National College of Ireland

BSc in Computing (Software Development)

2019/2020

123 Navigation

Technical Report

Wayne Hartigan

x16348136



[Executive Summary 4](#_Toc40023878)

[Introduction 5](#_Toc40023879)

[Background 6](#_Toc40023880)

[Technical Approach 7](#_Toc40023881)

[Aims 8](#_Toc40023882)

[Technologies 8](#_Toc40023883)

[Evaluation 9](#_Toc40023884)

[User Requirements Definition 10](#_Toc40023885)

[Intuitive 10](#_Toc40023886)

[Understanding 10](#_Toc40023887)

[Beneficial 10](#_Toc40023888)

[Reliable & Consistent 10](#_Toc40023889)

[Cross-Platform 10](#_Toc40023890)

[Requirements Specification 11](#_Toc40023891)

[Functional requirements 11](#_Toc40023892)

[Requirement 1: Installation 11](#_Toc40023893)

[Requirement 2: Giving Microphone Permission 12](#_Toc40023894)

[Requirement 3: Select Webpage Element 13](#_Toc40023895)

[Requirement 4: Google Search 15](#_Toc40023896)

[Requirement 5: Refresh Page 16](#_Toc40023897)

[Requirement 6: Go Back 17](#_Toc40023898)

[Requirement 7: Go Forward 18](#_Toc40023899)

[Requirement 8: Scroll Up 19](#_Toc40023900)

[Requirement 9: Scroll Down 20](#_Toc40023901)

[Requirement 10: Scroll Top 21](#_Toc40023902)

[Requirement 11: Scroll Bottom 22](#_Toc40023903)

[Requirement 12: Mic Not Working 23](#_Toc40023904)

[Non-Functional Requirements 24](#_Toc40023905)

[Performance/Response time requirement 24](#_Toc40023906)

[Availability requirement 24](#_Toc40023907)

[Recovery requirement 24](#_Toc40023908)

[Security requirement 24](#_Toc40023909)

[Reliability requirement 25](#_Toc40023910)

[Portability requirement 25](#_Toc40023911)

[Reusability requirement 25](#_Toc40023912)

[Resource utilization requirement 25](#_Toc40023913)

[Interface requirements 26](#_Toc40023914)

[Overview 26](#_Toc40023915)

[Design Choices 26](#_Toc40023916)

[GUI 27](#_Toc40023917)

[Implementation 31](#_Toc40023918)

[Chrome Extension Manifest 32](#_Toc40023919)

[Background.js & Content.js 33](#_Toc40023920)

[WebkitSpeechRecognition 35](#_Toc40023921)

[Permissions 38](#_Toc40023922)

[Google Chrome Web Store 40](#_Toc40023923)

[Testing 41](#_Toc40023924)

[User Testing 41](#_Toc40023925)

[System Evolution 62](#_Toc40023926)

[Project Proposal 63](#_Toc40023927)

[Objectives 64](#_Toc40023928)

[Background 64](#_Toc40023929)

[Technical Approach 65](#_Toc40023930)

[Special resources required 65](#_Toc40023931)

[Project Plan 65](#_Toc40023932)

[Technical Details 65](#_Toc40023933)

[Evaluation 66](#_Toc40023934)

[Reflective Journals 67](#_Toc40023935)

[Bibliography 72](#_Toc40023936)

# Executive Summary

The purpose of this report is to outline the aspects of Wayne Hartigan’s final year project for The National College of Ireland. The fundamental idea behind this project is to create a Google Chrome Extension that would allow a user to navigate throughout Google Chrome without the use of a mouse or trackpad but instead with their voice. This project is called “123 Navigation”. This project aims to seamlessly allow users to navigate throughout Google Chrome using voice commands. The main command “Navigation” will create a small box beside each link, button and image on a webpage with a number inside, the user can then select which link, button or image they wish to click by saying the number in the box beside it. Other commands that will be incorporated will include “Scroll Up”, “Scroll Down”, “Go Back” and “Go Forward” to allow the user a much easier and immersive experience whilst using 123 Navigation.

This idea will utilize the Google Chrome extension’s developer framework and Google’s speech recognition API “WebkitSpeechRecognition” and will be deployed to the Google web store.

This project will incorporate multiple languages, mainly developed in JavaScript, HTML, CSS and JSON.

The main aim of this project is to help users seamlessly and intuitively control Google Chrome. This project’s target audience will be mainly people with low levels of motor skills but will not be limited to this audience as the purpose of this extension could be used for multiple applications such as a hands free ticket kiosk, for people who do not wish to touch public surfaces or a lecturer wishing to control their PC as they walk through their classroom. 123 Navigation could have endless use cases.

This report will explain the concept of 123 Navigation, how the technologies needed have been implemented and how the final product has been developed.

# Introduction

123 Navigation is a Google Chrome extension designed to allow users to navigate throughout Google Chrome with their voice.

123 navigation will use a wake word to begin the system, once the user says the designated wake word, the navigation icons will appear on screen and the system will begin listening for the user’s response.

“Navigation Icons” are small icons with a unique number value in each of them, that will be displayed beside each link, button or image on the webpage the user is currently on. Using these icons, the user will intuitively be able to select the link in which they would like to click by saying the number value within the icon.

I believe 123 Navigation will alleviate a lot of stress for users that struggle to use a mouse or trackpad by allowing them to identify and select an item on the webpage without having to touch their mouse or keyboard. 123 Navigation can appeal to many different users, not just people with disabilities, 123 Navigation could appeal to any user that wishes to use Google Chrome without having to use their mouse or trackpad.

As 123 Navigation is a Google Chrome extension, it will immediately become accessible to any person that has Google Chrome installed, allowing users to install 123 Navigation with very little hassle.

## Background

Throughout my lifetime my mother has always been disabled, she suffers from multiple sclerosis (MS), a progressive, immune-mediated disorder which turns the host’s immune system against itself. Due to this illness my mother’s motor-skills have greatly deteriorated, making day to day tasks such as browsing the internet very difficult. This has inspired me to create *123 Navigation.* I have always felt the need to find anything to help my mother out, whether through software, hardware, or gadgets to make her life a little bit easier but one thing that has always fallen short is the software.

I have tried on multiple occasion to install different applications, chrome extensions, etc. to my mother’s computer to help her use her computer more efficiently but they all are either too slow, too complex or too expensive and require external hardware.

I tried a Chrome Extension before that was designed to navigate Chrome with speech but it was not easy, the user had to either hold the control key to speak or have the voice recognition listening all the time, which would cause problems with general speech being picked up and navigating. The navigation commands were generic “Scroll”, “Click”, etc. but there was no clear guide to what to say and the command had to be perfect or it would not pick it up. Creating a lot of confusion and problems when trying to use it.

After that I decided I would create *123 Navigation,* a more user friendly and simple to use voice navigation system by giving the user a “wake word” which will eliminate the need to press a key to have the system listen or have the voice system listening constantly, which will make people feel more comfortable using the extension. Also, by displaying a navigation icon on each link or button on the webpage the user is currently on, the user will then say the number value in the navigation icon next to the page element they wish to click and *123 Navigation* will select it for the user. I feel that this simple design, with clear instructions and easy integration to any Google Chrome could really help not just my mother but anybody else looking to control their browser through voice commands or hands free.

## Technical Approach

A Google Chrome extension is made three main languages, HTML, JavaScript and JSON, with CSS used change the presentation of the elements in the HTML page.

The HTML pages are displayed to the user in the drop-down menu on the extension’s toolbar, this typically holds an options menu or other functionality that the extension does not automatically do.

The JSON is used in the Manifest.json file, this file tells Google Chrome all the details of the extension such as name, icon image, permissions needed, all the background and content JavaScript files, what to load when the browser opens etc.

JavaScript will be used to give all added functionality to the extension. The JavaScript files are split into two categories, the “context.js” and the “background.js”. The “context.js” file is the JavaScript file used to connect the browser to the extension, this file can access the DOM of the webpage and browser console. The “background.js” file will handle the heavy functionality, in the case of 123 Navigation, it will handle the processing of the voice input, it will then pass a message to the “context.js” file telling it the specific message, it will then make changes to the DOM by inserting the navigation icons or clicking the element chosen by the user.

An external JavaScript file will also be invoked during the installation of 123 Navigation to provoke Google’s microphone permissions to allow the extension to use the user’s microphone, with their permission.

Google Chrome extensions are tested by uploading an “unpacked package” containing all the aforementioned files and then running the extension like any other extension you download. The Google Chrome extensions section the settings menu offers a great debugging service that allows you to make changes and refresh without stopping your extension.

To publish a Google Chrome extension, it is as simple as zipping your files, creating a developer account and uploading your app. This simple process allows 123 Navigation to reach a wide market very quick. There is also a section to select payment methods if you choose to charge for your app.

The entire development of 123 Navigation will be stored on GitHub in a public repository. GitHub will aid in version control and branches to allow me to make constant changes without the worry of breaking a previous component.

## Aims

* The system must consistently and responsively understand when the user has said the wake-word.
* The system must consistently and responsively display the navigation icons, with a unique number value in each icon, to the user.
* The navigation icons must display next to all elements of the page that can be clicked.
* The system must reliably understand the user’s input for the navigation icon.
* The system must allow the user to cancel the navigation icons by with saying “cancel”, clicking on the screen or scrolling the page.
* The system must reliably click the selected element designated by the user.
* The system must then return to stand-by mode awaiting the wake word again.

## Technologies

* JavaScript will be used in the backend and front end of 123 Navigation. JavaScript is an object-orientated programming language that is one of the main three languages in web-development alongside HTML and CSS, both of which will be used in 123 Navigation. JavaScript can be used to update both CSS and HTML making it very useful for 123 Navigation as it will be needed to add the navigation icons to the webpage. JavaScript is also very useful for its ability to take voice as an input, this will be main driving force the backend. (MDN contributors, 2019)
* WebkitSpeechRecognition will be the API used to retrieve and understand the user’s voice input. WebkitSpeechRecognition is a JavaScript API designed for use in Google Chrome that allows control and flexibility over speech recognition in Google Chrome webpages and extensions. (Shires, 2013)
* HTML will be used as the front end for the webpages attached to the Chrome Extension, such as the home page that will be displayed upon install which will prompt the user to give 123 Navigation access to their microphone. HTML (Hyper Text Mark-up Language) is the standard mark-up language for webpages and documents that are designed to be displayed in a web browser. (W3Schools, 2018)
* CSS will be used to style the navigation icons and webpages attached to 123 Navigation. CSS (Cascading Style Sheets) is a style sheet language used for editing and describing the presentation of a webpage and it’s elements. (W3 Schools, 2018)
* JSON will be used to link 123 Navigation to Google Chrome by giving important details such as scripts, name, icon etc. JSON (JavaScript Object Notation) is a lightweight format for storing and transporting data. (W3 Schools, 2018)

***Testing***

Testing will take place both internally and externally. Through extensive testing on multiple machines, multiple microphones and multiple versions of Google Chrome, this will ensure the most reliability for 123 Navigation and live up to the promise of staying fast, reliable and easy to use.

