KEY FEATURES:

- **User Management**:

- **Registration and Login**: Users can sign up and log in securely using an integrated authentication system. AWS Cognito or Flask-Login ensures data protection and seamless session management.
- **Role-Based Access**: Separate dashboards and functionalities are tailored for students, teachers, and administrators.

- **Course Materials Management**:

- **S3 Integration**: Course content, including lecture notes, videos, and other educational materials, are uploaded and stored in Amazon S3. S3 ensures high availability and secure access to these resources.
- **Version Control**: Educators can update course materials, and students always have access to the latest versions.

- **Data Management**:

- **Amazon RDS**: A relational database setup on RDS stores user information, course metadata, and other critical data. The database schema is designed for optimal performance and scalability.
- **Secure Queries**: SQLAlchemy, the ORM used with Flask, ensures safe interactions with the database.

- **Deployment and Scalability**:

- **AWS EC2**: The Flask application is deployed on an EC2 instance, providing the flexibility to scale resources based on user demand.
- **Load Balancing**: Elastic Load Balancing (ELB) ensures even distribution of traffic across instances.

- **Real-Time Communication**:

- Using AWS Chime SDK or WebRTC (optional for future development), the platform supports live lectures, discussions, and virtual office hours.

- **Analytics and Monitoring**:

- **AWS CloudWatch**: Real-time monitoring tracks system health and performance.

- **Data Analytics**: AWS QuickSight generates insights on student progress, attendance, and course engagement metrics.

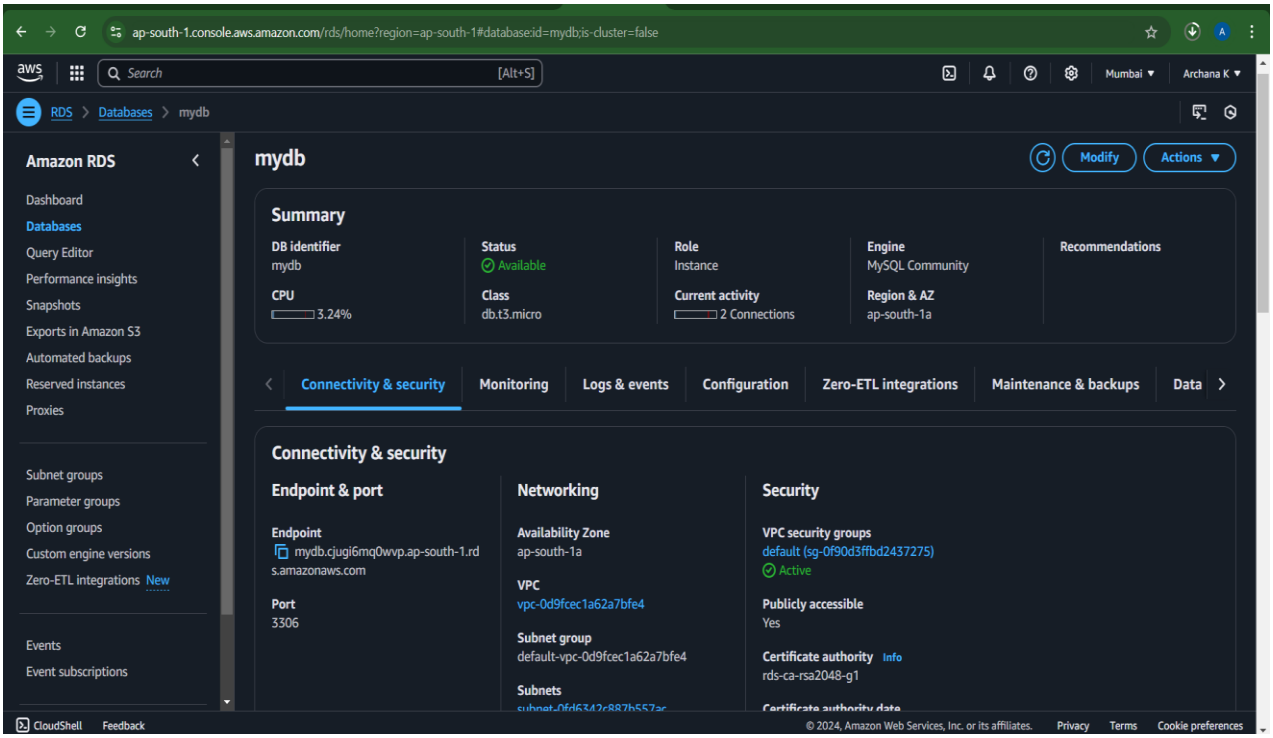
- **Security**:

- End-to-end encryption ensures secure transmission of sensitive data.
- Role-based IAM policies regulate access to AWS resources.

