

|  |
| --- |
| VNU HCM - University of Science  Faculty of Information Technology |
| Subject: Java Programming  Report #1  Object-Oriented Programming |
| November 11  Class: 19CLC-KTPM2  Authored by: Group 4 - Cookies  Phạm Trọng Vinh Khuê - 19127038  Trần Đại Hoàng Trung - 191270081  Trần Thanh Tùng - 19127311 |

1. **Introduction**

**1. Project context**

Since the end of 2019, the global pandemic of Coronavirus disease 2019, or COVID-19   
for short, has caused constant substantial damage in economy and people's livelihood all over the world. More devastatingly, it has taken millions of people's lives. Therefore, governments have taken initiative and medcial personnels are dispatched to deal with the situation.

In order to help make their work easier and accelerate the schedule, a COVID-19 patient management system is a must. It is an application in which administrators can keep track of the patients' information and status, as well as managing the necessities packages.

**2. Purpose of topic's selection**

Like everyone else, our group want this pandemic to end as quickly as possible, so that the world can return to its normal state and no one else has to suffer. We decide to help by developing an application that fulfills the requirements stated in the Project Context section above. Additionally, those on the "front lines" like medical personnels have worked so hard to keep us safe from COVID-19, so lifting some weight off their shoulders is one of our intentions. Our group's first target users are the Vietnamese people, as we are Vietnamese ourselves.

**3. Existing applications review**

As of November 2021, Vietnam has several applications with similar functionality. They includes common features such as vaccine certificate, medical declaration. However, each of them also contains their own specific features. Some notable applications are listed below.

* Sổ Sức khỏe Điện tử (Tentatively translated: Digital Health Manual) provides a wide range of functions like vaccination registration or arrangement for health examination.
* PC-Covid allows its users to check for COVID test information.
* VNEID can be used to declare the locations that users will go in the near future.

**4. System requirements**

Although fulfulling some important functionality, the applications mentioned above suffer from a few major problems such as late synchronization, dull user interface, or late OTP code delivery. As a result, these applications recieve negative to average reviews from the users. We want our own application to be without such issues, hence the requirements below.

* **Functional Requirements**
  + The application should allow Adminstrator account creation upon its first start-up.
  + Patients are required to create password if their accounts already exist in the system but no login has ever occured.
  + Only Administrators have the right to view the list of Patients, along with their   
    specific information; change their status and location, and manage necessity packages, in addition to compiling statistics.
  + Patients can view their information, history of management and necessity packages,   
    as well as their payment history and debit.
  + The software system should include banking API.
* **Non-Functional Requirements**
  + The user interface should follow UI/UX guidelines for better user experience.
  + The client component of the system shall not require more than 100 MB disc space and 4GB Ram, while requiring the operating system of Windows 10 or more.
  + The server component of the system shall utilize Microsoft SQL Server.
  + Most of the client component's responses and transactions should occur no more than 10 seconds.
  + The server should be capable of handling at least 100 concurrent users.
  + User inputs must be responded in less than 200 miliseconds.
  + All information must be synchronized less than 3 hours.

