Product Generation & Publishing Pipeline

# 🚀 Project Overview

This project simulates an end-to-end e-commerce product pipeline using multiple technologies:  
1. Product Content Generator (Python + GPT API)  
2. Mock Product Visualizer (JavaScript)  
3. Fake Product Publisher (PHP)  
4. Automation Orchestrator (Python)  
  
Bonus: AI-generated tags using GPT from product image prompt.

# 📦 Folder Structure

project-root/  
│  
├── generator/ # Python content generator  
│ └── generate\_product.py  
│  
├── visualizer/ # JS-based image visualizer  
│ └── index.html  
│ └── script.js  
│  
├── publisher/ # PHP endpoint  
│ └── fake\_publisher.php  
│  
├── orchestrator/ # Python script to run everything  
│ └── run\_pipeline.py  
│  
├── output/  
│ └── product.json # Final generated product data  
│ └── product\_mockup.png # Overlaid product image  
│  
└── README.md # This file

# 🔧 Tech Stack

|  |  |  |
| --- | --- | --- |
| Task | Language | Tools/APIs Used |
| Product Content Generator | Python | OpenAI GPT API |
| Image Mock Visualizer | JavaScript | HTML5 Canvas (Browser-based) |
| Fake Publisher | PHP | JSON Receiver |
| Automation Orchestrator | Python | `requests`, `subprocess` |

# ✅ How It Works

1. Product Generator (generate\_product.py)  
 - Generates product title, description, tags using GPT.  
 - Saves result to product.json.  
  
2. Visualizer (index.html + script.js)  
 - Simulates image upload  
 - Overlays the image on a product mockup (T-shirt)  
 - Outputs final preview  
  
3. Fake Publisher (fake\_publisher.php)  
 - Receives product JSON  
 - Responds with a fake product ID  
  
4. Orchestrator (run\_pipeline.py)  
 - Triggers all the steps sequentially  
 - Sends generated product to PHP publisher  
 - Simulates the full pipeline

# 🧪 How to Run

Prerequisites:  
- Python 3.8+  
- PHP Server (e.g., XAMPP, localhost)  
- Internet connection (for GPT API)  
  
Steps:  
1. Run Python generator:  
 python generator/generate\_product.py  
  
2. Open Visualizer in browser:  
 Navigate to visualizer/index.html  
 Upload or simulate an image  
  
3. Run PHP Server:  
 php -S localhost:8000 -t publisher/  
  
4. Run Orchestrator:  
 python orchestrator/run\_pipeline.py

# 📁 Sample Output

## 📝 product.json

{  
 "title": "Retro Galaxy T-Shirt",  
 "description": "A vintage-styled tee with a cosmic vibe...",  
 "tags": ["retro", "space", "tshirt", "fashion"],  
 "image\_url": "output/product\_mockup.png"  
}

## 🖼️ Generated Image

Located in output/product\_mockup.png

# ⭐ Bonus AI Features

- AI-generated tags based on GPT prompt  
- (Optional) Captioning for uploaded image

# 📌 Notes

- Modular code with clear separation of tasks  
- Easily deployable on local systems  
- No external database used (mocking only)

# 📤 Submission Format

- GitHub Repo / ZIP file  
- All code files named and sorted  
- README file included  
- Generated JSON and mock image included

# ✨ Author

[Pilla Lohith]  
Email: pillalohith45@email.com  
GitHub: https://github.com/lohith459