User testing will also take place near the end of development to ensure users can use the extension, that it is fast and reliable and works with multiple voices and accents. A Google survey will be handed out to each of these testers to find out how their experience was as well as any changes or add-ons they suggest.

***Deployment***

Once 123 Navigation has been tested and development has completed the extension will be hosted on the Google Chrome Web Store. The Google Chrome Web Store is an online store in which users can install Google applications, extensions and themes to their Google Chrome browser.

***Special resources required***

Although all speech recognition software requires a microphone and will recommend the use of an external microphone or headset 123 Navigation will not require any external resources such as microphones. 123 Navigation will work will all integrated microphones in laptops and webcams.

## Evaluation

As development for 123 Navigation progresses, JUnit tests will be implemented using Jest or Mocha to ensure developing further does not break any previous work, integration tests will also be implemented for extra precaution.

Physical testing will be complete once a user can successfully navigate throughout chrome using 123 Navigation.

# User Requirements Definition

From the user’s perspective, there are several requirements that the user would like to see implemented.

The following definitions explain the requirements requested:

## Intuitive

The extension must be intuitive to use and incorporate an easy to understand workflow. For example, the “wake word” used to begin the extension must not be a difficult word to pronounce, the colour scheme must be easy to read, with a font colour that stands out from the background colour. The application must feel comfortable and natural for the user to use. (WebDesignRankins, 2019)

## Understanding

The extension must have the ability to understand the user’s voice inputs, this will include a range of voice frequency’s, accents and speed. For the user to fully feel comfortable using 123 Navigation, they must not have to fight with the voice recognition, the ability for the extension to understand the user’s voice input is a must.

## Beneficial

123 Navigation must be beneficial to use over a mouse or trackpad. The user must feel that using 123 Navigation helps them navigate throughout Google Chrome and does not dampen the experience compared to using traditional navigation methods.

## Reliable & Consistent

123 Navigation must always work. The extension must reliably work while the user is navigating throughout Google Chrome. The extension must not crash or stop working before, during or after the user is navigating. 123 Navigation must work consistently on all web pages. The commands must stay the same on each website and each time the user navigates throughout Google Chrome using 123 Navigation.

## Cross-Platform

123 Navigation must run on all Operating Systems (non-mobile) and all Google Chrome applications on these Operating Systems. 123 Navigation must work with all types of microphones and must not require an external microphone, must work with integrated microphones.

# Requirements Specification

## Functional requirements

### Requirement 1: Installation

#### Description & Priority

123 Navigation must be installed through the Google Chrome Web Store. The requirement details the user installing 123 Navigation.

#### Use Case

**Scope**

The scope of “Installation” is to in install 123 Navigation from the Google Web.

**Description**

This requirement is describing the use case for the user installing 123 Navigation.

**Precondition**

Before 123 Navigation is installed it will be available on the Google Chrome Web Store.

**Activation**

The use case will begin once the user wishes to install 123 Navigation.

**Main flow**

1. The user navigates to <https://chrome.google.com/webstore/category/extensions>.
2. The user will fill the search bar with “123 Navigation”.
3. The user will click “Add to Chrome”.
4. The user will agree to permissions required.

**Alternate flow**

1. The user does clicks into “123 Navigation” after searching.
2. System will return to [M3].

**Post condition**

The system will now create a “Welcome” page that will invoke the permissions popup [Requirement 2].

### Requirement 2: Giving Microphone Permission

#### Description & Priority

The user’s microphone is required to use 123 Navigation. This requirement details the user giving 123 Navigation permission to use their PC’s microphone.

#### Use Case

**Scope**

The scope of “Giving Microphone Permission” is to allow 123 Navigation permission to use the user’s microphone.

**Description**

This requirement is describing the use case for the user permitting 123 Navigation to use their microphone.

**Precondition**

123 Navigation is installed, and user is on the “Welcome” page.

**Activation**

The use case will begin once the user installs 123 Navigation. Upon installation a webpage containing all information of 123 Navigation will open, this page will load the permission pop up for the microphone.

**Main flow**

1. The user will install 123 Navigation from the Google Web Store.
2. The “Welcome” page will open in the user’s browser.
3. A popup requesting permission to use the user’s microphone will appear.
4. The user will give permission to 123 Navigation to use their microphone.

**Alternate flow 1**

1. A popup requesting permission to use the user’s microphone will appear.
2. The user blocks permission.
3. The user follows the on-screen instructions
4. The user will click the video icon beside the URL.
5. The user will click “allow” in the drop-down menu.
6. The user refreshes the page.
7. System returns to [M4]

**Alternate flow 2**

1. A popup requesting permission to use the user’s microphone does not appear.
2. The user follows the on-screen instructions
3. The user will click the video icon beside the URL.
4. The user will click “allow” in the drop-down menu.
5. The user refreshes the page.
6. System returns to [M4]

**Post condition**

The system will now go into its stand-by mode, in which it is now waiting for the user to say the “wake word” to activate the navigation.

### Requirement 3: Select Webpage Element

#### Description & Priority

The user now has successfully installed and given 123 Navigation permission to use their microphone and can now begin navigating using the “wake word”.

#### Use Case

**Scope**

The scope of “Select Webpage Element” is say the “wake word” and to select the webpage element the user desires.

**Description**

This requirement is describing the use case for the user using 123 Navigation to navigate Google Chrome.

**Precondition**

User is on the webpage they wish to select an element on.

**Activation**

The use case will begin once the user says the “wake word”.

**Main flow**

1. The user says the “wake word”
2. Navigation Icons are now beside each clickable element.
3. The user says the numeric value with the Navigation Icon next to the element they wish to select.
4. The selected element will then be clicked.
5. Systems returns to a Precondition state.

**Alternate flow 1**

1. The user says the “wake word”.
2. Navigation Icons are now beside each clickable element.
3. The user says a numeric value that does not match any elements.
4. The system returns to [M2].

**Alternate flow 2**

1. The user says the “wake word”.
2. Navigation Icons are now beside each clickable element.
3. The user says “Cancel”.
4. All Navigation Icons are removed.
5. Systems returns to a Precondition state.

**Alternate flow 3**

1. The user says the “wake word”.
2. Navigation Icons are now beside each clickable element.
3. The clicks their mouse.
4. All Navigation Icons are removed.
5. Systems returns to a Precondition state.

**Alternate flow 4**

1. The user says the “wake word”.
2. Navigation Icons are now beside each clickable element.
3. The clicks their mouse or scrolls down the page
4. All Navigation Icons are removed.
5. Systems returns to a Precondition state.

**Post condition**

The system will now go into its stand-by mode, in which it is now waiting for the user to say a command.

### Requirement 4: Google Search

#### Description & Priority

The user wishes to use the in-built search functionality to Google search with their voice.

#### Use Case

**Scope**

The scope of “Google Search” is to explain the work flow of the user using the “search for” command followed by their search term to do a google search of that search term.

**Description**

This requirement is describing the use case for the user using the “Search For” command to search Google.

**Precondition**

User is on any webpage on Google Chrome.

**Activation**

The use case will begin once the user says the “search for”.

**Main flow**

1. The user says the “search for” followed by their search term.
2. The page will load to Google search results with the user’s search term.
3. System will return to preconditional state.

**Alternate flow**

1. The user says the “search for” without saying a search term.
2. System will return to preconditional state.

**Post condition**

The system will now go into its stand-by mode, in which it is now waiting for the user to say a command.

### Requirement 5: Refresh Page

#### Description & Priority

The user wishes to refresh the page.

#### Use Case

**Scope**

The scope of “Refresh Page is to explain what happens when the user says, “Refresh Page”.

**Description**

This requirement is describing the use case for the user uses the “Refresh Page” command to refresh the current web page.

**Precondition**

User is on any webpage on Google Chrome.

**Activation**

The use case will begin once the user says the “Refresh Page”.

**Main flow**

1. The user says, “Refresh Page”.
2. The current web page will reload.

**Post condition**

The system will now go into its stand-by mode, in which it is now waiting for the user to say a command.

### Requirement 6: Go Back

#### Description & Priority

The user wishes to return to the previous web page.

#### Use Case

**Scope**

The scope of “Go Back is to explain what happens when the user says, “Go Back”.

**Description**

This requirement is describing the use case for the user uses the “Go Back” command to return to the previous web page.

**Precondition**

User is on any webpage on Google Chrome.

**Activation**

The use case will begin once the user says the “Go Back”.

**Main flow**

1. The user says, “Go Back”.
2. The user will be brought back to the previous page.

**Alternate flow**

1. The user says the “Go Back”.
2. There is no previous web page.
3. User stays on current web page.
4. System will return to preconditional state.

**Post condition**

The system will now go into its stand-by mode, in which it is now waiting for the user to say a command.

### Requirement 7: Go Forward

#### Description & Priority

The user wishes to go to the next web page.

#### Use Case

**Scope**

The scope of “Go Forward is to explain what happens when the user says, “Go Forward”.

**Description**

This requirement is describing the use case for the user uses the “Go Forward” command to go to the next web page.

**Precondition**

User is on any webpage on Google Chrome.

**Activation**

The use case will begin once the user says the “Go Forward”.

**Main flow**

1. The user says, “Go Forward”.
2. The user will be brought to the next page.

**Alternate flow**

1. The user says the “Go Forward.
2. There is no next web page.
3. User stays on current web page.
4. System will return to preconditional state.

**Post condition**

The system will now go into its stand-by mode, in which it is now waiting for the user to say a command.

### Requirement 8: Scroll Up

#### Description & Priority

The user wishes to scroll up the web page.

#### Use Case

**Scope**

The scope of “Scroll Up is to explain what happens when the user says, “Scroll Up”.

**Description**

This requirement is describing the use case for the user uses the “Scroll Up” command to scroll up the current web page.

**Precondition**

User is on any webpage on Google Chrome.

**Activation**

The use case will begin once the user says the “Scroll Up”.

**Main flow**

1. The user says, “Scroll Up”.
2. The web page will be scrolled up by 500 pixels.

**Alternate flow**

1. The user says the “Scroll Up”.
2. The web page is already at the top.
3. System will return to preconditional state.

**Post condition**

The system will now go into its stand-by mode, in which it is now waiting for the user to say a command.

### Requirement 9: Scroll Down

#### Description & Priority

The user wishes to scroll down the web page.

#### Use Case

**Scope**

The scope of “Scroll Down” is to explain what happens when the user says, “Scroll Down”.

