

## 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:**

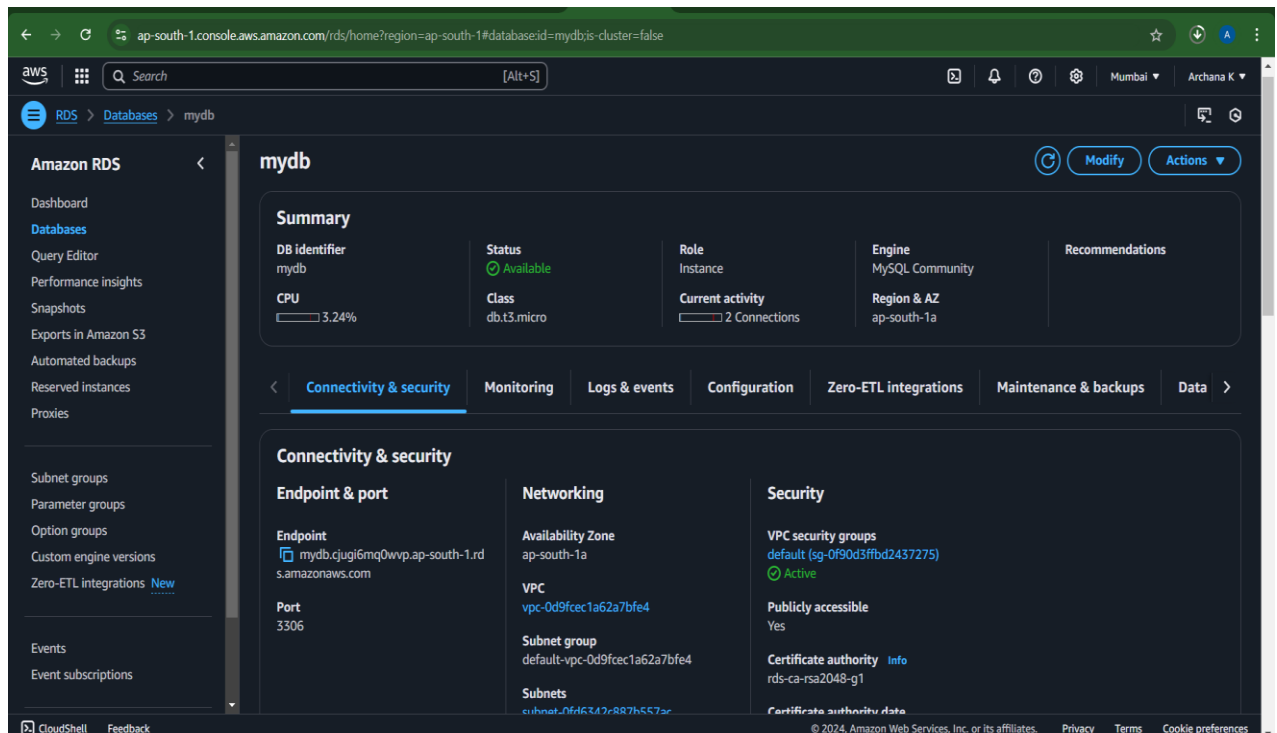
- **AWS CloudWatch:** Real-time monitoring tracks system health and performance.

