Software Requirements Specification Template

Dev Toys

Software Requirements Specification

Ver. 0.0023

4/24/2024

Kooper Young

Gabriel Mura

Divya Chelikani

Swathi Beepeta

Submitted in partial fulfilment

Of the requirements of

CSIS 44-691 Graduate Directed Project 11

# Revision History

Please see our Github that was assigned to us. It has a complete revision history, very well organized and sorted:

<git@github.com:44691-FA23/Dev-Toys.git>

# Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
| Kooper. Y | Kooper Young | Dev Toys | 4/24/2024 |
| Gabriel. M | Gabriel Mura | Dev Toys | 4/24/2024 |
| Divya. C | Divya C  helikani | Dev Toys | 4/24/2024 |
| Swathi. B | Swathi Beepeta | Dev Toys | 4/24/2024 |

**Table of Contents**

1. Introduction
   1. Purpose
   2. Scope
   3. Definitions, Acronyms, and Abbreviations
   4. References
   5. Overview
2. General Description
   1. Product Perspective
   2. Product Functions
   3. User Characteristics
   4. General Constraints
   5. Assumptions and Dependencies
3. Specific Requirements
   1. External Interface Requirements
      1. User Interfaces
      2. Hardware Interfaces
      3. Software Interfaces
      4. Communications Interface
   2. Functional Requirements
   3. Use Cases
   4. Class/Objects
   5. Non-Functional Requirements

3.5.1. Performance

3.5.2. Reliability

* + 1. Availability
    2. Security
    3. Portability
  1. Inverse Requirements
  2. Design Constraints
  3. Logical Database Requirements
  4. Other Requirements

3.10 Use Case Diagrams

1. Design

4.1. ER diagram

4.2 GUI

1. Analysis Models

5.1. Data Flow Diagram

5.2. Sequence Diagram

**1.Introduction**

**1.1 Purpose:**

The purpose of this web extension is to function as several smaller utilities inside of a web browser, very similar to Microsoft Power Toys, with which it shares a legacy name.

**1.2 Scope:**

The scope of this project can be defined as small or relatively small. It has only one folder and does not reference any external API or database. The project is roughly 1000 lines of code.

**1.3 Definitions, Acronyms, and Abbreviations:**

API – Application Programming Interface

**1.4 References:**

The only reference for this project is the Chrome Extension Manifest V3 Documentation:

<https://developer.chrome.com/docs/extensions/mv3/intro/>

[fullcalender](https://fullcalendar.io/)

[jQuery UI](http://jqueryui.com/)

**1.5 Overview:**

Our extension consists of 3 separate utilities.

1. A Tab Re-Namer
   1. This can rename tabs in a browser persistently
2. An Automatic Mouse
   1. This can store and then execute a series of mouse actions
3. A File Calendar

Users can preview information about calendar events, drag-and- drop events for scheduling ease, and directly create, edit, or delete events. Users can get a quick overview of the month's events without navigating away from their current view. This can be especially useful when planning future events or checking availability., while a list of events enables convenient drag-and-drop functionality to the main calendar, reflecting changes in both views. If a day has already passed, the events will change color to show they're over.

**2. General Description**

**2.1 Product Perspective:**

Given that we aren’t yet finished with our project, an external view of what we have done leaves much to be desired, especially when compared with more professional extensions. This is most noticeable in our UI, which feels clunky and unresponsive. This is a temporary issue and is expected to be fixed in the next semester. Where is does succeed is in the functionality, mostly. Our extension currently achieves 75% of its goals in terms of function, and we plan to reach no less than 95% of planned functionality during the next semester.

**2.2 Product Functions:**

1. Auto Mouse
   1. Can be toggled on and off
   2. Records mouse clicks on a webpage when on
   3. Can view mouse clicks on an archive page
2. Tab Re-Namer
   1. Can rename tabs
   2. Can change tabs favicon
   3. Can get favicons from a link
3. File Calendar

a. Combines file previews with calendar data for improved task management.

b. Displays file content and related calendar events in one view.

c. Allows direct creation/editing of calendar events within file preview.

**2.3 User Characteristics:**

Just like with most other chrome extensions, our product has no tightly defined user characteristics. It is intended to be used by whoever wishes to use it to solve their numerous problems. If we had to guess, most of our user base would be anywhere from 15-45, somewhat tech savy, and have a personality type aligning somewhere in the range of a person who values productivity though a person who is completely obsessed with neat tricks you can do with chrome extensions.

**2.4 General Constraints:**

Roughly speaking, our extension has the same constraints as any other chrome extension, in that it can do nearly anything that you could do with regular JavaScript, but only inside of a web page loaded inside of a chromium-based web browser. This means no reaching out to a desktop or even to the actual browser buttons without additional API’s.

**2.5 Assumptions and Dependencies:**

This extension assumes that you have a device capable of running a up-to-date Chrome web browser and know how to install chrome extensions. This extension only supports manifest version 3.

**3. Specific Requirements**

**3.1 External Interface Requirements:**

As of now, our extension has no external connections, API’s, or databases. It does utilize the chrome extensions internal storage however, but this is built in and requires no setup by the user or maintenance by the developer.

**3.1.1 User Interfaces:**

The user interfaces are built using HTML, CSS and Bootstrap, and use a mix between popup tabs and web pages for displaying information

**3.1.2 Hardware Interfaces:**

Our extension requires no hardware interfaces.

**3.1.3 Software Interfaces:**

Our extension only requires a chromium-based web browser with our extension installed.

**3.1.4 Communications Interface:**

Our extension requires no communication interfaces set up by the developer or user. All internal and external communication is handled by chrome and is proprietary information that cannot be listed here.

**3.2 Functional Requirements:**

1.Auto Mouse

* 1. Be toggled on and off
  2. Records mouse clicks on a webpage
  3. View recorded mouse clicks
  4. Get and Set data using chrome. Storage
  5. Preform recorded mouse actions
  6. Respond to programmable keyboard shortcuts

2.Tab Re-Namer

a. Persist data throughout sessions

b. Return icon to original state before change

c. Return title to original state before change

d. Change title

e. Change favicon

3. File Calendar

a. Users create events by dragging files onto the calendar.

b. Associate files with events via drag-and-drop

c. Real time calendar update

d. Easily reschedule events by dragging them to a new date/time.

e. View/open attached files from the calendar.

f. Implemented CRUD events.

**3.3 Use Cases:**

1. Auto Mouse

a. Click the extension icon to open the UI

b. Click record

c. Preform mouse actions

d. Stop recording

e. Reopen the extension UI to navigate to the archive

f. View recorded mouse actions

2. Tab Re-Namer

a. Click on the extension to open the UI

b. Insert a new title, favicon link, or favicon image

c. Press the submit button to show changes on tab

3. File Calendar

a. Created random events in calendar

b. Drag attachments for instant event creation.

c. Swiftly move meetings by dragging events.

d. After the day is passed, change the color of past events on the calendar.

e. Resolve conflicts during rescheduling with alerts.

f. Create, Read, Update, and delete events.

g. Recreate events for particular week

**3.4 Class/Objects:**

Chrome extensions all share a similar structure. You have a popup.html file for the UI, a service worker (background.js) that handles global communication and organization, and a content script that can interact with a web page. This is not a static structure, and sometimes extensions have more or less than these pages, such as Auto Mouse, which has an additional web page called archive.html.