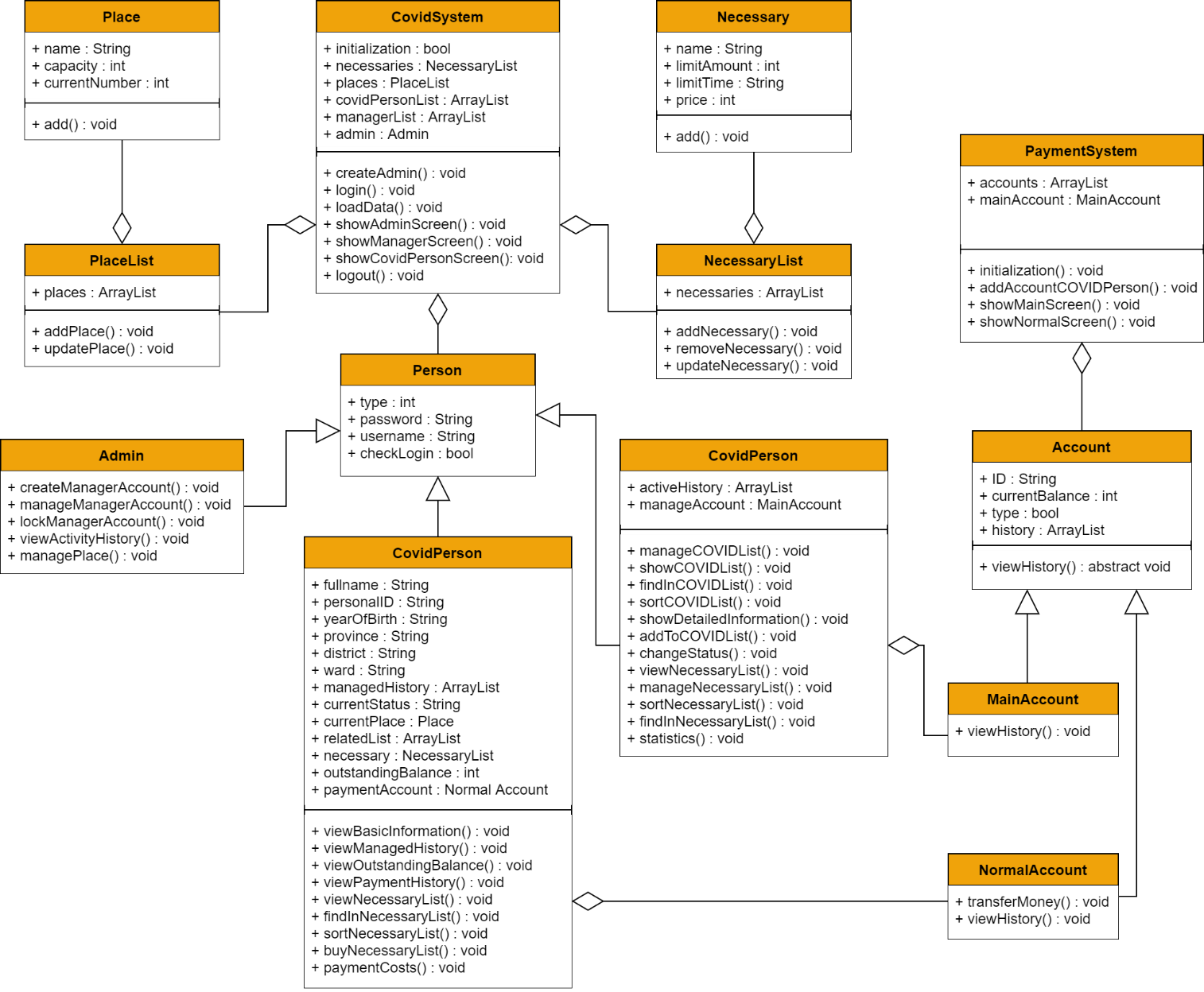
**5. Project scope**

The scope of this project is to create an environment where patients affected by COVID-19   
can be managed by medical personnels with little to no difficulty. Furthermore, said patients are provided with access to benefitial necessity packages and will have a pleasant experience, as well as the ability to obtain information about their health and status in general.

**6. Expected outcome**

At the end, we expect to have a fully functional desktop application which can used by both the Administrators and the Patients. This application will consist of every functions and features stated in the Requirements file, while satisfying all of the requirements mentioned in section I.4 above.

1. **Analysis and Design**



* **Person**: This is an abstract class stores basic attributes of an account of COVID System such as type, username, password, checkLogin.
* **Admin**: This is a child class from class Person. This class represents the Admin module and performs some admin-specific operations.
* **Manager**: This is a child class from class Person which has some own attributes such as activeHisory for storing activity history and manageAccount for manage main account of Payment System.This class represents the Manager module and performs some manager-specific operations.
* **COVIDPerson**: This is a child class from class Person which has many own attributes such as fullName, personalID, yearOfBirth, province, dictrict, ward, managedHistory, relatedList, currentPlace and currentStatus for basic information of a COVID person, necessary for list of bought necessary, outstandingBalance for outstanding balance and paymentAccount for own payment account of each COVID peson.This class represents the COVIDPerson module and performs some COVIDPerson-specific operations.
* **Place**: This class represents treatment place, stores basic attributes of a treatment place such as name, capacity, currentNumber and performs some management operations like add,….
* **PlaceList**: This is a class used for storing list of treatment place and performing management operations on the list of treatment place such as add a place to the list, update a place in list.
* **Necessary**: This class represents necessary for Covid person, stores basic attributes of a necessary such as name, limitAmount, limitTime, price and performs some management operations like add,….
* **NecessaryList**: This is a class used for storing list of necessary and performing management operations on the list of necessary such as add a necessary to the list, update a necessary in list, remove a necessary in list.
* **CovidSystem**: This is a class which manages the whole Covid System. This class used for storing all the necessary data of COVID System , performing authentication operations, showing menu and performing specific operations for each type of account.
* **Account**: This is an abstract class stores basic attributes of an account of Payment System such as ID, currentBalance, type, history. This class also has an abstract operations viewHistory for its childs.
* **MainAccount**: This is a child class from class Account. This class represents the main account of Payment System to receive payment from normal account and performs some own operations.
* **NormalAccount**: This is a child class from class Account. This class represents the normal account of Payment System. This account is created when a Covid person added to Covid System. This class also has some own operations.
* **PaymentSystem**: This is a class which manages the whole Payment System. This class stores list of normal payment account and a main payment account. This class also show menu and performs specific operations for each type of payment account.

1. **Implementation**

In order to transfer the diagrams listed above to Java code, we'll use Java-exclusive attributes such as Interface for classes that share the same functionality, or, similarly, the keyword extends for inheritance. This project will fulfill all of the Object-oriented Programming's characteristics, which are Encapsulation, Inheritance, and Polymorphism.

We will utilize Swing widget toolkit for Graphical User Interface, and SQL as the database.

