Project Overview

TigerTix is a comprehensive campus event ticketing system designed for Clemson University students. The application allows users to browse campus events, reserve tickets, and receive confirmations through an intuitive web interface. The system features JWT-based authentication, natural language processing for ticket booking via LLM integration, and voice interface capabilities for enhanced accessibility.

This project demonstrates real-world software engineering practices including microservices architecture, Agile development methodology, CI/CD pipelines, and accessibility-first design principles.

Tech Stack

Frontend

React (v18+) - UI framework
Web Speech API - Voice input/output for accessibility
Vercel - Deployment platform

Backend

Node.js (v18+) - Runtime environment
Express.js - Web application framework
SQLite - Lightweight relational database
bcryptjs - Password hashing
jsonwebtoken (JWT) - Authentication tokens
better-sqlite3 - Database driver
Axios - HTTP client for inter-service communication

AI/ML

Ollama (Local development) - LLM runtime
Llama 3 - Language model for natural language ticket booking
Alternative: OpenAI API for production deployment

DevOps & Deployment

GitHub Actions - CI/CD pipeline
Vercel - Frontend hosting
Render - Backend microservices hosting
Jest - Testing framework

Microservices

Client Service (Port 6001)

Manages event listings
Handles ticket purchases
Provides event availability data

Authentication Service (Port 5003)
User registration and login
JWT token generation and validation
Password hashing with bcrypt
Maintains user database

LLM Service (Port 5002)
Natural language processing for booking requests
Integrates with Ollama/Llama 3 or OpenAI
Parses user intent and extracts event information
Keyword-based fallback for reliability

Admin Service (Port 5001)
Event creation and management
System administration functions
Event database management

Data Flow
User authenticates through Auth Service
Frontend receives JWT token
Authenticated requests include JWT header
Client Service provides event data
LLM Service processes natural language booking requests
Ticket purchases update database in real-time
Frontend reflects updated ticket availability

Installation & Setup Instructions
Prerequisites
Node.js (v18 or higher)
npm or yarn
Git
Ollama
Local Development Setup
1. Clone the Repository

```
git clone https://github.com/RyanMurphy0/TigerTix.git
cd TigerTix
```

2. Install Frontend Dependencies

```
cd frontend
npm install
```

3. Install Backend Dependencies

```
# Client Service
cd backend/client-service
npm install

# Auth Service
cd ../user-authentication
npm install

# LLM Service
cd ../llm-service
npm install

# Admin Service
cd ../admin-service
npm install
```

4. Set Up Environment Variables

Frontend (.env in frontend/ directory):

```
REACT_APP_API_URL=http://localhost:6001
REACT_APP_AUTH_URL=http://localhost:5003
REACT_APP_LLM_URL=http://localhost:5002
```

Backend Services (.env in each service directory):

Auth Service (backend/user-authentication/.env):

```
PORT=5003
JWT_SECRET=your_secret_key_here_change_in_production
DATABASE_PATH=./database.db
NODE_ENV=development
```

Client Service (backend/client-service/.env):

```
PORT=6001
DATABASE_PATH=../shared-db/database.sqlite
NODE_ENV=development
```

LLM Service (backend/llm-service/.env):

```
PORT=5002
OLLAMA_URL=http://localhost:11434
CLIENT_SERVICE_URL=http://localhost:6001
NODE_ENV=development
# Optional: OPENAI_API_KEY=your_openai_key
```

Admin Service (backend/admin-service/.env):

```
PORT=5001
```

```
JWT_SECRET=your_secret_key_here_change_in_production
DATABASE_PATH=../shared-db/database.sqlite
NODE_ENV=development
```

6. Run the Application

Terminal 1 - Frontend:

```
cd frontend
npm start
```

Frontend will run at http://localhost:3000

Terminal 2 - Auth Service:

```
cd backend/user-authentication
npm start
```

Terminal 3 - Client Service:

```
cd backend/client-service
npm start
```

Terminal 4 - LLM Service:

```
cd backend/llm-service
npm start
```

Terminal 5 - Admin Service (Optional):

```
cd backend/admin-service
npm start
```

7. Access the Application

Open your browser and navigate to http://localhost:3000

Backend Tests

Run tests for each microservice:

```
# Client Service
cd backend/client-service
npm test

# Auth Service
cd backend/user-authentication
npm test

# LLM Service
cd backend/llm-service
npm test
```

```
# Admin Service
cd backend/admin-service
npm test
```

Team

Development Team

Ryan Murphy - Backend Development, Microservices Architecture, Database Design, Deployment

GitHub: @RyanMurphy0

Brady Barnes - Frontend Development, UI/UX Design, Component Integration

Academic Information

Instructor: Dr. Julian Brinkley

Course: CPSC 3720 - Software Engineering

Institution: Clemson University

Semester: Fall 2024

License

This project is licensed under the MIT License.

SOFTWARE.