ASSIGNMENT-5

Frason Francis / 201903020 / 25

Develop GUI Application for E-commerce
application use **(LO5)**

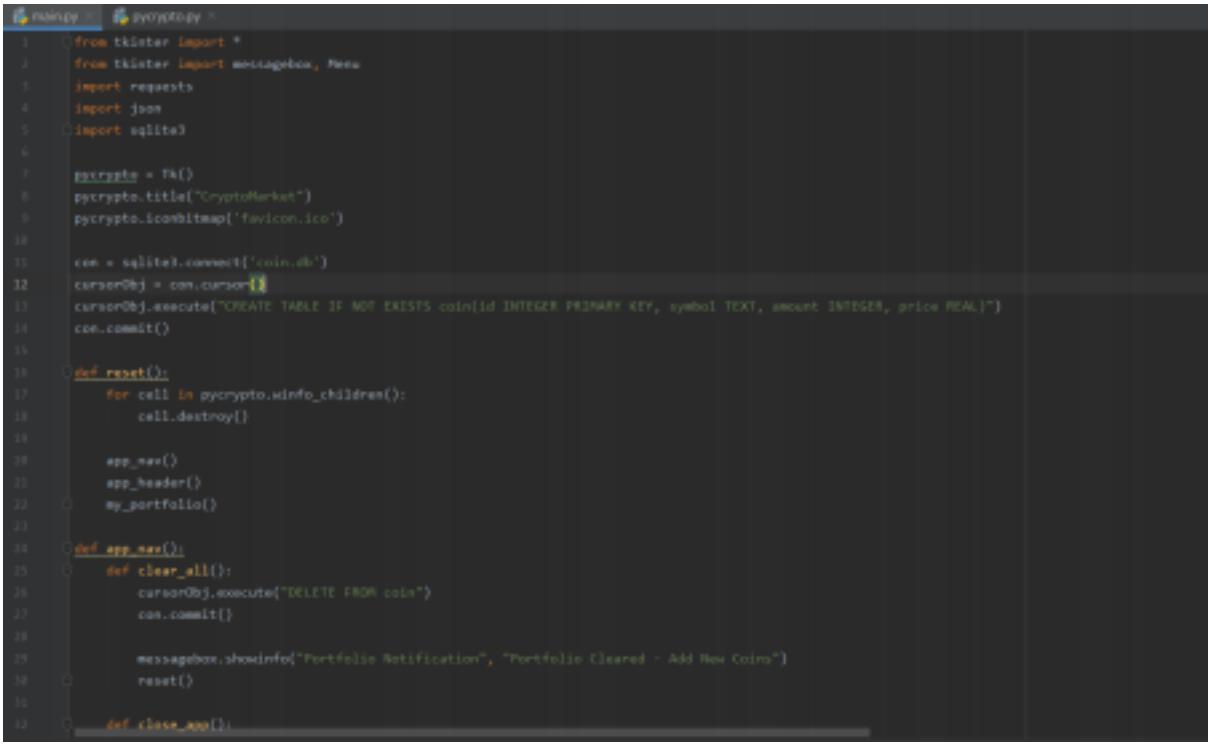
1) file, pickle, dictionary to show add, delete, update operations.

2) sqlite3 dictionary to show add, delete, update operations.

Theory : *This is a E-Commerce Cryptocurrency Market Application where one can purchase and sell their Cryptocurrency. This programme also perform add, delete, update operation and uses Sqlite3 as database of the Application.*