**3.5 Non-Functional Requirements:**

**3.5.1 Performance:**

Performance is expected to be on par with other small chrome extensions. The only performance problem that could occur currently would be if the user's internet extremely slow, something less than 2mbps. This could cause the content scripts to not access the web page correctly.

**3.5.2 Reliability:**

The software rarely actually crashes, but given that it is currently unfinished, it often fails to meet its functional requirements. Any failure points in code are currently already addressed with error catching, but as we continue to develop, more failure points will be exposed.

**3.5.3 Usability:**

Using our extension is intuitive, especially if you have ever used an extension before. However, the UI could be made more responsive to let the user know that work is being done, but that is a job for a later time.

**3.5.4 Security:**

There are no security issues. The database is managed by chrome and all user information is stored on their local account. Any security issues are on Google’s side and not on the extension.

**3.5.5 Portability:**

Because our extensions use a synced storage system, the extension can be used on any chromium-based browser that has the user’s google account signed in. It cannot be moved to a non-chromium-based browser.

**3.6 Inverse Requirements:**

The application should allow a single extension to perform multiple convenient tasks. Each piece of the extension should achieve different things but work together to make one productive tool. Example: You schedule a task with a file using the file calendar, automate the tedious aspects of that task with auto mouse, and use the tab rename to organize the manual tasks you have left.

**3.7 Design Constraints:**

The biggest design constraint is the fact that we are using chrome extension architecture. We are confined to the limits of JavaScript and the libraries provided for extensions to communicate with web pages. There are also constraints on storage size and extension speed.

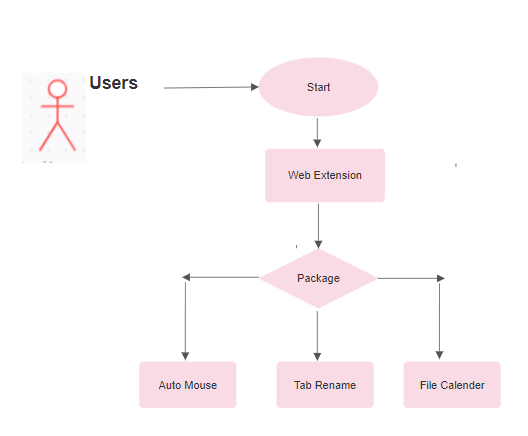
**3.8 Logical Database Requirements:**

There is no database requirement as chrome has storage that extensions can access to set data in. Since large amounts of data are not being shared, we can get away with just using the built in chrome storage.

**3.9 Other Requirements:**

Other requirements include being a quick extension that is easy to use. There should be no confusion about what the extension is doing to the user. It should be simple and intuitive. All actions performed should be correctly displayed to the user and actions done as expected.

**3.10 Use Case Diagrams:**

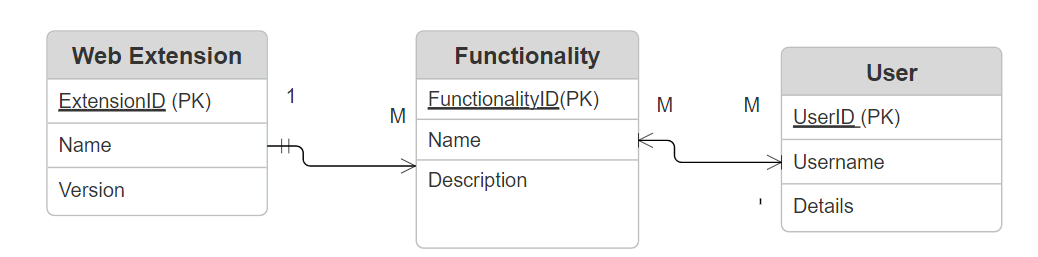


**Figure: Use Case Diagram**

**4. Design**

**4.1 ER diagram:**

* Extension to Functionality: One-to-Many.
* User to Extension: Many-to-Many.
* User to Functionality: Many-to-Many.



**Figure: ER Diagram**

**4.2 GUI**

I’m assuming this means you would either like to see our GUI or have it explained. The explanation is simple – because our GUI is not finished. It was deemed least important of all things, so it is a task for the next semester. Every GUI is written in HTML and usually Bootstrap is involved in some way or another.

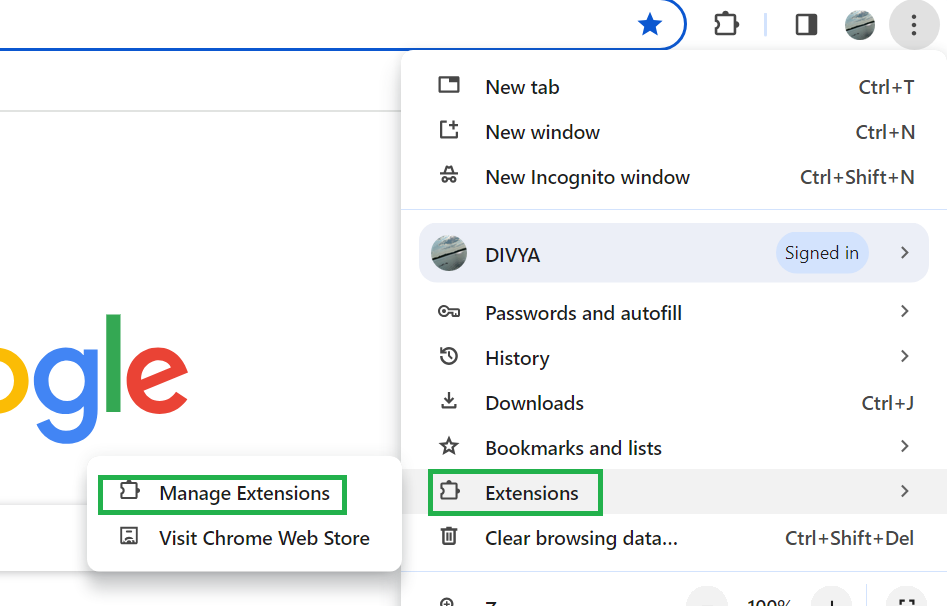
Creating a Chrome extension is quite straightforward. Here are the simplified steps.

