Computer History Museum Management System

CSE-2112: Object Oriented Programming Lab

–

Submitted to
Dr. Muhammad Ibrahim
Associate Professor
Md. Ashraful Islam
Assistant Professor

Submitted by
Meherun Farzana, roll - 05
Himel Chandra Roy, roll - 13
Aniket Joarder, roll - 48
Second year

Department of Computer Science And Engineering
University of Dhaka

Table of Contents

Introduction

- **1.1 The Concept**
- **1.2 Motivation**
- **1.3 Features**
- **1.4 Tools**

Design and Implementation

- **2.1 UX Flow**
- **2.2 UI**
- **2.3 System Design**
- **2.4 Discussion**
- **2.5 Use of OOP**
- **2.6 Code Repository**

Conclusion

- **3.1 Challenges and Solutions**
- **3.2 Future Development**
- **3.3 Conclusion**

Introduction

1.1 The Concept

“Bangladesh Computer History Museum Management System Software” is actually an appliance for managing and displaying the functionalities of a museum that features different types of artifacts and other activities related to computers and their history, managing people related to the museum. This project also keeps track of the ticketing system of the museum.

1.2 Motivation

The national museum of Bangladesh uses a manual system for keeping track of records of their employees and the managerial system in most cases. There are some major issues with that system, such as:

- The centralized employee addition and deletion take much time and lots of paper to keep records.
- Keeping information about each visitor is tough in a handwritten system.
- The manual price calculation system is tedious and there is room for error.
- Different inter-connected departments of a museum are hard to control using the analog system.
- The addition and deletion of new artifacts and keeping records of them are really very hard in the analog system.

Nowadays, the museum is trying to convert its functionalities to a digital system yet there are ways to bring the whole into one shade. To overcome these problems, we thought of a digitized version of the museum, especially a computer history museum which may help the managerial bodies of the museum to manage everything mentioned above with an application that is designed with a welcoming and modern interface. Besides, as the students of Computer Science and Engineering, we hope that in the near future, like other developed countries, Bangladesh will also have it's own museum of Computer History.

1.3 Features

The key features of this application software are:

1. It allows the use of multiple users including an admin in a single device.
2. An admin can add more users to this application giving some restrictions to the new users (i.e. new users can not add other users).
3. It provides a secured database containing all the personal information of password-protected user accounts.
4. It provides a very flexible and user-friendly dashboard to access the features of the application.
5. It allows the users to add or delete an employee of different categories and employ them in the departments of the museum.
6. There are two major departments in the museum, the curatorial department and the non-curatorial department. Both of these departments are divided into some more divisions too. An employee can work in any of these departments and that can be controlled by the users of this application.

7. It contains information about the artifacts and photo gallery of the museum that can be updated by the users.
8. An educational system is introduced here with some courses related to computers and their history and there are options for selecting educators for the students.
9. It keeps and shows information about the Board of directors for the museum.
10. It allows the ticket-selling agent to use this application.
11. Information about visitors and their past visits is kept here, visitors visiting this museum more than 3 times may get a discount in the national holidays.
12. The price of the ticket can be calculated here without any error; a dynamic ticket and a map (which can be updated from the Curatorial Department section) of the whole museum can also be printed here in a single click.

1.4 Tools and Technologies

We tried to make the best use of all the latest tools and technologies for product development; from UI/UX design to robust system design with the power of the Object-Oriented feature of Java and its GUI Framework JavaFX. Below are listed the tools and technologies we used :

1. UI/UX design: SceneBuilder
2. Code IDE: IntelliJ
3. CodeBase: JAVA
4. Framework: JavaFX
5. Database: SQLite
6. Design Code Base: CSS
7. UX Code Base: FXML
8. Ticket Printing Tool: Jaspersoft Studio

Design and Implementation

2.1 UX Flow

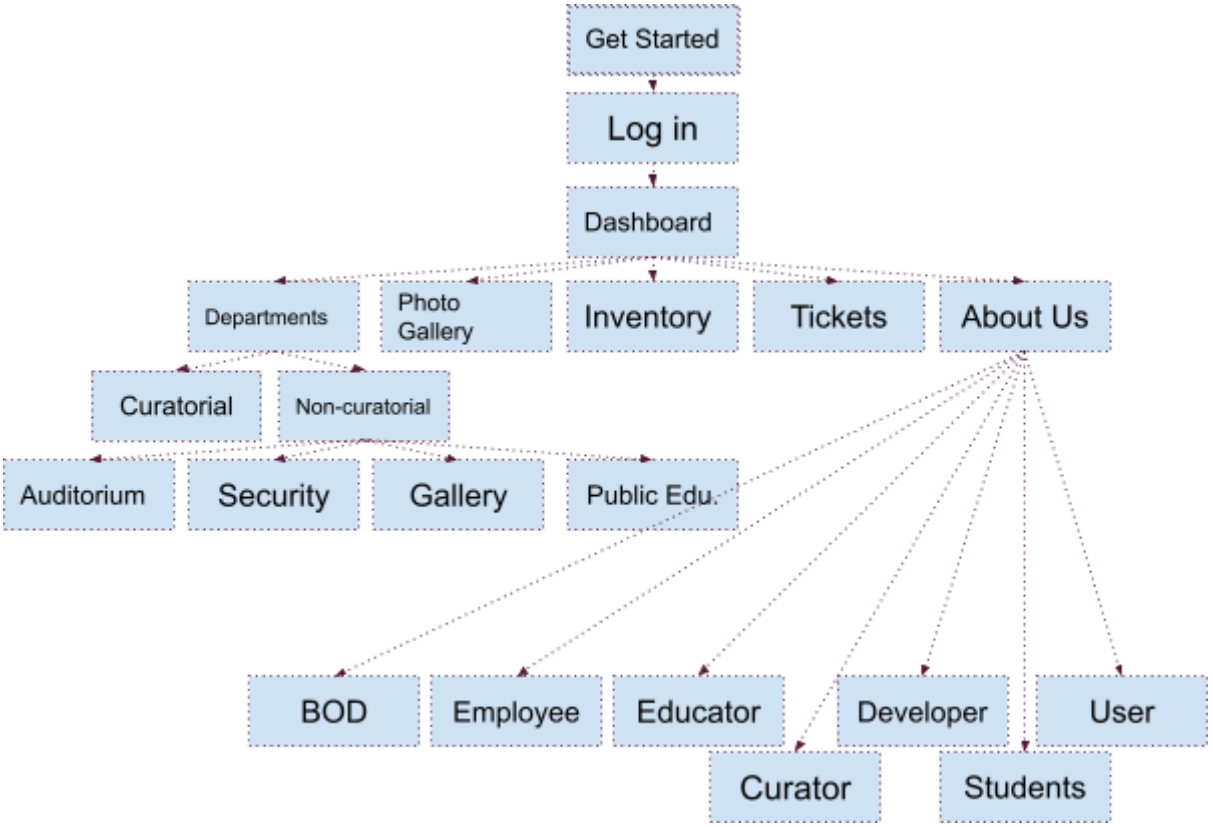
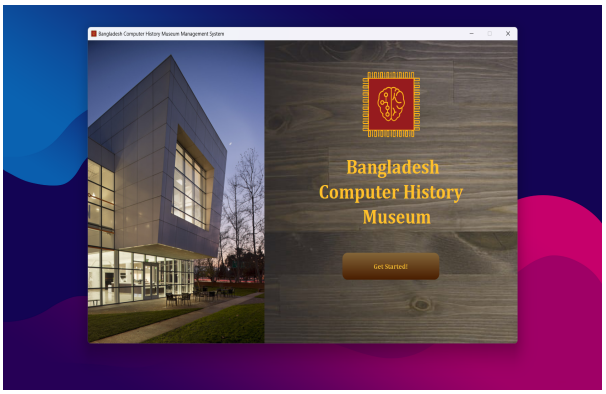


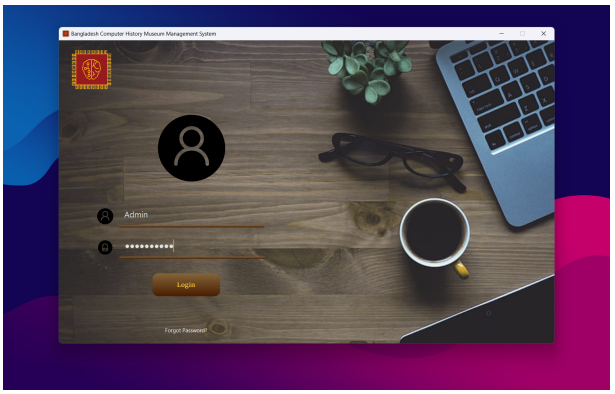
Figure: UI flow and use-case diagram

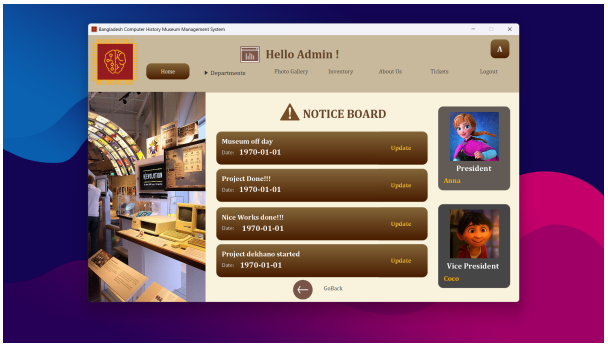
This figure shows the UX Flow of the project. Whenever a user enters the software, the user will be taken to the *Get Started* page and *Login* page followed by that.

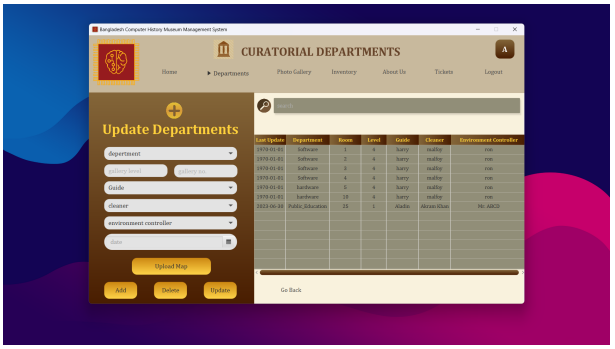
After logging in, The dashboard of the software will come in front with the four latest notices and two of the BOD members. Then the user will be able to access all other scenes of the application. The user may log out from the application at any time the user wants. So, there is a *Logout* button to perform that work.

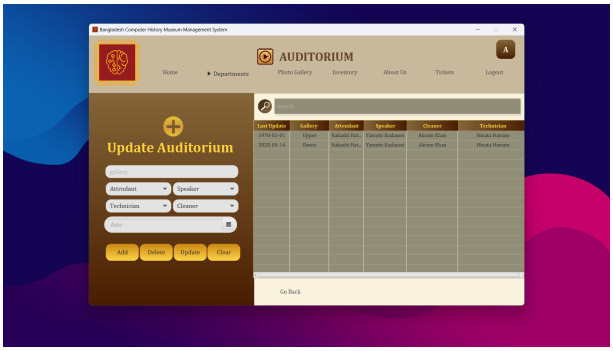
2.2 UI

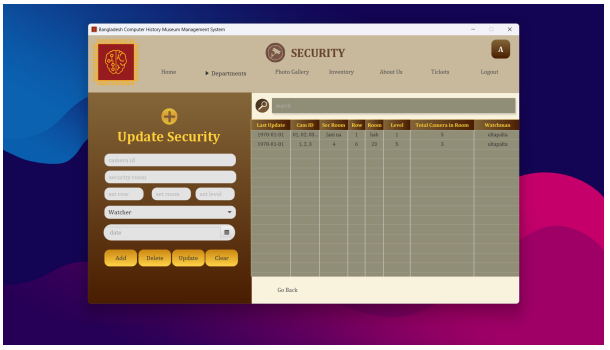

Bangladesh Computer History Museum
Get Started!


Admin
Login
Forgot Password?


Hello Admin !
NOTICE BOARD
Museum off day
Project Done!!!
Nice Works done!!!
Project delshano started
President
Vice President


CURATORIAL DEPARTMENTS
Update Departments


AUDITORIUM
Update Auditorium


SECURITY
Update Security

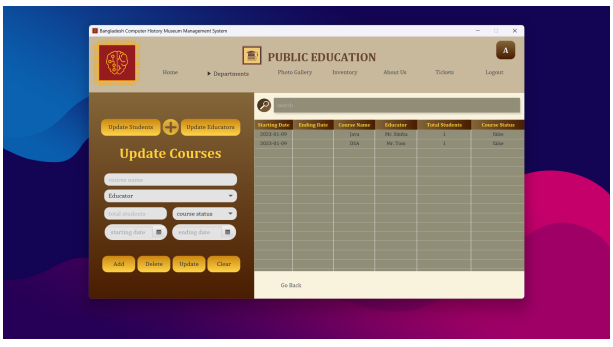
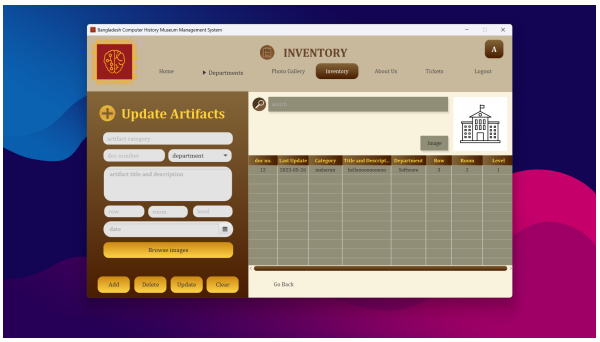
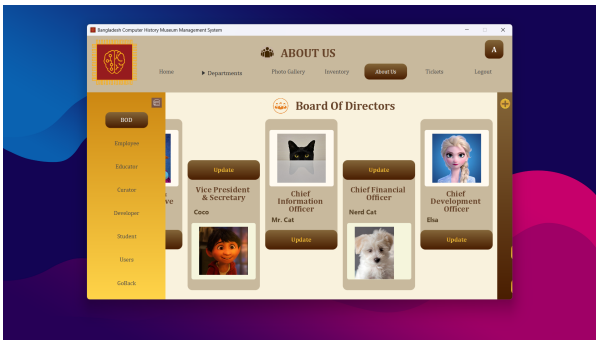
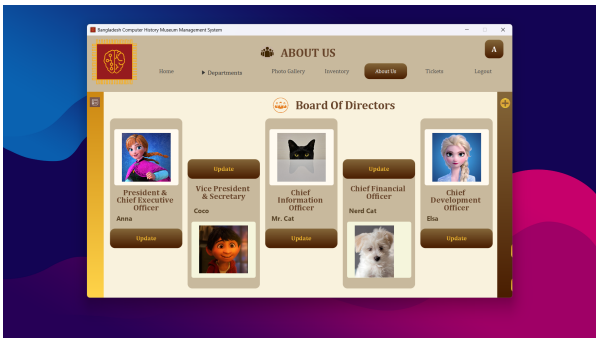
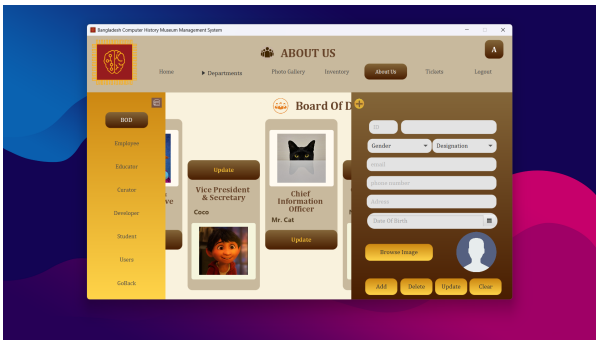

PUBLIC EDUCATION
Update Courses

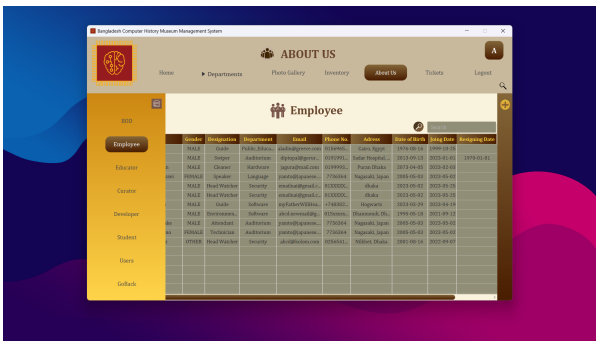

PHOTO GALLERY
Update Photos


INVENTORY
Update Artifacts
Browse Images
Add
Delete
Update
Clear


ABOUT US
Board Of Directors
Vice President & Secretary
Coco
Chief Information Officer
Mr. Cat
Chief Financial Officer
Nerd Cat
Chief Development Officer
Elsa
Update


ABOUT US
Board Of Directors
President & Chief Executive Officer
Anna
Vice President & Secretary
Coco
Chief Information Officer
Mr. Cat
Chief Financial Officer
Nerd Cat
Chief Development Officer
Elsa
Update

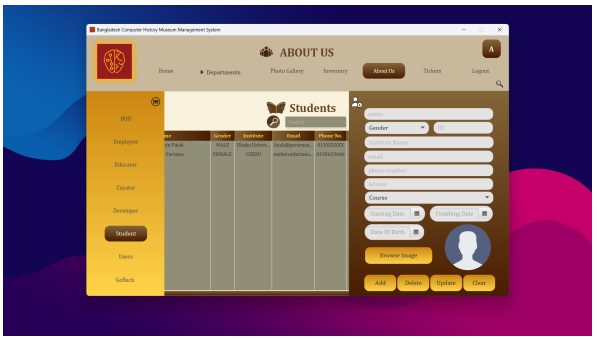

ABOUT US
Vice President & Secretary
Coco
Chief Information Officer
Mr. Cat
Browse Image
Add
Delete
Update
Clear

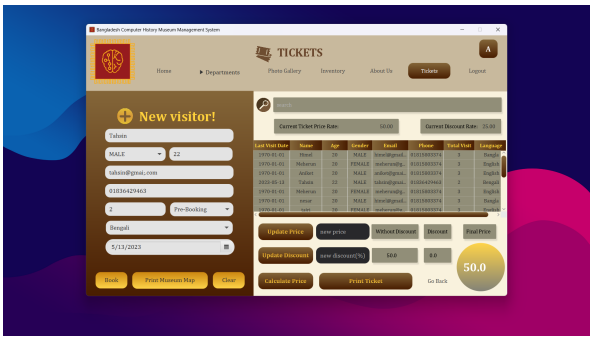

ABOUT US
Employee


ABOUT US
Educators
Browse Image
Add
Delete
Update
Clear


ABOUT US
Curator
Browse Image
Add
Delete
Update
Clear


ABOUT US
Developers
Browse Image
Add
Delete
Update
Clear





These are some snaps of the running application. These pages show (from the beginning):

The introductory page	The Login page
Dashboard	Curatorial Department
Auditorium	Security
Public Education	Photo Gallery
Inventory	About Us
BOD	BOD page with editing section
Employee	Educators
Curator	Developers
Users	Students
Ticket page	Sample Ticket

2.3 System Design

System design is the solution for the creation of a new system. This phase focuses on the detailed implementation of the feasible system. It emphasizes translating design. System design has two phases of development:

- Logical design
- Physical design

Here the logical design is done through data flow and database design. The physical design is followed by physical design or coding.

During the logical design phase, the analyst describes inputs (sources), outputs(destinations), databases (data sores), and procedures (data flows) all in a format that meets the user requirements. The physical design produces the working system by defining the design specifications, which specify exactly what the system must do.

1. Input and Output Design:

The input design involves determining the inputs, validating the data, minimizing data entry, and providing a multi-use facility. With the feature, users can be able to upload pdf, jpg, and png files. Also, the direct user inputs are handled in this section. Inaccurate inputs are the most common cause of errors in data processing. Errors entered by the data entry operators can be controlled by input design. All the input data are validated and if any data violates any conditions, the user is warned by a message.

Computer output is the most important and direct source of information to the user. Output design is a very important phase since the output needs to be in an efficient manner. An efficient and intelligible output design improves the system's relationship with the user. Allowing the user to view the sample screen is important because the user is the ultimate judge of the quality of

output. The output module has various options. Outputs (such as pdf of notices and tickets etc) can be printed and downloaded from the software directly.

2. Database Design:

Databases are storehouses of data used in software systems. The data is stored in tables inside the database. Several tables are created for the manipulation of the data for the system. In the whole project, a total of seven databases were used for storing curatorial data, employee data, other departmental data, security people's data, user id and passwords, educators' and students' data, etc. Here **SQLite** is used for database management.

Class Hierarchy

The class hierarchy of Computer History Museum Management System software is described below with the UML diagram. The main features of oop inheritance, Encapsulation and polymorphism are properly used in the software.

The class hierarchy for storing information about the departments is given below:

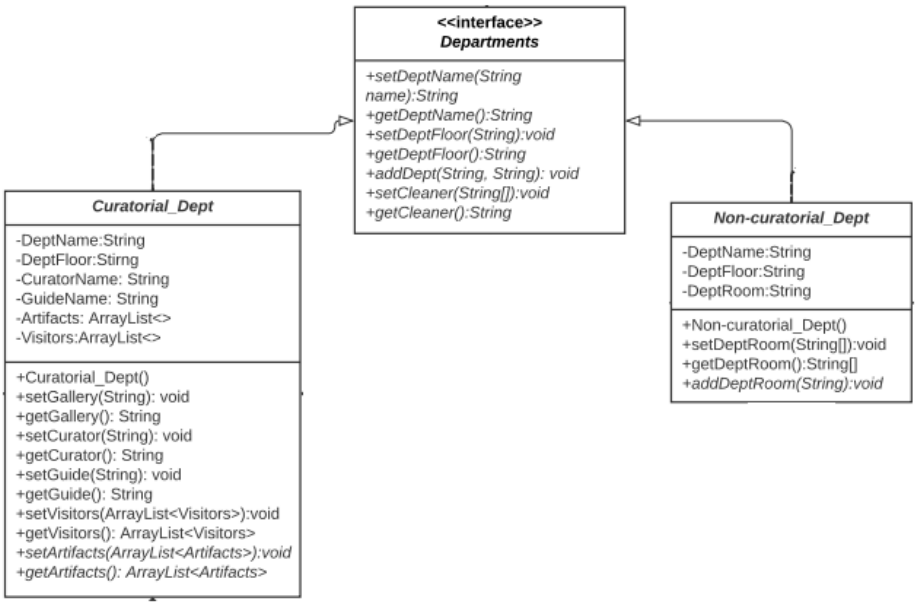
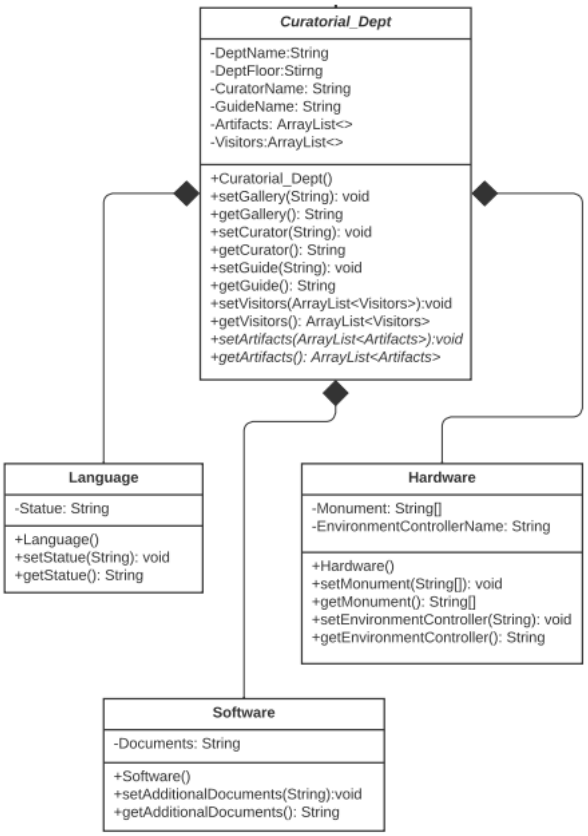


Figure: Class hierarchy of *Departments*

1. Departments

This interface is created for representing the departments of the museum. It contains all the common methods used in every department. There are two major divisions in it. They are:



[1.1] Curatorial_Dept

This class stores information about the curatorial departments, such as the name of the department, floor and level number, the names of the guide and curator, etc.

[1.2] Non_curatorial_Dept

This class contains information and methods about all the non-curatorial departments of the museum. There are four non-curatorial departments shown in this museum.

[1.2.1] Auditorium

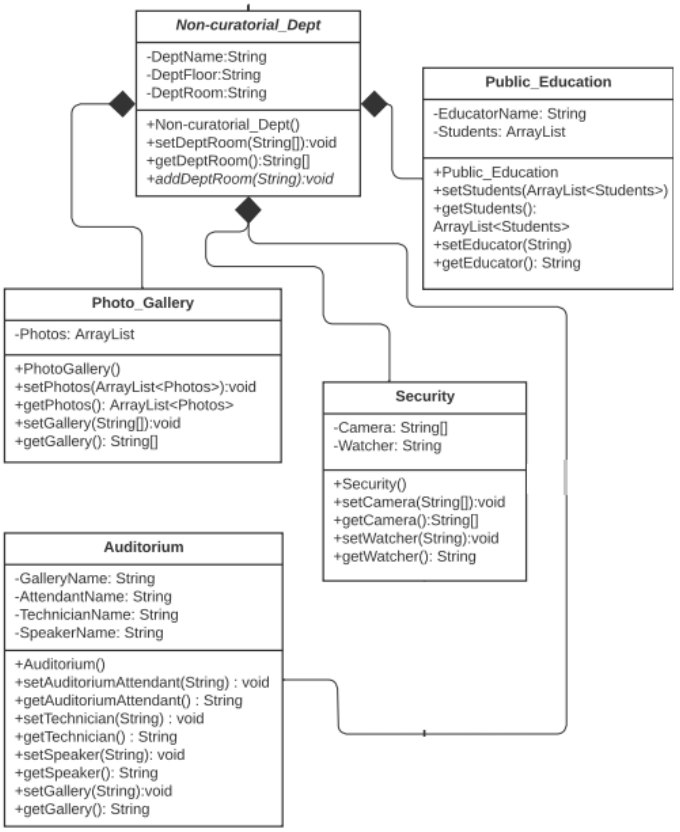
This sub-class contains information about the auditorium of the museum. Some important information is kept here such as gallery name, attendant name, technician's name, etc.

[1.2.2] Security

Security is actually a very important thing to consider in a museum. So, there is a security department under the non-curatorial departments. This class actually stores information about the cameras and the watchers.

[1.2.3] Photo Gallery

The photo gallery of a museum is absolutely one of the most crowded places. Here, in the photo gallery class, the storing system of photos is ensured.

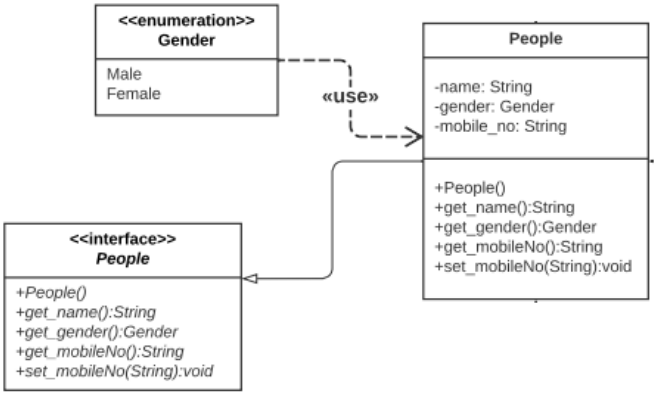


[1.2.4] Public_Education

One of the most important mottos of a museum is to spread knowledge to the people. This is why there is a public education class in the hierarchy. It deals with the educators and the students.

2. People

A museum is not imaginable without people. So, the people interface and the implementing class, named people, contain all the common information of all people involved in the museum.



[2.1] Internals

Not all people found in a museum are visitors. There are some internal bodies who always do the managerial part and other stuff in the museum. This class contains information about them. Internals can be divided into some other

subclasses like curators, BOD, students, developers, educators, etc. They are described here in brief.

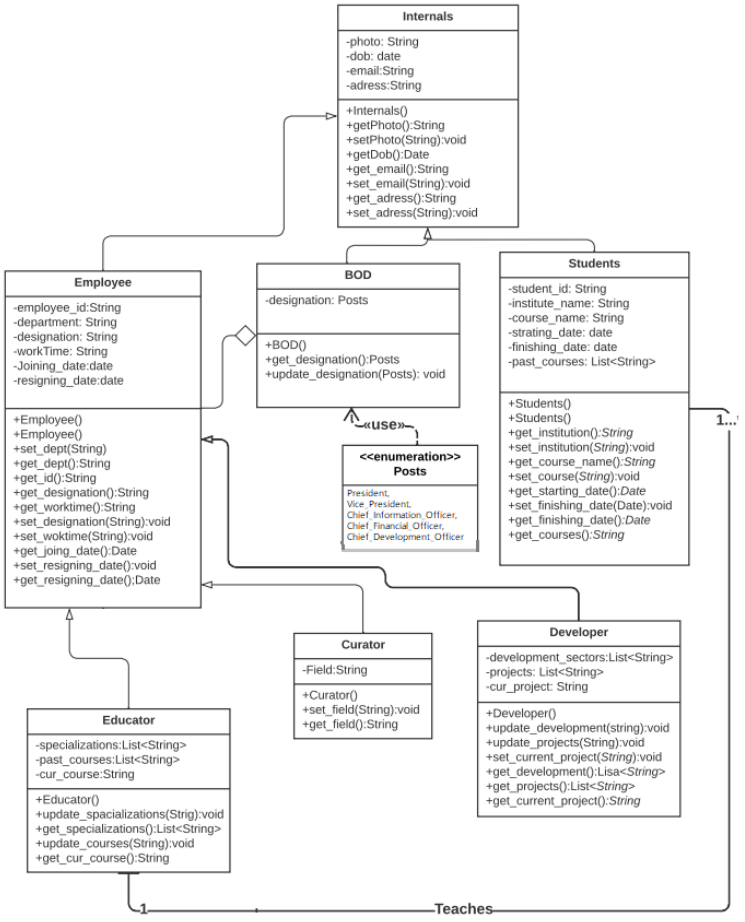


Figure: *Internal* class hierarchy

[2.1.1] BOD

There are five board members of the museum. BOD class contains information about those people including their dedicated designations.

[2.1.2] Students

This class stores information about the students of the public education department. It contains the institution name, course name, starting and ending date of the course, etc.

[2.1.3] Employee

There are several employees working in the museum. This class contains the id, department name, designation, worktime, joining date, and resigning date (in applicable cases). This class has some more subclasses.

[2.1.3.1] Curator

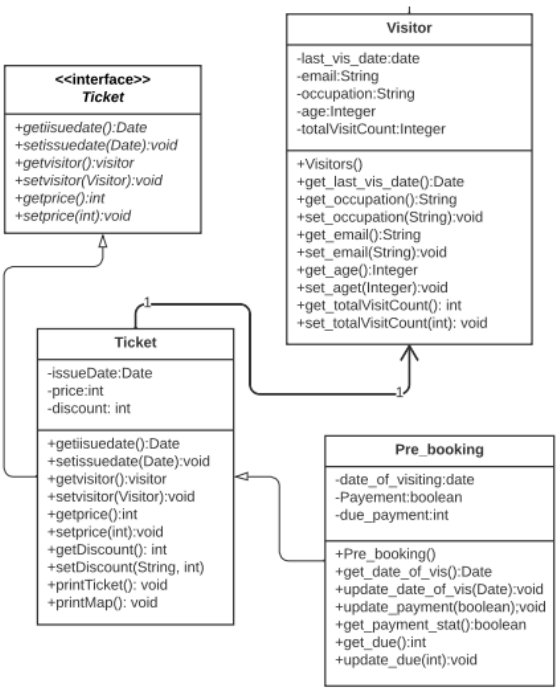
In the curatorial department, some curators do their job. This class contains information about them. There are three sections in the curatorial department, hardware, software, and language. All the curators take care of the artifacts of these three departments.

[2.1.3.2] Developers

This class stores information about the developers working in the museum. They mainly do software development. This class contains information about each of them, like, their previous projects, current project, and many more.

[2.1.3.3] Educators

This class contains information about the educators working in the public education department. That information includes the courses they teach, the programs they continue, etc. There is a teaching relationship between students and teachers.



[2.2] Visitors

A large number of people visit the museum every day. To store their information, this class is used. This class has a very close relationship with the ticketing system of the museum.

3. Ticket

This interface contains the necessary methods for calculating the price of a ticket and printing it. There is an implementing class with the same name. This class has some functionalities such as calculating discounted tickets on special days for regular visitors, issuing the date of selling tickets, choosing whether the ticket will be a pre-booking ticket or an onsite booking ticket, etc.

[3.1] Pre-booking

This class contains information about the pre-booked tickets, especially the visiting date set later.

Details about other classes:

- **AdminsController**

This class helps to run the admin and users page. Not all users can add or remove other users, this feature is implemented here. Again, adding or updating the users by the admin is also implemented in this class.

- **AuditoriumController**

This class is mainly created for controlling the auditorium department page. Adding, deleting, or updating the employees of the auditorium is managed using this class.

- **Curatorial_deptController**

This class contains all the methods of controlling the changes on the curatorial department page.

- **DashboardSceneController**

All the information and buttons we can find in the dashboard are controlled by the methods of this class.

- **DeveloperController**

This class helps with the controlling part of the Developer page. It contains methods for adding or removing a developer or updating information about the existing developers.

- **EducatorController**

This class helps with the controlling part of the Educator page. It contains methods for adding or removing an educator or updating information about the existing educators.

- **EmployeeController**

All the addition, deletion, or update of employee information are controlled by the methods of this class.

- **GalleryController**

The actions in the photo gallery page are controlled by the methods written in this controller class.

- **InventoryController**

The inventory class and the inventory page uses this controller class to perform the actions there.

- **LoginSceneController**

This class and its methods work with the login page. From ID and password matching to taking to the dashboard – this controller class does such jobs.

- **PEducationController**

This class works with the public education page. Selecting and maintaining the students and teachers for a course is done in this controller class. There is an interconnection of educators, students and the courses so that only valid educators can be set for each course and student.

- **SecurityController**

This controller class works with the security page of the application. Ensuring security by providing the necessary camera setup and the watcher is done here in this controller class.

- **StudentsController**

This controller class works with the student page in the application. Adding or removing a student from the database or updating a student's information is controlled in this class.

- **TicketController**

All the methods related to the ticketing section can be found here in this controller class. This class has methods for adding new visitors, calculating the price for the ticket for a particular visitor, printing the ticket and a map.

2.4 Discussion

In the project many javaFX is used for designing and demonstration purposes. In this project, many Java features are used. These features are described below with an explanation:

Interface:

An interface in Java is a blueprint of a class. It has static constants and abstract methods. The interface in Java is a mechanism to achieve abstraction. There can be only abstract methods in the Java interface, no method bodies. It is used to achieve abstraction and multiple inheritance in Java. Our main three interfaces are "Departments", "People" and "Ticket".

Encapsulation:

Encapsulation is defined as the wrapping up of data under a single unit. It is the mechanism that binds together code and the data it manipulates. Another way to think about encapsulation is that it is a protective shield that prevents the data from being accessed by the code outside this shield.

Polymorphism:

Polymorphism is a concept of object-oriented programming that allows us to perform a single action in different forms. It is an OOP design that empowers classes with various functionalities to execute or share a common interface. The helpfulness of polymorphism is that code written in various classes has no impact on which class it has a place in since they are utilized similarly.

Function overriding:

If a subclass (child class) has the same method as declared in the parent class, it is known as method overriding in Java. In other words, If a subclass provides the specific implementation of the method that has been declared by one of its parent classes, it is known as method overriding. We used it in the subclasses of people class where necessary.

Function overloading:

If a class has multiple methods having the same name but different in parameters, it is known as Method Overloading. We used it in the constructors and various other methods in different classes.

Collection framework:

The Collection in Java is a framework that provides an architecture to store and manipulate a group of objects. Java Collections can achieve all the

operations that you perform on data such as searching, sorting, insertion, manipulation, and deletion. Java Collection means a single unit of objects. The Java Collection framework provides many interfaces (Stack, Set, List, Queue, Deque) and classes (ArrayList, Vector, LinkedList, PriorityQueue, HashSet, LinkedHashSet, TreeSet). We used stack to implement the “GoBack” button so that each time you press the button, it takes you to the immediately visited previous page. We used map in the function of “printTicket” to keep value of the parameters in Jaspersoft studio to change information of a visitor dynamically. And the use of ObservableList and ArrayList is plenty in every controller class to work with the information of an object related to each class.

Exception Handling:

The Exception Handling in Java is one of the powerful mechanisms to handle the runtime errors so that the normal flow of the application can be maintained. In Java, an exception is an event that disrupts the normal flow of the program. It is an object which is thrown at runtime. The java.lang.throwable class is the root class of the Java Exception hierarchy inherited by two subclasses: Exception and Error. Exception handling helped us a lot to read the errors in the console and debug them smoothly. The use of exception handling in our project is huge. From just opening the window to printing the ticket, different exception handling is used everywhere. Some most common exceptions were IOException, RuntimeException, NullPointerException, FileNotFoundException, IllegalArgumentException, ClassNotFoundException etc.

These are the major features that are used in this project.

2.5 Use of OOP

In this project, we used all the object-oriented features of Java. In our UML diagram, the inheritance relationship can be seen. Here are some of them mentioned:

- There is a *people* class that is inherited by two classes, *Visitor* and *Internals.*
- The *Internal* class is inherited by *Employee, BOD,* and *Students* classes.
- *Educator, Curator,* and *Developer* classes inherit the *Employee* class.
- *Departments* is an interface that is implemented by two classes.
- The *Non-curatorial_Department* class is inherited by four different classes named as *Auditorium, Security, Photo_Gallery,* and *Inventory.*

The data here is provided in such a way that it is bound to a particular class only. No data goes across its boundary. This is a very good use and implementation of encapsulation. Maximum code reusability is ensured here by making some superclasses and keeping all the common methods there.

Again, we did not call the database multiple times for doing a particular task, rather we tried to store all data in a secured class. The codebase is divided into separate modules where children classes can send data to the parent classes using *super* variables.

Function overriding and overloading are two of the most useful and powerful features of Java. We have used these two features multiple times ensuring the code reuse and inheritance property of object-oriented programming.

Collection frameworks of Java have been used multiple times in this project. This use eases the difficulty of storing data. Almost every method in this project uses exception handling. It helped to figure out the exceptions raised in the codes and helped it solve in time.

2.6 Code repository

The codes with necessary files are kept in GitHub. Here is the GitHub repository link (public):

https://github.com/reckless-meherun/CSE-2112-JavaFx-Project-Bangladesh-Computer-History-Museum-Management

Conclusion

3.1 Challenges and Solutions

1. We faced difficulties in building the user interface as there is not much documentation about the JavaFX frontend library or using SceneBuilder. Still we managed to bring about an attractive UI at the end.
2. While including the database, we started using MySQL at first. But that was very tedious as the data had to be set in each member's pc manually. MySQL doesn't work if the other person doesn't have it installed in their PC. Then we managed to replace MySQL with SQLite. We got a chance to maintain and update the data in the database smoothly as SQLite didn't make us face those difficulties.
3. Implementation of the connection between two classes with no inherit-property between them was really a tough job to do. However, we pulled it out at the end.
4. We faced difficulties while setting the image paths and using the Jaspersoft to print ticket as a PDF as JavaFx caused a lot of problems with relative paths. However, we worked extremely hard to make our system path independent so that it can be used from any PC just by downloading some libraries.

3.2 Future Development

This application can be developed more by using networking for data sharing and storing in cloud storage services. Then multiple users will be able to use this app at a time and a user will be able to recover his/her password using email in case they forget it. We could not do it in this project because of the shortage of time and lack of networking-based knowledge. Besides, we plan on making everything more dynamic such as extending the notice board.

The concept of a Computer History Museum is not very popular in our country. As a result, there is no physical existence of such a museum in Bangladesh. Yet, we expect that there will be a computer history museum in our country and thus we will use this application precisely.

3.3 At the end

This project really helped us to use and learn many basic and advanced features of Java and JavaFX. The little amount of CSS used here to design the UI also helped us know the basics of it. This project helped us in gaining valuable information and practical knowledge on several topics like using Java features, use of oop principal, CSS, usage of responsive templates, designing ticket using Jaspersoft studio and management of databases using SQLite. The entire system is secured. Also, the project helped us understand the development phases of a project and the software development life cycle. We learned how to test different features of a project.

In this project, we have gained lots of experience and had to face many obstacles too. Making software is completely a new and practical experience for us. So, we had to learn things from the beginning. Through this work, we have introduced ourselves to the JavaFx (special effects in the Java language) library. Our thinking and imagination capability have grown. We have developed our

communication skills by cooperating with group members. It truly was an amazing experience for us.