

# **UB Study Hall Reservation System**

## **Final Report**

The "UB Study Hall Reservation System" is a dynamic digital platform developed to revolutionize the traditional classroom reservation process within the University at Bridgeport(UB). This project aims to develop an automated and user-friendly interface. The system caters to both administrators and students, providing an efficient and streamlined approach to managing study hall reservations. Moreover, the system introduces additional features such as an online library, PC rentals, a community-driven chat forum, study hall mapping, job opportunity submissions, a store for purchasing various items, a donation platform, and a rewarding points system. Leveraging MySQL as the database management system and Python as the primary programming language, the system is hosted on the Google Cloud Platform (GCP) to ensure scalability, reliability, and accessibility to a broad user base.

**by**

**Yagnam Venisri - 1174310**  
**Lalith S Lankalapalli - 1215621**  
**Abhishek Dyawarishetti - 1211926**  
**Brandon Galvan - 1107863**

## **Step 1: Project Team**

1. Yagnam Venisri
2. Lalith S Lankalapalli
3. Abhishek Dyawarishetti
4. Brandon Galvan

## **Step 2:**

1. **Problem Statement:** Ub Study Hall Reservation System - The "UB Study Hall Reservation System" aims to provide an efficient and user-friendly platform for managing study room reservations within the University at Bridgeport (UB). This is an integrated and automated "UB Study Hall Reservation System" that not only streamlines the reservation process but also offers additional services such as an online library, PC rental, community forums, a store, and a donation platform. These functionalities aim to enrich the academic and social aspects of the study hall, providing students with a comprehensive and user-friendly platform for their various needs.

- a. User Types:
  - i. The system will cater to two main types of users: Admin and Normal users.
  - ii. Admin users will have full system management capabilities and access to reports.
  - iii. Normal users can reserve, modify, and cancel classroom reservations.
- b. Authentication and Registration:
  - i. Normal users will be able to register in the system, and subsequently, sign in to access the reservation features.
- c. Reservation Functionality:
  - i. Normal users can book, modify, and cancel classroom reservations based on availability.
- d. Feedback and Rating:
  - i. Normal users can provide feedback and ratings regarding the service and quality of their reservation experience.
- e. Information Visibility:
  - i. All users can view the current, past, and future status of classroom reservations to make informed decisions.
- f. Communication:
  - i. Normal users can communicate with Admin users by sending messages regarding any issues related to their reservations.
- g. Additional Functionalities:

- i. Online Library:
  - 1. Normal users can browse and access books online through the website.
- ii. Rent a PC:
  - 1. Normal users can rent a PC for a specific duration.
- iii. Chat Forum:
  - 1. Normal users can post questions in the forum and receive replies from the community.
- iv. Map:
  - 1. Normal users can see a picture where they can see the map/blueprint of the Study Hall.
- v. Careers:
  - 1. Normal users can submit their resumes for positions within the study hall.
- vi. Store:
  - 1. Normal users can access a small store within the study hall to purchase stationery items, coffee mugs, t-shirts, etc.
- vii. Points:
  - 1. Normal users can earn points every time they book a hall and use them to purchase items from the store.
- viii. Donate:
  - 1. Normal users can contribute by donating items or funds that help fellow students.
- ix. Forgot Password:
  - 1. Users can reset passwords using this feature.
- x. FAQ:
  - 1. Users can find answers to commonly asked questions.
- xi. Contact Us:
  - 1. Normal users can contact via email or office phone number during office hours.
- xii. About Us:
  - 1. Normal users can explore the inspiration behind the features of the study hall and gather more information about it.

### **Step 3: Queries**

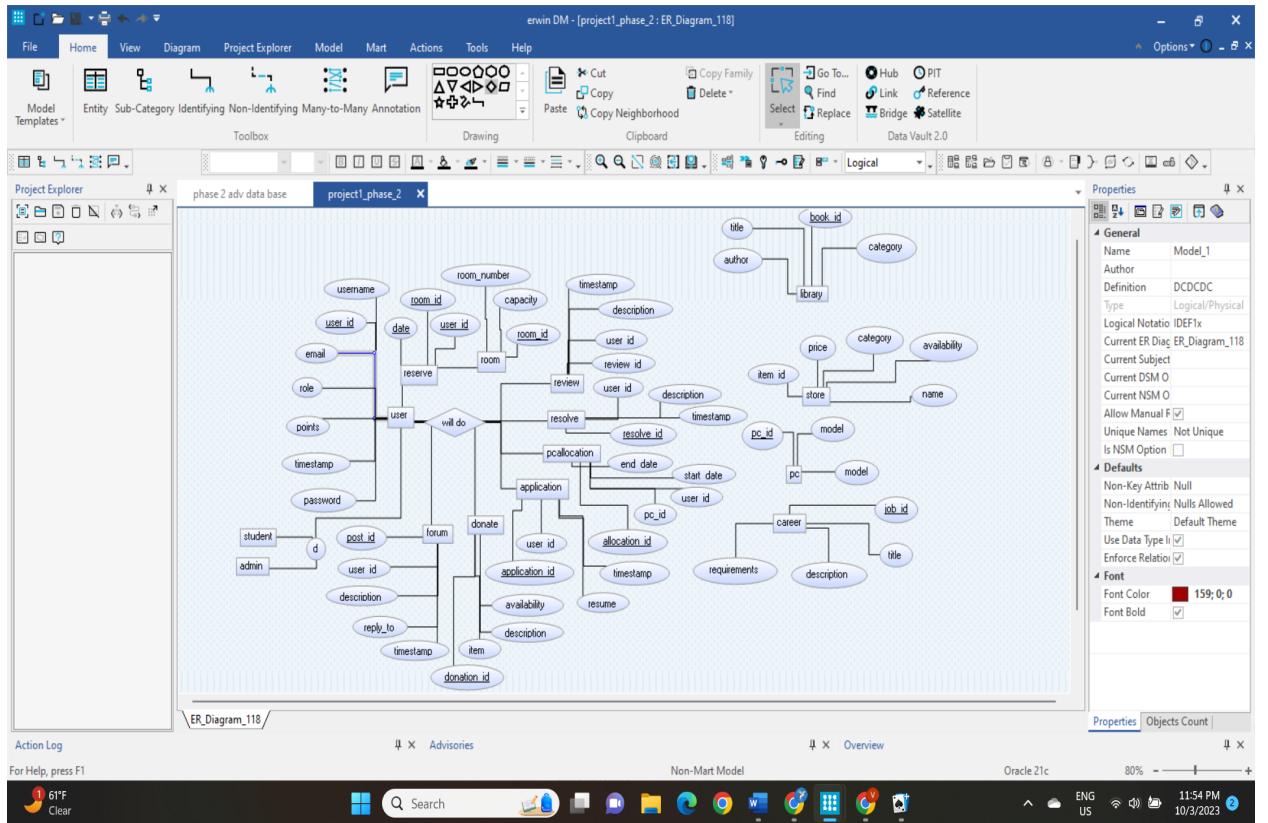
1. Retrieve a list of available rooms for a specific date.
2. Get all the available books in the online library.
3. Check the availability of PCs for rent during a specified duration.
4. Retrieve forum posts in a chronological order and their respective replies.

5. List out all available job positions within the study hall.
6. Retrieve a list of items available for purchase in the study hall store.
7. Retrieve the current points balance for a specific user.
8. Retrieve a list of donation items available for contribution.
9. Retrieve a list of past reservations for a specific user.
10. Retrieve a list of frequently asked questions and their answers.
11. Retrieve contact details (email, phone) for reaching out to the administration.
12. Retrieve detailed information about the study hall and its features.
13. Retrieve all the user and reservation information for the admin
14. Retrieve all the complaints for the admin to resolve.
15. Retrieve the room information such as how many people can accommodate, ac, fan etc.,
16. Retrieve all the allocated and unallocated pcs for admin.
17. Retrieve all the previous reservations for a user.

## **Step 4: Database Draft**

1. User: user\_id(primary key), username, password, email, role, points, timestamp.
2. Room: room\_id(primary key), room\_number, capacity, location.
3. Reserve: room\_id(partial primary key), user\_id(partial primary key, foreign key), date(partial primary key), timestamp.
4. Reviews: review\_id(primary key), username, description, rating, timestamp.
5. Resolve: resolve\_id(primary key), username, description, timestamp.
6. Library: book\_id(primary key), title, author, category.
7. PC: pc\_id(primary key), model, availability.
8. Forum: post\_id (primary key), user\_id (foreign key), description, replies, timestamp.
9. Store: item\_id (primary key), name, price, category, availability.
10. Career: job\_id (primary key), title, description, requirements.
11. Donate: donation\_id (primary key), item\_name, description, availability.
12. Application: application\_id(primary key), user\_id, resume, timestamp.
13. PCAllocation: allocation\_id(primary key), pc\_id(foreign key), user\_id(foreign key), duration, start\_timestamp, end\_timestamp.

