MIDDLE EAST TECHNICAL UNIVERSITY
NORTHERN CYPRUS CAMPUS

Computer Engineering Program

CNG 495 – Cloud Computing

Term Project Progress Report

FALL 2023

Prepared by:

Ece Erseven - 2385383

Egemen Aksöz - 2315083

Table of Contents

Table of Figures

Explanation of the project

Project Name: Second Hand METU

Topic: Second-Hand Web Application for METU Students

The web application for METU students aims to set up a dynamic and sustainable marketplace where users can share and discover second-hand items. Users will register and log in securely, with their identities verified, to ensure the authenticity of the listings. The platform will designate users to post, manage, and update detailed advertisements for their second-hand items, allowing for easy editing, updating, and removal of listings. Users can post an item to sell by providing the item name, price, category, condition (which could range from descriptions like "lightly used" to "brand new"), and price. Providing detailed information about the item's condition is crucial. It will help buyers understand its state and boost clearness and trust. Advanced search options based on item name and price will improve the user experience for buyers. For instance, users can effortlessly find specific items by entering the item name into the search bar, and the platform will display all relevant adverts associated with that item, simplifying the search experience for the user. Additionally, users can pick a price range and search for an item within that price range. The web app will also show user profiles, individual listings, and a review system, encouraging trust-building among buyers and sellers. It should be noted that a payment system will not be implemented. The buyers will get the communication information from the sellers' profile. Once an item is sold, the seller should manually mark that item as sold out. A user support system through FAQs and email will address questions and issues to facilitate user engagement. The technology stack will contain Django for backend development, integrating with AWS for cloud services and elastic search for efficient search functionality. HTML, CSS will ensure an intuitive and visually appealing front end, contributing to a user-friendly and sustainable second-hand marketplace. To sum up, the web application will have a positive impact in facilitating METU community members to buy and sell second-hand items effortlessly. In doing so, it will actively promote an eco-friendly lifestyle and enable responsible consumption among its users.

SaaS (Software as a Service)

The web application itself, which allows users to post and search for second-hand items, can be delivered as a SaaS. METU students access the application through a web browser without needing to install any software locally. The application handles all the functionality, and users interact with it through a user-friendly interface. While IaaS and PaaS models provide more control over infrastructure and platform configurations, they also require more involvement in system management. SaaS is a suitable choice for our web application, primarily focusing on providing a service rather than managing infrastructure.

Data Types

Binary Data:

- Item images uploaded by users.
- Binary data for user profile pictures.

Text Data:

- Item descriptions and details.
- User reviews and comments.
- User profile information.

Numerical Data:

- Price of the items.

Data Flow Diagram:

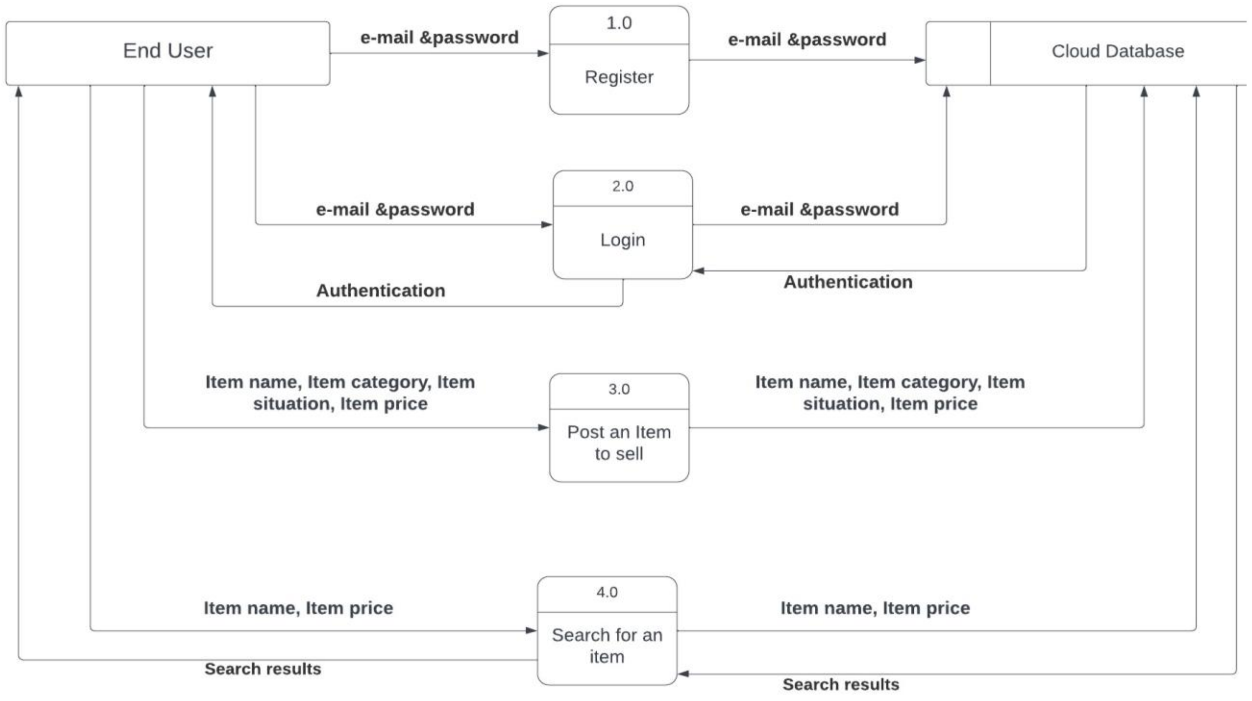


Figure 1:Data Flow Diagram

Computation Diagram:

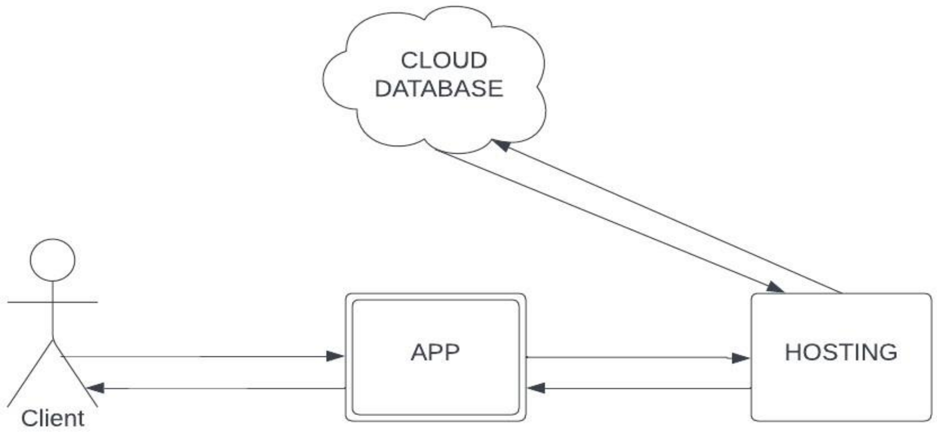


Figure 2:Computation Diagram

Client- Service Interaction Diagram:

