Voice Command Browsing (Extension for Google Chrome Browser)

MINI PROJECT REPORT

Submitted in partial fulfillment of the requirements for the award of degree

BACHELOR OF TECHNOLOGY

BY

RAJESH R NAIR **Reg.no:14015695**
UNNIKRISHNAN **Reg.no:14015716**

SCMS SCHOOL OF ENGINEERING AND TECHNOLOGY
(Affiliated to M.G. University)
VIDYA NAGAR, PALISSERY, KARUKUTTY
ERNAKULAM – 683 582

June, 2017

SCMS SCHOOL OF ENGINEERING AND TECHNOLOGY

(Affiliated to M.G. University)

VIDYA NAGAR, PALISSERY, KARUKUTTY

ERNAKULAM – 683 582

BONAFIDE CERTIFICATE

This is to certify that the mini project, titled "Voice Command Browsing (Extension for Google Chrome Browser)" by

RAJESH R NAIR	**Reg.no:14015695**
UNNIKRISHNAN	**Reg.no:14015716**

submitted in partial fulfilment of the requirement for the award of the degree of Bachelor of Technology, is a bonafide work carried under supervision, during the academic year 2015-2016.

Ms. DEEPA SREE VARMA	**Prof. VINOD P**
PROJECT GUIDE	**HEAD OF DEPARTMENT**

INTERNAL EXAMINER	**EXTERNAL EXAMINER**

Abstract

Surfing the internet has reached the top in the list of things done for entertainment in the world on a daily basis. As such browsers used for this purpose have unintentionally become the most used applications on a desktop or laptop. Browsing trends have evolved over the years, from the default used text searches to speech recognition based search and even image based search across the web. The goal of this project is to create a vocal command based extension which can be used to perform the basic and a few intermediate functions that are otherwise done in the browser using the keyboard or mouse. For instance, voice commands can be given to open a new tab rather than doing so using a mouse. A variety of such simple commands can be implemented to improve the functionalities of the extension. The goal of this project is to design and implement a extension or an add-on which performs functions based on simple voice commands. The extension will be set to work on any Google Chrome browser. The extension will be made available in the Google Chrome support page (which has all extensions and extensions). The users can install it and make it available on their Google Chrome browser. The project aims to improve the ease of accessibility of Google Chrome browser users using voice commands rather than the usual type-and-click method. Users will be free to leave their keyboards and mouse and can depend on just their own voices to perform most browser related operations such as opening a new tab, closing tabs, searching the web etc. Though no functions in websites itself can be performed, the browser related operations will be easily done using this extension.

ACKOWLEDGEMENT

We are greatly indebted to Prof.M.Madhavan, Principal, SSET, Ernakulam and Prof.Vinod, Head of department, Department of Computer Science and Engineering, SSET, who whole heartedly granted us the permission to carry out the mini project. We would like to thank our guide, Ms.Deepa Sree Varma, Assistant Professor, Department of Computer Science and Engineering, SSET who has given us valuable guidance and support throughout the project. Also, we would like to thank our project coordinators, Ms.Shilpa P C and Ms.Gayathri Assistant Professors, Department of Computer Science and Engineering, SSET, who supported and instructed us all the way. We would like to express our sincere gratitude to all the teachers of Computer Science Department who gave us moral and technical support through the course of our mini project. We would like to thank the supporting staff in the Computer lab whose dedicated work kept the lab working smoothly, thus ensuring our time at the lab went hassle free.

Contents

List of Figures

Chapter 1

Introduction

1.1 OVERVIEW

The 'Voice Enabled browser' consists of three main parts: Speech Engine , the Java application and the Interface. The relationships between the modules can be seen in the following simplified class diagram. The GUI is the part of Interface module of the project. The speech recognizer will listen to what you say and convert your words to a string. A voice browsing extension is a software application that presents an interactive voice user interface to the user in a manner analogous to the functioning of a web browser interpreting Hypertext Markup Language (HTML). Dialog documents interpreted by browser are often encoded in standards-based markup languages, such as Voice Dialog Extensible Markup Language (VoiceXML), a standard by the World Wide Web Consortium. A voice browser presents information aurally, using prerecorded audio file playback or text-to-speech synthesis software. A voice browser obtains information using speech recognition [1, 2, 3] and keypad entry, such as DTMF detection. As speech recognition and web technologies have matured, voice applications are deployed commercially in many industries and voice browsers are supplanting traditional proprietary interactive voice response (IVR) systems. Voice browser software is delivered in a variety of implementations models.

1.2 PROBLEM ANALYSIS

Problem analysis is the process of understanding the actual problems, user needs and proposing solutions to meet those needs. The goal of problem analysis is to gain a better understanding of the problem being solved before development begins. It is the process of gathering and interpreting facts, diagnosing problems and using the information to recommend improvements on the system. Problem analysis is problem solving activities that require intensive communication between users and the system developers. A problem can be defined as the difference between things as perceived and things as desired. The system is studied and analyzed. The system is viewed as a whole and the input to the system are identified. The output from the system is given to various processes.

1.3 EXISTING SYSTEM

1.3.1 KEYBOARD

In normal usage, the keyboard is used as a text entry interface to type text and numbers into a word processor, text editor or other programs. In a modern computer, the interpretation of key presses is generally left to the software. A computer keyboard distinguishes each physical key from every other and reports all key presses to the controlling software. Keyboards are also used for computer gaming, either with regular keyboards or by using keyboards with special gaming features, which can expedite frequently used keystroke combinations.

1.3.2 MOUSE

A computer mouse is a pointing device (hand control) that detects two-dimensional motion relative to a surface. This motion is typically translated into the motion of a pointer on a display, which allows a smooth control of the graphical user interface. Physically, a mouse consists of an object held in one's hand, with one or more buttons. Mice often also feature other elements, such as touch surfaces and "wheels", which enable additional control and dimensional input.

