Date	15 November 2022
Team ID	PNT2022TMID33084
Project name	Nutrition assistant application

index.html:

```
<!DOCTYPE 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">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
crossorigin="anonymous"></script>

  <title>Nutrition Assistant</title>
  <link rel="stylesheet" href="style.css">
  <style>

  </style>
</head>
<body>



  <div class="container-fluid">
   <div class="row">
    <div class="col-lg-5 colh heading">
     <p class="display-3">Nutrition Assistant</p>
    </div>
    <div class="col-lg-7 colh">

     <div>
      <ul class="lists">
       <li> <a id="c1" href="#cont_1">Home</a></li>
       <li onmouseover = "signinOne(this)" onmouseout="unsignin(this)"> <a id="c2"
href="#logins">SignIn</a></li>
       <li onmouseover = "signinTwo(this)" onmouseout="unsignin(this)"> <a id="c3"
href="#logins">SignUp</a></li>
       <li onmouseover = "visible(this)" onmouseout="unsignin(this)"> <a id="c4"
href="#abouts">About</a></li>
      </ul>
     </div>
    </div>
   </div>
  </div>

  <div class="container-fluid content">
   <div class="row">
      <img src="5912.jpg">
    </div>
    <div class ="image">

    </div>
    <div class="col-lg info" id="infos">
     <div class="cont_1 pb-0">

     <div id="logins" class="login">
      <div class="container" id="container">
       <div class="form-container sign-up-container">
        <form action="/register" method="post" enctype="multipart/form-data">
```

```
 <h1>Create Account</h1>

 <input type="text" name="name" placeholder="Name" required />
 <input type="email" name="email" placeholder="Email" required />
 <input type="password" name="pass" placeholder="Password" required />
 <button>Sign Up</button>
 </form>
</div>
<div class="form-container sign-in-container">
 <form action="/home" method="post" enctype="multipart/form-data">
 <h1>Sign in</h1>
  <input STYLE="display:none" type="text" name="name" placeholder="Name"/>
  <input type="email" name="email" placeholder="Email" required />
  <input type="password" name="pass" placeholder="Password" required />
  <a href="/forgot">Forgot your password?</a>
 <button>Sign In</button>
 </form>
</div>
<div class="overlay-container">
 <div class="overlay">
 <div class="overlay-panel overlay-left">
  <h1>Welcome Back!</h1>
  <p>To keep connected with us please login with your personal info</p>
  <button class="ghost" id="signIn">Sign In</button>
 </div>
 <div class="overlay-panel overlay-right">
  <h1>Hello, Friend!</h1>
  <p>Enter your personal details and start journey with us</p>
  <button class="ghost" id="signUp">Sign Up</button>
 </div>
 </div>
```

```
          </div>
          </div>
        </div>
        <div class="about" id="abouts">
         <p>
          Nutrition Assistant is a web-based tool that is capable of calculating nutrition
          from an image. It uses a food model to detect food items from images and a nutrition
          API to calculate nutrition information.
         </p>
        </div>
        </div>
       </div>
      </div>
     </div>

</body>
<script src="index.js">

 function ont() {
  document.getElementById('info').style.color = 'red';
 }
</script>
<script>
 window.watsonAssistantChatOptions = {
  integrationID: "4a603e76-8f59-4fcd-b826-8bf529498ba4", // The ID of this integration.
  region: "jp-tok", // The region your integration is hosted in.
  serviceInstanceID: "969aa05a-ff29-4d90-9012-ed8e7c6b1ccf", // The ID of your service instance.
  onLoad: function(instance) { instance.render(); }
 };
 setTimeout(function(){
  const t=document.createElement('script');
  t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') + "/WatsonAssistantChatEntry.js";
  document.head.appendChild(t);
 });
</script>
</html>
```