## Step 5: ER Model



## Step 6: Relational Database Schema

### User

user_id	username	email	password	role	points	timestamp
---------	----------	-------	----------	------	--------	-----------

### Room

room_id	room_number	capacity	location
---------	-------------	----------	----------

### Reserve

room_id	user_id	date	timestamp
---------	---------	------	-----------

### Review

review_id	user_id	description	rating	timestamp
-----------	---------	-------------	--------	-----------

## Resolve

<u>resolve_id</u>	user_id	description	timestamp
-------------------	---------	-------------	-----------

## Library

<u>book_id</u>	title	author	category
----------------	-------	--------	----------

## PC

<u>pc_id</u>	model	availability
--------------	-------	--------------

## Forum

<u>post_id</u>	user_id	description	reply_to	timestamp
----------------	---------	-------------	----------	-----------

## Store

<u>item_id</u>	name	price	category	availability
----------------	------	-------	----------	--------------

## Career

<u>job_id</u>	title	description	requirements
---------------	-------	-------------	--------------

## Donate

<u>donation_id</u>	item	description	availability
--------------------	------	-------------	--------------

## Application

<u>application_id</u>	user_id	resume	timestamp
-----------------------	---------	--------	-----------

## PCAllocation

<u>allocation_id</u>	pc_id	user_id	start_date	end_date	timestamp
----------------------	-------	---------	------------	----------	-----------

## Step 7: Relational Algebra Expressions

1. Retrieve a list of available rooms for a specific date.
  - a.  $\text{AvailableRooms}(\text{date}) = \pi \text{ room\_id} (\sigma \text{ date} = \text{'specific\_date'} (\text{Reserve}))$
2. Get all the available books in the online library.
  - a.  $\text{AvailableBooks} = \pi \text{ title} (\text{Library})$

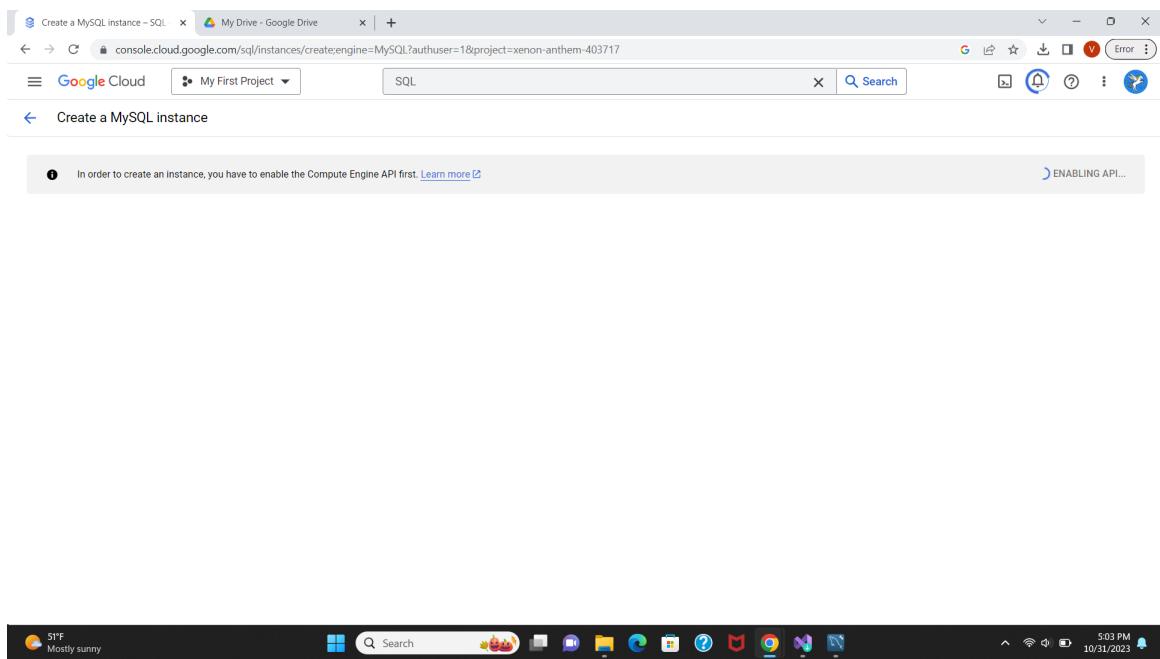
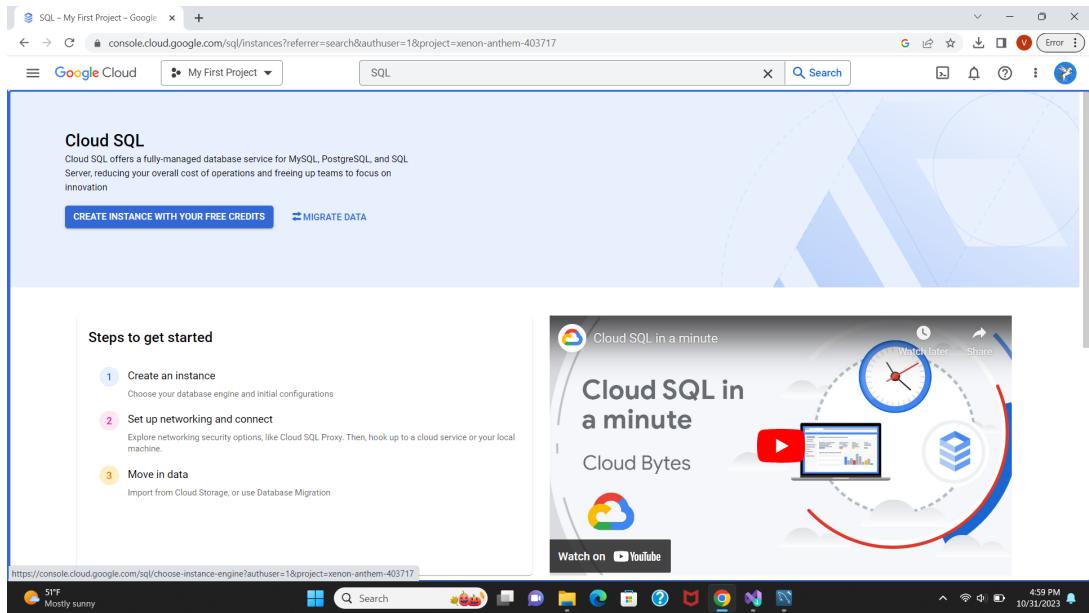
3. Check the availability of PCs for rent during a specified duration.
  - a.  $\text{AvailablePCs}(\text{startDate}, \text{endDate}) = \pi \text{ model, quantity } (\sigma \text{ start\_date} \geq \text{'specified\_start\_date'} \text{ AND } \text{end\_date} \leq \text{'specified\_end\_date'}) \text{ (PC)}$
4. Retrieve forum posts in a chronological order and their respective replies.
  - a.  $\text{AllForumPostsAndReplies} = \pi \text{ username, description, timestamp, replies } (\text{ForumPost}) \bowtie (\text{ForumPost.post\_id} = \text{ForumPost.reply\_to}) \rho (\text{post\_id, description, timestamp, replies})(\text{ForumPost})$
5. List out all available job positions within the study hall.
  - a.  $\text{AvailableJobPositions} = \pi \text{ position, description, requirements } (\text{Career})$
6. Retrieve a list of items available for purchase in the study hall store.
  - a.  $\text{AvailableStoreItems} = \pi \text{ name, price, category } (\text{Store})$
7. Retrieve the current points balance for a specific user.
  - a.  $\text{CurrentPoints}(\text{user}) = \pi \text{ points } (\sigma \text{ user\_id} = \text{user}) (\text{User})$
8. Retrieve a list of donation items available for contribution.
  - a.  $\text{DonatedItems} = \pi \text{ item\_name, description } (\text{Donate})$
9. Retrieve a list of past reservations for a specific user.
  - a.  $\text{PastReservations}(\text{user}) = \pi \text{ room\_number, date } (\sigma \text{ user\_id} = \text{user}) (\text{Reserve})$
10. Retrieve contact details (email, phone) for reaching out to the administration.
  - a.  $\text{AdminContactDetails} = \pi \text{ email, phone } (\text{Admin})$
11. Retrieve detailed information about the study hall and its features.
  - a.  $\text{StudyHallInformation} = \pi * (\text{StudyHall})$
12. Retrieve all the user and reservation information for the admin
  - a.  $\text{Reservations} = \pi * (\text{Reserve})$
13. Retrieve all the complaints for the admin to resolve.
  - a.  $\text{Complaints} = \pi * (\text{Resolve})$
14. Retrieve the room information such as how many people can accommodate, ac, fan etc.,
  - a.  $\text{RoomInfo} = \pi * (\text{Room})$
15. Retrieve all the allocated and unallocated pcs for admin.
  - a.  $\text{PCInfo} = \pi * (\text{PC})$
16. Retrieve all the upcoming reservations for a user.
  - a.  $\text{Upcoming}(\text{user}, \text{date}) = \pi \text{ room\_number } (\sigma \text{ user\_id} = \text{user} \text{ AND } \text{timestamp} \geq \text{date}) (\text{Reserve})$

## Step 8: Creating Database Server

To establish the Database Server, we followed the below steps

1. **Project Creation:** We initiated a GCP project. This project served as the central hub for our database and web application components.

**2. Billing Account:** We assigned a billing account to the project based on a coupon provided in the exercise lab. This is essential to manage the costs associated with running resources on GCP.

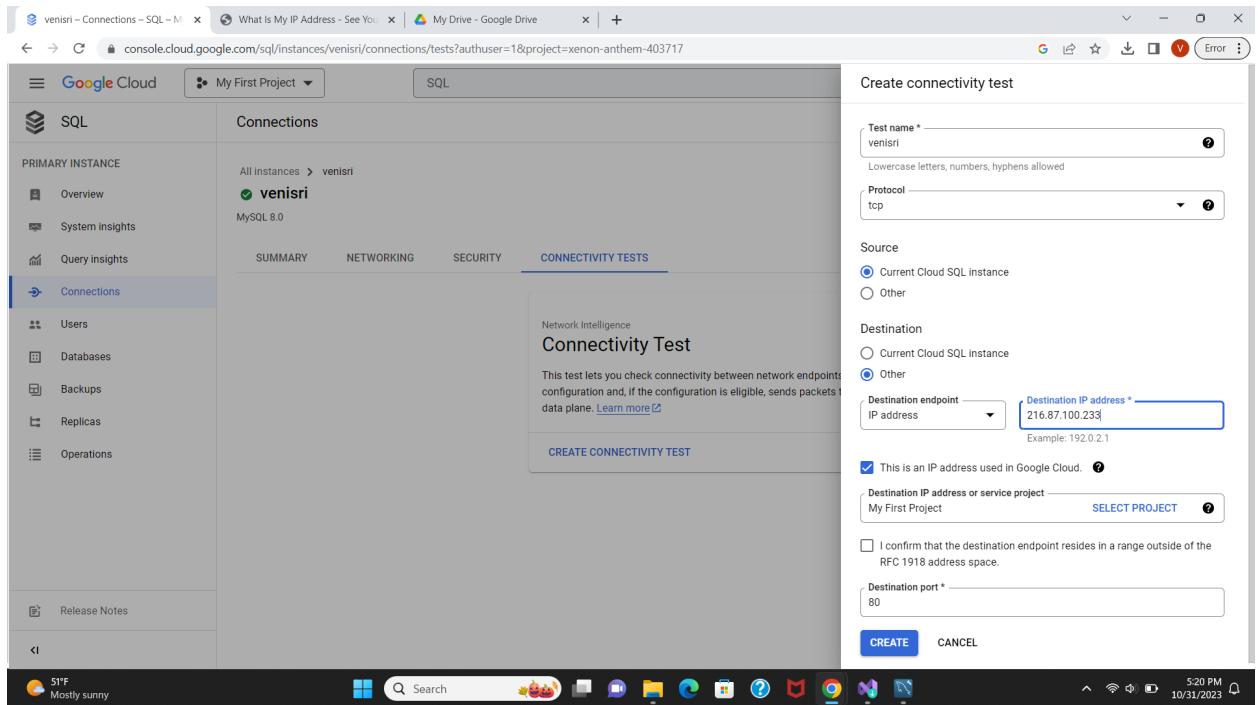


**3. RDBMS Selection and Installation:** We selected MySQL as our relational database management system (RDBMS) and installed it on a Google Compute Engine

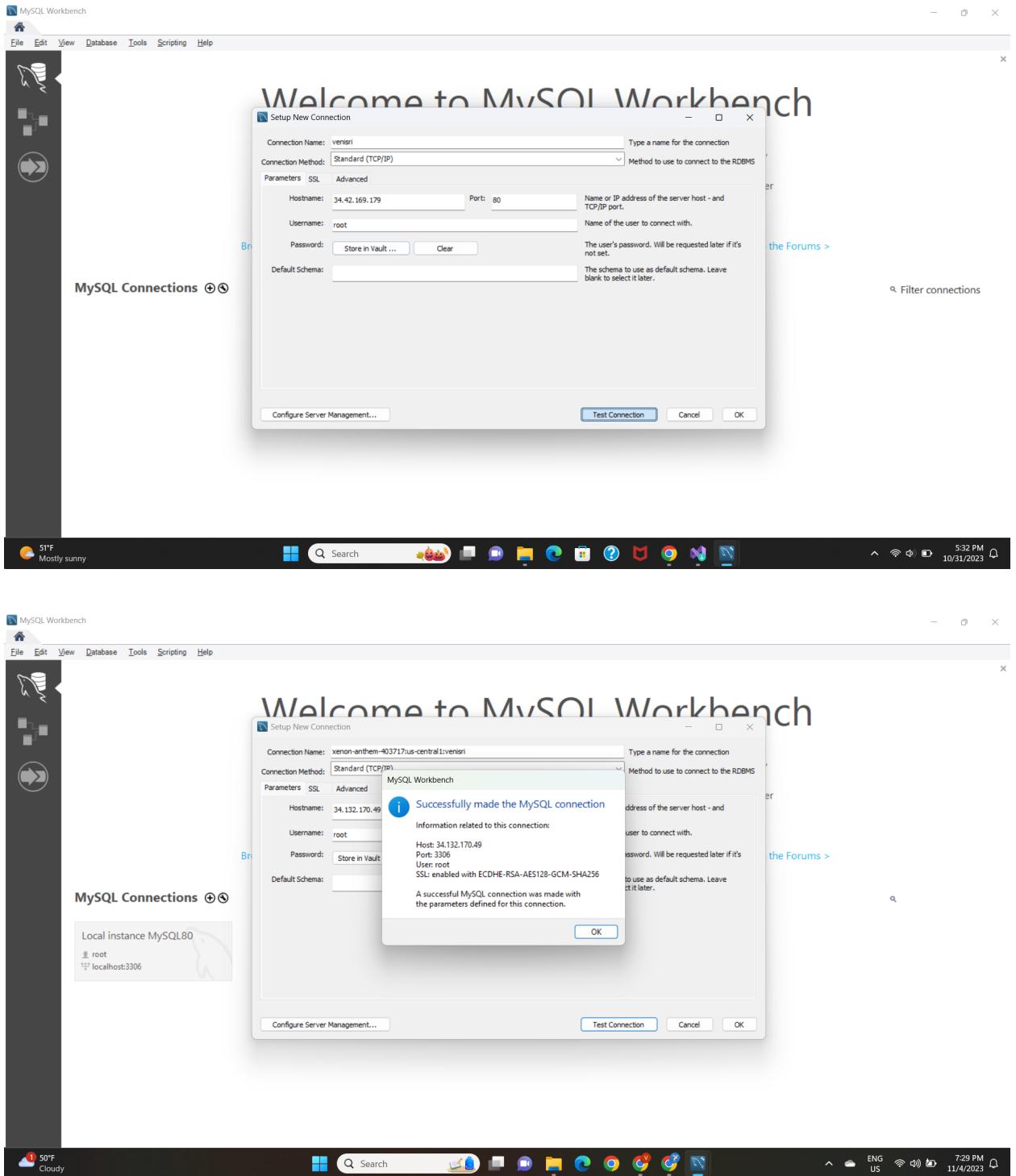
instance. This choice was made due to MySQL's reliability, scalability, and compatibility with various web frameworks.

The screenshot shows the 'Create a MySQL instance' page in the Google Cloud Platform. The 'Instance info' section includes fields for Instance ID (venisri), Password, and a note about root password requirements. The 'Pricing estimate' section shows a cost of \$2.30 per hour. The 'Summary' table provides details like Cloud SQL Edition (Enterprise Plus), Region (us-central1 (Iowa)), DB Version (MySQL 8.0), and Memory (64 GB). The 'Choose a Cloud SQL edition' section allows selecting Enterprise Plus or Enterprise, with Enterprise Plus selected. The status bar at the bottom shows it's 5:05 PM on 10/31/2023.

The screenshot shows the 'Connections' page for the MySQL instance 'venisri'. The left sidebar has 'SQL' selected. The 'Networking' tab is active. Under 'Instance IP assignment', 'Public IP' is selected. The 'Authorized networks' section shows a new network being created with the name 'venisri' and CIDR range '216.87.100.233'. The status bar at the bottom shows it's 5:13 PM on 10/31/2023.



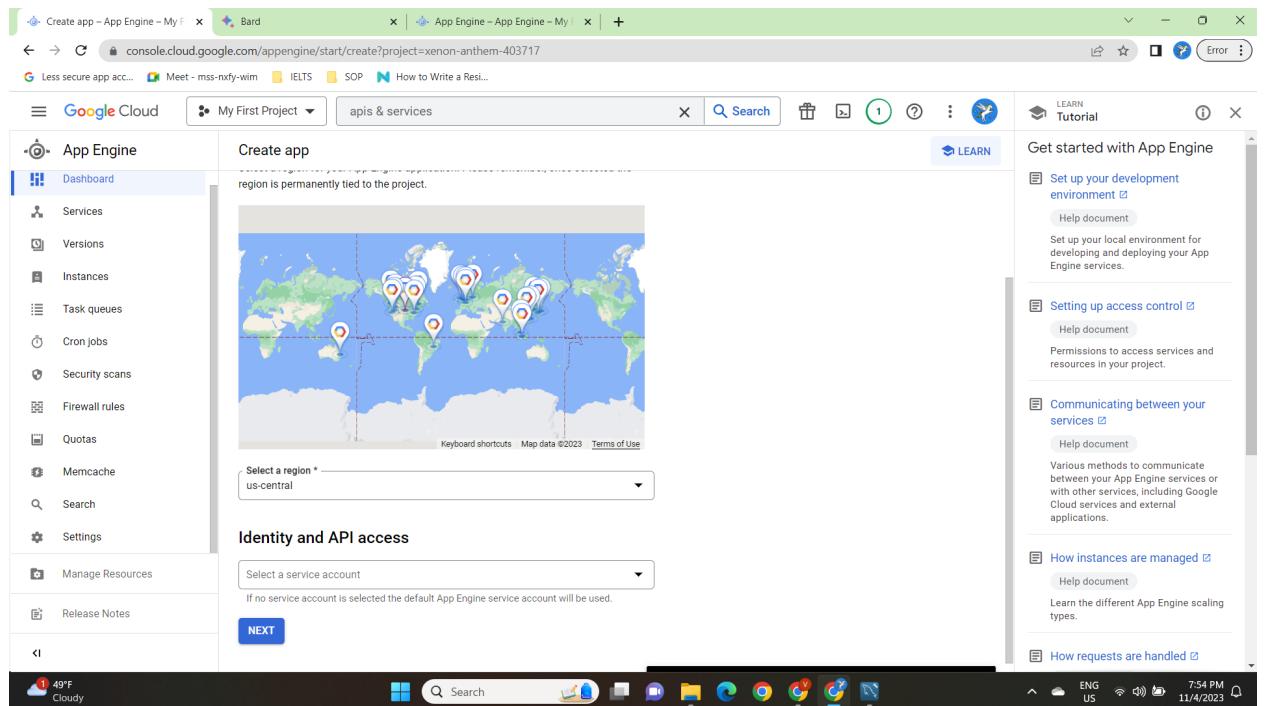
4. **Database Connectivity using MySQL Workbench:** In addition to setting up the database server on Google Compute Engine, we leveraged MySQL Workbench, a powerful database management tool, to facilitate the development and management of our MySQL database instance. This step enhanced our ability to interact with the database for various purposes, including schema design, SQL query development, and database administration.
5. **Connection to the GCP-hosted MySQL Database Instance:** We established a connection between MySQL Workbench and our MySQL database instance on GCP. To achieve this, we used the following steps:
  - a. **Hostname Configuration:** We obtained the hostname or IP address associated with our MySQL database instance on Google Compute Engine within the GCP project.
  - b. **MySQL Workbench Configuration:** In MySQL Workbench, we configured a new database connection, providing the GCP-hosted MySQL database instance's hostname.
  - c. **Testing and Usage:** With the successful connection, we were able to utilize MySQL Workbench to perform tasks such as database schema design, executing SQL queries, and managing the database server directly from our local development environment.



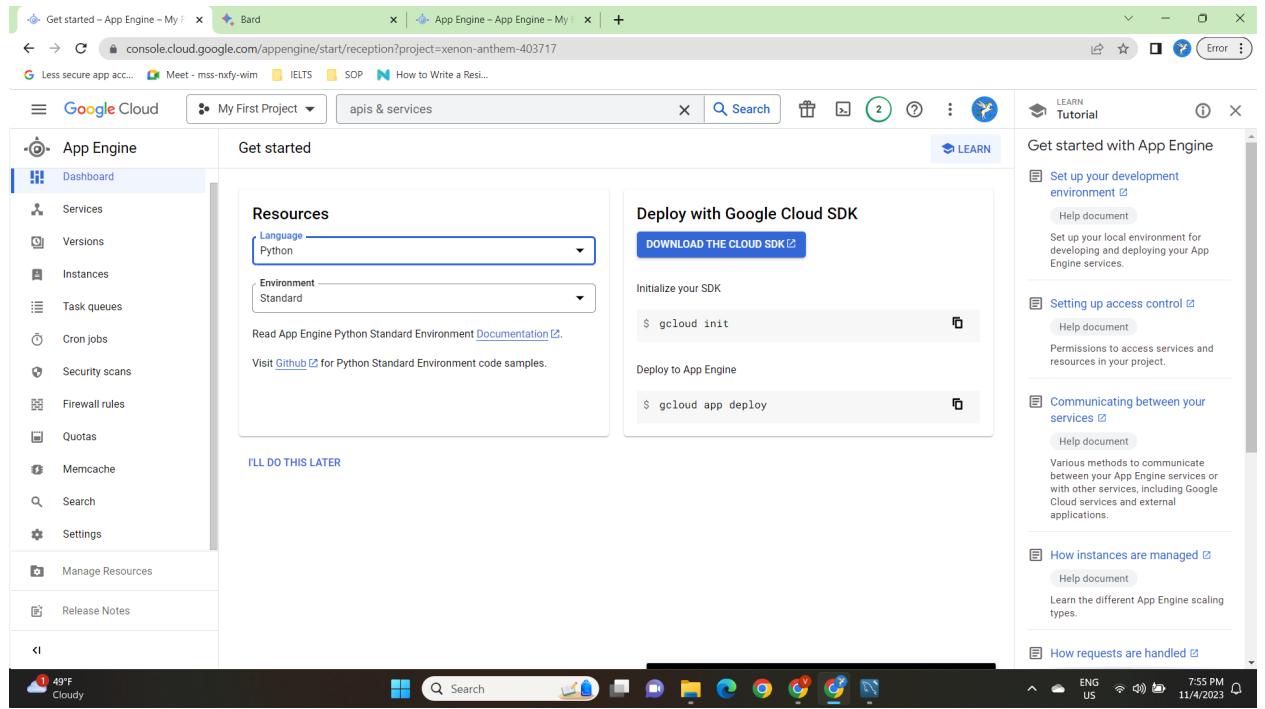
This integration between MySQL Workbench and our GCP-hosted MySQL database instance allowed us to streamline our development process, making it more efficient and enabling real-time database interaction for the development of the UB Study Hall Reservation System.

## Step 9: Creating a Web Application Server

- 1. Google App Engine Setup:** We created a Google App Engine instance to host our web application. Google App Engine provides a scalable and serverless platform for running web applications.



- 2. Framework Selection:** We selected the Django framework for our web application. Django is a popular and robust web framework that simplifies the development process and provides a wide range of tools and libraries for creating web applications.



3. **Application Deployment:** We are going to deploy our room reservation system using Django on Google App Engine. This allows us to create a dynamic and user-friendly web application for users to interact with the database.

## Step 9: Database Creation

- Created a database named reservation.
- Defined tables to model various entities: user, room, reserve, review, complaints, library, library\_history, pc, pc\_allocation, forum, store, career, donate, application.

## Step 10: DDL Script

- Produced a DDL script named dbDDL.sql that encompasses the entire database schema including tables with constraints, indexes, etc.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Schemas: reservation

SQL File 3: dbDML dbDDL dbSQL Limit to 1000 rows

```

1 • create database reservation;
2 • drop table students;
3 • create table user
4 • (
5     user_id INT PRIMARY KEY,
6     username VARCHAR(50) NOT NULL,
7     email VARCHAR(100) UNIQUE,
8     user_password VARCHAR(50) NOT NULL,
9     user_role VARCHAR(20),
10    points INT,
11    created_timestamp TIMESTAMP
12 );
13 • create table room
14 • (
15     room_number VARCHAR(20) PRIMARY KEY,
16     capacity INT,
17     location VARCHAR(100)
18 );
19 • create table reserve
20 • (
21     reserve_id INT PRIMARY KEY,
22     room_id INT REFERENCES Room(room_id).

```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Output:

#	Time	Action	Message	Duration / Fetch
1	22:06:09	create database reservation	Error Code: 1007. Can't create database 'reservation': database exists	0.047 sec
2	22:06:44	drop database reservation	0 row(s) affected	0.047 sec
3	22:07:19	create database reservation	1 row(s) affected	0.053 sec
4	22:07:19	create table user (user_id INT PRIMARY KEY, username VARCHAR(50) NOT NULL, email VARCHAR(100) UNIQUE, user_password VARCHAR(50) NOT NULL, user_role VARCHAR(20), points INT, created_timestamp TIMESTAMP)	Error Code: 1050. Table 'User' already exists	0.062 sec

Object Info Session

SQL script saved to 'C:\Users\14753\Desktop\Semester4\Adv Database\dbSQL.sql'

31% Clear

Search

11:32 PM 11/14/2023

## Step 11: DML Script

- Created a DML script named dbDML.sql to insert sample data into tables.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Schemas: reservation

SQL File 3: dbDML dbDDL dbSQL Limit to 1000 rows

```

1 • INSERT INTO user (user_id, username, email, user_password, user_role, points, created_timestamp)
2 VALUES
3     (1, 'JohnDoe', 'john@example.com', 'password123', 'admin', 100, CURRENT_TIMESTAMP),
4     (2, 'JaneSmith', 'jane@example.com', 'pass321', 'user', 50, CURRENT_TIMESTAMP),
5     (3, 'MikeJohnson', 'mike@example.com', 'mikepass', 'user', 75, CURRENT_TIMESTAMP),
6     (4, 'EmilyBrown', 'emily@example.com', 'emilypass', 'user', 80, CURRENT_TIMESTAMP),
7     (5, 'AlexClark', 'alex@example.com', 'alexpass', 'user', 60, CURRENT_TIMESTAMP),
8     (6, 'SophiaLee', 'sophia@example.com', 'sophiapass', 'user', 90, CURRENT_TIMESTAMP),
9     (7, 'WilliamTaylor', 'william@example.com', 'williampass', 'user', 70, CURRENT_TIMESTAMP);
10 • INSERT INTO room (room_number, capacity, location)
11 VALUES
12     ('101', 20, 'Floor 1'),
13     ('205', 15, 'Floor 2'),
14     ('310', 25, 'Floor 3'),
15     ('112', 18, 'Floor 1'),
16     ('410', 30, 'Floor 4'),
17     ('201', 12, 'Floor 2'),
18     ('301', 22, 'Floor 3');
19 • INSERT INTO reserve (reserve_id, room_id, user_id, reserve_date, reserve_timestamp)
20 VALUES
21     (1, 1, 3, '2023-11-15', CURRENT_TIMESTAMP),
22     (2, 4, 5, '2023-11-16', CURRENT_TIMESTAMP),
23     (3, 5, 6, '2023-11-17', CURRENT_TIMESTAMP),
24     (4, 6, 7, '2023-11-18', CURRENT_TIMESTAMP),
25     (5, 7, 8, '2023-11-19', CURRENT_TIMESTAMP);

```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Output:

#	Time	Action	Message	Duration / Fetch
9	22:11:54	INSERT INTO pc_pc_id.model,availability VALUES (1,'Dell Inspiron',true), (2,'HP Pavilion',false), (3,'Acer Aspire',true), (4,'Lenovo IdeaPad',true)	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0	0.052 sec
10	22:12:11	INSERT INTO pc_allocation (allocation_id,pc_id,user_id,start_date,end_date,timestamp,status) VALUES (1,1,1,'2023-11-15','2023-11-15',150000,0)	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0	0.063 sec
11	22:12:23	INSERT INTO forum (post_id,user_id,description,reply_to,timestamp) VALUES (1,2,'New to the forum. I am a beginner.',null,'2023-11-15 22:12:23')	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0	0.063 sec
12	22:13:33	INSERT INTO store (item_id,item_name,price,category,availability) VALUES (1,'Headphones',49.99,true)	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0	0.047 sec

Object Info Session

SQL script saved to 'C:\Users\14753\Desktop\Semester4\Adv Database\dbSQL.sql'

31% Clear

Search

11:33 PM 11/14/2023

## Step 12: Drop Script

- Formulated a drop script named dbDROP.sql to remove all database objects, facilitating a clean database state.

The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The main window has tabs for SQL File 3\*, dbDML, dbDDL, and dbSQL. The dbSQL tab is active, displaying the following SQL code:

```
FROM review RA
GROUP BY R.user_id
HAVING AVG(RA.rating) > 4
ORDER BY avg_rating DESC;

-- Drop all tables
DROP TABLE IF EXISTS application;
DROP TABLE IF EXISTS donate;
DROP TABLE IF EXISTS career;
DROP TABLE IF EXISTS store;
DROP TABLE IF EXISTS forum;
DROP TABLE IF EXISTS pc_allocation;
DROP TABLE IF EXISTS pc;
DROP TABLE IF EXISTS library_history;
DROP TABLE IF EXISTS library;
DROP TABLE IF EXISTS complaints;
DROP TABLE IF EXISTS review;
DROP TABLE IF EXISTS reserve;
DROP TABLE IF EXISTS room;
DROP TABLE IF EXISTS user;
```

The left sidebar shows the schema structure under the 'Schemas' tab, specifically the 'reserve' table with columns: reserve\_id, room\_id, user\_id, reserve\_date, and reserve\_time. The bottom left pane shows the definition for the 'room\_id' column: 'room\_id int'. The bottom right pane displays the 'Output' section with the following log entries:

#	Time	Action	Message	Duration / Fetch
1	22:06:09	create database reservation	Error Code: 1007. Can't create database 'reservation', database exists	0.047 sec
2	22:06:44	drop database reservation	0 row(s) affected	0.047 sec
3	22:07:19	create database reservation	1 row(s) affected	0.063 sec
4	22:07:19	create table user (user_id INT PRIMARY KEY, username VARCHAR(50) NOT NULL, email VARCHAR(100) U... Error Code: 1050. Table 'User' already exists	Error Code: 1050. Table 'User' already exists	0.062 sec

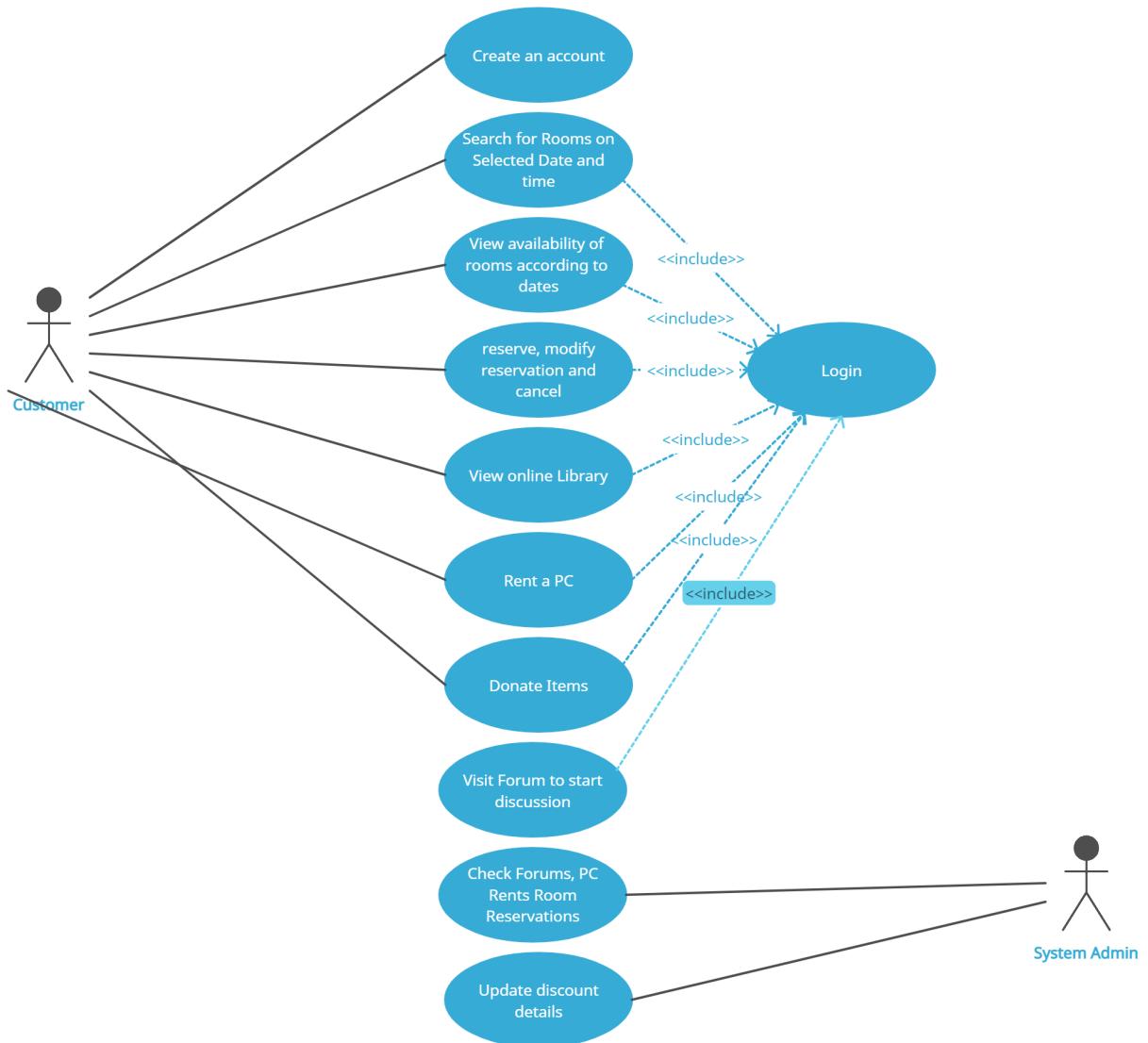
## Step 13: SQL Script

- Generated an SQL script named dbSQL.sql with complex queries satisfying specific criteria:
- Joins involving multiple relations.
- Aggregate queries using GROUP BY, HAVING, and ORDER BY clauses.
- Utilized nested subqueries.