1.4 PROPOSED SYSTEM

Aims to improve the ease of accessibility of chrome browser users using voice commands rather than the usual type-and-click method. A vocal command based extension which can be used to perform the basic and a few intermediate functions that are otherwise done in the browser using the keyboard or mouse. Users will be free to leave their keyboards and mouses and can depend on just their own voices to perform most browser related operations such as opening a new tab, closing tabs, searching the web etc.

1.5 FEASIBILITY STUDY

Feasibility study is a procedure that identifies, describes and evaluates candidate systems and selects the best system for the job. An estimate is made whether the identified users need may be satisfied using the current software and hardware technologies. The study will decide whether the proposed system will be cost effective from a business point of view and if it can be developed using the given existing budgetary constraints. The key considerations involved in the feasibility analysis are the following:

1. Economic feasibility
2. Technical feasibility
3. Operational feasibility

1.5.1 ECONOMIC FEASIBILITY

Economic study is the most frequently used method for evaluating the effectiveness of candidate system. More commonly known as cost/benefit analysis, the procedure is to determine the benefits and savings that are accepted from a candidate system and compares with costs. If benefit outweighs cost, then decisions are made to design and implement the system. Otherwise further alterations will have to be made if to have a chance of being approved. Less hardware is required and can also be mounted on the existing wheelchair with reduced complexity.Hence this project is economically feasible and is cost effective because of its compatibility and effort saving nature.

1.5.2 TECHNICAL FEASIBILITY

Technical feasibility is a measure of how feasible the project is technically. The effort and technology included in the conventional system is not needed as the whole process is automated. The hierarchy of the new system is very easier than the existing system. The new system is very much easier and user friendly. Operational cost is very easy. The maintenance and modification of the new system needs very less human effort.

1.5.3 OPERATIONAL FEASIBILITY

Its is very reliable since a microphone is supported by all existing systems and the voice recognition system enables the user to save time by using simple commands instead of typing externally .this facility

doesnot require any preexperience since all included invoking commands are simple names which is common to all.

Chapter 2

DESIGN

2.1 BLOCK DIAGRAM

2.1.1 CONTEXT LEVEL DFD

Figure 2.1: Context level DFD

1. Context level DFD shown in Figure 2.1 is the most basic representation of the system.
2. This indicates the basic working of the system.
3. The user controls the application.

2.1.2 LEVEL 0 DFD

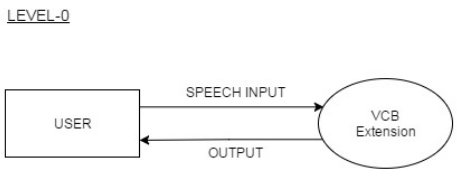

Figure 2.2: Level 0 DFD

1. Level 0 DFD shown in Figure 2.2 indicates the basic processes involved in the system.
2. This level indicates all the processes of our system.

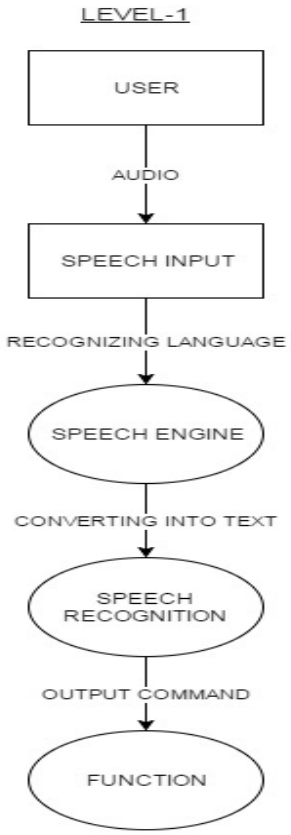

Figure 2.3: Level 1 DFD

2.1.3 LEVEL 1 DFD

1. Level 1 DFD shown in Figure 2.3 represents working of the extension

2.1.4 LEVEL 2 DFD

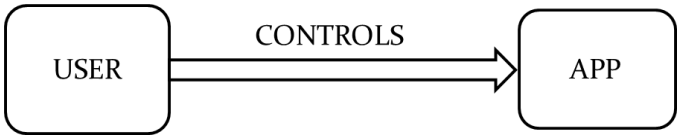

Figure 2.4: Level 2 DFD

1. Level 2 shown in Figure 2.4 includes the processing in speech api which fetch data from the server.

Chapter 3

IMPLEMENTATION

3.1 SYSTEM REQUIREMENTS

3.1.1 HARDWARE REQUIREMENTS

Requirements that helps in communication between the Android app and the hardware

- Laptop i5-6200 1tb HDD and 8gb RAM
- Laptop i5-6200 1tb HDD and 4gb RAM
- Microphone

3.1.2 SOFTWARE REQUIREMENTS

Requirements that helps in developing the software

- Online Database Storage
- Windows 7/8/10
- Wifi Connectivity
- Java script
- Chrome Browser

3.2 PLATFORM AND IDE

3.2.1 ONLINE DATABASE STORAGE

Online data storage refers to the practice of storing electronic data with a third party service accessed via the Internet. It is an alternative to traditional local storage (such as disk or tape drives) and portable storage (such as optical media or flash drives). It can also be called "hosted storage,""Internet storage" or "cloud storage." In recent years, the number of vendors offering online data storage for both consumers and businesses has increased dramatically. Some services store only a particular kind of data, such as photos, music or backup data, while others will allow users to store any type of file. Most of these vendors offer a small amount of storage for free with additional storage capacity available for a fee, usually paid on a monthly or annual basis.

