Photography Agent

- Sai Venkat Reddy Kopparthi

https://photographyproject-2715650954.us-west1.run.app/login/

username: kopparthi.saivenkat@gmail.com pwd: kopparthi.saivenkat

usernameL ksvr444job@gmail.com pwd: ksvr444job

1. Introduction

In today's digital era, the demand for personalized and AI-driven solutions is growing rapidly. This project aims to create a **web-based platform** that allows users to train **Flux-lora-dev** model for image generation and generate images based on user-defined prompts. The system integrates **Firebase** for user authentication and data storage, **Replicate** for model training and image generation, and **OpenAI** for generating image descriptions. Once training is complete, the user is notified via email. The system also provides a dropdown to select a trained LoRA and a text box to generate images based on user-defined prompts. The platform is built using **Django**, a robust Python web framework, and is containerized using **Docker** for easy deployment.

The key features of the system include:

- User authentication and session management.
- Users can upload 15 images with text descriptions.
- Training **ostris/flux-dev-lora-trainer** using user-uploaded images.
- Generating images based on user prompts.
- Displaying user-specific models and generated images.
- Sending email notifications upon training completion.

2. Existing Solutions in the Market and Their Limitations

Existing Solutions

1. **Runway ML**:
    - A platform for training and deploying machine learning models.
    - Provides pre-trained models for image generation.
    - **Limitation**: Limited customization options for training models.
2. **DeepArt.io**:
    - Allows users to generate artistic images using AI.
    - **Limitation**: No support for training custom models.
3. **DALL·E by OpenAI**:
    - Generates images from textual prompts.
    - **Limitation**: No user-specific model training or customization.

Limitations of Existing Solutions

- Lack of user-specific model training.
- Limited customization options for training data.
- No integration of user authentication and data storage.
- No email notifications for training completion.

3. Your Approach

Our solution addresses the limitations of existing systems by providing:

- **Custom Model Training**: Users can upload 15 images and train a custom Flux-Dev model using Replicate's API.
- **User Authentication**: Firebase handles user authentication and session management.
- **Image Generation**: Users can generate images using their trained models.
- **Email Notifications**: Users are notified via email when training is complete.
- **OpenAI Integration**: Image descriptions are generated using OpenAI's GPT-4 model.
- **User-Friendly Interface**: A simple and intuitive UI for uploading images, training models, and generating images.

The system is designed to be **user-friendly** and **scalable**, with a focus on providing a seamless experience for both training and generating images.