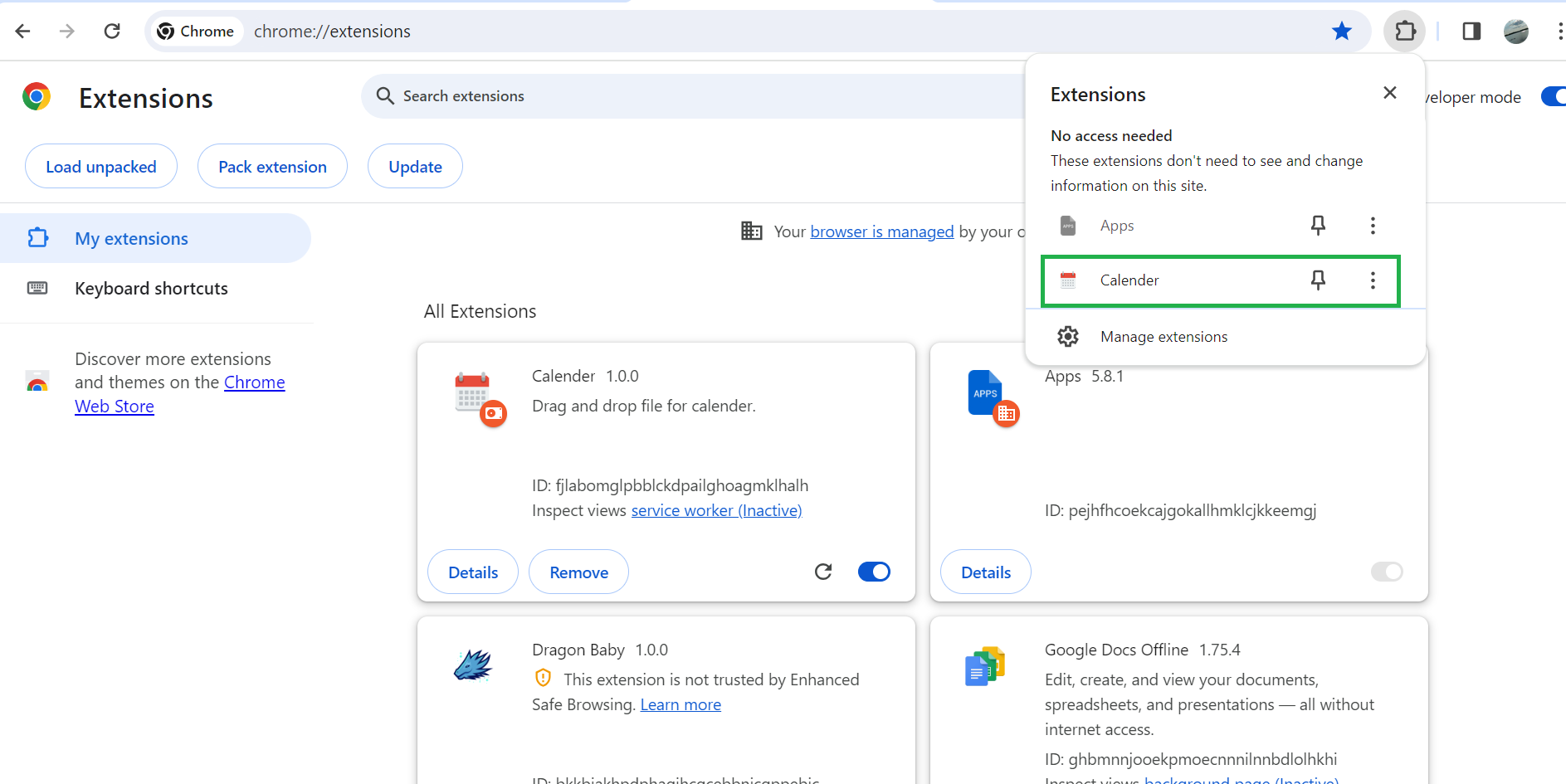
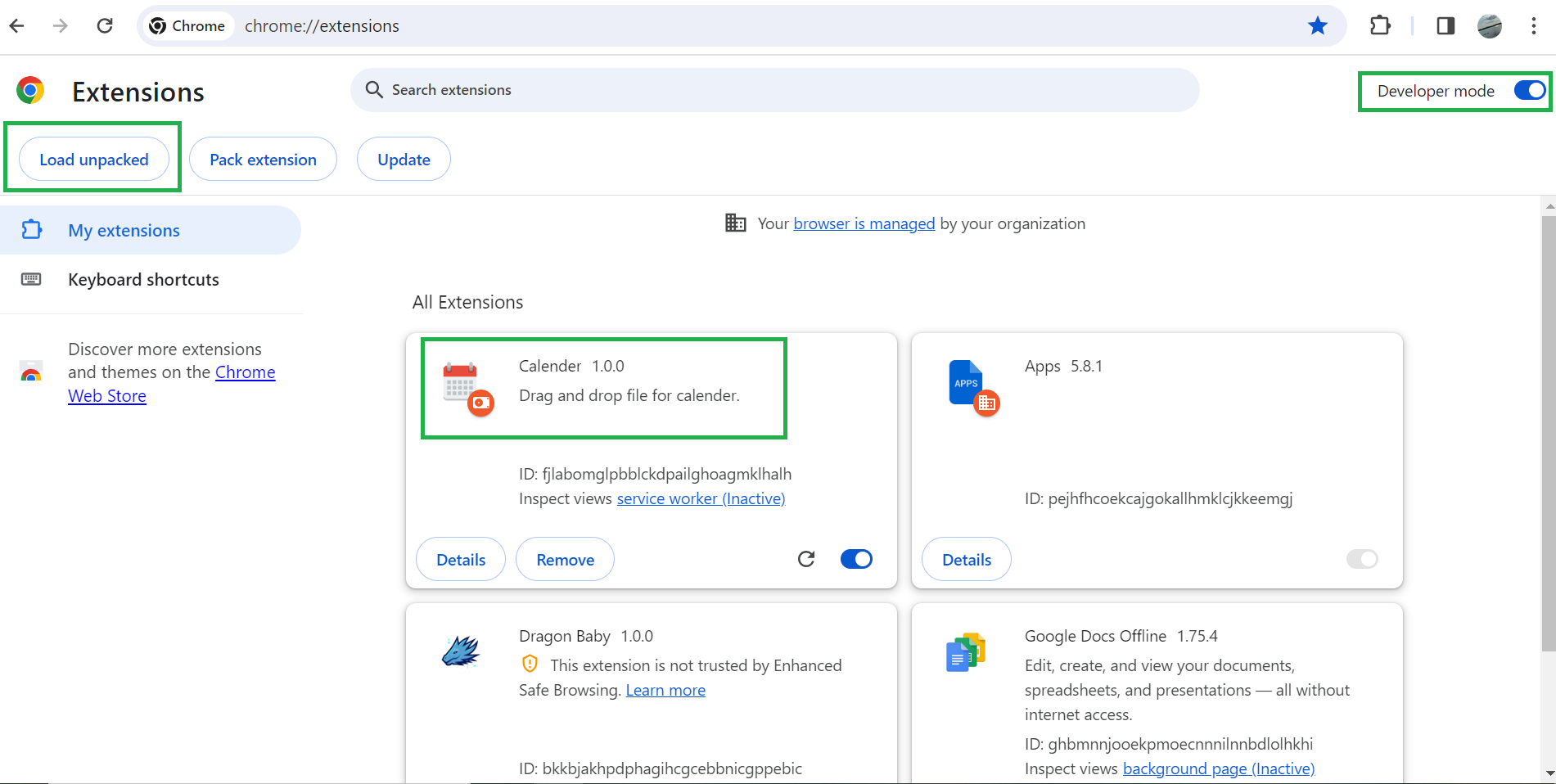
**Setup** : Get a text editor and install Google Chrome if you haven't already.

**Create\_Files** : Make a new folder for your extension. Inside, create a file named manifest.json to define your extension's details like its name and version. Then, create HTML, CSS, and JavaScript files for your extension's functionality.

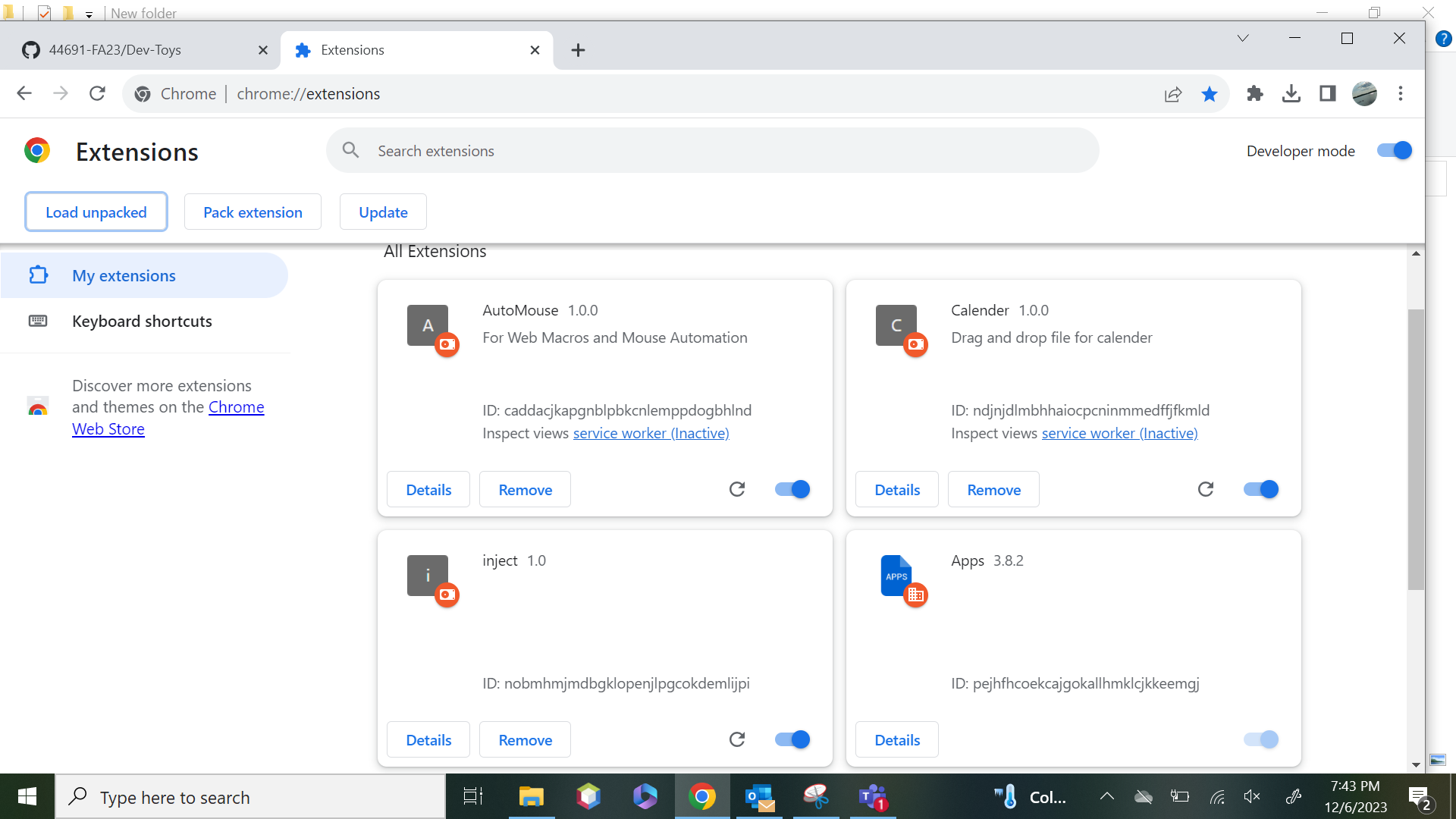
**Code\_Your\_Extension** : Write the code for your extension's features in the HTML, CSS, and JavaScript files you created.

**Test\_Locally** : Open Chrome and go to extensions and click on Manage Extension.

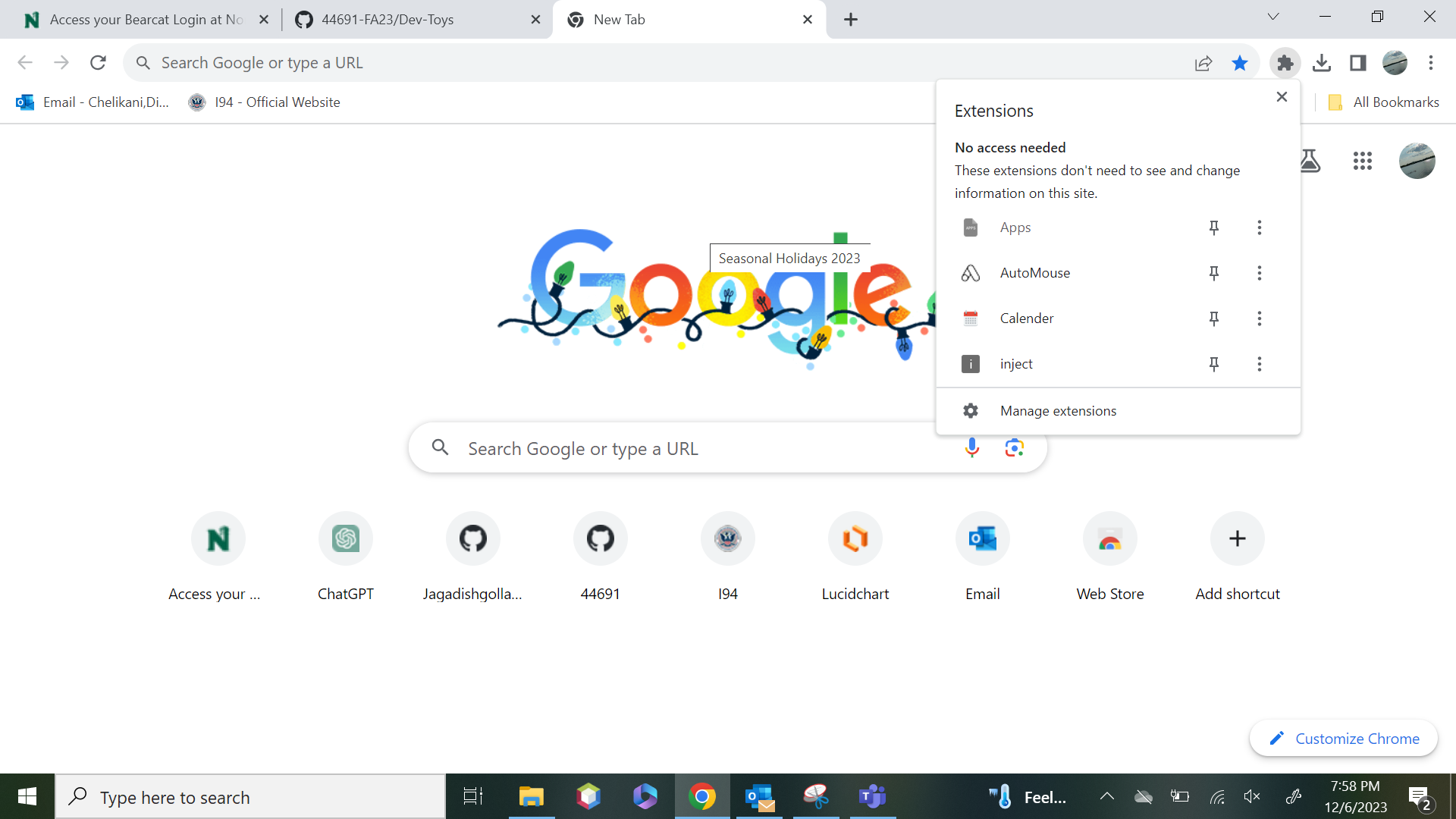


Turn on ***Developer mode***. Click ***Load unpacked*** and select the folder with your extension files.  