Step by Step Process of Implementation :

Step 1: Log in to AWS Management Console

- Navigate to the AWS Management console.
- Sign in with your AWS credentials.
- In the AWS Management Console, search for RDS in the search bar.
- Click on RDS to open the service dashboard.
- On the RDS dashboard, click Create database.
- Standard create: Offers full control over configuration options.
- Easy create: Automates most configuration decisions for quicker setup.
- For full customization, select Standard create.
- Choose your preferred database engine, such as: MySQL
- Select the version of the database engine.
- Deployment Option: Choose between Production or Dev/Test.
- DB Instance Identifier: Provide a name for the database instance.
- Master Username: Set the admin username (e.g., admin).
- Master Password: Set a strong password for the admin user.
- DB Instance Class: Select the instance size based on performance needs (e.g., db.t3.micro for free tier-eligible setups).
- General Purpose SSD (gp2)
- Provisioned IOPS SSD (io1)
- Magnetic
- Set allocated storage size (e.g., 20 GB).
- Select a VPC (Virtual Private Cloud) or allow RDS to create one automatically.
- Configure Public Access:
- Yes if the database needs to be accessed over the internet.
- No for internal access only.
- Choose or create a Subnet Group for high availability.
- Set VPC Security Groups to control inbound and outbound traffic.

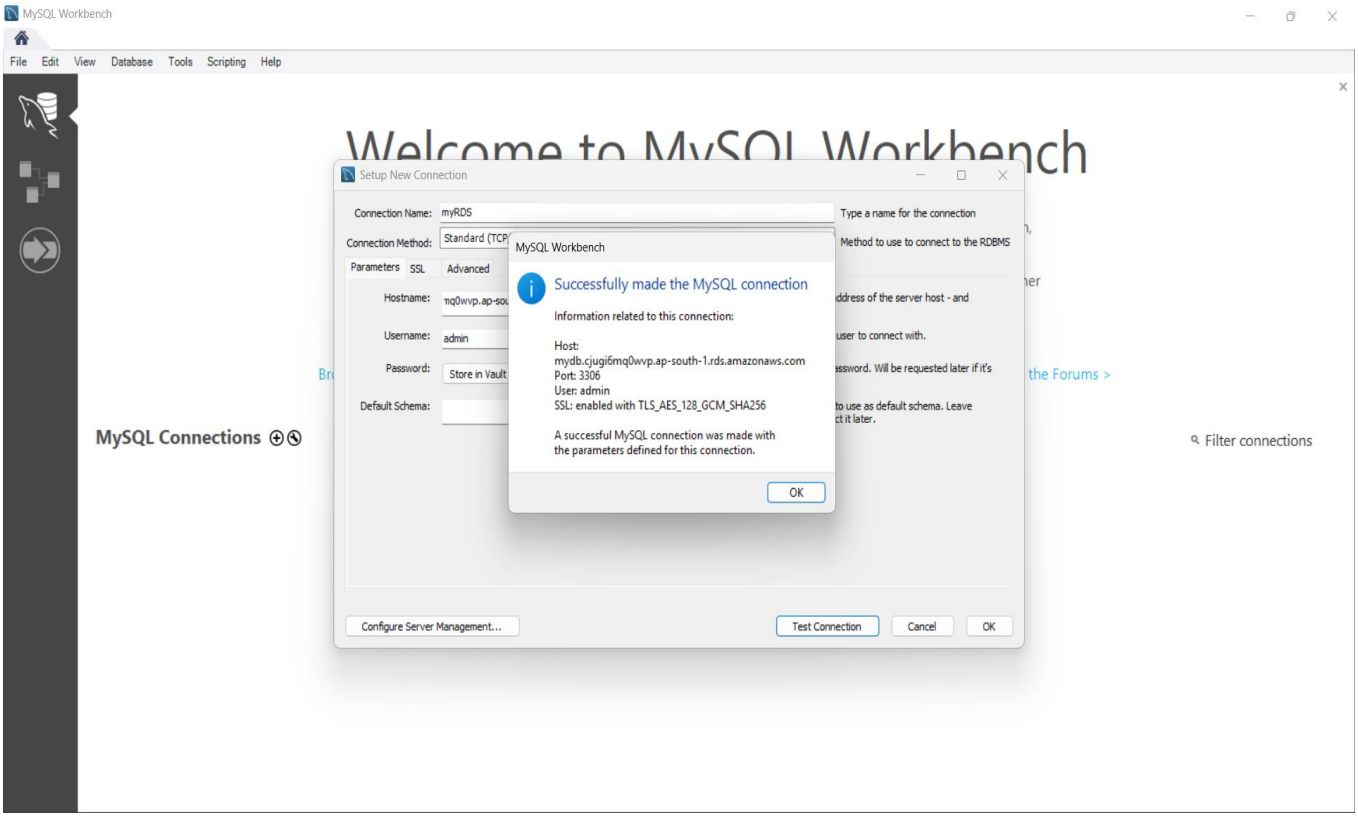


Step 2 : Connect to MySQL Workbench.

