BROWSER BUDDY REQUIREMENTS SPECIFICATION

SP-17 Browser Plugin



Elliott Brown Nathan Lynes Jacob Germana-McCray

Professor Perry | 02/04/24

Kennesaw State University | 4850-02/01 Senior Project

Table of Contents

1. Introduction

1.1 Overview

Contained within this document are the requirements for the Browser Buddy web extension. We detail these requirements with the intent of effectively communicating the needs our team will meet, and any third party should gather any information necessary to understand the project fully.

Moreover, this document does not provide scheduling, budgeting, or developing plans; these are detailed elsewhere. Additional details about the plan of this project are found in Appendix A.

Browser Buddy is a web extension that uses artificial intelligence to summarize web pages from *wikipedia.com*.

1.2 Project Goals

The team has identified three major goals for this project:

1. Provide a seamless user experience that works across Firefox web browsers.
2. Require no additional setup from the user upon installation.
3. Guarantee reliable summarizations of user-provided text data.

1.3 Scope

The objective of this project is to deliver a web browser extension for the summarization of web pages—specifically web pages from *wikipedia.com*.

Our system will require no databasing, as there is no planned functionality for saving user data. Browser Buddy will be installed by the user onto any Firefox web browser, and the user will access the functionality through the context menu.

Users will employ Browser Buddy to send data to and from ChatGPT-4 using the servers from OpenAI; the data sent will be user-specified, raw text data, and the data received will be a summary of that raw text data.

1.4 Definitions

- Browser extension - Software that extends the functionality of a web browser.
- API – Application programming interface.
- OpenAPI – producer of the GPT-4 Large Language Model (LLM).
- The project, the system, the extension, or the program– refers to Browser Buddy as a whole.

1.5 Assumptions

It is assumed that the user has the skills necessary to browse the web and access the website *wikipedia.com*. Additionally, it is assumed that the servers from OpenAI remain online and accessible for nearly 100% of execution. The absence of these servers would diminish any functionality the extension has.

2. Design Constraints

Contained in this section are constraints our team is placing upon the development of Browser Buddy.

2.1 Environment

Browser Buddy will work as a browser extension within the Firefox desktop browser on Windows and MacOS devices. Internet connection is a requirement due to the nature of the plugin.

2.2 User Characteristics

The target users are students, researchers, professionals or anyone who frequently reads lengthy online content. Users should have basic web browsing literacy and English reading/writing skills.

2.3 System

Browser Buddy will consist of a browser extension frontend using HTML/CSS/JavaScript that uses the WebExtensions API which is shared between Chrome and Firefox, though the main development goal is for a Firefox-specific extension. The stated technologies will be used as a front-end interface for our API connection with OpenAI's GPT-4 model.

3. Functional Requirements

All functional requirements that this system will implement are detailed below. This will include a description of every function and the level at which they are present.

3.1 Users

There is no plan to implement administrative accounts, as we feel this function is unnecessary considering there are no databases, transactions, or logs to be viewed or recorded. Appropriately, every user of the service will be treated the same.

Users to the service will receive the following functions:

3.1.1 Install/Uninstall

Clearly, users must possess the ability to add or remove the program from their browser as desired. The absence of this function would be unethical. Fortunately, our team will have no need to develop such a feature, as this is native to most modern web browsers. For our purposes, Firefox does possess this feature.

3.1.2 Text Highlighting and Context Menu

Browser Buddy is intended to be used within the context menu. The context menu is the pop-up that appears after right-clicking elements within the web browser—again, this is present in most modern web browsers. A simple button will be added into the native context menu, and the button will be appropriately labeled so that the user will understand its purpose.