**Description**

This requirement is describing the use case for the user uses the “Scroll Down” command to scroll down the current web page.

**Precondition**

User is on any webpage on Google Chrome.

**Activation**

The use case will begin once the user says the “Scroll Down”.

**Main flow**

1. The user says, “Scroll Down”.
2. The web page will be scrolled down by 500 pixels.

**Alternate flow**

1. The user says the “Scroll Down”.
2. The web page is already at the bottom.
3. System will return to preconditional state.

**Post condition**

The system will now go into its stand-by mode, in which it is now waiting for the user to say a command.

### Requirement 10: Scroll Top

#### Description & Priority

The user wishes to scroll to the top of the web page.

#### Use Case

**Scope**

The scope of “Scroll Down” is to explain what happens when the user says, “Scroll Top”.

**Description**

This requirement is describing the use case for the user uses the “Scroll Top” command to scroll to the top of the current web page.

**Precondition**

User is on any webpage on Google Chrome.

**Activation**

The use case will begin once the user says the “Scroll Down”.

**Main flow**

1. The user says, “Scroll Top”.
2. The web page will scroll to the top of the current web page.

**Alternate flow**

1. The user says the “Scroll Top”.
2. The web page is already at the top.
3. System will return to preconditional state.

**Post condition**

The system will now go into its stand-by mode, in which it is now waiting for the user to say a command.

### Requirement 11: Scroll Bottom

#### Description & Priority

The user wishes to scroll to the bottom of the web page.

#### Use Case

**Scope**

The scope of “Scroll Down” is to explain what happens when the user says, “Scroll Bottom”.

**Description**

This requirement is describing the use case for the user uses the “Scroll Bottom” command to scroll to the top of the current web page.

**Precondition**

User is on any webpage on Google Chrome.

**Activation**

The use case will begin once the user says the “Scroll Bottom”.

**Main flow**

1. The user says, “Scroll Bottom”.
2. The web page will scroll to the bottom of the current web page.

**Alternate flow**

1. The user says the “Scroll Bottom”.
2. The web page is already at the bottom.
3. System will return to preconditional state.

**Post condition**

The system will now go into its stand-by mode, in which it is now waiting for the user to say a command.

### Requirement 12: Mic Not Working

#### Description & Priority

The user is experiencing problems with their microphone not getting recognised and commands not getting completed.

#### Use Case

**Scope**

The scope of “Mic Not Working” is to explain what happens when is experiencing microphone issues and how to solve the issue.

**Description**

This requirement is describing the use case for when the user is experiencing microphone issues and how to solve the issue.

**Precondition**

User is experiencing microphone issues.

**Activation**

The use case will begin once the user is experiencing microphone issues.

**Main flow**

1. The user clicks the 123 Navigation icon in the extensions section of their browser’s toolbar.
2. The user clicks the “Mic not working?”.
3. The user follows the on-screen instructions.
4. The user refreshes the page.

**Post condition**

The system will now go into its stand-by mode, in which it is now waiting for the user to say a command.

## Non-Functional Requirements

### Performance/Response time requirement

123 Navigation is targeted towards people who normally struggle to navigate through google chrome, thus performance and response time is a key factor in the user’s experience. In a study conducted in 2018 by Dotcom-Monitor, even if a webpage takes between one and three seconds to load, on average the page has lost thirty two percent of all users travelling to the page by the time it loads (Dotcom Tools, 2018).

Thus, 123 Navigation needs to be as responsive as possible and have a load time of below one second.

My aim for the response time, between the user providing the wake word and “navigation icons” displaying, is five hundred milliseconds. This will hopefully provide the use without enough responsiveness that it will feel like an improvement on struggling with a mouse, trackpad or any other form on physical input.

### Availability requirement

123 Navigation is a Google Chrome extension; thus, it will be available to every computer that has Google Chrome version 25 and later installed.

An internet connection is required for 123 Navigation.

### Recovery requirement

As no data will be stored, a recovery system will not be needed.

If 123 Navigation became unresponsive, restarting the browser or the extension itself should fix the issue, no recovery plan is needed, and the extension will work as it did before

### Security requirement

123 Navigation is powered by voice controls; thus, it has access to the user’s microphone. The security requirement needed for 123 Navigation is to ensure the user, that no data will be stored, and all speech input will be purely used to navigate and control Google Chrome and not stored.

### Reliability requirement

Reliability is a massive requirement in 123 Navigation, as some users may rely on it as their only means of navigating through the internet.

123 Navigation must ensure that the user can consistently control the voice controls and that 123 Navigation consistently understands the user.

### Portability requirement

As 123 navigation is a Google Chrome extension, it will be available on all computers that have Google Chrome. Laptops will have the ability to use 123 Navigation anywhere they travel so long as it has an internet connection. 123 Navigation uses WebkitSpeechRecognition, which is a server sided speech processing API, thus an internet connection is required.

### Reusability requirement

123 Navigation must be reusable, if a user uses 123 Navigation to navigate to a webpage, they must have the ability to then navigate from the new page.

### Resource utilization requirement

The following resources will be required to run the system:

* Google Chrome v25 or higher
* A microphone, either built in or external.
* A desktop or laptop machine, mobile devices such as phones and tablets will not be compatible.
* An operating system that supports Google Chrome, i.e. Linux, MacOS or Windows, mobile operating systems such as Android, IOS, iPadOS etc. will not be compatible.

As Google Chrome is required to run 123 Navigation, the following system specs are the minimum recommended specs to ensure Google Chrome will run, based on operating systems (Google, 2019).

Windows:

* Windows 7, Windows 8, Windows 8.1, Windows 10 or later
* An Intel Pentium 4 processor or later that's SSE2 capable

MacOS

* OS X Yosemite 10.10 or later

Linux

* 64-bit Ubuntu 14.04+, Debian 8+, openSUSE 13.3+, or Fedora Linux 24+
* An Intel Pentium 4 processor or later that's SSE2 capable

# Interface requirements

## Overview

As 123 Navigation will rely solely on the user’s voice input and clear instructions on the screen the interface plays a vital role in the user’s experience.

The interface for 123 Navigation needs to be clear and precise with an intuitive feel for the user to have a comfortable and understanding experience.

All design and GUI will be designed purely with HTML and CSS and will be injected into a web page using JavaScript.

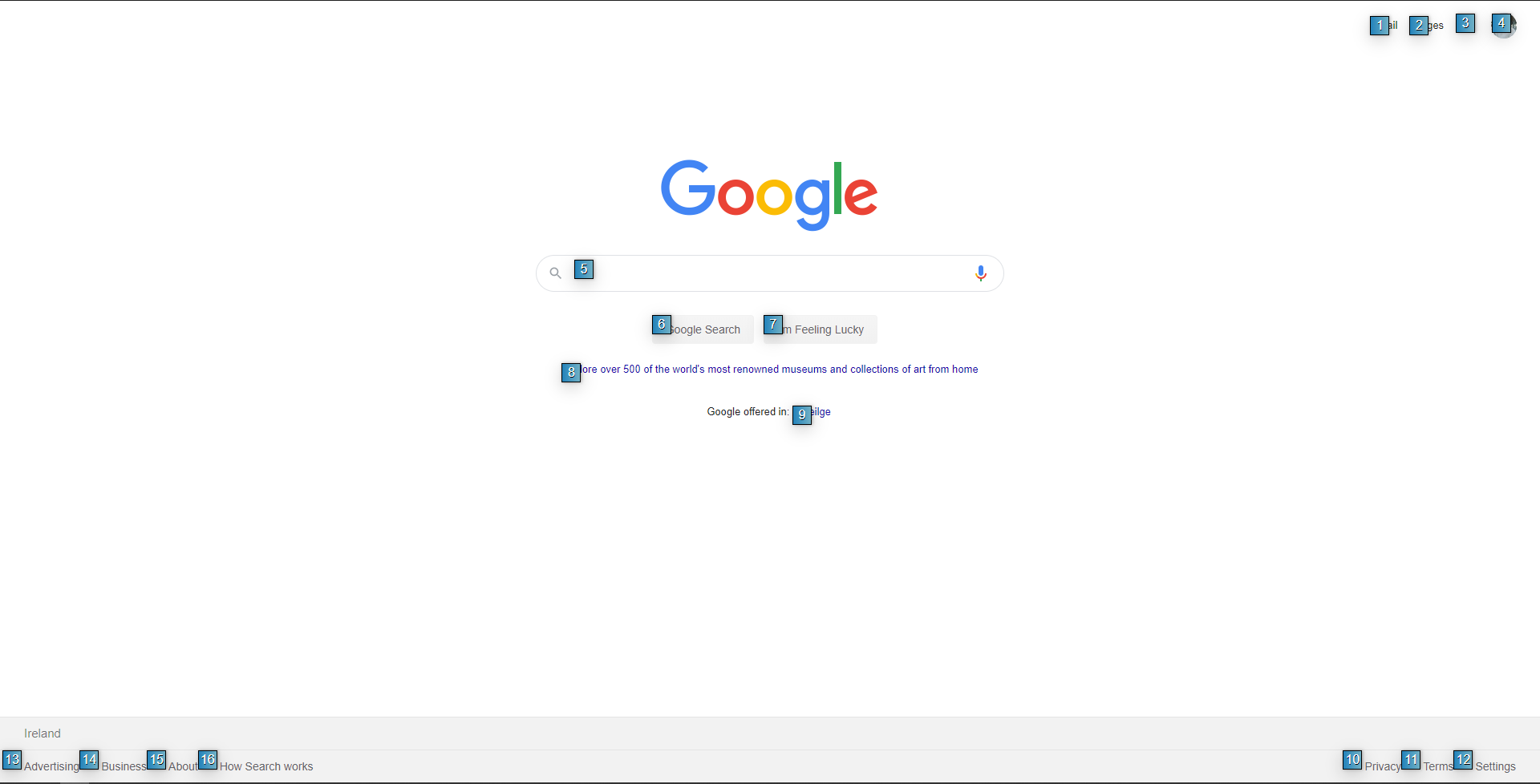
## Design Choices

For the navigation icons displayed next to each web page element, the design I went with was small box (22 x 22 pixels) with a blue gradient colour scheme and white background text. This colour scheme was chosen as it conveys clarity to the user whilst not blending it with any web page thus dampening the user’s experience.

