clone the below git repository:

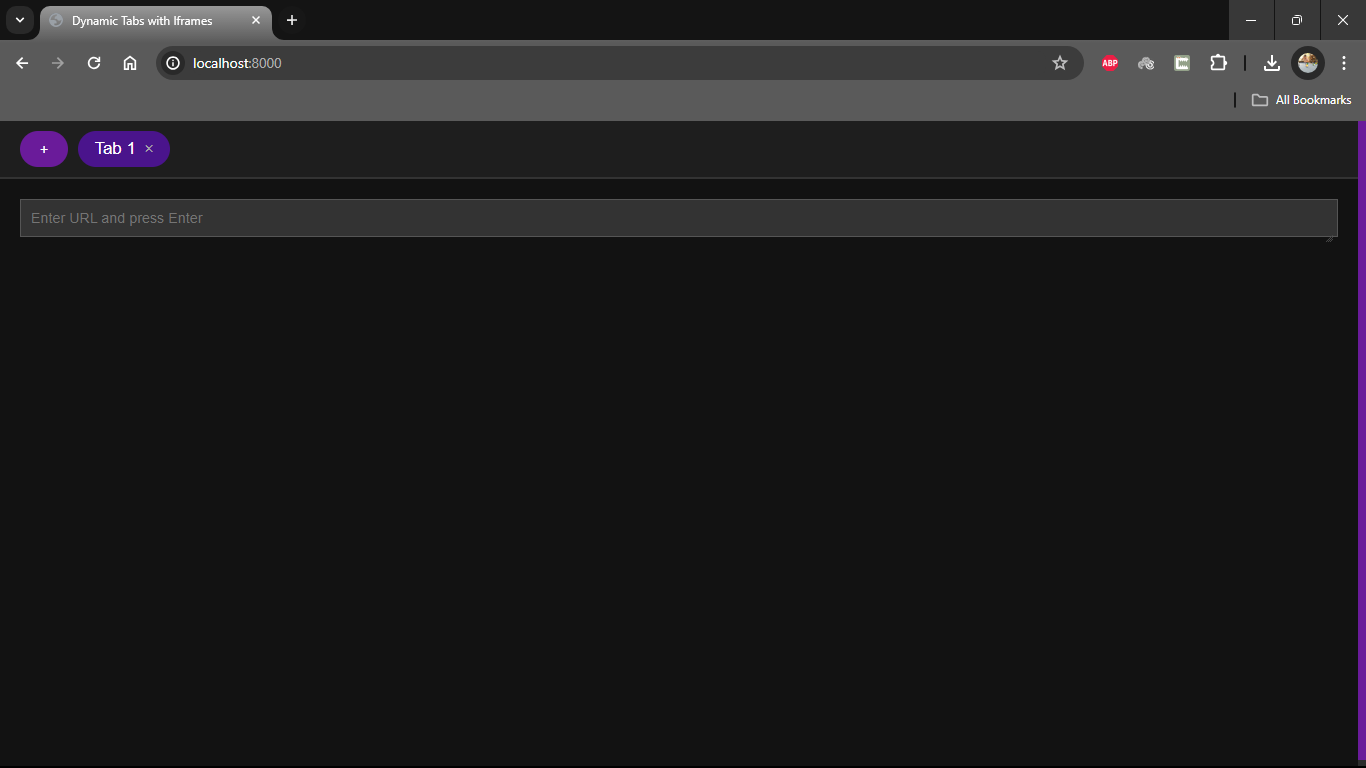
<https://github.com/Sabhari-CEG/Custom-browser-tab-implementation.git>

after cloning it open it in the VS code inside some folder. To run it, you need to start the server. I had installed python 3 on my system so I will use the below code.

python -m http.server



Now my server is running on the port 8000. Let us view it in the browser.

