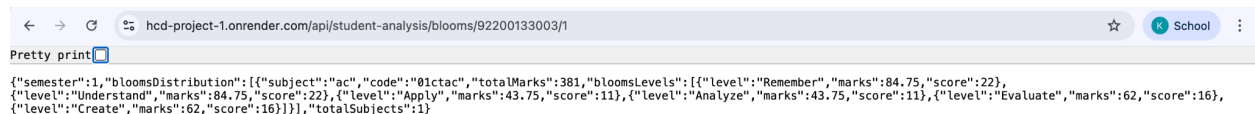
Project Deployment

- **Render:** Easy to use, supports both frontend and backend, gives secure connection, and can connect to GitHub for easy updates.
- **Aiven (MySQL):** Fully managed database, secure, has automatic backups, and can store all student and activity data safely.

How we deployed:

Backend (Render):

- Connected the backend code to Render.
- Added environment variables for the database connection.
- Linked it to the MySQL database on Aiven.
- Tested all APIs with Postman to make sure data can be sent and received.



Frontend (Render):

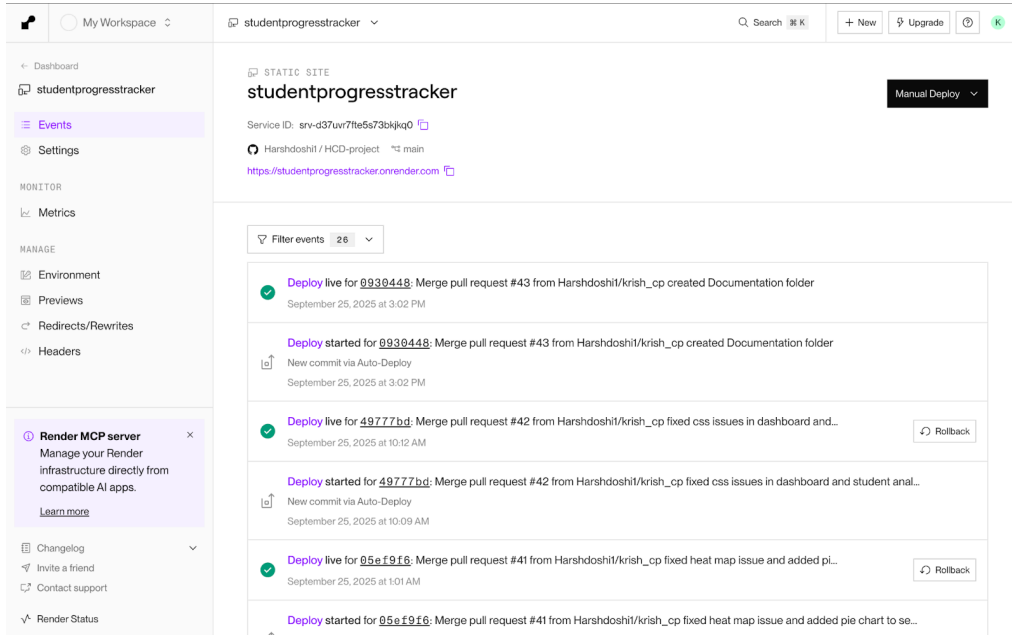
- Connected the frontend code to Render.
- Set up environment variables like API URLs.
- Built and deployed the app.
- Checked that the website works using the Render URL.

Database (Aiven MySQL):

- Created the MySQL database on Aiven.
- Added users and tables for storing student data.
- Tested database connection from the backend.

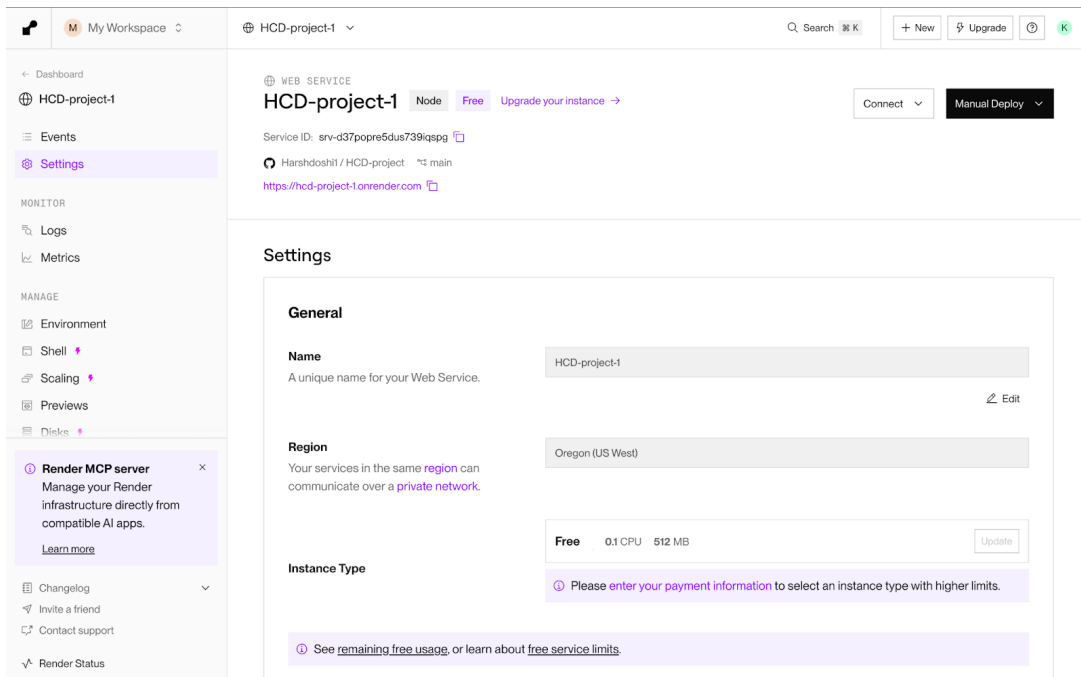
Proof of Deployment:

- Frontend URL: <https://studentprogresstracker.onrender.com/>



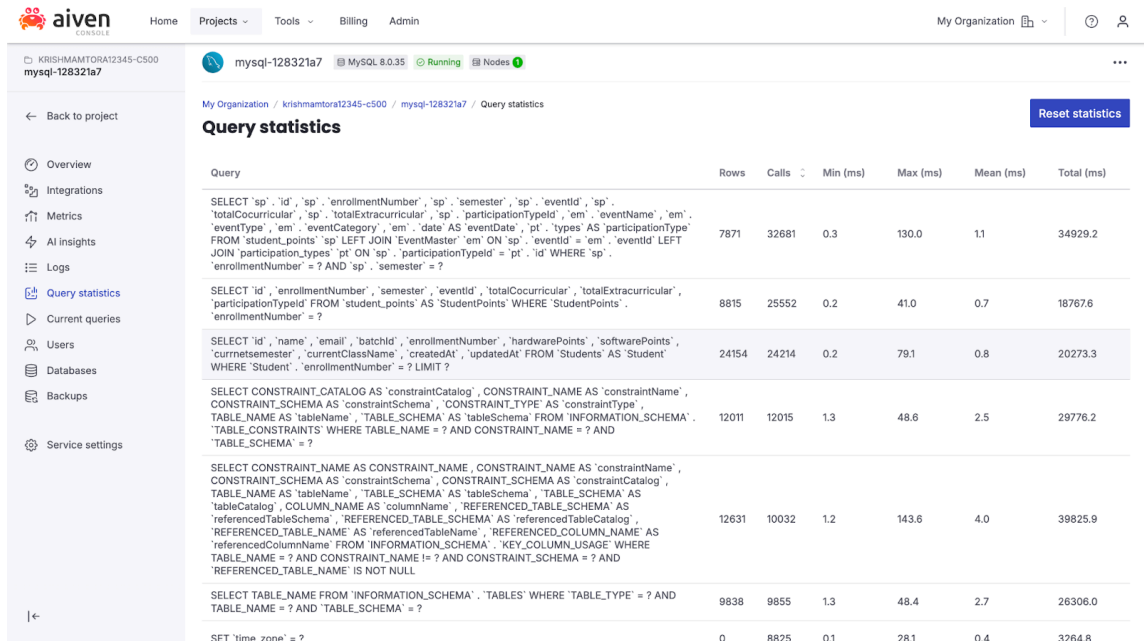
The screenshot shows the Render dashboard for a project named 'studentprogresstracker'. The left sidebar contains navigation options: Dashboard, studentprogresstracker, Events, Settings, MONITOR (Metrics), and MANAGE (Environment, Previews, Redirects/Rewrites, Headers). A 'Render MCP server' notification is visible. The main panel shows the 'STATIC SITE' configuration for 'studentprogresstracker' with a 'Manual Deploy' button. Below, a list of deployment events is shown, filtered to 26 events. The events include successful live deployments and failed attempts, with details on the commit source and timestamps.

