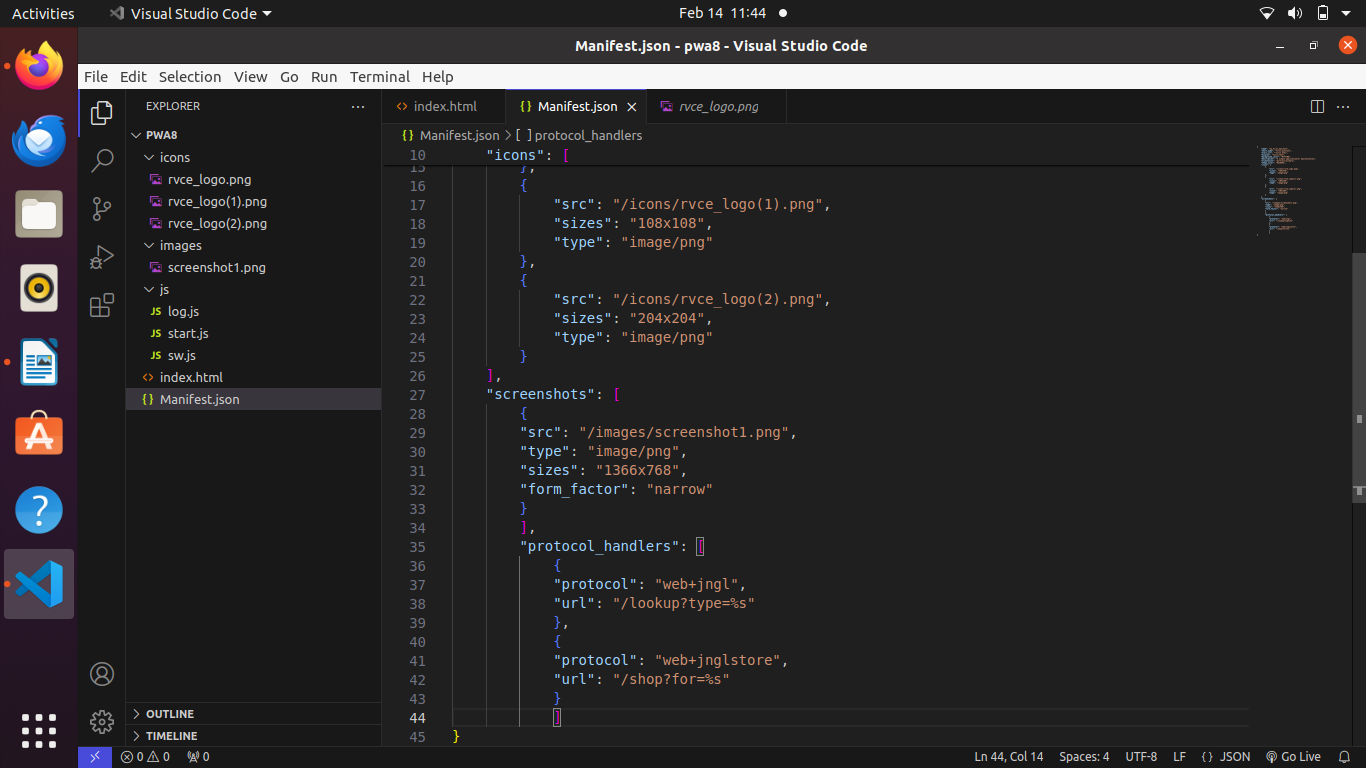
Program 8:

Create a login page to authenticate a user using PWA with Manifest file

Solution: RVCE New Logo, Screen shot can be taken from your desktop



 **Folder and File details:**

ICON Folder: contain 3 pictures

Images Folder: 1 screenshot

js Folder: 3 files, log.js, start.js, sw.js

index.html

manifest.json

**log.js**

localStorage.setItem("username","rvce");

localStorage.setItem("passme1","rvce");

function verify(){

let username = document.getElementById("username").value;

let password = document.getElementById("password").value;

let un = localStorage.getItem("username");

let pas = localStorage.getItem("passme1");

alert(username+" "+password+" "+un+" "+pas);

let stun =un.localeCompare(username);

let spass =pas.localeCompare(password)

if( stun !=-1 && spass!=-1 ){

alert("authentication done");

}else{

alert("authentication failure");

}

}

**start.js**

if ('serviceWorker' in navigator) {

navigator.serviceWorker

.register("/js/sw.js")

.then((reg)=>console.log("registered",reg))

.catch((err)=>console.log("err",err));

} else {

console.log('No service worker support in this browser');

}

**sw.js**

// method that runs when service worker is installed first time

self.addEventListener('install', event => {

console.log('service worker installing');

// loading static html into cache first

self.skipWaiting();

})

// method that runs when the service worker is activated

self.addEventListener('activate', event => {

console.log('service worker activating...')

})

// method that will run when the app makes fetch calls

self.addEventListener('fetch', event => {

console.log('fetching', event.request.url);

});

index.html

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

<link rel="manifest" href="./Manifest.json">

</head>

<body>

<script src="/js/log.js"></script>

<script src="/js/start.js"></script>

<p>Username: <input type="text" name="username" id="username"></p>

<p>Password: <input type="password" name="password" id="password"></p>

<p><input type="submit" value="check" onclick="verify()"></p>

inside the html page with manifest

</body>

</html>

**manifest.json**

{

"name": "my\_first\_manifest",

"short\_name": "first\_manifest",

"start\_url": "/first.html",

"display": "fullscreen",

"background\_color": "#ffff00",

"description": "A simple applicationfor manifestation",

"orientation": "portrait-primary",

"theme\_color": "#2196F3",

"icons": [

{

"src": "/icons/rvce\_logo.png",

"sizes": "768x768",

"type": "image/png"

},

{

"src": "/icons/rvce\_logo(1).png",

"sizes": "108x108",

"type": "image/png"

},

{

"src": "/icons/rvce\_logo(2).png",

"sizes": "204x204",

"type": "image/png"

}

],

"screenshots": [

{

"src": "/images/screenshot1.png",

"type": "image/png",

"sizes": "1366x768",

"form\_factor": "narrow"

}

],

"protocol\_handlers": [

{

"protocol": "web+jngl",

"url": "/lookup?type=%s"

},

{

"protocol": "web+jnglstore",

"url": "/shop?for=%s"

}

]

