Swe573 Software Development Practice
Fall 2024

Final Project Deliverables

Course: Swe573 Software Development Practice
Project Name: Mystique Matters
Date: 21.12.2024
Student Name: Egemen Öztürk

Submission Components

- **Software Git Repository**: Git Repo
- **Release Tag URL**: Release URL
- **Deployment URL**: Deployment URL
- **Demonstration Video**: Video Link

Related to the submission of all the project deliverables for the Swe573 Fall 2024 semester project reported in this report, I Egemen Öztürk declare that:

- I am a student in the Software Engineering MS program at Bogazici University and am registered for Swe573 course during the Fall 2024 semester.

- All the material that I am submitting related to my project (including but not limited to the project repository, the final project report, and supplementary documents) have been exclusively prepared by myself.

- I have prepared this material individually without the assistance of anyone else with the exception of permitted peer assistance which I have explicitly disclosed in this report.

Egemen Öztürk

Project Details

Overview

Mystique Matters is a dynamic web application designed for users to create, share, and engage with object-related posts. The platform focuses on enabling collaboration and knowledge sharing about unknown or mysterious objects. Users can register, log in, and manage their profiles while utilizing features such as object post creation, search with tags, attributes etc., commenting, and geolocation. A key highlight of the application is its integration with WikiData, which provides related tag suggestions during object creation. Users can also resolve objects through a dedicated workflow.

Software Requirements Specification (SRS)

Functional Requirements:

1. **Authentication System:** Secure user registration, login, and session management.
2. **Object Post Management:** Create, edit, view, and delete object-related posts.
3. **Profile Management:** Users can manage personal details and retrieve user-specific posts.
4. **Search:** Search posts by tags, attributes, and more.
5. **Object Resolve Workflow:** Enable creators to resolve unidentified objects with collaborative input.
6. **WikiData Integration:** Fetch and suggest related tags dynamically while creating posts.
7. **Detailed Object Attributes:** Support comprehensive object descriptions, including tags, geolocation, material, dimensions, and more.

Non-Functional Requirements:

1. **Responsiveness:** A seamless user experience across desktop, tablet, and mobile devices.
2. **Security:** Robust handling of user data with encryption and secure authentication mechanisms.
3. **Scalability:** Dockerized deployment ensures easy scalability and high availability.

Design Documents

Software Architecture:

1. **Frontend:** Developed using HTML, CSS, and JavaScript.
2. **Backend:** Built with Flask, a lightweight and flexible Python web framework.
3. **Database:** MySQL for structured data management and efficient query handling.
4. **Deployment:** Dockerized for portability and hosted on Heroku for reliable cloud-based deployment.
5. **Media Storage**: AWS S3 bucket is used to store uploaded photos, ensuring scalable and secure media handling.

Diagrams:

User Interaction Flow:

Step 1: The User interacts with the Frontend to initiate an action, such as creating a post.

Step 2: The Frontend sends the request to the API (backend server).

Step 3: The API processes the request and communicates with the MySQL Database to store or retrieve data.

Step 4: The API responds with the result, and the Frontend updates the interface for the User.

Example Scenario - Creating an Object:

Step 1: The User accesses the "Create Object" form on the Frontend.

Step 2: The User fills in the required fields:

1. Title: A brief title for the post (mandatory).
2. Description: Details about the object or topic (mandatory).
3. Tags: Keywords or topics related to the post (mandatory).
4. Photo: Photo of an object (mandatory).

5. Optional Attributes: Other fields like geolocation, material, and size are available but not required.

Step 3: As the User types in the "Tags" field, the Frontend dynamically fetches related tags from WikiData through API calls and displays suggestions. If a tag is selected, it is added to the list of tags for the post.

Step 4: The Frontend submits the data to the API as a POST request.

Step 5: The API validates the data, ensuring mandatory fields are complete, and uploads the photo to the AWS S3 bucket. The photo URL is saved in the MySQL Database alongside the post details.

Step 6: The API responds with a success message and details of the newly created post.

Step 7: The Frontend updates the UI to display the new post in the user's feed.

Project Status

Requirements Status: Partially completed. While core features are implemented, advanced search and object resolution workflows require further development.

Deployment Status:

- **URL:** Deployment URL
- **Dockerized:** Fully Dockerized with a provided Dockerfile.

Local Installation Instructions

Clone the Repository

```
git clone https://github.com/egemnoztrk/swe573-egemen

cd swe573-egemen
```

Environment Setup

Create an .env file in the root directory of the project and populate it with the required environment variables:

```
MYSQL_URI="mysql+pymysql://<username>:<password>@<host>:<port>/<database>"

FLASK_SECRET_KEY="your_secret_key"

AWS_ACCESS_KEY_ID="your_aws_access_key"

AWS_SECRET_ACCESS_KEY="your_aws_secret_access_key"

AWS_DEFAULT_REGION="your_aws_region"
```

Build and Run Docker Container

```
docker build -t mystique-matters .

docker run -p 5500:5500 mystique-matters
```

Access the Application

Open a browser and navigate to http://localhost:5500

User Manual

1. **Register:** Navigate to `/register`. Fill in the required fields and submit the form.
2. **Login:** Access `/login`, enter your email and password, and log in securely.
3. **View Posts:** The homepage displays a curated feed of the latest object posts.
4. **Create a Post:** Authenticated users can create object posts through the homepage form, with detailed attributes such as material, dimensions, and geolocation.
5. **Search:** Use the search bar to find posts by tags, attributes, or keywords. Related tags are dynamically suggested via WikiData integration.

6. **Profile Management:** Update and view personal details on the `/profile` page.
7. **Comment:** Add comments on object posts to contribute to discussions and resolutions.
8. **Object Resolution:** Resolve an object directly from its post page by completing the resolution workflow.

Users

1. **Username: egemnoztrk@gmail.com   password: 123**
2. **Username: ali@gmail.com   password: 123**
3. **Username: merve@gmail.com   password: 123**
4. **Username: egemen@gmail.com   password: 123**