**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KHOA HỌC MÁY TÍNH**

**A blue logo with a black background

Description automatically generated**

**BÁO CÁO ĐỒ ÁN**

**LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG - IT002.O21.TTNT**

**ĐỀ TÀI**

**TENGEE – APP QUẢN LÝ BÁN KHOÁ HỌC**

Giảng viên hướng dẫn ThS. Nguyễn Thanh Sơn

Sinh viên thực hiện: Huỳnh Việt Tiến - 23521770

**TP. Hồ Chí Minh, tháng 06 năm 2024**

**MỤC LỤC NỘI DUNG**

[**I.** **GIỚI THIỆU ĐỒ ÁN** 4](#_Toc168591087)

[**1.** **NỘI DUNG VÀ YÊU CẦU** 4](#_Toc168591088)

[**1.1** **Nội dung** 4](#_Toc168591089)

[1.2 **Yêu cầu** 4](#_Toc168591090)

[**2.** **SẢN PHẨM** 5](#_Toc168591091)

[**2.1**  **Link github** 5](#_Toc168591092)

[**II.** **QUÁ TRÌNH THỰC HIỆN** 5](#_Toc168591093)

[1. **Tuần 1:** 5](#_Toc168591094)

[2. **Tuần 2:** 6](#_Toc168591095)

[3. **Tuần 3:** 6](#_Toc168591096)

[4. **Tuần 4:** 7](#_Toc168591097)

[5. **Kết quả đạt được** 8](#_Toc168591098)

[6. **Sản phẩm cuối cùng** 14](#_Toc168591099)

[7. **Tài liệu tham khảo** 14](#_Toc168591100)

[**III.** **PHỤ LỤC DOCSTRING** 14](#_Toc168591101)

[DataManager: 15](#_Toc168591102)

[\_\_init\_\_(): 15](#_Toc168591103)

[insert\_data(type, data): 15](#_Toc168591104)

[fetch\_data(type: bool): 15](#_Toc168591105)

[get\_data(data\_type): 15](#_Toc168591106)

[update\_data(data\_type, data): 15](#_Toc168591107)

[find\_data(self, typeData, id): 15](#_Toc168591108)

[delete(self, data, type): 15](#_Toc168591109)

[getID(self): 15](#_Toc168591110)

[Data: 16](#_Toc168591111)

[\_\_init\_\_(id, data\_type): 16](#_Toc168591112)

[Course(Data): 16](#_Toc168591113)

[\_\_init\_\_(id, data\_type, title, author, description, price, oldprice, image) 16](#_Toc168591114)

[get\_this\_course() 16](#_Toc168591115)

[User(Data) 16](#_Toc168591116)

[\_\_init\_\_(id, data\_type, accountname, password, username, role) 16](#_Toc168591117)

[get\_this\_user() 16](#_Toc168591118)

[Teacher(User): 17](#_Toc168591119)

[\_\_init\_\_(id, data\_type, accountname, password, username, role) 17](#_Toc168591120)

[insert\_bank(self, bank\_name, card\_number, recipient): 17](#_Toc168591121)

[fetch\_bank(self): 17](#_Toc168591122)

[Bank: 17](#_Toc168591123)

[\_\_init\_\_(self, bank\_name, recipient, cardnumber): 17](#_Toc168591124)

[get\_this\_bank(self): 17](#_Toc168591125)

[class Student(User): 17](#_Toc168591126)

[\_\_init\_\_(self, id, data\_type, accountname, password, username, role): 17](#_Toc168591127)

1. **GIỚI THIỆU ĐỒ ÁN**
2. **NỘI DUNG VÀ YÊU CẦU**
   1. **Nội dung**
3. **Mục đích:**

* Ứng dụng kiến thức của môn học Lập trình hướng đối tượng;
* Chạy thử nghiệm ý tưởng để giải quyết các vấn đề thực tiễn trong đời sống;

1. **Ý tưởng:**

* Tạo dựng một ứng dụng dùng để quản lý đăng bán các khoá học online. Tạo ra sự tương tác giữa người cung cấp cấp khoá học với vai trò là Teacher (giáo viên) và ngiời mua khoá học là Student (học sinh)

1. **Ngôn ngữ lập trình, công cụ lập trình:**

* Công cụ lập trình: Visual Studio Code, Qt Designer
* Ngôn ngữ lập trình: Python
  1. **Yêu cầu**

1. **Yêu cầu chung:**

**-** phụ lụcdocstring cho từng lớp và phương thức được liệt kê trong sơ đồ lớp (UML) ✔

- có tính hướng đối tượng, kế thừa, đóng gói ✔

- có bản thiết kế UML ✔

- Code python theo bản thiết kế UML ✔

1. **Yêu cầu riêng:**

* **hệ thống đăng ký/ đăng nhập**

Chia người dùng thành 2 loại là học sinh/ giáo viên (student/ teacher) ✔

* **Trang cá nhân (Profile Page)**

Tất cả người dùng cung cấp thông tin cơ bản bao gồm: ✔

* accountname (tên tài khoản – const)
* username (tên người dùng)
* password (mật khẩu)

Teacher cung cấp thêm thông tin ***tài khoản ngân hàng*** vào **hệ thống profile** mới có thể tạo khoá học. ✔

* **Trang chủ (Home Page)**

Trang home yêu cầu hiện thị được các tính năng cơ bản: hiện khóa học từ dữ liệu, tìm kiếm khoá học, hiện các nút chuyển đổi giao diện, cửa sổ ✔

* **Trang quản lý khoá học (Courses Management Page)**
* Chỉ hiện thị với giao diện giáo viên ✔
* Cung cấp các tính năng quản lí cơ bản bao gồm đăng, sửa, và xoá khoá học✔
* **Trang Thanh toán (Purchasing Page)**
* Cung cấp màn hình thông tin ngân hàng của tác giả khoá học ✔
* Thiết kế chức năng thông báo đơn hàng đến giáo viên
* Thiết kế chức năng thông báo đơn hàng đã được chấp nhận đến học sinh
* **Trang khoá học (Purchasing Page)**
* Hiện thị các khoá học đã mua và đã được xác nhận từ giáo viên cho học sinh

1. **SẢN PHẨM**

**2.1 Link github**

Link: <https://github.com/SharkTien/Tengee>

1. **QUÁ TRÌNH THỰC HIỆN**
2. **Tuần 1:**

* Các thư viện và module có sẵn được sử dụng trong tuần 1:
* time: dùng cho màn hình loading
* pyqt5 (qtcore,qt,qtwidgets, uic, qmainwindow,…): dùng cho tổng thể giao diện
* Thiết kể màn hình loading
* Tạo hệ thống logic đăng nhập, đăng ký tài khoản
* Tạo cây thư mục dùng để lưu trữ các tệp dữ liệu
* Thiết kế cửa sổ đăng ký đăng nhập
* Thêm các điều kiện nhập khi đăng ký và kiểm tra độ chính xác khi đăng nhập.
* Tính năng remember me, ghi nhớ người dùng khi đóng ứng dụng
* Thiết kế màn hình home
* Thiết kế card khóa học
* Cách hiển thị dữ liệu khoá học:
* Đọc tệp dữ liệu lưu khoá học đưa vào cấu trúc danh sách (list) dưới dạng các object Course bao gồm các thuộc tính cơ bản như tiêu đề, tác giả, mô tả, giá và hình ảnh.
* Sử dụng thư viện giao diện tạo một Scrollbox (hộp cuộn), sử dụng vòng for để lần lượt đưa các object Course từ danh sách vào Scrollbox dưới dạng card khoá học.
* Thêm tính năng tìm kiếm thông tin khóa học (bao gồm tìm theo tiêu đề, tác giả và mô tả)
* Thêm các nút phân loại có sẵn để lọc khóa học theo từ khoá
* Thiết kế màn hình profile hiển thị thông tin người dùng và thêm các chức năng tương tác thông tin tài khoản bao gồm sửa tên, sửa mật khẩu, lưu thay đổi và đăng xuất
* Thiết kế cửa sổ xác nhận thoát ứng dụng Yes/ No nhằm tránh trường hợp ấn nhầm nút tắt làm gián đoạn tác vụ đang thực hiện
* Thiết kế sơ đồ lớp UML cơ bản
* Fix Bug:
* Thêm tính năng vô hiệu hoá tương tác của người dùng khi hộp thoại xác nhận thoát ứng dụng xuất hiện, chỉ khi xác nhận thì mới huỷ vô hiệu hoá.

1. **Tuần 2:**

* Cải thiện sự rõ ràng của tính hướng đối tượng, kế thừa và đóng gói. Theo đánh giá và nhận xét của gv hướng dẫn, lượt bỏ, bỏ qua các lớp UML không cần thiết của giao diện UI, thư viện và từ đó thiết kế lại sơ đồ UML đáp ứng được mục đích chính của vấn đề.

Xây dựng lại vấn đề: Tạo một **cơ sở quản lí** dữ liệu. Dữ liệu được chia làm 2 loại là người dùng và khoá học và đều được đánh **mã số ID**. Mỗi **người dùng** sẽ có **tên tài khoản**, **mật khẩu**, **tên người dùng**, **vai trò**. Mỗi **khoá học** sẽ có **tên khoá học**, **tác giả**, **mô tả**, **giá**, **thumbnail**. Biết người dùng chia tiếp thành 2 vai trò là giáo viên hoặc là học sinh. Giáo viên sẽ bao gồm thông tin về tài khoản ngân hàng (tên ngân hàng/ví điện tử, tên chủ thẻ, số tài khoản) và danh sách các khoá học đã tạo. Học sinh sẽ bao gồm thông tin các khoá học đã mua.

