# An overview

A platform where people can search, rent and loan furniture

The two actors in the system are:

- The admin :
  - Creates accounts and maintain inventory
- The Customer :
  - Loans/rents furniture

Storage of details for furniture:

- Type,
- Relevant images
- Rental price
- Cost price
- Interest rate
- Reviews

Useful for people who are looking to refurbish their house after relocation

*and this was our motivation!*

# Tools used

Tkinter

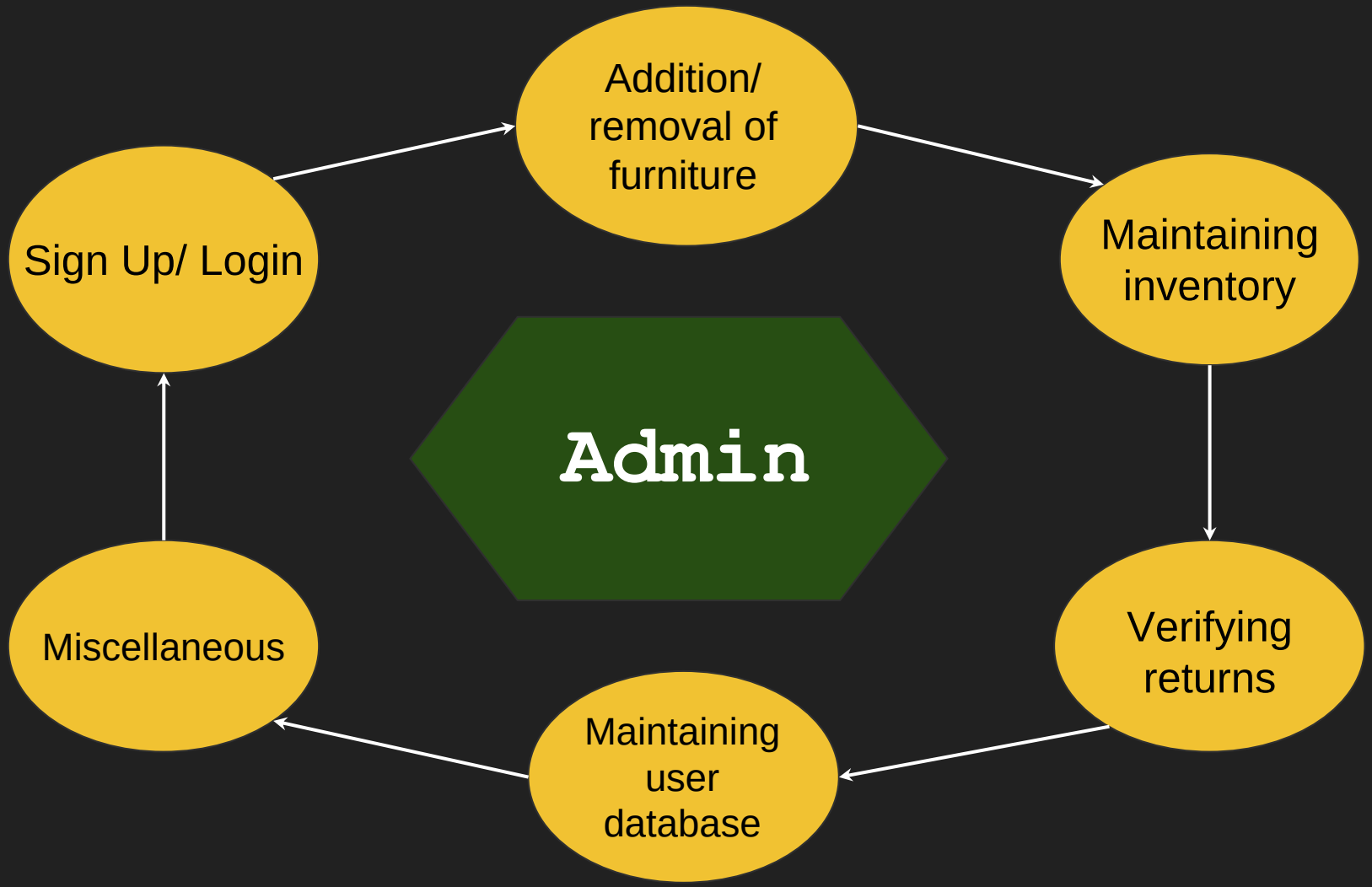
MySQL®

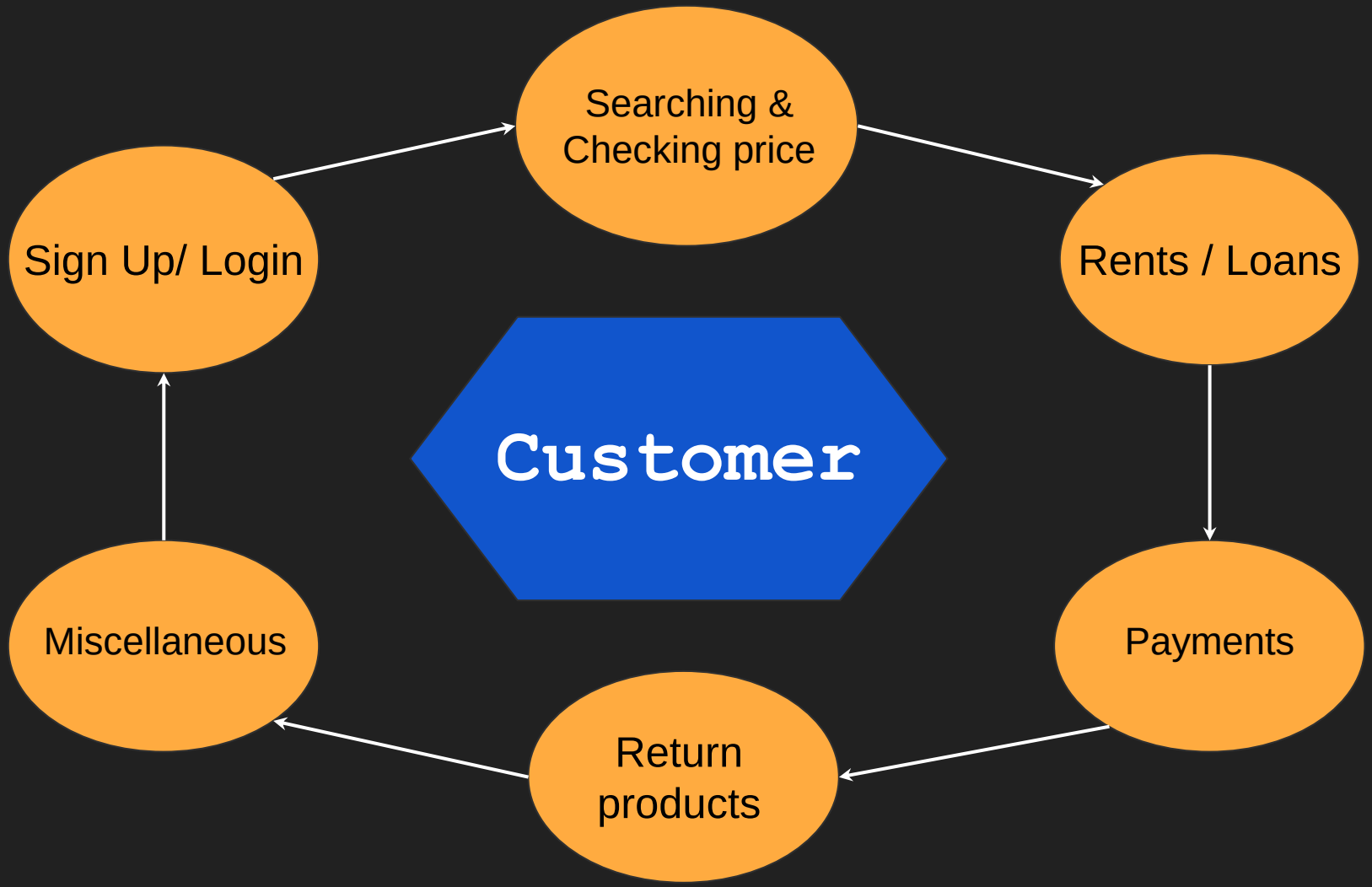

python + Gmail

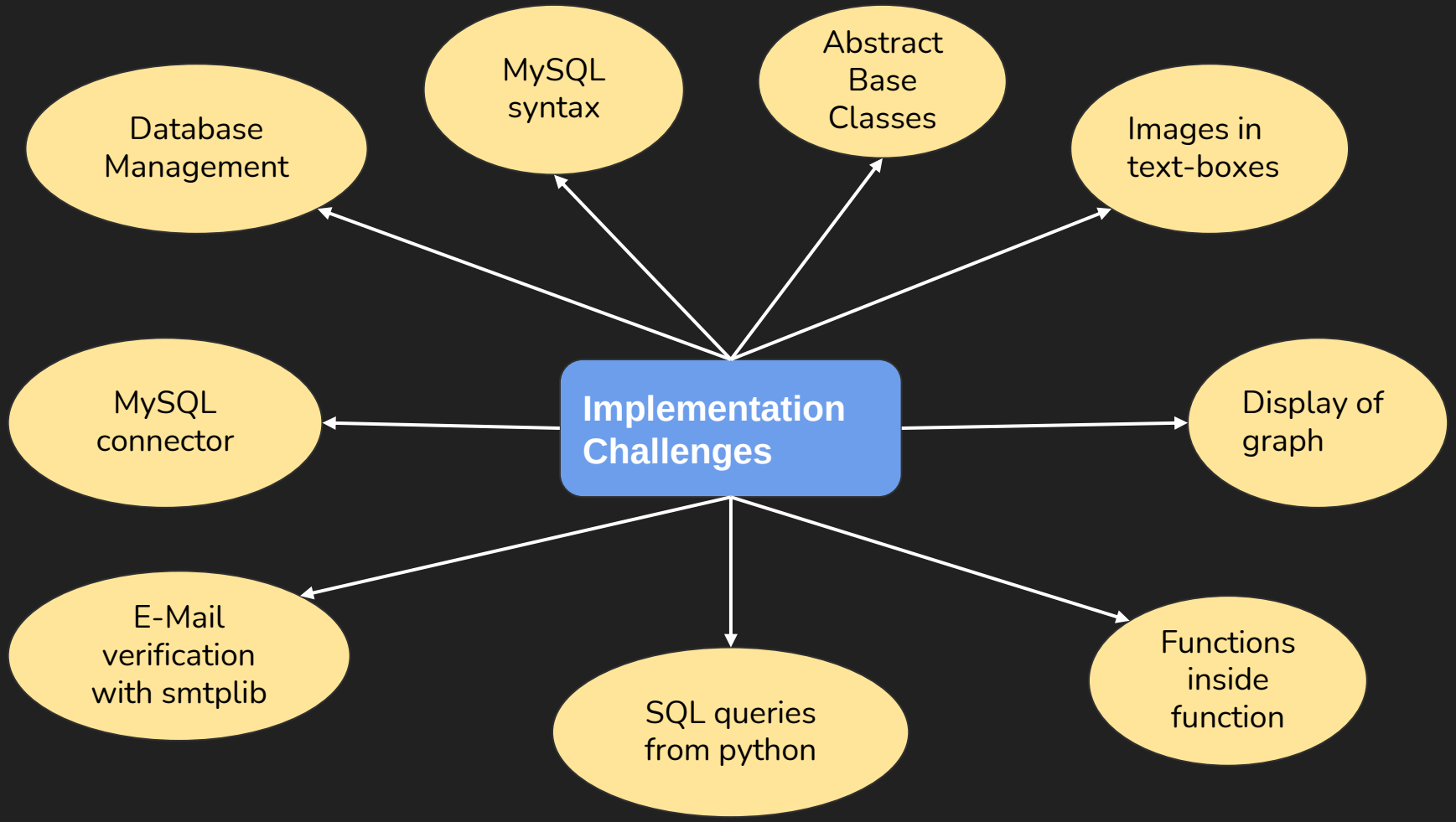
matplotlib

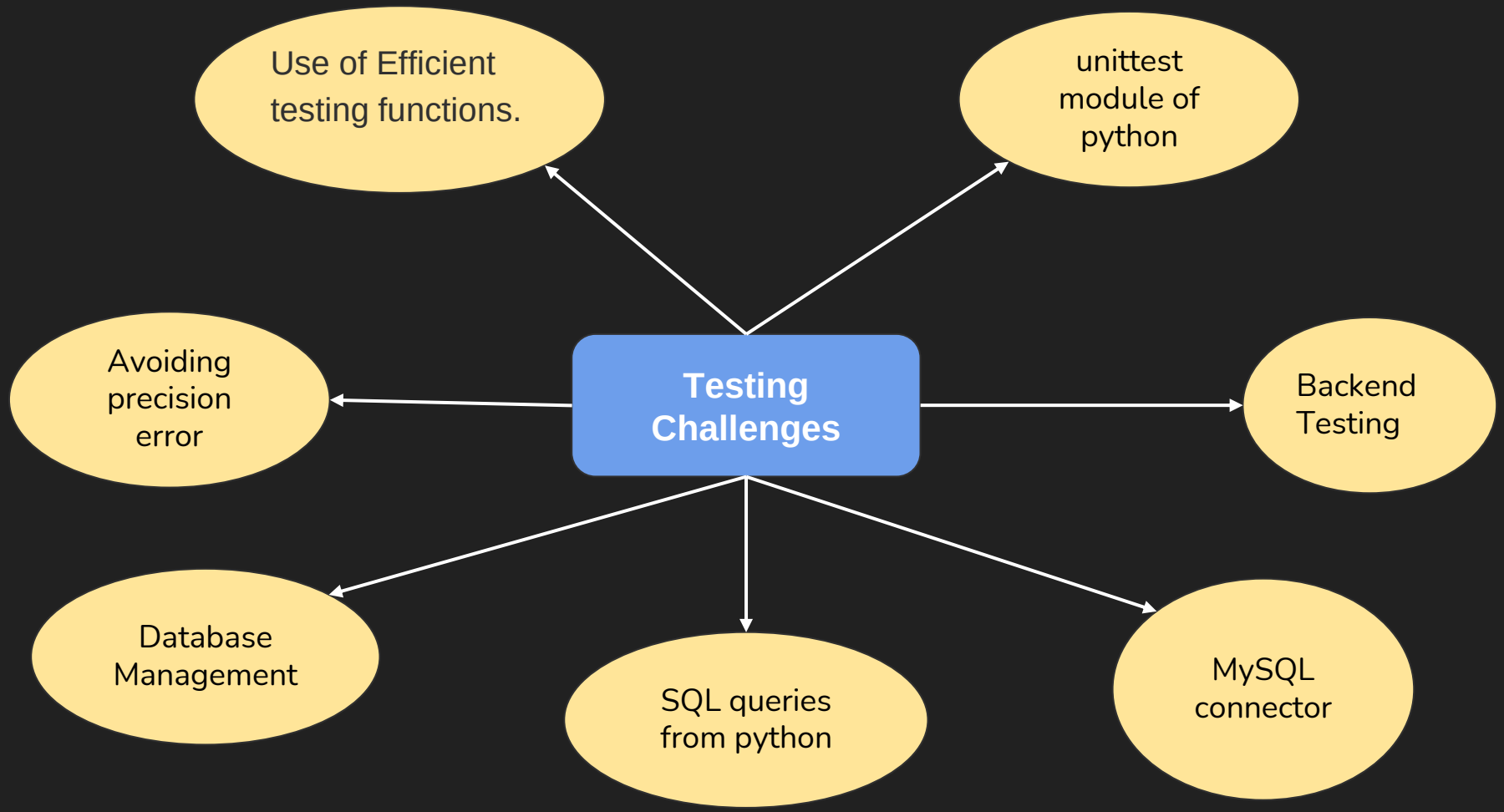
pillow

