SnapEnhance: AI-Powered Image Processing with CI/CD

01.02.2025

—

Team Binary_Girls
Farhana Akter Suci
Rifah Sajida Deya
Department of Computer Science and Engineering
Jagannath University, Dhaka

Project: AI-Powered Image Processing Pipeline with CI/CD

Tech Stack

- Backend: FAST API (Python)
- Frontend: HTML, CSS, JS
- Cloud: Render(backend), Vercel (frontend)
- Docker: For containerization
- CI/CD: GitHub Actions

Goals

This project aims to combine AI, cloud computing, and automation to create an efficient and scalable image processing pipeline. Here are the main goals:

1. Cloud-Native & Scalable Architecture

- Deploy a fully cloud-based solution without relying on local servers.
- Use free-tier services (Render, Railway, Vercel) to ensure accessibility and scalability.
- Containerize the entire project using Docker for consistent deployment across different environments.

2. AI-Powered Image Processing

- Develop an API that applies AI-based image processing (e.g., grayscale, edge detection, background removal).
- Implement real-time processing so users get instant results.
- Keep the backend efficient and optimized to handle multiple users.

3. Automation with CI/CD (DevOps Implementation)

- Set up GitHub Actions for automated testing & deployment.
- Automatically deploy new changes to Render (backend) and Vercel (frontend) without manual intervention.

4. **User-Friendly Interface & Experience**

- Create a simple and intuitive UI (HTML, CSS, JS) where users can upload images easily.
- Provide a progress indicator so users know when the image is being processed.
- Allow users to download the processed image once it's ready.

5. **Security & Reliability**

- Use environment variables (.env) to secure API keys and configurations.
- Implement basic authentication to prevent spam or abuse.

Final Outcome

Goal is to create a fully functional, cloud-hosted AI image processing web app that is:

- Automated (CI/CD pipeline)
- Scalable (Deployed on cloud services)
- User-friendly (Easy UI for uploading and downloading images)
-

Plan

1. Build and Set up the backend API (Image upload & processing)
2. Dockerize the backend (Containerize for easy deployment)
3. Deploy (Backend: Render)
4. Build the frontend (User uploads image, sees processed result)
5. Dockerize the frontend
6. Deploy (Frontend: Vercel)
7. CI/CD with GitHub Actions (Automate deployment)

Tools to Use

I. **GitHub Actions** →For CI/CD automation
II. **Docker** →For containerization
III. **Render** For backend deployment
IV. **Vercel** → For frontend hosting

Implementation:

Step-1: Set up the backend API (Image upload & processing)

- Create the backend server with FAST API (Language: Python).
- Add python code to let users register and login so that images can be stored in an individual account respectively
- Add python code to take an image from user and upload it in a folder of server and let user choose which effect to apply
- Change the image by applying effects
- Save the processed image in the users dashboard.

Step-2: Dockerize the backend (Containerize for easy deployment)

- Build the Docker image [Build Docker Container with Custom Name. code: docker build -t snapenhance]
- Run the Container with a Custom Name [code: docker run -d -p 5000:5000 snapenhance]

- Check the running Containers [code: docker ps. Should see snapEnhance in the list.]
- Stop & Remove the Container (If Needed) [to stop use: docker stop snapenhance]; [to remove use: docker rm snapenhance]

Step-3: Deploy (Backend: Render)

- After Dockerization deploy in render
- After deployment, the API for the project can be found
- Use this in browser or test in Postman

Step-4: Build the frontend

- Do all front end works in frontend folder
- Create the frontend interface with HTML, CSS, JS
- Get backend API and connect with frontend to upload and fetch data from frontend

Step-5: Dockerize the frontend(Containerize for easy deployment)

- Build the Docker image
- Run the Container with a Custom Name [docker run -d -p 8080:80 snapenhance]

Step-6: Deploy (Front end: Vercel)

- After Dockerization deploy in Vercel
- Deploy from git repository simply

Step-7: CI/CD with GitHub Actions (Automate deployment)

- In .github>workflow and frontend.yml to automate deployment of frontend
- In .github>workflow and backend.yml to automate deployment of backend
- Then save
- CI/CD is implemented, now make changes in any code it can deploy itself without any manual work
- Go to project repository and in upper navbar see “Actions” or in project repository in right side see “Deployments”, press there to see how many times you deployed via CI/CD and the logs too
- Done till now.

Finally:

- To view project code go to: https://github.com/rifah07/SnapEnhance-Pro/
- To see live go to: https://snap-enhance-pro.vercel.app/
- For more detail: https://drive.google.com/drive/folders/1XoP6jKJY71JWBBPvZ9ng7TL5PQ7Qc41p?usp=drive_link