

“Remote Computer Administration”

*A project report submitted in partial fulfillment of the requirements
for the Award of Degree,*

**Bachelor of Technology
in
Computer Science & Engineering**

By

Nishant Shekhar	R100211033
Rimjhim Gupta	R100211043
Anshu Karn	R100211011
Ashish Sharma	R100211015

Under the Esteemed Guidance of
Ms. Chhavi Gupta



**Department of Computer Science & Engineering
Centre for Information Technology
University of Petroleum & Energy Studies
Bidholi, Via Prem Nagar, Dehradun, UK**

April – 2015



The innovation driven
E-School

CANDIDATE'S DECLARATION

We hereby certify that the project work entitled “ **Remote Computer Administration**” in partial fulfilment of the requirements for the award of the Degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING with specialization in Open Source & Open Standards and submitted to the Department of Computer Science & Engineering at Center for Information Technology, University of Petroleum & Energy Studies, Dehradun, is an authentic record of our work carried out during a period from **August, 2014** to **April, 2015** under the supervision of **Ms. Chhavi Gupta, Asst. Professor, UPES.**

The matter presented in this project has not been submitted by us for the award of any other degree of this or any other University.

Nishant Shekhar R100211033 Rimjhim Gupta R10211043 Anshu Karn R100211011 Ashish Sharma R100211015 .

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date: April 4,2015

Ms Chhavi Gupta
Project Guide

Dr. Vinay Awasthi
Program Head – B.Tech CS-OSS
Center for Information Technology
University of Petroleum & Energy Studies
Dehradun – 248 001 (Uttarakhand)

ACKNOWLEDGEMENT

We wish to express our deep gratitude to our guide **Ms. Chhavi Gupta**, for all advice, encouragement and constant support he has given us through out our project work. This work would not have been possible without his support and valuable suggestions.

We sincerely thank to our respected Program Head of the Department, **Dr. Vinay Awasthi**, for his great support in doing our project in **Area** at **CIT**.

We are also grateful to **Dr. Manish Prateek, Associate Dean** and **Dr. Kamal Bansal, Dean, CoES, UPES** for giving us the necessary facilities to carry out our project work successfully.

We would like to thank all our **friends** for their help and constructive criticism during our project work. Finally we have no words to express our sincere gratitude to our **parents** who have shown us this world and for every support they have given us.

**Nishant
Shekhar**

R100211033

**Rimjhim
Gupta**

R100211043

**Anshu
Karn**

R100211011

**Ashish
Sharma**

R100211015

ABSTRACT

“Remote Computer Administration” is an application that uses wi-fi network to connect computer and smart phone and thus provide the functionality of mouse as well as keyboard remotely.

The application provides the functionality of the mouse and keyboard through the smart phone’s web browser, provided the computer and the smart phone are connected in the same wi-fi network. The goal of project is to provide long range connectivity and remote access, unlike the limited range bluetooth connection, as well as full-duplex communication through use of web sockets.

TABLE OF CONTENTS

S.No.	Contents	Page No
	Candidate's Declaration	ii
	Acknowledgement	iii
	Abstract	iv
	Table of Contents	v
	List of Figures	vii
	List of Tables	viii
1.	Introduction	1
1.1.	History	1
1.2.	Requirement Analysis	1
1.3.	Main Objective	2
1.4.	Sub Objectives	2
2.	System Analysis	3
2.1.	Existing System	3
2.2.	Proposed System	5
2.3.	Modules	6
2.3.1	PY Mouse	6
2.3.2	Mod Py Websockets	6
2.3.3	PIP	7
2.3.4	PY Keyboard	7
2.3.5	QR	8
3.	Design	9
3.1	Client Server Architecture	9
3.2	Flowchart	10
3.3	Sequence Diagram	11
3.4	UseCase Diagram	13
3.5	Implementation	14

4. COMPONENTS	16
4.1. Web Sockets	16
4.2. Python	17
4.3. Javascript	18
4.4. JSON	19
4.5 PyQt4	20
4.6 jQuery Keypad	22
5. Testing and debugging	24
5.1 Testing tool	24
5.2 Debugging tool	25
6. Output screens	26
7. Conclusion	30
8. Future Scope	31
References	32

List of Figures

Figure Number	Title	Page Number
3.1	Client Server Architecture	9
3.2	Flowchart	10
3.3	Sequence Diagram	11
3.4	Sequence Diagram	12
3.5	Use Case Diagram	13
3.6	Data Flow Diagram	15
4.1	Python	17
4.2	JavaScript	18
4.3	JSON	19
4.4	PYQt4	20
5.1	Firebug	24
5.2	JSLint	25
6.1	O/P Screen	26
6.2	O/P Screen	27
6.3	O/P Screen	27
6.4	O/P Screen	28
6.5	O/P Screen	29
6.6	O/P Screen	29

List of tables

Table Number	Title	PageNumber
2.1	Comparison with Application	3

INTRODUCTION

1.1 History

A group of friends sitting together and enjoying movie on a projector screen at their home at a pleasant winter evening felt the need of some application that could access the mouse of their laptop without any of them getting out of bed.

Those engineering students being smart enough could surely do that by getting out of their warm bed, wish if their so called “smart phones” could do that.

There exists a number of application that provide with remote access using bluetooth, but a need for longer range of connectivity was identified that could provide a real time full-duplex communication.

1.2 Requirement Analysis

- Need felt for a handy application that provides long range connectivity of the computer with smart phone.
- Class- 2 bluetooth present in mobile phones limit the range within 10 metres of distance. Hence, a wi-fi connection could settle the range connectivity problem.
- Use of client-server architecture while solving the problem could be too helpful.
- Installation of a separate application on phone would not create any new object of convenience, on the other hand communication needs to be full duplex in nature, so web sockets could serve the purpose quiet efficiently.
- Linux system could facilitate us with the easy server connection as well as device connection.
- Also an attempt to use Open Source and Open Standards.

1.3 Main Objective

To create an application that provides connectivity of computer and smart phone through wi-fi network and thus provide the functionality of mouse as well as keyboard through mobile's web browser.

1.4 Sub Objectives

The objective is divided into the following achievable units:

- Web browser to be used as mouse touchpad, hence the use of HTML 5 Canvas.
- Establish connection between phone and computer using wi-fi network.
- Write web socket request handlers for connection.
- Authenticate the user using QR code.
- Fetch mouse co-ordinates using javascript.
- PyKeyboard module to handle the jQuery keypad.
- Transport and exchange of data using JSON.

SYSTEM ANALYSIS

2.1 Existing System

The growing popularity and spread of smart phones has changed the design of computer systems as they were known in recent years. Technological developments have enabled the creation of mobile devices with technical features previously only conceived in PC architectures or similar devices. With this evolution comes the need to integrate these devices with others so they can take actions and monitor interaction on mobile devices .Other aspect to be considered is the remote visualization mechanisms that are useful for achieve a remote display of the devices.

There were android applications using the remote mouse but lacked real time communication.We have developoed a real time full duplex communication.