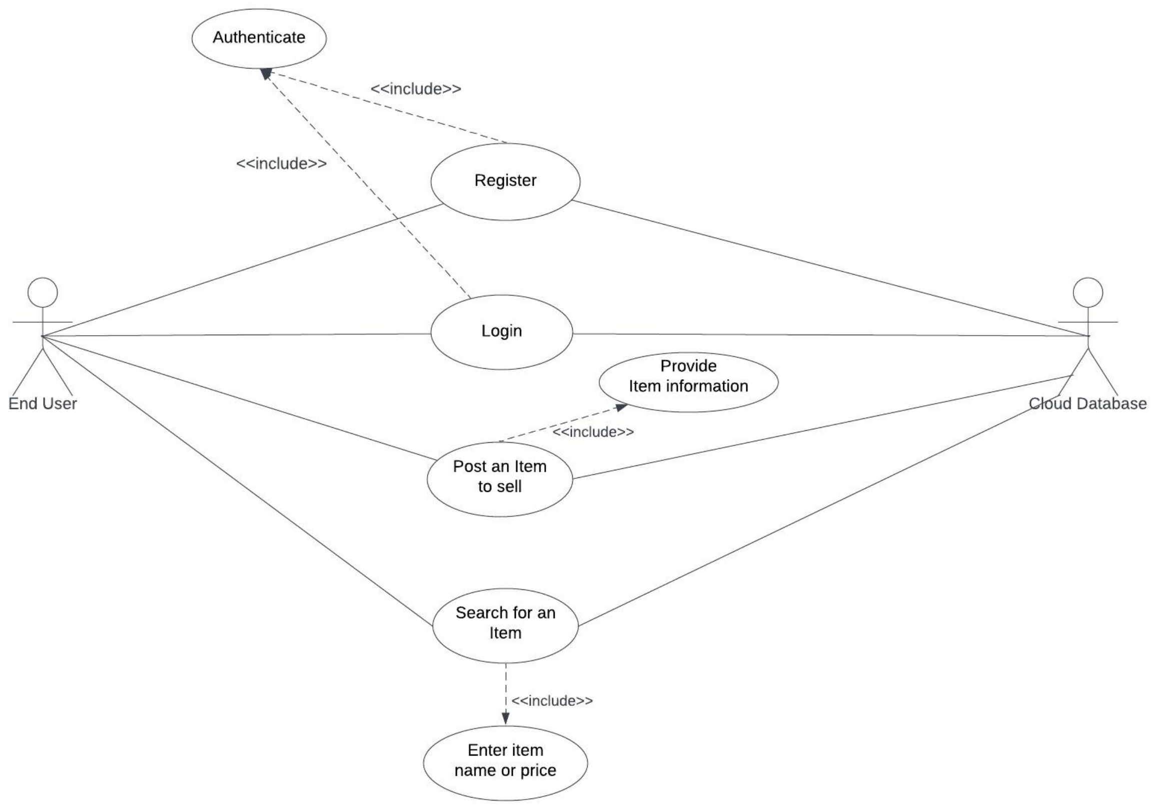


Figure 3:Client-Service Interaction Diagram

Expected contribution for each project group member:

We have decided that the implementation of front-end will be done by Egemen Aksöz and back-end code and MySQL database connection will be implemented by Ece Erseven. We also decided that integration with Cloud (Amazon RDS) and Search functionality will be implement together since we both want to learn about Cloud Technologies.

Milestones Achieved

October 23-27: Brainstorming and Cloud Technology Research

During this phase, our team brainstormed to generate creative ideas for the project. Various cloud technologies were searched to determine the most appropriate ones for our project's needs. It is decided to use Amazon RDS for our cloud database. Our project's objectives and proposal report are completed at the end of this period.

October 30 – November 3: Initial Implementation

We started to work on the initial stages of implementation. For example, necessary environments for our project were downloaded and set, such as PyCharm and MySQL Workbench. We shared the tasks and started to work on front-end and back-end codes separately. It is decided that the front-end code will be implemented by Egemen (including the design of the pages, HTML, and CSS codes) and the back-end code (including models, views, and forms using Python) and MySQL database connection by Ece. Cloud-related parts are decided to implement together as a team (Amazon RDS connection).

November 13 – 19: Back-end and Front-end Development

In this stage, our team developed the back-end and front-end codes separately. This parallel approach allowed us to progress a well-coordinated development process.

The work completed by Ece:

First, two apps, called "userAuthentication" and "marketplace," are created, each serving different purposes within the project. The "userAuthentication" app manages user-related functionalities, focusing on authentication, registration, login, and profile management. Within the "userAuthentication" app, the model, particularly the UserProfile class in "models.py," holds critical user information, such as email and phone number, improving the default Django User model with additional user-specific details. Also, the views defined in "views.py" handle the logic implementation for user registration, login, profile display, and logout.

On the other hand, the "marketplace" app is used to post and display items within the project. Its purpose is to create a marketplace where users can post items for sale and manage their posts. In the "marketplace/models.py" file, the Item class defines the properties such as item name, price, category, condition, description, and image. The views in "marketplace/views.py" are designed to implement functionalities such as posting new items for sale and displaying a user's posts.

userAuthentication/models.py:
UserProfile class is implemented, which holds user-related information such as email, phone number, etc. Some of the fields are implemented for the next stage. For example, the profile_picture area will be used in the project's next step.

userAuthentication/views.py:
UserCreationForm is imported, part of Django's built-in authentication system that handles registration, password validation, and confirmation. Then, the "register" function is implemented that registers and redirects the user to the home page after successful registration. The "user_login" function is created using AuthenticationForm and redirects the user to the dashboard page after successful login. The "dashboard" function is designed to get the user profile associated with the currently logged-in user, and the "user_logout" function is implemented to redirect the user to the home page. Additionally, the user_profile function is implemented for the Profile page.

userAuthentication/forms.py:
When the user visits the profile page, the user_profile function calls the UserProfileForm, which enables the user to add the phone number and email to the system.

userAuthentication/urls.py:
The "urls.py" is created for the URL patterns (routing) for the authentication-related views.

marketplace/models.py:
Item class is implemented, and it holds item_name,price, category,condition,description and image. Items are also stored with the user foreign key that adds that item to the system.

marketplace/views.py:

The sell_item function implemented in this view is called SellItemForm so that users can add the item's properties to the system when they visit the Sell Item page. After saving an item, this function redirects the user to the dashboard. Additionally, the my_posts function is responsible for displaying the posts made by the currently authenticated user.

marketplace/forms.py:

This form facilitates the process of creating Item instances. It includes fields for category, condition, and image, and it is designated to work with the Item model.

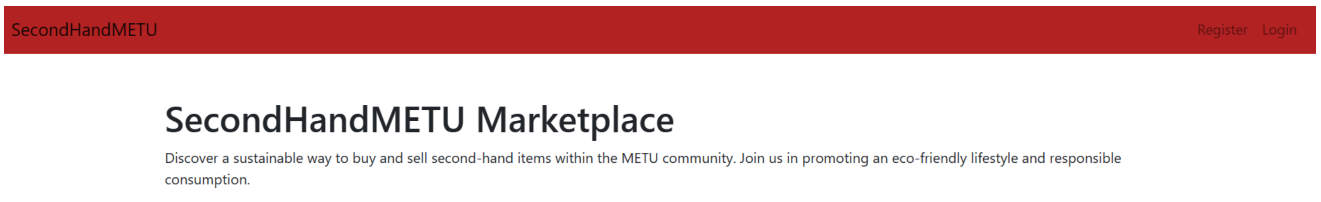
marketplace/urls.py:

The "urls.py" is created for the URL patterns (routing) for marketplace-related views.

