재미한인정보과학자협회
KOCSEA

Mini-Project 2: Passport Shop
- Engineering Project -
(40 mins)

KOCSEA Vibe (AI) Coding Winter Camp

KOCSEA Education Committee

Created by Jeongkyu Lee, Last Update: 2025-11.28

Setup

- LLM: Select Any available LLM
  - E.g., ChatGPT, Gemini, Llama, Claude ...

- IDE: Select your preferred IDE
  - PyCharm, VSCode

Learn from this project

- Software Engineering Process model:
  - PassportShop Follows an **Incremental Development Model**
  - For example:
    1. Load and display an image
    2. Face detection
    3. Cropping and centering
    4. Background adjustment (white background)
    5. Resize to 2x2 inches (600×600)
    6. Compliance validation (face size, brightness, alignment)
  - Students experience this directly as they watch their PassportShop evolve from:
    - a basic image loader → into a face detector → into an auto-cropper → into a full compliance checker.

Vibe Coding Table

#	Time	Activity	Vibe Coding Technique	Sample Prompt	Expected Output
0	**0:00–0:05**	**Requirements Analysis** — Understand U.S. passport photo rules	–	“What are the official U.S. passport photo requirements? Include dimensions, background, head position, and facial expression.”	List of compliant requirements (2×2 inches, 600×600 px, white background, centered head, neutral expression, etc.)
1-1	**0:05–0:10**	**Design 1 — System Overview**	**T1: Generative Design Partner**	“Suggest a component diagram for a PassportShop app. Identify modules such as Loader, FaceDetector, Cropper, BackgroundCleaner, Resizer, and Validator.”	High-level system architecture + pipeline diagram
1-2	**0:10–0:13**	**Design 2 — Detailed Face Detection Flow**	**T1: Generative Design Partner**	“Provide a design flow for detecting and centering a face in an image. Include input, process steps, and expected output.”	Flowchart describing detect → crop → align → output
2-1	**0:13–0:17**	**Implementation 1 — Image Loader Module**	**T2: Modular Assembly**	“Generate Python code for an ImageLoader module that loads an image and returns it in a usable format. No GUI yet.”	ImageLoader module code
2-2	**0:17–0:21**	**Implementation 2 — Face Detection Module**	**T2: Modular Assembly**	“Generate Python code for a FaceDetector module using OpenCV Haar cascades. Return a bounding box. Keep it independent.”	FaceDetector module with detect_face()
2-3	**0:21–0:25**	**Implementation 3 — Auto Cropper Module**	**T2: Modular Assembly**	“Generate a Cropper module that takes the face bounding box and produces a centered crop.”	Cropper module with crop_to_face()
2-4	**0:25–0:29**	**Implementation 4 — Background Adjustment Module**	**T2: Modular Assembly**	“Create a BackgroundCleaner module that whitens the background while keeping the face natural.”	BackgroundCleaner module
2-5	**0:29–0:33**	**Implementation 5 — Combined App**	**T2: Modular Assembly**	“Create main.py that create a pipeline using ImageLoader, FaceDetector, Cropper and BackgroundCleaner."	
3	**0:33–0:40**	**Testing — Unit Tests for Modules**	**T4: Automated Test Generation**	“Generate Python unittest cases that verify: (1) image loads, (2) face detection returns a box, (3) cropped output is centered, (4) resized image is 600×600.”	Automated test file with assertions
4	**0:40–0:45**	**Verification & Maintenance — Debug Iteration**	**T3: Debugging Collaboration**	“My app crashes when no face is detected. Explain why and fix the code by adding error handling.”	Explanation + improved code with try/except or fallback behavior
2-6	**0:45–0:50**	**Implementation 6 — Final Compliance Validator**	**T2: Modular Assembly**	“Generate a Validator module that checks the final image for correct size, background brightness, and head ratio according to U.S. passport rules.”	Validator module confirming compliance

PassportShop

- PassportShop
- A Python application to generate compliant passport photos
- “Generate compliant passport photos with ease”

Project Overview

- **Goal:**
  - Create an app that automatically produces passport photos satisfying U.S. Department of State requirements.
- **Key Features:**
  - Auto crop and resize to 2x2 inches (600x600 pixels)
  - Detect and center face
  - Adjust background to white
  - Validate compliance
- **Target Users:**
  - Individuals needing digital passport photos for U.S. passport applications.

