PRESIDENCY COLLEGE
(AUTONOMOUS)
AFFILIATED TO BENGALURU CITY UNIVERSITY, APPROVED BY AICTE, DELHI & RECOGNISED BY THE GOVT. OF KARNATAKA
RE-ACCREDITED BY NAAC WITH 'A+' GRADE

DEPARTMENT OF COMPUTER APPLICATIONS

MCA - D SECTION (Batch: 2024-2025)
Session: May 2024 to August 20225

PROJECT WORK – SYNOPSIS

TITLE OF THE PROJECT	:	Fluently (Language Leaning Web-app)
STUDENT NAME & SECTION	:	OMPRASAD B L MCA E Section
REGISTRATION NO.	:	23P01261
GUIDE NAME	:	Ms. Neha Jaswani

Abstract:

Fluently is a modern, interactive language-learning web application built with the PERN stack (Postgres, Express, React, Node.js), designed to make mastering a new language engaging, accessible, and secure. At its core, the platform features bite-sized, gamified lessons—ranging from vocabulary drills to listening comprehension and pronunciation exercises—empowered by AI-based voice feedback. Users receive spoken prompts via an AI Text-to-Speech engine and respond through a built-in Speech-to-Text interface, enabling real-time pronunciation assessment and personalized feedback.

User authentication and data security are handled seamlessly through Clerk.js, offering pre-built, customizable UI components and robust session management including social login, multi-factor authentication, and role-based access control. The backend, built with Node.js and Express, interfaces with Postgres to manage learning modules, user progress, and performance statistics. The React frontend delivers a polished, responsive experience with intuitive navigation and dynamic content rendering.

The voice-enabled features leverage the Web Speech API alongside AI services such as Google Speech-to-Text, enabling immersive user interactions and improving pronunciation strategies. Gamification elements—such as XP points, Progress tracking, hearts, and leaderboards—promote sustained engagement and motivation. An administrative dashboard supports content creation, user management, and analytics, making the platform viable for both individual learners and educational institutions.

Overall, “Fluently” combines secure, scalable architecture, advanced speech AI, and compelling UX design to offer a unique, effective solution for language learners seeking interactive, voice-driven learning experiences.

Introduction:

Fluenlty is an immersive, AI-powered language-learning web application built on the robust **PERN** stack—**PostgreSQL**, **Express**, **React**, and **Next.js**. Designed to help learners master languages through interactive, bite-sized lessons, Fluenlty offers a secure, scalable, and cutting-edge educational platform. It integrates **Clerk.js** for seamless authentication, providing features like email, social sign-in, and multi-factor login, all optimized for Next.js routing and middleware.

The core of Fluenlty is its voice-driven learning experience: lessons prompt users through AI-generated speech via the **Web Speech API** and cloud TTS services, while user responses are captured through speech recognition. This enables dynamic pronunciation feedback—leveraging AI or cloud services like Azure or OpenAI—to offer real-time analysis of accent, fluency, and accuracy, creating an immersive and personalized learning loop.

On the frontend, the Next.js React application delivers a smooth, responsive user interface, incorporating gamified elements such as XP tracking, streaks, hearts, and leaderboards to boost motivation. The backend, powered by Express and PostgreSQL, securely manages user profiles, lesson content, progress metrics, and speech logs. Using Next.js API routes, protected by Clerk's middleware and auth helpers, ensures security across both client and server.

Aim of the Project: Fluenlty aims to bridge interactive web technologies and AI voice tools to create an engaging, effective language-learning platform. By combining secure authentication, voice-enabled lessons, and performance tracking with a scalable PERN architecture, Fluenlty aims to redefine online language education as intuitive, motivating, and deeply personalized.

Review of Literature

Existing Systems

Duolingo

- **Strengths**: Widely used; gamified progression with XP, streaks, and social elements; adaptive exercise generation aids vocabulary, reading, listening, and speaking. Studies confirm Duolingo's effectiveness users

often reach similar proficiency to traditional coursework after intensive use .

