Notler

A Notes Application

Made by

Aravinth .S.S

Shvuayu Sarkar

Index

Acknowledgement

I would like to express my special thanks of gratitude to my teacher Mr. Christy Thomas and our Principal Mrs. Smriti Singh who gave me the golden opportunity to do this project on the topic Notes Application. It helped me in doing a lot of research and I came to know about many things related to this topic.

Finally, I would also like to thank Atharva Gondhali, Dev Godhaniya, my parents and friends who helped me a lot in finalizing this project within the limited time frame

Chapter-1: Introduction

1. Preface

In today's fast developing world we keep on getting a lot of work and we tend to maintain a lot of short notes or memos. Now as the technology started developing we moved on to digital notes and memos but they were not synchronised between our devices, so to solve this problem we came up with a solution and that is nothing but a cloud synchronised notes application that can be accessed from any device and from anywhere and at any time.

2. Abstract

There are already a lot of notes application in the market that allows people to make and save notes or memos but none of them synchronize the notes between multiple devices which was a major drawback of such applications.

Our main aim with this project was to remove this drawback and keep the notes synchronized between multiple devices so that the users don't have to check different devices for a particular note.

We achieved this by use of a technology called cloud computing which allowed us to make our application globally available and on every device that has a web browser.

3. Project Overview

Notler is a cloud based notes application that can be accessed in nearly all devices that has a web browser in it.

This program stores user's notes and shows them on demand.

The program will create a database where it'll be storing user's login information and user's notes.

The values in the database will be edited according to the user's input like the notes, email-id and their account's password

4. Administrative and user modules

When someone tries to access this program firstly they'll be asked to sign in or sign up which is accessed or stored in a MySQL database respectively. Then the user will be taken to the home screen where all of their notes will be displayed and they can either add new or edit, from there they can go to settings update their credentials or go to their profile to access those credentials and after using they can simply logout of the application. The users will be automatically logged in if they don't log out of the program. The program also has a profile photo system to make it look more attractive in the viewers perspective this is done using Gravatar.

5. Features

The features of the program are as follows:

- Notes synchronization
  - Synchronises all the data in the MySQL database so no need to worry about data loss.
- Updating notes
  - Change something or replace an existing note.
- Available in almost every device
  - Works in any device that has a web browser and an active internet connection.
- Automatic login
  - No need to enter login details every time to access the notes.
- Profile Photo System
  - Allowing users to set profile picture.
- Changing personal information
  - Allows users to changes their personal information like email-id and password without any hesitation.

6. Objectives

The objective of this project are:

1. Make notes synchronized between devices.
2. Making the application accessible in all devices that has a web browser in it.
3. Giving users a completely new experience in notes app.
4. Helping people to manage notes easily

Chapter-2: Requirements and Specification

1. User interface

The user interface for this program is created using Python and HTML (Hyper Text Markup Language).

Python in a high level interpreter language and an object oriented programing language which helps programmers to write logical codes for large-scale projects. Unlike other programming languages Python is very user friendly language which makes it a go to language for computer beginners and it is also the most used/learned programing language as of December 2021

Hyper Text Markup Language (HTML) is a standard markup language for documents designed to be displayed in a web browser. It can also be assisted by technologies such as Cascading Style Sheets (CSS) and Python

2. Database

For Database in this project we have used MySQL, MySQL is an open-source Relational Database Management System (RDBMS) which organises data in rows and columns.

MySQL is based on Structured Query Language (SQL) which is a programming language used to create, modify, extract data from a RDBMS

MySQL is an open-source software under GNU License which is made by Oracle popularly known for their programming language JAVA

In this project MySQL database has been used to store user's login information and notes

3. Functions and Modules used

The Function used are as follows:

1. app.route()
2. app.run()

The Modules user are as follows:

