

# Notler

A Notes Application

Made by

Aravinth .S.S

Shvuayu Sarkar

# Index

Sr.No	Tittle	Page
	Acknowledgement	3
	<b>Chapter 1: Introduction</b>	
1	Preface	4
2	Abstract	5
3	Project Overview	6
4	Administrative and User Modules	7
5	Features	8
6	Objective	9
	<b>Chapter 2: Requirement and specification</b>	
1	User Interface	10
2	Database	11
3	Functions and Modules used and Functions and Modules Created	12
4	Algorithm	13
	<b>Chapter 3: Program</b>	14
	<b>Chapter 4: Test Cases</b>	18
	<b>Chapter 5: Conclusion</b>	22
	<b>Bibliography</b>	22

# **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 a lot of 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.

## 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

## 5. Features

The features of the program are as follows:

- Cloud synchronization
- Updating existing note
- Available in almost every device
- Automatic login
- Profile Photo System



## 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

## **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 is a high level interpreter language and an object oriented programming 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 programming 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:

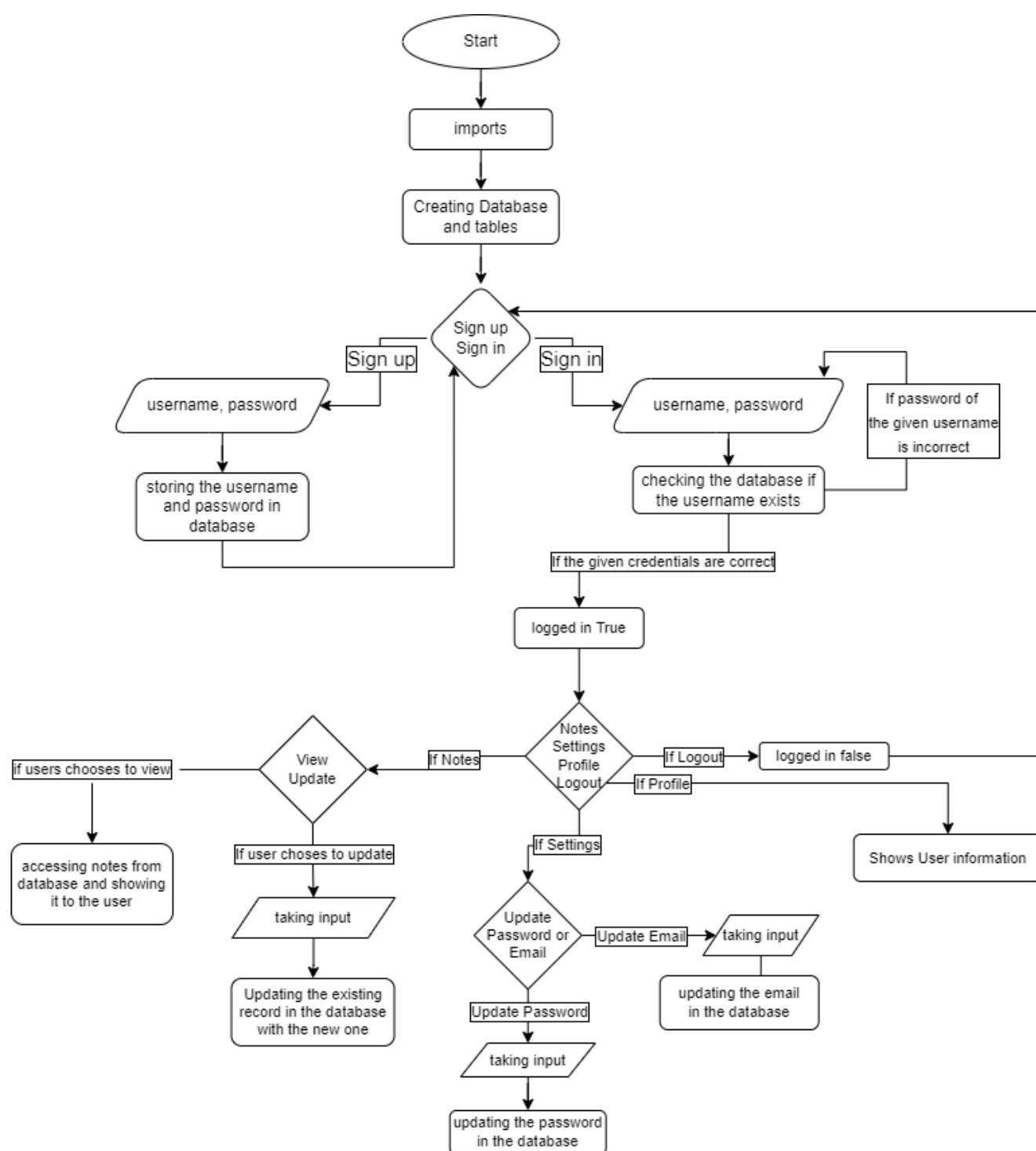
1. flask
2. flask\_mysqlldb
3. mysql.connector
4. urllib
5. hashlib

