ASE Lab 3 Report

UI Mashup

Nutritionix API

Text to Speech API

*FEBRUARY 14, **2019***

Authored by:

Dharani Muli (Class ID: 18),

Chakra Pavan Kumar Kota (Class ID :13)

Introduction

Objectives

1. Build Registration and Login pages using local storage
2. Integrate with Nutritionix and text to speech API by creating own keys
3. Should build single page web application with webservices mashup

Features

Below are the features that we implemented as part of this application:

1. Registration with form validation.
2. Login with validation.
3. Use can view the Calories and Weight (in grams) information of the food item searched.
4. Our "text to speech" service will read out the information provided visible in the search screen.

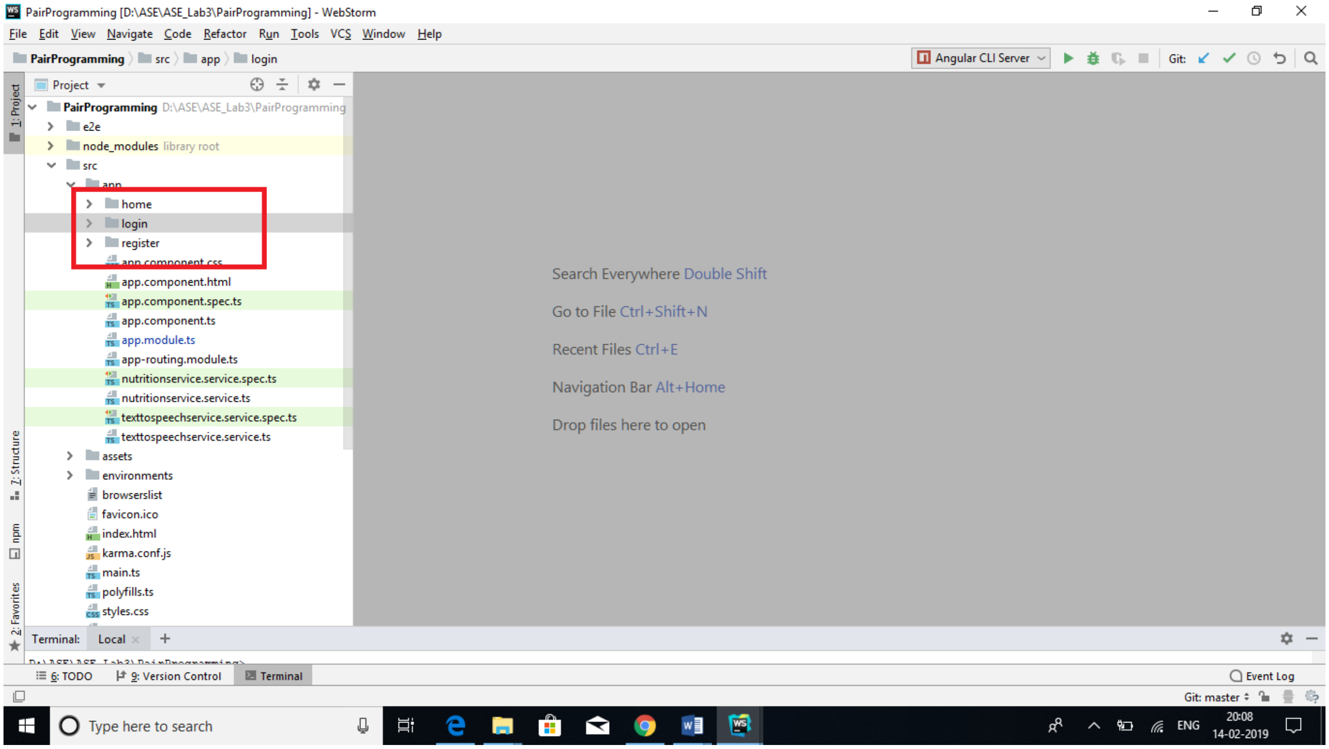
Design/Implementation

We have followed below steps to successfully complete this lab assignment:

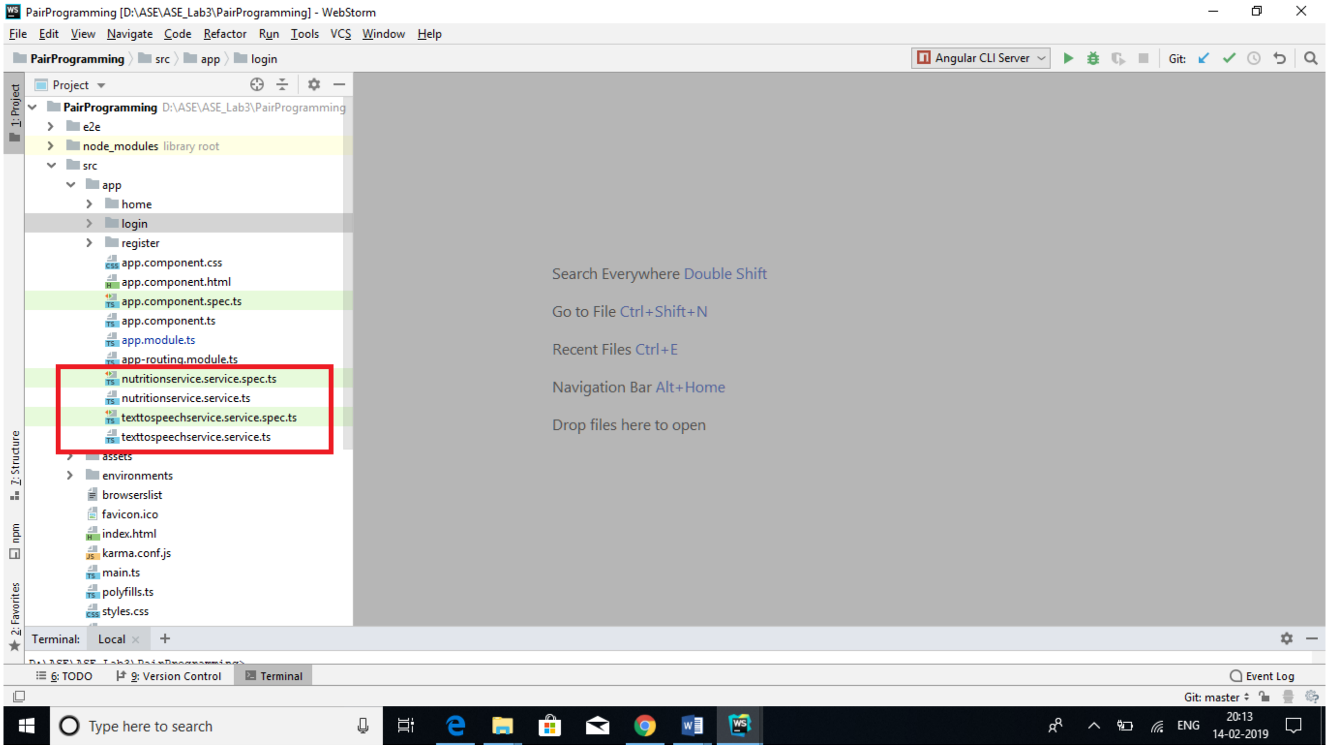
Step -1: We make sure all the pre-requisites are ready before start of the project and below are the technologies/languages used:

1. WebStorm IDE
2. npm
3. Angular 7
4. TypeScript
5. Bootstrap
6. Generated keys for Nutritionix and text to speech service integration

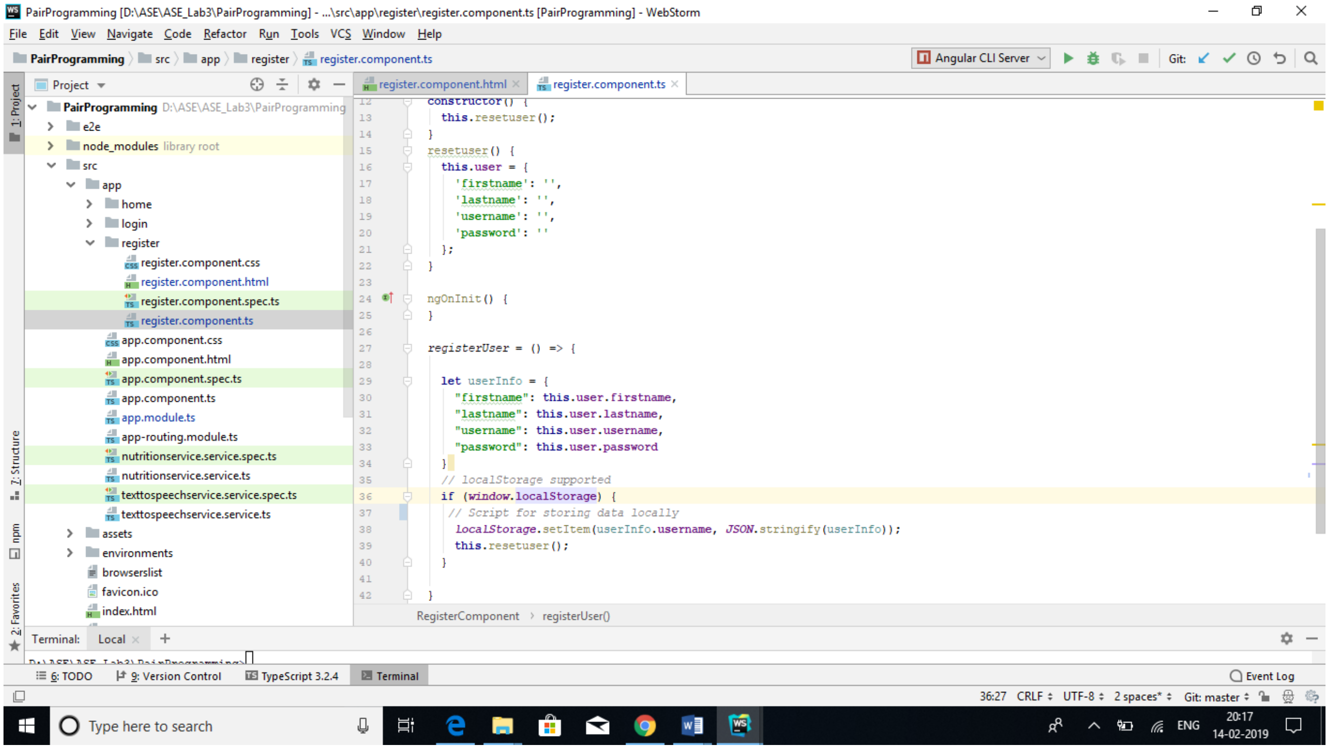
Step-2: We have created a new project and 3 components under the project i.e; Register, Login and Home components



Step-3: Along with components we have also created services i.e; Nutritionservice and texttospeechservice

