**HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY**

**SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY**



**MINI-PROJECT REPORT**

**Virus Demonstration – Object-oriented Programming project**

**Course ID: IT3100E – Class code: 141177**

**Supervisor: Prof. NGUYEN Thi Thu Trang, PhD.**

**Group 14**

|  |  |
| --- | --- |
| **Student Name** | **Student ID** |
| Nguyễn Viết Minh | 20214917 |
| Tạ Ngọc Minh | 20214918 |
| Nguyễn Tùng Lương | 20214913 |
| Đào Hà Xuân Mai | 20210562 |

**Hanoi – 2023**

**Table of Contents**

[**1.** **Assignment of members** 3](#_Toc140441797)

[**2.** **Mini-project description** 3](#_Toc140441798)

[**2.1** **Mini-project overview** 3](#_Toc140441799)

[**2.2** **Mini-project requirement** 3](#_Toc140441800)

[**2.3** **Use Case Diagram** 4](#_Toc140441801)

[**3.** **Design** 5](#_Toc140441802)

[**3.1** **General Class Diagram** 5](#_Toc140441803)

[**3.2** **Other Class Diagram and Explanation** 6](#_Toc140441804)

[**4.** **References** 8](#_Toc140441805)

1. **Assignment of members**

|  |  |
| --- | --- |
| Task | Member |
| UseCase Diagram, Class Diagram | Nguyen Viet Minh |
| General Design, Initial implementation, Model | Nguyen Viet Minh |
| Image, video, information for input and GUI | All members |
| GUI Design | Nguyen Tung Luong (70%)  Ta Ngoc Minh (30%) |
| Controller implementation | Nguyen Tung Luong (20%)  Ta Ngoc Minh (80%) |
| Report writing | Nguyen Viet Minh (70%)  Nguyen Tung Luong (20%)  Ta Ngoc Minh (10%) |
| Slide outlining and design | Dao Ha Xuan Mai |
| Demo video recording | Ta Ngoc Minh |

1. **Mini-project description**
   1. **Mini-project overview**

In this project, we provide knowledge about different types of viruses, as well as the way they infect to have the basic comprehension to prevent them.

Basic structure of virus:

* Every virus has 2 basic elements: acid nucleic and capsid.
* Based on their structure, viruses are divided into 2 categories: with and without lipid envelope.
  + Viruses without envelopes will dissolve their capsid when reach the target cell.
  + Viruses with envelopes usually have anchors, called glycoprotein. The mechanism for infecting in this case is by lock–key: when reaching the host cell with the suitable outer structure, it uses its glycoproteins to attach, then injects its acid nucleic into the cell.

You can view our project in GitHub [here](https://github.com/Min-KiD/OOP.20222.14) or watch our demo on [YouTube](https://www.youtube.com/watch?v=zGCq_iQrOS0).

* 1. **Mini-project requirement**

In this project, our team needs to build a system for users to interact with. Main purpose of this project is that the user can get the information about different types of viruses that we provided in the application. Users can choose which viruses they want to see and which part that they want to know more about.

For that mission, if our team wants to complete this project. We need to have the knowledge about viruses we want to give in, therefore reading documents, understanding, choosing viruses involved and finally taking some or all parts of it is the most important and first step thing to do.

Besides choosing image, information, and type of virus for involving in is designing, video of how it infects other species. Understand its construction in image to give to user help them easy to Imaginate and remember for long time.

* 1. **Use Case Diagram**

A diagram of a person with text

Description automatically generated

*Figure 1. UseCase Diagram*

As we can see from the use case, we can choose the virus we find interest in and view its image and details structure of it if we want. We can find the information when we click on it, it will pop up, also we can choose to play video to understand how viruses work or infect the species it connects with.

It also has a help menu when we first open the app for the purpose of helping the user first interact to understand how its work, where to click on and finally an exit button for escape from the app.

A screenshot of a computer

Description automatically generated

For easy to understand how the app work more, please visit our [demo](https://www.youtube.com/watch?v=zGCq_iQrOS0).