Software Engineering Phases:

- Requirements Analysis
- Design (T1: Generative Design Partner)
- Implementation (T2: Modular Assembly)
- Testing (T4: Automated Test Generation)
- Verification & Maintenance (T3: Debugging Collaboration)

0. Requirements Analysis

- Ask LLM
  - Requirement of US Passport photo
  - For example:

```
What are the official U.S. passport photo
requirements? Include dimensions, background, head
position, and facial expression
```

  - Keep the answers

1. Design (T1: Generative Design Partner)

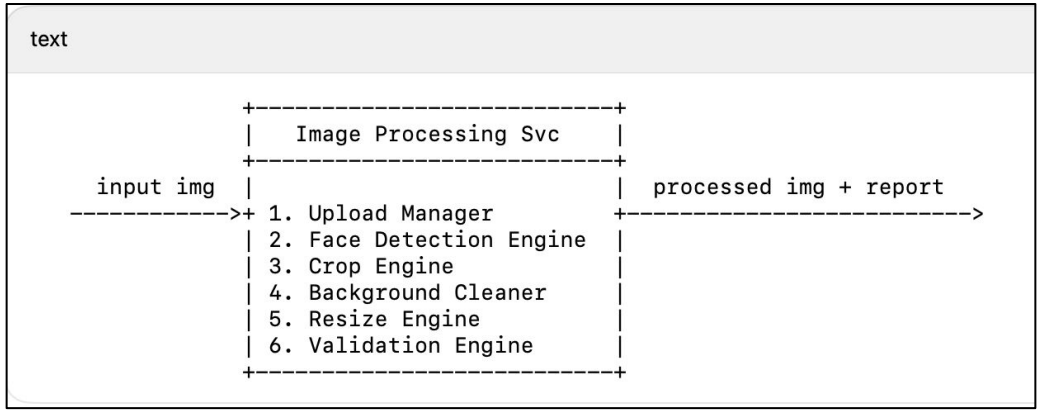
- Sample Prompt 1

"Suggest a component diagram for a PassportShop app. Identify modules that allow users to
- Upload an image
- Detect a face
- Crop an image
- Clean background
- Resize
- Validate"

- Expected Output

- List of modules and their functions
- Suggested tools/libraries
- Data flow diagram or architecture outline
- Example interfaces between components

text



1-1. Design (T1: Generative Design Partner)

- Sample Prompt 2

```
I’m developing a Python app called PassportShop that generates U.S. passport photos automatically.
The app should allow users to upload an image, detect and crop the face, adjust the background to white, and verify photo compliance (size, head position, lighting).
Please propose a modular architecture for this system, including main components, their responsibilities, and data flow between modules.
Also suggest which Python libraries (e.g., OpenCV, Pillow, face-recognition, tkinter) to use for each module.
Finally, show the architecture visually in a text-based diagram or structured outline.”
```

- Expected Output

- List of modules and their functions
- Suggested tools/libraries
- Data flow diagram or architecture outline
- Example interfaces between components

Summary of Sample 1 Output

Module	Key Function	Main Libraries
`ui.py`	User interface & control	tkinter
`image_loader.py`	Load & preprocess images	Pillow
`face_cropper.py`	Detect & crop face	OpenCV, face-recognition
`background_editor.py`	Adjust/replace background	OpenCV, numpy
`compliance_checker.py`	Validate photo specs	OpenCV, numpy
`exporter.py`	Save final image	Pillow
`utils.py`	Logging & helpers	logging

Design (T1: Generative Design Partner)