Style.css:

```
@media only screen and (max-width :768px ) {
  .colh {
    height: auto;
  }

  .lists {
    height: 330px;
    overflow: auto;
    flex-direction: column;
  }

 }



 body {
  background-color: rgb(255, 255, 255);
  font-family: Futura ,Trebuchet MS,Arial,sans-serif;
  min-height: 100vh;
  font-size: 1.3rem;

 }

 .heading {
  color: #330c3a;
```

```
}

.colh {
  height: auto;
  display: flex;
  justify-content: center;
  align-items: center;
  padding: 50px 0;
}

.colh > div {
  width: 100%;
}

.spaceimp {
  margin-bottom: 200px;
}

.lists {
  list-style-type: none;
  display: flex;
  font-size: 1.2rem;
  justify-content: space-around;
  align-items: center;
  align-content: center;
  transition: 2.8s;
}
.image{
  margin-top: 30px;
```

```
}
 .lists > li {
   text-decoration: none;
 }

 .lists > li > a {
   text-decoration: none;
   padding: 10px 40px;
   color: rgb(11, 9, 9);
   transition: .8s ease-in;
   border-radius: 26px;
   background: #e3e3e3;
   box-shadow: inset -17px 17px 33px #d1d1d1,
       inset 17px -17px 33px #f5f5f5;

 }

 .lists > li > a:hover {
   color: rgb(251, 249, 249);
   border-radius: 26px;
   background: #131111;
   box-shadow:  -11px 11px 22px #8a8484,
        11px -11px 22px #a49d9d;
 }

 p {
   text-align: justify;
 }

 .info {
   min-height: 50vh;
```

```
  color: rgb(1, 9, 9);
  min-width: 10vw;
  padding-left: 30px;
}

.content {
  margin-top: 50px;
  padding-bottom: 50px;
  box-sizing: border-box;
}

.content > info > p {
  font-size: 2.5rem;
}

.cont_1 {
  width: 100%;
  height: 100%;
  display: flex;
  justify-content: top;
  align-items: flex-end;
  flex-direction: column;
  flex-grow: 1;
  padding: 0 10px 0 0;
  position: relative;
}

.login {
  width: 100%;
  height: 100%;
  left: 0;
```

```
   top: 0;
   position: absolute;
   z-index: 1;
   display: none;
   margin-bottom: 50px;
   animation-name: loginhidder;
   animation-duration: 2s;
  }

  .about {
   background-color:  #691977;
   min-height: 200px;
   width: 80%;
   border-radius: 25px;
   margin-right: 10%;
   margin-top: 100px;
   padding: 20px;
   -webkit-box-shadow: 3px 3px 5px 6px rgb(89, 82, 82);  /* Safari 3-4, iOS 4.0.2 - 4.2, Android 2.3+
*/
   -moz-box-shadow:    3px 3px 5px 6px rgb(89, 82, 82);  /* Firefox 3.5 - 3.6 */
   box-shadow:         3px 3px 5px 6px rgb(89, 82, 82);
   display: none;
   animation-name: loginhidder;
   animation-duration: 2s;
  }

  @keyframes loginhidder {
   0% {opacity: 0;}
   50% {opacity: .5;}
   100% {opacity: 1;}
  }
```

```
.list {
  padding-left: 80px;
}

.options{
  padding: 0px 15px;
  background-color: rgba(188, 196, 196,0.8);
  box-sizing: border-box;
  text-align: center;
  margin-bottom: 20px;
  border-radius: 25px;
  display: flex;
  width: 90%;
  transition: opacity .9s;
}

.options > p {
  display: flex;
  justify-content: center;
  align-items: center;
  align-content: center;
  font-size: 1.2rem;
  font-style: bold;
}


#l1 {
  position: relative;
  animation-name: example;
  animation-duration: 1s;
```

```
}

#l2 {
  position: relative;
  animation-name: example;
  animation-duration: 1.5s;
}

#l3 {
  position: relative;
  animation-name: example;
  animation-duration: 2s;
}

#l4 {
  position: relative;
  animation-name: example;
  animation-duration: 2.5s;
}

@keyframes example {
0% {left:800px;}
     1% {left:792px;}
     2% {left:784px;}
     3% {left:776px;}
     4% {left:768px;}
     5% {left:760px;}
     6% {left:752px;}
     7% {left:744px;}
     8% {left:736px;}
     9% {left:728px;}
```

```
10% {left:720px;}

11% {left:712px;}

12% {left:704px;}

13% {left:696px;}

14% {left:688px;}

15% {left:680px;}

16% {left:672px;}

17% {left:664px;}

18% {left:656px;}

19% {left:648px;}

20% {left:640px;}

21% {left:632px;}

22% {left:624px;}

23% {left:616px;}

24% {left:608px;}

25% {left:600px;}

26% {left:592px;}

27% {left:584px;}

28% {left:576px;}

29% {left:568px;}

30% {left:560px;}

31% {left:552px;}

32% {left:544px;}

33% {left:536px;}

34% {left:528px;}

35% {left:520px;}

36% {left:512px;}

37% {left:504px;}

38% {left:496px;}

39% {left:488px;}

40% {left:480px;}
```

```
41% {left:472px;}

42% {left:464px;}

43% {left:456px;}

44% {left:448px;}

45% {left:440px;}

46% {left:432px;}

47% {left:424px;}

48% {left:416px;}

49% {left:408px;}

50% {left:400px;}

51% {left:392px;}

52% {left:384px;}

53% {left:376px;}

54% {left:368px;}

55% {left:360px;}

56% {left:352px;}

57% {left:344px;}

58% {left:336px;}

59% {left:328px;}

60% {left:320px;}

61% {left:312px;}

62% {left:304px;}

63% {left:296px;}

