Project Report / Synopsis: Image Generation App

1. Introduction

This project is a Flask-based web application that allows users to generate images from text prompts using the Hugging Face API. It features user authentication, stores generated prompts and images in a MySQL database, and includes API integration for generating images. The core functionalities include user registration/login, image generation based on text prompts, and storing/logging prompt data.

2. Technology Stack

- Backend Framework: Flask (Python)
- Database: MySQL
- API Integration: Hugging Face API for text-to-image generation
- Authentication: Password encryption with Passlib
- Testing: MySQL database connection test via Python
- Dependencies: Listed in requirements.txt

3. Application Workflow

3.1. User Authentication

- Login (`/`): Users provide their username and password. On successful login, the user is redirected to the prompt generation page. If login fails, a flash message is displayed.
- Registration (`/register`): New users can create accounts. Passwords are hashed using SHA-256 encryption via passlib.

3.2. Image Generation

- Prompt Page (`/prompt`): Logged-in users can submit a text prompt to generate an image.
- Generate Image (`/generate_image`): The prompt is sanitized, sent to the Hugging Face API, and

if successful, the generated image is stored in the database and shown to the user.

3.3. Data Handling and Storage

- Prompts and image paths are stored in MySQL.

- Session management ensures only authenticated users access key pages.

4. Code Overview

- app.py defines routes for login, registration, prompt submission, and image generation.

- test.py tests MySQL connectivity.

5. Installation and Setup Instructions

1. Install Dependencies: `pip install -r requirements.txt`

2. Create a MySQL database named `image_gen_app`.

3. Configure environment variables (e.g., Hugging Face API key).

4. Run the app: `python app.py`

6. Security Considerations

- Use environment variables to store sensitive keys.

- Passwords are hashed using sha256_crypt from passlib.

- User inputs are sanitized to prevent injection attacks.

7. Dependencies (requirements.txt)

```
Flask==2.3.2

mysql-connector-python==8.0.33

passlib==1.7.4

Pillow==9.5.0

requests==2.31.0
```

8. Conclusion

This application integrates Flask, MySQL, and Hugging Face API to provide a functional image generation system. With user authentication and session management, it ensures secure access. It showcases the use of third-party APIs and database management, making it suitable for enhancements like multi-user support or cloud deployment.