Team Techtonics-

Quizito

1. Problem Statement

Educational institutions, training companies, and online learning platforms require a fast, scalable, and intelligent way to create assessments. Traditional quiz-building tools require manual content creation, lack real-time analytics, and do not adapt to user capability.

Key challenges in existing systems:

- Manual quiz creation is time-consuming and inconsistent
- Limited personalization or adaptive difficulty
- Lack of semantic answer evaluation
- Absence of rich analytics and AI-driven insights
- Fragmented workflows across frontend, backend, and AI components

Quizito addresses these gaps with an AI-powered solution:

- Automatic quiz generation based on any topic
- Intelligent scoring engine with semantic analysis
- Insights dashboard visualizing strengths, weaknesses, and time metrics
- Fully connected frontend-backend-AI pipeline demonstrating feasibility for scalable use

By the end of Round 2, Quizito achieves a **functional prototype** demonstrating the **core AI logic, backend APIs, and end-to-end workflow**, satisfying the 60–80% development target.

2. Approach & AI Components (Expanded)

Quizito integrates three core AI modules forming the backbone of the system:

2.1 Quiz Generation Module (LLM-driven)

This module accepts:

- Topic
- Number of questions
- Optional difficulty level

It outputs:

- Structured MCQs or descriptive questions
- Correct answers

- Distractor options (for MCQs)

Prompt engineering used:

- Topic decomposition
- Multi-format templating
- Difficulty scaling
- Layered prompting ensuring consistent structure

2.2 Semantic Scoring & Evaluation Engine

The scoring model compares:

- User answer
- Expected AI-generated answer
- Context of the question

It uses:

- Sentence embeddings
- Fuzzy semantic similarity
- Threshold-based scoring strategy

Output includes:

- Score
- Explanation
- Confidence score
- Highlight of key missing elements

This creates a more human-like evaluation system beyond keyword matching.

2.3 Analytics Engine

This module transforms raw interaction logs into insights:

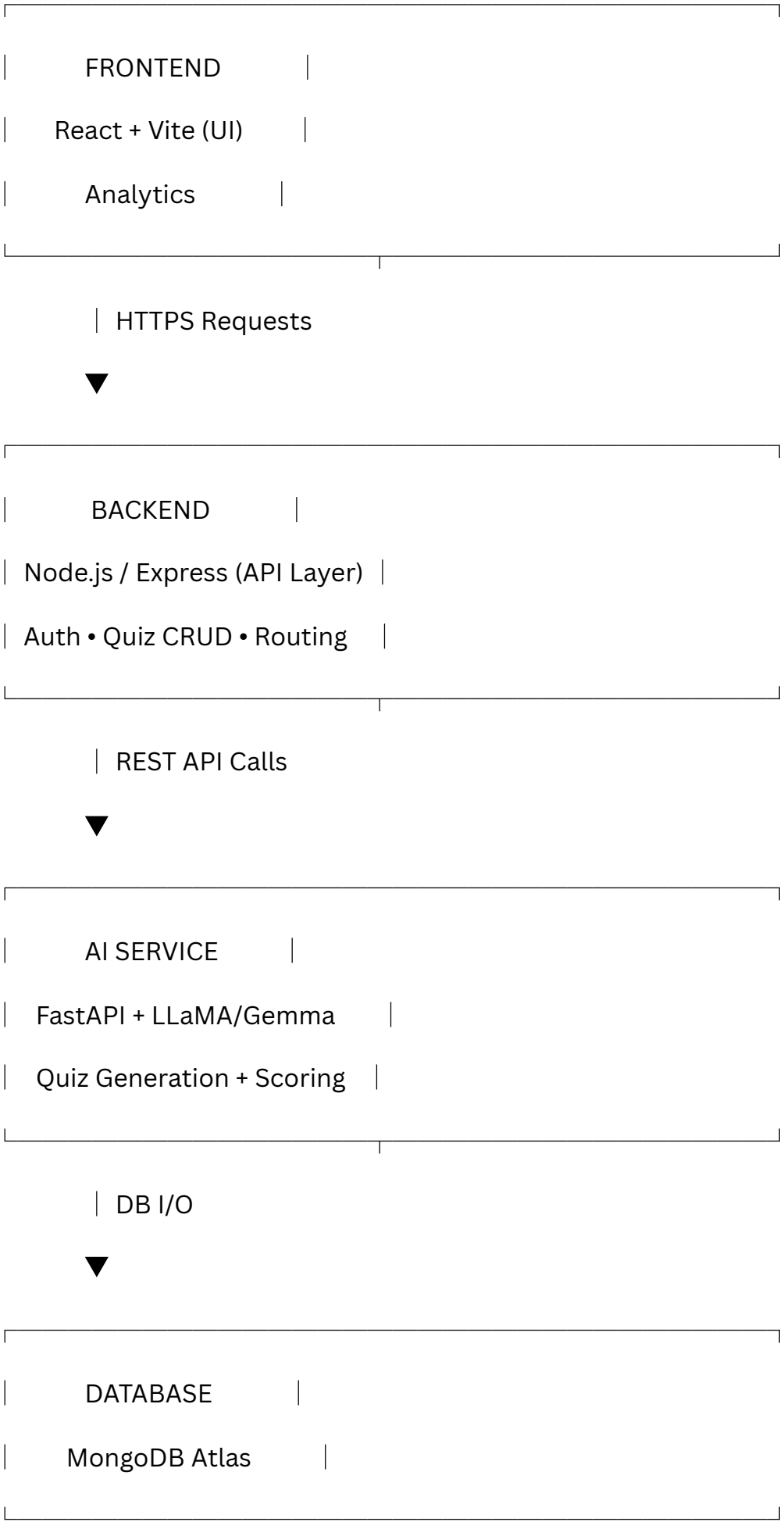
- Accuracy per question
- Time-per-question analysis
- Difficulty vs. performance graph
- Topic mastery estimation
- Behavioral indicators (strengths, weaknesses)