64% {left:288px;}

65% {left:280px;}

66% {left:272px;}

67% {left:264px;}

68% {left:256px;}

69% {left:248px;}

70% {left:240px;}

71% {left:232px;}
```

```
    72% {left:224px;}
    73% {left:216px;}
    74% {left:208px;}
    75% {left:200px;}
    76% {left:192px;}
    77% {left:184px;}
    78% {left:176px;}
    79% {left:168px;}
    80% {left:160px;}
    81% {left:152px;}
    82% {left:144px;}
    83% {left:136px;}
    84% {left:128px;}
    85% {left:120px;}
    86% {left:112px;}
    87% {left:104px;}
    88% {left:96px;}
    89% {left:88px;}
    90% {left:80px;}
    91% {left:72px;}
    92% {left:64px;}
    93% {left:56px;}
    94% {left:48px;}
    95% {left:40px;}
    96% {left:32px;}
    97% {left:24px;}
    98% {left:16px;}
    99% {left:8px;}
    100% {left:0px;}

}
```

```
h1 {
        font-weight: bold;
        margin: 0;
 }

 h2 {
        text-align: center;
 }

 p {
        font-size: 14px;
        font-weight: 100;
        line-height: 20px;
        letter-spacing: 0.5px;
        margin: 20px 0 30px;
 }

 .info > p {
  font-size: 1.2rem ;
  font-weight: 20;
         line-height: 30px;
         letter-spacing: 1px;
         margin: 20px 0 30px;
 }

 a {
        color: #333;
        font-size: 14px;
        text-decoration: none;
        margin: 15px 0;
```

```
}

button {
	border-radius: 20px;
	border: 1px solid hsl(276, 76%, 8%);
	background-color: #2e0637;
	color: #FFFFFF;
	font-size: 12px;
	font-weight: bold;
	padding: 12px 45px;
	letter-spacing: 1px;
	text-transform: uppercase;
	transition: transform 80ms ease-in;
}

button:active {
	transform: scale(0.95);
}

button:focus {
	outline: none;
}

button.ghost {
	background-color: transparent;
	border-color: #FFFFFF;
}

form {
	background-color: #FFFFFF;
	display: flex;
	align-items: center;
	justify-content: center;
	flex-direction: column;
	padding: 0 50px;
	height: 100%;
	text-align: center;
}

input {
	background-color: #eee;
	border: none;
	padding: 12px 15px;
	margin: 8px 0;
	width: 100%;
}

.container {
	background-color: #fff;
	border-radius: 10px;
	box-shadow: 0 14px 28px rgba(0,0,0,0.25),
			0 10px 10px rgba(0,0,0,0.22);
	position: relative;
	overflow: hidden;
	width: 768px;
	max-width: 100%;
	min-height: 480px;
}

.form-container {
	position: absolute;
	top: 0;
	height: 100%;
	transition: all 0.6s ease-in-out;
}

.sign-in-container {
	left: 0;
	width: 50%;
	z-index: 2;
}

.container.right-panel-active .sign-in-container {
	transform: translateX(100%);
}

.sign-up-container {
	left: 0;
	width: 50%;
	opacity: 0;
	z-index: 1;
}

.container.right-panel-active .sign-up-container {
	transform: translateX(100%);
	opacity: 1;
	z-index: 5;
	animation: show 0.6s;
}

@keyframes show {
	0%, 49.99% {
	  opacity: 0;
```

```
      z-index: 1;
    }

    50%, 100% {
      opacity: 1;
      z-index: 5;
    }
}

.overlay-container {
    position: absolute;
    top: 0;
    left: 50%;
    width: 50%;
    height: 100%;
    overflow: hidden;
    transition: transform 0.6s ease-in-out;
    z-index: 100;
}

.container.right-panel-active .overlay-container{
    transform: translateX(-100%);
}

.overlay {
    background: #330c3a;
    background: -webkit-linear-gradient(to right,#330c3a, #330c3a);
    background: linear-gradient(to right, #330c3a, #330c3a);
    background-repeat: no-repeat;
    background-size: cover;
    background-position: 0 0;
```

```
        color: #FFFFFF;
        position: relative;
        left: -100%;
        height: 100%;
        width: 200%;
        transform: translateX(0);
        transition: transform 0.6s ease-in-out;
}

.container.right-panel-active .overlay {
          transform: translateX(50%);
}

.overlay-panel {
        position: absolute;
        display: flex;
        align-items: center;
        justify-content: center;
        flex-direction: column;
        padding: 0 40px;
        text-align: center;
        top: 0;
        height: 100%;
        width: 50%;
        transform: translateX(0);
        transition: transform 0.6s ease-in-out;
}

.overlay-left {
        transform: translateX(-20%);
}
```

```
 .container.right-panel-active .overlay-left {
	transform: translateX(0);
 }

 .overlay-right {
	right: 0;
	transform: translateX(0);
 }

 .container.right-panel-active .overlay-right {
	transform: translateX(20%);
 }


/*input[type=text], input[type=password] {
	width: 100%;
	padding: 2px 10px;
	margin: 8px 0;
	display: inline-block;
	border: 0px solid #ccc;
	box-sizing: border-box;
  border: 1;
 }
 button {
  width: 100%;
  margin-top: 30px;
  background-color: green;
  border: 0;
  padding: 5px;
 }*/
```

