Application Challenges Fall 2025

- Developer Challenge
- Designer Challenge
- Product Manager Challenge

Developer Challenge

Context and Problem Statement

SecureLog is an non profit organization that provides encryption services. They want to revamp their encryption service with modern technologies. Build a full stack application that encrypts and decrypts payloads while providing searchable logs to past requests.

Requirements

UI

Create a React application with the ability to do the following:

- Enter an encryption key and a payload, call this service's API, log the request, and display the encrypted data
- Enter a decryption key and a payload, call this service's API, log the request, and display the decrypted data
- Paginated view of past requests

Our evaluation will not only be based on the design and layout of the application. However, we are looking for good UX and error handling.

API

Create a FastAPI or SpringBoot API that meets the following details:

1. POST /api/v1/encrypt

- Accept a JSON object containing { key: str, data: str }
- Respond with { data: str } where data is the resulting encrypted data from encrypting the data with the public key
- POST /api/v1/decrypt

- Accept a JSON object containing { key: str, data: str } where key is a private key and data is some encrypted data from private key's keypair
- Respond with { data: str } where data is the resulting data from decrypting the payload with the private key
- GET /api/v1/logs

- Paginate logs of requests made to this service, include two URL parameters: size (log count) and offset (number of logs from the beginning)

- Logs should be of the following type { id: str, timestamp: datetime, ip: str, data: str }
  - ID should be a UUID
  - Note that datetime should be a UNIX timestamp
- The response of this endpoint should be a collection of logs

PostgreSQL should be used to store logs. Logs should match the format defined in the API section.

The application as a whole should be containerized. Use docker-compose to spin up an instance of PostgresSQL and create a Dockerfile to build and image of your API. Include a linter workflow for both the web and server in your repository.

Using github actions include a linter workflow for the front-end and backend. The action should be triggered on every push.

Submission Guidelines

- Github repository link. The repo should contain both the frontend and backend (Remember to make it public so we can access)
- Deployed URL (optional, bonus): Deploy your app in AWS or any other provider of your choice.
- Add a README.md that includes installation and development instructions. A brief overview of how the application works. Bonus points if you include an architecture diagram.
- The repository should contain a docker-compose.yaml to run the full stack application (web + api + db), and a Dockerfile for the frontend and backend.
- .github/workflows/web-lint.yml and .github/workflows/server-lint.yml
- Repository structure

```
.github/workflows/
├web-lint.yml
    server-lint.yml
README.md
docker-compose.yml
web/
├package.json
    package-lock.json
    src/
    Dockerfile
server/
├src/
    Dockerfile
```

- ***You must submit your repo and deployment link via this Google Form***

SUBMISSION DEADLINE: October 10th @ 11:59 PM

Tips

- Get the ecryption/decryption working first, then add logging, pagination, and finally polish the UI/UX
- Be clear in the README.md. Include commands for steup, a description of each service, adn sample API requests
- Show clear messages in the UI for invalid keys, empty payloads, or server errors instead of generic errors
- Focus on UX details, add loading states, disable buttons while requests are being served
- Make sure your gh workflows actually run and pass

Designer Challenge

Context and Problem Statement

Craigslist is an advertisement website founded in 1995 by Craig Newman. Despite being one of the most visited sites on the internet (ranking as the 19th most visited website globally in 2023), its UI/UX design has remained the same since the early days.

Your task will be to redesign the homepage and classified listing page of the site, with a focus on modernizing the UI while maintaining the simplicity that defines the platform.

Requirements

- The design shall be completed in Figma.
- A design system shall be implemented that defines the visual style of the website, including colors, typography, and spacing. See example.
- The submission shall include wireframes and mockups for each section (homepage and classified listing page) and each device (mobile, desktop). See example of a wireframe.

Submission Guidelines

- Create a new Figma file.
- The name of the file shall be "Your Name - Designer Challenge F25" (e.g. "Ivan Farfan - Designer Challenge F25")
- The file must include two Figma pages: Wireframes and mockups.
- ***You must submit your Figma link via this Google Form.***

SUBMISSION DEADLINE: October 10th @ 11:59 PM

Tips

- Start by creating the design system. Consider the colors and typography you want to use throughout the entire site.
- Seek inspiration by exploring other designs. A useful resource is appshots.design
- Develop the desktop wireframes first, then adapt them for mobile versions.
- Base your mockups on the wireframes you created.
- Use brand assets (e.g. logos) from the original Craigslist site to make your mockups more realistic.

Product Manager Challenge

Part 1 – Product Scenario

Your team has been tasked with creating a **platform that helps students better collaborate on coursework and projects**.

1. **Problem Exploration**
   - Define **two key pain points** students face with collaboration today.
   - Describe your approach to validating whether these are real and worth solving.
2. **Vision & Features**
   - Propose a **vision statement** for the platform.
   - List **3 major features** you would prioritize, and explain why (reference feasibility, impact, and effort).
3. **MVP Definition**
   - If you had only **6 weeks and a small 4-person engineering team**, what would your **minimum viable product** look like?
   - Provide a **clear prioritization rationale** (e.g., using MoSCoW, RICE, or another framework).

Part 2 – Execution

1. **Roadmap & Planning**
   - Create a **high-level 8-week roadmap** breaking down 3 phases (e.g., discovery, MVP build, iteration).
   - Assign approximate ownership (e.g., engineers, designers, PM responsibilities).
2. **Communication & Proactivity**
   - Draft a short **status update email** to your team at the midpoint of the project when you realize one of the features may be delayed. Show how you would:
     - Keep morale positive
     - Communicate clearly
     - Suggest next steps
3. **Technical Awareness**
   - Suppose engineers propose two implementation options for a core feature:
     - **Option A**: Quick to build but minimal scalability.
     - **Option B**: More complex but allows growth and integrations.
   - Which would you choose for an MVP? Justify your decision based on product management principles.
4. **Ticket Cutting**
   - Please **create 2 tickets/tasks** for one part of one of your potential features
   - The Ticket should include:

- User Story
- Acceptance Criteria
- Technical Details
- Out of Scope
- Level of Effort in the form of story points or estimated development time with **justification**

Submission Guidelines

Submission Requirements

PDF with:

1. **Product Scenario**
    1. PDF with:
        1. 2 Key Pain Points
        2. Vision Statement
        3. 3 Prioritized Features
        4. MVP with included features
        5. Prioritization Rationale
2. **Execution**
    1. 8 Week roadmap
    2. Status update email
    3. Option A or Option B and justification

Public Github Repository including:

1. Kanban Board with 2 feature Tickets

Make sure to make your Kanban **public** so we can evaluate your tickets.

Submission Conventions

PDF should be formatted "Your Name - PM Challenge F25" (e.g. "Lucas Ha - PM Challenge F25")

You must submit your PDF and Github Repository link via this Google Form.

SUBMISSION DEADLINE: October 10th @ 11:59 PM