Here is the comparison between other existing applications and 'Remote Computer Administration':

Table 2.1: Comparison with existing system

Features	Other Applications	Remote Computer Administration
Latency	More latency (0.5-1sec approx.)	No significant latency (<1/10 sec approx.)
Connection	Socket	Web Socket
Range	Short Range (~10 m) (Bluetooth)	Long Range (~32m) (Wi-Fi)
Language	Java	Python
Development	Comparitively slow development	Fast Development (developed in python)

Comparison with similar applications that provide few of the features of “Remote Computer Administration”:

1. Andro MouseServer 2.5:

Provides the functionality of only mouse and connectivity is required through bluetooth. Remote Computer Administration on the other hand provides functionality of keyboard in extra and also connectivity through wifi provides a wider range of connectivity.

2. Mouse Kit:

It provides the functionality of mouse as well as keyboard but connection is required through bluetooth and hence limiting the range of connectivity. Also an application needs to be installed on the user’s phone unlike our application.

3. Ultimate Mouse Lite:

Provides functionality of mouse as well as keyboard and also connectivity through wifi is done but comparatively more latency and installation of an application on the user’s phone puts Remote Computer Administration forward.

2.2 Proposed System

Our proposed system has following features

- Establish connection between phone and computer using wi-fi network.
- Authenticate the user through scanning the QR code.
- Web socket request handlers that establishes connection.
- JSON to transport and exchange of data.
- Use of HTML 5 Canvas for web browser to be used as touchpad.
- Javascript to fetch mouse co-ordinates.
- PyKeyboard module to handle the jQuery keypad.

2.3 Modules

Modules of the system includes

2.3.1 PY Mouse

To use this plugin put the python files somewhere on the python path. The next example illustrates basic usage:

```
# import the module
from pymouse import PyMouse

# instantiate an mouse object
m = PyMouse()

# move the mouse to int x and int y (these are absolute positions)
m.move(200, 200)

# click works about the same, except for int button possible values are 1: left, 2: middle, 3: right
m.click(500, 300, 1)

# get the screen size
m.screen_size()
# (1024, 768)

# get the mouse position
m.position()
# (500, 300)
```

2.3.2 Mod PY websocket

- ✓ The py websocket project aims to provide a WebSocket standalone server and a WebSocket extension for Apache HTTP Server, mod_pywebsocket.
- ✓ pywebsocket is intended for **testing** or **experimental** purposes. To run with Apache HTTP Server, mod_python is required. For wss, mod_ssl is also required.
- ✓ Pywebsocket supports RFC 6455 (and some legacy protocols) and the following extension.
- ✓ WebSocket Per-message Compression (permessage-deflate)

2.3.3 PIP 1.5.6

- ✓ A tool for installing and managing Python packages.
- ✓ To install a package from pip

```
$ pip install SomePackage
[...]
Successfully installed SomePackage
```

To view the packages installed

```
$ pip show --files SomePackage
Name: SomePackage
Version: 1.0
Location: /my/env/lib/pythonx.x/site-packages
Files:
  ../somepackage/__init__.py
  [...]
```

2.3.4 PY Keyboard

To use this plugin put the python files somewhere on the python path. The next example illustrates basic usage:

```
if sys.platform.startswith('java'):
    from .java_ import PyKeyboard

elif sys.platform == 'darwin':
    from .mac import PyKeyboard, PyKeyboardEvent

elif sys.platform == 'win32':
    from .windows import PyKeyboard, PyKeyboardEvent

else:
    from .x11 import PyKeyboard, PyKeyboardEvent
```

2.3.5 QR Code

We are generating the QR code using QR module. This is generated for the user authentication .

The QR code is generated on the PC screen and scanned by our smart phone.

Generating QR Code:

It is done for generating server's address . The interface's address is extracted through os module.

We take the ip address of the interface that is connected to internet. The QR module generates the QR image that is drawn on the canvas in the main loop of our program.

