## JS Back-End Exam – Crypto Trade

## Exam Rules:

1. You have **4 hours**
2. When you are ready, delete the ***node\_modules***folder, make sure all dependencies are listed in the ***package.json*** file and submit your archiving project.
3. You are provided with **HTML & CSS** resources
4. You may **add attributes** (such as **class** and **dataset**), but it is forbidden to **change** **existing** attributes (such as **class** and **id**)
5. You may **change "href"** attributes on links and add/change the **method** and **action** attributes of HTML Forms.
6. Use **Express.js** as a back-end framework
7. Use **MongoDB** as a database with [**mongoose**](https://mongoosejs.com/docs/)
8. You can use whatever **view engine** you like ([**express-handlebars**](https://www.npmjs.com/package/express-handlebars), EJS, Pug, etc.)
9. Use [**bcrypt**](https://www.npmjs.com/package/bcrypt) **for hashing the password**
10. The application **must start** from the file **"index.js"** on port **3000**
11. It is **forbidden** to **use** **React**, **Vue**, **Angular,** etc.
12. **Only the last submission will be evaluated!**

## Application Overview

Get acquainted with the provided **HTML and CSS** and create an application for the **crypto trade**.

The visitors can **view** the **Home page** and **All Crypto** catalog with available offers. They can also **register** with an **email**, **username**, and **password**, allowing them to create their offers for crypto trade and **buy crypto** (if the **current user** is **not** the **owner of the crypto** offer). Authors can edit or delete posts at any time.

## Functional Requirements

### Guest (not logged in)

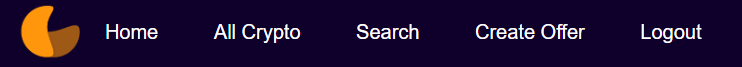
**Guest** navigation example:



The **application** should provide Guest (not logged in) users with the functionality to Login, Register, and **view** the Home page, **All Crypto** page, and the **Details** page.

### Users (logged in)

**User** navigation example:



The **application** should provide **Users** (logged in) with the functionality to:

* **View the Home page and all other pages with logged-in navigation**
* **View All Crypto page**
* **Create а new crypto offer [Create Offer]**
* **Access crypto details page [Details]**
* **Buy crypto (**if the **current user** is **not** the **owner of the offer)**
* **Delete or Edit the offer depending on the user's authentication (only for the owner of the current offer for crypto)**

## Database Models

The **Database** of the **Crypto Trade** application needs to support **two entities**:

### User

* Username - string (required),
* Email - string (required),
* Password - string (required)

### Crypto

* **Name - String (required),**
* **Image: String (required),**
* **Price: Number (required),**
* **Crypto Description: String (required),**
* **Payment Method: String (crypto-wallet, credit-card, debit-card, paypal) required,**
* **Buy a crypto - a collection of Users (a reference to the User model)**
* **Owner - object Id (a reference to the User model)**

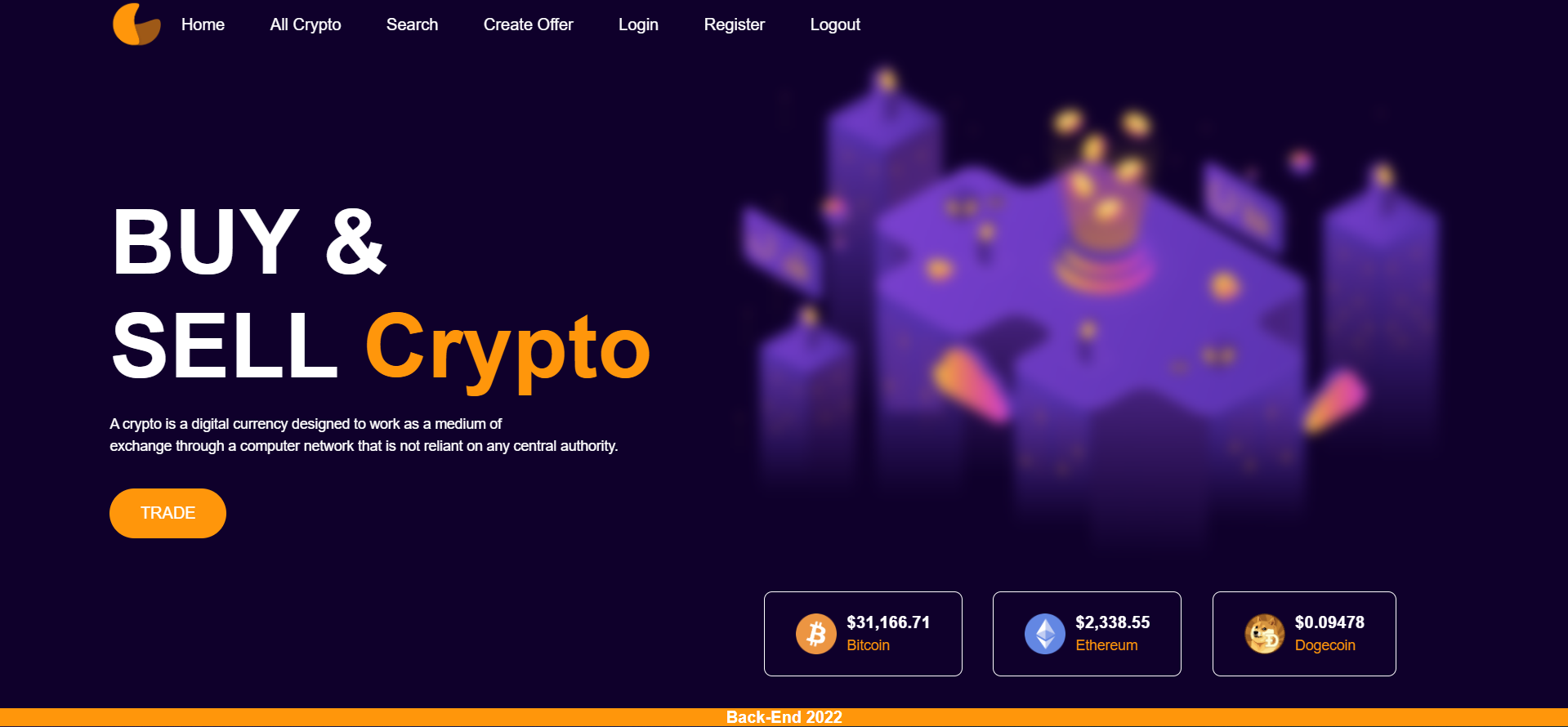
**Note:** When a user **buys crypto**, their **id** is added to that collection (**Buy a crypto**)

Implement the entities with the correct data types.