These insights drive **educational personalization**, meeting the hackathon's "impact potential" criterion.

3. Technical Architecture (Deep Explanation)

Quizito follows a **three-service distributed architecture** ensuring modularity and scalability.

3.1 High-Level Architecture



3.2 Component Responsibilities

Frontend

- Displays quizzes
- Sends user responses
- Renders analytics
- Interacts with backend APIs
- Ensures responsive UI and minimal load time

Backend

- Central hub for coordination
- Handles authentication & authorization
- Stores all quiz attempts
- Routes AI requests
- Ensures stable, consistent API behavior

AI Microservice

- Heavy lifting of intelligence
- Stateless inference server
- Future-ready for GPU-based scaling

3.3 Data Flow (End-to-End)

[User Request] → Frontend → Backend API → AI Engine

AI Output → Backend → MongoDB Save → Frontend Visualization

3.4 API Flow Example

1. **Frontend** requests quiz generation
2. **Backend** forwards structured request to AI service
3. **AI** generates questions → returns JSON
4. **Backend** stores quiz metadata
5. **Frontend** displays quiz to the user

4. Development Workflow & Milestone Mapping

The project strictly followed the hackathon's Week-wise milestone structure:

Week 1 – Architecture & Setup (Completed)

- Selected **React + Node.js + FastAPI** as the tech stack

- Finalized system architecture diagrams
- Set up complete GitHub repository with /src, /ai, /backend, /frontend, /docs
- Created environment files: package.json, requirements.txt
- Initialized backend scaffolding with Express + MongoDB connection
- Designed initial UI wireframes

Deliverable:

✓ Architecture document
✓ Repository with clear structure
✓ Working database connection
✓ Landing page scaffold

Week 2 – AI Model Integration & Backend API Development (Completed)

- Integrated LLaMA/Gemma-based model for question generation
- Implemented semantic scoring module
- Built AI microservice using FastAPI
- Implemented core API endpoints:
  - /quiz/generate
  - /quiz/submit
  - /analytics/:id
- Enabled backend ⇔ AI service HTTP communication
- Started internal testing with mock data

Deliverable:

✓ Fully functional AI inference pipeline
✓ Documented APIs (README)
✓ Working backend + AI

Week 3 – Frontend Integration & End-to-End Workflow (Completed)

- Developed full quiz participation flow
- Connected frontend to backend APIs
- Displayed quiz results + analytics via UI components
- Implemented auth, error handling, and responsiveness
- Validated entire prototype from input → inference → output

Deliverable:

✓ End-to-end working prototype (meets 60–80% completeness)
✓ Deployed on Netlify + Render
✓ Real-time, functional demo ready

5. Challenges and Mitigations (Expanded)

Challenge	Impact	Mitigation
Inconsistent AI-generated formatting	Breaks UI rendering	Introduced templated prompting + regex cleanup
Long AI inference times	Slower quiz creation	Switched to optimized lightweight models
Cross-service communi	Blocked frontend ›	Set stable CORS policy +

cation errors	backend › AI flow	endpoint normalization
Environment variable conflicts	Deployment instability	Separated env files by service & cleaned secrets from repo
MongoDB connection timeouts on Render	Failed requests	Implemented retry logic & correct connection string
CORS	UI could	Implemen

failures between Netlify & Render	not access backend	ted dynamic origin detection

6. Expected Round 3 Upgrades (Roadmap)

- Add adaptive learning & difficulty scaling
- Add real-time collaboration and leaderboard
- Improve analytics visualizations
- Introduce role-based dashboards (Educator mode)
- Deploy a GPU-backed AI server for higher model accuracy
- Full mobile UI revamp

7. Conclusion

Quizito successfully meets the **Round 2 objective** of achieving a **functional prototype** with:

- AI-powered quiz generation
- End-to-end workflow implementation
- Backend–Frontend–AI service integration
- Documentation & architecture clarity
- Deployed live demo

This positions Quizito strongly for **Round 3 at IIT Bombay**, where the focus shifts to refinement, UX polishing, and model performance optimization.