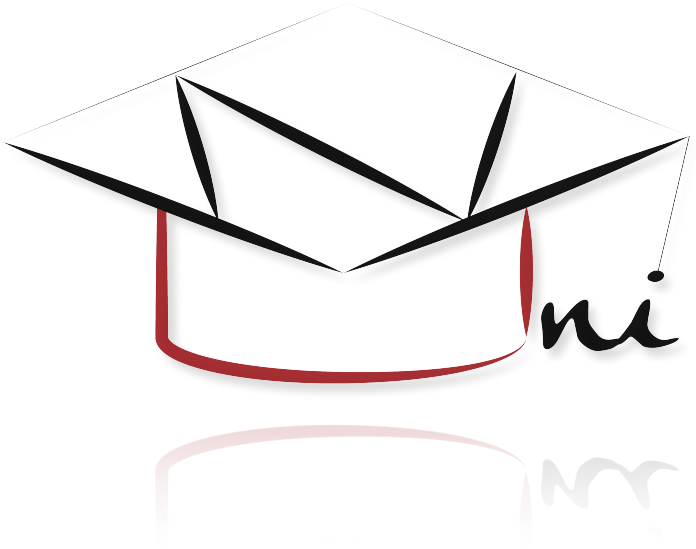
VnUni

Object Oriented Analysis and Design



Instructor: Dr. Truong Ninh Thuan
Author: Team OOAD K57CA
March 23, 2015

Course project

Instructor

Dr. Truong Ninh Thuan - University of Engineering and Technology - VNU

Team members

- Truong Quoc Tuan (Leader) K57CA Student ID: 12020416
- Nguyen Thac Thong K57CA Student ID: 12020624
- Le Van Giap K57CA Student ID: 12020493

Requirement

Give the analysis and design documentation of the software.

Overview

VnUni is a website for high school students to get information about Vietnamese universities' admission. This report is the documentation for our website including three main parts: software's requirement, analysis and design.

Acknowledgment

We would like to express our special thanks to Dr. Truong Ninh Thuan who gave us the enthusiastic instruction and support.

We would like to thank Team OOAD K55CA and K56CA for their reports that we used as references and sample documents.

Table Of Contents

1. Requirements

1.1. Problem statement

There are millions of high school students constantly taking university entrance examination every year in Vietnam. Due to the great number of enquiries about universities' and colleges' information on majors, quality teaching, admission marks, etc from students and their parents, many kinds of handbooks and websites are being published with diverse and up-to-date data to meet this demand.

However, these sources are sometimes not reliable; the interface is not friendly and unscientific – which make users confused and worried. To tackle this issue, we want a useful website called VnUni for students to help them approach easily the necessary information about universities and colleges by a brand-new way, which would hopefully become a trustful connection between students and schools in near future.

This website provides an information search engine based on different filters: location, majors, recently admission marks, etc. Moreover, users could have an objective and comprehensive view with "Comparison and Ranking" function that compares and rates schools according to various criteria.

Each university or college is supplied with a separate account to log in the system so that they could modify their data very conveniently.

This website must have an administrator that manages all operations of the system: add/remove a school, add/remove a user, modify database. Anonymous users must not have the privilege to register a new school account. It is only provided by administrator.

Users could connect to VnUni via client devices such as laptops, PC, tablets or smartphones.

1.2. Glossary

Brief Description

The glossary contains the working definitions for all classes in the VnUni System.

Term

University

An education organization in Vietnam.

University Admission

An activity of universities to admit students.

Entrance exam

An exam of Vietnam universities for graduated high-school students who want to get higher education.

Faculty

A group of university departments concerned with a major division of knowledge.

Admission Mark

A level of mark for students to take to the university.

Admission Staff

An user of the university who has the permission to edit the information page of this university.

Filter Catalog

The fields catalog of filter system, helping user to filter out suitable universities quickly.

User

A person who view the website in their browser.

Admin

A person who develop, maintain the system and update database.