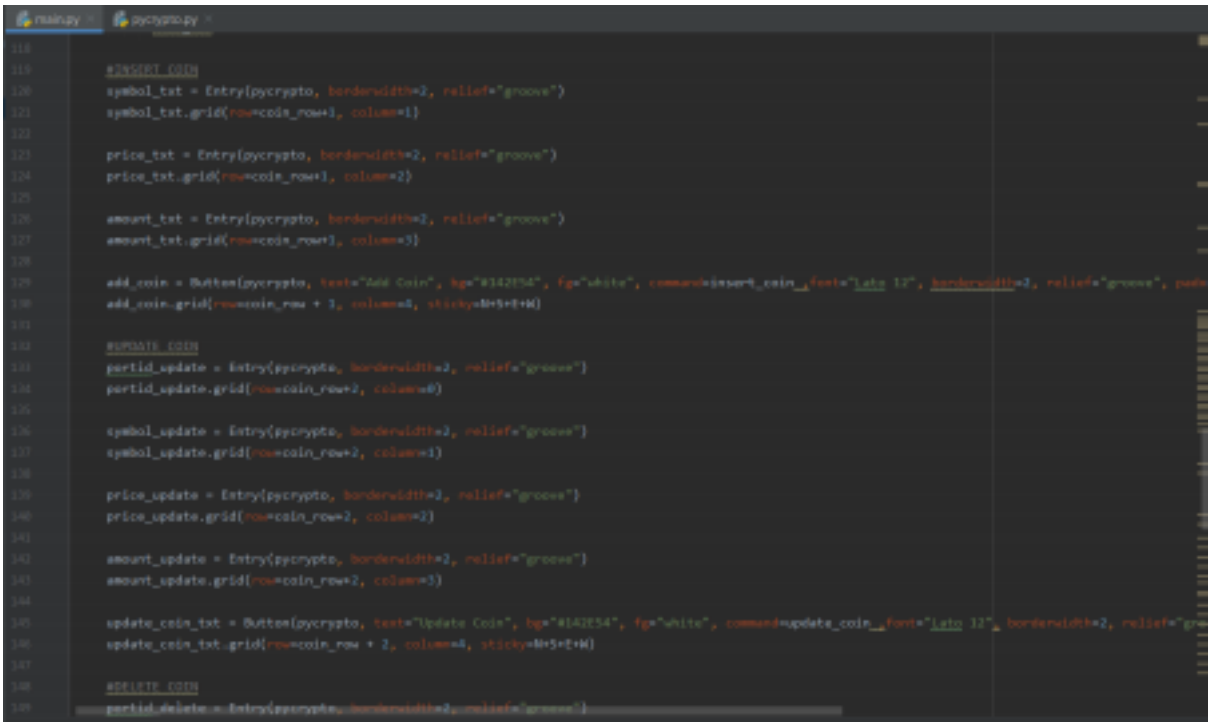
Code:

Main.py



```
def close_app():
    pycrypto.destroy()

menu = Menu(pycrypto)
file_item = Menu(menu)
file_item.add_command(label='Clear Portfolio', command=clear_all)
file_item.add_command(label='Close App', command=close_app)
menu.add_cascade(label="File", menu=file_item)
pycrypto.config(menu=menu)

def my_portfolio():
    api_request = requests.get("https://pro-api.coinmarketcap.com/v1/cryptocurrency/listings/latest?start=1&limit=5000&convert=USD&CMC_PRO_API_KEY=3B
    api = json.loads(api_request.content)

    cursorObj.execute("SELECT * FROM coin")
    coins = cursorObj.fetchall()

    def font_color(amount):
        if amount >= 0:
            return "green"
        else:
            return "red"

    def insert_coin():
        cursorObj.execute("INSERT INTO coin(symbol, price, amount) VALUES(?, ?, ?)", (symbol_txt.get(), price_txt.get(), amount_txt.get()))
        con.commit()

        messagebox.showinfo("Portfolio Notification", "Coin Added To Portfolio Successfully!")
        reset()

    def update_coin():
        cursorObj.execute("UPDATE coin SET symbol=?, price=?, amount=? WHERE id=?", (symbol_update.get(), price_update.get(), amount_update.get(),
```

```
    def update_coin():
        cursorObj.execute("UPDATE coin SET symbol=?, price=?, amount=? WHERE id=?", (symbol_update.get(), price_update.get(), amount_update.get(),
        con.commit()

        messagebox.showinfo("Portfolio Notification", "Coin Updated Successfully!")
        reset()

    def delete_coin():
        cursorObj.execute("DELETE FROM coin WHERE id=?", (portid_delete.get(),))
        con.commit()

        messagebox.showinfo("Portfolio Notification", "Coin Deleted From Portfolio")
        reset()

    total_pl = 0
    coin_row = 1
    total_current_value = 0
    total_amount_paid = 0

    for i in range(0, 300):
      for coin in coins:
        if api["data"][i]["symbol"] == coin[1]:
          total_paid = coin[2] * coin[3]
          current_value = coin[2] * api["data"][i]["quote"]["USD"]["price"]
          pl_percoin = api["data"][i]["quote"]["USD"]["price"] - coin[3]
          total_pl_coin = pl_percoin * coin[2]

          total_pl += total_pl_coin
          total_current_value += current_value
          total_amount_paid += total_paid

          portfolio_id = Label(pycrypto, text=coin[0], bg="#F3F4F6", fg="black", font="Lato 12", borderwidth=2, relief="groove", padx="3", pady="3")
```