- Sample Prompt 3

```
Python app that creates U.S.
passport-compliant photos.
I need help designing the user interface
and workflow using tkinter including
these steps:
- Upload photo
- Auto crop and background fix
- Preview and validate compliance
-  Save/export result

Feel free to give suggestions for
improving usability and accessibility.
```

- Expected Output

- Proposed UI screen structure (with tkinter widget ideas)
- Text-based or ASCII wireframe
- Workflow diagram or sequence
- UX recommendations

Summary of Sample 2 Output

Step	Action	Result
1	Upload	Loads image + validates size
2	Auto Crop	Detects face, centers it
3	Background Fix	Replaces background with plain white
4	Preview & Validate	Shows compliance indicators
5	Save	Exports compliant image or printable sheet

2-1. Implementation 1 — Image Loader Module (T2: Modular Assembly)

- If needed, adjust modules suggested in ”Design”
  - Merge, Split, Add or Remove
- Sample Prompt

```
“Generate Python code for an ImageLoader module that loads
an image and returns it in a usable format. No GUI yet”
```

- Create python files as suggested
- Debug your Code
  - Any Errors? Frequent errors:
    - ModuleNotFoundError: No module named 'pygame’
      - $ pip3 install pygame
    - IndentationError: unexpected indent
      - Adjust indentation, i.e., 4 spaces
  - Any Problems or Issues?

2-2. Implementation 2 — Face Detection Module (T2: Modular Assembly)

- If needed, adjust modules suggested in ”Design”
  - Merge, Split, Add or Remove
- Sample Prompt

```
“Generate Python code for a FaceDetector module using OpenCV
Haar cascades. Return a bounding box. Keep it independent”
```

- Create python files as suggested
- Debug your Code

2-3. Implementation 3 — Face Detection Module (T2: Modular Assembly)

- If needed, adjust modules suggested in ”Design”
  - Merge, Split, Add or Remove
- Sample Prompt

```
“Generate a Cropper module that takes the face bounding box
and produces a centered crop”
```

- Create python files as suggested
- Debug your Code

2-4. Implementation 4 — Background Adjustment Module (T2: Modular Assembly)

- If needed, adjust modules suggested in ”Design”
  - Merge, Split, Add or Remove
- Sample Prompt

```
“Create a BackgroundCleaner module that whitens the
background while keeping the face natural”
```

- Create python files as suggested
- Debug your Code

2-5. Implementation 5 — Combined App (T2: Modular Assembly)

- If needed, adjust modules suggested in ”Design”
  - Merge, Split, Add or Remove
- Sample Prompt

```
“Create main.py that create a pipeline using ImageLoader,
FaceDetector, Cropper and BackgroundCleaner”
```

- Create python files as suggested
- Debug your Code

3. Testing — Unit Tests for Modules (T4: Automated Test Generation)

- Takes several passport photos
- Download several passport photos from internet
- Sample Prompt

```
“Generate Python unittest cases that verify: (1) image
loads, (2) face detection returns a box, (3) cropped output
is centered, (4) resized image is 600×600.”
```

4. Verification & Maintenance — Debug Iteration (T3: Debugging Collaboration)

- Test error images, such as no face, small face . . .
- Sample Prompt when no face is detected

```
“My app crashes when no face is detected. Explain why and
fix the code by adding error handling.”
```

2-6. Implementation 6 — Final Compliance Validator (T2: Modular Assembly)

- If needed, adjust modules suggested in ”Design”
  - Merge, Split, Add or Remove
- Sample Prompt

```
“Generate a Validator module that checks the final image for
correct size, background brightness, and head ratio
according to U.S. passport rules”
```

- Create python files as suggested
- Debug your Code

5. Order to Walgreen photo

Discussion

- Topic 1: More functions or improved app
  - For example,
    - Auto Lighting & Contrast Correction
    - Smart Head Position Checker
    - Background Replacement with Smart Segmentation
    - Batch processing
    - Others …
- Topic 2: UI/UX what about GUI?