The work completed by Egemen:

Home Page:

It is designed the Dashboard simply. Firstly, a red navigation bar is created; it serves as a user interface element, allowing users to navigate our platform. It contains two buttons, "Register" and "Login," allowing users to create accounts or log in to access the platform's features. In the middle of the page, a section providing brief information about our platform is included.



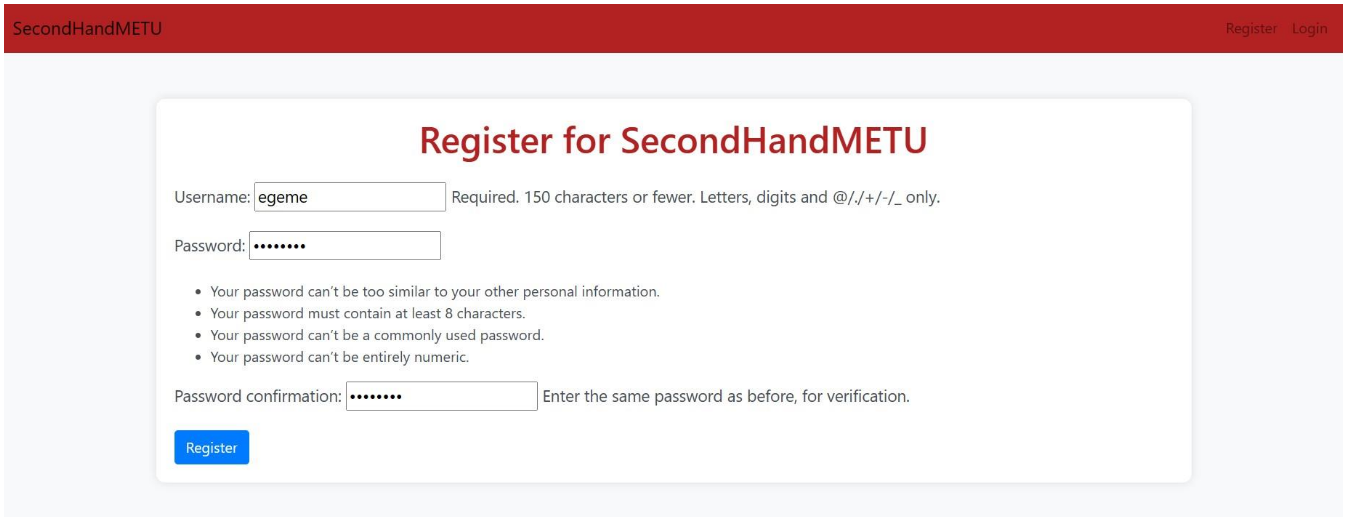
SecondHandMETU Marketplace

Discover a sustainable way to buy and sell second-hand items within the METU community. Join us in promoting an eco-friendly lifestyle and responsible consumption.

Figure 4:Home Page

Registration Page:

The register page enables users to register with specific input elements. This page includes text boxes for users to input their wanted username and password and verify the password during registration. The register button enables users to submit their registration details. Also, Django set some user username and password restrictions; for example, the password must contain at least eight characters or can't be entirely numeric.



Register for SecondHandMETU

Username: egeme Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Password: ••••••••

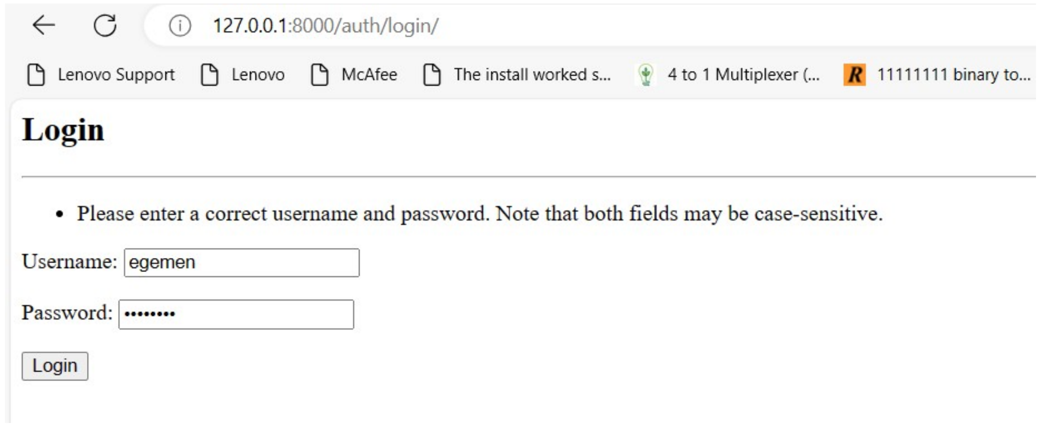
- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation: •••••••• Enter the same password as before, for verification.

Figure 5:Registration Page

Login Page:

The login page includes username and password textboxes and a submit button displaying their login credentials. Users can enter their login information and access the dashboard.



Login

- Please enter a correct username and password. Note that both fields may be case-sensitive.

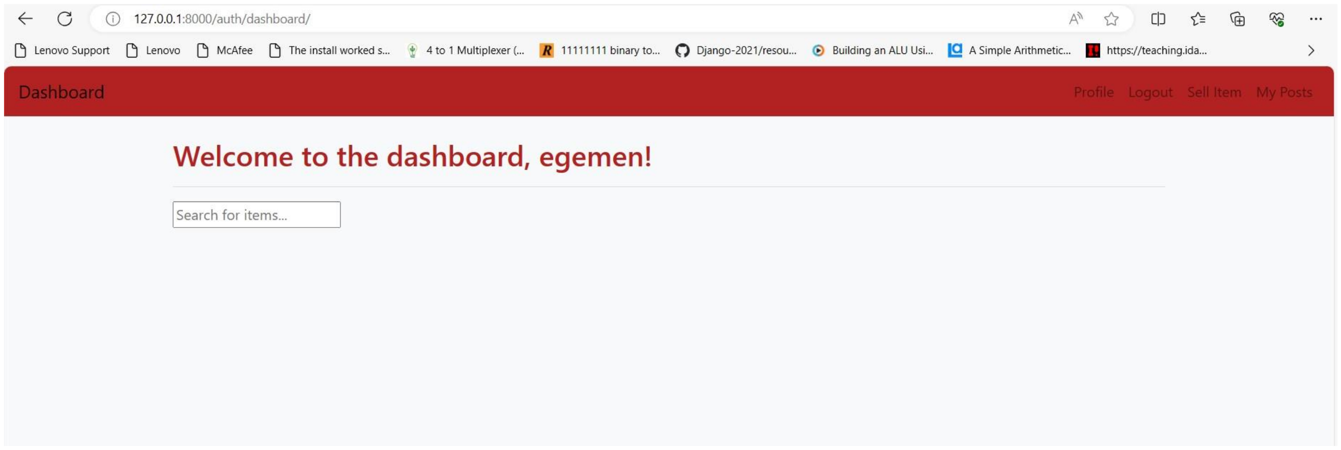
Username: egemen

Password: ••••••••

Figure 6:Login Page

Dashboard Page:

The navigation bar includes buttons such as Profile, My Posts, Sell Item, and Logout. A search bar is displayed on this page but will be implemented in the next step. Clicking the "Profile" button takes users to their profile page, where they can edit their personal information. 'Logout' button: Allows users to log out of their accounts. An "Item for Sale" button takes users to a form or page where they can list and sell items. The "My Posts" button is implemented to view what users have listed for sale.