## Application Pages (80 pts)

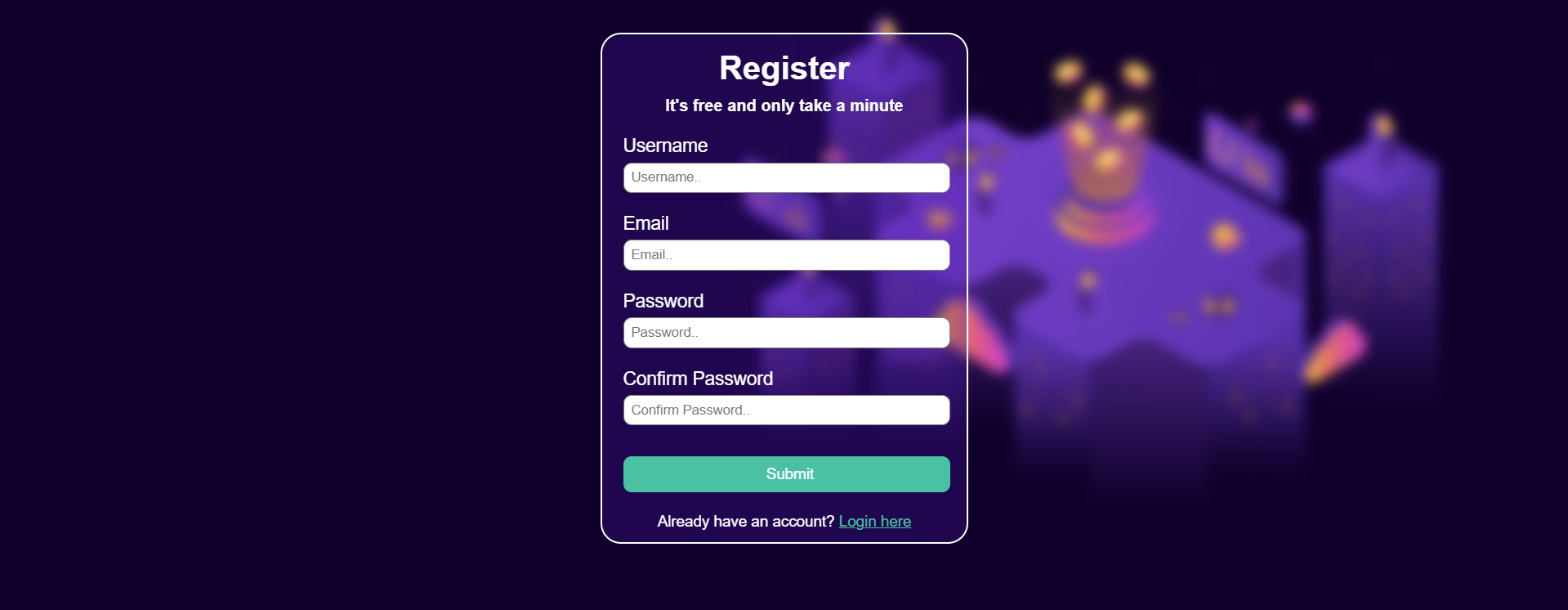
#### Home Page (For logged in users and logged-out users)

Visualize **static home page**.



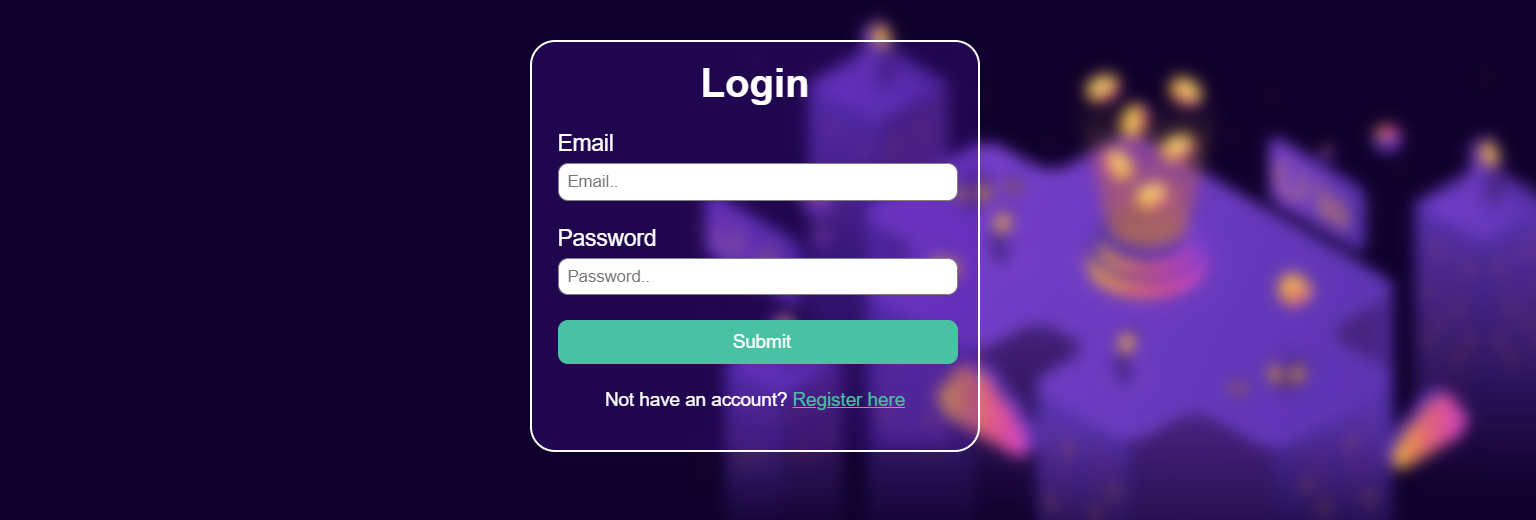
**Register Page (Logged Out User)**

**Register** a user inside the Database with a **username, email, and password**. **Password** inside the **Database** must be hashed (**use bcrypt**), and both passwords **must match**! After successful registration, you should **redirect** to the **Home page**.



