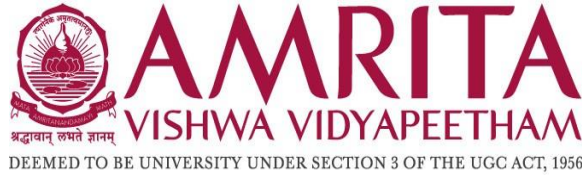


# “Simplified Timetable Management and Substitution System for Amrita College using Java”

*As a part of the subject*

**OBJECT ORIENTED PROGRAMMING in Java**



**Centre for Computational Engineering and Networking**

**AMRITA SCHOOL OF ENGINEERING**

**AMRITA VISHWA VIDYAPEETHAM**

**COIMBATORE - 641 112**

**(INDIA)**

**JULY- 2023**

**BATCH B GROUP 5**

## **GROUP MEMBERS:**

S.NO	NAME	ROLL NO
1	M NILAVARASAN	CB.EN.U4AIE22139
2	M SANTHOSH	CB.EN.U4AIE22152
3	KL AMRITHA NANDINI	CB.EN.U4AIE22126
4	CH ARJUN	CB.EN.U4AIE22106

# MODULE-1 “Linking All Class & Admin-Side”

## 1.1. DESCRIPTION OF THE MODULE:

A part of this module is linking all other modules using the concept of OOP's (interface's, abstract, inheritance, etc) so that the code will be more precise, and also each Page represents a GUI and a class. And all class's have many things in common, so instead of writing the same thing again in each class, I have used interface and abstract-class's to store everything in common-place, and all the other class extends this abstract and interface class's. So using the concept of inheritance the child class can inherit all the methods, and variables are inherited. So in short a part of this module is to link all class's. In this part of my module there are three class's

***Interface (“InterfaceClass”):*** This is a interface class where I will store all the variables, methods etc. which are common to both the admin and teacher side, in a common place, so that I don't need to redeclare again

***Abstract class (“TeacherStorage”):*** This is abstract class where I will store the things which are common only to the teacher-side class's.

***Abstract class (“AdminStorage”):*** This is abstract class where I will store the things which are common only to the Admin-side class's.

The other part in my module is the admin side class's where, the admin is the head of all teachers he can access all teachers and he can fetch all their personal information, their timetable and also message admin. So the full admin side is taken control by me. In short this class will accept or reject the pass that the teachers send to me, also the admin will recruit new teachers, with the credentials the admin gives to a teacher, the teacher can finish the verification process only with the help of the details given by the admin. So, this module has a total of 5 classes, and all these class's are related only to the admin side.

***Class (“AdminAllTeachersInfo”):*** This class will display all the details that the teacher has filled in the signup page, using a table. This class is basically to show all the info of the teachers in a common page.

***Class (“AdminHomePage”):*** This class is the home-page of the admin side class's, so this class just mainly contains only a GUI-part of the admin side class.

***Class (“AdminNotification”):*** This class is a important part in our java project, where the admin recivies the notification from the teacher stating that they need

of approval of the pass, so In this part admin will either approve or reject the pass, send by that specific teacher. In short this class is for accepting the pass that the teacher has send.

***Class (“AdminRecruitTeachers”):*** Suppose if there is a scenario where the admin recruit a new teacher, then that teacher will be given credential details name, collegeID (unique) and password. So with the help of this details only a teacher can go through the verification page.

***Class (“TeacherAdmin”):*** This is the place where the collegeID given to teachers that ID is entered by the admin, and now the admin can access all the details their personal-information, their timetable and also message admin etc.

Overall this module focus on linking all the other class’s in a common page, and path. And also all the class’s related to the admin side.

## 1.2. MOTIVATION OF THE MODULE:

The motivation behind developing this module was driven by many factors the main reason is to create an admin side and also to differentiate the admin and teachers side. And the interface and abstract class make the code very optimized and save’s time and space also. So the main motivation of this module is the following goals:

***Reusing Code:*** We wanted to avoid repeating code and make development more efficient. By using interfaces and abstract classes, we could store common elements, variables, and methods in one place. This way, we could easily reuse them in different parts of the project. It also made it easier to update or modify the code since changes made in one location would automatically apply to all related parts.

***Organized and Easy-to-Read Code:*** We aimed to make the code well-structured and easy to understand. Each page in the project represents a specific graphical interface and class. By using inheritance, the code became more organized and modular. It improved readability, making it easier for developers to navigate and comprehend the different parts of the project. This also made troubleshooting and maintenance simpler.

***Flexibility and Scalability:*** We wanted to ensure that the project could easily adapt to future changes and additions. By using interfaces and abstract classes,

we created a flexible foundation. This allowed us to extend or modify functionalities without major code rewrites. New features and modules could be integrated smoothly by extending existing classes or implementing interfaces. This flexibility ensures that the project can grow and evolve over time.

***Efficient Admin-Side Management:*** We focused on streamlining administrative tasks and making them easier to handle. The admin-side classes were designed to provide dedicated interfaces and functionalities for administrators. This empowered administrators to efficiently manage teacher-related information. They could access teacher details, handle notifications, approve or reject requests, and recruit new teachers through user-friendly interfaces. This streamlined the administrative process and reduced manual work.

In summary, the motivation behind Module-1 was to improve code reusability, enhance code organization and readability, support flexibility and scalability, and streamline administrative management. By linking classes and using object-oriented principles, the project became easier to maintain, expand, and handle administrative tasks efficiently.

## **1.3 RELEVANCE OF THE MODULE IN THE SYSTEM:**

Module-1, which focuses on linking all classes and implementing the admin-side functionalities, is very important for the system. It plays a key role in managing the system effectively and making things easier for administrators. The relevance of this module can be seen in the following ways:

***Admin Control:*** The module allows the system to have an administrator who oversees all the teachers. This helps in maintaining proper control and supervision of teacher-related activities within the system.

***Simplifying Admin Tasks:*** By providing specific classes for admins, the module makes administrative tasks simpler and more organized. Admins can easily access teacher information, manage notifications, approve or reject requests, and recruit new teachers using user-friendly interfaces. This reduces manual work and increases efficiency.

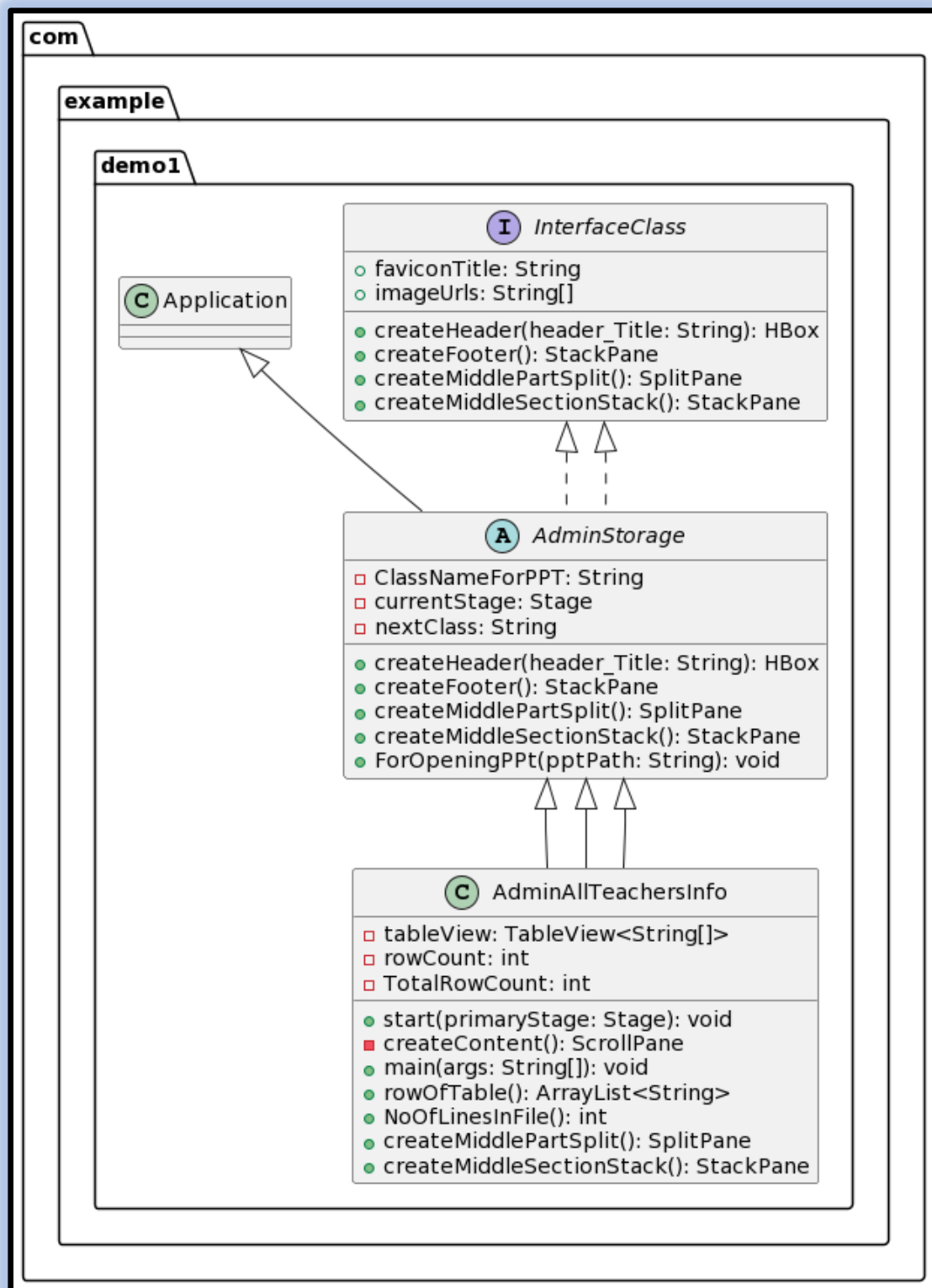
***Ensuring Secure Verification:*** The module ensures a secure process for verifying new teachers. Admins assign unique credentials like names, college IDs (unique), and passwords to each teacher. These credentials are necessary for teachers to complete the verification process. By managing the verification, admins ensure that only eligible and trustworthy teachers are recruited.

***Scalability:*** The module's use of object-oriented concepts, like interfaces, abstract classes, and inheritance, makes the system flexible and scalable. It becomes easier to add new features and modules to the system by extending existing classes or implementing interfaces. This flexibility allows the system to adapt to future needs and expand smoothly.

In summary, Module-1 is highly relevant as it establishes admin control, simplifies administrative tasks, ensures secure verification of teachers, and provides scalability to the system. By connecting classes and implementing admin-side functionalities, this module greatly contributes to the smooth and efficient management of the entire system.

## 1.4. UML OF THE MODULE

### 1.4.1. UML of “AdminAllTeachersInfo” Class.



*Figure 1 UML of AdminAllTeachersInfo Class.*

## 1.4.2. UML Of “AdminHome” Class.

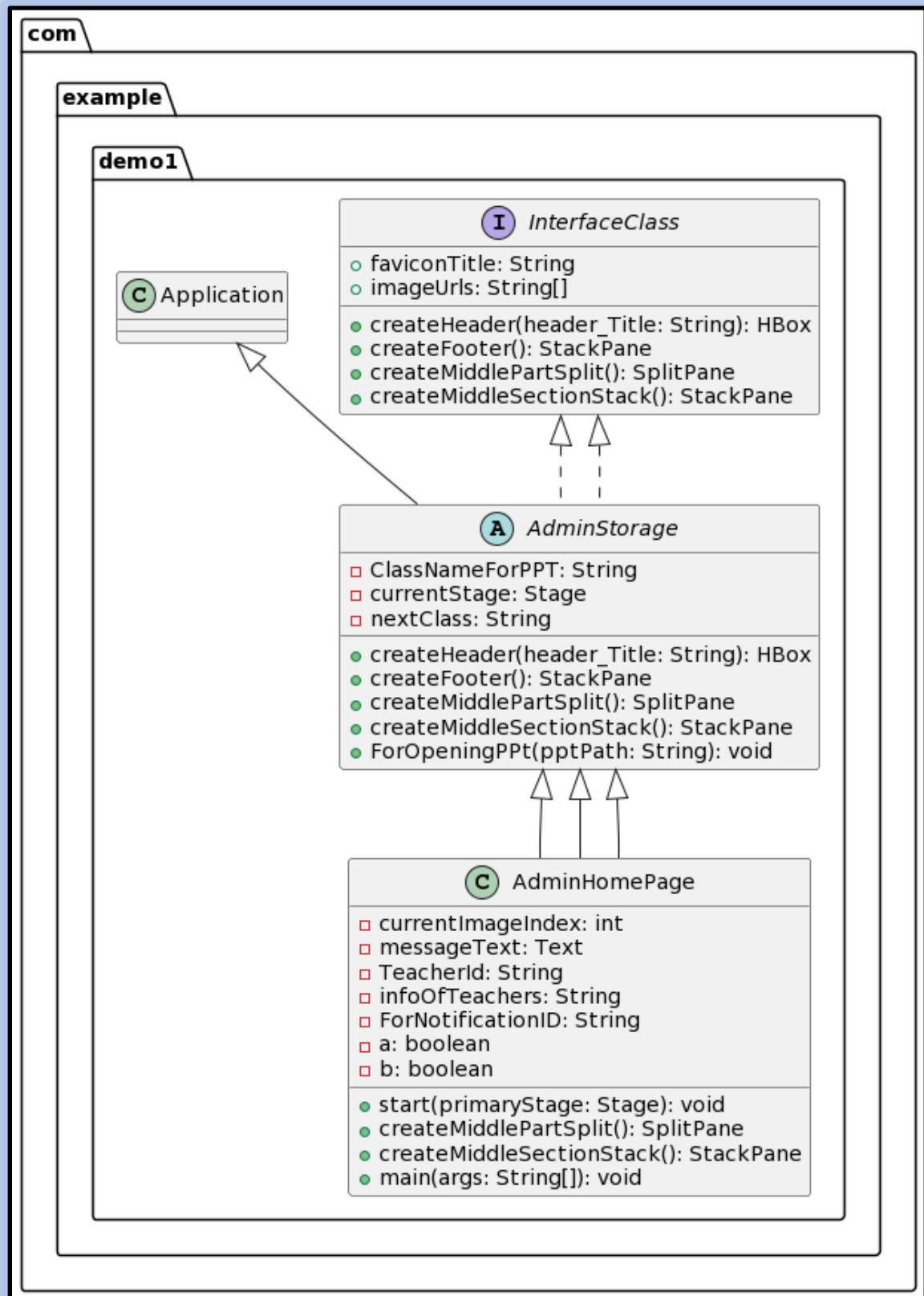
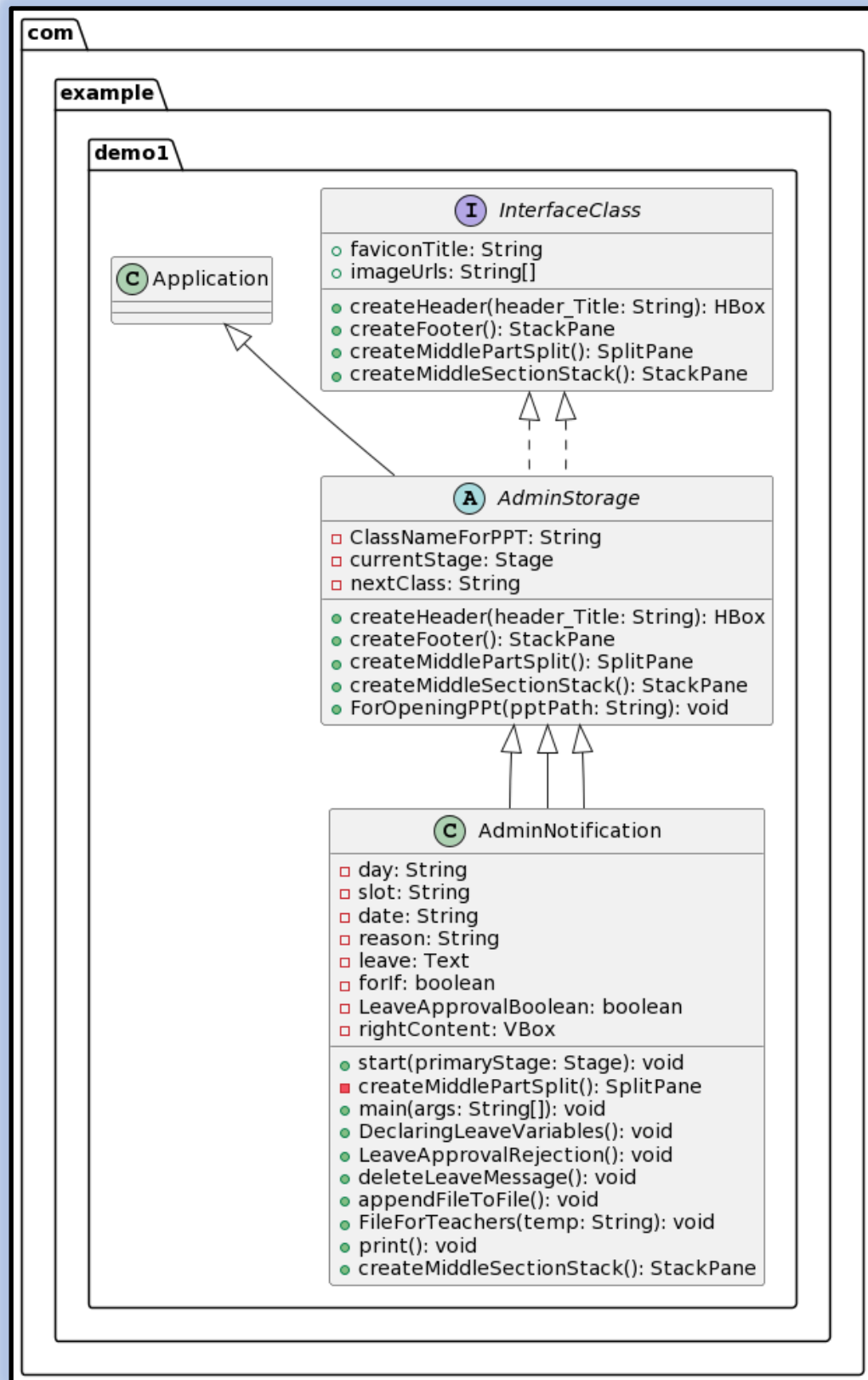


Figure 2 UML Of “AdminHome” Class.

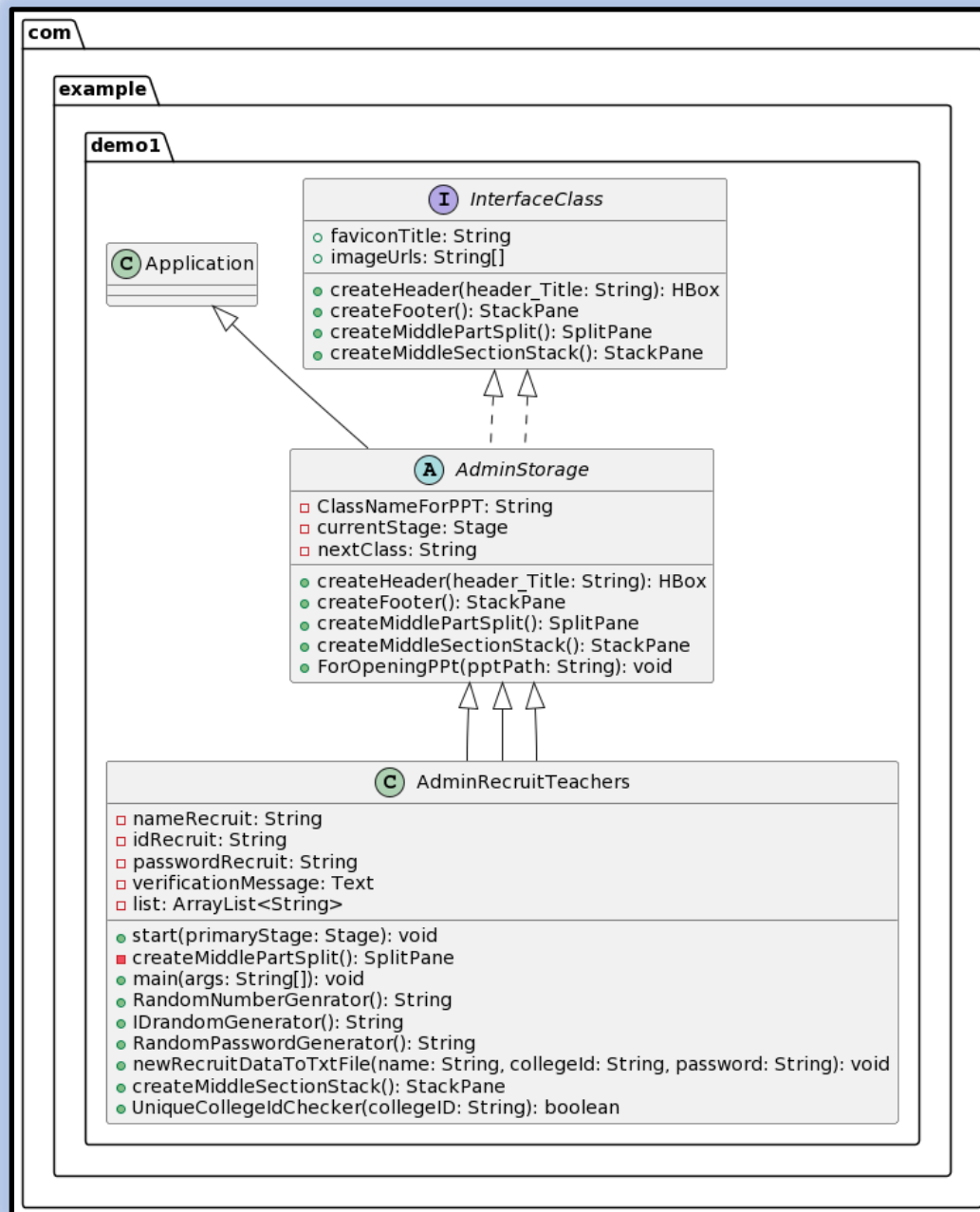
### 1.4.3. UML Of AdminNotification Class.



*Figure 3 UML Of AdminNotification Class.*



## 1.4.4. UML Of AdminRecruitTeachers Class.



**Figure 4** UML Of AdminRecruitTeachers Class.

## 1.4.5. UML Of AdminStorage Abstract Class.

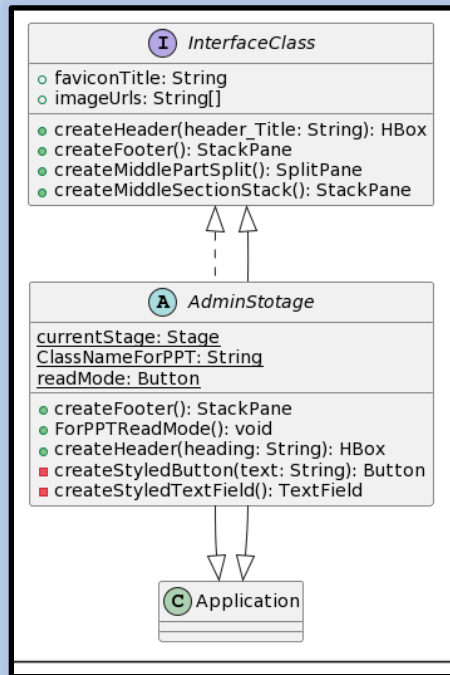


Figure 5 AdminStorage Abstract Class.

## 1.4.6. UML Of TeacherAdmin Class.

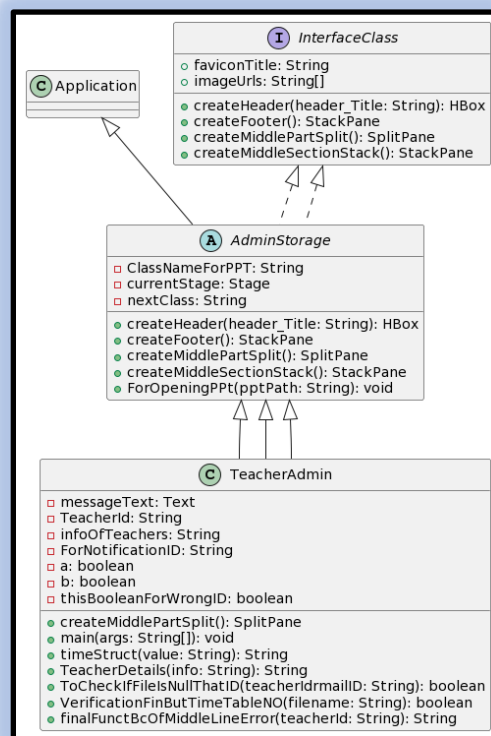


Figure 6 TeacherAdmin Class.

## 1.4.7. UML Of InterfaceClass interface Class.

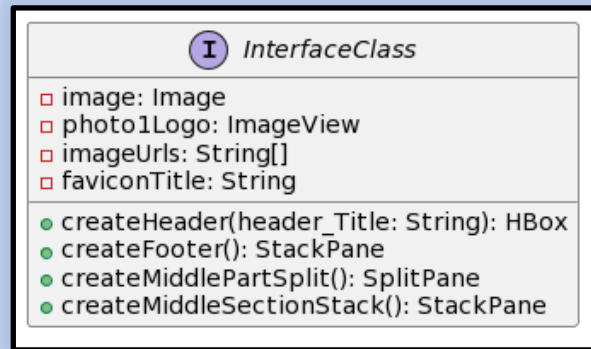


Figure 7 UML Of InterfaceClass interface Class.

## 1.4.8. UML Of TeacherStorage Abstract Class.

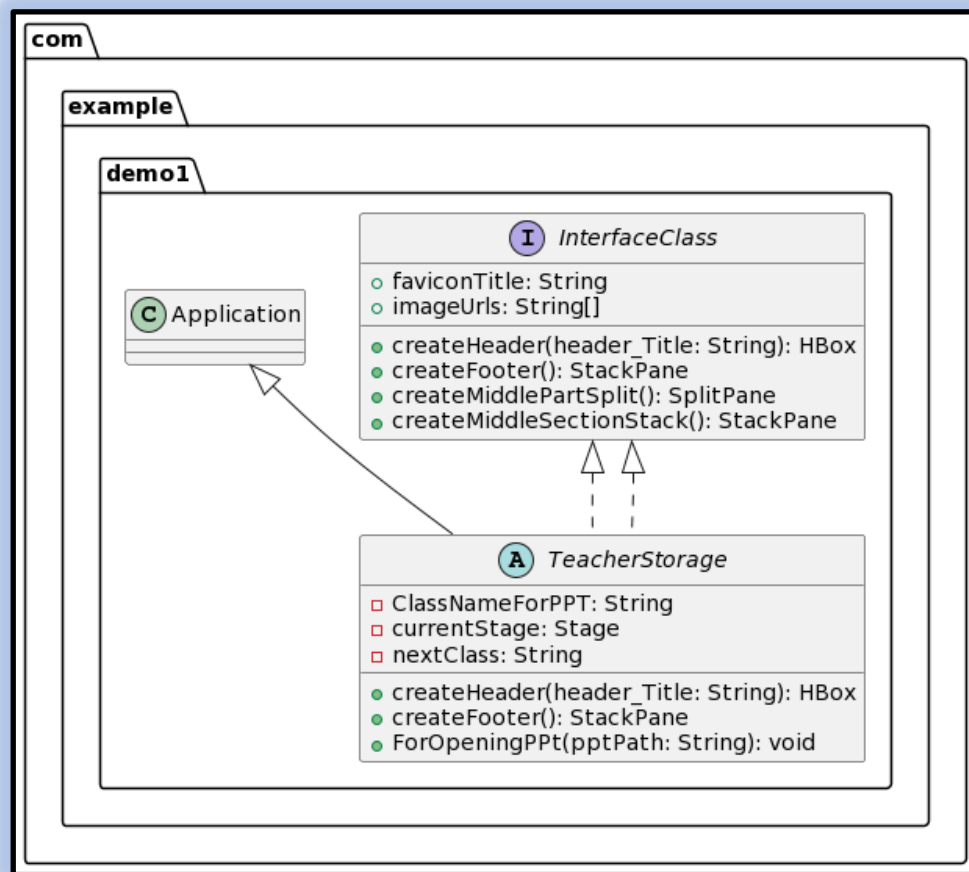
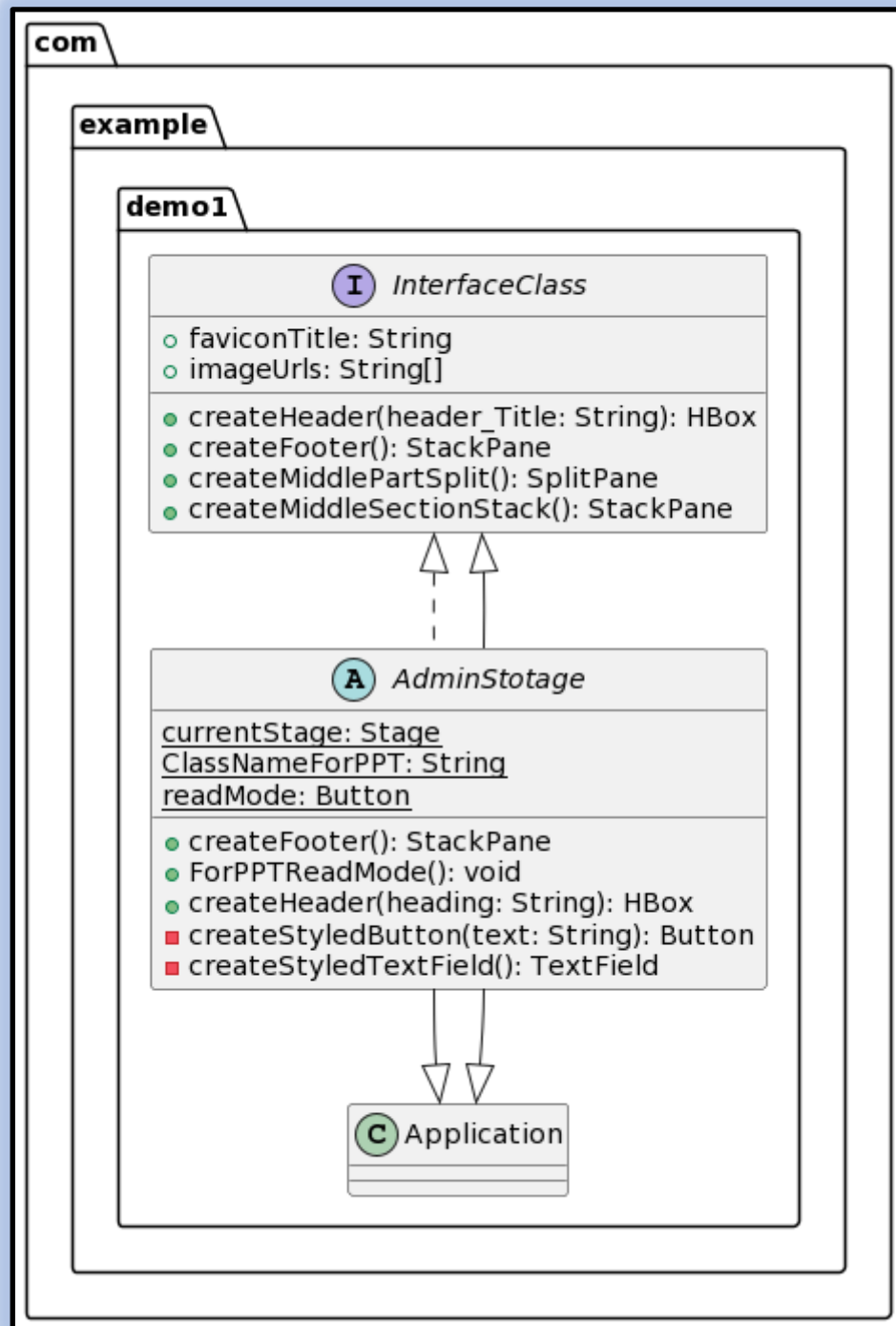


Figure 8 UML Of TeacherStorage Abstract Class.

## 1.4.9. UML Of AdminStorage Abstract Class.



*Figure 9 UML Of AdminStorage Abstract Class.*

## 1.5. OUTPUT OF THE MODULE

### 1.5.1. AdminHeader



*Figure 10 Output Image Of AdminHeader.*

### 1.5.2. AdminFooter



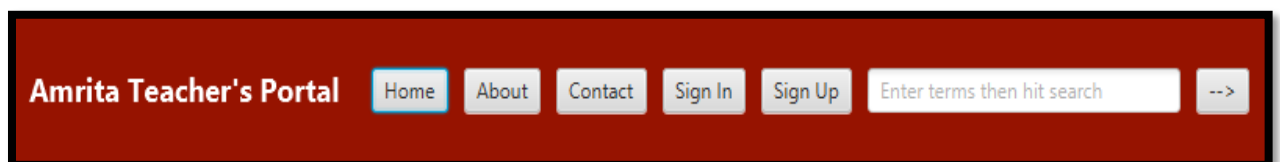
*Figure 11 Output Image Of AdminFooter*

### 1.5.3. TeacherFooter



*Figure 12 Output Image Of TeacherFooter.*

### 1.5.4. TeacherHeader



*Figure 13 Output Image Of TeacherHeader.*

## 1.5.5. AdminAllTeachersInfo

All Teachers Info														
<a href="#">Home</a> <a href="#">Teacher-Info</a> <a href="#">All Teachers info</a> <a href="#">NewRecruit</a> <a href="#">Period Reservation</a> <input type="text" value="Enter terms then hit search"/> <a href="#">--&gt;</a>														
Name	College ID	Password1	Mail ID	Password2	Name	Gender	D.O.B	Phone No	Address	Qualification	ID	Campus	Subject	Department
Nilavarasan	1234012	nila123	01nilavarasan@gmail.com	nilavijay	Nilavarasan	male	30/3/2005	996259204	trichy	btech	1234012	coimb	JAVA	AI
Manimaran	121212	mani321	smmanimaran@yahoo.com	maninilapriya	Manimaran	male	30/3/2005	9962592024	trichy	M-Tech	121212	chennai	CODER	CSE
Priya	321321	priyaluvnila	priya@gmail.com	priyaluvnila	Priya Sengamalam	female	1/6/1975	9962500191	Chennai trichy-snakoiaie	Arts	321321	chennai	Cook	food
Magima	454545	AnnaLuv	Anna@gmail.com	nilaanna	Magima	female	3/3/2010	1234567890	Bangalore	10thstd	454545	bangalore	social	school
Bhagyaraj	1234567	bestfriend	bestfriendever@gmail.com	sjirfriend	Bhagyaraj	male	26/3/2005	9988776655	assam	12th	1234567	amritapuri	CSE	sjirs
Tarun	1234321	FakeFriend	tarunE@gmail.com	tarun321	Tarun E	male	13/12/2004	4321432112	pudukothaie	completed	1234321	trichy	python	CSE
nila	892125	jMBEi79i	-	-	-	-	-	-	-	-	-	-	-	-
nilaaaa	844194	5yVt5EU8	-	-	-	-	-	-	-	-	-	-	-	-
Vijay	116952	LeoLeo	-	-	-	-	-	-	-	-	-	-	-	-
LEoLEo	679469	75vDsKB	-	-	-	-	-	-	-	-	-	-	-	-

Figure 14 Output Image Of AdminAllTeachersInfo.

## 1.5.6. AdminHome

Amrita University Teacher's-Management

[Home](#)
[...](#)
[Teacher...](#)
[All Teacher...](#)
[NewR...](#)
[Period Reserv...](#)

[...](#)

### HOME PAGE

Current Time: 13:14:40

Current Date: 23 June 2023

Select a date and add notes here

Add notes on that day,.

Website for Admin  
A Project by Group 5

Figure 15 Output Image Of AdminHome.

## 1.5.7. AdminNotification

The screenshot shows a web application titled "Amrita University Teacher's-Management". The navigation bar includes buttons for "Notification", "Teach...", "All Teache...", "New...", and "Period Reser...", along with a search bar. The main content area is split into two columns. The left column features the Amrita Vishwa Vidyapeetham logo and the text "AMRITA VISHWA VIDYAPEETHAM UNIVERSITY" with the motto "श्रद्धावान् लभते ज्ञानम्". The right column displays a notification for ID:121212, titled "Notification's". It includes fields for "LEAVE REQUEST:", "ID: 121212", "DAY: Wednesday", "SLOT: 10:40 AM-12:20 PM", "DATE: 2023-06-23", and "Reason: simply". Below these fields are "Approve" and "Reject" buttons. The footer of the application reads "Website for Admin" and "A Project by Group 5".

*Figure 16 Output Image Of AdminNotification.*

## 1.5.8. AdminRecruitTeachers

The screenshot shows a web application titled "Amrita University Teacher's-Management". The navigation bar includes buttons for "Recruit Teachers", "Teac...", "All Teach...", "Ne...", and "Period Res...", along with a search bar. The main content area is split into two columns. The left column features the Amrita Vishwa Vidyapeetham logo and the text "AMRITA VISHWA VIDYAPEETHAM UNIVERSITY" with the motto "श्रद्धावान् लभते ज्ञानम्". The right column displays a form titled "New Recruit". It includes input fields for "\*enter name of new recruit", "\*assign college-ID", and "\*assign password", each with a corresponding "Random Number" or "Random Password" button. A green "Fill" button is located below the password field. The footer of the application reads "Website for Admin" and "A Project by Group 5".

*Figure 17 Output Image Of AdminRecruitTeachers.*

# MODULE-2 “Teacher Verification and Sign-Up Page”

## 2.1 DESCRIPTION OF THE MODULE:

This module is responsible for the verification page and its following Sign-Up page as well as the Personal Details page that is needed for signing up a new teacher. This module also includes the home page for the Teacher’s side that opens up when starting the application.

This module consists of the following classes:

**Class (“Step1”):** This is the home page of the teacher-side classes, where the admin-side and the teacher-side classes are differentiated using color. Red is for the Teacher side classes, and blue is for the Admin Side classes. This is just a home page displaying a quote and an image.

**Class (“Step2”):** This is the sign-up, page or to be more precise Verification page, where a user can come to this page, and get verified. The name, college ID, and password will be provided by the admin. Else; they cannot enter after this page. Only when the user enters the proper college ID and others. The user will be redirected to a sign-up page.

**Class (“Step3”):** This class is where the user comes, if and only if he/she passes the “Step2” class. Once the user comes to this class, they have to enter their email, and password. These credentials will be stored, and further, when they sign in.

**Class (“Step4”):** This is the final page in the flow or the process of signing up a new teacher. The teacher now has to fill in all their personal details on this page. And all this data again is stored, and all this data of the teacher is visible to the admin. And once the staff fill’s the page, he/she will be a teacher in this respective college. With this, the process of signing up a new user is finished. Once this part is done, the program automatically goes to the Sign In page.

After this, the user can now log/sign in and access and utilize this application for timetable updation.



## 2.2 MOTIVATION OF THE MODULE:

The motivation behind developing this module is to provide a secure and user-friendly Sign-up and verification system and a home page for the application. The home page is essential for any application we make and is the first page that comes up when we start the program. Login, timetable updation, getting the teacher information, and other options provided by the other modules of this project can only be accessed once the teacher signs up.

## 2.3 RELEVANCE OF THE MODULE IN THE SYSTEM:

The user must be first recruited by the admin on the admin side of the application, and using the information entered, the newly added teacher should verify with their name, college ID, and password provided by the admin.

For verification, the teacher has to enter the name, college ID, and password given by the admin. If these are correct then it verifies it and goes on to get the teacher's sign up where the user can give their email ID and password followed by a personal information page.

Only once these details are filled in, can a teacher update their timetable and do use the other facilities given by this program.

1. ***User Verification:*** The program then checks if the data entered and the basic teacher information on the admin side are the same and verifies it.
2. ***New teacher sign-up:*** This module on the sign-up page enables teachers to register themselves as users of the application. It involves the inputting of the email ID and password of the user.
3. ***User Management:*** This module also allows the application to maintain a database of registered teachers, and store their credentials such as name, email id, address, educational qualifications, department, etc securely.

## 2.4 UML OF THE MODULE:

### 2.4.1 UML for Step1 class:

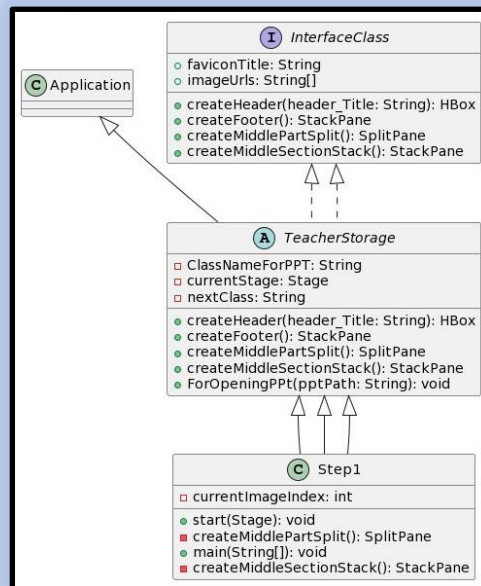


Figure 18 UML for step1 class

### 2.4.2 UML for Step2 class:

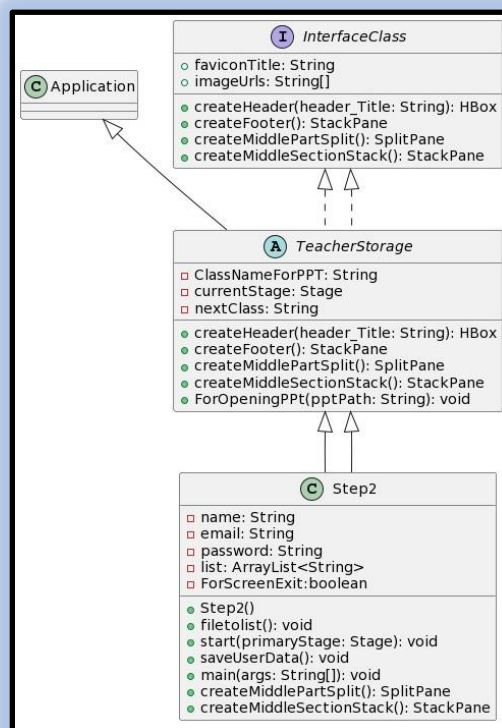


Figure 19 UML for step2 class

## 2.4.3 UML for Step3 class:

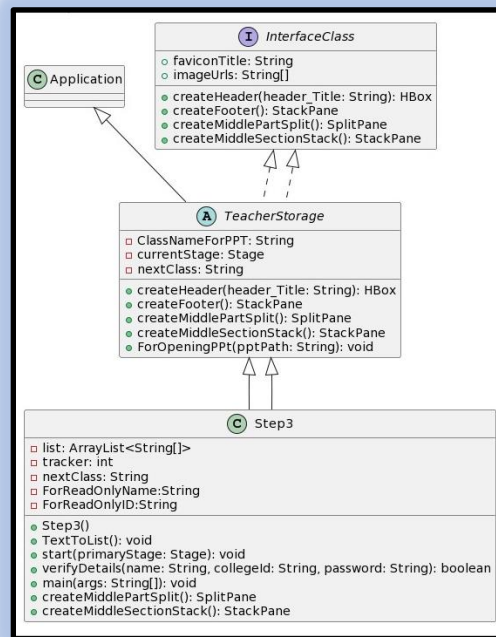


Figure 20 UML for step3 class

## 2.4.4 UML for Step4 class:

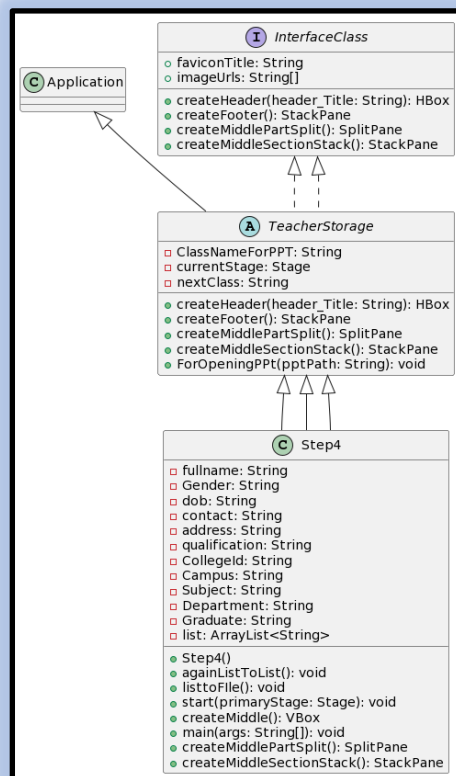
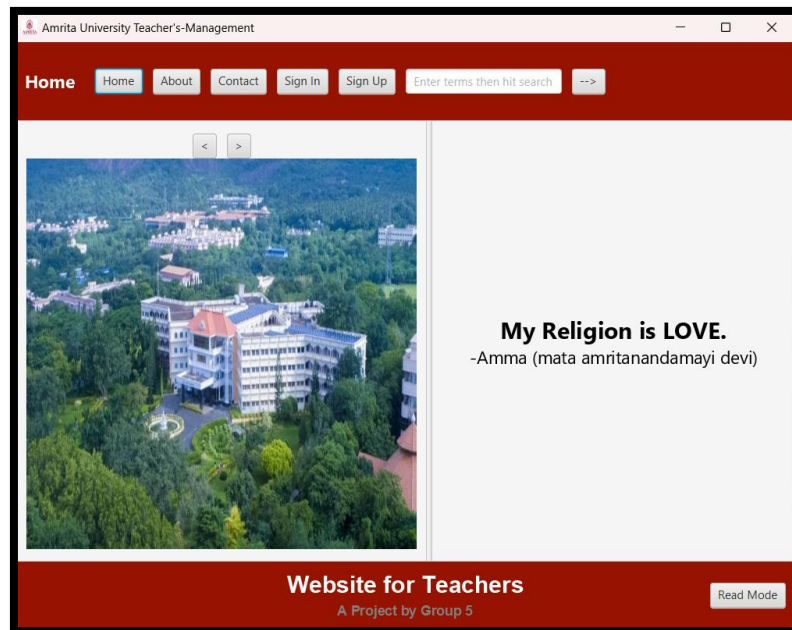


Figure 21 UML for step4 class

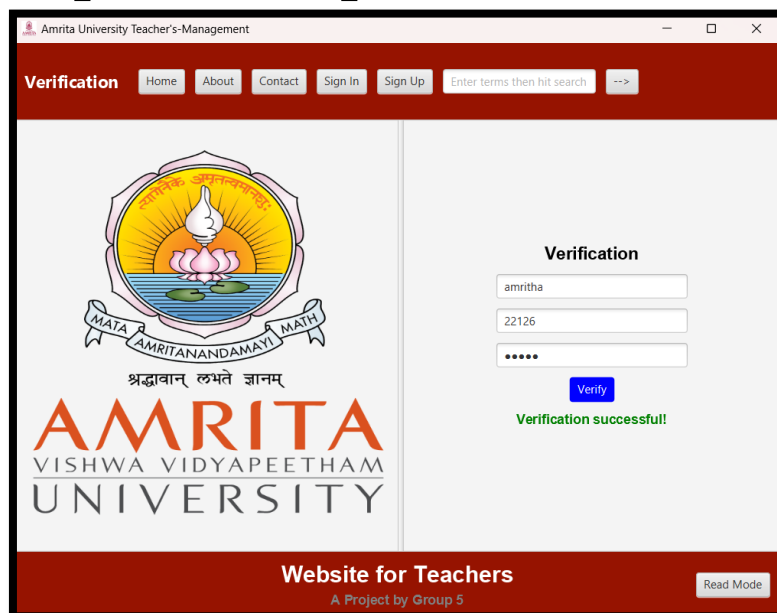
## 2.5 OUTPUT OF THE MODULE:

### 2.5.1 Output for Step1 class:



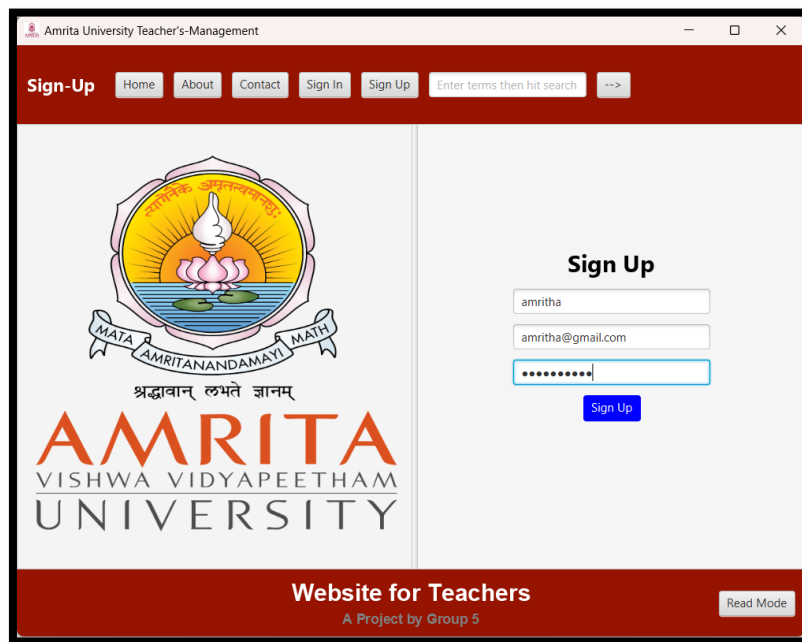
*Figure 22 Home page for teacher's side*

### 2.5.2 Output for Step2 class:



*Figure 23 Verification page*

## 2.5.3 Output for Step3 class:



The screenshot shows a web browser window titled "Amrita University Teacher's-Management". The page has a red header with navigation links: "Sign-Up", "Home", "About", "Contact", "Sign In", and "Sign Up". A search bar is also present. The main content area features the Amrita University logo on the left and a "Sign Up" form on the right. The form includes input fields for "amritha", "amritha@gmail.com", and a password field with masked characters. A "Sign Up" button is at the bottom of the form. The footer contains the text "Website for Teachers" and "A Project by Group 5", along with a "Read Mode" button.

Figure 24 Sign Up page

## 2.5.4 Output for Step4 class:



The screenshot shows a web browser window titled "Amrita University Teacher's-Management". The page has a red header with navigation links: "Personal-Info", "Home", "About", "Contact", "Sign In", and "Sign Up". A search bar is also present. The main content area features the Amrita University logo on the left and a "Personal Information" form on the right. The form includes input fields for "amritha", "female", "9/11/2003", "983434343", "abcde, cdddef, qwerty", and "aaaaaaaaa". A "Submit" button is at the bottom of the form. The footer contains the text "Website for Teachers" and "A Project by Group 5", along with a "Read Mode" button.

Figure 25 Personal information page

## MODULE-3 “Sign-In and Timetable Creation”

### 3.1. DESCRIPTION OF THE MODULE:

**Sign in:** The “Step5” module allows user to sign into their account. This class is sign-in page for teachers, teachers can sign-in if they sign-ups, also. The user can either enter his name, collegeID, emailID and the password the user enters is the password, that the user enter initially during sign-up. Only if the password, and the name or collegeID or email ID matches the user can enter, this page is similar to any sign-IN page, also it has another forgot-password.

**Forgot password:** The “Step 6” module allows the user to change the existing password.

**Teacher’s Timetable:** The “Step7” module allows the teachers to set their timetable.

This is one of the important pages in the project. It allows the new teachers to allocate their own timetable. Just after signing in the teachers can enter this page, and they can update their old timetable.

### 3.2. MOTIVATION OF THE MODULE:

The sign in class “Step5” was to implement a sign-in functionality for a Java application. The class was developed as part of a larger project that required user authentication and access control.

By providing a sign-in feature, the class enables users to securely access specific functionalities or resources within the application.

The Timetable class “Step7” to be building a graphical user interface (GUI) application for creating and managing timetables.

The motivation behind this class could be to streamline the process of timetable creation and management, making it more efficient and user-friendly for teachers or administrators who need to organize their schedules.

### 3.3. RELEVANCE OF THE MODULE IN THE SYSTEM:

The sign in part in the module assures that all user's are valid individuals(teachers) of the collage. It also makes it easier for the admin to get the details of the faculties inside the campus. The other modules can be only accessed after logging in with your respective data.

The Timetable part in the module helps the faculties to allocate as well as update their timetables. The new teachers can create their own timetable. The faculties can update their timetables when they need and as a result, time can be utilized in a more fruit full way. The teachers whose syllabi are pending can update their timetables to complete the timetables as well as use time effectively.

### 3.4. UML OF THE MODULE

#### 3.4.1. UML of “Step5” Class.

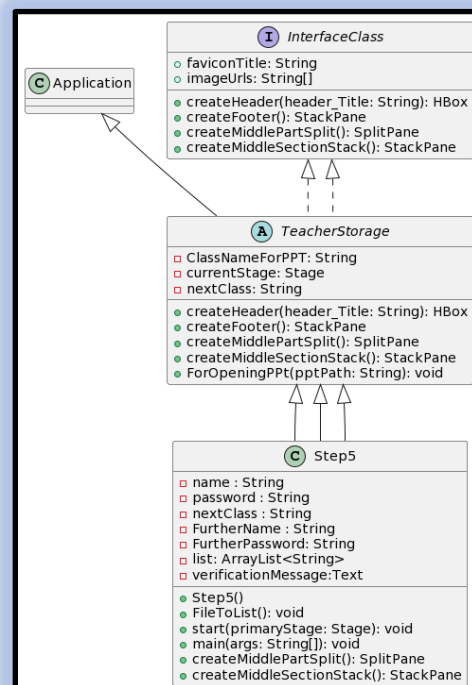


Figure 26 UML of “AdminAllTeachersInfo” Class.

### 3.4.2. UML of “Step6” Class.

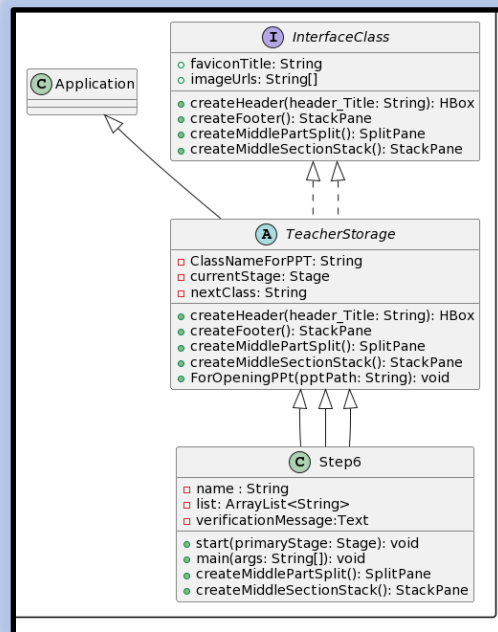


Figure 27 UML of Step6 Class.

### 3.4.2. UML Of “Step7” Class

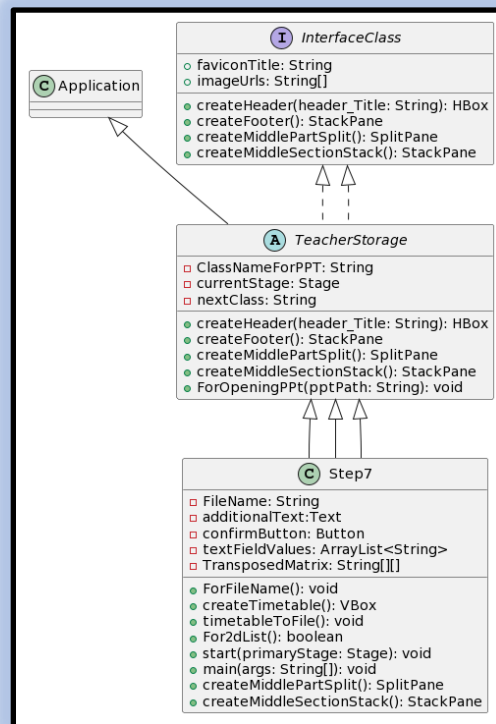


Figure 28 UML Of “Step7” Class.



### 3.4.3. UML Of Step1Contact Class

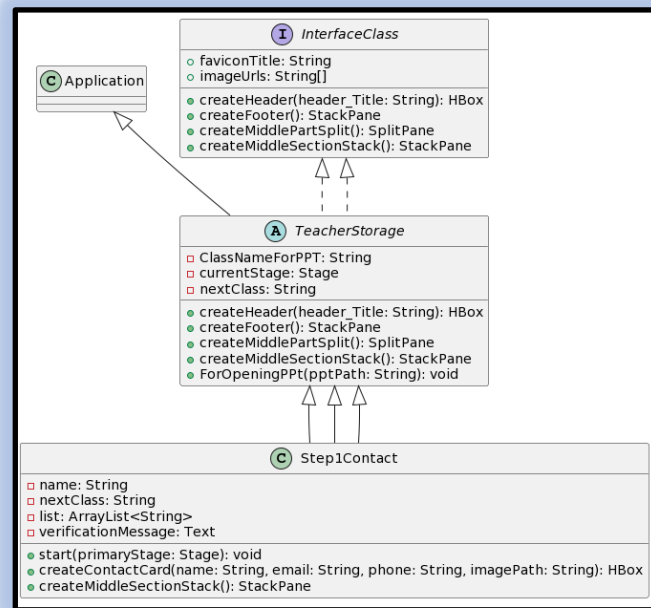


Figure 29 UML Of Step1Contact Class.

## 3.5. OUTPUT OF THE MODULE

### 3.5.1. SignIn Page

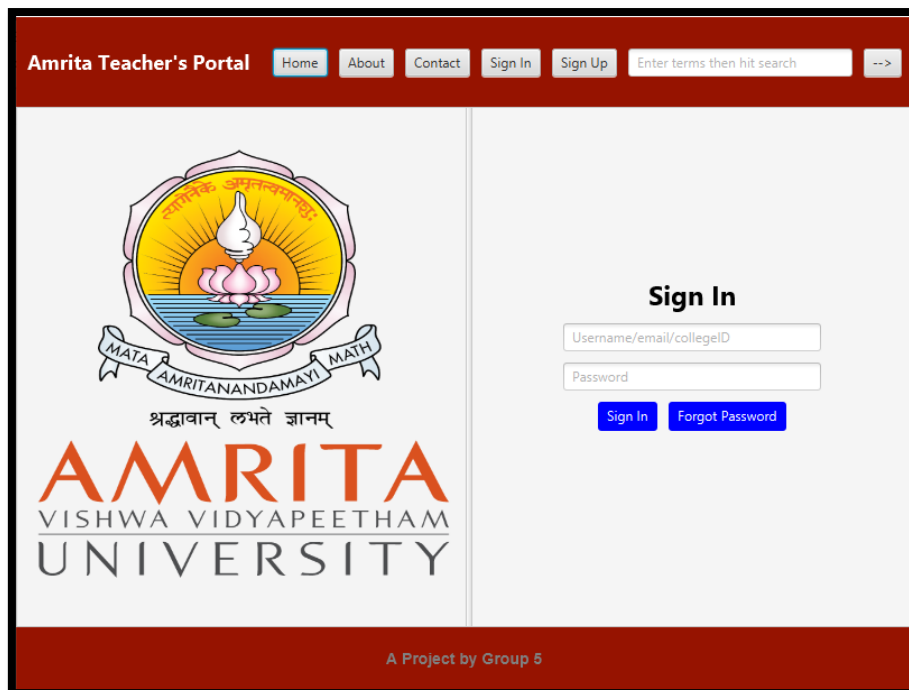


Figure 30 Output Image Of SignIn.

## 3.5.2. Forgot Password

Amrita Teacher's Portal

Home About Contact Sign In Sign Up Enter terms then hit search -->

**Forgot Password**

email

Next

A Project by Group 5

*Figure 31 Output Image Of ForgotPasssword.*

## 3.5.3. Timetable

Amrita Teacher's Portal

Home About Contact Sign In Sign Up Enter terms then hit search -->

**Time Table**

Monday Tuesday Wednesday Thursday Friday Saturday

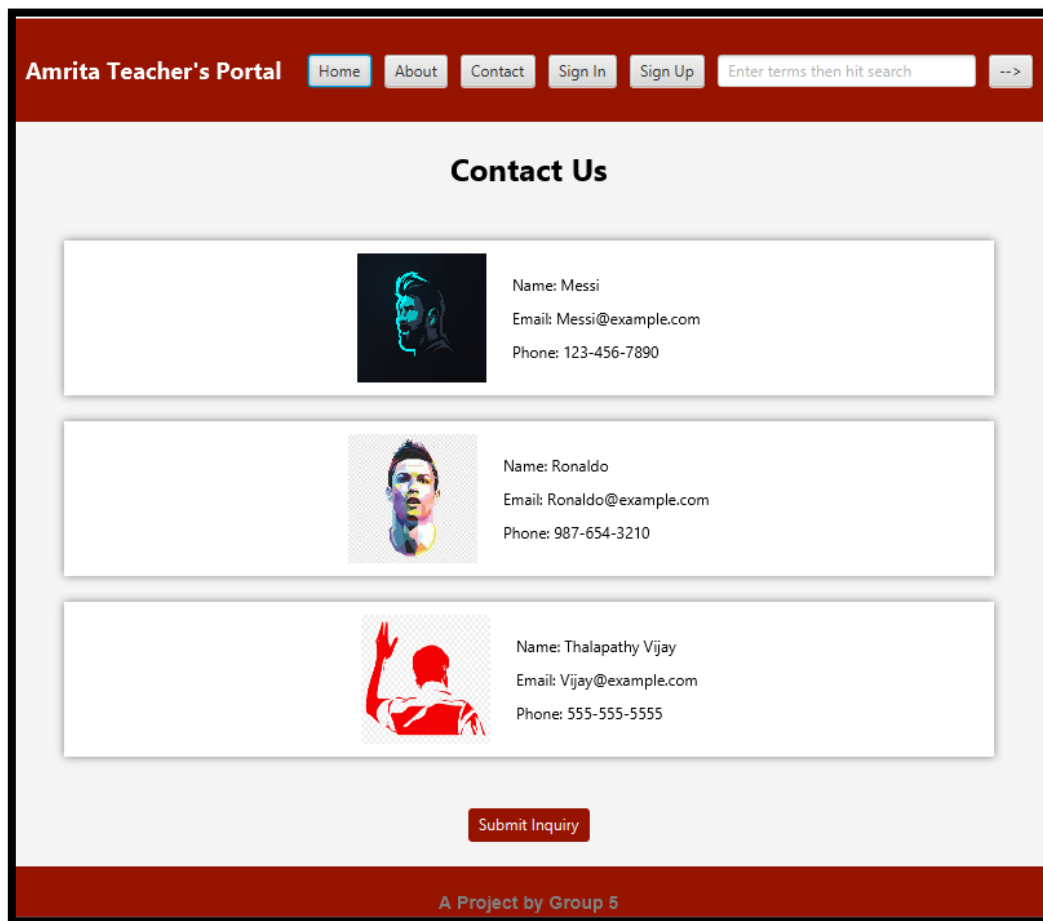
Class 1	Class 2	Class 3	Class 1	Class 2	Class 3	Class 1	Class 2	Class 3	Class 1	Class 2	Class 3	Class 1	Class 2	Class 3	Class 1	Class 2	Class 3

Confirm

A Project by Group 5

*Figure 32 Output Image Of TimeTable.*

### 3.5.4. Contact.



*Figure 33 Output Image Of Step1Contact.*

# MODULE-4 “Updating The Teacher’s Timetable”

## 4.1. DESCRIPTION OF THE MODULE

The “LeaveReq” module and “TeacherNotification” module of a part of main module SignInfoPage the application, these modules allows the user to request leave to the admin and update the timetable accordingly. Here’s the description of the following module.

**SignInfoPage:** The primary objective of this code is to consolidate all the methods and functions relevant to teachers into a single module, called "SignInfoPage," which extends the functionality of another module called "TeacherStorage."

JavaFX is a framework for creating desktop applications with a graphical user interface (GUI). In this case, the code aims to create an interactive interface for teachers, allowing them to perform various tasks related to their work. The application uses JavaFX layout containers, event handling, and styling to achieve this.

By extending the "TeacherStorage" module, the "SignInfoPage" module inherits or enhances the functionalities provided by the "TeacherStorage" module. It includes methods and functions specific to the sign-in information page, such as updating timetables, applying for leave, displaying notifications, and sending emails.

The graphical user interface (GUI) is constructed using JavaFX layout containers. Layout containers are components that arrange other components (buttons, labels, text fields, etc.) in a structured manner. Examples of JavaFX layout containers include VBox (vertical box), HBox (horizontal box), GridPane (grid-based layout), and BorderPane (border-based layout).

The event handling aspect of the code deals with defining actions or behaviors that occur when the user interacts with the GUI elements. For instance, clicking a button might trigger a method that updates the timetables or displays notifications. Event handling allows the application to respond to user input and perform the necessary actions accordingly.

Overall, the provided code aims to create a user-friendly and efficient sign-in information page for teachers using JavaFX. It combines various functionalities into a single module, enabling teachers to access and perform their tasks conveniently through the graphical user interface.

**LeaveReq:** The "LeaveReq" module is designed to facilitate a user-friendly graphical user interface (GUI) that enables users to easily request leave from the administrator. When initiated, the module opens a window that presents several input fields and controls, carefully designed to collect all the necessary information for requesting an absence on a specific day.

Through this intuitive interface, users can conveniently enter details such as the desired date of leave, the reason for the absence, and any additional comments or relevant information. The GUI provides a seamless experience, allowing users to navigate and interact with the various fields effortlessly.

By utilizing the "LeaveReq" module, users can efficiently communicate their leave requirements to the administrator, streamlining the leave request process and enhancing overall productivity.

Within the input grid layout, the following information is collected:

**1.)Select Day:** The user can choose a day of the week from a list view.

**2.)Select Period:** The user can choose a time slot from a list view.

**3.)Mandatory Grounds of Leave:** The user can enter the reason or grounds for leave in a text field.

**4.)Select Date:** The user can choose a date from a date picker.

The user needs to provide values for these input fields and click the "Request" button to submit the leave request. The code validates whether all the input fields are filled, and if so, it saves the leave request data separated by comma to a file named "LeaveFile.txt" and displays a success message.

This request is sent to the admin end, where the admin can either accept or reject. The details of the "LeaveFile.txt" is stored in new file for further actions as per the action performed by the admin.

The module employs appropriate event handlers and input validation to ensure the entered data is processed correctly and necessary actions are performed accordingly.

**TeacherNotification:** The "TeacherNotification" module is an important component of a system designed to handle notifications related to leave requests for teachers. Its purpose is to display notifications regarding the acceptance or rejection of leave requests and to send notifications to other available teachers regarding the availability of free periods. Additionally, it reserves those free periods for the teachers.

To provide a user-friendly interface, the module defines event handlers for button clicks, allowing users to interact with the system. It relies on two data files, namely "ForTeachersLeave.txt" and "user\_data.txt," to gather the necessary information for generating notifications and identifying free periods. By processing and displaying this information, the module ensures that teachers are kept informed of important updates.

Consider the scenario where a teacher applies for leave, and their leave request is approved. In this case, the module generates a message specifically notifying the user about the approval of their leave. Simultaneously, other teachers who have free periods during the same time receive a message urging them to reserve the classroom for their own use.

When a teacher receives a leave approval or rejection notification, a "Confirm" button appears as part of the message. This button serves as a means for the user to provide a response. Upon clicking the "Confirm" button, the message is updated accordingly based on the user's response. For example, if the teacher confirms their acceptance of the leave approval, the message may display a confirmation message or provide additional information related to the approved leave. On the other hand, if the teacher confirms their acceptance of the leave rejection, the message might include alternative options or instructions.

By incorporating these functionalities, the "TeacherNotification" module streamlines the process of notifying teachers about leave requests, managing free periods, and facilitating their responses. It enhances communication within the system and promotes efficient utilization of resources such as classrooms.

Overall, the SignInInfo module provides a user-friendly interface for users accessibly information, saves the details, and checks for potential matches among found items. It facilitates the process of allocating proper image.

## 4.2 MOTIVATION OF THE MODULE.

The module plays a crucial role in offering convenient and interactive functions for users. It ensures that data is stored and utilized in a responsible manner by the administrator. Based on the available data, the module facilitates the provision of services to the users.

***Ease of Use:*** The module utilizes a user-friendly graphical interface with clearly labeled input fields and controls. This design approach simplifies the reporting process, making it accessible even to users with limited technical skills. The intuitive layout and familiar form-like structure help users navigate and fill out the required information effortlessly.

***Data Collection:*** The module collects various details about the date, time, slot. This comprehensive data collection ensures that users can provide as much information as possible, facilitating the matching process and assisting the user with accurate details.

***User Interaction:*** By offering a dedicated module for providing information, the application empowers users to take an active role in the updating process. It allows them to provide detailed information about the date of leave, and also allows to allocate a slot for the free user according to their own wish.

***Integrated Notification System:*** This module send the user notifications according to the admins answer. This integration ensures that users are promptly notified of any potential matches, providing them with valuable information and facilitating efficient communication between users and the application.

## 4.3 RELEVANCE OF THE MODULE IN THE SYSTEM:

The module plays a crucial role in the system by facilitating efficient communication and interaction between the users and the system. It serves as a bridge, connecting the users' actions with the system's responses.

Firstly, the module's ability to process and display information from the "ForTeachersLeave.txt" and "user\_data.txt" data files is vital for the system's

operation. It ensures that teachers are kept up-to-date with important updates, such as leave approvals or rejections. This feature enhances the system's transparency and allows for a smoother workflow.

Secondly, the module's event handlers for button clicks enable users to interact with the system effectively. For instance, the "Confirm" button allows users to respond to notifications, thereby promoting active user engagement.

Thirdly, the module's user-friendly graphical interface enhances the system's usability. It simplifies the reporting process, making it accessible to users with varying technical skills. This feature is particularly relevant in a diverse user environment, ensuring that all users can navigate and use the system with ease.

Lastly, the integrated notification system is a key feature of the module that enhances the system's efficiency. By sending user notifications based on the admin's responses, the module ensures that users are promptly informed of any updates. This feature not only provides users with valuable information but also facilitates efficient communication within the system.

In conclusion, the module's relevance in the system lies in its ability to facilitate user-system interaction, enhance system transparency, promote user engagement, improve system usability, and ensure efficient communication.



## 4.4. UML OF THE MODULE

### 4.4.1. UML of “SignInInfoPage” Class.

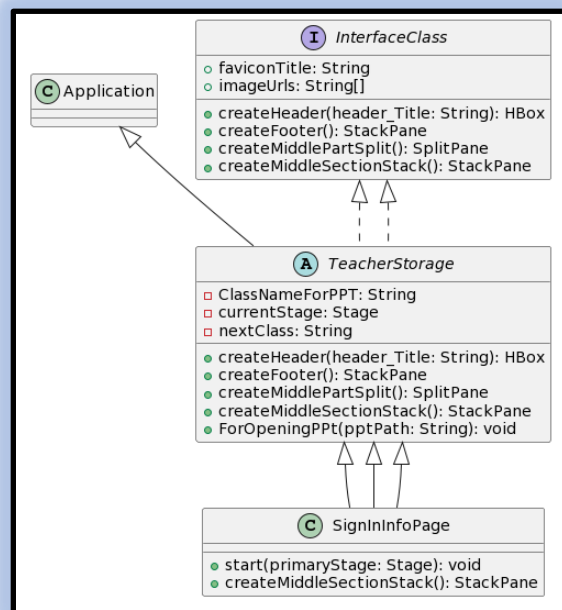


Figure 34 UML diagram for SignInInfoPage

### 4.4.2. UML of “LeaveReq” Class

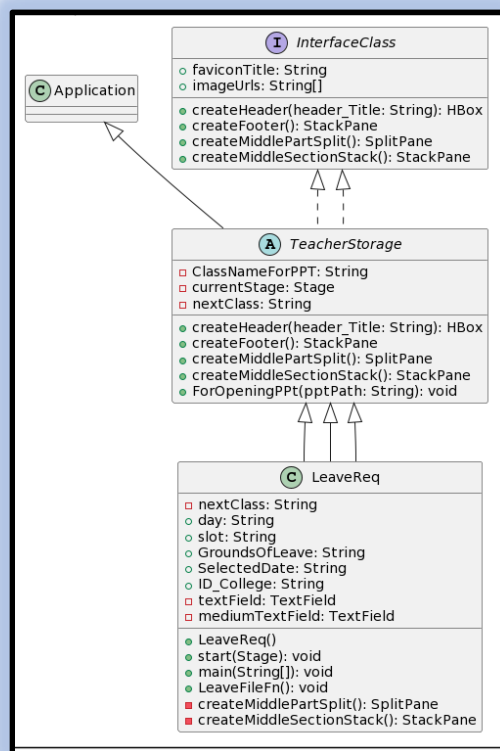


Figure 35 UML diagram for LeaveReq

### 4.4.3. UML of “TeacherNotification” Class.

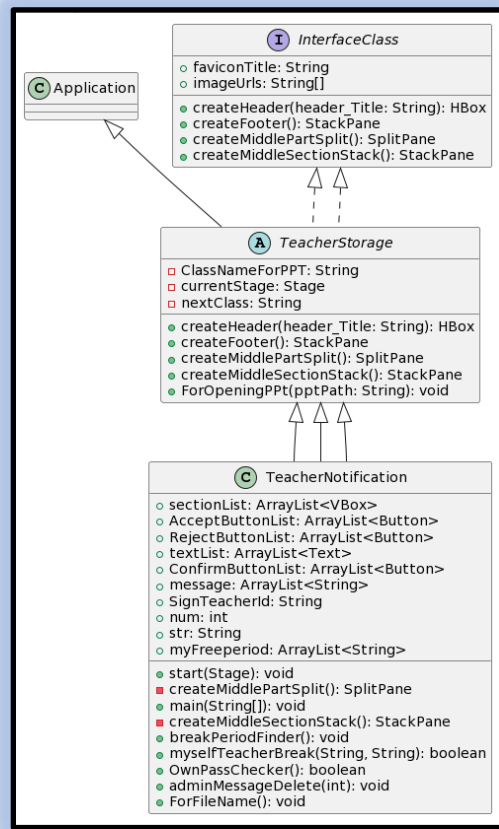


Figure 36 UML diagram for TeacherNotification

### 4.4.4. UML of “Step1About” Class.

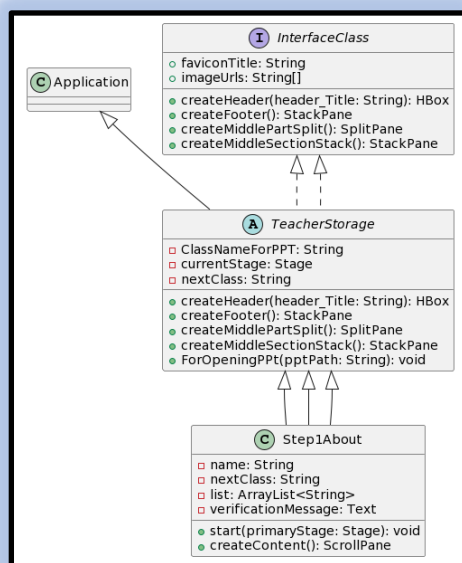


Figure 37 UML diagram for Step1About

## 4.5. OUTPUT OF THE MODULE

### 4.5.1. SignInInfoPage

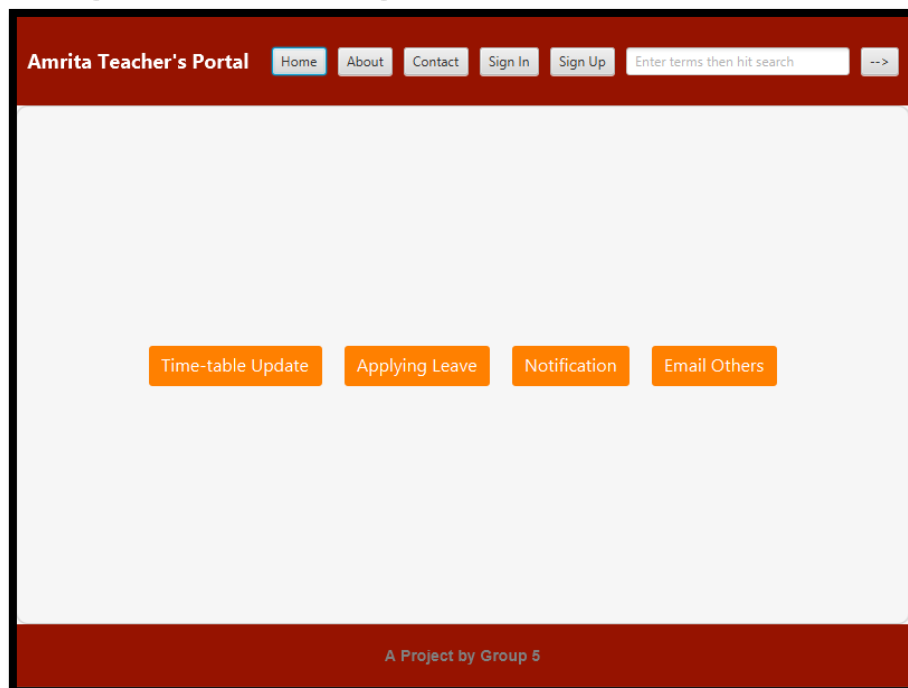


Figure 38 Output for SignInInfoPage

### 4.5.2. LeaveReq

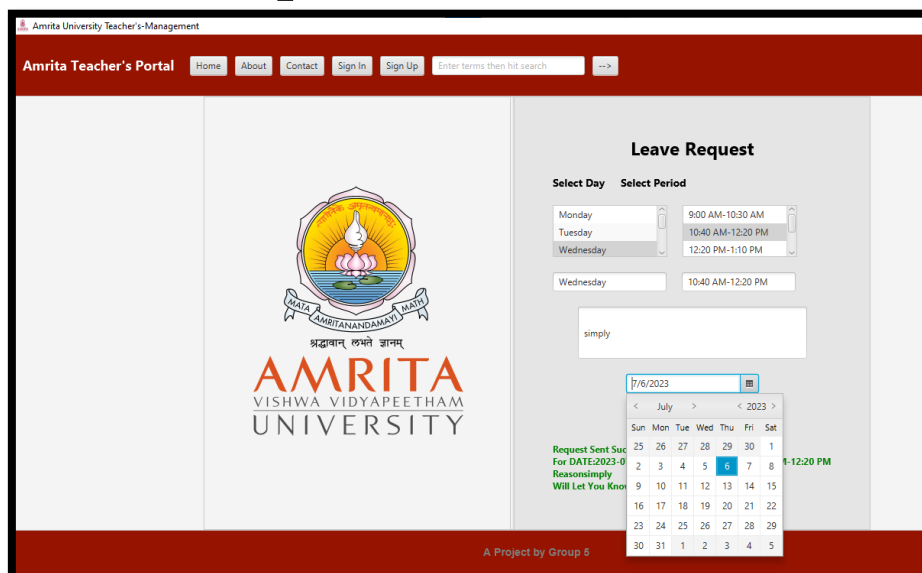


Figure 39 Output for LeaveReq

## 4.5.3. TeacherNotification

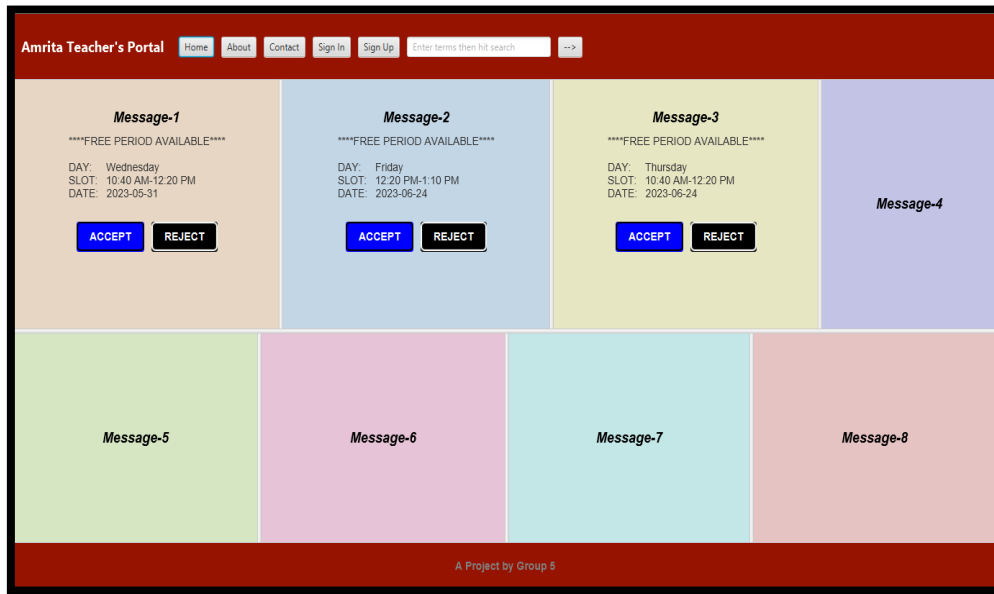


Figure 40 Output for TeacherNotification

## 4.5.4. Step1About

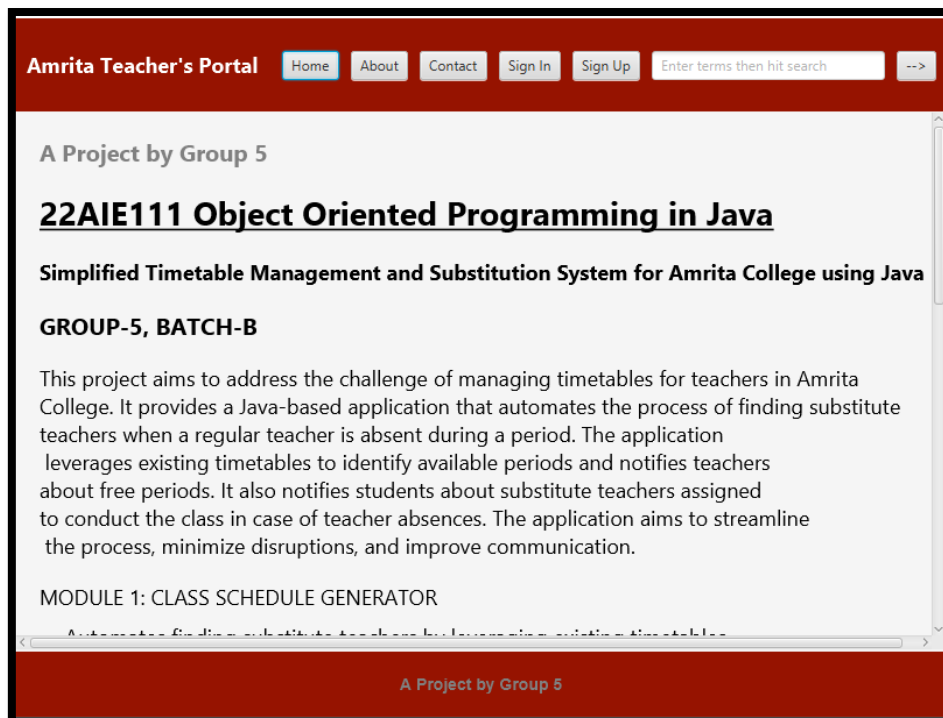
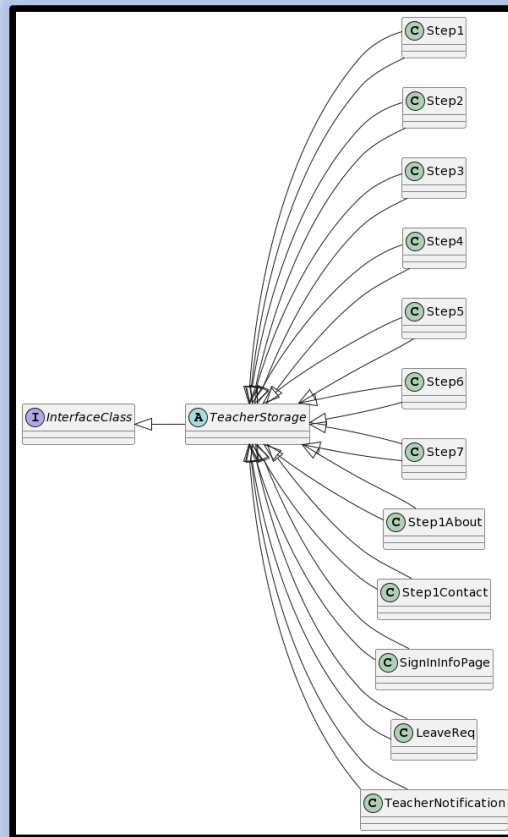


Figure 41 Ouput for Step1About

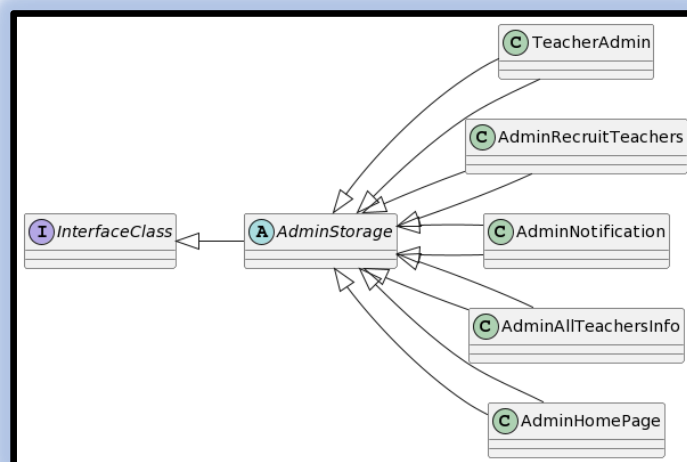
# Module 5 “Interlinking Of All Classes”

## 5.1. Uml Of TeacherSide Classes



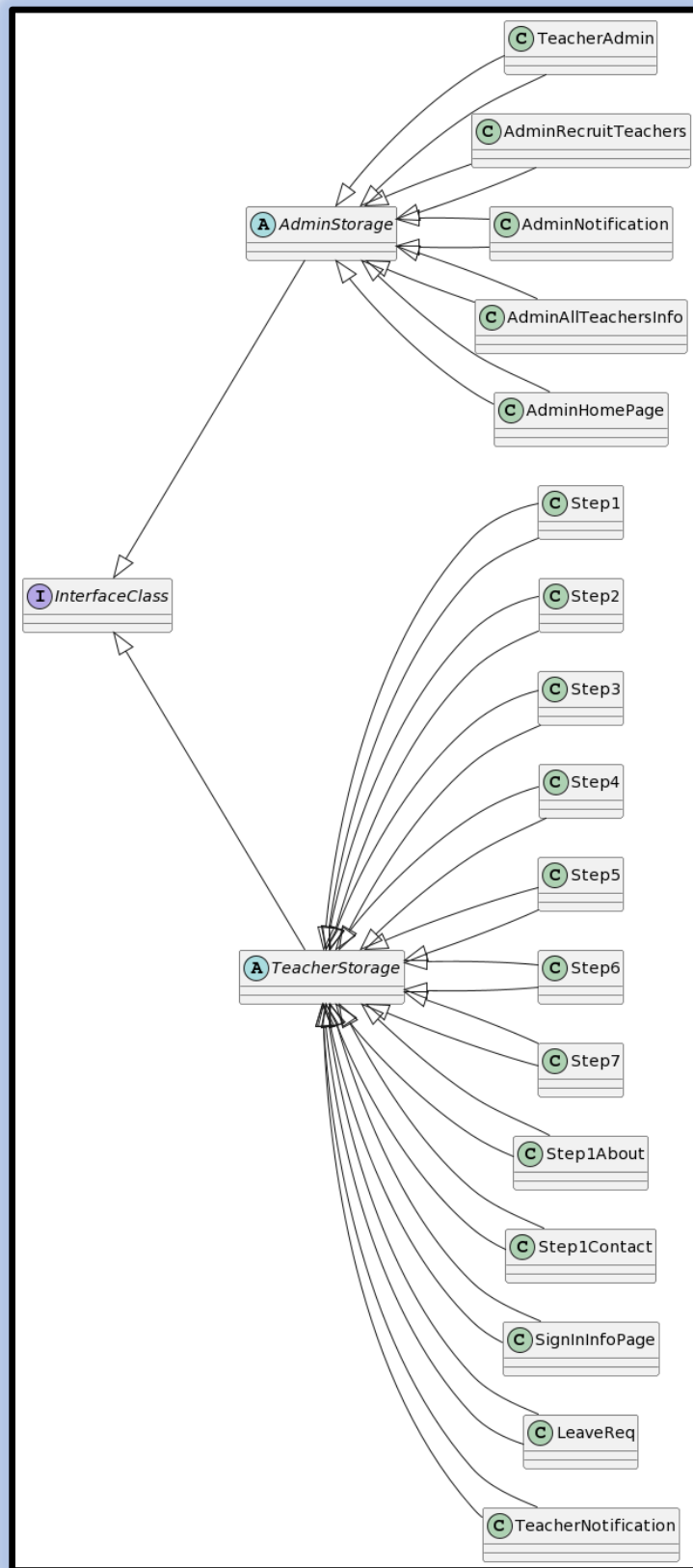
*Figure 42 UML diagram for TeacherSide Classes*

## 5.2. UML for AdminSide Classes



*Figure 43 UML diagram for AdminSide classes*

### 5.3. UML for all Classes



*Figure 44 UML diagram for all classes*