**Extension:**   
  
**Loaded Auto mouse, Tab Rename and File Calendar in Web Page**

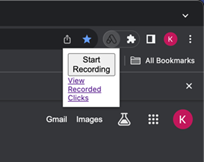


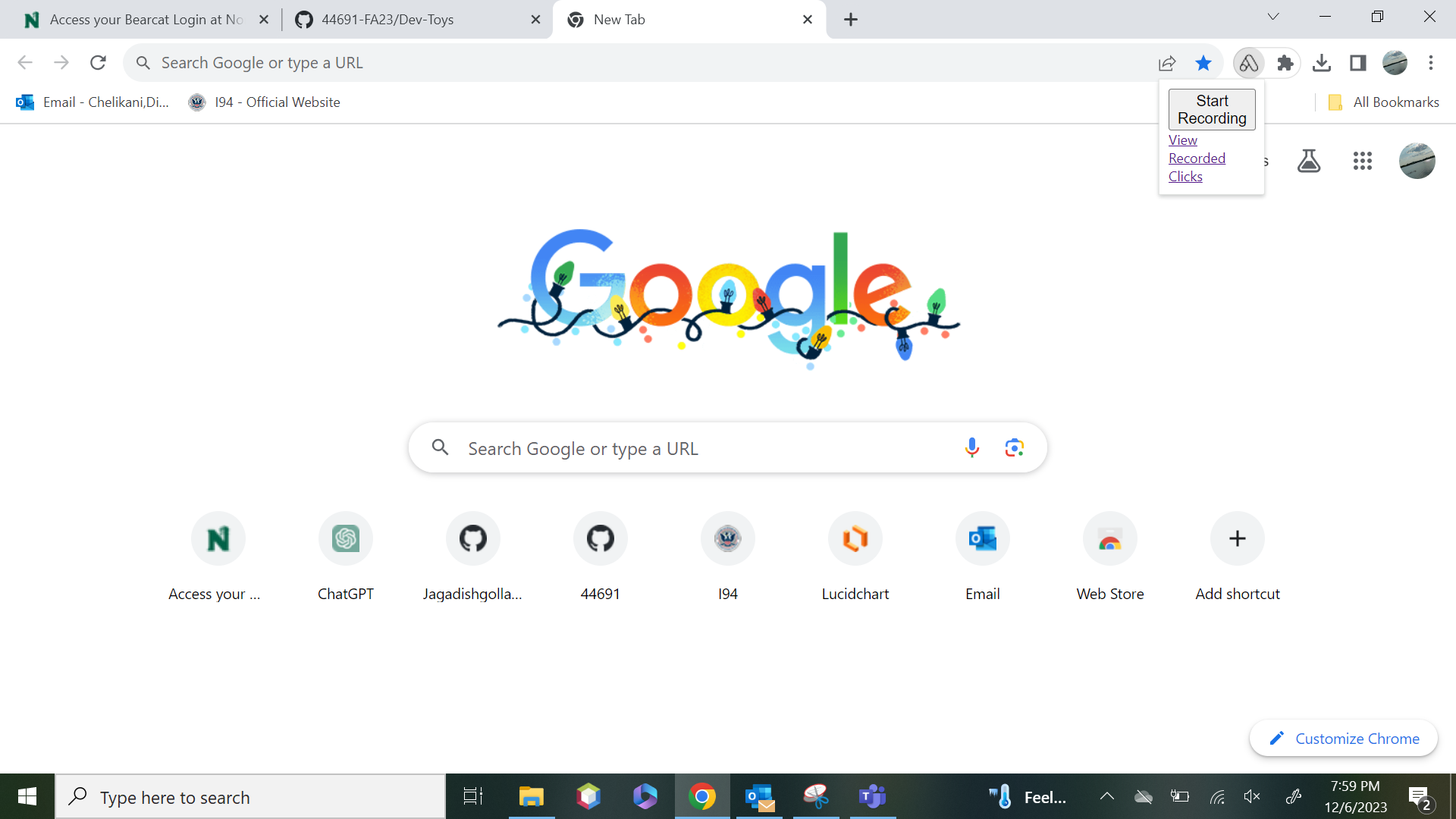
**Here you can see the Auto mouse, Calendar and Tab Rename in extensions**

