5th DECEMBER 2024

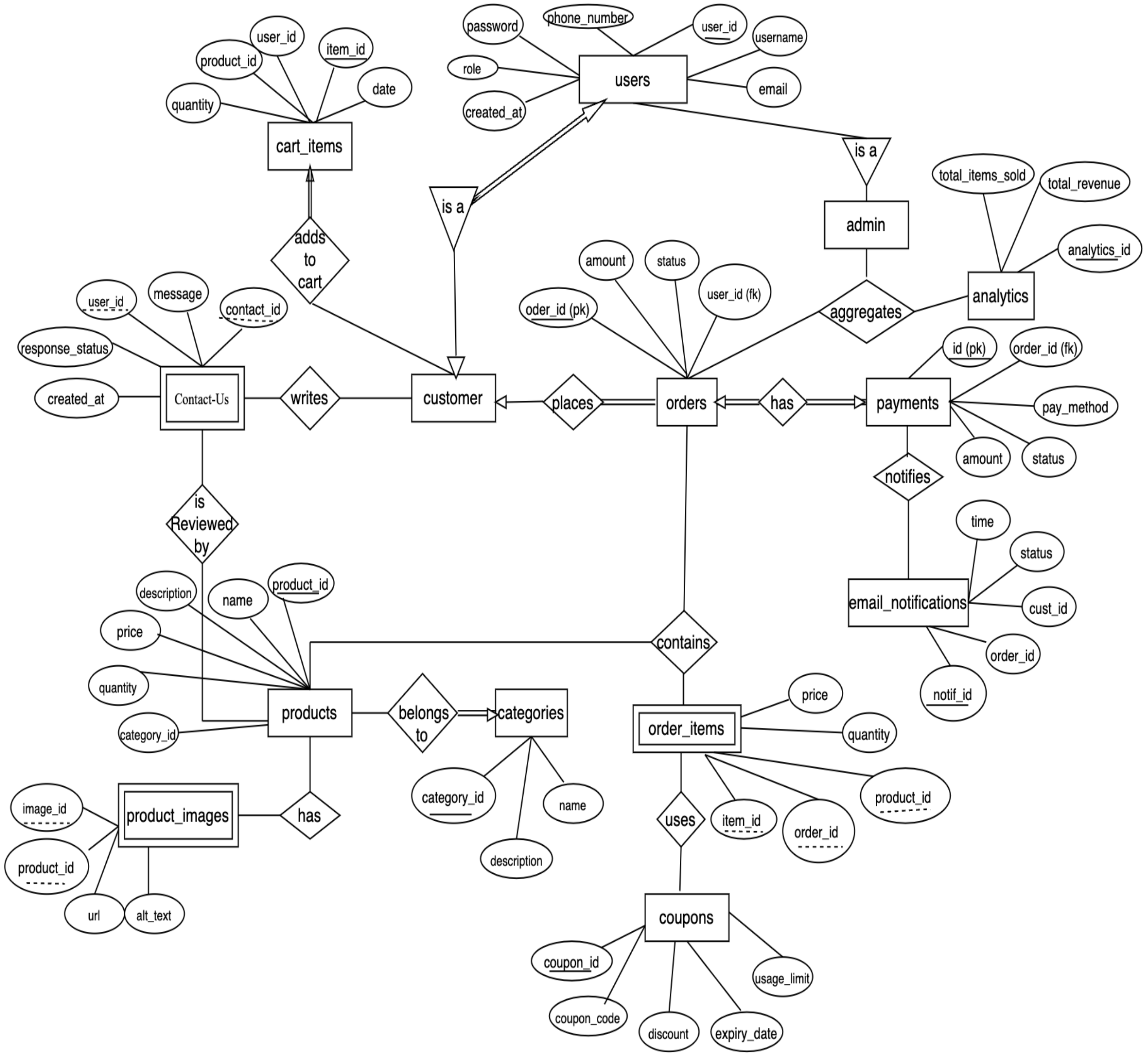
Final Deliverable

ShopWave Web Application

VENKATA VEERA SOMASEKHAR DARISI (001624347)
ANITHA REDDY BAPATHU (001623789)
DARCY WOODRUFF (001668118)

1. Database Design

Final ER Diagram



Final version of tables written in Schema Statements

1. cart_items (item_id, date, user_id, product_id, quantity)

2. ContactUs(contact_id, message, user_id, response_status, created_at)

3. **Orders** (order_id, amount, status, user_id)

4. **Users** (user_id, username, email, phone_number, password, role, created_at)

5. **Analytics** (analytics_id, total_revenue, total_items_sold)

6. **Payments** (payment_id, order_id, pay_method, amount, status)

7. **EmailNotifications** (notification_id, order_id, customer_is, status, time)

8. **Products** (product_id, name, description, price, quantity, category_id)

9. **Categories** (category_id, name, description)

10. **ProductImages** (image_id, product_id, url, alt_text)

11. **Coupons** (coupon_id, coupon_code, discount, expiry_date, usage_limit)

12. **OrderItems** (order_id , item_id, product_id, quantity, price)

13. **aggregates** (analytics_id, order_id)

