Technical Design Document for AI Powered Learning Management System

Table of Contents

1.Introduction

AI Powered Learning Management System is a web application designed to provide an enhanced learning experience by leveraging advanced AI capabilities. The system enables real-time doubt resolution, efficient content delivery, and intuitive interaction through state-of-the-art technologies such as Next.js, Express.js, MongoDB, and the Gemini API.

This document details the technical design, architecture, and components, serving as a reference for developers, contributors, and stakeholders.

---

2. System Overview

The system integrates:

- **Frontend**: Developed with Next.js for server-side rendering and static site generation.
- **Backend**: Built with Express.js to handle server-side logic and API endpoints.
- **Database**: MongoDB for scalable data storage.
- **Gemini API**: Provides AI-powered assistance.
- **Flowbite**: Ensures responsive UI design.

---

3. Architecture

High-Level Architecture

The system follows a three-tier architecture:

1. **Frontend**: Handles user interaction and video playback.
2. **Backend**: Manages server-side processing, API requests, and AI integrations.
3. **Database**: Stores user data, video metadata, and chat logs.

Component Interactions

- **Ask Doubt Feature**:
    1. Captures a video screenshot upon user action.
    2. Sends the screenshot to the Gemini API.
    3. Displays context-aware responses in the chat interface.

---

4. Backend Design

Technologies Used

- **Node.js**: JavaScript runtime for server-side programming.

- **Express.js**: Framework for building APIs.
- **MongoDB**: NoSQL database.
- **Mongoose**: ODM library for MongoDB.
- **Gemini API**: AI interaction.
- **Nodemon**: Development utility.
- **JWT**: Authentication.

Project Structure

server

├— controllers

├— models

├— routes

├— middlewares

├— utils

├— .env

├— index.js

API Design

1. **Authentication**
    - POST /api/auth/login: User login.
    - POST /api/auth/logout: User logout.
2. **Doubt Handling**
    - POST /api/doubts: Handles video-related doubts.
3. **Testing**
    - GET /test: Confirms server status.

Database Schema

User Model

- username: String, unique.
- email: String, unique.
- password: String, hashed.
- profilePic: String, optional.

Doubt Model

- userId: References User.

- videoId: String.
- timestamp: Number.
- contextData: String.

Middleware

- **JWT Validation**: Verifies tokens for protected routes.
- **Error Handling**: Manages errors and unauthorized access.

---

5. Frontend Design

Technologies Used

- **Next.js**: Framework for SSR.
- **Flowbite**: UI components.
- **TypeScript**: Ensures type safety.
- **Firebase**: Authentication and hosting.

Project Structure

client

├— public

├— src

| ├— components

| ├— pages

| ├— utils

├— .env

├— package.json

Routing

- **Public Routes**:
    - /: Landing page.
    - /login: Authentication.
- **Protected Routes**:
    - /dashboard: User dashboard.
    - /video/:id: Video player with doubt resolution.

State Management

- **Context API**: Handles global state.

- **Local State**: Manages component-specific state.

Key Components

- **Video Player**: Plays videos and integrates doubt resolution.
- **Chat Interface**: Displays Gemini API responses.
- **Navbar**: Navigation and authentication controls.

---

6. Security Considerations

- **Authentication**:
    - JWT-based token authentication.
- **Authorization**:
    - Role-based access control.
- **CORS**:
    - Configured for trusted origins.
- **Environment Variables**:
    - Secrets stored securely in .env files.

---

7. Deployment Plan

1. **Backend**:
    - Host on Render
2. **Frontend**:
    - Deploy static files on Vercel.
3. **Database**:
    - MongoDB Atlas with IP whitelisting.
4. **CI/CD**:
    - GitHub Actions for automated deployments.

---

8. Unit Testing

- Use **Jest** and **Supertest** for API testing.
- Coverage:
    - Authentication endpoints.
    - Ask Doubt functionality.

---

9. Future Plans

- **Feature Enhancements**:
    - Add a video annotation tool.
    - Integrate additional AI models.
- **Performance**:
    - Optimize video processing latency.

---

10. Conclusion

This technical design document outlines the architecture and functionalities of the AI Powered Learning Management System, providing a roadmap for development and future enhancements. By integrating cutting-edge technologies and AI capabilities, this platform aims to revolutionize the learning experience.