DESIGN

3.1 Client Server Architecture

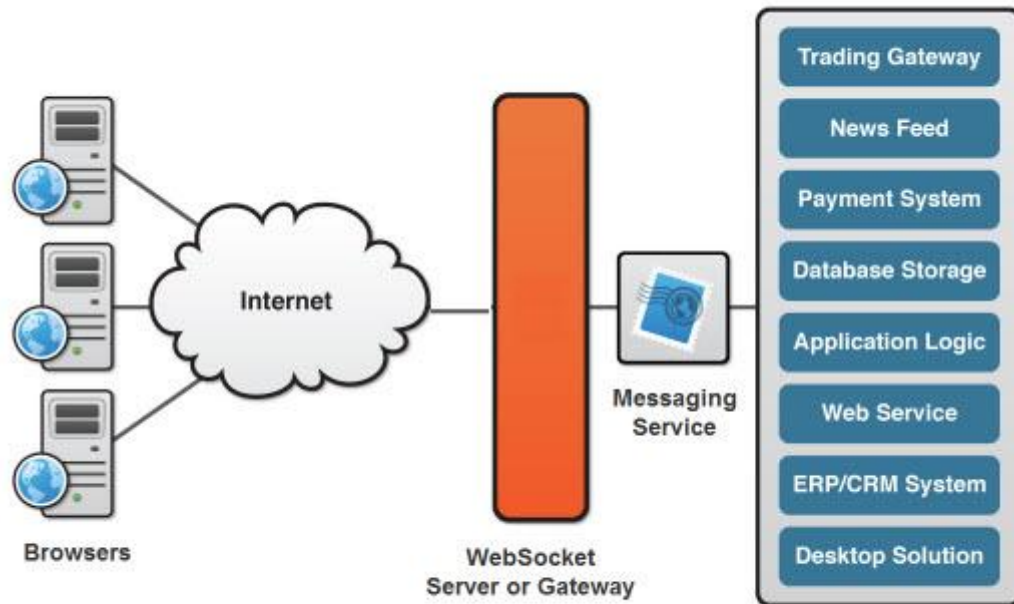


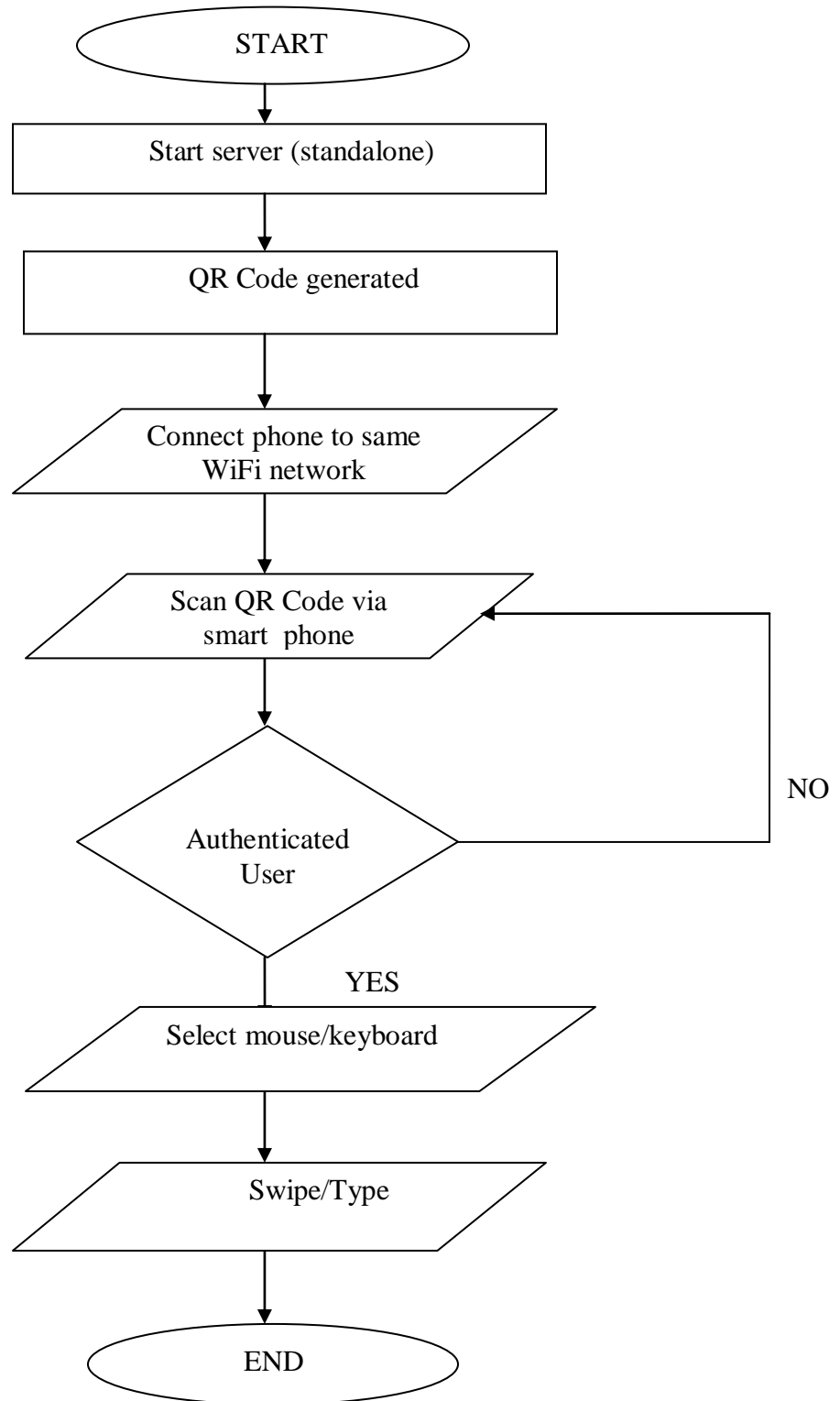
Fig No 3.1: Client Server Architecture

Web mouse uses pywebsocket as its server . pywebsocket is a experimental server written and maintained by google , it is written in python programming language. We connect the browser and the server via web socket protocol for achieving rela time full duplex communication . Web Socket also provide api which is accessed by javascript on the client side . java script sends mouse movement data in JSON format and is decoded at the server end in respective web socket handler.

CLIENT : WEB BROWSER (with WebSocket support)