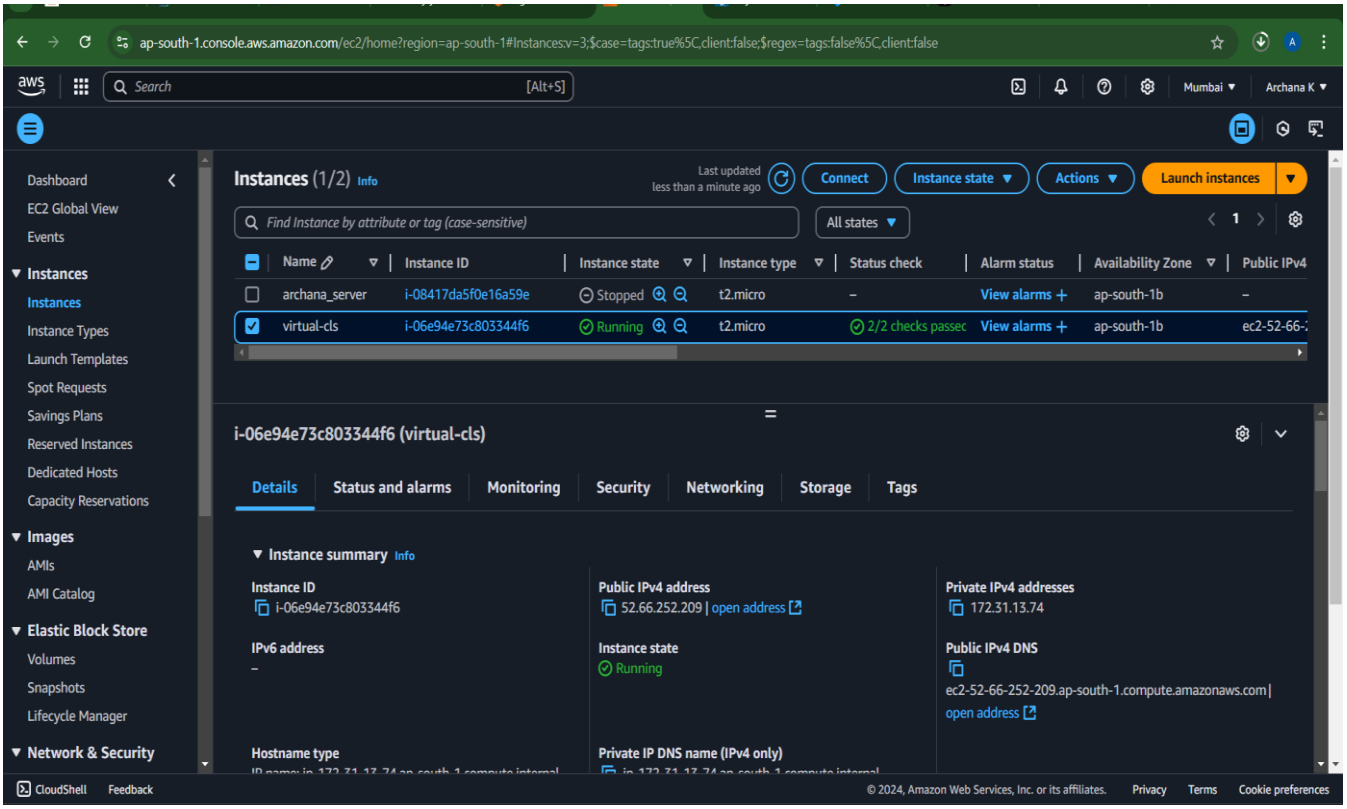
- Download and install **MySQL Workbench** from the official MYSQL website.
- Launch MySQL Workbench after installation.
- Log in to the AWS Management Console.
- Navigate to the **RDS** dashboard.
- Select your MySQL RDS instance from the list.
- Locate the **Endpoint** and **Port** in the instance details.
  - Example: `mydb-instance.abc123xyz.us-east-1.rds.amazonaws.com`
  - Port: Default is `3306`.
- In the AWS Management Console, go to **EC2 > Security Groups**.
- Find the security group associated with your RDS instance.
- Add an inbound rule to allow your IP address to access the RDS instance:
  - **Type**: MySQL/Aurora
  - **Protocol**: TCP

- **Port Range**: 3306
- **Source**: Your IP (use **My IP** option for simplicity).