### 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



# Chapter-3: Program

## Main-File

```

1  |from flask import *
2  |from flask_mysql import *
3
4  #custom module
5  from mysqlcreate import accounttable
6  from mysqlcreate import notetittletable
7  from mysqlcreate import notetable
8  from gravatar import avatar
9
10 #creating tables and running it
11 accounttable()
12 notetittletable()
13 notetable()
14
15 app = Flask(__name__)
16
17 app.secret_key = 'Ar7'
18
19 #mysql database config
20 app.config['MYSQL_HOST'] = 'sql6.freesqldatabase.com'
21 app.config['MYSQL_USER'] = 'sql6447629'
22 app.config['MYSQL_PASSWORD'] = 'MHS4PnTuiJ'
23 app.config['MYSQL_DB'] = 'sql6447629'
24
25 mysql = MySQL(app) #Hello
26
27 @app.route('/')
28 def main():
29     username = request.cookies.get('username')
30     cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
31     cursor.execute('SELECT * FROM account WHERE username = %s', (username, ))
32     account = cursor.fetchone()
33     if account:
34         session['loggedin'] = True
35         session['username'] = account['username']
36         resume = make_response(redirect('/home'))
37         return resume
38     return render_template('home.html')
39
40 # Basic Sign In Sign Up Sign Out System
41
42 @app.route('/signin', methods = ['GET', 'POST'])
43 def login():
44     msg = ''
45     if request.method == 'POST' and 'username' in request.form and 'password' in request.form:
46         username = request.form['username']
47         password = request.form['password']
48         cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
49         cursor.execute('SELECT * FROM account WHERE username = %s and password = %s', (username, password, ))
50         account = cursor.fetchone()
51         if account:
52             session['loggedin'] = True
53             session['username'] = account['username']
54             msg = 'Logged In Successfully'
55             res = make_response(redirect('/home'))
56             res.set_cookie('username', f'{username}', max_age=60*60*24*30)
57             res.set_cookie('password', f'{password}', max_age=60*60*24*30)
58             return res
59         else:
60             msg = 'Incorrect username/password'
61             return render_template('signin.html', msg=msg)
62     return render_template('signin.html')
63

