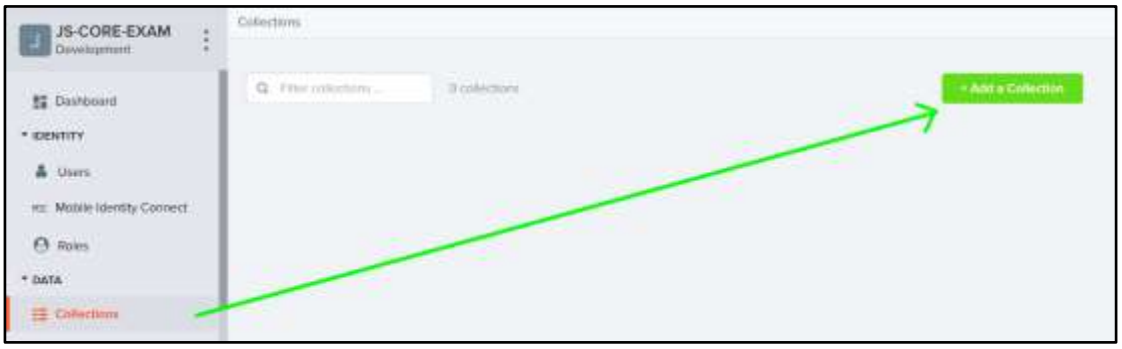
JS Applications Exam - Movies SPA

You are assigned to implement a **Web application** (SPA) using HTML5, JavaScript, AJAX, REST and JSON with cloud-based backend (Kinvey). Using libraries like **jQuery**, **Handlebars** and **Sammy** is allowed but is not obligatory. The app keeps **users** and **movies**. Guests should be able to **register** and **login**. Logged-in users should be able to view **all movies**, **create movies**, **buy tickets** for the movies, see **details** about a **movie** and **logout**. Logged-in users should also be able to **edit** or **delete** the movies **they have created**. There should also be a **section** where users can **see only the movies they have created**.

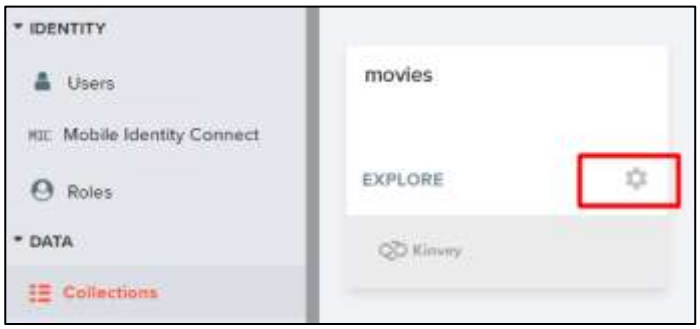
1. Create a Kinvey REST Service

Register at **Kinvey.com** and create an application to keep your data in the cloud.

Create a collection called **movies**. Each **movie** has a **`title`, `description`, `imageURL`, `tickets`** and **`genres` (array)**.



In order to be able to keep track of the **tickets** of each movie, you need to **allow all users to edit this collection**. So go to the **properties** of the collection:



Then go to **permissions** and edit them to look like this:

2. Test the Kinvey REST Services

Using **Postman** or other HTTP client tool (you can use Kinvey's built-in **API Console**), test the REST service endpoints:

User Registration (Sign Up)

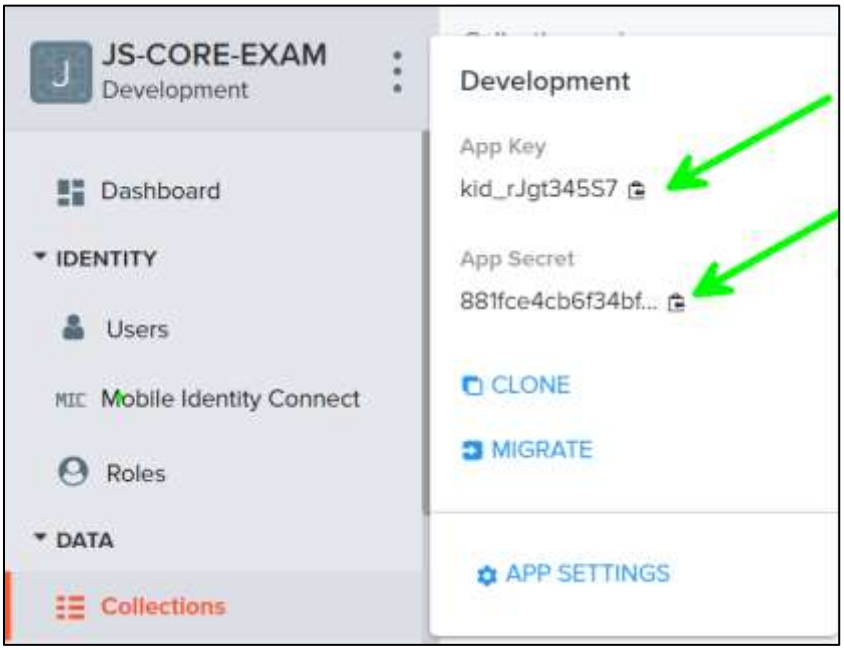
POST https://baas.kinvey.com/user/***app_id***	
Request headers	**Authorization:** Basic base64(app_id:app_secret) **Content-Type:** application/json