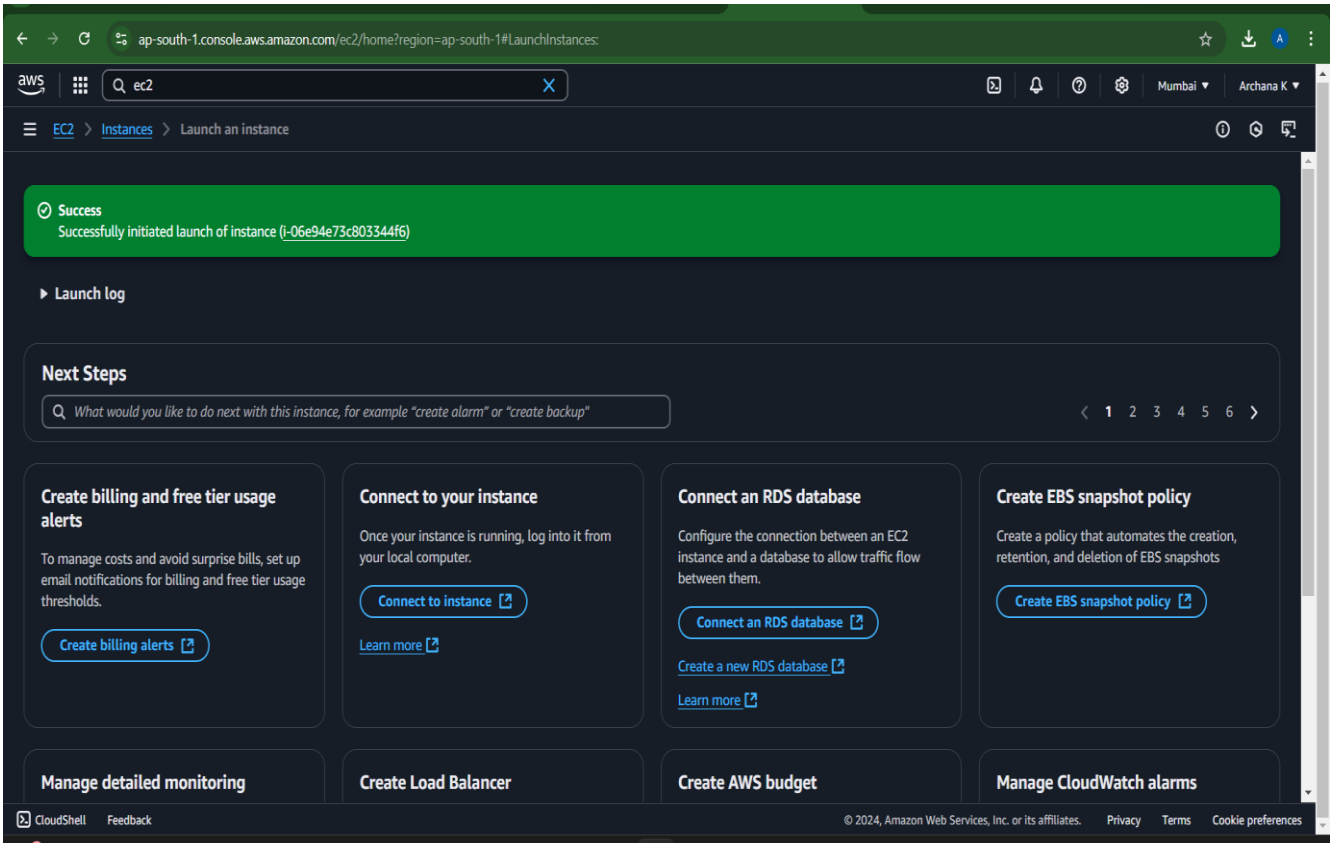
- Save the changes.
- Open **MySQL Workbench**.
- Click the + symbol next to "MySQL Connections" to create a new connection.
- **Connection Name**: Provide a name for the connection (e.g., `AWS RDS MySQL`).
- **Hostname**: Enter the **Endpoint** of your RDS instance (e.g., `mydb-instance.abc123xyz.us-east-1.rds.amazonaws.com`).
- **Port**: Enter the port number (default: `3306`).
- **Username**: Enter the master username you configured for your RDS instance.
- **Password**:
  - Click **Store in Vault** or **Store Password** and enter the master password.
- Click the **Test Connection** button.
- If prompted, select the appropriate authentication method (default is **Standard**).
- If the connection is successful, a confirmation dialog will appear.
- Click **OK** to save the connection.
- Double-click the new connection in MySQL Workbench to connect to your RDS instance.
- Once connected, you can:
  - View existing databases.
  - Run SQL queries.
  - Perform administrative tasks.