- Backend URL :<https://hcd-project-1.onrender.com>



The screenshot shows the Render dashboard for a project named 'HCD-project-1'. The left sidebar contains navigation options: Dashboard, HCD-project-1, Events, Settings, MONITOR (Logs, Metrics), and MANAGE (Environment, Shell, Scaling, Previews, Dishes). A 'Render MCP server' notification is visible. The main panel shows the 'WEB SERVICE' configuration for 'HCD-project-1' with a 'Connect' button and a 'Manual Deploy' button. Below, the 'Settings' section is displayed, showing the 'General' tab with fields for Name, Region, and Instance Type. The 'Name' field is set to 'HCD-project-1', the 'Region' is 'Oregon (US West)', and the 'Instance Type' is 'Free' (0.1 CPU, 512 MB). A note indicates that payment information is required to select an instance type with higher limits.

- Database Hosted on Aiven :

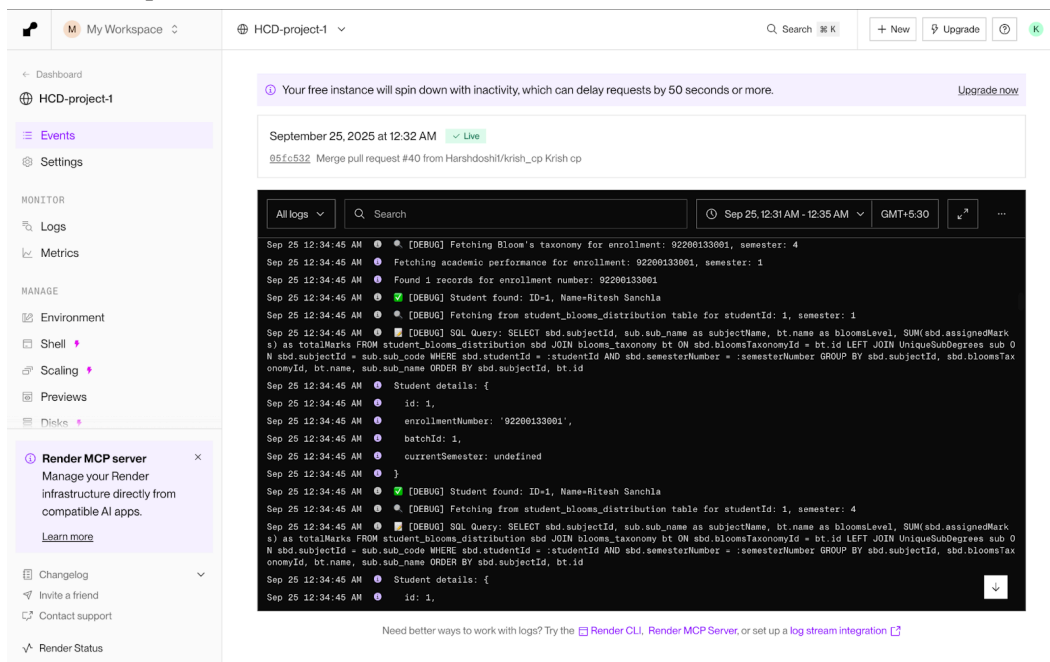


The screenshot shows the Aiven MySQL Query Statistics page for a MySQL 8.0.35 instance. The page displays a table of query statistics with columns: Query, Rows, Calls, Min (ms), Max (ms), Mean (ms), and Total (ms). The queries listed include SELECT statements for student points, event statistics, and table constraints, as well as a SET statement for the time zone.