3.2.2 WINDOWS

Microsoft Windows (or simply Windows) [4] is a metafamily of graphical operating systems developed, marketed, and sold by Microsoft. It consists of several families of operating systems, each of which cater to a certain sector of the computing industry with the OS typically associated with IBM PC compatible architecture. Active Windows families include Windows NT and Windows Embedded; these may encompass subfamilies, e.g. Windows Embedded Compact (Windows CE) or Windows Server. Defunct Windows families include Windows 9x, Windows Mobile and Windows Phone. Microsoft introduced an operating environment named Windows on November 20, 1985, as a graphical operating system shell for MS-DOS in response to the growing interest in graphical user interfaces (GUIs). Microsoft Windows came to dominate the world's personal computer (PC) market with over 90% market share, overtaking Mac OS, which had been introduced in 1984. Apple came to see Windows as an unfair encroachment on their innovation in GUI development as implemented on products such as the Lisa and Macintosh (eventually settled in court in Microsoft's favor in 1993). On PCs, Windows is still the most popular operating system. However, in 2014, Microsoft admitted losing the majority of the overall operating system market to Android, because of the massive growth in sales of Android smartphones. In 2014, the number of Windows devices sold was less than 25% that of Android devices sold. This comparison however may not be fully relevant, as the two operating systems traditionally target different platforms. As of September 2016, the most recent version of Windows for PCs, tablets, smartphones and embedded devices is Windows 10. The most recent versions for server computers is Windows Server 2016. A specialized version of Windows runs on the Xbox One game console.

3.2.3 WIFI CONNECTIVITY

Wi-Fi or WiFi is a technology for wireless local area networking with devices based on the IEEE 802.11 standards. Wi-Fi is a trademark of the Wi-Fi Alliance, which restricts the use of the term Wi-Fi Certified to products that successfully complete interoperability certification testing. Devices that can use Wi-Fi technology include personal computers, video-game consoles, smartphones, digital cameras, tablet computers, digital audio players and modern printers. Wi-Fi compatible devices can connect to the Internet via a WLAN network and a wireless access point. Such an access point (or hotspot) has a range of about 20 meters (66 feet) indoors and a greater range outdoors. Hotspot coverage can be as small as a single room with walls that block radio waves, or as large as many square kilometers achieved by using multiple overlapping access points. Depiction of a device sending information wirelessly to another device, both connected to the local network, in order to print a document Wi-Fi most commonly uses the 2.4 gigahertz (12 cm) UHF and 5 gigahertz (6 cm) SHF ISM radio bands. Having no physical connections,it is more vulnerable to attack than wired connections, such as Ethernet.

3.2.4 JAVASCRIPT

JavaScript [5, 6] often abbreviated as "JS", is a high-level, dynamic, untyped, interpreted run-time language. It has been standardized in the ECMAScript language specification. Alongside HTML[7] and CSS[8], JavaScript is one of the three core technologies of World Wide Web content production; the majority of websites employ it, and all modern Web browsers support it without the need for plug-ins[9]. JavaScript is prototype-based with first-class functions, making it a multi-paradigm language, supporting object-based, imperative, and functional programming styles.It has an API for working with text, arrays, dates and regular expressions, but does not include network, storage, or graphics APIs, relying instead upon APIs made available by its host environment[10] . Although there are strong outward similarities between JavaScript and Java, including language name, syntax, and respective standard libraries, the two languages are distinct and differ greatly in design; JavaScript was influenced by programming languages such as Self and Scheme. JavaScript is also used in environments that are not Web-based, such as PDF documents, site-specific browsers, and desktop widgets. Newer and faster JavaScript virtual machines

(VMs) and platforms built upon them have also increased the popularity of JavaScript for server-side Web applications. On the client side, developers have traditionally implemented JavaScript as an interpreted language, but more recent browsers perform just-in-time compilation. Programmers also use JavaScript in video-game development and in desktop and mobile applications.

3.2.5 GOOGLE CHROME BROWSER

Google Chrome is a freeware web browser developed by Google, It was first released in 2008, for Microsoft Windows, and was later ported to Linux[11] , macOS, iOS and Android. Google Chrome is also the main component of Chrome OS, where it serves as a platform for running web apps [12]. Google releases the majority of Chrome's source code as the Chromium open-source project.A notable component that is not open-source is the built-in Adobe Flash Player (that Chrome has disabled by default since September 2016). Chrome used the WebKit layout engine until version 27. As of version 28, all Chrome ports except the iOS port use Blink, a fork of the WebKit engine. As of February 2017, StatCounter estimates that Google Chrome has a 62% worldwide usage share of web browsers as a desktop browser. It also has 52% market share across all platforms combined,because it is also the most popular browser for smartphones. Its success has led to Google expanding the "Chrome" brand name on various other products such as Chromecast, Chromebook, Chromebit, Chromebox and Chromebase.

3.3 DETAILS OF IMPLEMENTATION

The main objective of our project is to design a wheelchair that will be controlled wirelessly and will be very easy to operate it with no physical efforts.Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective. The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

