HTTP AND OBSERVABLES

IIHT

Time To Complete: 10 to 12 hr

CONTENTS

1 Project Abstract

Managing data interaction between a client and a server is crucial in modern web development. This assignment focuses on implementing **HTTP Requests** and handling **Observables** in Angular 15 to perform CRUD operations.

The objective is to build a **Post Management Single Page Application (SPA)**, where users can view, add, edit, and delete posts. The application should demonstrate the use of Angular's `HttpClient` service combined with observables for asynchronous data handling.

2 Problem Statement

You are tasked with developing a **Post Management SPA** using Angular 15.

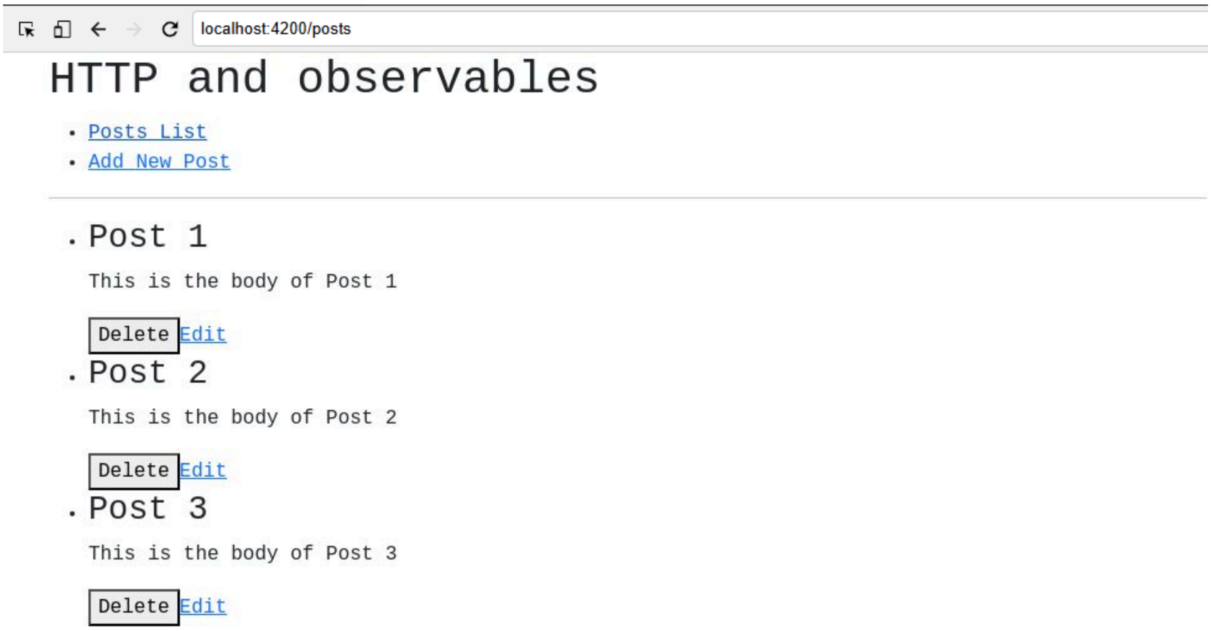
The application should:

- Fetch and display a list of posts.
- Allow the user to add a new post.
- Allow editing of existing posts.
- Provide the ability to delete a post.
- Use Angular's `HttpClient` service and observables to manage all HTTP requests.
- Handle errors gracefully and provide feedback.

3 Proposed HTTP and observables Application Wireframe

UI needs improvisation and modification as per given use case and to make test cases passed.

