علی ناصری نیا
۴۰۱۳۱۰۵۳

راه اندازی پایگاه داده به وسیله کتابخانه psycopg2 در پایتون انجام شد.

کتابخانه termcolor را برای چاپ رنگی در خروجی نصب داشته باشید.

در فایل config.py و connect.py به پایگاه داده متصل میشویم و آبجکت connect در فایل cli.py ذخیره میشود و به وسیله آن به cursor دست میابیم.

دستور ساخت جداول در فایل CreatTableCommand وجود دارد و در فایل cli آن ها را میسازیم.

شرط ساخت جدول این است که قبلا وجود نداشته باشد بنابراین اگر از قبل دیتایی وجود داشته باشد آنها پاک نمیشوند.

دستور ها با argparser هندل شده مانند پروژه یک

تمام عملیات ها به صورت یک تابع در Actions.py قرار دارد.

Register

```python
def register(self, ID, studentID, major, birth_date, first_name, last_name, balance):
    try:

        self.cursor.execute(f"INSERT INTO students(ID, studentID, major, birth_date, first_name, last_name, balance) VALUES('{ID}',
        ok("Student Registrated Successfully")

    except psycopg2.errors.UniqueViolation as e:
        war("national ID or studentID is already exists")

    except psycopg2.errors.CheckViolation as e:
        if str(e).find("studentid") != -1:
            war("studentID should have 7 or 8 digits")
        elif str(e).find("id") != -1:
            war("ID should have 10 digits")
        elif str(e).find("balance") != -1:
            war("balance should be positive")

    except psycopg2.errors.StringDataRightTruncation:
        war("values length of id and first_name and last_name and major should be less than 20")

    except psycopg2.errors.DatetimeFieldOverflow:
        war("value of birth date should like this format yyyy-mm-dd")

    except Exception as e:
        war(e)
```

register

اطلاعات را insert میکند و سپس اگر خطایی از جمله تکراری بودن کدملی و شماره دانشجویی و تعداد ارقام و فرمت تاریخ تولد و منفی نبودن موجودی را ارایه میدهد.

Remove

```python
def remove(self, ID):
    self.cursor.execute(f"SELECT * FROM students WHERE ID = '{ID}'")

    if self.cursor.rowcount == 0:
        war("No students with this national ID were found")
        return

    data = self.cursor.fetchall()[0]
    for i in range(7):
        detail(ut.studentDetails[i], data[i])
    print()

    war("Are you sure to remove student with this ditails? [Y/n]")
    x = input()
    if x == "Y":
        self.cursor.execute(f"DELETE FROM students WHERE ID = '{ID}'")
        if self.cursor.rowcount == 1:
            ok("removed successfully")
        else:
            war("error removing student")
    else:
        war("remove operation cancelled")
```

ابتدا برسی میکند دانشجویی با چنین کد ملی وجود دارد بعد اگر وجود داشت تاییدیه میگیرد که آن را حذف کند یا نه

با حذف دانشجو با پیام موفقیت یا عدم موفقیت مواجه میشویم.

credit

```
def credit(self, studentID, money):
    if int(money) <= 0:
        war("value of money should be positive")
        return

    self.cursor.execute(f"UPDATE students SET balance = balance + {money} WHERE studentID = '{studentID}'")
    if self.cursor.rowcount == 0:
        war("Error: studentID not found")
    else:
        ok("credit charged successfully")
```

credit

افزایش شماره حساب با گرفتن شماره دانشجویی

پول باید مثبت باشد

add

```
def add(self, name, date, price, inventory):
    try:
        self.cursor.execute(f"INSERT INTO foods(name, date, price, inventory) VALUES('{name}', '{date}', '{price}', '{inventory}') R
        ID = self.cursor.fetchone()[0]
        ok("food added successfully with ID = " + str(ID))
    except Exception as e:
        war(e)
```

add food

برای اضافه کردن غذا مانند اضافه کردن دانشجو عمل می کنیم

delete

```
def delete(self, ID):
    try:
        self.cursor.execute(f"DELETE FROM foods WHERE ID = '{ID}' AND inventory = 0")
        if self.cursor.rowcount == 0:
            war(f"there is no food with ID '{ID}' and inventory = 0")
        else:
            ok("food with ID = " + ID + " removed successfully")
    except Exception as e:
        war(e)
```

delete food

اگر مقدار غذا inventory صفر شده باشد شما اجازه حذف غذا را دارید.

Reserve

```python
def reserve(self, studentID, foodID):
    try:
        self.checkReservePossibelity(studentID, foodID)
        self.cursor.execute(f"INSERT INTO reservations(studentID, foodID) VALUES('{studentID}', '{foodID}') RETURNING ID")
        ID = self.cursor.fetchone()[0]
        ok("food reserved with ID = " + str(ID))
        self.reserveOpt(studentID, foodID, '-')
        self.cursor.execute(f"INSERT INTO transactions(SRCreservationID, DSTreservationID) VALUES(NULL, {ID})")
    except Exception as e:
        war(e)
```

reserve food

ابتدا چک میکنیم که موجودی دانشجو و موجودی غذا داریم یا نه

اگر عملیات موفقیت آمیز باشد شماره آیدی رزور غذا را به عنوان خروجی می دهیم.

بعد عملیات تراکنش مالی را انجام می دهیم.

change

```python
def change(self, SRC, DST):
    SRC = self.setNull(SRC)
    DST = self.setNull(DST)

    try:
        self.checkReserveID(SRC, 'SRC')
        self.checkReserveID(DST, 'DST')
        self.cursor.execute(f"INSERT INTO transactions(SRCreservationID, DSTreservationID) VALUES({SRC}, {DST})")
        self.updateIsReserved(SRC, '+')
        self.updateIsReserved(DST, '-')

    except Exception as e:
        war(e)
```

change reserve

ابتدا مقدار src و dst اگر none باشد را 'null' میکنیم.

حالا برسی میکنیم که آیا رزور با این آیدی وجود دارد یا خیر.

اگر درست بود isReserved هر رزور را آپدیت میکنیم.

DeleteDB

```python
def deleteDB(self):
    for table in Actions.TABLE_NAMES:
        self.cursor.execute(f"DROP TABLE IF EXISTS {table} CASCADE")
```

delete all talbe in database

با این کار تمام جداول دیتابیس را پاک میکنیم.

showDB

```python
def showDB(self):
    print()
    for table in Actions.TABLE_NAMES:
        cprint(table, "magenta", attrs=["bold"])
        cprint("=" * len(table), "yellow")
        self.cursor.execute(f"SELECT * FROM {table}")
        rows = self.cursor.fetchall()
        for row in rows:
            cprint(row, "cyan")
        print()
```

show all table

تمام جداول دیتابیس را نشان میدهد. با رنگ :)