14. **addsToCart** (item_id, customerId)

15. **uses** (coupon_id, item_id, product_id, order_id)

16. **belongs_to** (category_id, product_id)

17**. writes** (user_id, contact_id)

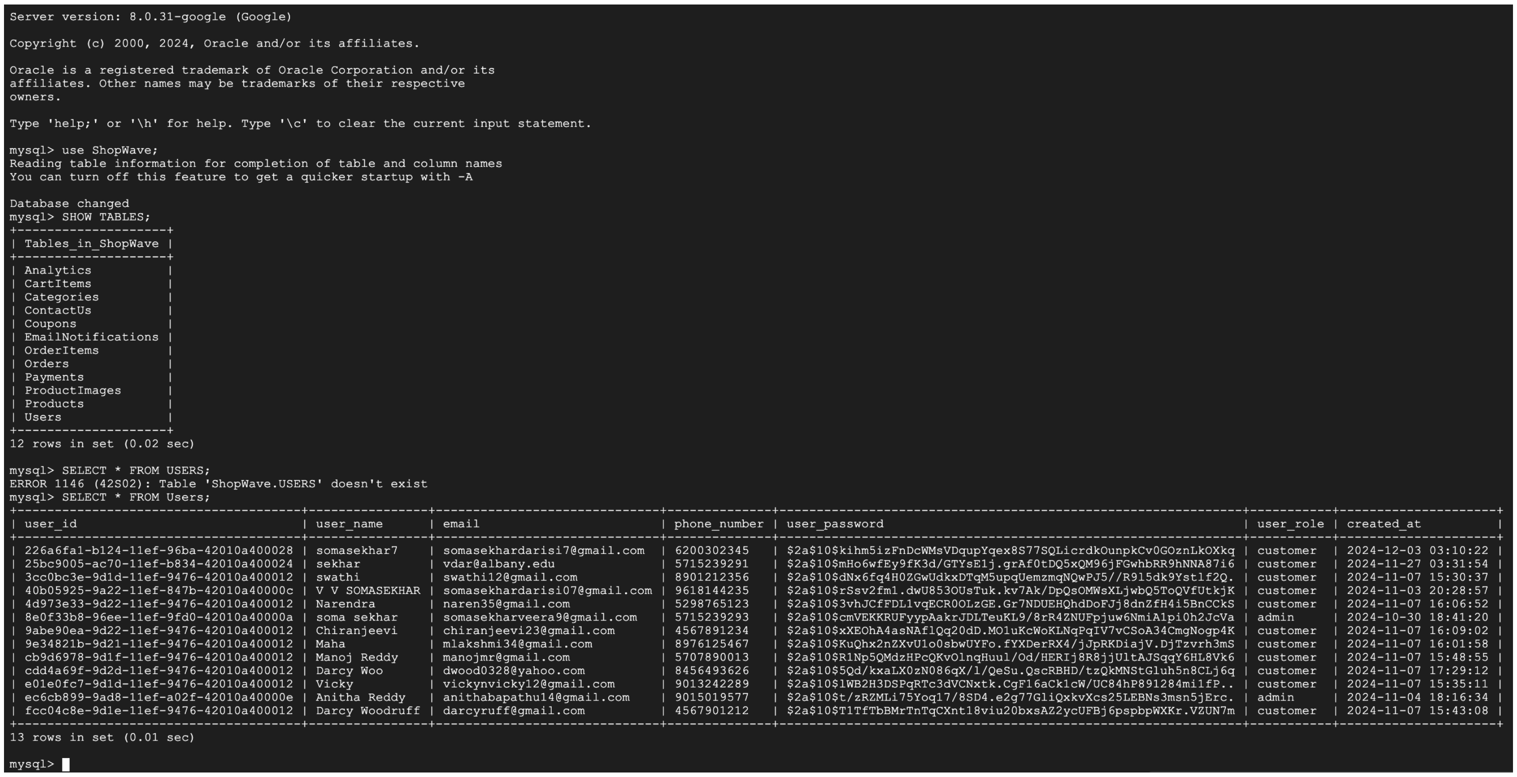
18. **places** (customer_id, order_id)

19. **has** (payment_id, order_id)

20. **has** (product_id, image_id)

2. Database Programming

I have hosted the Database in the GCP. Below, screenshot provides the evidence to the hosted DB and the tables created in the hosted Database named “ShopWave”.



```
Server version: 8.0.31-google (Google)

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use ShopWave;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SHOW TABLES;
+--------------------+
| Tables_in_ShopWave |
+--------------------+
| Analytics          |
| CartItems          |
| Categories         |
| ContactUs          |
| Coupons            |
| EmailNotifications |
| OrderItems         |
| Orders             |
| Payments           |
| ProductImages      |
| Products           |
| Users              |
+--------------------+
12 rows in set (0.02 sec)

mysql> SELECT * FROM USERS;
ERROR 1146 (42S02): Table 'ShopWave.USERS' doesn't exist
mysql> SELECT * FROM Users;
+--------------------------------------+----------------+------------------------------+--------------+--------------------------------------------------------------+-----------+---------------------+
| user_id                              | user_name      | email                        | phone_number | user_password                                                | user_role | created_at          |
+--------------------------------------+----------------+------------------------------+--------------+--------------------------------------------------------------+-----------+---------------------+
| 226a6fa1-b124-11ef-96ba-42010a400028 | somasekhar7    | somasekhardarisi7@gmail.com  | 6200302345   | $2a$10$kihm5izFnDcWMsVDqupYqex8S77SQLicrdkOunpkCv0GOznLkOXkq | customer  | 2024-12-03 03:10:22 |
| 25bc9005-ac70-11ef-b834-42010a400024 | sekhar         | vdar@albany.edu              | 5715239291   | $2a$10$mHo6wfEy9fK3d/GTYsE1j.grAf0tDQ5xQM96jFGwhbRR9hNNA87i6 | customer  | 2024-11-27 03:31:54 |
| 3cc0bc3e-9d1d-11ef-9476-42010a400012 | swathi         | swathi12@gmail.com           | 8901212356   | $2a$10$dNx6fq4H0ZGwUdkxDTqM5upqUemzmqNQwPJ5//R915dk9Ystlf2Q. | customer  | 2024-11-07 15:30:37 |
| 40b05925-9a22-11ef-847b-42010a40000c | V V SOMASEKHAR | somasekhardarisi07@gmail.com | 9618144235   | $2a$10$rSsv2fm1.dwU853OUsTuk.kv7Ak/DpQsOMWsXLjwbQ5ToQVfUtkjK | customer  | 2024-11-03 20:28:57 |
| 4d973e33-9d22-11ef-9476-42010a400012 | Narendra       | naren35@gmail.com            | 5298765123   | $2a$10$3vhJCfFDL1vqECR0OLzGE.Gr7NDUEHQhdDoFJj8dnZfH4i5BnCCkS | customer  | 2024-11-07 16:06:52 |
| 8e0f33b8-96ee-11ef-9fd0-42010a40000a | soma sekhar    | somasekharveera9@gmail.com   | 5715239293   | $2a$10$cmVEKKRUFyypAakrJDLTeuKL9/8rR4ZNUFpjuw6NmiA1pi0h2JcVa | admin     | 2024-10-30 18:41:20 |
| 9abe90ea-9d22-11ef-9476-42010a400012 | Chiranjeevi    | chiranjeevi23@gmail.com      | 4567891234   | $2a$10$xXEOhA4asNAflQq20dD.MO1uKcWoKLNqPqIV7vCSoA34CmgNogp4K | customer  | 2024-11-07 16:09:02 |
| 9e34821b-9d21-11ef-9476-42010a400012 | Maha           | mlakshmi34@gmail.com         | 8976125467   | $2a$10$KuQhx2nZXvU1o0sbwUYFo.fYXDerRX4/jJpRKDiajV.DjTzvrh3mS | customer  | 2024-11-07 16:01:58 |
| cb9d6978-9d1f-11ef-9476-42010a400012 | Manoj Reddy    | manojmr@gmail.com            | 5707890013   | $2a$10$R1Np5QMdzHPcQKvOlnqHuul/Od/HERIj8R8jjUltAJSqqY6HL8Vk6 | customer  | 2024-11-07 15:48:55 |
| cdd4a69f-9d2d-11ef-9476-42010a400012 | Darcy Woo      | dwood0328@yahoo.com          | 8456493626   | $2a$10$5Qd/kxaLX0zN086qX/l/QeSu.QscRBHD/tzQkMNStGluh5n8CLj6q | customer  | 2024-11-07 17:29:12 |
| e01e0fc7-9d1d-11ef-9476-42010a400012 | Vicky          | vickynvicky12@gmail.com      | 9013242289   | $2a$10$lWB2H3DSPqRTc3dVCNxtk.CgF16aCk1cW/UC84hP891284mi1fP.. | customer  | 2024-11-07 15:35:11 |
| ec6cb899-9ad8-11ef-a02f-42010a40000e | Anitha Reddy   | anithabapathu14@gmail.com    | 9015019577   | $2a$10$t/zRZMLi75Yoq17/8SD4.e2g77GliQxkvXcs25LEBNs3msn5jErc. | admin     | 2024-11-04 18:16:34 |
| fcc04c8e-9d1e-11ef-9476-42010a400012 | Darcy Woodruff | darcyruff@gmail.com          | 4567901212   | $2a$10$T1TfTbBMrTnTqCXnt18viu20bxsAZ2ycUFBj6pspbpWXKr.VZUN7m | customer  | 2024-11-07 15:43:08 |
+--------------------------------------+----------------+------------------------------+--------------+--------------------------------------------------------------+-----------+---------------------+
13 rows in set (0.01 sec)

mysql> 
```

I have deployed my web application in the render.io

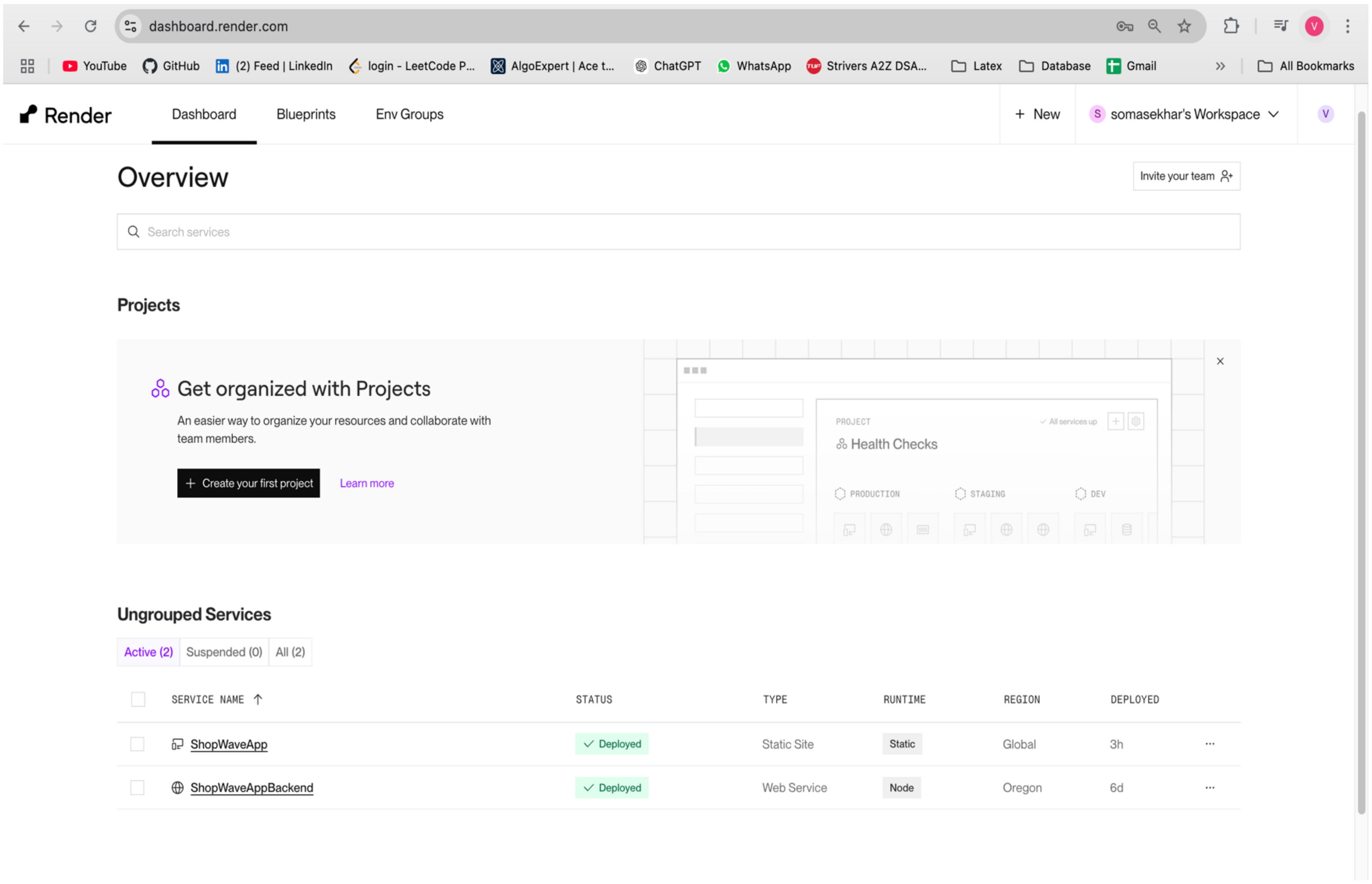
This is the hosted application URL: https://shopwaveapp-p2kq.onrender.com/

As I used React for my web application, to deploy an application like this, we need to push the code to a GitHub repository. After that, we need to create an account on Render. Once the account is created, we link our GitHub repository where we hosted the code.

Next, we click the **+New** icon on the Render dashboard, choose **Static Site**, fill in the required details, and deploy our frontend.

Then, we deploy the backend code following the same steps. However, this time, we choose **Web Service** instead of **Static Site**. We also configure the environment variables used by our application before deploying.

Once both the frontend and backend are deployed on Render, we can access the web application through the URL generated by the frontend. The URL is attached above.



SERVICE NAME	STATUS	TYPE	RUNTIME	REGION	DEPLOYED
ShopWaveApp	✓ Deployed	Static Site	Static	Global	3h
ShopWaveAppBackend	✓ Deployed	Web Service	Node	Oregon	6d

This is how the render dashboard will look like once we deployed the application successfully.

If we would like to run the application in your local environment. We need to get the code from this github repository https://github.com/somasekhar7/ShopWaveWebApp.git

Once you have successfully cloned the code into your local environment, you need to set up the .env file as instructed. Follow the details mentioned in the .env file.

Next, install the Node modules in both the frontend and backend folders. To do this, navigate to the frontend folder by entering the command:

```
cd frontend
```

Then, install the Node modules using: npm install

For the backend, stay in the root directory and run the command:

npm install

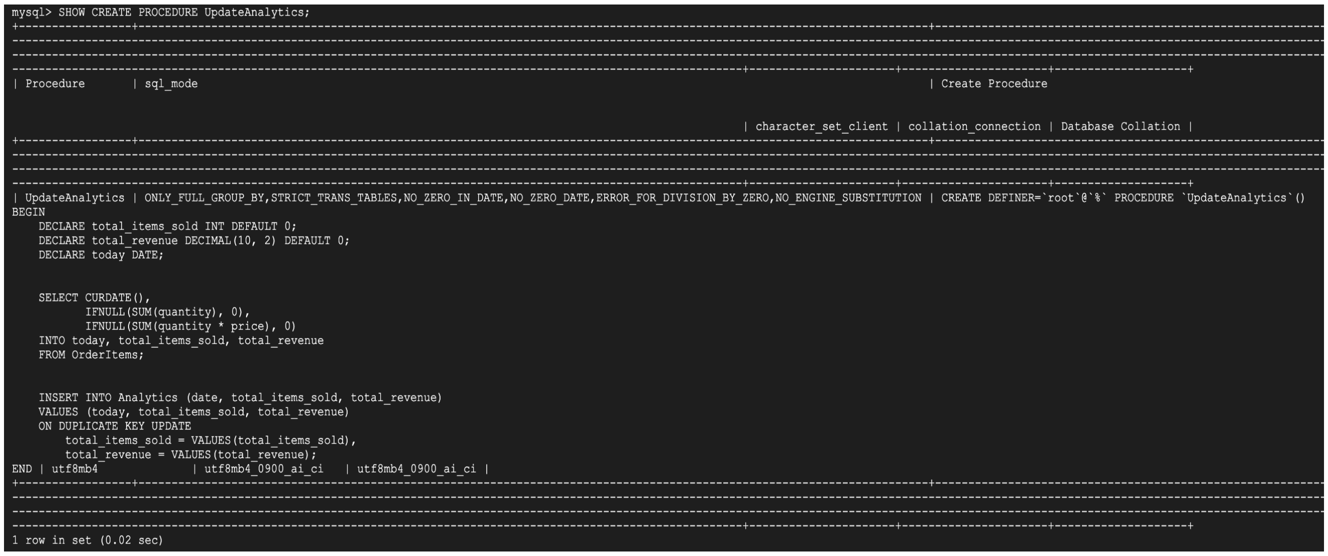
After installing the dependencies, use the following command in both the frontend and backend to start the development servers:

npm run dev

This will allow you to see the application running in your web browser.

I have used two Advanced SQL Commands which are as follows:

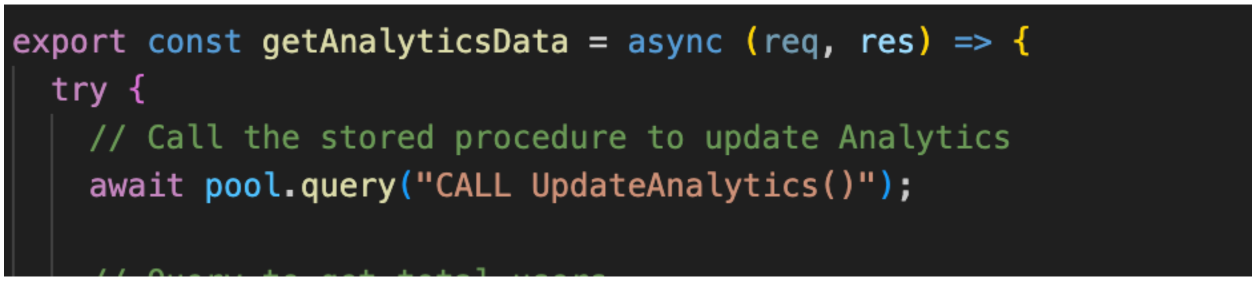
1. Stored Procedure



```
mysql> SHOW CREATE PROCEDURE UpdateAnalytics;
+-----------------+------------------------------------------------------------------------------------------------------------------------+-----------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------+----------------------+----------------------+--------------------+
| Procedure       | sql_mode                                                                                                               | Create Procedure


                                                                                                             | character_set_client | collation_connection | Database Collation |
+-----------------+------------------------------------------------------------------------------------------------------------------------+-----------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------+----------------------+----------------------+--------------------+
| UpdateAnalytics | ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION | CREATE DEFINER=`root`@`%` PROCEDURE `UpdateAnalytics`()
BEGIN
    DECLARE total_items_sold INT DEFAULT 0;
    DECLARE total_revenue DECIMAL(10, 2) DEFAULT 0;
    DECLARE today DATE;


    SELECT CURDATE(),
           IFNULL(SUM(quantity), 0),
           IFNULL(SUM(quantity * price), 0)
    INTO today, total_items_sold, total_revenue
    FROM OrderItems;


    INSERT INTO Analytics (date, total_items_sold, total_revenue)
    VALUES (today, total_items_sold, total_revenue)
    ON DUPLICATE KEY UPDATE
        total_items_sold = VALUES(total_items_sold),
        total_revenue = VALUES(total_revenue);
END | utf8mb4              | utf8mb4_0900_ai_ci   | utf8mb4_0900_ai_ci |
+-----------------+------------------------------------------------------------------------------------------------------------------------+-----------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------+----------------------+----------------------+--------------------+
1 row in set (0.02 sec)
```

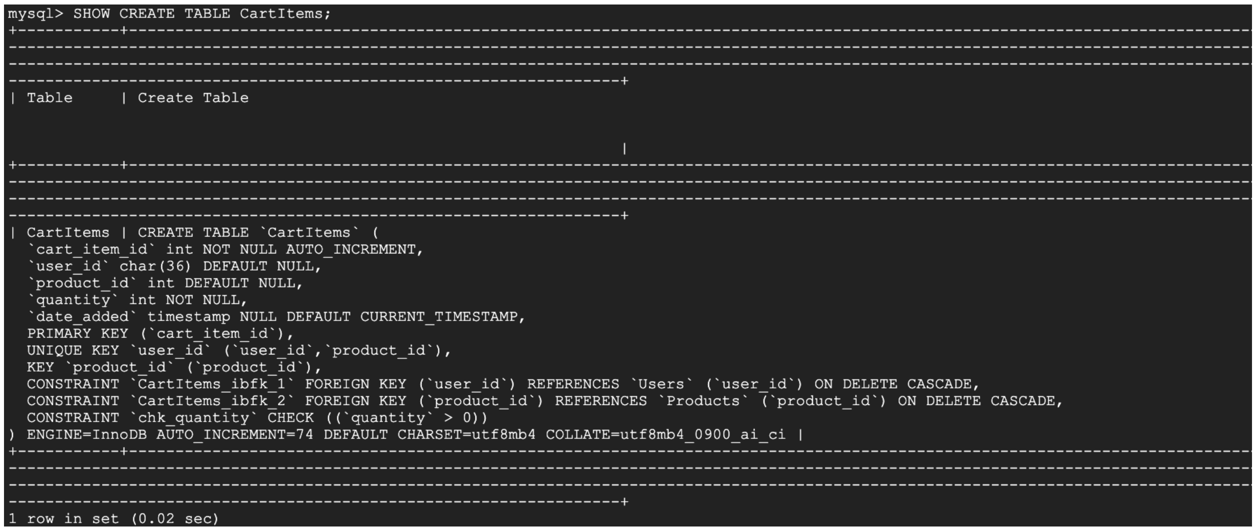
This is the stored procedure that I have used to update the Analytics each time by fetching the updated data from the necessary tables in the DB to display the analytics in the frontend of the application.

In the application I have called this stored procedure in the analytics controller of this route.



```
export const getAnalyticsData = async (req, res) => {
  try {
    // Call the stored procedure to update Analytics
    await pool.query("CALL UpdateAnalytics()");
```

2. Also, I have used the CHECK in the cart table which satisfies the advanced SQL criteria.



```
mysql> SHOW CREATE TABLE CartItems;
+-----------+------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------+
| Table     | Create Table


                                                                  |
+-----------+------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------+
| CartItems | CREATE TABLE `CartItems` (
  `cart_item_id` int NOT NULL AUTO_INCREMENT,
  `user_id` char(36) DEFAULT NULL,
  `product_id` int DEFAULT NULL,
  `quantity` int NOT NULL,
  `date_added` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`cart_item_id`),
  UNIQUE KEY `user_id` (`user_id`,`product_id`),
  KEY `product_id` (`product_id`),
  CONSTRAINT `CartItems_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES `Users` (`user_id`) ON DELETE CASCADE,
  CONSTRAINT `CartItems_ibfk_2` FOREIGN KEY (`product_id`) REFERENCES `Products` (`product_id`) ON DELETE CASCADE,
  CONSTRAINT `chk_quantity` CHECK ((`quantity` > 0))
) ENGINE=InnoDB AUTO_INCREMENT=74 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+-----------+------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------+
1 row in set (0.02 sec)
```

I have used the CHECK constraint here which allows us to insert the data record only when the quantity of the product is greater than 0.

3. Database Security at Database level

As I had set up the database security for the developers end.

So, I created two users. One of them is “root” user who has the access to the entire database which means he can be able to create, update, drop schema and alter tables.

Also, I have created one more user named “new_user” who performs only the CRUD operations. But he cannot be able to drop (or) alter the tables and schema of DB.

In this way we ensure that only the user with dedicated access can perform all the operations.

Below, is the SQL command used to setup the privileges to the user.

CREATE USER 'new_user'@'%' IDENTIFIED BY 'new_password';

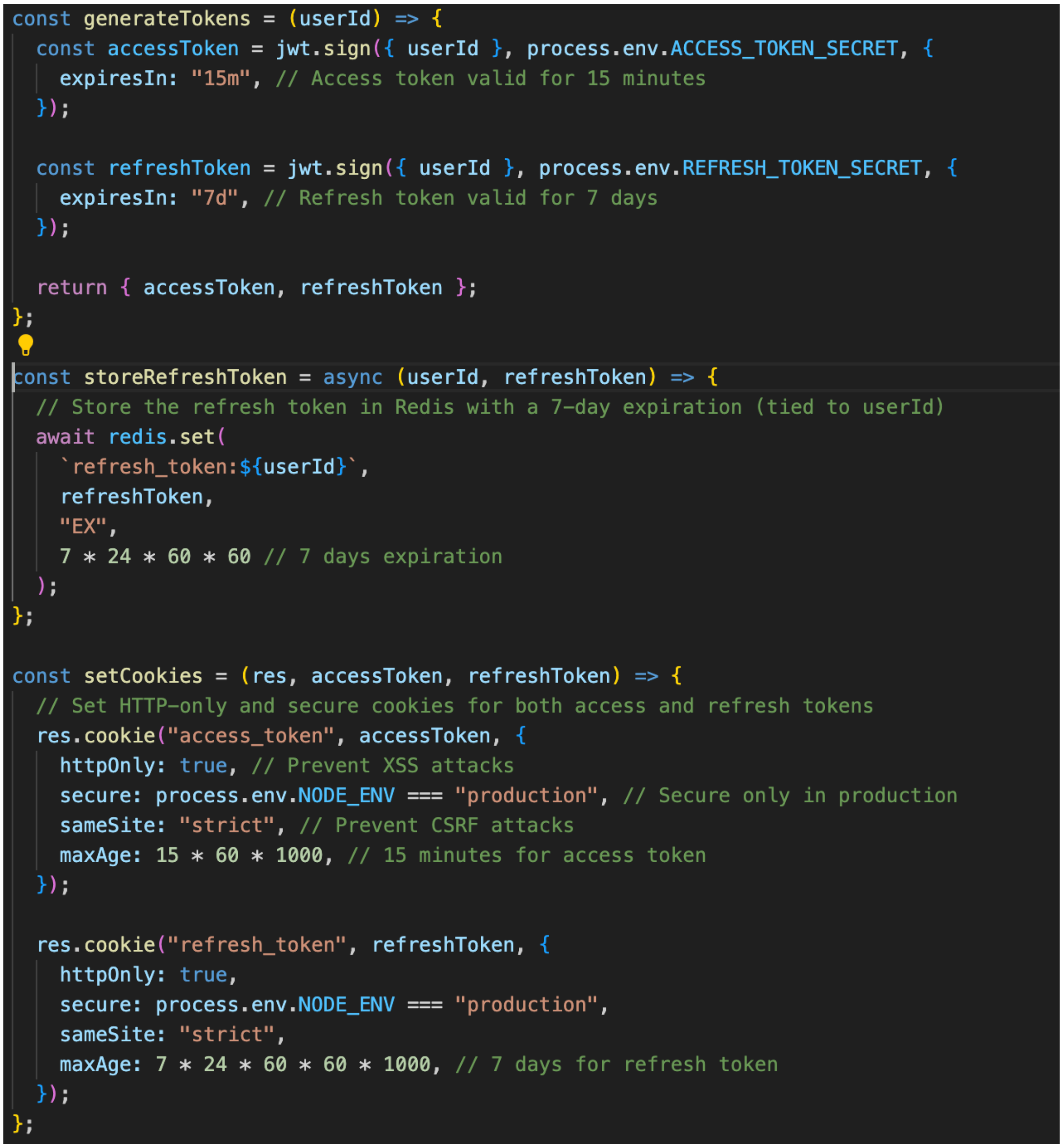
GRANT SELECT, INSERT, UPDATE, DELETE ON `ShopWave`.* TO 'new_user'@'%';

FLUSH PRIVILEGES;

REVOKE CREATE, ALTER, DROP ON `ShopWave`.* FROM 'new_user'@'%';
FLUSH PRIVILEGES;

4. Database Security at Application level

1. At the application level I have implemented the JWT (JSON Web Token). So, whenever a user logs in the application we will be assigning them a JWT token which is the stored in the REDIS where the refresh_token stays for a period of 15Minutes and the authentication_token stays for a period of 7days. This is how we secure the user at the application level. This code snippet shows the implementation of JWT and store it in cookies.

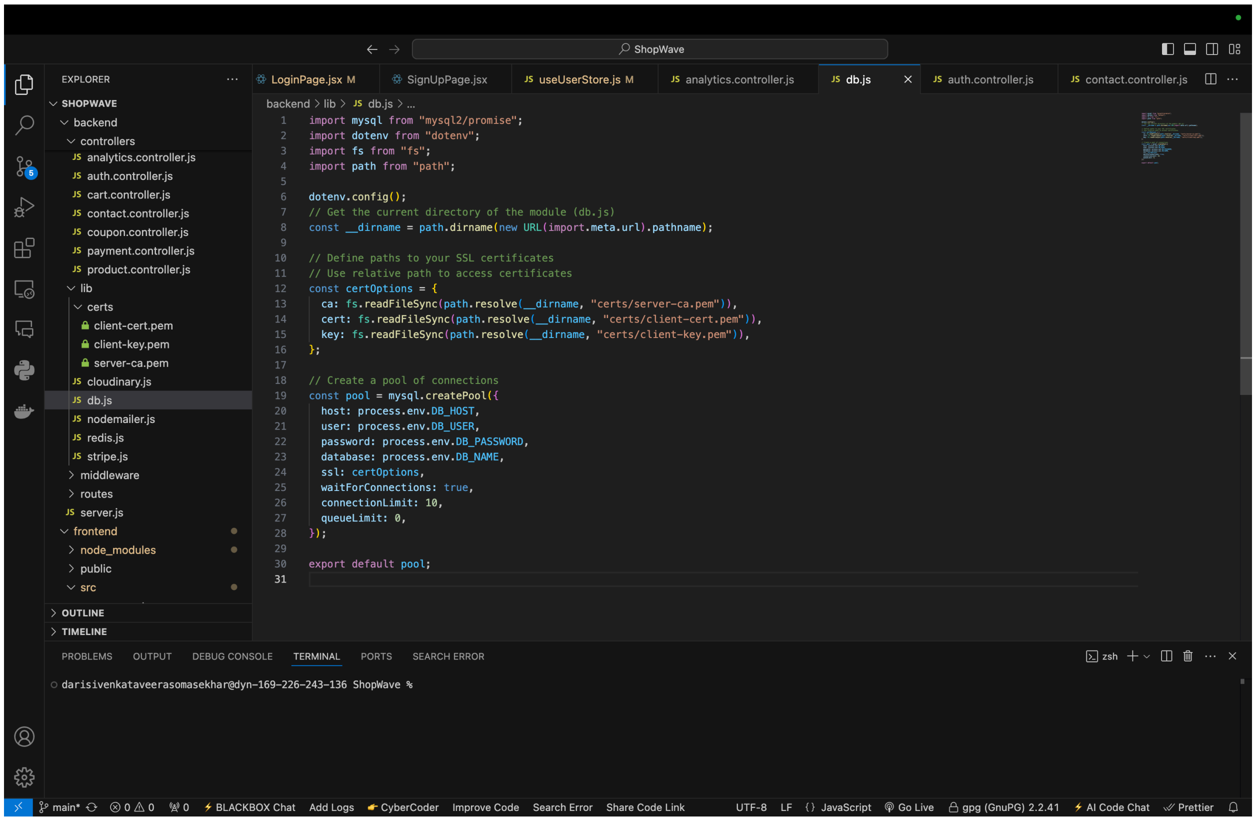


```
const generateTokens = (userId) => {
  const accessToken = jwt.sign({ userId }, process.env.ACCESS_TOKEN_SECRET, {
    expiresIn: "15m", // Access token valid for 15 minutes
  });

  const refreshToken = jwt.sign({ userId }, process.env.REFRESH_TOKEN_SECRET, {
    expiresIn: "7d", // Refresh token valid for 7 days
  });

  return { accessToken, refreshToken };
};

const storeRefreshToken = async (userId, refreshToken) => {
  // Store the refresh token in Redis with a 7-day expiration (tied to userId)
  await redis.set(
    `refresh_token:${userId}`,
    refreshToken,
    "EX",
    7 * 24 * 60 * 60 // 7 days expiration
  );
};

const setCookies = (res, accessToken, refreshToken) => {
  // Set HTTP-only and secure cookies for both access and refresh tokens
  res.cookie("access_token", accessToken, {
    httpOnly: true, // Prevent XSS attacks
    secure: process.env.NODE_ENV === "production", // Secure only in production
    sameSite: "strict", // Prevent CSRF attacks
    maxAge: 15 * 60 * 1000, // 15 minutes for access token
  });

  res.cookie("refresh_token", refreshToken, {
    httpOnly: true,
    secure: process.env.NODE_ENV === "production",
    sameSite: "strict",
    maxAge: 7 * 24 * 60 * 60 * 1000, // 7 days for refresh token
  });
};
```

2. Also, I had configured the SSL Certificates to communicate with the Database from the backend of the application. So, we need to provide client-cert.pem, client-key.pem, server-ca.pem. In this way we ensure the security at the DB level.

3. I have encrypted the user password in the application which satisfies the application-level security.

This screenshot shows the evidence of the configured SSL certificates.



```javascript
import mysql from "mysql2/promise";
import dotenv from "dotenv";
import fs from "fs";
import path from "path";

dotenv.config();
// Get the current directory of the module (db.js)
const __dirname = path.dirname(new URL(import.meta.url).pathname);

// Define paths to your SSL certificates
// Use relative path to access certificates
const certOptions = {
  ca: fs.readFileSync(path.resolve(__dirname, "certs/server-ca.pem")),
  cert: fs.readFileSync(path.resolve(__dirname, "certs/client-cert.pem")),
  key: fs.readFileSync(path.resolve(__dirname, "certs/client-key.pem")),
};

// Create a pool of connections
const pool = mysql.createPool({
  host: process.env.DB_HOST,
  user: process.env.DB_USER,
  password: process.env.DB_PASSWORD,
  database: process.env.DB_NAME,
  ssl: certOptions,
  waitForConnections: true,
  connectionLimit: 10,
  queueLimit: 0,
});

export default pool;
```

```
darisivenkataveerasomasekhar@dyn-169-226-243-136 ShopWave %
```