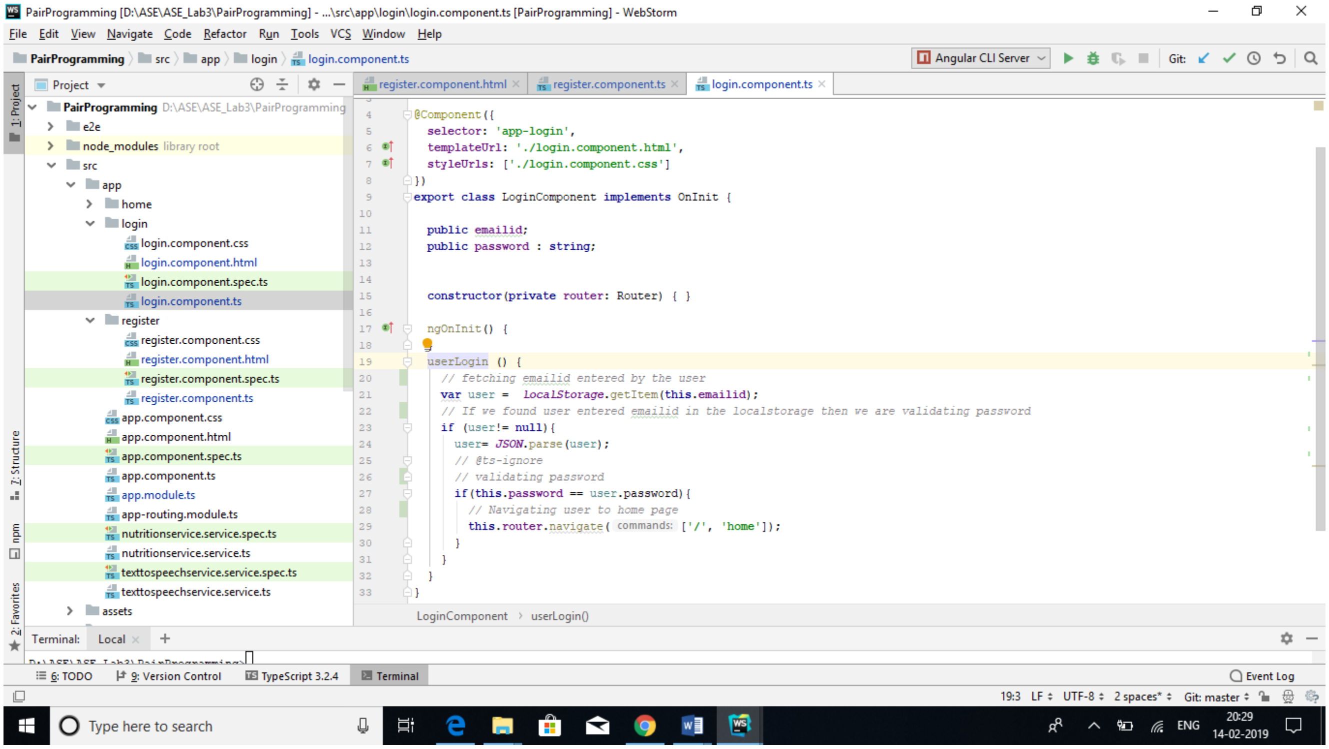


Step-4: Using HTML5 element “localStorage” we are fetching data of the form and storing locally


PairProgramming [D:\ASE\ASE_Lab3\PairProgramming] - ...\src\app\register\register.component.ts [PairProgramming] - WebStorm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
PairProgramming › src › app › register › register.component.ts
Angular CLI Server
Git:
Project
register.component.html
register.component.ts
PairProgramming D:\ASE\ASE_Lab3\PairProgramming
e2e
node_modules library root
src
app
home
login
register
register.component.css
register.component.html
register.component.spec.ts
register.component.ts
app.component.css
app.component.html
app.component.spec.ts
app.component.ts
app.module.ts
app-routing.module.ts
nutritionservice.service.spec.ts
nutritionservice.service.ts
texttospeechservice.service.spec.ts
texttospeechservice.service.ts
assets
environments
browserslist
favicon.ico
index.html
RegisterComponent › registerUser()
Terminal: Local +
6: TODO
9: Version Control
TypeScript 3.2.4
Terminal
Event Log
36:27 CRLF UTF-8 2 spaces* Git: master
Type here to search
ENG
20:17
14-02-2019

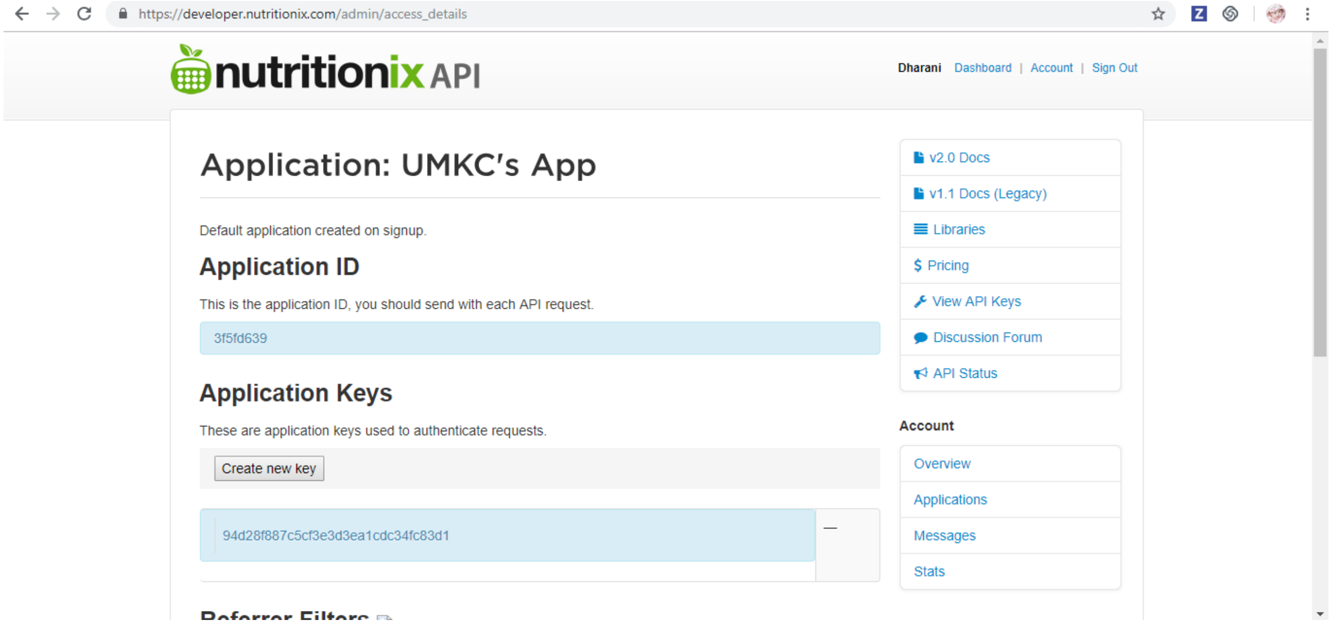
```
constructor() {
  this.resetuser();
}
resetuser() {
  this.user = {
    'firstname': '',
    'lastname': '',
    'username': '',
    'password': ''
  };
}

ngOnInit() {
}

registerUser = () => {

  let userInfo = {
    "firstname": this.user.firstname,
    "lastname": this.user.lastname,
    "username": this.user.username,
    "password": this.user.password
  }
  // localStorage supported
  if (window.localStorage) {
   // Script for storing data locally
    localStorage.setItem(userInfo.username, JSON.stringify(userInfo));
    this.resetuser();
  }

}
```

Step-5: Once user try to login we are fetching details from localStorage validating and allowing user to login if the credentials are correct