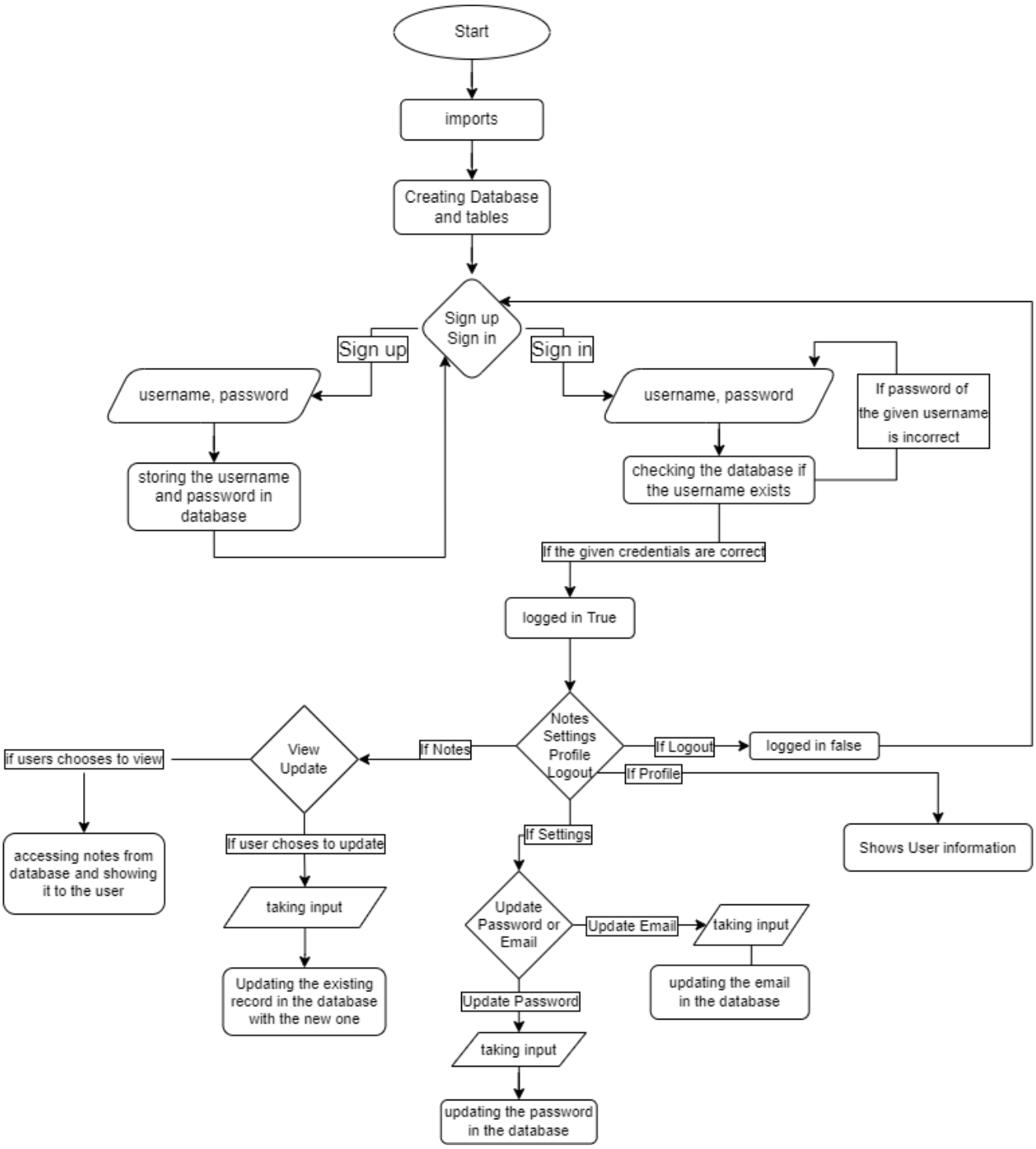
1. flask
2. flask_mysqldb
3. mysql.connector
4. urllib
5. hashlib

Functions and Modules created

The Functions created are as follows:

1. accountable – this function creates a table in database for storing user information's
2. notetable – this function creates a table to store notes in it
3. main – this function manages the automatic login system
4. login – this function is used to login the user into the application
5. signup – this function is used to create new user account
6. signout – this function logs out the user from the app
7. home – this function takes the user to the main page of the app
8. updatenote – this function is used to update note
9. profile – this function is used to take the user to their profile page
10. settings – this function is used to take the user to the settings menu
11. avatar – used for profile photo system

4. Algorithm


Start
imports
Creating Database and tables
Sign up Sign in
Sign up
Sign in
username, password
storing the username and password in database
username, password
If password of the given username is incorrect
checking the database if the username exists
If the given credentials are correct
logged in True
Notes Settings Profile Logout
If Notes
View Update
if users chooses to view
accessing notes from database and showing it to the user
If user choses to update
taking input
Updating the existing record in the database with the new one
If Logout
logged in false
If Profile
Shows User information
If Settings
Update Password or Email
Update Email
taking input
updating the email in the database
Update Password
taking input
updating the password in the database