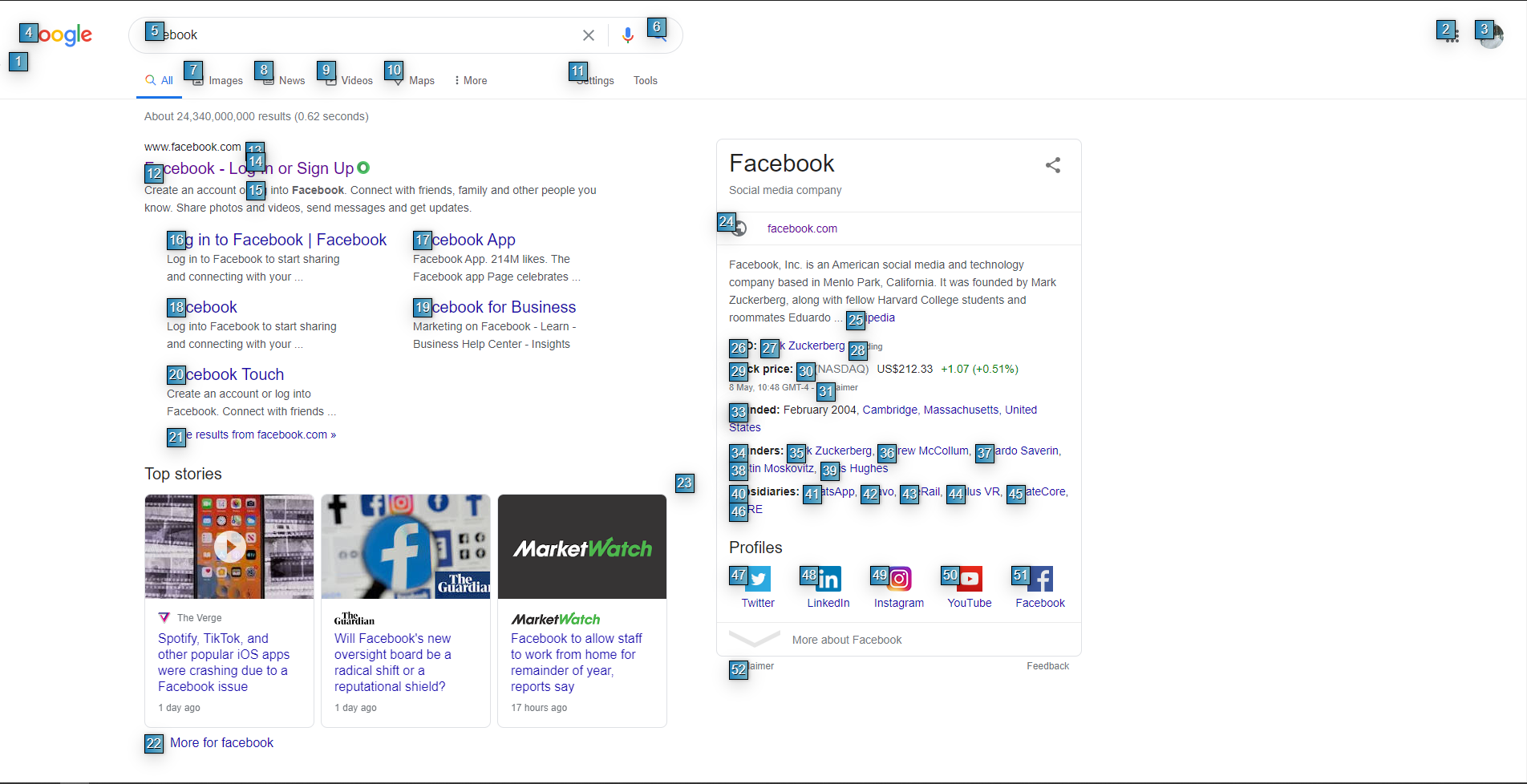
This design will make all naviagtion icons clear and easy to read whilst navigation throughout Google Chrome. The gradient colour scheme with a black border helps the navigation icon stand out from all web pages.

## GUI

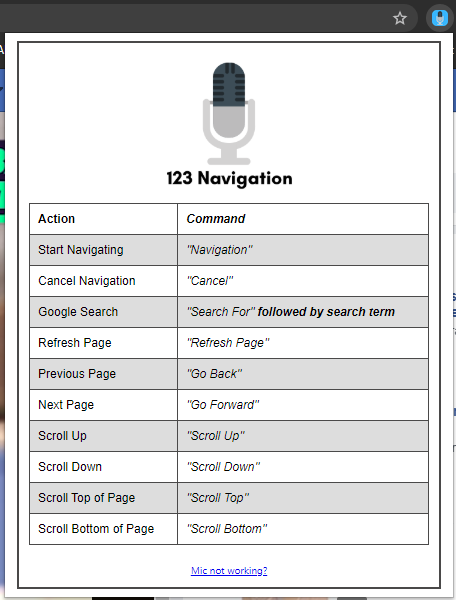


As mentioned in the previous section, the blue navigation icons with a gradient colour scheme and white numbers make it very clear and obvious which numbers match with which HTML element.

This layout is very intuitive and easy to follow and stays consistent throughout every web page.

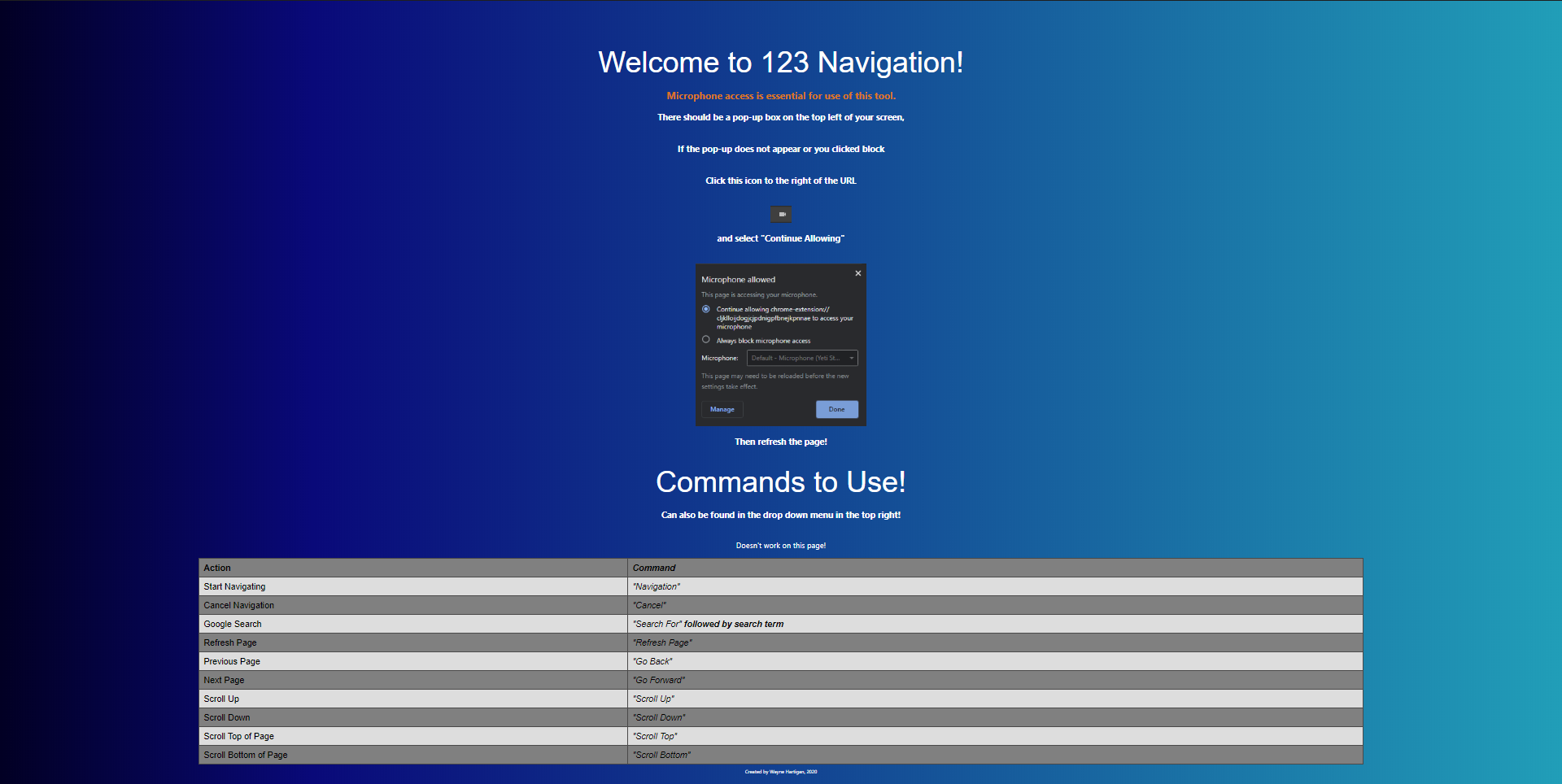


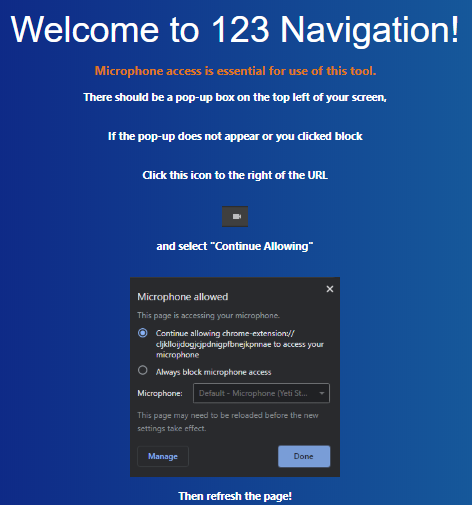
Although on some crowded web pages the navigation icons can become crowded and can overlap.

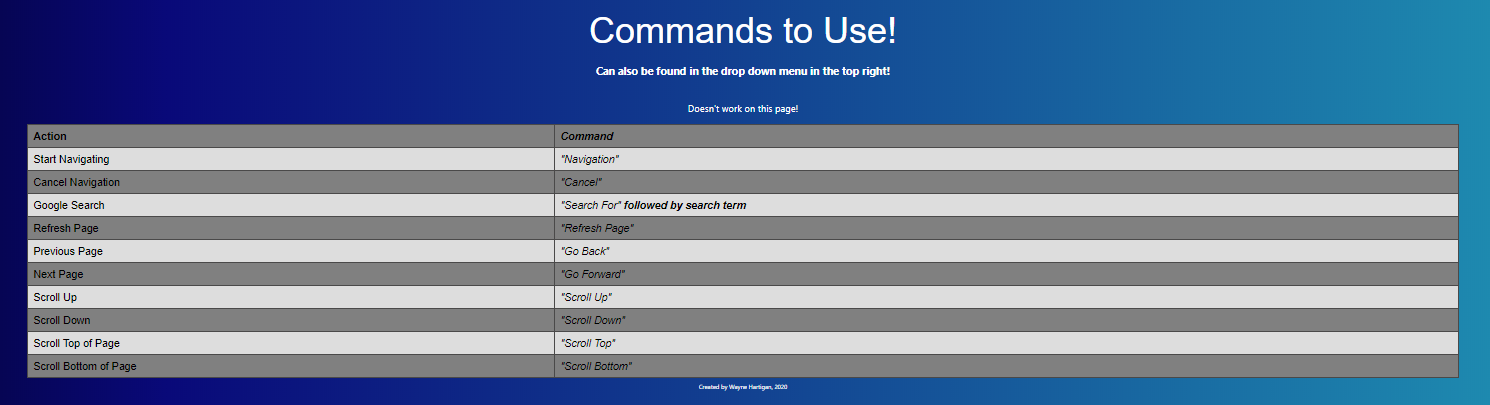


The dropdown menu in the Chrome menu was designed to clearly display to the user which commands are available and how to use them. A table format with an offset colour scheme was used to clearly display to the user which command to use for each action.