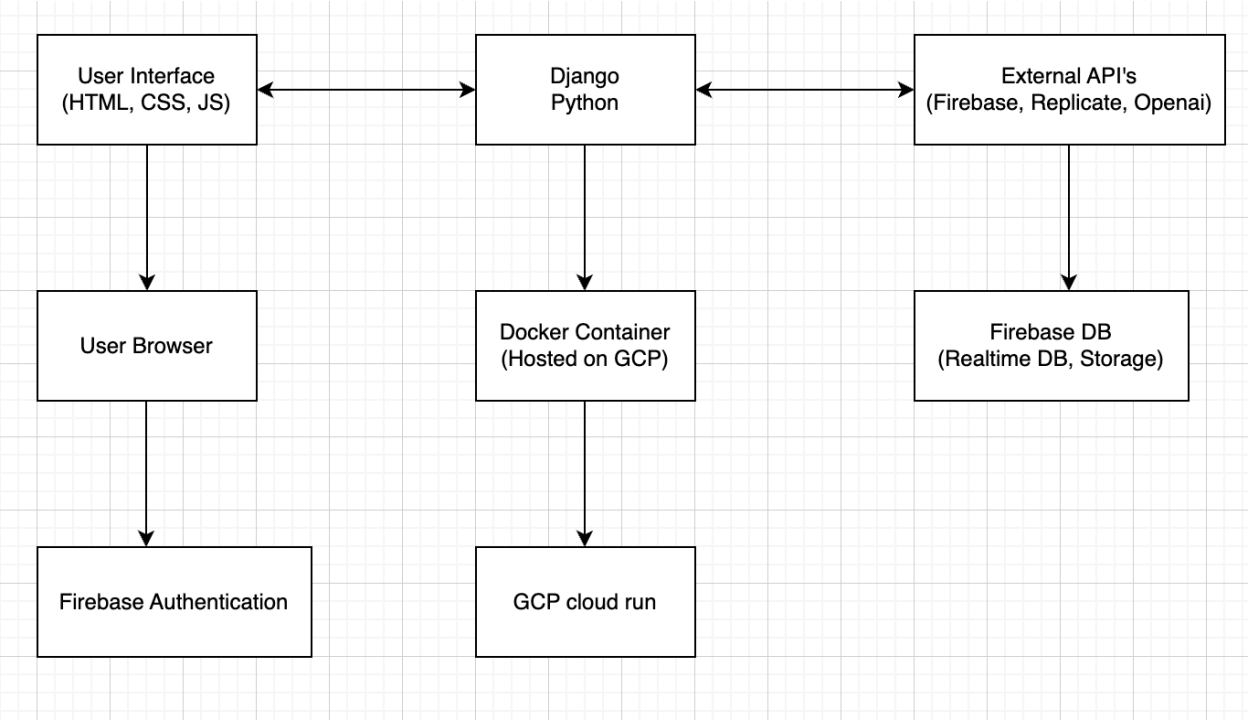
4. Data Sources Used

1. **User-Uploaded Data**:
   Users upload 15 images (5 of person1, 5 of person2, and 5 of both together) with text descriptions.
2. **Firebase**:
   Used for user authentication, session management, and storing user-specific data (e.g., model details, training images, training responses).
3. **Replicate**:
   Used for training the Flux-Dev model and generating images.
4. **OpenAI**:
   Used for generating descriptions of uploaded images.
5. **SendGrid**:
   Used for sending email notifications to users.
6. **User-Uploaded Data**:
   Users upload their own images for training custom models

5. Explanation of the Code and System Design

System Design

Below is a block diagram of the system design:

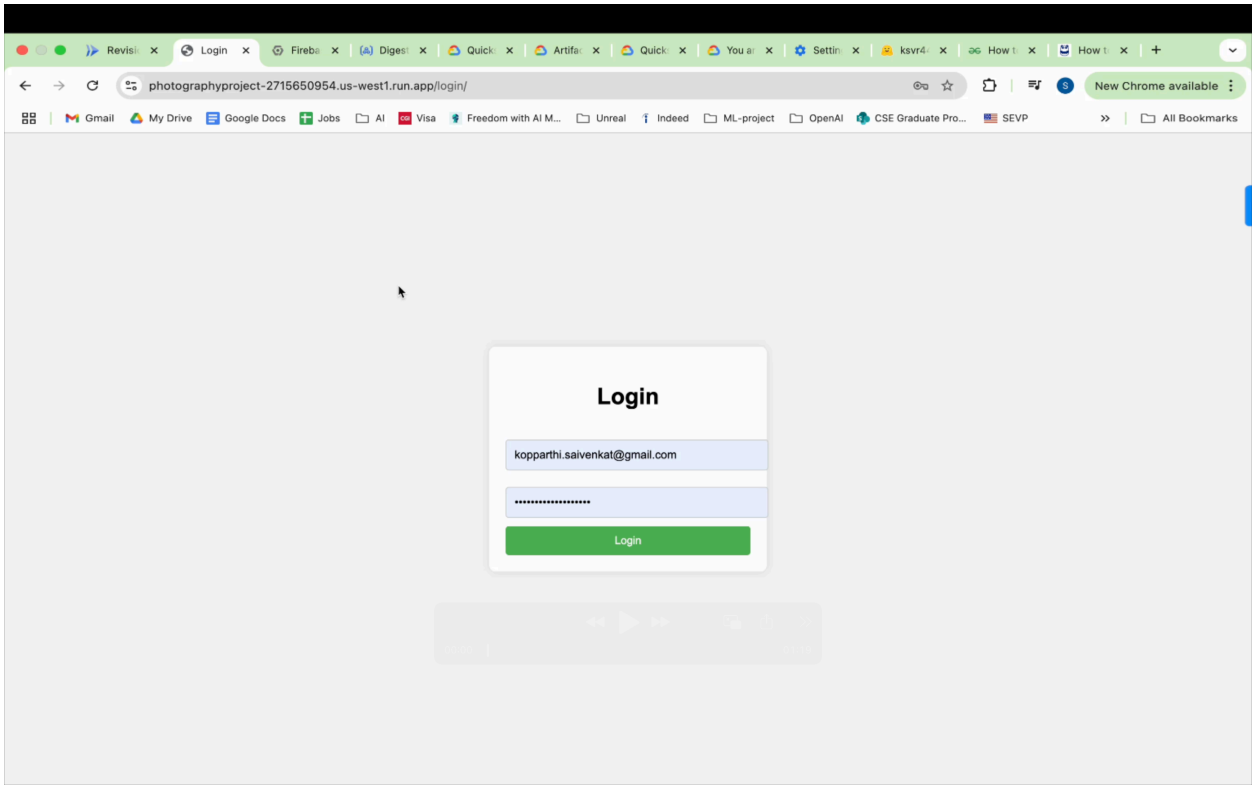


Explanation of the Code

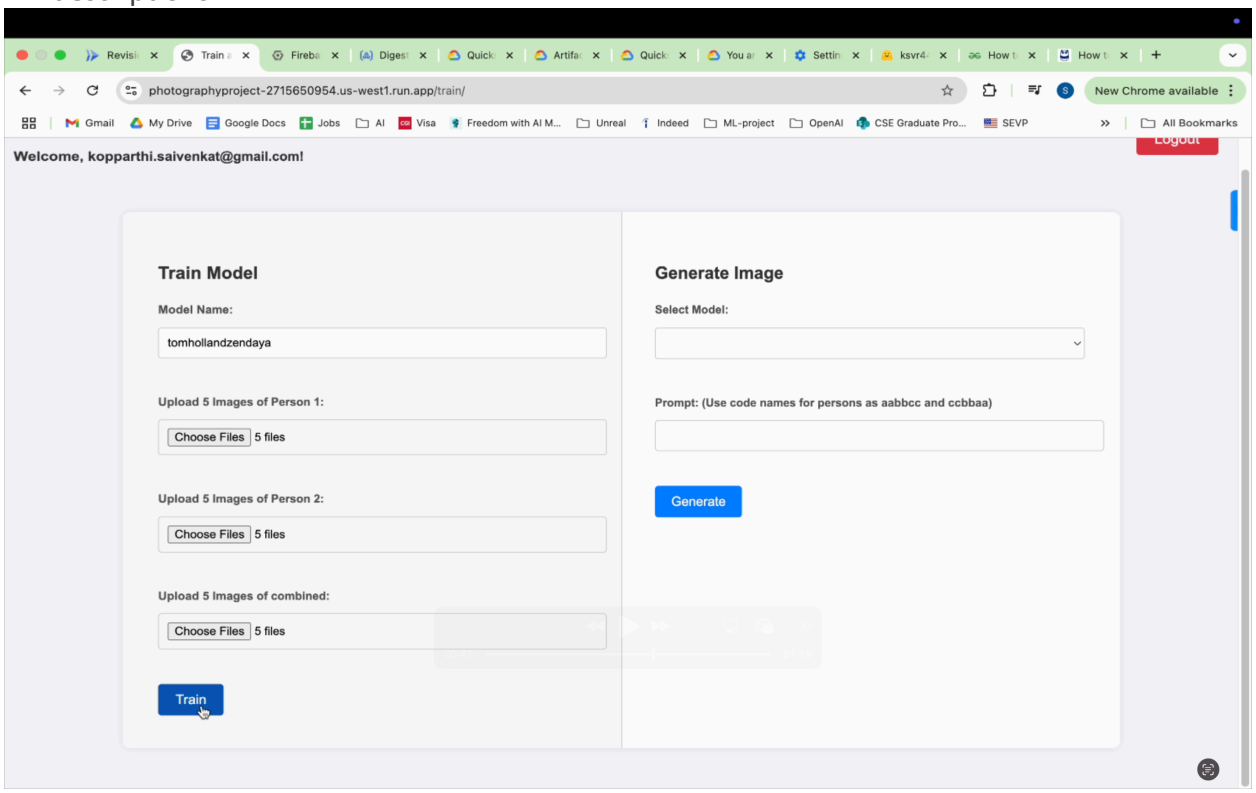
1. **User Authentication**: (login.html), views.py -> login, logout
    - Firebase handles user login and session management.
    - The user's email is stored in the session and displayed on the UI.
2. **Model Training**: views.py -> train_model(request) ->[Openai -> image descriptions]
    - Users upload images and provide a model name and trigger word.
    - The images are uploaded to Firebase Storage, and their descriptions are generated using OpenAI.
    - Replicate is used to train the custom model.
3. **Image Generation**: views.py -> generate_image(request)
    - Users select a trained model and provide a prompt.
    - Replicate generates an image based on the prompt and model.
4. **Email Notifications**: replicate_webhook(request), send_email_notification(user_email, status, model_name)
    - SendGrid sends an email to the user when training is complete.
5. **Build Docker image:** .dockerignore, docker-compose.yml
6. **Frontend**: (train.html) ,
    - The UI is built using HTML, CSS, and JavaScript.
    - Error and success messages are displayed as banners that vanish after a few seconds.