index.js:

```
  const signUpButton = document.getElementById('signUp');

  const signInButton = document.getElementById('signIn');

  const container = document.getElementById('container');


  const switchone = document.getElementById('c1');

  const switchtwo = document.getElementById("c2");

  const switchthree = document.getElementById('c3');

  const switchfour = document.getElementById('c4');


  const Fswitchone = document.getElementById('l1');

  const Fswitchtwo = document.getElementById("l2");

  const Fswitchthree = document.getElementById('l3');

  const Fswitchfour = document.getElementById('l4');


  const space = document.getElementById('infos');


  var pre_state = 0;

  var stateone = 0;

  var statetwo = 0;

  var statethree = 0;


  signUpButton.addEventListener('click', () => {

  container.classList.add("right-panel-active");

});


  signInButton.addEventListener('click', () => {

  container.classList.remove("right-panel-active");

});


switchone.addEventListener('click', remover);
```

```
switchtwo.addEventListener('click', signin);
switchthree.addEventListener('click', Signup)
switchfour.addEventListener('click', about);

Fswitchone.addEventListener('click', remover);
Fswitchtwo.addEventListener('click', signin);
Fswitchthree.addEventListener('click', Signup)
Fswitchfour.addEventListener('click', about);


function remover() {

  if(pre_state == 1){
    pre_state = 0;
    space.classList.remove("spaceimp");
    document.getElementById("abouts").style.display = "none";
    document.getElementById("logins").style.display = "none";
    document.getElementById("l1").style.display = "flex";
    document.getElementById("l2").style.display = "flex";
    document.getElementById("l3").style.display = "flex";
    document.getElementById("l4").style.display = "flex";
}
}

function div_adder () {
  space.classList.add("spaceimp");
  document.getElementById("abouts").style.display = "none";
  document.getElementById("logins").style.display = "block";
  document.getElementById("l1").style.display = "none";
  document.getElementById("l2").style.display = "none";
  document.getElementById("l3").style.display = "none";
  document.getElementById("l4").style.display = "none";
}

function about_adder () {
  //space.classList.add("spaceimp");
 // remover();
  document.getElementById("abouts").style.display = "block";
  document.getElementById("l1").style.display = "none";
  document.getElementById("l2").style.display = "none";
  document.getElementById("l3").style.display = "none";
  document.getElementById("l4").style.display = "none";
}


function signin() {

  if(pre_state == 0) {
    pre_state = 1;
    stateone = 1;
    statetwo = 0;
    statethree = 0;
    container.classList.remove("right-panel-active");
    div_adder();
  }else {
    if(stateone == 0) {
      pre_state = 1;
      stateone = 1;
      statetwo = 0;
      statethree = 0;
      container.classList.remove("right-panel-active");
      div_adder();
    }else {
      remover();
    }
  }

}

function Signup() {

  if(pre_state == 0) {
    pre_state = 1;
    stateone = 0;
    statetwo = 1;
    statethree = 0;
    container.classList.add("right-panel-active");
    div_adder();
  }else {
    if(statetwo == 0) {
      pre_state = 1;
      stateone = 0;
      statetwo = 1;
      statethree = 0;
      container.classList.add("right-panel-active");
      div_adder();
    }else {
      remover();
    }
  }
}

function about() {
  if(pre_state == 0){
    pre_state = 1;
    stateone = 0;
    statetwo = 0;
    statethree = 3;
    about_adder();
  }else{
    if(statethree == 0){
      remover();
      pre_state = 1;
      stateone = 0;
      statetwo = 0;
      statethree = 3;
      about_adder();
    }else{
      remover();
    }
  }

}

function unvisible(x) {
    if(pre_state == 0) {
      document.getElementById("l1").style.display = "none";
      document.getElementById("l2").style.display = "none";
      document.getElementById("l3").style.display = "none";
      document.getElementById("l4").style.display = "none";
    }

  }
```

```
function visible(x){
  if(pre_state == 0) {
    document.getElementById("abouts").style.display = "block";
    //space.classList.add("spaceimp");
    container.classList.add("right-panel-active");
    document.getElementById("l1").style.display = "none";
    document.getElementById("l2").style.display = "none";
    document.getElementById("l3").style.display = "none";
    document.getElementById("l4").style.display = "none";
  }
}

function unsignin(x) {
  if(pre_state == 0){
    container.classList.remove("right-panel-active");
    space.classList.remove("spaceimp");
    document.getElementById("logins").style.display = "none";
    document.getElementById("abouts").style.display = "none";
    document.getElementById("l1").style.display = "flex";
    document.getElementById("l2").style.display = "flex";
    document.getElementById("l3").style.display = "flex";
    document.getElementById("l4").style.display = "flex";
  }

}



function signinOne(x){
  if(pre_state == 0) {
    container.classList.remove("right-panel-active");
    space.classList.add("spaceimp");
    document.getElementById("logins").style.display = "block";
    document.getElementById("l1").style.display = "none";
    document.getElementById("l2").style.display = "none";
    document.getElementById("l3").style.display = "none";
    document.getElementById("l4").style.display = "none";
  }
}

function signinTwo(x){
  if(pre_state == 0) {
    document.getElementById("logins").style.display = "block";
    space.classList.add("spaceimp");
    container.classList.add("right-panel-active");
    document.getElementById("l1").style.display = "none";
    document.getElementById("l2").style.display = "none";
    document.getElementById("l3").style.display = "none";
    document.getElementById("l4").style.display = "none";
  }

function setcon(x) {
  if(pre_state == 0) {
    document.getElementById("abouts").style.display = "block";
      //space.classList.add("spaceimp");
    container.classList.add("right-panel-active");
    document.getElementById("l1").style.display = "none";
    document.getElementById("l2").style.display = "none";
    document.getElementById("l3").style.display = "none";
    document.getElementById("l4").style.display = "none";
    }
}
```

```
}
```

