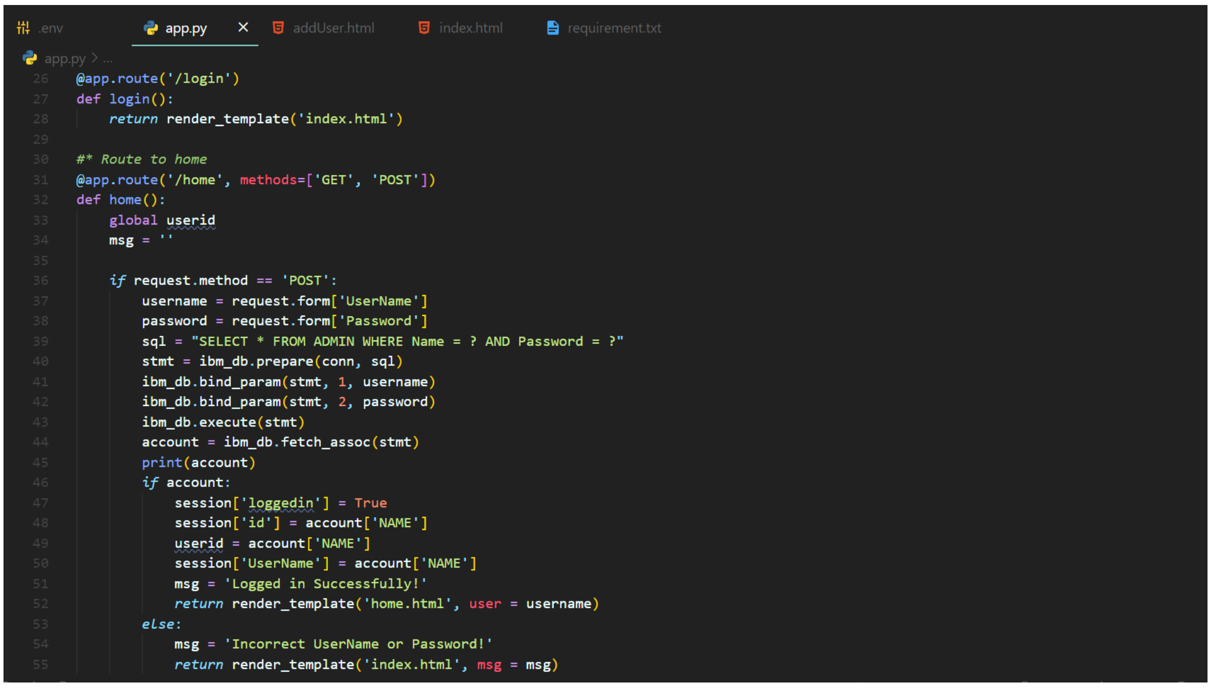
Creating APIs In Flask

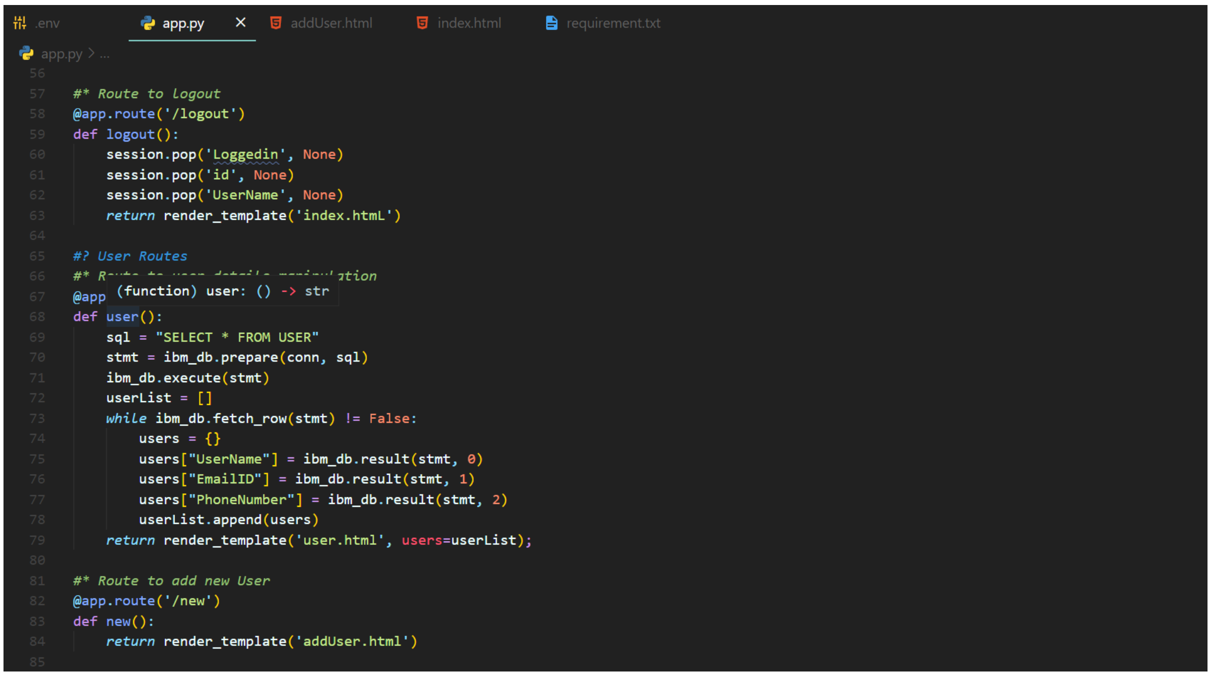
Sample APIs used in our Project.