SERVER : pywebsocket

3.2 Flowchart



3.3 Sequence Diagram

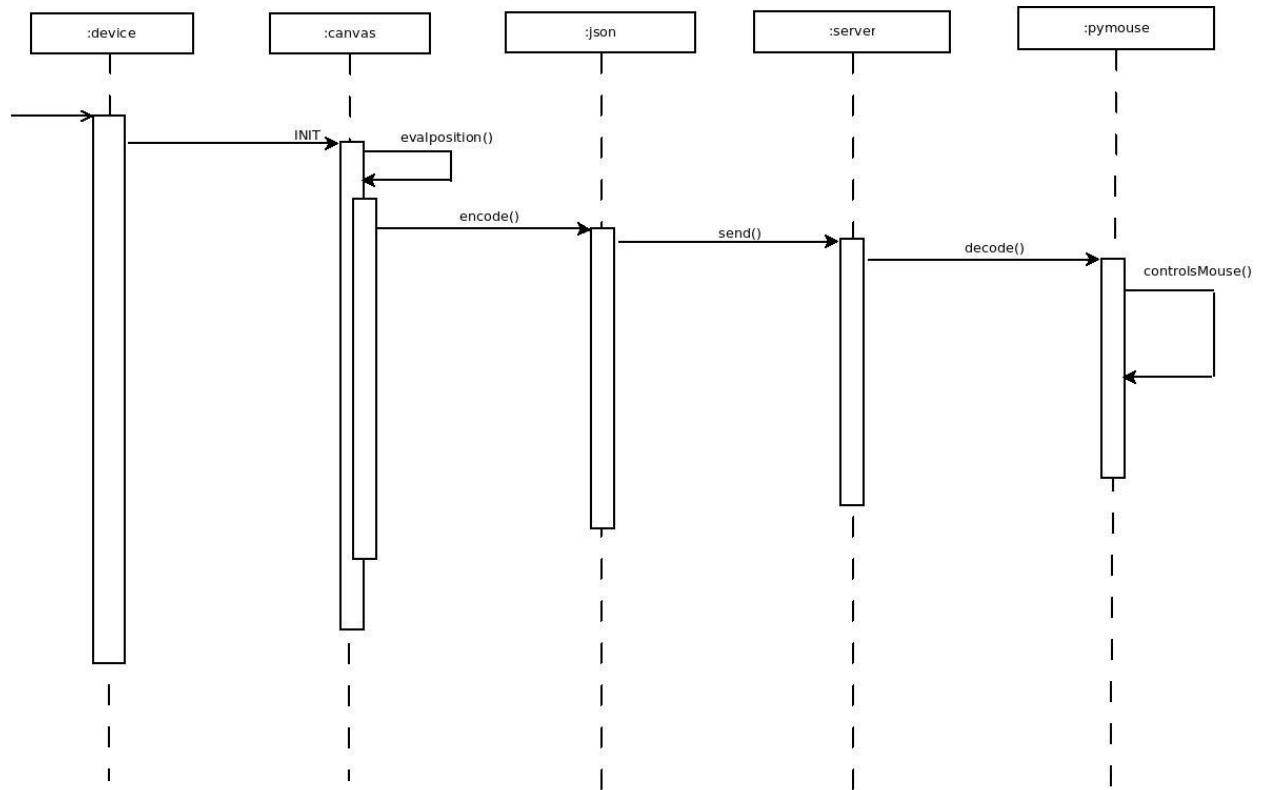


Fig No 3.3 : Sequence Diagram for Mouse

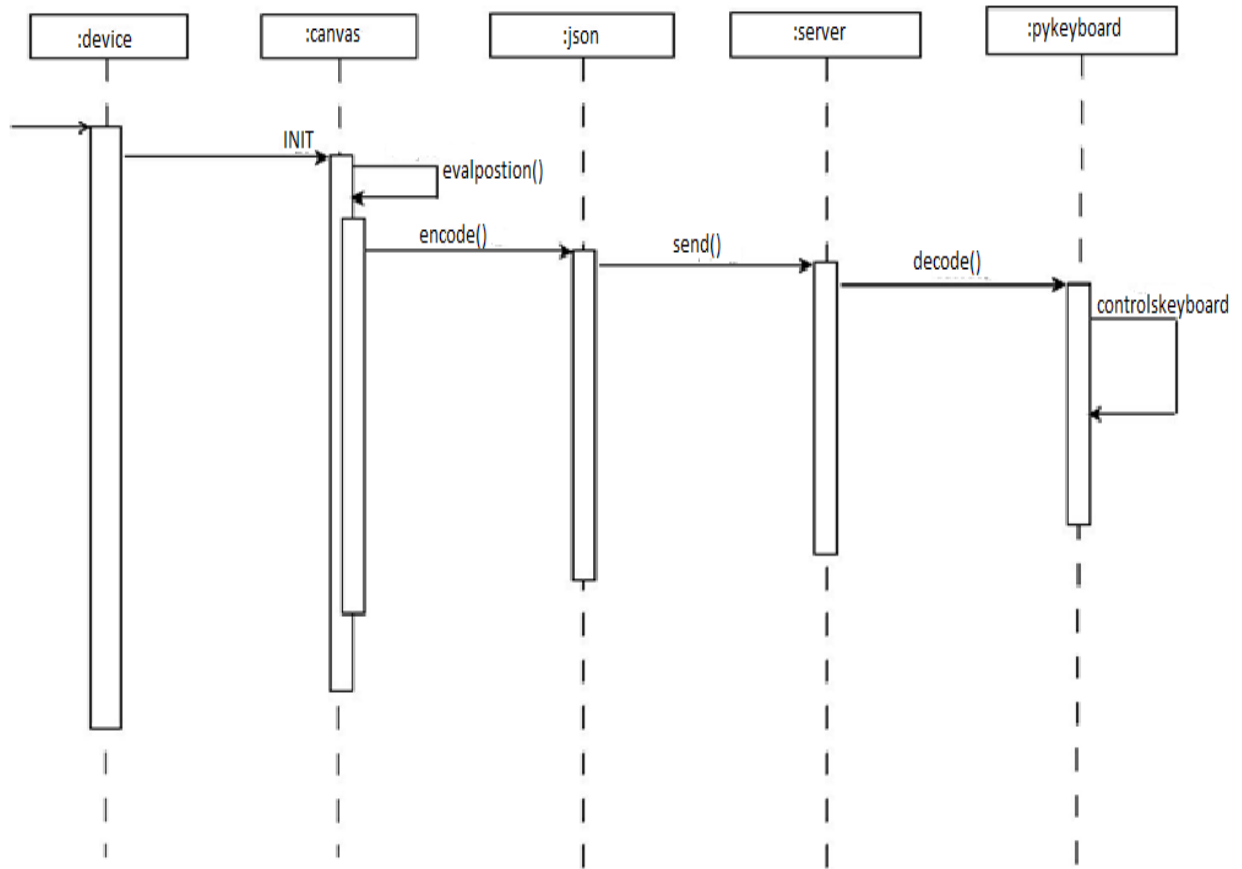


Fig 3.4: Sequence diagram for PY Keyboard

3.4 UseCase Diagram

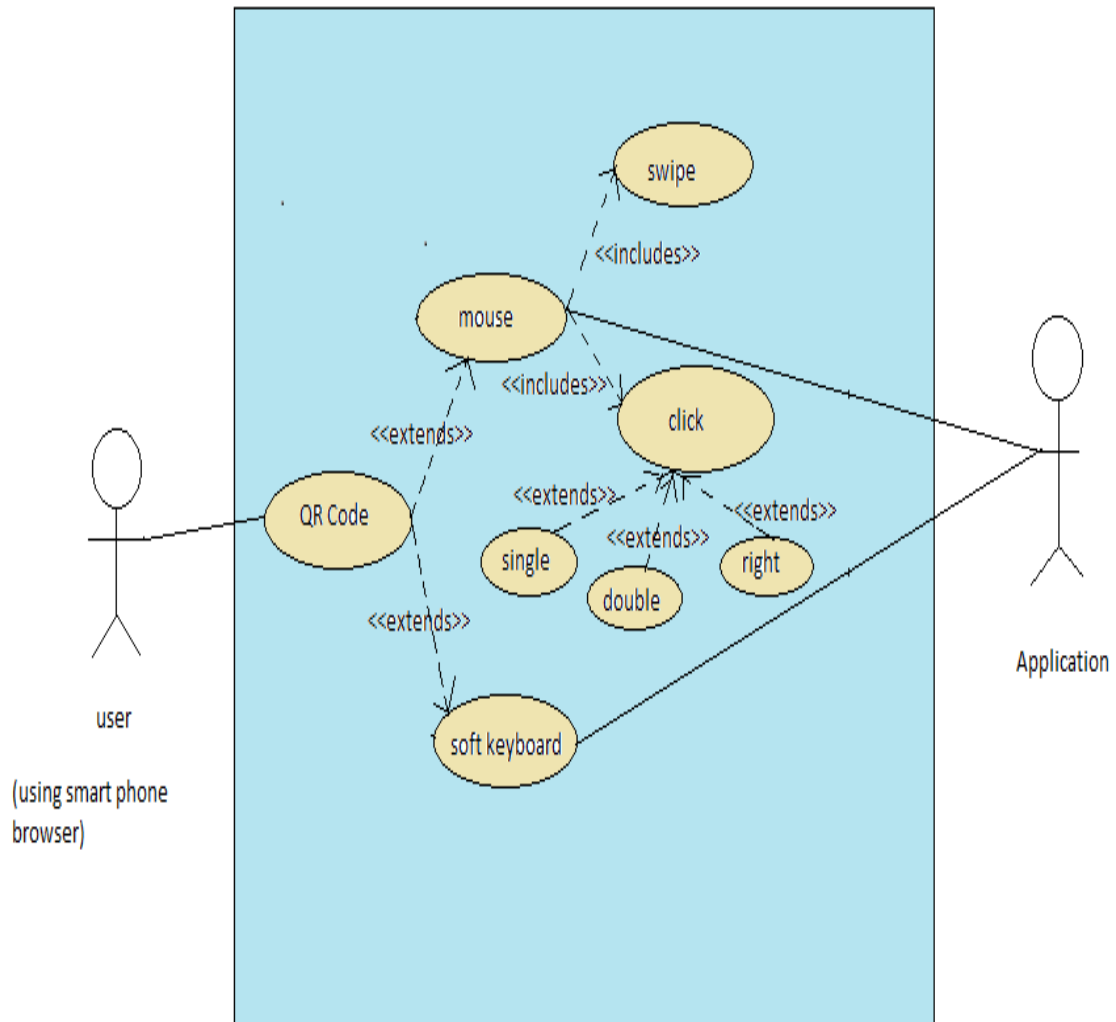


Fig No 3.5 : Use Case Diagram

3.5 Implementation

Remote computer administration is implemented as a client server model in which the client can be any web browser with web socket support. Web Socket is a communication protocol which supports full duplex bi-directional communication. Web socket provides an API also which can be accessed through JavaScript. `mod_pywebsocket` is used for server which in our case is being used as standalone server, we provide the port number and the websocket request handler directory, request handlers are in the format `<<name>>_wsh`. The client evaluates the mouse position through event in our landing page which then is encoded to JSON format and is sent to server in the form of request. The server decodes the JSON and data in it.

