



Topic : Online App Store

Group no : MLB_7.1_6

Campus : Malabe

Submission Date : 18.05.2022

We declare that this is our own work, and this Assignment does not incorporate without acknowledgment any material previously submitted by anyone else in SLIIT or any other university/Institute. And we declare that each one of us equally contributed to the completion of this Assignment.

Registration No	Name	Contact Number
IT21158186	A.R.D. Pinsara	076 937 9089
IT21167614	D.N. Mahagedara	076 335 5762
IT21158636	R.M.U.P.D. Rathnayake	071 331 1593
IT21377662	D.C. Herath	075 040 6427
IT21346668	D.A.S. Vimansa	071 012 5385

1) List down the requirements you have identified, for the system you need to implement. There should be at least 10 different requirements excluding the user login.

1. Before Downloading any App, the User should provide user details and relevant details to the Registration form and register to the system.
2. An unregistered user can register as a usual user or either Developer.
3. After the registration of a developer, the registration request will be sent to the admins, and they can review the request and accept it or not.
4. After the registration usual user/ developer can Login to the system.
5. All the apps available in the system are categorized as the type that provided by the developer.
6. A usual user can download any app that available in the system without doing any payment.
7. There can be several admins and they can manage users, apps, and the developers in the system by using admin panel.
8. An admin has admin name, password, email, admin id.
9. Developers can upload their apps to the system with the required details.
10. One app can have details like App Name, Category, Description, Developer name and more.
11. A user can add reviews and rate the apps available.
12. A Developer can reply to the reviews available for their app.
13. Site owner can manage admins available in the system.

Noun/ Verb Analysis

- Nouns

- Verb

1. Before downloading any **app**, the **user** should **provide** user details and relevant details to the registration form and **register** to the **system**.
2. An **unregistered user** can register as a usual user or either **developer**.
3. After the registration of a developer, the registration request will be sent to the **admins**, and they can **review** that registration request and **accept** it or not.
4. After the registration of usual user/ developer can **login** to the system.
5. All the apps available in the system are **categorized** as the type that provided by the developer.
6. A usual user can **download** any app that available in the system without doing any payment.
7. There can be several admins and they can **manage** users, apps, and developers in the system by using admin panel.
8. An admin has **admin name**, **password**, **email** and an **admin id**.
9. Developers can **upload** their apps to the system with the required details.
10. One app can have details like **app name**, **category**, **description**, **developer name** and more.
11. A user can **add reviews** and **rate** the apps available.
12. A developer can **reply** to the **reviews** available for their app.
13. **Site owner** can manage admins available in the system.

Identifying classes using noun/ verb analysis

- App - class
- User - class
- System - out of scope
- unregistered user - out of scope
- developer - class
- admin - class
- review - class
- Site owner -not a class (Only have login credential)
- admin name - attributes
- password - attributes
- email - attributes
- admin id - attributes
- app name - attributes
- category - class
- description - attributes
- developer name - attributes

CRC Cards

App	
Responsibilities	Collaboration
Keep app details and requirements	
Keep developer details	Developer

User	
Responsibilities	Collaborations
Keep user details	

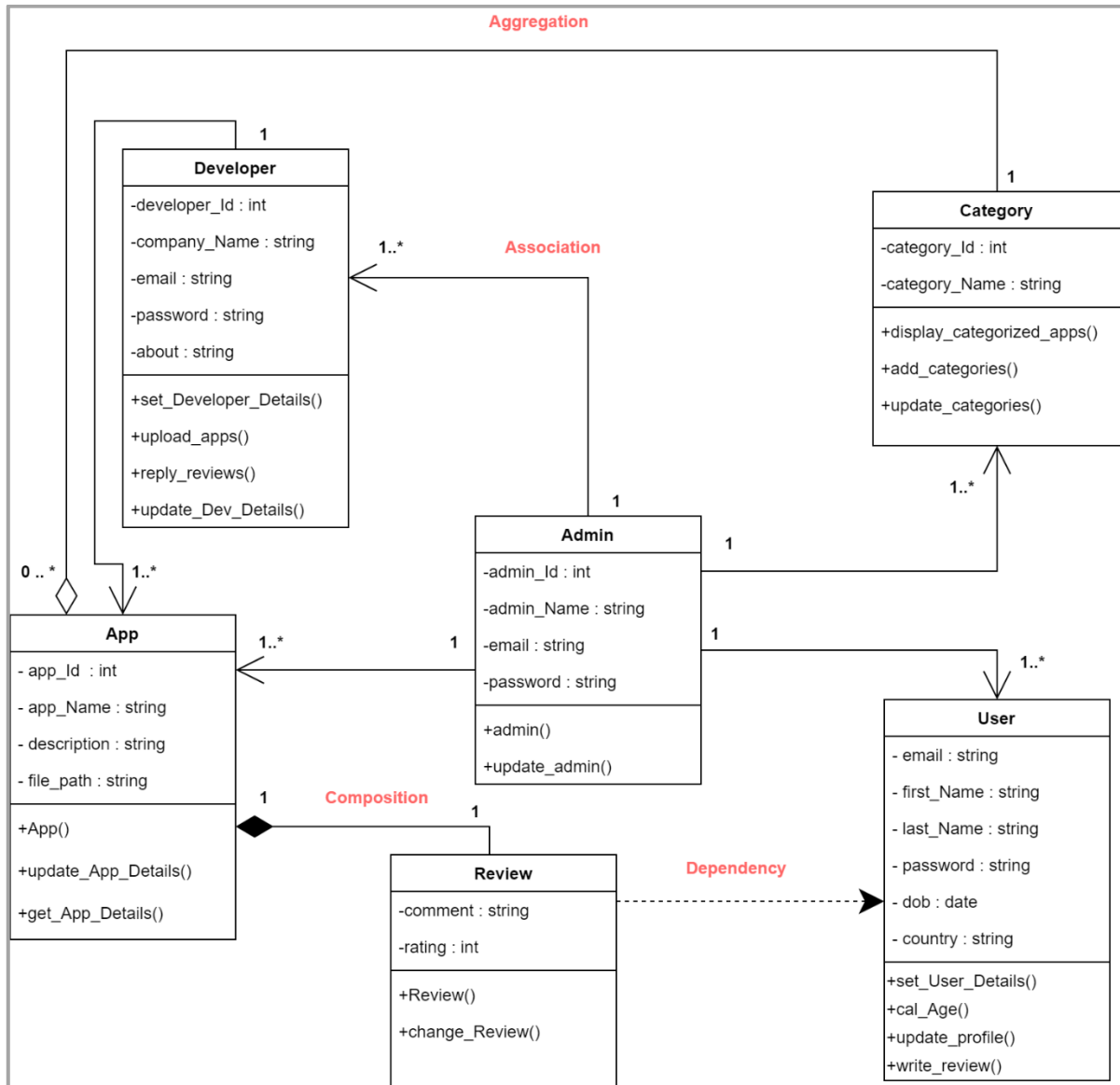
Developer	
Responsibilities	Collaborations
Keep developer details	