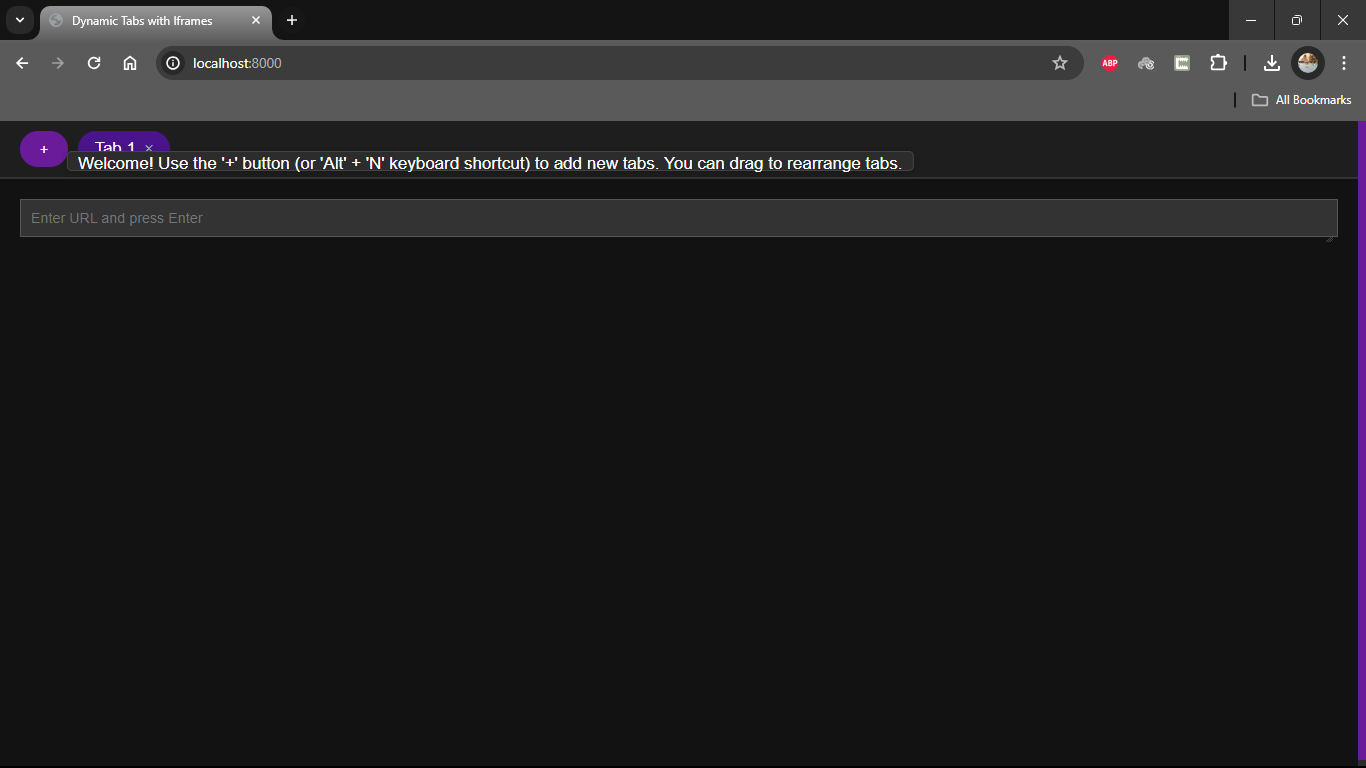


I can see my tab running on my port 8000. I had used a dark theme (my own preference). Let’s look at the functionalities.

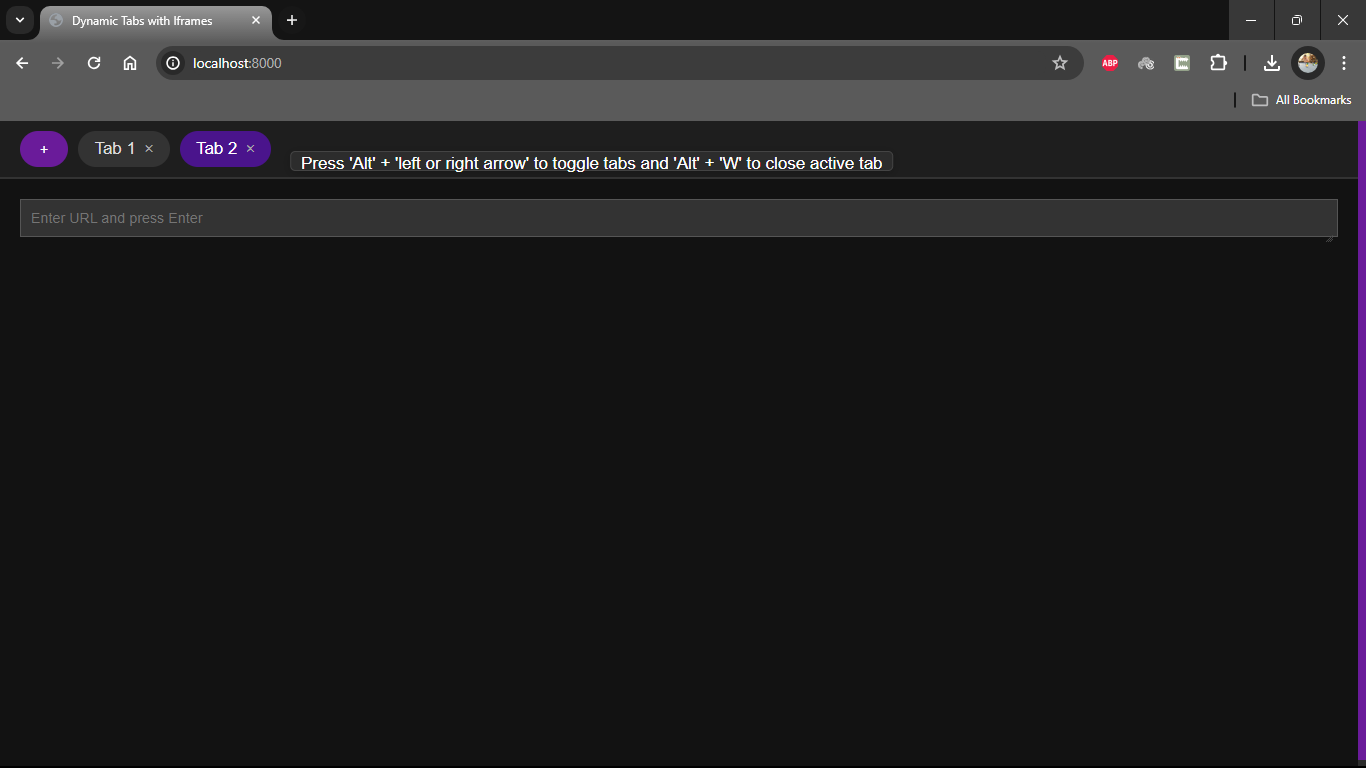
**Create tabs:**

The user can create tabs in two methods,

1. By clicking the ‘+’ button near the tab
2. By clicking the ‘Alt’ + ‘N’ keyboard combinations



On hovering the tab 1, I get this info message, additionally I can drag and drop the tabs to rearrange it. Now I’m creating a new tab by both ways.