Bổ sung file data\_init.py chứa tất cả các class mô tả vấn đề như trên ứng dụng tính đóng gói (public, private), tính kế thừa và quan hệ 1…\*, \*…1, 1…1, \*…\* và docstring

Chỉnh sửa nâng cấp tệp python ui để chạy với tệp data\_init.py

1. **Tuần 3:**

* Cập nhật thay đổi sơ đồ lớp:
* Giới hạn quan hệ giữa lớp Teacher với lớp Bank (ngân hàng) từ 1..\* thành 1..1
* Thêm và chỉnh các phương thức chính của cấu trúc dữ liệu bao gồm Fetch (nhập dữ liệu), Insert (thêm), Get (nhận dữ liệu), update (thay đổi thông tin tệp dữ liệu), find (tìm kiếm), delete (xoá) và hàm getID tăng giá trị của ID
* Gộp khoá học đã tạo của giáo viên và khoá học của học sinh làm một đưa vào lớp cha, nếu dữ liệu vai trò là 1 thì là danh sách tạo, dữ liệu vai trò là 0 thì là danh sách mua.
* Thêm chức năng cập nhật, thêm thông tin ngân hàng cho giáo viên insert\_bank vào lớp Teacher. Sau khi thêm thì mở khoá nút tạo khoá học (ban đầu vô hiệu hoá)
* Thiết kế màn hình quản lý khoá học của giáo viên

Cách hiển thị các khoá học đã tạo: Danh sách sở hữu khoá học của giáo viên được lưu trữ dưới dạng danh sách ID với key là id của giáo viên. Sử dụng phương thức tìm kiếm ở cấu trúc dữ liệu DataManager để tìm kiếm ID giáo viên, rồi hiển thị giá trị của thuộc tính data\_courses. Sau đó tương tự như cách hiện thị dữ liệu khoá học.

* + Thiết kế màn hình edit khoá học.
  + Thiết kế chức năng tạo, chỉnh sửa và xoá khoá học.
  + Nghiên cứu cách đăng ảnh lên ứng dụng khi người dùng sử dụng ảnh trong đường dẫn local để làm ảnh của khoá học: thư viện shutil, hàm shutil.copy(src, dct) thực hiện sao chép thư mục ảnh từ folder local của user vào thư mục src của ứng dụng.
  + Màn hình enrollment hiện thông tin chi tiết khoá học.
  + Màn hình xác nhận thanh toán khoá học.

1. **Tuần 4:**
2. **Kết quả đạt được**
   1. Các bản thiết kế

* A screenshot of a computer

  Description automatically generatedMàn hình loading
* Cửa sổ đăng nhập

A computer screen shot of a person's hand

Description automatically generated

Figure 1Màn hình đăng nhập

A computer screen shot of a person's hands

Description automatically generatedA screen shot of a computer

Description automatically generated

Figure 2 Tab đăng ký

Figure 3 Tooltip điều kiện nhập

* Thiết kế thẻ khoá học

A screenshot of a computer

Description automatically generated

* Màn hình trang chủ

A screenshot of a computer

Description automatically generated

Figure 4 Home page

* Màn hình thông tin cá nhân

A screenshot of a computer

Description automatically generated

Figure 5 Profile settings page

* Màn hình quản lý khoá học đã tạoA screenshot of a computer

  Description automatically generated

Figure 6 Màn hình quản lý khoá học đã tạo

* Màn hình Sửa thông tin khoá học

A screenshot of a computer

Description automatically generated

Figure 7 Courses Edit page < Edit button

* Màn hình tạo khoá học

A screenshot of a computer

Description automatically generated

Figure 8 Create Course Page > Add course button

* Màn hình hiển thị chi tiết thông tin khoá học

A screenshot of a computer

Description automatically generated

* A screenshot of a computer

  Description automatically generatedMàn hình thanh toán khoá học <- Buy now button
* Màn hình thông báo
* Thông báo các đơn đăng ký khoá học cho giáo viên

Giáo viên sẽ kiểm tra đã nhận được chuyển khoản từ học sinh hay không. Để kiểm tra thì giáo viên sẽ vào tài khoản ngân hàng, xem lịch sử giao dịch có giao dịch nào từ tên tài khoản của học sinh với số tiền chuyển đúng với giá trong đơn thông báo

Nếu đồng ý thì ấn vào Accept để chuyển thông tin khoá học về cho học sinh, nếu không thì huỷ đơn khoá học. Cả hai đều gửi thông báo về học sinh.