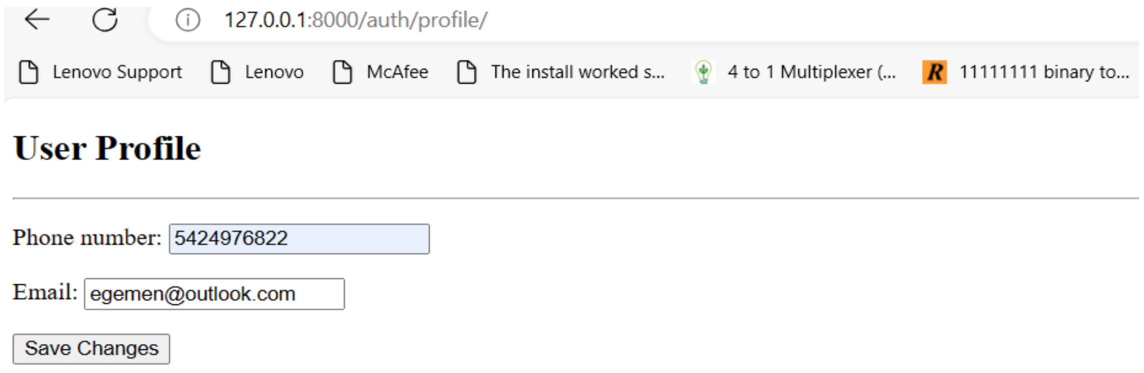


Welcome to the dashboard, egemen!

Figure 7:Dashboard Page

Profile Page:

The user's profile page includes input fields for their email address and phone number. Also, the "Save Changes" button allows users to submit changes. Users can easily manage and update their personal information by editable fields and a save button on the profile page.



User Profile

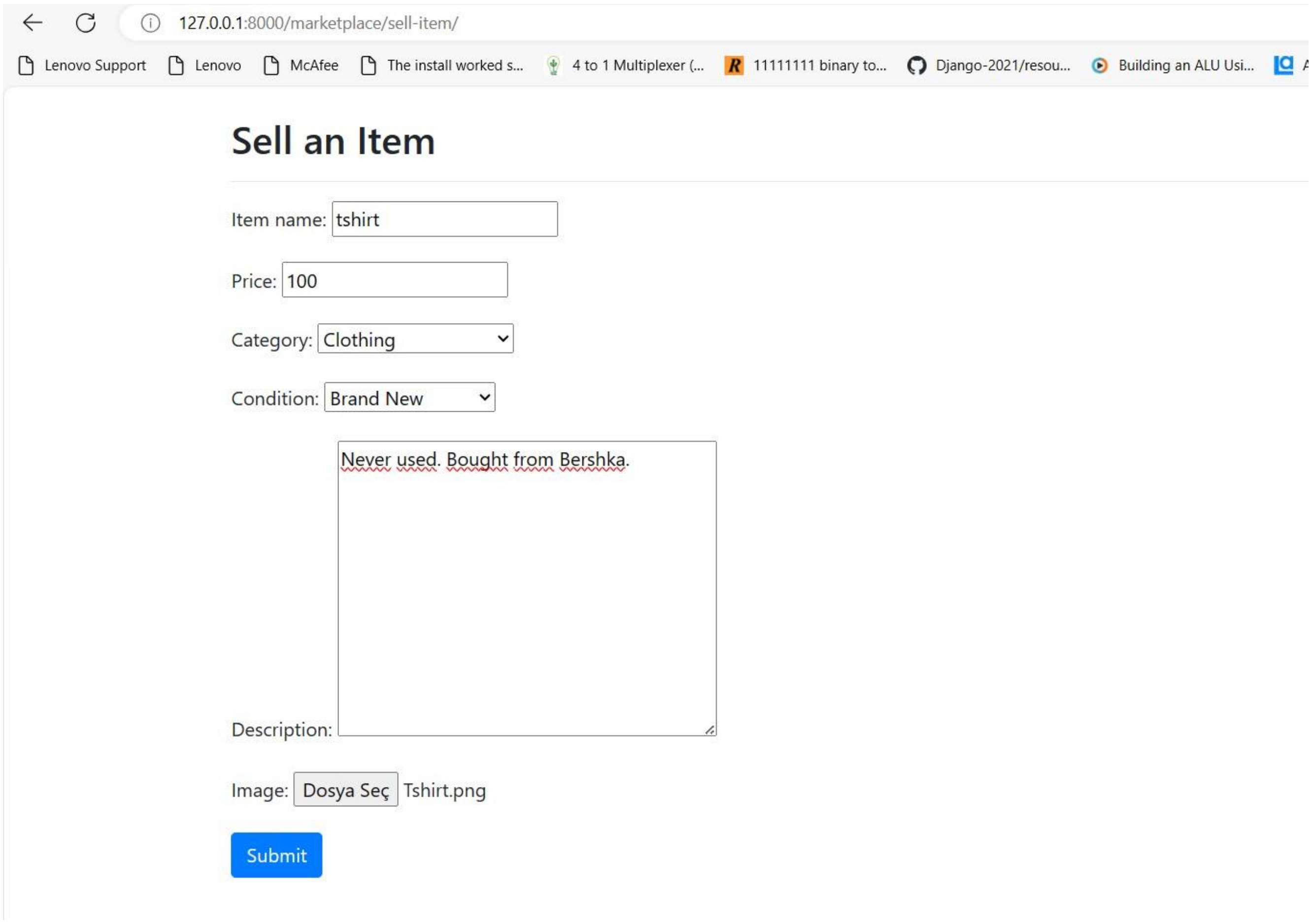
Phone number: 5424976822

Email: egemen@outlook.com

Figure 8:Profile Page

Sell Item Page:

The sell item page contains input fields for the item's name, price, category, and condition. The category and condition are dropdown menus that offer users predetermined alternatives. Also, users can add some information about the item with related photos. By establishing a separate page for users to sell items with specific features such as name, price, category, and condition, we simplify and make listing them user-friendly. This improves the user experience and encourages users to sell their items on our platform.



Sell an Item

Item name: tshirt

Price: 100

Category: Clothing

Condition: Brand New

Description: Never used. Bought from Bershka.

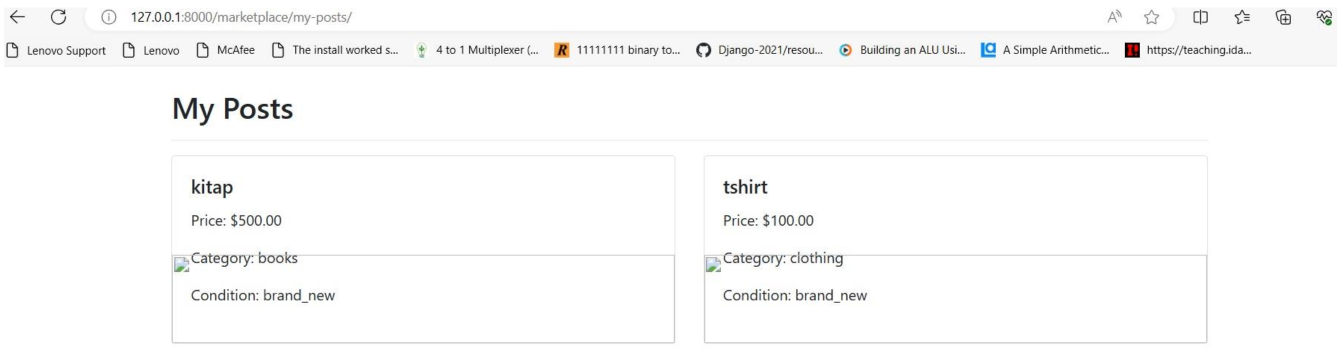
Image: Dosya Seç Tshirt.png

Submit

Figure 9:Sell Item Page

My Posts Page:

The My Posts page includes the item's information, including picture, name, price, category, and condition. Users can quickly review the details of what they have posted on the web app by accessing a summary of them on this page. The challenge here was displaying the image to the users; it is confirmed that the image was added to the database correctly, but for some reason, it cannot be displayed to the user in this step and hopefully will be fixed in the next stage.


127.0.0.1:8000/marketplace/my-posts/

My Posts

kitap
Price: $500.00
Category: books
Condition: brand_new

tshirt
Price: $100.00
Category: clothing
Condition: brand_new

Figure 10:My Posts Page

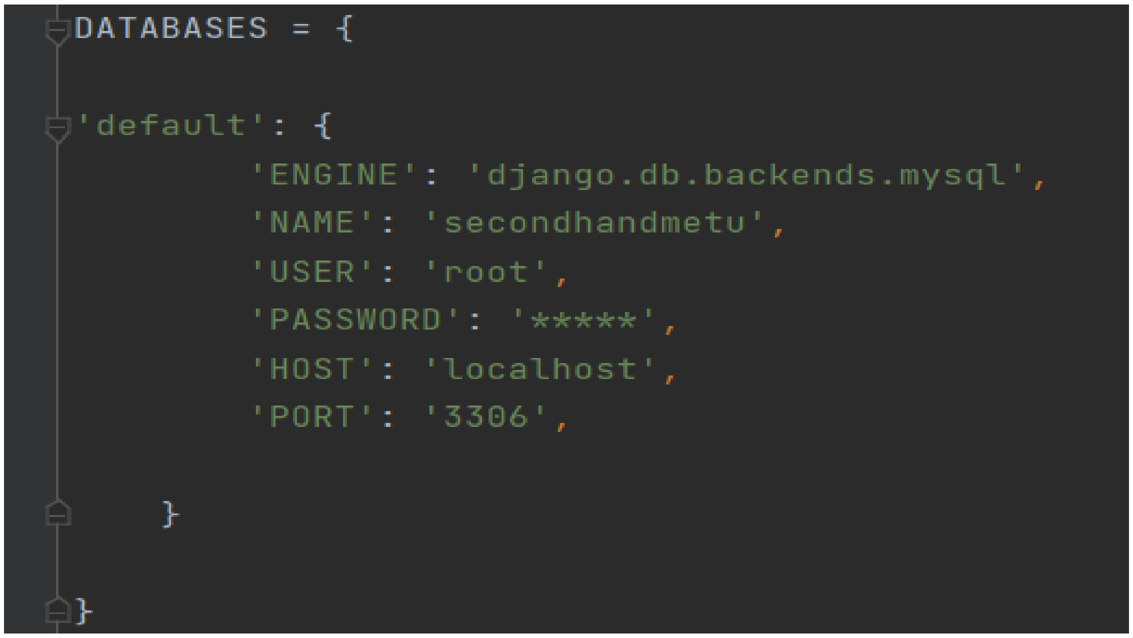
November 20 – 26: Integration and Bug Fixing

We focused on integrating back-end and front-end components. During this phase, we tried to identify and address any bugs or issues that occurred. Even though we had some difficulties integrating front and back-end codes, we successfully joined them.

The work completed by Ece:

Database tables are connected with MySQL. The connection is established by setting the engine as 'Django.db. backends.mysql', database name, user name, password, host, and port; then setting up a new connection on the MySQL side and matching the host, port, and password information. To ensure the integration with the database, it is tested by registering as a user and adding an item to sell. The database tables are controlled using MySQL workbench and checked, for example, if the item information is inserted into the table correctly or usernames are visible in table. Django Documentations (2023) are used to understand the connection steps and the related link is provided in references.

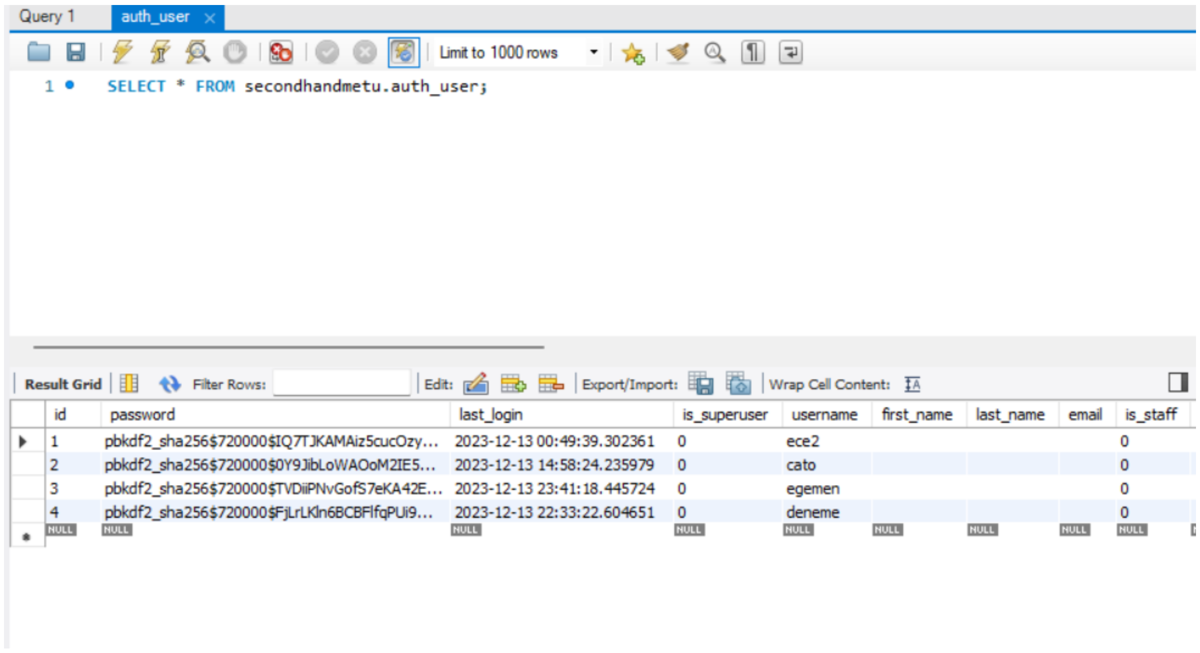
Establishing the connection between Django application and MySQL database:



```
DATABASES = {

'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'secondhandmetu',
        'USER': 'root',
        'PASSWORD': '*****',
        'HOST': 'localhost',
        'PORT': '3306',

    }

}
```