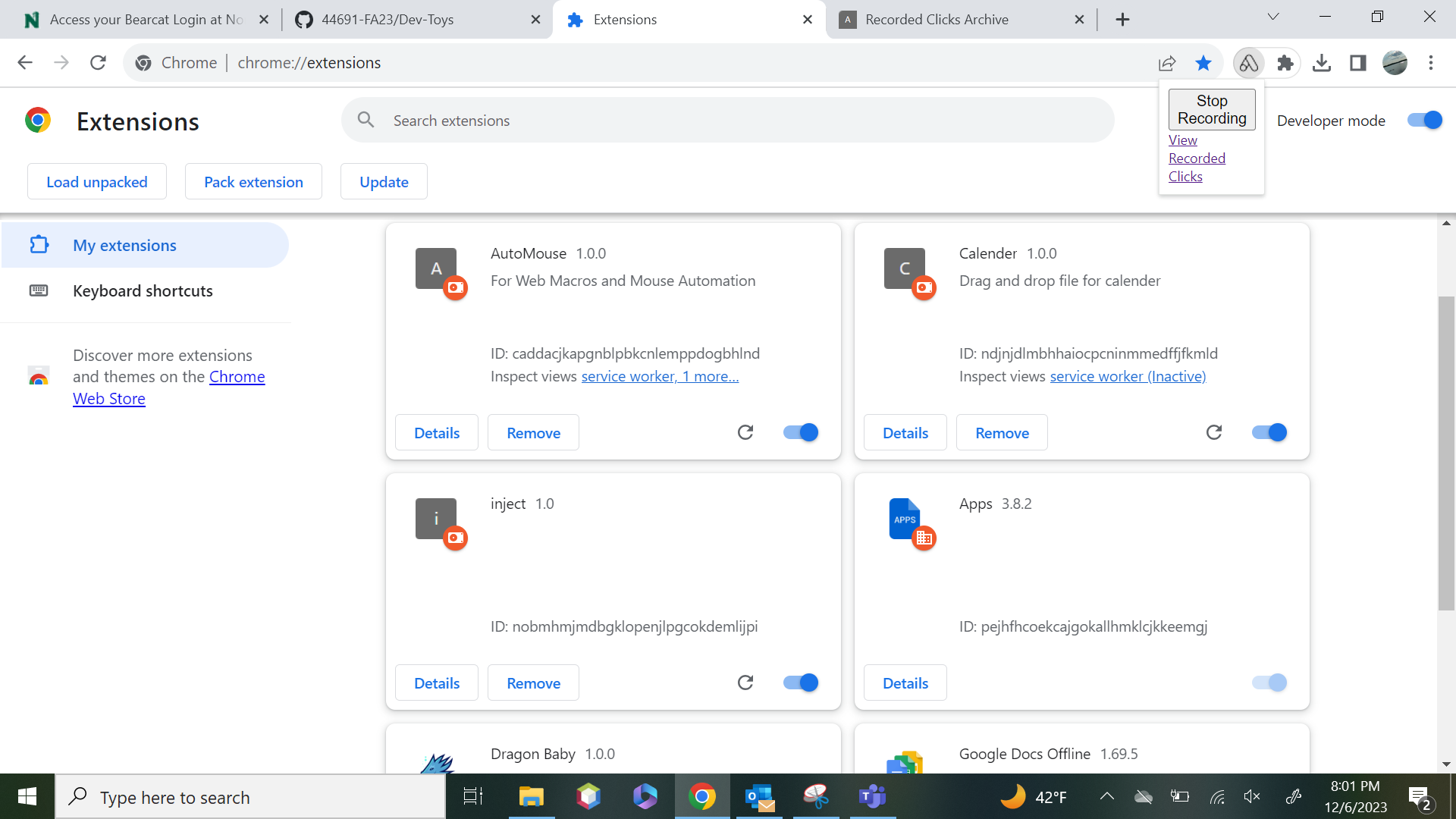


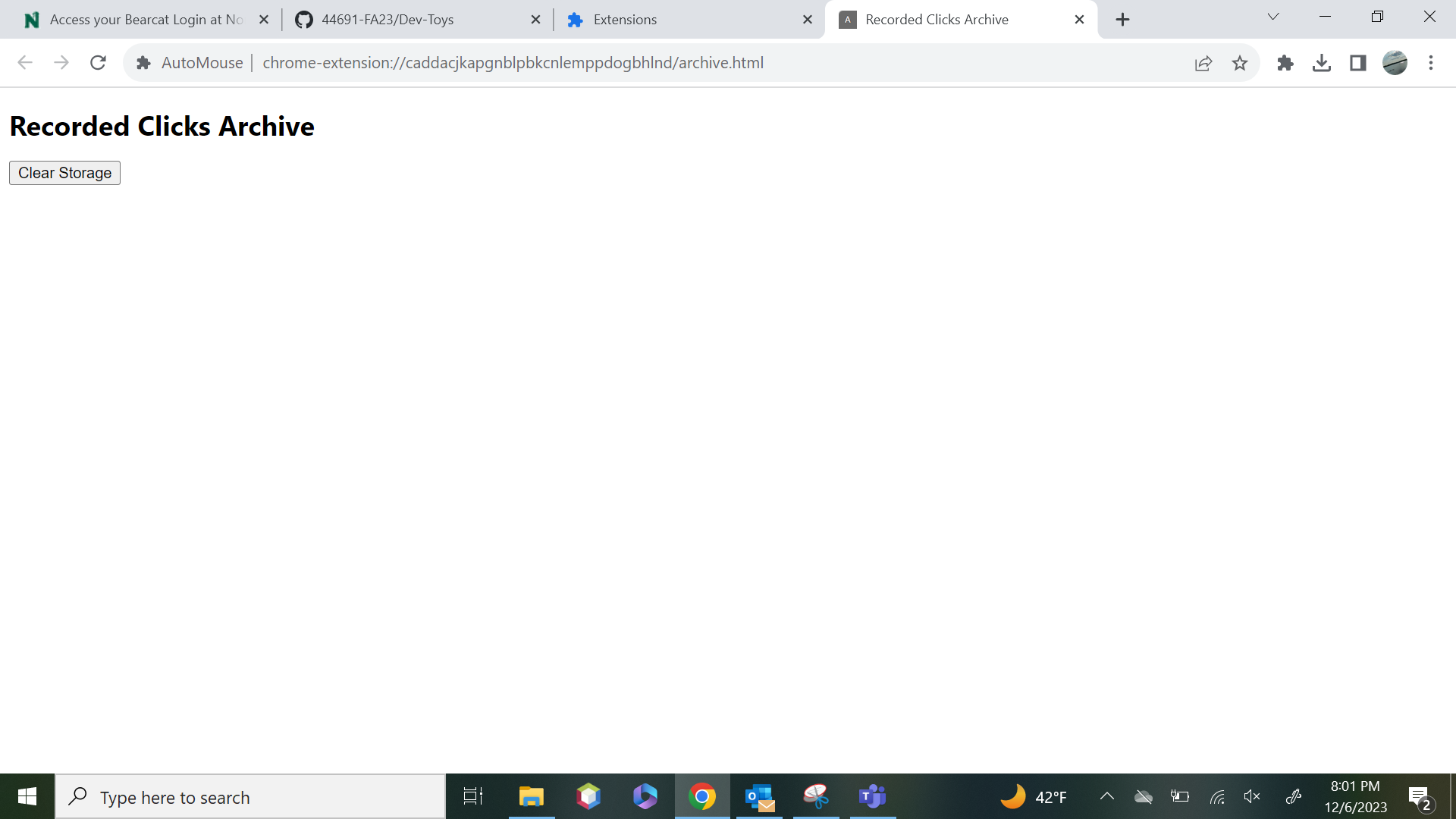
**a. Auto mouse**

Here are some screenshots of our app. Remember, our app is a chrome extension, so the entire GUI is a popup in the corner of the screen.



