Apprendre – AI-Powered Real-Time Quiz Platform

1. Problem Statement:

In modern digital learning environments, engagement and assessment are crucial. Traditional quiz platforms lack interactivity, real-time multiplayer support, and AI-driven automation. Educators and learners need a system that allows instant quiz generation, live competition, performance analytics, and smooth user experience.

Key gaps addressed:

- Manual quiz creation is slow and repetitive.
- Lack of real-time multiplayer and live leaderboards.
- Limited insights into performance and adaptive difficulty.
- PDF/text content cannot be directly transformed into quizzes.

2. Approach & AI Components:

- Frontend: React + Tailwind CSS

- Responsive UI for quizzes, dashboards, multiplayer sessions.

- Integrated with Firebase services for auth, data, and storage.

- Backend: Firebase (Functions, Firestore, Storage, Realtime DB)

- Cloud Functions handle all CRUD operations for quizzes, sessions, results.

- Firestore stores quizzes, questions, results; Realtime DB handles live session updates.

- Storage manages PDF uploads for AI quiz generation.

- AI Integration: Google AI Studio (Gemini API)

- Generates multiple-choice questions from:
    1. Topic keywords
    2. Raw text/articles
    3. PDF uploads (extracted via pdf-parse)

- Returns structured quiz objects: questions, answers, explanations.

Workflow:

- ➢ User inputs topic/text/PDF → triggers /quizzes/:id/ai-generate.
    - ○
- ➢ Function calls Gemini API → returns quiz payload.
    - ○
- ➢ Quiz stored in Firestore → accessible via frontend for solo or multiplayer play.

3. Technical Architecture:

- Frontend ↔ Backend Integration:

- All API calls via Firebase Functions (HTTPS) with ID token verification.

- Frontend API wrappers handle auth, quizzes, results, sessions, PDF upload.

- Realtime updates for multiplayer via Firestore listeners.

- Data Model:

    - ➔ *Quizzes Collection:*

```
{
    "id": "quizId",
    "title": "Quiz Title",
    "questions": [ ... ],
    "ownerUid": "userId",
    "is_public": true
}
```

    - ➔ Sessions Collection:

```
{
    "id": "sessionId",
    "quiz_id": "quizId",
    "game_code": "ABC123",
    "status": "waiting|question|results|finished",
    "players": [ { "uid": "", "score": 0 } ]
}
```

    - ➔ Realtime Flow:

Player submits answer → Firestore triggers update → leaderboard updates for all participants.
Storage: PDFs uploaded → parsed → sent to Gemini API → converted to quiz

4. Challenges & Mitigations:

Challenge	Mitigation
Real-time multiplayer without Socket.io	Used Firestore Realtime listeners for session updates.
AI quiz generation from PDFs	Integrated pdf-parse to extract text; structured input for Gemini API.
Auth + protected routes	Firebase Auth with ID token verification; req.user attached in functions.
Frontend-backend sync	API client in React automatically injects ID token; fetch wrapper ensures consistency.
Emulator setup	Used Firebase Emulator Suite for local testing (Functions, Firestore, Storage, Realtime DB, Hosting).

5. Roadmap to Final Build

Completed:

- React frontend wired to Firebase backend.
- AI quiz generation from topics, text, and PDFs.
- Multiplayer sessions, leaderboards, solo play.
- PDF upload + extraction.
- Dashboard, analytics, and user profile.
- Next Steps / Optional Enhancements:
  Adaptive quiz difficulty using AI.
  Speech-based question integration.
- Analytics visualization using charts (Recharts/Framer Motion).
- Production deployment on Firebase Host