```

```

64 @app.route('/signup', methods = ['POST', 'GET'])
65 def signup():
66     msg = ''
67     color = ''
68     if request.method == 'POST' and 'username' in request.form and 'email' in request.form and 'password' in request.form and 'cnfrmpassword' in request.form:
69         username = request.form['username']
70         email_id = request.form['email']
71         password = request.form['password']
72         note1title = 'Empty Note. Edit to add something'
73         note2title = 'Empty Note. Edit to add something'
74         note3title = 'Empty Note. Edit to add something'
75         note4title = 'Empty Note. Edit to add something'
76         note5title = 'Empty Note. Edit to add something'
77         note1 = 'Empty Note. Edit to add something'
78         note2 = 'Empty Note. Edit to add something'
79         note3 = 'Empty Note. Edit to add something'
80         note4 = 'Empty Note. Edit to add something'
81         note5 = 'Empty Note. Edit to add something'
82         cnfrmpassword = request.form['cnfrmpassword']
83         cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
84         cursor.execute('SELECT * FROM account WHERE username = %s', (username, ))
85         account = cursor.fetchone()
86         if account:
87             msg = 'Account already exists'
88             color = 'red-text'
89         else:
90             if password == cnfrmpassword:
91                 cursor.execute('INSERT INTO account VALUES (%s, %s, %s)', (username, email_id, password))
92                 cursor.execute('INSERT INTO notetitle VALUES (%s, %s, %s, %s, %s, %s)',
93                             (username, note1title, note2title, note3title, note4title, note5title))
94                 cursor.execute('INSERT INTO notes VALUES (%s, %s, %s, %s, %s, %s)', (username, note1, note2, note3, note4, note5))
95                 cursor.connection.commit()
96                 msg = 'Account created successfully'
97                 color = 'green-text'
98                 return render_template('home.html', msg=msg, color=color)
99             else:
100                 color = 'red-text'
101                 msg = 'You have either not confirmed the password or you might have entered wrong password'
102                 return render_template('signup.html', color=color, msg=msg)
103     return render_template('signup.html', msg=msg, color=color)
104
105 @app.route('/signinout')
106 def singout():
107     session.pop('loggedin', None)
108     session.pop('username', None)
109     res = make_response(redirect('/'))
110     res.delete_cookie('username')
111     res.delete_cookie('password')
112     return res
113
114 #sign functions end here
115
116 # Functional Features Start here
117
118 @app.route('/home')
119 def home():
120     if 'loggedin' in session:
121         username = session['username']
122         notecur = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
123         notecur.execute('SELECT * FROM notes WHERE username = %s', (username, ))
124         notes = notecur.fetchone()
125         notecur.execute('SELECT * FROM notetitle WHERE username = %s', (username, ))
126         title = notecur.fetchone()
127         return render_template("todo.html", notes=notes, title=title)
128     else:
129         return redirect('/')
130
131 @app.route('/update1', methods = ['GET', 'POST'])
132 def updatenote1():
133     notenum = '1'
134     if 'loggedin' in session:
135         if request.method == 'POST' and 'noteupdate' in request.form and 'notetitleupdate' in request.form:
136             notetitle = request.form['notetitleupdate']
137             noteupdate = request.form['noteupdate']
138             username = session['username']
139             cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
140             cursor.execute('SELECT * FROM notes WHERE username = %s', (username, ))
141             cursor.execute('UPDATE notetitle SET notetitle = %s WHERE username = %s', (notetitle, username))
142             cursor.execute('UPDATE notes SET note1 = %s WHERE username = %s', (noteupdate, username))
143             mysql.connection.commit()
144             return redirect('/home')
145         return render_template('update1.html', notenum=notenum)
146     return redirect('/')
147
148 @app.route('/update2', methods = ['GET', 'POST'])
149 def updatenote2():
150     notenum = '2'
151     if 'loggedin' in session:
152         if request.method == 'POST' and 'noteupdate' in request.form and 'notetitleupdate' in request.form:
153             notetitle = request.form['notetitleupdate']
154             noteupdate = request.form['noteupdate']
155             username = session['username']
156             cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
157             cursor.execute('SELECT * FROM notes WHERE username = %s', (username, ))
158             cursor.execute('UPDATE notetitle SET notetitle = %s WHERE username = %s', (notetitle, username))
159             cursor.execute('UPDATE notes SET note2 = %s WHERE username = %s', (noteupdate, username))
160             mysql.connection.commit()
161             return redirect('/home')
162         return render_template('update2.html', notenum=notenum)
163     return redirect('/')