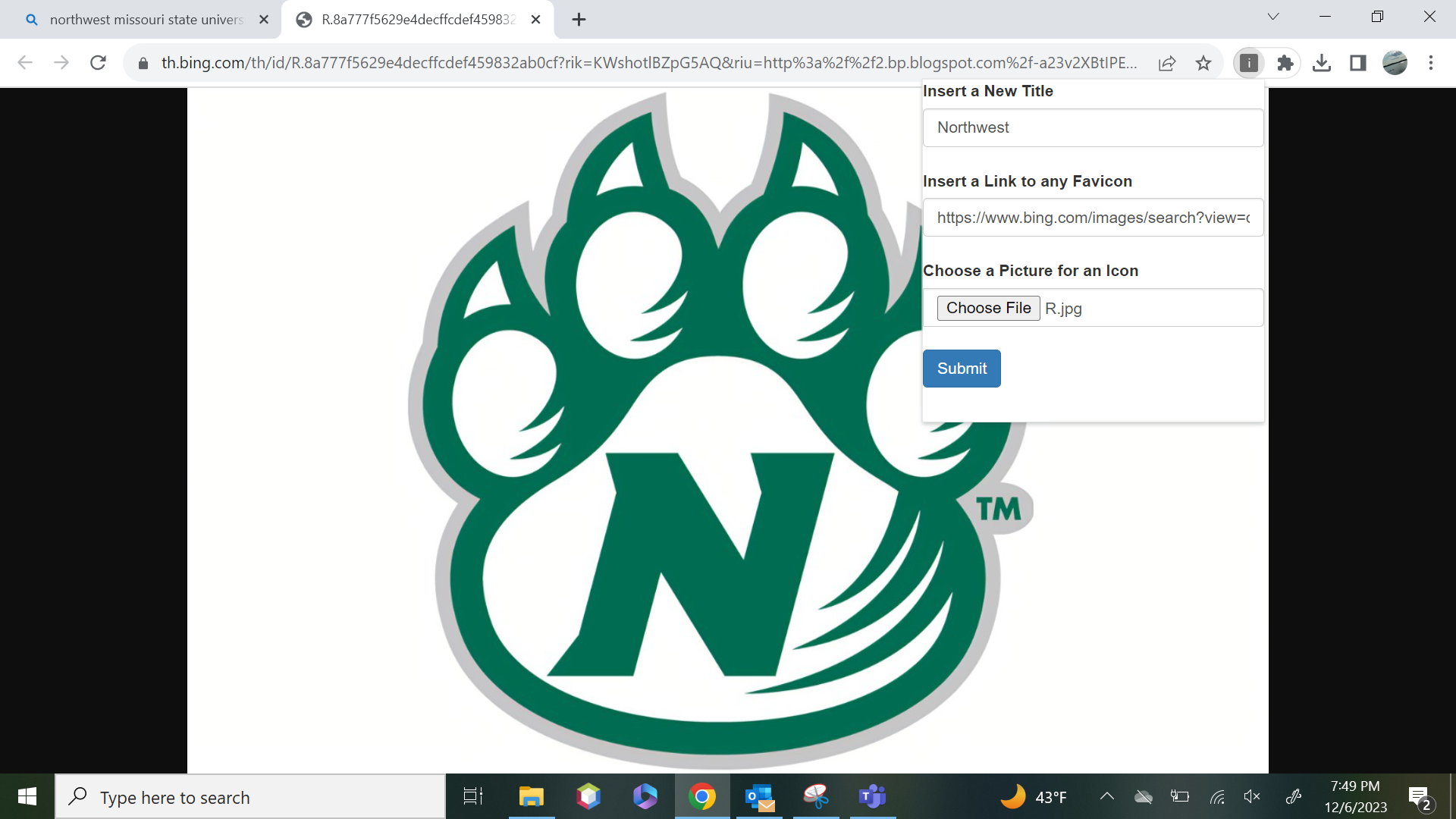




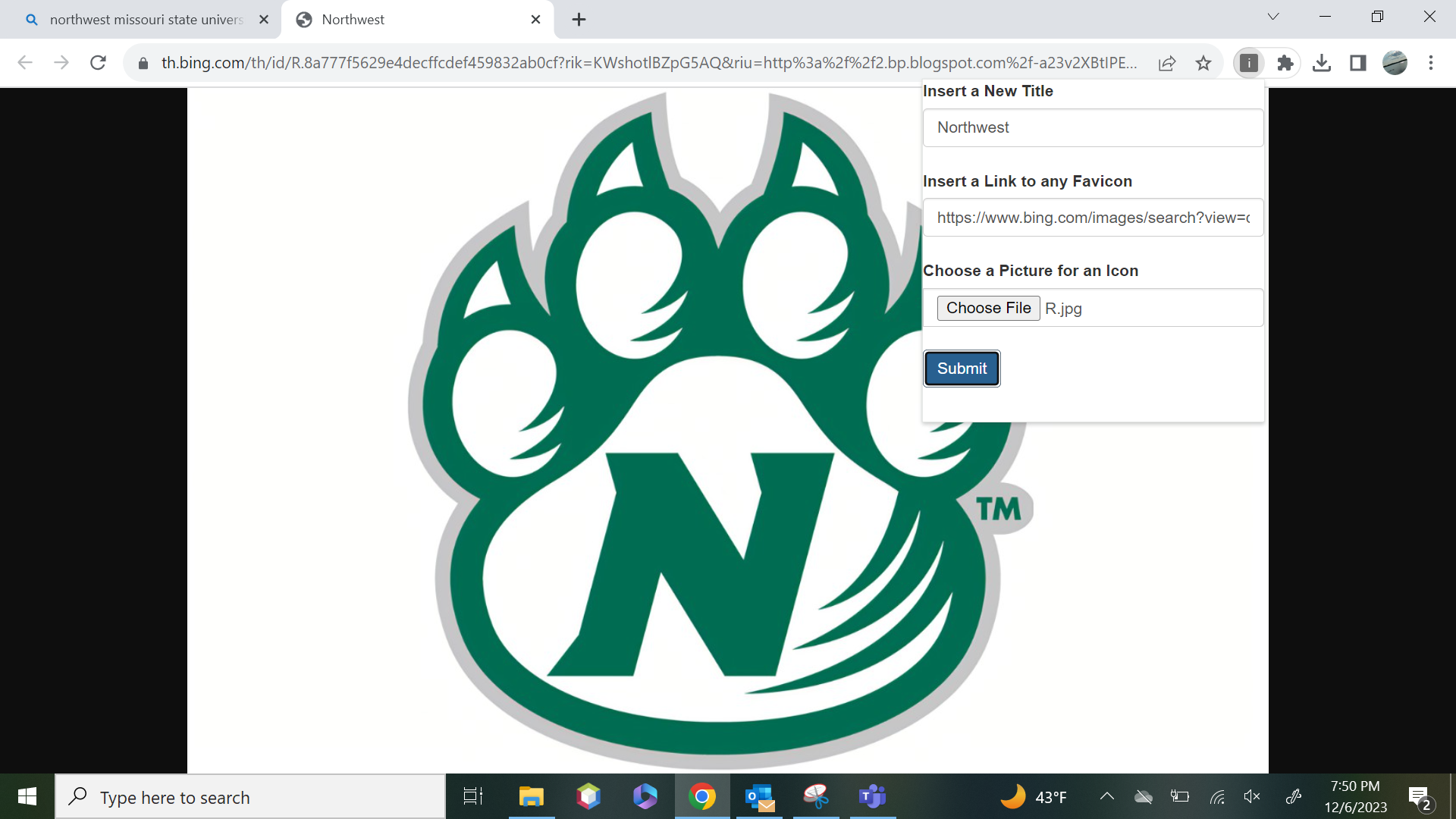


**b. Tab Rename**

We successfully created a web extension named 'Northwest' by copying and pasting a URL link, accompanied by a carefully chosen picture to enhance its visual appeal."



Here you can see the tab renamed as **NORTHWEST**.