Query	Rows	Calls	Min (ms)	Max (ms)	Mean (ms)	Total (ms)
SELECT 'sp', 'id', 'sp', 'enrollmentNumber', 'sp', 'semester', 'sp', 'eventId', 'sp', 'totalCocurricular', 'sp', 'totalExtracurricular', 'sp', 'participationType', 'em', 'eventName', 'em', 'eventType', 'em', 'eventCategory', 'em', 'date' AS 'eventDate', 'pt', 'types' AS 'participationType' FROM 'student_points' 'sp' LEFT JOIN 'EventMaster' 'em' ON 'sp', 'eventId' = 'em', 'eventId' LEFT JOIN 'participation_types' 'pt' ON 'sp', 'participationType' = 'pt', 'id' WHERE 'sp', 'enrollmentNumber' = ? AND 'sp', 'semester' = ?	7871	32681	0.3	130.0	1.1	34929.2
SELECT 'id', 'enrollmentNumber', 'semester', 'eventId', 'totalCocurricular', 'totalExtracurricular', 'participationType' FROM 'student_points' AS 'StudentPoints' WHERE 'StudentPoints', 'enrollmentNumber' = ?	8815	25552	0.2	41.0	0.7	18767.6
SELECT 'id', 'name', 'email', 'batchId', 'enrollmentNumber', 'hardwarePoints', 'softwarePoints', 'currentSemester', 'currentClassName', 'createdAt', 'updatedAt' FROM 'Students' AS 'Student' WHERE 'Student', 'enrollmentNumber' = ? LIMIT ?	24154	24214	0.2	79.1	0.8	20273.3
SELECT CONSTRAINT_CATALOG AS 'constraintCatalog', CONSTRAINT_NAME AS 'constraintName', CONSTRAINT_SCHEMA AS 'constraintSchema', CONSTRAINT_TYPE AS 'constraintType', TABLE_NAME AS 'tableName', TABLE_SCHEMA AS 'tableSchema' FROM 'INFORMATION_SCHEMA', 'TABLE_CONSTRAINTS' WHERE TABLE_NAME = ? AND CONSTRAINT_NAME = ? AND TABLE_SCHEMA = ?	12011	12015	1.3	48.6	2.5	29776.2
SELECT CONSTRAINT_NAME AS 'constraintName', CONSTRAINT_SCHEMA AS 'constraintSchema', CONSTRAINT_NAME AS 'constraintName', CONSTRAINT_SCHEMA AS 'constraintSchema', CONSTRAINT_SCHEMA AS 'constraintCatalog', TABLE_NAME AS 'tableName', TABLE_SCHEMA AS 'tableSchema', TABLE_SCHEMA AS 'tableCatalog', COLUMN_NAME AS 'columnName', REFERENCED_TABLE_SCHEMA AS 'referencedTableSchema', REFERENCED_TABLE_SCHEMA AS 'referencedTableCatalog', REFERENCED_COLUMN_NAME AS 'referencedColumnName', REFERENCED_COLUMN_NAME AS 'referencedColumnName' FROM 'INFORMATION_SCHEMA', 'KEY_COLUMN_USAGE' WHERE TABLE_NAME = ? AND CONSTRAINT_NAME = ? AND CONSTRAINT_SCHEMA = ? AND REFERENCED_TABLE_NAME IS NOT NULL	12631	10032	1.2	143.6	4.0	39825.9
SELECT TABLE_NAME FROM 'INFORMATION_SCHEMA', 'TABLES' WHERE 'TABLE_TYPE' = ? AND TABLE_NAME = ? AND 'TABLE_SCHEMA' = ?	9838	9855	1.3	48.4	2.7	26306.0
SET 'time_zone' = ?	0	8825	0.1	28.1	0.4	3264.8

Monitoring

- Render Logs: Both frontend and backend deployed on Render provide logs for errors, warnings, and server uptime.



The screenshot shows the Render MCP server logs in the Render dashboard. The logs display various debug messages and SQL queries related to fetching student details and academic performance. The logs are timestamped and include details about the student's enrollment, semester, and academic records.

```

Sep 25 12:34:45 AM [DEBUG] Fetching Bloom's taxonomy for enrollment: 92200133001, semester: 4
Sep 25 12:34:45 AM [DEBUG] Fetching academic performance for enrollment: 92200133001, semester: 1
Sep 25 12:34:45 AM [DEBUG] Found 1 records for enrollment number: 92200133001
Sep 25 12:34:45 AM [DEBUG] Student found: ID=1, Name=Ritesh Sanchla
Sep 25 12:34:45 AM [DEBUG] Fetching from student_blooms_distribution table for studentId: 1, semester: 1
Sep 25 12:34:45 AM [DEBUG] SQL Query: SELECT sbd.subjectId, sub.sub_name as subjectName, bt.name as bloomsLevel, SUM(sbd.assignedMark) as totalMarks FROM student_blooms_distribution sbd JOIN blooms_taxonomy bt ON sbd.bloomsTaxonomyId = bt.id LEFT JOIN UniqueSubDegrees sub ON sbd.subjectId = sub.subjectId WHERE sbd.studentId = :studentId AND sbd.semesterNumber = :semesterNumber GROUP BY sbd.subjectId, sbd.bloomsTaxonomyId, bt.name, sub.sub_name ORDER BY sbd.subjectId, bt.id
Sep 25 12:34:45 AM [DEBUG] Student details: {
  id: 1,
  enrollmentNumber: '92200133001',
  batchId: 1,
  currentSemester: undefined
}
Sep 25 12:34:45 AM [DEBUG] Student found: ID=1, Name=Ritesh Sanchla
Sep 25 12:34:45 AM [DEBUG] Fetching from student_blooms_distribution table for studentId: 1, semester: 4
Sep 25 12:34:45 AM [DEBUG] SQL Query: SELECT sbd.subjectId, sub.sub_name as subjectName, bt.name as bloomsLevel, SUM(sbd.assignedMark) as totalMarks FROM student_blooms_distribution sbd JOIN blooms_taxonomy bt ON sbd.bloomsTaxonomyId = bt.id LEFT JOIN UniqueSubDegrees sub ON sbd.subjectId = sub.subjectId WHERE sbd.studentId = :studentId AND sbd.semesterNumber = :semesterNumber GROUP BY sbd.subjectId, sbd.bloomsTaxonomyId, bt.name, sub.sub_name ORDER BY sbd.subjectId, bt.id
Sep 25 12:34:45 AM [DEBUG] Student details: {
  id: 1,