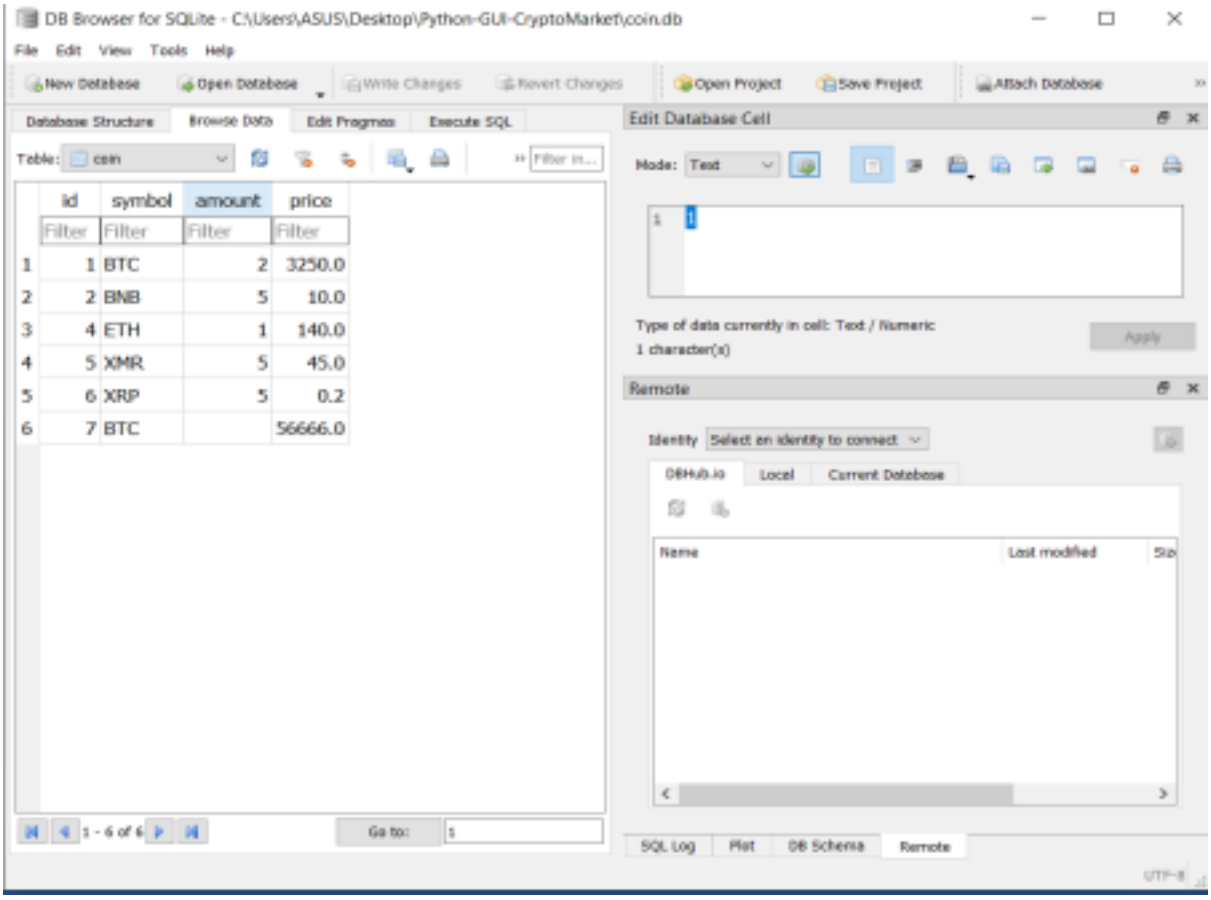
```
    #INSERT COIN
    symbol_txt = Entry(pycrypto, borderwidth=2, relief="groove")
    symbol_txt.grid(row=coin_row+3, column=1)

    price_txt = Entry(pycrypto, borderwidth=2, relief="groove")
    price_txt.grid(row=coin_row+1, column=2)

    amount_txt = Entry(pycrypto, borderwidth=2, relief="groove")
    amount_txt.grid(row=coin_row+1, column=3)

    add_coin = Button(pycrypto, text="Add Coin", bg="#142E54", fg="white", command=insert_coin, font="Lato 12", borderwidth=2, relief="groove", padx
    add_coin.grid(row=coin_row + 1, column=4, sticky=N+S+E+W)

    #UPDATE COIN
    portid_update = Entry(pycrypto, borderwidth=2, relief="groove")
    portid_update.grid(row=coin_row+2, column=0)

    symbol_update = Entry(pycrypto, borderwidth=2, relief="groove")
    symbol_update.grid(row=coin_row+2, column=1)

    price_update = Entry(pycrypto, borderwidth=2, relief="groove")
    price_update.grid(row=coin_row+2, column=2)

    amount_update = Entry(pycrypto, borderwidth=2, relief="groove")
    amount_update.grid(row=coin_row+2, column=3)

    update_coin_txt = Button(pycrypto, text="Update Coin", bg="#142E54", fg="white", command=update_coin, font="Lato 12", borderwidth=2, relief="gr
    update_coin_txt.grid(row=coin_row + 2, column=4, sticky=N+S+E+W)

    #DELETE COIN
    portid_delete = Entry(pycrypto, borderwidth=2, relief="groove")
```

```
    portfolio_id = Label(pycrypto, text="Portfolio ID", bg="#142E54", fg="white", font="Lato 12 bold", padx="5", pady="5", borderwidth=2, relief="gr
    portfolio_id.grid(row=0, column=0, sticky=N+S+E+W)

    name = Label(pycrypto, text="Coin Name", bg="#142E54", fg="white", font="Lato 12 bold", padx="5", pady="5", borderwidth=2, relief="groove")
    name.grid(row=0, column=1, sticky=N+S+E+W)

    price = Label(pycrypto, text="Price", bg="#142E54", fg="white", font="Lato 12 bold", padx="5", pady="5", borderwidth=2, relief="groove")
    price.grid(row=0, column=2, sticky=N+S+E+W)

    no_coins = Label(pycrypto, text="Coin Owned", bg="#142E54", fg="white", font="Lato 12 bold", padx="5", pady="5", borderwidth=2, relief="groove")
    no_coins.grid(row=0, column=3, sticky=N+S+E+W)

    amount_paid = Label(pycrypto, text="Total Amount Paid", bg="#142E54", fg="white", font="Lato 12 bold", padx="5", pady="5", borderwidth=2, relief
    amount_paid.grid(row=0, column=4, sticky=N+S+E+W)

    current_val = Label(pycrypto, text="Current Value", bg="#142E54", fg="white", font="Lato 12 bold", padx="5", pady="5", borderwidth=2, relief="gr