1) Home Route.



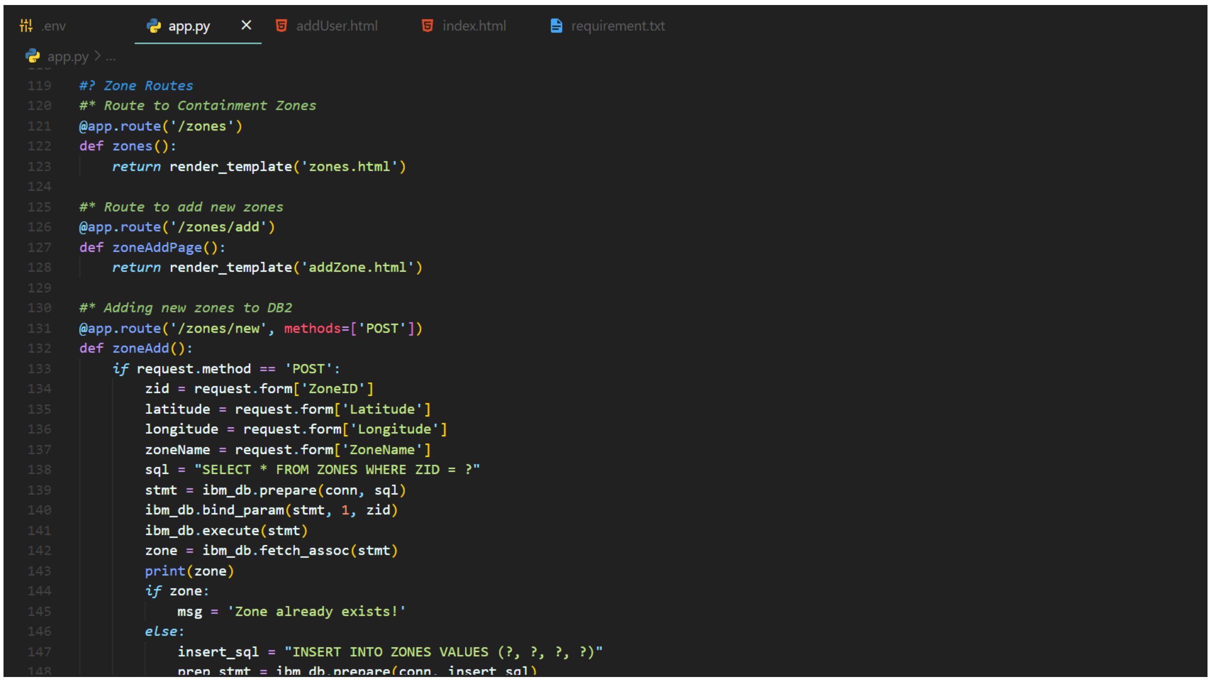
```python
@app.route('/login')
def login():
    return render_template('index.html')

#* Route to home
@app.route('/home', methods=['GET', 'POST'])
def home():
    global userid
    msg = ''

    if request.method == 'POST':
        username = request.form['UserName']
        password = request.form['Password']
        sql = "SELECT * FROM ADMIN WHERE Name = ? 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['NAME']
            userid = account['NAME']
            session['UserName'] = account['NAME']
            msg = 'Logged in Successfully!'
            return render_template('home.html', user = username)
        else:
            msg = 'Incorrect UserName or Password!'
            return render_template('index.html', msg = msg)
```

2) User Route.



