Bilkent University

Department of Computer Engineering

SENIOR DESIGN PROJECT

Project Name: UniStud

Low Level Design Report

Supervisor:

Prof. H. Altay Güvenir

Jury Members:

Prof. Özgür Ulusoy

Prof. Uğur Güdükbay

Innovation Expert:

Melih Gezer

Presented by:

Aurel Hoxha

Albjon Gjuzi

Arba Hoxha

Eniselda Tusku

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfilment of the requirements of the Senior Design Project, course CS491/2.

Table of Contents

1. Introduction

In the recent years, with the wide popularization of the web, the number of people who actively use different online platforms and applications has increased to an incredible rate, reaching 3.58 billion in 2017 [1]. Among these users' statistics show that the majority of them belong to a group age between 17-28 years old, making these people a very ideal target group for application developers. Students seem to use these platforms excessively but even though here are different applications that help them with several individual needs, there is a lack in the market for application that allow students to access several services in the same time, saving their time, money and energy.

If you are a student enrolled in a university or even an aspiring student, you have to navigate between several applications in order to access services related to the search of the opportunities, tutorials and online courses, finding books or even exploring nearby events. What we aim is to introduce an alternative that allows them to access these services in a single application in a simpler and more effective way.

UniStud will be offered as a web service and as an Android application, in this way reaching a higher number of users and operating better according to the demand of the users. What makes UniStud innovative, is the fact that a student is fed with relevant information without passing from link to another. UniStud will keep users updated with everything that is happening around them. The users can not miss the new opportunities and will easily engage into their educational path. In this report, detailed information about current platforms, proposed system and how it will be implemented will be provided.

1.1 Design Trade-Offs

1.1.1 Security vs. Cost

UniStud collects several data from users, which is composed of multimedia format data, videos or images and private information of different students. Thus, it is crucial for the system to ensure security and keep the information of the users secure. For security, we rely on encrypted databases. Relatedly, the security introduces monetary, time, and labour cost.

1.1.2 Space vs. Speed

The large amount of data introduces difficulties in saving and fetching the data from the servers. Such difficulties are the delays between the operations. The greatest space requirements are introduced by the continuous storage of the livestream tutorial components and its frequent update. This naturally causes an increment in the processing time and slows down the system. The communication between the server and the client for the majority of the data in the system will not slow down the system. However, in case of overloading, to achieve the same result with the livestream component, the quality of the videos will be decreased. All connections and data exchanges with the server will be handled in background threads. This way, we will be able to keep the system fast and highly responsive while managing lots of data at the same time.

1.2 Engineering Standards

In all documentations, UML [2] design principles are used in the description of class interfaces, diagrams, scenarios and use cases, subsystem compositions, and hardware-software components depiction. UML is a commonly used standard that allows simpler description of the components of a software project. With standard UML models, we were able to represent the system structure, software components, and functionalities.

1.3 Interface Documentation Guidelines

All the classes that are going to be used in the system will be documented following some conventions. Class, variables and method names will use ‘Camel-Case’ to diminish the difficulty in understanding the code. Every class will follow the same design pattern where the class name comes first, the attributes follow, and finally the methods are listed. The detailed outline is provided below:

1.4 Definitions, Keywords, Acronyms and Abbreviations

Career Opportunities: Browse section for different universities and internships.

UniTrade: The section where students will be able to sell/buy/loan study materials.

UniStream: The section where students will be able to watch or create tutorials.

UI: User Interface

API: Application Programming Interface

HTTP: Hypertext Transfer Protocol

TCP: Transmission Control Protocol

Client: The part of the system that user interacts with.

Server: The part of the system that responds to client’s requests. It is responsible for data management, API interactions and logical operations.

2. Packages

2.1 Client

The client service corresponds to both mobile and web applications, because all functionalities will be implemented in both platforms. The client is the presentation layer of our system. Most of the requests that will be made will be sent to the server and data will be returned and displayed to the user accordingly. Client is responsible for managing users' operations on the system, presenting the data from the server to the user and also notifying the user when it is necessary. Client subsystem includes Presentation Tier and Control Tier.

2.1.1 Presentation Package

Presentation Tier is responsible for all of the user interface interactions and it uses Control Tier in order to communicate with the server.