Chapter-3: Program

Main-File

```python
from flask import *
from flask_mysqldb import *

#custom module
from mysqlcreate import accounttable
from mysqlcreate import notetittletable
from mysqlcreate import notetable
from gravatar import avatar

#creating tables and running it
accounttable()
notetittletable()
notetable()

app = Flask(__name__)

app.secret_key = 'Ar7'

#mysql database config
app.config['MYSQL_HOST'] = 'sql6.freesqldatabase.com'
app.config['MYSQL_USER'] = 'sql6447629'
app.config['MYSQL_PASSWORD'] = 'MHS4PnTuiJ'
app.config['MYSQL_DB'] = 'sql6447629'

mysql = MySQL(app) #Hello

@app.route('/')
def main():
    username = request.cookies.get('username')
    cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
    cursor.execute('SELECT * FROM account WHERE username = %s', (username, ))
    account = cursor.fetchone()
    if account:
        session['loggedin'] = True
        session['username'] = account['username']
        resume = make_response(redirect('/home'))
        return resume
    return render_template('home.html')

# Basic Sign In Sign Up Sign Out System

@app.route('/signin', methods = ['GET', 'POST'])
def login():
    msg = ''
    if request.method == 'POST' and 'username' in request.form and 'password' in request.form:
        username = request.form['username']
        password = request.form['password']
        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute('SELECT * FROM account WHERE username = %s and password = %s', (username, password, ))
        account = cursor.fetchone()
        if account:
            session['loggedin'] = True
            session['username'] = account['username']
            msg = 'Logged In Sucessfully'
            res = make_response(redirect('/home'))
            res.set_cookie('username',f'{username}',max_age=60*60*24*30)
            res.set_cookie('password',f'{password}',max_age=60*60*24*30)
            return res
        else:
            msg = 'Incorrect username/password'
            return render_template('signin.html', msg=msg)
    return render_template('signin.html')
```

```python
@app.route('/signup', methods = ['POST', 'GET'])
def signup():
    msg = ''
    color = ''
    if request.method == 'POST' and 'username' in request.form and 'email' in request.form and 'password' in request.form and 'cnfrmpassword' in request.form:
        username = request.form['username']
        email_id = request.form['email']
        password = request.form['password']
        note1tittle = 'Empty Note. Edit to add something'
        note2tittle = 'Empty Note. Edit to add something'
        note3tittle = 'Empty Note. Edit to add something'
        note4tittle = 'Empty Note. Edit to add something'
        note5tittle = 'Empty Note. Edit to add something'
        note1 = 'Empty Note. Edit to add something'
        note2 = 'Empty Note. Edit to add something'
        note3 = 'Empty Note. Edit to add something'
        note4 = 'Empty Note. Edit to add something'
        note5 = 'Empty Note. Edit to add something'
        cnfrmpassword = request.form['cnfrmpassword']
        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute('SELECT * FROM account WHERE username = %s', (username, ))
        account = cursor.fetchone()
        if account:
            msg = 'Account already exists'
            color = 'red-text'
        else:
            if password == cnfrmpassword:
                cursor.execute('INSERT INTO account VALUES (%s, %s, %s)', (username, email_id, password))
                cursor.execute('INSERT INTO notetittle VALUES (%s, %s, %s, %s, %s, %s)',
                    (username, note1tittle, note2tittle, note3tittle, note4tittle, note5tittle))
                cursor.execute('INSERT INTO notes VALUES (%s, %s, %s, %s, %s, %s)', (username, note1, note2, note3, note4, note5))
                cursor.connection.commit()
                msg = 'Account created sucessfully'
                color = 'green-text'
                return render_template('home.html', msg=msg, color=color)
            else:
                color = 'red-text'
                msg = 'You have either not confirmed the password or you might have entered wrong password'
                return render_template('signup.html', color=color, msg=msg)
    return render_template('signup.html', msg=msg, color=color)
```

```python
@app.route('/signout')
def singout():
    session.pop('loggedin', None)
    session.pop('username', None)
    res = make_response(redirect('/'))
    res.delete_cookie('username')
    res.delete_cookie('password')
    return res

#sign functions end here

# Functional Features Start here

@app.route('/home')
def home():
    if 'loggedin' in session:
        username = session['username']
        notecur = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        notecur.execute('SELECT * FROM notes WHERE username = %s', (username, ))
        notes = notecur.fetchone()
        notecur.execute('SELECT * FROM notetittle WHERE username = %s', (username, ))
        tittle = notecur.fetchone()
        return render_template("todo.html", notes=notes, tittle=tittle)
    else:
        return redirect('/')
```

```python
@app.route('/update1', methods = ['GET', 'POST'])
def updatenote1():
    notenum = '1'
    if 'loggedin' in session:
        if request.method == 'POST' and 'noteupdate' in request.form and 'notetittleupdate' in request.form:
            notetittle = request.form['notetittleupdate']
            noteupdate = request.form['noteupdate']
            username = session['username']
            cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
            cursor.execute('SELECT * FROM notes WHERE username = %s', (username, ))
            cursor.execute('UPDATE notetittle SET note1tittle = %s WHERE username = %s', (notetittle, username))
            cursor.execute('UPDATE notes SET note1 = %s WHERE username = %s', (noteupdate, username))
            mysql.connection.commit()
            return redirect('/home')
        return render_template('update1.html', notenum=notenum)
    return redirect('/')

@app.route('/update2', methods = ['GET', 'POST'])
def updatenote2():
    notenum = '2'
    if 'loggedin' in session:
        if request.method == 'POST' and 'noteupdate' in request.form and 'notetittleupdate' in request.form:
            notetittle = request.form['notetittleupdate']
