# JS Back-End Exam – Auction House

## Exam Rules

1. You have **4 hours**. When you are ready, make sure all **dependencies** are listed in the ***package.json*** file and submit your project **as described at the end of this file.** **Only the last submission will be evaluated!**
2. You are provided with **HTML & CSS** files. You may **add attributes** (such as **class** and **dataset**), but shouldn’t **change** **existing** ones, except for **href** on links and the **method** and **action** attributes of HTML forms.
3. Use **express**, **MongoDB** as a database with [**mongoose**](https://mongoosejs.com/docs/) and [**bcrypt**](https://www.npmjs.com/package/bcrypt) **for hashing of passwords.**
4. You can use any **view engine** of your choice ([**handlebars**](https://www.npmjs.com/package/express-handlebars), EJS, Pug, etc.). **Client-side code is not allowed.**
5. The application **must start** from a file named **"index.js"** on port **3000.**

## Application Overview

Get acquainted with the provided HTML and CSS and create an application for **online auctions**. Visitors can view the **Home page** and **Browse** **page**, they can also **register** withan **email**, **password**, and **name**, which will allow them to publish their **auctions**, as well as **bid** for existing auctions (but only if the **current user** is **not** the **author**). Authors can **edit** or **delete** their own publications.

  Graphical user interface, website

Description automatically generated Graphical user interface

Description automatically generated

Graphical user interface

Description automatically generated Graphical user interface, application

Description automatically generated

## Database Models

The **Database** of the **Auction House** application needs to support **2 entities.**

**Note**: **You may add additional properties to the models if this will help you solve the task**.

### User

* **Email** – string (required)
* Password – string (required)
* **First name** –string (required)
* **Last name** –string (required)

You should make the following validations:

* The **email** should be **in the following format: <name>@<domain>.<extension>**
  + Only English letters are allowed for any of the parts of the email
  + There must be a **@** after the **<name>**
  + There must be a **dot** after the **<domain>**
  + Example of a valid email - "**petar@softuni.bg**"
* The **password** should be **at least 5 characters** long
* The **first** and **last names** should be **at least 1 character long**

### Auction

* **Title** – string (required)
* **Description** – string
* **Category** – string (required)
* **Image URL** – string
* **Price – number** (required)
* **Author** –**reference to the User model (required)**
* **Bidder** – **reference to the User model**

When a user bids on an auction, their ID is stored in the Bidder field of the auction.

You should make the following validations:

* The **title** should be a minimum of **4 characters** long
* The **description** should be a maximum of **200 characters** long
* The **category** should be one of the following: Vehicles, Real Estate, Electronics, Furniture, Other
* The **price** cannot be negative

## Functional Requirements (80 pts)

**Note:** Don't forget to change the paths to the CSS files, and the images.

### Navigation

The navigation bar at the top of the page should be dynamic and display different links, depending on the role of the current user.

**Guest** navigation example:



**User** navigation example:



### Home Page

Show a welcome screen with links to **Browse Listings**. If the current user is **logged in**, show a link to **Publish Auction**.



### Register Page

Register a user inside the database with **email**, **password**, **first** and **last names**. **Password** inside the database must be **hashed** and both **passwords** must **match**! After successful registration, **redirect to the Home page as an already logged-in user.**

Graphical user interface

Description automatically generated

### Login Page

Logging an already registered user with the correct **email** and **password**. After successful login, **redirect to the Home page as an already logged-in user.**

Graphical user interface

Description automatically generated

### Logout

The logout action is available to **logged-in** users. Upon success, clear any session information and **redirect** the user to the **Home page**.

### Browse Auctions

List of all **auctions**. **Each** **auction** is shown with its title and current price, as well as the **Details** button that leads to the details page for the specific auction.



If **there are** **no auctions** in the **database**, display the following view:

Graphical user interface, application

Description automatically generated

### Details Page

Information about the auction:

* **Title**
* **Category**
* **Description**
* **Current price**
* **Publisher full name**
* **Actions** (Depending on the status of the current user)

#### Guest

If the user has **not logged in**, **no buttons** should be displayed

Graphical user interface, text, application

Description automatically generated

#### Author

If the **currently logged-in** user is the **author** of the auction, the **[Edit]** and **[Delete]** buttons should be displayed.