PairProgramming [D:\ASE\ASE_Lab3\PairProgramming] - ...\src\app\login\login.component.ts [PairProgramming] - WebStorm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
PairProgramming › src › app › login › login.component.ts
Angular CLI Server
Git:
Project
register.component.html
register.component.ts
login.component.ts
PairProgramming D:\ASE\ASE_Lab3\PairProgramming
e2e
node_modules library root
src
app
home
login
login.component.css
login.component.html
login.component.spec.ts
login.component.ts
register
register.component.css
register.component.html
register.component.spec.ts
register.component.ts
app.component.css
app.component.html
app.component.spec.ts
app.component.ts
app.module.ts
app-routing.module.ts
nutritionservice.service.spec.ts
nutritionservice.service.ts
texttospeechservice.service.spec.ts
texttospeechservice.service.ts
assets
LoginComponent › userLogin()
Terminal: Local +
6: TODO
9: Version Control
TypeScript 3.2.4
Terminal
Event Log
19:3 LF UTF-8 2 spaces* Git: master
Type here to search
ENG
20:29
14-02-2019

```
@Component({
  selector: 'app-login',
  templateUrl: './login.component.html',
  styleUrls: ['./login.component.css']
})
export class LoginComponent implements OnInit {

  public emailid;
  public password : string;


  constructor(private router: Router) { }

  ngOnInit() {
  }
  userLogin () {
    // fetching emailid entered by the user
    var user =  localStorage.getItem(this.emailid);
    // If we found user entered emailid in the localstorage then we are validating password
    if (user!= null){
      user= JSON.parse(user);
      // @ts-ignore
      // validating password
      if(this.password == user.password){
        // Navigating user to home page
        this.router.navigate( commands: ['/', 'home']);
      }
    }
  }
}
```

Step-6: Later we integrated with Nutritionix service and fetched the data required. In order to use this service we should register to Nutritionix development version and get the “Appid” and “AppKey”



https://developer.nutritionix.com/admin/access_details

Dharani | Dashboard | Account | Sign Out

Application: UMKC's App

Default application created on signup.

Application ID

This is the application ID, you should send with each API request.

3f5fd639

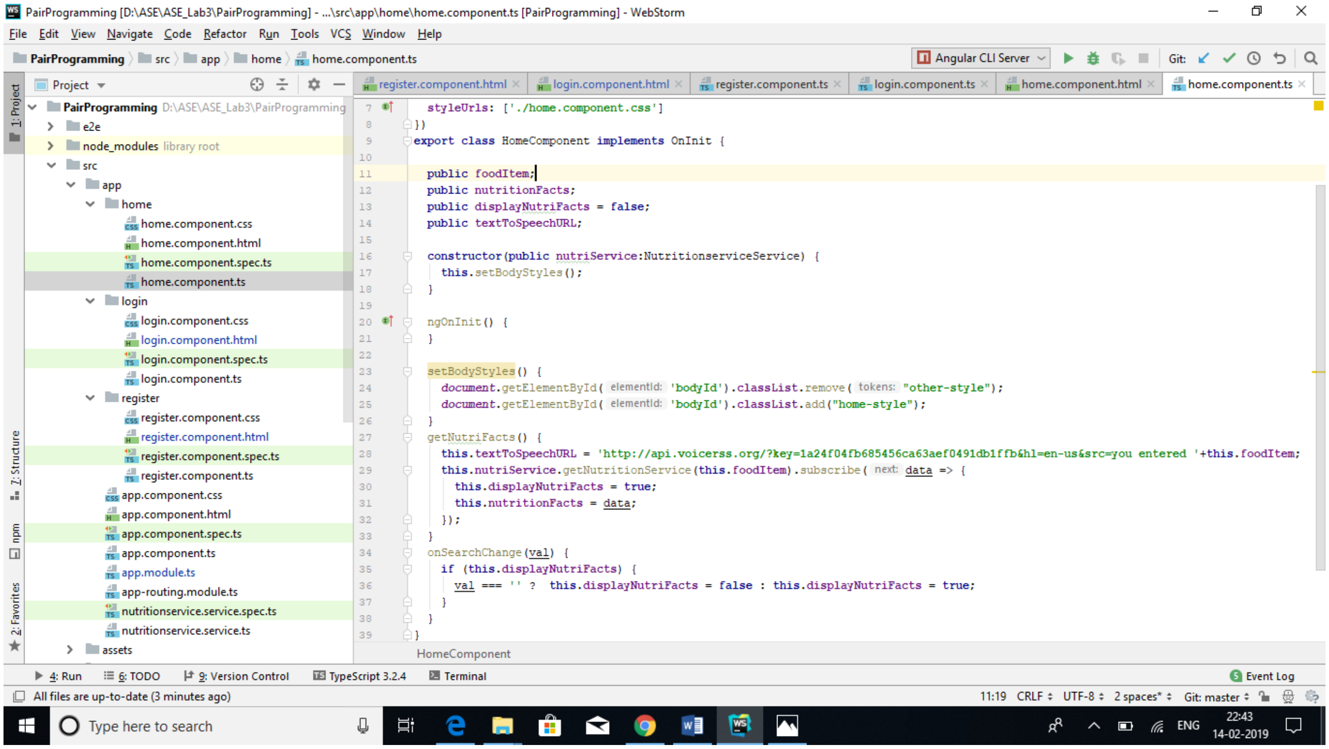
Application Keys

These are application keys used to authenticate requests.

Create new key

94d28f887c5cf3e3d3ea1cdc34fc83d1

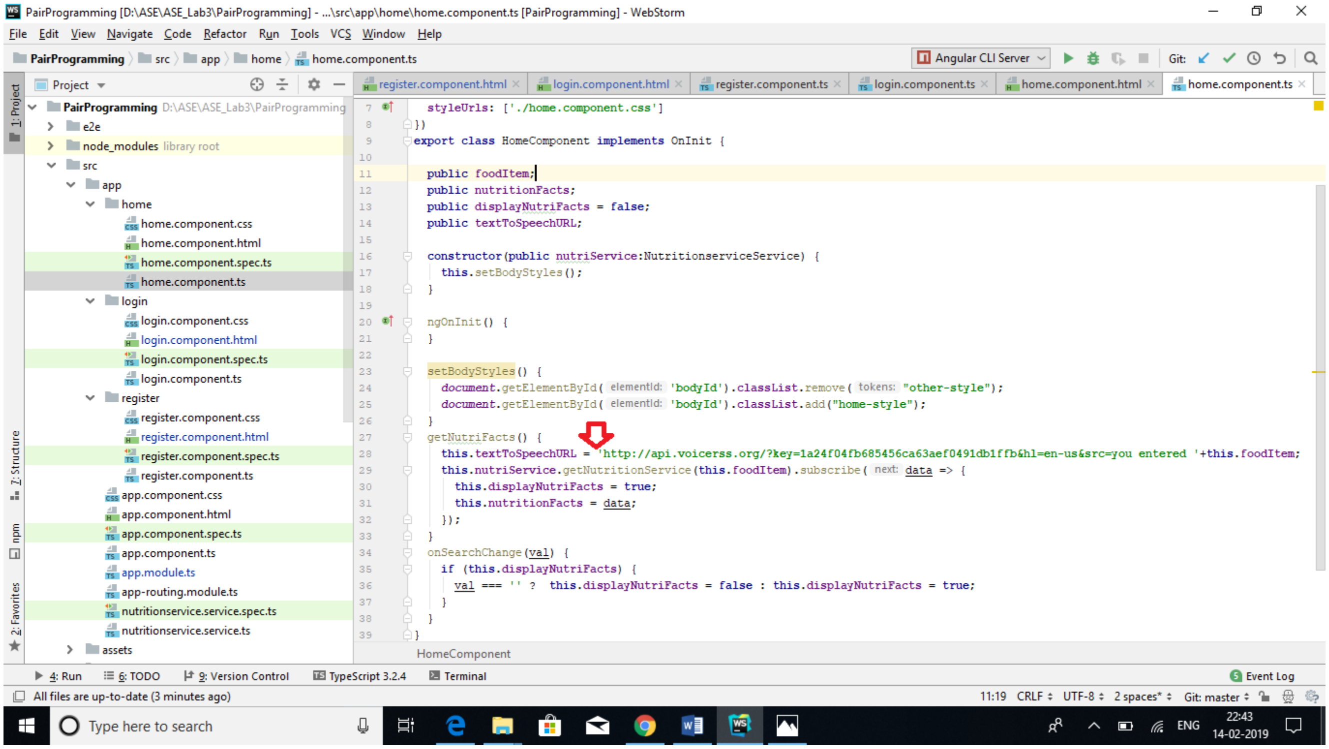
Referrer Filters



PairProgramming [D:\ASE\ASE_Lab3\PairProgramming] - ...\src\app\home\home.component.ts [PairProgramming] - WebStorm

```
  styleUrls: ['./home.component.css']
})
export class HomeComponent implements OnInit {

  public foodItem;
  public nutritionFacts;
  public displayNutriFacts = false;
  public textToSpeechURL;

  constructor(public nutriService:NutritionserviceService) {
    this.setBodyStyles();
  }

  ngOnInit() {
  }

  setBodyStyles() {
    document.getElementById('bodyId').classList.remove("other-style");
    document.getElementById('bodyId').classList.add("home-style");
  }
  getNutriFacts() {
    this.textToSpeechURL = 'http://api.voicerss.org/?key=1a24f04fb685456ca63aef0491db1ffb&hl=en-us&src=you entered '+this.foodItem;
    this.nutriService.getNutritionService(this.foodItem).subscribe(data => {
      this.displayNutriFacts = true;
      this.nutritionFacts = data;
    });
  }
  onSearchChange(val) {
    if (this.displayNutriFacts) {
      val === '' ?  this.displayNutriFacts = false : this.displayNutriFacts = true;
    }
  }
}
```

Step-7: We have done the same with text to speech service and made our application speak according to what user entered on the home screen.

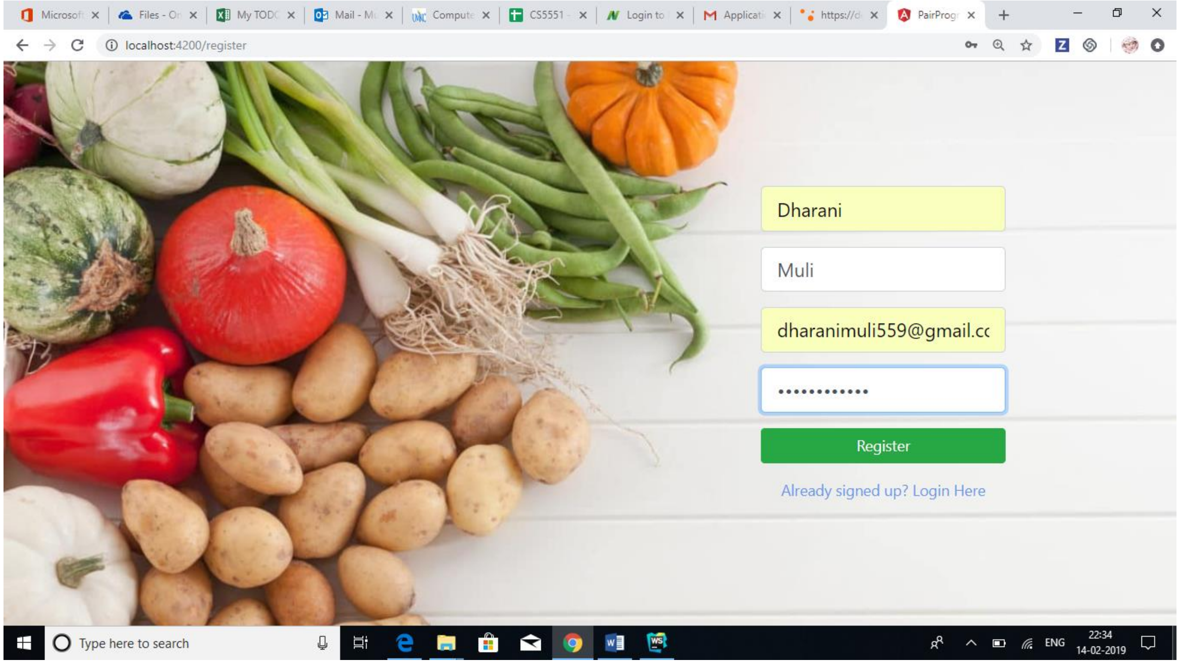


```
  styleUrls: ['./home.component.css']
})
export class HomeComponent implements OnInit {

  public foodItem;
  public nutritionFacts;
  public displayNutriFacts = false;
  public textToSpeechURL;

  constructor(public nutriService:NutritionserviceService) {
    this.setBodyStyles();
  }

  ngOnInit() {
  }

  setBodyStyles() {
    document.getElementById( elementId: 'bodyId').classList.remove( tokens: "other-style");
    document.getElementById( elementId: 'bodyId').classList.add("home-style");
  }
  getNutriFacts() {
    this.textToSpeechURL = 'http://api.voicerss.org/?key=1a24f04fb685456ca63aef0491db1ffb&hl=en-us&src=you entered '+this.foodItem;
    this.nutriService.getNutritionService(this.foodItem).subscribe( next: data => {
      this.displayNutriFacts = true;
      this.nutritionFacts = data;
    });
  }
  onSearchChange(val) {
    if (this.displayNutriFacts) {
      val === '' ?  this.displayNutriFacts = false : this.displayNutriFacts = true;
    }
  }
}
```

Input/Output:

Registration Page:

Input: User registers by entering required details



localhost:4200/register

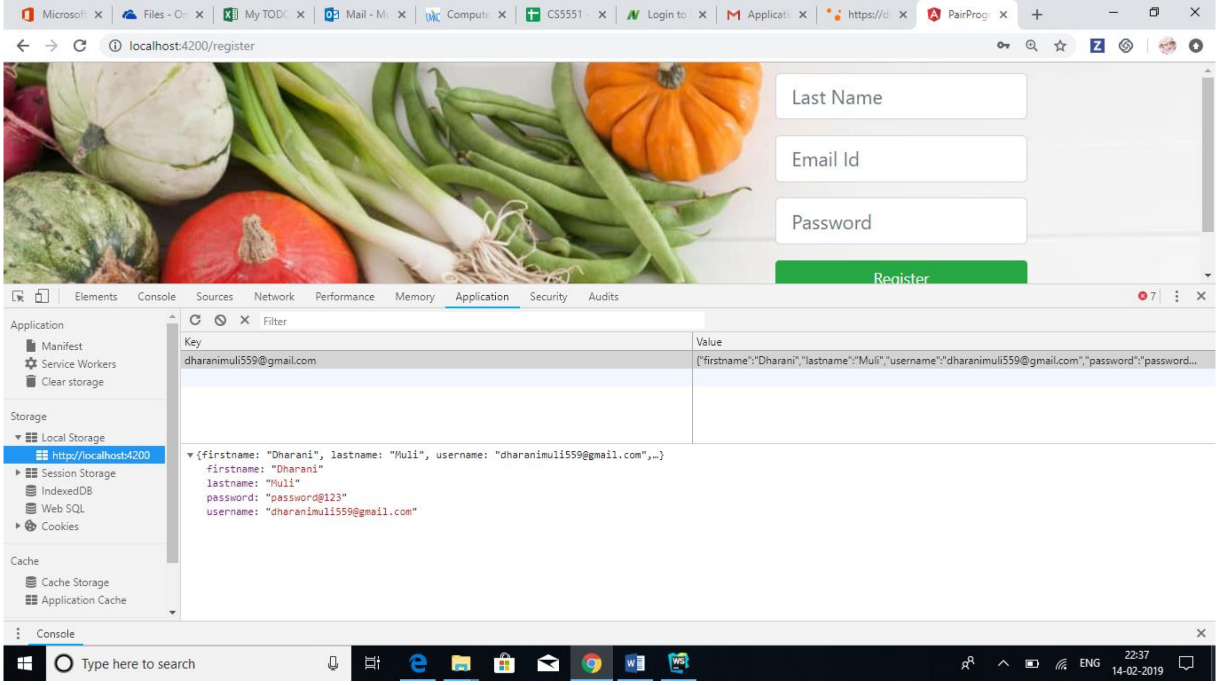
Dharani

Muli

dharanimuli559@gmail.cc

•••••••••••••

Output: We stored all the details locally



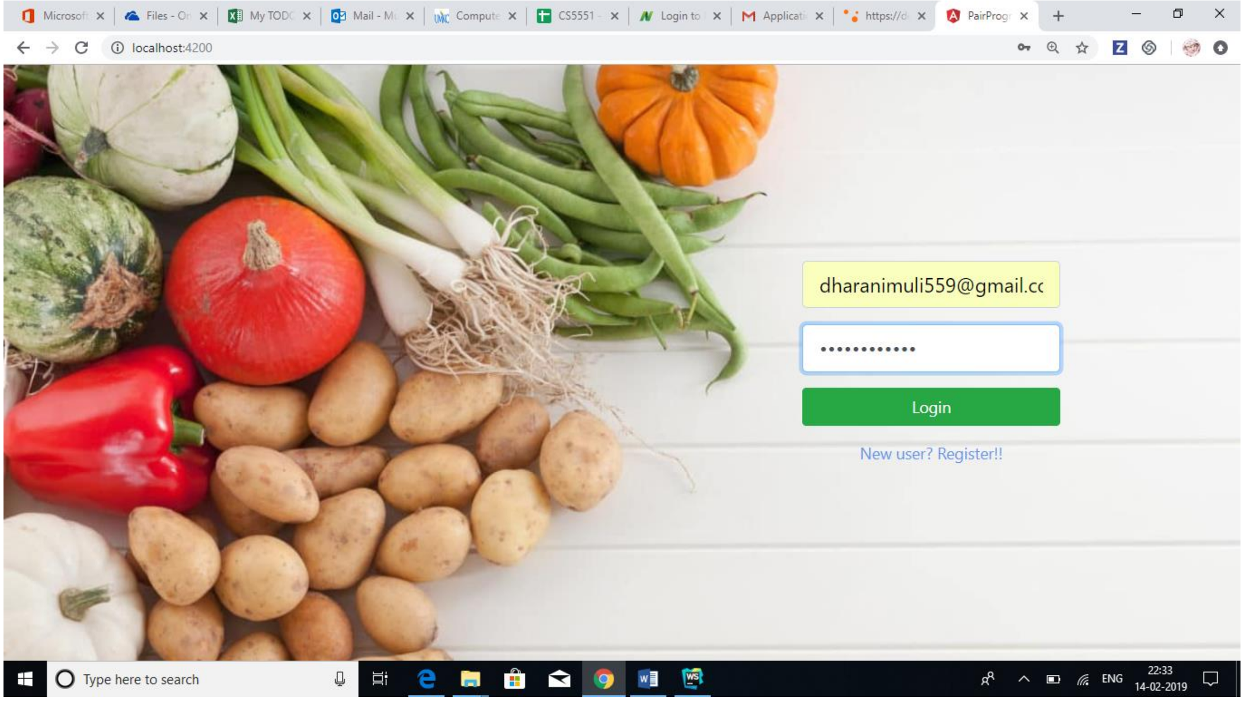
localhost:4200/register

Key	Value
dharanimuli559@gmail.com	{"firstname":"Dharani","lastname":"Muli","username":"dharanimuli559@gmail.com","password":"password...