3.3.1 SOURCE CODE

3.3.1.1 Javascript code

```
function win(){
window.open('http://localhost/new-tab/jj.html','_self',\
 'windowName', "height=400,width=600");
}
try{
chrome.tabs.getSelected(null,function(tab) {
    var tablink = tab.url;
    console.log(tablink);
    if(tablink=='chrome://newtab/'){
        win();
    }
});
document.getElementById("rec").addEventListener("click", win);
}
catch(e){

}
function OpenTabs(cases) {
    switch(cases){
    case 'history':
    chrome.tabs.create({url: 'chrome://history/'});
    break;
```

```
    case 'download':
    chrome.tabs.create({url: 'chrome://downloads/'});
    break;
    case 'settings':
    chrome.tabs.create({url: 'chrome://settings/'});
    break;
    case 'flags':
    chrome.tabs.create({url: 'chrome://flags/'});
    break;
    case 'version':
    chrome.tabs.create({url: 'chrome://version/'});
    break;
    case 'help':
    chrome.tabs.create({url: 'chrome://help/'});
    break;
    }
}

window.setInterval(function(){
httpGet();
}, 500);
function httpGet()
{

    var xmlHttp = new XMLHttpRequest();
    // false for synchronous request
    xmlHttp.open( "GET", 'http://localhost/new-tab/text.php', false );
    xmlHttp.send( null );
    var funct=xmlHttp.responseText;
    console.log(funct);
    switch(funct){
        case 'history':OpenTabs(funct);
    httpClear();
        break;
        case 'download':OpenTabs(funct);
    httpClear();
        break;
        case 'settings':OpenTabs(funct);
    httpClear();
        break;
        case 'version' :OpenTabs(funct);
    httpClear();
        break;
         case 'flags':OpenTabs(funct);
    httpClear();
        break;
        case 'help':OpenTabs(funct);
    httpClear();
        break;
    }
}

function httpClear(){
    var clearHttp = new XMLHttpRequest();
    // false for synchronous request
    clearHttp.open( "GET", 'http://localhost/new-tab/text.php?data', false );
    clearHttp.send( null );
}

var recognition = new (window.SpeechRecognition
  || window.webkitSpeechRecognition || window.mozSpeechRecognition
  || window.msSpeechRecognition)();
```

```
function reco(){
console.log('Recognition Function Called');
recognition.lang = 'en-US';
recognition.continuous     = false;
recognition.interimResults = false;
recognition.maxAlternatives = 5;
recognition.start();
    recognition.onresult = function(event) {
        console.log('You said: ','.'+event.results[0][0].transcript+'.');
        var str = event.results[0][0].transcript;
        if(str.includes('search')){
            console.log('Google Called');
                var value=str.replace('search', '');
                window.open('https://www.google.co.in/search?q='+value,'_self');
        }
        if(str.includes('open Facebook')){
            console.log('Facebook Called');
                window.open('https://www.facebook.com','_self');
        }
        if(str.includes('open Gmail')){
            console.log('Gmail Called');
                window.open('https://mail.google.com','_self');
        }
        if(str.includes('open YouTube')){
            console.log('YouTube Called');
                window.open('https://youtube.com','_self');
        }
        if(str.includes('open Twitter')){
            console.log('Twitter Called');
                window.open('https://twitter.com','_self');
        }
        if(str.includes('history')==true){
            console.log('History Called');
            httpGet('history');
            str="";
        }
        if(str.includes('download')==true){
            console.log('Downloads Called');
            httpGet('download');
            str="";
        }
        if(str.includes('settings')==true){
            console.log('Settings Called');
            httpGet('settings');
            str="";
        }
       if(str.includes('help')==true){
            console.log('help Called');
            httpGet('help');
            str="";
        }
       if(str.includes('flags')==true){
            console.log('Flags Called');
            httpGet('flags');
            str="";
        }
       if(str.includes('version')==true){
            console.log('version Called');
            httpGet('version');
            str="";
       }
}
```

```
147
148
149     function httpGet(val)
150     {
151         var xmlHttp = new XMLHttpRequest();
152         // false for synchronous request
153         xmlHttp.open( "GET",
154         'http://localhost/new-tab/text.php?data='+val, false );
155         xmlHttp.send( null );
156         console.log(xmlHttp.responseText);
157     }
158 }
```

3.3.1.2 Json code

```
1  {
2        "background": {
3         "scripts": [ "tab.js","jquery.js" ]
4     },
5     "browser_action": {
6        "default_title": "chrome:history"
7     },
8    "name": "Improved new tab page",
9    "description": "Display bookmarks on the new tab page",
10   "version": "0.1",
11   "incognito": "split",
12   "chrome_url_overrides": {"newtab": "jj.html"},
13   "web_accessible_resources": ["text.php"],
14   "permissions": ["bookmarks","history","storage","tabs","background", "
       http://localhost/"],
15   "manifest_version": 2
16 }
```

3.3.1.3 HTML code

```
1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <meta charset="UTF-8" />
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1">
7      <title>New Tab</title>
8      <meta name="description" content="Animated icons powered by the motion graphics
           library mo.js by Oleg Solomka" />
9      <meta name="keywords" content="animated icons, svg, webfont, mo.js, facebook, thumbs
           up, animation, web design" />
10     <meta name="author" content="Codrops" />
11     <link rel="shortcut icon" href="favicon.ico">
12     <link href='https://fonts.googleapis.com/css?family=Patrick+Hand+SC' rel='stylesheet'
           type='text/css'>
13     <link rel="stylesheet" type="text/css" href="fonts/font-awesome-4.5.0/css/font-
           awesome.min.css" />
14     <link rel="stylesheet" type="text/css" href="css/normalize.css" />
15     <link rel="stylesheet" type="text/css" href="css/demo.css" />
16     <link rel="stylesheet" type="text/css" href="css/icons.css" />
17     <!--[if IE]><script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script
           ><![endif]-->
18   </head>
```

```
19    <body>
20
21      <div class="container">
22
23        <section class="content">
24          <ol class="grid">
25            <span class="icobutton__text"></span>
26            <li class="grid__item">
27              <button class="icobutton icobutton--microphone" onclick="reco()" id="rec"><
                    span class="fa fa-microphone"></span></button>
28            </li>
29
30          </ol>
31        </section>
32
33      </div>
34      <!-- /container -->
35      <script src="js/mo.min.js"></script>
36      <script src="js/demo.js"></script>
37      <script src="tab.js"></script>
38    </body>
39  </html>
```

Chapter 4

TESTING

Testing is the most important activity in the development phase. Testing is the process of finding errors or bugs in the system. Testing ensuresthe satisfaction of the users. In other words it isprocess by which one detects the defects in the system. Software testing methods are traditionally divided into black box testing and white box testing. These two approaches are used to describe the point of view that a test engineer takes when designing test cases.

4.1 TEST PLANS

The implementation of a computer-based system requires that test data can be prepared and that the system and its elements be tested in a planned, structured manner. The computer program component is a major subsystem of the computer-based information system, and particular attention should be given to the testing of these system elements as it is developed. Testing is a process of executing a program with the interest of finding an error. A good test is one that has a high probability of finding the yet undiscovered errors. The primary objective for the test case design is to drive a set of tests that has the highest likelihood for systematically uncovering different classes of errors in the software. A series of testing are performed for this project before the system is ready for acceptance. Some of the testing strategies applied for the system are listed here.