1.3. Supplementary Specification

Objectives

The purpose of this document is to define requirements of the VnUni System. This Supplementary Specification lists the requirements that are not readily captured in the use cases of the use-case model. The Supplementary Specifications and the use-case model together capture a complete set of requirements on the system.

Scope

This Supplementary Specification applies to the VnUni System which will be developed by the VnUni Team – Computer Science Department – University of Technology and Engineering, Vietnam National University.

The VnUni System will enable users to search and view the admission information of universities and colleges in Vietnam. This system also allows admission staffs to provide or modify the data of their schools.

This specification defines the non-functional requirements of the system; such as reliability, usability, performance, and supportability as well as functional requirements that are common across a number of use cases. (The functional requirements are defined in the Use Case Specifications.

References

Applicable references are:

- IBM Rational Software Documentation (Version 2004).
- Document of K55CA OOAD project - Restaurant Management System.
- Document of K56CA OOAD project - YouTube Video Player.

Functionality

- Multiple users must be able to perform their works concurrently.
- All functionality shall be available remotely through an internet connection. This may require applications or controllers running on the remote computers.

Usability

This section lists all of those requirements that relate to, or affect, the usability of the system.

Browser Compliance

The user-interface should run smoothly on all browsers, for instant: Chrome, Firefox, and Internet Explorer, etc.

Design for Ease-of-Use

The user interface of the VnUni System shall be designed for ease-of-use and shall be appropriate for a computer-literate user community with no additional training on the System. The user interface has to be nice and clear.

Reliability

The VnUni System shall be available 24 hours a day, 7 days a week. There shall be no more than 4% down time.

Performance

The performance characteristics of the system are outlined in this section.

Simultaneous Users

The system shall support up to 10000 simultaneous users against the central database at any given time, and up to 5000 simultaneous users against the local servers at any one time.

Database Access Response Time

The system shall provide access to the VnUni database with no more than a 5 second latency.

Supportability

None.

Security

The system must prevent anonymous users to modify the database. Almost changes of the system databases can only be done by managers. Require confirm account before submit the changes.

Design Constraints

This system will provide only for any browsers supporting JavaScript.