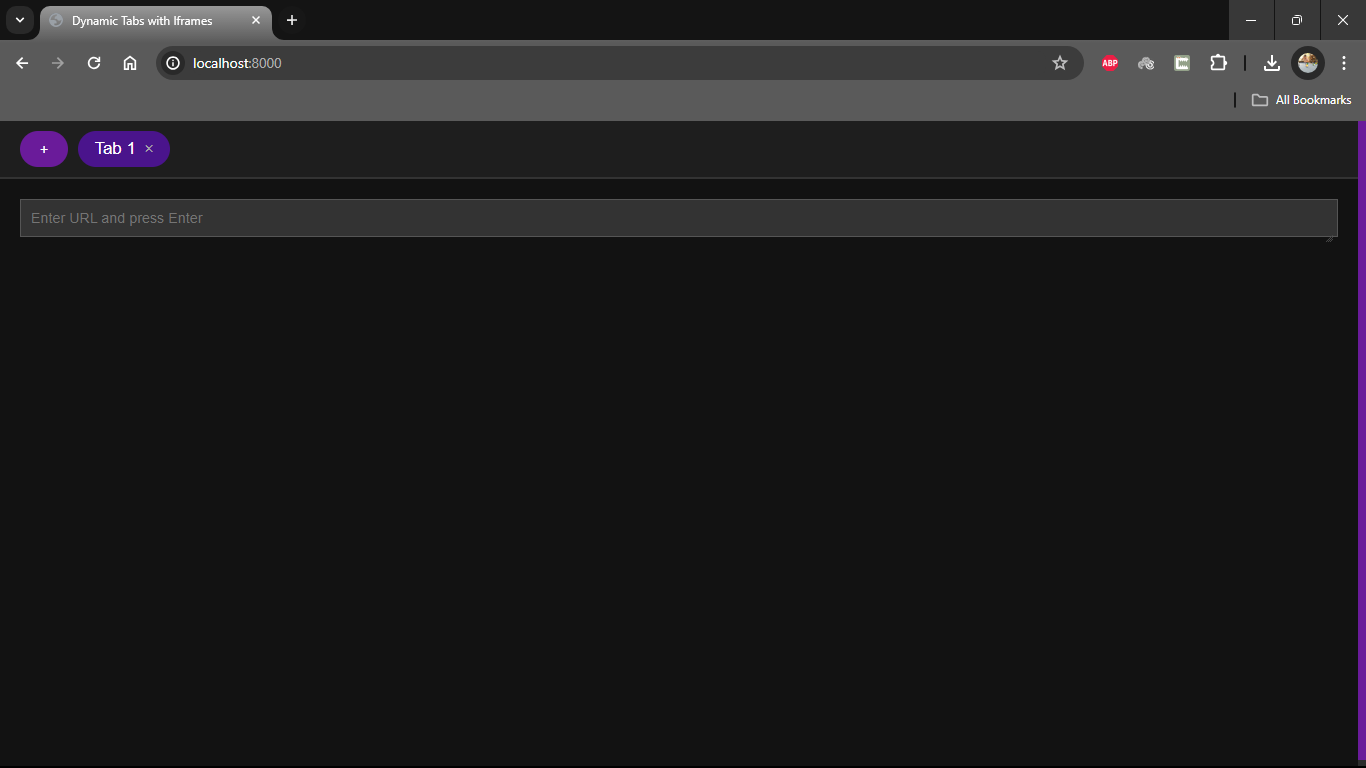
Both ways work and new tab is created.