Testing Strategies:

1. Black Box Testing
2. White Box Testing

4.1.1 Black Box Testing

Black Box testing also called functional testing, focuses on the functional requirements of the software. Knowing the specified function that a product designed to perform the test can be conducted to ensure that each function is fully operational. Black Box tests are carried out to test that input to function is properly accepted and output is correctly produced. It finds the errors in the following categories:

1. Interface errors.
2. Performance in data structures or external database access.
3. Performance errors.
4. Initialization and termination errors.
5. Incorrect or missing functions.

Black box testing is used to detect errors of type incorrect or missing functions, interface errors, errors in data structures or external database access, performance, initialization and termination errors.

4.1.2 White Box Testing

White box testing, sometimes called – Glass box testing, and is a test case design uses the control structure of the procedural design to check the errors. It involves following procedures:

1. All independent paths within module have been exercised at least once.
2. All logical decisions were checked for the true and false values.
3. All loops were executed to check their boundary values.
4. Internal data structure was tested for their validity.

4.1.3 TYPES OF TESTING

1. Unit Testing
2. Integration Testing
3. Validation Testing
4. Output Testing
5. User Acceptance Testing
6. Product transitions qualities

4.2 TESTCASE TABLES

Table 4.1: CONNECTION PAGE

SL NO.	TEST CASE	RESULT
1		
2		
3		

Chapter 5

RESULTS AND DISCUSSIONS

5.1 PROBLEMS FACED

First, the obvious: speech is a complex audio signal, made up of a large number of component sound waves. Speech can easily be captured in wave form, transmitted and reproduced by common equipment; this is how the telephone has worked for a century. However, once we move up the complexity scale and try to make a computer understand the message encoded in speech, the actual wave form is unreliable. Vastly different sounds can produce similar wave forms, while a subtle change in inflection can transform a phoneme's wave form into something completely alien. In fact, much of the speech signal is of no value to the recognition process. Worse still: any reasonably accurate mathematical representation of the entire signal would be far too large to manipulate in real time. Therefore, a manageable number of discriminating features must somehow be extracted from the wave before recognition can take place. A common scheme involves "cepstral coefficients" (cepstral is a mangled form of spectral); the recognizer collects 8,000 speech samples per second and extracts a "feature vector" of at most a few dozen numbers from each one, through a mathematical analysis process that is far beyond the scope of this article. The two primary limitations of current speech recognition technology are that it does not yet transcribe free-form speech input, and that it makes mistakes. The previous sections discussed how speech recognizers are constrained by grammars. Speech recognizers make mistakes. So do people. But recognizers usually make more. Understanding why recognizers make mistakes, the factors that lead to these mistakes and how to train users of speech recognition to minimize errors are all important to speech application developers. The reliability of a speech recognizer is most often defined by its recognition accuracy. Accuracy is usually given as a percentage and is most often the percentage of correctly recognized words. Because the percentage can be measured differently and depends greatly upon the task and the testing conditions it is not always possible to compare recognizers simply by their percentage recognition accuracy. A developer must also consider the seriousness of recognition errors: misrecognition of a bank account number or the command "delete all files" may have serious consequences. The following is a list of major factors that influence recognition accuracy.

- Recognition accuracy is usually higher in a quiet environment.
- Higher-quality microphones and audio hardware can improve accuracy.
- Users that speak clearly (but naturally) usually achieve better accuracy.
- Users with accents or atypical voices may get lower accuracy.
- Applications with simpler grammars typically get better accuracy.

5.2 ACHIEVEMENTS

The software was run with real time inputs and the security constraints and validation was performed. The software developed was implemented and tested with real data and was found to be error free. Also, it was found that the system will work correctly and successfully. We have tried to make the system maximum user friendly and light on the device. Any user with the basic knowledge of using a computer device with internet enable connectivity can access it with ease.

Chapter 6

CONCLUSION AND FUTURE SCOPE

The entire project has been developed and deployed as per the requirements. It is found to be bug free as per the testing standards that are implemented. The developed software was tested with real data and was found to work correctly and successfully. The functioning of scouts are made simpler. They are made aware of potential prospects by a simple search system. Also maintaining, verifying and updating user details and achievements are made easier. The system is highly flexible for future enhancements and it in itself has huge scope for development. It will also provide good opportunity for software developing companies and other such organizations involved in the field of software and information technology.

A vocal command based extension which can be used to perform the basic and a few intermediate functions that are otherwise done in the browser using the keyboard or mouse. Users will be free to leave their keyboards and mouses and can depend on just their own voices to perform most browser related operations such as opening a new tab, closing tabs, searching the web etc.