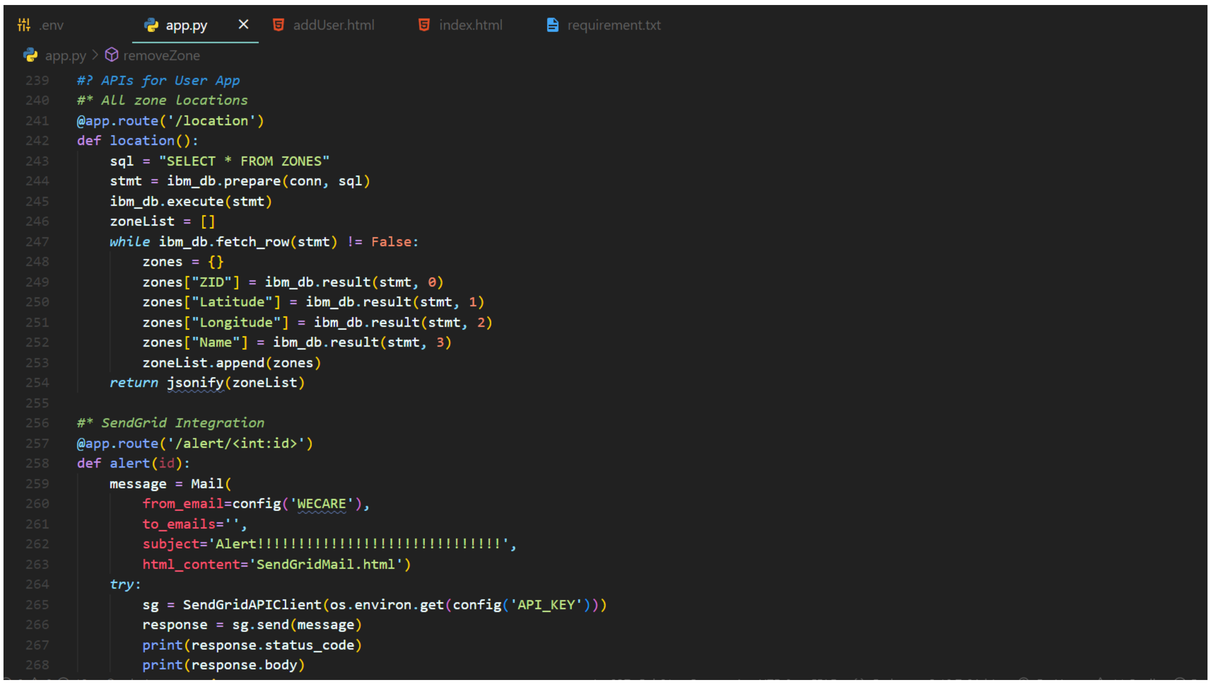
```python

#* Route to Logout
@app.route('/logout')
def logout():
    session.pop('Loggedin', None)
    session.pop('id', None)
    session.pop('UserName', None)
    return render_template('index.htmL')

#? User Routes
#* Route to user details manipulation
@app (function) user: () -> str
def user():
    sql = "SELECT * FROM USER"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.execute(stmt)
    userList = []
    while ibm_db.fetch_row(stmt) != False:
        users = {}
        users["UserName"] = ibm_db.result(stmt, 0)
        users["EmailID"] = ibm_db.result(stmt, 1)
        users["PhoneNumber"] = ibm_db.result(stmt, 2)
        userList.append(users)
    return render_template('user.html', users=userList);

#* Route to add new User
@app.route('/new')
def new():
    return render_template('addUser.html')

```

3) Zone Route.



```python
#? Zone Routes
#* Route to Containment Zones
@app.route('/zones')
def zones():
    return render_template('zones.html')

#* Route to add new zones
@app.route('/zones/add')
def zoneAddPage():
    return render_template('addZone.html')

#* Adding new zones to DB2
@app.route('/zones/new', methods=['POST'])
def zoneAdd():
    if request.method == 'POST':
        zid = request.form['ZoneID']
        latitude = request.form['Latitude']
        longitude = request.form['Longitude']
        zoneName = request.form['ZoneName']
        sql = "SELECT * FROM ZONES WHERE ZID = ?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, zid)
        ibm_db.execute(stmt)
        zone = ibm_db.fetch_assoc(stmt)
        print(zone)
        if zone:
            msg = 'Zone already exists!'
        else:
            insert_sql = "INSERT INTO ZONES VALUES (?, ?, ?, ?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
```

4) APIs for Mobile Requests.



```python
#? APIs for User App
#* All zone locations
@app.route('/location')
def location():
    sql = "SELECT * FROM ZONES"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.execute(stmt)
    zoneList = []
    while ibm_db.fetch_row(stmt) != False:
        zones = {}
        zones["ZID"] = ibm_db.result(stmt, 0)
        zones["Latitude"] = ibm_db.result(stmt, 1)
        zones["Longitude"] = ibm_db.result(stmt, 2)
        zones["Name"] = ibm_db.result(stmt, 3)
        zoneList.append(zones)
    return jsonify(zoneList)

#* SendGrid Integration
@app.route('/alert/<int:id>')
def alert(id):
    message = Mail(
        from_email=config('WECARE'),
        to_emails='',
        subject='Alert!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!',
        html_content='SendGridMail.html')
    try:
        sg = SendGridAPIClient(os.environ.get(config('API_KEY')))
        response = sg.send(message)
        print(response.status_code)
        print(response.body)
```