Documentation for Online Bidding System

Project Attribution

This bidding website project was developed by:

- **YUVARAJ KRISHNAN B (EC21B1059)**
- **M HEMACHANDIRAN (CS21B1015)**

Table of Contents:

PROJECT OVERVIEW:

The goal of this project is to build a full-stack web application that allows users to participate in an online bidding system. This application includes user authentication, auction item listing, bidding functionality, and a user-friendly interface for managing and viewing bids. The project is designed to assess the candidate's ability to design, implement, and document a complete web application.

Features of the Project :

- **User Authentication**: Secure registration and login system.
- **Auction Item Management**: Users can create, view, update, and delete auction items.
- **Bidding Functionality**: Users can place and track bids on auction items.
- **User Interface**: Intuitive and responsive design for easy navigation and use.

Pre-requisites:

- Node.js (>=14.x)
- Angular 17
- MongoDB
- Express.js

Installation

1. Node js:

Installs a node binary into your project, which because npm runs scripts with the local ./node_modules/.bin in the PATH ahead of the system copy means you can have a local version of node that is different than your system's, and manage node as a normal dependency.

Install

```
> npm i node
```

2. **Angular :**
   a) **Install Angular CLI**

```
npm install -g @angular/cli
```

b) Create a New Project

```
ng new my-angular-project
```

C) Navigate to the Project Directory

```
cd my-angular-project
```

d. Start the Development Server

```
ng serve
```

3) Mongoose

Npm install mongoose

4) Express

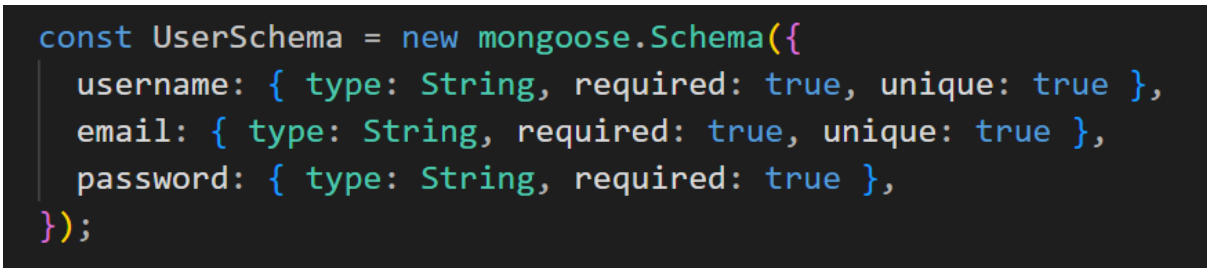
Npm install express

License:

This project is licensed under the MIT License - see the LICENSE.md file for details.

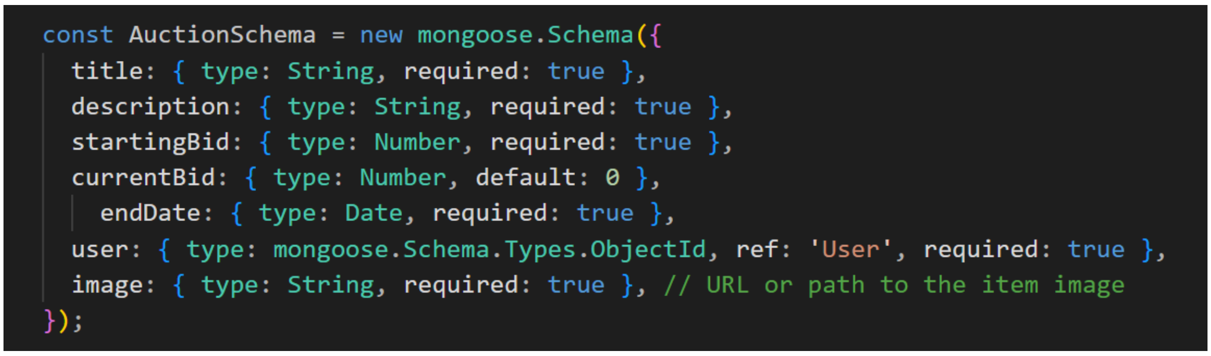
Database Schema:

Users Collection:



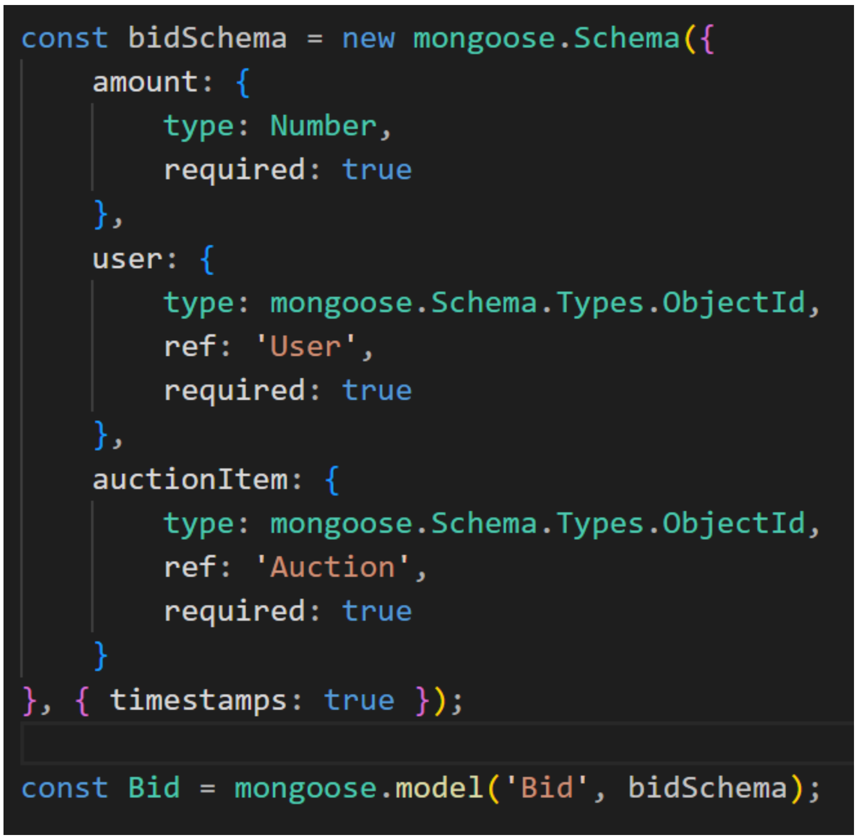
```
const UserSchema = new mongoose.Schema({
  username: { type: String, required: true, unique: true },
  email: { type: String, required: true, unique: true },
  password: { type: String, required: true },
});
```

Auction Items Collection:



```
const AuctionSchema = new mongoose.Schema({
  title: { type: String, required: true },
  description: { type: String, required: true },
  startingBid: { type: Number, required: true },
  currentBid: { type: Number, default: 0 },
    endDate: { type: Date, required: true },
  user: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true },
  image: { type: String, required: true }, // URL or path to the item image
});
```

Bids Collection:



```
const bidSchema = new mongoose.Schema({
    amount: {
        type: Number,
        required: true
    },
    user: {
        type: mongoose.Schema.Types.ObjectId,
        ref: 'User',
        required: true
    },
    auctionItem: {
        type: mongoose.Schema.Types.ObjectId,
        ref: 'Auction',
        required: true
    }
}, { timestamps: true });

const Bid = mongoose.model('Bid', bidSchema);
```

Application Flow:

1. User Registration and Login:

- **Users register with their username, email, and password.**
- **Users log in with their email and password to access their account.**

2. Auction Management:

- **Users can create auction items by providing a title, description, starting bid, and end date.**
- **Users can view, update, and delete their auction items.**

3. Bidding Functionality:

- **Users can place bids on auction items.**
- **Users can view the current highest bid and bid history.**
- **Users receive notifications when they are outbid.**

Technologies Used:

- **Frontend: Angular 17**
- **Backend: Node.js, Express.js**
- **Database: MongoDB**

API Documentation :

User Authentication :

- **Register: POST /signup**
  - **Request Body: { "username": "string", "email": "string", "password": "string" }**
  - **Response: 201 Created**

- **Login: POST /login**
  - **Request Body: { "email": "string", "password": "string" }**
  - **Response: 200 OK**

Auction Management :

- **Create Auction Item: POST /auction-items**
  - **Request Body: { "title": "string", "description": "string", "startingBid": "number", "endDate": "string" }**
  - **Response: 201 Created**

- **View All Auction Items: GET /auction-items**
  - **Response: 200 OK**

- **Update Auction Item: PUT /auction-items/:id**
  - **Request Body: { "title": "string", "description": "string", "startingBid": "number", "endDate": "string" }**
  - **Response: 200 OK**

- **Delete Auction Item: DELETE /auction-items/:id**
  - **Response: 200 OK**

Bidding Functionality

- **Place Bid: POST /auction-items/:id/bids**
  - **Request Body: { "amount": "number" }**
  - **Response: 201 Created**

- **View Current Highest Bid and Bid History: GET /auction-items/:id/bids**
  - **Response: 200 OK**