Request body	{
	"username": "testuser",
	"password": "testuserpass890"
	}

Response
201 Created
	{
	"_id": "59930c78a743e20c7d3fca77",
	"username": "testuser",
	"password": "testuserpass890"
	}

Error response **409 Conflict**	{ "error": "UserAlreadyExists", "description": "This username is already taken. Please retry your request with a different username", "debug": "" }
Error response **401 Unauthorized**	{ "error": "InvalidCredentials", "description": "Invalid credentials. Please retry your request with correct credentials", "debug": "" }

The request needs "**Basic**" authentication. Use the Kinvey **App Key** and Kinvey **App Secret** as credentials.



User Login

POST https://baas.kinvey.com/user/*app_id*/login	
Request headers	`Authorization: Basic base64(app_id:app_secret)` `Content-Type: application/json`

Request body	`{`
	`  "username": "testuser",`
	`  "password": "testuserpass890"`
	`}`

Response
200 OK
	`{`
	`  "_id": "59930c78a743e20c7d3fca77",`
	`  "username": "testuser"`
	`  "_kmd": {`
	`  "authtoken": "8e6471bc-3712-4cfb-b92e-50e62a0c80….Duj5fHdM/7XHIe6KdY="`
	`  …`
	`  },`
	`  …`
	`}`

Error response **401 Unauthorized**	`{ "error": "InvalidCredentials", "description": "Invalid credentials. Please retry your request with correct credentials", "debug": "" }`

Successful login returns an **authtoken** which is later used to authenticate the CRUD operations.

User Logout

POST https://baas.kinvey.com/user/***app_id***/_logout	
Request headers	`Authorization: Kinvey authtoken`
Response **204 No Content**	
Error response **401 Unauthorized**	`{ "error": "InvalidCredentials", "description": "Invalid credentials. Please retry your request with correct credentials", "debug": "" }`

To logout, you need to provide the **authtoken** given by login/register as "**Kinvey**" authorization header.

List All Movies

GET https://baas.kinvey.com/appdata/***app_id***/movies?query={}&sort={}

Request headers	`Authorization: Kinvey authtoken`

Response

200 OK


```
[
    {
        "_id": "5c9234b6cad7d87d9f873b90",
        "title": "Us (2019)",
        "imageUrl":"https://m.media-amazon.com/images/M/MV5BZTliNWJhM2YtNDc1MC00YTk1LWE2MGYtZmE4M2Y5ODdlNzQzXkEyXkFqcGdeQXVyMzY0MTE3NzU@._V1_UX140_CR0,0,140,209_AL_.jpg",
        "description": "A family's serenity turns to chaos when a group of doppelgängers begins to terrorize them.",
        "genres": [
            "Horror,Triller"
        ],
        "tickets": "52",
        "_acl": {
            "creator": "5c91f497dc781901a7e5745c"
        },
        "_kmd": {
            "lmt": "2019-03-21T08:31:38.656Z",
            "ect": "2019-03-20T12:40:22.352Z"
        }
    },
    {
        "_id": "5c92353231f05528ddabc64d",
        "title": "Triple Threat (2019)",
        "imageUrl": "https://m.media-amazon.com/images/M/MV5BYTY1MjRhYmYtZDg4Yy00ZWRiLWIwYzktZThkY2E0YjZlNjgxXkEyXkFqcGdeQXVyMTc3MjY3NTY@._V1_UY209_CR0,0,140,209_AL_.jpg",
        "description": "A hit contract is taken out on a billionaires daughter intent on bringing down a major crime syndicate. A down and out team of mercenaries must take on a group of professional assassins and stop them before they kill their target.",
        "genres": [
            "Action",
            "Thriller"
        ],
```