    current_val.grid(row=0, column=5, sticky=N+S+E+W)

    pl_coin = Label(pycrypto, text="P/L Per Coin", bg="#142E54", fg="white", font="Lato 12 bold", padx="5", pady="5", borderwidth=2, relief="groove"
    pl_coin.grid(row=0, column=6, sticky=N+S+E+W)

    totalpl = Label(pycrypto, text="Total P/L With Coin", bg="#142E54", fg="white", font="Lato 12 bold", padx="5", pady="5", borderwidth=2, relief="
    totalpl.grid(row=0, column=7, sticky=N+S+E+W)

app_nav()
app_header()
my_portfolio()
pycrypto.mainloop()

cursorObj.close()
con.close()
```

```python
import requests
import json

api_request = requests.get("https://pro-api.coinmarketcap.com/v1/cryptocurrency/listings/latest?start=1&limit=5&convert=USD&CMC_PRO_API_KEY=1889ce05-
api = json.loads(api_request.content)

print("----------------")
print("----------------")

coins = [
  {
    "symbol":"BTC",
    "amount_owned": 2,
    "price_per_coin": 3200
  },
  {
    "symbol":"EOS",
    "amount_owned": 100,
    "price_per_coin": 2.05
  }
]

total_pl = 0

for i in range(0, 5):
  for coin in coins:
    if api["data"][i]["symbol"] == coin["symbol"]:
      total_paid = coin["amount_owned"] * coin["price_per_coin"]
      current_value = coin["amount_owned"] * api["data"][i]["quote"]["USD"]["price"]
      pl_percoin = api["data"][i]["quote"]["USD"]["price"] - coin["price_per_coin"]
      total_pl_coin = pl_percoin * coin["amount_owned"]
```

```python
  }
]

total_pl = 0

for i in range(0, 5):
  for coin in coins:
    if api["data"][i]["symbol"] == coin["symbol"]:
      total_paid = coin["amount_owned"] * coin["price_per_coin"]
      current_value = coin["amount_owned"] * api["data"][i]["quote"]["USD"]["price"]
      pl_percoin = api["data"][i]["quote"]["USD"]["price"] - coin["price_per_coin"]
      total_pl_coin = pl_percoin * coin["amount_owned"]

      total_pl = total_pl + total_pl_coin

      print(api["data"][i]["name"] + " - " + api["data"][i]["symbol"])
      print("Price - ${0:.2f}".format(api["data"][i]["quote"]["USD"]["price"]))
      print("Number Of Coin:", coin["amount_owned"])
      print("Total Amount Paid:", "${0:.2f}".format(total_paid))
      print("Current Value:", "${0:.2f}".format(current_value))
      print("P/L Per Coin:", "${0:.2f}".format(pl_percoin))
      print("Total P/L With Coin:", "${0:.2f}".format(total_pl_coin))
      print("----------------")

print("Total P/L For Portfolio:", "${0:.2f}".format(total_pl))
```

Database:



Output: