# Genix Auctions App Documentation

Welcome to the Genix Auctions App setup guide. This document provides comprehensive and easy-to-follow instructions to help you set up and run the application seamlessly. As a senior full-stack developer, you will find best practices embedded in each of these steps to ensure smooth deployment and maintainability of the application.

## Overview

Genix Auctions App is a full-stack web application designed to facilitate online auctions. Users can register, create auction items and place bids. The application follows a modular approach and is built with Angular on the front end, Node.js with Express for the backend, and MySQL as the database.

### Tech Stack:

* **Frontend**: Angular 16 with NgRx for state management.
* **Backend**: Node.js with Express.js.
* **Database**: MySQL.

## Prerequisites

1. **Node.js & npm**: Ensure that Node.js (v14 or later) and npm (v6 or later) are installed.
2. **MySQL**: Install MySQL and ensure it's running locally or on a server you can access.
3. **Angular CLI**: Install Angular CLI globally using npm:

## Developer Information

## Developed by Muthu Krishna Santhosh

## Security

## Password Hashing: We use bcrypt for secure password hashing.

## Authentication: User authentication is handled securely using JSON Web Tokens (JWT).

## Database Setup

1. **Create Database** Open MySQL Workbench or any MySQL client, and execute the following script to create the required database schema:

## Backend Setup

1. **Navigate to the Backend Directory**
2. **Install Dependencies**
3. **Environment Configuration** Create a .env file in the backend directory and add the following configurations:

Replace your\_password with the actual MySQL password.

1. **Run the Backend**

The backend should now be running on <http://localhost:60532>.

## Frontend Setup

1. **Navigate to the Frontend Directory**
2. **Install Dependencies**
3. **Run the Frontend**

The frontend should now be accessible at <http://localhost:4200>/auctions/auctions-dashboard.

## NgRx State Management

The frontend utilizes NgRx for managing the state of authentication, auctions, and bidding. This ensures that the application state is predictable and maintainable.

* **Store Module**: All store modules for authentication, auctions, and bidding are imported in store.module.ts.
* **Auth Actions**: Actions such as login, register, logout are dispatched to update the store.
* **Effects**: The side effects, such as login success and navigation, are managed through AuthEffects.

## Auth Guard and Route Configuration

* **AuthGuard**: Protects certain routes from being accessed without authentication.
* **Avoid AuthGuard**: To make the landing page http://localhost:4200/auctions/auctions-dashboard accessible without authentication, avoid applying AuthGuard on this route in the AuctionsRoutingModule.

## Swagger API Documentation

To add Swagger API documentation:

1. **Install Swagger Dependencies**
2. **Add Swagger to the Backend** Update app.js with the following:
3. **Access Swagger Documentation** Visit http://localhost:60532/api-docs/ to see the API documentation.

## Testing the Application

* **Postman**: You can use Postman to test the API endpoints.
* **NgRx DevTools**: Use the NgRx DevTools to inspect the application state and dispatched actions during development.

## Conclusion

The Genix Auctions App is designed to be scalable and maintainable by following modern web development best practices. By using Angular, Node.js, MySQL, and NgRx, this application provides a robust solution for online auctions.

For any issues, please refer to the logs generated by Angular or Express, or consult the documentation for each framework/library used.