	"tickets": "22",
	"_acl": {
	"creator": "5c9234f4dc781901a7e87ce9"
	},
	"_kmd": {
	"lmt": "2019-03-20T13:47:12.419Z",
	"ect": "2019-03-20T12:42:26.360Z"
	}
	}
	]

Error response **401 Unauthorized**	{ "error": "InvalidCredentials", "description": "Invalid credentials. Please retry your request with correct credentials", "debug": "" }

Create Movie

POST https://baas.kinvey.com/appdata/***app_id***/movies	
Request headers	**Authorization: Kinvey authtoken** **Content-Type: application/json**
Request body	{"title": "test movie", "description": "test movie description", "imageUrl": "https://m.media-amazon.com/images/M/MV5BMTQzOTUyODMyOV5BMl5BanBnXkFtZTcwNzY2MzU1MQ@@._V1_.jpg", "genres": ["Thriller", "Drama"], "tickets": 50}

Response
201 Created
	{
	"title": "test movie",
	"description": "test movie desciption",
	"imageUrl":"https://m.media-amazon.com/images/M/MV5BMTQzOTUyODMyOV5BMl5BanBnXkFtZTcwNzY2MzU1MQ@@._V1_.jpg",
	"genres": [
	"Thriller",
	"Drama"
	],
	"tickets": 50,
	"_acl": {
	"creator": "5c91f497dc781901a7e5745c"
	},
	"_kmd": {

	"lmt": "2019-03-21T08:57:28.073Z", "ect": "2019-03-21T08:57:28.073Z" }, "_id": "5c9351f8b66da201acf6ffb8" }
Error response **401 Unauthorized**	{ "error": "InvalidCredentials", "description": "Invalid credentials. Please retry your request with correct credentials", "debug": "" }

Edit Movie

PUT https://baas.kinvey.com/appdata/***app_id***/movies/***movie_id***	
Request headers	Authorization: Kinvey authtoken Content-Type: application/json

Request body	{
	"title": "test movie",
	"description": "test movie desciption",
	"imageUrl":"https://m.media-amazon.com/images/M/MV5BMTQzOTUyODMyOV5BMl5BanBnXkFtZTcwNzY2MzU1MQ@@._V1_.jpg",
	"genres": [
	"Thriller",
	"Drama"
	],
	"tickets": 49
	}

Response **200 Ok**	{ "title": "test movie", "description": "test movie desciption", "imageUrl":"https://m.media-amazon.com/images/M/MV5BMTQzOTUyODMyOV5BMl5BanBnXkFtZTcwNzY2MzU1MQ@@._V1_.jpg", "genres": ["Thriller", "Drama"], "tickets": 49,

	<pre>"_id": "5c9351f8b66da201acf6ffb8",
	"_acl": {
	"creator": "5c91f497dc781901a7e5745c"
	},
	"_kmd": {
	"lmt": "2019-03-21T09:02:01.340Z",
	"ect": "2019-03-21T08:57:28.073Z"
	}
	}</pre>

Error response **401 Unauthorized**	<pre>{ "error": "InvalidCredentials", "description": "Invalid credentials. Please retry your request with correct credentials", "debug": "" }</pre>

Delete Movie

DELETE https://baas.kinvey.com/appdata/*app_id*/movies/**movie_*id***	
Request headers	<pre>Authorization: Kinvey authtoken</pre>

Response
200 OK
	<pre>{
	"count": 1
	}</pre>

Error response **404 Not Found**	<pre>{ "error": "EntityNotFound", "description": "This entity not found in the collection", "debug": "" }</pre>
Error response **401 Unauthorized**	<pre>{ "error": "InvalidCredentials", "description": "Invalid credentials. Please retry your request with correct credentials", "debug": "" }</pre>

Buy Tickets

PUT https://baas.kinvey.com/appdata/*app_id*/movies/**movie_*id***	

Request headers	<pre>Authorization: Kinvey authtoken
	Content-Type: application/json</pre>

Request body	<pre>{
	"title": "test movie",
	"description": "test movie description",
	"imageUrl": "https://m.media-</pre>

	amazon.com/images/M/MV5BMTQzOTUyODMyOV5BMl5BanBnXkFtZTcwNzY2MzU1MQ@@._V1_.jpg", "genres": ["Thriller", "Drama"], "tickets": 49 }

Response
200 Ok
	{
	"title": "test movie",
	"description": "test movie desciption",
	"imageUrl":"https://m.media-amazon.com/images/M/MV5BMTQzOTUyODMyOV5BMl5BanBnXkFtZTcwNzY2MzU1MQ@@._V1_.jpg",
	"genres": [
	"Thriller",
	"Drama"
	],
	"tickets": 49,
	"_id": "5c9351f8b66da201acf6ffb8",
	"_acl": {
	"creator": "5c91f497dc781901a7e5745c"
	},
	"_kmd": {
	"lmt": "2019-03-21T09:02:01.340Z",
	"ect": "2019-03-21T08:57:28.073Z"
	}
	}

Error response **401 Unauthorized**	{ "error": "InvalidCredentials", "description": "Invalid credentials. Please retry your request with correct credentials", "debug": "" }

My Movies

GET https://baas.kinvey.com/appdata/***app_id***/movies?query={"_acl.creator":"${***user_id***}"}	
Request headers	**Authorization: Kinvey authtoken**
Response	[

200 OK	{
	"_id": "5c9234b6cad7d87d9f873b90",
	"title": "Us (2019)",
	"imageUrl": "https://m.media-amazon.com/images/M/MV5BZTliNWJhM2YtNDc1MC00YTk1LWE2MGYtZmE4M2Y5ODdlNzQzXkEyXkFqcGdeQXVyMzY0MTE3NzU@._V1_UX140_CR0,0,140,209_AL_.jpg",
	"description": "A family's serenity turns to chaos when a group of doppelgängers begins to terrorize them.",
	"genres": [
	"Horror,Triller"
	],
	"tickets": "52",
	"_acl": {
	"creator": "5c91f497dc781901a7e5745c"
	},
	"_kmd": {
	"lmt": "2019-03-21T08:31:38.656Z",
	"ect": "2019-03-20T12:40:22.352Z"
	}
	}
	…
	]

Error response **401 Unauthorized**	{ "error": "InvalidCredentials", "description": "Invalid credentials. Please retry your request with correct credentials", "debug": "" }

3. Movies - HTML and CSS

You have been given the web design of the application as **HTML** + **CSS** files.

- Initially all views and forms are shown by the HTML. Your application may **hide**/**show elements** by CSS (**`display: none`**) or **delete**/**reattach** from and to the DOM all unneeded elements, or just display the views it needs to display.
- You may render the views/forms/components with **jQuery** or **Handlebars**.

Important: Don't change the elements' **class names** and **ids**. Don't rename form fields/link names/ids. You are **allowed** to add **data attributes** to any elements. You may modify **`href`** attributes of links and add **`action`**/**`method`** attributes to **forms**, to allow the use of a routing library.

4. Movies - Client-Side Web Application

Design and **implement** a client-side front-end app (SPA) for managing **movies**. Implement the functionality described below.

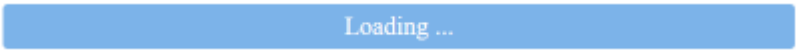
Notifications (5 pts)

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

- In case of successful action an **informational (green) notification message** should be shown, which disappears automatically after 3 seconds or manually when the user clicks it.

- In case of **error**, an **error notification message** (red) should be shown which disappears on user click.

- During the AJAX calls a **loading notification message (blue)** should be shown. It should disappear automatically as soon as the AJAX call is completed.



- **NOTE: You get all 5 point if the notifications contain the right messages.**

Navigation Bar (5 pts)

Navigation links should correctly change the current page (view). **The navbar should be displayed in every view!**

- Clicking on the links in the **NavBar** should display the view behind the link (views are represented as sections in the HTML code).
- Your application may **hide/show elements** by CSS (`display: none`) or **delete/reattach** from and to the DOM all unneeded elements, or just display the views it needs to display.
- The links [`Cinema`], [`Add Movie`], [`My Movies`], [`Logout`], and the user caption "`Welcome, {username}!`" should be visible **only** for logged-in users.

- Anonymous users have access only to the following links: [`Home`], [`Login`] and [`Register`].

