Intelligent Document Processing (IDP) - 12 Hour Hackathon Roadmap

Team Size: 4 Members

Hackathon Start Time: 11 PM

----

1. Overview

This document provides a complete 12-hour execution roadmap, task splitting, technology stack, and presentation preparation strategy for building an Intelligent Document Processing system during an overnight hackathon. The goal is to build a lean and functional MVP that supports:

- Document upload (PDF, DOCX, Images)
- Text extraction
- Summarization
- Document classification
- Automatic routing
- Semantic search using FAISS
- Simple, impressive UI (Streamlit)

----

2. Technology Stack

Backend:

- Python
- FastAPI
- LangChain
- FAISS Vector Store
- PyPDF2 / pdfplumber (PDF extraction)
- docx2txt (DOCX extraction)

- Optional: Tesseract or EasyOCR for image OCR

AI/ML:

- Gemini 2.0 Flash / Pro (Summarization & Classification)

- Google Generative AI Embeddings

- SentenceTransformers

- Optional ML classifier (RandomForest / Logistic Regression)

Frontend:

- Streamlit (recommended for hackathon speed)

- HTML/CSS optional for enhancements

Storage:

- Local storage for uploaded docs

- FAISS index locally stored as faiss_index.bin

- Optional SQLite DB for metadata

----

3. Team Division (4 Members)

Member 1 – Backend & Extraction

- Build FastAPI endpoints: /upload, /extract

- Implement PDF, DOCX extraction logic

- Optional: Integrate OCR using easyocr

- Implement text cleaning pipeline

Member 2 – AI Pipeline (Summarization, Classification, Embeddings)

- Build summarization function using Gemini

- Build classification prompt for document types

- Implement routing logic (rule-based + LLM)

- Create chunking + embedding + FAISS indexing

- Build /summarize, /classify, /search endpoints

Member 3 – UI/Frontend Developer

- Build Streamlit UI

- Document upload interface

- Display extracted text, summary, category, routing

- Build semantic search UI

- Add document history visual panel

Member 4 – Integrations, QA, Pitch Deck

- Integrate Streamlit with FastAPI

- Build demo script

- Create slides (Problem → Solution → Demo → Impact)

- Prepare sample documents

- Conduct testing and optimizations

----

4. 12-Hour Smart Execution Timeline

----

11 PM – 12 AM (1 hour) — Planning & Setup

- Finalize features of the MVP

- Create GitHub repo

- Set up FastAPI skeleton

- Set up Streamlit skeleton

- Confirm folder structure

- Decide classification categories

----

12 AM – 3 AM (3 hours) — Backend + AI Foundation

Member 1:

- Implement /upload

- Implement text extraction for PDF & DOCX

- Test extraction on 3 sample files

Member 2:

- Implement chunking + embedding pipeline

- Build FAISS index creation & update

- Write summarization prompt

- Write classification prompt

----

3 AM – 6 AM (3 hours) — Frontend + API Integration

Member 3:

- Build Streamlit UI components:

- Upload section

- Display extracted text

- Display summary

- Display routing & category

- Search page

Member 4:

- Integrate UI ↔ backend endpoints

- Build search workflow

- Fix request/response formatting

- Verify full flow: Upload → Extract → Summarize → Classify → Search

----

6 AM – 8 AM (2 hours) — Polish & Additional Features

All Members:

- Add similarity search between documents
- Improve UI layout
- Add error handling & loading states
- Add multi-document search sorting
- Improve prompts for classification

----

8 AM – 10 AM (2 hours) — Testing, Sample Docs, Pitch Deck

Member 4:

- Build pitch slides:

1. Problem Statement
2. Why it matters
3. Proposed Solution
4. Architecture Diagram
5. Demo Flow
6. Impact & Future Scope

Members 1–3:

- Test all features with:
- HR letter
- Invoice
- Engineering doc
- Purchase order
- Safety document
- Final fixes

----

10 AM – 11 AM (1 hour) — Final Dry Run

- Run the whole demo end-to-end
- Ensure UI looks clean
- Fix performance & formatting issues
- Finalize GitHub README
- Prepare for judging

----

5. Final Deliverables for Judges

----

1. Working demo:

- Document upload → extraction → summary → classification → routing → semantic search

2. Clean UI/UX using Streamlit

3. Clear architecture:

- FastAPI Backend
- FAISS Vector Search
- Gemini AI for NLP

4. Pitch Deck with impact and future scope

5. GitHub Repository with:

- Code
- Instructions
- Sample docs
- Demo screenshots

----

6. Conclusion

This roadmap ensures that a team of 4 can confidently build an impressive, fully functional Intelligent Document Processing MVP within a 12-hour hackathon window. Proper division of tasks, simplified architecture, and leveraging LLMs allow rapid development while maintaining high impact.