3.1 ScreenShots



localhost:4200/posts

HTTP and observables

- Posts List
- Add New Post

- Post 1

  This is the body of Post 1

  Delete Edit
- Post 2

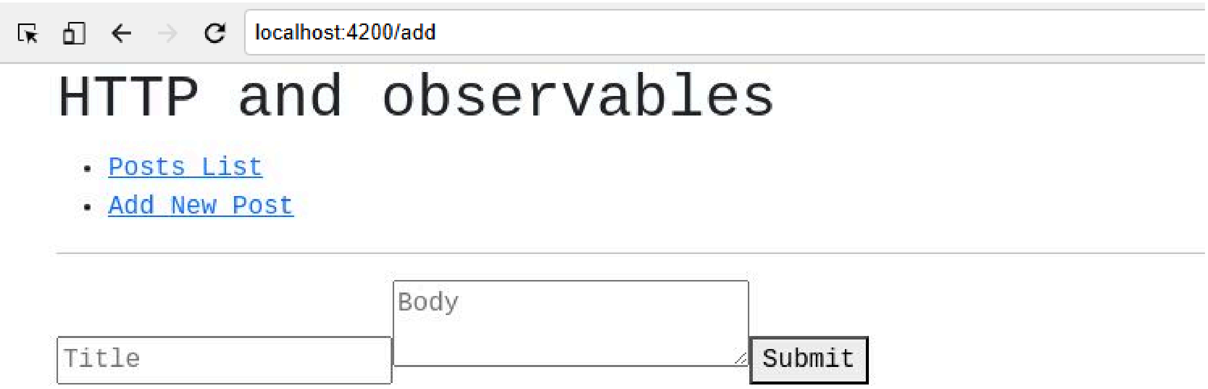
  This is the body of Post 2

  Delete Edit
- Post 3

  This is the body of Post 3

  Delete Edit

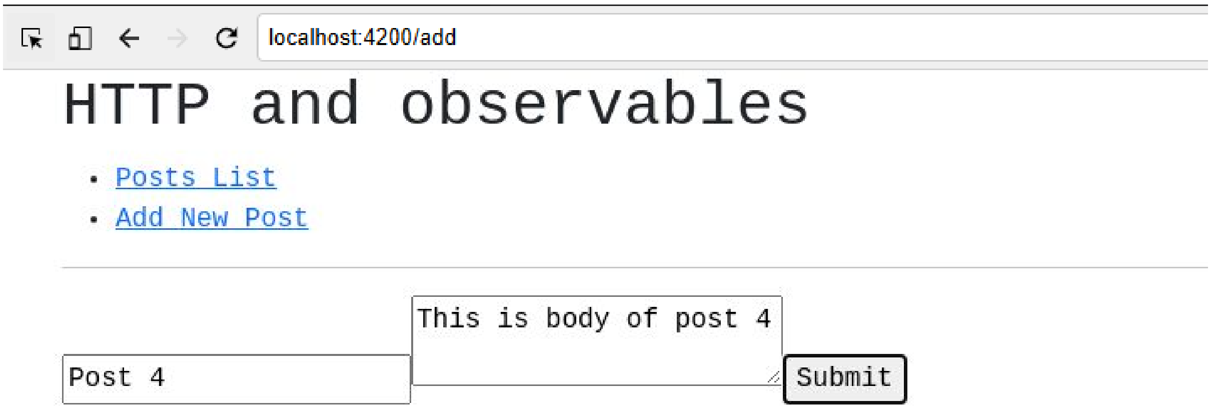
*****Add New Post*****



localhost:4200/add

HTTP and observables

- Posts List
- Add New Post



localhost:4200/add

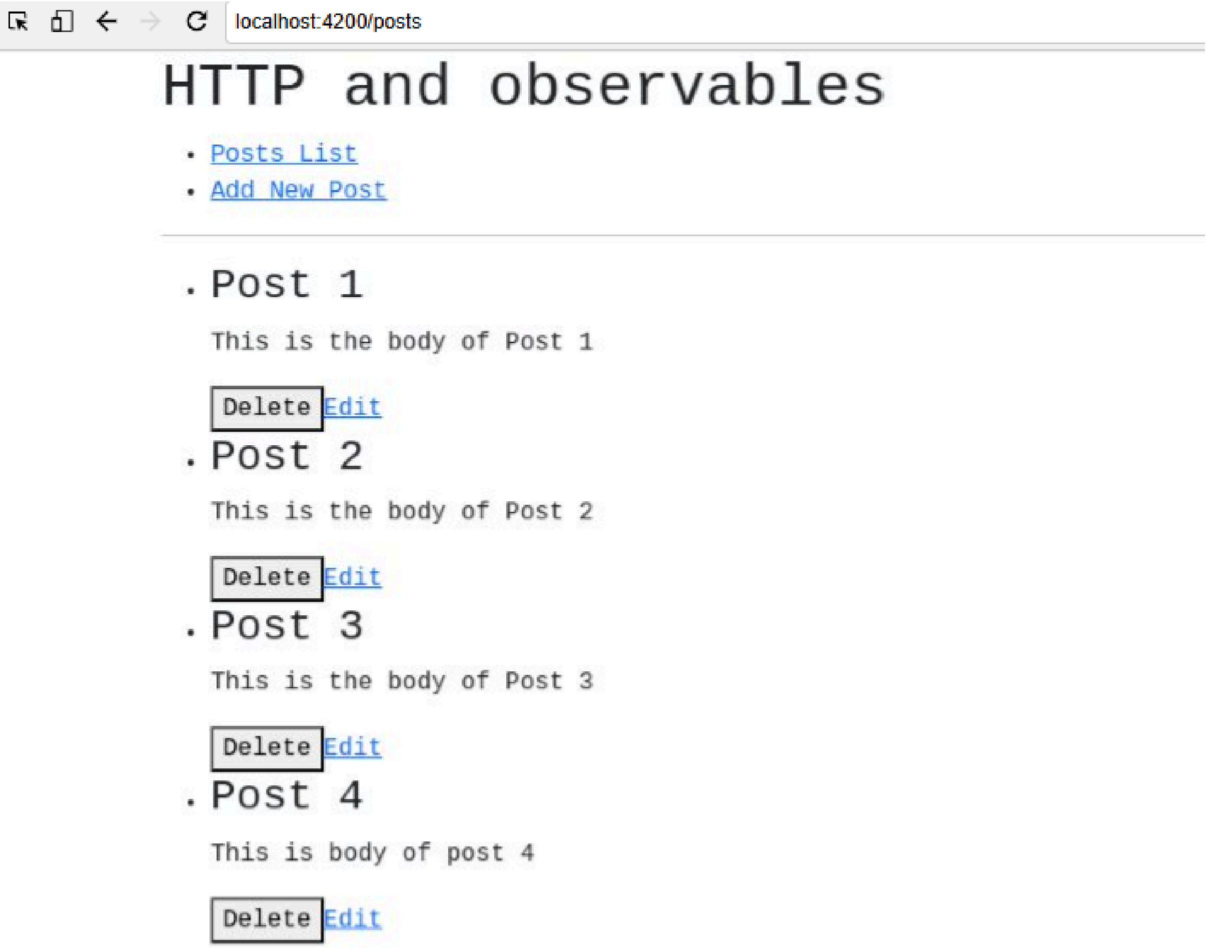
HTTP and observables

- Posts List
- Add New Post

Post 4

This is body of post 4

Added Post



localhost:4200/posts

HTTP and observables

- Posts List
- Add New Post

- Post 1

  This is the body of Post 1

  Delete Edit
- Post 2

  This is the body of Post 2

  Delete Edit
- Post 3

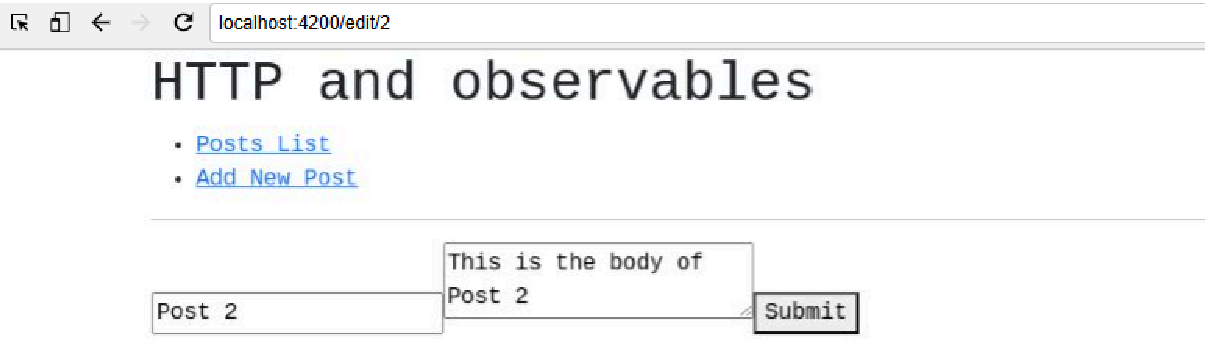
  This is the body of Post 3

  Delete Edit
- Post 4

  This is body of post 4

  Delete Edit

*****Edit Post*****

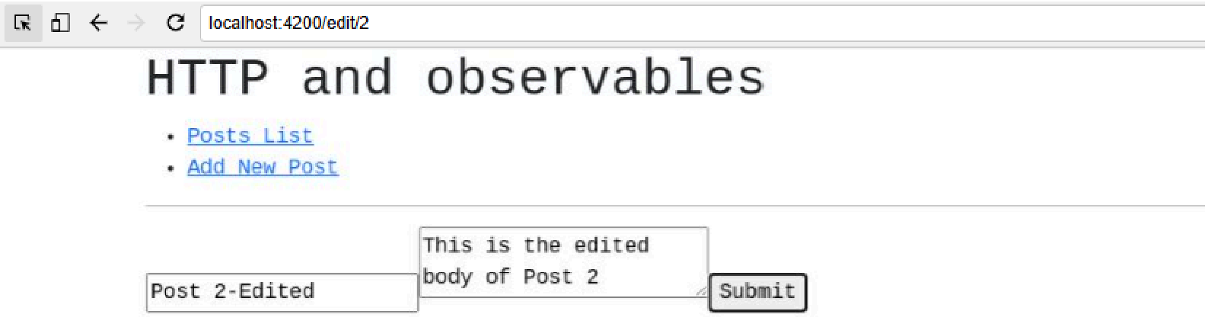

localhost:4200/edit/2

HTTP and observables

- Posts List
- Add New Post

Post 2