**c. File Calendar**

# **Main\_Calendar**

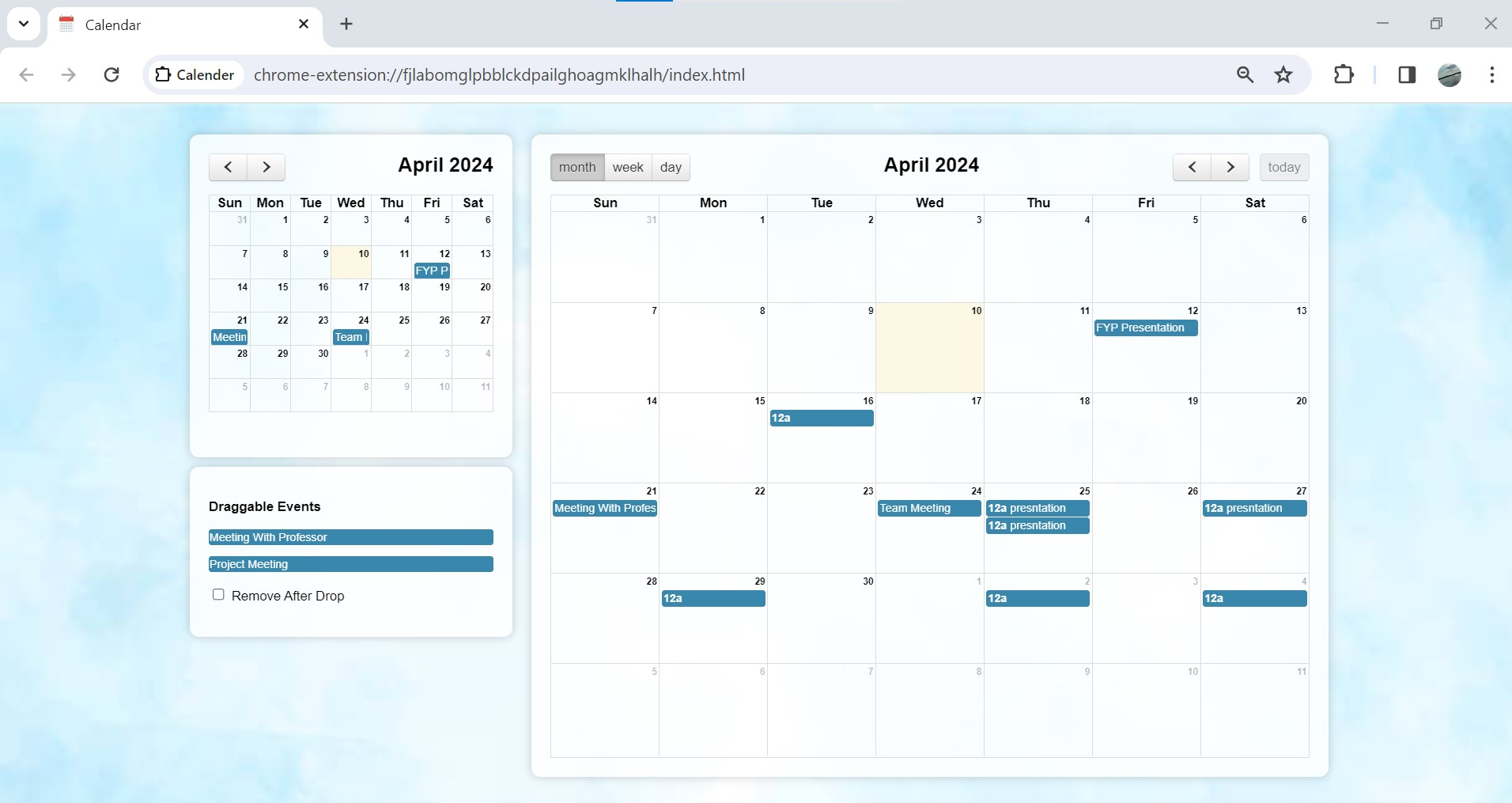
* If you want to schedule an event for a specific day, just click on that day on the main calendar. You'll see options to add the event's title, description, start date, and end date. Once you've filled in the details, click on "Create Event" to add it to your calendar.
* Once the event is created, you can preview the information about the meeting by clicking on it. If you need to change the date or time, you can drag the event to the updated time.
* If you need to update any other information about the event, just click on the "Edit" button, make your changes, and click "Save."
* Each event will be highlighted by a calendar color, making it easy to see when you have appointments or commitments.
* We've added a visual indicator for past events. After both the event day and scheduled time have passed, the event block's appearance changes, signaling that it's already happened. This "grid off" status assists users in promptly recognizing completed events, streamlining their focus on upcoming or ongoing ones.
* And if you decide you no longer need an event, simply click on it and then click "Delete" to remove it from your calendar.
* If you want to create an event that repeats weekly, you have the option to customize which days of the week the event occurs on. We've implemented a checkbox system where you can simply check the box next to the days you want the event to be recreated. This gives you flexibility and control over your recurring events, allowing you to tailor them to your specific needs.
* This way, managing your schedule is easy and intuitive, allowing you to stay organized and on top of your commitments.

# **Mini Calendar**

* The small calendar is like a mini version of your whole month's schedule. It lets you quickly check what's coming up without leaving your current view. It's great for planning future events or seeing when you're free. This way, you can stay top  
   of schedule and never miss a thing.

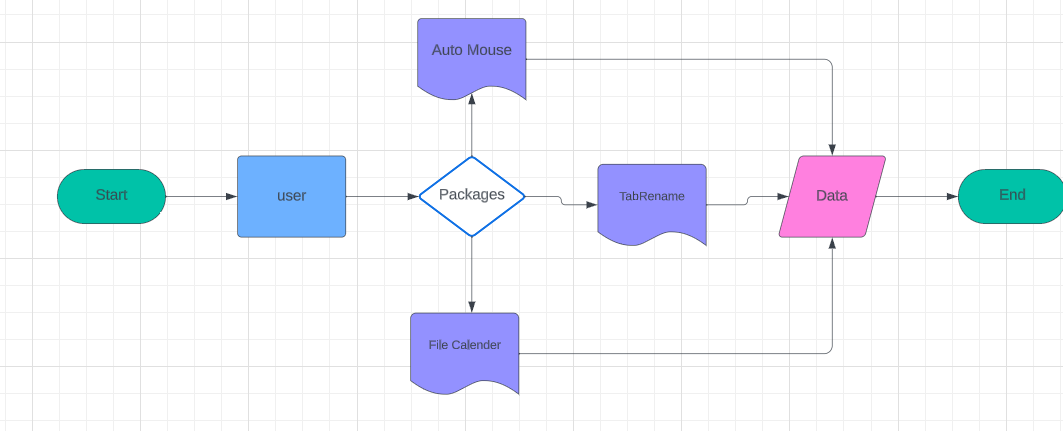
# **Draggable Events**

* Here, You can create draggable events for things like meetings with your professors, team meetings, project meeting, or presentations for your final year project. These events are like virtual sticky events that you can easily move around on your calendar to schedule them at different times.
* If you have no gatherings longer need an event or want to reschedule it, you can simply drag it to a new time slot or delete it after dropping it on the Main calendar. There's an option to enable a checkbox for "remove after drop," which automatically removes the event from its original position after you drop it onto the Main calendar.
* you drag one of these events onto your main calendar, it shows up as a colored pop or a highlighted block in both calendars as well.
* This visual representation helps you manage your time effectively and stay organized amidst your busy schedule.



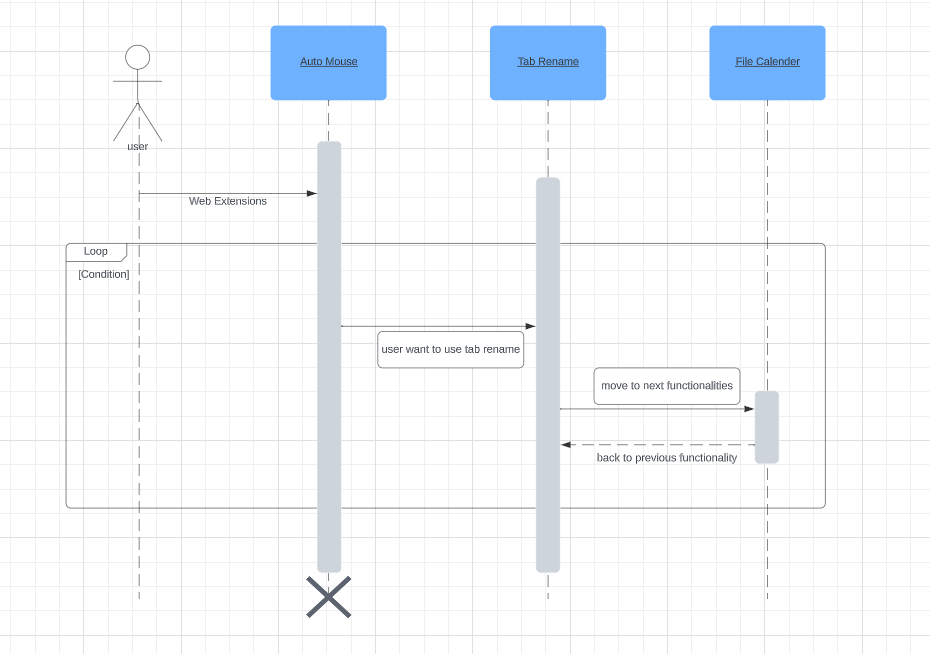
**5. Analysis Models**

**5.1 Data Flow Diagram:**



**Figure: Data Flow Diagram**

**5.2 Sequence Diagram:**



**Figure: Sequence Diagram**