```

```

164 @app.route('/update3', methods = ['GET', 'POST'])
165 def updatenote3():
166     notenum = '3'
167     if 'loggedin' in session:
168         if request.method == 'POST' and 'noteupdate' in request.form and 'notetitleupdate' in request.form:
169             notetitle = request.form['notetitleupdate']
170             noteupdate = request.form['noteupdate']
171             username = session['username']
172             cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
173             cursor.execute('SELECT * FROM notes WHERE username = %s', (username, ))
174             cursor.execute('UPDATE notetitle SET note3title = %s WHERE username = %s', (notetitle, username))
175             cursor.execute('UPDATE notes SET note3 = %s WHERE username = %s', (noteupdate, username))
176             mysql.connection.commit()
177             return redirect('/home')
178         return render_template('update3.html', notenum=notenum)
179     return redirect('/')
180
181 @app.route('/update4', methods = ['GET', 'POST'])
182 def updatenote4():
183     notenum = '4'
184     if 'loggedin' in session:
185         if request.method == 'POST' and 'noteupdate' in request.form and 'notetitleupdate' in request.form:
186             notetitle = request.form['notetitleupdate']
187             noteupdate = request.form['noteupdate']
188             username = session['username']
189             cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
190             cursor.execute('SELECT * FROM notes WHERE username = %s', (username, ))
191             cursor.execute('UPDATE notetitle SET note4title = %s WHERE username = %s', (notetitle, username))
192             cursor.execute('UPDATE notes SET note4 = %s WHERE username = %s', (noteupdate, username))
193             mysql.connection.commit()
194             return redirect('/home')
195         return render_template('update4.html', notenum=notenum)
196     return redirect('/')
197
198 @app.route('/update5', methods = ['GET', 'POST'])
199 def updatenote5():
200     notenum = '5'
201     if 'loggedin' in session:
202         if request.method == 'POST' and 'noteupdate' in request.form and 'notetitleupdate' in request.form:
203             notetitle = request.form['notetitleupdate']
204             noteupdate = request.form['noteupdate']
205             username = session['username']
206             cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
207             cursor.execute('SELECT * FROM notes WHERE username = %s', (username, ))
208             cursor.execute('UPDATE notetitle SET note5title = %s WHERE username = %s', (notetitle, username))
209             cursor.execute('UPDATE notes SET note5 = %s WHERE username = %s', (noteupdate, username))
210             mysql.connection.commit()
211             return redirect('/home')
212         return render_template('update5.html', notenum=notenum)
213     return redirect('/')
214
215 @app.route('/profile')
216 def profile():
217     if 'loggedin' in session:
218         cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
219         cursor.execute('SELECT * FROM account WHERE username = %s', (session['username'], ))
220         account = cursor.fetchone()
221         pfp = avatar(account['email_id'])
222         return render_template('profile.html', account=account, pfp=pfp)
223     return redirect('/home')
224
225 @app.route('/settings', methods = ['GET', 'POST'])
226 def settings():
227     if 'loggedin' in session:
228         msg = ''
229         color = ''
230         if request.method == 'POST' and 'email_up' in request.form and 'cn_pass' in request.form:
231             email_id = request.form['email_up']
232             cnfrmpassword = request.form['cn_pass']
233             username = session['username']
234             cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
235             cursor.execute('SELECT * FROM account WHERE username = %s', (username, ))
236             account = cursor.fetchone()
237             if cnfrmpassword == account['password']:
238                 cursor.execute('UPDATE account SET email_id = %s WHERE username = %s', (email_id, username, ))
239                 msg = 'Email-id Updated'
240                 color = 'green-text'
241                 mysql.connection.commit()
242                 return render_template('settings.html', msg=msg, color=color)
243             else:
244                 msg = 'Incorrect Password Entered'
245                 color = 'red-text'
246                 return render_template('settings.html', msg=msg, color=color)
247         elif request.method == 'POST' and 'c_pass' in request.form and 'pass_up' in request.form and 'n_pass' in request.form:
248             username = session['username']
249             new_password = request.form['n_pass']
250             old_password = request.form['c_pass']
251             password = request.form['pass_up']
252             cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
253             cursor.execute('SELECT * FROM account WHERE username = %s', (username, ))
254             account = cursor.fetchone()
255             if old_password == account['password']:
256                 if password == new_password:
257                     cursor.execute('UPDATE account SET password = %s WHERE username = %s', (password, username, ))
258                     mysql.connection.commit()
259                     msg = 'Password Updated'
260                     color = 'green-text'
261                     return render_template('settings.html', color2=color, msg2=msg)

```



```

262         else:
263             msg = 'Your new password and conformation password dose not match retry again'
264             color = 'red-text'
265             return render_template('setting.html', msg2=msg, color2=color)
266         else:
267             msg = 'Your old password dose not match retry again'
268             color = 'red-text'
269             return render_template('settings.html', msg2=msg, color2=color)
270     return render_template('settings.html')
271     return redirect('/')
272
273 if __name__ == '__main__':
274     app.run(threaded=True, port=5000)

```

## MySQL-module

```

1  import mysql.connector
2
3  def accountable():
4      db = mysql.connector.connect(
5          host='localhost',
6          user='root',
7          passwd='ssassk@2004',
8          database='notler')
9
10     mycursor = db.cursor()
11
12     mycursor.execute("CREATE TABLE IF NOT EXISTS account(username VARCHAR(100) PRIMARY KEY, email_id VARCHAR(300), password VARCHAR(300))")
13
14     def notetable():
15         db = mysql.connector.connect(
16             host='localhost',
17             user='root',
18             passwd='ssassk@2004',
19             database='notler')
20
21         mycursor = db.cursor()
22
23         mycursor.execute("CREATE TABLE IF NOT EXISTS notes(username VARCHAR(100) PRIMARY KEY, note1 VARCHAR(1000), note2 VARCHAR(1000),\
24             note3 VARCHAR(1000), note4 VARCHAR(1000), note5 VARCHAR(1000))")
25