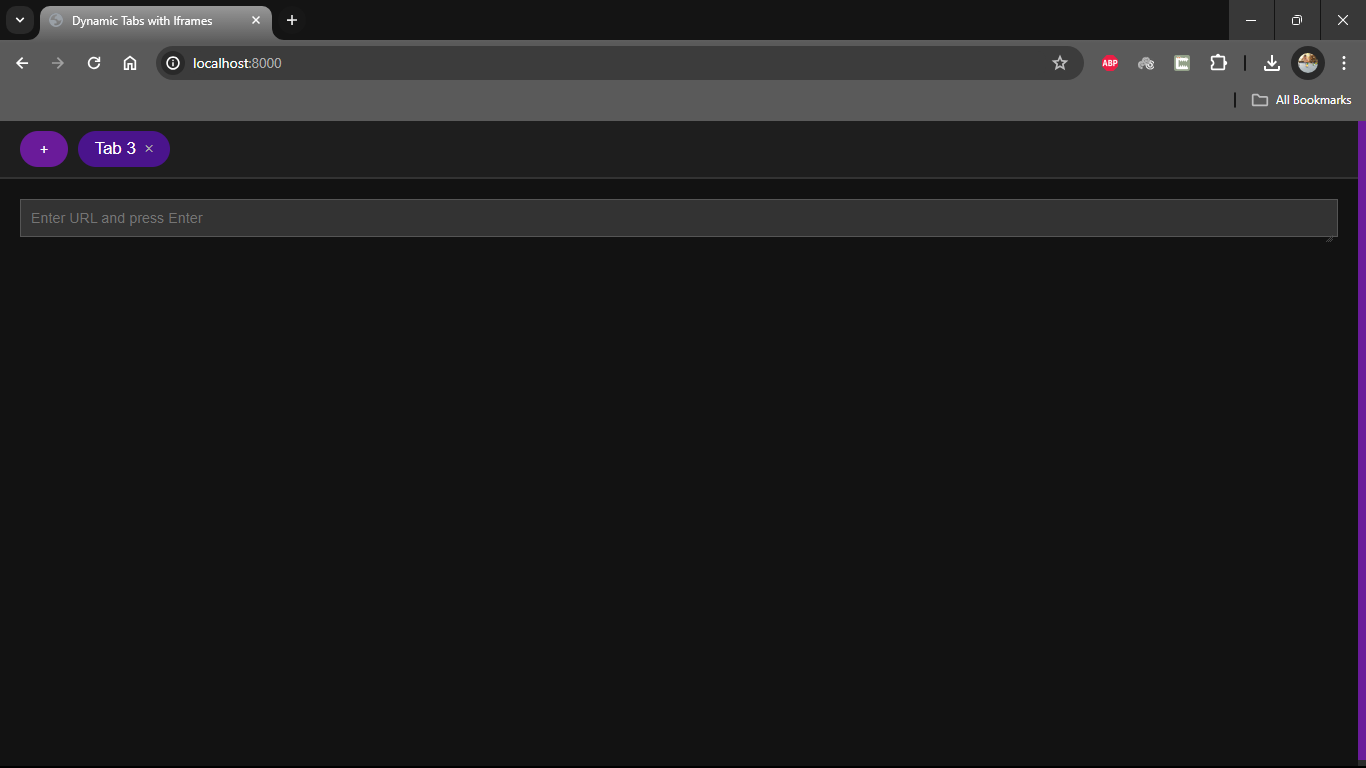
**Close a tab**

I can close a tab in two ways

1. By clicking the cross symbol next to the tab name.
2. By using the ‘Alt’ + ‘w’ keyboard combinations.

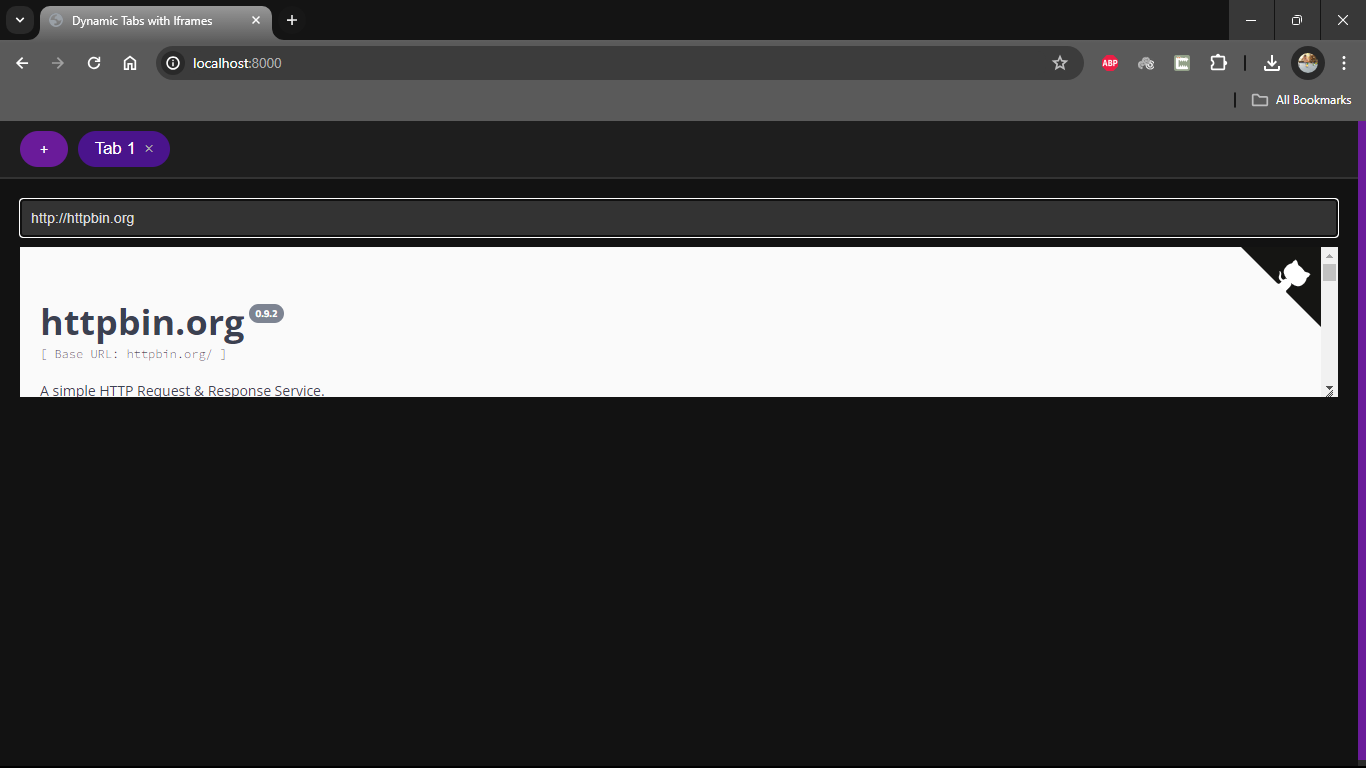
This info is given once when we create a new tab, let’s try both ways of closing a tab.