A “Mic not working?” link was added to return the user to the “welcome page” if they are experiencing microphone issues.

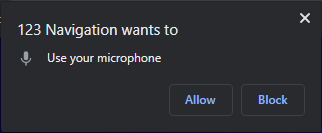






The welcome page is the page the user will be greeted with upon installation.

This page explains to the user that microphone permissions is needed. This page is also the page in which the microphone permission prompt will be provoked.



This page also explains what to do if the microphone permission does not appear or is blocked. These instructions are also the instructions used if the user is having microphone problems.

Below them instructions the user will also be greeted with all commands available in the same table format as the drop-down menu. Allowing the user to study the commands before using.

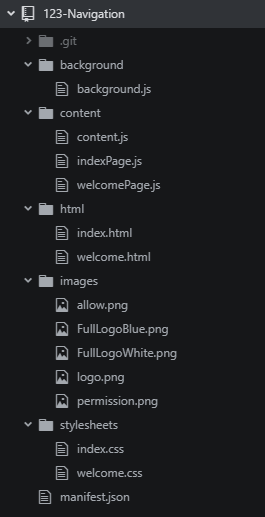
# Implementation

This section is to outline and explain the primary functions, the development process and all components that were needed to create these functions. As 123 Navigation was developed across the entire academic year the development process has changed and been modified many times.

The main sections of development are broken down into the following sections.

* Chrome Extension Manifest
* Background.js
* WebkitSpeechRecognition
* Permissions
* Google Chrome Web Store

Full Project Directory:



## Chrome Extension Manifest

The manifest.json file is a required file when developing a Google Chrome extension. This file describes all information needed for the extension. Including the name, version, browser actions, required permissions and which files are run by the extension itself.

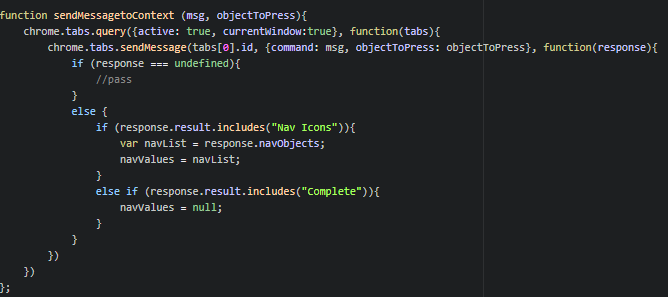


The manifest is also where the background script is provoked. This is the main JavaScript file that will run in the background of the extension.

## Background.js & Content.js

Chrome extension JavaScript files are split into two categories, background scripts and content scripts. The content script can interact with the DOM and all html elements, while the background script cannot. The WebkitSpeechRecognition API I am using does not work in the content script; thus, all speech recognition and processing is done within the background script.

The background script and content script cannot communicate each other as a normal JavaScript file could. Thus, I had to use a messaging service which send a request message to the background script and retrieves the response from the content script.



The above function would send a message to the current tab which it retrieves from the “tabs[0],id” and sends an object containing a “command” which is the message I would pass through, and an “objectToPress” which would pass through the element that the use has requested to press. I would call this function like so if there was no object to be pressed.



I would call this function like so if there was an object to be pressed.



Which passes in the element id, to be used in the content script. All code within the content script is within a message listener.

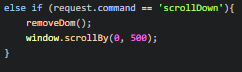


Which takes in the request sent by the background script. From this request I can then get the message that was sent across and check if there is an object to press. By calling “request.command” I can check in the content script which command the background script has sent.

For example, if the user has said “scroll down”, the background script will send a message to the content script with a message “scrollDown” and object to press set to null.



In the content script the request message is checked and the command is executed.



To return the response to the background script, a function is created which formats the response correctly.



Which can be called like as follows:



This tells the background script that the response has been completed.

This function can also be called if the navigation icons have been created, with a list of all the ids of the elements for the user to say.



## WebkitSpeechRecognition

WebkitSpeechRecognition is a server sided speech processing API I used to process the speech input of the user. As this API only works in the background.js file all configuration is within this file.

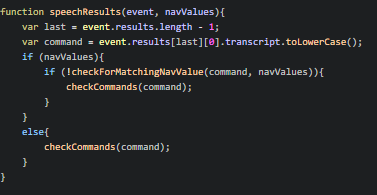


On line 2, the WebkitSpeechRecognition object is created. With the corresponding grammar list created on line 3.

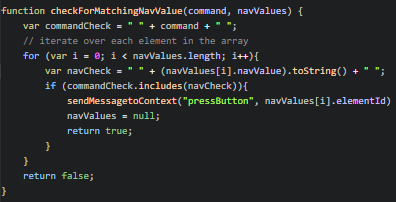
The “recognition.continuous” configuration is a setting that is to set the speech to never stop listening, and do not stop, however I had some problems with this function as whenever the system received an error it would stop, errors such as “no-speech” were a common occurrence, which is when the speech recognition received no input, thus breaking the recognition. A work around for that, that seems to work very consistently, is using the “recognition.onend” functionality which runs every time the speech recognition would stop. Now every time the recognition stops, I run “recognition.start()” which will start the recognition again. This work around seems to be the best system for having continuous listening.

The error handling as seen on line 17 has a small catch, if the error is the “no-speech” error, it will not print, as this is not a true error and just the recognition noting that it has not received input. All other errors will be logged.

The “recognition.onresult” functionality is a function that will run whenever the speech has received its input. This will then run the function “speechResults()” while passing in the event and the navValues list created at the top.

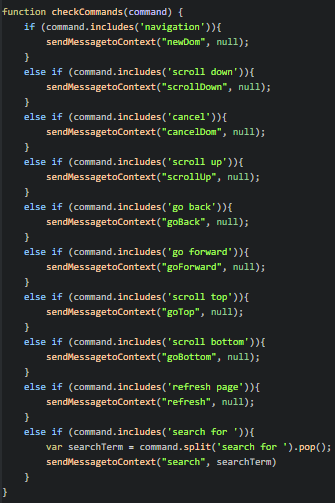


This function will change the command to a string and check for navValues that have been passed through the response of the content script. If there are navValues it will run the “checkForMatchingNavValue()” function.



This function will iterate throughout the voice input to see if it includes any of the nav values. If it does it will send a response to the content script to press that element and return true, or else it will return false.

If the function returns false, the next function of checking for other commands will run. This function will also run if there was no nav values.



This function includes all other commands, such as scroll up, down and search for.

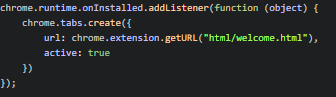
As WebkitSpeechRecognition is a server sided voice processing API, this functionality will not work offline. All voice processing is completed through Google Chrome itself. This API is also compatible with Mozilla Firefox, leaving the possibility of transferring this project to other platforms.

## Permissions

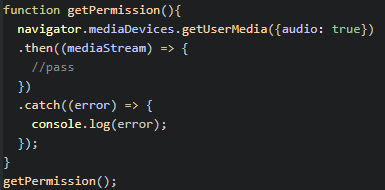
Microphone permissions are essential for 123 Navigation to work; however, Google Chrome does not natively allow microphone permissions to be used by Chrome Extensions. For normal permissions, they just need to be declared in the manifest file, but there is no option for microphone permissions.

To work around this and give permission to 123 Navigation to use the user’s microphone, any html page within 123 Navigation needed to receive permission. This would then allow the extension itself to use this permission.

When the user first installs 123 Navigation, the “welcome.html” page is loaded.



This page then provokes the “welcome.js” script.



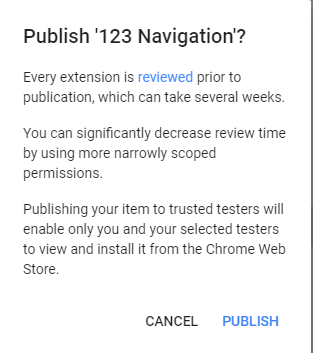
Which prompts the user with a popup to use their microphone. This function will then give 123 Navigation permission to use the use’s microphone.

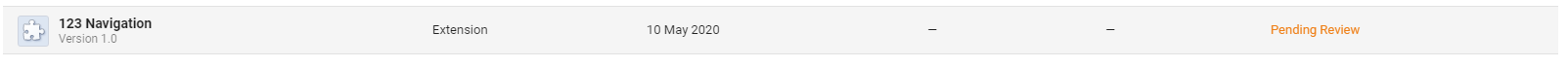
The html of the welcome page can be found below.



## Google Chrome Web Store

For deployment, Google Chrome extensions are deployed to the Google Chrome Web Store. However, Google review all extensions before they can be published, this review process can take up to several weeks, currently the review process has not completed for 123 Navigation.





When the review is complete, 123 Navigation will be deployed on the Google Chrome store, making the extension extremely accessible to anybody that wishes to use it.