Figure 11:Database Configurations 1

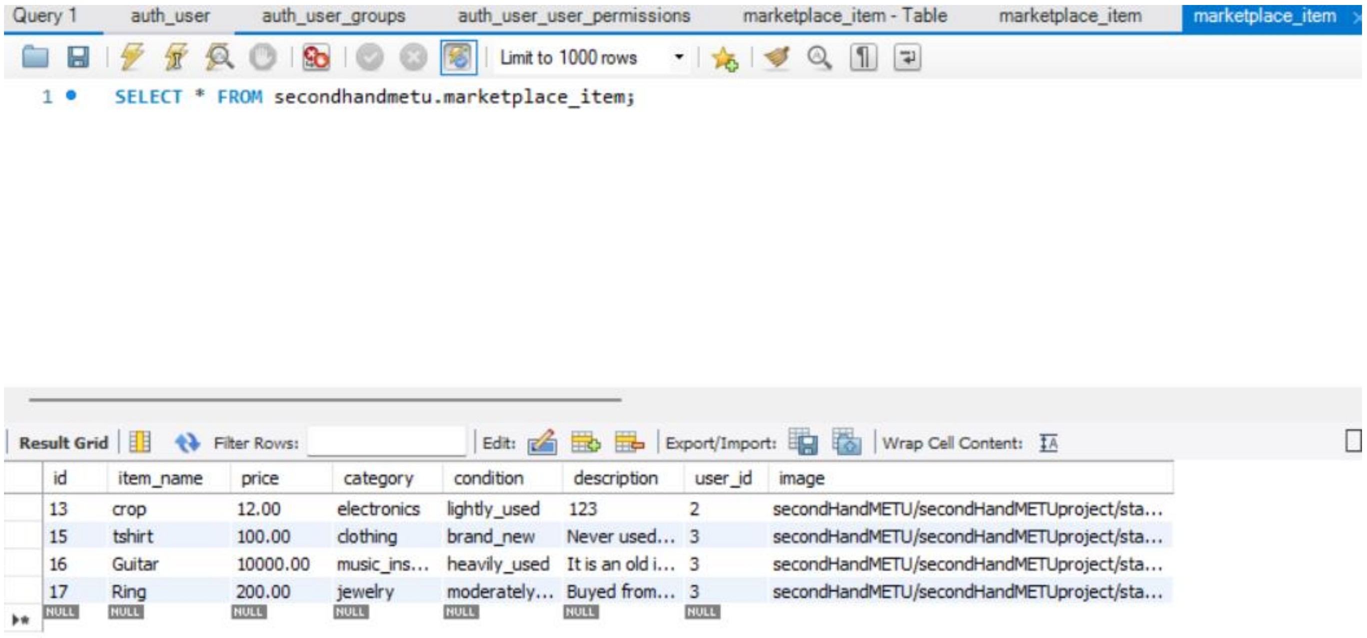
User Database Table: It can be seen that usernames and passwords are correctly inserted into table.



```
SELECT * FROM secondhandmetu.auth_user;
```

id	password	last_login	is_superuser	username	first_name	last_name	email	is_staff
1	pbkdf2_sha256$720000$IQ7TJKAMAiz5cucOzy...	2023-12-13 00:49:39.302361	0	ece2				0
2	pbkdf2_sha256$720000$0Y9JibLoWAOoM2IE5...	2023-12-13 14:58:24.235979	0	cato				0
3	pbkdf2_sha256$720000$TVDiiPNvGofS7eKA42E...	2023-12-13 23:41:18.445724	0	egemen				0
4	pbkdf2_sha256$720000$FjLrLKln6BCBFlfqPUi9...	2023-12-13 22:33:22.604651	0	deneme				0
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 12:User Table in MySQL



```
SELECT * FROM secondhandmetu.marketplace_item;
```

id	item_name	price	category	condition	description	user_id	image
13	crop	12.00	electronics	lightly_used	123	2	secondHandMETU/secondHandMETUproject/sta...
15	tshirt	100.00	clothing	brand_new	Never used...	3	secondHandMETU/secondHandMETUproject/sta...
16	Guitar	10000.00	music_ins...	heavily_used	It is an old i...	3	secondHandMETU/secondHandMETUproject/sta...
17	Ring	200.00	jewelry	moderately...	Buyed from...	3	secondHandMETU/secondHandMETUproject/sta...
NULL	NULL	NULL	NULL	NULL	NULL	NULL	

Figure 13:Item Table in MySQL

November 27 – December 3: Midterm Evaluation

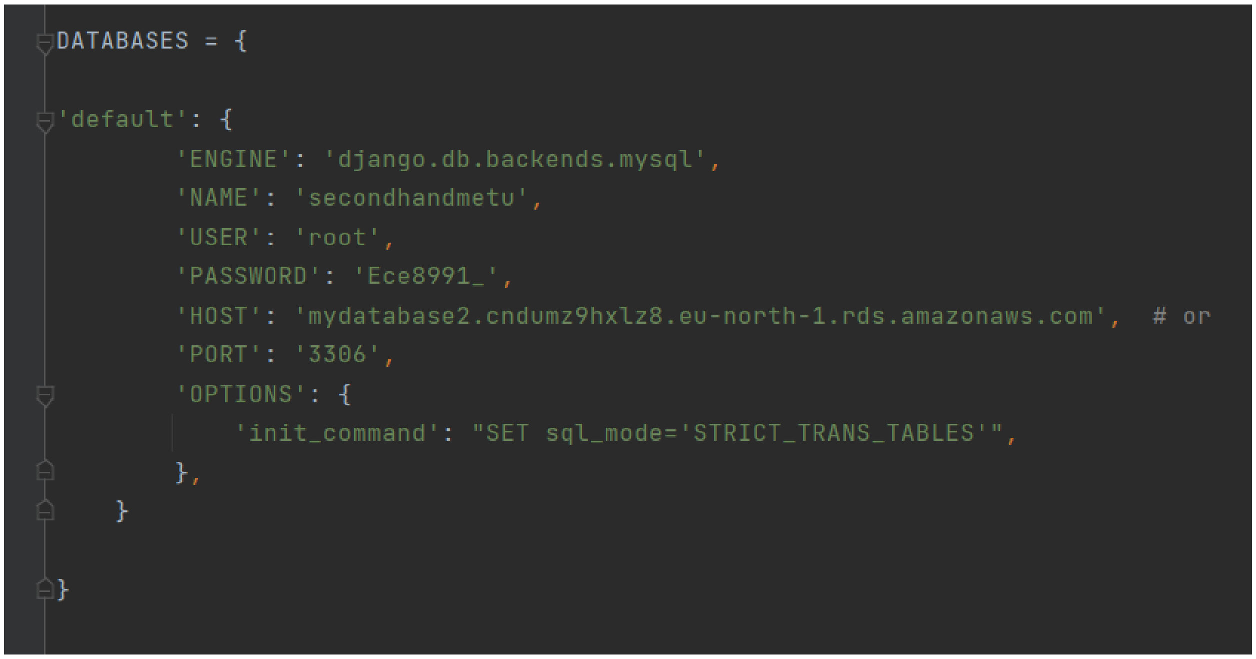
The team was on a break during the midterm week.

December 4 – 10: Database Connection with Amazon RDS

The work completed by Ece & Egemen together:

Our primary focus was establishing a connection with the cloud at this phase. We have created an Amazon RDS instance using the free tier, which provides up to 20 GBs. Then, we connected the MySQL workbench and Django project with the created instance. It is updated the 'HOST' as 'mydatabase2.cndumz9hxlz8.eu-north-1.rds.amazonaws.com'. The challenge of this part was the configuration of VPC security groups. After editing the inbound and outbound rules and allowing "All traffic," we were able to connect our project database with Amazon RDS. In addition to that we tried to finalize code parts. While doing this part we have followed Amazon RDS Documentations(n.d) and a tutorial from Medium.com(2021); and the related links are provided in references.

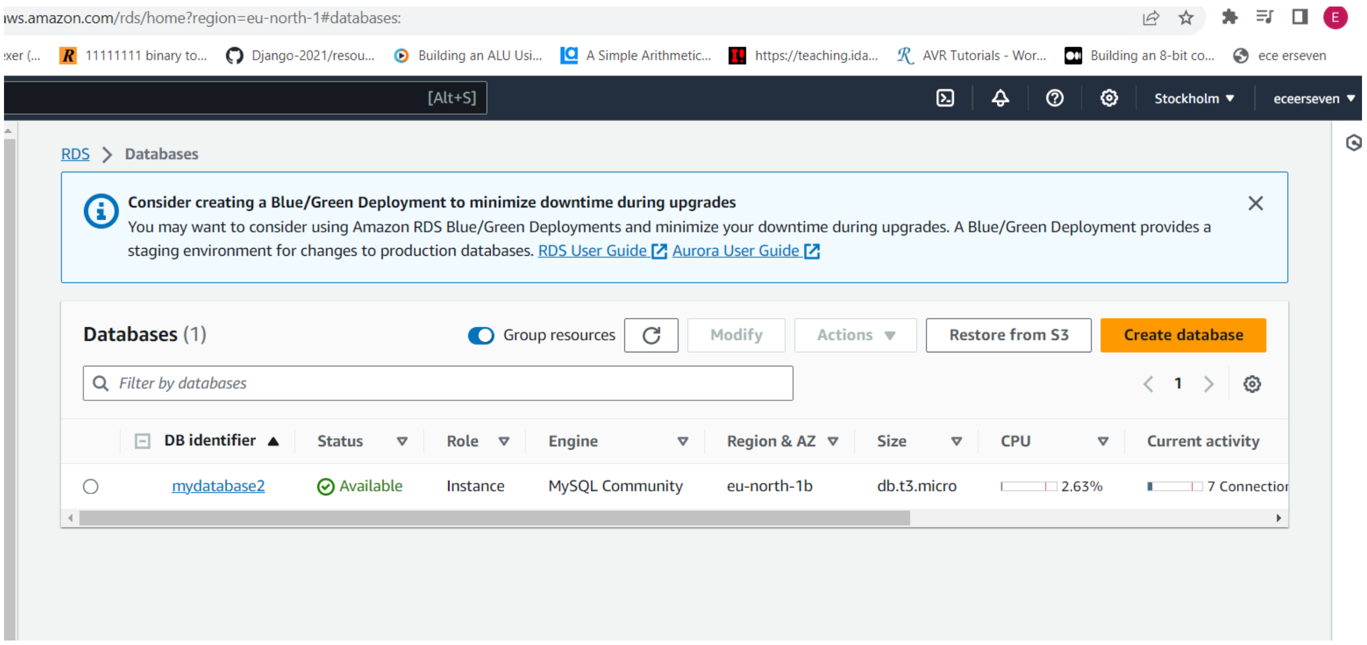
The configurations done for connecting the database in setting.py, the end point of the Amazon RDS is added instead of the local host.



```
DATABASES = {

'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'secondhandmetu',
        'USER': 'root',
        'PASSWORD': 'Ece8991_',
        'HOST': 'mydatabase2.cndumz9hxlz8.eu-north-1.rds.amazonaws.com',  # or
        'PORT': '3306',
        'OPTIONS': {
            'init_command': "SET sql_mode='STRICT_TRANS_TABLES'",
        },
    }

}
```

Figure 14:Database Configurations 2

It can be seen that the database is available and connected:



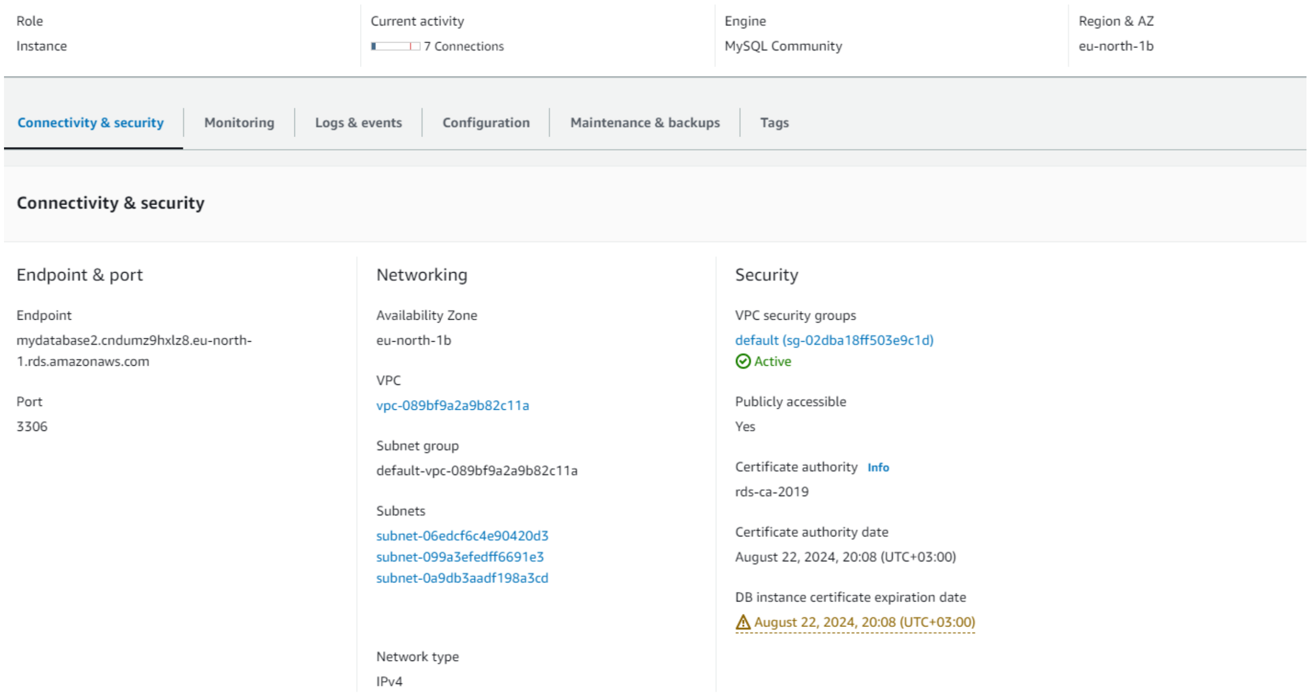
Consider creating a Blue/Green Deployment to minimize downtime during upgrades
You may want to consider using Amazon RDS Blue/Green Deployments and minimize your downtime during upgrades. A Blue/Green Deployment provides a staging environment for changes to production databases. RDS User Guide Aurora User Guide

