ĐỀ XUẤT ĐỒ ÁN GIỮA KỲ

Web Photobooth Application - Python Backend

MỤC LỤC

---

1. TỔNG QUAN DỰ ÁN

1.1 Mô Tả Ngắn Gọn

Ứng dụng web photobooth cho phép người dùng:

- Chụp ảnh từ webcam
- Áp dụng các hiệu ứng xử lý ảnh (filters, effects)
- Sử dụng AI/ML cho face detection và background removal
- Lưu trữ và quản lý ảnh
- Tạo collage, GIF, và các content sáng tạo khác **Backend được xây dựng 100% bằng Python.**

1.2 Stack Công Nghệ

Backend (Python 100%)

- **Framework**: Flask 3.0+
- **Image Processing**: OpenCV, Pillow (PIL)
- **Machine Learning**: face-recognition, rembg, NumPy
- **Database**: SQLite
- **File Handling**: Python File I/O

Frontend (Minimal)

- **HTML5, CSS3**: Giao diện
- **JavaScript**: ~50 dòng CHỈ để access webcam

1.3 Tại Sao Chọn Web App?

Tiêu chí	Desktop App (Tkinter)	Web App (Flask)
Python %	100%	90%
UI/UX	Kém, không professional	Modern, đẹp
Deployment	Khó, chỉ local	Dễ, deploy lên cloud
Sharing	Phải copy file .exe	Chỉ cần link
Real-world	Limited	Production-ready
Portfolio value	Trung bình	Cao

Kết luận: Web app cân bằng tốt nhất giữa tính thực tế và tỷ lệ Python code.

---

2. PHÂN TÍCH VAI TRÒ PYTHON

2.1 Phân Tích Tỷ Trọng Code

Component	Ngôn ngữ	Mô tả	Tỷ trọng	Số dòng code
Backend Framework	Python	Flask routing, middleware	15%	~120
Image Processing	Python	OpenCV, Pillow algorithms	25%	~200
AI/ML Models	Python	Face detection, segmentation	15%	~120
Business Logic	Python	Validation, workflow	10%	~80
Database Layer	Python	SQLite operations	5%	~40
API Design	Python	RESTful endpoints	10%	~80
File System	Python	Save/load images	5%	~40
Utilities	Python	Helpers, decorators	5%	~40
Frontend Logic	**JavaScript**	**Camera access only**	**10%**	**~50**

→ Tổng: Python ~720 dòng (90%) vs JavaScript ~50 dòng (10%)

---

3. PHẠM VI CHỨC NĂNG

3.1 Chức Năng Cơ Bản

A. Chụp Ảnh Từ Webcam

- Giao diện web hiển thị camera stream
- Button chụp ảnh với countdown timer (3-2-1)
- Gửi ảnh lên Python server qua API

B. Xử Lý Ảnh Cơ Bản

Sử dụng Pillow/OpenCV:

- Chuyển đổi màu (Grayscale, Sepia)
- Điều chỉnh độ sáng/tương phản
- Crop và resize ảnh
- Xoay và lật ảnh
- Thêm border/frame

C. Lưu và Quản Lý Ảnh

- Lưu ảnh vào thư mục với timestamp
- Tạo thumbnail tự động
- API lấy danh sách ảnh đã chụp
- Download ảnh về máy

D. Gallery Đơn Giản

- Hiển thị grid ảnh đã chụp
- Xem full size
- Xóa ảnh

3.2 Chức Năng Nâng Cao (Python OpenCV)

A. Filters & Effects Nâng Cao

1. Blur Effects:

- Gaussian Blur
- Motion Blur
- Box Blur

2. Edge Detection:

- Canny Edge Detection
- Sobel Operator

3. Artistic Filters:

- Cartoon Effect
- Pencil Sketch Oil
- Painting Effect

4. Color Grading:

- Vintage
- Cool Tone
- Warm Tone

5. Instagram-style Filters:

- Nashville
- Valencia
- X-Pro II

** Sử dụng ma trận convolution, color space transformation

B. Template Photobooth (4 ảnh ghép)

- Chụp 4 ảnh liên tiếp với countdown
- Python ghép thành layout 2x2 hoặc 4x1
- Thêm logo, text, khung trang trí