Graphical user interface, website

Description automatically generated

Also, **only the author** should see the **name** of the current **bidder**:

Graphical user interface, website

Description automatically generated

If there are no bids yet, show the text **No bids**:

Graphical user interface, website

Description automatically generated

#### User

**Registered users** can **place a bid** on any auction that isn’t their own.

Graphical user interface, website

Description automatically generated

If the current user is the **highest bidder** for the auction, show this information instead of the Place bid form:

Graphical user interface, website

Description automatically generated

### Place Bid

Any **registered user** who **isn’t the author** of an auction and **isn’t** **the current bidder**, can place a bid. After a successful bid, the **ID** of the user must be stored in the **bidder** field of the auction and the amount that they bid becomes the new value for **price**. **Redirect** the user to the **Details** page for the auction**.**

If a user is already the current bidder, or if the **amount** they submitted is **less than or equal** to the current price, an **error message** should be displayed.

### Publish Auction

The Publish page is available to **logged-in users**. It contains a form for adding a new **auction**. Upon success, **redirect** the user to the **Browse** **page**.

Graphical user interface, application

Description automatically generated

### Edit Auction

The **Edit** **page** is available to logged-in users and it allows authors to **edit** their publications. Clicking the [**Edit**] **button** of a particular auction on the **Details** **page** should display the **Edit** **page**, with **all fields filled** with the data for the auction. It contains a form with input fields for all relevant properties.

Graphical user interface, application

Description automatically generated

Auctions **without** a bidder may have their price adjusted. If the current auction **has a bidder**, the **price cannot be changed** – the form field should be disabled.

Upon success, **redirect** the user to the **Details** **page** for the current auction.

### Delete Auction

**Authors** should be able to click over the [**Delete**] **button** – deleting the current ad from the **database** and the user should be **redirected** to the **Browse page.**

## Security Requirements (10 pts)

Limit the following functionality to be accessible only to certain user roles:

* Only Guests can access the Login **page** and functionality
* Only Guests can access the Register **page** and functionality
* Only Users can access the **Publish Auction** **page** and functionality
* Only Users can access the Place bid functionality and only if they aren’t the author of the auction
* Only Users can access the Logout functionality
* Only Authors can access the Delete and Edit functionalities for their own auction

If Guests try to **access** а page they should **not be able to**, you must **redirect** them to the **Login** page.

If **Users** try to **access** а page they should **not be able to**, you must **redirect** them to the **Home Page**.

## Validation and Error Handling (10 pts)

Use the following view for **invalid** **paths**:

Timeline

Description automatically generated

The application should **provide feedback** to the user about the result of their actions. **You can choose to display the first error or all of them. You are free to choose the content of the error message.**

In case of errors, like form validation or placing an invalid bid, you **should display a div** with class **"error-box"**



## Bonus – Closed Auctions (15 pts)

Authors can **close their auctions** and view a **list of auctions** they have closed.

On the **Details page** of auctions created by the current user, if there is a bidder, a **Close Auction** button should appear near the price information. Clicking this link should **redirect** the user to the **Closed Auctions page** and causes the auction to **no longer be displayed** when browsing the **catalog**. Instead, it will now be visible in the **Closed Auctions page** for the current user.

Graphical user interface, website

Description automatically generated

In the **navigation bar** of Users, an additional link is displayed, that leads to a list of their **Closed Auctions**:

Graphical user interface, website

Description automatically generated

The list of closed auctions shows the **title** of each auction as well as the **final price** and the **name of the bidder**. Closed auctions **do not have a details page**.

Graphical user interface, website

Description automatically generated

If the current user hasn’t closed any auction, display the following message instead:

Graphical user interface, application

Description automatically generated

## Submitting Your Solution

Place in a **ZIP** file your project folder. Exclude the node\_modules folder. Upload the archive to Judge.









*GOOD LUCK! 😊*