A screenshot of a group of women

Description automatically generated

* Màn hình khoá học của học sinh

A screenshot of a computer

Description automatically generatedHiển thị các khoá học đã mua của học sinh sau khi được giáo viên accept

* A screenshot of a computer

  Description automatically generatedCửa sổ xác nhận thoát khỏi ứng dụng
* A computer screen shot of a computer

  Description automatically generatedUML class diagram

1. **Sản phẩm cuối cùng**
2. **Tài liệu tham khảo**

Đề tài NCKH: <https://github.com/hoangm960/Astraea>

1. **PHỤ LỤC DOCSTRING**

## DataManager:

        Class DataManager initiates list users = [], list courses = [] with basic method

        method:

            + \_\_init\_\_()

            + fetch\_data(type: bool)

            + insert\_data(type:bool)

            + get\_data(type:bool)

### \_\_init\_\_():

initiates original variables and const value

insert\_data(type: bool, data: list): insert a data (user/ course based on [type]) into user list/ course list

            data is a list with format: [accountname, password, username, role]

### insert\_data(type, data):

insert\_data(type: bool, data: list): insert a data (user/ course based on [type]) into user list/ course list. Data is a list with format: [id, type, accountname, password, username, role]

### fetch\_data(type: bool):

get data from data file and add into user list/ course list

            user: [id, 1, accountname, password, username, role]

            course: [id, 0, title, author, description, price, oldprice, thumbnails url]

### get\_data(data\_type):

return list of users or courses based on [data\_type]

### update\_data(data\_type, data):

update\_data(data\_type): update list of users or courses based on [data\_type]

### find\_data(self, typeData, id):

            find\_data(typeData, id): find data from data users [1] and data courses [0] based on [typeData] and [id].

            If type = -1, find accountname of users

### delete(self, data, type):

           delete(data, type): function delete account or course based on [type] (bool)

### getID(self):

          getID(self): create new object's ID

## Data:

class Data: generate a data storaging id and data\_type, which is the familiar attributes of Course and User data.

method:

            \_\_init\_\_(id, data\_type)

### \_\_init\_\_(id, data\_type):

initiates original variables

## Course(Data):

class Course(Data): generate a data storaging title, author, description, price, oldprice and image then inherites id and data type

method:

            \_\_init\_\_(id, data\_type, title, author, description, price, oldprice, image)

            get\_this\_course()

### \_\_init\_\_(id, data\_type, title, author, description, price, oldprice, image)

\_\_init\_\_(id, data\_type, title, author, description, price, oldprice, image): initiates original variables and inherites variables from parent classes

### get\_this\_course()

get\_this\_course(): return a dictionary with keys including: title, author, description, price, oldprice and image

## User(Data)

class User(Data): generate a data storaging accountname, password, username and role then inherites id and data type

method:

            \_\_init\_\_(id, data\_type, accountname, password, username, role)

Update\_courses()

            get\_this\_user()

change\_password(data)

change\_username(data)

### update\_courses(self):

            update\_courses(): update the list courses of users

### \_\_init\_\_(id, data\_type, accountname, password, username, role)

\_\_init\_\_(id, data\_type, accountname, password, username, role): initiates original variables and inherites variables from parent classes

### get\_this\_user()

get\_this\_user(): return a dict {id, accountname, password, username, role}

## Teacher(User):

        class Teacher: generate a data storaging bank\_account (-> Bank) then inherites id, data type, accountname, password, username and role

        method:

            \_\_init\_\_(id, data\_type, accountname, password, username, role)

            insert\_bank(bankname, card\_number, recipient)

fetch\_bank()

### \_\_init\_\_(id, data\_type, accountname, password, username, role)

\_\_init\_\_(id, data\_type, accountname, password, username, role): initiates original variables and inherites variables from parent classes

### insert\_bank(self, bank\_name, card\_number, recipient):

            insert\_bank(bank\_name, card\_number, recipient): insert new bank value to this Teacher account

### fetch\_bank(self):

            fetch\_bank(self): get bank account data

## Bank:

            class Bank: generate a series data for bank account

            method:

                \_\_init\_\_(bank\_name, cardholder, cardnumber)

get\_this\_bank()

### \_\_init\_\_(self, bank\_name, recipient, cardnumber):

    \_\_init\_\_(bank\_name, cardholder, cardnumber): initiate attributes

### get\_this\_bank(self):

    get\_this\_bank(): return a dictionary of Bank information

## class Student(User):

  class Student(User): inherites id, data type, accountname, password, username and role

        method:

            \_\_init\_\_(id, data\_type, accountname, password, username, role)

### \_\_init\_\_(self, id, data\_type, accountname, password, username, role):

            \_\_init\_\_(id, data\_type, accountname, password, username, role): inherites and initiates attributes