

Creating APIs in Flask

Team ID	PNT2022TMID35229
Project Name	CONTAINMENT ZONE ALERTING

```
app.py 9+ X
D:\> PRASHANTH > Projects > Final Deliverables > Flask_Application > app.py > ...
78
79 @app.route('/', methods=['GET', 'POST'])
80 def register():
81     message = ''
82     if request.method == 'POST':
83         # get the data from the form
84         name = request.form['username']
85         email = request.form['email']
86         password = request.form['password']
87         confirm_password = request.form['confirm_password']
88         # if nothing is entered in the form
89         if not name or not email or not password or not confirm_password:
90             message = 'Please fill all the fields!'
91             return render_template('register.html', message=message)
92         # if the password and confirm password do not match
93         elif password != confirm_password:
94             message = 'Passwords do not match!'
95             return render_template('register.html', message=message)
96
97         # password length must be 8 or above
98         if len(password) < 8:
99             message = 'Password must be 8 or more characters'
100             return render_template('register.html', message=message)
101         # check if the email is valid
102         if re.match(r"^[^@]+@[^@]+\.[^@]+$ ", email):
103             # insert the data into the database
104             # check if email already exists in the database
105             sql = "SELECT * FROM users WHERE email = '" + email + "'"
106             stmt = ibm_db.exec_immediate(conn, sql)
107             # print("stmt", stmt)
108             result = ibm_db.fetch_assoc(stmt)
109             # print("result", result)
110             if result:
111                 message = 'The username or email already exists!'
112             else:
113                 sql = "INSERT INTO users (id, username, email, password,type) VALUES (seq_person.nextval,'" + name + \
114                     "' , '" + email + "', '" + password + "', 1) "
115                 ibm_db.exec_immediate(conn, sql)
116                 # send confirmation email
117                 send_conf_email(email)
118                 return redirect(url_for('login'))
119             else:
120                 message = 'The email is invalid!'
121             return render_template('register.html', message=message)
122
123
124 @app.route('/login', methods=['GET', 'POST'])
125 def login():
126     message = ''
127     if request.method == 'POST':
128         # get the data from the form
129         email = request.form['email']
130         password = request.form['password']
131         # if nothing is entered in the form
132         if not email or not password:
133             message = 'Please fill all the fields!'
134             return render_template('login.html', message=message)
135         # check if the username and password are valid
136         sql = "SELECT * FROM users WHERE email = '" + email + "' AND password = '" + password + "'"
137         stmt = ibm_db.exec_immediate(conn, sql)
138         result = ibm_db.fetch_assoc(stmt)
139         # print("result", result)
140         if result:
141             # message = 'You have successfully logged in!'

```

```

app.py 9+ X
D: > PRASHANTH > Projects > Final Deliverables > Flask_Application > app.py > ...
142     session['id'] = result['ID']
143     session['username'] = result['USERNAME']
144     session['email'] = result['EMAIL']
145     # print("id ==", session['id'])
146     return redirect(url_for('home'))
147 else:
148     message = 'The email or password is incorrect!'
149     return render_template('login.html', message=message)
150
151
152 @app.route('/logout')
153 def logout():
154     session.clear()
155     return redirect(url_for('login'))
156
157
158 # create a route for the home page and open only if the user is logged in
159 @app.route('/home', methods=['GET', 'POST'])
160 def home():
161     # print(name)
162
163     if 'id' in session:
164         if request.method == 'GET':
165             return render_template('home.html', name=session['username'])
166         if request.method == "POST":
167             # get data
168             lat = request.form["lat"]
169             lon = request.form["lon"]
170             if lat == "" or lon == "":
171                 return render_template('home.html', name=session['username'], email=session['email'], id=session['id'],
172                                     success=0)
173             # create a query to insert the data into the database

```

```

app.py 9+ X
D: > PRASHANTH > Projects > Final Deliverables > Flask_Application > app.py > ...
175         + lat + ", " + lon + ", 0)"
176         # execute the query
177         ibm_db.exec_immediate(conn, sql)
178         return render_template('home.html', name=session['username'], email=session['email'], id=session['id'],
179                             success=1)
180     return render_template('home.html', success=0)
181 else:
182     return redirect(url_for('login'))
183
184
185 # create a route for the data page and open only if the user is logged in
186 @app.route('/data')
187 def data():
188     if 'id' not in session:
189         return redirect(url_for('login'))
190     else:
191         # create a query to fetch the data from the database
192         sql = "SELECT * FROM inf_location"
193         stmt = ibm_db.exec_immediate(conn, sql)
194         # print("stmt", stmt)
195         # fetch all the data from the database and store it in the result dictionary
196         result = ibm_db.fetch_assoc(stmt)
197
198         # create a list to store the data
199         data = []
200         # loop through the result dictionary and append the data to the list
201         while result:
202             data.append(result)
203             result = ibm_db.fetch_assoc(stmt)
204         # print(data)
205         return render_template('data.html', data=data)
206