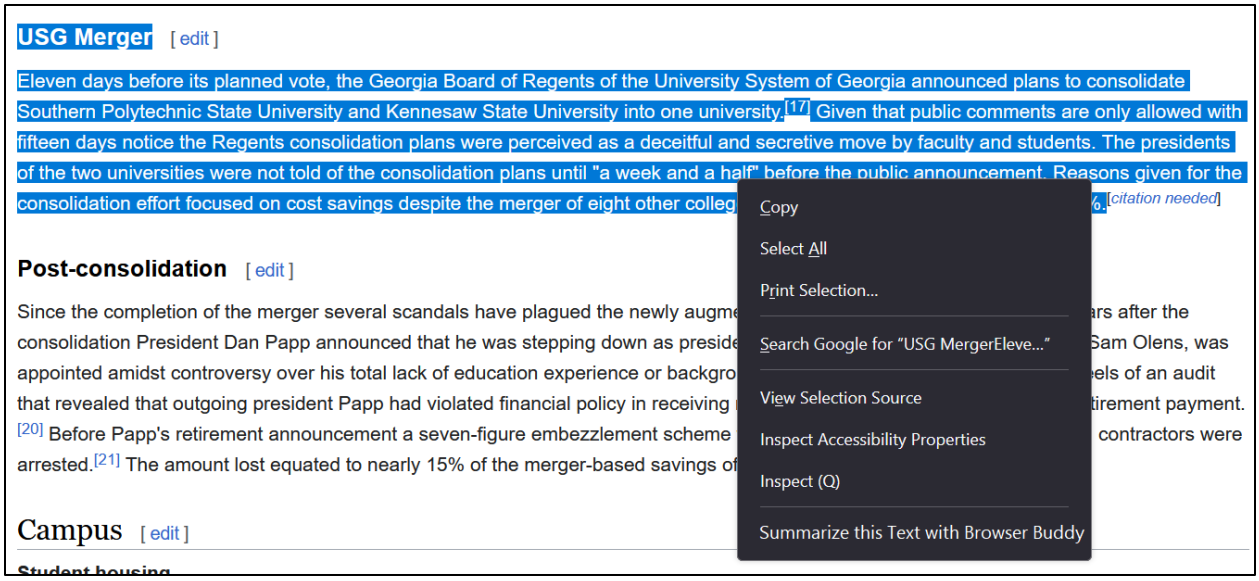

Figure 1: Mockup of Wikipedia page with Context Menu

Moreover, this button will become available after highlighting text within the webpage. It is this text that the extension will send through API calls and retrieve a summary from. Figure 1 depicts what we expect this extension to look like using the Wikipedia page for Southern Polytechnic State University [1].

3.1.3 Summary Window

Upon selecting the button outlined in Section 3.1.2, the summarized text will be displayed to a new pop-up window near the top of the user's screen. Figure 2 provides a mockup of this implementation. This window will feature a text box that holds the summarized text, and the box will disappear upon the user clicking any other element on the page.

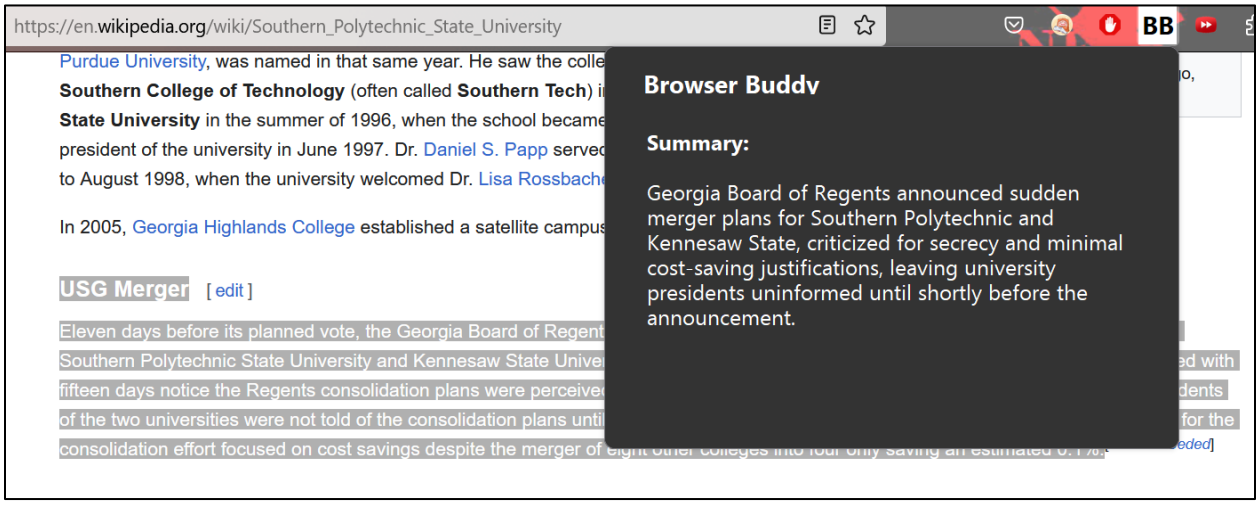

Figure 2: Mockup of Summary Window

3.2 Backend

Backend refers to any functions the user does not see or interact with. Of course, the user interacts with features of the GUI, but the functions described here are required for the APIs to function properly.

3.2.1 Obtain API Key

Every instance of the program will require a key to communicate with OpenAI's API. Keys exist so that only verified users have access to proprietary functions defined by companies, organizations, or institutions—such is the case here. A key will be obtained before the program attempts to call the API after a predefined user input is received.

3.2.2 Communicate with OpenAI

Browser Buddy sends highlighted user input to OpenAI so that a summarized text may be returned.

4. Nonfunctional Requirements

This section lists arbitrary attributes of our program.

4.1 Security

Users should not fear a breach of personal information or loss of their private key.

4.2 Capacity

As this extension relies upon the availability of the OpenAI servers, the capacity for users to send/receive data is limited to various features like server downtime, Internet speed, location, etc. Users should not expect to wait for a request to be filled.

Moreover, the local speed of the program should not be an issue. Observing the scope of this project, there is no need for memory-intensive or processor-intensive algorithms or tasks; therefore, this program should not require more than the minimum requirements for running just the browser.

4.3 Usability

The extension should be usable and understandable to most users. We intend on implementing proper UIE techniques to ensure that our target demographic will require the least amount of assistance or training possible.

4.4 Other

Depending on how OpenAI marshals the use of their API, we may or may not need to modify the code of this project so that future users can use the more recent versions of the API. Our maintenance should be very minimal, and it must be noted that every call to the API costs $0.001. Since this project is to remain within the realm of KSU, this cost should be negligible.

5. External Interface Requirements

5.1 User Interface

Users will interact with the software via their web browser. Due to the high variance in web browser layouts and designs, we are designing a GUI that pairs well with the default Firefox web browser layout. Browser Buddy's GUI does not require too much attention either; a simple popup window does not impact the user's overall interaction with their web browser.

5.2 Software Interface

Our application is run exclusively through Mozilla Firefox. Therefore, our application needs to use Firefox's appropriate extensions for usage of extensions.

5.3 API Interface

To communicate with OpenAI's ChatGPT-4 model, we require an interface for sending and receiving data. Again, this will be configured to work with Mozilla Firefox.

6. Testing Plan

Due to the nature of this project, the extent of the testing routine should be as complex as testing every user story. Again, the lack of a database and lack of different levels of users does not open many opportunities to test the project outside of its required functionality.

Of course, we intend on reinforcing all the requirements. Given there is enough time and resources remaining, we may modify the source of our code to allow for faster execution times and faster communication. Since our project heavily relies on the use of an external API, the speed we will observe should be reliant on various factors outside the control of the program itself—such as internet speed, server uptime, etc.

The results of this testing will be compiled and reported towards the end of development.

Appendix A: Project Plan

1. Overview

1.1 Background

In today's world of information overload, people need tools to distill online content quickly into concise, digestible summaries. Our team aims to develop an intuitive browser extension that helps users grasp the key points of a large webpage in seconds.

1.2 Objectives

The following are the team's objectives:

1. Develop a Firefox extension using the WebExtensions API.
2. Integrate machine learning capabilities by interfacing with OpenAI's GPT-4 model via their private API.
3. At the click of a context-menu button, summarize webpages into smaller & more readable paragraphs and bulleted lists.
4. Summaries must be clean, visually appealing, and built for readability and accessibility first.
5. Provide users with customization options such as summary length or text-to-speech.

