Ticket Management System Documentation

By
Kartikey Singh
IIT ROORKEE

Table of Contents:

Introduction
The Ticket Management System is a web-based application developed in React for efficient tracking and organization of tickets.

Getting Started

Prerequisites
Node.js (v14 or higher)
npm (v6 or higher)

Installation:

Clone the repository:
git clone [repository-url]

Navigate to the project directory:
cd ticket-management-system

Install dependencies:
npm install

Project Structure
The project is structured to enhance maintainability and scalability. Key directories include:

/src: Contains the source code of the application.
/public: Includes the HTML template and other static files.
Dependencies
Key dependencies include:

react: JavaScript library for building user interfaces.
axios: Promise-based HTTP client for making API requests.
uuid: Library for generating unique identifiers.
Configuration
The application is configured to interact with an API. Configuration settings can be found in the /src/config.js file.

Usage
Run the Application

To run the application locally, use the following command:
npm start

View the Application

Access the application in a web browser at http://localhost:3000.

Features

Group tickets by status, user, and priority.
Sort tickets based on priority and title.
Visualise tickets in a clean and organised manner.
Code Overview
App.js
App.js is the main entry point, rendering the main components and initialising the application.

Navbar Component

The Navbar component provides options for grouping and ordering tickets. It utilises SVG icons for enhanced visual appeal.

List Component

The List component handles the rendering of ticket lists based on the selected group.

Card Component

The Card component represents an individual ticket and includes relevant details such as ID, user profile, title, tags, and priority.

Styling

CSS Structure

Styling is managed through CSS files. Key classes and styles are defined in /src/styles.

Customization

Users can customise the styling by modifying the CSS files.

API Integration

Endpoint
The application interacts with a provided API endpoint for fetching ticket and user data.

Data Refactoring

Data from the API is refactored to enhance readability and organisation. Refactored data is used to populate ticket details.

Functionality

Grouping and Sorting
Tickets can be grouped by status, user, or priority. Sorting options include priority and title.

Local Storage

User preferences for grouping and sorting are stored in local storage for persistence.