Assignment -2

Question-1:

1. Create a Flask App
2. Add the Home page, About Page
3. Add the Bootstrap
4. Add the Sign in page and App the Signup Page + database connectivity

Solution:

*****index.js*****

```
function deleteNote(noteId) {
  fetch("/delete-note", {
  method: "POST",
    body: JSON.stringify({ noteId: noteId }),
  }).then((_res) => {
    window.location.href = "/";
  });
}
```

*****base.html*****

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <link
      rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
      integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
      crossorigin="anonymous"
    />
    <link
      rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css"
      crossorigin="anonymous"
    />

    <title>{% block title %}Home{% endblock %}</title>
```

```
 </head>
 <body>
  <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
   <button
    class="navbar-toggler"
    type="button"
    data-toggle="collapse"
    data-target="#navbar"
   >
    <span class="navbar-toggler-icon"></span>
   </button>
   <div class="collapse navbar-collapse" id="navbar">
    <div class="navbar-nav">
     {% if user.is_authenticated %}
     <a class="nav-item nav-link" id="home" href="/">Home</a>
     <a class="nav-item nav-link" id="logout" href="/logout">Logout</a>
     {% else %}
     <a class="nav-item nav-link" id="login" href="/login">Login</a>
     <a class="nav-item nav-link" id="signUp" href="/sign-up">Sign Up</a>
     {% endif %}
    </div>
   </div>
  </nav>

  {% with messages = get_flashed_messages(with_categories=true) %} {% if
  messages %} {% for category, message in messages %} {% if category ==
  'error' %}
  <div class="alert alert-danger alter-dismissable fade show" role="alert">
   {{ message }}
   <button type="button" class="close" data-dismiss="alert">
    <span aria-hidden="true">&times;</span>
   </button>
  </div>
  {% else %}
  <div class="alert alert-success alter-dismissable fade show" role="alert">
   {{ message }}
   <button type="button" class="close" data-dismiss="alert">
    <span aria-hidden="true">&times;</span>
   </button>
  </div>
  {% endif %} {% endfor %} {% endif %} {% endwith %}

  <div class="container">{% block content %} {% endblock %}</div>
  <script
   src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
   integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
   crossorigin="anonymous"
  ></script>
  <script
   src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
   integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
   crossorigin="anonymous"
  ></script>
  <script
   src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
```

```
      integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
      crossorigin="anonymous"
    ></script>

    <script
      type="text/javascript"
      src="{{ url_for('static', filename='index.js') }}"
    ></script>
  </body>
</html>
```

home.hml

```
{% extends "base.html" %} {% block title %}Home{% endblock %} {% block content
%}
<h1 align="center">Notes</h1>
<ul class="list-group list-group-flush" id="notes">
  {% for note in user.notes %}
  <li class="list-group-item">
    {{ note.data }}
    <button type="button" class="close" onClick="deleteNote({{ note.id }})">
      <span aria-hidden="true">&times;</span>
    </button>
  </li>
  {% endfor %}
</ul>
<form method="POST">
  <textarea name="note" id="note" class="form-control"></textarea>
  <br />
  <div align="center">
    <button type="submit" class="btn btn-primary">Add Note</button>
  </div>
</form>
{% endblock %}
```

login.html

```
{% extends "base.html" %} {% block title %}Login{% endblock %} {% block content
%}
<form method="POST">
  <h3 align="center">Login</h3>
  <div class="form-group">
    <label for="email">Email Address</label>
    <input
      type="email"
      class="form-control"
      id="email"
      name="email"
      placeholder="Enter email"
    />
  </div>
  <div class="form-group">
    <label for="password">Password</label>
    <input
      type="password"
      class="form-control"
      id="password"
      name="password"
      placeholder="Enter password"
    />
  </div>
  <br />
  <button type="submit" class="btn btn-primary">Login</button>
</form>
{% endblock %}
```

*****sign_up.html*****

```
{% extends "base.html" %} {% block title %}Sign Up{% endblock %} {% block
content %}
<form method="POST">
  <h3 align="center">Sign Up</h3>
  <div class="form-group">
    <label for="email">Email Address</label>
    <input
      type="email"
      class="form-control"
      id="email"
      name="email"
      placeholder="Enter email"
    />
  </div>
  <div class="form-group">
    <label for="firstName">First Name</label>
    <input
      type="text"
      class="form-control"
      id="firstName"
      name="firstName"
      placeholder="Enter first name"
    />
  </div>
  <div class="form-group">
    <label for="password1">Password</label>
    <input
      type="password"
      class="form-control"
      id="password1"
      name="password1"
      placeholder="Enter password"
    />
  </div>
  <div class="form-group">
    <label for="password2">Password (Confirm)</label>
    <input
      type="password"
      class="form-control"
      id="password2"
      name="password2"
      placeholder="Confirm password"
    />
  </div>
  <br />
  <button type="submit" class="btn btn-primary">Submit</button>
</form>
{% endblock %}
```

init.py