The data is the argument of `pymouse`. module `pymouse` uses X11 protocols and is a python implementation for providing interfaces to them. by using X11 interface it controls the mouse.

Authentication of user is done through scanning of QR code by smartphone camera. The QR module of the application obtains the ip address of the interface that is connected to internet. It passes the obtained ip as string and generates the corresponding QR code. This is further used as connection as well as authentication.

Working of keyboard is very similar to mouse, here the module `PYkeyboard` implements the functionality of keyboard with the help of web socket handlers.

Data Flow Diagram:

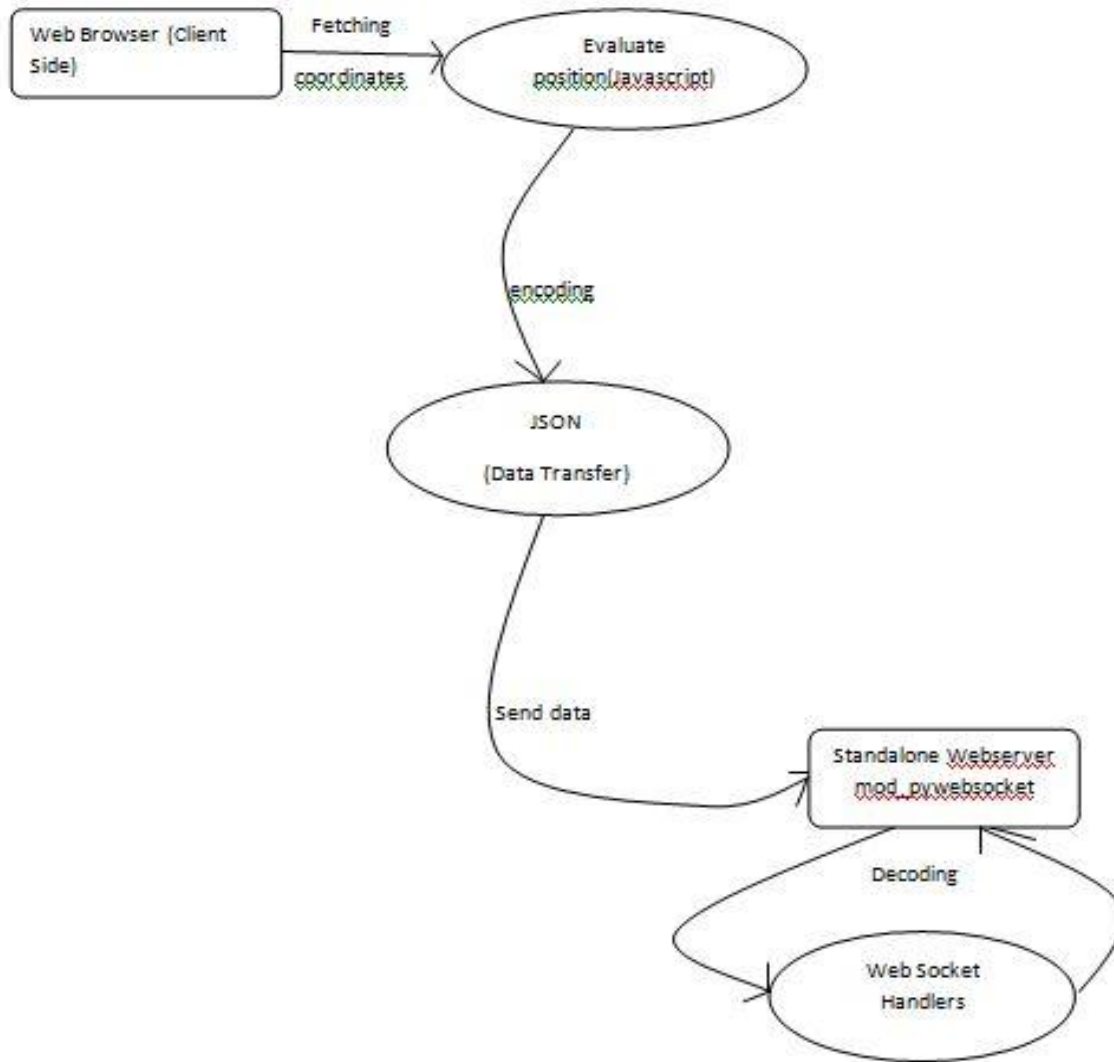


Fig 3.6: Data flow diagram

COMPONENTS

4.1 Web Socket

Utilizing the WebSocket interface couldn't be simpler. To connect to an end-point, just create a new WebSocket instance, providing the new object with a URL that represents the end-point to which you wish to connect, as shown in the following example. Note that a ws:// and wss:// prefix are proposed to indicate a WebSocket and a secure WebSocket connection, respectively.[2]

```
var myWebSocket = new WebSocket("ws://www.websockets.org");
```

A WebSocket connection is established by upgrading from the HTTP protocol to the WebSockets protocol during the initial handshake between the client and the server. The connection itself is exposed via the "onmessage" and "send" functions defined by the WebSocket interface.

Before connecting to an end-point and sending a message, you can associate a series of event listeners to handle each phase of the connection life-cycle as shown in the following example

```
myWebSocket.onopen = function(evt) { alert("Connection open ..."); };  
myWebSocket.onmessage = function(evt) { alert("Received Message: " + evt.data); };  
myWebSocket.onclose = function(evt) { alert("Connection closed."); };
```

To send a message to the server, simply call "send" and provide the content you wish to deliver. After sending the message, call "close" to terminate the connection, as shown in the following example. As you can see, it really couldn't be much easier

```
myWebSocket.send("Hello WebSockets!"); myWebSocket.close();
```

4.2 Python



Fig No 4.1 : Python Logo

Python is a widely used general-purpose, high-level programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C++ or Java. The language provides constructs intended to enable clear programs on both a small and large scale.

Python is object-oriented, imperative and functional programming or procedural styles. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library.

Python interpreters are available for installation on many operating systems, allowing Python code execution on a majority of systems. Using third-party tools, such as Py2exe or Py installer, Python code can be packaged into stand-alone executable programs for some of the most popular operating systems, allowing for the distribution of Python-based software for use on those environments without requiring the installation of a Python interpreter.[3] [4]

4.3 JavaScript

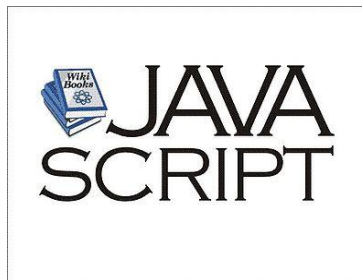


Fig No 4.2 JavaScript Logo

It is a dynamic computer programming language. It is most commonly used as part of web browsers, whose implementations allow client-side scripts to interact with the user, control the browser, communicate asynchronously, and alter the document content that is displayed. It is also used in server-side network programming with frameworks such as Node.js, game development and the creation of desktop and mobile applications.[1] [2]

JavaScript is classified as a prototype-based scripting language with dynamic typing and first-class functions. This mix of features makes it a multi-paradigm language, supporting object-oriented, imperative, and functional programming styles.