1.3 Scope

The scope of the project will be as follows:

1. A desktop Firefox extension for MacOS and Windows.
2. Integration with OpenAI's GPT-4 model for robust summarization capability.
3. An intuitive browser pop-up window for summary viewing and controls.
4. Summarization of text-heavy webpage sections

2. Deliverables

2.1 Requirements

- Functional requirements
- Non-functional requirements
- User stories
- Data requirements

2.2 Design Plan

- UI/UX specification
- Interaction flow diagram
- Mockup
- Style guide

- Class specification

2.3 Prototype

- Basic extension with core MVP features
- Demonstration with feedback

2.4 Documentation

- Source code w/ comments
- Testing plans and results
- User manuals
- Final project report

3. Milestones

Milestone	**Notes**	**Due date**
Requirements & Design Document	Requirements for the project. Design of the project.	Feb. 13th
Website summarization achieved	Fetch from openAI's API with a webpage's contents and display a response.	Feb. 24th
Prototype & Presentation	Extension which has a button that sends webpage data to openAI and relays a response with functioning UI.	Mar. 4th
Minimum Viable Product	All functional requirements accomplished.	Mar. 31st
Final Report Draft	Draft of the final report. Will be nearing the final build of the project.	Apr. 6th
Final Project Package	Compilation of all previous work, including the final product.	Apr. 21st

4. Scheduled Meetings

Our team is scheduled to have an in-person meeting every Thursday from 2:00pm to 4:30pm. Every other day is intended to be a workday—except for Saturday, which will be a swing-day. This setup for meetings should give us ample time to discuss our progress throughout the previous and upcoming week. Of course, virtual meetings in Teams will be scheduled as required.

5. Gantt

We have planned 194 work hours. This, of course, may be inaccurate, but is appropriate considering the length of time necessary for every listed task.

Project Name:	SP-17 Browser Plugin																													
Report Date:	1/30/2024																													
					Requirements & Design				*Website Summarization*				*Prototype & Presentation*			*Minimum Viable Product*				*Final Report Draft*			*Final Report Package*			
Phase	**Tasks**	**Complete%**	**Current Status Memo**	**Assigned To**	02/01	02/06	02/08	02/13	02/15	02/20	02/22	02/24	02/29	03/02	03/04	03/07	03/14	03/21	03/31	04/02	04/04	04/06	04/09	04/16	04/18	04/21
	[illegible]	0%		[illegible]																						
	Data requirements	0%		Jacob, Elliott			3	4																		
Project design	UI/UX specification	0%		Jacob				2	3	2																
	Interaction flow diagram	0%		Jacob					2	2																
	Class specification	0%		Jacob, Elliott						4	4															
	Style guide	0%		Elliott, Nathan						4	3	1														
	Full mockup	0%		Team						5	6	5														
Development	Design prototype	0%		Jacob, Nathan							6	9	9	7												
	Prepare prototype presentation	0%		Jacob, Elliott								5	8	6	4											
	Rework requirements	0%		Nathan											3	2	3									
	Document updated design	0%		Nathan, Elliott												4	4									
	Prepare MVP	0%		Team											5	8	8	10	7							
Final product	Test and debug MVP	0%		Nathan, Jacob														6	6	2						
	Touch up MVP	0%		Jacob, Elliott															2	1						
	Prepare final product	0%		Team															3	6	8	4	3			
Final report	Compile final report	0%		Team																			10	8	3	
	Prepare final product presentation	0%		Jacob																				3	3	
	Assmble repository and deliverables	0%		Jacob																					2	4
			Total work hours	194	2	8	10	8	5	17	19	20	17	13	12	14	15	16	18	9	8	4	13	11	8	4
	* formally define how you will develop this project including source code management																									

Legend	
Planned	194
Delayed	0

6. Risk Assessment

Outlaid here are some risks associated with the projects corresponding mitigation efforts.

1. Slow API performance

a. Cache summaries to limit API calls & cost.
b. Optimize preprocessing to lower token counts.
c. Test on a few machines to identify issues with performance.

2. OpenAI usage limits

a. Explore using alternative models during downtime.
b. Provide a warning to users about throttling.

3. Privacy concerns

a. Limit data collection to bare minimum for basic functionality.
b. Only store user information such as cached summaries locally.

7. Version Control

Git through GitHub hosted by an organizational account, all members have equal authorization. We will maintain proper version control techniques—every new feature to the project will be placed in its own branch. Proper documentation will be created as well.

Appendix B: Sources

[1] Wikipedia contributors. (2023, October 26). Southern Polytechnic State University. In *Wikipedia, The Free Encyclopedia*. Retrieved 19:35, February 9, 2024, from https://en.wikipedia.org/w/index.php?title=Southern_Polytechnic_State_University&oldid=1181918685