```
from flask import Flask
from flask_sqlalchemy import SQLAlchemy
from os import path
from flask_login import LoginManager

db = SQLAlchemy()
DB_NAME = "database.db"


def create_app():
    app = Flask(__name__)
    app.config['SECRET_KEY'] = 'hjshjhdjah kjshkjdhjs'
    app.config['SQLALCHEMY_DATABASE_URI'] = f'sqlite:///{DB_NAME}'
    db.init_app(app)

    from .views import views
    from .auth import auth

    app.register_blueprint(views, url_prefix='/')
    app.register_blueprint(auth, url_prefix='/')

    from .models import User, Note

    create_database(app)

    login_manager = LoginManager()
    login_manager.login_view = 'auth.login'
    login_manager.init_app(app)

    @login_manager.user_loader
    def load_user(id):
        return User.query.get(int(id))

    return app


def create_database(app):
    if not path.exists('website/' + DB_NAME):
        db.create_all(app=app)
        print('Created Database!')
```

auth.py

```
from flask import Blueprint, render_template, request, flash, redirect, url_for
from .models import User
from werkzeug.security import generate_password_hash, check_password_hash
from . import db
from flask_login import login_user, login_required, logout_user, current_user


auth = Blueprint('auth', __name__)


@auth.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        email = request.form.get('email')
        password = request.form.get('password')

        user = User.query.filter_by(email=email).first()
        if user:
            if check_password_hash(user.password, password):
                flash('Logged in successfully!', category='success')
                login_user(user, remember=True)
                return redirect(url_for('views.home'))
            else:
                flash('Incorrect password, try again.', category='error')
        else:
            flash('Email does not exist.', category='error')

    return render_template("login.html", user=current_user)


@auth.route('/logout')
@login_required
def logout():
    logout_user()
    return redirect(url_for('auth.login'))


@auth.route('/sign-up', methods=['GET', 'POST'])
def sign_up():
    if request.method == 'POST':
        email = request.form.get('email')
        first_name = request.form.get('firstName')
        password1 = request.form.get('password1')
        password2 = request.form.get('password2')

        user = User.query.filter_by(email=email).first()
        if user:
            flash('Email already exists.', category='error')
        elif len(email) < 4:
            flash('Email must be greater than 3 characters.', category='error')
        elif len(first_name) < 2:
            flash('First name must be greater than 1 character.', category='error')
        elif password1 != password2:
            flash('Passwords don\'t match.', category='error')
        elif len(password1) < 7:
            flash('Password must be at least 7 characters.', category='error')
        else:
            new_user = User(email=email, first_name=first_name, password=generate_password_hash(
                password1, method='sha256'))
            db.session.add(new_user)
            db.session.commit()
            login_user(new_user, remember=True)
            flash('Account created!', category='success')
            return redirect(url_for('views.home'))

    return render_template("sign_up.html", user=current_user)
```

models.py

```
afrom . import db
from flask_login import UserMixin
from sqlalchemy.sql import func


class Note(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    data = db.Column(db.String(10000))
    date = db.Column(db.DateTime(timezone=True), default=func.now())
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'))