# Features and Functionalities


Addition/ removal of furniture
Maintaining inventory
Verifying returns
Maintaining user database
Miscellaneous
Sign Up/ Login
Admin

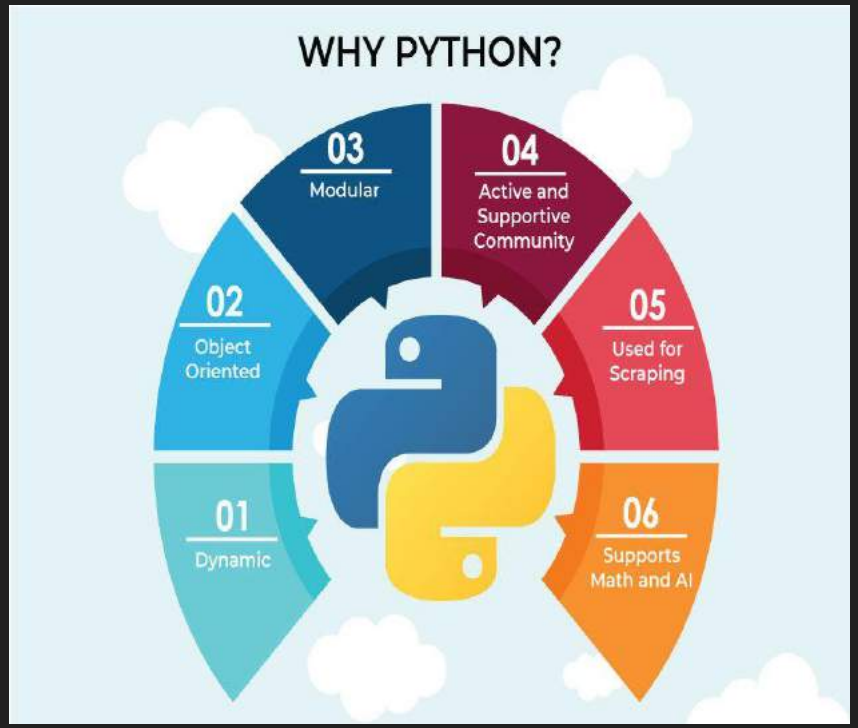

Customer
Searching & Checking price
Rents / Loans
Payments
Return products
Miscellaneous
Sign Up/ Login


Implementation Challenges
Database Management
MySQL syntax
Abstract Base Classes
Images in text-boxes
MySQL connector
Display of graph
E-Mail verification with smtplib
SQL queries from python
Functions inside function


Testing Challenges
Use of Efficient testing functions.
unittest module of python
Avoiding precision error
Backend Testing
Database Management
SQL queries from python
MySQL connector