```

## ProfilePicture-module

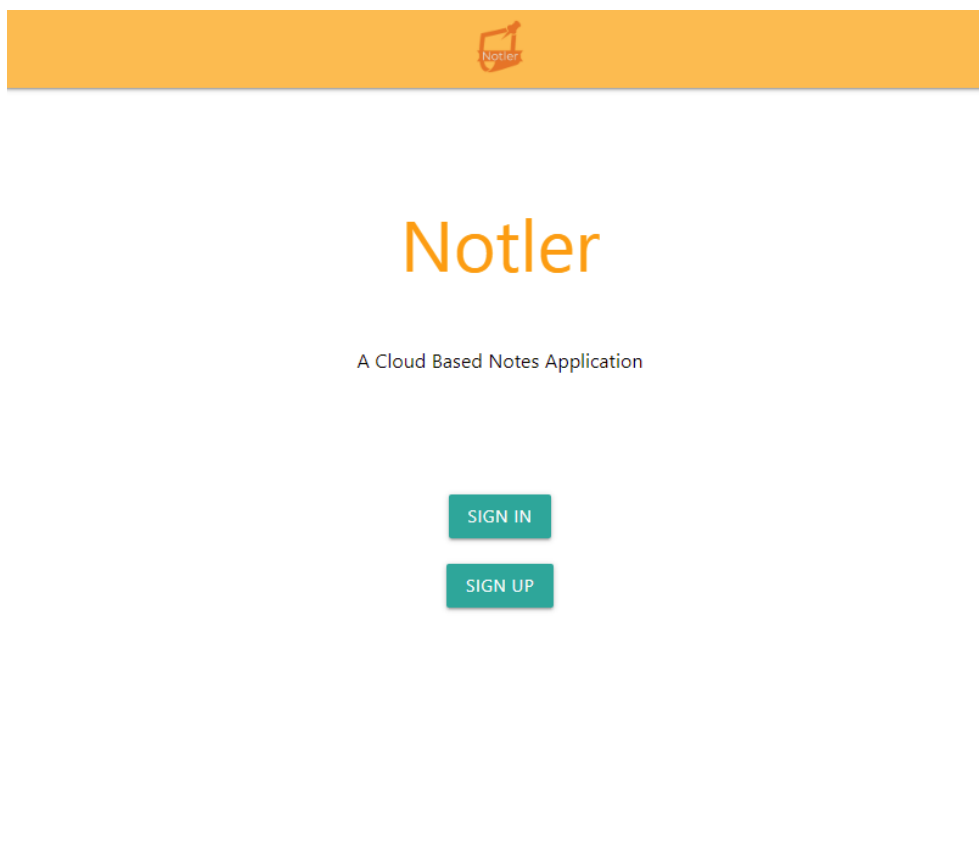
```

1  import urllib, hashlib
2
3  def avatar(usermail):
4
5      #hashing the url
6      hash = hashlib.md5(usermail.encode('utf-8')).hexdigest()
7
8      #output image
9      gravatar_image = "https://www.gravatar.com/avatar/" + hash + "?"
10
11     return gravatar_image

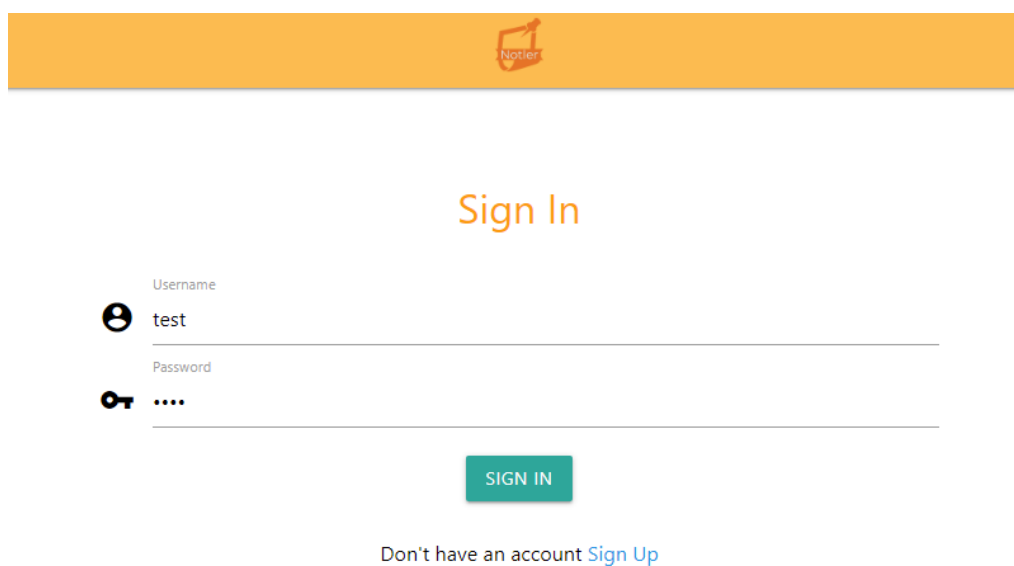
```

## Chapter-4: Test Case


### 1. Home Page:




### 2. Sign-in:



### 3. Sign-up:




## Sign Up




Username

test




Email ID

test@ar07.ml



Password

....




Confirm Password

....

SIGN UP

Already have an account [Sign In](#)

### 4. Notes-Page:

Your Notes

test ntoe

this is a note for testing

EDIT NOTE

this will be note number 2

this is note number 2


EDIT NOTE


note 3


this project is computer science project

EDIT NOTE

## 5. Update-Note:


Note 1

Enter Note Title (Less than 1000 words)  
this is note 1


Write Here (Less than 1000 words)  
note 1 content


UPDATE NOTE

## 6. Settings-Page:

Settings


Update Email-id


Enter Email-id


Confirm Password

UPDATE EMAIL ID

Update Password

Enter your old Password

Enter New Password

Confirm New Password

UPDATE PASSWORD

## 7. Profile-Page:



**Username :** test

**Email ID :** csproject@csproject.com

## **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