The screenshot shows the MySQL Workbench interface with the 'dbSQL' tab selected. The central pane displays a multi-line SQL script with several numbered comments (12, 18, 20, 28) explaining different parts of the query. The left pane shows the database schema with tables like 'pc\_allocation', 'review', and 'room'. The bottom pane shows the execution history with four actions: creating the 'reservation' database, dropping it, creating the 'user' table, and finally creating the 'user' table again due to a primary key constraint error. The status bar at the bottom right indicates the date and time as 11/14/2023 11:36 PM.

```
12 • SELECT R.user_id, AVG(RA.rating) AS avg_rating
13   FROM review RA
14   GROUP BY R.user_id
15   HAVING AVG(RA.rating) > 4
16   ORDER BY avg_rating DESC;
17
18   -- Query 3: Aggregate Query with nested subquery
19   -- List users who have allocated a PC and their total number of allocations
20 • SELECT U.username, COUNT(PCA.pc_id) AS total_allocations
21   FROM user U
22   JOIN pc_allocation PCA ON U.user_id = PCA.user_id
23   WHERE PCA.pc_id IN (
24       SELECT pc_id FROM pc WHERE availability = true
25   )
26   GROUP BY U.username;
27
28   -- Query 4: Join Query involving multiple tables
29   -- Retrieve the username, room number, and item name for each item in the store reserved by a user
30 • SELECT U.username, R.room_number, S.item_name
31   FROM reserve RS
32   JOIN user U ON RS.user_id = U.user_id
33   JOIN room R ON RS.room_id = R.room_id
```

## System configuration of the project:



## Step 14: Application Development

In step 14, we concentrated on building a user-friendly front-end application that interfaces seamlessly with the backend database management system (DBMS) developed using MySQL. The application aimed to simplify the process of reserving study hall rooms for students.

### **Technology and Tools Utilized:**

1. Backend Language: Python with Django Framework
2. Database Management System: MySQL Workbench
3. Frontend Interface: Developed using Django templates for HTML rendering
4. Integration Tools: Google Cloud Platform (GCP) for hosting the database

### **Functionalities Implemented:**

1. User Authentication: Users can register, log in, and authenticate credentials securely.
2. User Registration: New user registration involves capturing essential information such as username, email, and password.
3. Reservation Management: Students can reserve study hall rooms for a specified time slot.
4. Points System: Upon successful reservation, users are awarded points, encouraging consistent use and engagement.
5. Feedback Submission: Allows users to submit messages along with their usernames and email addresses for improved communication and issue resolution.
6. Administrative Review: Enables administrators to access submitted feedback, review messages, and address user concerns or suggestions promptly.
7. Administrative Reservation Monitoring: Provides a comprehensive list of reservations, detailing room bookings, timestamps, and user information accessible to authorized administrators.
8. Reservation Modification: Enabled users to modify their reservations before the scheduled time through an intuitive user interface. Users have the flexibility to adjust reservation details like time, date, or room number based on availability and their needs.
9. Cancellation Feature: Implemented a cancellation option, allowing users to revoke their reservations within a specified window before the scheduled reservation time. Ensures users have the autonomy to manage their bookings, promoting a user-friendly experience.
10. User-Friendly Navigation: Integrated a navigation bar across all pages to enhance user experience and enable easy traversal between different sections of the application. Consistently accessible navigation across the interface for effortless movement between functionalities and pages.

### **Database Connectivity:**

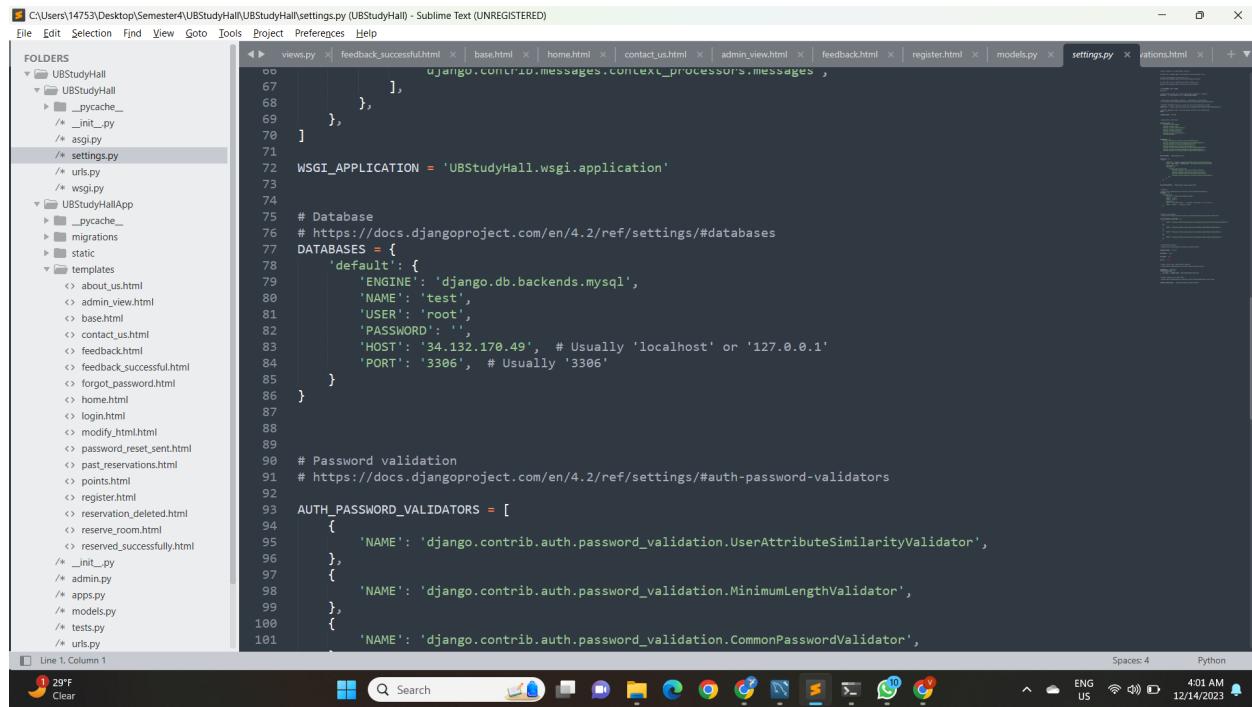
1. MySQL Workbench: Utilized for designing, creating, and managing the database schema.
2. Google Cloud Platform (GCP): The backend Django application was connected to the MySQL database hosted on GCP, ensuring secure and scalable access.

## User Interface:

1. Graphical User Interface (GUI): Incorporated a simple, intuitive interface for users to navigate through various functionalities.
2. Menu-Driven Input: Designed a menu-driven approach for users to interact with the database without needing SQL queries, enhancing usability.