- **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**

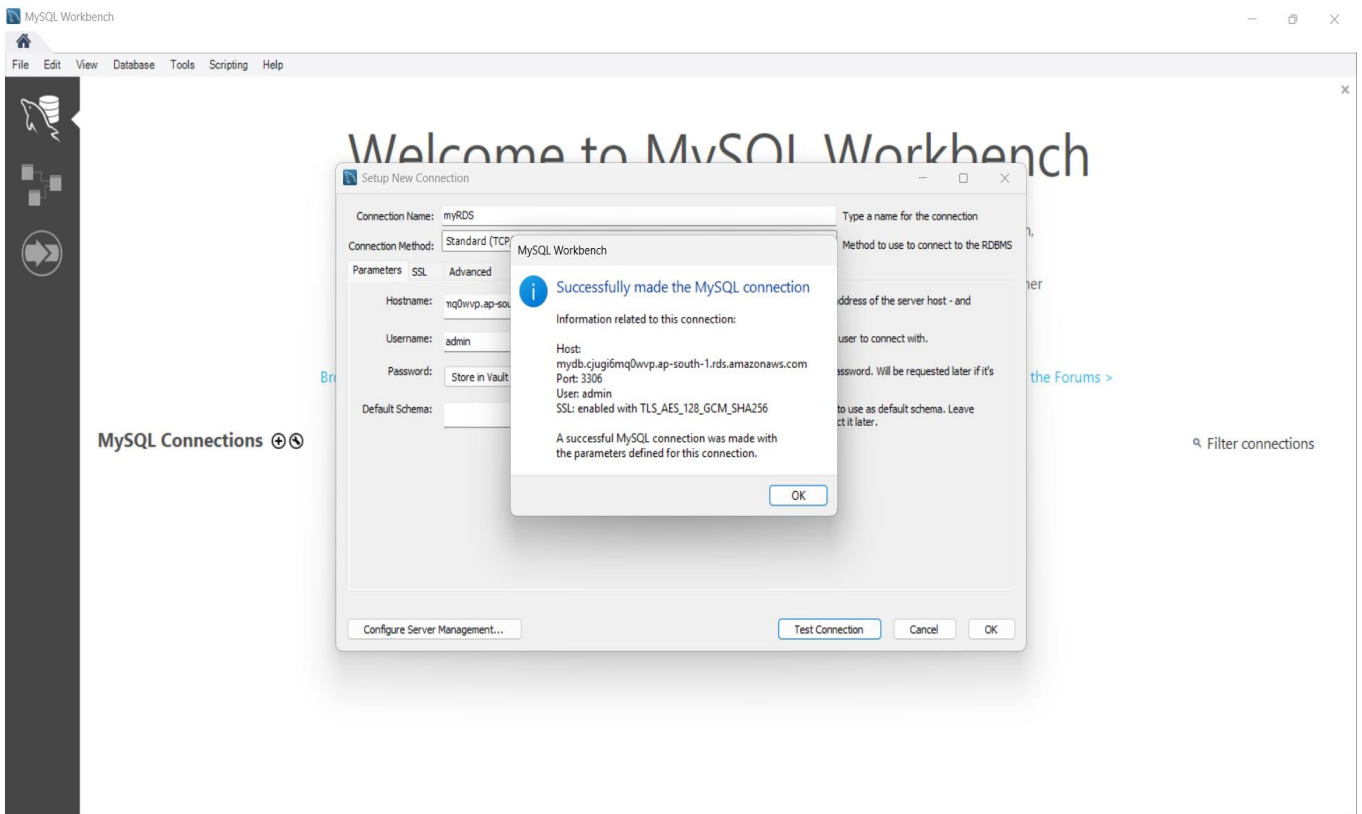
- Navigate to the AWS Management console.
- Sign in with your AWS credentials.
- In the AWS Management Console, search for RDS in the search bar.
- Click on RDS to open the service dashboard.
- On the RDS dashboard, click Create database.
- Standard create: Offers full control over configuration options.
- 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.



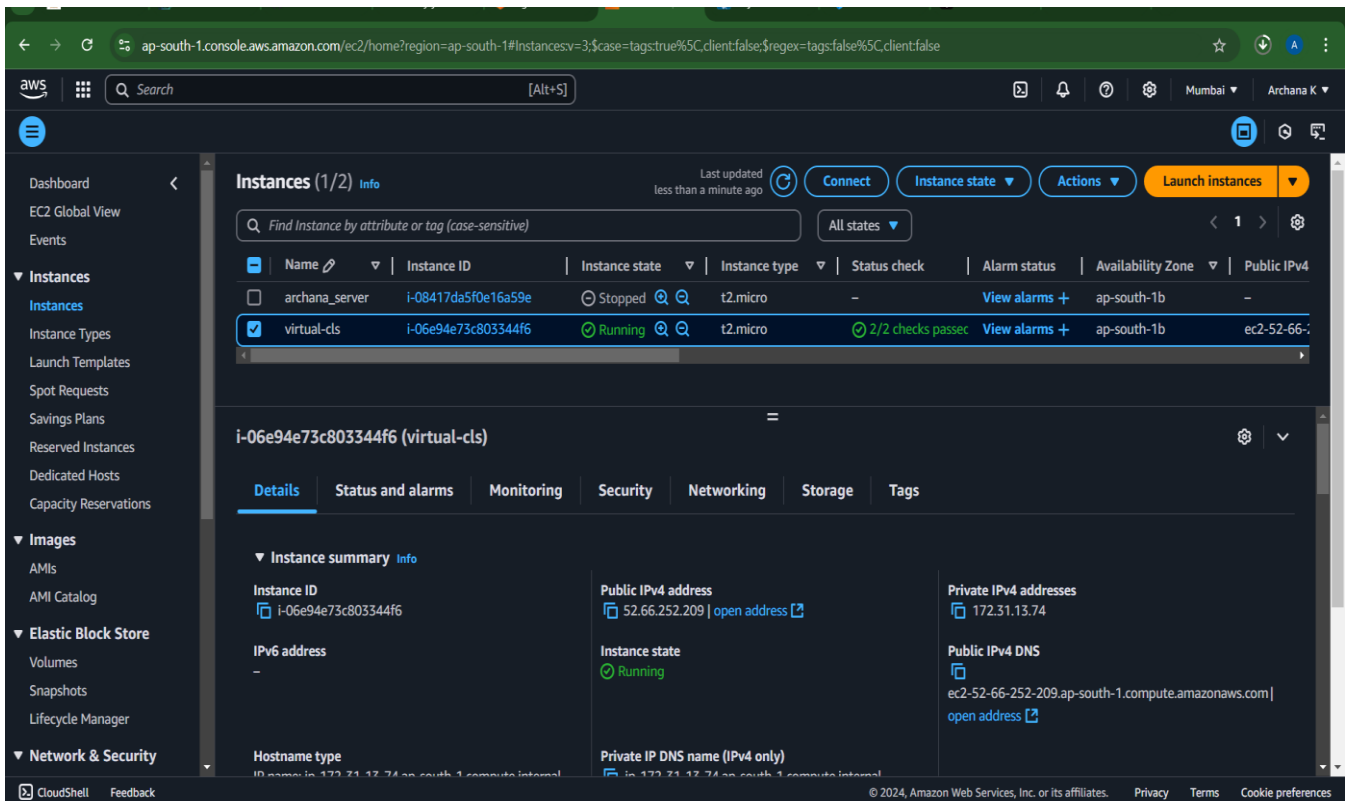
## Step 2 : Connect to MySQL Workbench.