6. Sample Interactions with Your System

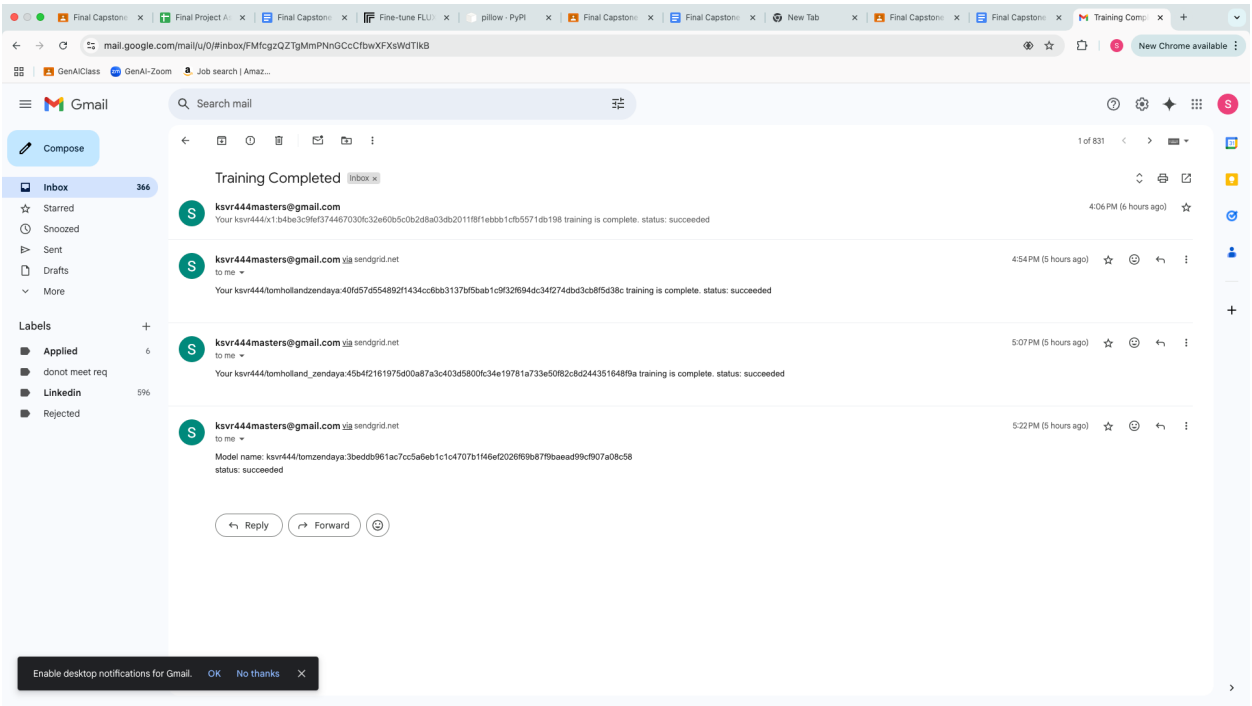
1. **User Login**:
    - The user logs in using their email and password.
    - The user's email is displayed at the top of the page.