- **Limitations**: Over-reliance on gamification can distract users from learning goals and encourage competition-focused behavior. Content often lacks contextual depth and explanatory feedback, limiting deeper language understanding. Recent AI-driven features have drawn criticism for inaccuracies and reduced human oversight.

Proposed System: Fluenlty

Contextual Depth & Feedback

- Integrates grammar explanations directly into lessons, addressing the common gap identified in Duolingo.
- Utilizes Next.js server-side capabilities combined with Clerk.js to deliver personalized, secure learning experiences.

Balanced Gamification

- Retains engagement-enhancing features (XP, streaks), but introduces learning-focused incentives and limits on game-first behaviors to prevent misuse.

Continuous Improvement with Analytics

- Leverages PostgreSQL metrics and speech logs to track user progress and tailor content dynamically.
- AI-generated exercises align with each learner's evolving skill level—drawing from research on adaptive learning systems .

Technology used

HARDWARE	
PROCESSOR	Intel i5
RAM	8 GB
Hard disk	1TB
SOFTWARE	REQUIREMENTS
Operating system	Windows 10
Frontend Technologies	ReactJS, Tailwind CSS,ShadCn
Backend Technologies	Nextjs, ExpressJs, Eleven Labs, ClerkJs
Database	PostGreSQL
Browser	Google Chrome/ FireFox
IDE (Development Environment)	Vs Code
Technology Tools	GitHub,Netlify,Git

Estimated Project Timeline – Fluently

Phase	Modules / Milestones	Estimated Time
1. Requirement Gathering & Planning	Define features, user roles, UX expectations, wireframes, tech stack decisions, database schema design	7–14 days
2. UI/UX Design Phase	Design landing page, learning interface, dashboard, mobile views using Figma or other tools	4–6 days
3. Project Setup	Set up Next.js 14/15 with TypeScript, folder structure, Tailwind + Shadcn UI, routing, Clerk authentication	2–3 days

4. Core Development Phase

Feature / Module	Subtasks	Time Estimate
User Authentication	Clerk integration, session management, protected routes	2–3 days
Landing Page	Responsive UI, animations	2–3 days
Lesson System	Lesson UI (questions, answers), progress tracker, lesson data fetch/store	5–7 days
AI Voice Integration	ElevenLabs API setup, pronunciation playback in lessons	2–3 days
Gamification System	XP engine, streak logic, achievement tracker	3–4 days
Hearts System	Deduction/addition logic, heart popups, practice unlock	2–3 days
Shop Module	UI, item logic, XP redemption, heart purchase flow	2 days
Subscription with Stripe	Stripe integration, pricing plans, webhooks, upgrade/downgrade handling	2–3 days
Leaderboard & Quests	XP tracking, global leaderboard logic, milestone quest logic	2–3 days
Sound Effects	SFX integration for user actions and feedback	1 day
Exit/Error Handling	Unsaved progress warning, custom 404s, heart exhaustion warnings	1–2 days
Admin Dashboard	React Admin setup, lesson CRUD, user management, shop item updates	2–3 days
Practice Module	List past lessons, retry mode, regain heart conditions	1–2 days

---

5. Backend & Database Phase

Module	Tasks	Time Estimate
Database Modeling	Define schema for users, XP, lessons, shop, subscriptions	4–7 days
Drizzle ORM Integration	Model tables, relationships, migrations	2 days
API Routes / Server Actions	Server logic for fetching/storing lesson data, XP, purchases	3–4 days

---

6. Finalization & Deployment

Activity	Tasks	Time Estimate
Mobile Responsiveness Polishing	Ensure all views adapt to mobile/tablet	1–2 days
Testing & Debugging	Bug fixing, UX testing, error handling	3–4 days
Deployment	Vercel deployment, env config, Stripe/Clerk/ElevenLabs production setup	1-3 days
Buffer Time	Unexpected issues, polish, feedback rounds	7–10 days

Gantt Chart:

Requirement specification

SRS /SRA	System Design	Coding	Testing	Deployment and Report
2 weeks	1 week	4 weeks	2 weeks	1 week

Guide Signature

Student Signature