

## Workshop: Cubicle – Part 3

"Cubicle" is a place, where you can browse some of the most popular Rubik cubes in the world and add some new cubes that you have discovered.

If you missed the first two parts of this workshop, make sure you complete them before you continue because all parts of this workshop are related to each other.

### Main Task

Now it's time to implement **user service** in your app, so people can **register**, **login** and **logout**. And each cube can be **edited** or **deleted**. Some of the functionality should **require authentication** such as (create a cube, create accessory) and **authorization** (such as edit and delete). Also, all **routes** should be **protected**!

### Installing Dependencies

You should install a few more packages which you will use. They are:

1. [jsonwebtoken](#) - allows you to decode, verify and generate JWT
2. [bcrypt](#) - a library to help you hash passwords
3. [cookie-parser](#) - parse **cookie header** and populate **req.cookies** with an object keyed by the cookie names (if you choose to store the **jwt** as a **cookie**)

### Model

The **User** Model structure:

- **Id** - ObjectId
- **username** - string
- **password** - string (**hashed**) - Use **bcrypt** to **hash** and **compare** the password

Make sure, when you successfully create a new user into the database, you generate a **jsonwebtoken** and use it later for **authentication** and **authorization**.

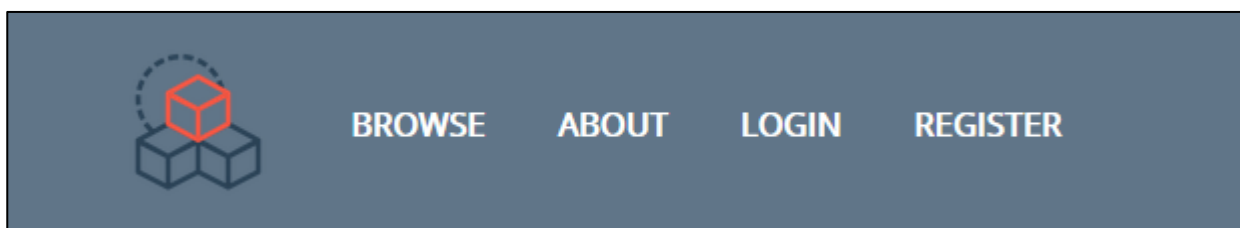
Also, you have to add property on **Cube Model**, which is **creatorId** (type: **String** and it's **required**), so you can keep tracking every cube's creator.

### Routes Protection

Make sure the **anonymous** (guest) users **can't reach** the functionality which requires authentication, such as creating a cube view. And **already logged-in** users have generated and stored **jwt**, **can** see the correct navigation, and **can't reach** the login and register form. If some of these scenarios happen, make sure the current user is redirected to the home page.

### Authentication

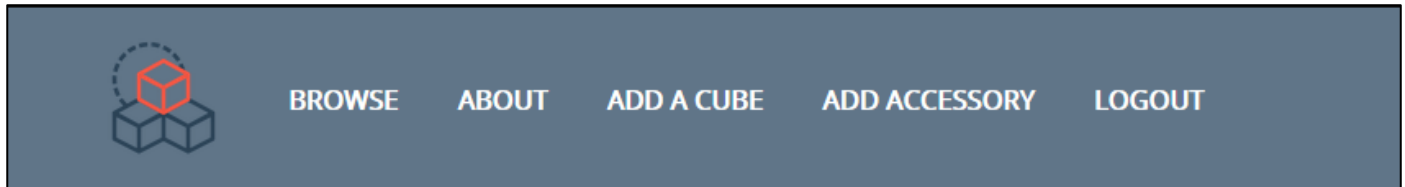
**Guest** users can **see** and **access** the following URLs:



- Home page (Browse)
- About page
- Login page
- Register page
- Cube details page

And **can't access** and **see** everyone else...