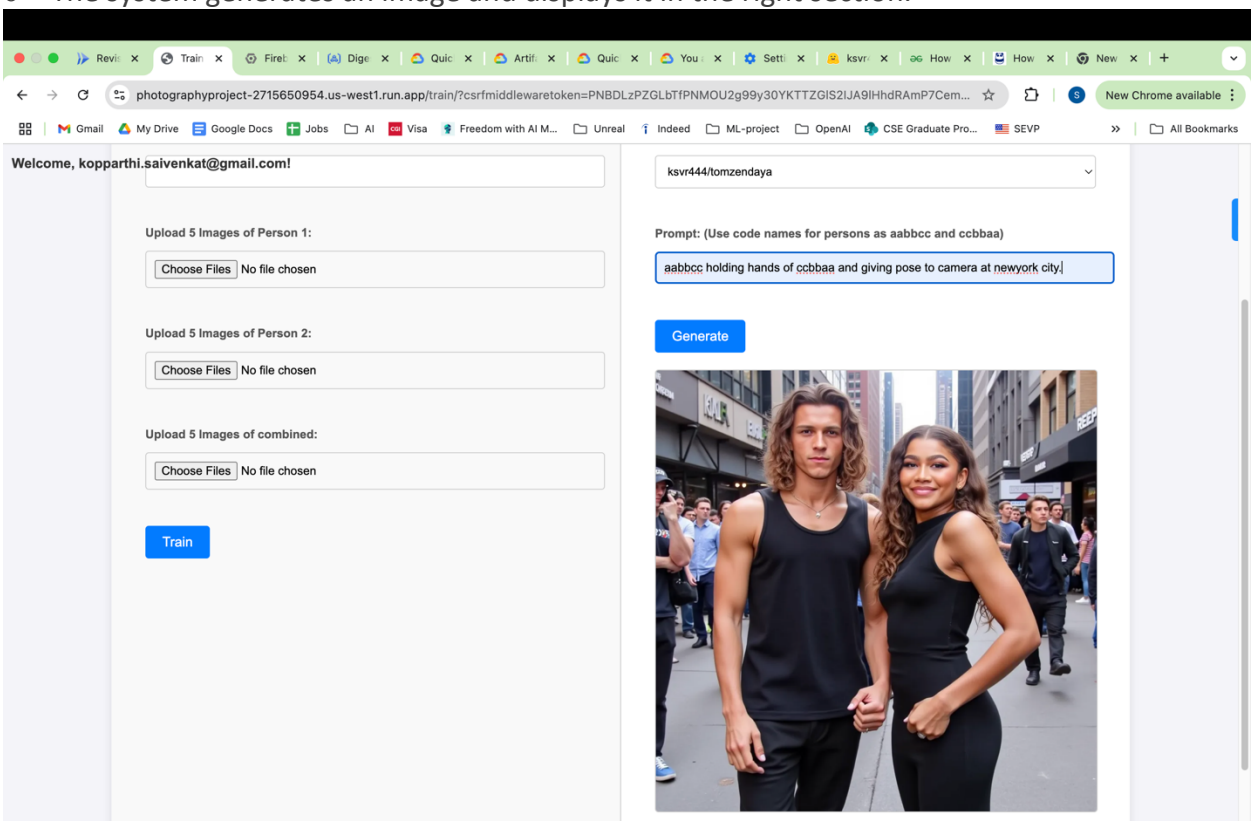
2. **Image Upload**:
    - The user uploads 15 images (5 of person1, 5 of person2, and 5 of both together) with text descriptions.

