Title: Ticketing System

Authors: Amirmahdi Monzavi & Mahdiar Omid Moghadami

University: Science and Research branch

Course: database

Instructor: Dr. Hamid Torkashvand

Abstract

This document presents a full-stack web application for a ticketing system. The application allows users to sign up, log in, and submit tickets, explaining issues they face. An admin can log in, view a list of all tickets, and see each ticket's details. The frontend is developed using React, styled with Tailwind CSS, and handles client-side routing with React Router. TanStack Query also manages HTTP requests and error handling. The backend, built with Node.js and Express.js, includes user authentication with bcrypt and JWT, and communicates with a MySQL database.

Introduction

The Ticketing System Web Application is designed to show the process of issue reporting and resolution within a system. Users can register and log in to submit tickets, which are then reviewed by an admin. This project demonstrates the integration of modern web technologies to create a responsive and functional application that meets user needs efficiently.

The primary objectives of this project are:

- To provide a user-friendly interface for issue submission.
- To enable efficient ticket management for administrators.
- To ensure secure and reliable data handling through authentication and database management.

Technologies Used

Frontend:

React: For building the user interface.

Tailwind CSS: For styling the application.

React Router: For client-side routing.

TanStack Query: For handling HTTP requests and error management.

Backend:

Node.js: The runtime environment.

Express.js: For setting up routes and middleware.

CORS: For managing cross-origin requests.

bcrypt: For hashing passwords.

JWT (JSON Web Token): For authentication.

MySQL2: For database connection and queries.

Database:

MySQL: The relational database management system.

Architecture

The application follows a client-server architecture, with a clear separation between the frontend and backend. The frontend is a single-page application (SPA) built with React, communicating with the backend via RESTful API endpoints provided by Express.js. The backend handles system logic and authentication in a MySQL database.

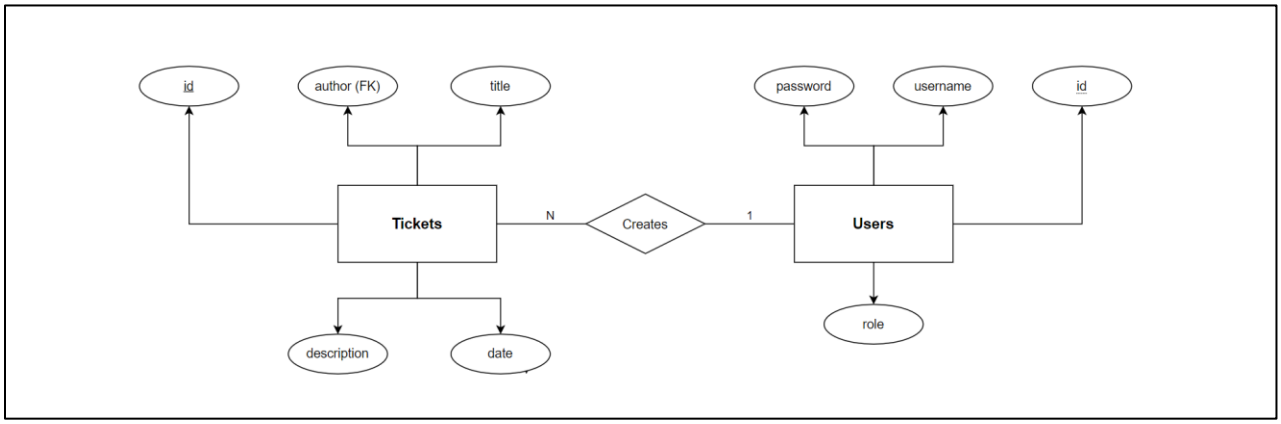
Key components:

Frontend: React components, styled with Tailwind CSS, handle the user interface and interactions.

Backend: Node.js with Express.js sets up API endpoints for user and ticket management.

Database: MySQL stores user information and tickets, with tables for Users and Tickets.

Entity Relation Diagram



Setup Instructions

First make sure to have Node.js installed.

Here is the link to download Node.js:

https://nodejs.org/en/download/prebuilt-installer

Database Setup:

1. Open the MySQL Command Line.
2. Create a new user: CREATE USER `admin`@`localhost` IDENTIFIED BY 'admin';
3. Create your database: CREATE DATABASE ticketing_system_db;
4. Grant all privileges: GRANT ALL PRIVILEGES ON ticketing_system_db.* TO 'admin'@'localhost';
5. Refresh permissions: FLUSH PRIVILEGES;

Frontend Setup:

1. Navigate to the frontend directory.
2. Run “npm install” to install dependencies.
3. Run “npm run dev” to start the development server.

Backend Setup:

1. Navigate to the backend/src directory.
2. Run “npm install” to install dependencies.
3. Ensure MySQL server is running and accessible.
4. Create a .env file with database credentials and JWT secret.
5. Set these in .env file: DB_HOST=localhost, DB_USER= admin, DB_PASSWORD=admin, DB_DATABASE=ticketing_system_db and JWT_SECRET=7b8e2f1a9c4d3e5b0a6f8c7d2e1b4a9e.
6. Run “node app.js” to start the backend server.

Running the Application:

Access the frontend at http://localhost:5173.

The backend runs on http://localhost:3000.

Features

- User Authentication: Users can sign up and log in securely using bcrypt for password hashing and JWT.
- Ticket Submission: Authenticated users can submit tickets with a title and description.
- Admin Panel: An admin can log in to view all tickets, with each ticket showing the id, author, title, description, and date of submission.
- Detailed Ticket View: Admins can click on a ticket link to view its details individually.

Conclusion

The Ticketing System Web Application successfully meets its objectives by providing a secure and efficient platform for issue tracking. The use of modern web technologies ensures a responsive and smooth user experience.