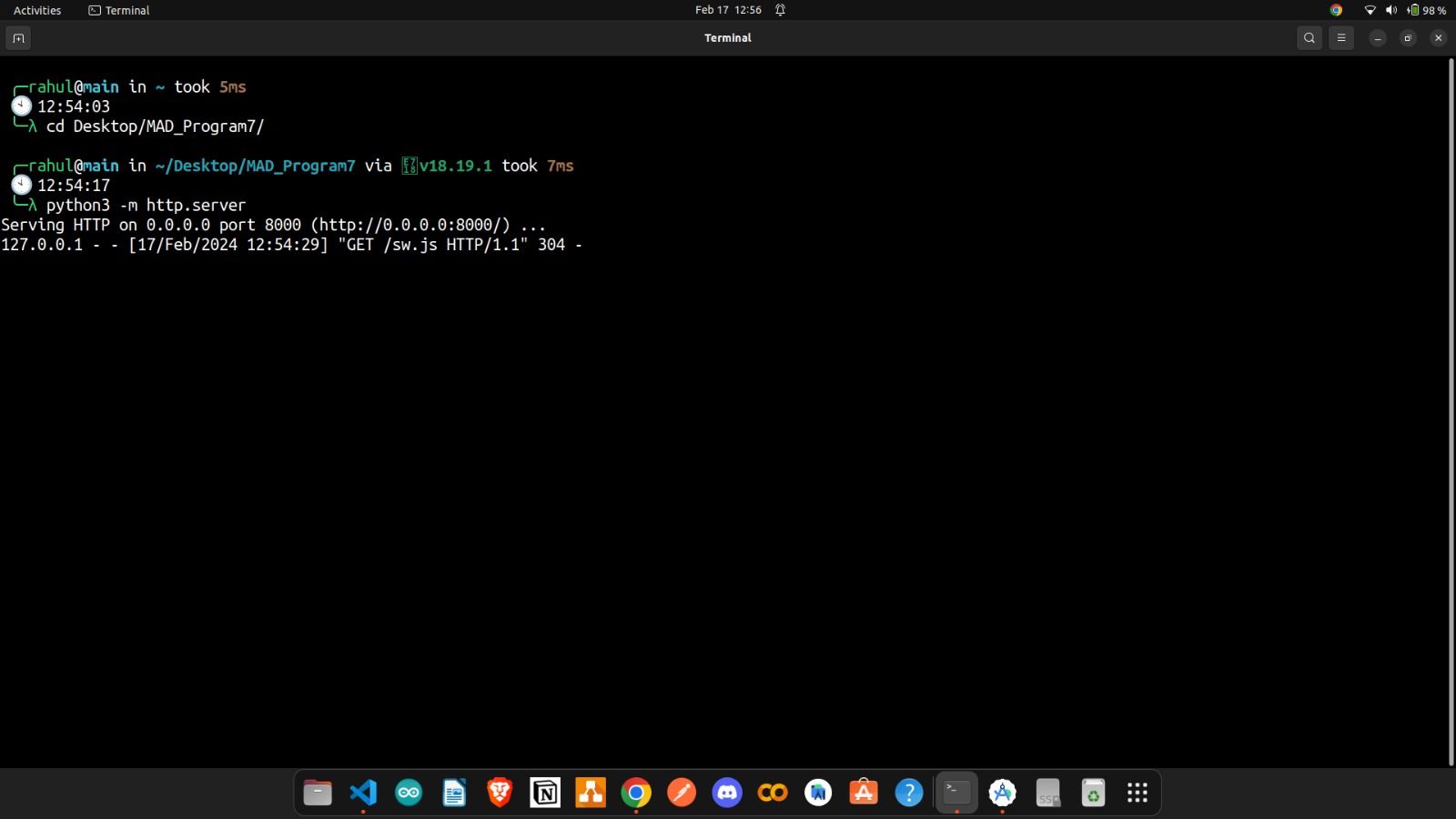
}

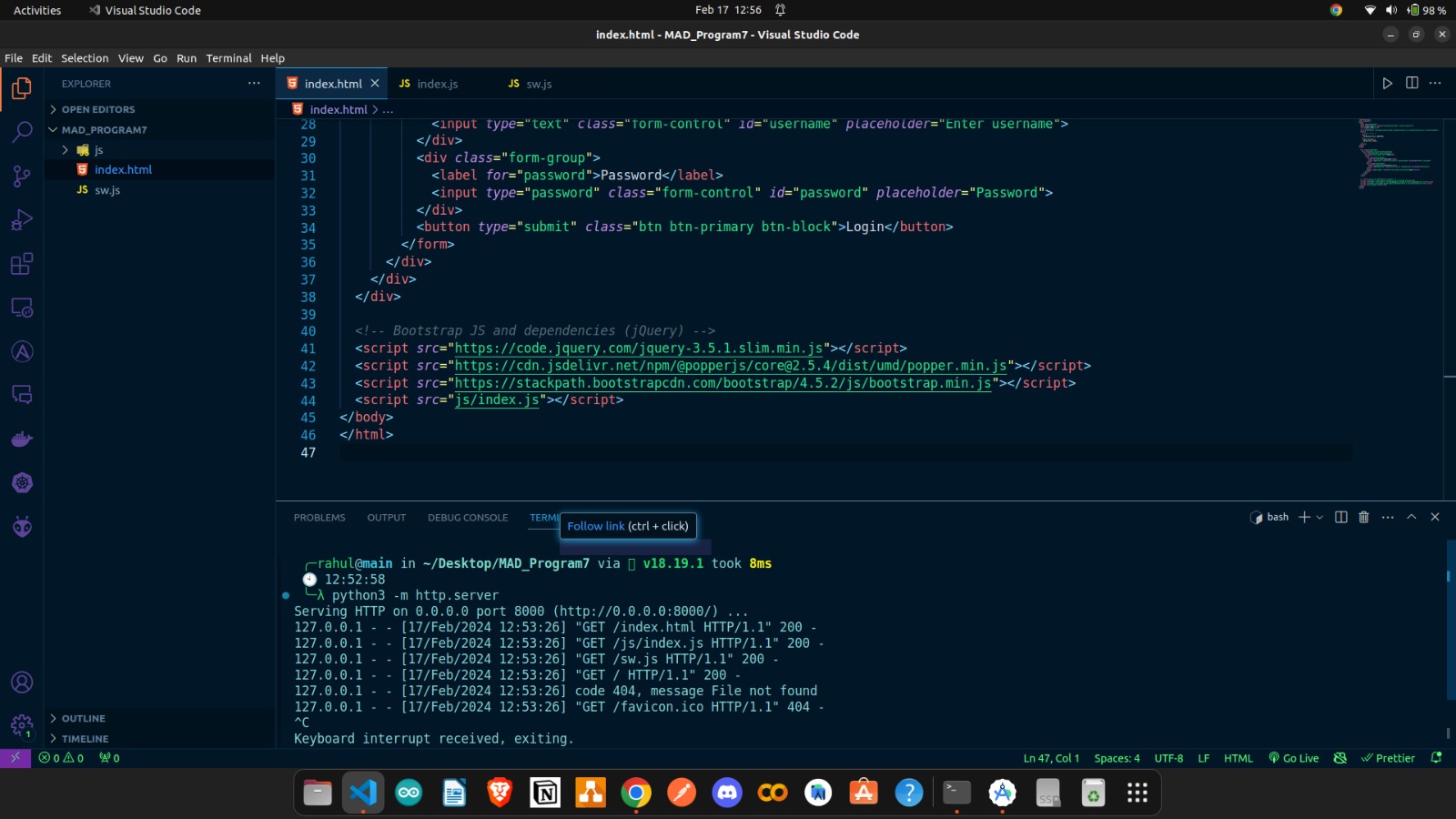
OUTPUT:

Additonal method to run server using cli

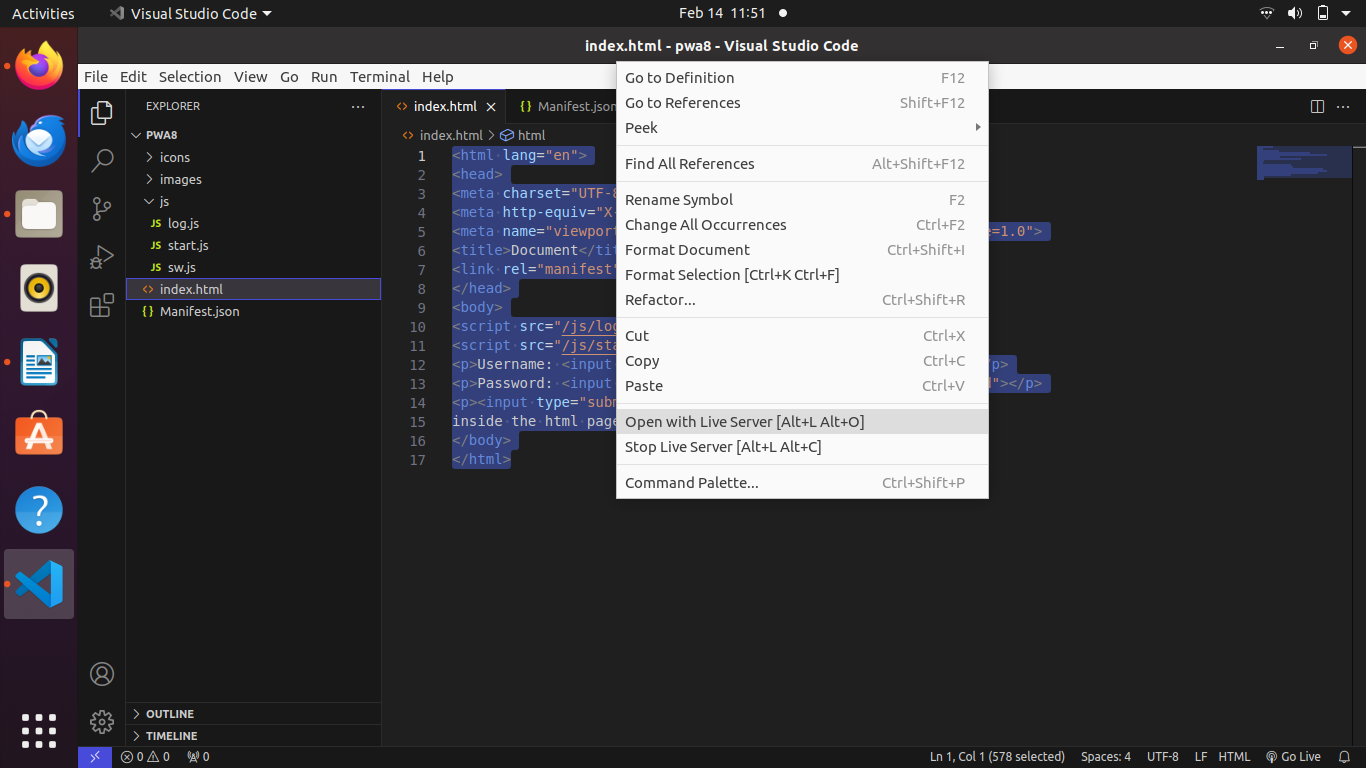
Run the program through terminal using commad  
$ python3 -m http.server <portnumber>