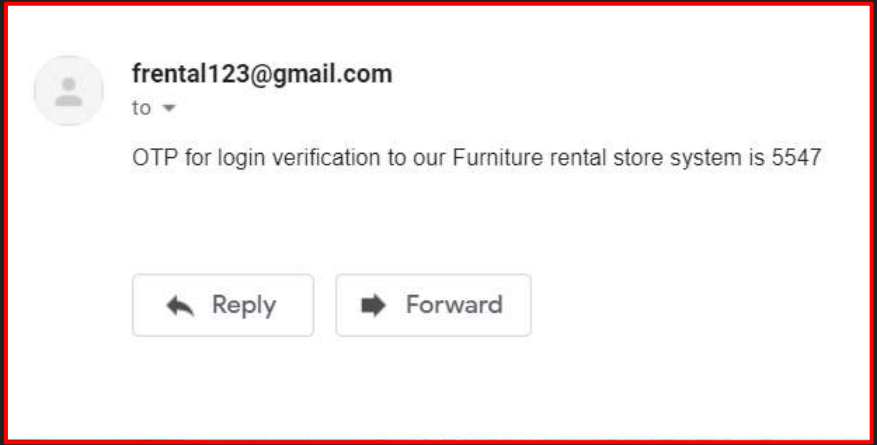
# Motivation to Choose Python

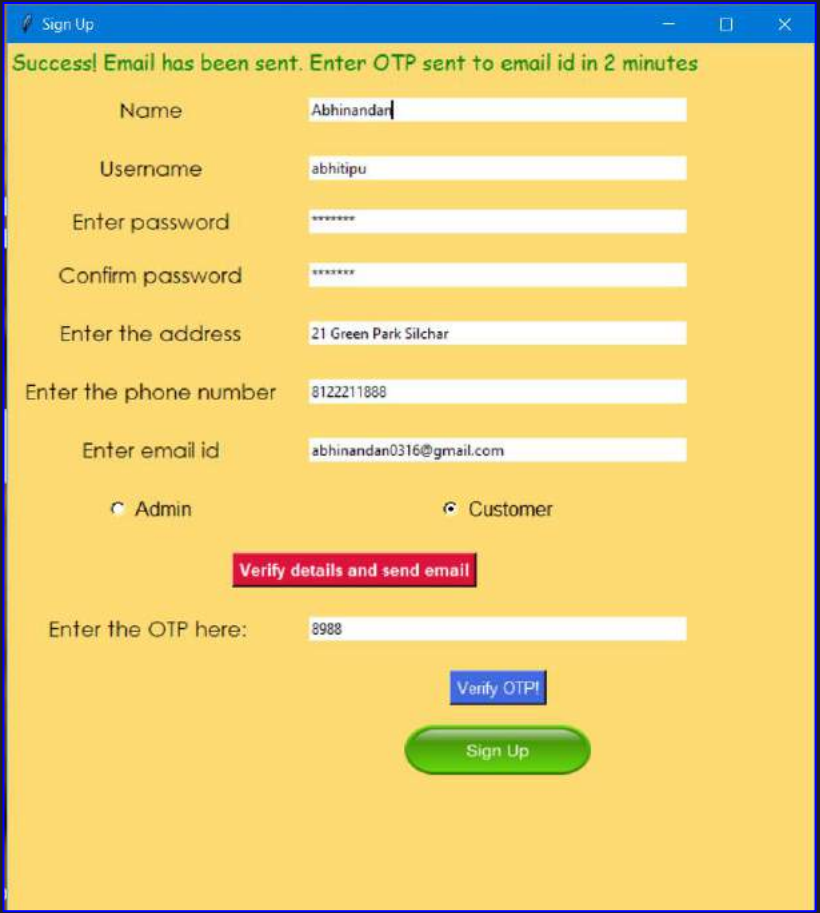
- Simplified Syntax and similarity with natural language.
- Excellent libraries which reduce time and effort of development.
- Ultimate tool to play with graphs and data science.
- Great integration of MySQL as a database management tool.
- Easy to use GUI with Tkinter.
- Vast support of error detection with the VSCode editor.

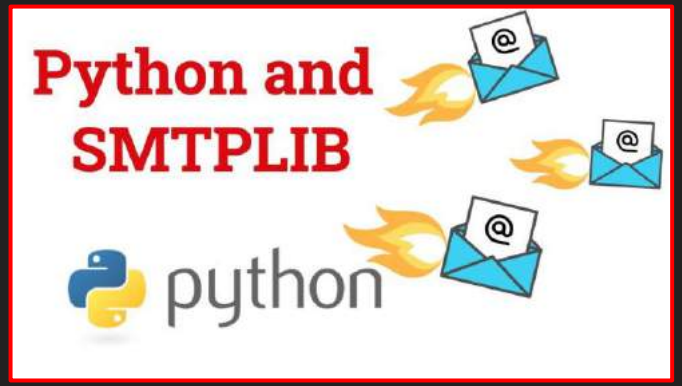


# e-mail verification

ft. smtplib



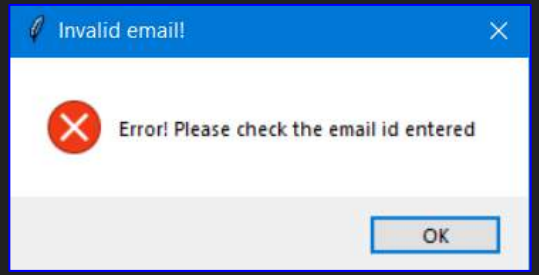



```python
import smtplib
import random
import time
```

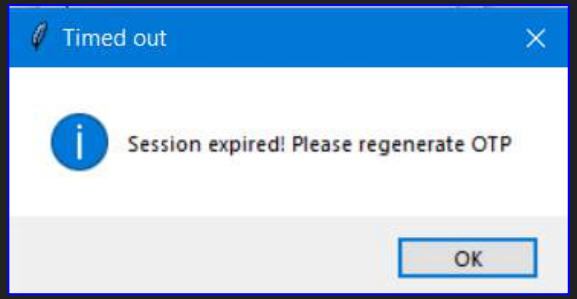
Use of the smtplib library to verify e-mail

Using random library to generate random OTPs.

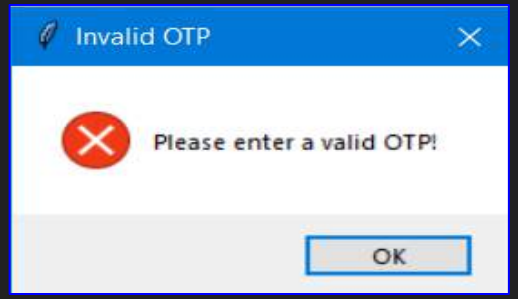
ERRORS!!




