|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Name | HTTP Method | Path Name | Purpose |
| Assignment 1 | | | | |
| 1 | Home Page | GET | /  (server.js) | Sends the html file. Showing button linking the tasks of both student, team name |
| 2 | Add category page | GET | /category/33254834/add-category  (category-route.js) | Sends the add-category html file. Showing form for user to key in the category details to add them to the database for further access |
| 3 | Add event category post | POST | /category/33254834/add\_cat  (category-route.js) | Get the category input and create a new category to push the data to the backend.  Then redirect to the list-category html page |
| 4 | List event page | GET | /category/33254834/event-categories  (category-route.js) | Sends the event-categories html page, listing all the categories available in the database |
| 5 | Get the keyword for the listing of event category page | GET | /category/33254834/search-category  (category-route.js) | Get the keyword from the URL query and go to the page that list all the filtered event categories that matches |
| 6 | Post method for keyword input | POST | /category/33254834/search-category-detail  (category-route.js) | Get the keyword from the input form and list all filtered event categories that matches |
| 7 | List event category that matching keyword | GET | /category/33254834/event-categories?keyword=  (category-route.js) | Sends the event-categories html page, listing all the categories that have the keyword matching the keyword from query |
| 8 | Get event id for event detail searching | GET | /category/33254834/search-event  (category-route.js) | Get the event id from the form to link to the page that shows the event details |
| 9 | Show event detail page | GET | /category/33254834/event/:eventID  (category-route.js) | Show the event details by passing the parameter in the URL |
| 10 | Search event post | POST | /category/33254834/search-event  (category-route.js) | Get the event id to search from the form and direct to the page that shows the event detail |
| 11 | Delete category page | GET | /category/33254834/delete-category  (category-route.js) | Go to the delete category page that gets the input of the category id that is required for deletion then go to the list category page |
| 12 | Delete category post | POST | /category/33254834/delete\_cat  (category-route.js) | Get the delete id from the form |
| 13 | Add Event Page | GET | /John/event/add  (event.js) | Renders the “add-event.html” file where the user can submit event details through a form. |
| 14 | Add event post | POST | /add-event-post  (event.js) | Creates a new Event object based on the form input, and pushes it onto the backend. Then redirects to the List all events page. |
| 15 | List All Events page | GET | /John/events  (event.js) | Renders the “event-list.html” file, which lists the currently existing events available in the database. |
| 16 | List All Sold Out Events Page | GET | /John/sold-out-events  (event.js) | Renders the "event-sold-out.html" file, which lists the sold out events from the database. |
| 17 | Category Detail Input page | GET | /John/category/  (event.js) | Renders the "category-detail-input.html" file, where users can input a category ID to be searched for. |
| 18 | Category Detail Page | GET | /John/category/:id  (event.js) | Parses the ‘id’ information from the query, looks through the category database for details of a category with that specific ID, and shows category details along with events that have matching category ID’s. |
| 19 | Delete Event by ID input page | GET | /John/event/delete  (event.js) | Renders the "delete-event.html” file, where users can enter an event ID to be deleted. |
| 20 | Delete Event by ID page | GET | /John/event/remove/?id=  (event.js) | Parses the id from the query, then filters the event database to exclude any events with a matching id, then redirects to list all events page. |
| 21 | 404 Not found page | GET | \*  (server.js) | Everything else accept intended link will bring to 404 not found page |
| Assignment 2 | | | | |
| 22 | Add Category API | API POST | /category/33254834/api/v1/add-categories  (category-api-route.js) | Add Category to Category database by using http post request and pass JSON object |
| 23 | List Category API | API GET | /category/33254834/api/v1/list-categories  (category-api-route.js) | List down all Category in database using http get request |
| 24 | Delete Category API | API DELETE | /category/33254834/api/v1/delete-category  (category-api-route.js) | Delete the category from the database by matching category id |
| 25 | Update Category API | API PUT | /category/33254834/api/v1/update-category  (category-api-route.js) | Update the category name and description by matching category id |
| 26 | Add Event API | API POST | John/api/v1/add-event  (event-api.js) | Add new event to the database and insert ID of the new event to the 'eventList' array field in all the categories under this event. |
| 27 | List All Events API | API GET | John/api/v1/events  (event-api.js) | List all events and the details for their categories in a JSON format. |
| 28 | Delete event by ID API | API DELETE | John/api/v1/delete-event  (event-api.js) | Deletes event based on inputted eventID. The ID of the deleted event is removed from the 'eventList' array in all the categories that are listed in the 'categoryList'. |
| 29 | Update event API | API PUT | John/api/v1/update-event  (event-api.js) | Updates event name and capacity by ID. Returns a status message to confirm the update or if the ID is not found. |
|  |  |  |  |  |

USER MANUAL

1. Run server.js on local host with port number 8080. <http://localhost:8080/>
   1. This direct to the main page of the app
   2. Containing the team name at the end of the page
   3. Having button to navigate around to each task
2. Add category
   1. Add details about the category into the input then click the submit button to add category
   2. After valid category is added, it will navigate to the list all category page
3. List all category
   1. List all category available in the application
   2. Contains the view button linked to the page that shows the category detail
4. List category by keyword
   1. Can filter category by keyword by using query (/?keyword=)
   2. Can also filter category keyword using search bar
5. Show event details
   1. Enter the event id in the search event page (<http://localhost:8080/category/33254834/search-event>)
   2. Or pass in event id as parameter at the end of the url (<http://localhost:8080/category/33254834/event>)
6. Delete category by id
   1. Enter the id of category to delete in the input page
   2. If the id is valid, the category will be removed from database
   3. If the id is invalid, nothing will be removed
   4. Both will end up in the list all category page
7. Add event
8. List all event
9. List sold out event
10. Category detail page
11. Delete event by id

* Every page regarding to student #1 task has a navigation bar that could click on to return to home page
* Every page regarding to student # 2 task has a home icon to click on to return home page
* There is a 404-not-found page that shown when the pathname is invalid
  + (eg. http://localhost:8080/not-valid)