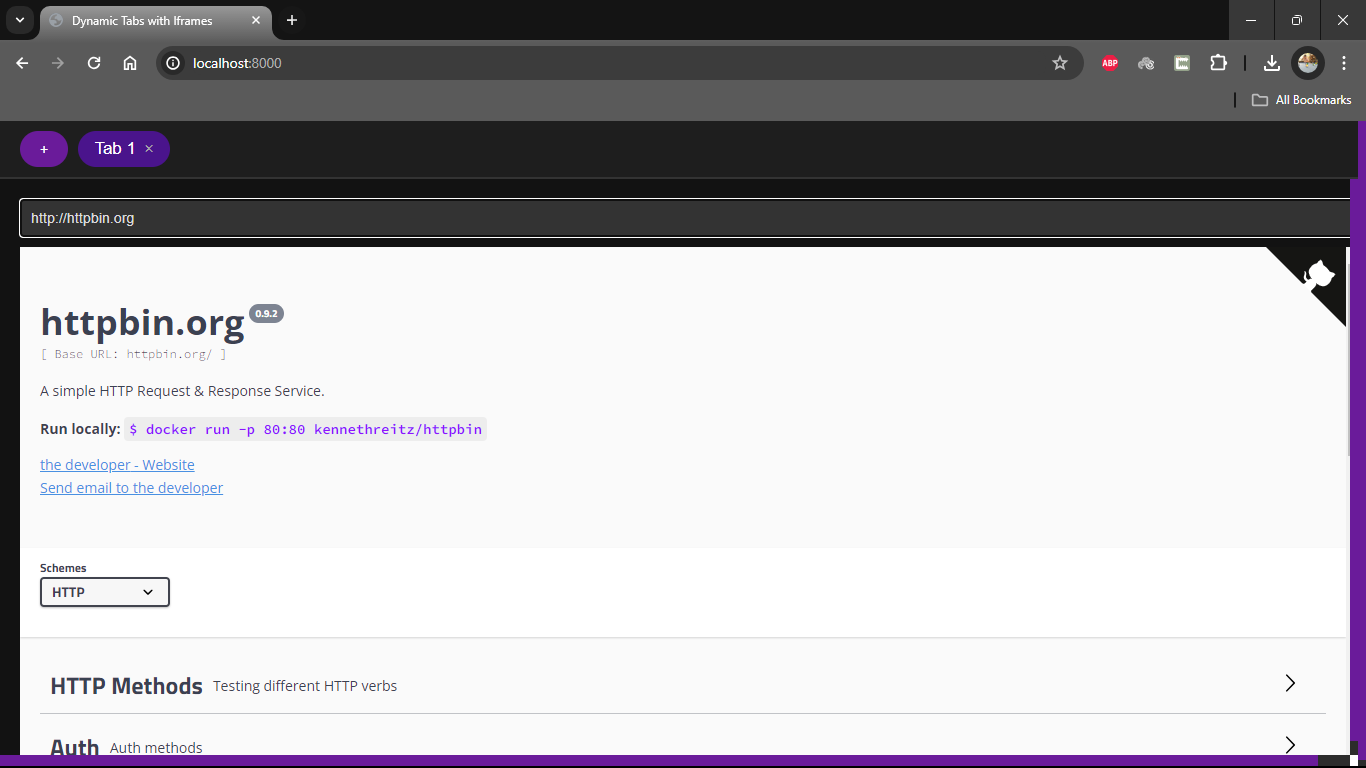


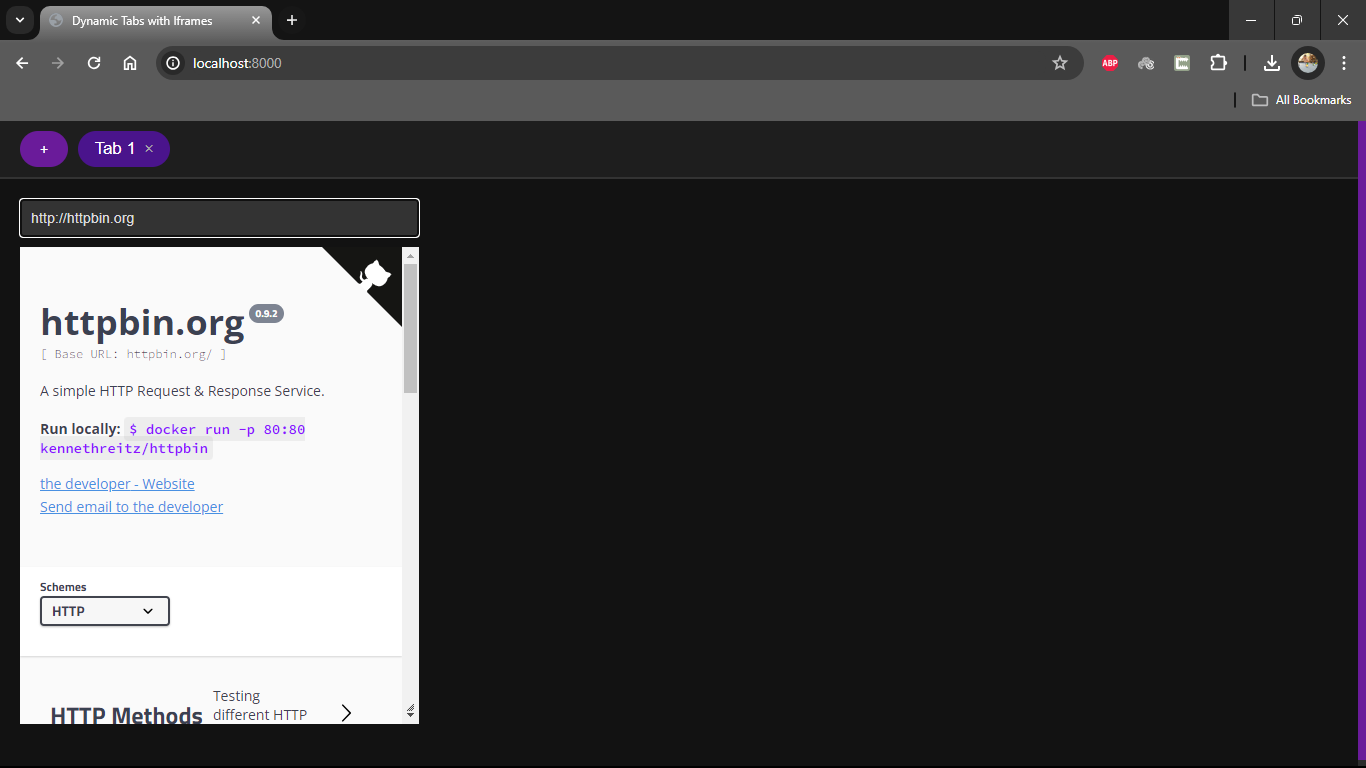


Both ways work.

**Load a content in Iframe:**

As soon as I enter the http url and hit enter the web page loads.  
  
I also had added a feature to resize the browser window



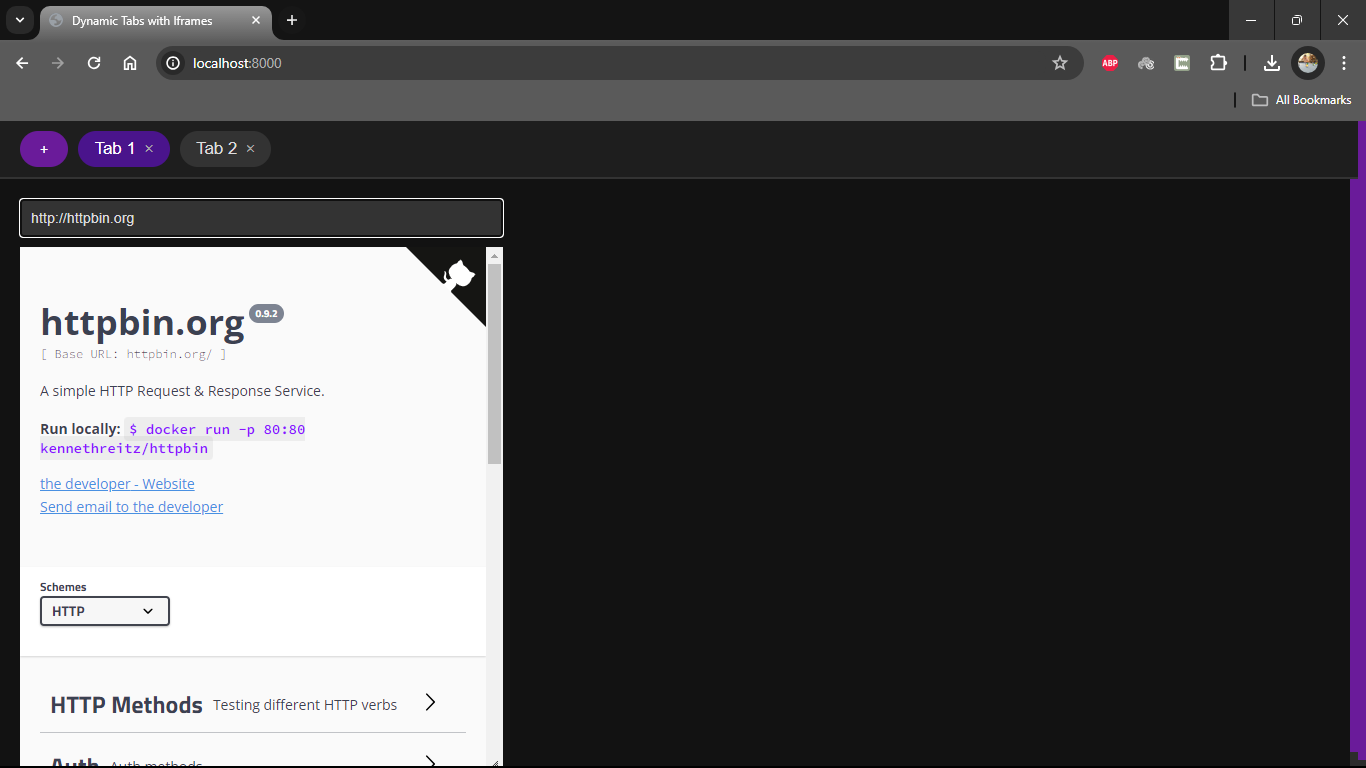


**Switch between tabs**

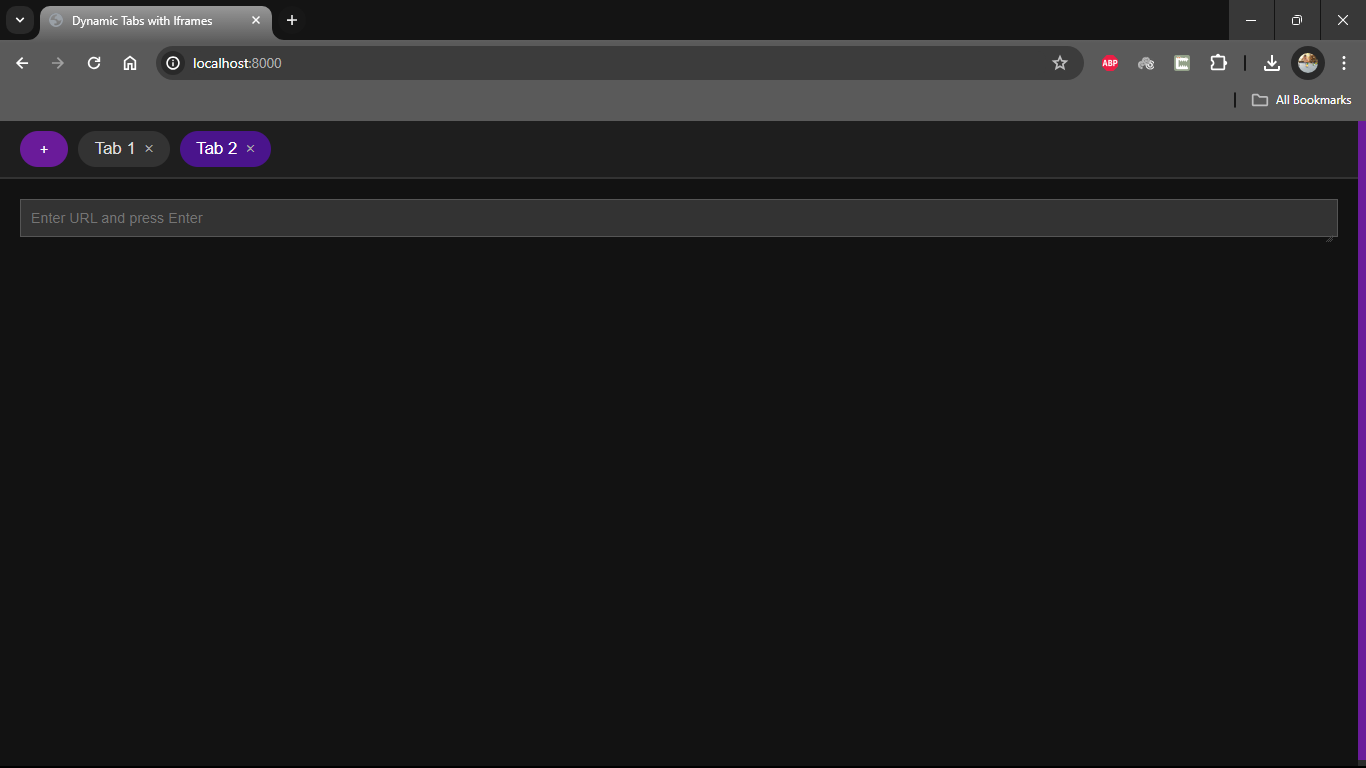
You can toggle between the tabs in two ways