Home Screen (5 pts)

The initial page (view) should display:

- **NavBar** with the following links [`Home`], [`Login`], [`Register`]
- The **image** from the resources (`background.jpg`)
- The **Footer**

Register User (5 pts)

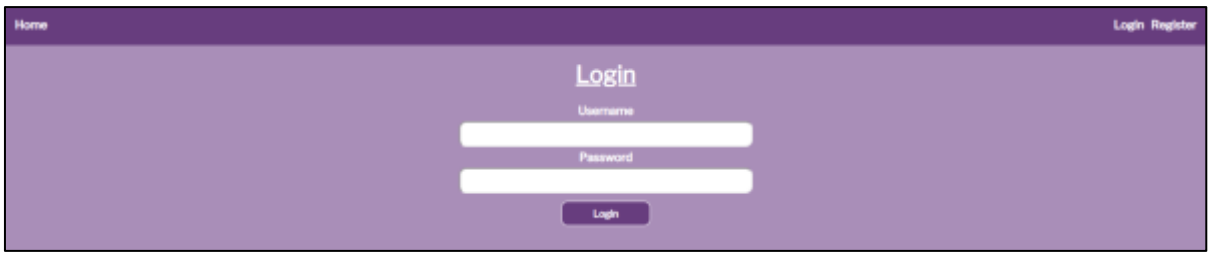
By given **username** and **password** the app should register a new user in the system.

- You should make the following validations:
  - The **username** should be **at least 3 characters** long
  - The **password** should be **at least 6 characters** long
  - The **repeat password** should be **equal to the password**
- After a **successful registration**, a notification message **`"User registration successful."`** should be **displayed** and the user should be **logged-in**.
  - **Home page** should be **displayed** with the **User navbar**.
- In case of **error** (eg. **duplicate** username, **non-matching** passwords), an appropriate **error message** should be **displayed** and the client should be able to **try** to register again.
- Keep the **user session data** in the browser's **session storage**.

Login User (5 pts)

By given **username** and **password** the app should be able to login an existing user.

- After a **successful login**, a notification message **`"Login successful."`** should be displayed and the user should be **logged-in**.
  - **Home page** should be **displayed** with the **User navbar**.
- In case of **error**, an appropriate **error message** should be **displayed** and the user should be able to fill the **login form** again.
- Keep the **user session data** in the browser's **session storage**.



Logout (5 pts)

Logged-in users should be able to **logout** from the app.

- After a **successful** logout, a **notification** message **`"Logout successful."`** should be displayed and the user should be **logged-out**.
    - **`Login page`** should be **displayed** with the **user navbar**.
- The **Logout REST service** at the back-end should be obligatory **called** at logout.
- All local information in the browser (**user session data**) about the current user should be **deleted**.

Cinema (30 pts)

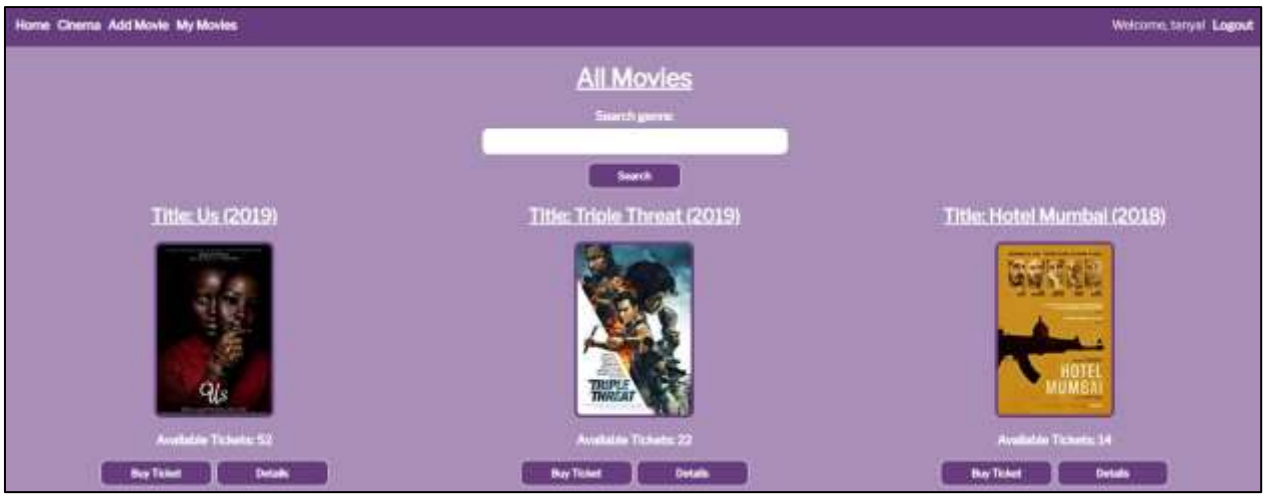
Logged-in users should be able to **view all movies**.