```

```

app.py 9+ X
D:\> PRASHANTH > Projects > Final Deliverables > Flask_Application > app.py > ...
208 # android_signup api
209 @app.route('/android_signup', methods=['POST'])
210 def android_signup():
211     if request.method == 'POST':
212         # get the data from the form
213         name = request.json['name']
214         email = request.json['email']
215         password = request.json['password']
216         # if nothing is entered in the form
217         # check if the email is valid
218         if re.match(r"^[^@]+@^[^@]+\.[^@]+$", email):
219             # insert the data into the database
220             # check if email already exists in the database
221             sql = "SELECT * FROM users WHERE email = '" + email + "'"
222             stmt = ibm_db.exec_immediate(conn, sql)
223             # print("stmt", stmt)
224             result = ibm_db.fetch_assoc(stmt)
225             # print("result", result)
226             if result:
227                 return jsonify({"message": "The username or email already exists!"})
228             else:
229                 sql = "INSERT INTO users (id, username, email, password,type) VALUES (seq_person.nextval,'" + name + \
230                     "', '" + email + "', '" + password + "', 2) "
231                 ibm_db.exec_immediate(conn, sql)
232                 # pass the id of the user to the android app
233                 sql = "SELECT * FROM users WHERE email = '" + email + "' AND password = '" + password + "'"
234                 stmt = ibm_db.exec_immediate(conn, sql)
235                 result = ibm_db.fetch_assoc(stmt)
236                 return {"status": "success", "message": "You have successfully registered!", "id": result['ID']}
237             else:
238                 return jsonify({'message': 'The email is invalid!'})
239         return jsonify({'message': 'The email is invalid!'})
240

```

```

263 @app.route("/post_user_location_data", methods=["POST"])
264 def post_user_location_data():
265     # get data
266     lat = request.json["lat"]
267     lon = request.json["long"]
268     id1 = request.json["id"]
269     ts = request.json['timestamp']
270     # create a query to insert the data into the database
271     sql = "INSERT INTO location (LOCATE_LAT, LOCATE_LONG, USER_ID, TIME_STAMP) VALUES ('" + lat + "', '" + lon + "', '" + str(
272         id1) + "', '" + ts + "'"
273     # execute the query
274     ibm_db.exec_immediate(conn, sql)
275     return {"status": "success", "message": "You have successfully registered!"}
276
277
278 @app.route("/location_data")
279 def location_data():
280     # create a query to fetch the data from the database
281     sql = "SELECT * FROM inf_location"
282     stmt = ibm_db.exec_immediate(conn, sql)
283     # print("stmt", stmt)
284     # fetch all the data from the database and store it in the result dictionary
285     result = ibm_db.fetch_assoc(stmt)
286
287     # create a list to store the data
288     data = []
289     # loop through the result dictionary and append the data to the list
290     while result:
291         data.append(result)
292         ibm_db.fetch_assoc(stmt)
293     # print(data)
294     return json.dumps(data)

```

```

app.py 9+ X
D: > PRASHANTH > Projects > Final Deliverables > Flask_Application > app.py > ...

295
296     else:
297         return {"response": "failure"}
298
299
300 @app.route("/get_all_users")
301 def get_users():
302     # create a query to fetch the data from the database
303     sql = "SELECT * FROM users"
304     stmt = ibm_db.exec_immediate(conn, sql)
305     # print("stmt", stmt)
306     # fetch all the data from the database and store it in the result dictionary
307     result = ibm_db.fetch_assoc(stmt)
308     if result:
309         # create a list to store the data
310         data = []
311         # loop through the result dictionary and append the data to the list
312         while result:
313             data.append(result)
314             result = ibm_db.fetch_assoc(stmt)
315         # print(data)
316         return json.dumps(data)
317
318     # if(user_result > 0):
319     #     rv = signup_cursor.fetchall()
320     #     row_headers = [x[0] for x in signup_cursor.description]
321     #     json_data = []
322     #     for result in rv:
323     #         json_data.append(dict(zip(row_headers, result)))
324     #     return json.dumps(json_data)
325
326

```

```

app.py 9+ X
D: > PRASHANTH > Projects > Final Deliverables > Flask_Application > app.py > ...

328 def send_trigger():
329     if request.method == "POST":
330         # get the data from the form
331         email = request.json['email']
332         location_id = request.json['id']
333         # print("email and loc", email, location_id)
334         # get location data
335         sql = "SELECT VISITED FROM INF_LOCATION WHERE LOCATE_ID = '" + str(location_id) + "'"
336         stmt = ibm_db.exec_immediate(conn, sql)
337         print("stmt", stmt)
338         if stmt:
339             result = ibm_db.fetch_assoc(stmt)
340             if result:
341                 visited = result['VISITED']
342                 visited = visited + 1
343                 sql = "UPDATE INF_LOCATION SET VISITED = '" + str(visited) + "' WHERE LOCATE_ID = '" + str(
344                     location_id) + "'"
345                 ibm_db.exec_immediate(conn, sql)
346
347                 ibm_db.exec_immediate(conn, sql)
348                 # send email
349                 # print("email ->", email)
350                 sendemail(email)
351                 return {"response": "Mail success"}
352             else:
353                 return {"response": "Mail failed"}
354         #sendemail(email)
355         #return {"response": "Mail success"}
356
357
358 if __name__ == '__main__':
359     app.run(debug=True, host='0.0.0.0', port=5000)

```