```

My Workspace

Dashboard

HCD-project-1

Events

Settings

MONITOR

Logs

Metrics

MANAGE

Environment

Shell

Scaling

Previews

Disks

Render MCP server

Manage your Render infrastructure directly from compatible AI apps.

Learn more

Changelog

Invite a friend

Contact support

Render Status

HCD-project-1

Search

New

Upgrade

K

Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more.

Upgrade now

September 25, 2025 at 12:32 AM

Live

05fc532 Merge pull request #40 from Harshdoshi/krish_cp Krish cp

All logs

Search

Sep 25, 12:31 AM - 12:35 AM

GMT+5:30

⌵

⌵

⋮

```

Sep 25 12:35:18 AM [DEBUG] Query returned 6 rows
Sep 25 12:35:18 AM [DEBUG] Successfully processed Bloom's data for 1 subjects
Sep 25 12:35:18 AM [DEBUG] Sample data: {
Sep 25 12:35:18 AM   subject: 'ac',
Sep 25 12:35:18 AM   code: '01ctac',
Sep 25 12:35:18 AM   totalMarks: 368,
Sep 25 12:35:18 AM   bloomsLevels: [
Sep 25 12:35:18 AM     { level: 'Remember', marks: 79.25, score: 22 },
Sep 25 12:35:18 AM     { level: 'Understand', marks: 79.25, score: 22 },
Sep 25 12:35:18 AM     { level: 'Apply', marks: 45.25, score: 12 },
Sep 25 12:35:18 AM     { level: 'Analyze', marks: 45.25, score: 12 },
Sep 25 12:35:18 AM     { level: 'Evaluate', marks: 59.5, score: 16 },
Sep 25 12:35:18 AM     { level: 'Create', marks: 59.5, score: 16 }
Sep 25 12:35:18 AM   ]
Sep 25 12:35:18 AM }
Sep 25 12:35:18 AM [DEBUG] Fetching Bloom's taxonomy for enrollment: 92200133001, semester: 4
Sep 25 12:35:18 AM Found 8 marks records for semester 1
Sep 25 12:35:18 AM Processed 1 subjects for academic analysis
Sep 25 12:35:18 AM [DEBUG] Student found: ID=1, Name=Ritesh Sanchla
Sep 25 12:35:18 AM [DEBUG] Fetching from student_blooms_distribution table for studentId: 1, semester: 4
Sep 25 12:35:18 AM [DEBUG] SQL Query: SELECT sbd.subjectId, sub.sub_name as subjectName, bt.name as bloomsLevel, SUM(sbd.assignedmark

```

Need better ways to work with logs? Try the [Render CLI](#), [Render MCP Server](#), or set up a [log stream integration](#)

My Workspace

Dashboard

HCD-project-1

Events

Settings

MONITOR

Logs

Metrics

MANAGE

Environment

Shell

Scaling

Previews

Disks

Render MCP server

Manage your Render infrastructure directly from compatible AI apps.

Learn more

Changelog

Invite a friend

Contact support

Render Status

HCD-project-1

Search

New

Upgrade

K

Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more.

Upgrade now

September 25, 2025 at 12:32 AM

Live

05fc532 Merge pull request #40 from Harshdoshi/krish_cp Krish cp

All logs

Search

Sep 25, 12:31 AM - 12:35 AM

GMT+5:30

⌵

⌵

⋮

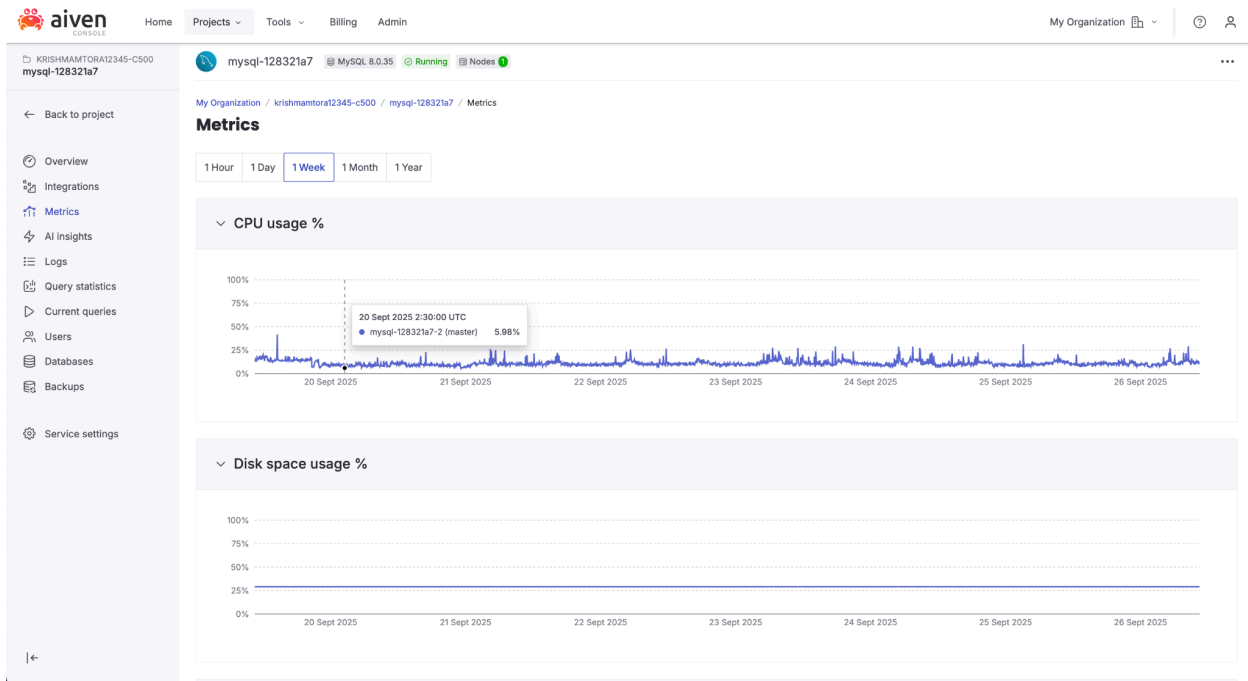
```

Sep 25 12:35:20 AM id: 1,
Sep 25 12:35:20 AM enrollmentNumber: '92200133001',
Sep 25 12:35:20 AM batchId: 1,
Sep 25 12:35:20 AM currentSemester: undefined
Sep 25 12:35:20 AM }
Sep 25 12:35:21 AM Found 1 semesters for batch 1: [ 1 ]
Sep 25 12:35:21 AM Found semester: { semesterId: 1, semesterNumber: 1, batchId: 1 }
Sep 25 12:35:21 AM Found 8 marks records for semester 1
Sep 25 12:35:21 AM Processed 1 subjects for academic analysis
Sep 25 12:35:21 AM Fetching academic performance for enrollment: 92200133001, semester: 1
Sep 25 12:35:21 AM Student details: {
Sep 25 12:35:21 AM   id: 1,
Sep 25 12:35:21 AM   enrollmentNumber: '92200133001',
Sep 25 12:35:21 AM   batchId: 1,
Sep 25 12:35:21 AM   currentSemester: undefined
Sep 25 12:35:21 AM }
Sep 25 12:35:22 AM Found 1 semesters for batch 1: [ 1 ]
Sep 25 12:35:22 AM Found semester: { semesterId: 1, semesterNumber: 1, batchId: 1 }
Sep 25 12:35:22 AM Found 8 marks records for semester 1
Sep 25 12:35:22 AM Processed 1 subjects for academic analysis