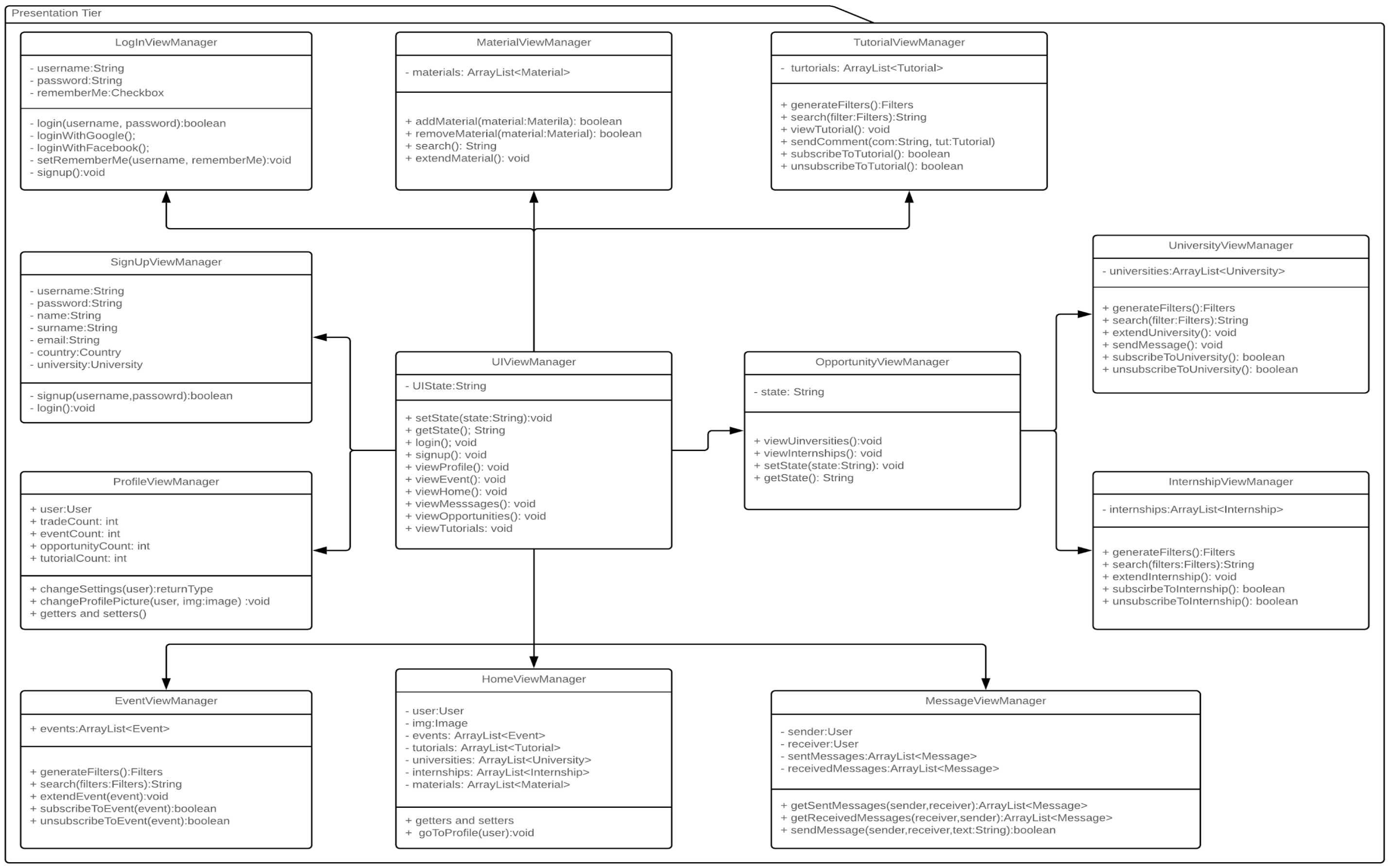

Presentation Tier
LogInViewManager
- username:String
- password:String
- rememberMe:Checkbox
- login(username, password):boolean
- loginWithGoogle();
- loginWithFacebook();
- setRememberMe(username, rememberMe):void
- signup():void
MaterialViewManager
- materials: ArrayList<Material>
+ addMaterial(material:Materila): boolean
+ removeMaterial(material:Material): boolean
+ search(): String
+ extendMaterial(): void
TutorialViewManager
- turtorials: ArrayList<Tutorial>
+ generateFilters():Filters
+ search(filter:Filters):String
+ viewTutorial(): void
+ sendComment(com:String, tut:Tutorial)
+ subscribeToTutorial(): boolean
+ unsubscribeToTutorial(): boolean
SignUpViewManager
- username:String
- password:String
- name:String
- surname:String
- email:String
- country:Country
- university:University
- signup(username,passowrd):boolean
- login():void
UIViewManager
- UIState:String
+ setState(state:String):void
+ getState(); String
+ login(); void
+ signup(): void
+ viewProfile(): void
+ viewEvent(): void
+ viewHome(): void
+ viewMesssages(): void
+ viewOpportunities(): void
+ viewTutorials: void
OpportunityViewManager
- state: String
+ viewUinversities():void
+ viewInternships(): void
+ setState(state:String): void
+ getState(): String
UniversityViewManager
- universities:ArrayList<University>
+ generateFilters():Filters
+ search(filter:Filters):String
+ extendUniversity(): void
+ sendMessage(): void
+ subscribeToUniversity(): boolean
+ unsubscribeToUniversity(): boolean
ProfileViewManager
+ user:User
+ tradeCount: int
+ eventCount: int
+ opportunityCount: int
+ tutorialCount: int
+ changeSettings(user):returnType
+ changeProfilePicture(user, img:image) :void
+ getters and setters()
InternshipViewManager
- internships:ArrayList<Internship>
+ generateFilters():Filters
+ search(filters:Filters):String
+ extendInternship(): void
+ subscirbeToInternship(): boolean
+ unsubscribeToInternship(): boolean
EventViewManager
+ events:ArrayList<Event>
+ generateFilters():Filters
+ search(filters:Filters):String
+ extendEvent(event):void
+ subscribeToEvent(event):boolean
+ unsubscribeToEvent(event):boolean
HomeViewManager
- user:User
- img:Image
- events: ArrayList<Event>
- tutorials: ArrayList<Tutorial>
- universities: ArrayList<University>
- internships: ArrayList<Internship>
- materials: ArrayList<Material>
+ getters and setters
+ goToProfile(user):void
MessageViewManager
- sender:User
- receiver:User
- sentMessages:ArrayList<Message>
- receivedMessages:ArrayList<Message>
+ getSentMessages(sender,receiver):ArrayList<Message>
+ getReceivedMessages(receiver,sender):ArrayList<Message>
+ sendMessage(sender,receiver,text:String):boolean

Figure 1 – Presentation Class Diagram

- **UIViewManager:** Class that control all views of the system and coordinates the change from one to another.
- **LoginViewManager:** This class handles the first page operations and UI that the user will see upon opening the system.
- **SignupViewManager:** This class handles the Sign-Up operations and UI for users that are not yet registered to our system.
- **ProfileViewManager:** This class handles all operations related to UI for a user’s account and settings and the UI.
- **MessageViewManger:** This class handles all operation related to UI for a user sent and received messages.
- **EventViewManager:** This class handles all event related operations and the UI.
- **HomeViewManager:** This class handles all Home Page related operations and the UI.
- **MaterialViewManager:** This class handles all Study Material related operations and the UI.
- **OpportunityViewManager:** This class can direct user to either University or Internship view and coordinates movement between those two.
- **InternshipViewManger:** This class handles all Internship related operations and the UI.
- **UniversityViewManager:** This class handles all Universities related operations and the UI.
- **TutorialViewManager:** This class handles all Tutorial related operations and the UI.