C. Sticker Overlay

- Upload stickers (PNG với alpha channel)
- Python composite stickers lên ảnh gốc
- Vị trí custom hoặc auto (trên mặt nếu có face detection)

D. Text Watermark

- Thêm text vào ảnh
- Custom font, size, color, position
- Sử dụng Pillow ImageDraw

3.3 Chức Năng Cao Cấp

A. Face Detection & Recognition (Python ML)Sử dụng: face-recognition library, OpenCV Haar Cascades

Features:

- Tự động phát hiện khuôn mặt trong ảnh
- Đếm số người trong ảnh
- Crop ảnh theo khuôn mặt (auto-focus)
- Face landmarks detection (mắt, mũi, miệng)

Ứng dụng:

- Auto-focus vào khuôn mặt
- Beauty filters (làm mờ da, to mắt)
- Thêm sticker đúng vị trí (mắt kính, mũ, râu)

B. Background Removal/Blur (AI-powered)

Sử dụng: rembg library (U2-Net model) **Features:**

- Tự động xóa phông nền
- Thay background mới
- Blur background (bokeh effect)
- Green screen effect

C. Batch Processing

- Upload nhiều ảnh cùng lúc
- Áp dụng filter/effect cho tất cả
- Export as ZIP file
- Progress bar realtime

D. Advanced Features

1. GIF Creator:

2. Video to Frames: Extract frames từ video

3. QR Code Generator: Tạo QR code link đến ảnh

4. Image Compression: Tối ưu dung lượng ảnh

---

4. KIẾN TRÚC HỆ THỐNG

4.1 Project Structure

```
photobooth-app/photobooth-app/
├── app.py                  # Flask main application ├── app.py                  # Flask main application
├── requirements.txt        # Python dependencies ├── requirements.txt        # Python dependencies
├── config.py               # Configuration settings ├── config.py               # Configuration settings
```

```
│
├── models/                 # Business Logic Layer (Python)
│   ├── __init__.py
│   ├── image_processor.py     # Image processing classes
│   ├── face_detector.py       # Face detection logic
│   ├── filter_engine.py       # Filter algorithms
│   └── database.py            # Database operations
│
├── routes/                 # API Layer (Python)
│   ├── __init__.py
│   ├── api.py                 # RESTful API endpoints
│   └── views.py               # Template routes
│
├── utils/                  # Utilities (Python)
│   ├── __init__.py
│   ├── helpers.py             # Helper functions
│   ├── validators.py          # Input validation
│   └── decorators.py          # Custom decorators
│
├── static/                 # Frontend Assets
│   ├── css/
│   │   └── style.css
│   ├── js/
│   │   └── camera.js          # ~50 lines - Camera only!
│   └── uploads/               # Saved images
│       ├── originals/
│       ├── processed/
│       └── thumbnails/
│
├── templates/              # HTML Templates
│   ├── index.html             # Main page
│   ├── gallery.html           # Gallery page
│   └── base.html              # Base template
│
├── tests/                  # Unit Tests (Python)
│   ├── test_filters.py
│   ├── test_api.py
│   └── test_face_detection.py
│
└── docs/                   # Documentation
    ├── README.md
    ├── API.md
    └── ALGORITHMS.md
```

4.2 RESTful API Design

Core Endpoints:

```
POST   /api/capture            # Nhận và lưu ảnh
POST   /api/filter/<name>      # Áp dụng filter
POST   /api/collage            # Tạo collage
GET    /api/gallery            # Lấy danh sách ảnh
GET    /api/photo/<id>         # Lấy ảnh cụ thể
DELETE /api/photo/<id>         # Xóa ảnh
POST   /api/face-detect        # Phát hiện khuôn mặt
POST   /api/remove-bg          # Xóa background
POST   /api/create-gif         # Tạo GIF
POST   /api/batch-process      # Xử lý nhiều ảnh
```

---

5. KỸ NĂNG PYTHON THỂ HIỆN

5.1 Web Development với Python

Flask Framework - Routing, Middleware, Error Handling:

5.2 Image Processing - Core Python Skills

OpenCV & NumPy - Ma trận và xử lý pixel:

5.3 Machine Learning Integration

Face Detection với dlib/face_recognition:

5.4 Object-Oriented Programming

Design Patterns, Inheritance, Encapsulation:

---

6. KẾ HOẠCH THỰC HIỆN CHI TIẾT

6.1 TUẦN 1: Setup & Core Features

Ngày 1-2: Project Setup

- [ ] Cài đặt Python 3.10+
- [ ] Tạo virtual environment: `python -m venv venv`
- [ ] Cài đặt dependencies cơ bản:

```bash
pip install Flask==3.0.0
pip install Pillow==10.1.0
pip install opencv-python==4.8.1
pip install numpy==1.24.3
```

- [ ] Tạo project structure theo mục 4.1
- [ ] Setup Git repository
- [ ] Tạo `.gitignore` cho Python

Ngày 3-4: Camera Capture

- [ ] Tạo `templates/index.html` với camera interface
- [ ] Implement JavaScript (~50 dòng) cho:
  - getUserMedia() để access webcam
  - Canvas để capture frame
  - Convert canvas to base64
  - AJAX gửi lên server
- [ ] Tạo Flask route `/api/capture` để nhận base64 image
- [ ] Decode base64 và lưu file với timestamp
- [ ] Test camera capture end-to-end
- [ ]

Ngày 5-6: Basic Filters

- [ ] Implement 5 filters cơ bản:
  1. Grayscale
  2. Sepia
  3. Brightness adjustment
  4. Contrast adjustment
  5. Blur
- [ ] Tạo Flask route `/api/filter/<filter_name>`
- [ ] Test từng filter

Ngày 7: Gallery & Database

- [ ] Setup database
- [ ] Tạo table `photos` theo schema 4.3
- [ ] Implement save metadata function
- [ ] Tạo gallery page hiển thị thumbnails
- [ ] Test save và retrieve

Deliverable Week 1:

- Camera capture hoạt động
- 5 basic filters
- Save images với metadata
- Basic gallery
- **Estimated Score: 6.5-7.0đ**

6.2TUẦN 2: Advanced Processing

Ngày 8-9: Advanced Filters (Part 1)

- [ ] Implement 5 filters phức tạp:
  1. Cartoon effect (như code example ở mục 5.2)
  2. Pencil sketch
  3. Edge detection (Canny)
  4. Emboss effect
  5. Vintage filter

Ngày 10-11: Advanced Filters (Part 2)

- [ ] Implement 5 Instagram-style filters:
  1. Nashville (warm tones, contrast boost)
  2. Valencia (faded, warm)
  3. X-Pro II (high contrast, cool shadows)
  4. Walden (vintage, increased exposure)
  5. Kelvin (warm, amber glow)

Ngày 12-13: Collage Maker

- [ ] Implement 4-photo capture sequence với countdown
- [ ] Tạo function `create_collage()` với layouts:
  - 2x2 grid
  - 4x1 horizontal strip
  - 1x4 vertical strip

- Thêm spacing, borders, background colors
- Thêm text overlay (date, event name)

Ngày 14: Stickers & Watermarks

- [ ] Implement sticker overlay với PNG alpha channel
- [ ] Tạo function `add_text_watermark()`
- [ ] Test với different positions và fonts

Deliverable Week 2:

- 15+ filters hoạt động
- Collage maker với multiple layouts
- Stickers và watermarks

6.4 TUẦN 3: AI Features

Ngày 15-16: Face Detection Setup

- [ ] Cài đặt face-recognition library:

```bash
pip install face-recognition==1.3.0
pip install dlib  # May need cmake
```

- [ ] Implement `/api/detect-faces` endpoint
- [ ] Test face detection với nhiều ảnh

Ngày 17-18: Face-based Features

- [ ] Auto-crop theo khuôn mặt
- [ ] Face counter
- [ ] Face landmarks detection
- [ ] Auto-place stickers (glasses, hats) dựa vào landmarks

Ngày 19-20: Background Removal

- [ ] Cài đặt rembg:

```bash
pip install rembg==2.0.50
```

- [ ] Implement `/api/remove-bg` endpoint
- [ ] Tạo function thay background
- [ ] Implement blur background (bokeh effect)

Ngày 21: Advanced Features

- [ ] GIF creator từ nhiều ảnh
- [ ] QR code generator
- [ ] Batch processing (apply filter to multiple images)