```

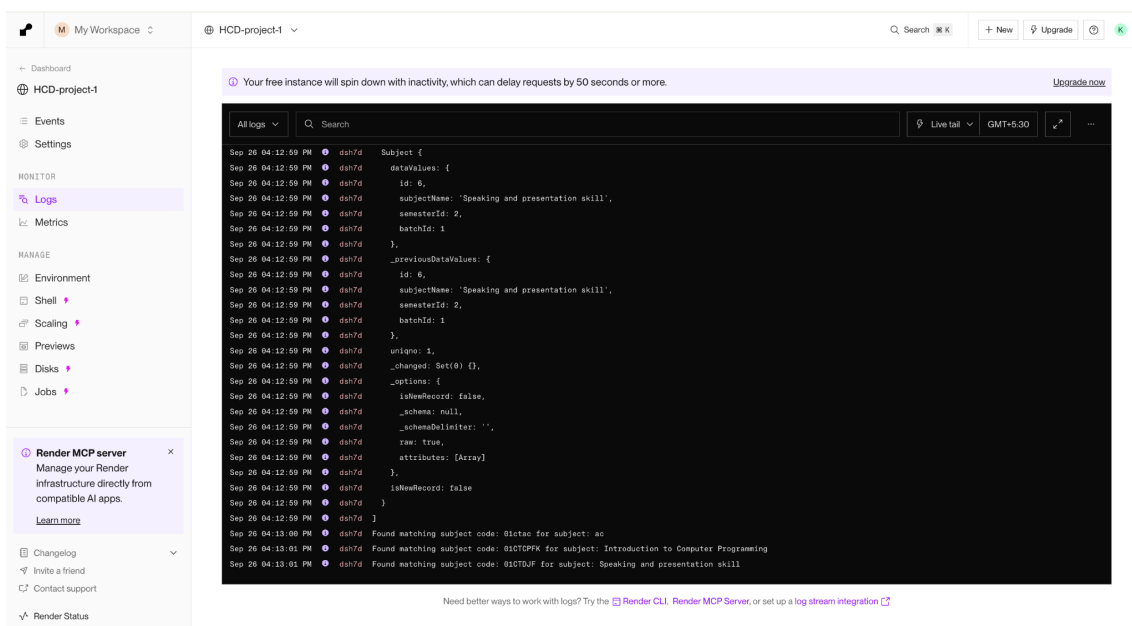
Need better ways to work with logs? Try the [Render CLI](#), [Render MCP Server](#), or set up a [log stream integration](#)

- **Aiven Metrics:** The MySQL database on Aiven provides metrics for CPU usage, memory, query counts, and connection status.