1. **Design**
   1. **General Class Diagram**

A screenshot of a computer

Description automatically generated

*Figure 2. General Class Diagram*

Running.java is the main for running the app, it takes information in the setup.java after being constructed and takes the GUI in MainController.java for graphical design, loading and making a completed application.

* 1. **Other Class Diagram and Explanation**

A close-up of a diagram

Description automatically generated

*Figure 3. Virus Class Diagram*

Virus Class Diagram is Model part, the constructor.

For more details, the 4 OOP techniques used in model as follows:

**- Encapsulation** is often used to hide the internal representation, or state, of an object from the outside. So major attributes are set private for protecting information from our accessor and ​​store the data members and data methods of a class together.

A computer screen shot of a virus

Description automatically generated

For example, private String *envelopeProtein*, private String *mProtein*, private String *spike*, private String *hEsterase*, restrict the access to ***CoronaVirus*** class and its structure.

**- Abstraction** is the process by which data and programs are defined with a representation similar in form to its meaning (*semantics*) while hiding away the implementation details.

A close-up of a computer screen

Description automatically generated

For example, abstract class ***Virus*** is used to prevent creating any Object instance and optimize method structure and reduce the object to its essence so that only the necessary characteristics are exposed to the users.

**- Inheritance** is a mechanism which enables one class to acquire properties of another class which means that child class (the one that inherits) can *reuse*attributes or methods of super class (the one that is being inherited).

A close up of a white background

Description automatically generated

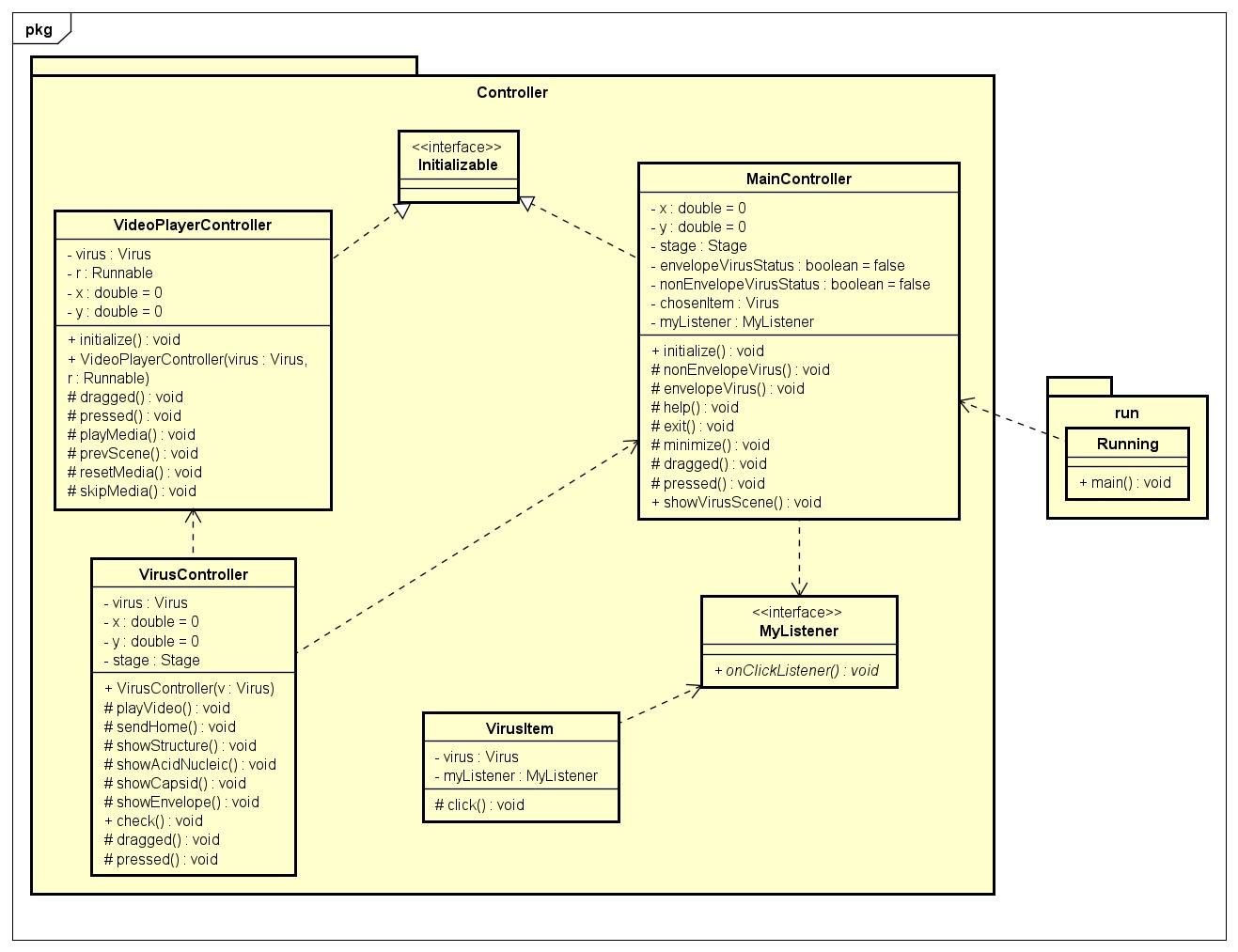
For example, ***EnvelopedVirus*** inherit from ***Virus*** because they share the similarity of virus.