# Testing

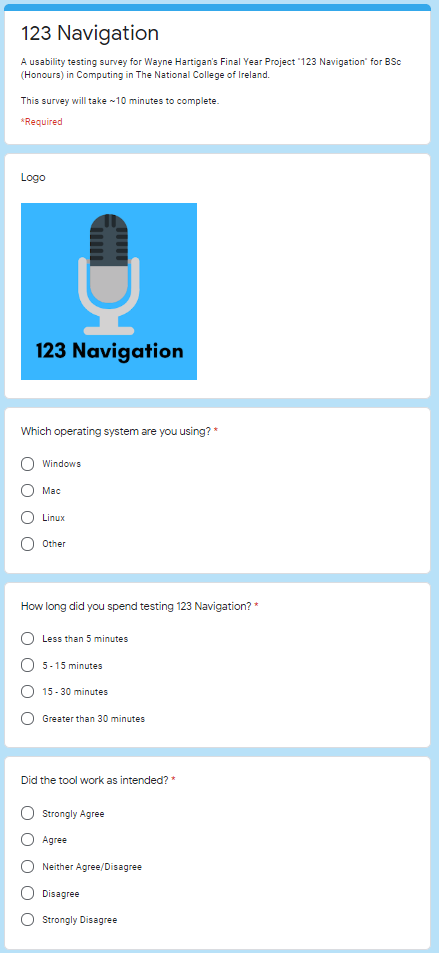
## User Testing

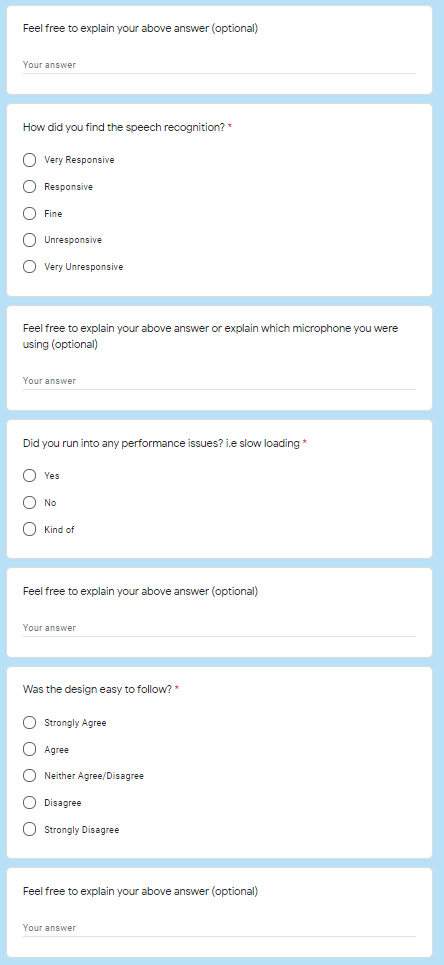
When 123 Navigation was at a working state, and the development had reached near the end, I handed the project out to different people. All people used their own machines and microphones. This ensured a wide range of testing environments.

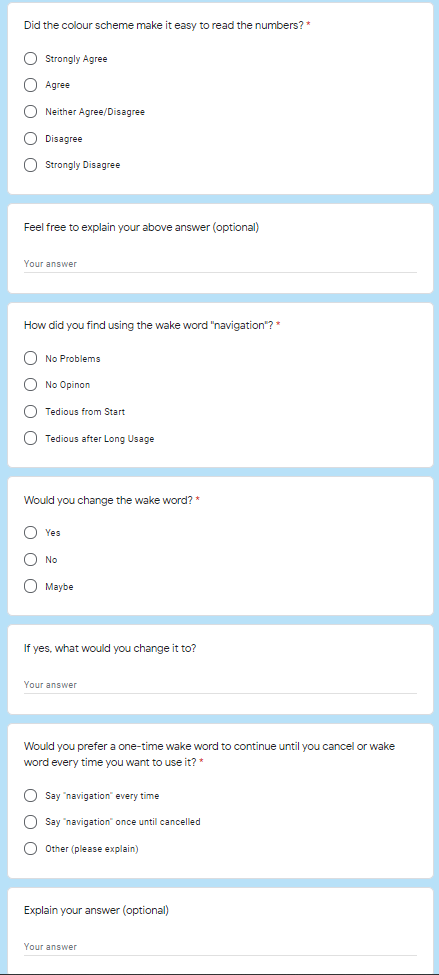
Once the testers had finished their testing, I sent them a Google Survey, to receive feedback.

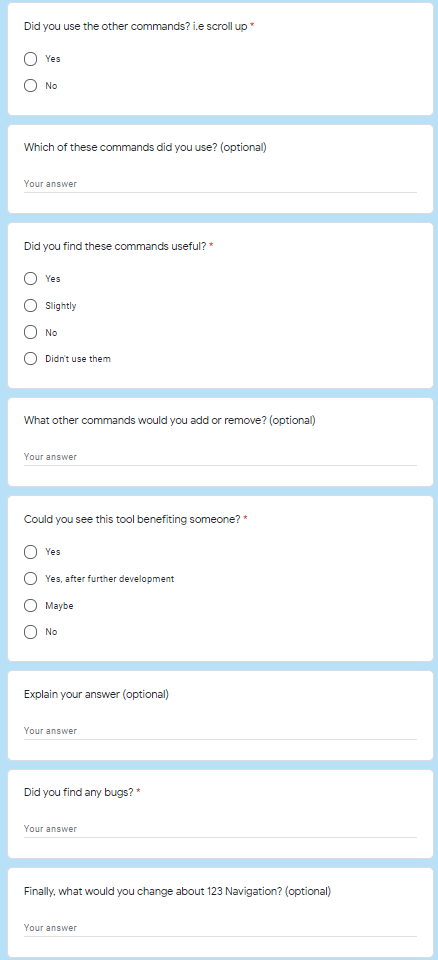
I received a total of nine responses to the survey with very helpful feedback and suggestions.

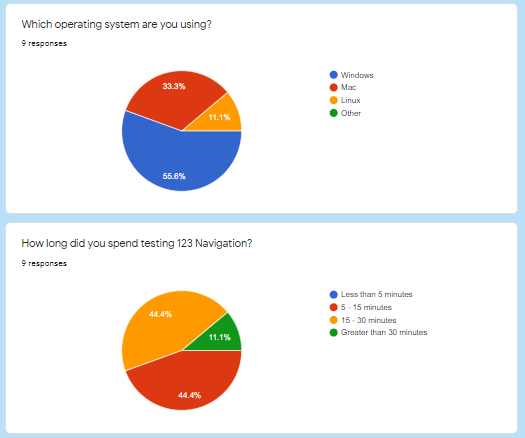
The survey is listed below.



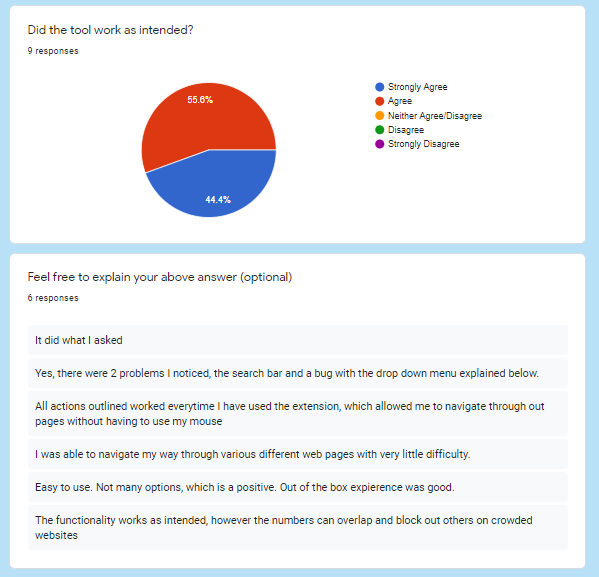




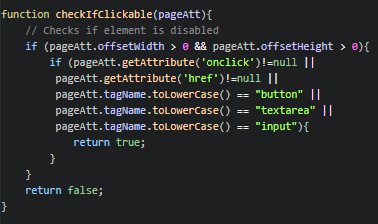




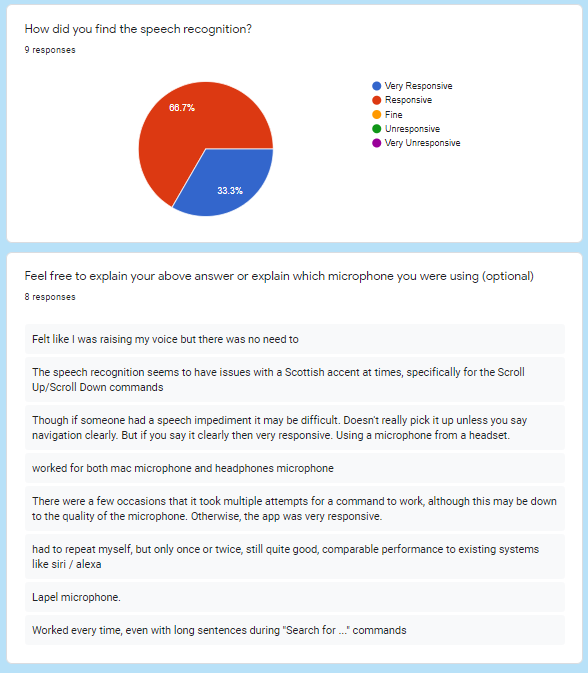
Most users used 123 Navigation on Windows; however, some did use MacOS and Linux. Most tested it for up to 15 minutes.



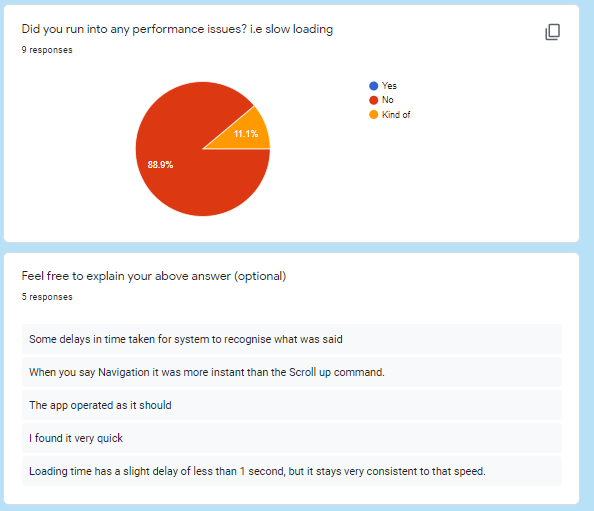
Every single responded very positively to “Did the tool work as intended?”, all responses either agreed that 123 Navigation did as it should, or they reported very minor glitches which would include overlapping navigation icons on crowded websites, this feedback was put into the practise and the crowding on certain web pages was brought down significantly by narrowing down the elements selected, now only non-disabled and specific elements will be shown.







All responses for the speech recognition itself again were very positive. Nobody felt the speech recognition for unresponsive. All comments on the speech recognition itself however show there is some issues with some accents, the Scottish accent in particular and comments about how a clear speech input is required and if a user could not speak clear and fast it might mis-understand them. This leaves the possibility of perhaps slowing down the result function to allow more time for the user to speak.



Some users found the performance of 123 Navigation faster than others, seemingly depending on the command they were using.

This could also depend on which web page they were using; how many HTML elements were in each of these web pages.