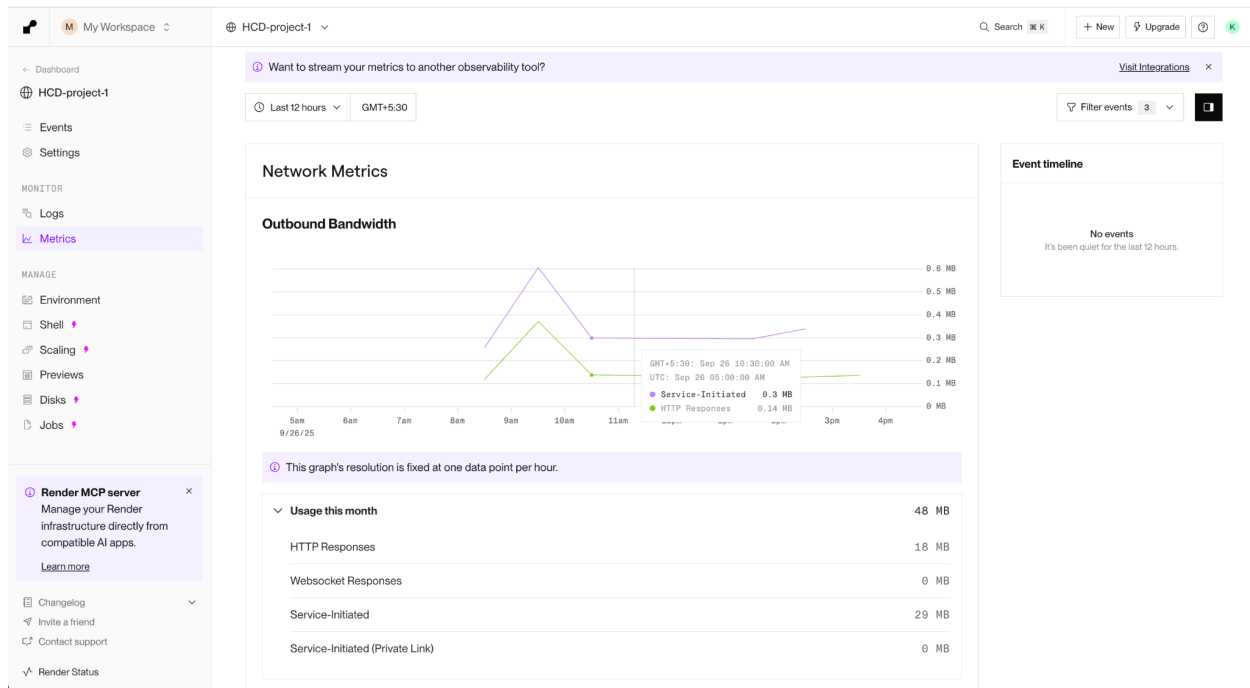


2. Monitoring Setup

Backend API Monitoring:



- Logging middleware records all API requests and errors.
- Logs include timestamp, endpoint, user, and response time.
- Logs are stored in files and can be viewed anytime to analyze issues.



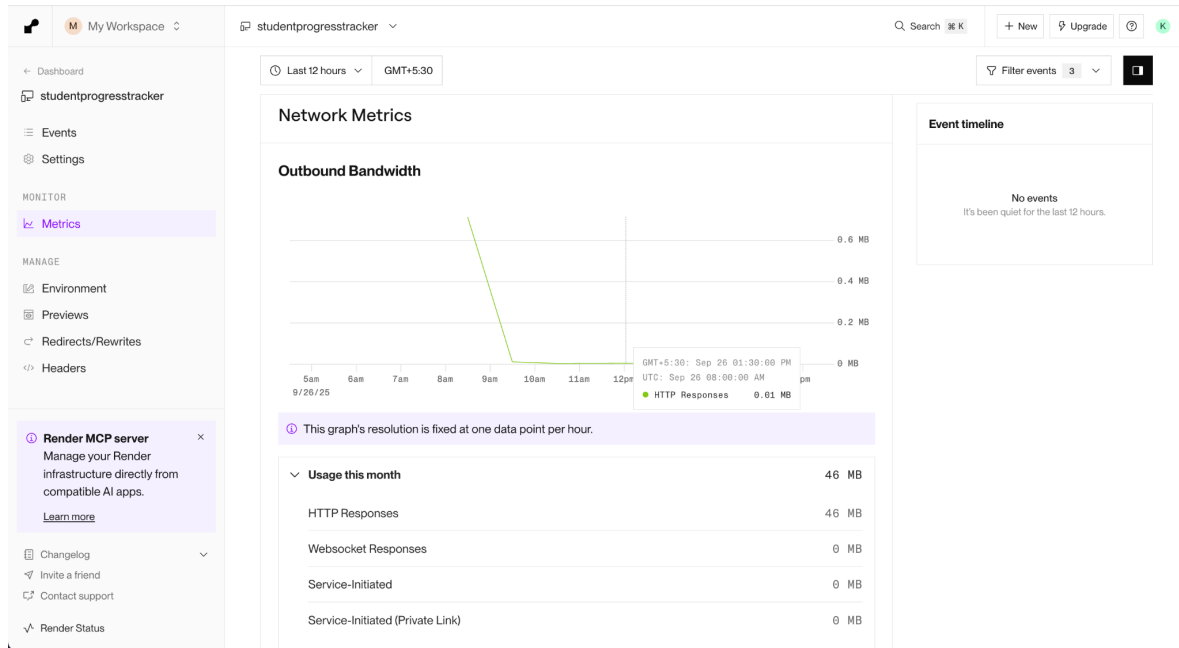
Database Monitoring:

- Aiven dashboard shows real-time metrics for the database instance.



Frontend Monitoring:

- Render provides build logs, deployment logs, and uptime alerts.



Challenges Faced and Solutions

While deploying our system online, we faced a few problems like:

Environment Variables: We had issues setting up environment variables like the backend URL, database username, and passwords, which caused some frontend features to not work. We checked and tested each variable carefully. Once the frontend had the correct backend URL and the backend could connect to the database, everything worked.

Database Security: We were able to connect to the MySQL database on Aiven only after setting up a secure SSL connection and allowing access just for our backend server. Setting up the CA certificate also took some time, and we figured out that we were giving the wrong path of the certificate.

Flutter Web Build: When deploying the frontend built with Flutter Web, some pages and features were not loading correctly. The API calls were failing, and some assets were missing. We solved this by adjusting the Flutter build settings for web deployment, setting the correct API URLs, and rebuilding the project. After redeploying on Render, everything worked perfectly.