2.1.2 Controller Package

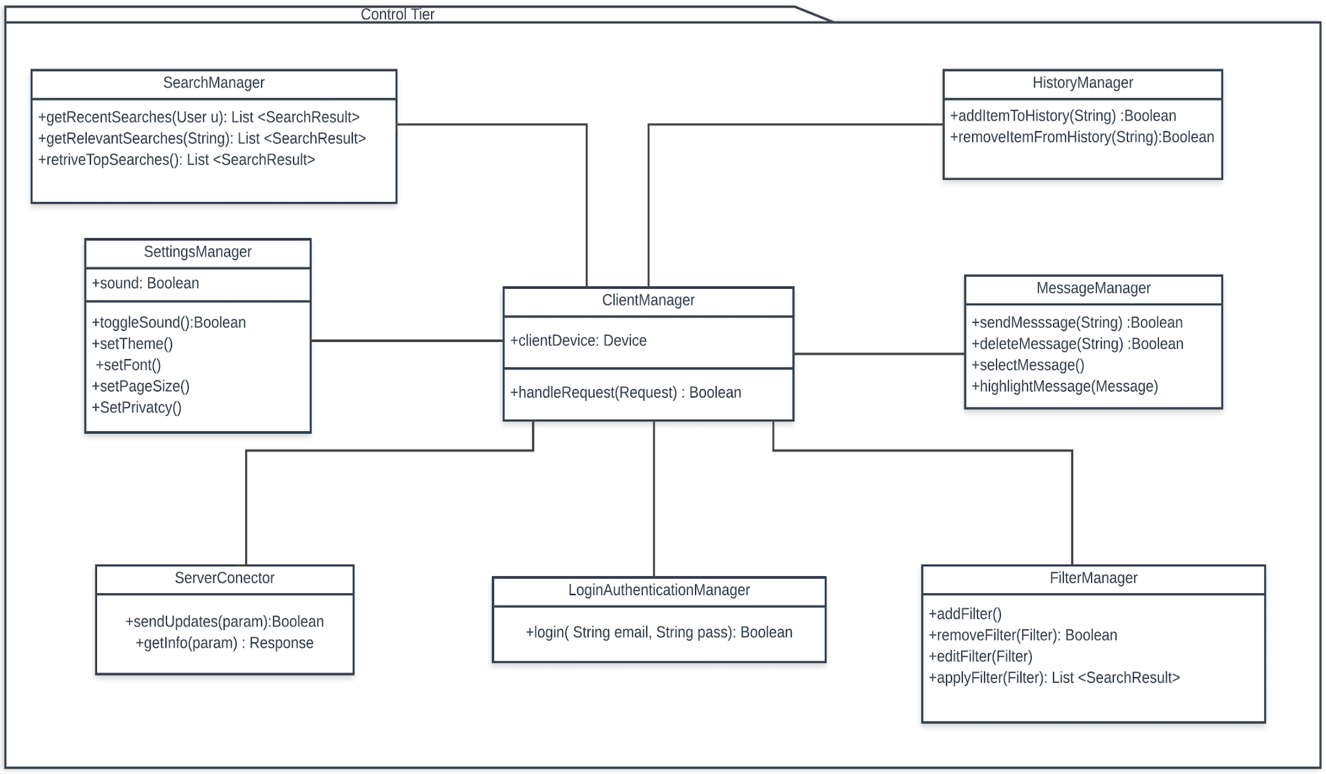

Control Tier
SearchManager
+getRecentSearches(User u): List <SearchResult>
+getRelevantSearches(String): List <SearchResult>
+retriveTopSearches(): List <SearchResult>
HistoryManager
+addItemToHistory(String) :Boolean
+removeItemFromHistory(String):Boolean
SettingsManager
+sound: Boolean
+toggleSound():Boolean
+setTheme()
+setFont()
+setPageSize()
+SetPrivatcy()
ClientManager
+clientDevice: Device
+handleRequest(Request) : Boolean
MessageManager
+sendMesssage(String) :Boolean
+deleteMessage(String) :Boolean
+selectMessage()
+highlightMessage(Message)
ServerConector
+sendUpdates(param):Boolean
+getInfo(param) : Response
LoginAuthenticationManager
+login(String email, String pass): Boolean
FilterManager
+addFilter()
+removeFilter(Filter): Boolean
+editFilter(Filter)
+applyFilter(Filter): List <SearchResult>