1.4. Use-case model

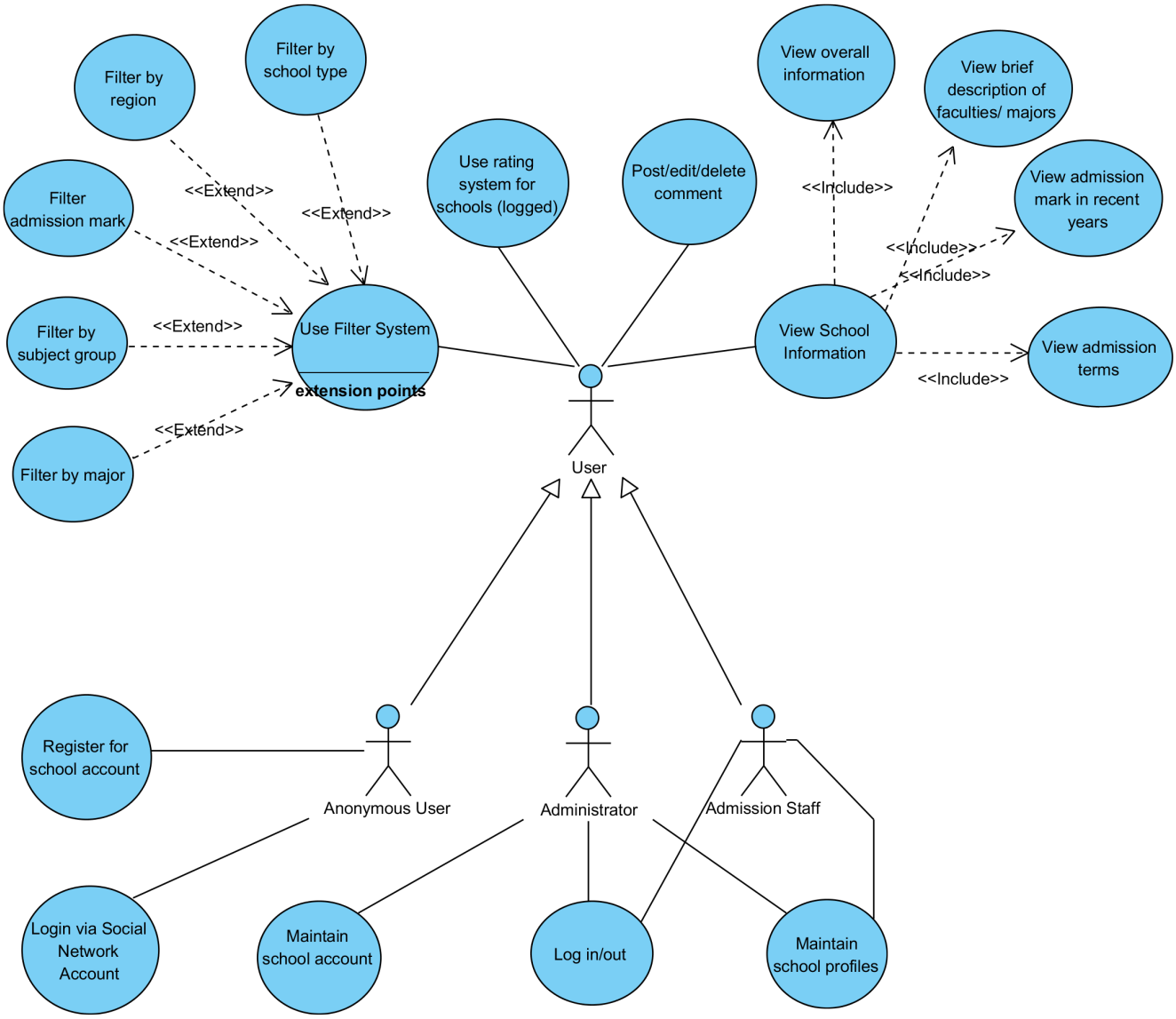


Figure 1 - Use-case Model

1.4.1. Use-case Specification - <Login>

Brief Description

This use case describes how a user logs into the VnUni System.

Flow of Events

Basic Flow

- The system validates the actor's password and logs him/her into the system.
- The system displays the Main Form and the use case ends.

Invalid Name/Password

If in the basic flow the system cannot find the name or the password is invalid, an error message is displayed. The actor can type in a new name or password or choose to cancel the operation, at which point the use case ends.

Associations

Actor

The actor starting this use case is:

- Administrator
- Admission Staff

Associations to Other Use Cases

None.

Associations from Other Use Cases

None.

Pre-Conditions

None.

Special Requirements

None.

1.4.2. Use-case Specification - <Maintain School Profile>

Brief Description

This use case allows the Administrators and Admission staffs to maintain school information including “add and modify school information” in VnUni database system.

Flow of Events

Basic Flow - Add data to a blank university record

- The administrator/admission staff selects “Add new information”.
- The system displays a blank university information form.
- The administrator/admission enters the following information for the university: full name, date of established, short introduction, etc.
- The system validates data to insure the proper data. If data is valid, the system updates new information to the blank university record in the database system.

Modify information of a university

- The administrator/admission staff selects "Modify information”.
- The system displays an editable form with old information.
- The administrator/admission staff types in the field he/she wishes to modify.
- When changes are complete, the administrator/admission staff selects “Save”.
- The database system updates the university information.

Associations

Actor

The actor starting this use case is:

- Administrator
- Admission Staff

Associations to Other Use Cases

None.

Associations from Other Use Cases

None.

Pre-Conditions

Before this use case begins the administrator or admission staff has logged onto the system.

Special Requirements

None.

1.4.3. Use-case Specification - <Use Filter System>

Brief Description

This use case describes how a user use the filter system to search universities.