Despite some naming, syntactic, and standard library similarities, JavaScript and Java are otherwise unrelated and have very different semantics. The syntax of JavaScript is actually derived from C, while the semantics and design are influenced by self and scheme programming languages.

4.4 JSON

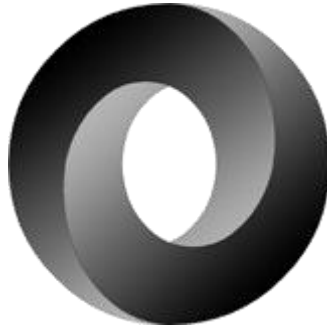


Fig No 4.3 JSON Logo

JSON or **JavaScript Object Notation**, is an open standard format that uses human-readable text to transmit data objects consisting of attribute–value pairs. It is used primarily to transmit data between a server and web application, as an alternative to XML.

Although originally derived from the JavaScript scripting language, JSON is a language-independent data format. Code for parsing and generating JSON data is readily available in a large variety of programming languages.

The JSON format was originally specified by Douglas Crockford. It is currently described by two competing standards, RFC 7159 and ECMA-404. The ECMA standard is minimal, describing only the allowed grammar syntax, whereas the RFC also provides some semantic and security considerations.[2] [4]

4.5 PyQt4



Fig 4.4 PYQt Logo

PyQt4 is a toolkit for creating GUI applications. It is a blending of Python programming language and the successful Qt library. Qt library is one of the most powerful GUI libraries. PyQt4 is implemented as a set of Python modules. It has 440 classes and 6000 functions and methods. It is a multiplatform toolkit which runs on all major operating systems, including Unix, Windows, and Mac OS. PyQt4 is dual licensed. Developers can choose between a GPL and a commercial license. Previously, GPL version was available only on Unix. Starting from PyQt version 4, GPL license is available on all supported platforms. [6]

PyQt4's classes are divided into several modules:

- QtCore
- QtGui
- QtNetwork
- QtXml
- QtSvg

- QtOpenGL
- QSql

The **QtCore** module contains the core non GUI functionality. This module is used for working with time, files and directories, various data types, streams, URLs, mime types, threads or processes.

The **QtGui** module contains the graphical components and related classes. These include for example buttons, windows, status bars, toolbars, sliders, bitmaps, colours, and fonts.

The **QtNetwork** module contains the classes for network programming. These classes facilitate the coding of TCP/IP and UDP clients and servers by making the network programming easier and more portable.

The **QtXml** contains classes for working with XML files. This module provides implementation for both SAX and DOM APIs.

The **QtSvg** module provides classes for displaying the contents of SVG files. Scalable Vector Graphics (SVG) is a language for describing two-dimensional graphics and graphical applications in XML.

The **QtOpenGL** module is used for rendering 3D and 2D graphics using the OpenGL library. The module enables seamless integration of the Qt GUI library and the OpenGL library.

The **QtSql** module provides classes for working with databases.

4.6 jQuery Keypad

This attaches a popup keyboard to a text field for mouse-driven entry or adds an inline keypad in a division or span.

Usage

- Include the jQuery library (1.7+) in the head section .

```
<script type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.0/jquery.min.js"></script>
```

- Download and include the jQuery Keypad CSS and JavaScript in the head section.

```
<link type="text/css" href="css/jquery.keypad.css" rel="stylesheet">
<script type="text/javascript" src="js/jquery.plugin.js"></script>
<script type="text/javascript" src="js/jquery.keypad.js"></script>
```

- Connect the keypad functionality to your input fields.

```
$(selector).keypad();
```

- You can include custom settings as part of this process.

```
$(selector).keypad({prompt: 'Enter here'});
```

Default Settings

The keypad functionality can easily be added to a text field with appropriate default settings. Use it with a password field for more secure text entry. You can also remove the keypad widget if it is no longer required.

The defaults are:

- Text is in English
- Numeric keys only are shown
- Only the keypad may be used for entry

You can override the defaults globally as shown below:

```
$.keypad.setDefaults({prompt: 'Hide/Show keyboard'});
```

Invocation

You can trigger the keypad by focus on the field (the default), via an associated button, or by both. Disable or enable the fields and their triggers via a command.

Inline Keypad

You can present the keypad inline by applying it to a `div` or `span`. Make use of the value entered via the `onKeyPress` or `onClose` callbacks. Or connect it with an external input field via the `target` setting.

Layouts

You can alter the layout of the keypad by specifying the keys per row as a string array. Use special characters for controls:

- `$.keypad.CLOSE` - close the keypad
- `$.keypad.CLEAR` - erase the entire field
- `$.keypad.BACK` - erase the previous character
- `$.keypad.SHIFT` - toggle between upper and lower case
- `$.keypad.SPACE_BAR` - an extended space key
- `$.keypad.ENTER` - the *Enter* key
- `$.keypad.TAB` - the *Tab* key
- `$.keypad.SPACE` - an empty space
- `$.keypad.HALF_SPACE` - half an empty space

Long Keys

You can have keys with more than a single character on them by providing a `separator` setting and delimiting the layout with it.

TESTING AND DEBUGGING

5.1 Firebug

It is a free and open-source web browser extension that facilitates the live debugging, editing, and monitoring of any website's CSS, HTML, DOM, XHR, and JavaScript.

It Uses the most advanced JavaScript debugger available for any browser.

Firebug includes a powerful JavaScript debugger that lets you pause execution at any time and see what each variable looked like at that moment. If your code is a little sluggish, use the JavaScript profiler to measure performance.