- 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.abcd123xyz.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

The screenshot shows the AWS Management Console in the 'ap-south-1' region. The breadcrumb navigation indicates the path: EC2 > Instances > Launch an instance. A green success banner at the top states: 'Success Successfully initiated launch of instance (i-06e94e73c803344f6)'. Below this is a 'Launch log' section. The 'Next Steps' section contains a search bar and several recommended actions: 'Create billing and free tier usage alerts', 'Connect to your instance', 'Connect an RDS database', and 'Create EBS snapshot policy'. At the bottom, there are links for 'Manage detailed monitoring', 'Create Load Balancer', 'Create AWS budget', and 'Manage CloudWatch alarms'.

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

The screenshot shows the 'Security Groups' page in the AWS Management Console for the 'ap-south-1' region. The left sidebar shows the navigation menu with 'Network & Security' > 'Security Groups' selected. The main content area shows a list of security groups:

Name	Security group ID	Security group name	VPC ID	Description
-	sg-0f90d3ffbd2437275	default	vpc-0d9fcec1a62a7bfe4	default VPC
-	sg-03007134d8afeb41a	launch-wizard-1	vpc-0d9fcec1a62a7bfe4	launch-wizard-1

Below the list, the 'Inbound rules' tab is selected, showing a table of rules:

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-03aa12d0c2f52101a	IPv4	MySQL/Aurora	TCP	3306
-	sgr-0f4d408e46ebc2ca	IPv4	HTTP	TCP	80
-	sgr-049d4edc50ceba293	IPv4	SSH	TCP	22
-	sgr-008873e460a65e09a	IPv4	HTTPS	TCP	443

## Step 6 : Deploy Flask App on EC2

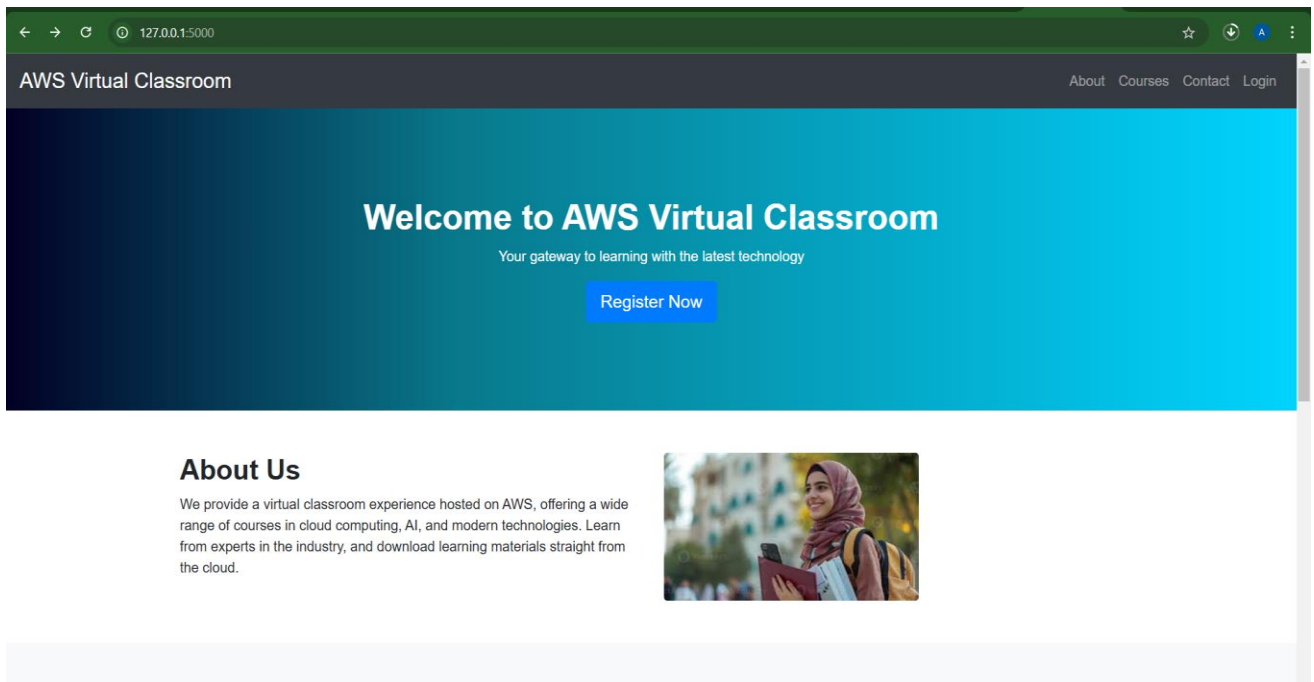
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

+ FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\archa\Downloads\azam_aws_prjct-main> python -m venv venv
PS C:\Users\archa\Downloads\azam_aws_prjct-main> venv\scripts\activate
(venv) PS C:\Users\archa\Downloads\azam_aws_prjct-main> pip install -r requirements.txt
ERROR: Could not open requirements file: [Errno 2] No such file or directory: 'requirements.txt'
```

```
Windows PowerShell
File "C:\Users\archa\Downloads\azam_aws_prjct-main\azam_aws_prjct-main\app.py", line 27, in register
    cursor.execute('INSERT INTO users (username, password hash) VALUES (%s, %s)', (username, hashed_password))
File "C:\Users\archa\AppData\Local\Programs\Python\Python312\Lib\site-packages\mysql\connector\cursor.py", line 568
, in execute
    self._handle_result(self._connection.cmd_query(stmt))
File "C:\Users\archa\AppData\Local\Programs\Python\Python312\Lib\site-packages\mysql\connector\connection.py", line
846, in cmd_query
    result = self._handle_result(self._send_cmd(ServerCmd.QUERY, query))
File "C:\Users\archa\AppData\Local\Programs\Python\Python312\Lib\site-packages\mysql\connector\connection.py", line
656, in _handle_result
    raise errors.get_exception(packet)
mysql.connector.errors.ProgrammingError: 1146 (42S02): Table 'course_app.users' doesn't exist
127.0.0.1 - - [06/Dec/2024 00:32:14] "GET /register?__debugger__=yes&cmd=resource&f=style.css HTTP/1.1" 200 -
127.0.0.1 - - [06/Dec/2024 00:32:14] "GET /register?__debugger__=yes&cmd=resource&f=debugger.js HTTP/1.1" 200 -
127.0.0.1 - - [06/Dec/2024 00:32:14] "GET /register?__debugger__=yes&cmd=resource&f=ubuntu.ttf HTTP/1.1" 200 -
127.0.0.1 - - [06/Dec/2024 00:32:14] "GET /register?__debugger__=yes&cmd=resource&f=console.png HTTP/1.1" 200 -
127.0.0.1 - - [06/Dec/2024 00:32:14] "GET /register?__debugger__=yes&cmd=resource&f=console.png HTTP/1.1" 200 -
127.0.0.1 - - [06/Dec/2024 00:32:38] "POST /register HTTP/1.1" 302 -
127.0.0.1 - - [06/Dec/2024 00:32:38] "GET /login HTTP/1.1" 200 -
127.0.0.1 - - [06/Dec/2024 00:32:58] "POST /login HTTP/1.1" 302 -
127.0.0.1 - - [06/Dec/2024 00:32:58] "GET /dashboard HTTP/1.1" 200 -
PS C:\Users\archa\Downloads\azam_aws_prjct-main\azam_aws_prjct-main> python app.py
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 950-312-977
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Step 7 : Display the home page of the website .