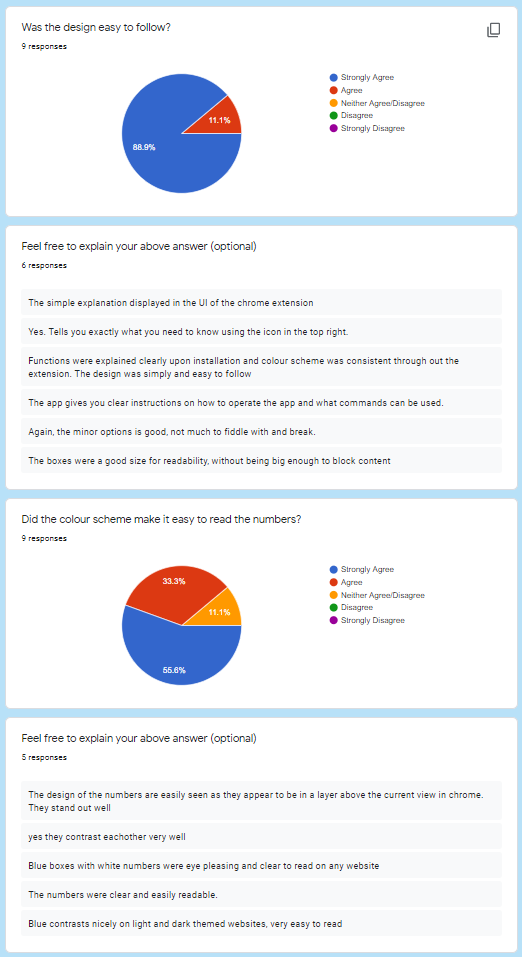
I have taken small steps in the code to improve performance and hopefully speed up the application in some areas. Mainly in the processing of checking the user’s voice input.

A before and after of the code changes can be viewed below.



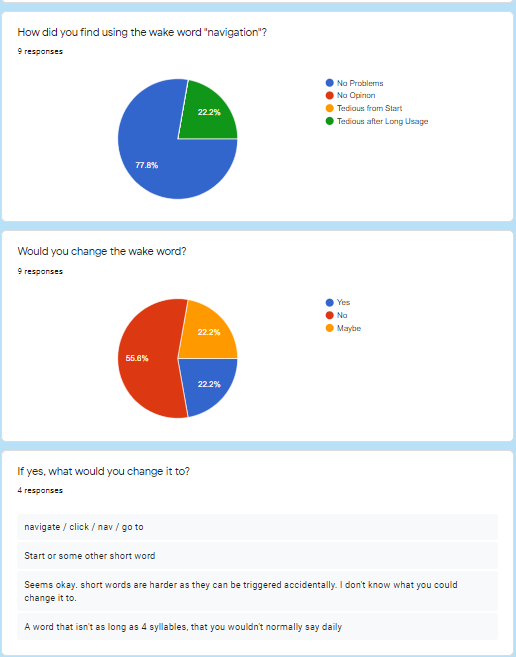


As seen above, the code is much shorter while keeping the same functionality, it also sets the command to lower case when creating the command, rather than every time it wishes to check the command.



The design and colour scheme got overwhelmingly positive feedback. All users believed the colour scheme and design was easy to follow and did not blend in with any web pages.

The welcome page and index page were both met with positive feedback.



Most users found no problems with the main wake word “navigation” however, some did find it tedious after long usage.

Most users reported back that they would not change the wake word, however many did report that they would prefer a shorter word. Changing the wake word would take extensive testing to find a word that would suit everyone. For this reason and the mostly positive feedback towards the wake word, it has remained as “navigation”.



An interesting insight to some of the testers, was the question asking if they would prefer to have a one-time wake word for navigating or use the wake word every time.

Nearly 70% said they would prefer to use a one-time wake word. I personally believe in keeping it as a wake word every time you wish to use it but this definitely opens up the possibility to change and maybe even give the option to choose in the future.

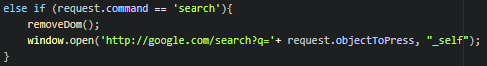




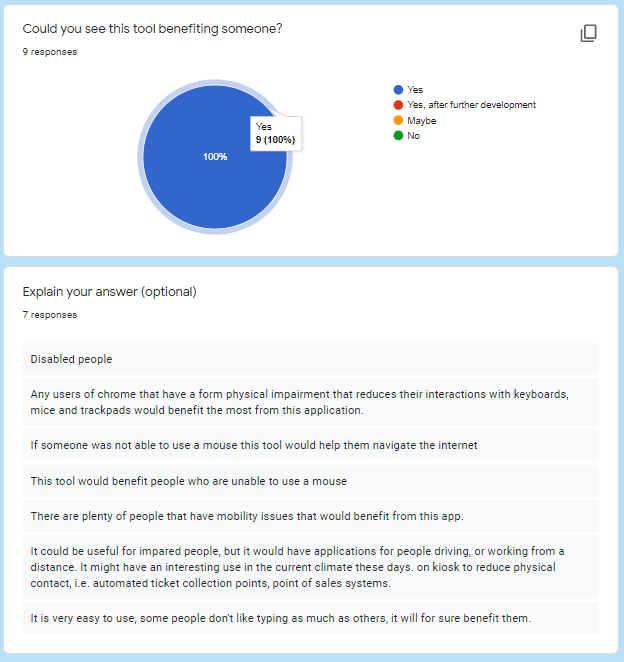
The biggest impact this survey gave to 123 Navigation is the suggested commands. All extra commands, such as scroll up and down, go back and go forward were met with extremely positive feedback, giving the project a much more completed feel.

Some extra feedback that was given was the suggestions to “Search for”, “go to top and bottom of pages”. All these suggestions were implemented and now the users can simply say “go top” or “go bottom” to quickly navigate to the top or bottom of their current webpage or say “search for {search term}” to quickly search Google using their voice. All implementations of these commands were met with overwhelming positive feedback. If the user does not say anything after “search for” the code does not run, stopping blank searches from becoming a problem. Not one user suggested removing any commands, which is good and allows a broader use case for users without the user feeling overwhelmed.

The code to search Google can be seen below.

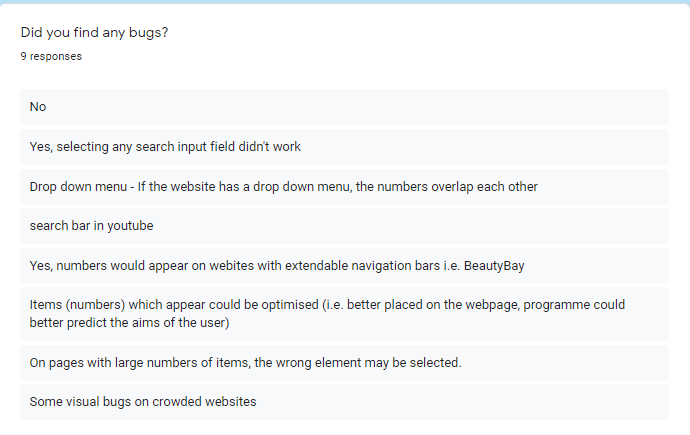


The “\_self” pamarameter, opens the URL in the current tab. It originally openeded in a new tab but users felt it would be better in the current tab.



All testers felt that 123 Navigation would help someone. Some believed that people with disabilities, such as user that could not work their mouse would benefit from using 123 Navigation, others believed that other fields could benefit, such as people wishing to keep distance from using public machines such as ticket kiosks.

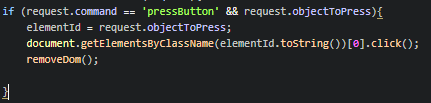
123 Navigation got overwhelmingly positive feedback that it could help somebody in many different scenarios.



The bugs that the users found were very consistent, which is good as it allows focus to be spent mainly on these bugs. Most of the bugs however were to do with visuals, such as the overlapping navigation icons, which I have spoken about previously.

A bug that was mentioned was on pages with larger volumes of navigation icons the wrong element would be clicked. I researched into this more and discovered that the manner in which I was selecting the elements was by ID, however some pages such as YouTube would use the same ID for many of their elements, thus when clicking it would just select the first. I worked around this by using the element classes instead. I can make the class of each element unique, without disrupting the styling of the page by adding the navigation value to the end of the class, finding the element with this exact class and clicking that. The code for that can be seen below.





This ensures each element is always unique. This code change solved the issue of the wrong element getting clicked when on certain web pages.

The other bug that was reported a lot was the report that clicking search bars did not work. This is a bug I noticed early on but have not managed to solve. It appears when a search or input field element is clicked using 123 Navigation, it selects the div around the element rather than the input field itself. This is a bug I would need more time to fix out.

Other than these two bugs, not many bugs were reported, some did not report any bugs, which is good and means that the bugs that were found were specific to certain use cases and not every use.

# System Evolution

123 Navigation has lots of room to evolve, as the use cases for a voice-controlled internet browser are endless.

In the future the ability to implement a speech to text functionality to allow users to type without the use of their keyboard, this feature would act seamlessly with the current navigation, for example if the user decides to click on a text box (using the current navigation) speech to text would automatically turn on allowing the user to type without using any other commands.

The implementation of a bookmarks feature, which would allow users to save URLs to a nickname such as “Bookmark 1” so the user can just say “Bookmark 1” to go to that page, making navigating through their browser much quicker and responsive as many of us use bookmarks without thinking.

The possibility to develop 123 Navigation to work as an extension to other browsers such as Mozilla Firefox and Opera is very possible, as the JavaScript API WebkitSpeechRecognition will be compatible with Firefox (MDN contributors, 2019).

The ability to save settings, such as having the choice of using a one-time wake or say it each time, or change the font size or colour would be small improvements but would make for a better user experience.

The possibilities of growth from 123 Navigation are endless and I am very excited to see where this project goes and how it evolves.

# Project Proposal

**ABC Navigation**

Hands Free Internet Surfing

Wayne Hartigan

x16348136

[x16348136@student.ncirl.ie](mailto:x16348136@student.ncirl.ie)

[hartiganwayne@gmail.com](mailto:hartiganwayne@gmail.com)

BSc (Hons) in Computing

Software Development

16/10/2019

Note that this project proposal was created prior to changing the name from “ABC Nav” to “123 Navigation” and using numeric values instead of character values.

## Objectives

*ABC Navigation* is a project designed to create a hands-free navigation system (Chrome Extension) for users in Google Chrome. This will work by assigning each link, button, text field, or anywhere that can be clicked on a web page a letter, i.e. a small box with ‘‘a’’ beside it the user will then say the letter ‘’a’’ and that will be clicked.

The main objective of *ABC Navigation* is to allow users that lack the motor skills or ability to use their mouse, trackpad or keyboard consistently navigate throughout Google Chrome.

My mother suffers from Multiple sclerosis (MS) and as her MS has developed her motor skills have begun to degrade and tasks as menial as navigating throughout the internet is becoming a more challenging task. If *ABC Navigation* can help her return to navigating the internet, then the goals of *ABC Navigation* have been met, and if it can help my mother, I believe it can help a lot of people.

The global objective is to help people who cannot help themselves.

## Background

As stated above in the “Objectives” my mother suffers from Multiple sclerosis (MS), this is what have inspired me to create *ABC Navigation.* I have always felt the need to find anything to help my mother out, whether through software, hardware, or gadgets to make her life a little bit easier but one thing that has always fallen short is the software.