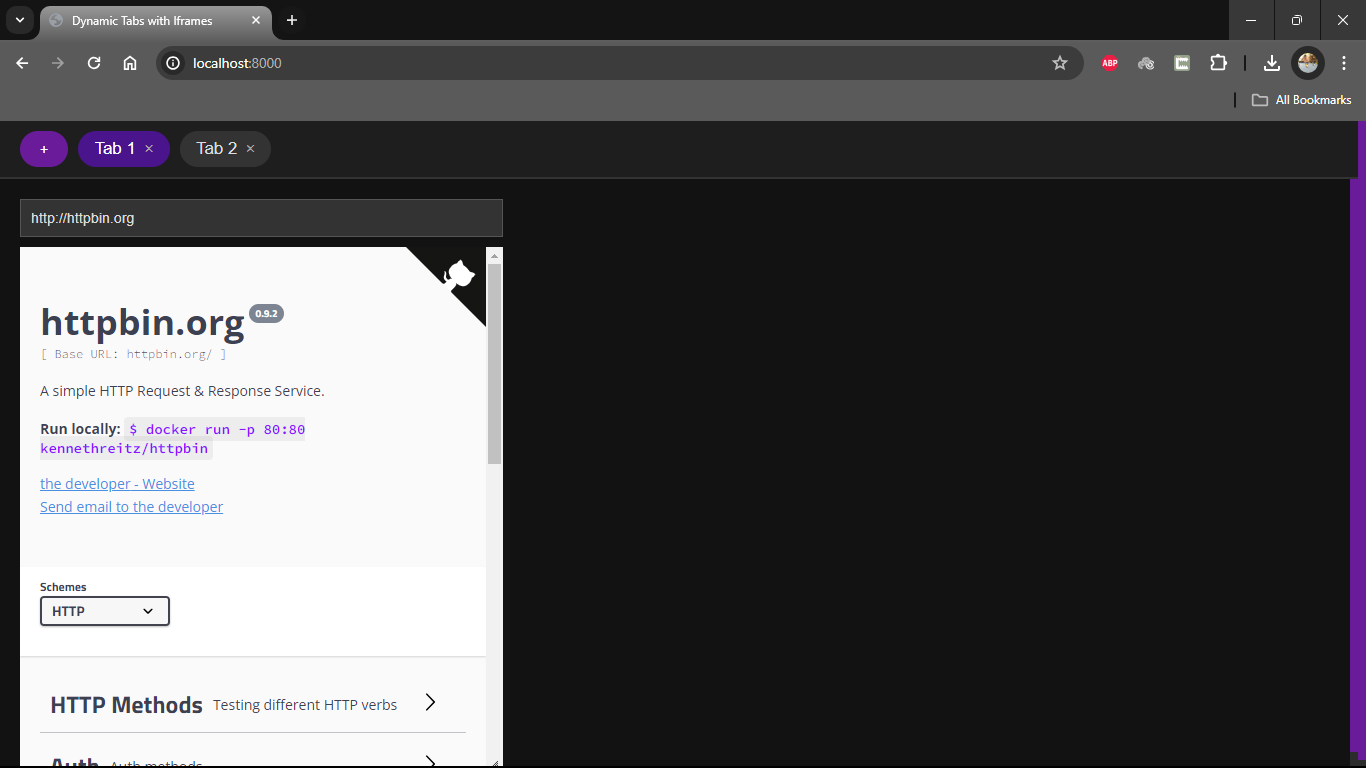
1. By clicking on the actual tab name
2. By using the “Alt” + “left arrow” to move to previous tab and “Alt” + “Right Arrow” to move to next tab.

