00-intro

IELTS AI Platform
An AI-powered learning platform for IELTS preparation.
Features:
- Auto-scoring for Writing & Speaking (band descriptors, detailed feedback).
- Reading & Listening test generation and auto-grading.
- Personalized dashboard & learning path.
Tech stack:
- **Frontend**: Next.js (App Router), Tailwind, shadcn/ui.
- **Backend**: FastAPI (Python), PostgreSQL (pgvector), Redis, RQ/Celery.
- **AI Services**: LLM (OpenAI GPT), Whisper (ASR), TTS.
- **Infra**: Docker Compose (dev), Kubernetes/Cloud (prod).

01-architecture

System Architecture
Frontend (Next.js), API Gateway (FastAPI), Workers (RQ/Celery), PostgreSQL + pgvector, Redis (queue + KV), S3/MinIO (assets), AI Providers (LLM/Whisper/TTS).
Observability: Sentry + OpenTelemetry; Auth: NextAuth/Clerk; Billing: Stripe (later).

02-project-structure

Project Structure
ielts-platform/
■■ apps/web (Next.js)
■■ apps/api (FastAPI)
■■ workers/{scorer,generator}
■■ packages/{shared-schemas,scoring-metrics,rag-kit}
■■ infra/{docker,db,k8s}
■■ .github/workflows/{ci.yml,cd.yml}
■■ docs/

03-setup-dev

Development Setup
Requirements:
- Docker & Docker Compose
- Node.js 20+ with pnpm
- Python 3.11+
Quick Start:
git clone https://github.com/khanhnguyendev/ielts-platform.git
cd ielts-platform
make up
API: http://localhost:8000/healthz
Web: http://localhost:3000
Database Migrations:
make migrate

04-database

Database Schema
Tables:
- users
- items (test items)
- sessions
- submissions
- scores
- assets
- ai_runs
- rag_knowledge
Extensions:
- pgvector for embeddings.
Migrations:
- Alembic used for versioning.

05-api-design

API Design
Health
GET /healthz
Writing
POST /api/writing/score
Input: { question, essay, task_type }
Output: JSON with sessionId, band, subScores, feedback
Speaking
POST /api/speaking/score (audio upload → job → transcript + band)
Reading & Listening: similar pattern

06-ai-pipelines

AI Pipelines
Writing:
- Input essay → LLM scoring prompt → JSON schema.
- Self-consistency: 3 runs → median band.
- Store in scores.
Speaking:
- Audio → Whisper ASR → metrics → LLM summarization.
Reading/Listening:
- Two-phase generation: Draft → Verifier.
- Store passage/script, questions, rationale.

07-frontend

Frontend (apps/web)
- Next.js (App Router).
- Tailwind + shadcn/ui.
- Features:
- Auth (NextAuth).
- Test pages: Writing, Speaking, Reading, Listening.
- Dashboard: scores history, weak skills.

08-ci-cd

CI/CD
- GitHub Actions:
- ci.yml: lint, test, type-check, build.
- cd.yml: build Docker, push, deploy.
- Production: Kubernetes (Helm).

09-contributing

Contributing Guide
1. Fork & clone.
2. Branch naming: feature/..., fix/...
3. Run make dev.
4. Submit PR with description.

10-roadmap

Roadmap
Phase 0 — Setup & Environment ■
Phase 1 — Auth & Users
Phase 2 — Writing Score MVP
Phase 3 — Speaking
Phase 4 — Reading
Phase 5 — Listening
Phase 6 — RAG & Knowledge Base
Phase 7 — Dashboard & Learning Path
Phase 8 — Moderation & Safety
Phase 9 — Payment & Plans
Phase 10 — Production & Observability