Realization & Technologies Used

1) Backend (FastAPI with Python)

The backend is built using **FastAPI** with SQLAlchemy ORM and PostgreSQL.

Endpoints Implemented

- **Auth endpoints (/auth)**
    - /auth/register → register new users with hashed passwords.
    - /auth/login → authenticate user and return JWT token.
    - /auth/me → get current user info from token.
- **Analyze endpoints (/analyze)**
    - /analyze/ → accept a message, call AI + threat intelligence checks, return structured result (judgment, explanation, tips).
    - /analyze/history → fetch all past analyses by user (with pagination support).
- **Stats endpoints (/stats)**
    - /stats/judgments → breakdown of judgments (Safe, Phishing, Other).
    - /stats/monthly → monthly counts of analyses (used in bar chart).
    - /stats/daily → daily counts of analyses (used in line chart).

Backend Features

- **Database Models**
    - User: manages email, password hash, role, created_at.
    - Analysis: stores analyzed message, sender, AI result JSON, created_at, linked to user.
- **Authentication**
    - JWT token with role inside payload.
    - Passwords securely hashed with bcrypt.
- **Generative AI integration**
    - Calls AI model (structured JSON output).
    - Integrated with heuristic checks (URLs, phishing keywords).
- **Seeding scripts**

- seed_data.py for dashboard testing.
- reassign_analyses.py to fix data ownership for specific users.

The backend satisfies the requirement of **2+ working endpoints** (actually >5).

2) Frontend (Next.js + React + shadcn/ui)

The frontend is built with **Next.js 13 App Router** and **shadcn/ui** components.

Pages Implemented

- **Auth**
  - /login → login form, saves token in localStorage + cookie.
  - /register → registration form.
- **Dashboard**
  - /dashboard → charts (line, bar, pie) with real backend data.
- **Analyze**
  - /analyze → chat-like interface to paste suspicious emails/messages, send to backend, and display AI analysis (judgment, explanation, tips).
- **History**
  - /history → list of past analyses with timestamps, newest first.
- **Settings**
  - /settings → change theme (light/dark), logout.

UI Features

- **Sidebar**: links to dashboard, analyze, history, settings. Shows avatar with user initial.
- **Dark/Light theme toggle** integrated with shadcn/ui.
- **Charts**:
  - Line chart → daily usage.
  - Bar chart → monthly usage.
  - Pie chart → judgment breakdown (Safe, Phishing, Other).

The frontend satisfies **3+ working pages** requirement (actually >4).

3) Technologies Used

- **Backend**
    - FastAPI (Python)
    - SQLAlchemy ORM
    - PostgreSQL
    - JWT (PyJWT)
    - bcrypt (password hashing)
    - Seed scripts for testing
- **Frontend**
    - Next.js 13 App Router
    - React
    - shadcn/ui components
    - Recharts (charts)
    - LocalStorage + cookie-based auth
- **Tools**
    - GitHub for version control
    - Postman for API testing
    - Swagger UI for endpoint testing
    - Docker Compose (optional for DB)

Conclusion for C

- Backend → more than 2 functional endpoints (Auth, Analyze, Stats).
- Frontend → more than 3 working pages (Login, Dashboard, Analyze, History, Settings).
- Tech stack → modern full-stack (FastAPI + Next.js + shadcn/ui).