Flow of Events

Basic Flow

- The system displays a list of options in filter system.
- The user selects (or not) filter options and search.
- The system retrieves a list of all universities that were satisfied the filter options and displays in three lists which were represented in proportion to three regions of Vietnam.
- For each university on lists, the system will displays some information of each university such as admission marks and number of admitted students per year when user moves the mouse on.

No Universities Found

If in the basic flow, the system cannot find the university which satisfied user's options in the filter, at which point the use case ends.

Associations

Actor

The actor starting this use case is:

- User

Associations to Other Use Cases

None.

Associations from Other Use Cases

None.

Pre-Conditions

None.

Special Requirements

None.

1.4.4. Use-case Specification - <View School Information>

Brief Description

This use case allows any users to see the information of a school. The information contains: full name of school, date of established, short introduction, etc.

Flow of events

Basic Flow – view the information of a specific school.

- The user apply the search or filter system to find schools that appropriate with him.
- The user select the desire school that appears from the result list.
- The page viewing information of the school appears and the user start discovering school information.

Associations

Actors

The actor starting this use case is:

- Any user.

Associations to Other Use Cases

None.

Associations from Other Use Cases

None.

Pre-Conditions

None.

Special Requirements

None.

1.4.5. Use-case Specification - <Register School Account>

Brief Description

This use-case allows anonymous users to send a request to administrator to get an account for him/her school. After receiving the request, admin will examine the information, accept or reject the request and send the response to users via email.

Flow of events

Basic Flow - Register school account

- Click the "Register account" button, the registration form will appear.
- Fill in the registration form.
- Click "Send" button to send the request to the administrator.

Associations

Actors

The actor starting this use case is:

- Anonymous users

Associations to Other Use Cases

None.

Associations from Other Use Cases

None.

Pre-Conditions

None.

Special Requirements

None.

1.4.6. Use-case Specification - <Maintain School Account>

Brief Description

This use case allows the Administrators to add or remove accounts for schools.

Flow of events

Basic Flow – Add account for a school.

- The administrator receivers the request of making a school account from an admission staff of the school.
- The administrator check the request and consider to make a new school account.
- The administrator selects "Add a new school account".
- The system displays a blank form for making new school account. The administrator enters the following information for the new account: school id, school name, account username, account password, etc.
- The administrator send the new account information to the person who sent the request. The information include: account nickname, account password, etc.
- The admission staff receive the account information and using it to work with VnUni system with specific privileges.

Remove an admission staff account

- The administrator receivers the request of removing a school account from an admission staff of the school.
- School already has an account
- The administrator receivers the request of making a school account from an admission staff of the school.
- The administrator check the request and create an account for valid staff.

Associations

Actors

The actor starting this use case is:

- Administrators.

Actors also involved in this use case:

- Admission staffs.

Associations to Other Use Cases

None.

Associations from Other Use Cases
None.

Pre-Conditions
Before this use case begins the administrator has logged onto the system.

Special Requirements
None.

1.4.7. Use-case Specification - <Use Rating System>

Brief Description

This use case allows the logged users using rating system to rate a specific school.

Flow of events

Basic Flow – Using rating system in result list

- The logged user apply the search or filter system to find schools that appropriate with him.
- The result list appears and the logged user can rate any school presenting in that list.
- Using rating system in school information web page
- The logged user apply the search or filter system to find schools that appropriate with him.
- The logged user select the desire school that appears from the result list.
- The page viewing information of the school appears and the logged user start discovering school information.
- While discovering school information, logged user can anytime use rating system to rate the present school.

Associations

Actors

The actor starting this use case is:

- Any logged users.

Associations to Other Use Cases

None.

Associations from Other Use Cases

None.

Pre-Conditions

Before this use case begins the user has logged onto the system.

Special Requirements

None.

1.4.8. Use-case Specification - <Post/Edit/Delete Comment>

Brief Description

This use-case allows the anonymous users and admin and admission staff to post, edit or remove comments for a articles or a schools in this website. This include: post a comment, edit a comment, delete a comment.