**Login Page (Logged Out User) - (5 Pts)**

Logging an already registered user with the correct **email** and **password**. After successful login, you should **redirect to the Home page.**



#### Logout (logged in user)

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

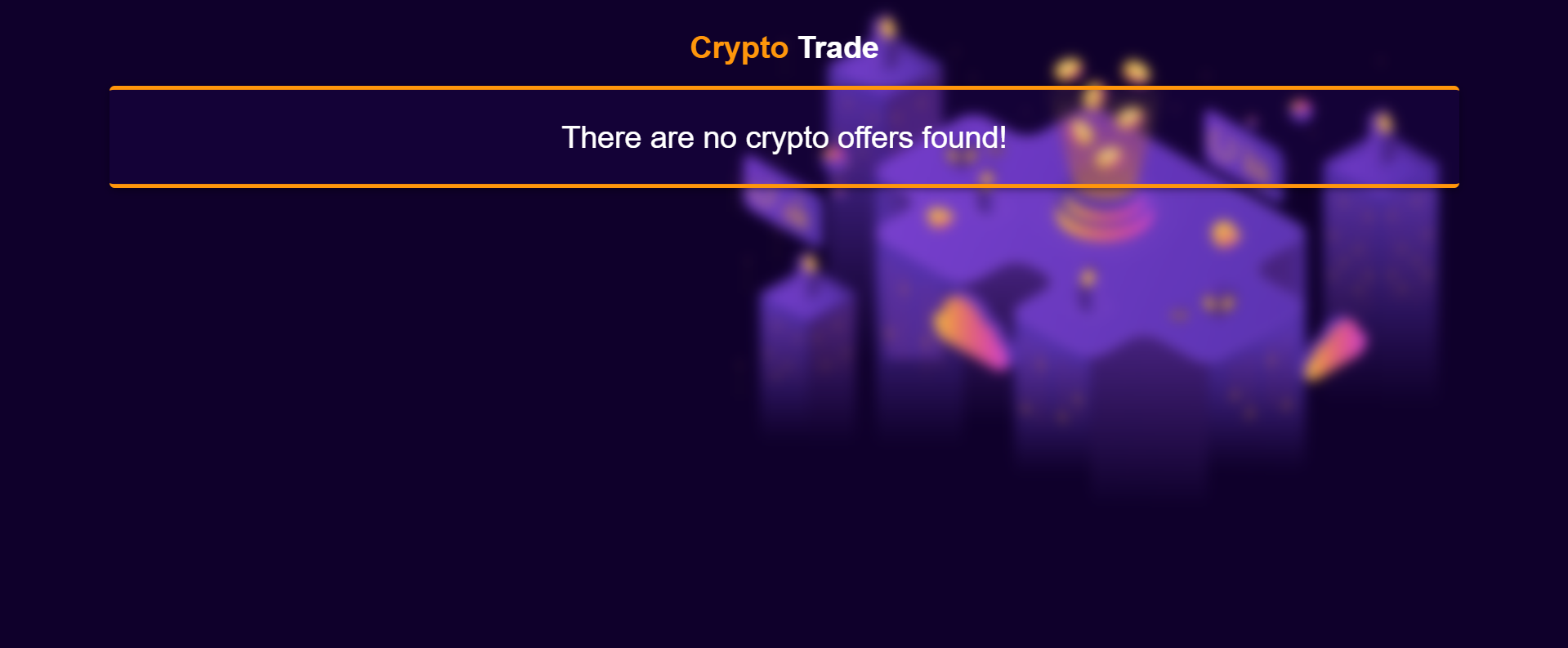
#### Crypto Catalog (For logged in users and logged out users)

List all crypto offers. Each offer must display information about the **crypto image**, the **name**, the **price**, **payment,** and a button for **details** about the **specific crypto**. As in the picture below:



[**Details**] button should be a link to the **details** **page** for the current crypto.

If there are **NO** crypto offers in the Database yet, display **"There are no crypto offers found! "**



**Details Page - (for logged in users and logged out users)**

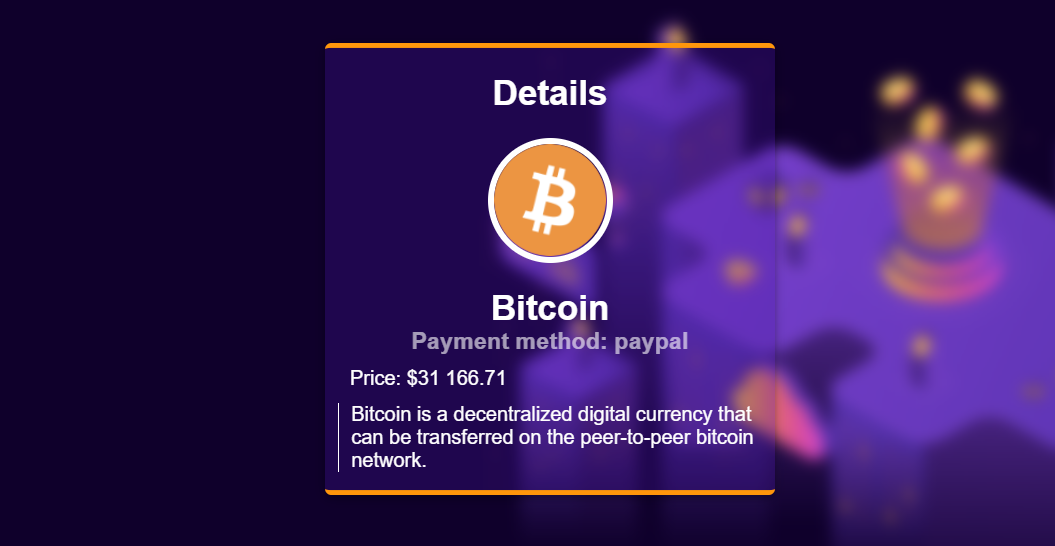
All users should be able **to see details**. Clicking the **Details button** on the crypto card should display the details page. If the currently registered user **is the creator** of the crypto offer, the **Edit** and **Delete** buttons should be displayed. Otherwise, they **should not be available**.

Information about the crypto:

* **Name**
* **Payment method**
* **Price**
* **Description**
* **Buttons** (Depending on the status of the currently logged in user)

#### Details Page (logged out users)

If there are **no logged**-in users, **no buttons** should be displayed.



#### Details Page (logged in user and creator of the current offer)

If the **currently logged-in** user is the **owner** (the user who **created the crypto offer**), he should see the **[Delete]** and **[Edit]** buttons.