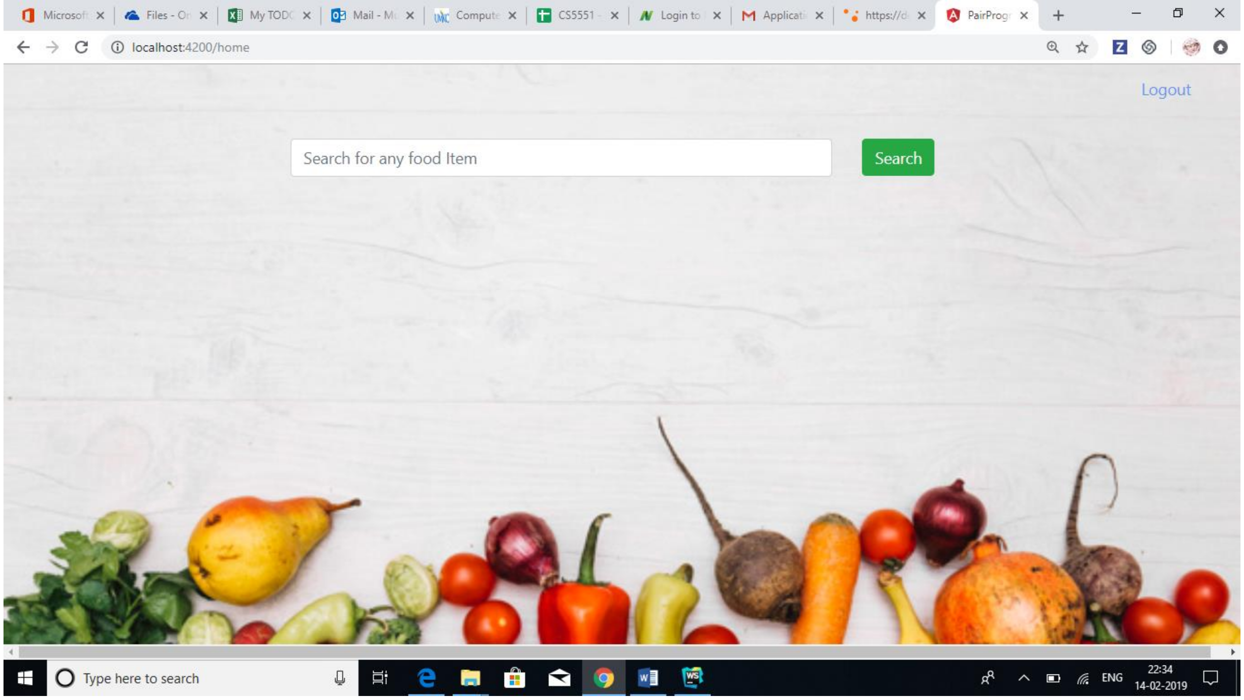
```
▼{firstname: "Dharani", lastname: "Muli", username: "dharanimuli559@gmail.com",…}
   firstname: "Dharani"
   lastname: "Muli"
   password: "password@123"
   username: "dharanimuli559@gmail.com"
```

Login page

Input: User entered login credentials.