Admin	
Responsibilities	Collaborations
Keep admin details	
Manage users	User
Manage developers	Developer

Review	
Responsibilities	Collaboration
Keep review details	App
Keep reviewer's details	User

Category	
Responsibilities	Collaboration
Categorize apps	App
Keep available categories	

Exercise 1 - UML Class Diagram



Exercise 2 - Coding for the classes

```
#include <iostream>
#include <string>

class developer {
private:
    int developer_id;
    string company_name;
    string email;
    string password;
    string about;

public:
    void set_developer(int devId,string companyName,string mail,string pwd,string About);
    void update_dev_details();
    void upload_app();
    void update_app();
    void reply_review();
};

class category {
private:
    int category_id;
    string category_name;

public:
    void display_categorized_apps();
    void add_categories(int id,string catName);
    void update_categories();
};

class admin{
private:
    int admin_id;
    string admin_name;
    string email;
    string password;

public:
    admin(int id, string name, string email, string pwd);
    void update_admin();
}

class app {
private:
    int app_id;
    string app_name;
    string description;
    string file_path;

public:
    void app(int Id,string Name,string desc,string fpath);
    void update_app_details();
    void get_app_details();
};
```



```

class user {
private:
    string email;
    string first_name;
    string last_name;
    string password;
    string dob;
    string country;

public:
    void set_user_details(string mail,string fname,string lname,string pwd,string dob,string ctry);
    int cal_age();
    void update_profile();
    void write_review();
};

class review {
private:
    int userId,appId;
    string comment;
    int rating;

public:
    void review(string cmt);
    void change_review();
};

```

```

int main() {
    string Name,FirstName,LastName, Email, Password,
           About,CompanyName,Dob,Country,Description,Filepath,Comment;

    int ID;

    //Creating Dynamic Developer Objects
    developer* dev1, * dev2;
    dev1 = new developer;
    dev2 = new developer;

    //Creating Dynamic User Objects
    user* us1, * us2;
    us1 = new user;
    us2 = new user;

    //Creating Dynamic Category Objects
    category* cat1, * cat2;
    cat1 = new category;
    cat2 = new category;

    //Initializing Dynamic App Objects with Overload constructor
    app* app1, * app2;
    app1 = new app(ID, Name, Description, Filepath);
    app2 = new app(ID, Name, Description, Filepath);

    //Creating Dynamic Review Objects
    review* rev1, * rev2;
    rev1 = new review(Comment);
    rev2 = new review(Comment);
}

```

//Creating Dynamic Admin Objects With Overloaded Constructor

admin* ad1, * ad2;

ad1 = new admin(ID, Name, Email, Password);

ad2 = new admin(ID, Name, Email, Password);

//Initializing Developers

dev1->set_developer(ID,CompanyName,Email>Password>About);

dev2->set_developer(ID, CompanyName, Email, Password, About);

//Initializing Users

us1->set_user_details(Email, FirstName, LastName, Password, Dob, Country);

us2->set_user_details(Email, FirstName, LastName, Password, Dob, Country);

//Initializing Category

cat1->add_categories(ID, Name);

cat2->add_categories(ID, Name);

delete dev1, dev2;

delete us1, us2;

delete cat1, cat2;

delete rev1, rev2;

delete ad1, ad2;

delete app1, app2;

}