1. **Result**

During the two weeks of research & writing this report, we have acquired information about the current situation of the COVID-19 pandemic, and the related applications with similar features. This allows us to analyze and come up with suitable functions and requirements for our application.

Furthermore, we've got full comprehension of the application's structure with the help of diagrams designed by our group such as UML Class Diagram or the Package Diagram.

With our project properly planned out, we'll be able to develop the application using the guidelines stated in the Implementation section of the document.

1. **Plan**
2. **Task Decomposition**Diagram

   Description automatically generated
3. **Project Management**
   1. **Project Estimates**

The estimated time for the whole project is 8 weeks, following the detailed plan in part 2.2 below. This estimate is based on the fortnightly tasks for each phase of the project. As this is a voluntary program, the user satisfaction would be the compensation for our development team. Re-estimation would only occur in the event of feedback received once the program published.

* 1. **Project Plan**
* ***Sprint 1:***
* Duration: 14 days (01/11/2021 - 14/11/2021).
* Outcome: Class diagram, Package diagram, Report #1.
* Task information:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task | Member | Task type | Duration | Start date | End date |
| Report - Introduction | Tùng | Document | 6 | 01/11 | 07/11 |
| Report - Plan | Khuê | Document | 6 | 01/11 | 07/11 |
| Class diagram - Basic | Trung | Document | 6 | 01/11 | 07/11 |
| Database and Network Research | All | Training | 6 | 01/11 | 07/11 |
| Class diagram | All | Document | 2 | 07/11 | 09/11 |
| Package diagram | All | Document | 2 | 09/11 | 11/11 |
| Report - Analysis and Design | Khuê, Trung | Document | 3 | 11/11 | 14/11 |
| Report - Result | Tùng | Document | 3 | 11/11 | 14/11 |
| Report Redesign | Tùng | Document | 1 | 13/11 | 14/11 |

* Task information:
* ***Sprint 2:***
* Duration: 14 days (15/11/2021 - 28/11/2021).
* Outcome: Patient System, Report #2.
* Task information:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task | Member | Task type | Duration | Start date | End date |
| Implementation  Patient System - Admin | Trung | Coding | 13 | 15/11 | 28/11 |
| Implementation  Patient System - Manager | Khuê, Trung | Coding | 13 | 15/11 | 28/11 |
| Implementation  Patient System - Patient | Tùng | Coding | 13 | 15/11 | 28/11 |
| Report Update & Redesign | Tùng | Document | 3 | 25/11 | 28/11 |

* Task information:
* ***Sprint 3:***
* Duration: 14 days (29/11/2021 - 12/12/2021).
* Outcome: Payment System, Report #3.
* Task information:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task | Member | Task type | Duration | Start date | End date |
| Implementation  Payment System - Manager | Trung | Coding | 13 | 29/11 | 12/12 |
| Implementation  Patient System - Patient | Tùng | Coding | 13 | 29/11 | 12/12 |
| System Synchronization | Khuê | Coding | 13 | 29/11 | 12/12 |
| Report Update & Redesign | Tùng | Document | 3 | 9/12 | 12/12 |

* Task information:
* ***Sprint 4:***
* Duration: 14 days (13/12/2021 - 26/12/2021).
* Outcome: Final System, Report #3.
* Task information:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task | Member | Task type | Duration | Start date | End date |
| Implementation  Data Transfer via Network | All | Coding | 6 | 13/12 | 19/12 |
| Component Testing | All | Coding | 2 | 19/12 | 21/12 |
| System Testing | All | Coding | 3 | 21/12 | 24/12 |
| Report Update & Redesign | Tùng | Document | 3 | 23/12 | 26/12 |
| Meeting Preparation | All | Document | 2 | 24/12 | 26/12 |

* Task information:

1. **Risk Management**

Risks will be identified in the Inception Phase using the steps identified in the RUP for   
Small Projects activity “Identify and Assess Risks”. Project risks are evaluated at least once per iteration and documented in this table. The risks of the greatest magnitude are listed first in the table.

|  |  |  |
| --- | --- | --- |
| Risk Magnitude | Risk Description | Mitigation Strategy |
| High | To complete the stakeholder’s requirements, the team will deal with 3 aspects of programming: Java, Database and Networking. Applying all of those to the app might have trouble and easily lead to poor-quality products. | Each member have to do some study and research in all of those fields. |
| Medium | The team consists of 3 members instead of 4 as the maximum number. This will be a disavantage for our team to compete with other teams. Overdue and poor-quality products might be the result of this point. | The whole project must be carefully seperated into phases to ensure balanced workloads. Each member is required to read each phase’s plan and self-managed time budget to complete his/her tasks on time. |
| Low | There might be a little gap in skill level among team’s members. Misunderstandings may appear and slow down the team’s process | Each member needs to keep a cooperative and supportive attitude towards teammates. |
| Low | A team member suddenly quits the project may lead to an unexpected result to the whole project. | All members must understand the whole project to be albe to handle quited member’s work. |