Flow of events

Post a comment

- Enter text into the comment-box
- Push "Enter" key on the keyboard or "Comment" button on the left of comment-box to post comment.

Edit a comment

- Click "Edit" action bellow the comment, a box-form appear for editing.
- Enter the text into the edit box .
- Click "Save" button to save the comment

Delete a comment

- Click the "Delete" action bellow the comment, beside "Edit" action, a alert dialog appear to verify the action.
- Click "Confirm" button to confirm delete action.

Associations

Actors

The actor starting this use case is:

- Administrators or Admission staffs or Logged-users

Associations to Other Use Cases

None.

Associations from Other Use Cases

Login via Social Network Account.
Login/logout.

Pre-Conditions

Before this use case begins the users have logged onto the system.

Special Requirements

None.

2. Analysis

2.1. Architectural Analysis

2.1.1. Logical View

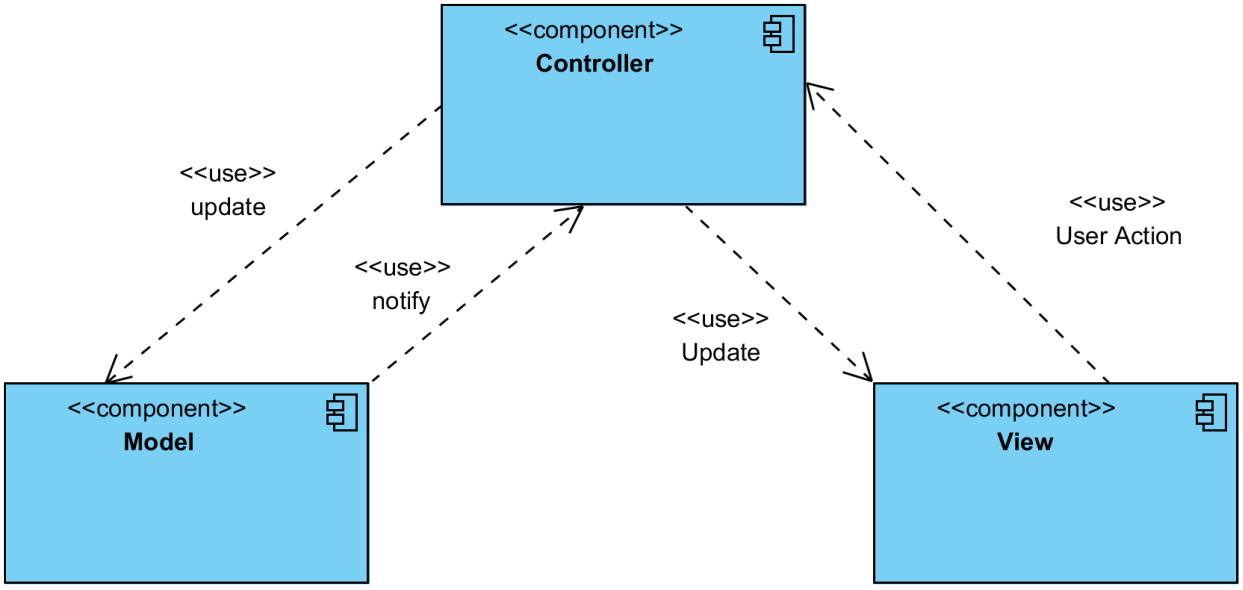


Figure 2 - Logical View

Definition

- *Model*: represents the underlying, logical structure of data in a software application and the high-level class associated with it. This object model does not contain any information about the user interface.
- *View*: This is a collection of classes representing the elements in the user interface (all of the things the user can see and respond to on the screen, such as buttons, display boxes, and so forth)
- *Controller*: represents the classes connecting the model and the view, and is used to communicate between classes in the model and view.