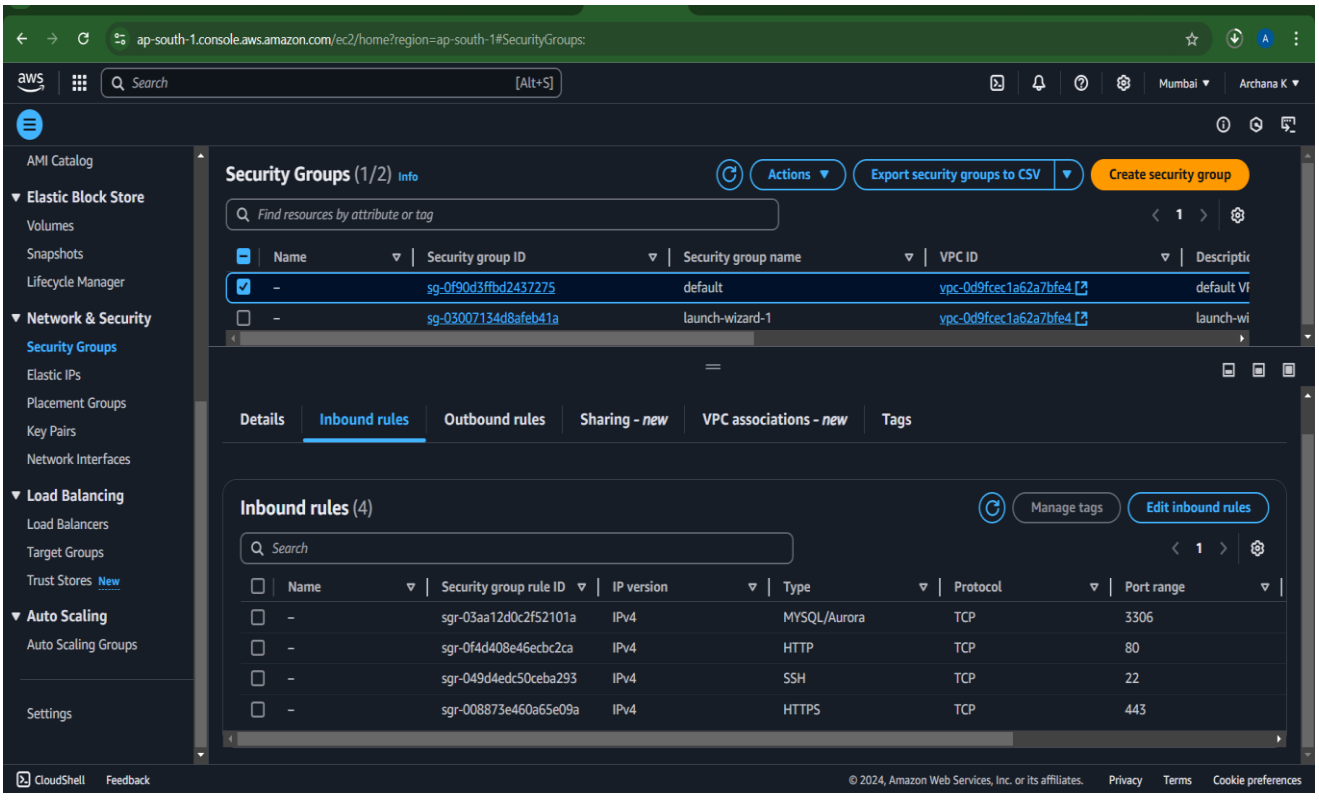
Step 3 : Create an EC2 Instance



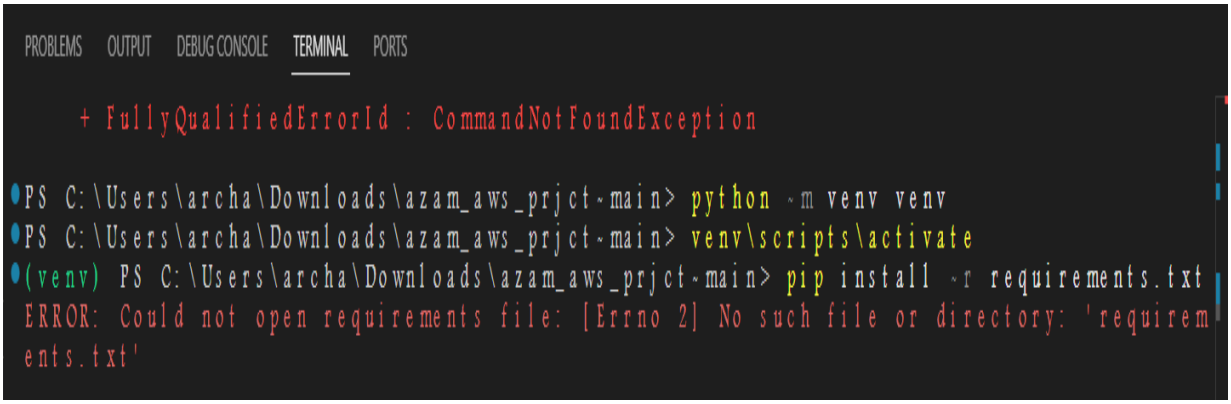
Step 4 : EC2 instance successfully created