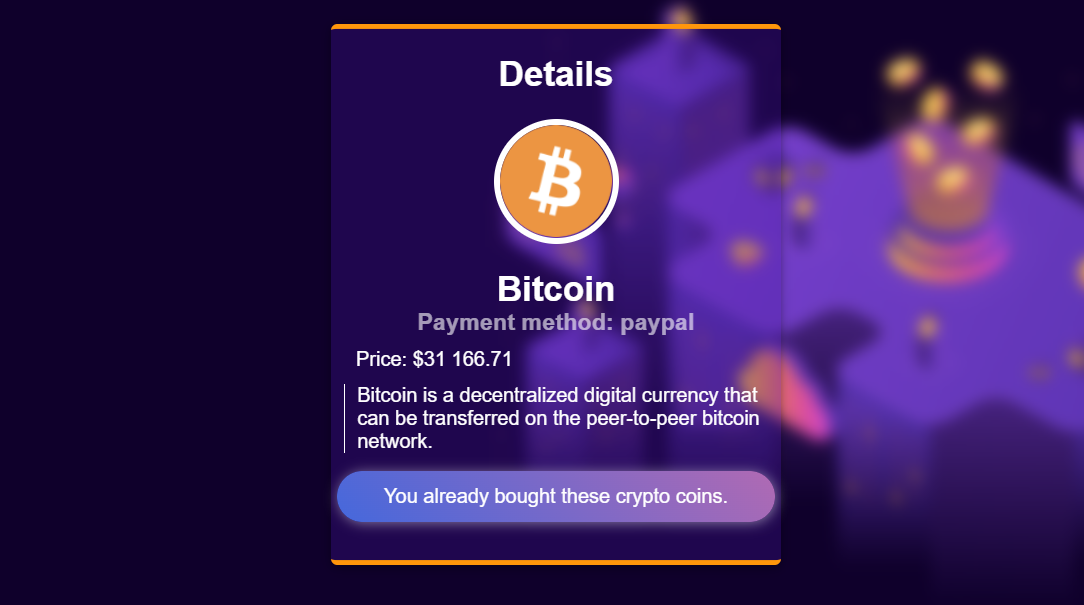
#### Details Page (logged in user with available places)

If the currently logged-in user is **not the owner** (a user who is not the creator of this crypto offer) and has not bought that crypto, he should see a **buy** button.



#### Details Page (logged in user already bought crypto)

If the currently logged in user **is not the owner** and has **already bought crypto**, he should see **[You have already bought this crypto].**



#### Buy Crypto (logged in user who is not the current crypto owner)

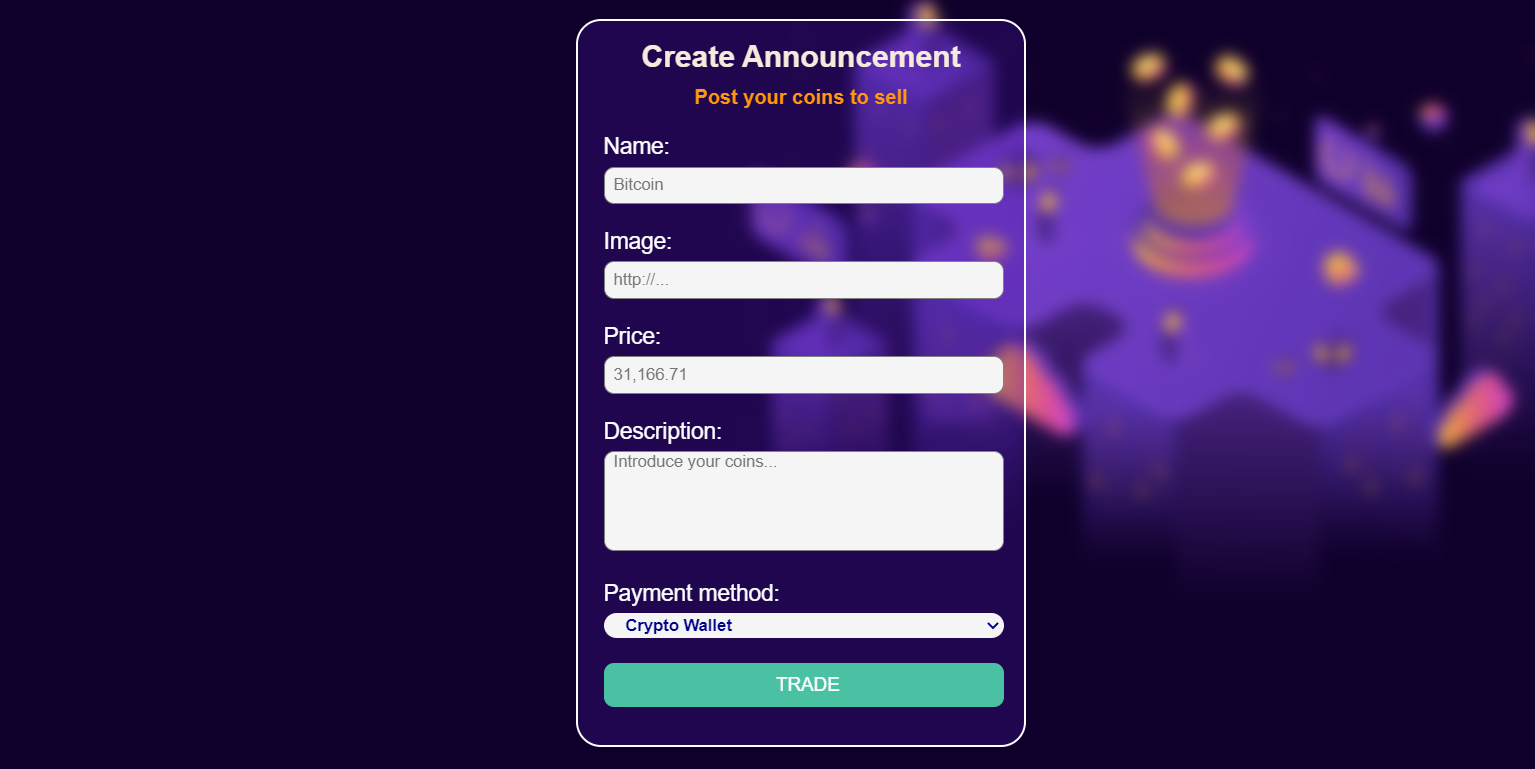
Any registered user who is **not** the current **owner** of the crypto offer must be able **to buy crypto** (if any).

If he manages to **buy the crypto successfully**, his **userId** must be added to the collection of **Buy a crypto** and **redirect** the user to the **Details** page for the current **crypto offer**.

If a user has once bought current crypto, he should see " **You already bought these crypto coins.**".

**Create Offer (Logged in User)**

The Create Offer page is available to logged-in users. It contains a form for adding a new crypto offer. Upon success, redirect the user to the All Crypto page.

