ILMATRIX — Documentation

ILMATRIX is an AI-powered study companion for university students. It helps explain materials, answer or generate quizzes, draft forum replies, prepare for exams responsibly, chat about context, practice via a Quiz Trainer, and simulate peer discussions — all centered on your uploaded materials.

Brand: ILMATRIX
Primary audience: Students and educators
Tech core: Hono.js API, Groq (Meta Llama models), PDF/DOCX/PPTX/OCR extraction, static Tailwind UI, Netlify-ready

1) Product Overview

- What it is:
  - A study assistant that focuses on using your uploaded materials first. It explains, answers, and trains directly from your content where possible, quoting relevant snippets for transparency and integrity.
- What it is not:
  - It is not a replacement for studying. It includes guardrails and encourages academic honesty.
- Key differentiators:
  - Material-first design, strong extraction support (PDF, DOCX, PPTX, images via OCR), fast responses powered by Groq, and multiple learning modalities (explain, quiz, trainer, peer simulation).

2) Feature Summary

- Upload
  - Supported: PDF, TXT, PNG, JPG/JPEG, DOCX, PPTX
  - Extracts text and stores a materialId for referencing
- Explain
  - Structured and concise explanations, with short citations from material when possible
- Quiz Helper
  - Solve existing quizzes or generate N questions; includes “Jawaban” section when applicable
- Forum Reply
  - Drafts respectful, academically coherent replies to forum prompts

- Exam Helper
  - Study-first guidance; encourages integrity and responsible use
- Chat (General)
  - Free-form chat with optional material context
- Quiz Trainer
  - Start: generate practice questions (numbered)
  - Score: evaluate answers, provide score, per-question feedback, weaknesses, and a study plan
- Peer Simulation
  - Role-play a classmate in friendly or challenging style for multiple rounds

UI color scheme:

- Chat tab: Primary #0f0e85
- Upload tab: Secondary #e44c99
- Other tabs: Tertiary #1b612f

3) User Flows (Narrative)

3.1 Upload and Explain

- User opens the App page
- Uploads a PDF/DOCX/PPTX/image/TXT
- System extracts text and returns a materialId
- User navigates to Explain tab, adds a prompt (optional), runs Explain
- System returns a structured explanation with citations (where possible)
  3.2 Quiz Helper (solve or generate)
- If solving: paste quiz text in the prompt; run Quiz
- If generating: set number of questions; leave prompt blank or add topic hints; run Quiz
- System returns solution or generated questions, with “Jawaban” at the end where relevant
  3.3 Forum Reply
- Provide forum prompt or topic; run Draft Reply
- System returns a respectful, academically aligned draft
  3.4 Exam Helper
- Provide study instructions or sample questions; run Exam Helper
- System returns guidance that prioritizes learning and ethical preparation
  3.5 Chat (General)
- Type messages to discuss content, clarify answers, or iterate on results
- If a material is uploaded, the assistant uses it as context

3.6 Quiz Trainer

- Start: generate numbered practice questions based on uploaded material
- User writes numbered answers in the provided textarea
- Score: system grades, provides feedback/weaknesses, and a study plan

3.7 Peer Simulation

- Choose style (friendly/challenging) and rounds (1–5)
- Start simulation; get a multi-turn role-play dialogue to practice reasoning and articulation