Chapter 7

APPENDIX

7.1 SCREEN SHOTS

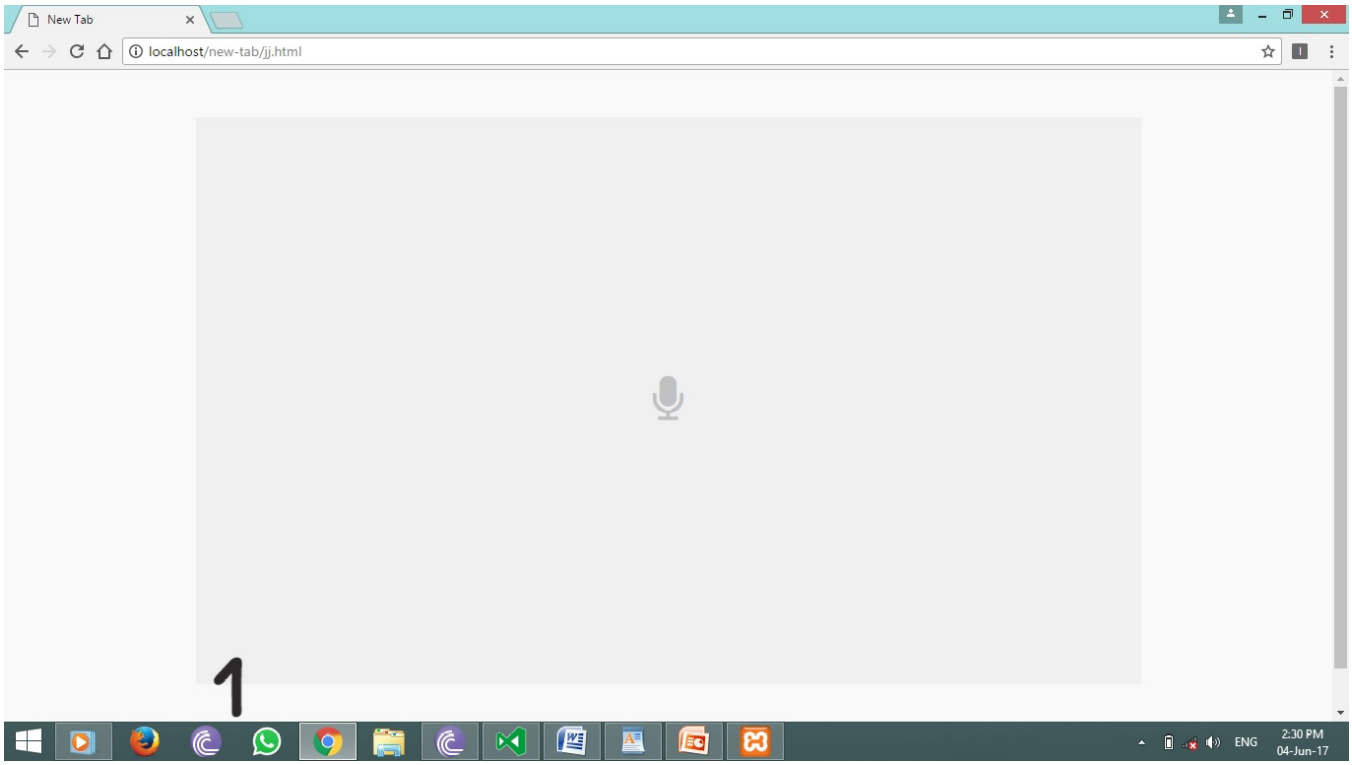

Figure 7.1: Screen short 1

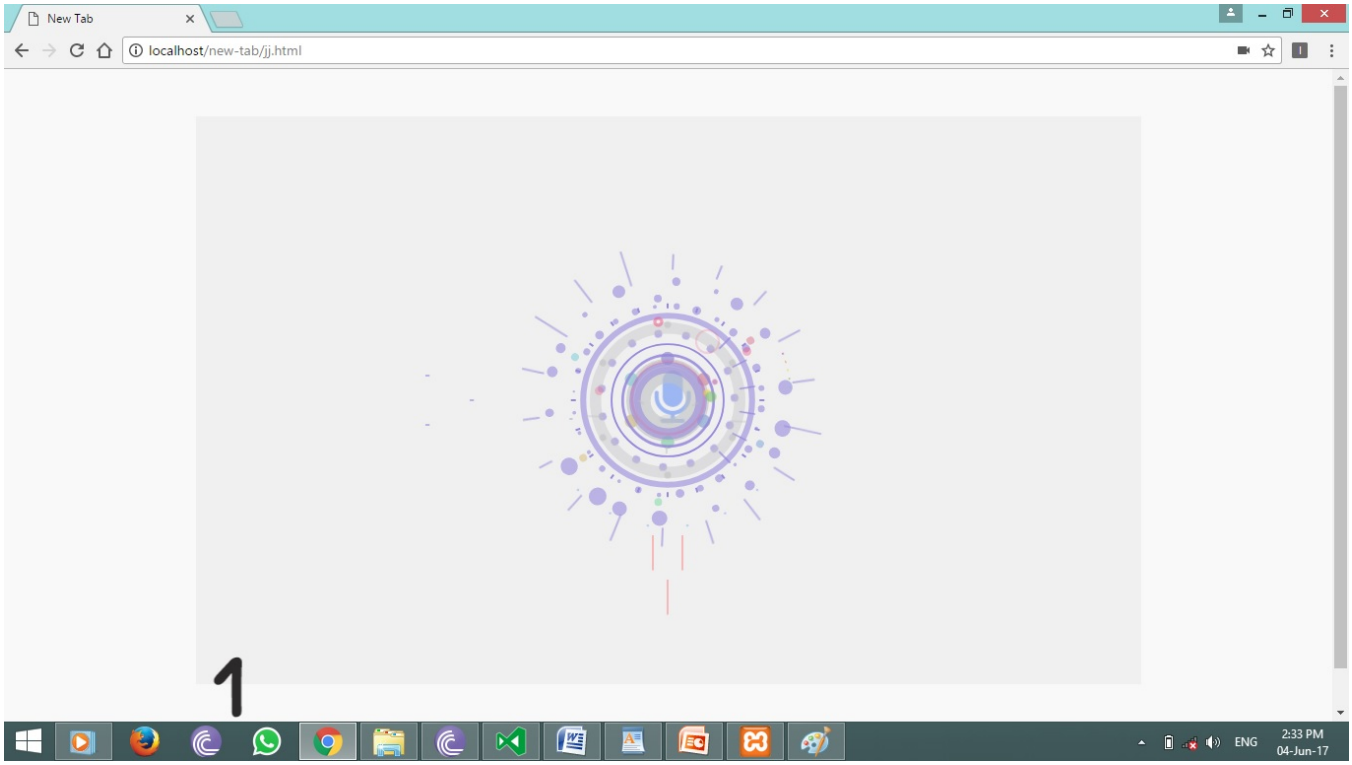

Figure 7.2: Screen short 2

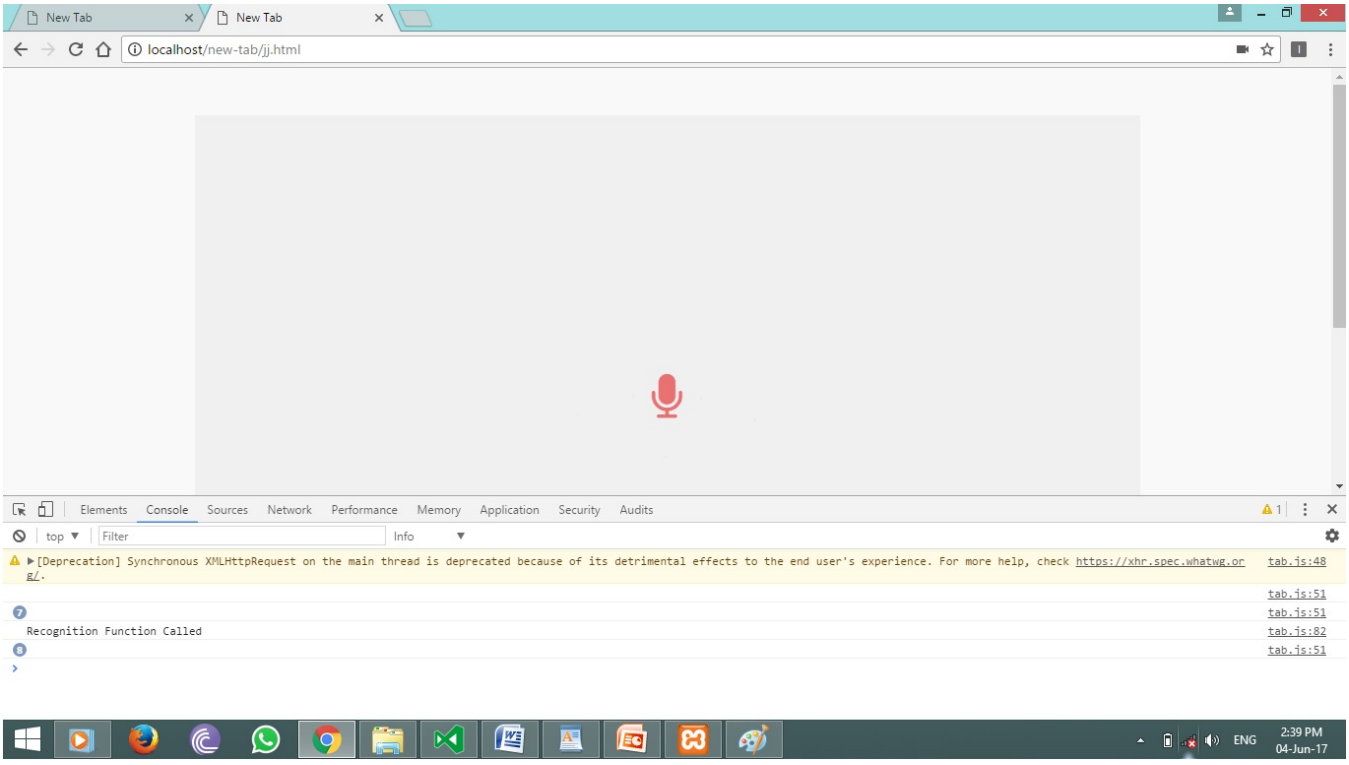

Figure 7.3: Screen short 3

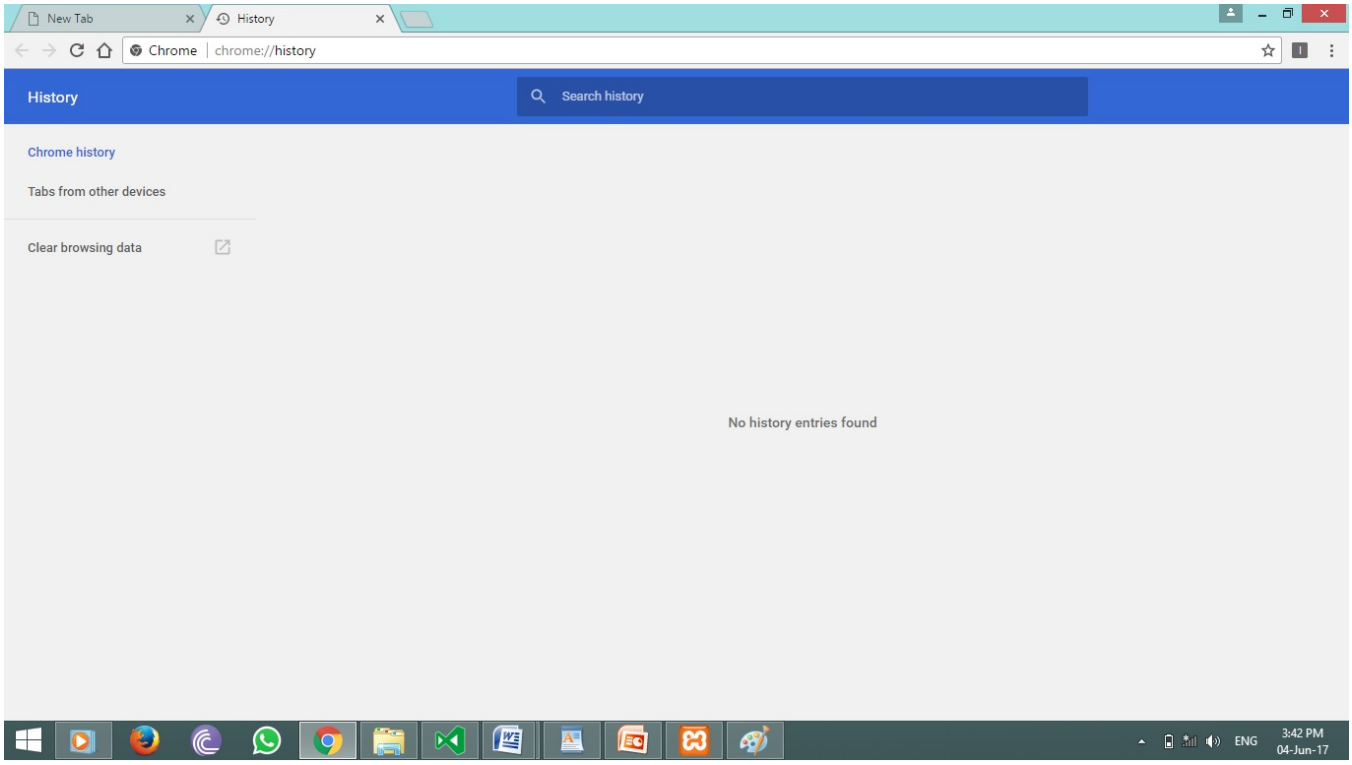

Figure 7.4: Screen short 4

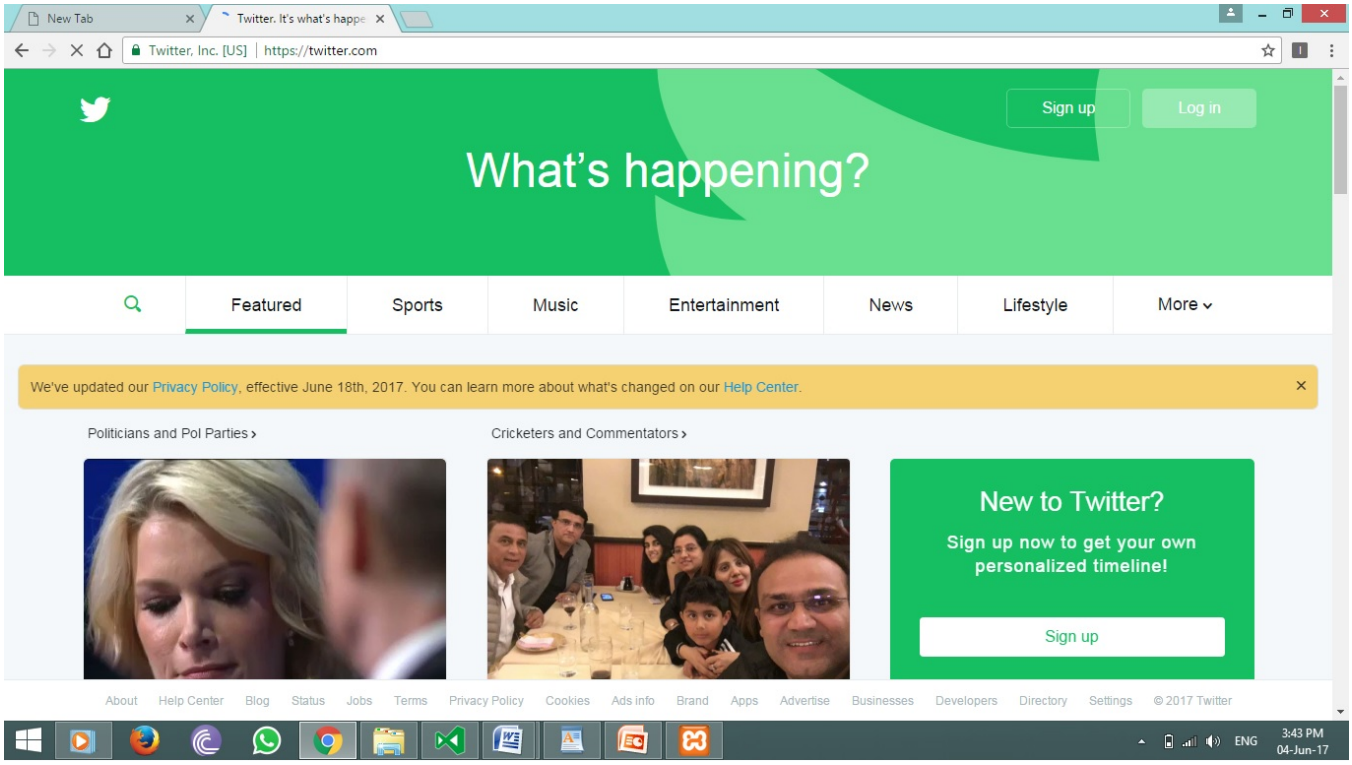

Figure 7.5: Screen short 5

Bibliography

[1] George Ornbo. The HTML5 speech recognition api. https://shapeshed.com/html5-speech-recognition-api/.

[2] David Walsh. Javascript speech recognition. https://davidwalsh.name/speech-recognition.