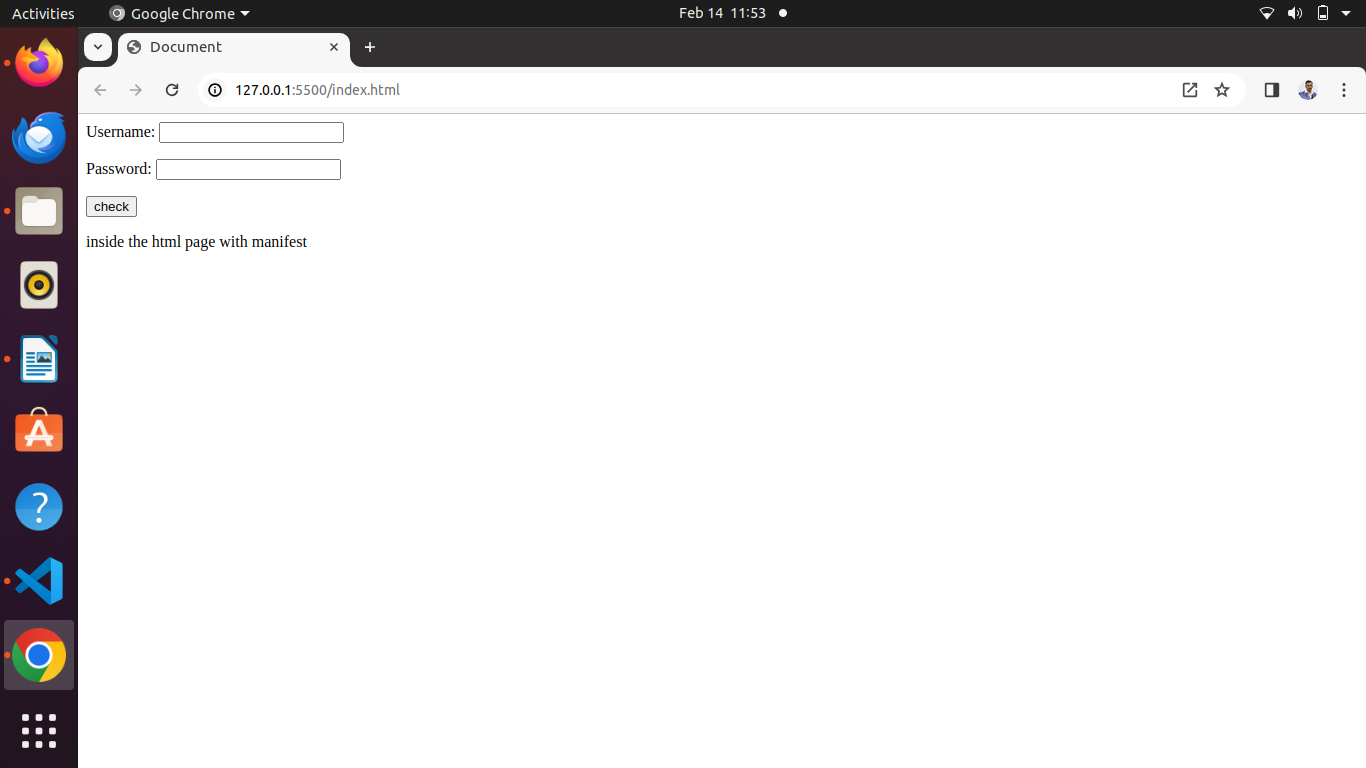
Ex : python3 -m http.server 8001



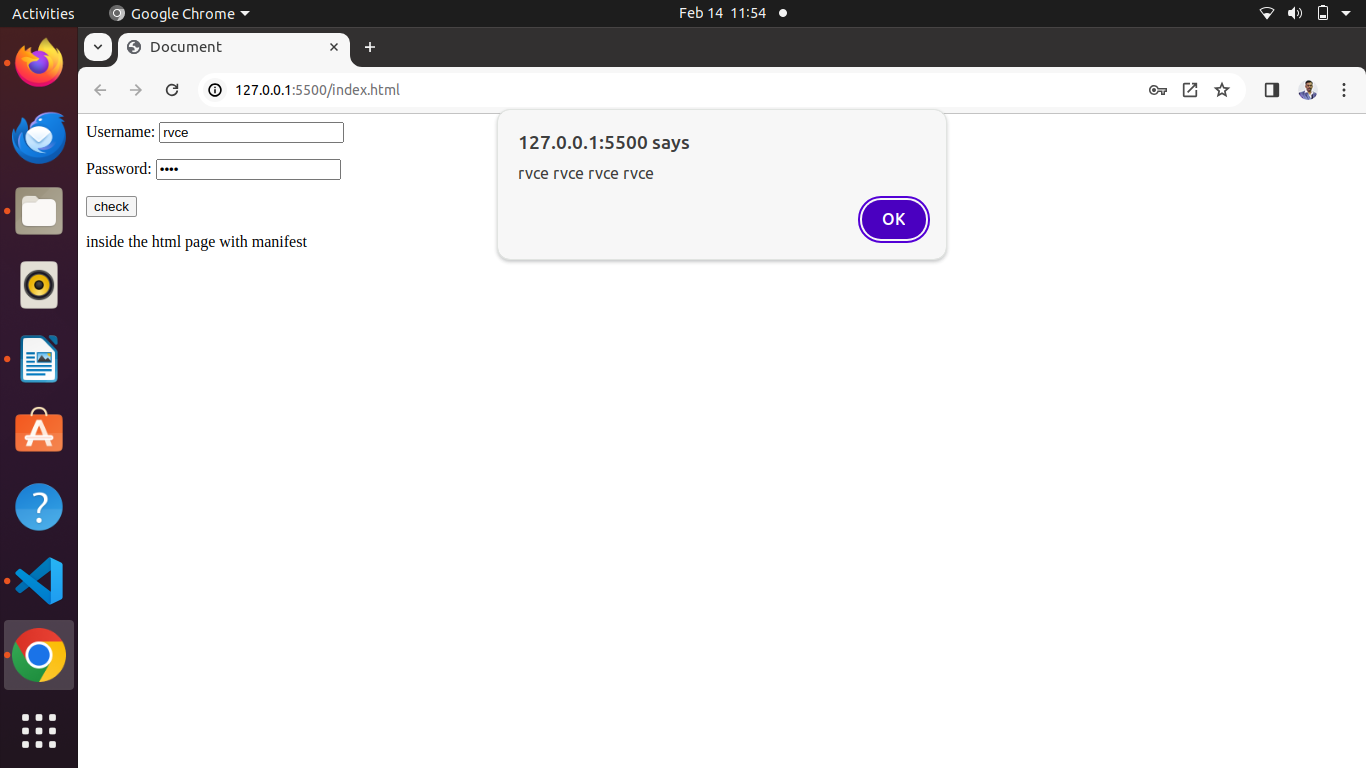


Method 2  
rightclick on HTML file and open with Live server option

crome browser



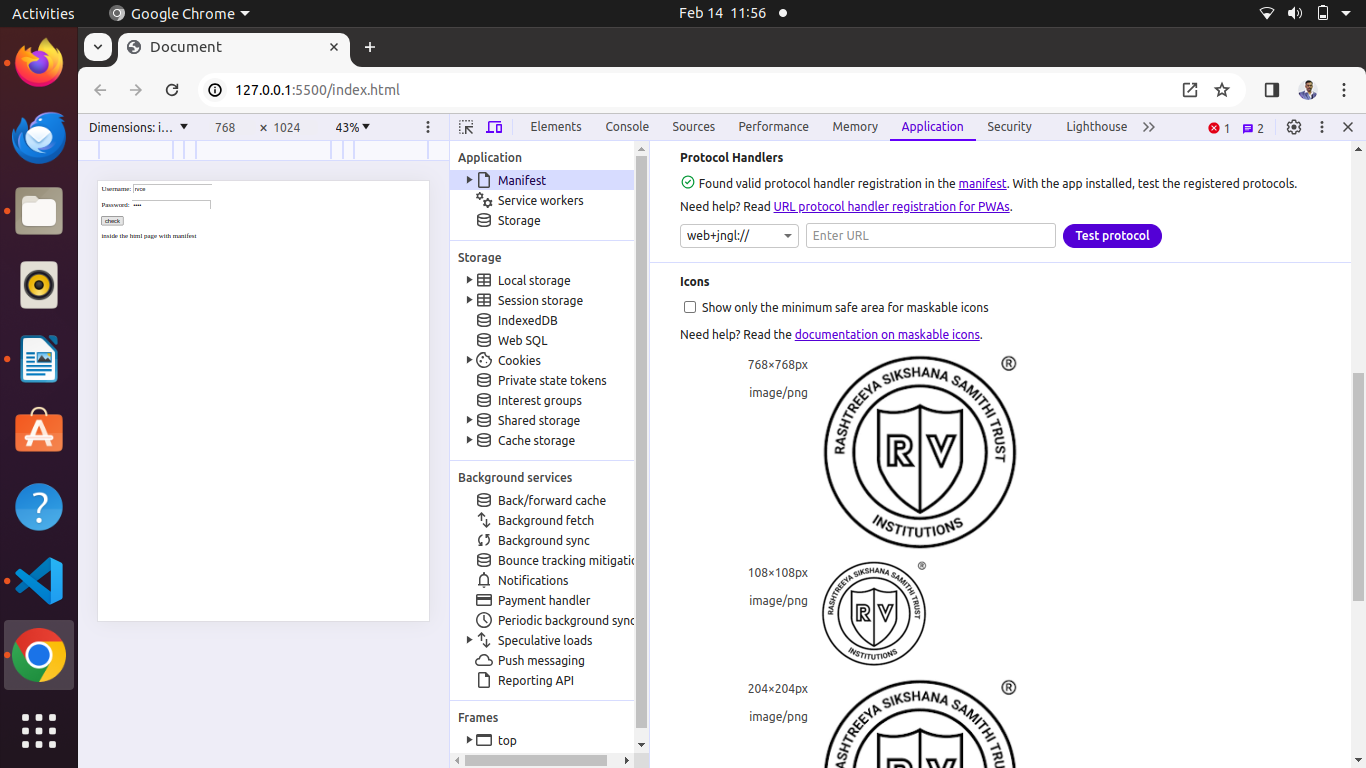
check user authentication , click on ok for authentication done.

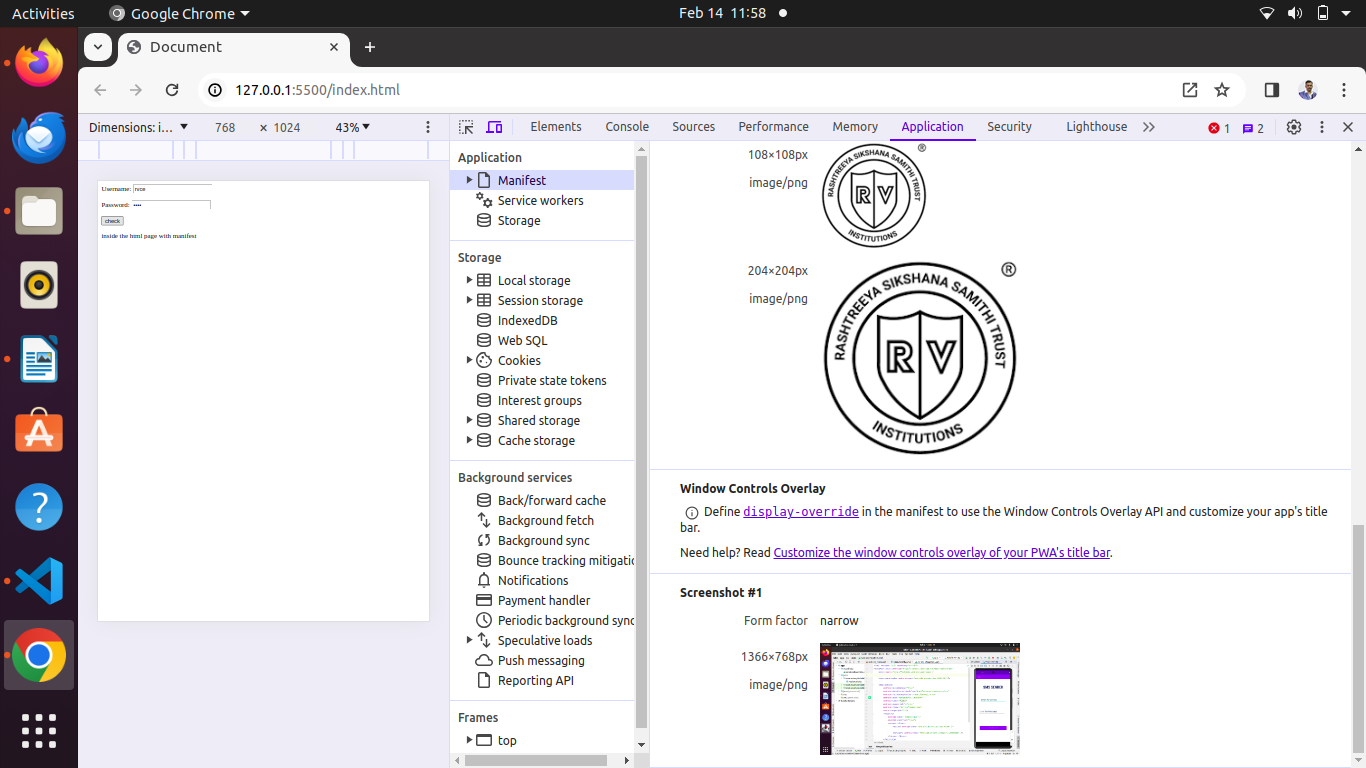


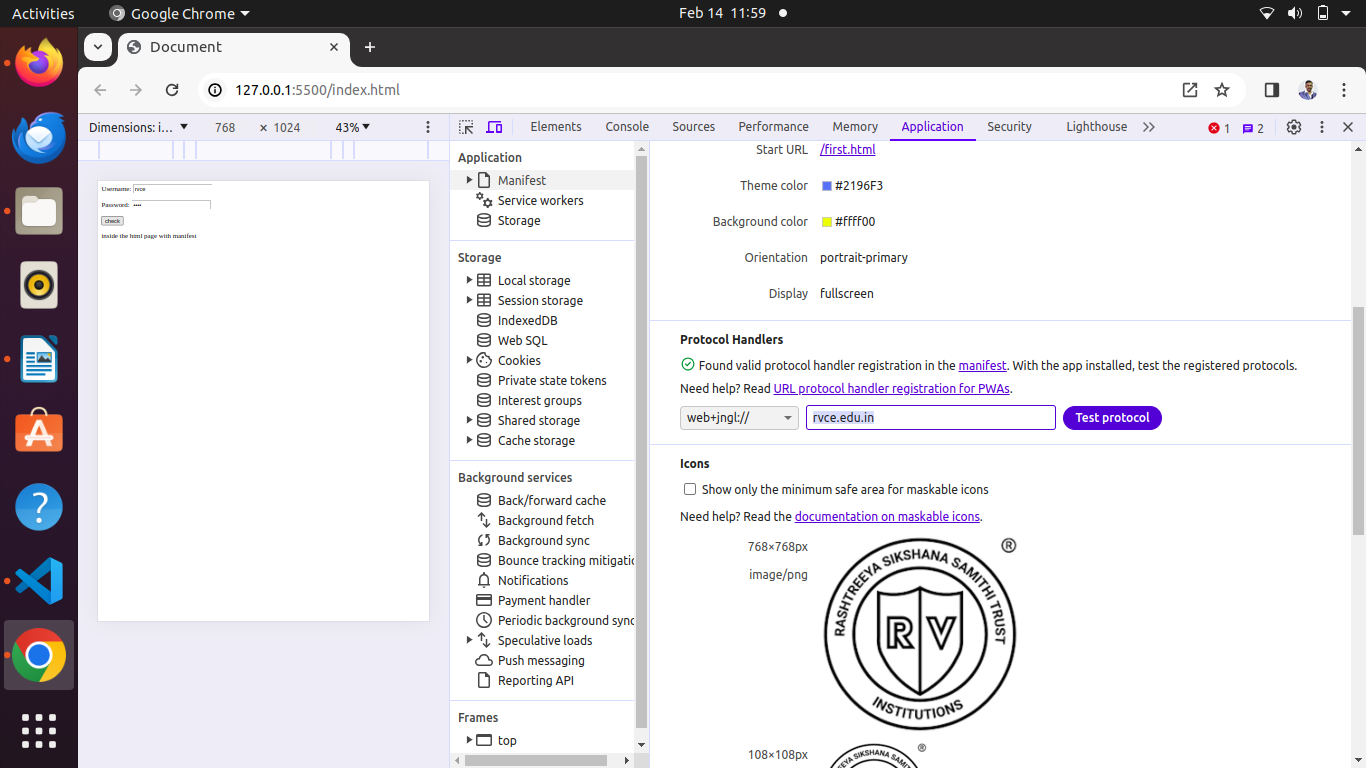
press f12 in the keyboard

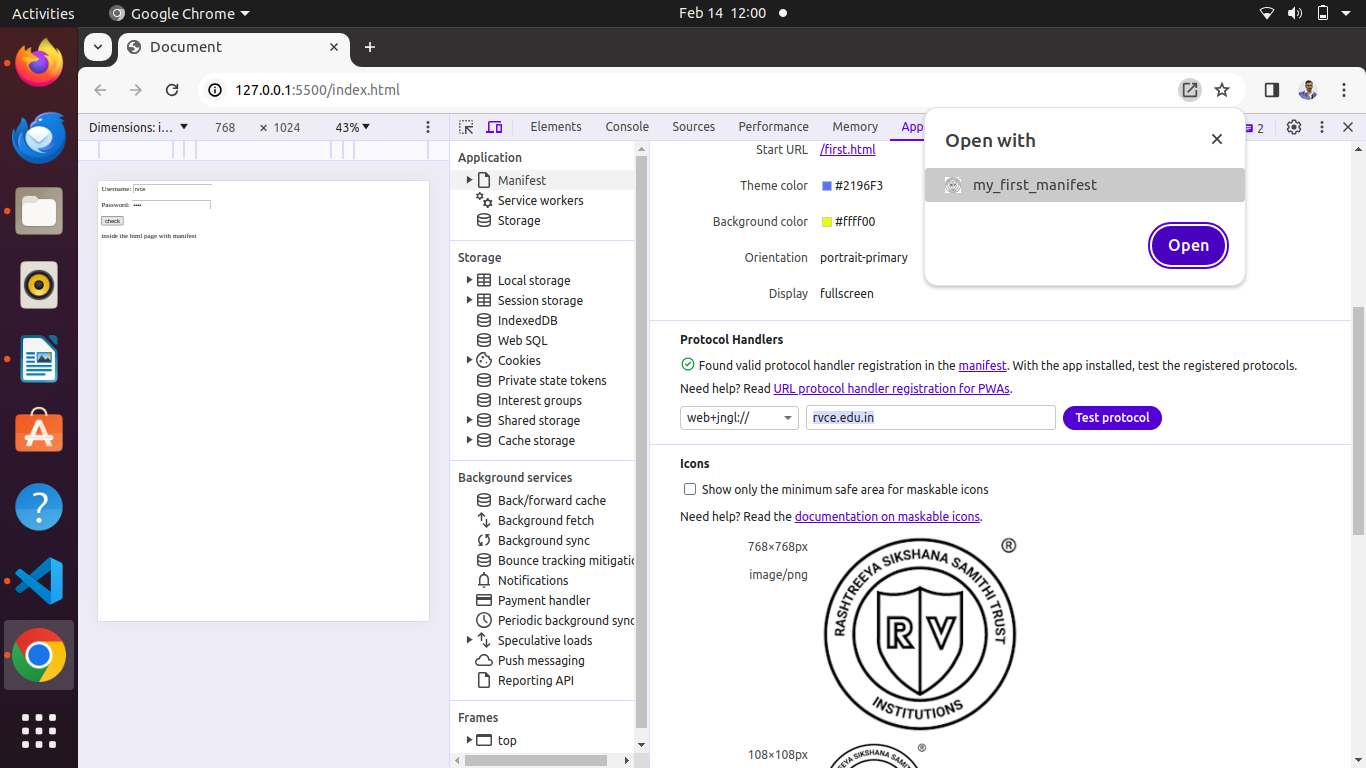
click on application and manifest.json option

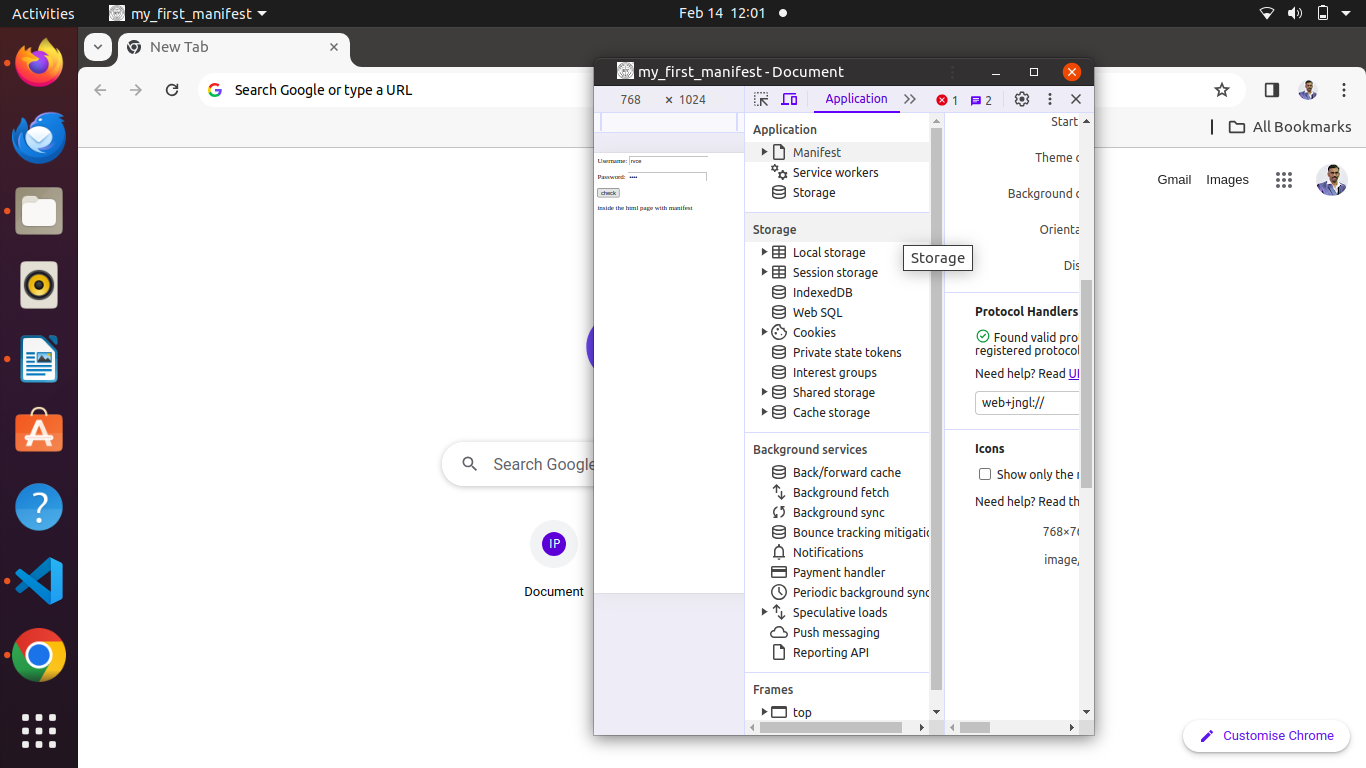
listed with all icons with various size mapped in the manifest.json file

Screenshot uploaded in the maniest.json file

Protocol handler details

PWA is working as a app by click on the option

click on open

SEE application is working like an app in the browser

Now this app can be viewed in your android studio emulator also

open emulator

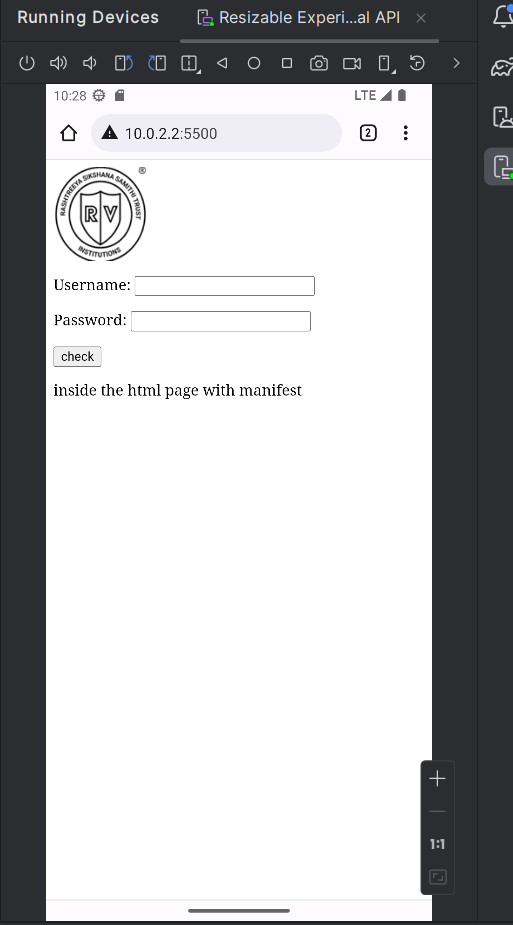
**Resizable (Experimental) API 33**

Then type the ip of browser

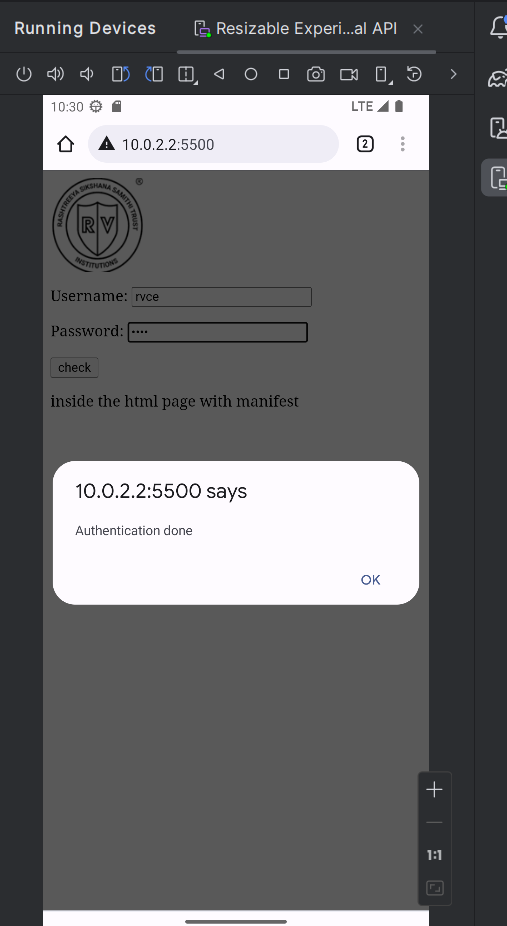
[http://10.0.2.2:5500](http://10.0.2.2:5500/) static ip of the browser index.html

the app should load in the emulator and the app can be placed in the home screen also.

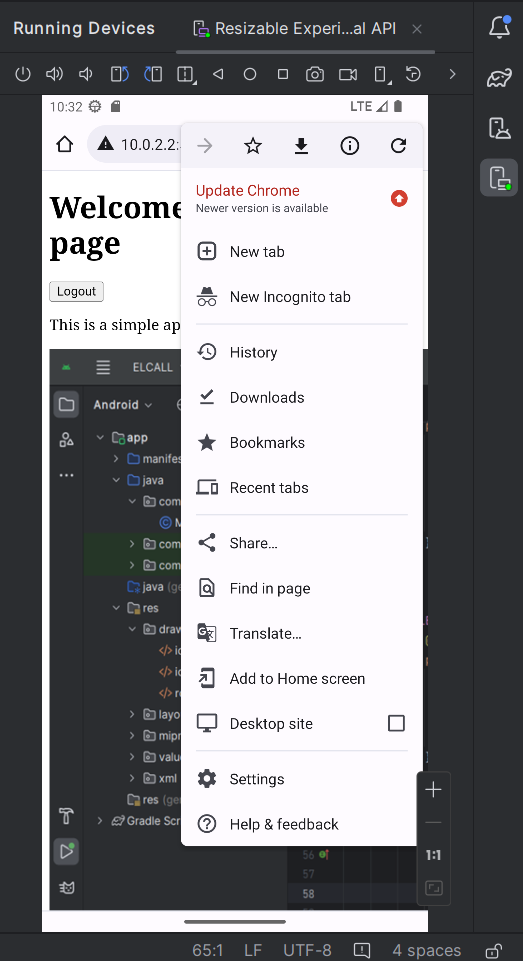
From home screen icon this web browser should open.

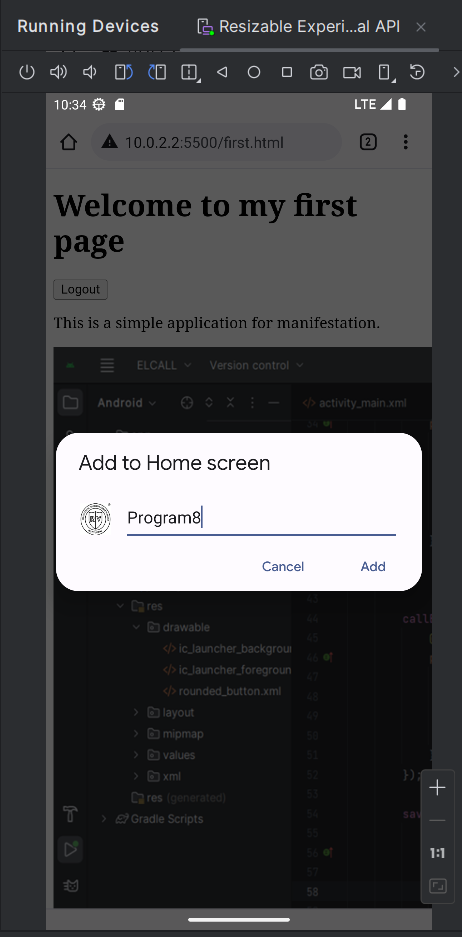


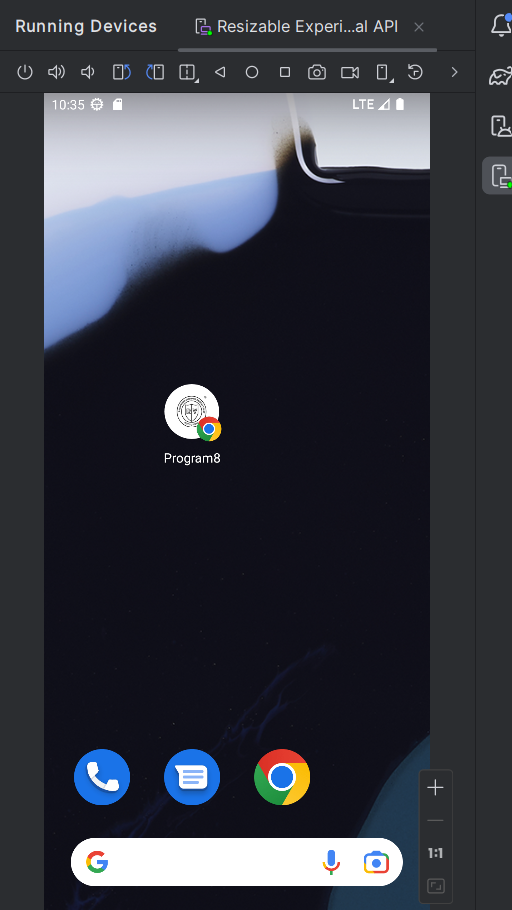
**In the emulator (Android Studio) web page is looking like this through the static ip**

****

**Once login credentials is working(Authentication done)**

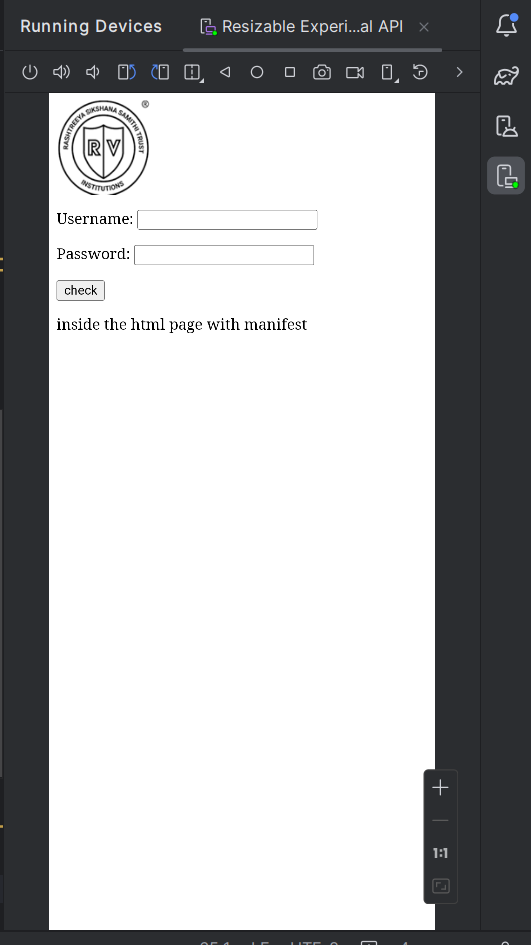
****

**Go to settings select Add to home screen option **

****

**Program 8 App is stored in the desktop**

**Through this icon/app program will run**

****