Step 8: Student Registration and Access

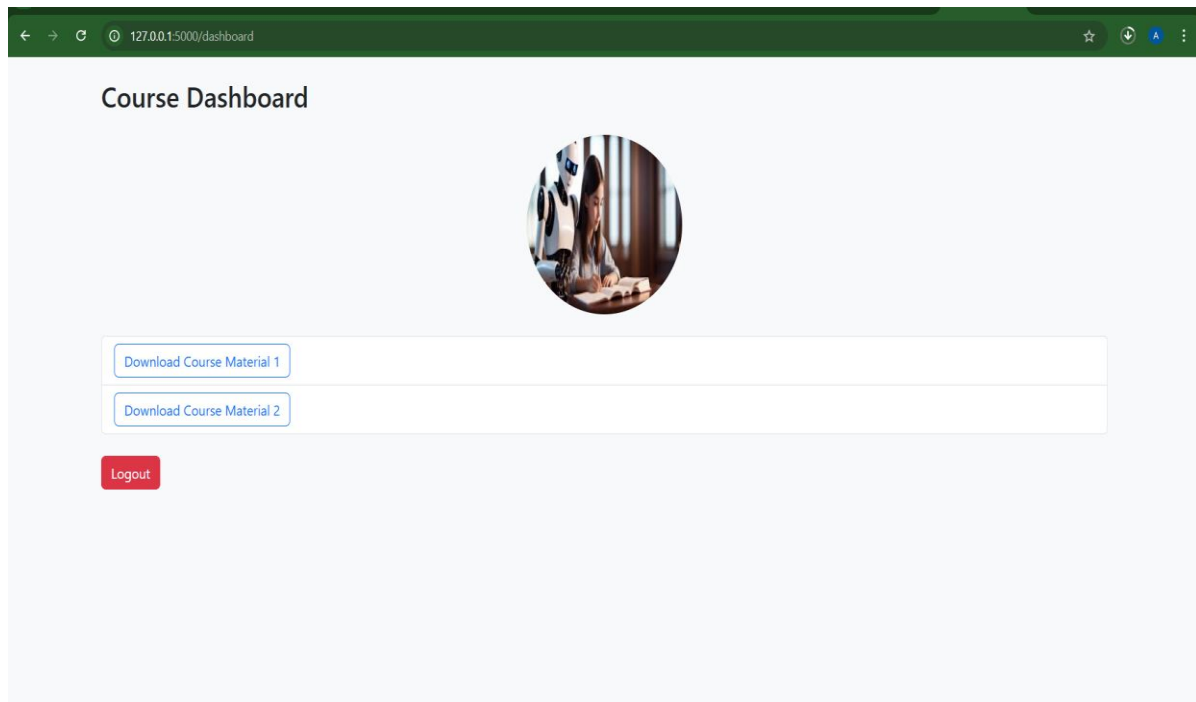
A screenshot of a web browser displaying the registration page of 'AWS Virtual Classroom'. The browser's address bar shows '127.0.0.1:5000/register'. The page has a dark blue header with the site name 'AWS Virtual Classroom' on the left and navigation links 'About', 'Courses', 'Contact', and 'Login' on the right. The main content area features a large blue gradient banner with the text 'Welcome to AWS Virtual Classroom' and the subtitle 'Your gateway to learning with the latest technology'. A blue 'Register Now' button is centered below the banner. Below the banner, there is an 'About Us' section with a paragraph of text and a small image of a woman in a hijab holding a tablet. The page is otherwise empty.

Step 9 : User login

A screenshot of a web browser displaying the login page of 'AWS Virtual Classroom'. The browser's address bar shows '127.0.0.1:5000/login'. The page has a dark blue header with the site name 'AWS Virtual Classroom' on the left and navigation links 'About', 'Courses', 'Contact', and 'Login' on the right. The main content area features a large blue gradient banner with the text 'Welcome to AWS Virtual Classroom' and the subtitle 'Your gateway to learning with the latest technology'. A blue 'Register Now' button is centered below the banner. Below the banner, there is an 'About Us' section with a paragraph of text and a small image of a woman in a hijab holding a tablet. The page is otherwise empty.



## Step 10 : Instructor Uploads Materials



## Step 11: Downloading Course Content