I have tried on multiple occasion to install different applications, chrome extensions, etc. to my mother’s computer to help her use her computer more efficiently but they all are either too slow, too complex or too expensive and require external hardware.

I tried a Chrome Extension before that was designed to navigate Chrome with speech but all it was not easy, the user had to either hold the control key to speak or have the voice service listening all the time, which would cause problems with general speech being picked up and navigating. The navigation commands were generic “Scroll”, “Click”, etc. but there was no clear guide to what to say and the command had to be perfect or it would not pick it up. Creating a lot of confusion and problems when trying to use it.

After that I decided I would create *ABC Navigation,* a more user friendly and simple to use voice navigation system by giving the user a “wake word” which will eliminate the need to press a key to have the system listen or have the voice system listening constantly, which will make people feel more comfortable using the extension. Also by displaying a value on each link or button to click on, the user is given clear and obvious instructions to navigate.

## Technical Approach

As this is the first instance of developing a chrome extension for myself, there will be some research needed.

The main languages used to develop Chrome Extensions are HTML, CSS, JavaScript and JSON. My preferred language to code in is Python, I would prefer to create ABC Navigation in Python but that appears to be very difficult, in order to do this I would need to develop in Python and use either ‘RapydScrypt’ or ‘Brython’ both of which translate Python code to JavaScript, I have begun testing this and it seems to work well on smaller codebases but I am not sure how it would handle something of this size and the libraries for python will not work in this format. I will be looking into other options such as just using JavaScript or possibly a codebase called ‘Coffee’, which is a JavaScript based language that uses the formatting and syntax of Python, which if ‘Brython’ and ‘RapydScript’ fall through might be a better option personally.

I will be also looking into the different libraries ‘Coffee’, ‘Python’ & ‘JavaScript’ offer for controlling voice commands.

Research that will need to be completed for the actual use case of the app include. Deciding on a ‘wake word’ for the user to say to begin the navigation similar to Amazon’s ‘Alexa’ and Apple’s ‘Hey Siri’. Other smaller details but that are just as important include, font, font size, box colour etc. for the overall look and feel. This research will be conducted by handing out a survey with images and examples to the demographic this app will target, which are people in my mother’s hospital.

## Special resources required

As stated above my plan is to develop *ABC Navigation* to be used without an external hardware for ease of use.

Most voice-controlled application would recommend an external microphone for clarity, but the extension will work without one just fine.

## Project Plan

A full project plan has not been conducted yet as I am yet to meet with my supervisor, this will be discussed in detail at a later date.

## Technical Details

JavaScript, HTML, CSS, jQuery, JUnit Tests & integration tests. JavaScript has a API specifically for speech recognition, I will be researching that further to implement into the extension.

## Evaluation

As I develop ABC Navigation, I will be implementing unit tests to ensure developing further does not break any previous work, I will then be implementing integration tests near the end.

In terms of user testing I will be ensuring that my target audience can easily navigate throughout chrome. This will be the main objective.

# Reflective Journals

Reflective Journal October

Student name: Wayne Hartigan

Programme: BSc in Computing (Software Development)

Month: October 2019

**My Achievements**

As this was the first month of work on the project, not much progress has been made, I have been researching languages and method used to create a chrome extension as this is my first experience with developing one. Other than that I have completed the ethics forms and have met with Peamount Hospital about a possible test plan being rolled out to patients willing to partake in the testing, this will include surveys and physical testing of the chrome extension.

I met with Ron Elliot but as was made very clear, Ron was more focused on business analytics, while I am in the Software Development stream. I do believe this was a mistake. I now have been assigned a new supervisor, David Kelly, I have emailed him and as of writing, I am yet to receive an email back. When I meet with David I will discuss the ethics, proposal and work flow in greater detail.

**My Reflection**

I feel the research going towards the chrome extension is going very well, I am studying JavaScript to sharpen my skills, I have also discovered a method in which I can write JavaScript using Python, although this is not a very efficient means of development.

**Intended Changes**

Next month I intend to have a better supervisor meeting and discuss in detail, what needs to be done, when and how, as I have no received sufficient help thus far. I also plan to begin working on a beta version for the chrome extension.

**Supervisor Meetings**

Date of Meeting (with Ron Elliot): 22nd of October 2019

Date of Meeting (with David Kelly): N/a

Reflective Journal November

Student name: Wayne Hartigan

Programme: BSc in Computing (Software Development)

Month: November 2019

**My Achievements**

This month I met with my new supervisor, David Kelly. David walk me through all that is needed to be completed before the mid-point presentation along, he went into great details about what is needed for the requirement specification document, mid-point presentation slides and ethics form.

David helped me break down what aspect of the prototype I should target for the mid-point presentation. Thus, I am now breaking my prototype into 3 separate parts, a demonstration of voice being picked up by a programme, a DOM injection and a working chrome extension. This has really helped me understand and pick up the pace for the upcoming weeks.

I have also been working hard on my JavaScript skills, and be looking to expand my skills into NodeJS and looking at server level python.

The requirement specification document was also due this month which, with David’s help and advice, was completed.

**Next Month Plan**

Within the next I plan to work heavily on my prototypes and to have a working prototype for each component needed to display my project.

I also plan to work closely and weekly with David, in order to stay on the correct path.

**Supervisor Meetings**

Date of Meeting: 28th November 2019

Items discussed: Mid-point presentation

Reflective Journal

Student name: Wayne Hartigan

Programme: BSc in Computing (Software Development)

Month: December 2019

**My Achievements**

This month’s work was all to prepare for the mid-point presentation, as the presentation was worth 25%, I spent many days in meetings with my supervisor David Kelly, he helped me prepare for the presentation with the slides and the prototype as well as the documentation needed. I met with David three times this month before my presentation and we went into detail about my documentation and where it needed to improve.

I also demonstrated my prototype for David, my prototype had the functionality to take input from a user’s voice and using DOM manipulation the chrome extension will input the navigation icons I have designed.

The midpoint presentation went very well in my opinion, my slides and prototype were highly received. My documentation will need to improve as per the feedback I received. My grade was 65.7%.

**Next Month Plan**

Within the next month I plan to have developed my prototype into a working beta of the project, with the ability to actually control the webpage using the user’s voice.

I also plan on working very hard on my documentation to try bring up the quality and quantity.

**Supervisor Meetings**

Date of Meeting: 5th, 10th, 12th of December

Items discussed:

* Mid-point presentation
* Documentation
* Prototype.



Reflective Journal

Student name: Wayne Hartigan

Programme: BSc in Computing (Software Development)

Month: February 2020

**My Achievements**

This month’s work was all about preparing for the second semester, we had classes on preparing our projects and all the deliverables needed. I have put my name down for a seminar on the 3rd of February to help work on my bio and posters for the showcase.

Progress for the project has been slow, I have spent most of my time researching and testing out new ways to speed up the process I am currently using for my speech recognition.

**Next Month Plan**

Next month I plan to attend as many seminars as possible to really prepare for the final deliverables.

I plan to have most of the major features implemented by the end of next month and then begin on expanding these processes.

**Supervisor Meetings**

I did not have any meetings with supervisor this week. I emailed David Kelly on the 21st of January but did not hear back in time to schedule a meeting this month.



Reflective Journal

Student name: Wayne Hartigan

Programme: BSc in Computing (Software Development)

Month: February 2020

**My Achievements**

This month I began furthering my development of ABC Navigation, I have managed to complete the development allowing the user to click a button (or link etc.) using only their voice and saying the letters that have appeared on the screen. Other work put into the project has included refining the speed and reliability of the project, a lot has yet to be done before the project is user friendly, but the project is starting to take shape.

Other work done throughout the month has included documentation work, expanding the documentation to include the newer up to date features and use case.

**Next Month Plan**

Next month I plan to work hard on development and by the end of march I want to have the ability to fully navigate throughout Google Chrome using only my voice.

I plan to meet with my supervisor to discuss documentation as David has informed me that the documentation of my project is what I will require help with as the development side has progressed very well.

**Supervisor Meetings**

Meeting Date: Friday, 14th February

During my meeting with David, we discussed where the development is and where I plan to take it and how, I explained all my plans to further develop ABC Nav and David agreed and said I do not need guidance for the development thus, we discussed in heavily the documentation for the project, we reviewed my documentation from the mid-point presentation and where needs to be expanded. David has helped a lot with the documentation and my documents are currently under review, which I will correct once I have received the review.



# Bibliography

Dotcom Tools, 2018. *How Fast Should My Website Load?.* [Online]   
Available at: https://www.dotcom-tools.com/blog/how-fast-should-my-website-load/  
[Accessed 10 12 2019].

Google, 2019. *Chrome Browser system requirements.* [Online]   
Available at: https://support.google.com/chrome/a/answer/7100626?hl=en  
[Accessed 10 12 2019].

Google, 2019. *What is the Chrome Web Store?.* [Online]   
Available at: https://developer.chrome.com/webstore/about\_webstore  
[Accessed 10 12 2019].

MDN contributors, 2019. *Using the Web Speech API.* [Online]   
Available at: https://developer.mozilla.org/en-US/docs/Web/API/Web\_Speech\_API/Using\_the\_Web\_Speech\_API  
[Accessed 10 12 2019].

MDN contributors, 2019. *What is JavaScript?.* [Online]   
Available at: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First\_steps/What\_is\_JavaScript  
[Accessed 10 12 2019].

Shires, G., 2013. *Voice Driven Web Apps: Introduction to the Web Speech API.* [Online]   
Available at: https://developers.google.com/web/updates/2013/01/Voice-Driven-Web-Apps-Introduction-to-the-Web-Speech-API  
[Accessed 01 12 2019].

Shires, G., 2013. *Voice Driven Web Apps: Introduction to the Web Speech API.* [Online]   
Available at: https://developers.google.com/web/updates/2013/01/Voice-Driven-Web-Apps-Introduction-to-the-Web-Speech-API  
[Accessed 10 12 2019].

W3 Schools, 2018. *CSS Introduction.* [Online]   
Available at: https://www.w3schools.com/css/css\_intro.asp  
[Accessed 10 12 2019].

W3 Schools, 2018. *What is JSON?.* [Online]   
Available at: https://www.w3schools.com/whatis/whatis\_JSON.asp  
[Accessed 10 12 2019].

W3Schools, 2018. *What is HTML?.* [Online]   
Available at: https://www.w3schools.com/whatis/whatis\_html.asp  
[Accessed 10 12 2019].

WebDesignRankins, 2019. *What is intuitive web design?.* [Online]   
Available at: https://www.webdesignrankings.com/what-is-intuitive-web-design/  
[Accessed 10 12 2019].