app.py:

```
from flask import Flask, render_template, request, redirect, url_for, session

import ibm_db

import re

from sendemail import sendmail,sendgridmail

import smtplib


app = Flask(__name__)


app.secret_key = 'a'


conn = ibm_db.connect("DATABASE=bludb;HOSTNAME="2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328";UID=bsb19147;PWD=M8Q6aCGQuLHwiHkU",'','')


@app.route('/')


def homer():
   return render_template('home.html')




@app.route('/login',methods =['GET', 'POST'])
def login():
   global userid
   msg = ''




   if request.method == 'POST' :
      username = request.form['username']
```

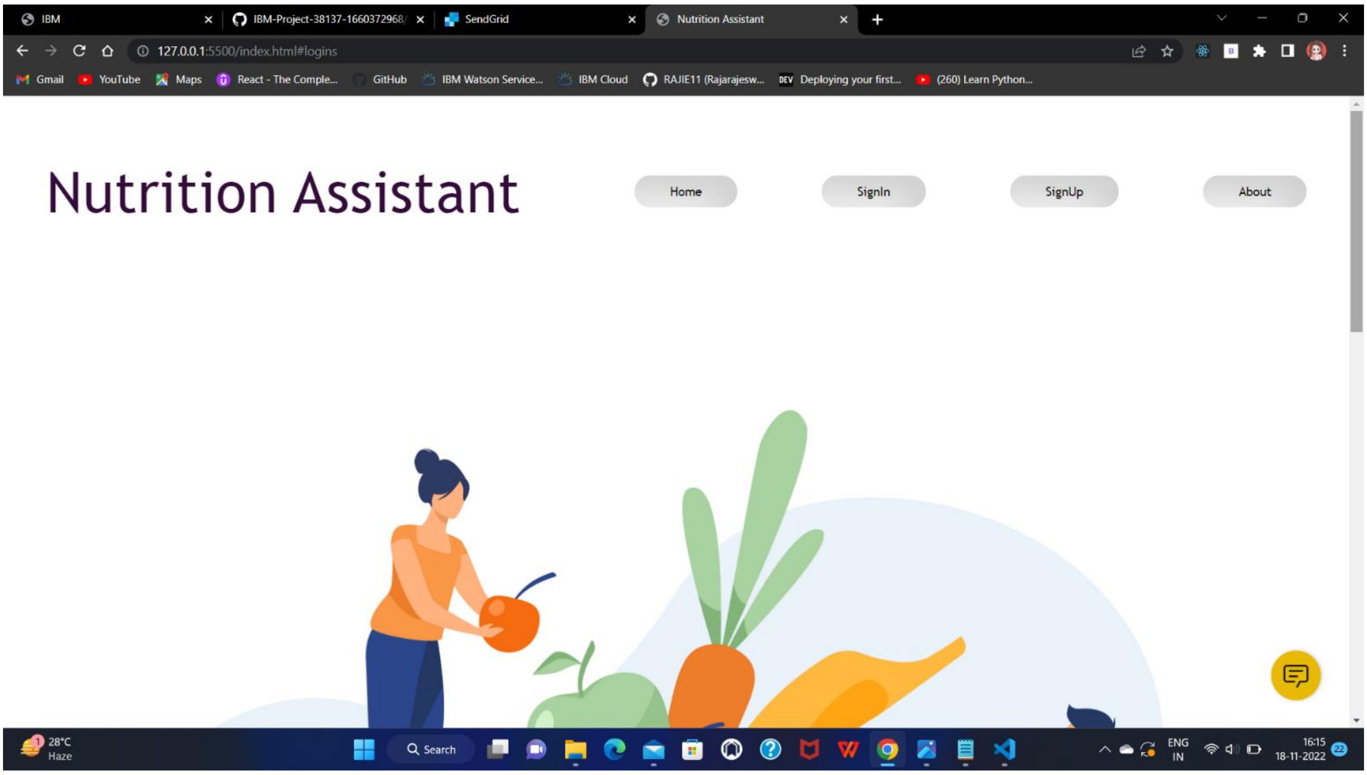
```
        password = request.form['password']
        sql = "SELECT * FROM users WHERE username =? AND password=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print (account)
        if account:
            session['loggedin'] = True
            session['id'] = account['USERNAME']
            userid=  account['USERNAME']
            session['username'] = account['USERNAME']
            msg = 'Logged in successfully !'

            msg = 'Logged in successfully !'
            return render_template('dashboard.html', msg = msg)
        else:
            msg = 'Incorrect username / password !'
    return render_template('login.html', msg = msg)




@app.route('/register', methods =['GET', 'POST'])
def registet():
    msg = ''
    if request.method == 'POST' :
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']
        sql = "SELECT * FROM users WHERE username =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            msg = 'Account already exists !'
        elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):
            msg = 'Invalid email address !'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'name must contain only characters and numbers !'
        else:
            insert_sql = "INSERT INTO  users VALUES (?, ?, ?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, username)
            ibm_db.bind_param(prep_stmt, 2, email)
            ibm_db.bind_param(prep_stmt, 3, password)
            ibm_db.execute(prep_stmt)
            msg = 'You have successfully registered !'
            TEXT = "Hello "+username + ",\n\n"+ """Thanks for applying registring at smartinterns """
            message  = 'Subject: {}\n\n{}'.format("smartinterns Carrers", TEXT)
            #sendmail(TEXT,email)
            sendgridmail(email,TEXT)
    elif request.method == 'POST':
        msg = 'Please fill out the form !'
    return render_template('register.html', msg = msg)


@app.route('/dashboard')
def dash():
    return render_template('dashboard.html')


@app.route('/apply',methods =['GET', 'POST'])
def apply():
    msg = ''
    if request.method == 'POST' :
        username = request.form['username']
        email = request.form['email']

        qualification= request.form['qualification']
        skills = request.form['skills']
        jobs = request.form['s']
        sql = "SELECT * FROM users WHERE username =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            msg = 'there is only 1 job position! for you'
            return render_template('apply.html', msg = msg)



        insert_sql = "INSERT INTO  job VALUES (?, ?, ?, ?, ?)"
        prep_stmt = ibm_db.prepare(conn, insert_sql)
        ibm_db.bind_param(prep_stmt, 1, username)
        ibm_db.bind_param(prep_stmt, 2, email)
        ibm_db.bind_param(prep_stmt, 3, qualification)
        ibm_db.bind_param(prep_stmt, 4, skills)
        ibm_db.bind_param(prep_stmt, 5, jobs)
```