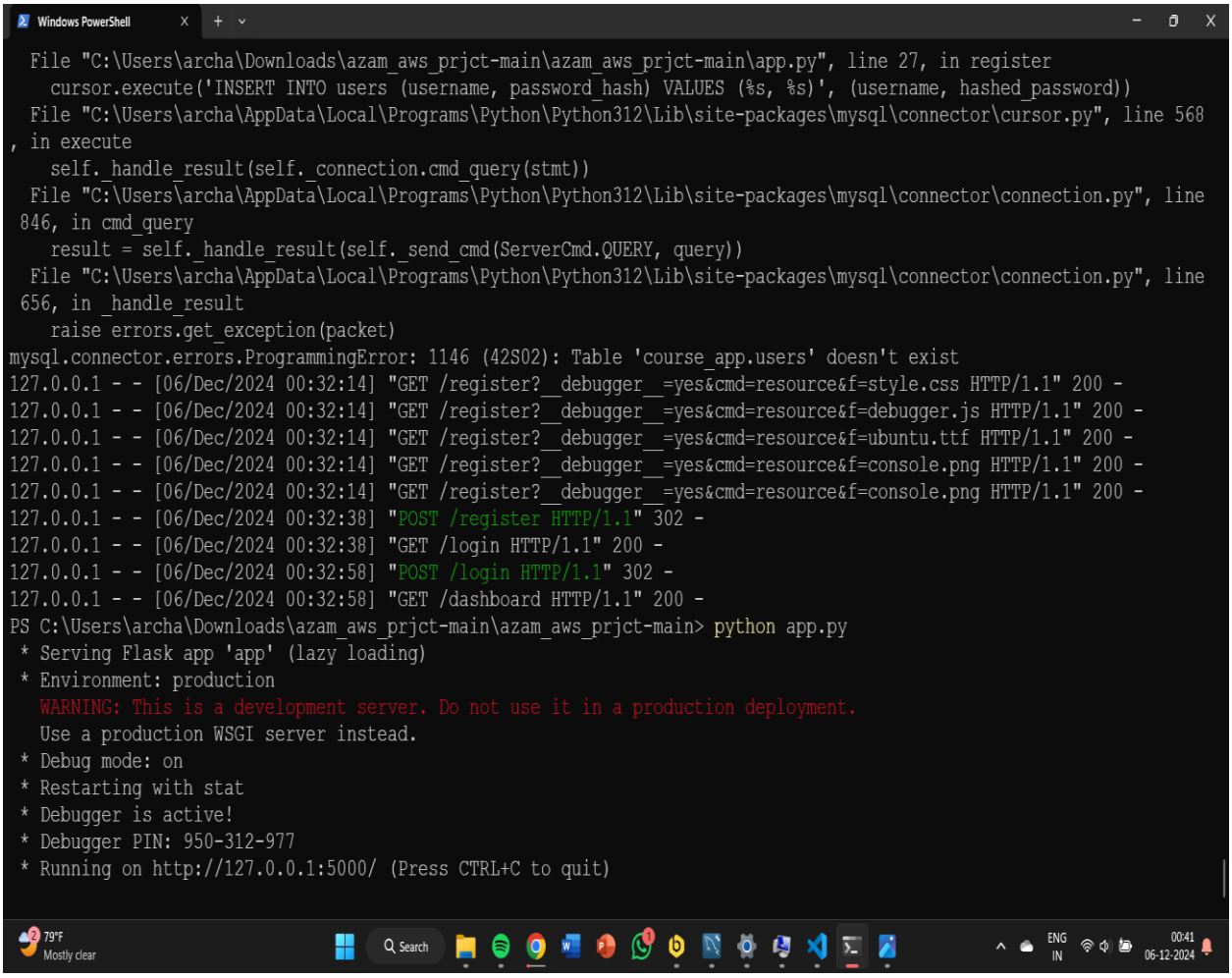


Step 5 : Choose instance type, configure security groups, and key pair

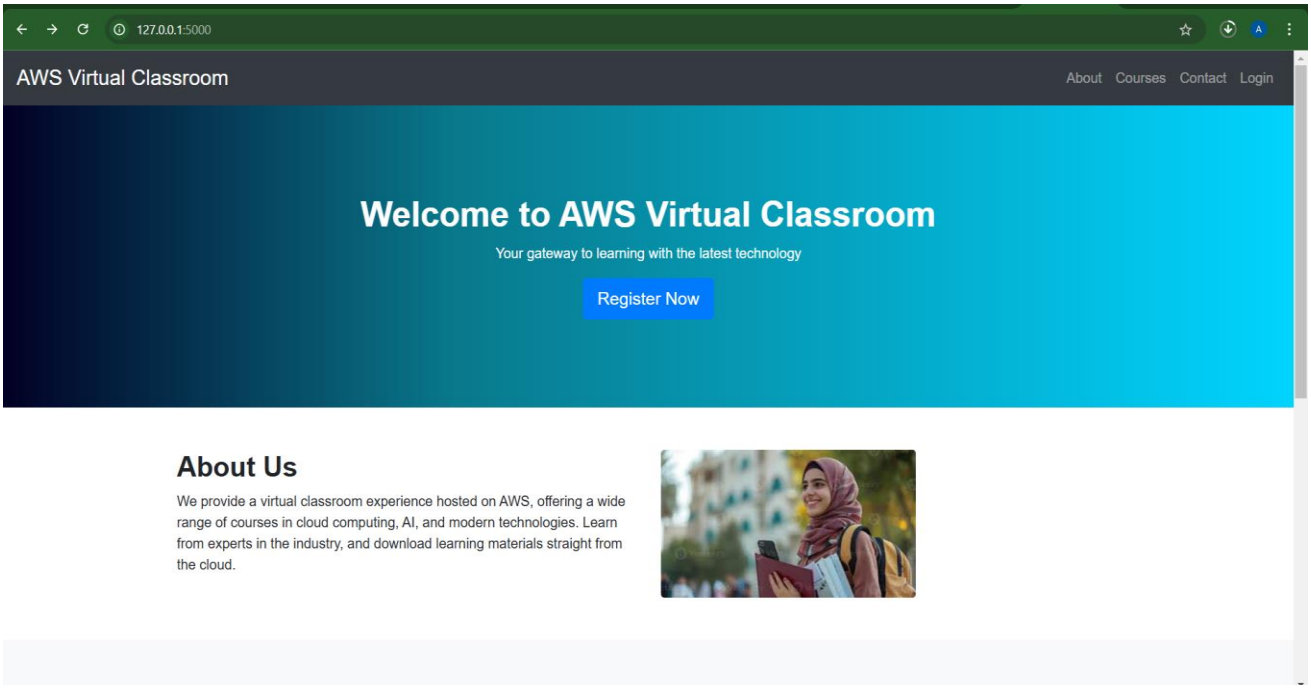