Deliverable Week 3:

- Face detection hoạt động
- Background removal/blur
- GIF creator
- Batch processing

6.5 TUẦN 4: Polish & Testing

Ngày 22-23: Code Refactoring

- [ ] Refactor code theo OOP patterns (như mục 5.4)
- [ ] Tạo abstract Filter base class
- [ ] Implement Strategy pattern cho filters
- [ ] Add comprehensive error handling
- [ ] Add logging

Ngày 24-25: Testing

- [ ] Write unit tests cho filters
- [ ] Test API endpoints với Postman
- [ ] Integration testing
- [ ] Performance testing
- [ ] Fix bugs

Ngày 26-27: UI/UX Improvements

- [ ] Improve gallery UI

- [ ] Add loading indicators
- [ ] Add filter preview thumbnails
- [ ] Responsive design
- [ ] Add animations

Ngày 28: Documentation

- [ ] Write README.md với:
  - Project description
  - Installation instructions
  - Usage guide
  - Screenshots
- [ ] Write API documentation
- [ ] Add code comments
- [ ] Create requirements.txt

Deliverable Week 4:

- Clean, well-structured code
- Comprehensive error handling
- Tests passing
- Good documentation

6.6 TUẦN 5 (Optional): Excellence

Ngày 29-30: Algorithm Documentation

- [ ] Write detailed algorithm explanations
- [ ] Include mathematical formulas
- [ ] Create diagrams for complex algorithms
- [ ] Explain design patterns used

Ngày 31-32: Presentation Preparation

- [ ] Create presentation slides
- [ ] Prepare demo script
- [ ] Record backup demo video
- [ ] Practice Q&A responses

Ngày 33-35: Final Polish

- [ ] Performance Optimization
- [ ] Code review và cleanup
- [ ] Final testing
- [ ] Deploy demo version (optional)

Deliverable Week 5:

- Excellent documentation
- Professional presentation
- Confident Q&A handling

---

7. YÊU CẦU KỸ THUẬT

7.1 Python Libraries Cần Cài Đặt

requirements.txt:

```
# Core Framework
Flask==3.0.0
Flask-CORS==4.0.0

# Image Processing
Pillow==10.1.0
opencv-python==4.8.1
numpy==1.24.3

# Machine Learning
face-recognition==1.3.0
rembg==2.0.50

# Utilities
python-dotenv==1.0.0
qrcode==7.4.2

# Testing (Optional)
pytest==7.4.0
pytest-flask==1.2.0
```

Installation command:

```
bash

pip install -r requirements.txt
```

7.2 System Requirements

Minimum:

- Python 3.10+
- 4GB RAM
- 2GB free disk space
- Webcam/Camera device
- Modern web browser (Chrome 90+, Firefox 88+)

Recommended:

- Python 3.11+
- 8GB RAM
- 5GB free disk space
- Good lighting for face detection
- Fast internet for library downloads

7.3 Troubleshooting Common Issues

Issue 1: face-recognition installation fails

```
bash

# Solution: Install dlib first
# Windows: Download wheel from https://github.com/z-mahmud22/Dlib_Windows_Python3.x
pip install dlib-19.24.0-cp310-cp310-win_amd64.whl
pip install face-recognition
```

Issue 2: OpenCV import error

```
bash

# Solution: Uninstall and reinstall
pip uninstall opencv-python
pip install opencv-python-headless==4.8.1
```

Issue 3: rembg model download issues

```
bash

# Solution: Manually download model
# Set environment variable
export U2NET_HOME=/path/to/models
```

7.4 Development Environment Setup

```
# 1. Create project directory
mkdir photobooth-app
cd photobooth-app

# 2. Create virtual environment
python -m venv venv

# 3. Activate virtual environment
# Windows:
venv\Scripts\activate
# Mac/Linux:
source venv/bin/activate

# 4. Install dependencies
pip install -r requirements.txt

# 5. Create .env file
echo "FLASK_APP=app.py" > .env
echo "FLASK_ENV=development" >> .env
echo "SECRET_KEY=your-secret-key-here" >> .env

# 6. Initialize database
python init_db.py

# 7. Run application
flask run
```