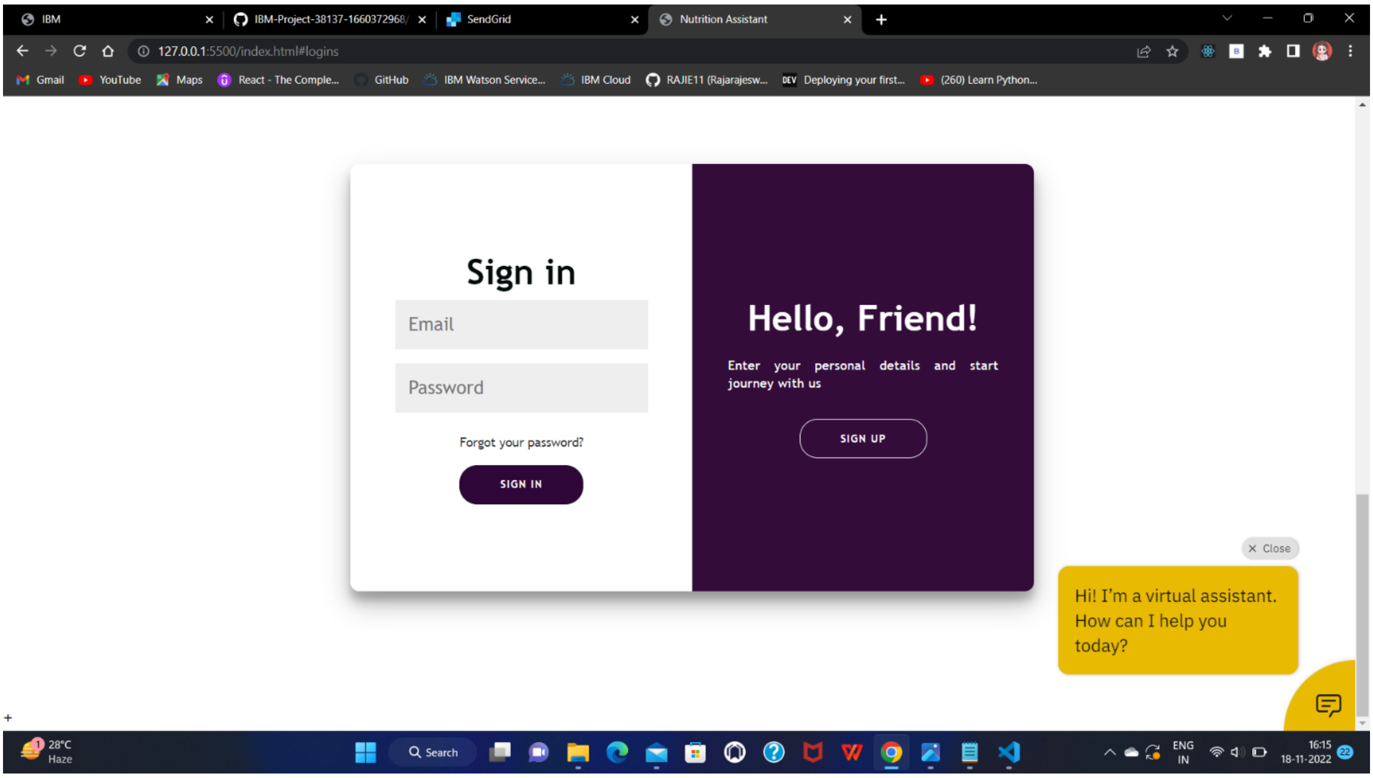
```
         ibm_db.execute(prep_stmt)
         msg = 'You have successfully applied for job !'
         session['loggedin'] = True
         TEXT = "Hello sandeep,a new appliaction for job position" +jobs+"is requested"

         #sendmail(TEXT,"sandeep@thesmartbridge.com")
         sendgridmail("sandeep@thesmartbridge.com",TEXT)



     elif request.method == 'POST':
         msg = 'Please fill out the form !'
     return render_template('apply.html', msg = msg)

@app.route('/display')
def display():
    print(session["username"],session['id'])

    cursor = mysql.connection.cursor()
    cursor.execute('SELECT * FROM job WHERE userid = % s', (session['id'],))
    account = cursor.fetchone()
    print("accountdislay",account)



    return render_template('display.html',account = account)

@app.route('/logout')

def logout():
   session.pop('loggedin', None)
   session.pop('id', None)
   session.pop('username', None)
   return render_template('home.html')




if __name__ == '__main__':
   app.run(host='0.0.0.0') from flask import Flask, render_template, request, redirect, url_for, session
import ibm_db
import re
from sendemail import sendmail,sendgridmail
import smtplib

app = Flask(__name__)

app.secret_key = 'a'

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME="2d46b6b4-cbf6-40eb-bbce-
6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328";UID=bsb19147;PWD=M8
Q6aCGQuLHwiHkU",'','')

@app.route('/')

def homer():
    return render_template('home.html')



@app.route('/login',methods =['GET', 'POST'])
def login():
    global userid
    msg = ''
    if request.method == 'POST' :
        username = request.form['username']
        password = request.form['password']
        sql = "SELECT * FROM users WHERE username =? AND password=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print (account)
        if account:
            session['loggedin'] = True
            session['id'] = account['USERNAME']
            userid=  account['USERNAME']
            session['username'] = account['USERNAME']
            msg = 'Logged in successfully !'

            msg = 'Logged in successfully !'
            return render_template('dashboard.html', msg = msg)
        else:
            msg = 'Incorrect username / password !'
    return render_template('login.html', msg = msg)




@app.route('/register', methods =['GET', 'POST'])
def registet():
    msg = ''
    if request.method == 'POST' :
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']
        sql = "SELECT * FROM users WHERE username =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            msg = 'Account already exists !'
        elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):
            msg = 'Invalid email address !'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'name must contain only characters and numbers !'
        else:
            insert_sql = "INSERT INTO  users VALUES (?, ?, ?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, username)
            ibm_db.bind_param(prep_stmt, 2, email)
            ibm_db.bind_param(prep_stmt, 3, password)
            ibm_db.execute(prep_stmt)
            msg = 'You have successfully registered !'
            TEXT = "Hello "+username + ",\n\n"+ """Thanks for applying registring at smartinterns """
            message  = 'Subject: {}\n\n{}'.format("smartinterns Carrers", TEXT)
            #sendmail(TEXT,email)
            sendgridmail(email,TEXT)
    elif request.method == 'POST':
        msg = 'Please fill out the form !'
    return render_template('register.html', msg = msg)


@app.route('/dashboard')
```

```
def dash():


    return render_template('dashboard.html')


@app.route('/apply',methods =['GET', 'POST'])
def apply():
     msg = ''
     if request.method == 'POST' :
         username = request.form['username']
         email = request.form['email']


         qualification= request.form['qualification']
         skills = request.form['skills']
         jobs = request.form['s']
         sql = "SELECT * FROM users WHERE username =?"
         stmt = ibm_db.prepare(conn, sql)
         ibm_db.bind_param(stmt,1,username)
         ibm_db.execute(stmt)
         account = ibm_db.fetch_assoc(stmt)
         print(account)
         if account:
            msg = 'there is only 1 job position! for you'
            return render_template('apply.html', msg = msg)




         insert_sql = "INSERT INTO  job VALUES (?, ?, ?, ?, ?)"
         prep_stmt = ibm_db.prepare(conn, insert_sql)
         ibm_db.bind_param(prep_stmt, 1, username)
         ibm_db.bind_param(prep_stmt, 2, email)
         ibm_db.bind_param(prep_stmt, 3, qualification)
```

```
        ibm_db.bind_param(prep_stmt, 4, skills)
        ibm_db.bind_param(prep_stmt, 5, jobs)
        ibm_db.execute(prep_stmt)
        msg = 'You have successfully applied for job !'
        session['loggedin'] = True
        TEXT = "Hello sandeep,a new appliaction for job position" +jobs+"is requested"

        #sendmail(TEXT,"sandeep@thesmartbridge.com")
        sendgridmail("sandeep@thesmartbridge.com",TEXT)




    elif request.method == 'POST':
        msg = 'Please fill out the form !'
    return render_template('apply.html', msg = msg)

@app.route('/display')
def display():
    print(session["username"],session['id'])

    cursor = mysql.connection.cursor()
    cursor.execute('SELECT * FROM job WHERE userid = % s', (session['id'],))
    account = cursor.fetchone()
    print("accountdislay",account)



    return render_template('display.html',account = account)

@app.route('/logout')

def logout():
  session.pop('loggedin', None)
  session.pop('id', None)
  session.pop('username', None)
  return render_template('home.html')



if __name__ == '__main__':
  app.run(host='0.0.0.0')
```