2.1.2. Key Abstractions

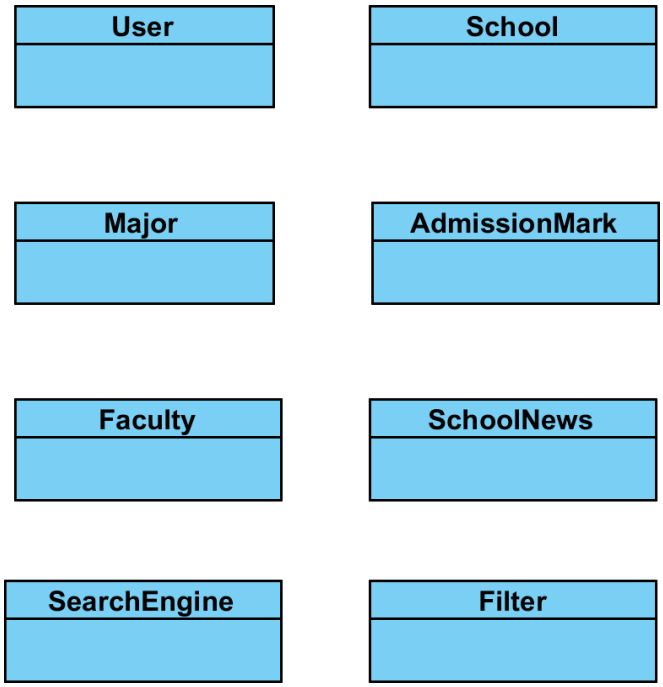


Figure 3 - Key Abstractions

Definition

- *User:* Is an account of the system, may be an normal user, school user or administrator.
- *School*: A record that contains all information about school information.
- *Major:* A record that contains all information about a school major.
- *Division:* A record that contains all information about a division in university entrance exam.
- *AdmissionMark:* A record that contains admission mark of specific year of a school major.
- *SchoolNews* : The article was written by the school user or administrator, contains contents about useful school news.
- *SearchEngine:* The component that is responsible for searching all information about schools' information.
- *Filter:* The component that provides specific options for user to quickly search appropriate schools.