Figure 2 – Controller Class Diagram

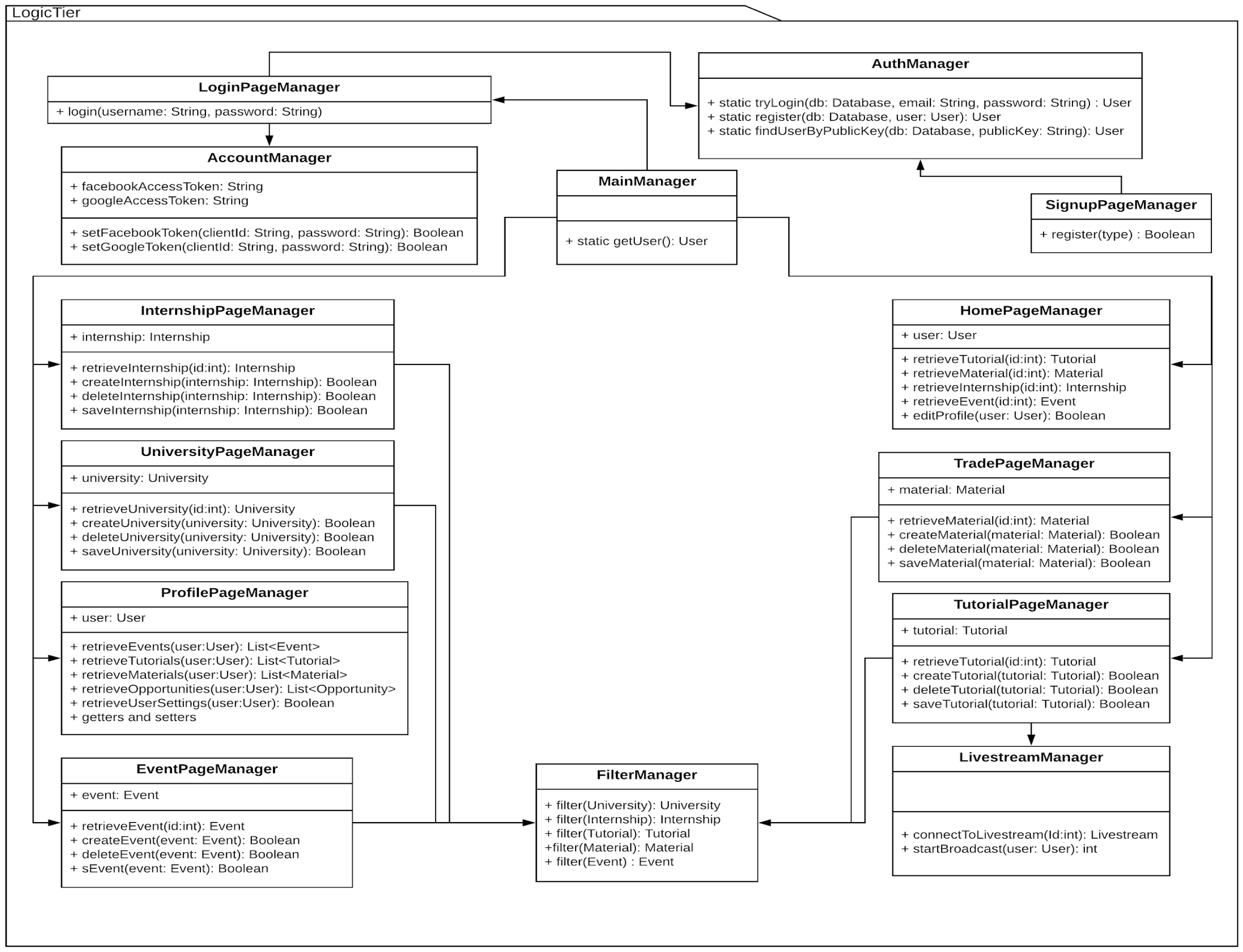
- **ServerConnector:** This class handles the communication between the client and the servers.
- **SettingsManager:** This class updates the settings of the user according to their changes and preferences.
- **SearchManager:** Class that stores the previous searches in order to help the user and also directs the search to the server to get the results.
- **LoginAuthenticationManager:** Class that handles login operations.
- **FilterManager:** Class that handles filters for all types of searches and applies the search according to them.
- **MessageManager:** Class that handles message sending and receiving operations.
- **HistoryManager:** Class that saves and updates the history according to user operations and interaction with the system.

2.2 Server

Server is a crucial part of our system that will be responsible for the all the interactions that the user will have with the system. The tutorials that will be offered will be recorded on the client side and delivered to the server. The server will receive this data and will make it accessible for the other users in the platform. It will also handle the information that will be entered regarding the other categories such as opportunities, trading and events and will make the necessary adjustments of this information accordingly.

2.2.1 Logic Tier

The Logic tier is the application layer responsible for the control of the flow of information between presentation layer and data layer. It accommodates all the heavy operations the application needs to handle.



LogicTier
LoginPageManager
+ login(username: String, password: String)
AuthManager
+ static tryLogin(db: Database, email: String, password: String) : User
+ static register(db: Database, user: User): User
+ static findUserByPublicKey(db: Database, publicKey: String): User
AccountManager
+ facebookAccessToken: String
+ googleAccessToken: String
+ setFacebookToken(clientId: String, password: String): Boolean
+ setGoogleToken(clientId: String, password: String): Boolean
MainManager
+ static getUser(): User
SignupPageManager
+ register(type) : Boolean
InternshipPageManager
+ internship: Internship
+ retrieveInternship(id:int): Internship
+ createInternship(internship: Internship): Boolean
+ deleteInternship(internship: Internship): Boolean
+ saveInternship(internship: Internship): Boolean
HomePageManager
+ user: User
+ retrieveTutorial(id:int): Tutorial
+ retrieveMaterial(id:int): Material
+ retrieveInternship(id:int): Internship
+ retrieveEvent(id:int): Event
+ editProfile(user: User): Boolean
UniversityPageManager
+ university: University
+ retrieveUniversity(id:int): University
+ createUniversity(university: University): Boolean
+ deleteUniversity(university: University): Boolean
+ saveUniversity(university: University): Boolean
TradePageManager
+ material: Material
+ retrieveMaterial(id:int): Material
+ createMaterial(material: Material): Boolean
+ deleteMaterial(material: Material): Boolean
+ saveMaterial(material: Material): Boolean
ProfilePageManager
+ user: User
+ retrieveEvents(user:User): List<Event>
+ retrieveTutorials(user:User): List<Tutorial>
+ retrieveMaterials(user:User): List<Material>
+ retrieveOpportunities(user:User): List<Opportunity>
+ retrieveUserSettings(user:User): Boolean
+ getters and setters
TutorialPageManager
+ tutorial: Tutorial
+ retrieveTutorial(id:int): Tutorial
+ createTutorial(tutorial: Tutorial): Boolean
+ deleteTutorial(tutorial: Tutorial): Boolean
+ saveTutorial(tutorial: Tutorial): Boolean
EventPageManager
+ event: Event
+ retrieveEvent(id:int): Event
+ createEvent(event: Event): Boolean
+ deleteEvent(event: Event): Boolean
+ sEvent(event: Event): Boolean
FilterManager
+ filter(University): University
+ filter(Internship): Internship
+ filter(Tutorial): Tutorial
+filter(Material): Material
+ filter(Event) : Event
LivestreamManager
+ connectToLivestream(Id:int): Livestream
+ startBroadcast(user: User): int

Figure 3 – Logic Class Diagram

- **AccountManager:** The main function of this class is to maintain the users' account.
- **AuthManager:** This class is used to save, retrieve and update information about the users from the database.
- **MainManager:** Fundamental class that will be responsible for accessing the services provided by the application. It will gather all the information and deal with the interactions of the user leading to each of these individual services.
- **HomePageManager:** Class that will be responsible for storing information and directing the user to specific services.
- **TutorialPageManager:** This class is responsible for the tutorials and will have all the necessary information regarding them. It will be responsible for the interaction of the user with the tutorials both in offering and accessing.
- **InternshipPageManager:** This class is responsible for the internships and will have all the necessary information regarding them. It will be responsible for the interaction of the user with the internships.
- **UniversityPageManager:** This class is responsible for the university related search and will have all the necessary information regarding them.
- **EventPageManager:** This class is responsible for the events related and will have all the necessary information regarding them. It will deal with the interactions on both offering and accessing an event.
- **TradePageManager:** This class that is responsible for the displaying and trading of all of the academic materials and will be responsible for the interactions of the user with these materials.
- **LogicPageManager:** This class is responsible for handling the Login process of the user and dealing with the provided data.
- **SignUpPageManager:** This class is responsible for handling the SignUp process of the user, dealing with the provided data and managing the interactions to make the user part of the platform.
- **ProfilePageManager:** This class is responsible for the user information and will store all the necessary data.
- **FilterManager:** This class is responsible for filtering all the data content that is found in the system. It will use the data according to the particular type.
- **LivestreamManager:** This class is responsible for streaming the tutorial in real-time to other users.

4.1.2 Data Tier

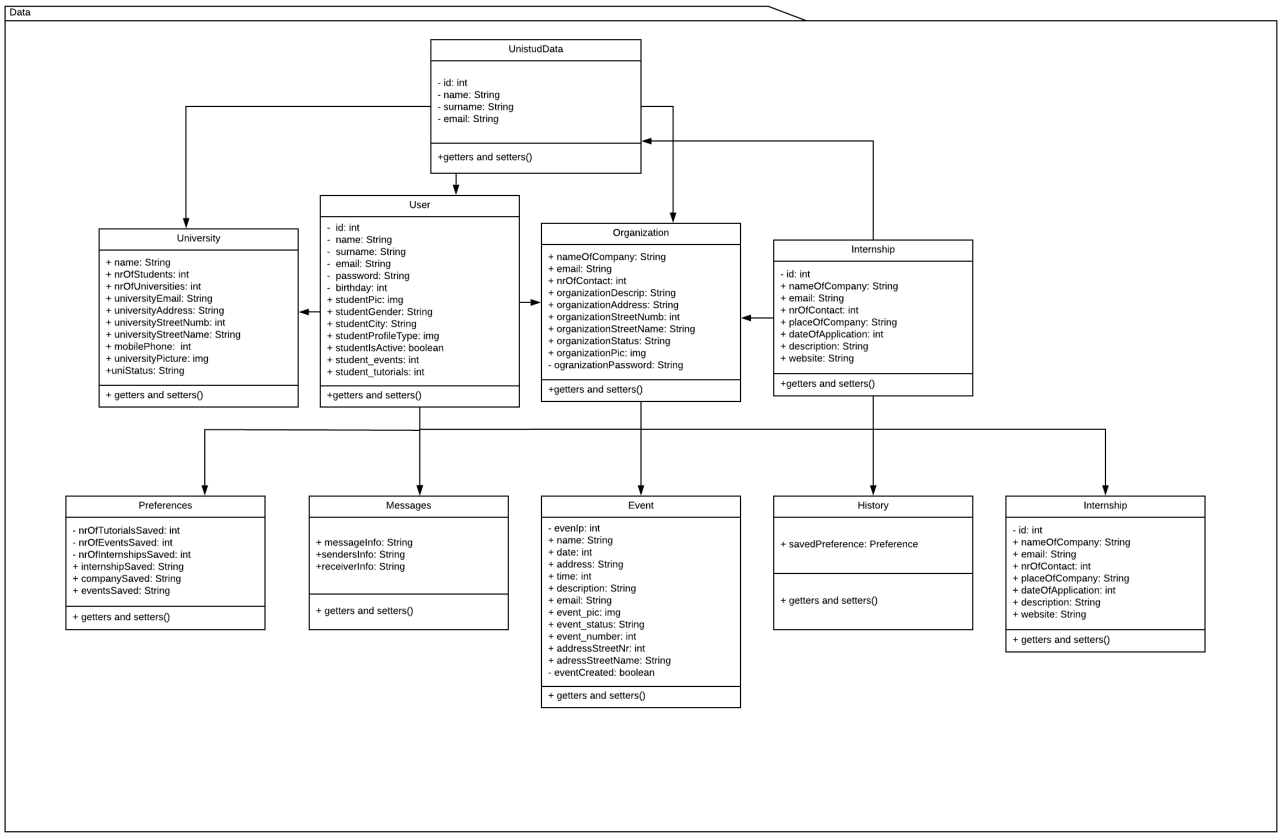


Figure 4 – Data Class Diagram

- **User:** Data class representing users.
- **University:** Data class representing all the universities in UniStud.
- **Internship:** This class contains the information about the internships offered by the companies in the platform.
- **Organization:** This data class concerns with the information connecting the other data classes such as preferences, messages, event, history and tutorial.
- **Preferences:** This class handles all the user's preferences by the filters provided in the platform.
- **Messages:** This class represents the data concerning the messages between the users of the platform.
- **Event:** This data class represents all the events provided and advertised in the platform.
- **History:** The data class which handles the information of the user's history.
- **Tutorial:** This data class represents all the available tutorials in the platform.

3. Class Interfaces

3.1 Client

3.1.1 Presentation

class UIViewManager	
Attributes	
private String UIState	
Methods	
Public void setState(String state)	This method changes the state according to user interaction with the system. Default is login.
Public String getState()	This method returns the current state of the system, which is needed from controller in order to display correct information to the user.
Public void login()	This method redirects the user to login page when he presses logout or opens the app.
Public void signup()	This method redirects the user to signup page when user clicks “I don’t have an account”
Public void viewProfile()	This method redirects the user to his/her profile after user presses his/her profile picture.
Public void viewEvent()	This method redirects the user to events page after user selects events from menu.
Public void viewHome()	This method redirects the user to his/her home page after user logs in or after user selects home from menu.
Public void viewMessages()	This method redirects the user to his/her messages after user presses envelope icon from homepage.
Public void viewOpportunities()	This method redirects the user to opportunities page after user selects opportunities from menu.
Public void viewTutorials()	This method redirects the user to tutorials page after user selects tutorials from menu.

class LogInViewManager	
Attributes	
private String username private String password private Checkbox rememberMe	
Methods	
Public boolean login(String username, String password)	This method sends the information to controller when user presses login button.
Public boolean loginWithGoogle()	This method shows user a google login window and asks for credentials and permission to get necessary information.
Public boolean loginWithFacebook()	This method shows user a facebook login window and asks for credentials and permission to get necessary information.
Public void setRememberMe(String user, String passoword)	This method sets remember me checkbox to true for the current device. Default is false.
Public void signup()	This method redirects the user to signup page when user clicks “I don’t have an account”

class HomeViewManager	
Attributes	
private User user private Image img private ArrayList<Event> events private ArrayList<Tutorial> tutorials private ArrayList<Material> materials private ArrayList<Internship> internships private ArrayList<Material> materials	
Methods	
Public void goToProfile(User user)	This method allows user to go to its profile page.
Getters and setters	

class SignUpViewManager	
Attributes	
private String username private String password private String name private String surname private String email private Country country private University uni	
Methods	
Public boolean signup(String username, String password, String name, String surname, String email, Country country, University uni)	This method sends the information to controller when user presses sign up button.
Public boolean login()	This method redirects user to login page when it presses “Already have an account”.

class ProfileUpViewManager	
Attributes	
private User user private int tradeCount private int eventCount private int opportunityCount private int tutorialCount	
Methods	
Public void changeSettings(User user)	This method allows user to change the settings for its account and application.
Public void changeProfilePicture(User user, Image img)	This method allows user to change its profile picture.
Getters and setters	

class MaterialViewManager	
Attributes	
private ArrayList<Material> materials	
Methods	
Public ArrayList<Filter> generateFilters()	This method allows user to generate filters and conduct their search according to them.
Public ArrayList<Material> search(ArrayList<Filter> filters)	This method returns the results according to filters edited from the user.
Public void extendMaterial()	This method is invoked when user clicks on a university to see a more detailed information about it and have the option to subscribe/unsubscribe to it.
Public boolean addMaterial(Material material)	This method is invoked when user want to add an item that he wants to sell in the list of the materials for sale.

class UniversityViewManager	
Attributes	
private ArrayList<University> universities	
Methods	
Public ArrayList<Filter> generateFilters()	This method allows user to generate filters and conduct their search according to them.
Public ArrayList<University> search(ArrayList<Filter> filters)	This method returns the results according to filters edited from the user.
Public void extendUniversity()	This method is invoked when user clicks on a university to see a more detailed information about it and have the option to subscribe/unsubscribe to it.
Public void sendMessage()	This method opens a new screen, allowing the user to send a message to any of the students.
Public boolean subscribeToUniverisity()	This method allows user to get information and notifications about selected university.
Public boolean unsubscribeToUniversity()	This method allows user to stop getting notifications about selected university.

class InternshipViewManager	
Attributes	
private ArrayList<Internship> internships	
Methods	
Public ArrayList<Internship> generateFilters()	This method allows user to generate filters and conduct their search according to them.
Public ArrayList<Internship> search(ArrayList<Filter> filters)	This method returns the results according to filters edited from the user.
Public void extendInternship()	This method is invoked when user clicks on a university to see a more detailed information about it and have the option to subscribe/unsubscribe to it.
Public boolean subscribeToInternship()	This method allows user to get information and notifications about selected internship.
Public boolean unsubscribeToInternship()	This method allows user to stop getting notifications about selected internship.

class EventViewManager	
Attributes	
private ArrayList<Event> events	
Methods	
Public ArrayList<Filter> generateFilters()	This method allows user to generate filters and conduct their search according to them.
Public ArrayList<Event> search(ArrayList<Filter> filters)	This method returns the results according to filters edited from the user.
Public void extendEvent()	This method is invoked when user clicks on a university to see a more detailed information about it and have the option to subscribe/unsubscribe to it.
Public boolean subscribeToEvent()	This method allows user to get information and notifications about selected event.
Public boolean unsubscribeToEvent()	This method allows user to stop getting notifications about selected event.

class TutorialViewManager	
Attributes	
private ArrayList<Tutorial> tutorials	
Methods	
Public ArrayList<Filter> generateFilters()	This method allows user to generate filters and conduct their search according to them.
Public ArrayList<Material> search(ArrayList<Filter> filters)	This method returns the results according to filters edited from the user.
Public void extendTutorial()	This method is invoked when user clicks on a university to see a more detailed information about it and have the option to subscribe/unsubscribe to it.
Public boolean subscribeToTutorial()	This method allows user to get information and notifications about selected tutorial.
Public boolean unsubscribeToTutorial()	This method allows user to stop getting notifications about selected tutorial.
Public void viewTutorial()	This method can only work when livestream has started, or the livestream has ended and is saved in the archive of the owner.
Public void sendComment(String com, Tutorial tutorial)	This method allows user to send a comment, which can be question for something they don't understand in case the livestream has not ended yet.

class OpportunityViewManager	
Attributes	
private String state	
Methods	
Public void viewUniversities()	This method allows redirects user to University screen.
Public void viewInternships()	This method allows redirects user to Internship screen.
Public void setState(String state)	This method allows the change of state according to user operations.
Public String getState()	This method gets the current state and sends it to UIViewManager class in order to give correct information.

class MessageViewManager	
Attributes	
private User sender private User receiver private ArrayList<Message> sentMessages private ArrayList<Message> receivedMessages	
Methods	
Public void getSentMessages(User sender, User receiver)	This method allows app to get list of sent messages for the user.
Public void getReceivedMessages(User sender, User receiver)	This method allows app to get list of received messages for the user.
Public boolean sendMessage(User sender, User receiver, String msg)	This method is invoked when user presses send message button.
Getters and setters	

3.1.2 Controller

class LoginAuthenticationManager	
Methods	
+login(String email, String pass): Boolean	This method allows user to login into the system.

Class SearchManager	
Methods	
getRecentSearches(User u): List <SearchResult>	Returns a list of results for recent searches the user has done.
getRelevantSearches(String): List <SearchResult>	This method returns a list of search results related to keyword of search.
retriveTopSearches(): List <SearchResult>	This method returns a list of top searches overall - this is relevant in cases of popular items for selling, tutorials, opportunities and events.

Class FilterManager	
Methods	
addFilter()	This method creates a filter.
removeFilter(Filter): Boolean	This method removes a filter.
editFilter(Filter)	This method edits a certain filter adding or removing characteristics according to the user’s preference.
applyFilter(Filter): List <FilterResult>	This method returns a list of results related to the characteristics of the filter applied.

Class MessageManager	
Methods	
sendMessage(String): Boolean	This method sends a message from one use to another.
deleteMessage(String): Boolean	This method deletes a selected message.
selectMessage()	This method selects one or more messages.
highlightMessage(Message)	This message highlights one or more message so that the user can access them easier.

Class HistoryManager	
Methods	
addItemToHistory(String): Boolean	This method adds an item to the user history.
removeItemFromHistory(String): Boolean	This method removes an item from user history.

Class SettingsManager	
Methods	
toggleSound(): Boolean	This method allows for toggling of sound in the device as on/off.
setTheme()	This method sets the theme in the program according to user’s preference.
setFont()	This method sets Font Size in the program according to user’s preference.
setPageSize()	This method sets the Page Size in the program according to user’s preference.
setPrivacy()	This method edits the privacy according to the user’s preferences.

Class ServerConnector	
Methods	
sendUpdates(param): Boolean	This method sends updates to the server.
+getInfo(param): Response	This method receives the information from the server.

Class ClientManager	
Attributes	
clientDevice: Device	
Methods	
handleRequest(Request): Boolean	This method handles the Client Request.

3.2 Server

3.2.1 Logic

class MainManager	
Methods	
public static User getUser ()	This method retrieves the User that has currently using the system and all the necessary information.

class LoginPageManager	
Methods	
public void login (String username, String password)	This method will perform the authentication.

class AccountManager	
Attributes	
String facebookAccessToken	
String googleAccessToken	
Methods	
public boolean setFacebookToken(String clientId, String password)	This method uses the OAuth 2.0 to get a token for the user’s Facebook account.
public boolean setGoogleToken(String clientId, String password)	This method uses the OAuth 2.0 to get a token for the user’s Google account.

class SignupPageManager	
Methods	
public boolean register(type)	This method will be used to register the users into platform

class AuthManager	
Methods	
public static User tryLogin(Database db, String email, String password)	Given a user name and a password this method checks if a user is not null and a hashed pass- word exists in the database.
public static User register(Database db, User user)	This method creates a new user in the database.
public static User findUserByPublicKey(Database db, String publicKey)	Given a public key this method returns a user that satisfies given information.

class InternshipPageManager	
Attributes	
private Internship internship	
Methods	
public Internship retrieveInternship (int id)	This method retrieves an Internship entity given an id for an Internship.
public boolean createInternship(Internship internship)	This method creates an Internship from the system entity given an Internship.
public boolean deleteIntenship(Internship internship)	This method deletes an Internship from the system entity given an Internship.
public boolean saveInternship(Internship internship)	This method saves an Internship in the HomePage history given an Internship.

class LivestreamManager	
Methods	
public Livestream connectToLivestream (int id)	This method connects a User to a broadcast given an id of a broadcast.
public int startBroadcast(User user)	This method initiates a new broadcast session for a given user and returns the id of the initiated broadcast.

class UniversityPageManager	
Attributes	
private University university	
Methods	
public University retrieveUniversity (int id)	This method retrieves a University entity given an id for a University.
public boolean createUniversity (University university)	This method creates a University from the system entity given a University.
public boolean deleteUniversity (University university)	This method deletes a University from the system entity given a University.
public boolean saveUniversity (University university)	This method saves a University in the HomePage history given a University.

class EventPageManager	
Attributes	
private Event event	
Methods	
public Event retrieveEvent (int id)	This method retrieves an Event entity given an id for an Event.
public boolean createEvent (Event event)	This method creates an Event from the system entity given an Event.
public boolean deleteEvent (Event event)	This method deletes an Event from the system entity given an Event.
public boolean saveEvent (Event event)	This method saves an Event in the HomePage history given an Event.

class TradePageManager	
This class that is responsible for the displaying and trading of all of the academic materials and will be responsible for the interaction of the user with these materials.	
Attributes	
private Material material	
Methods	
public Material retrieveMaterial (int id)	This method retrieves a Material entity given an id for a Material.
public boolean createMaterial (Material material)	This method creates a Material from the system entity given a Material.
public boolean deleteMaterial (Material material)	This method deletes a Material from the system entity given a Material.
public boolean saveMaterial (Material material)	This method saves a Material in the HomePage history given a Material.

class TutorialPageManager	
Attributes	
private Tutorial tutorial	
Methods	
public Tutorial retrieveTutorial (int id)	This method retrieves a Tutorial entity given an id for a Tutorial.
public boolean createTutorial (Tutorial tutorial)	This method creates a Tutorial from the system entity given a Tutorial.
public boolean deleteTutorial (Tutorial tutorial)	This method deletes a Tutorial from the system entity given a Tutorial.
public boolean saveTutorial (Tutorial tutorial)	This method saves a Tutorial in the HomePage history given a Tutorial.

class ProfilePageManager	
Attributes	
private User user	
Methods	
public List<Opportunity> retrieveOpportunity (User user)	This method retrieves the list of the Opportunity objects in the Profile Page.
public List<Tutorial> retrieveTutorial (User user)	This method retrieves the list of the Tutorial objects in the Profile Page.

public List<Material> retrieveMaterial (User user)	This method retrieves the list of the Material objects in the Profile Page.
public List<Event> retrieveEvent (User user)	This method retrieves the list of the Event objects in the Profile Page.
public boolean retrieveUserSettings(User user)	This method retrieves the user's settings (photos, personal info, etc.)
getters and setters	

class HomePageManager	
Attributes	
private User user	
Methods	
public List<Opportunity> retrieveOpportunity (User user)	This method retrieves the list of the Opportunity objects in the Home Page.
public List<Tutorial> retrieveTutorial (User user)	This method retrieves the list of the Tutorial objects in the Home Page.
public List<Material> retrieveMaterial (User user)	This method retrieves the list of the Material objects in the Home Page.
public List<Event> retrieveEvent (User user)	This method retrieves the list of the Event objects in the Home Page.
public boolean editProfile(User user)	This method retrieves the user's settings (photos, personal info, etc.)
getters and setters	

class FilterManager	
Methods	
public List<Opportunity> filterUniversity()	This method filters the universities according to features.
public List<Opportunity> filterTutorial()	This method filters the tutorials according to features.
public List<Opportunity> filterMaterial()	This method filters the materials according to features.
public List<Opportunity> filterEvent()	This method filters the events according to features.
public List<Opportunity> filterInternship()	This method filters the internships according to some features.

3.2.2 Data

class UnistudData
Attributes
private int id
private String name
private String surname
private String email
Methods
getters and setters

class Messages
Attributes
private String messageInfo
public String sendersInfo
public String receiverInfo
Methods
getters and setters

class Preferences
Attributes
private int nrOfTutorialsSaved
private int nrOfInternshipsSaved
private int nrOfEventsSaved
private String internshipSaved
private String universitySaved
public String eventsSaved
Methods
getters and setters

class University
Data class that keeps information about universities in the platform.
Attributes
private int universityID
public int nrOfStudents
public String name
public int nrOfUniversities
public String country
public String universityAddress
public int universityStreetNumber
public String universityStreetName
public int mobilePhone
public img universityPicture
public String uniDescription
private String uniStatus
public String universityEmail
Methods
getters and setters

class History
Attributes
private Preferences savedPreference
Methods
getters and setters

class Event
Attributes
private int eventIP
public int date
public String name
public String place
public String eventDescription
public String eventAddress
public int adressStreetNumber
public String addressStreetName
public int event_number
public int time
public String email
public img event_pic
public String event_status
private boolean eventIsCreated
Methods
getters and setters

class User
Data class that keeps information about users in Unistud platform.
Attributes
private int id
public String name
public String surname
private String email

private String password
public int birthday
public img studentPic
public String studentGender
public String studentCity
public String studentProfileType
public int student_events
public int student_tutorial
public boolean studentIsActive
Methods
getters and setters

class Internship
Data class that keeps information about the internships offered in the platform.
Attributes
public int dateOfApplication
private int InternshipID
public String placeOfCompany
public String nameOfCompany
public String email
public int nrOfContact
public String description
public String website
Methods
getters and setters

class Organization
Data class that keeps information about users who enter the system as organizations.
Attributes
private int id
public String nameOfCompany
public String email
public int nrOfContact
public String organizationDescription
public String organizationAddress
public int organizationStreetNumber
public string organizationStreetName
public img organizationPic
public String organizationStatus
private String organizationPassword
private String organizationStatus
public String organizationWebsite
Methods
getters and setters

4. References

[1] INTERNET USAGE STATISTICS The Internet Big Picture. (n.d.). 2018 https://www.internetworldstats.com/stats.htm. Accessed: 2018-10-14.

[2] “UML - Basics, ibm. http://www.ibm.com/developerworks/rational/ library/769.html.” Accessed: 2018-11-04.