Step 6 : Deploy Flask App on EC2

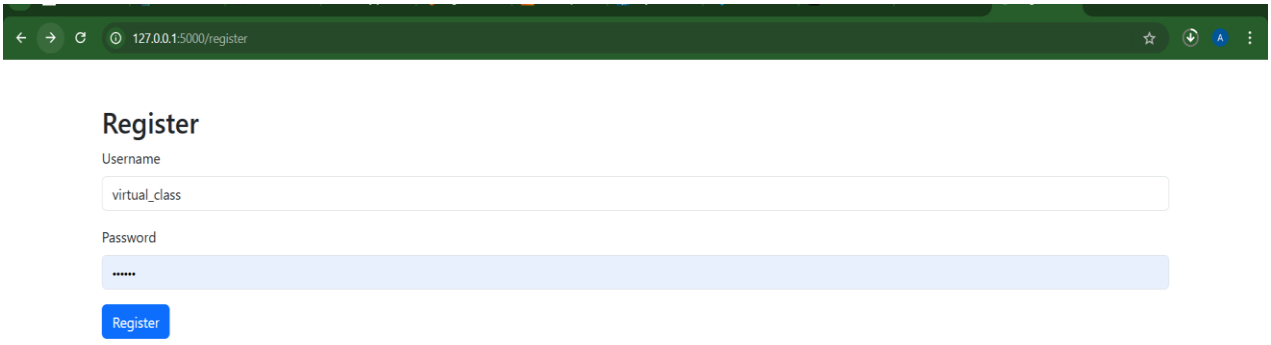




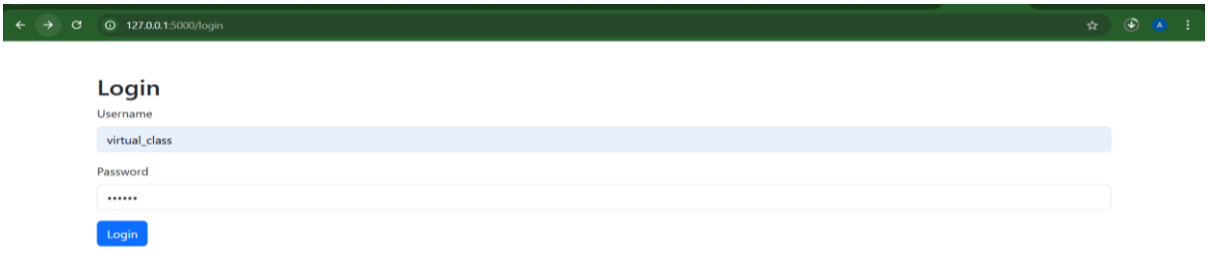
Step 7 : Display the home page of the website .