Fig No 5.1 Firebug

5.2 Debugging

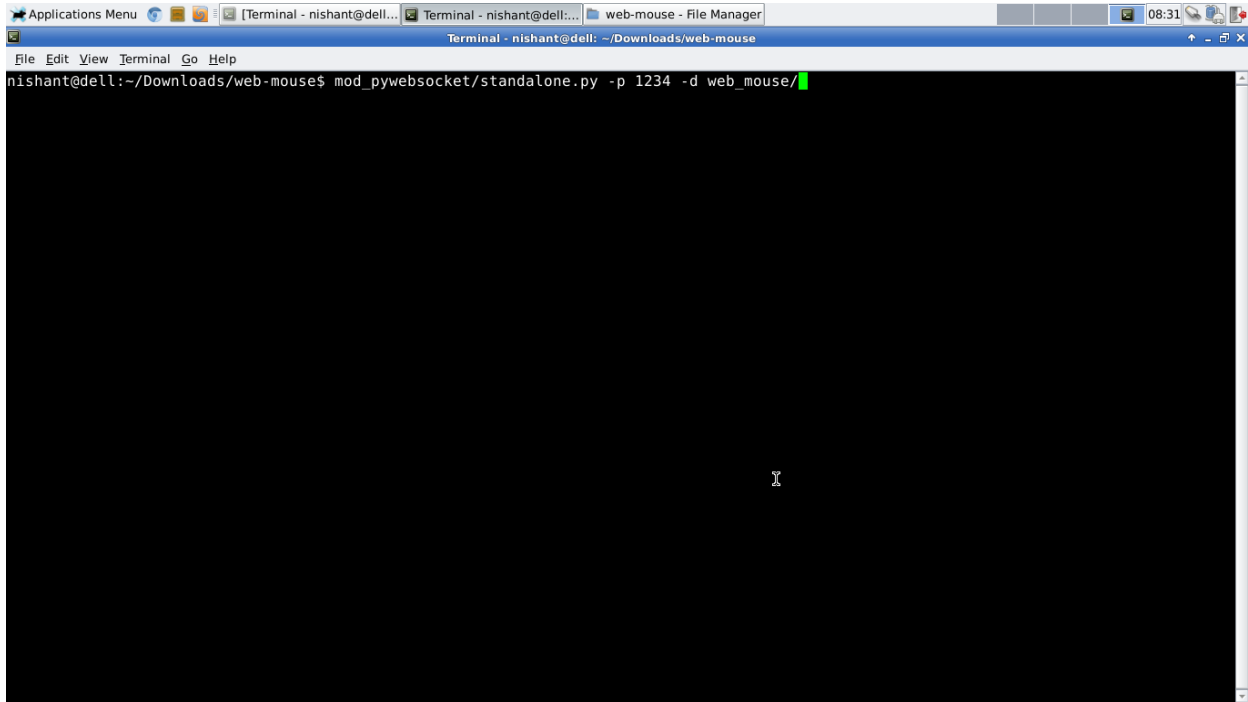
JSLint – The JavaScript Code Quality Tool

An online JavaScript program that looks for problems in JavaScript programs. JSLint takes a JavaScript source and scans it – if it finds a problem, it returns a message describing the problem and an approximate location within the source.



Fig No 5.2 JSlint

OUTPUT SCREENS

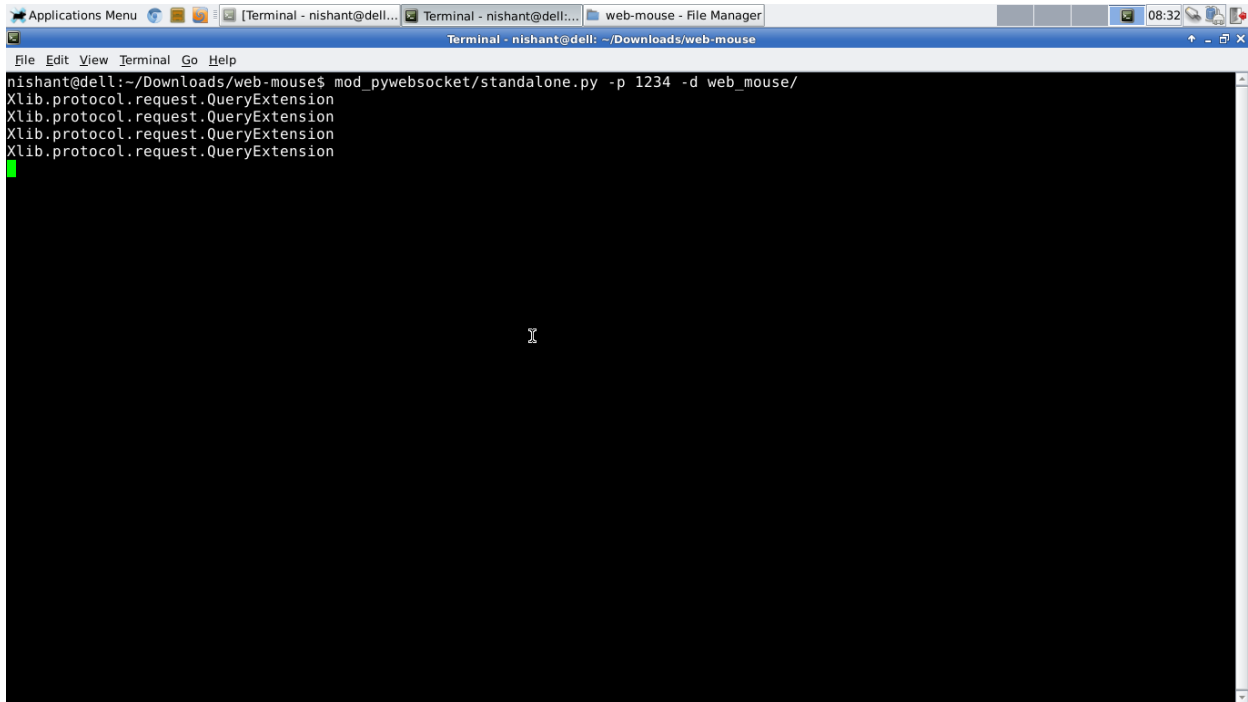


The image shows a terminal window with the following content:

```
Applications Menu [Terminal - nishant@dell... Terminal - nishant@dell... web-mouse - File Manager] 08:31  
Terminal - nishant@dell: ~/Downloads/web-mouse  
File Edit View Terminal Go Help  
nishant@dell:~/Downloads/web-mouse$ mod_pywebsocket/standalone.py -p 1234 -d web_mouse/
```

Fig 6.1 Output Screen : Running WEB MOUSE server

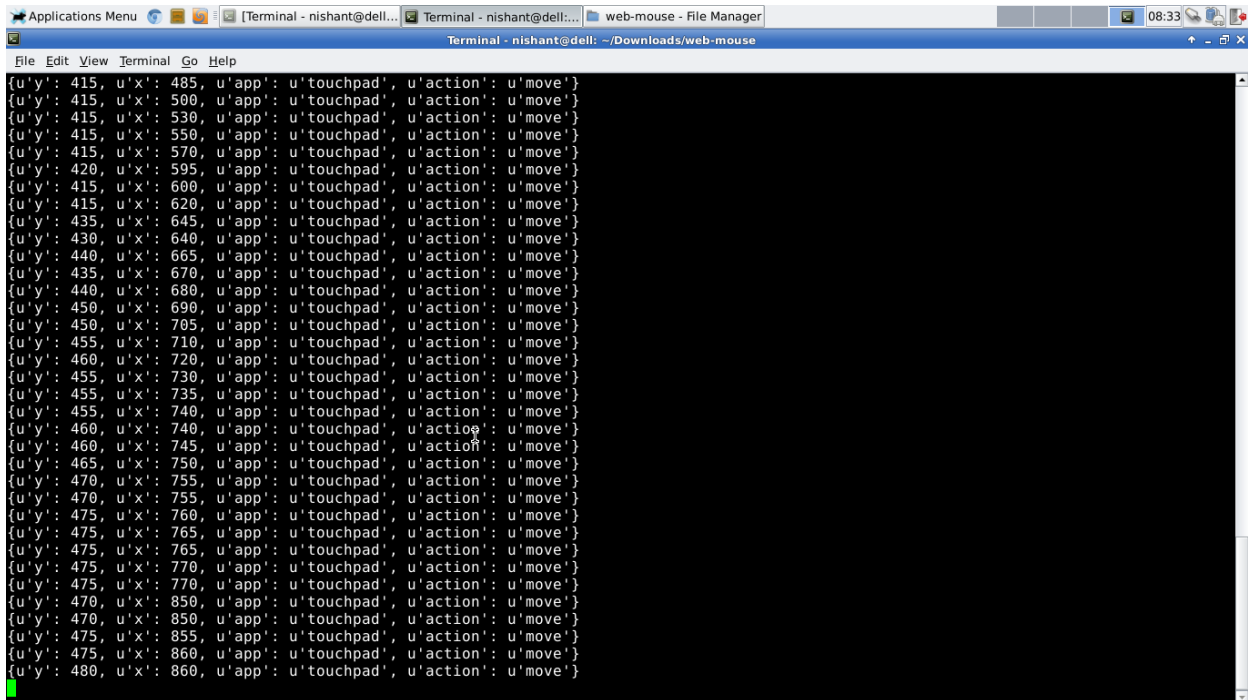
Running the web mouse and entering the socket number



```
Applications Menu [Terminal - nishant@dell... Terminal - nishant@dell... web-mouse - File Manager] 08:32
Terminal - nishant@dell: ~/Downloads/web-mouse
File Edit View Terminal Go Help
nishant@dell:~/Downloads/web-mouse$ mod_pywebsocket/standalone.py -p 1234 -d web_mouse/
Xlib.protocol.request.QueryExtension
Xlib.protocol.request.QueryExtension
Xlib.protocol.request.QueryExtension
Xlib.protocol.request.QueryExtension
```

Fig 6.2 Output Screen : Running WEB MOUSE server

Execution Started



```
Applications Menu [Terminal - nishant@dell... Terminal - nishant@dell... web-mouse - File Manager] 08:33
Terminal - nishant@dell: ~/Downloads/web-mouse
File Edit View Terminal Go Help
{u'y': 415, u'x': 485, u'app': u'touchpad', u'action': u'move'}
{u'y': 415, u'x': 500, u'app': u'touchpad', u'action': u'move'}
{u'y': 415, u'x': 530, u'app': u'touchpad', u'action': u'move'}
{u'y': 415, u'x': 550, u'app': u'touchpad', u'action': u'move'}
{u'y': 415, u'x': 570, u'app': u'touchpad', u'action': u'move'}
{u'y': 420, u'x': 595, u'app': u'touchpad', u'action': u'move'}
{u'y': 415, u'x': 600, u'app': u'touchpad', u'action': u'move'}
{u'y': 415, u'x': 620, u'app': u'touchpad', u'action': u'move'}
{u'y': 435, u'x': 645, u'app': u'touchpad', u'action': u'move'}
{u'y': 430, u'x': 640, u'app': u'touchpad', u'action': u'move'}
{u'y': 440, u'x': 665, u'app': u'touchpad', u'action': u'move'}
{u'y': 435, u'x': 670, u'app': u'touchpad', u'action': u'move'}
{u'y': 440, u'x': 680, u'app': u'touchpad', u'action': u'move'}
{u'y': 450, u'x': 690, u'app': u'touchpad', u'action': u'move'}
{u'y': 450, u'x': 705, u'app': u'touchpad', u'action': u'move'}
{u'y': 455, u'x': 710, u'app': u'touchpad', u'action': u'move'}
{u'y': 460, u'x': 720, u'app': u'touchpad', u'action': u'move'}
{u'y': 455, u'x': 730, u'app': u'touchpad', u'action': u'move'}
{u'y': 455, u'x': 735, u'app': u'touchpad', u'action': u'move'}
{u'y': 455, u'x': 740, u'app': u'touchpad', u'action': u'move'}
{u'y': 460, u'x': 740, u'app': u'touchpad', u'action': u'move'}
{u'y': 460, u'x': 745, u'app': u'touchpad', u'action': u'move'}
{u'y': 465, u'x': 750, u'app': u'touchpad', u'action': u'move'}
{u'y': 470, u'x': 755, u'app': u'touchpad', u'action': u'move'}
{u'y': 470, u'x': 755, u'app': u'touchpad', u'action': u'move'}
{u'y': 475, u'x': 760, u'app': u'touchpad', u'action': u'move'}
{u'y': 475, u'x': 765, u'app': u'touchpad', u'action': u'move'}
{u'y': 475, u'x': 765, u'app': u'touchpad', u'action': u'move'}
{u'y': 475, u'x': 770, u'app': u'touchpad', u'action': u'move'}
{u'y': 475, u'x': 770, u'app': u'touchpad', u'action': u'move'}
{u'y': 470, u'x': 850, u'app': u'touchpad', u'action': u'move'}
{u'y': 470, u'x': 850, u'app': u'touchpad', u'action': u'move'}
{u'y': 475, u'x': 855, u'app': u'touchpad', u'action': u'move'}
{u'y': 475, u'x': 860, u'app': u'touchpad', u'action': u'move'}
{u'y': 480, u'x': 860, u'app': u'touchpad', u'action': u'move'}
```

Fig 6.3 Output Screen : Running WEB MOUSE server

Using web mouse

Mobile Browser with websocket support)



Fig 6.4 Output Screen : Running WEB MOUSE server



Fig 6.5 GUI for connecting



Fig 6.6: QR authentication screen

LIMITATIONS & FUTURE SCOPE

Limitations

- Requires wi-fi network for connectivity.
- HTML5 enabled web browser is required on the smartphone.
- Can be used in same network only.
- Linux required as operating system on the laptop/desktop.

Future Scope

- Connectivity through different network like the application team viewer.
- Compatibility with other operating systems.
- Control of volume of laptop through phone

CONCLUSION

Remote access provides the users with convenience and cost effectiveness. There exist a number of applications in the market that provide remote access through bluetooth and thus limiting the range of connectivity.

Remote Computer Administration provides all the major advantages of remote access through wi-fi connectivity and long ranged connection without installing an extra application on phone.

Our work provides convenience to desktops/laptops users and save money of customers. It provides best cost effective solution to their problems and key highlight is to have multitasking ability. It will also threaten the most of peripheral developer industries.

By using websockets protocol we are not only providing full duplex communication but making the app adaptable for new generation web technologies.

REFERENCES

- [1] Michael Morrison, Head First JavaScript, 1005 Gravenstein Highway North, Sebastopol, O'Reilly Media, Dec (2007).
- [2] Vanessa Wang| Frank Salim| Peter MoskovitsThe Definitive Guide to HTML5 Web Socket,Washington DC, Apress Media, Feb(2005).
- [3] Pepijendevos,"Documentation-Introduction",PyMouse.cross-platform mouse control with python,Oct-8,2009
- [4] Python software foundation, "PY websockets", Python.websockets,Dec-13,2014.
- [5] Bhakti Mehta|Masoud Kalali, JavaScript and JSON essentials, Ohio, Packt Publishing, Jan (2013).
- [6] Zetcode."Introduction to PYQt4".PYQt4.Jan-27,2015