**Virtual Environment in Python**

**In command prompt:**

Method1:

1. python -m venv my\_env
2. .\my\_env\Scripts\activate
3. pip install <packages >
4. pip freeze > requirements.txt
5. deactivate
6. pip install -r requirements.txt

Method2:

1. pip install pipenv
2. pipenv install <packages>
3. pipenv shell [activation]
4. pipenv lock
5. exit
6. pipenv install

**Database integration in python**

**In MongoDB:**

* use a;
* db.createCollection(“sample”);
* db.sample.insertOne({“name” : “balaji” , “num”:1});
* db.sample.insertMany( [ {“name ” : ”balaji” , ”num” : 1 } , {“name” : ”b” , ”num” : 2} ]);
* db.student.find({});
* db.student.updateOne({“num” : 2} , { “$set” : {“name” : “bheesetti”} });
* db.student.deleteOne({“num” : 2});
* db.student.deleteOne({“\_id” : ObjectId(“ “)});

(In command prompt: pip install pymongo, import it in flask)

1. my\_client = MongoClient(“localhost”,27017)
2. my\_db = my\_client[“db\_name”]
3. my\_collection = my\_db[“collection\_name”]

**In MySQL:**

(In command prompt: pip install pymysql, import it in flask)

1. connection = pymysql.connect(host ="localhost" ,port=3306,user="root" ,password ="MYSQLpassword" ,database="db\_name")
2. my\_cursor =connection.cursor() (cursor is used to execute sql queries and it manages db resources and connections)
3. my\_cursor.execute("create database name") (to create database if database is not written in connect method)
4. my\_cursor.execute("create table if not exist table\_name (name varchar(20), password varchar(20),mobile\_no int(10)")
5. my\_cursor.execute(f"insert into table table\_name(name,password) values('{name}','{password}')") or ("insert into table table\_name(name,password) values(%s,%s),(f"{name}",f"{password}")")
6. connection.commit() (after every execution to save data in table use commit)
7. my\_cursor.execute("select \* from table\_name where name = (%s),(f"{name}")")

**Mail in Flask**

(in command prompt: pip install flask\_mail, import it in flask)

* app.config[“MAIL\_SERVER”] = “smtp.gmail.com”
* app.config[“MAIL\_PORT”] = 587
* app.config[“MAIL\_USE\_TLS”] = True
* app.config[“MAIL\_USERNAME”] = “mail@gmail.com”
* app.config[“MAIL\_PASSWORD”] = “password”
* created\_mail = Mail(app)
* msg = Message(subject= ””,recipients =[“”,””],sender= “mail@gmail.com” ,body = “”)
* created\_mail.send(msg)

**password hashing in flask**

(in flask from werkzeug.security import generate\_password\_hash, check\_password\_hash)

**FLASK**

----------in command prompt--------------

create a folder and open it in command prompt

* pip install flask
* pip show flask (to see installed or not)
* flask run (or) flask --app appname.py run (if app name is app.py then just flask run)
* Debug (to ignore stop and start the server):

flask --debug run

* To change port number:

flask --debug run -p 1000

--------------------FOLDER STRUCTURE-----------------------------

Folder\_name

* Static

style1.css

style2.css

* Templates

index.html

name.html

app.py

---------------FORMS :-------------

GET :- data is visible in url after sending data

POST :- data is hidden in url after sending data

\* In html at form if you add action="name" then you have to create a new route("/name") and write a function to see data .

^ if method is get then no errors are found ^

^ if method is POST then it displays error (to solve this we have to check condition is get or post) ^

render\_template:

-->render\_template("sample.html")

it is used to display html pages

redirect:

-->redirect("/")

In forms to check form is get or post then use it in 2nd url(route) if it is used in main url it cause error ^by default forms always in get method so else (get)condition became true and it occurs continuously and display error (this page isn't working)

url\_for:

-->redirect(url\_for("function name"))

In forms at 2nd url(condition checking url) instead of writing redirect ,add url\_for also (incase path change then output of redirect also change to ensure that add FunctionName which is fixed)

flash:

--> (in condition if username already exist)

------------------------Mail in project----------------

pip install flask-mail