Databases (1)

DB identifier	Status	Role	Engine	Region & AZ	Size	CPU	Current activity
mydatabase2	Available	Instance	MySQL Community	eu-north-1b	db.t3.micro	2.63%	7 Connectio

Figure 15:MySQL database is connected to Amazon RDS

The details including the endpoint and Security configurations is as follows :



Role	Current activity	Engine	Region & AZ
Instance	7 Connections	MySQL Community	eu-north-1b

Connectivity & security | Monitoring | Logs & events | Configuration | Maintenance & backups | Tags

Connectivity & security

Endpoint & port

Endpoint
mydatabase2.cndumz9hxlz8.eu-north-1.rds.amazonaws.com

Port
3306

Networking

Availability Zone
eu-north-1b

VPC
vpc-089bf9a2a9b82c11a

Subnet group
default-vpc-089bf9a2a9b82c11a

Subnets
subnet-06edcf6c4e90420d3
subnet-099a3efedff6691e3
subnet-0a9db3aadf198a3cd

Network type
IPv4

Security

VPC security groups
default (sg-02dba18ff503e9c1d)
Active

Publicly accessible
Yes

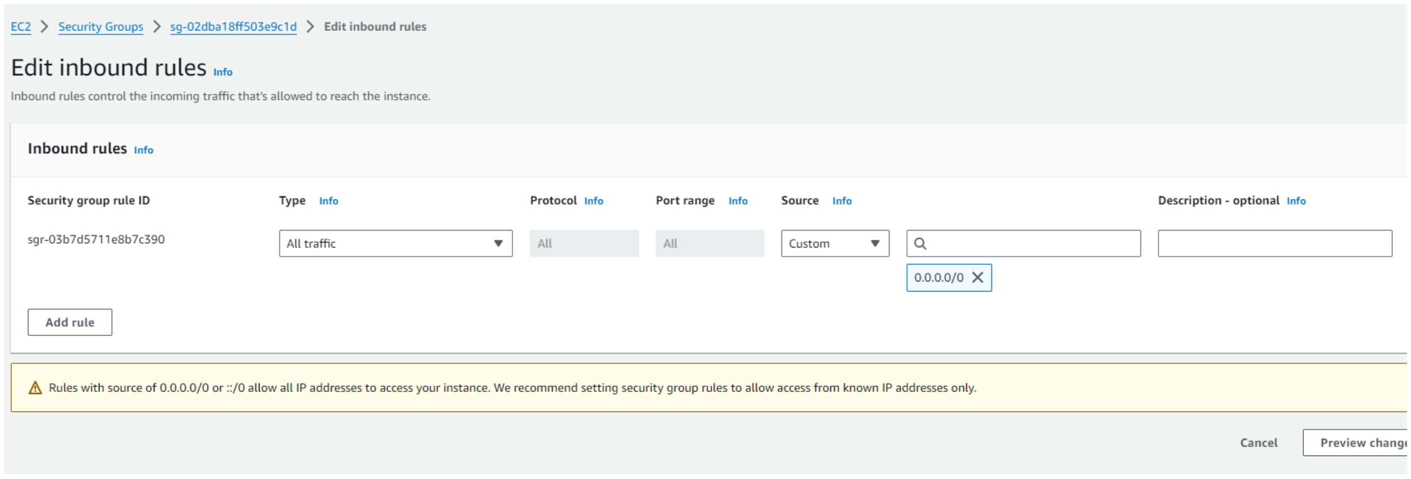
Certificate authority Info
rds-ca-2019

Certificate authority date
August 22, 2024, 20:08 (UTC+03:00)

DB instance certificate expiration date
August 22, 2024, 20:08 (UTC+03:00)

Figure 16:Connection details of RDS -end point , port, security details

Editing inbound & outbound rules and allowing the all traffic enable us to establish the connection successfully:



EC2 > Security Groups > sg-02dba18ff503e9c1d > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info
sgr-03b7d5711e8b7c390	All traffic	All	All	Custom	0.0.0.0/0	

Add rule

Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Cancel | Preview chang

Figure 17:Updating the Inbound rules & allowing all traffic

December 11 – 17: Report Preparation

In the final period of the project, we started to prepare a complete report that covers the project's development process, challenges faced, solutions implemented, and overall outcomes.

Report contribution of Ece:
The milestones achieved, future milestones, read-me file, references and preparing final version of the report is done by Ece.

Report contribution of Egemen:
The explanation of the user interface and HTML files prepared by Egemen.

Milestones remained

December 18 – 24:

User Profile page will be enhanced; for example, adding a navbar and enabling the user to add a profile picture will be done by Egemen. Ece will work on allowing the users to add multiple item photos and display them to the user on the MyPosts page. Ece and Egemen will implement the search functionality together using cloud technologies.

December 25 – 31:

Egemen will keep working on enhancing the design of the user interface. Ece will update the posts to show sellers' phone numbers and e-mails and try to fix back-end-related bugs (if they occur). Both team members will keep working on search functionality in this step. A feedback meeting may be conducted with Zekican at the end of this period.

January 1 – January 7: According to the feedback, we will fix the parts that need to be changed and work on hosting our app together as a team; since both team members are not experienced in this task, the contribution of both team members is needed.

January 8 – January 14: The team will prepare the final project report by sharing the parts.

The Github Link for the project: https://github.com/eceerseven/secondHandMETU

References

Amazon Web Services. (n.d.). Amazon RDS Documentation: Overview of DB Instances. Retrieved from https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Overview.DBInstance.html

Django. (2023). MySQL Notes. Django documentation. Retrieved from https://docs.djangoproject.com/en/5.0/ref/databases/#mysql-notes

Lukis, S. (2021). Connecting Django to Amazon RDS. *Medium*. Retrieved from https://medium.com/@stevelukis/connecting-django-to-amazon-rds-c563bad0483e