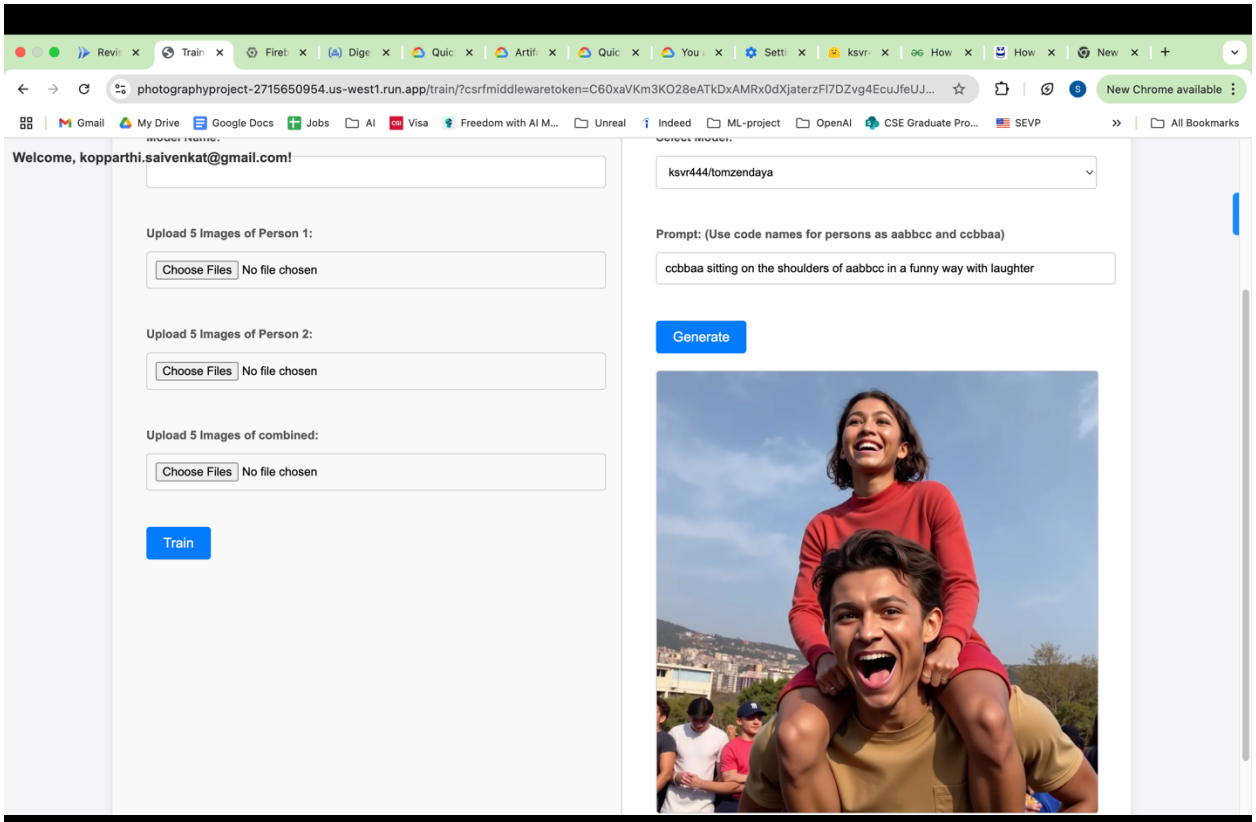


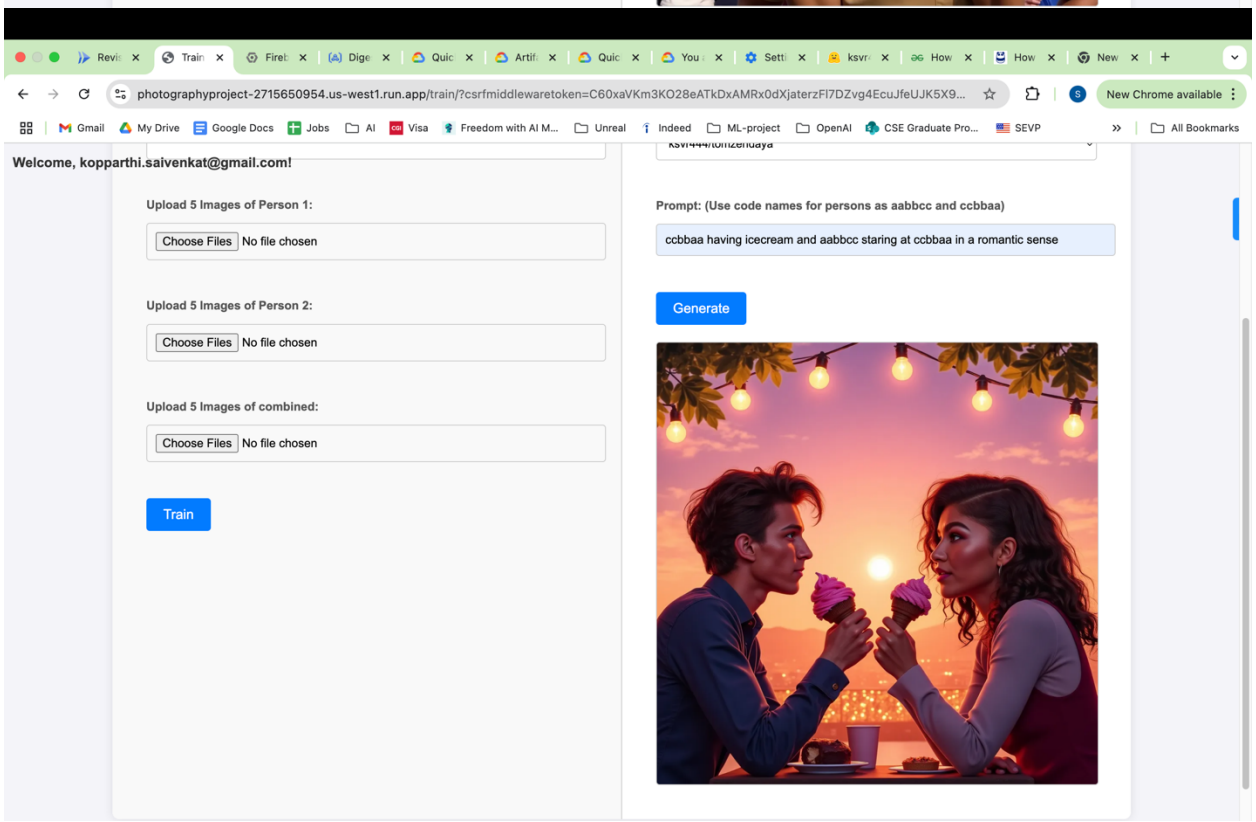
3. **Model Training**:
    - The user uploads images, provides a model name and trigger word, and clicks "Train."
    - The system trains the model and sends an email notification upon completion.



4. **Image Generation**:
   - The user selects a trained model, provides a prompt, and clicks "Generate."
   - The system generates an image and displays it in the right section.







7. Limitations

1. **Dependency on External APIs**:

    - The system relies on Firebase, Replicate, and OpenAI, which may introduce latency or downtime.
2. **Limited Customization**:
    - Users cannot fine-tune hyperparameters for model training.
3. **Scalability**:
    - The system may face scalability issues with a large number of users or training requests.
4. **Cost**:
    - Using external APIs (e.g., OpenAI, Replicate) incurred costs.

8. Future Scope

1. **Advanced Model Customization**:
    - Allow users to fine-tune hyperparameters for model training.
2. **Enhanced UI/UX**:
    - Add more interactive features, such as drag-and-drop image uploads and real-time progress tracking.
3. **Build a chat bot with whats app notification**:
    - coming up with a chatbot to load images and perform training and image generation and send training notification to whatsapp.

Conclusion

This project provides a **user-friendly and scalable solution** for training Flux-Dev model and generating images. By integrating Firebase, Replicate, and OpenAI, the system offers a seamless experience for users while addressing the limitations of existing solutions. With future enhancements, the platform can become a powerful tool for AI-driven image generation and customization.