Email id validation




2 min limit




Final validation

# Price Calculation while Buying Furniture

$$NewInterest = InterestRate \cdot 2^{-count}$$

$$count = Number\ of\ \ orders\ in\ the\ past$$

Interest Rate based on past order history

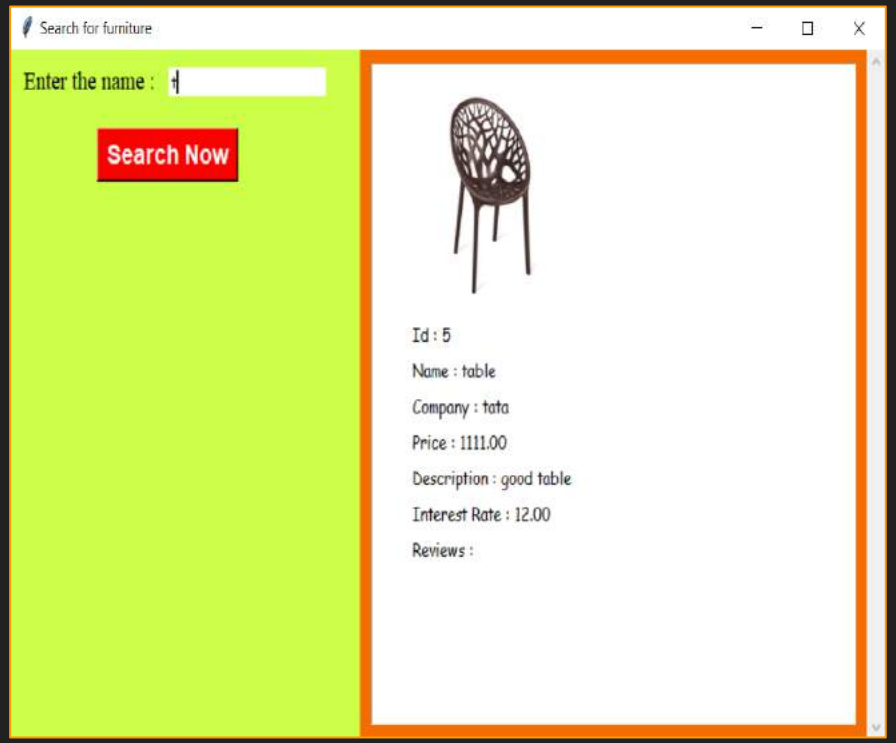
$$Price = Principal \cdot \left(1 + InterestRate \cdot \frac{TimePeriod}{100}\right)$$

