



education.smartinternz.com

4



IBM



ICTACADEMY

PROJECT ORIENTATION SESSIONS

CLOUD APP DEVELOPMENT

LIVE ON : 02 AUGUST 2022,
9AM - 12PM AND 6PM - 9PM



LIVE



104



Agile Methodology - Session 1

Portfolio



zoom



LIVE



484



Join to chat



Zoom

Leave

REC

```
File Edit Selection View Go Run Terminal Help app.py - module4 - Visual Studio Code
EXTENSION
MODULE4
  flask_with_sqlite
  flask-with-ibm-db2
  __pycache__
  static
  templates
  app.py
  db.py
  DigiCertGlobalRoot...
  ibmda2-connect.py
  revivedata.py
  flaskfiles
OUTLINE
TIMELINE
flask-with-ibm-db2 > app.py > home
1 from flask import *
2 from flask import Flask, render_template, request, redirect, url_for, session
3 from flask_sqlalchemy import SQLAlchemy
4
5 import ibm_db
6 conn = ibm_db.connect("DATABASE=blusdb;HOSTNAME=2548b6b4-chf6-40a8-bbce-6251e6ba8300.bs21o90l08kqb1od8lcp.dp
7 # conn = ibm_db.connect("DATABASE=blusdb;HOSTNAME=your_hostname;PORT=your_number;SECURITY=SSL;S
8
9 app = Flask(__name__)
10
11
12
13 @app.route('/')
14 def home():
15     return render_template('home.html')
16
17 @app.route('/addstudent')
18 def new_student():
19     return render_template('add_student.html')
20
21 @app.route('/addrec', methods = ['POST', 'GET'])
22 def addrec():
23     if request.method == 'POST':
24
25         name = request.form['name']
26         address = request.form['address']
```



Chat



Raise Hand



Q&A



More

```

def handleSignup(request):
    if request.method == 'POST':
        # get the post parameters
        uname = request.POST["uname"]
        fname=request.POST["fname"]
        lname=request.POST["lname"]
        email = request.POST["email"]
        profession = request.POST['profession']
        Savings = request.POST['Savings']
        income = request.POST['income']
        pass1 = request.POST["pass1"]
        pass2 = request.POST["pass2"]
        profile = UserProfile(Savings = Savings,profession=profession,income=income)
        # check for errors in input
        if request.method == 'POST':
            try:
                user_exists = User.objects.get(username=request.POST['uname'])
                messages.error(request," Username already taken, Try something else!!!")
                return redirect("/register")
            except User.DoesNotExist:
                if len(uname)>15:
                    messages.error(request," Username must be max 15 characters, Please")
                    return redirect("/register")

                if not uname.isalnum():
                    messages.error(request," Username should only contain letters and")
                    return redirect("/register")

                if pass1 != pass2:
                    messages.error(request," Password do not match, Please try again")
                    return redirect("/register")

```

```

def addmoney_submission(request):
    if request.session.has_key('is_logged'):
        if request.method == "POST":
            user_id = request.session["user_id"]
            user1 = User.objects.get(id=user_id)
            addmoney_info1 = Addmoney_info.objects.filter(user=user1).order_by('-Date')
            add_money = request.POST["add_money"]
            quantity = request.POST["quantity"]
            Date = request.POST["Date"]
            Category = request.POST["Category"]
            add = Addmoney_info(user = user1,add_money=add_money,quantity=quantity,Date
            add.save()
            paginator = Paginator(addmoney_info1, 4)
            page_number = request.GET.get('page')
            page_obj = Paginator.get_page(paginator,page_number)
            context = {
                'page_obj' : page_obj
            }
            return render(request,'home/index.html',context)
        return redirect('/index')
def addmoney_update(request,id):
    if request.session.has_key('is_logged'):
        if request.method == "POST":
            add = Addmoney_info.objects.get(id=id)
            add .add_money = request.POST["add_money"]
            add.quantity = request.POST["quantity"]
            add.Date = request.POST["Date"]
            add.Category = request.POST["Category"]
            add .save()
            return redirect("/index")
    return redirect("/home")

```

```

def expense_month(request):
    todays_date = datetime.date.today()
    one_month_ago = todays_date-datetime.timedelta(days=30)
    user_id = request.session["user_id"]
    user1 = User.objects.get(id=user_id)
    addmoney = Addmoney_info.objects.filter(user = user1,Date__gte=one_month_ago,Date__lte=todays_date)
    finalrep ={}

    def get_Category(addmoney_info):
        # if addmoney_info.add_money=="Expense":
        return addmoney_info.Category

    Category_list = list(set(map(get_Category,addmoney)))

    def get_expense_category_amount(Category,add_money):
        quantity = 0
        filtered_by_category = addmoney.filter(Category = Category,add_money="Expense")
        for item in filtered_by_category:
            quantity+=item.quantity
        return quantity

    for x in addmoney:
        for y in Category_list:
            finalrep[y]= get_expense_category_amount(y,"Expense")

    return JsonResponse({'expense_category_data': finalrep}, safe=False)

def stats(request):
    if request.session.has_key('is_logged') :
        todays_date = datetime.date.today()
        one_month_ago = todays_date-datetime.timedelta(days=30)
        user_id = request.session["user_id"]
        user1 = User.objects.get(id=user_id)
        addmoney_info = Addmoney_info.objects.filter(user = user1,Date__gte=one_month_ago,Date__lte=todays_date)
        sum = 0

```