            noteupdate = request.form['noteupdate']
            username = session['username']
            cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
            cursor.execute('SELECT * FROM notes WHERE username = %s', (username, ))
            cursor.execute('UPDATE notetittle SET note2tittle = %s WHERE username = %s', (notetittle, username))
            cursor.execute('UPDATE notes SET note2 = %s WHERE username = %s', (noteupdate, username))
            mysql.connection.commit()
            return redirect('/home')
        return render_template('update2.html', notenum=notenum)
    return redirect('/')
```

```
@app.route('/update3', methods = ['GET', 'POST'])
def updatenote3():
    notenum = '3'
    if 'loggedin' in session:
        if request.method == 'POST' and 'noteupdate' in request.form and 'notetittleupdate' in request.form:
            notetittle = request.form['notetittleupdate']
            noteupdate = request.form['noteupdate']
            username = session['username']
            cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
            cursor.execute('SELECT * FROM notes WHERE username = %s', (username, ))
            cursor.execute('UPDATE notetittle SET note3tittle = %s WHERE username = %s', (notetittle, username))
            cursor.execute('UPDATE notes SET note3 = %s WHERE username = %s', (noteupdate, username))
            mysql.connection.commit()
            return redirect('/home')
        return render_template('update3.html', notenum=notenum)
    return redirect('/')

@app.route('/update4', methods = ['GET', 'POST'])
def updatenote4():
    notenum = '4'
    if 'loggedin' in session:
        if request.method == 'POST' and 'noteupdate' in request.form and 'notetittleupdate' in request.form:
            notetittle = request.form['notetittleupdate']
            noteupdate = request.form['noteupdate']
            username = session['username']
            cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
            cursor.execute('SELECT * FROM notes WHERE username = %s', (username, ))
            cursor.execute('UPDATE notetittle SET note4tittle = %s WHERE username = %s', (notetittle, username))
            cursor.execute('UPDATE notes SET note4 = %s WHERE username = %s', (noteupdate, username))
            mysql.connection.commit()
            return redirect('/home')
        return render_template('update4.html', notenum=notenum)
    return redirect('/')

@app.route('/update5', methods = ['GET', 'POST'])
def updatenote5():
    notenum = '5'
    if 'loggedin' in session:
        if request.method == 'POST' and 'noteupdate' in request.form and 'notetittleupdate' in request.form:
            notetittle = request.form['notetittleupdate']
            noteupdate = request.form['noteupdate']
            username = session['username']
            cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
            cursor.execute('SELECT * FROM notes WHERE username = %s', (username, ))
            cursor.execute('UPDATE notetittle SET note5tittle = %s WHERE username = %s', (notetittle, username))
            cursor.execute('UPDATE notes SET note5 = %s WHERE username = %s', (noteupdate, username))
            mysql.connection.commit()
            return redirect('/home')
        return render_template('update5.html', notenum=notenum)
    return redirect('/')

@app.route('/profile')
def profile():
    if 'loggedin' in session:
        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute('SELECT * FROM account WHERE username = %s', (session['username'], ))
        account = cursor.fetchone()
        pfp = avatar(account['email_id'])
        return render_template('profile.html', account=account, pfp=pfp)
    return redirect('/home')


@app.route('/settings', methods = ['GET', 'POST'])
def settings():
    if 'loggedin' in session:
        msg = ''
        color = ''
        if request.method == 'POST' and 'email_up' in request.form and 'cn_pass' in request.form:
            email_id = request.form['email_up']
            cnfrmpassword = request.form['cn_pass']
            username = session['username']
            cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
            cursor.execute('SELECT * FROM account WHERE username = %s', (username, ))
            account = cursor.fetchone()
            if cnfrmpassword == account['password']:
                cursor.execute('UPDATE account SET email_id = %s WHERE username = %s', (email_id, username, ))
                msg = 'Email-id Updated'
                color = 'green-text'
                mysql.connection.commit()
                return render_template('settings.html', msg=msg, color=color)
            else:
                msg = 'Incorrect Password Entered'
                color = 'red-text'
                return render_template('settings.html', msg=msg, color=color)
        elif request.method == 'POST' and 'c_pass' in request.form and 'pass_up' in request.form and 'n_pass' in request.form:
            username = session['username']
            new_password = request.form['n_pass']
            old_password = request.form['c_pass']
            password = request.form['pass_up']
            cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
            cursor.execute('SELECT * FROM account WHERE username = %s', (username, ))
            account = cursor.fetchone()
            if old_password == account['password']:
                if password == new_password:
                    cursor.execute('UPDATE account SET password = %s WHERE username = %s', (password, username, ))
                    mysql.connection.commit()
                    msg = 'Password Updated'
                    color = 'green-text'
                    return render_template('settings.html', color2=color, msg2=msg)
```

```
                else:
                    msg = 'Your new password and conformation password dose not match retry again'
                    color = 'red-text'
                    return render_template('setting.html', msg2=msg, color2=color)
            else:
                msg = 'Your old password dose not match retry again'
                color = 'red-text'
                return render_template('settings.html', msg2=msg, color2=color)
        return render_template('settings.html')
    return redirect('/')

if __name__ == '__main__':
    app.run(threaded=True, port=5000)
```