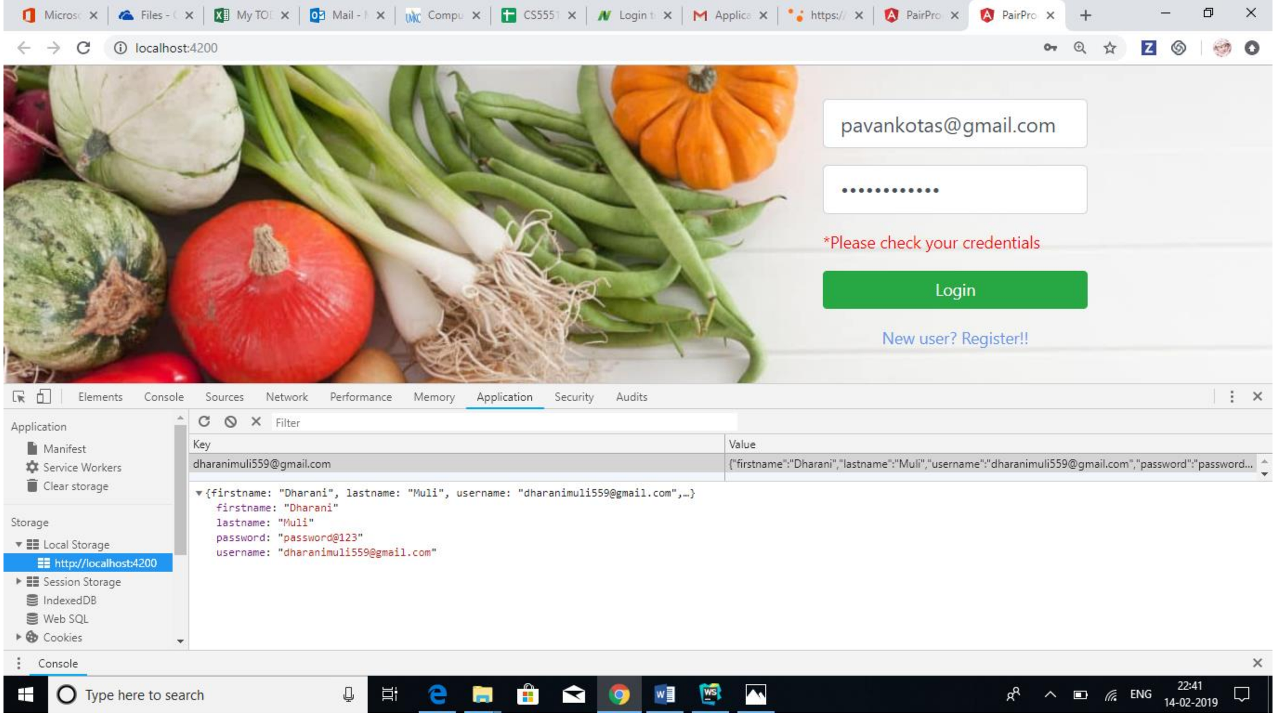


Output: If she is a valid user then we are navigating to Home page



Input: If non-registered used tries to login then we are throwing an error

Output: Throws an error



pavankotas@gmail.com

•••••••••••••

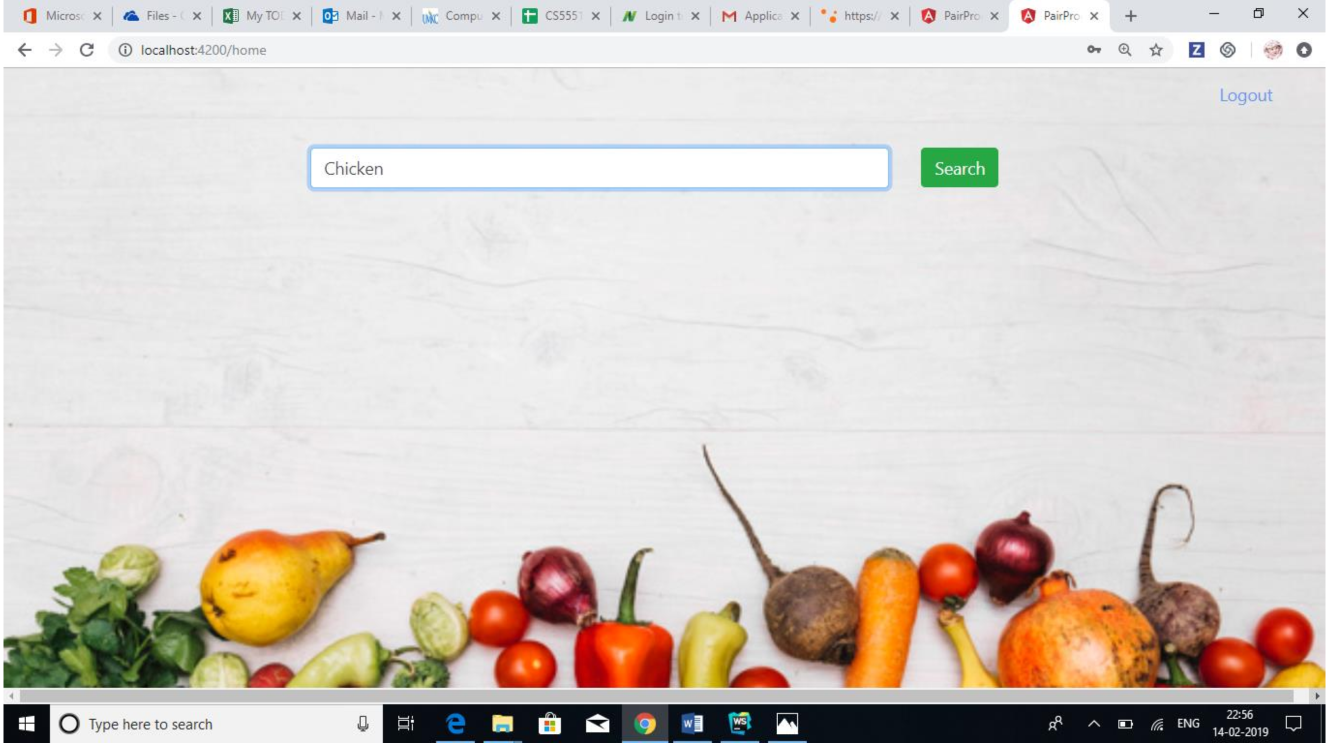
*Please check your credentials

Key	Value
dharanimuli559@gmail.com	{"firstname":"Dharani","lastname":"Muli","username":"dharanimuli559@gmail.com","password":"password...

```
▼{firstname: "Dharani", lastname: "Muli", username: "dharanimuli559@gmail.com",…}
   firstname: "Dharani"
   lastname: "Muli"
   password: "password@123"
   username: "dharanimuli559@gmail.com"
```

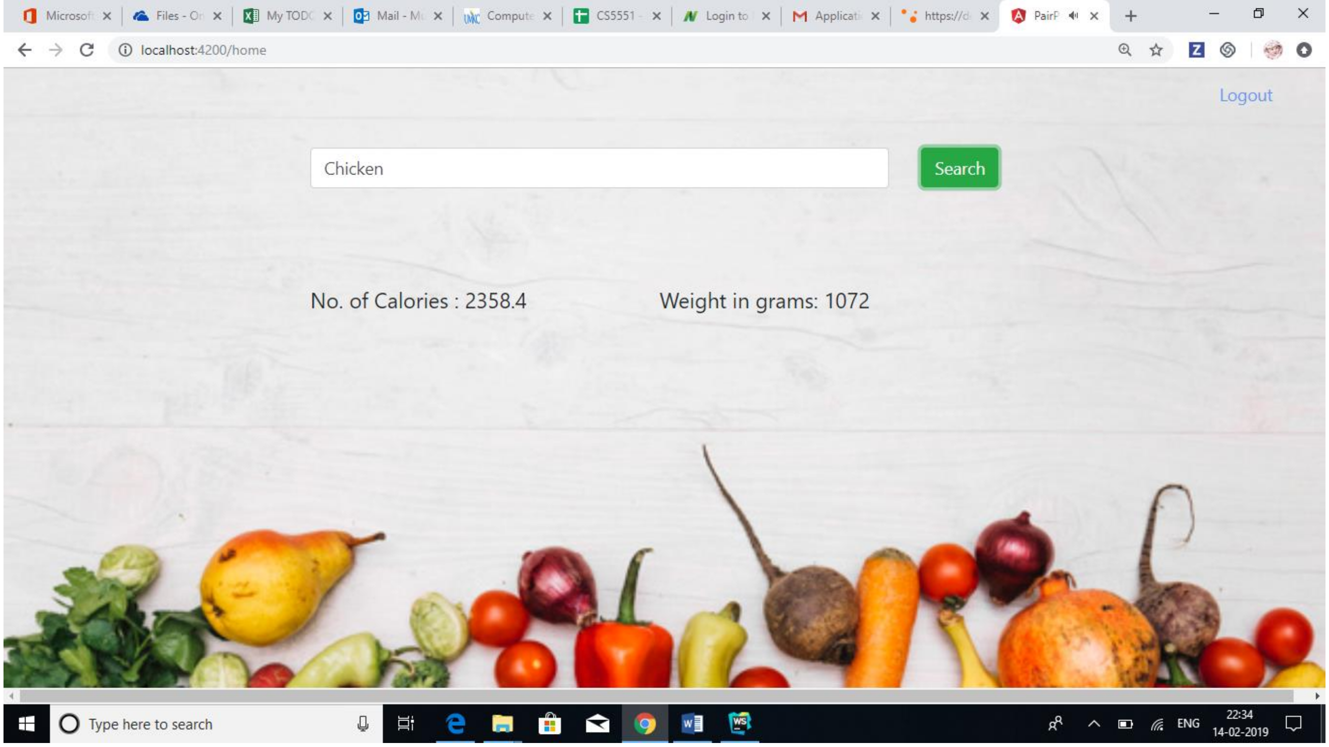
Home Page

Input: User searches for food item



Chicken

Output: Displaying Calories and Weight information of that particular food item. Moreover, text to speech api will speakout the searched item.



No. of Calories : 2358.4

Weight in grams: 1072

Issues/Limitations:

IBM doesn’t provide API keys so we chose different API.

Conclusion:

This Lab assignment helped us getting hands on experience on Angular 7 and localStorage concepts. Apart from that we learned how to integrate with services on the UI side doing UI mashup.