[3] Stackoverflow. speech recognition. https://stackoverflow.com/questions/tagged/speech-recognition.

[4] John Monyjok Maluth. *Microsoft Windows 7: A Guide to Windows 7 with Advanced Features*. CreateSpace Independent Publishing Platform, USA, 2012.

[5] Tutorials point. Javascript tutorial. https://www.tutorialspoint.com/javascript/.

[6] T. Powell and F. Schneider. *JavaScript: The Complete Reference, 2nd Edition*. Complete Reference. McGraw-Hill Education, 2004.

[7] W3 Schools. HTML. https://www.w3schools.com/html/default.asp.

[8] W3schools. CSS. https://www.w3schools.com/css/default.asp.

[9] Riaz Ahmed. *Web Development in PHP, MySQL, JavaScript, HTML & CSS: Step-by-Step Web Project*. CreateSpace Independent Publishing Platform, USA, 2014.

[10] W3 Schools. Javascript validation api. https://www.w3schools.com/js/js_validation_api.asp.

[11] Ellen Siever, Stephen Figgins, Aaron Weber, Robert Love, and Arnold Robbins. *Linux in a Nutshell (In a Nutshell (O'Reilly))*. O'Reilly Media, Inc., 2005.

[12] Google Chrome. Extensions. https://developer.chrome.com/extensions.

[13] Wikipedia. Hypertext transfer protocol. https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol.

[14] Stackoverflow. Java HTTP session. https://stackoverflow.com/questions/5140556/java-httpsession.