4) High-level Architecture

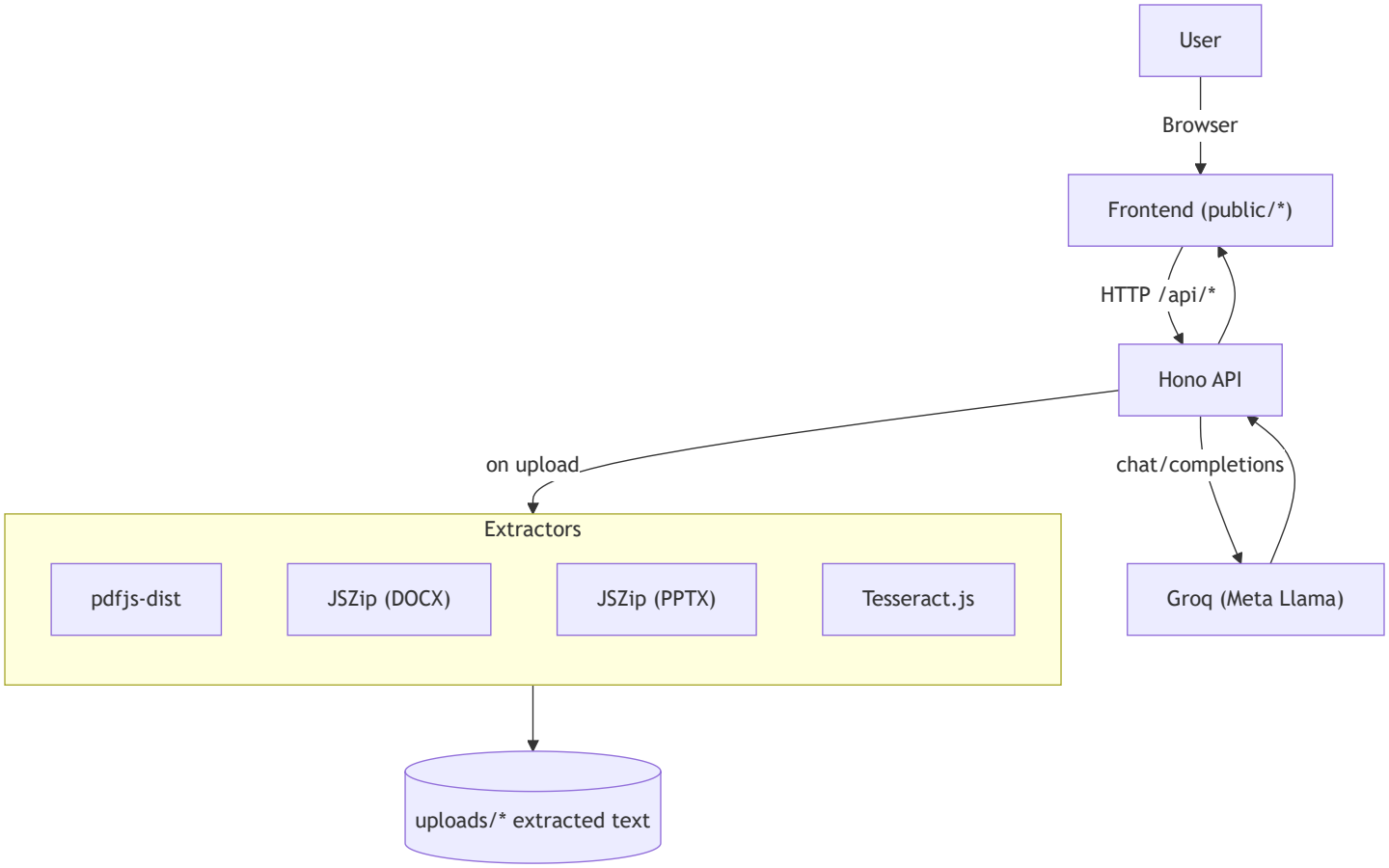
- Frontend
  - Static HTML (Tailwind CSS via CDN) in /public
  - Tabs per feature; each tab renders its own results (no result-only page)
- Backend
  - Hono.js app exposing /api routes
  - Groq SDK (Meta Llama models) for generation/analysis
  - Extraction utilities for PDF (pdfjs-dist), DOCX/PPTX (JSZip), OCR for images (tesseract.js)
  - Temp file storage for extracted text; returns a materialId
- Deployment
  - Netlify-compatible (functions for API, static hosting for /public)

Key files (for reference):

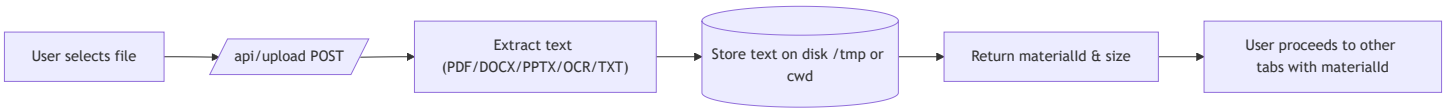
- public/index.html — app UI (tabs/features)
- public/home.html — marketing-style Home
- public/about.html — About Us page
- src/server.ts — Hono server/bootstrap
- src/routes.ts — API routes (upload, explain, quiz, forum, exam, chat, trainer, peer)
- src/groqClient.ts — Groq client and prompting logic
- src/extract/*.ts — extraction utilities (pdf, docx, pptx, image)
- netlify.toml + netlify/functions/api.ts — serverless adapter and routing

5) Flowcharts (Mermaid)

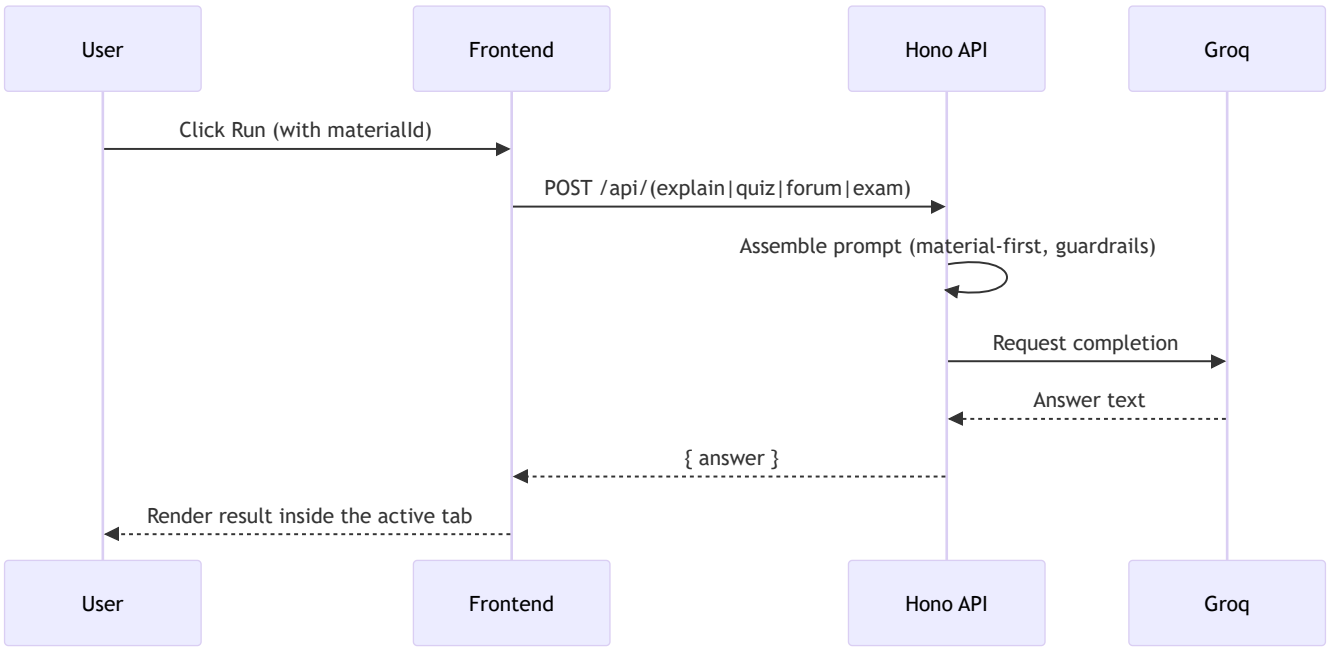
5.1 System Overview (Flowchart)



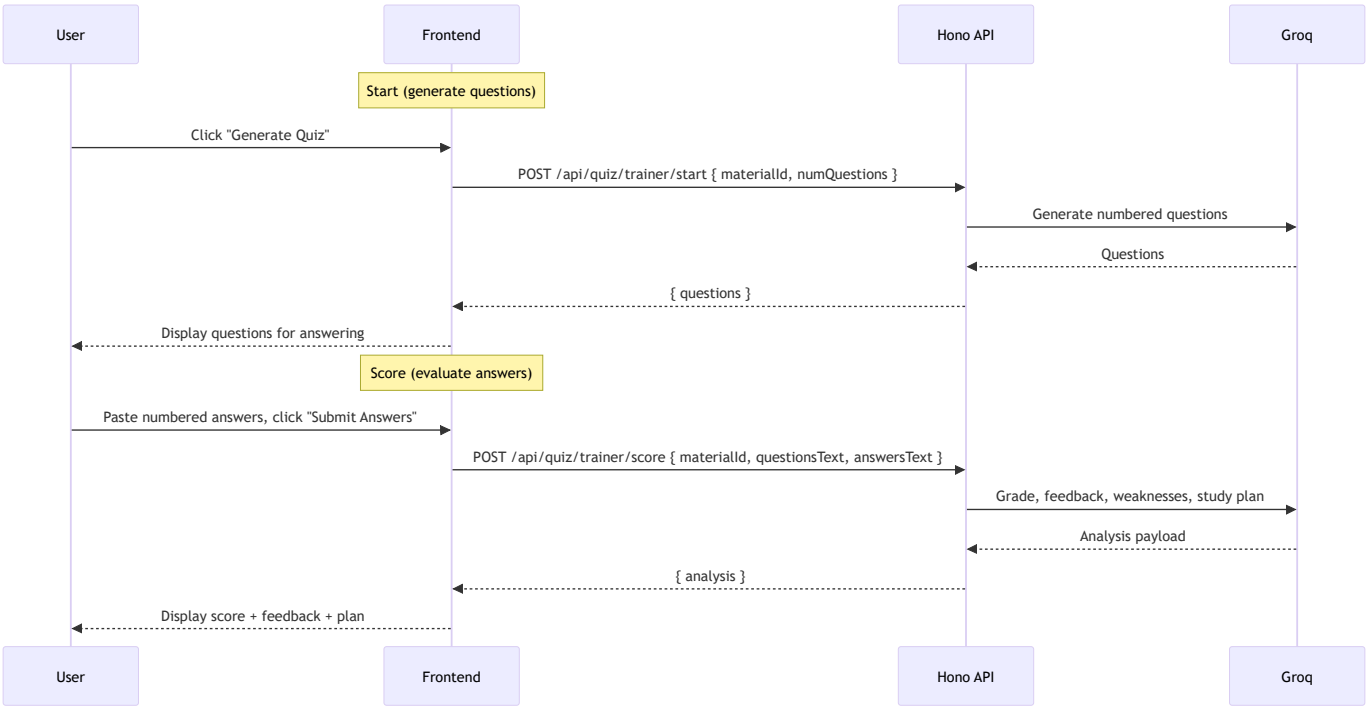
5.2 Upload Flow



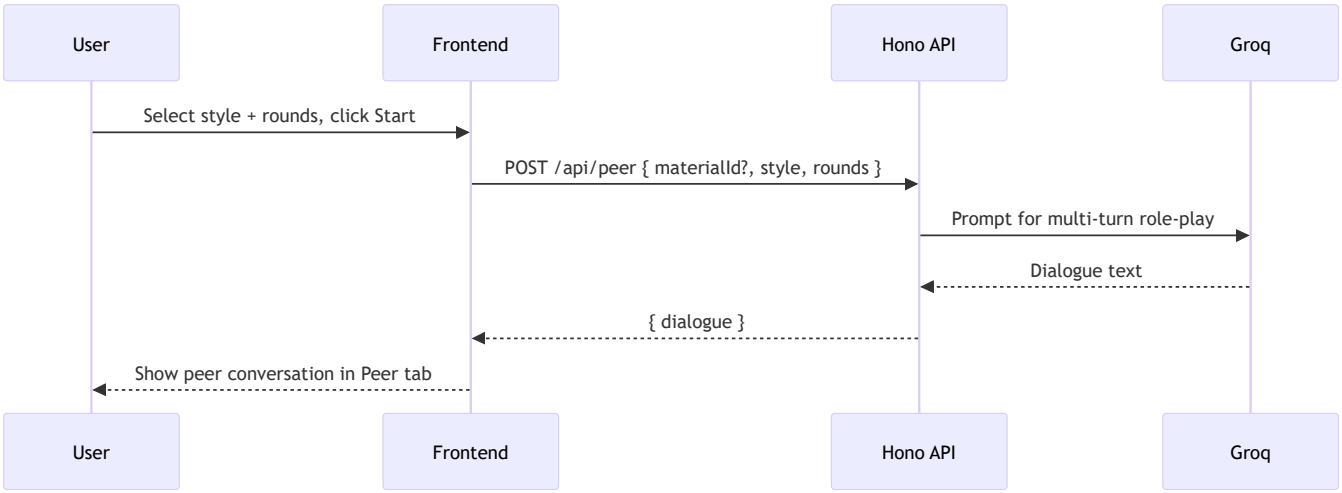
5.3 Explain/Quiz/Forum/Exam Flow



5.4 Quiz Trainer (Start + Score)



5.5 Peer Simulation



6) API Reference (Brief)

Base: /api

- POST /api/health
  - Response: { ok: true, model: "...", time: ... }
- POST /api/upload
  - Form-data: file (PDF/TXT/PNG/JPG/DOCX/PPTX)
  - Response: { materialId: string, size: number }
- POST /api/explain
  - JSON: { materialId: string, prompt?: string }

    - Response: { answer: string }
- POST /api/quiz
    - JSON: { materialId: string, prompt?: string, numQuestions?: number }
    - Response: { answer: string } // may include “Jawaban” section
- POST /api/forum
    - JSON: { materialId: string, prompt: string }
    - Response: { answer: string }
- POST /api/exam
    - JSON: { materialId: string, prompt: string }
    - Response: { answer: string } // includes integrity/ethics guidance
- POST /api/chat
    - JSON: { materialId?: string, messages: Array<{ role: "user"|"assistant"|"system", content: string }> }
    - Response: { answer: string }
- POST /api/quiz/trainer/start
    - JSON: { materialId: string, numQuestions: number }
    - Response: { questions: string } // numbered list only
- POST /api/quiz/trainer/score
    - JSON: { materialId: string, questionsText: string, answersText: string }
    - Response: { analysis: string } // score + feedback + weaknesses + study plan

7) Frontend Structure

- Pages
    - /public/home.html — marketing front page (hero, features, how-it-works, CTA)
    - /public/index.html — app with tabbed UI (Chat first, Upload second, others follow)
    - /public/about.html — about us (mission, story, values, tech, contact)
- Tabs in App (index.html)
    - Chat (primary)
    - Upload (secondary)
    - Explain, Quiz, Forum, Exam, Trainer, Peer (tertiary)
- UX notes
    - Each feature renders its result within its own section
    - No “Result” tab; reduce context switching
    - Keyboard focus moves to the first element of the active tab

8) Extraction Support

- PDF: pdfjs-dist configured with fonts and CMaps; supports local and serverless paths
- DOCX: JSZip reads word/document.xml and related parts; converts XML to text
- PPTX: JSZip reads ppt/slides/slideN.xml; extracts <a:t> runs as text
- Images (PNG/JPG): tesseract.js OCR; language fallback from “eng+ind” to “eng”
- TXT: read as-is

Performance tips:

- Prefer textual PDFs over image-only for faster and cleaner extraction
- Large DOCX/PPTX may need extra time; avoid extremely large uploads on serverless

9) Configuration & Deployment

- Environment
  - GROQ_API_KEY must be set for the backend
  - ESM with TypeScript in NodeNext mode; ensure .js extensions on local imports post-emit
- Local development
  - Install dependencies; run dev/start scripts
  - Open /public/index.html (via the local server if configured) or deploy to Netlify
- Netlify
  - netlify.toml routes /api/* to functions
  - Static site lives in /public
  - Functions adapter wraps Hono app

10) Security, Safety, Integrity

- Material-first prompts encourage quoting short snippets for transparency
- No chain-of-thought; focus on succinct, verifiable reasoning
- Exam Helper and other tasks include academic integrity guidance
- Consider adding:
  - Basic auth or key gating if deploying publicly
  - Rate limiting and file size limits
  - Redaction for PII in uploaded materials if needed

11) Roadmap Ideas

- Streaming responses (SSE) for better perceived latency
- Persist chat and trainer sessions (localStorage or backend session)
- Markdown rendering in results and chat bubbles
- Per-user profiles and saved materials
- Multi-language OCR packs as optional downloads
- Export to DOCX/PDF of generated study notes

12) FAQ (Concise)

- Q: Do I need an account?
  - A: For local usage, no. For public deployments, you may add auth.
- Q: How private are my uploads?
  - A: Files are processed for study purposes and stored temporarily. Review and customize retention for your deployment.
- Q: Can it handle photos/screenshots of slides?
  - A: Yes, via OCR (tesseract.js), but clarity and contrast improve results.
- Q: Which models are used?
  - A: Groq with Meta Llama models (configurable).

13) Pseudo Data Models

- UploadResponse
  - { materialId: string, size: number }
- ChatMessage
  - { role: "user" | "assistant" | "system", content: string }
- TrainerStartRequest
  - { materialId: string, numQuestions: number }
- TrainerScoreRequest
  - { materialId: string, questionsText: string, answersText: string }

14) Example Requests

14.1 Upload (curl)

```
curl -X POST http://localhost:3000/api/upload \
  -F "file=@/path/to/file.pdf"
```

Response:

```
{ "materialId": "a1b2c3d4", "size": 42037 }
```

14.2 Explain

```
curl -X POST http://localhost:3000/api/explain \
  -H "Content-Type: application/json" \
  -d '{ "materialId": "a1b2c3d4", "prompt": "Jelaskan bab 2" }'
```

14.3 Quiz (generate 5)

```
curl -X POST http://localhost:3000/api/quiz \
  -H "Content-Type: application/json" \
  -d '{ "materialId": "a1b2c3d4", "numQuestions": 5 }'
```

14.4 Trainer (start + score)

```
curl -X POST http://localhost:3000/api/quiz/trainer/start \
  -H "Content-Type: application/json" \
  -d '{ "materialId": "a1b2c3d4", "numQuestions": 5 }'


curl -X POST http://localhost:3000/api/quiz/trainer/score \
  -H "Content-Type: application/json" \
  -d '{ "materialId": "a1b2c3d4", "questionsText": "1) ...", "answersText": "1) ..." }'
```

15) Home & About Pages (Content Pointers)

- Home (/public/home.html)
  - Hero: “Learn smarter with ILMATRIX”
  - Features summary, how it works, tech highlights, CTA to App
- About (/public/about.html)
  - Mission, story, values, technology, collaboration/contact CTA

Open the App: /public/index.html

Explore marketing pages: /public/home.html and /public/about.html

End of ILMATRIX Documentation