MySQL-module

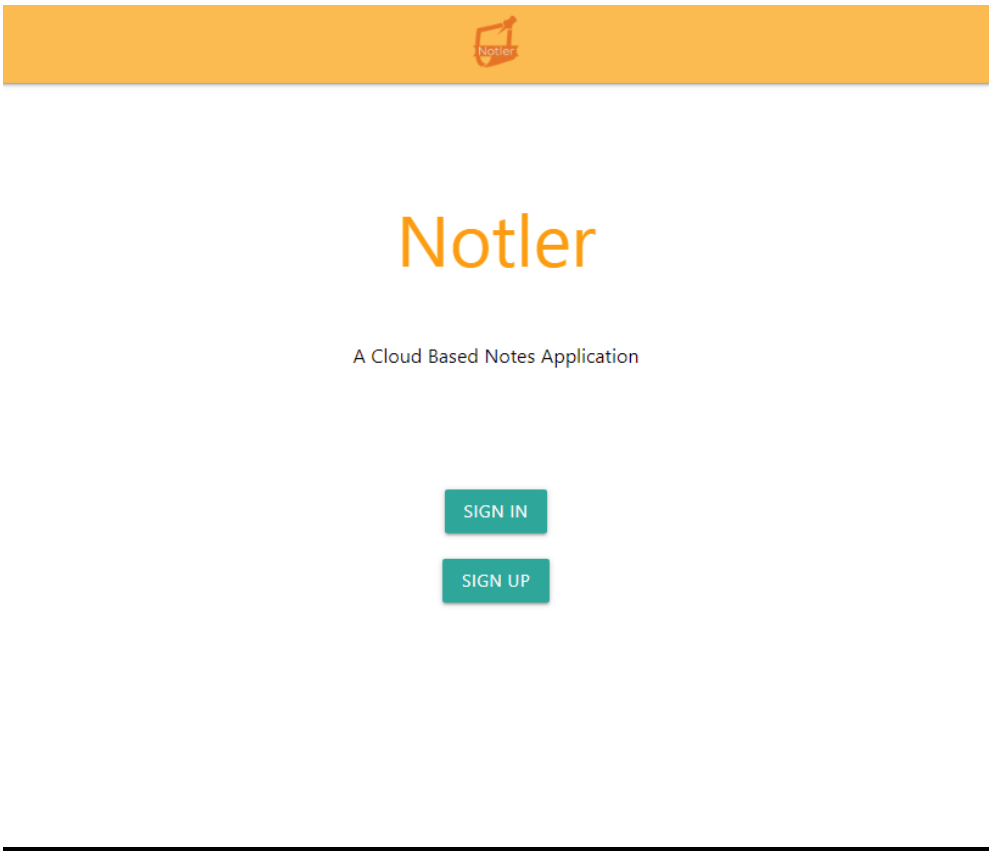
```
import mysql.connector

def accounttable():
    db = mysql.connector.connect(
        host='localhost',
        user='root',
        passwd='ssassk@2004',
        database='notler')

    mycursor = db.cursor()

    mycursor.execute("CREATE TABLE IF NOT EXISTS account(username VARCHAR(100) PRIMARY KEY, email_id VARCHAR(300), password VARCHAR(300))")

def notetable():
    db = mysql.connector.connect(
    host='localhost',
    user='root',
    passwd='ssassk@2004',
    database='notler')

    mycursor = db.cursor()

    mycursor.execute("CREATE TABLE IF NOT EXISTS notes(username VARCHAR(100) PRIMARY KEY, note1 VARCHAR(1000), note2 VARCHAR(1000),\
                note3 VARCHAR(1000), note4 VARCHAR(1000), note5 VARCHAR(1000))")