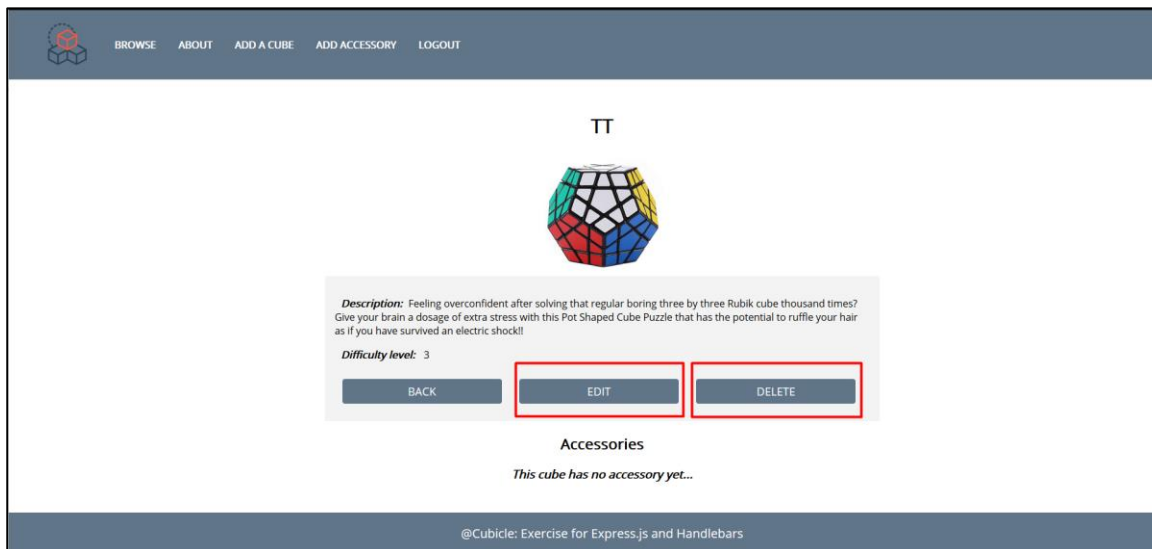
**Logged in** users can **see** and **access** the following URLs:



- Home page (Browse)
- About page
- Add cube
- Add accessory
- Logout
- Cube details page
- Cube accessories page
- Edit Cube page
- Delete Cube page

## Authorization

Only **authorized** users should see the **[Edit]** and **[Delete]** buttons and if the **currently logged-in user is the creator of this cube**. Otherwise, they should be **hidden**.



## Additional Pages

You should implement **4** new routes:


- **/login** - should render the login form
- **/register** - should render the register form
- **/edit** - should render the edit form

- **/delete** – should render the delete form

Make sure when you access **/edit** and **/delete** routes, they show the current cube information.


Use the provided [RESOURCES](#) to create the additional templates using Handlebars (Use **username: student**, **password: student** credentials to do that). Identify the dynamic parts and use appropriate syntax for interpolating and rendering the application context. Replace the old **CSS** file with the given one.

## Login Page




The screenshot shows a web application interface for a login page. At the top, there is a dark blue header bar containing a logo on the left and navigation links: BROWSE, ABOUT, LOGIN, and REGISTER. The main content area is white and features the title "LOGIN FORM" centered at the top. Below the title, there are two input fields: "Username" and "Password", each with a dark blue label on the left and a white input box. A dark blue button labeled "LOGIN" is positioned below the password field. At the bottom of the page, there is a dark blue footer bar with the text "@Cubicle: Exercise for Express.js and Handlebars" centered.

## Register Page



The screenshot shows a web application interface for a register page. At the top, there is a dark blue header bar containing a logo on the left and navigation links: BROWSE, ABOUT, LOGIN, and REGISTER. The main content area is white and features the title "REGISTER FORM" centered at the top. Below the title, there are three input fields: "Username", "Password", and "Re-Password", each with a dark blue label on the left and a white input box. A dark blue button labeled "REGISTER" is positioned below the "Re-Password" field. At the bottom of the page, there is a dark blue footer bar with the text "@Cubicle: Exercise for Express.js and Handlebars" centered.

## Edit Cube Page

 BROWSE ABOUT ADD A CUBE ADD ACCESSORY LOGOUT

### EDIT CUBE

Name

Pot Shaped Cube Puzzle

Description

Feeling overconfident after solving that regular boring three by three Rubik cube thousand times? Give your brain a dosage of extra stress with this Pot Shaped Cube Puzzle that has the potential to ruffle your

ImageUrl

https://thingsidesire.com/wp-content/uploads/2018/06/Pot-Shaped-Cube-Puzzle1.jpg


Difficulty

3 - Medium (Standard 3x3)

EDIT

@Cubicle: Exercise for Express.js and Handlebars

## Delete Cube Page

 BROWSE ABOUT ADD A CUBE ADD ACCESSORY LOGOUT

### DELETE CUBE

Name

Pot Shaped Cube Puzzle

Description

Rubik cube thousand times? Give your brain a dosage of extra stress with this Pot Shaped Cube Puzzle that has the potential to ruffle your hair as if you have survived an electric shock!!

ImageUrl

https://thingsidesire.com/wp-content/uploads/2018/06/Pot-Shaped-Cube-Puzzle1.jpg

Difficulty

1 - Very Easy

DELETE

@Cubicle: Exercise for Express.js and Handlebars

Good Luck!