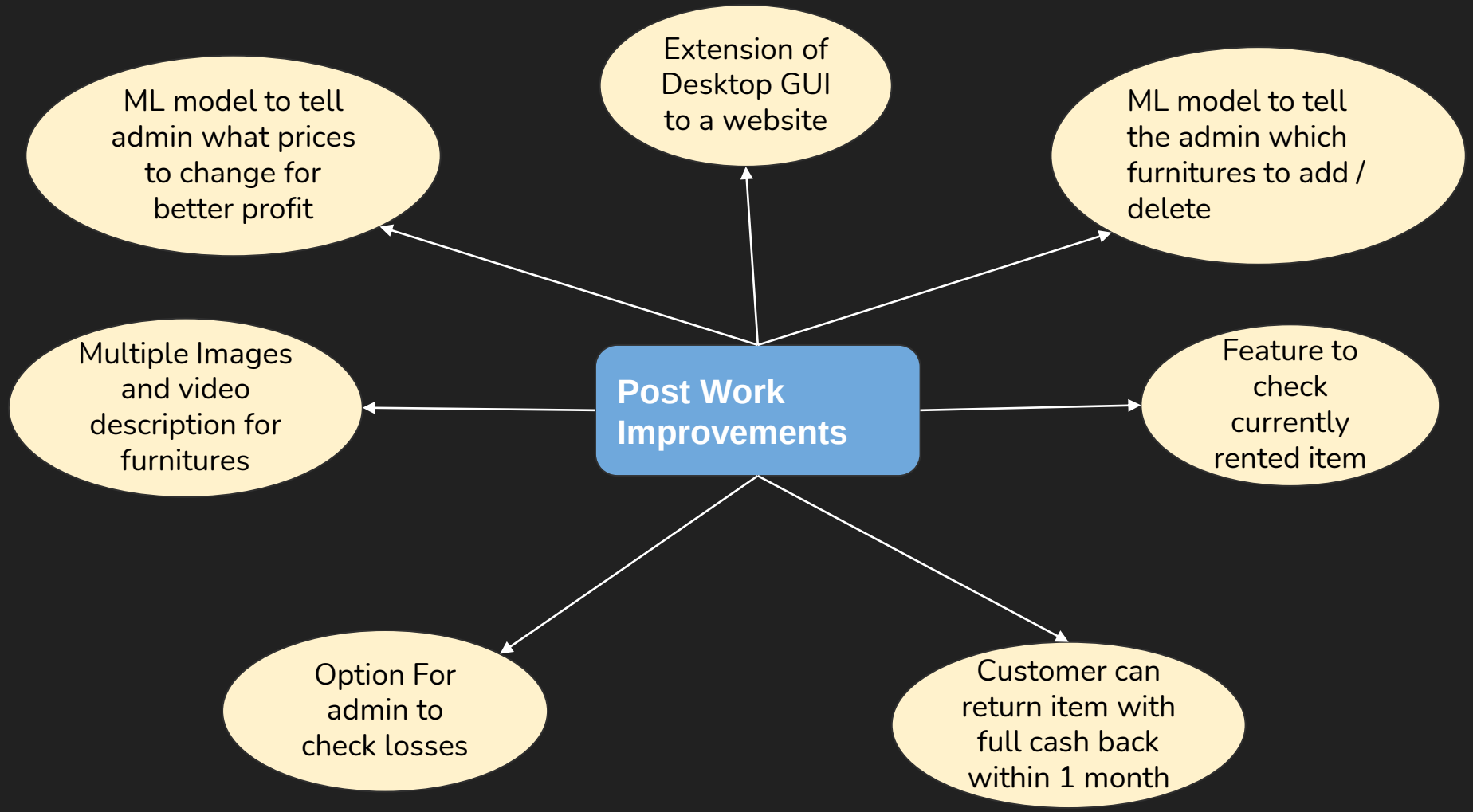
Price Calculation while buying on loan

$$NewPrice = OldPrice \cdot \frac{(DaysRented + 9)}{10}$$

Price Calculation while buying on rent


Substring Search
Furniture id , image , description
Non-rented items
Searching Furniture
Distinct items
Past reviews , price
Search based on name


Search for furniture
Enter the name :
Search Now
Id : 5
Name : table
Company : tata
Price : 1111.00
Description : good table
Interest Rate : 12.00
Reviews :


Post Work Improvements
Extension of Desktop GUI to a website
ML model to tell admin what prices to change for better profit
ML model to tell the admin which furnitures to add / delete
Multiple Images and video description for furnitures
Feature to check currently rented item
Option For admin to check losses
Customer can return item with full cash back within 1 month

# unittest module of python

01 **UnitTest Module**

- Verification of each functionality
- Validation by the QA team

02 **Validation and Verification**

- First level of software testing
- Unit testing each class
- Integration testing

03 **OOP Concepts supported by framework**

- Test Fixture
- Test Case
- Test Suite
- Test Runner

04 **Possible Outcomes**

- OK
- FAIL
- ERROR

# How to Use unittest module of python

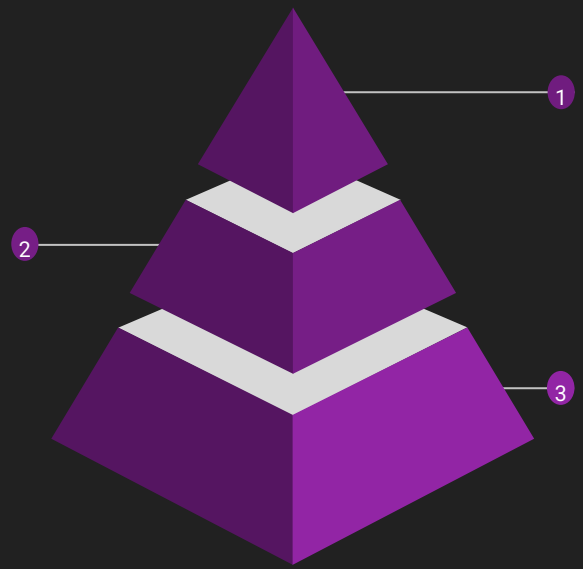
| Process 1 | Process 2 | Process 3 | Process 4 | Process 5 |
|---|---|---|---|---|
| Import unittest module and Define Testcase Subclass | Define a function or a class (in our case) that is to be tested. | Define a Test class as a subclass of unittest.TestCase | In the class defined in Process 3 test each of the functions or methods of the class one by one. | Then Execute the tests by the command $ python -m unittest {filename}.py |

```
import unittest

class TestFurnitureMethods(unittest.TestCase):
    myfurniture = Furniture(1, "Center Table", "neelka

    def test_id(self):
        self.assertEqual(self.myfurniture.getId(), 1)
```

```
................
----------------------------------------
Ran 16 tests in 0.003s

OK
```

# Backend / Database Testing

test.py is used to Test all the operations that involve any kind of reading and writing to the database.