Clicking the [**`Cinema`**] **link** in the **NavBar** should **display** the **`Cinema page`**.

The **`Cinema page`** should **list all** of the **movies**, in a **formatted** and **ordered** manner:

- The **movies** should be listed in the **format** as shown in the web design (see the screenshot below).
- The **movies** must be **sorted** by the **available tickets** in **descending**.
- Each **movie** has **title**, **image, available tickets, buttons:** [**Buy `Ticket`**] **and** [**`Details`**].

- The **`Cinema page`** should look like this:



Add Movie (10 pts)

Logged-in users should be able to **Add** movies.

Clicking the [**Add Movie**] **link** in the **NavBar** should **display** the **Add Movie page**.

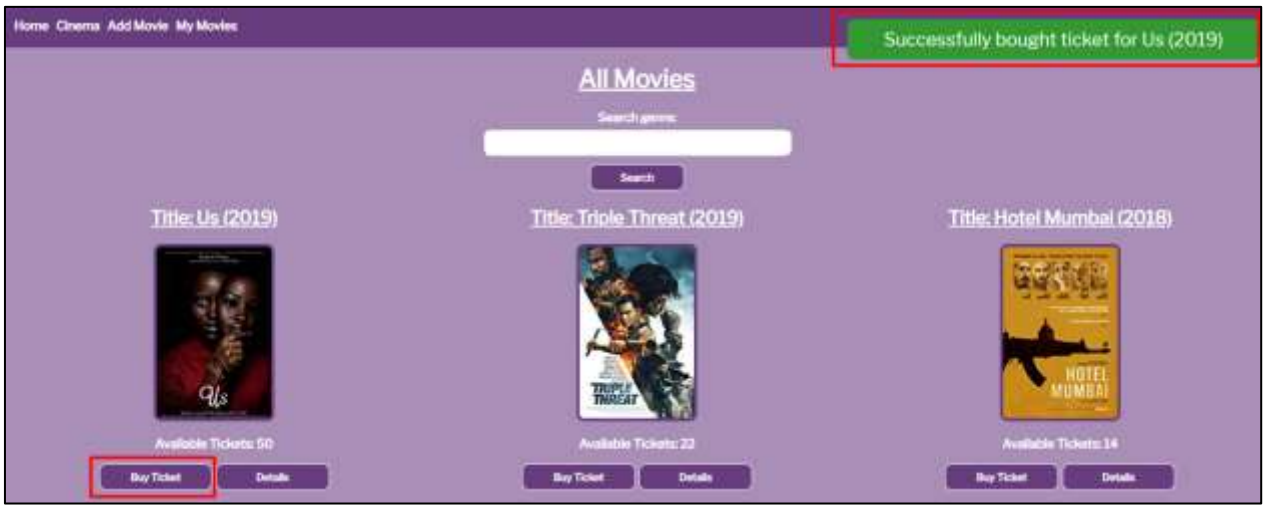
- The form should contain the following validations:
  - The **title** should be **at least 6 characters** long.
  - The **description** should be **at least 10 characters** long.
  - The **image** should start with **"http://"** or **"https://"**.
  - The **available tickets** should be a **number**.
  - The **genres** must be separated by a **single space**.
- After a **successful movie creation**, a notification message **"Movie created successfully."** should be **displayed** and the **Home page** should be shown.
- The inputs fields in the form **should be cleared.**
- The newly created movie should be stored in the Kinvey collection "**movies**".

Buy Ticket (5 pts)

Logged-in users should be able to **Buy tickets** for movies.

