Version: 1.0

[Synkers]
Architecture/Design Document

Table of Contents

Change History

Version: <1.0>

Modifier: <Houssam Mehdi>

Date: 10/20/2017

Description of Change: First skeleton was made, including the main functionalities, main classes and operations.

This would include: Main frame, Log in frame, Messages frame, Sessions frame, Register frame, Explore frame and Profile frame.

Version: <1.0>

Modifier: <Houssam Mehdi>

Date: 10/25/2017

Description of Change: All the frames design were modified to an absolute border layout for a better design. JDBC class implementation was made including the main preparations for the databases on Postgres using PgAdmin.

Version: <1.0>

Modifier: <Houssam Mehdi>

Date: 11/05/2017

Description of Change: Databases is completely ready, including 4 main tables: users, inbox, sessions and images.

Version: <1.0>

Modifier: <Houssam Mehdi>

Date: 11/10/2017

Description of Change: All database operation were implemented and checked for functionality, added some extra features to the project.

Version: <1.0>

Modifier: <Houssam Mehdi>

Date: 11/20/2017

Description of Change: All the action listeners were checked, implemented and tested. Started creating users for test, including teachers and Students. A lot of bugs were fixed. The main project works perfectly.

1 Introduction

Architecture and Design

The purpose of this application is to be able to connect students with tutors in the real world, in an easy and affordable way.

The main idea comes from real life, were students are struggling to find the right tutors for some material, instead of asking around, or searching for a tutor, we're going to start using Synkers in order to match these students with their corresponding tutors.

The architecture/design of this application was inspired from the same application which was made in "Lebanon" for android and IOS, this java implemented application is the same concept on Windows.

Software architecture and designs are typically expressed with a mix of UML models (class and sequence diagrams being the two most common) and prose. Dataflow diagrams are also helpful for understanding the interaction between components and overall flow of data through the system.

About this document

This document makes sure that the following checklist passes for the project:

- Are design objectives clearly stated? For example, if performance is more important than reusability.
- Does the architecture partition the implementation into clearly defined subsystems or modules with well-defined interfaces?
- Does the architecture express in a clear way the main patterns of communication between subsystems and modules?
- Does the architecture satisfy the requirements?
- Is the architecture traceable to requirements?
- Any models created should either be expressed with a well-known modeling language, or if a well-known modeling language isn't used, the syntax and semantics of the symbols that are used should be defined.

The purpose of this document is to describe the architecture and design of the **Synkers** application in a way that addresses the interests and concerns of all major stakeholders. For this application the major stakeholders are:

- Users and the customer – they want assurances that the architecture will provide for system functionality and exhibit desirable non-functional quality requirements such as usability, reliability, etc.
- Developers – they want an architecture that will minimize complexity and development effort.
- Project Manager – the project manager is responsible for assigning tasks and coordinating development work. He or she wants an architecture that divides the system into components of roughly equal size and complexity that can be developed simultaneously with minimal dependencies. For this to happen, the modules need well-defined interfaces. Also, because most individuals specialize in a particular skill or technology, modules should be designed around specific expertise. For example, all UI logic might be encapsulated in one module. Another might have all business logic.
- Maintenance Programmers – they want assurance that the system will be easy to evolve and maintain on into the future.

1. Logical View – major components, their attributes and operations. This view also includes relationships between components and their interactions. When doing OO design, class diagrams and sequence diagrams are often used to express the logical view.
2. Process View – the threads of control and processes used to execute the operations identified in the logical view.
3. Development View – how system modules map to development organization.
4. Use Case View – the use case view is used to both motivate and validate design activity. At the start of design the requirements define the functional objectives for the design. Use cases are also used to validate suggested designs. It should be possible to walk through a use case scenario and follow the interaction between high-level components. The components should have all the necessary behavior to conceptually execute a use case.

2 Design Goals

The design goals for this project were set:

- One graphical interface for both Students and tutors, working in a dynamic way according to the type of the client, which is determined when the user logs in.
- A Log in frame, which is static for both students and tutors.
- A Register frame which is static for both, including one dynamic field where the type of the client is precised.
- The main menu frame for both students and tutors, which will include the following options:
- Messages frame.
- Sessions frame.
- Profile frame.
- Explore frame.

Below we can see the following screens from the project:

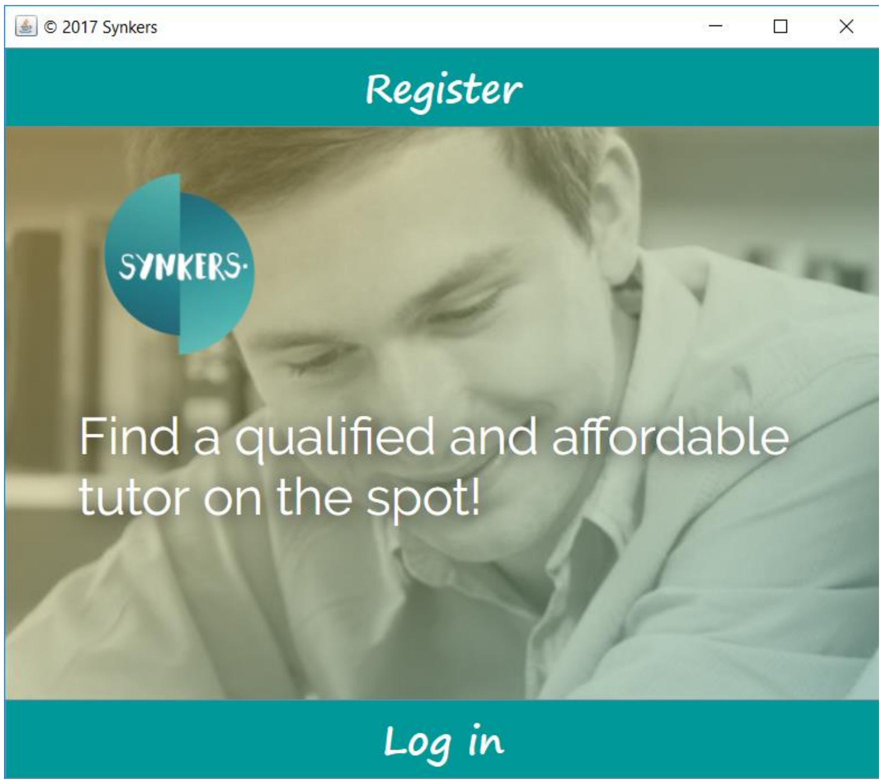


Figure 1: Main frame, two options : Log in or Register.

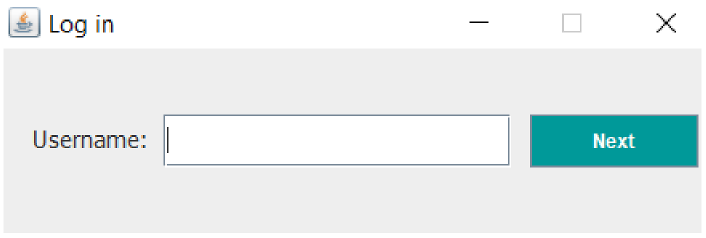


Figure 2: Log in frame, username authentication

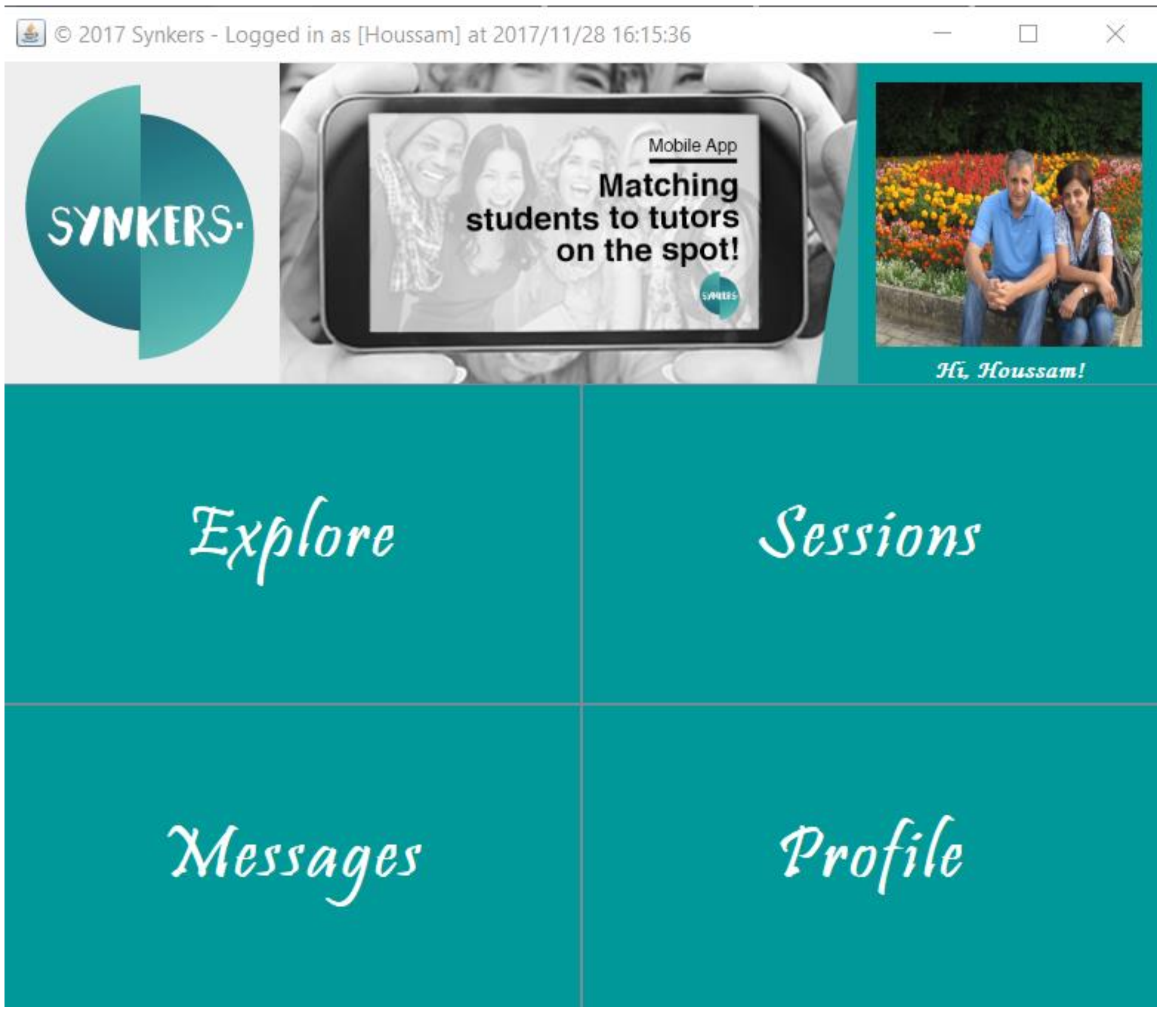
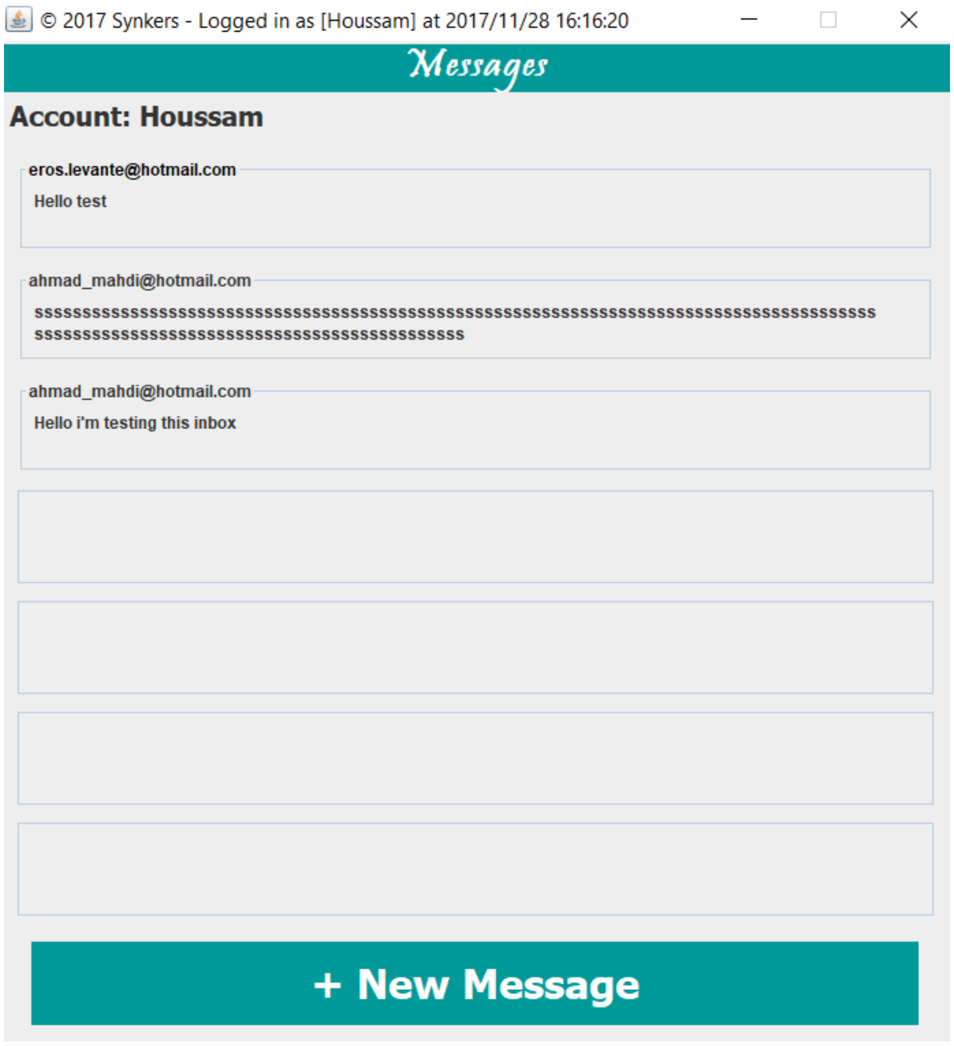

© 2017 Synkers - Logged in as [Houssam] at 2017/11/28 16:15:36
SYNKERS
Mobile App
Matching students to tutors on the spot!
Hi, Houssam!
Explore
Sessions
Messages
Profile

Figure 3: Successful log in, main menu options.


© 2017 Synkers - Logged in as [Houssam] at 2017/11/28 16:16:20

Account: Houssam

eros.levante@hotmail.com
Hello test

ahmad_mahdi@hotmail.com
ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
sssssssssssssssssssssssssssssssssssssssssssss

ahmad_mahdi@hotmail.com
Hello i'm testing this inbox

Figure 4: Messages option, checking your inbox or sending a new message.

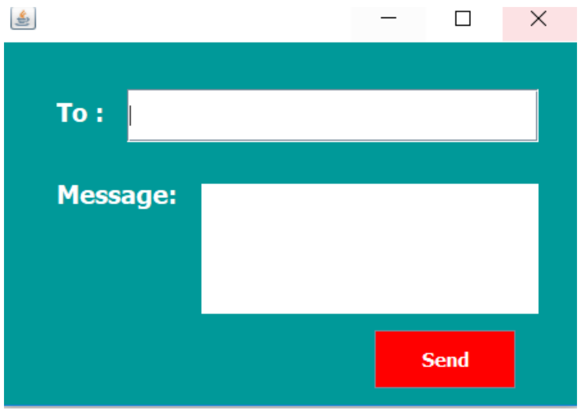
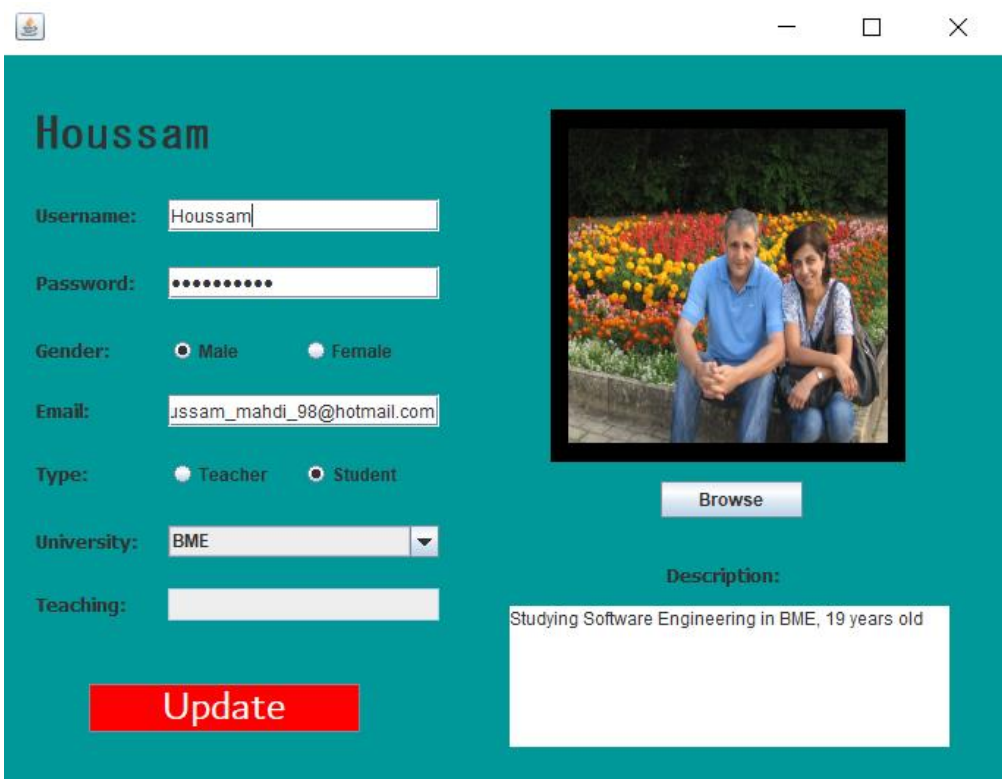


Figure 5: Sending a new message, entering the receiver username and your message.



Houssam

Username: Houssam

Password: ••••••••••

Gender: [✓ Male] Female

Email: ıssam_mahdi_98@hotmail.com

Type: Teacher [✓ Student]

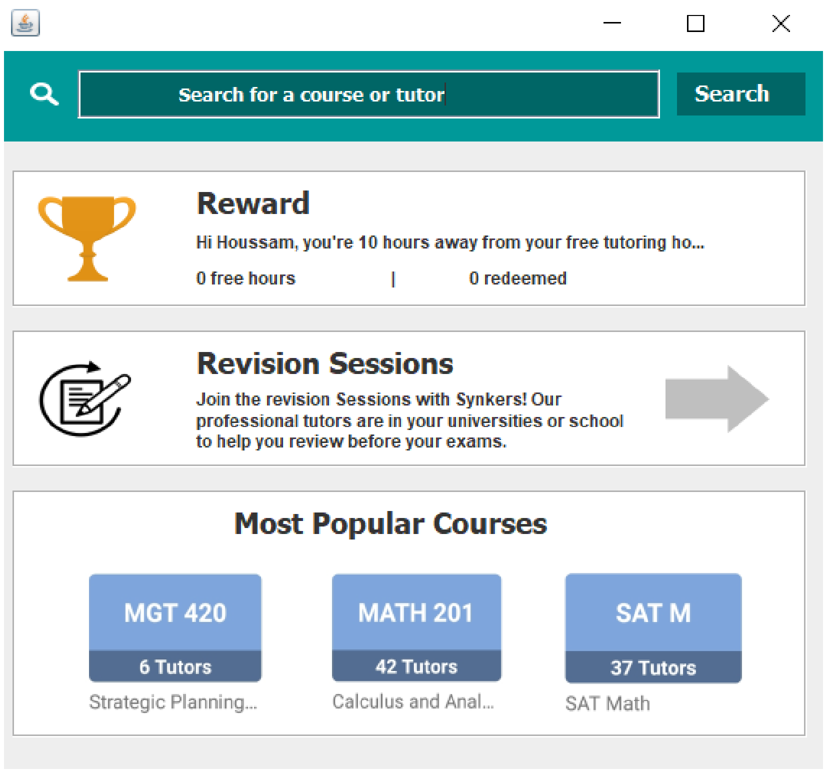
University: BME

Teaching:

Description:

Studying Software Engineering in BME, 19 years old

Figure 6: Profile frame, updating any missing information, or uploading a picture.



Reward

Hi Houssam, you're 10 hours away from your free tutoring ho...

0 free hours | 0 redeemed

Revision Sessions

Join the revision Sessions with Synkers! Our professional tutors are in your universities or school to help you review before your exams.

Most Popular Courses

MGT 420 — 6 Tutors — Strategic Planning...

MATH 201 — 42 Tutors — Calculus and Anal...

SAT M — 37 Tutors — SAT Math

Figure 7: Explore frame, searching for tutors, courses, or checking rewards.

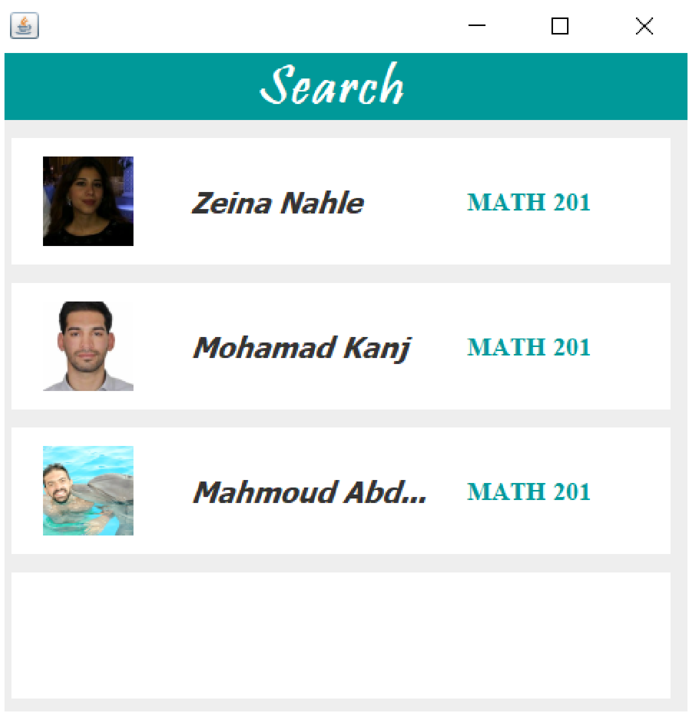


Figure 8: Searching for a tutor by name or course name.

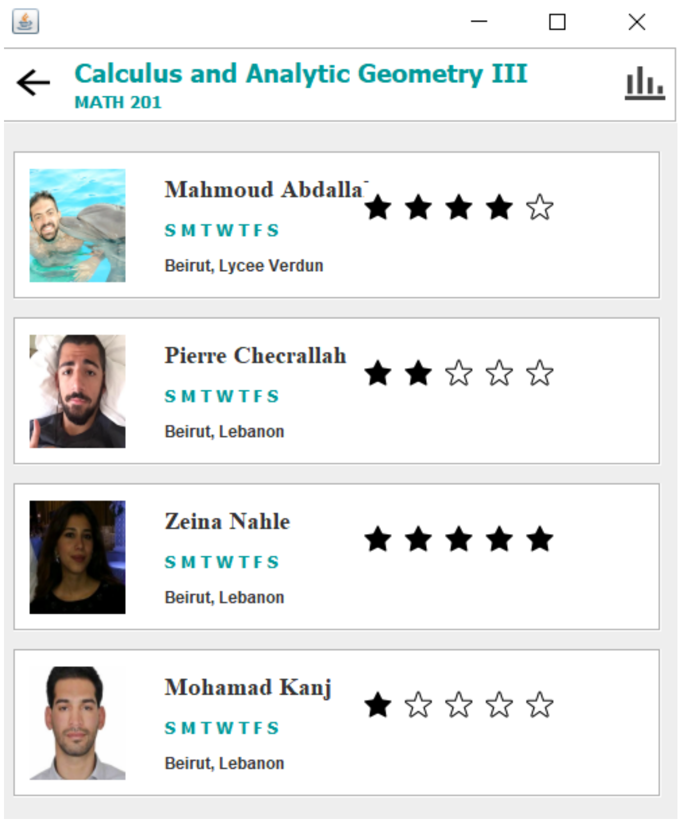
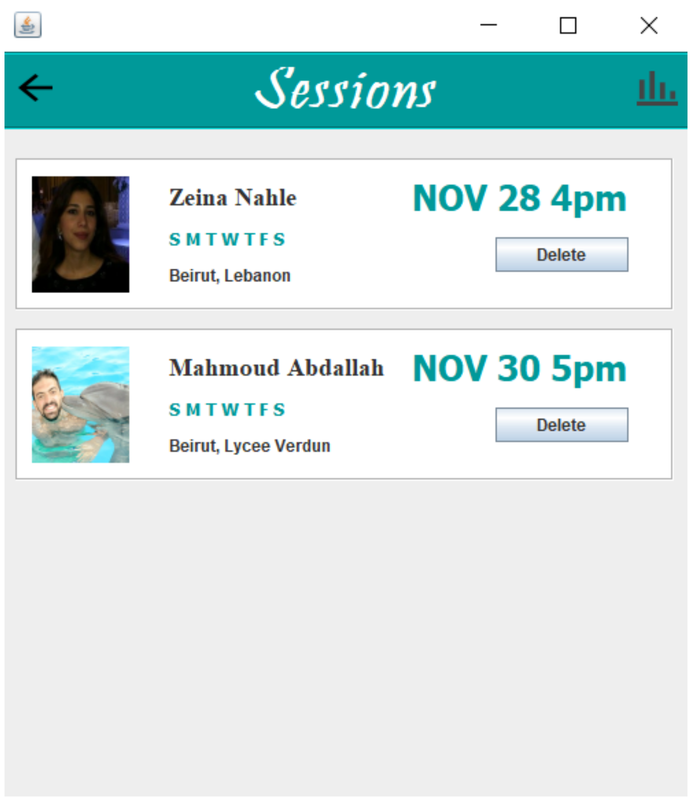


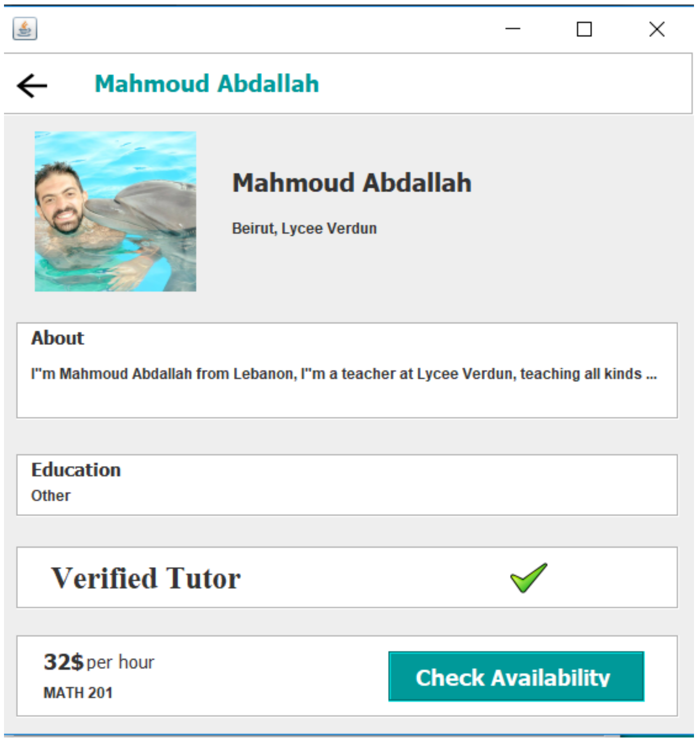
Figure 9: Exploring tutors for a specific course, including ratings, availability and location.



Zeina Nahle
S M T W T F S
Beirut, Lebanon
NOV 28 4pm

Mahmoud Abdallah
S M T W T F S
Beirut, Lycee Verdun
NOV 30 5pm

Figure 10: Checking/deleting your existing sessions.



Mahmoud Abdallah

Mahmoud Abdallah
Beirut, Lycee Verdun

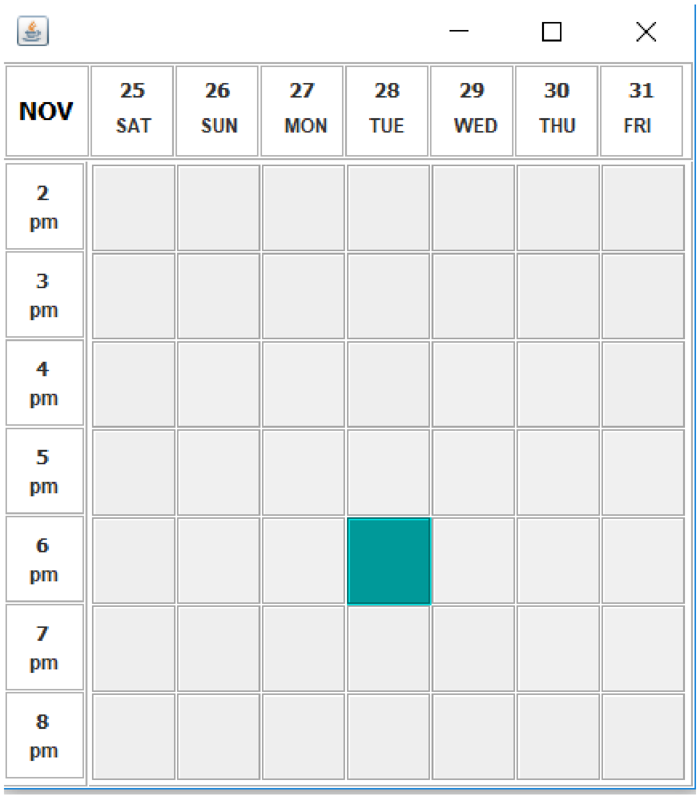
About
I"m Mahmoud Abdallah from Lebanon, I"m a teacher at Lycee Verdun, teaching all kinds ...

Education
Other

Verified Tutor ✓

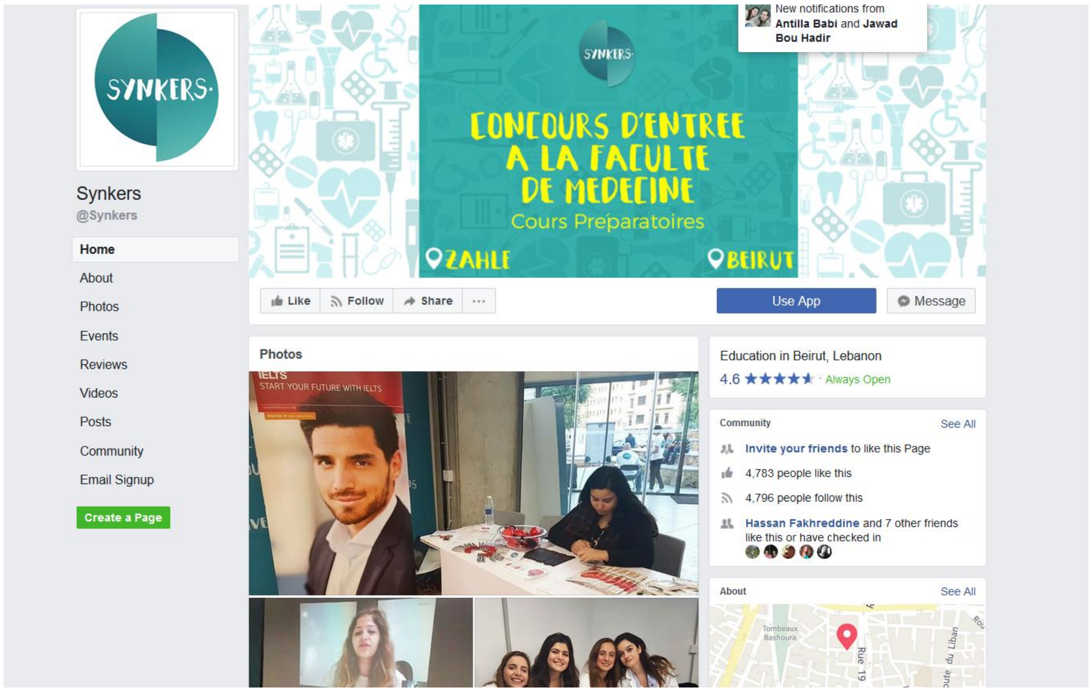
32$ per hour
MATH 201

Figure 11: Checking a tutors profile including description, education, price and availability.



NOV	25 SAT	26 SUN	27 MON	28 TUE	29 WED	30 THU	31 FRI
2 pm							
3 pm							
4 pm							
5 pm							
6 pm				🟩			
7 pm							
8 pm							

Figure 12: Booking a session with a tutor, choosing between the available time slots.



New notifications from Antilla Babi and Jawad Bou Hadir

Education in Beirut, Lebanon
4.6 ★★★★★ · Always Open

Community
Invite your friends to like this Page
4,783 people like this
4,796 people follow this
Hassan Fakhreddine and 7 other friends like this or have checked in

Figure 13: Automatic redirection to the Synkers page using your default browser.

3 System Behavior

The behavior view is used to both drive the design phase and validate the output of the design phase. The architecture description presented here starts with a review of the expect system behavior in order to set the stage for the architecture description that follows.

In our project, the system would have two main behaviors according to the client type. If our client is a Student, then he will be logged in as a “Student” where our goal is to find a tutor according to the needs of that student. If our client is a Teacher then he will be logged in as a “Teacher” where our goal is to find students for that teacher according to the courses he teaches.

These will be done using our application, where the teachers have to register and enter their corresponding information, and then these information will be visible for the registered Students in order to choose a Tutor. When the choosing phase is done, the student is redirected to the tutor’s profile where the tutor’s calendar is visible.

The student will then choose out of the times available on the tutor’s calendar, and that when the session will be registered for both.

In addition to that, a student can communicate with the tutor using the inbox section in order to get additional information about the course or the session.

Each student will receive a “Synkers Reward” which is a free session offered by the Synkers team, and this will happen whenever a student books 10 sessions with our application.

The students can pay with credit card using our application, but this is applied only for the tutors which allow this feature, if a teacher allows it then it will be automatically transferred into his account.

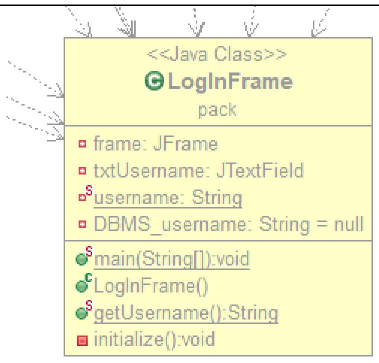
Every tutor has added a location in the address field when registering, but this address can be changed on the Profile section, using the Update button. Also if a location is not suitable for the student, then this student can communicate using inbox in order to choose a new location suitable for both, the student and the tutor.

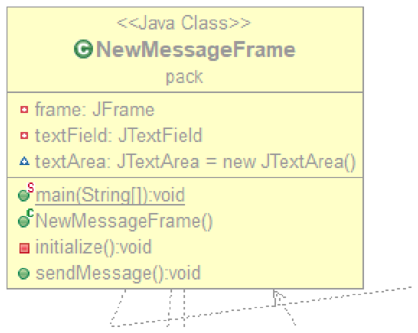
4 Logical View

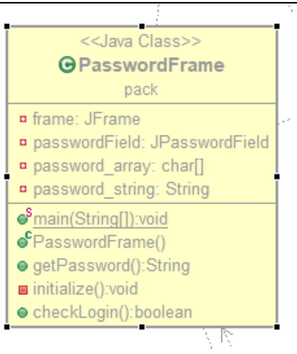
The logical view describes the main functional components of the system. This includes modules, the static relationships between modules, and their dynamic patterns of interaction.

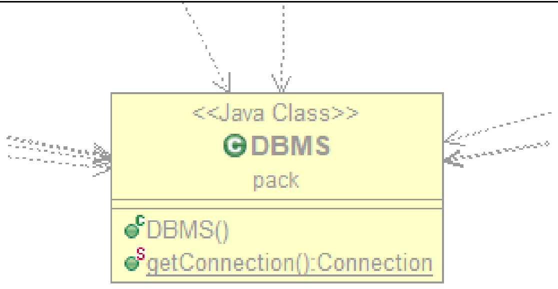
In this section the modules of the system are first expressed in terms of high level components (architecture) and progressively refined into more detailed components and eventually classes with specific attributes and operations.

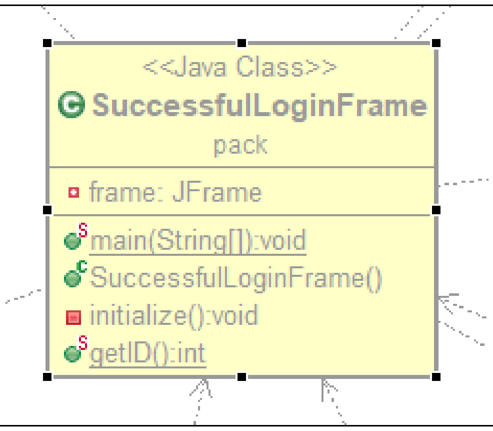
The following are the classes of our project represented in UML:



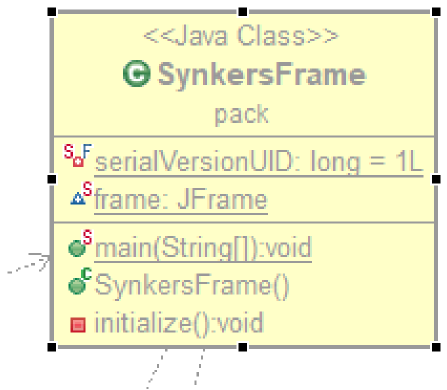




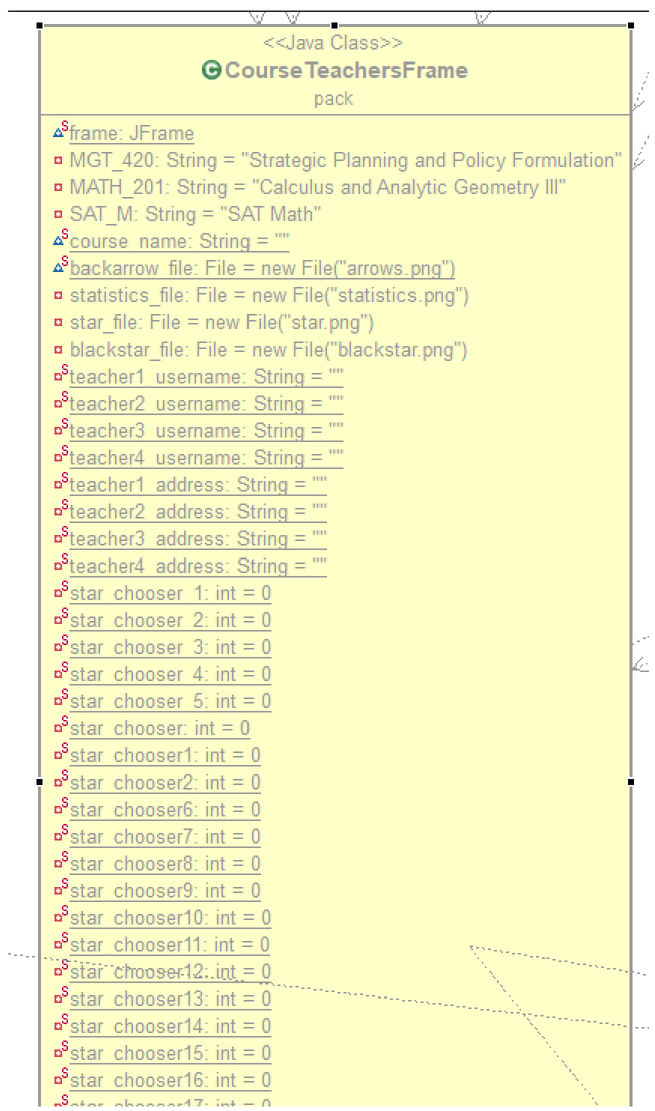




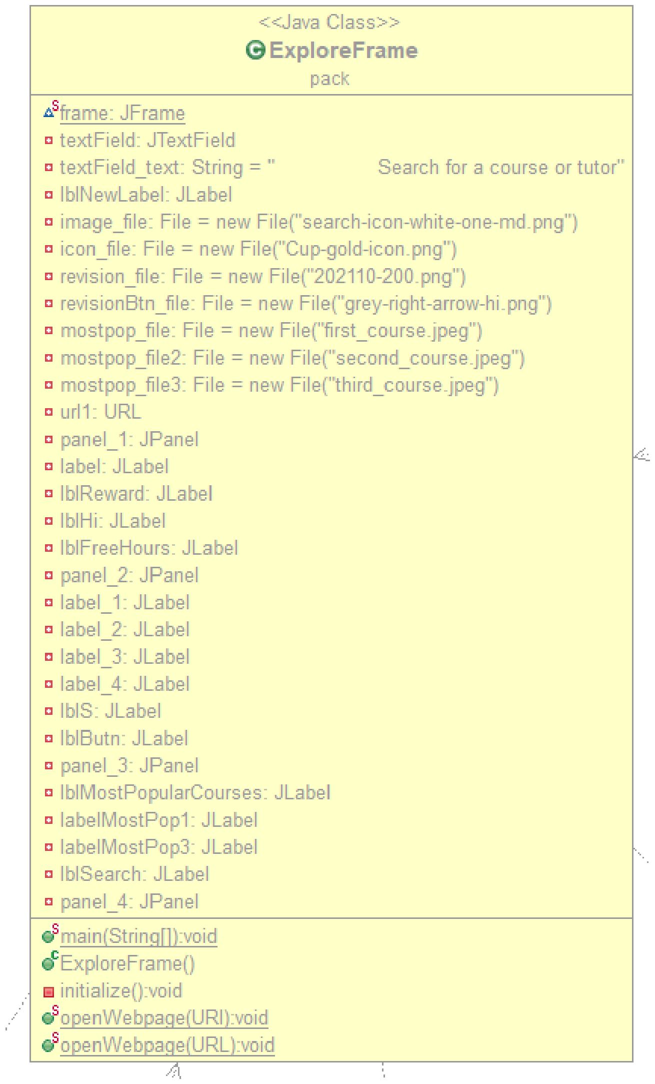
```
<<Java Class>>
SuccessfulLoginFrame
pack
frame: JFrame
main(String[]):void
SuccessfulLoginFrame()
initialize():void
getID():int
```



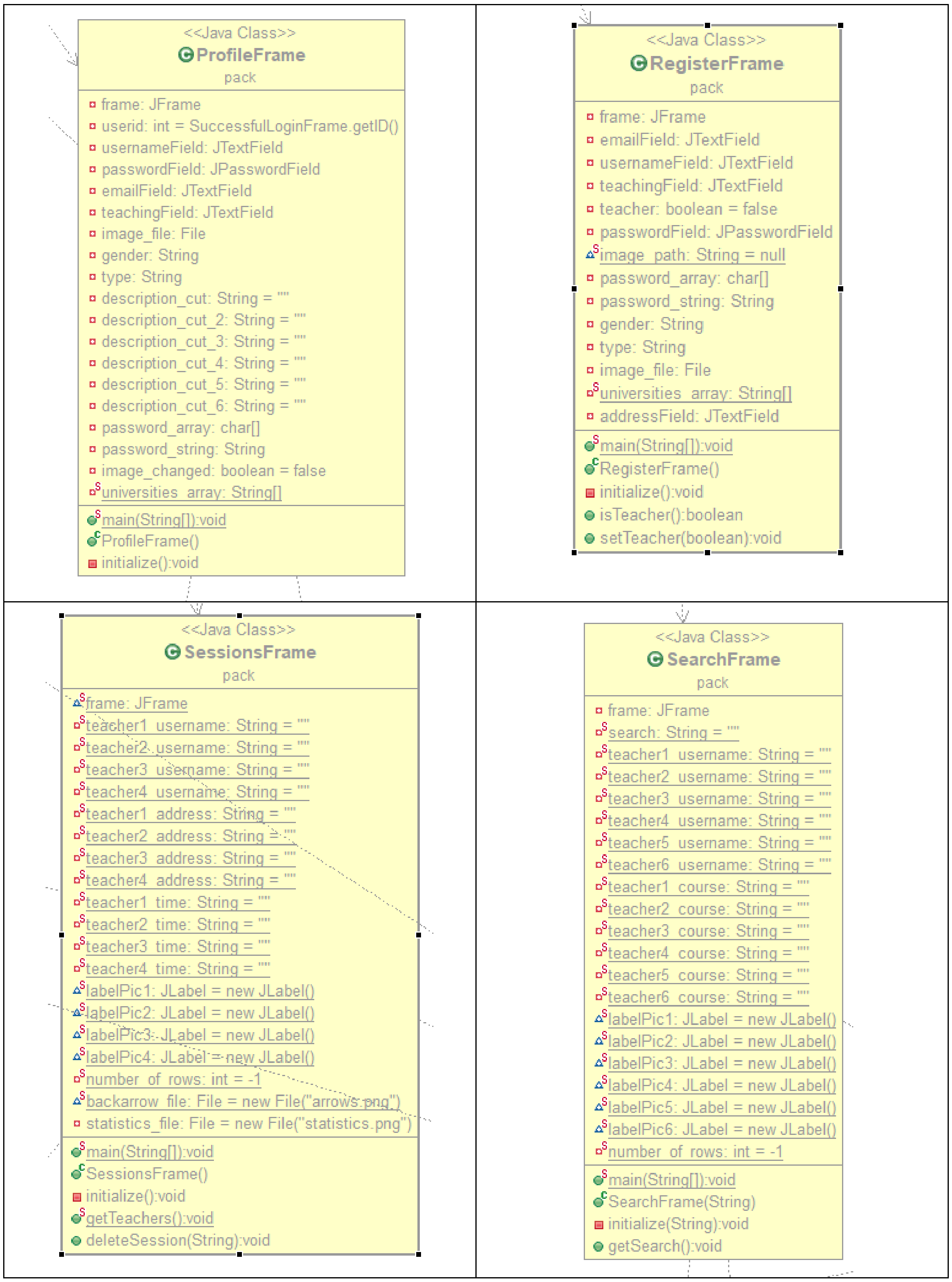
```
<<Java Class>>
SynkersFrame
pack
serialVersionUID: long = 1L
frame: JFrame
main(String[]):void
SynkersFrame()
initialize():void
```

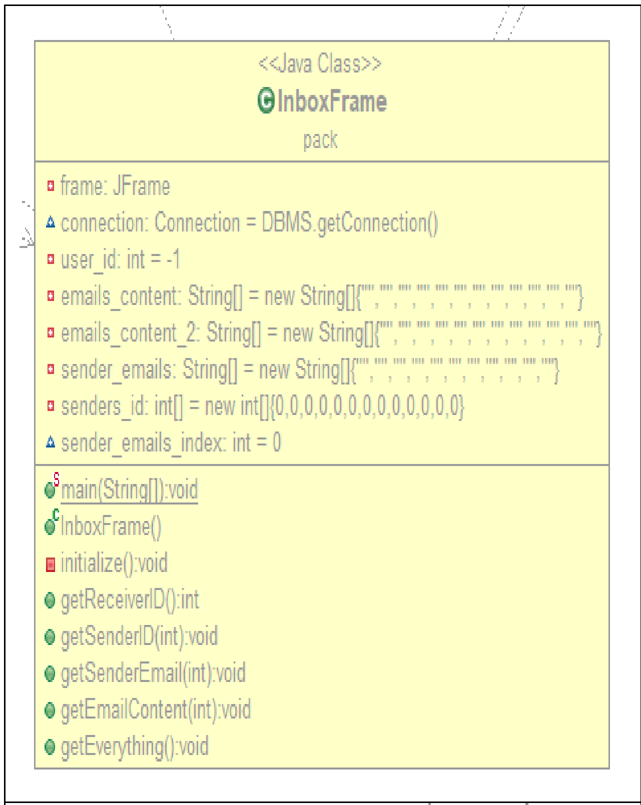


```
<<Java Class>>
CourseTeachersFrame
pack
frame: JFrame
MGT_420: String = "Strategic Planning and Policy Formulation"
MATH_201: String = "Calculus and Analytic Geometry III"
SAT_M: String = "SAT Math"
course_name: String = ""
backarrow_file: File = new File("arrows.png")
statistics_file: File = new File("statistics.png")
star_file: File = new File("star.png")
blackstar_file: File = new File("blackstar.png")
teacher1_username: String = ""
teacher2_username: String = ""
teacher3_username: String = ""
teacher4_username: String = ""
teacher1_address: String = ""
teacher2_address: String = ""
teacher3_address: String = ""
teacher4_address: String = ""
star_chooser_1: int = 0
star_chooser_2: int = 0
star_chooser_3: int = 0
star_chooser_4: int = 0
star_chooser_5: int = 0
star_chooser: int = 0
star_chooser1: int = 0
star_chooser2: int = 0
star_chooser6: int = 0
star_chooser7: int = 0
star_chooser8: int = 0
star_chooser9: int = 0
star_chooser10: int = 0
star_chooser11: int = 0
star_chooser12: int = 0
star_chooser13: int = 0
star_chooser14: int = 0
star_chooser15: int = 0
star_chooser16: int = 0
```

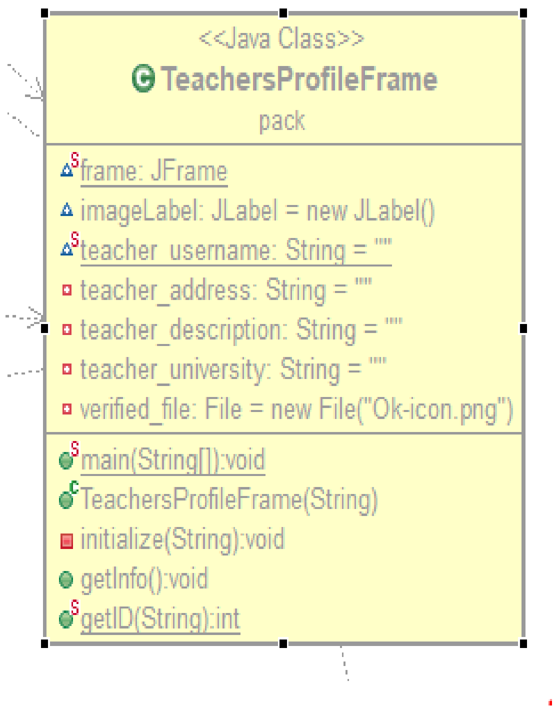


```
<<Java Class>>
ExploreFrame
pack
frame: JFrame
textField: JTextField
textField_text: String = "            Search for a course or tutor"
lblNewLabel: JLabel
image_file: File = new File("search-icon-white-one-md.png")
icon_file: File = new File("Cup-gold-icon.png")
revision_file: File = new File("202110-200.png")
revisionBtn_file: File = new File("grey-right-arrow-hi.png")
mostpop_file: File = new File("first_course.jpeg")
mostpop_file2: File = new File("second_course.jpeg")
mostpop_file3: File = new File("third_course.jpeg")
url1: URL
panel_1: JPanel
label: JLabel
lblReward: JLabel
lblHi: JLabel
lblFreeHours: JLabel
panel_2: JPanel
label_1: JLabel
label_2: JLabel
label_3: JLabel
label_4: JLabel
lblS: JLabel
lblButn: JLabel
panel_3: JPanel
lblMostPopularCourses: JLabel
labelMostPop1: JLabel
labelMostPop3: JLabel
lblSearch: JLabel
panel_4: JPanel
main(String[]):void
ExploreFrame()
initialize():void
openWebpage(URI):void
openWebpage(URL):void
```

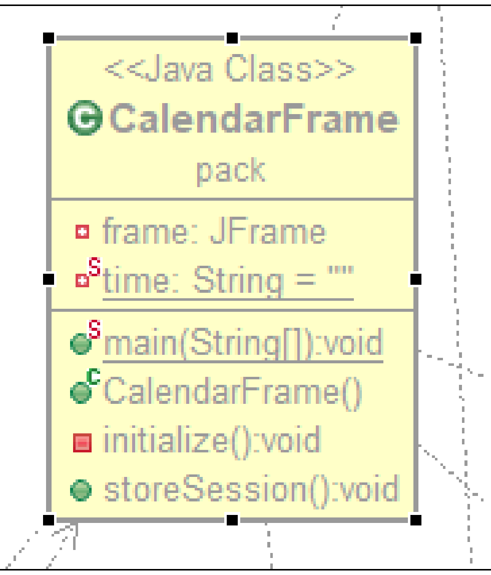

<<Java Class>>
ProfileFrame
pack
frame: JFrame
userid: int = SuccessfulLoginFrame.getID()
usernameField: JTextField
passwordField: JPasswordField
emailField: JTextField
teachingField: JTextField
image_file: File
gender: String
type: String
description_cut: String = ""
description_cut_2: String = ""
description_cut_3: String = ""
description_cut_4: String = ""
description_cut_5: String = ""
description_cut_6: String = ""
password_array: char[]
password_string: String
image_changed: boolean = false
universities_array: String[]
main(String[]):void
ProfileFrame()
initialize():void
<<Java Class>>
RegisterFrame
pack
frame: JFrame
emailField: JTextField
usernameField: JTextField
teachingField: JTextField
teacher: boolean = false
passwordField: JPasswordField
image_path: String = null
password_array: char[]
password_string: String
gender: String
type: String
image_file: File
universities_array: String[]
addressField: JTextField
main(String[]):void
RegisterFrame()
initialize():void
isTeacher():boolean
setTeacher(boolean):void
<<Java Class>>
SessionsFrame
pack
frame: JFrame
teacher1_username: String = ""
teacher2_username: String = ""
teacher3_username: String = ""
teacher4_username: String = ""
teacher1_address: String = ""
teacher2_address: String = ""
teacher3_address: String = ""
teacher4_address: String = ""
teacher1_time: String = ""
teacher2_time: String = ""
teacher3_time: String = ""
teacher4_time: String = ""
labelPic1: JLabel = new JLabel()
labelPic2: JLabel = new JLabel()
labelPic3: JLabel = new JLabel()
labelPic4: JLabel = new JLabel()
number_of_rows: int = -1
backarrow_file: File = new File("arrows.png")
statistics_file: File = new File("statistics.png")
main(String[]):void
SessionsFrame()
initialize():void
getTeachers():void
deleteSession(String):void
<<Java Class>>
SearchFrame
pack
frame: JFrame
search: String = ""
teacher1_username: String = ""
teacher2_username: String = ""
teacher3_username: String = ""
teacher4_username: String = ""
teacher5_username: String = ""
teacher6_username: String = ""
teacher1_course: String = ""
teacher2_course: String = ""
teacher3_course: String = ""
teacher4_course: String = ""
teacher5_course: String = ""
teacher6_course: String = ""
labelPic1: JLabel = new JLabel()
labelPic2: JLabel = new JLabel()
labelPic3: JLabel = new JLabel()
labelPic4: JLabel = new JLabel()
labelPic5: JLabel = new JLabel()
labelPic6: JLabel = new JLabel()
number_of_rows: int = -1
main(String[]):void
SearchFrame(String)
initialize(String):void
getSearch():void



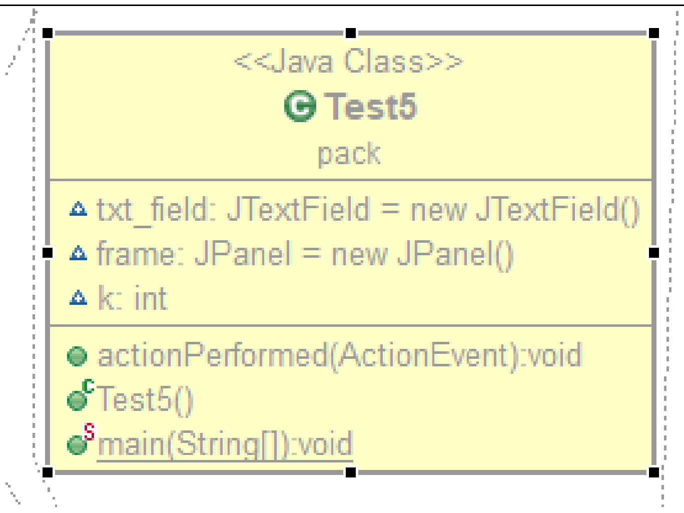
<<Java Class>>
InboxFrame
pack
frame: JFrame
connection: Connection = DBMS.getConnection()
user_id: int = -1
emails_content: String[] = new String[]{"","","","","","","","","","","",""}
emails_content_2: String[] = new String[]{"","","","","","","","","","","","",""}
sender_emails: String[] = new String[]{"","","","","","","","","","","",""}
senders_id: int[] = new int[]{0,0,0,0,0,0,0,0,0,0,0,0,0}
sender_emails_index: int = 0
main(String[]):void
InboxFrame()
initialize():void
getReceiverID():int
getSenderID(int):void
getSenderEmail(int):void
getEmailContent(int):void
getEverything():void



<<Java Class>>
TeachersProfileFrame
pack
frame: JFrame
imageLabel: JLabel = new JLabel()
teacher_username: String = ""
teacher_address: String = ""
teacher_description: String = ""
teacher_university: String = ""
verified_file: File = new File("Ok-icon.png")
main(String[]):void
TeachersProfileFrame(String)
initialize(String):void
getInfo():void
getID(String):int



<<Java Class>>
CalendarFrame
pack
frame: JFrame
time: String = ""
main(String[]):void
CalendarFrame()
initialize():void
storeSession():void



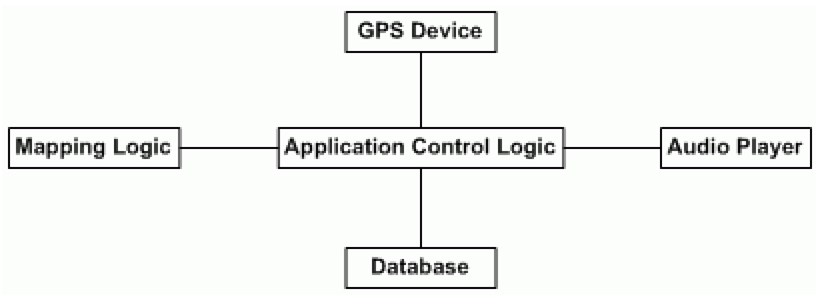
<<Java Class>>
Test5
pack
txt_field: JTextField = new JTextField()
frame: JPanel = new JPanel()
k: int
actionPerformed(ActionEvent):void
Test5()
main(String[]):void

4.1 High-Level Design (Architecture)

The high-level view or architecture consists of Synkers major components:

- Explore
- Messages
- Profile
- Sessions

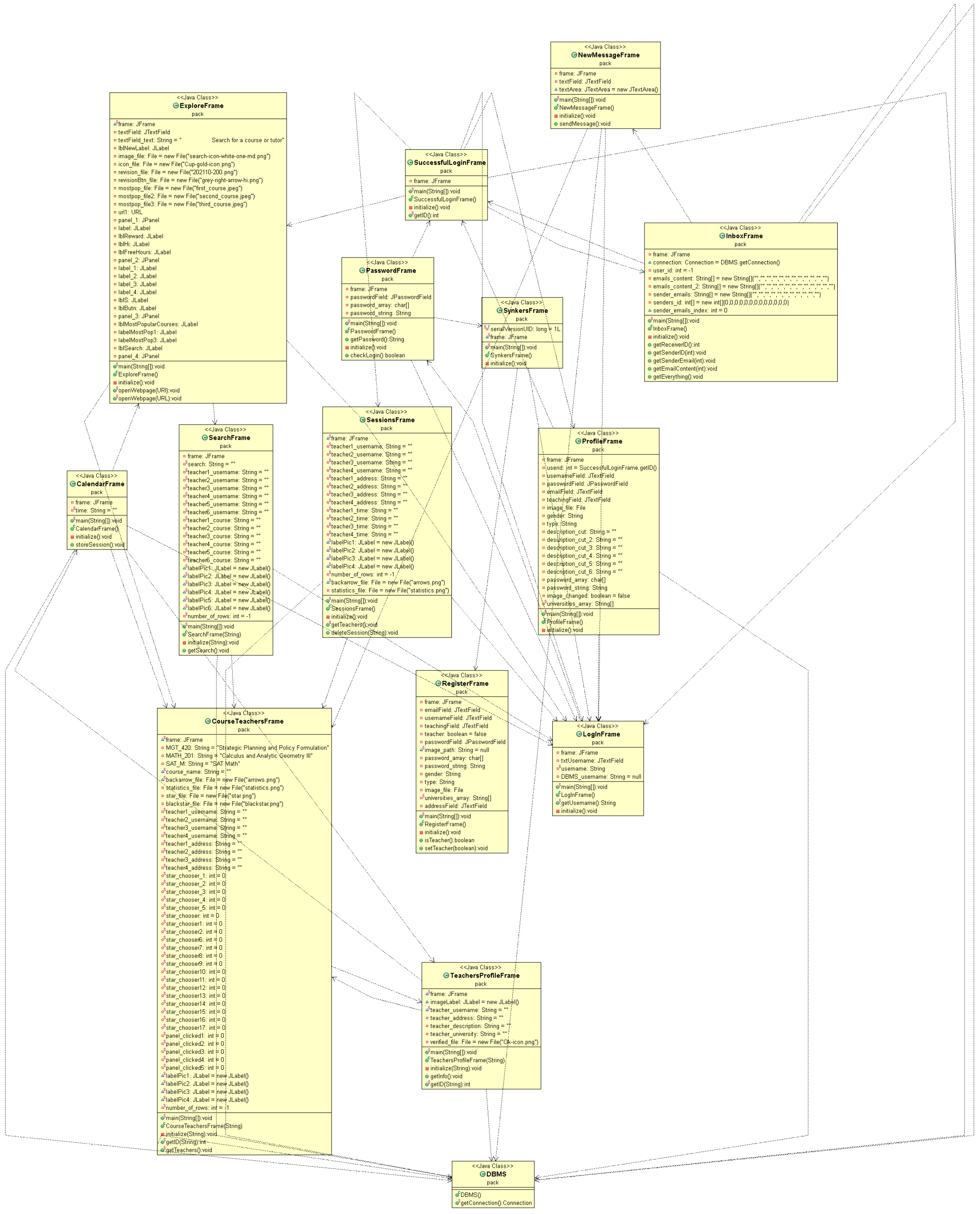
Logical view:



System Architecture

- The **GPS device** provides the user's location on campus (longitude and latitude coordinates). In basic mode, the user's position is used to decide which tutors are close to which student.
- The **Database** is a central repository for data on students and teachers, their names, passwords, addresses, description, sessions, inbox, images etc.
- The **Audio Player** controls playback of audio files where sessions can be recorded.
- Given a position on earth, the **Mapping Logic** will calculate nearby teachers..
- The **Application Control Logic** is the main driver of the application. It presents information to the user and reacts to user inputs.

4.2 Detailed Class Design


<<Java Class>>
ExploreFrame
pack
frame: JFrame
textField: JTextField
textField_text: String = "   Search for a course or tutor"
lblNewLabel: JLabel
image_file: File = new File("search-icon-white-one-md.png")
icon_file: File = new File("Cup-gold-icon.png")
revision_file: File = new File("202110-200.png")
revisionBtn_file: File = new File("grey-right-arrow-hi.png")
mostpop_file: File = new File("first_course.jpeg")
mostpop_file2: File = new File("second_course.jpeg")
mostpop_file3: File = new File("third_course.jpeg")
url1: URL
panel_1: JPanel
label: JLabel
lblReward: JLabel
lblHi: JLabel
lblFreeHours: JLabel
panel_2: JPanel
label_1: JLabel
label_2: JLabel
label_3: JLabel
label_4: JLabel
lblS: JLabel
lblButn: JLabel
panel_3: JPanel
lblMostPopularCourses: JLabel
labelMostPop1: JLabel
labelMostPop3: JLabel
lblSearch: JLabel
panel_4: JPanel
main(String[]):void
ExploreFrame()
initialize():void
openWebpage(URI):void
openWebpage(URL):void
<<Java Class>>
NewMessageFrame
pack
frame: JFrame
textField: JTextField
textArea: JTextArea = new JTextArea()
main(String[]):void
NewMessageFrame()
initialize():void
sendMessage():void
<<Java Class>>
SuccessfulLoginFrame
pack
frame: JFrame
main(String[]):void
SuccessfulLoginFrame()
initialize():void
getID():int
<<Java Class>>
InboxFrame
pack
frame: JFrame
connection: Connection = DBMS.getConnection()
user_id: int = -1
emails_content: String[] = new String[]{"","","","","","","","","","","",""}
emails_content_2: String[] = new String[]{"","","","","","","","","","","","",""}
sender_emails: String[] = new String[]{"","","","","","","","","","","",""}
senders_id: int[] = new int[]{0,0,0,0,0,0,0,0,0,0,0,0,0,0}
sender_emails_index: int = 0
main(String[]):void
InboxFrame()
initialize():void
getReceiverID():int
getSenderID(int):void
getSenderEmail(int):void
getEmailContent(int):void
getEverything():void
<<Java Class>>
PasswordFrame
pack
frame: JFrame
passwordField: JPasswordField
password_array: char[]
password_string: String
main(String[]):void
PasswordFrame()
getPassword():String
initialize():void
checkLogin():boolean
<<Java Class>>
SynkersFrame
pack
serialVersionUID: long = 1L
frame: JFrame
main(String[]):void
SynkersFrame()
initialize():void
<<Java Class>>
CalendarFrame
pack
frame: JFrame
time: String = ""
main(String[]):void
CalendarFrame()
initialize():void
storeSession():void
<<Java Class>>
SearchFrame
pack
frame: JFrame
search: String = ""
teacher1_username: String = ""
teacher2_username: String = ""
teacher3_username: String = ""
teacher4_username: String = ""
teacher5_username: String = ""
teacher6_username: String = ""
teacher1_course: String = ""
teacher2_course: String = ""
teacher3_course: String = ""
teacher4_course: String = ""
teacher5_course: String = ""
teacher6_course: String = ""
labelPic1: JLabel = new JLabel()
labelPic2: JLabel = new JLabel()
labelPic3: JLabel = new JLabel()
labelPic4: JLabel = new JLabel()
labelPic5: JLabel = new JLabel()
labelPic6: JLabel = new JLabel()
number_of_rows: int = -1
main(String[]):void
SearchFrame(String)
initialize(String):void
getSearch():void
<<Java Class>>
SessionsFrame
pack
frame: JFrame
teacher1_username: String = ""
teacher2_username: String = ""
teacher3_username: String = ""
teacher4_username: String = ""
teacher1_address: String = ""
teacher2_address: String = ""
teacher3_address: String = ""
teacher4_address: String = ""
teacher1_time: String = ""
teacher2_time: String = ""
teacher3_time: String = ""
teacher4_time: String = ""
labelPic1: JLabel = new JLabel()
labelPic2: JLabel = new JLabel()
labelPic3: JLabel = new JLabel()
labelPic4: JLabel = new JLabel()
number_of_rows: int = -1
backarrow_file: File = new File("arrows.png")
statistics_file: File = new File("statistics.png")
main(String[]):void
SessionsFrame()
initialize():void
getTeachers():void
deleteSession(String):void
<<Java Class>>
ProfileFrame
pack
frame: JFrame
userid: int = SuccessfulLoginFrame.getID()
usernameField: JTextField
passwordField: JPasswordField
emailField: JTextField
teachingField: JTextField
image_file: File
gender: String
type: String
description_cut: String = ""
description_cut_2: String = ""
description_cut_3: String = ""
description_cut_4: String = ""
description_cut_5: String = ""
description_cut_6: String = ""
password_array: char[]
password_string: String
image_changed: boolean = false
universities_array: String[]
main(String[]):void
ProfileFrame()
initialize():void
<<Java Class>>
RegisterFrame
pack
frame: JFrame
emailField: JTextField
usernameField: JTextField
teachingField: JTextField
teacher: boolean = false
passwordField: JPasswordField
image_path: String = null
password_array: char[]
password_string: String
gender: String
type: String
image_file: File
universities_array: String[]
addressField: JTextField
main(String[]):void
RegisterFrame()
initialize():void
isTeacher():boolean
setTeacher(boolean):void
<<Java Class>>
LogInFrame
pack
frame: JFrame
txtUsername: JTextField
username: String
DBMS_username: String = null
main(String[]):void
LogInFrame()
getUsername():String
initialize():void
<<Java Class>>
CourseTeachersFrame
pack
frame: JFrame
MGT_420: String = "Strategic Planning and Policy Formulation"
MATH_201: String = "Calculus and Analytic Geometry III"
SAT_M: String = "SAT Math"
course_name: String = ""
backarrow_file: File = new File("arrows.png")
statistics_file: File = new File("statistics.png")
star_file: File = new File("star.png")
blackstar_file: File = new File("blackstar.png")
teacher1_username: String = ""
teacher2_username: String = ""
teacher3_username: String = ""
teacher4_username: String = ""
teacher1_address: String = ""
teacher2_address: String = ""
teacher3_address: String = ""
teacher4_address: String = ""
star_chooser_1: int = 0
star_chooser_2: int = 0
star_chooser_3: int = 0
star_chooser_4: int = 0
star_chooser_5: int = 0
star_chooser: int = 0
star_chooser1: int = 0
star_chooser2: int = 0
star_chooser6: int = 0
star_chooser7: int = 0
star_chooser8: int = 0
star_chooser9: int = 0
star_chooser10: int = 0
star_chooser11: int = 0
star_chooser12: int = 0
star_chooser13: int = 0
star_chooser14: int = 0
star_chooser15: int = 0
star_chooser16: int = 0
star_chooser17: int = 0
panel_clicked1: int = 0
panel_clicked2: int = 0
panel_clicked3: int = 0
panel_clicked4: int = 0
panel_clicked5: int = 0
labelPic1: JLabel = new JLabel()
labelPic2: JLabel = new JLabel()
labelPic3: JLabel = new JLabel()
labelPic4: JLabel = new JLabel()
number_of_rows: int = -1
main(String[]):void
CourseTeachersFrame(String)
initialize(String):void
getID(String):int
getTeachers():void
<<Java Class>>
TeachersProfileFrame
pack
frame: JFrame
imageLabel: JLabel = new JLabel()
teacher_username: String = ""
teacher_address: String = ""
teacher_description: String = ""
teacher_university: String = ""
verified_file: File = new File("Ok-icon.png")
main(String[]):void
TeachersProfileFrame(String)
initialize(String):void
getInfo():void
getID(String):int
<<Java Class>>
DBMS
pack
DBMS()
getConnection():Connection