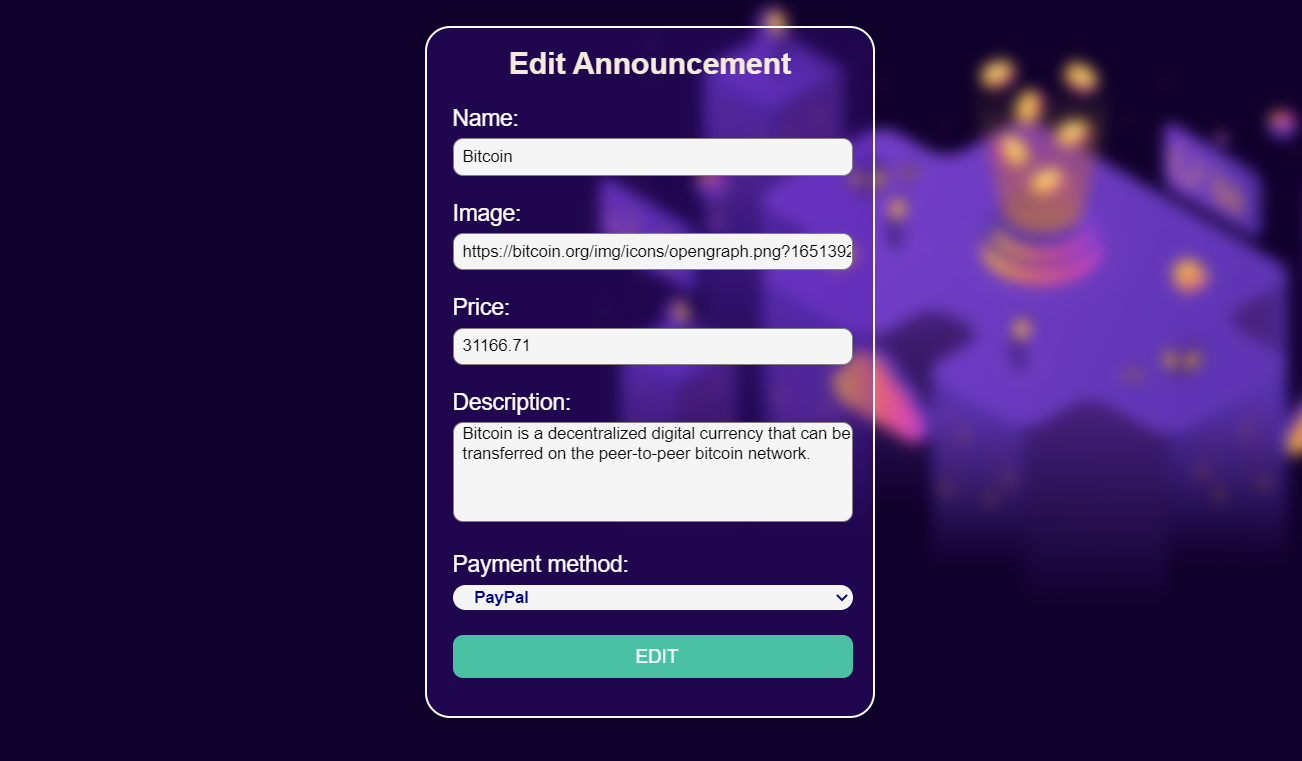


#### Delete Offer (logged in user and owner of the current offer)

Each **owner** of the crypto offer must be able to click on the **[Delete]** button and **delete the current crypto** from the Database, and the user must be redirected to the **All Crypto** page.

#### Edit Offer (logged in user and owner of the current offer)

Each **owner** can edit their **crypto offer**. Clicking the **[Edit]** button for a specific offer on the details page should display the **Edit page**, all fields being populated with crypto data. It contains a form with input fields for all relevant properties. If successful, redirect the user to the **current crypto details page**.

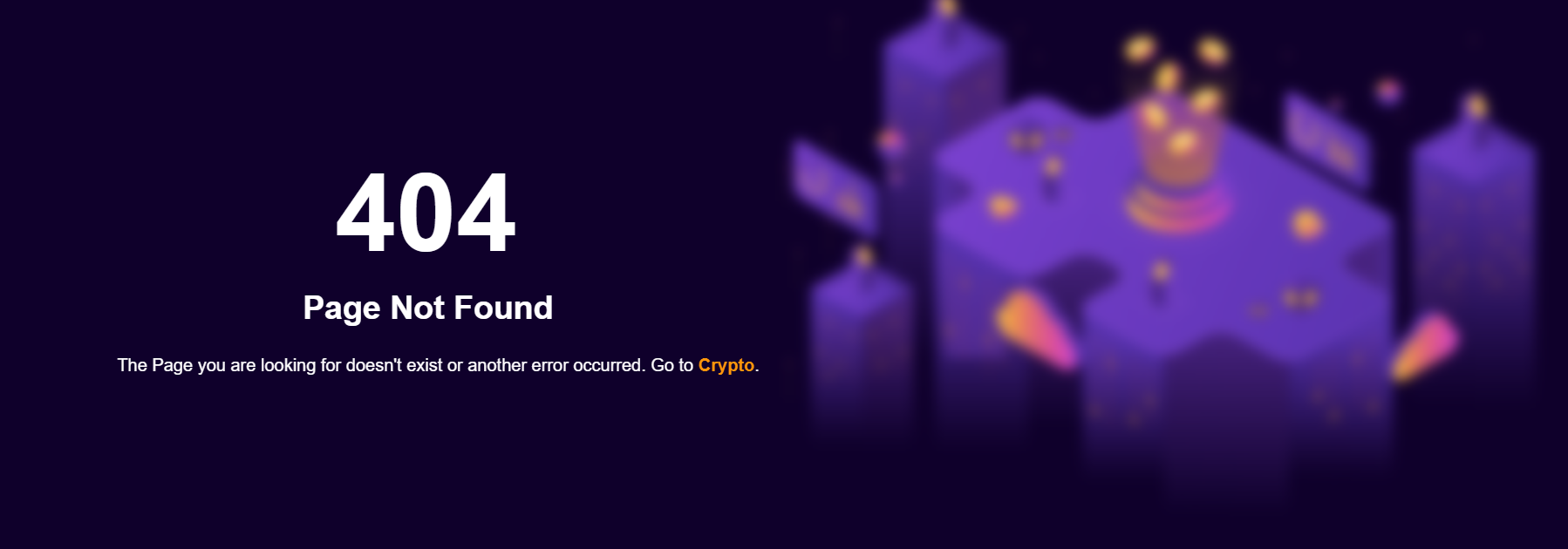


## Security Requirements (Routes Guards) - (10 Pts)

The Security Requirements are mainly **access** requirements. Configurations about which users can access specific functionalities and pages.