class User(db.Model, UserMixin):
    id = db.Column(db.Integer, primary_key=True)
    email = db.Column(db.String(150), unique=True)
    password = db.Column(db.String(150))
    first_name = db.Column(db.String(150))
    notes = db.relationship('Note')
```

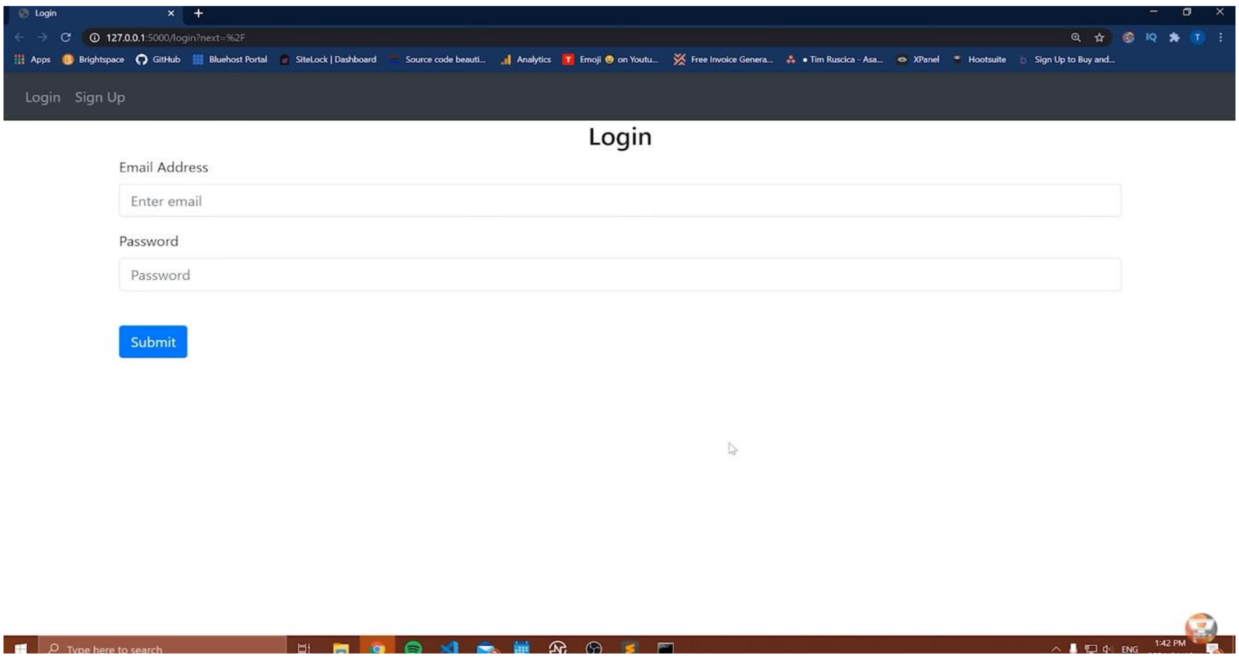
views.py

```
from flask import Blueprint, render_template, request, flash, jsonify
from flask_login import login_required, current_user
from .models import Note
from . import db
import json

views = Blueprint('views', __name__)


@views.route('/', methods=['GET', 'POST'])
@login_required
def home():
    if request.method == 'POST':
        note = request.form.get('note')

        if len(note) < 1:
            flash('Note is too short!', category='error')
        else:
            new_note = Note(data=note, user_id=current_user.id)
            db.session.add(new_note)
            db.session.commit()
            flash('Note added!', category='success')

    return render_template("home.html", user=current_user)
```

```
@views.route('/delete-note', methods=['POST'])
def delete_note():
    note = json.loads(request.data)
    noteId = note['noteId']
    note = Note.query.get(noteId)
    if note:
        if note.user_id == current_user.id:
            db.session.delete(note)
            db.session.commit()