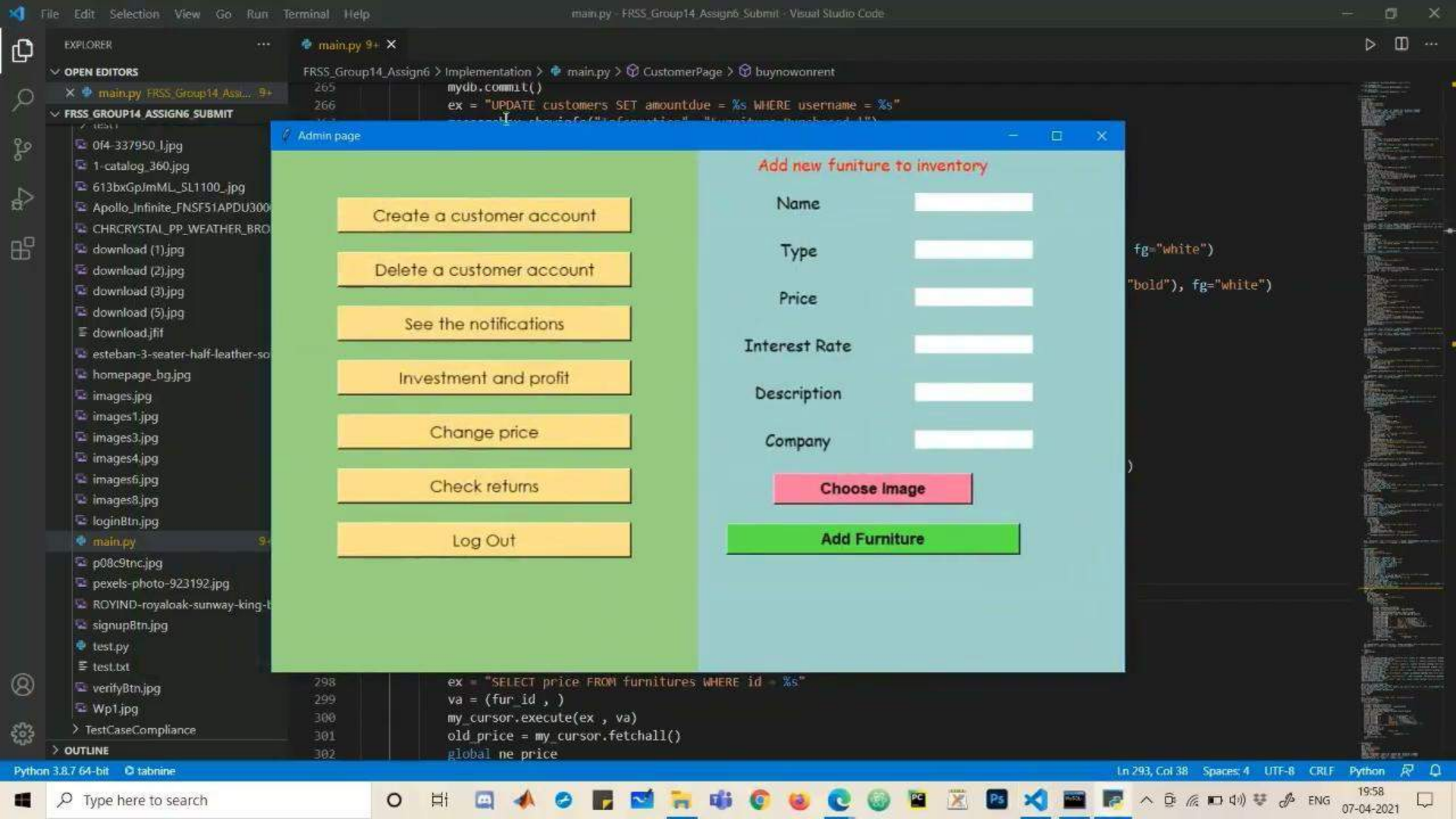

**Made a test.py file**

test.py file contains all the function used for testing

Imported test.py in the main program and checked at each step whether the right query is made and the desired changes are being done or not in the main database.

```
def testfurnituredelete(fur_id):
    exe = "SELECT * FROM furnitures WHERE id = %s"
    va = (fur_id,)
    my_cursor1.execute(exe, va)
    res = my_cursor1.fetchall()
    mydb1.commit()
    if len(res) > 0:
        print("DELETE DAMAGED FURNITURE : FAILED")
    else:
        print("DELETE DAMAGED FURNITURE : PASSED")
```

# A SHORT DEMO


File Edit Selection View Go Run Terminal Help
main.py - FRSS_Group14_Assign6_Submit - Visual Studio Code
EXPLORER
OPEN EDITORS
main.py FRSS_Group14_Assi... 9+
FRSS_GROUP14_ASSIGN6_SUBMIT
0f4-337950_l.jpg
1-catalog_360.jpg
613bxGpJmML_SL1100_jpg
Apollo_Infinite_FNSF51APDU300
CHRCRYSTAL_PP_WEATHER_BRO
download (1).jpg
download (2).jpg
download (3).jpg
download (5).jpg
download.jfif
esteban-3-seater-half-leather-so
homepage_bg.jpg
images.jpg
images1.jpg
images3.jpg
images4.jpg
images6.jpg
images8.jpg
loginBtn.jpg
main.py
p08c9tnc.jpg
pexels-photo-923192.jpg
ROYIND-royaloak-sunway-king-b
signupBtn.jpg
test.py
test.txt
verifyBtn.jpg
Wp1.jpg
TestCaseCompliance
OUTLINE
main.py 9+
FRSS_Group14_Assign6 > Implementation > main.py > CustomerPage > buynowonrent
mydb.commit()
ex = "UPDATE customers SET amountdue = %s WHERE username = %s"
fg="white")
"bold"), fg="white")
)
ex = "SELECT price FROM furnitures WHERE id = %s"
va = (fur_id , )
my_cursor.execute(ex , va)
old_price = my_cursor.fetchall()
global ne price
Admin page
Create a customer account
Delete a customer account
See the notifications
Investment and profit
Change price
Check returns
Log Out
Add new funiture to inventory
Name
Type
Price
Interest Rate
Description
Company
Choose Image
Add Furniture
Python 3.8.7 64-bit
tabnine
Ln 293, Col 38
Spaces: 4
UTF-8
CRLF
Python
Type here to search
ENG
19:58
07-04-2021