This is the body of Post 2


localhost:4200/edit/2

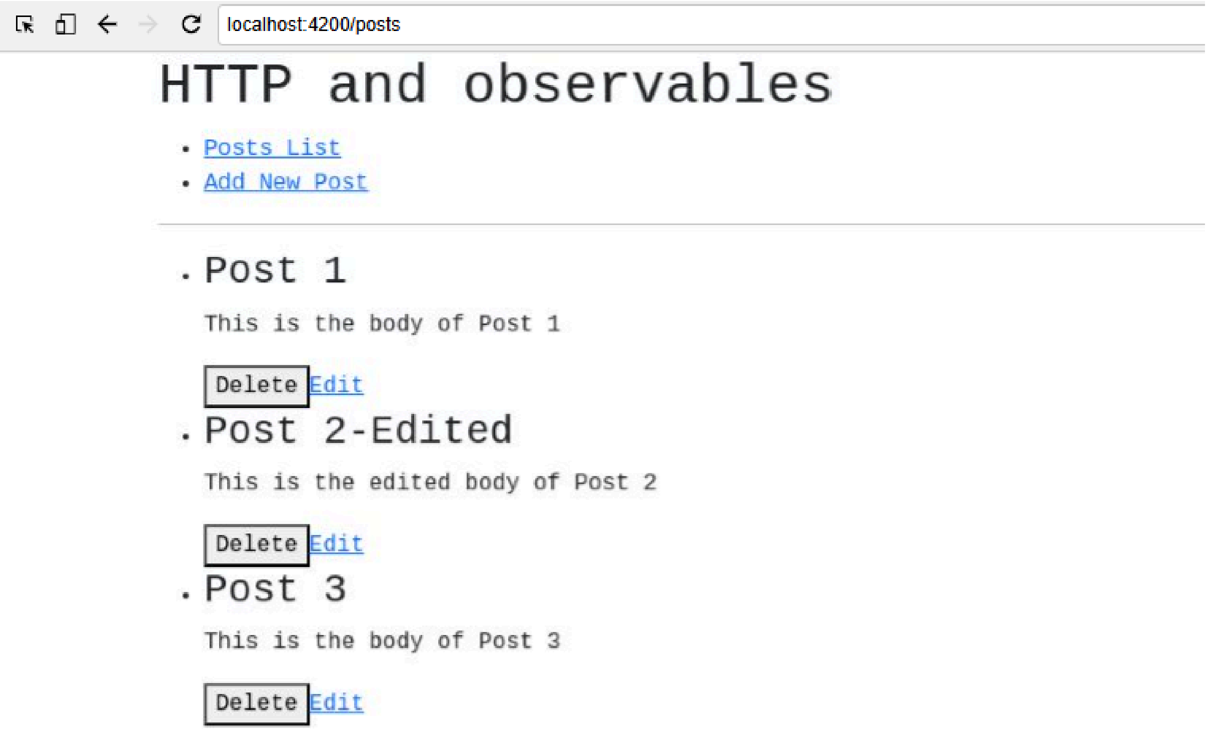
HTTP and observables

- Posts List
- Add New Post

Post 2-Edited

This is the edited body of Post 2

*****Delete Post*****



HTTP and observables

- Posts List
- Add New Post

- Post 1

  This is the body of Post 1

- Post 2-Edited

  This is the edited body of Post 2

- Post 3

  This is the body of Post 3

4 Business-Requirement:

As an application developer, develop the HTTP and observables (Single Page App) with below guidelines:

User Story #	User Story Name	User Story
US_01	Welcome Page	As a user I should be able to visit the welcome page as the default page. Acceptance criteria: **AppComponent:** 1. Should display the heading **"HTTP and observables"** in an **h1** tag. 2. Provide navigation links for **Posts List** as /posts and **Add New Post** as /add inside a **ul tag** using **li tags** with each link wrapped inside an **a tag** using `routerLink` directives. 3. Clicking on **Add New Post** should navigate to `/add`. Clicking on **Posts List** should navigate to `/posts`. 4. Include the `<router-outlet>` directive to render routed components. **PostListComponent:** 5. Should fetch and display the list of posts dynamically. 6. Display each post's **Title** in an **h3** tag and **Body** in a **p tag**, all inside **li tags**. 7. Provide a **Delete** button next to each post to remove it. 8. Provide an **Edit** link next to each post that navigates to `/edit/:id`. 9. Display error messages if any HTTP request fails. **PostFormComponent:** 10. For adding a new post: ● Display input fields for **Title** and **Body** using input and textarea elements. ● Provide a **Submit** button. 11. For editing an existing post: ● Pre-fill the fields with the selected post's data. ● Allow editing of Title and Body, and save changes on submit. 12. Show appropriate error message if the form submission fails. **HttpService:** 13. Create a **`HttpService`** to manage HTTP GET, POST, PUT, DELETE requests using Angular's **`HttpClient`**. **Get All Posts** ● Create a method to return the entire list of posts.

		● Return the posts wrapped inside an RxJS `Observable` using the `of()` function. ● Add error handling using the `catchError` operator. **Create a New Post** ● Accept a new post object (without an ID). ● Generate a new `id` by taking the length of the posts array and adding 1. ● Add the new post to the array. ● Return the newly created post wrapped in an `Observable`. ● Use `catchError` for error handling. **Update a Post by ID** ● Accept two parameters: `id` (number) and the updated post data. ● Find the post in the array using the given ID. ● If the post exists: ○ Merge the updated data into the existing post. ○ Return the updated post in an `Observable`. ● If the post is not found: ○ Return `null` **wrapped in an** `Observable`**.** ● **Use** `catchError` to handle any potential errors. **Delete a Post by ID** ● Accept the ID of the post to be deleted. ● Find the post using the ID. ● If found: ○ Remove it from the array. ○ Return the deleted post as an `Observable`. ● If not found: ○ Return `null` as an `Observable`. ● Include `catchError` for safety. **** Kindly refer to the screenshots for any clarifications. ****

5 Constraints

1. You should be able to press the “TAB” key and “SHIFT + TAB” to navigate from top field to bottom field and vice-versa.

6 Mandatory Assessment Guidelines

1. **All actions like build, compile, running application, running test cases will be through Command Terminal.**
2. **To open the command terminal the test takers, need to go to**

   Application menu (Three horizontal lines at left top) -> Terminal ->New Terminal.
3. **This editor Auto Saves the code.**
4. **These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.**
5. **This is a web-based application, to run the application on a browser, use the internal browser in the workspace. Click on the second last option on the left panel of IDE, you can find Browser Preview, where you can launch the application.**

   Note: The application will not run in the local browser
6. **You can follow series of command to setup Angular environment once you are in your project-name folder:**
   a. **npm install -> Will install all dependencies -> takes 10 to 15 min.**
   b. **npm run start -> To compile and deploy the project in browser. You can press the <Ctrl> key while clicking on localhost:4200 to open the project in the browser -> takes 2 to 3 min.**
   c. **npm run test -> to run all test cases. It is mandatory to run this command before submission of workspace -> takes 5 to 6 min.**
7. **Once you are done with development and ready with submission, you may navigate to the previous tab and submit the workspace. It is mandatory to click on "Submit Assessment" after you are done with code.**