    return jsonify({})
```

OUTPUT SCREENSHOTS

Step 1 : enter our email address and password



Login

Email Address

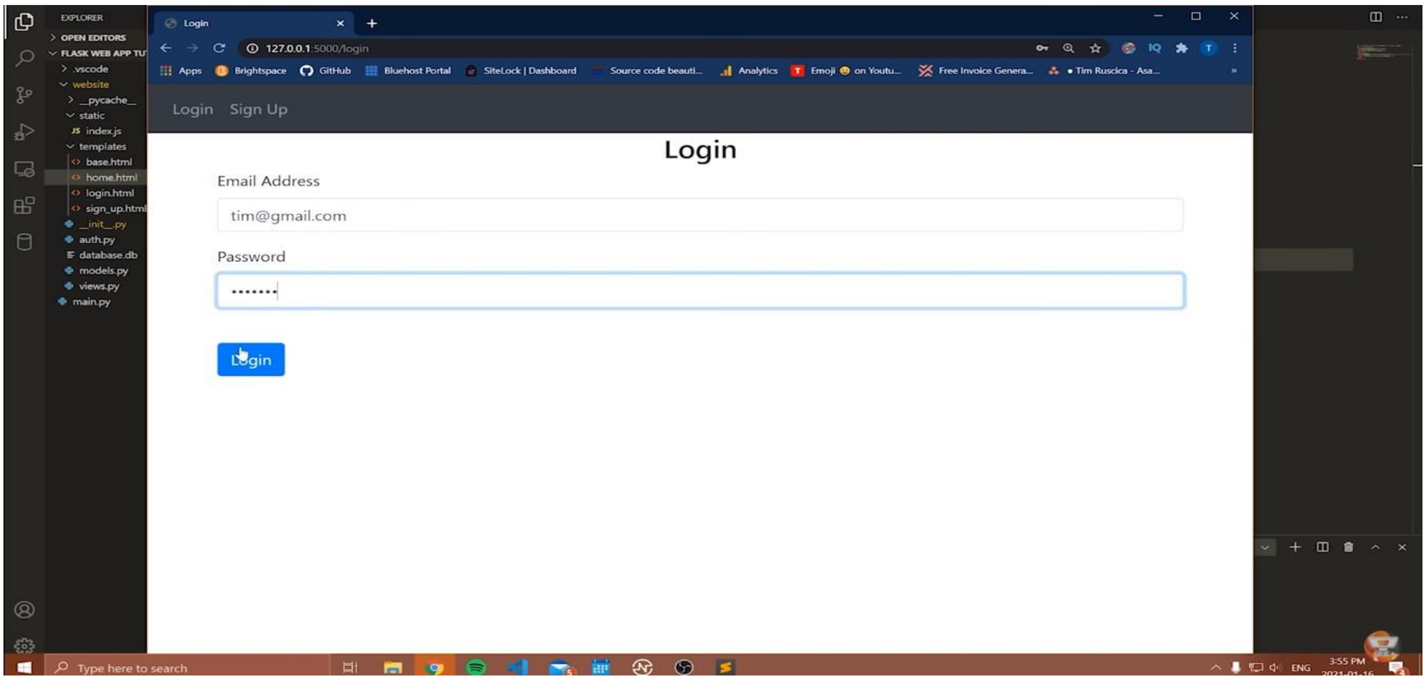
Enter email

Password

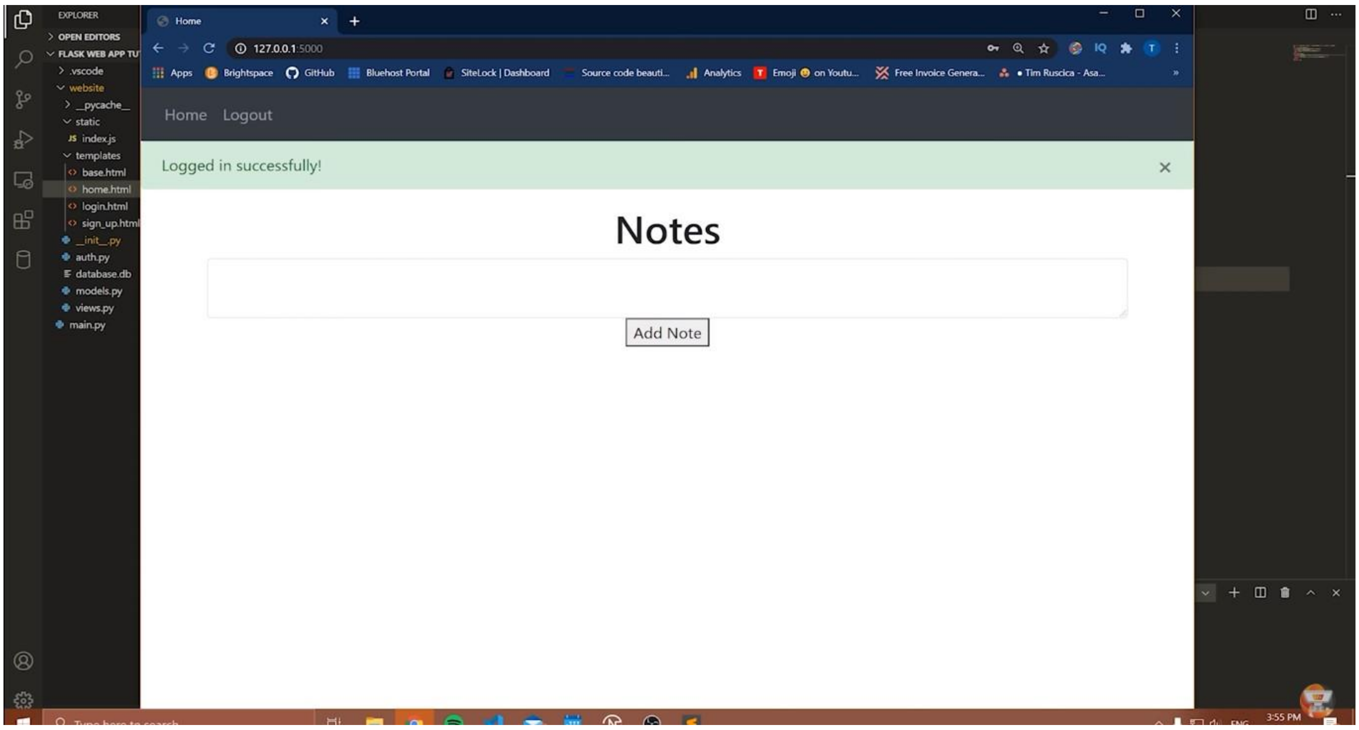
Password

Submit

STEP 2 : After enter that click “Login”



After that website redirect you to notes :



Home
127.0.0.1:5000
Home Logout
Logged in successfully!
Notes
Add Note