Output:





Sign in

Email

Password

Forgot your password?

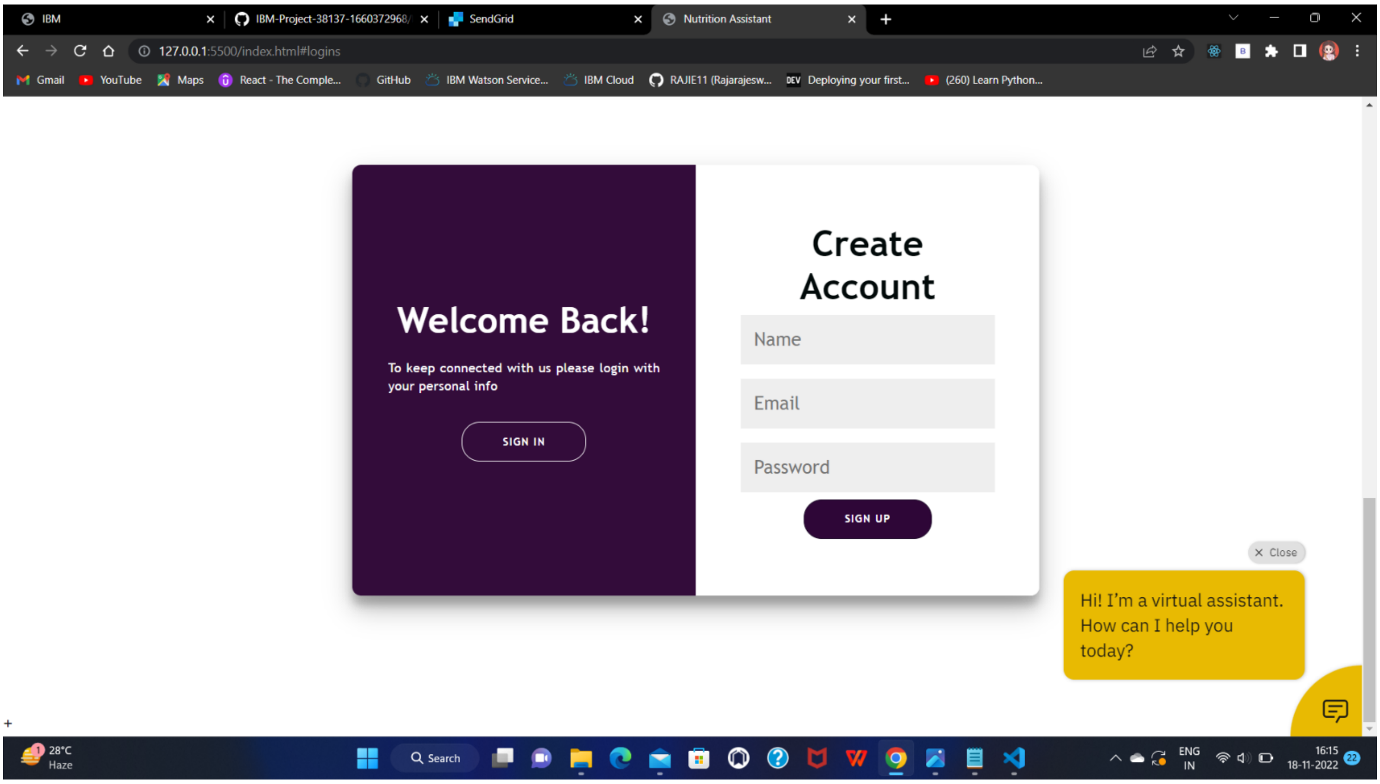
SIGN IN

Hello, Friend!

Enter your personal details and start journey with us

SIGN UP

Hi! I'm a virtual assistant. How can I help you today?



Welcome Back!

To keep connected with us please login with your personal info

SIGN IN

Create Account

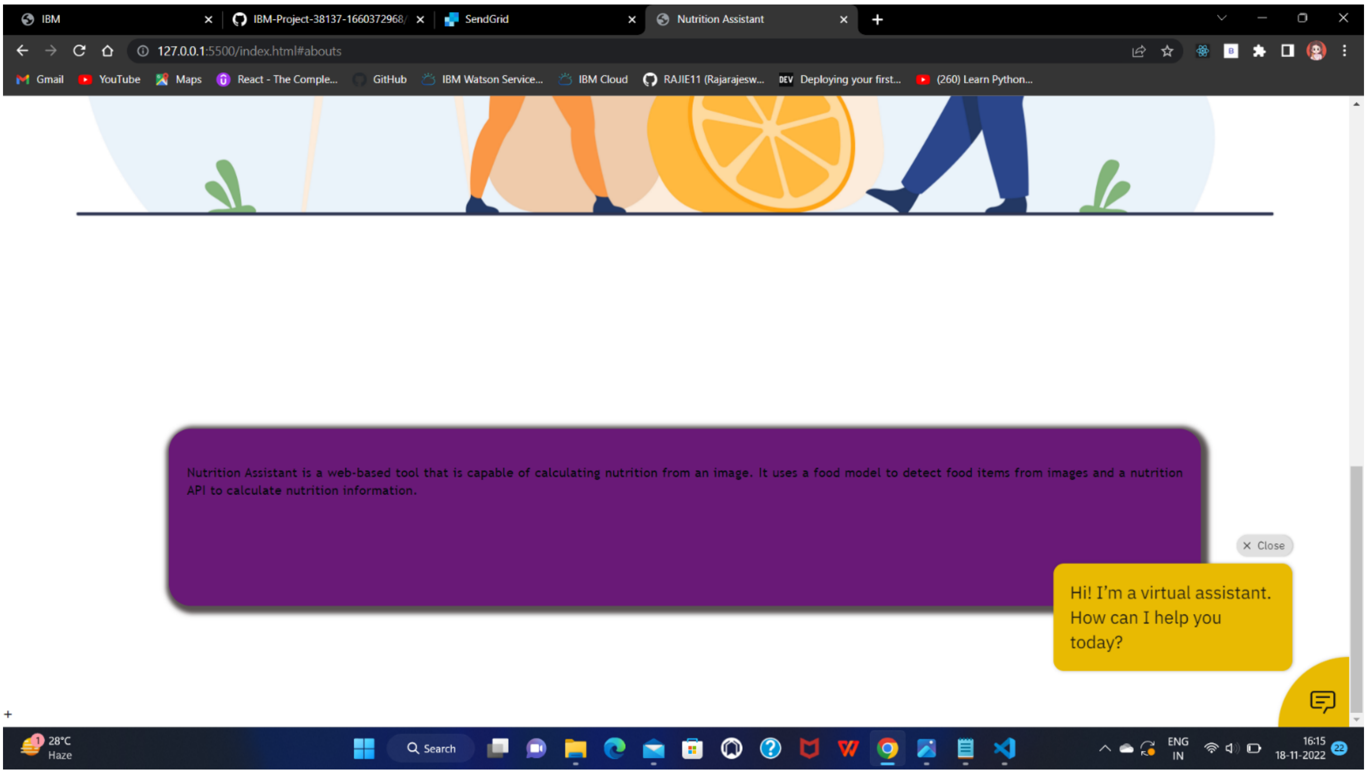
Name

Email

Password

SIGN UP

Hi! I'm a virtual assistant. How can I help you today?



Nutrition Assistant is a web-based tool that is capable of calculating nutrition from an image. It uses a food model to detect food items from images and a nutrition API to calculate nutrition information.

Hi! I'm a virtual assistant. How can I help you today?