## Results:

### Database Connectivity:

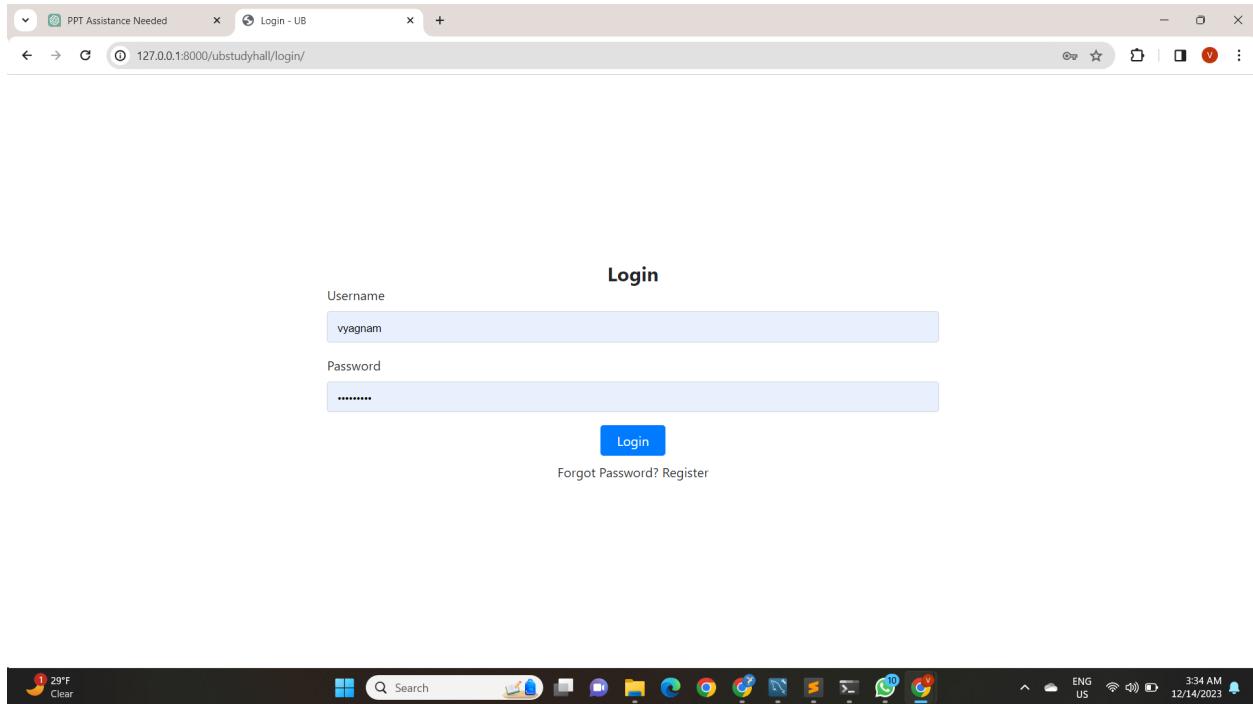


The screenshot shows the Sublime Text editor with the settings.py file of a Django project open. The file contains configuration for the application, database, and password validation. The code is as follows:

```
File Edit Selection Find View Goto Tools Project Preferences Help
C:\Users\14753\Desktop\Semester4\UBStudyHall\UBStudyHall\settings.py - Sublime Text (UNREGISTERED)
FOLDERS
  UBStudyHall
    UBStudyHall
      __pycache__
      __init__.py
      asgi.py
      settings.py
      urls.py
      wsgi.py
    UBStudyHallApp
      __pycache__
      migrations
      static
      templates
        about_us.html
        admin_view.html
        base.html
        contact_us.html
        feedback.html
        feedback_successful.html
        forgot_password.html
        home.html
        login.html
        modify.html.html
        password_reset_sent.html
        past_reservations.html
        points.html
        register.html
        reservation_deleted.html
        reserve_room.html
        reserved_successfully.html
      __init__.py
      admin.py
      apps.py
      models.py
      tests.py
      urls.py
views.py  feedback_successful.html  base.html  home.html  contact_us.html  admin_view.html  feedback.html  register.html  models.py  settings.py  rations.html
66         ],
67         ],
68     },
69   ],
70 ]
71
72 WSGI_APPLICATION = 'UBStudyHall.wsgi.application'
73
74
75 # Database
76 # https://docs.djangoproject.com/en/4.2/ref/settings/#databases
77 DATABASES = {
78     'default': {
79         'ENGINE': 'django.db.backends.mysql',
80         'NAME': 'test',
81         'USER': 'root',
82         'PASSWORD': '',
83         'HOST': '34.132.170.49', # Usually 'localhost' or '127.0.0.1'
84         'PORT': '3306', # Usually '3306'
85     }
86 }
87
88
89 # Password validation
90 # https://docs.djangoproject.com/en/4.2/ref/settings/#auth-password-validators
91 AUTH_PASSWORD_VALIDATORS = [
92     {
93         'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
94     },
95     {
96         'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
97     },
98     {
99         'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
100    }
101 ]
```

The status bar at the bottom indicates "Spaces: 4" and "Python". The taskbar at the bottom shows icons for various applications including Microsoft Edge, Google Chrome, and File Explorer.

## Page 1: Login Page



Users can log in using their credentials if they are already registered. The login form prompts for a username/email and password to access their accounts.

New users who don't have accounts can easily register by clicking on the registration link/button. This process involves providing essential details such as username, email, and password for creating a new account.

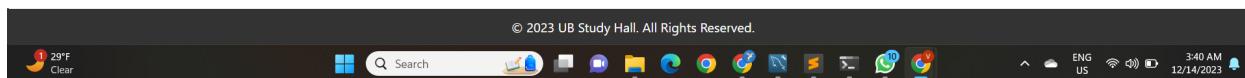
For users encountering login issues due to forgotten passwords, this feature provides a streamlined process to reset their password.

## Page 2: Home Page - Resource Navigation

The Home Page serves as a central hub, offering users an intuitive and comprehensive navigation system to access various resources and functionalities within the application. It prominently showcases essential options that users can explore for managing their study hall experience.

The screenshot shows the homepage of the UB Study Hall Reservation system. At the top, there are tabs for 'Adv Database Project - Go', 'UB Study Hall Reservation', 'Adv Database Project - Go', 'UB Study Hall Reservation', 'UB Study Hall Reservation', and 'UB Study Hall - UB'. The URL in the address bar is 127.0.0.1:8000/ubstudyhall/home/. The page has a dark header with 'Home' on the left and 'Account' with a dropdown arrow on the right. Below the header, there's a 'About Us' section with text about the platform's purpose and services. There are six service cards arranged in two rows of three:

- Study Room Reservations**: Reserve study rooms hassle-free for group or individual study sessions.
- My Reservations**: Check your upcoming and past reservations.
- Points System**: Earn points for booking halls and use them to purchase items from the store.
- About Us**: Explore the inspiration behind the study hall features.
- Forgot Password**: Reset passwords using this feature.
- Contact Us**: Contact via email or office phone number during office hours.



## Page 3: Room Reservation - Time and Date Selection

The Room Reservation page empowers users to book study hall rooms based on their preferred date and time slots. It serves as a platform to explore room availability and make reservations according to individual study schedules.

The screenshot shows the 'Room Availability' page. The URL in the address bar is 127.0.0.1:8000/ubstudyhall/reserve/. The page has a dark header with 'Home', 'Services', and 'Account' with a dropdown arrow. Below the header, there's a form for selecting a date and time:

Select Date:

Select Time:

Below the form, there's a section titled 'Available Rooms' with a table:

Room Number	Action
Room 101	<input type="button" value="Reserve"/>
Room 102	<input type="button" value="Reserve"/>
Room 103	<input type="button" value="Reserve"/>
Room 201	<input type="button" value="Reserve"/>
Room 202	<input type="button" value="Reserve"/>

## **Page 4: Upcoming and Past Reservations - Modification and Cancellation**

Page 4 serves as a centralized hub for users to manage their study hall reservations, both upcoming and past ones. It empowers users to view, modify, or cancel their upcoming reservations, providing a convenient platform for reservation management.

**Upcoming Reservations**

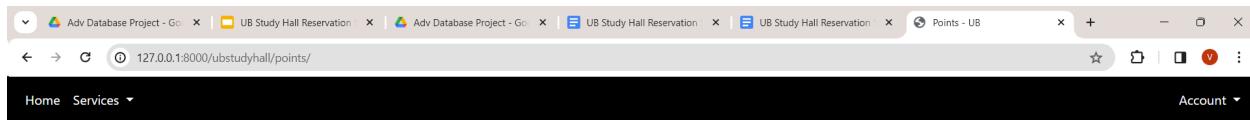
Date	Time	Room Number	Actions
2023-12-14	5:00am - 6:00am	101	<a href="#">Modify</a> <a href="#">Cancel</a>

**Past Reservations**

You have no past reservations

## **Page 6: Earning Points for Reservations**

Page 6 is dedicated to the points system integrated into the study hall reservation platform. It allows users to earn points upon successful completion of room reservations, encouraging consistent engagement and rewarding users for their frequent usage.



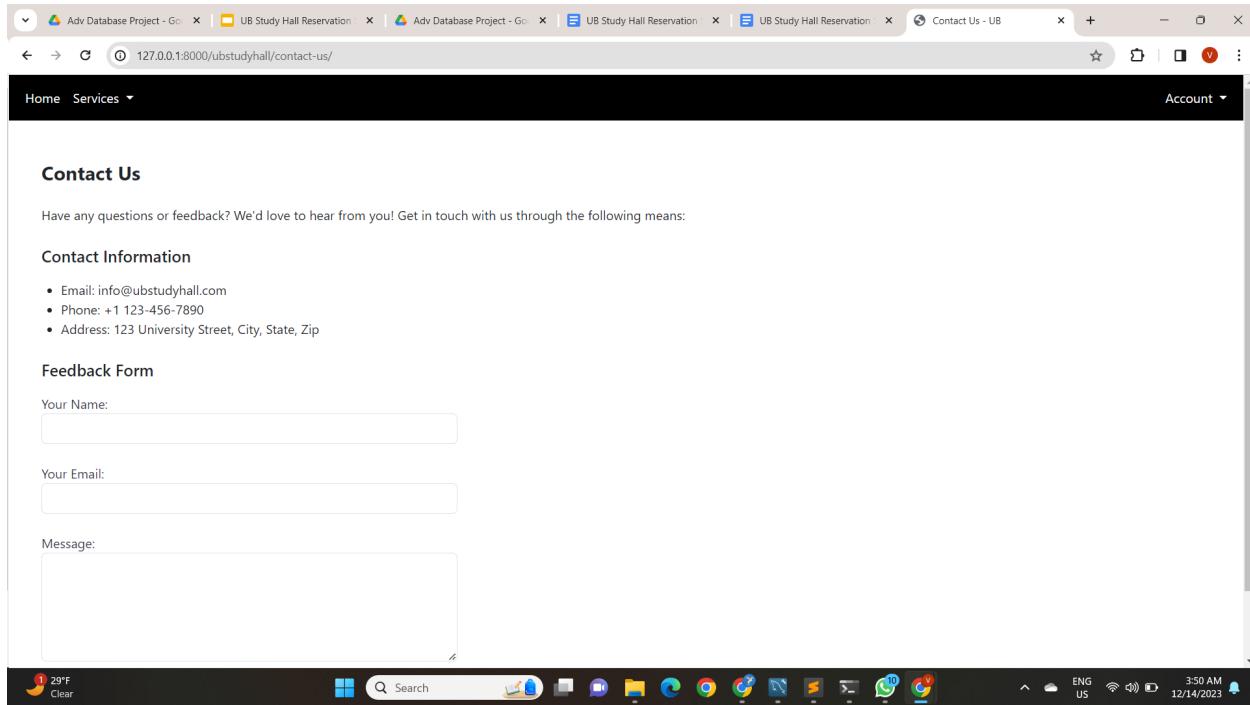
Hey student,

Your total points: 10



## Page 7: Contact Us

The "Contact Us" page serves as a channel for users to reach out to the administration or support team for any inquiries, concerns, or assistance related to the study hall reservation system.



## Page 8: Feedback Form

The "Feedback Form" is a dedicated section allowing users to provide feedback, suggestions, or report issues encountered while using the study hall reservation system.

The screenshot shows a web browser window with multiple tabs open. The active tab is titled 'Contact Us - UB' and displays a form for providing feedback. The page has a dark header with 'Home' and 'Services' links. Below the header, a message encourages users to contact them. The 'Contact Information' section lists email, phone number, and address. The 'Feedback Form' section contains fields for 'Your Name', 'Your Email', and a large 'Message' area, followed by a 'Send Message' button. The browser's status bar at the bottom shows system icons and the date/time.

## Navigation Bar:

The "Navigation Bar" is a persistent component placed across all pages of the study hall reservation system. It acts as a quick-access menu, allowing users to effortlessly transition between different sections and functionalities of the application.

127.0.0.1:8000/ubstudyhall/contact-us/#

29°F Clear

Search

3:54 AM 12/14/2023

## Admin View: All Reservations

The "Admin View" provides an overview of all reservations made within the study hall reservation system. It's an exclusive feature accessible only to authorized administrators, allowing them to monitor, manage, and analyze reservation activities comprehensively.

All Reservations    Feedbacks

Account

**Study Hall Reservations**

Username	Date	Time	Room Number
vyagnam	2023-11-25	5:00am - 6:00am	101
vyagnam	2023-11-25	6:00am - 7:00am	102
vyagnam	2023-11-26	7:00am - 8:00am	103
vyagnam	2023-11-26	8:00am - 9:00am	104
vyagnam	2023-11-27	9:00am - 10:00am	105
vyagnam	2023-11-25	5:00am - 6:00am	102
vyagnam	2023-12-10	5:00am - 6:00am	101
vyagnam	2023-12-10	5:00am - 6:00am	102
vyagnam	2023-12-10	5:00am - 6:00am	103
vyagnam	2023-12-13	6:00am - 7:00am	101
vyagnam	2023-12-20	10:00am - 11:00am	101
venisri	2023-12-13	5:00am - 6:00am	101
student	2023-12-14	5:00am - 6:00am	101

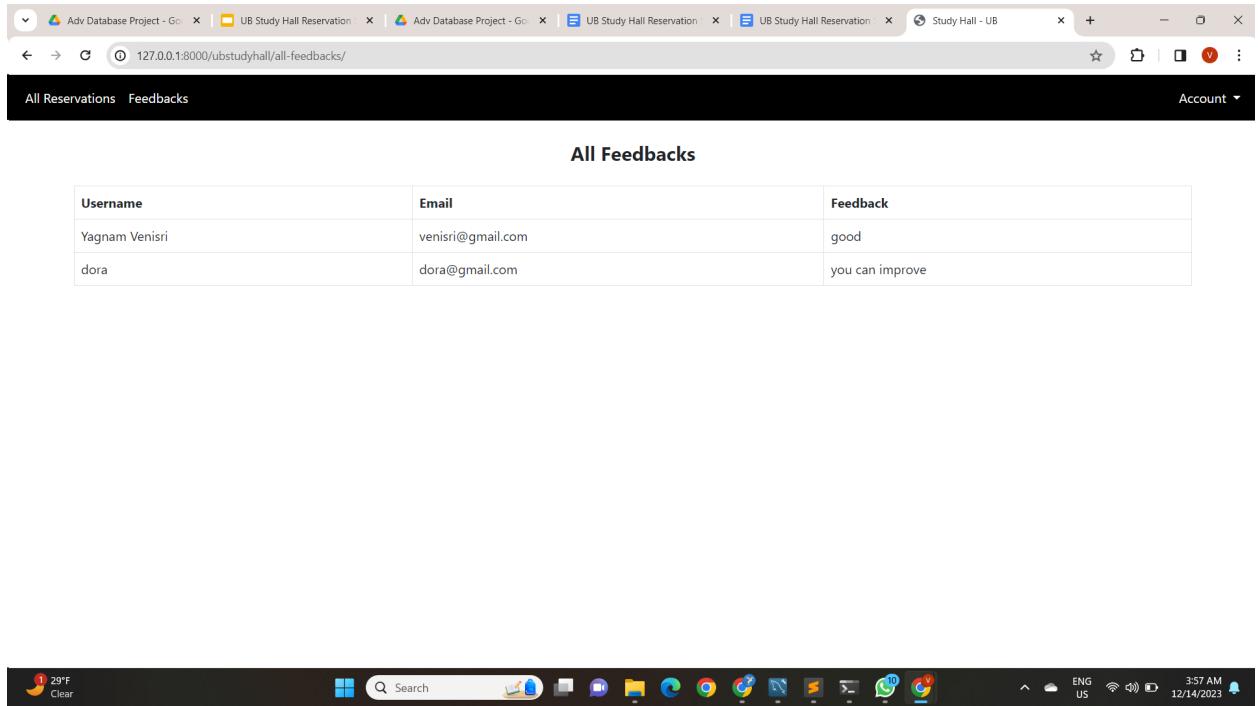
127.0.0.1:8000/ubstudyhall/studyhall-reservations/

29°F Clear

Search

3:56 AM 12/14/2023

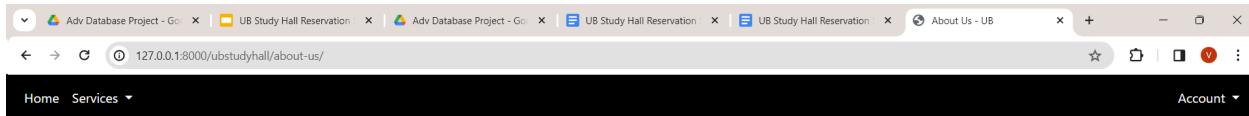
## Admin View: All Feedbacks



Username	Email	Feedback
Yagnam Venisri	venisri@gmail.com	good
dora	dora@gmail.com	you can improve

The "Admin View: All Feedbacks" section offers administrators a comprehensive overview of all feedback submitted by users within the study hall reservation system. This exclusive feature provides insights into user opinions, suggestions, or concerns, facilitating efficient management and prompt responses.

## About Us: Project Motto and Mission



## About Us

Welcome to UB Study Hall! We provide a convenient and efficient room reservation service for students. Our goal is to offer a seamless experience for booking study rooms on campus.

### Our Services

- Easy Room Reservation: We offer a simple and user-friendly platform to reserve study rooms based on availability.
- Flexible Timings: Our system allows you to select from a range of timings throughout the day, making it convenient for your study schedule.
- Room Details: Get comprehensive information about each room, including capacity and location, to help you choose the perfect study environment.
- Points System: Earn points with every reservation, which you can redeem for rewards or benefits.

We understand the importance of having a dedicated space for studying. Our services aim to make your academic journey more productive and enjoyable.

Start using UB Study Hall today and make your study sessions more organized and efficient!



The "About Us" section embodies the ethos and aspirations of the UB Study Hall Reservation System. It serves as a platform to articulate the core values, objectives, and mission driving the project and services provided.

## Conclusion:

The "UB Study Hall Reservation System" project has not only met but exceeded the initial objectives set for the system. By leveraging modern web technologies, a robust database management system, and cloud hosting capabilities, we've created a comprehensive solution that enhances the student experience in reserving study hall rooms.

This project allowed us to gain insights into the complexities of developing an integrated system, managing databases, and designing user-friendly interfaces. Additionally, it emphasized the importance of user feedback and adaptability in refining the system for optimal user satisfaction.

The successful completion of this project underscores the importance of collaborative teamwork, technical proficiency, and adaptability in delivering efficient solutions to real-world problems. Our learnings from this project will undoubtedly influence our future endeavors in developing innovative and user-centric applications.

Teamwork played a pivotal role, demonstrating the importance of effective collaboration, communication, and division of tasks, ensuring the project progressed smoothly.

Future iterations could focus on enhancing functionalities like the points system, feedback mechanisms, and reservation management for increased user engagement and efficiency. Consideration of additional features, such as real-time chat support or advanced analytics, to further enrich the user experience.