```

ProfilePicture-module

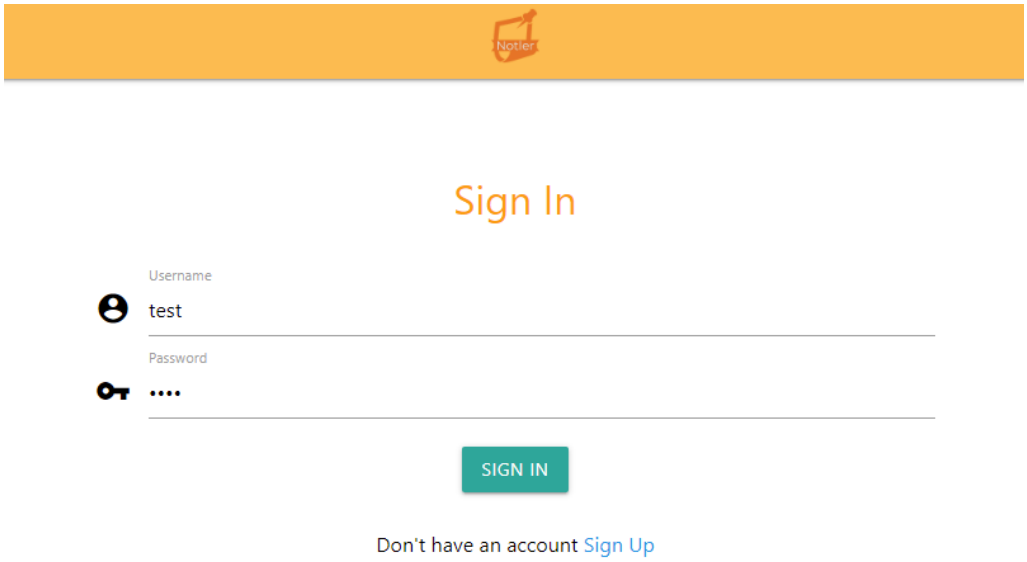
```
import urllib, hashlib

def avatar(usermail):

    #hashing the url
    hash = hashlib.md5(usermail.encode('utf-8')).hexdigest()

    #output image
    gravatar_image = "https://www.gravatar.com/avatar/" + hash + "?"

    return gravatar_image
```

Chapter-4: Test Case

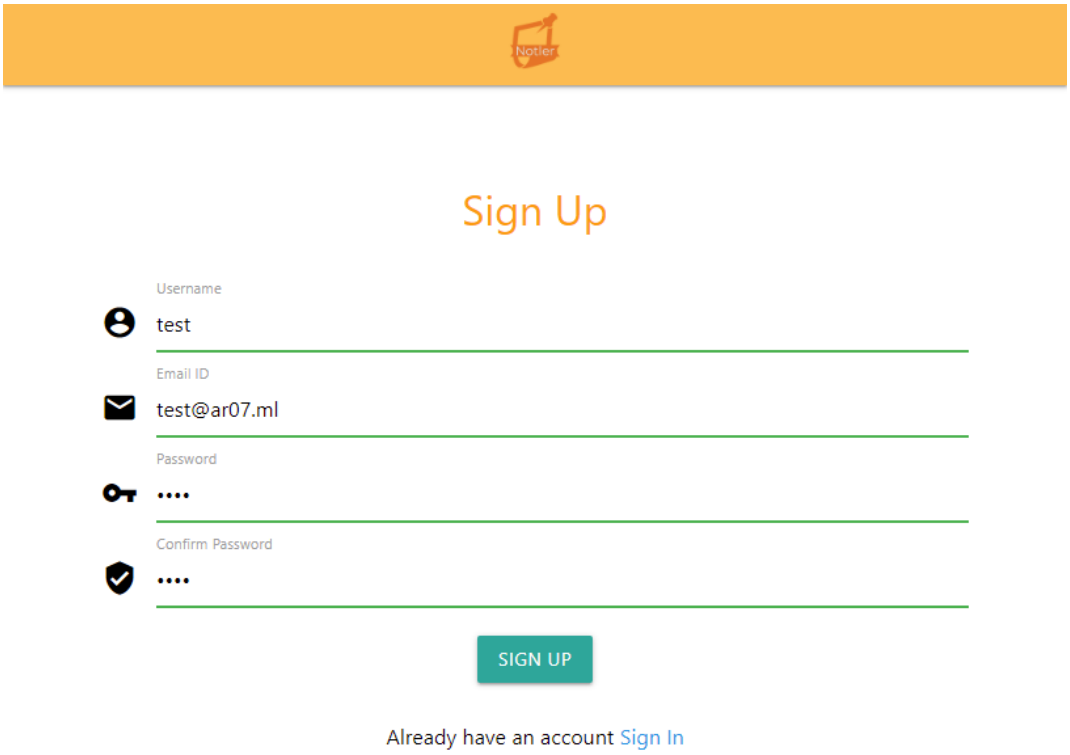
1. Home Page:



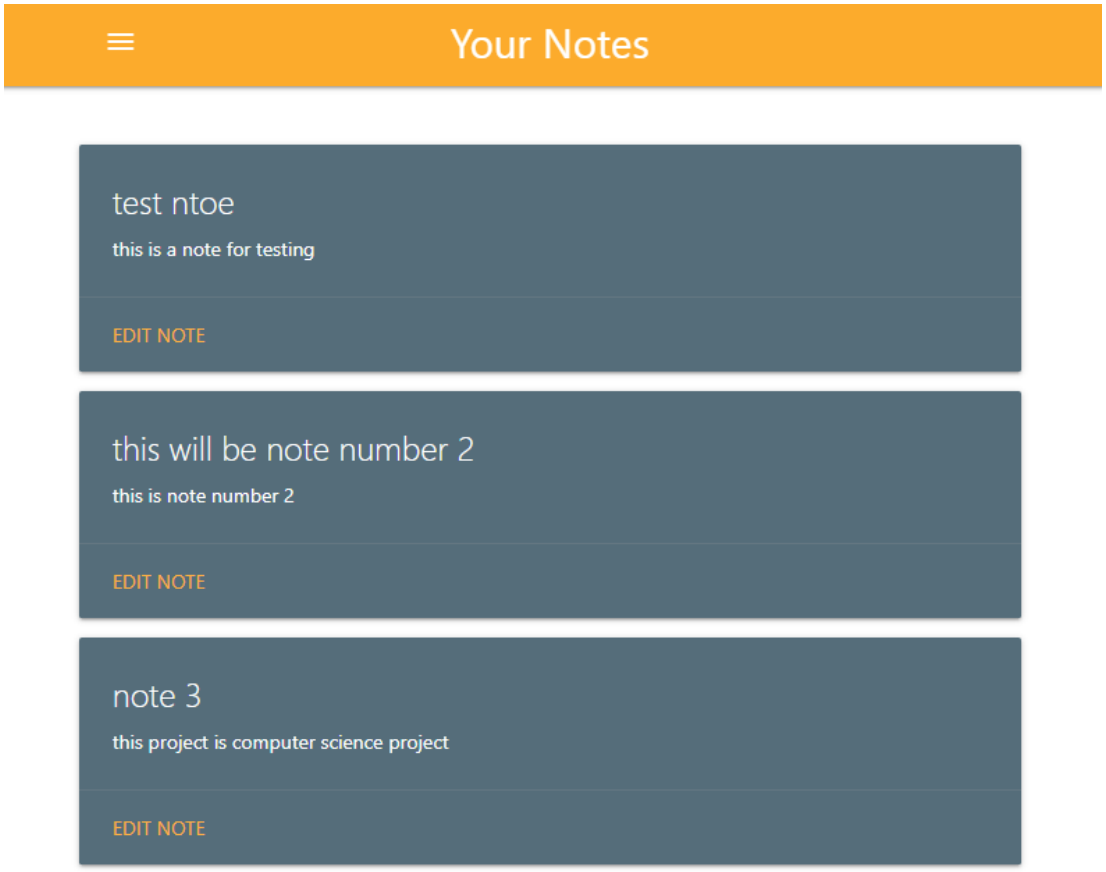
2. Sign-in:



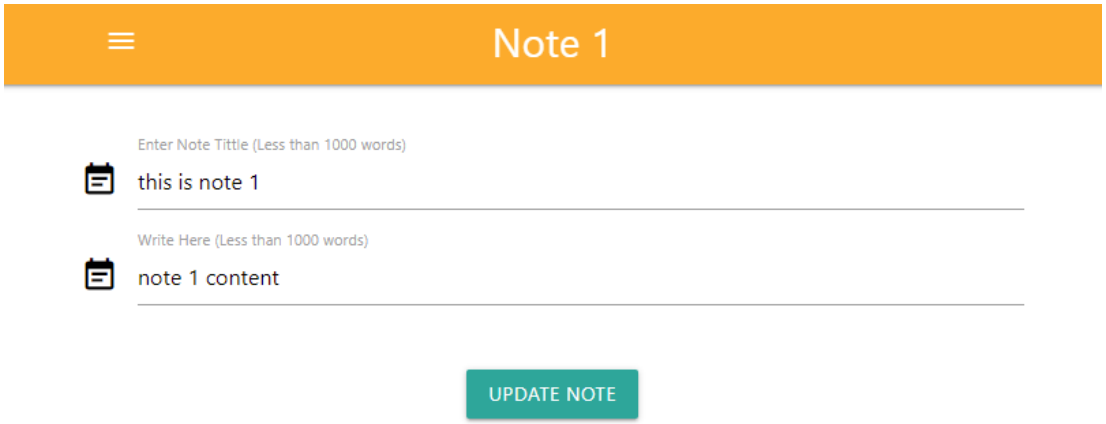
3. Sign-up:



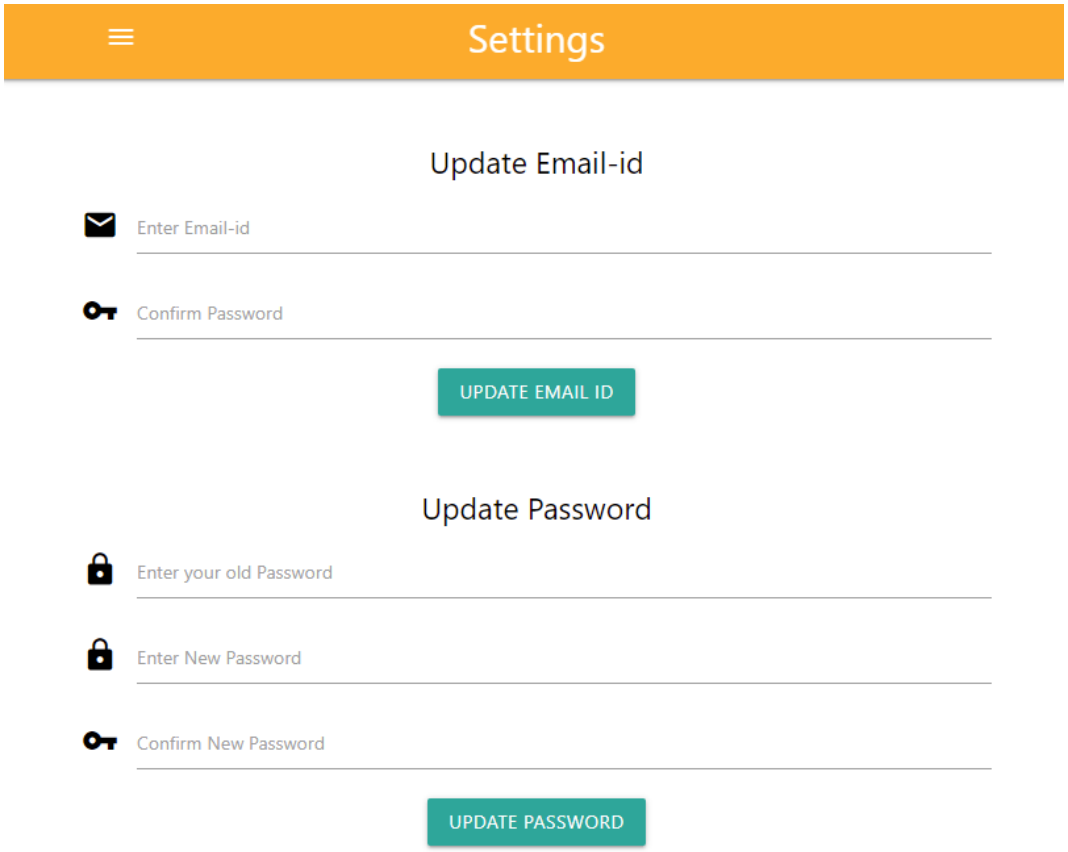
4. Notes-Page:



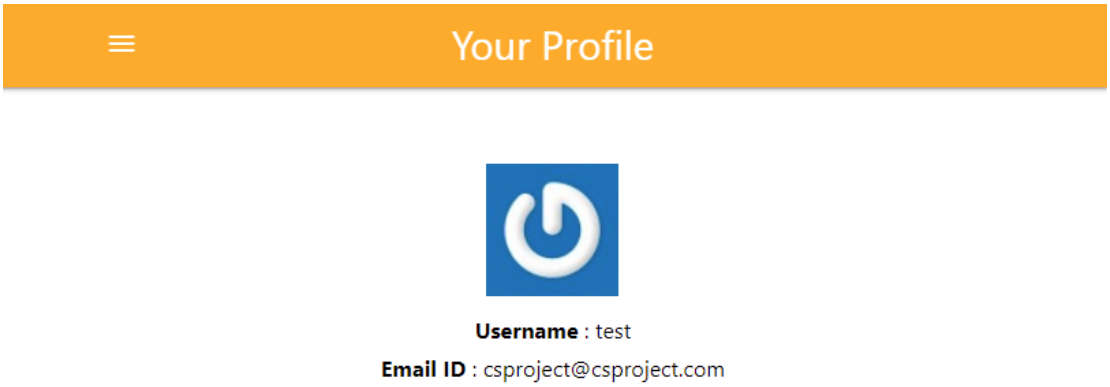
5. Update-Note:



6. Settings-Page:



7. Profile-Page:



Chapter-5: Conclusion

Through the project we've learned to use Python and MySQL and it has shown as a wide variety of projects that can be made using the combination of these two programs

Bibliography:

1. GitHub
2. Stack overflow
3. W3school
4. Geeks for Geeks