Clicking the [**Buy `Ticket`**] **link** of a **particular movie** should **decrease** its **available tickets**. If there are **no available tickets**, an appropriate **notification** should be **displayed** and the **available tickets** should **not be changed**.
Users can **buy multiple tickets** for **one movie.**

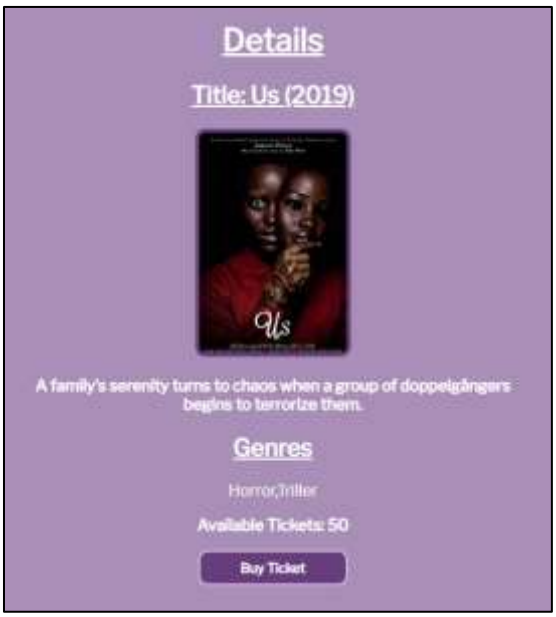
- After **successfully buying** a **ticket**, a notification message **`"Successfully bought ticket for {movie name}!"`** should be displayed.



Movie Details (5 pts)

Logged-in users should be able to **view details** about movies.

Clicking the [**`Details`**] **link** in of a **particular movie** should **display** the **`Movie Details` page**.



- From this **view**, the user should also be able to **buy tickets** - [**Buy `Tickets`**] link.

My Movies (10 pts)

Logged-in users should be able to **view their** movies.

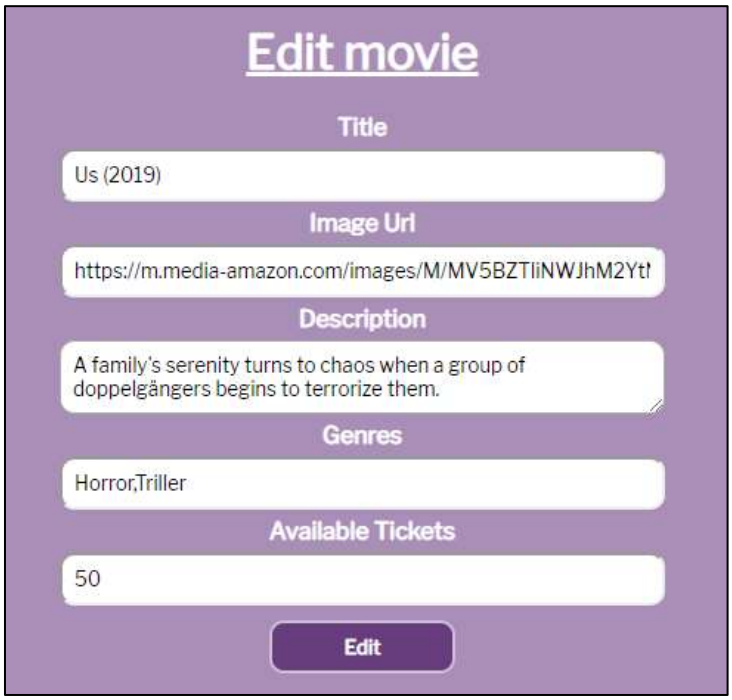
Clicking the [**My `Movies`**] **link** in the **NavBar** should **display** the **My `Movies` page**.

- The **movies** should be listed like in the **cinema** section, except the user should see only the **movies** that he created and each movie should have [**`Edit`**] and [**`Delete`**] buttons, which will **show** the **corresponding pages**.

Edit Movie (5 pts)

Logged-in users should be able to **edit their** movies.

Clicking the [**`Edit`**] **link** of a **particular movie** on the **`My Movies` page** should **display** the **`Edit Movie` page**:

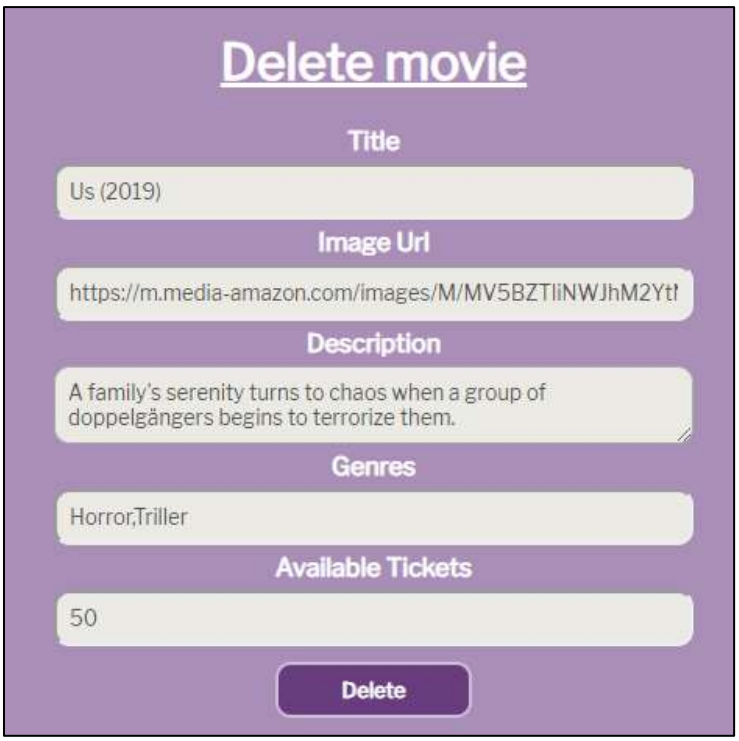


- After a **successful edit**, an appropriate notification should be **displayed**!

Delete Movie (5 pts)

Logged-in users should be able to **delete their** movies.

Clicking the [**`Delete`**] **link** of a **particular movie** on the **`My Movies` page** should **display** the **`Delete Movie` page**:

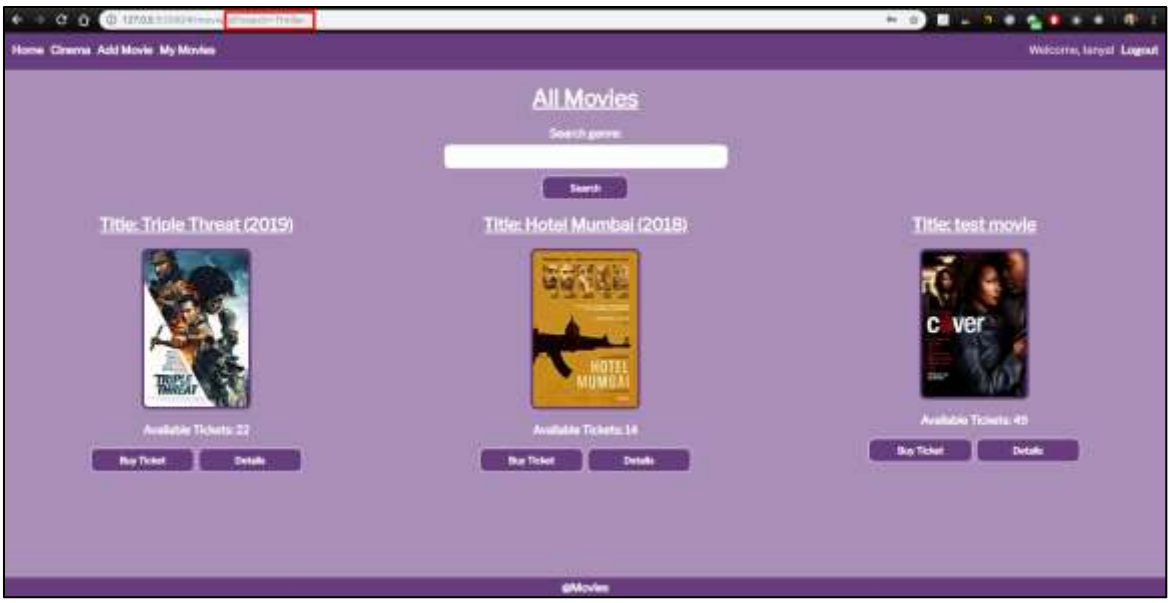


- The **input fields** should be **disabled**.
- After **successful movie delete** a notification message **`"Movie removed successfully!"`** should be displayed and the **Home page** should be **shown**.

(BONUS) Search: (5 pts)

Logged-in users should be able to **search** for movies, by typing some **genre** (**case sensitive**) on the **search field** of the **`Cinema page`**.

Clicking the [**`Search`**] link should **display** the **same view** (**`Cinema page`**), but **only** with the **movies** which **contain** the **given genre**:



5. Submitting Your Solution

Submit a ZIP file with your project folder. Exclude the **`node_modules`** folder. Upload the archive to the Judge.