Project Report: Text-to-Image Generation Web Application

1. Introduction

This project focuses on building a web-based application that allows users to generate images from text prompts using deep learning models. It leverages Flask as the web framework and integrates MySQL for handling user credentials and prompts. The application supports user registration, login, and image generation functionalities. The backend uses models from the `diffusers` library to generate images based on text inputs.

---

2. Technology Stack

- **Backend**: Flask (Python web framework)
- **Database**: MySQL (for user authentication and storing prompts)
- **Image Generation**: `diffusers` (deep learning library for generative AI)
- **Frontend**: HTML, CSS (for user interfaces)
- **Other Libraries**:
  - `mysql-connector-python`: For MySQL database connectivity
  - `passlib`: For password hashing and verification
  - `Pillow`: For handling image file processing
  - `transformers`, `torch`, `safetensors`: For managing models
  - `requests`: For handling HTTP requests

---

3. System Architecture

1. **User Authentication**:
   - The application uses secure login and registration systems. Passwords are hashed using the `sha256_crypt` hashing algorithm provided by `passlib` before being stored in the MySQL database.
   - Session management is used to track user logins.
2. **Prompt Submission & Image Generation**:
   - After logging in, users can submit prompts for image generation.
   - The application uses pre-trained deep learning models from the `diffusers` library to generate the images based on user input.
   - Generated images are stored locally, and their paths are saved in the database.
3. **Database Design**:
   - Two primary tables: `users` and `prompts`.
     - `users` table stores user credentials.

        - `prompts` table stores user-submitted text prompts and the generated image paths.

---

4. File Overview

1. **`app.py` (Main Application File)**
    - **Flask Routes**:
        - `/`: Displays the login page.
        - `/login`: Authenticates users using the credentials stored in the MySQL database.
        - `/register`: Allows new users to register by storing their username and hashed password in the database.
        - `/prompt`: Placeholder route to submit a prompt for generating images.
    - **Database Connection**:
        - Connects to a MySQL database using credentials from environment variables.
        - Handles database queries for user authentication and registration.
    - **Session Management**:
        - Manages user sessions using `session` in Flask, enabling users to remain logged in across multiple requests.
2. **HTML Templates**:
    - **`templates/login.html`**: Renders the login page with username and password fields, including form validation messages.
    - **`templates/register.html`**: Renders the registration page for new users to sign up.
    - Both templates use Flask's `url_for` function to link the static assets (CSS, JavaScript).
3. **CSS (`static/css/styles.css`)**:
    - Styles the user interface, including forms for login and registration.
    - Provides a responsive design for smaller screen sizes using media queries.
    - Background is set using an image, and containers are styled to offer a clean, user-friendly experience.
4. **JavaScript (`static/script/script.js`)**:
    - Handles client-side validation for login and registration forms.
    - Displays an error message if the username or password fields are left empty.
5. **Database Schema (`image_gen_app.sql`)**:
    - Contains SQL commands for creating the `users` and `prompts` tables.
    - The `users` table includes fields for `id`, `username`, `password`, and timestamps.
    - The `prompts` table stores user-generated prompts and the associated image file paths.

6. **`requirements.txt`**:
    - Lists all dependencies required for the project, such as Flask, MySQL connector, and libraries for deep learning models.

---

5. Key Functionalities

1. **User Registration & Login**:
    - Registration form collects a username and password, which are stored in the database after hashing.
    - The login form checks the username and password combination by verifying the password hash and starts a session upon successful login.
2. **Prompt Submission**:
    - Once logged in, users can submit text prompts for image generation.
    - (Placeholder for the prompt generation feature to be integrated in the future).
3. **Password Security**:
    - User passwords are stored securely using `sha256_crypt` hashing.
    - During login, passwords are verified by comparing the hashed values rather than plain text.
4. **Database Operations**:
    - `INSERT`, `SELECT`, and other MySQL queries are used for handling user data and prompts.
    - Error handling for database operations ensures that exceptions are caught and appropriate feedback is provided to the user.

---

6. Installation and Setup

Pre-requisites:

- Python 3.7 or higher
- MySQL database

Steps:

Clone the repository:
bash
Copy code

```
git clone <repo-link>
cd project-directory
```

1.

Install required Python packages:
bash
Copy code
```
pip install -r requirements.txt
```

2.
3. Set up the MySQL database:

Import the provided `image_gen_app.sql` file into your MySQL instance:
bash
Copy code
```
mysql -u root -p < image_gen_app.sql
```

   ○

Run the Flask application:
bash
Copy code
```
python app.py
```

4.
5. Access the application at `http://localhost:5000`.

---

7. Future Enhancements

- **Image Generation Integration**: Add the functionality to generate images from prompts using the models provided by the `diffusers` library.
- **User Dashboard**: Allow users to view and manage their submitted prompts and generated images.
- **Improved Error Handling**: Implement better error handling for various operations (e.g., login failures, database errors).

---

8. Conclusion

This project demonstrates a foundational web application integrating Flask with MySQL for user management and prompt submission. With additional features like image generation, this application could serve as a robust platform for generating AI-based images from text inputs.