This is my current tab



The active tab is highlighted, and the other tab name is visible. Now I’m using both the methods to toggle between the tabs,

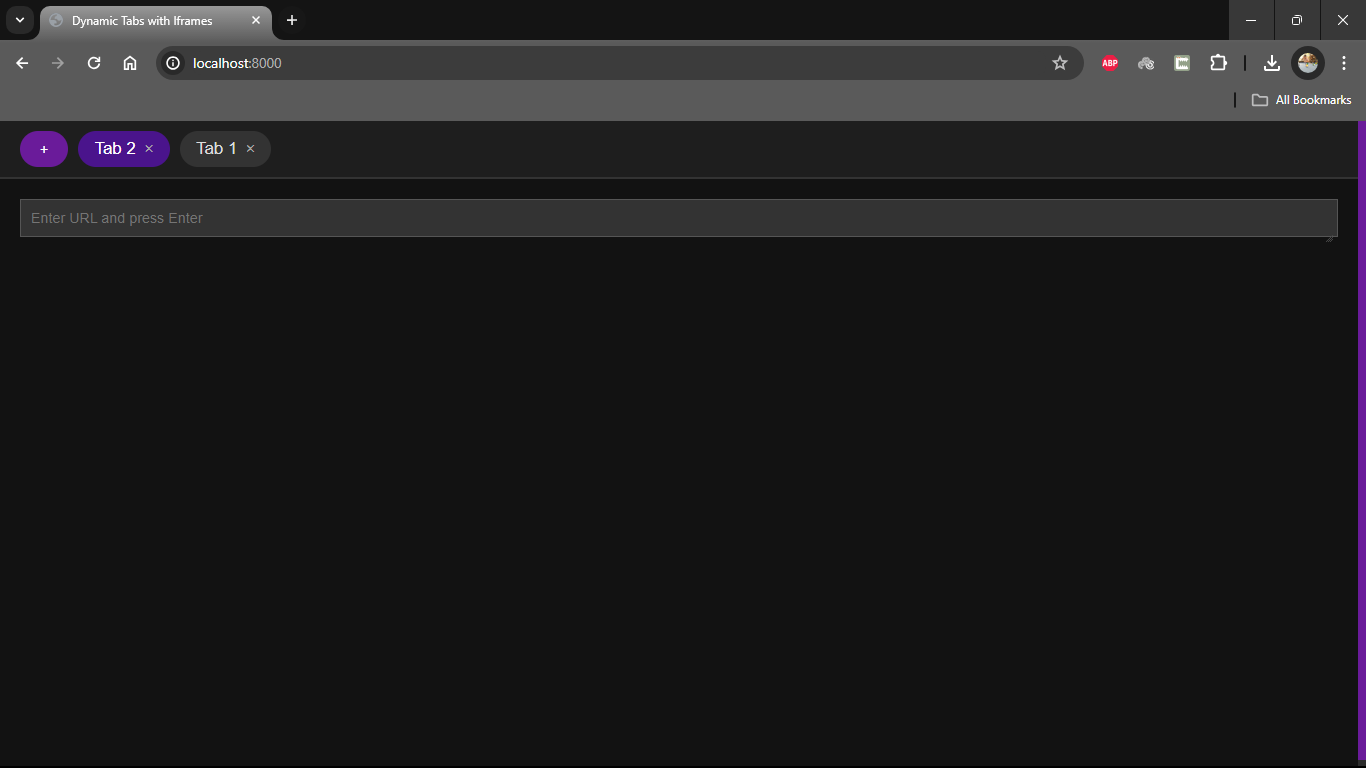




Both method works.

**Rearranging the tabs**

We can rearrange the tabs by simply dragging and dropping it.



Here I had moved the tab 2 before tab 1.

My design pattern most closely resembles the Module Pattern in JavaScript, combined with a touch of Revealing Module Pattern where functionality is organized into separate functions within a single encapsulated module (in this case, the $(document).ready() function).

**Analysis of the Design Pattern Used:**

**Module Pattern**

**Description:** This pattern is used to encapsulate functionalities in a single closure, protecting pieces from the global scope, which helps avoid function and variable collisions across the script.

**Implementation**: Your script wraps all functions (createTab, switchTab, closeTab, makeTabsSortable, cycleTabs, closeCurrentTab) inside the jQuery $(document).ready() function, which acts as a module initializer. This ensures that all functions and variables are localized rather than global, reducing the risk of conflicts.

**Revealing Module Pattern**

**Description**: An extension of the Module pattern where all functions and variables are kept private inside the closure until explicitly exposed.

**Implementation**: Although your script does not explicitly return an object that reveals public functions while keeping others private, it does isolate implementation details from the global scope, which is a key concept of this pattern.

**Efficiency of the Design Pattern:**

**Advantages**

**Namespace Management**: Encapsulating the functions in a single block reduces potential naming conflicts, especially important in a broader codebase.

**Maintainability**: Having all related functions and handlers within a single module makes the code more maintainable. It's easier to see and manage interactions between the elements and event handlers.

**Clarity and Organization**: The code is organized logically, where all related functionalities are grouped together, making it easier to understand and manage.