**- Polymorphism** is a mechanism that enables us to do the same action in different ways. In other words, in object-oriented programming, it is the capability of a method to act differently based on the object it is acting upon. There are 2 types of polymorphism:

Static: the type of polymorphism that occurs in *the same class* by *overloading* methods.

Dynamic: the type of polymorphism that occurs when child class *overrides* method declared in super or base class

For example, when we override some application of JavaFX library for suitable for our problem or when we use static for setup.java to make sure our information about virus is public but unchanged by the user.



*Figure 4. Controller Class Diagram*

Controller is responsible for controlling views. Here we want to design the controller as general as possible (can be used with all viruses). We use polymorphism technique, all controllers only interact with the parent class Virus, so it will be able to work with any of its child classes.

However, there are some special cases such as enveloped and non-enveloped viruses, we test them with instanceof and give interactions on View according to their class. Furthermore, for viruses with special properties, we store that data in a list and create a getter in the model to retrieve it.

As for the attack method, in fact there are 3 types of attack methods in nature but here with the viruses we show, they only have 2 types of attack methods, one of enveloped and one of non-enveloped viruses.

In addition, to be able to create more beautiful Views instead of just normal buttons, we have designed a panel that listens to action and clicks. And with that panel we created our own listener and simple controller for it.

1. **References**
2. Herrity, K. (2023, February 4). What is object-oriented programming (OOP)? A complete guide. Indeed.com. <https://www.indeed.com/career-advice/career-development/what-is-object-oriented-programming>
3. Movsisyan, S. (2020, April 1). Coronavirus disease and OOP paradigm in Java. Medium. <https://medium.com/javarevisited/coronavirus-disease-and-oop-paradigm-in-java-affae9bde0b4>
4. Ideal of initial code and OOP characteristics come from here.
5. [Virus - Wikipedia](https://en.wikipedia.org/wiki/Virus#Structure), [Viral envelope - Wikipedia](https://en.wikipedia.org/wiki/Viral_envelope),
6. Most of the knowledge taken about virus go from here, all the images taken from google and video taken from YouTube.
7. [JavaFX Tutorial - GeeksforGeeks](https://www.geeksforgeeks.org/javafx-tutorial/).
8. [Stack Overflow - Where Developers Learn, Share, & Build Careers](https://stackoverflow.com/).
9. JavaFX Documentation, <https://openjfx.io/javadoc/20/>.
10. CSS GUI FXML Formatting and Design: <https://github.com/k33ptoo/javafx-sample-dashboard/blob/master/src/css/fullpackstyling.css>