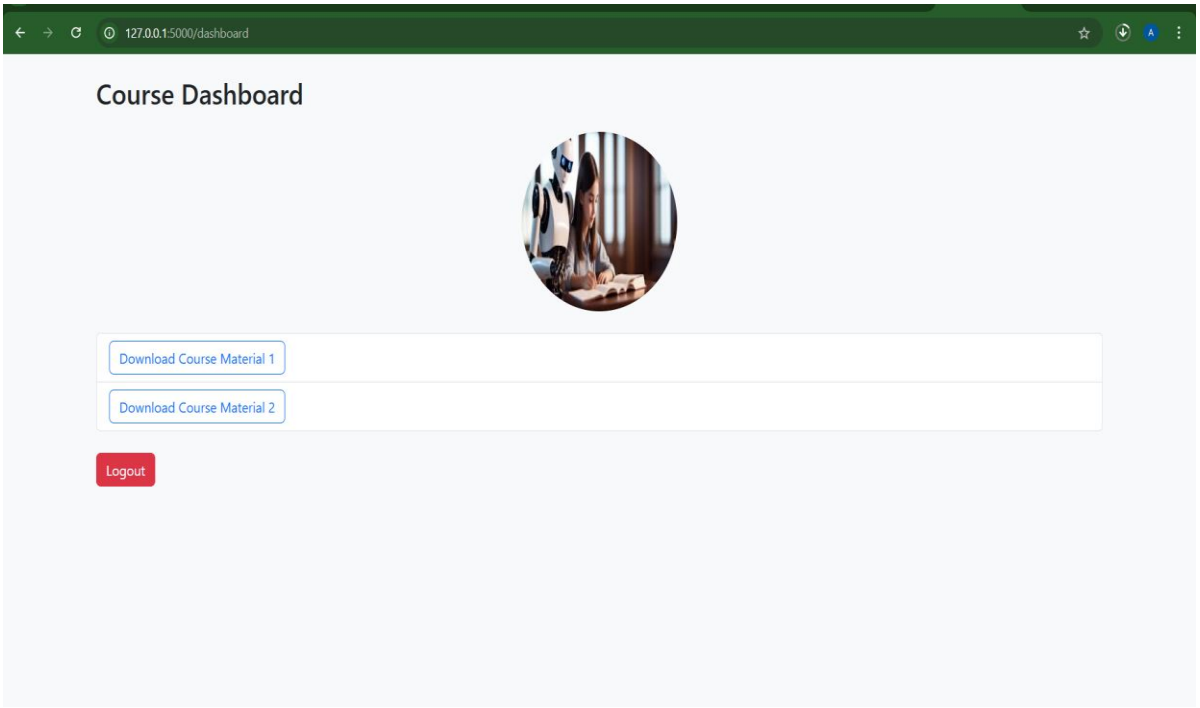
Step 8: Student Registration and Access



Step 9 : User login



Step 10 : Instructor Uploads Materials



Step 11: Downloading Course Content