* Guest (not logged in) users can access the Home page.
* Guest (not logged in) users can access the Login page and functionality.
* Guest (not logged in) users can access the Register page and functionality.
* Guest (not logged in) and Users (logged in) can access the **All Crypto** page (Listed all crypto offers).
* Guest (not logged in) can access the **Details** page without functionality.
* Users (logged in) can access the Home page.
* Users (logged in) can access the Details page and functionality.
  + **Users** (not crypto offer owners) can **buy crypto**.
  + **Users** (crypto offer owners) can **Edit** and **Delete** the current crypto offer
* Users (logged in) can access **Create Offer** page and functionality.
* Users (logged in) can access Logout functionality.

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



## Validation and Error Handling (10 Pts)

### The application should notify the users about the result of their actions.

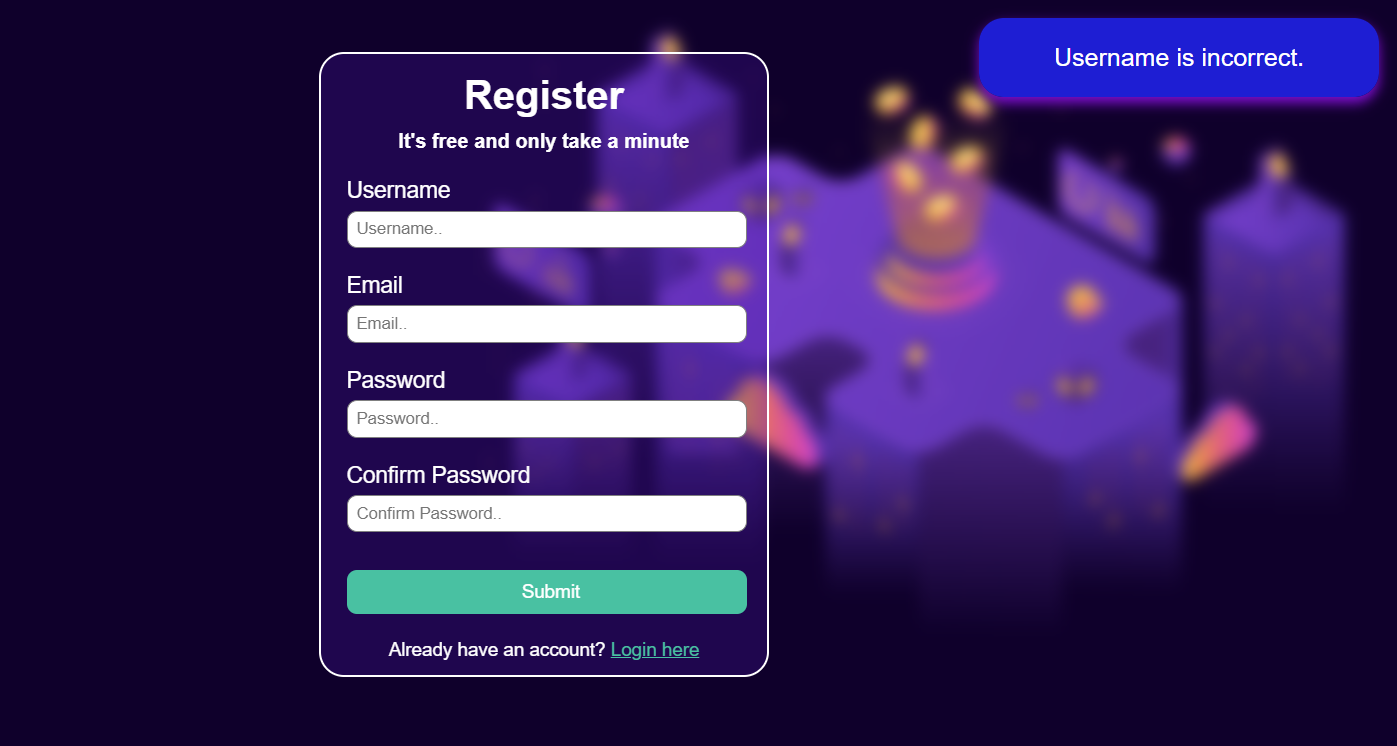
In case of error, you should display div with class "**errorContainer**"

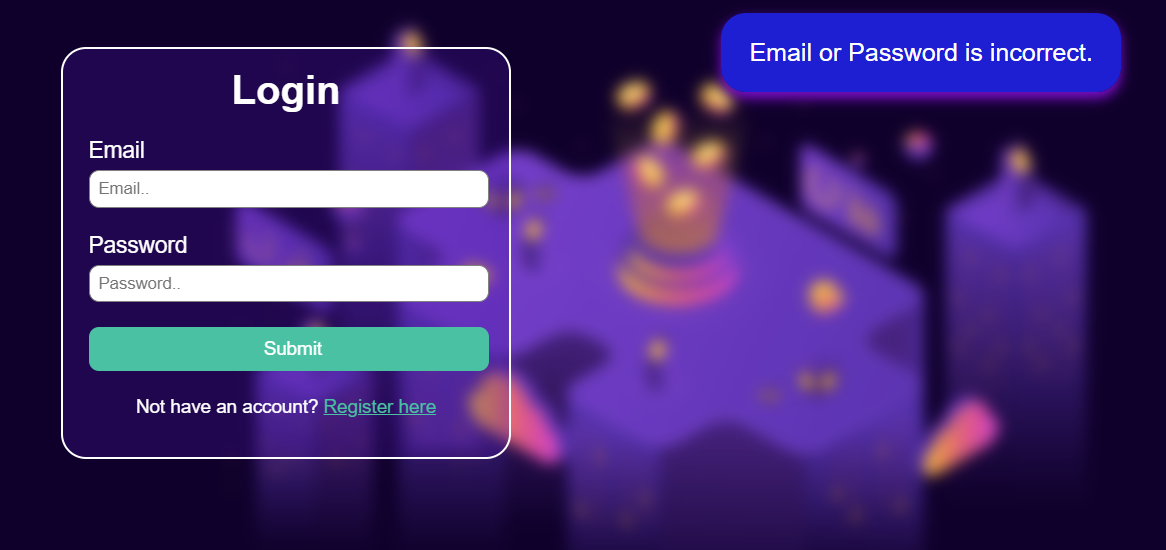
### You can choose to display the first error or all of them. You have complete freedom to choose the content of the error message you will display.

