A TECHNICAL PROJECT REPORT ON

"Hotel Management Systems"

TEAM MEMBERS

Sandeep Anumula - S02057678

Srinivas Madala - S02060306

Sai Prashanth Reddy Dyapa - S02060447

Ruthik Juvva - S02060044

Akash Badavath - S02059991

Purushottam Reddy Tippani - S02058159

SOUTHEAST MISSOURI STATE UNIVERSTY

INSTRUCTOR NAME: ZIPPING LU

COURSE: DISTRIBUTED CLOUD COMPUTING

SPRING - 2024

Introduction:

The goal of the Hotel Management System is to make managing a hotel easier while still meeting the needs of both employees and visitors. Without the limitations of conventional manual methods, this system provides a digital platform that facilitates the effective handling of reservations, guest services, and administrative tasks.

Objective:

The Hotel Management System's three primary objectives are to enhance guest experiences, expedite administrative work, and manage hotel operations. In addition to improving overall guest experience, this project aims to increase operational efficiency and optimize revenue potential for the hotel by developing a user-friendly platform.

Project Analysis:

Functional Requirements:

Accommodations can be provided to guests by registering their details in advance.

Check-in and check-out times are displayed in real time for guests, making reservations simple.

Access to the Blog: For news on events, local attractions, and hotel updates, visitors can visit the blog section.

Integration of Dashboard: Guests can effectively manage reservations, requests, and preferences with the integration of an extensive dashboard system.

Non-Functional Requirements: -

Security: Mandatory password changes improve the security of the initial login.

Portability: The system is made to function flawlessly on a variety of hardware and operating systems.

Audit Trail: An audit trail records every login attempt, whether successful or not, to preserve data integrity and keep an eye on system access.

SDLC:

Planning, designing, implementing, testing, and maintaining software applications are all part of the Software Development Life Cycle (SDLC), which is an organized method. The goal of the SDLC process is to guarantee that software projects are finished on schedule, within budget, and without errors. Software developers can achieve the best possible results in software development by adhering to the SDLC methodology.

The project follows a structured SDLC approach: -

1. Planning
2. Design
3. Implementation
4. Development
5. Testing
6. Maintenance

Modules:

- Room Management: Manages room inventory, descriptions, and pricing.
- Reservation System: Facilitates room booking, modification, and cancellation.

- Payment Gateway: Ensures secure online transactions for room bookings.
- Guest Services: Manages requests such as room service, housekeeping, and concierge.
- Customer Management: Manages guest accounts, preferences, and loyalty programs.
- Content Management: Handles website content, including room details, amenities, and local attractions.
- Staff Management: Manages staff assignments, schedules, and performance tracking.
- Cloud Technologies and Services Used:
- The hotel management system utilized several cloud technologies and services for reliability and scalability:

Amazon Web Services (AWS) for hosting and server management.

Google Cloud Platform (GCP) for data storage and processing.

Firebase for real-time guest interactions and notifications.

Key services included:

- The application servers were hosted on AWS Elastic Compute Cloud (EC2), which was one of the key services.
- The Relational Database Service (RDS) offered by AWS facilitates the effective management of MySQL databases.
- To secure access to AWS services, use AWS Identity and Access Management (IAM).
- Static files and media can be stored on Amazon S3.

- Guni corn: Used to administer and execute the Python application as a WSGI server.
- Nginx: A reverse proxy that improves security, controls SSL/TLS, and boosts capacity to handle loads.

Implementation:

The hotel management system was developed and deployed using a combination of technologies. Python and Django were employed for the back-end functionalities, while HTML5, CSS3, JavaScript, and Bootstrap were utilized for the front-end design.

Key implementation steps included:

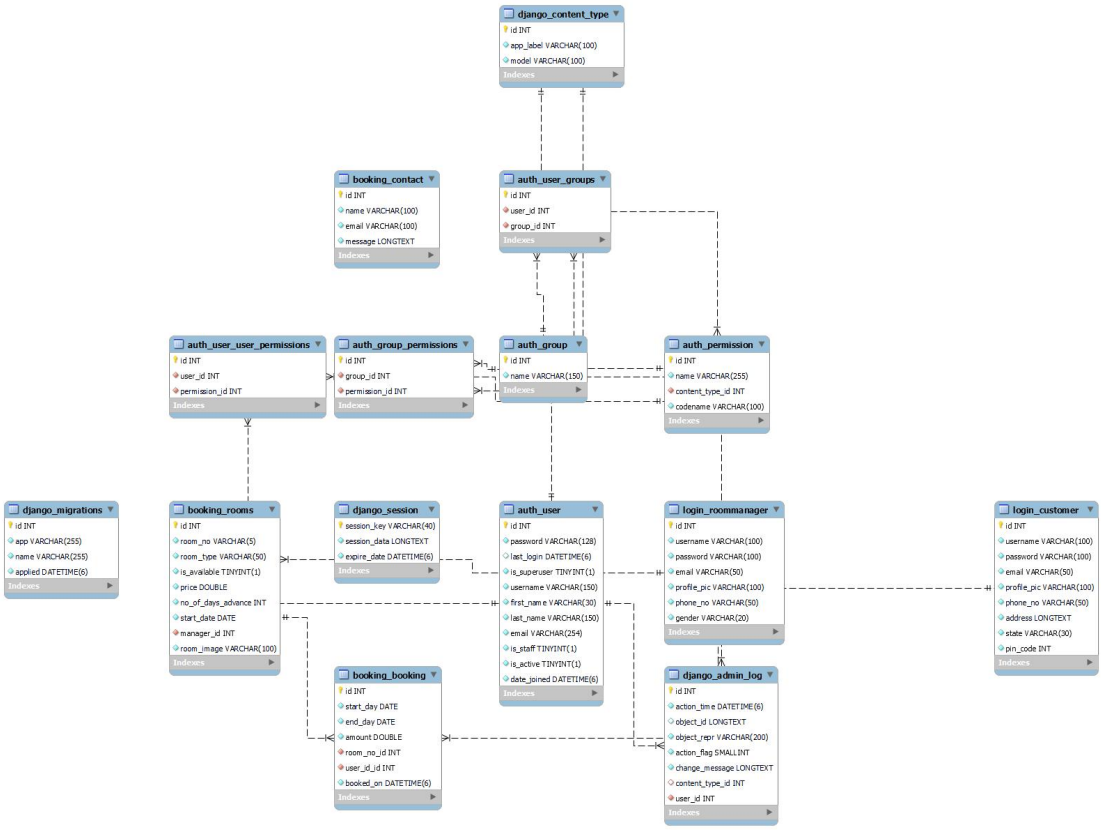
Server Configuration: Configured EC2 instances on AWS running Ubuntu, installed necessary software packages, and set up Guni corn with Nginx to serve the Django application, ensuring optimal performance and scalability.

Database Setup: Utilized AWS RDS for MySQL to host the hotel management system's database, ensuring data security, reliability, and scalability to accommodate the system's requirements.

Deployment: Implemented continuous integration and deployment processes using GitHub Actions. Automated testing and deployment procedures were established to streamline the deployment of updates and ensure the system's stability on AWS EC2 instances.

Tools and Technologies Used for Application Development:

UML diagrams:



Front-End:

HTML5: Utilized for structuring the web content and defining the layout.

CSS3: Applied for styling and enhancing the presentation of web pages.

JavaScript: Implemented to add dynamic behavior and interactivity to the user interface.

Bootstrap: Employed as a front-end framework for creating responsive and mobile-friendly designs.

Back-End:

Python: Chosen as the primary programming language for implementing server-side logic.

Django: Utilized as a high-level Python web framework to expedite development and ensure a clean architecture.

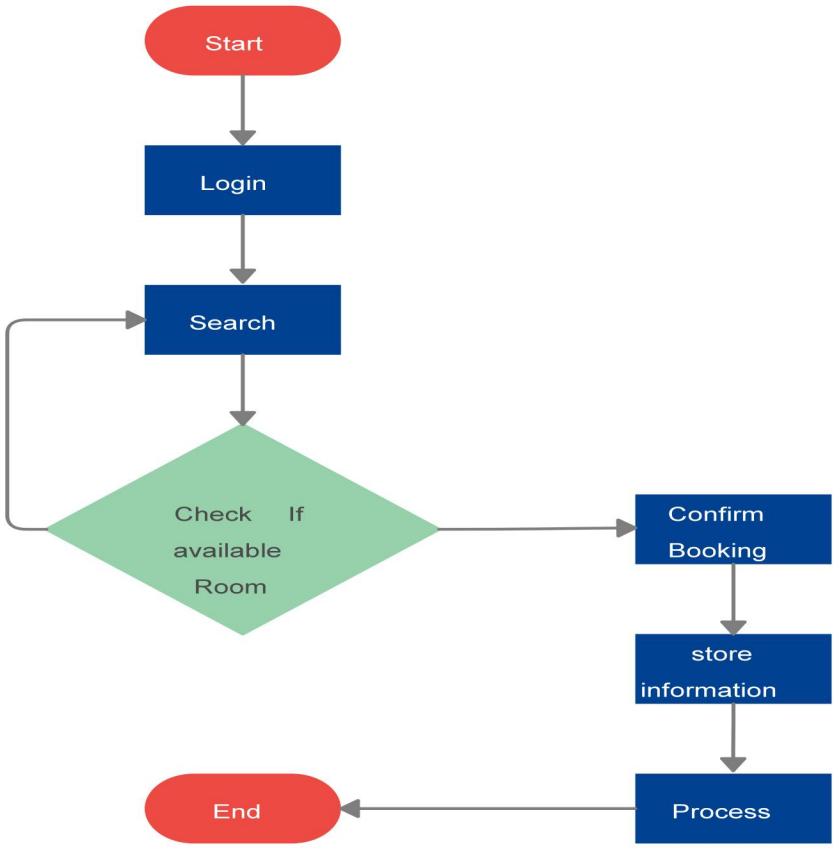
Pandas: Used for efficient data manipulation and analysis, particularly beneficial for handling extensive data sets.

Database:

AWS RDS for MySQL: Deployed as a managed relational database service to store and manage hotel data securely and efficiently.

MySQL Workbench: Utilized as a visual tool for database development and administration, offering features for designing, modeling, and querying the database schema.

Block Diagram:



The Hotel Management System is designed to ensure high availability and scalability, utilizing AWS EC2 instances for hosting and AWS RDS for database management. The system is built using Django, Guni corn, and Nginx to handle server-side logic and client requests efficiently. The front-end interface, developed with HTML5, CSS3, JavaScript, and Bootstrap, provides a user-friendly and interactive experience.

Challenges and Solutions:

Scalability: Addressed by leveraging cloud-based services such as AWS EC2 and RDS, allowing for automatic scaling of

resources based on demand. This ensures that the hotel management system can handle varying levels of traffic without experiencing performance issues.

Security: Managed by integrating AWS security features like security groups and IAM roles to protect both the application and its data. Encryption protocols and access control mechanisms were implemented to safeguard sensitive information and prevent unauthorized access.

Deployment Complexity: Handled by utilizing tools like Gunicorn and Nginx, which simplify the deployment process and enhance the system's performance. Automated deployment pipelines, configured through services like AWS Code Pipeline, streamline the deployment process and reduce the risk of errors.

Results and Outcomes:

The deployment of the Hotel Management System on AWS yielded significant improvements in handling user requests and data processing speeds. The system demonstrated robustness in managing simultaneous connections, showcasing the effectiveness of the cloud infrastructure utilized.

Insights gained from the deployment include the importance of proper initial configuration and the benefits of cloud scalability in adapting to changing demands. The hotel management system's performance enhancements and increased reliability contribute to improved guest experiences and operational efficiency.

Implementation:

The implementation of a hotel management system follows a similar process to that of an e-commerce platform, with adjustments tailored to the hospitality industry. Here's a breakdown of the implementation process:

Planning:

Define the hotel management system's architecture, identifying key features such as room management, reservation system, guest services, and staff management tools.

Design the user interface and layout, considering factors like room booking process, guest check-in/check-out flow, and staff management interfaces.

Outline the integration of necessary functionalities such as payment processing, guest communication tools, and third-party services for booking management.

Platform Selection:

Choose a suitable technology stack for developing the hotel management system, considering factors like scalability, security, and ease of integration.

Select a hotel management software or framework that aligns with the system's requirements and budget, ensuring it provides essential features for hotel operations.

Development:

Develop the user interface for the hotel management system, focusing on intuitive design and seamless navigation for guests and staff.

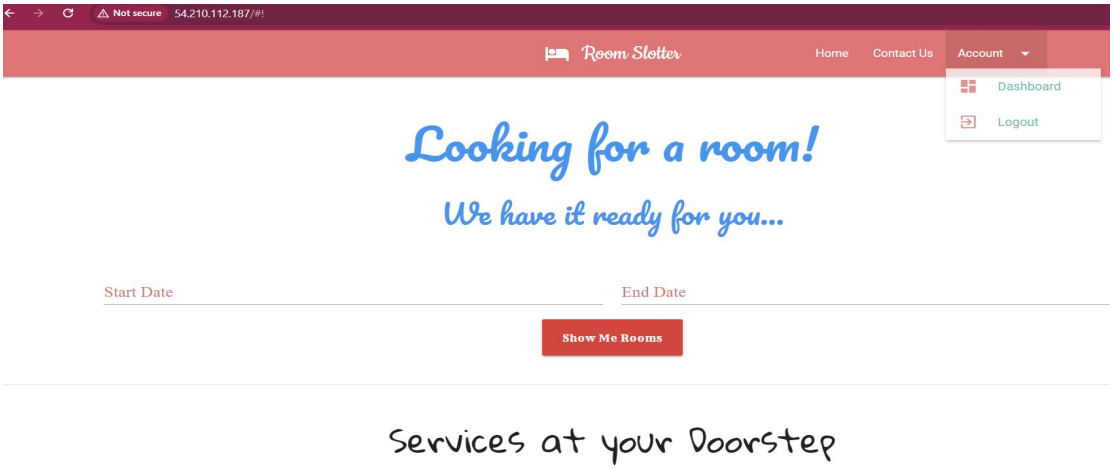
Implement backend functionalities using chosen technologies like Python and Django, ensuring efficient management of rooms, reservations, and guest services.

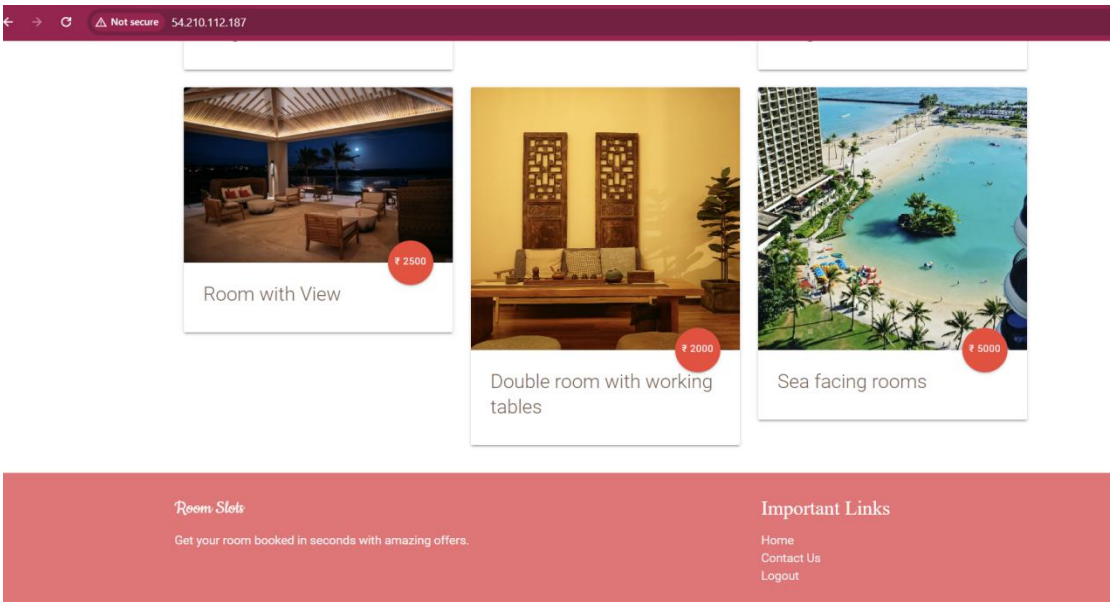
Integrate necessary features such as payment gateways, communication tools, and database management for storing guest information and transaction records.

Testing:

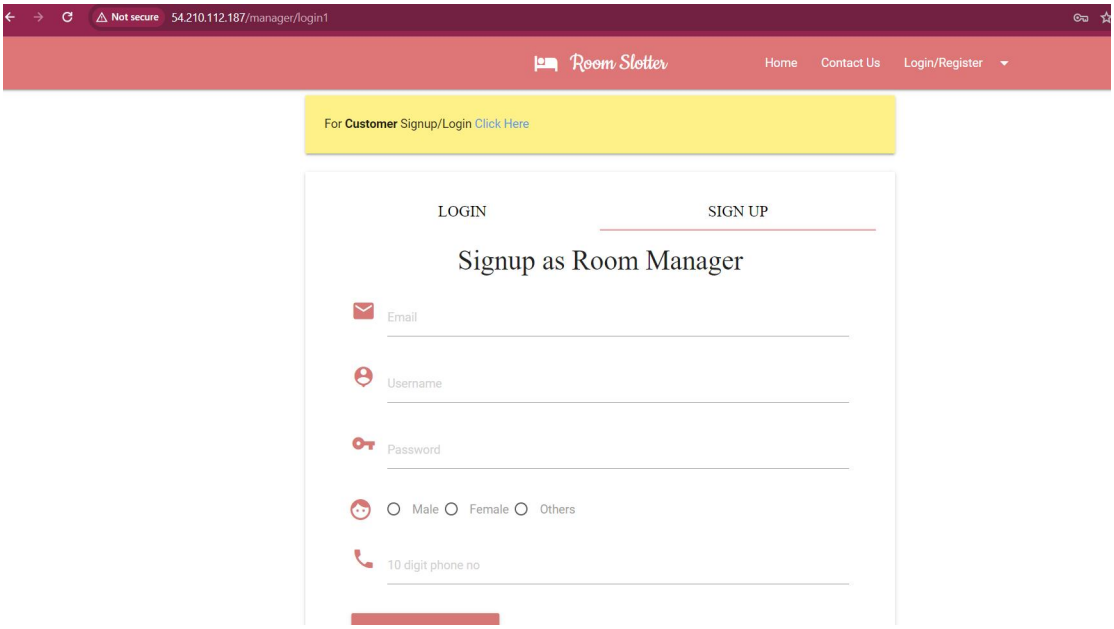
Main page:

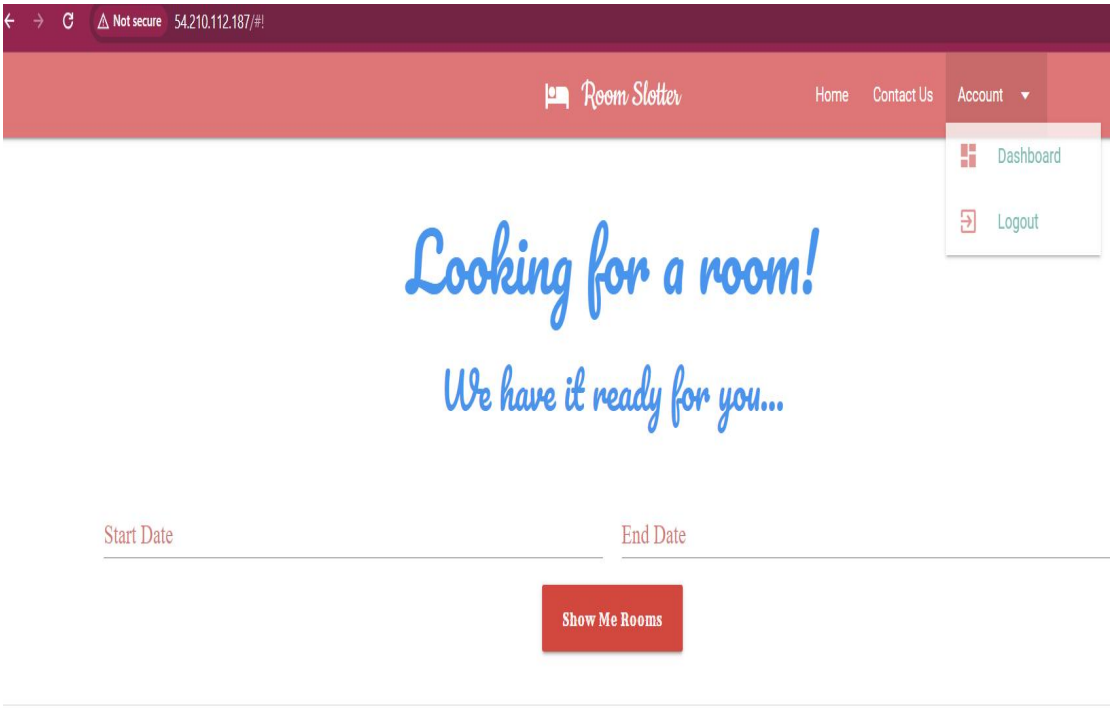
Dashboard:





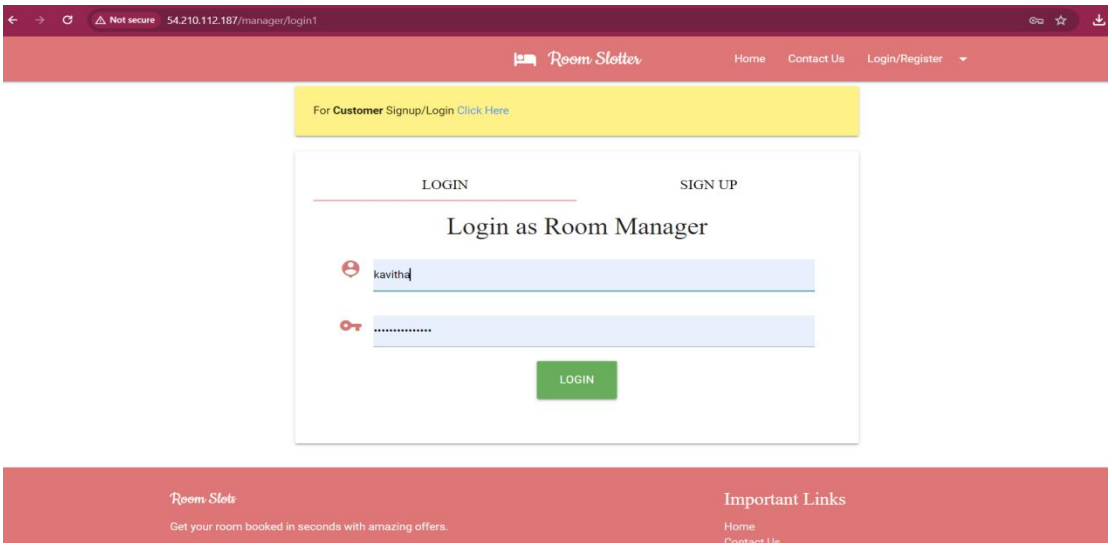
Signup as room manager:



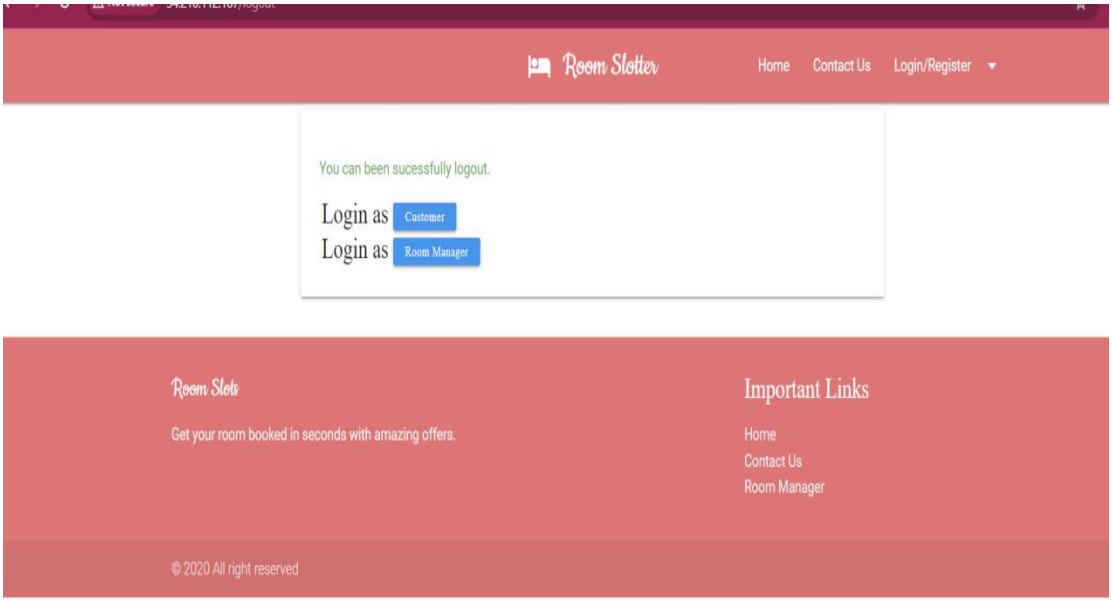


Services at your Doorstep

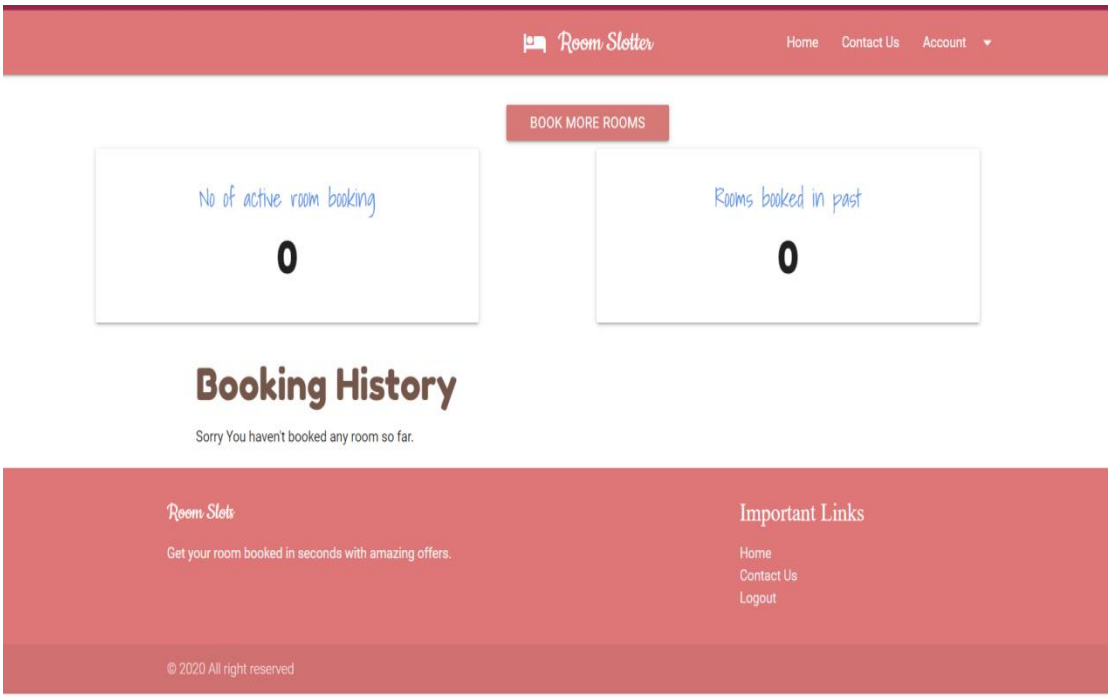
Sign in or login page



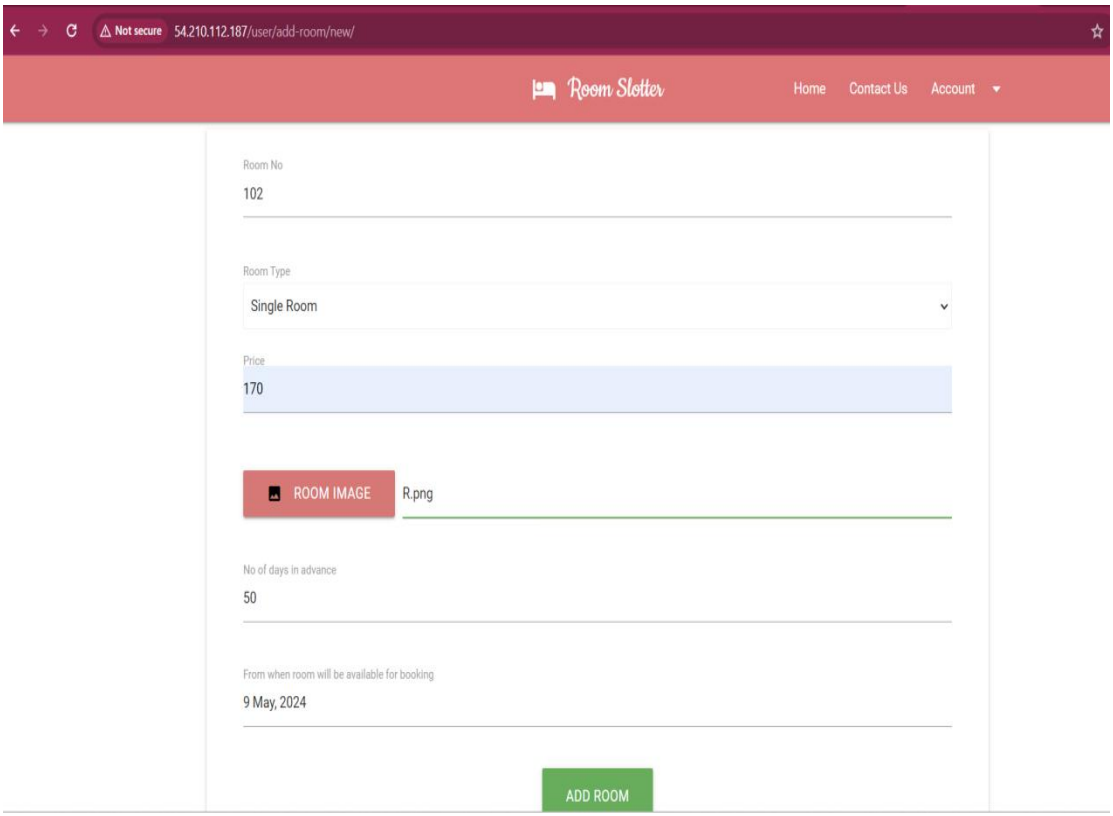
Login successful:

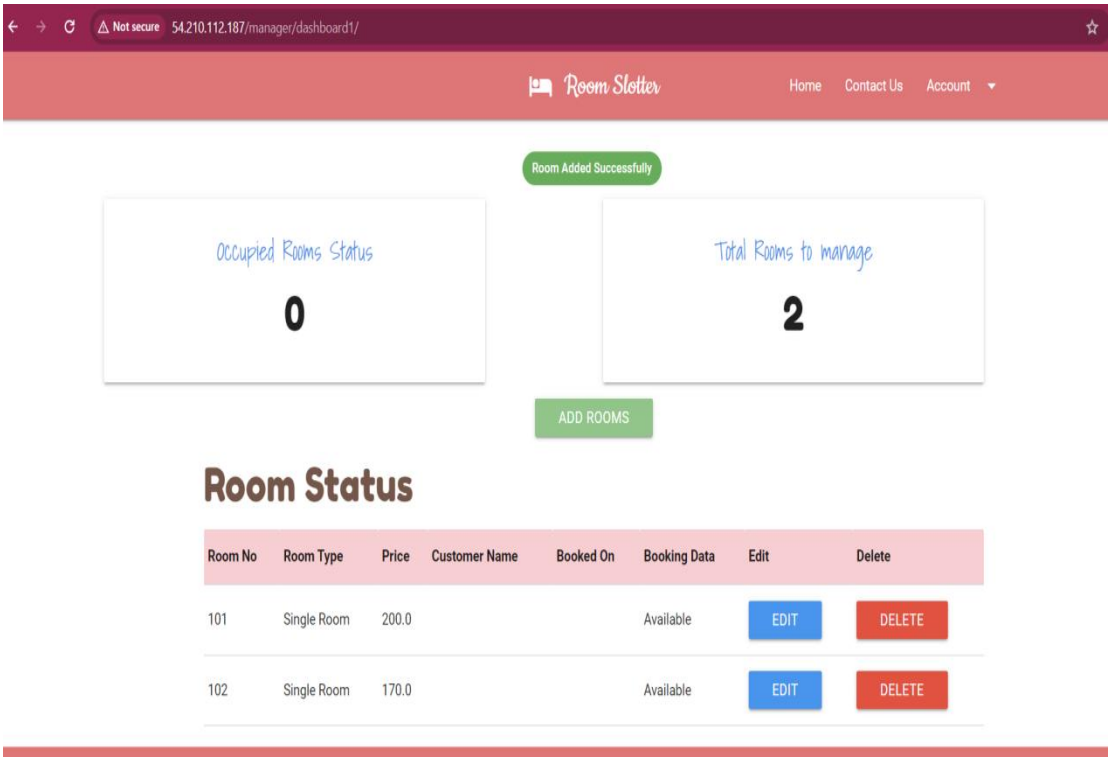


Before adding the room page;

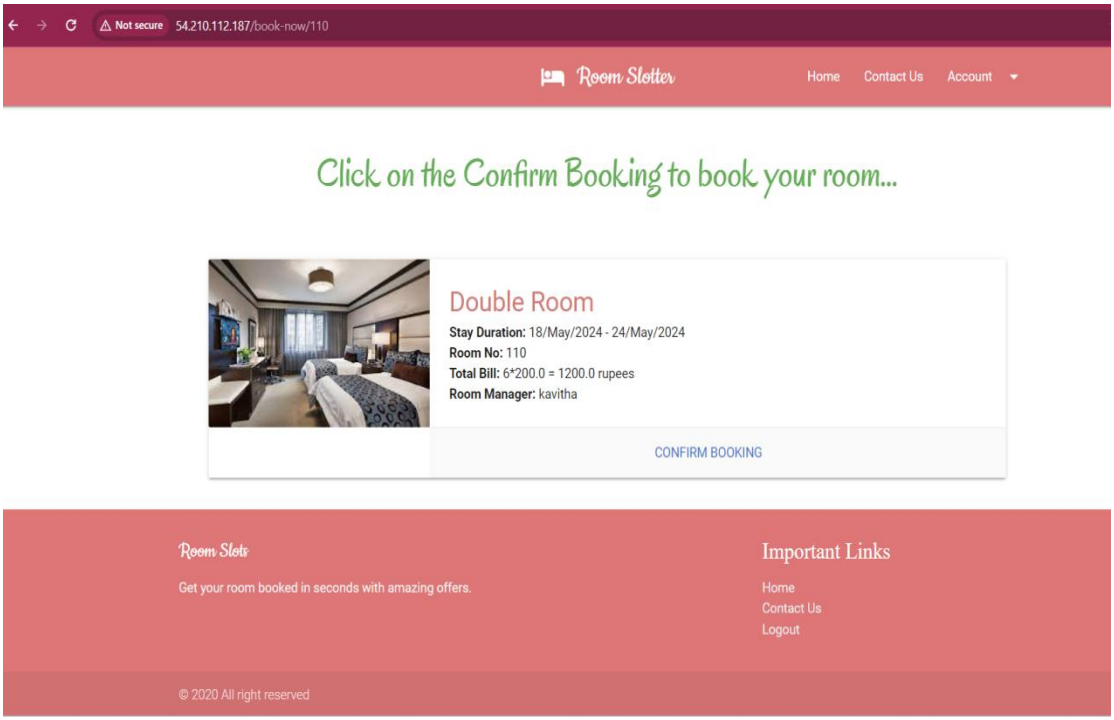


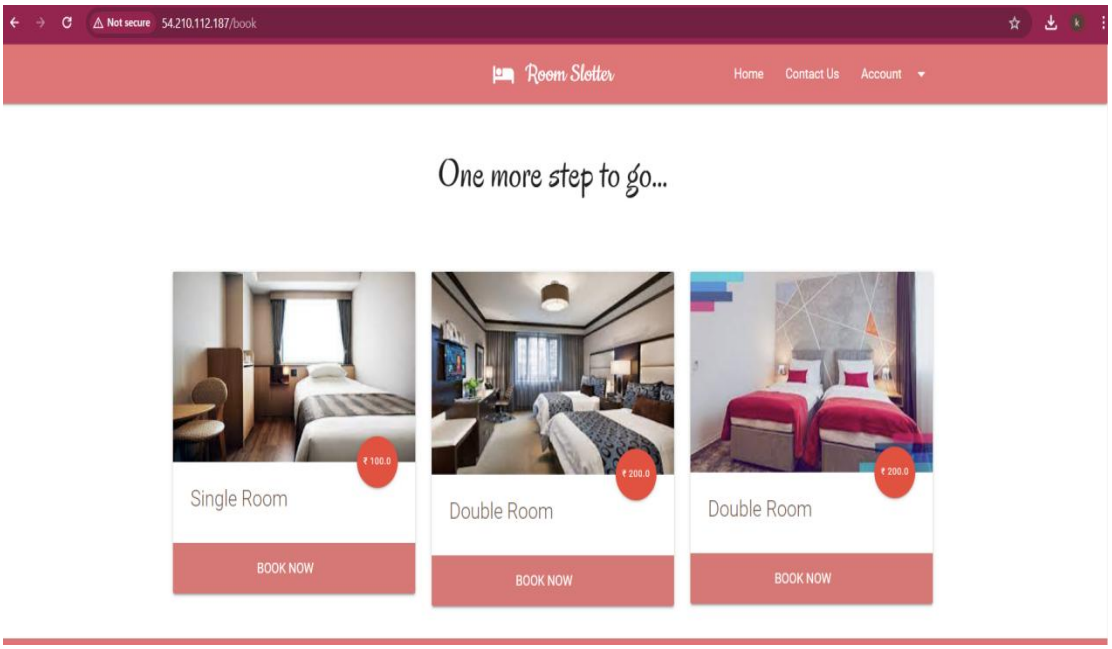
Add the room details:



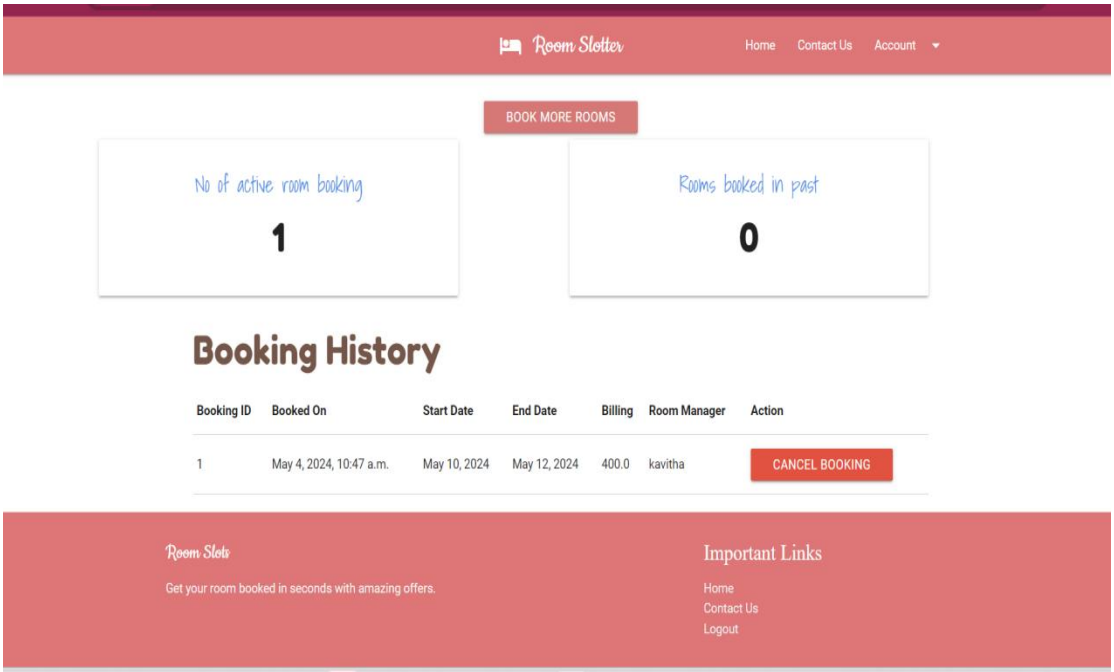


Confirm booking your room



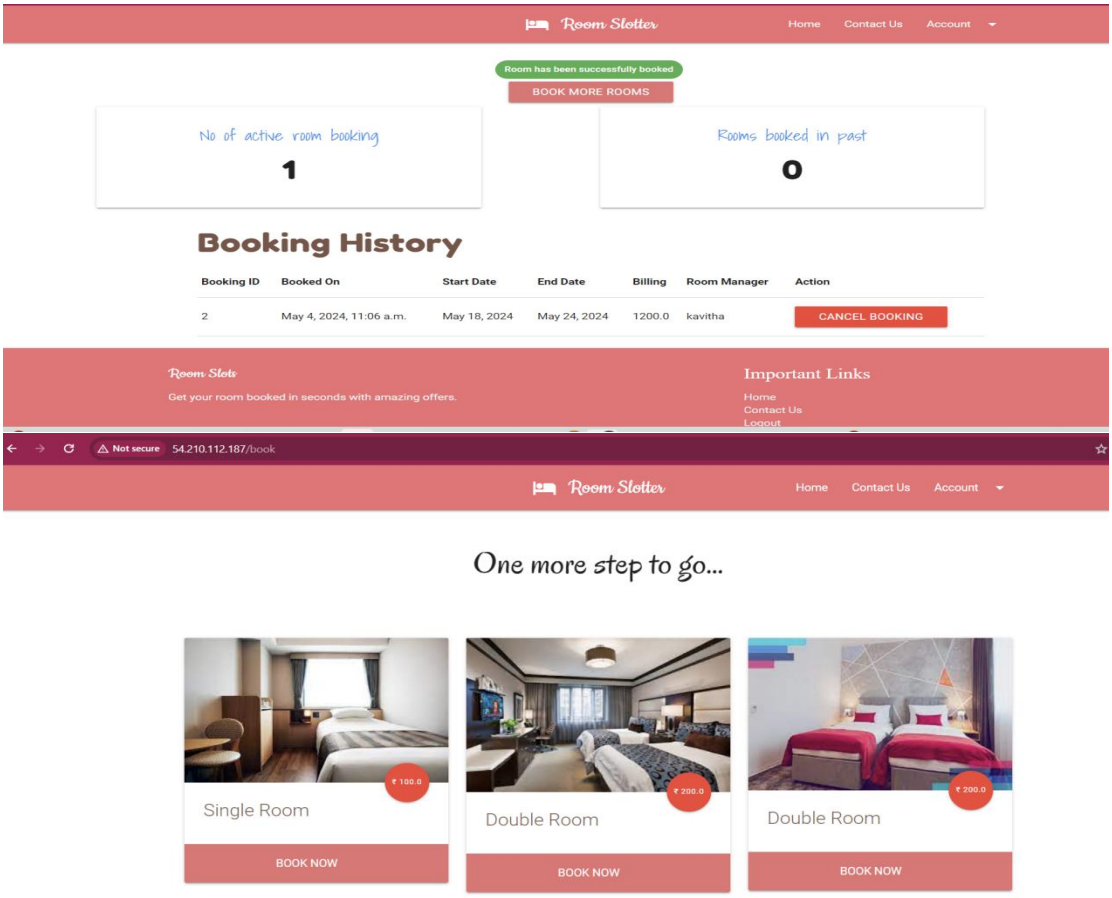


Booking room details:



Click on Book now

Confromation for booking page



Conduct thorough testing of the hotel management system to identify and address any bugs, errors, or usability issues.

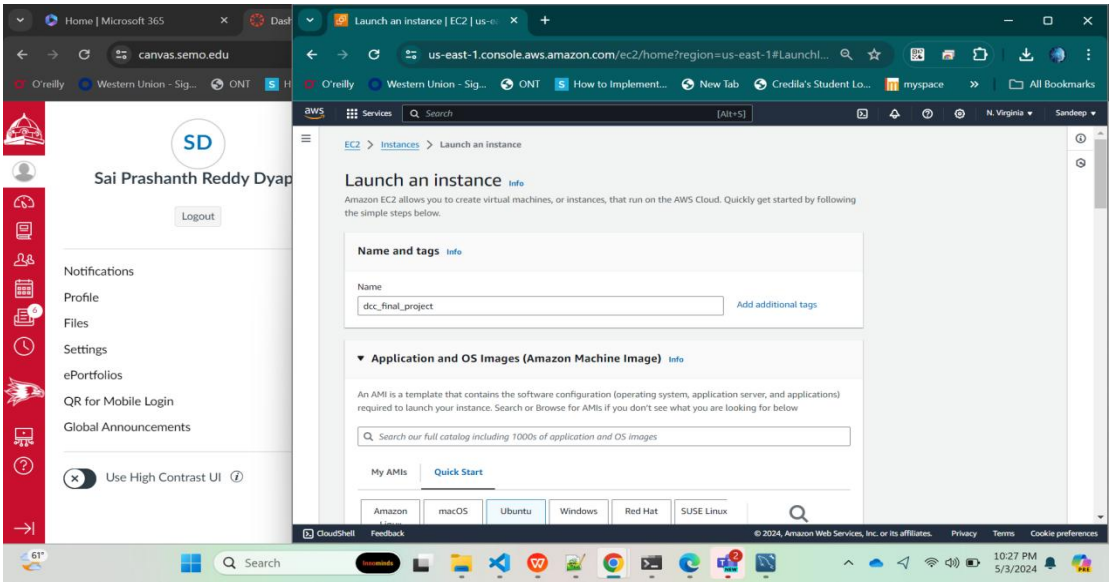
Perform functional testing to ensure all features work as intended, including room booking, reservation management, and staff tools.

Test the system's performance under various conditions to optimize speed and responsiveness.

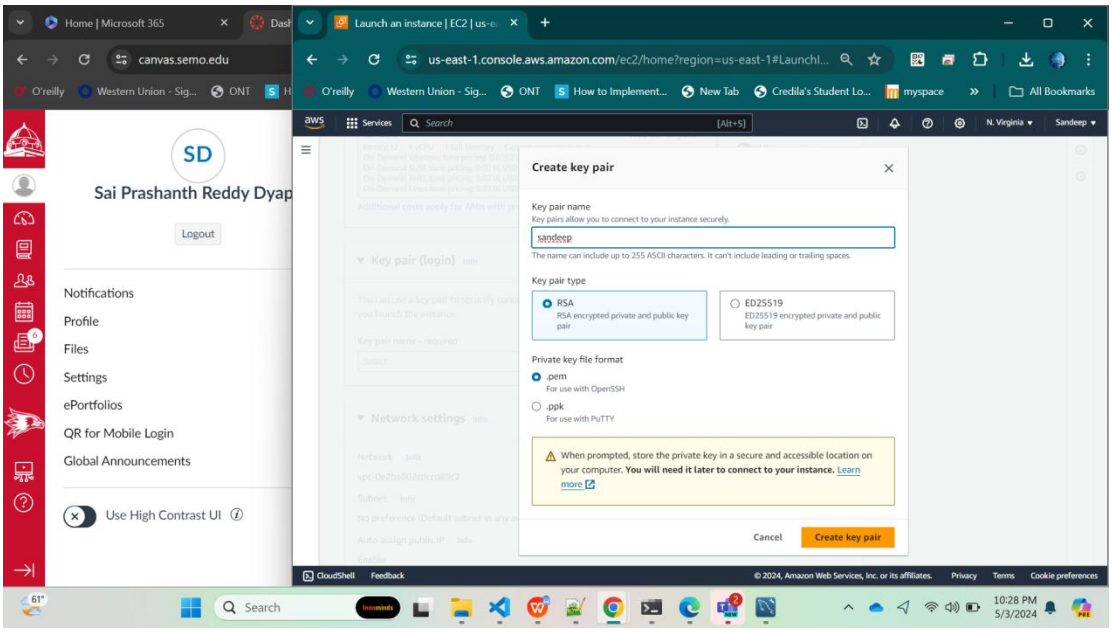
Conduct security testing to identify vulnerabilities and implement measures to protect guest data and system integrity.

Deployement steps:

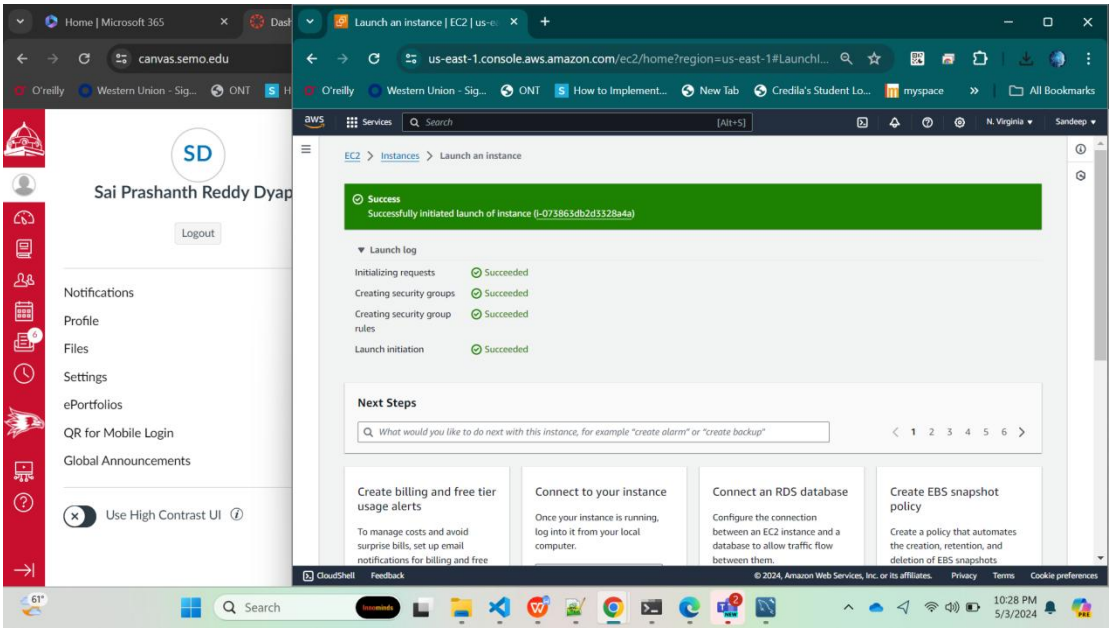
1.Launch Instance

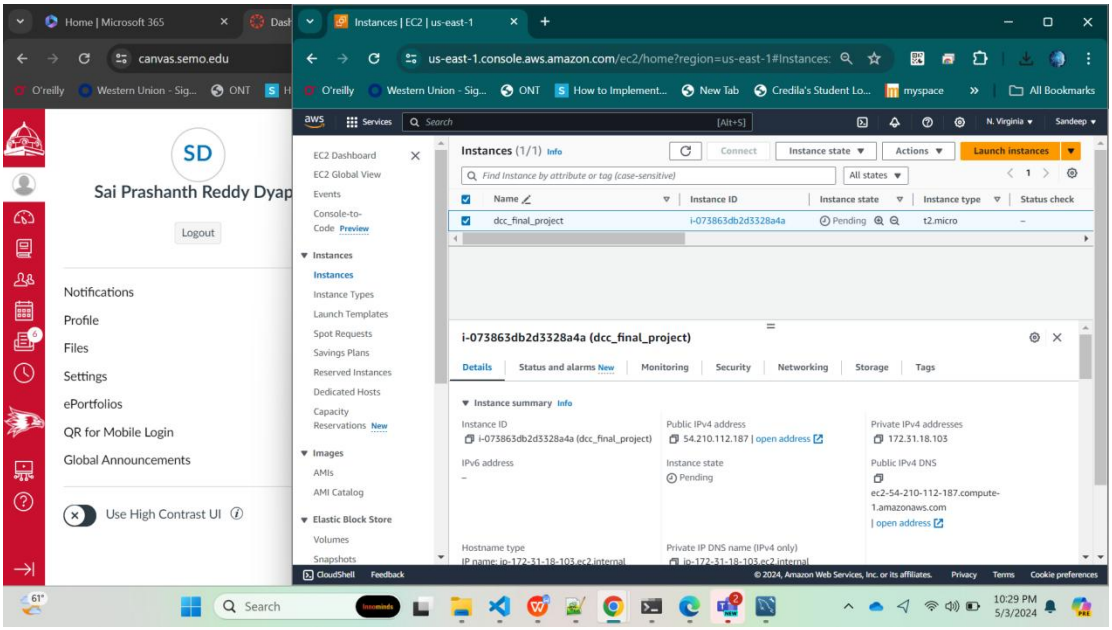


2. create key pair and rsa and .pem select then click on create key pair



Instance sources successfully.

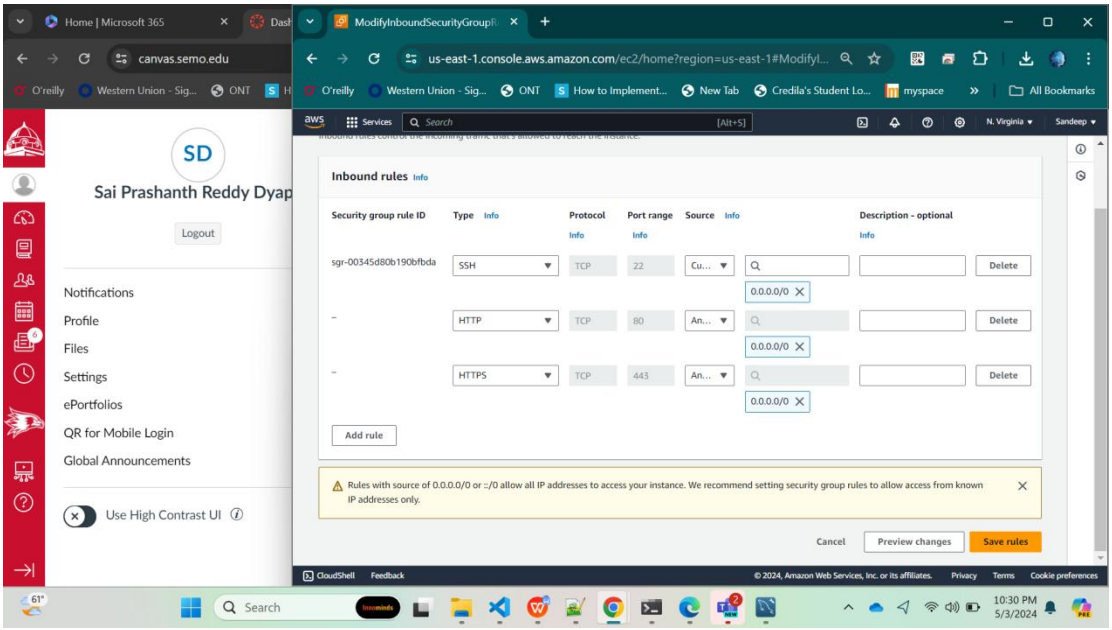




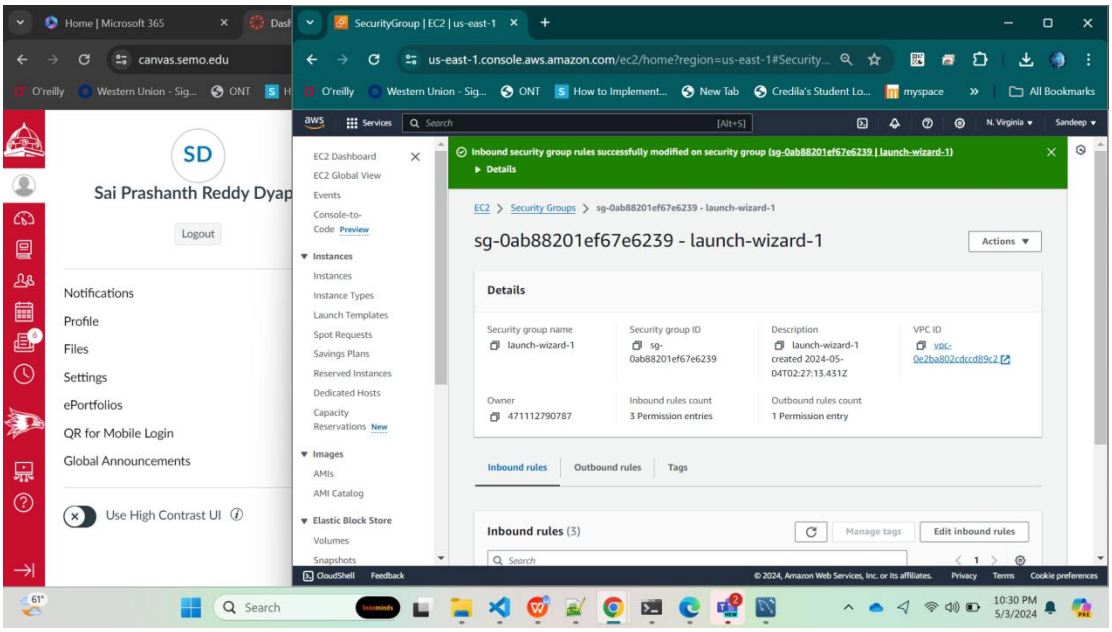
Click on inbound rules

Select SSH,ANY and save rules.

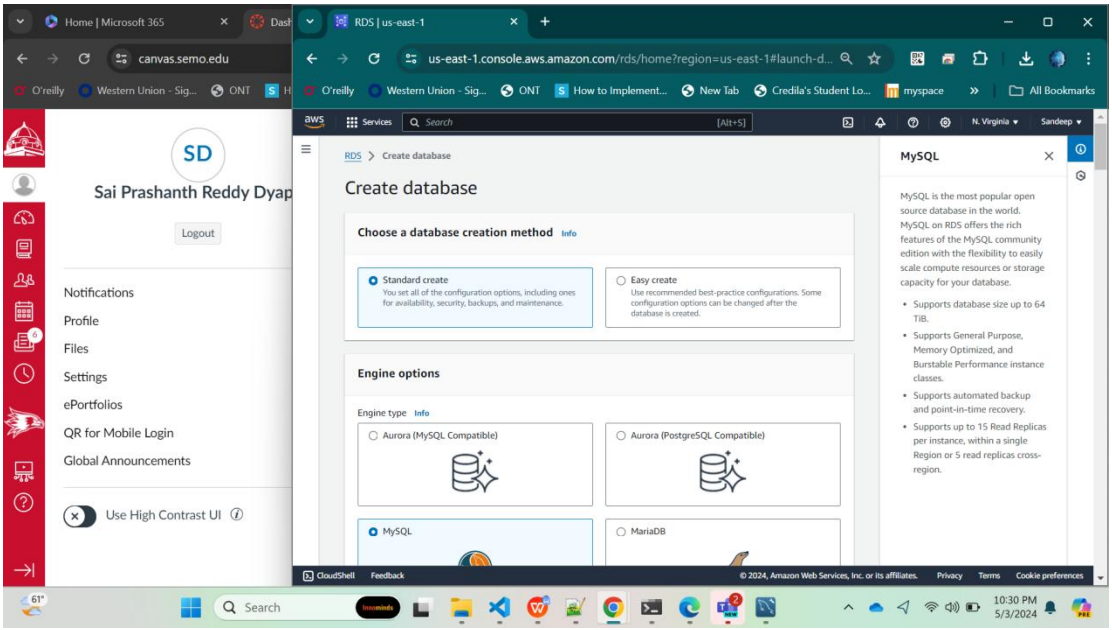
Check inbound rules



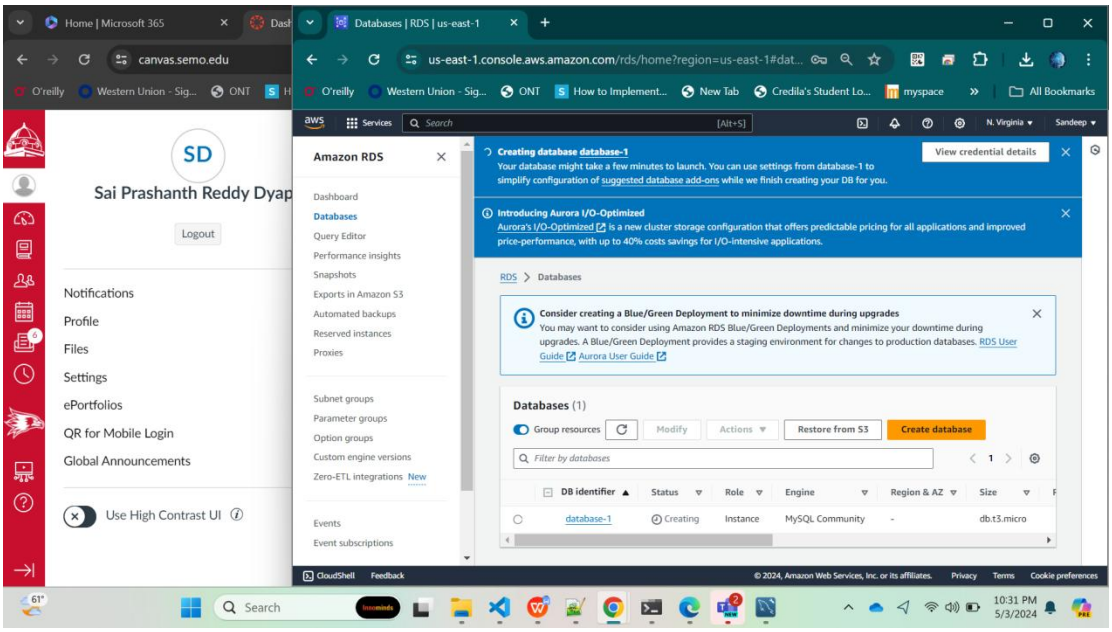
Instance group rules successfully added.



Database connection to RDS:

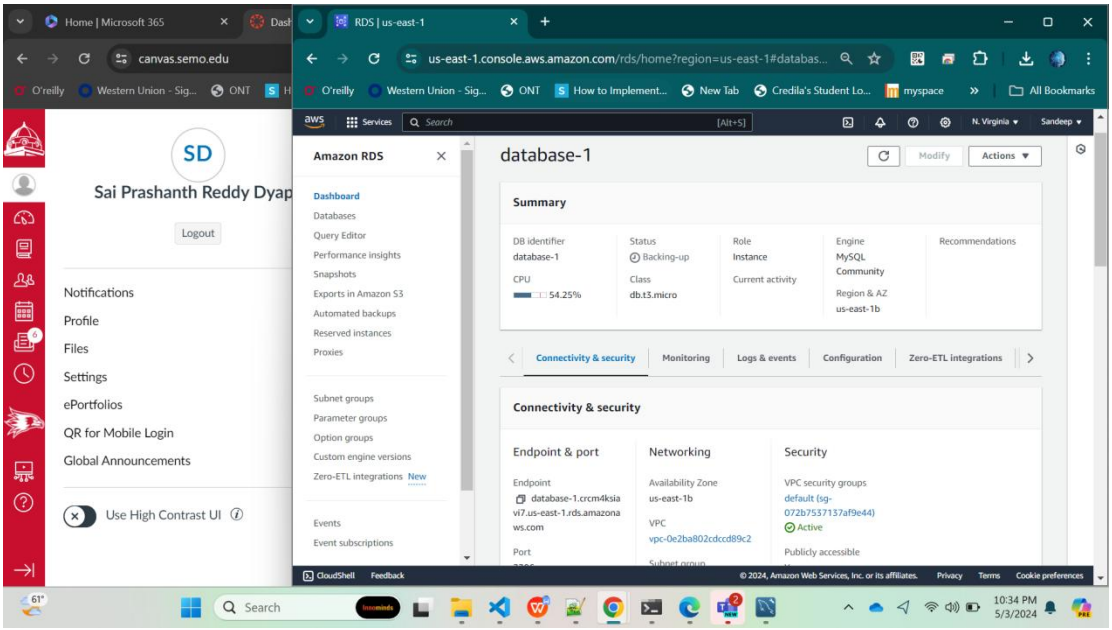


Creating database dabase-1.

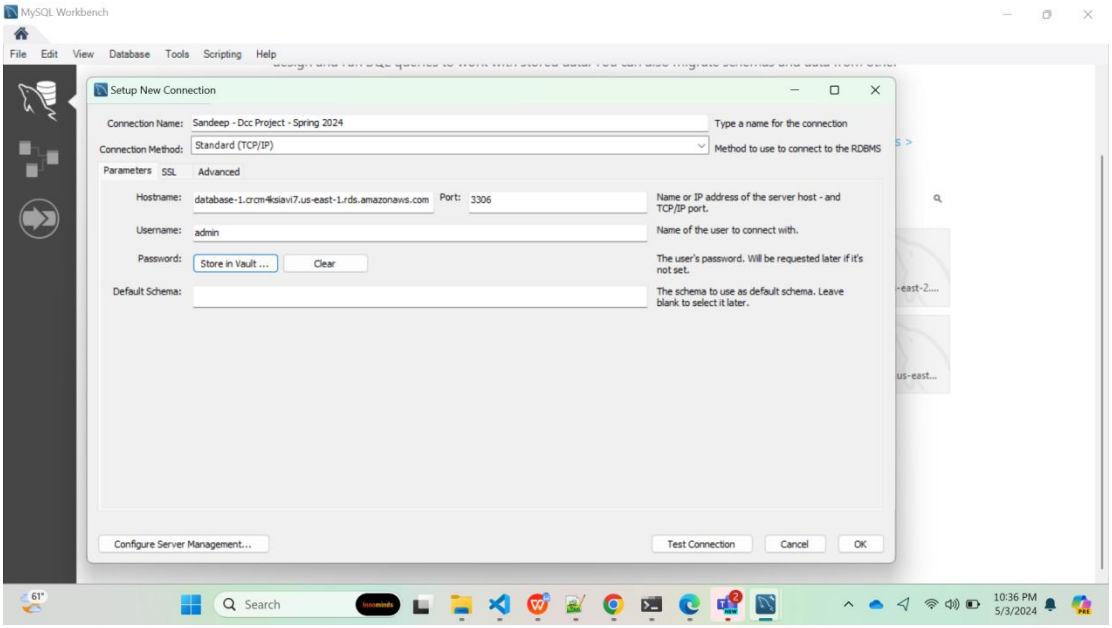


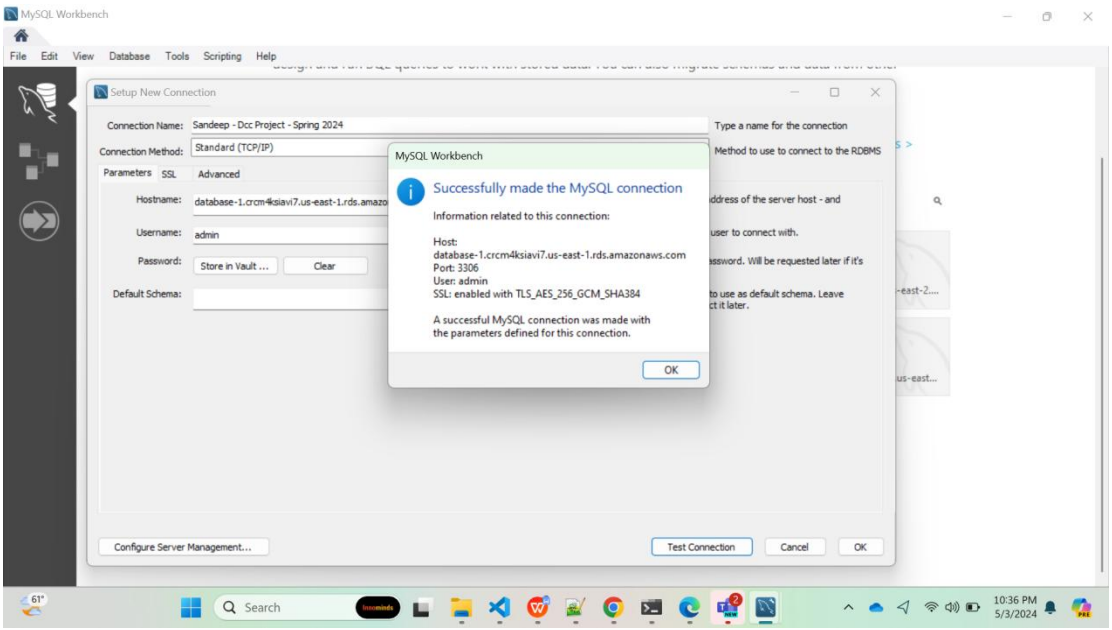
Creating database dabase-1.

Lets check community & security.

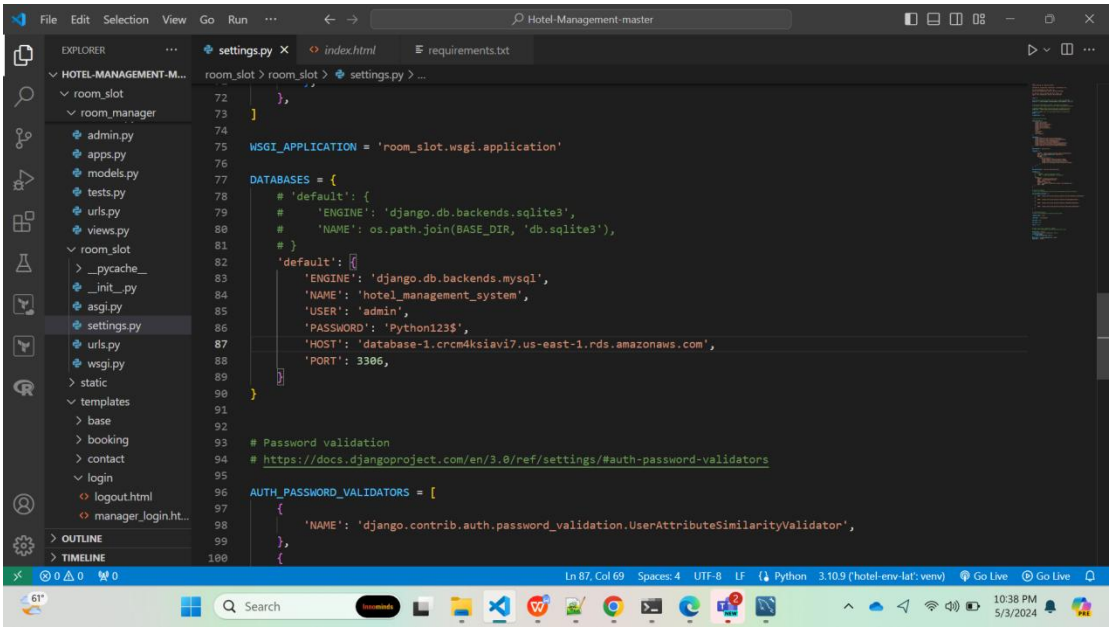

Home | Microsoft 365
canvas.semo.edu
SD
Sai Prashanth Reddy Dyap
Logout
Notifications
Profile
Files
Settings
ePortfolios
QR for Mobile Login
Global Announcements
Use High Contrast UI
RDS | us-east-1
us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#databas...
Amazon RDS
Dashboard
Databases
Query Editor
Performance insights
Snapshots
Exports in Amazon S3
Automated backups
Reserved instances
Proxies
Subnet groups
Parameter groups
Option groups
Custom engine versions
Zero-ETL integrations New
Events
Event subscriptions
database-1
Modify
Actions
Summary
DB identifier
database-1
Status
Backing-up
Role
Instance
Engine
MySQL Community
Recommendations
CPU
54.25%
Class
db.t3.micro
Current activity
Region & AZ
us-east-1b
Connectivity & security
Monitoring
Logs & events
Configuration
Zero-ETL integrations
Endpoint & port
Networking
Security
Endpoint
database-1.crcm4ksiavi7.us-east-1.rds.amazonaws.com
Availability Zone
us-east-1b
VPC security groups
default (sg-072b7537137af9e44)
Active
VPC
vpc-0e2ba802cdccd89c2
Publicly accessible
Port
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
10:34 PM
5/3/2024

Workbench:

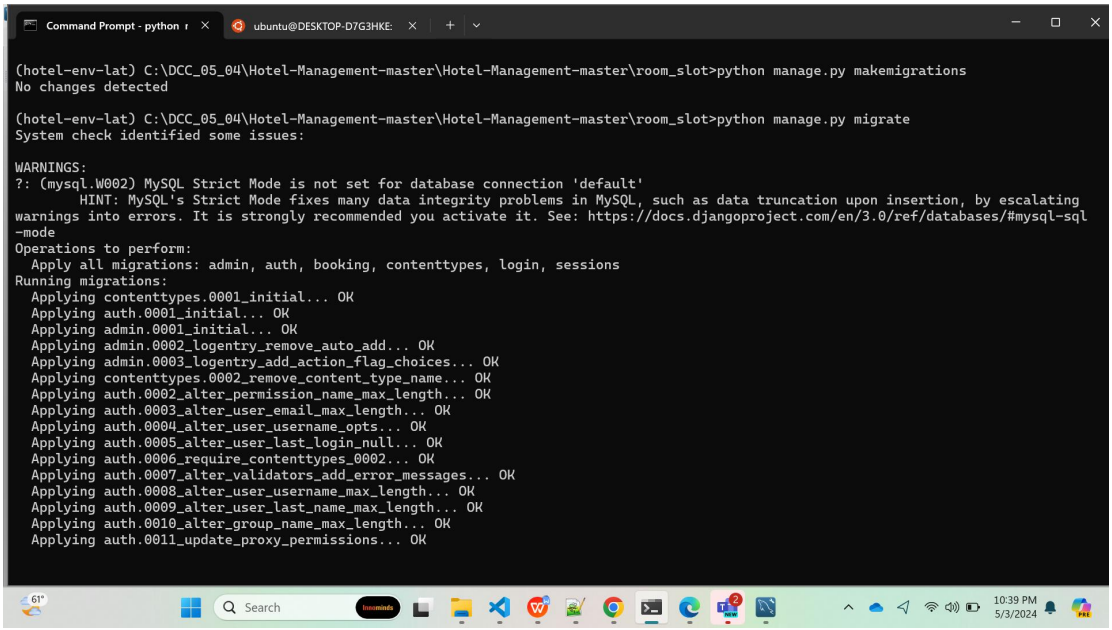

MySQL Workbench
File Edit View Database Tools Scripting Help
Setup New Connection
Connection Name: Sandeep - Dcc Project - Spring 2024
Type a name for the connection
Connection Method: Standard (TCP/IP)
Method to use to connect to the RDBMS
Parameters SSL Advanced
Hostname: database-1.crcm4ksiavi7.us-east-1.rds.amazonaws.com
Port: 3306
Name or IP address of the server host - and TCP/IP port.
Username: admin
Name of the user to connect with.
Password: Store in Vault ... Clear
The user's password. Will be requested later if it's not set.
Default Schema:
The schema to use as default schema. Leave blank to select it later.
Configure Server Management...
Test Connection
Cancel
OK
Search
10:36 PM
5/3/2024



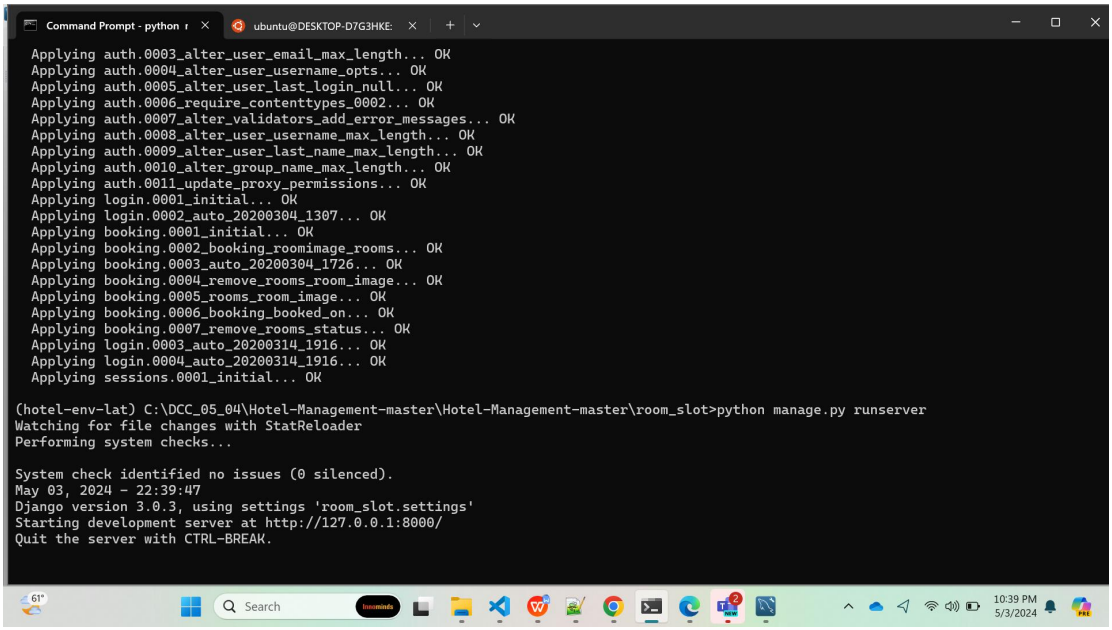
Databases:



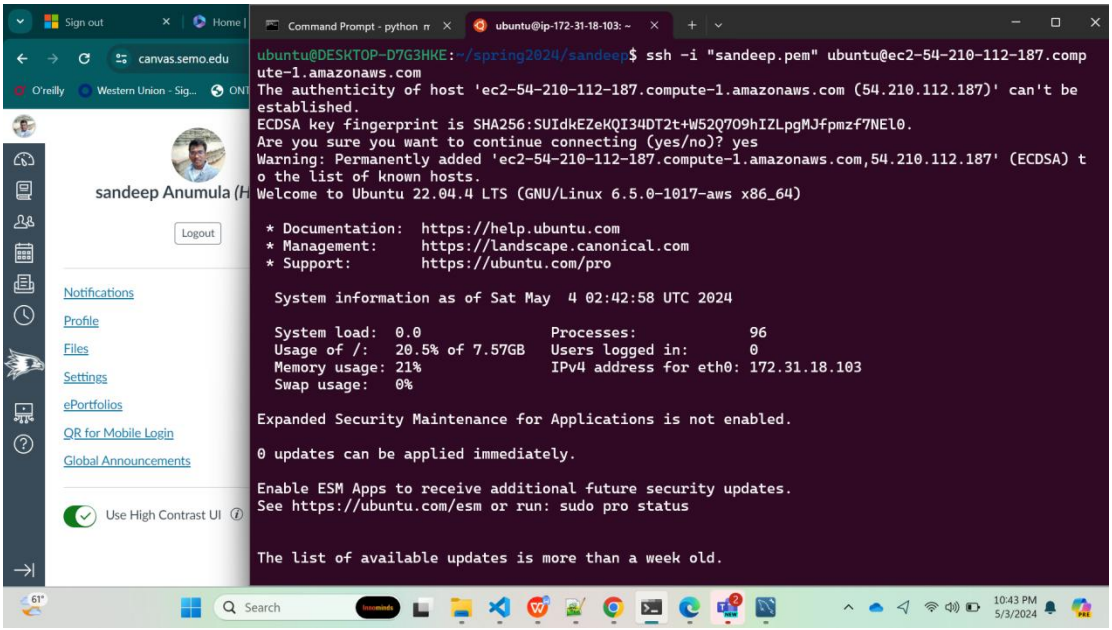
File directory cmd open



Python manage.py run server

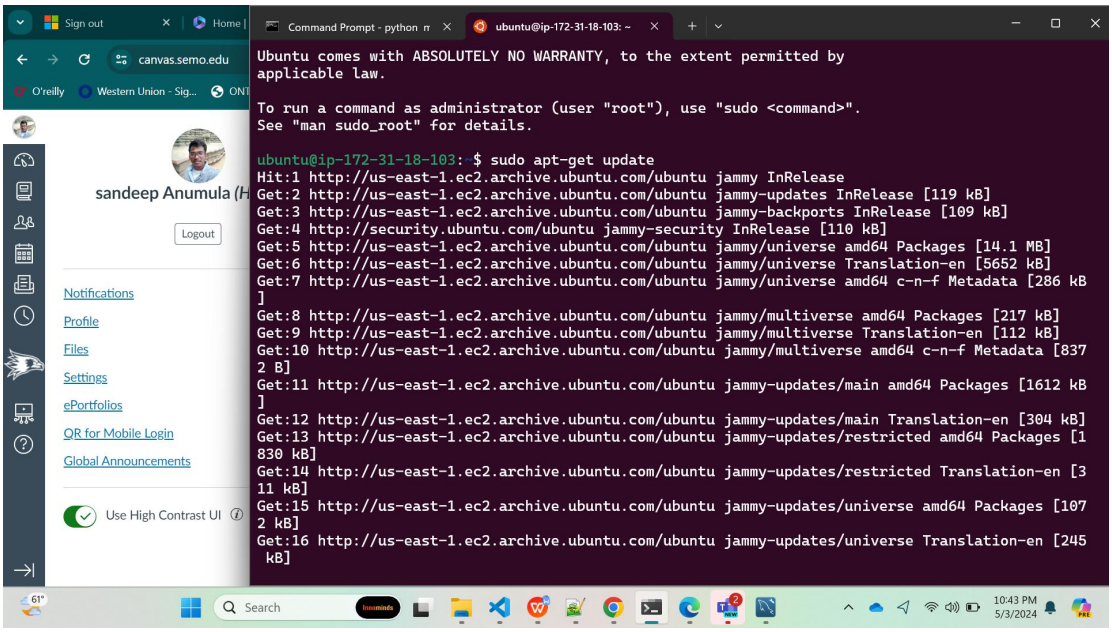


UBUNTU deployed:

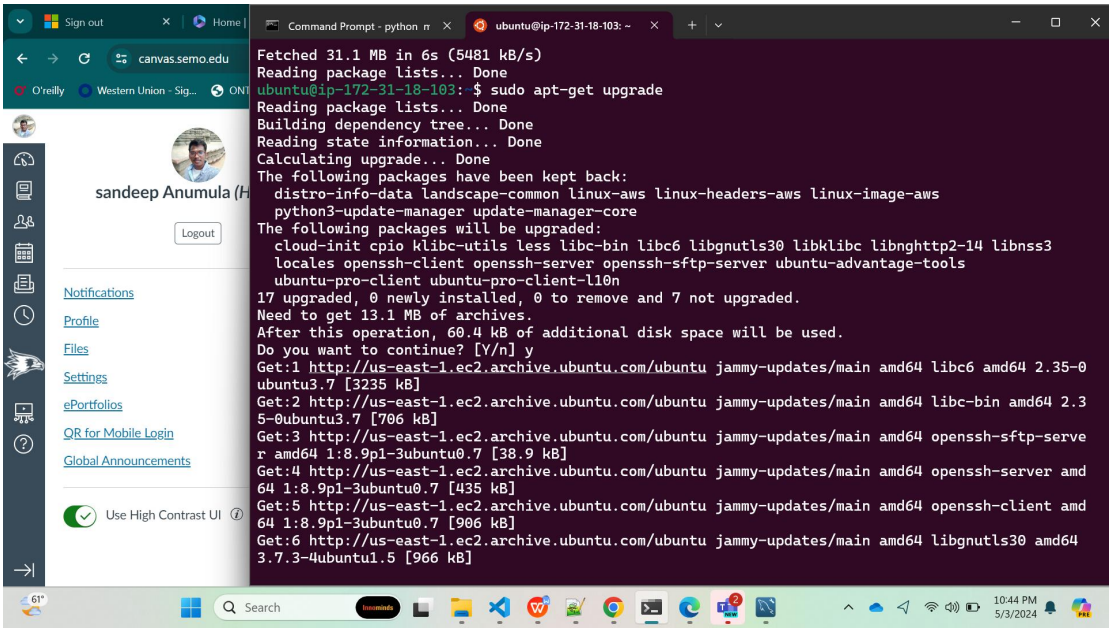


Command :sudo apt-get install

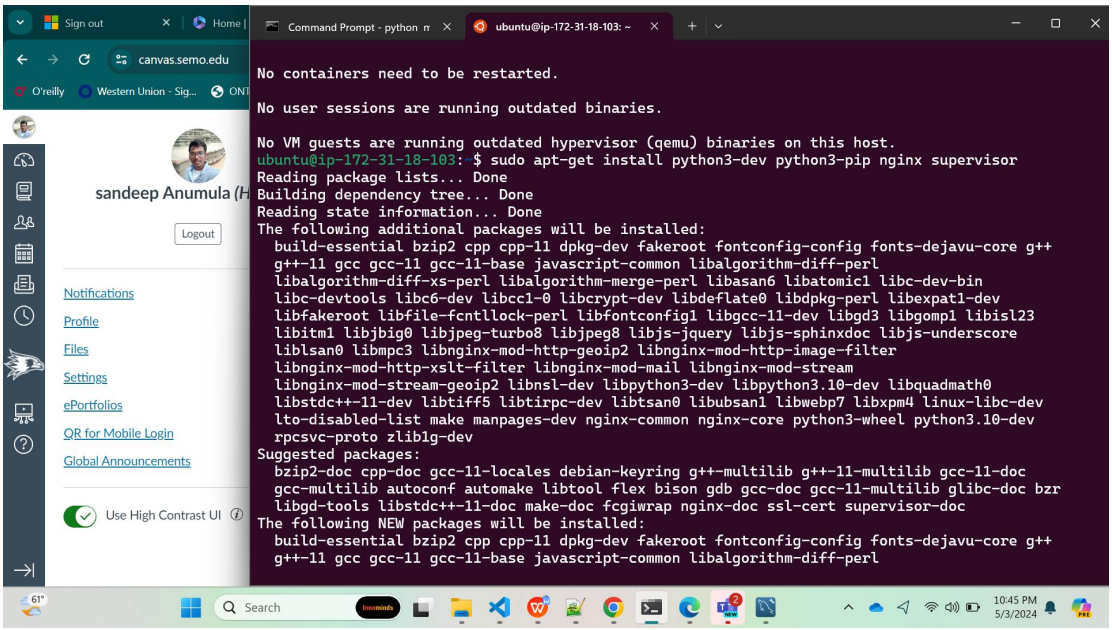
Sudo apt-get update

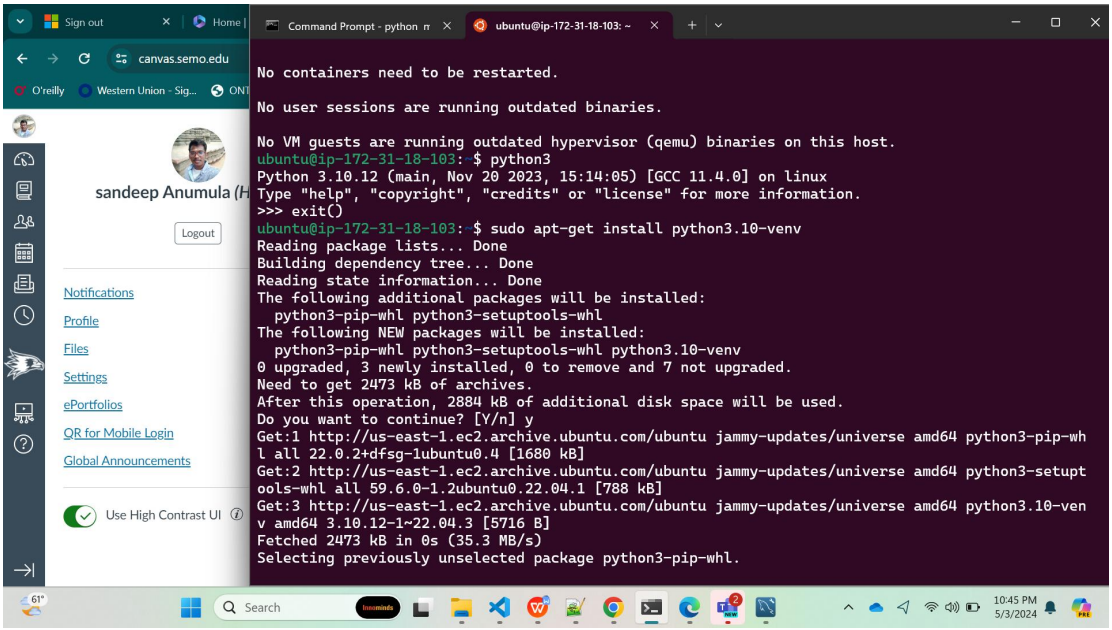


Next command sudo apt-get upgrade

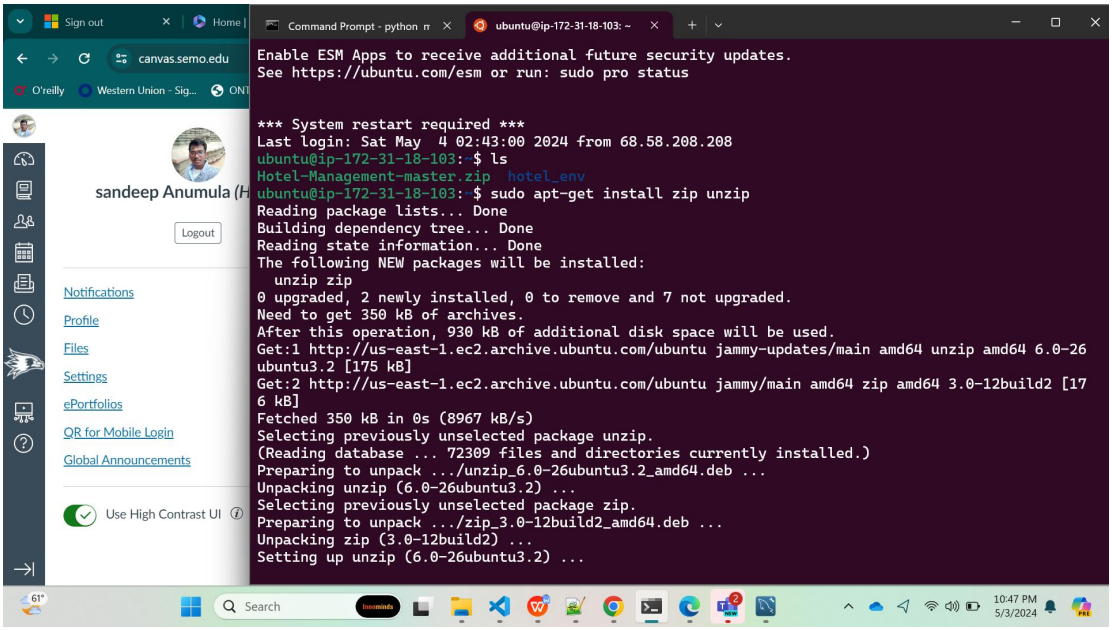


Command:sudo apt-get install supervisor nginx

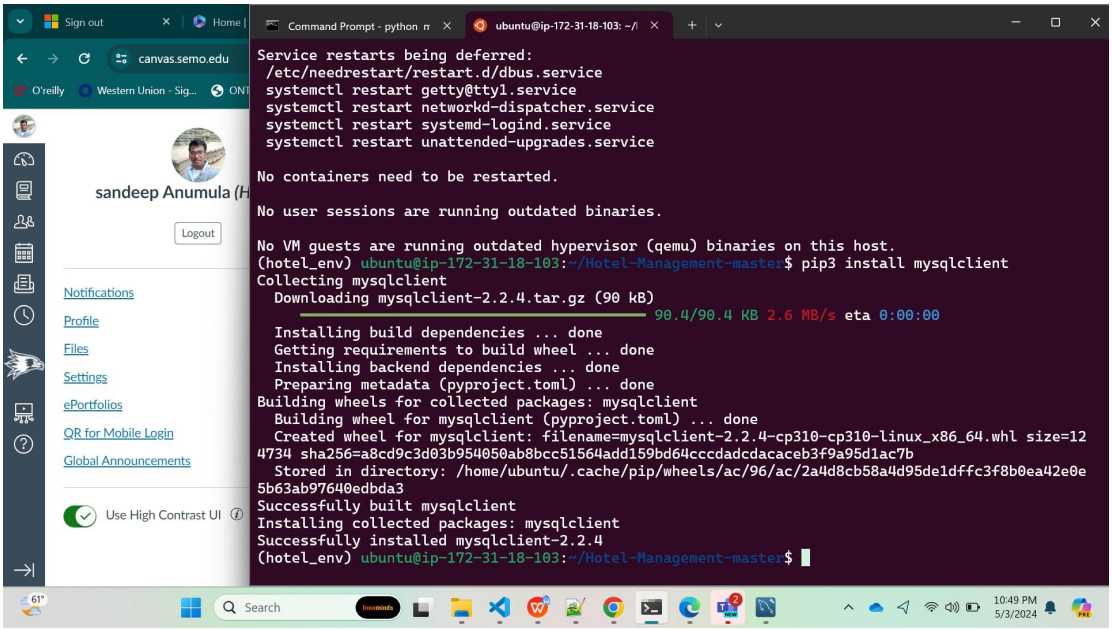


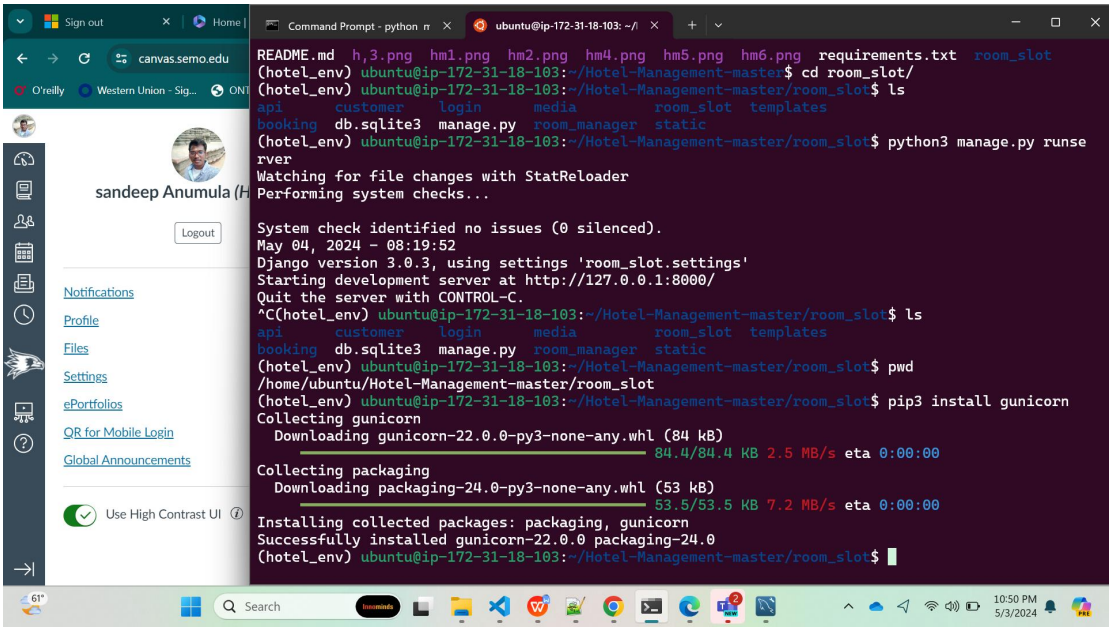


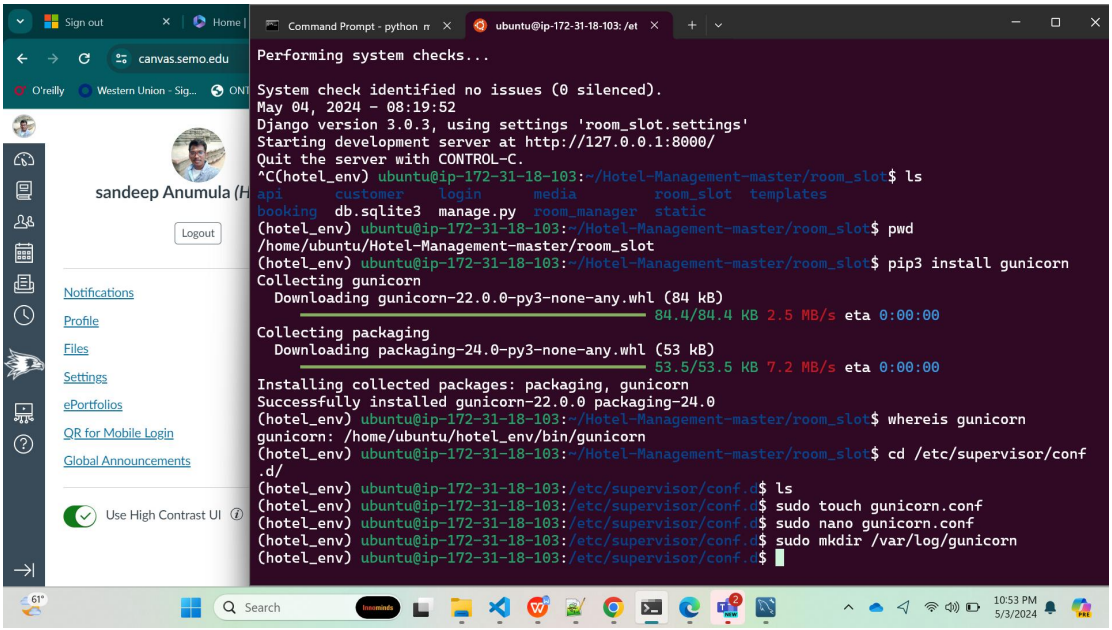
Sudo apt-get install zip unzip



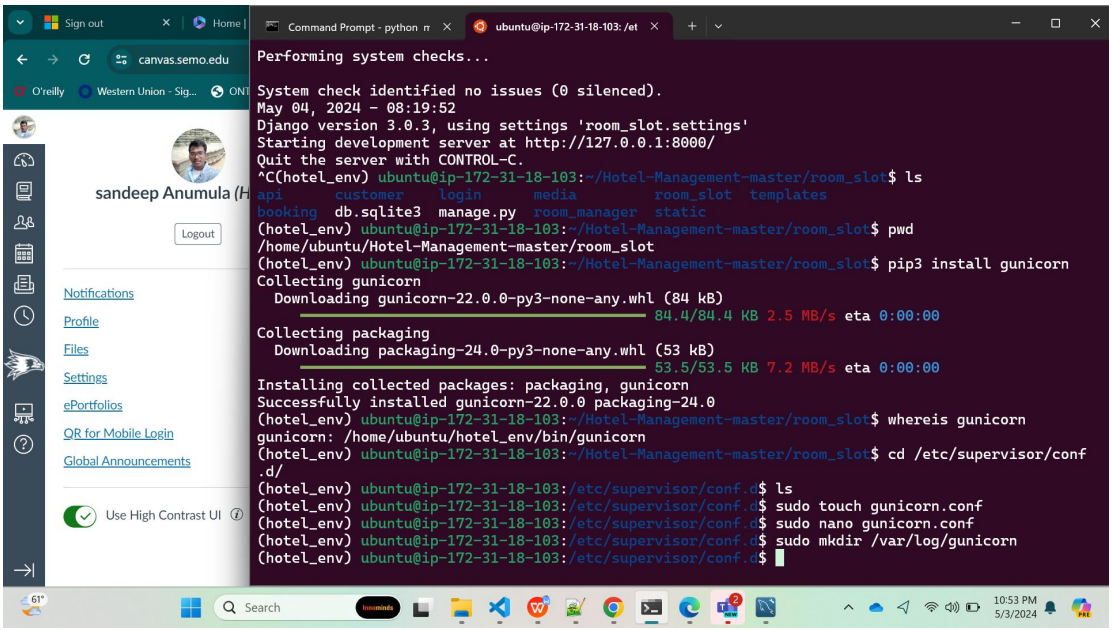
Pip3 install mysqlclient


Sign out
Home
canvas.semo.edu
O'reilly
Western Union - Sig...
sandeep Anumula (H
Logout
Notifications
Profile
Files
Settings
ePortfolios
QR for Mobile Login
Global Announcements
Use High Contrast UI
Command Prompt - python
ubuntu@ip-172-31-18-103: ~/
Service restarts being deferred:
/etc/needrestart/restart.d/dbus.service
systemctl restart getty@tty1.service
systemctl restart networkd-dispatcher.service
systemctl restart systemd-logind.service
systemctl restart unattended-upgrades.service
No containers need to be restarted.
No user sessions are running outdated binaries.
No VM guests are running outdated hypervisor (qemu) binaries on this host.
(hotel_env) ubuntu@ip-172-31-18-103:~/Hotel-Management-master$ pip3 install mysqlclient
Collecting mysqlclient
Downloading mysqlclient-2.2.4.tar.gz (90 kB)
90.4/90.4 KB 2.6 MB/s eta 0:00:00
Installing build dependencies ... done
Getting requirements to build wheel ... done
Installing backend dependencies ... done
Preparing metadata (pyproject.toml) ... done
Building wheels for collected packages: mysqlclient
Building wheel for mysqlclient (pyproject.toml) ... done
Created wheel for mysqlclient: filename=mysqlclient-2.2.4-cp310-cp310-linux_x86_64.whl size=124734 sha256=a8cd9c3d03b954050ab8bcc51564add159bd64cccdadcdacaceb3f9a95d1ac7b
Stored in directory: /home/ubuntu/.cache/pip/wheels/ac/96/ac/2a4d8cb58a4d95de1dffc3f8b0ea42e0e5b63ab97640edbda3
Successfully built mysqlclient
Installing collected packages: mysqlclient
Successfully installed mysqlclient-2.2.4
(hotel_env) ubuntu@ip-172-31-18-103:~/Hotel-Management-master$
Search
10:49 PM
5/3/2024


Sign out
Home
canvas.semo.edu
O'reilly
Western Union - Sig...
sandeep Anumula (H
Logout
Notifications
Profile
Files
Settings
ePortfolios
QR for Mobile Login
Global Announcements
Use High Contrast UI
Command Prompt - python
ubuntu@ip-172-31-18-103: ~/
README.md h,3.png hm1.png hm2.png hm4.png hm5.png hm6.png requirements.txt room_slot
(hotel_env) ubuntu@ip-172-31-18-103:~/Hotel-Management-master$ cd room_slot/
(hotel_env) ubuntu@ip-172-31-18-103:~/Hotel-Management-master/room_slot$ ls
api customer login media room_slot templates
booking db.sqlite3 manage.py room_manager static
(hotel_env) ubuntu@ip-172-31-18-103:~/Hotel-Management-master/room_slot$ python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...
System check identified no issues (0 silenced).
May 04, 2024 - 08:19:52
Django version 3.0.3, using settings 'room_slot.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
^C(hotel_env) ubuntu@ip-172-31-18-103:~/Hotel-Management-master/room_slot$ ls
api customer login media room_slot templates
booking db.sqlite3 manage.py room_manager static
(hotel_env) ubuntu@ip-172-31-18-103:~/Hotel-Management-master/room_slot$ pwd
/home/ubuntu/Hotel-Management-master/room_slot
(hotel_env) ubuntu@ip-172-31-18-103:~/Hotel-Management-master/room_slot$ pip3 install gunicorn
Collecting gunicorn
Downloading gunicorn-22.0.0-py3-none-any.whl (84 kB)
84.4/84.4 KB 2.5 MB/s eta 0:00:00
Collecting packaging
Downloading packaging-24.0-py3-none-any.whl (53 kB)
53.5/53.5 KB 7.2 MB/s eta 0:00:00
Installing collected packages: packaging, gunicorn
Successfully installed gunicorn-22.0.0 packaging-24.0
(hotel_env) ubuntu@ip-172-31-18-103:~/Hotel-Management-master/room_slot$
Search
10:50 PM
5/3/2024



Server deployement started:



Url: http://127.0.0:8000/

Deployment:

Set up hosting for the hotel management system, choosing a reliable and scalable hosting provider.

Configure the domain name and ensure proper DNS settings for the website's accessibility.

Deploy the system to the hosting environment, ensuring all components are properly configured and functioning.

Maintenance and Updates:

Regularly maintain and update the hotel management system to address any issues, enhance functionality, and improve user experience.

Manage room inventory, update pricing, and handle guest requests through the system's administrative interface.

Monitor system performance and security, implementing necessary updates and patches to ensure smooth operation.

By following these steps, a hotel management system can be effectively implemented to streamline hotel operations and enhance guest satisfaction.

Evaluation:

PROs:

Global Reach: Using assistance of hotel management systems, lodging establishments can reach a wider audience outside of their immediate neighborhood.

Convenience: By enabling guests to make reservations at any time and from any location without having to physically visit the hotel, these systems provide a convenient booking experience.

Cost-effective: By automating procedures like guest services and reservation administration, hotel management systems help lower operating expenses.

More Insights: Hotels can customize their services and marketing plans by gaining important insights into the behavior and preferences of their guests.

Personalized experiences from booking to check-out are made possible by these systems, which improves overall satisfaction.

24/7 Availability: Visitors can make reservations and enquiries at any time thanks to the hotel management systems' round-the-clock accessibility.

Scalability: These systems have the capacity to expand to

CONs:

Security Risks: Hotel management systems are vulnerable to cybersecurity threats such as data breaches and hacking, putting guest information at risk.

Customer Service Challenges: Providing quality customer service can be challenging, particularly with inquiries related to reservations, room preferences, and billing.

Operational Complexity: Managing a hotel management system requires dealing with various operational tasks, including room inventory, housekeeping schedules, and staff management.

Payment Processing Fees: Hotels incur fees for processing credit card transactions, which can impact profitability.

Lack of Personal Touch: Despite providing convenience, hotel management systems may lack the personal touch of traditional hospitality, affecting guest relationships and loyalty.

Competitive Market: The hotel industry is highly competitive, requiring hotels to continually innovate and adapt to stay ahead of competitors.

Future Works:

The submission of individualized experiences catered to the tastes and requirements of every visitor is expected to become a greater emphasis for hotel management systems.

Mobile-Friendly Solutions: To accommodate the increasing number of tourists who use smartphones, a bigger focus will be placed on mobile-friendly interfaces and mobile booking capabilities.

Integration with Emerging Technologies: Operational efficiency and guest experiences will be improved through integration with technologies like voice assistants, blockchain, and augmented reality.

Sustainability Initiatives: To satisfy the rising demand for environmentally friendly travel options, hotels will give priority

to sustainability initiatives, such as eco-friendly operations and ethical sourcing.

Upgraded Delivery Services: To provide guests with same-day service and amenity delivery, hotels can investigate forming alliances with nearby couriers and deploying technology.

Constant Innovation: To stay competitive, hotels must make constant investments in new ideas and adjust meet changing demands from visitors and market trends.

Conclusion: -

To sum up, the Hotel Management System is a solid solution created as a component of the curriculum for Distributed Cloud Computing. A scalable, secure, and high-performing hotel management platform is created by this project, which showcases the effective use of modern web development techniques and cloud technologies. The system's goals are to increase operational effectiveness, boost guest satisfaction, and satisfy the changing demands of the hospitality sector by integrating cutting-edge features and seamless user experiences.