### Login / Register

You should make the following validations:

* The **username** should be **at least** **five characters** long
* The **email** should be **at least ten character** long
* The **password** should be **at least four characters** long
* The **repeat password** should be **equal to the password**

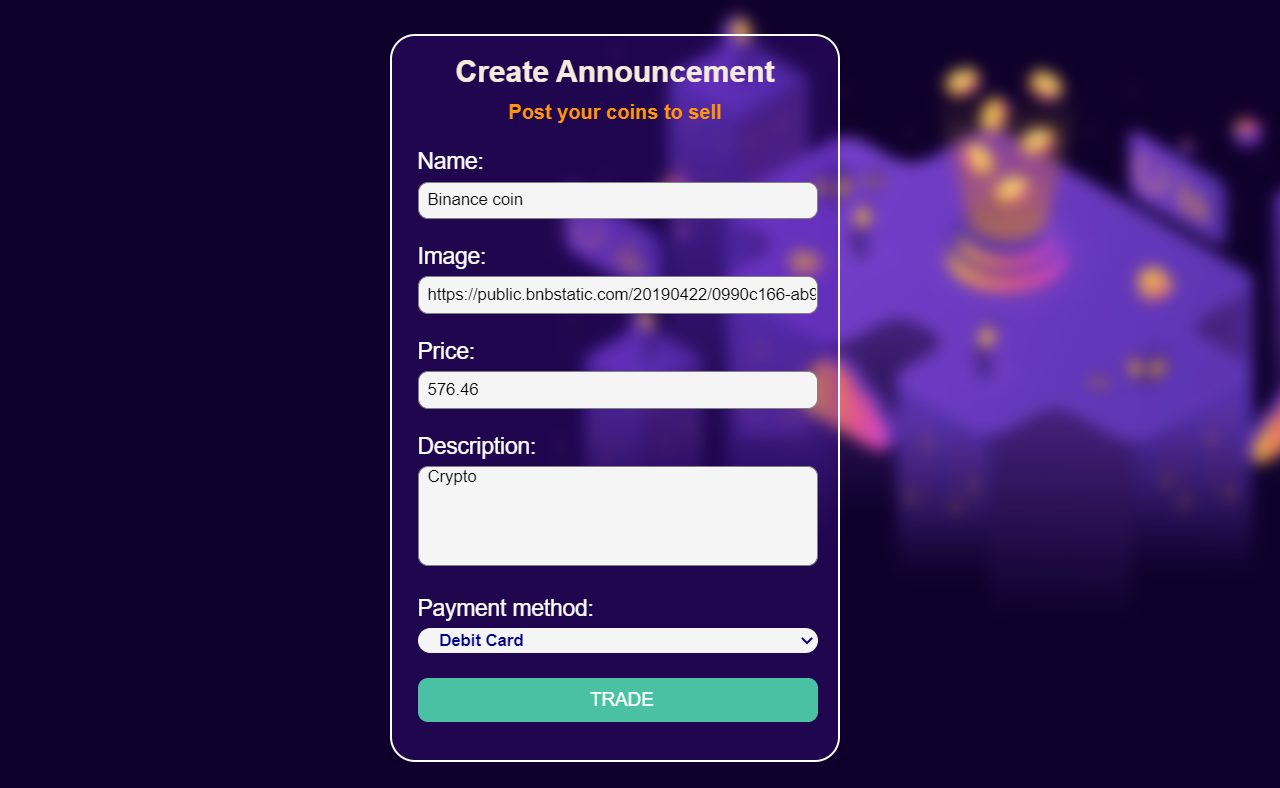


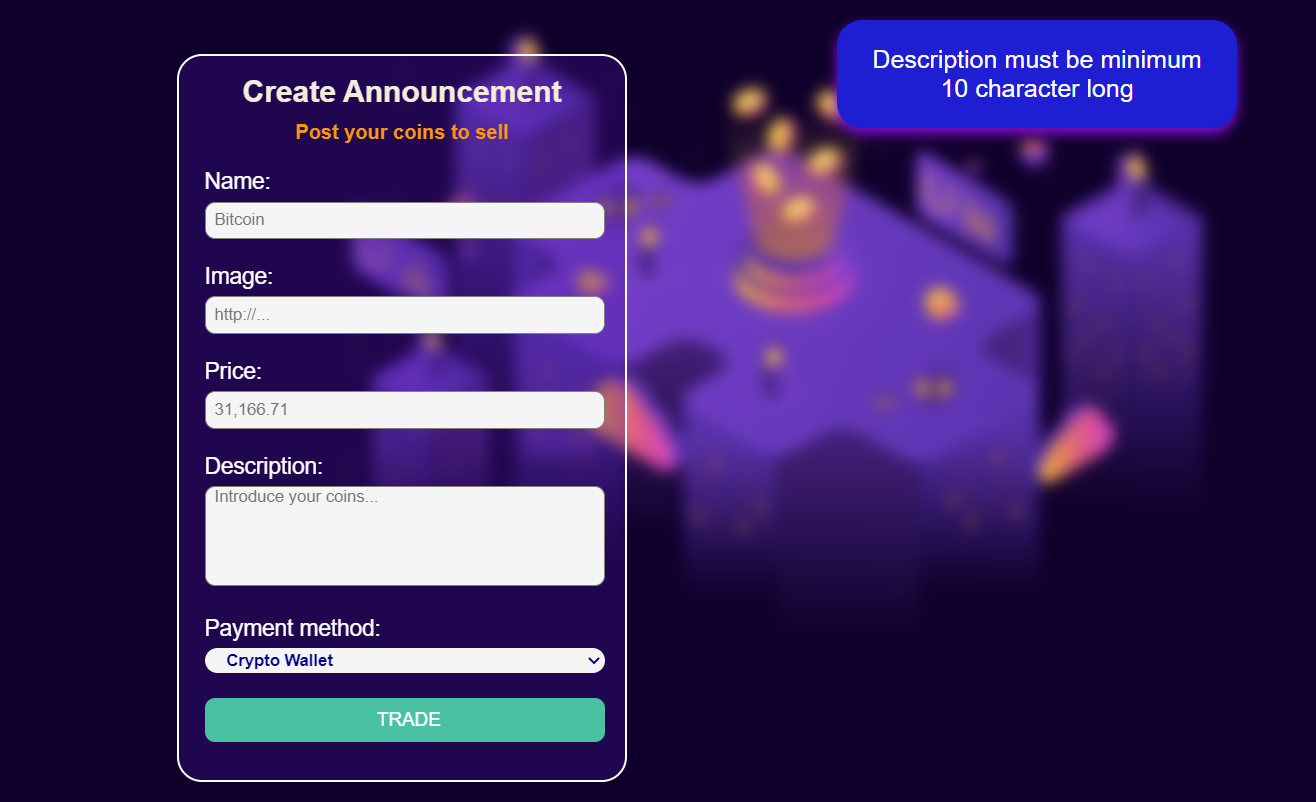


### Crypto

You should make the following validations while **creating** or **editing a** **crypto offer**:

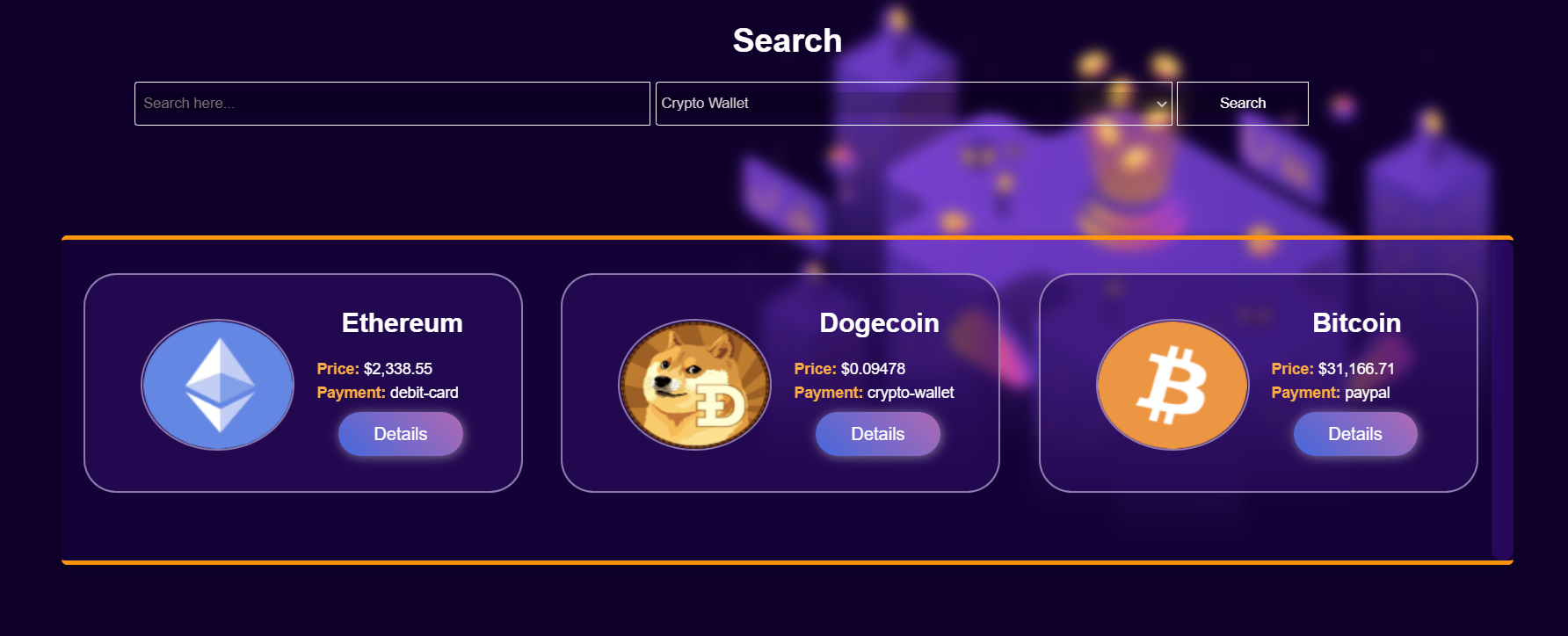
* The **Name** should be **at least two characters**
* The **Price** should be a **positive number**
* The **Crypto Image** should start with **http:// or https://**.
* The **Description** should be a minimum of **10 characters** long.
* The **Payment Method** must be one of the options



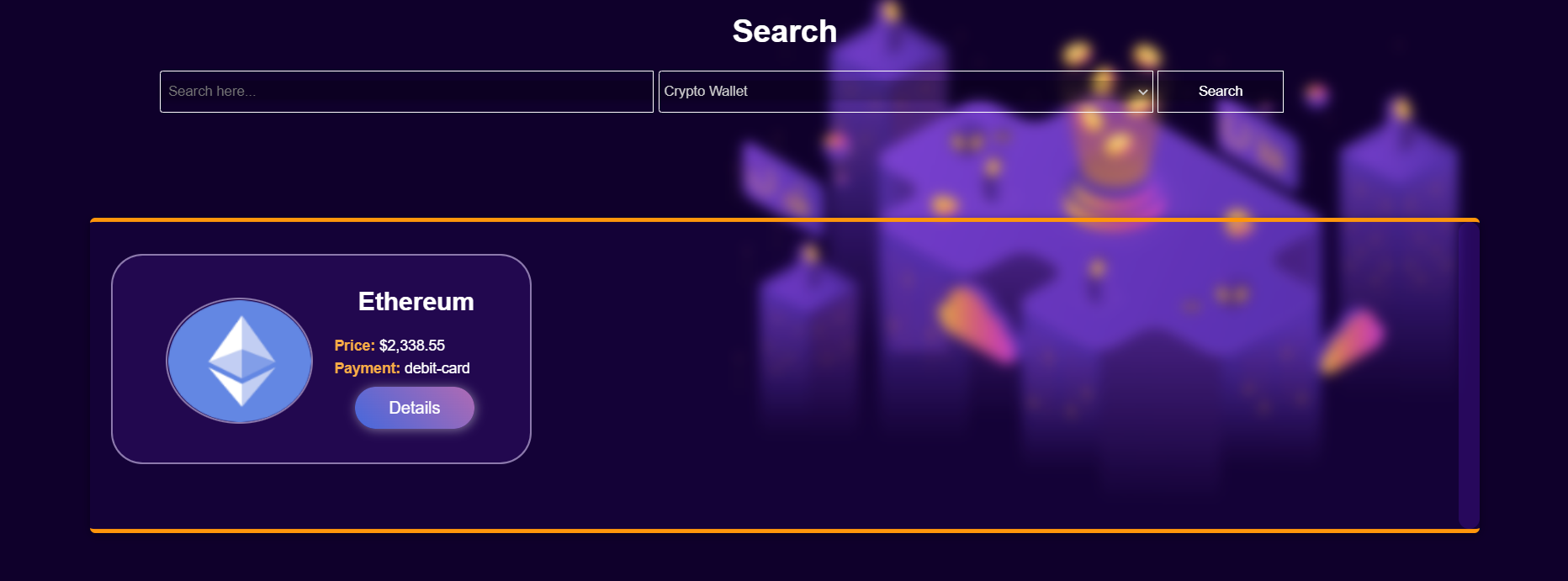


**\* Bonus – Search (10 Pts)**

Search **for crypto**. When you enter the search page, first, you must see all the crypto offers. Filter all matches by name and payment method (case insensitive).



After clicking on the **[Search]** button, and If there are any matching matches from the search, show each.



If there are **no** search matches, display:



## Submitting Your Solution

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









*GOOD LUCK! 😊*