2.2. Use-case Analysis

2.2.1. Interaction diagrams

2.2.1.1. Login

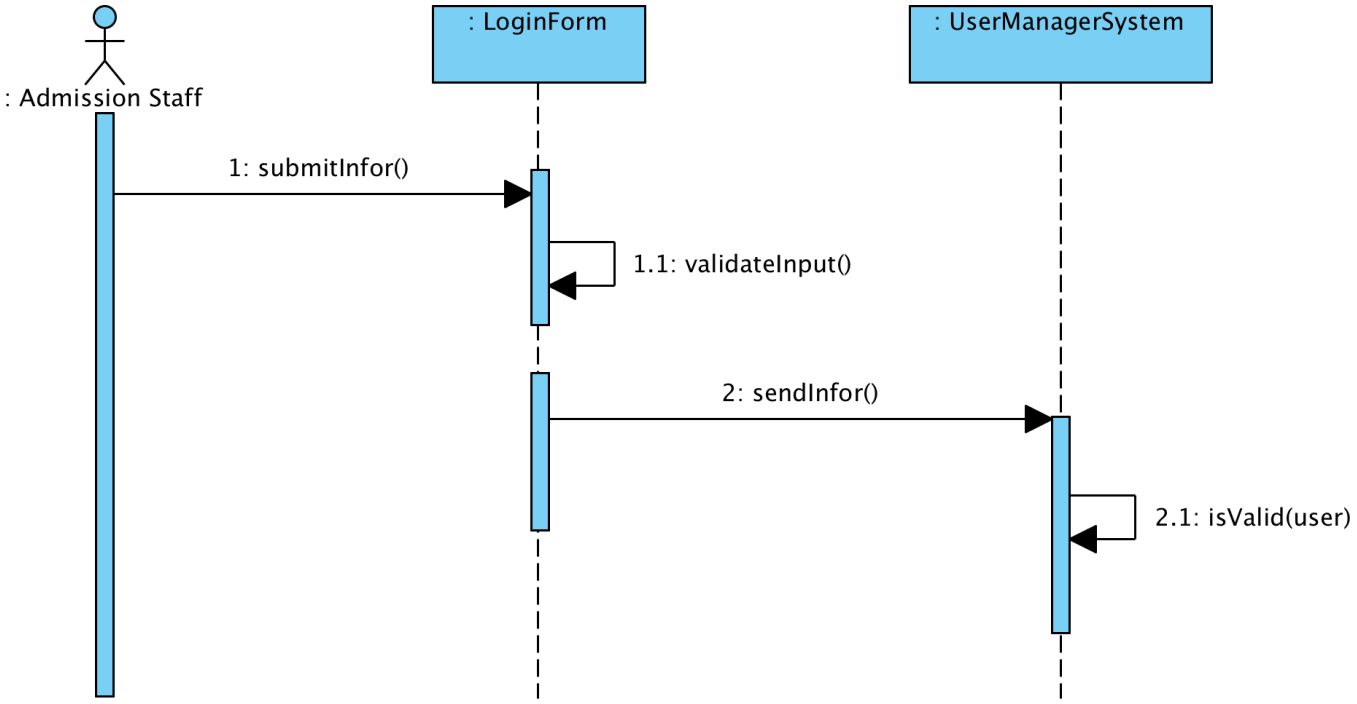


Figure 4 - Login Interaction Diagram - Basic Flow

2.2.1.2. Maintain School Profile

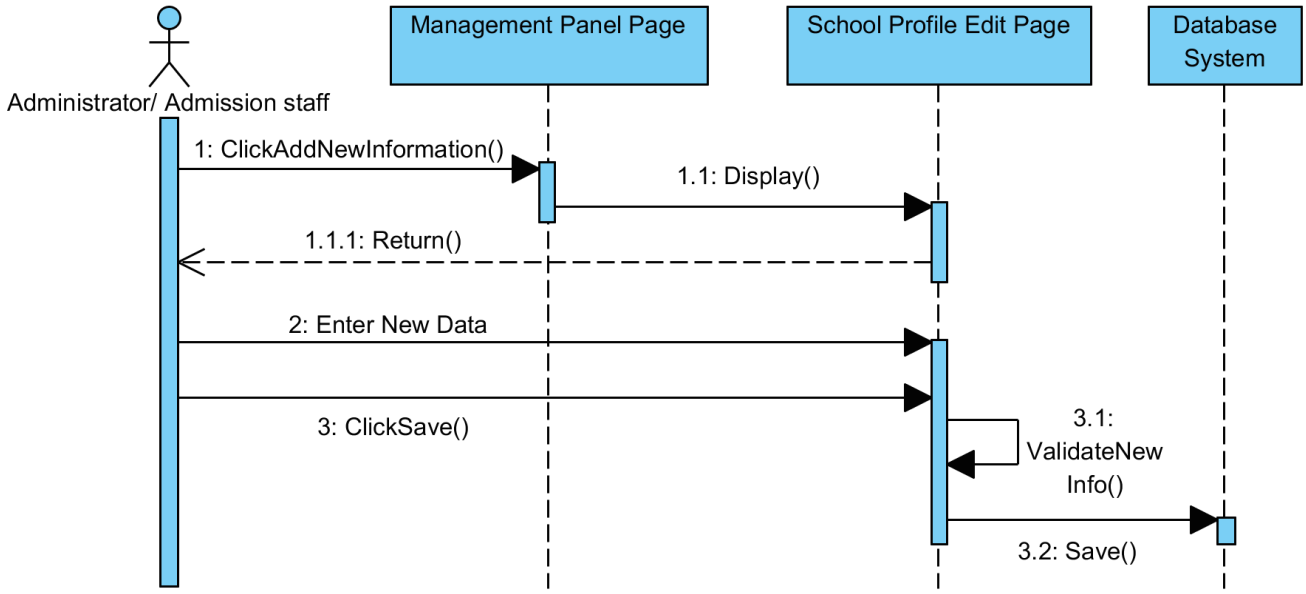


Figure 5 - Maintain School Profile Interaction Diagram - Basic Flow

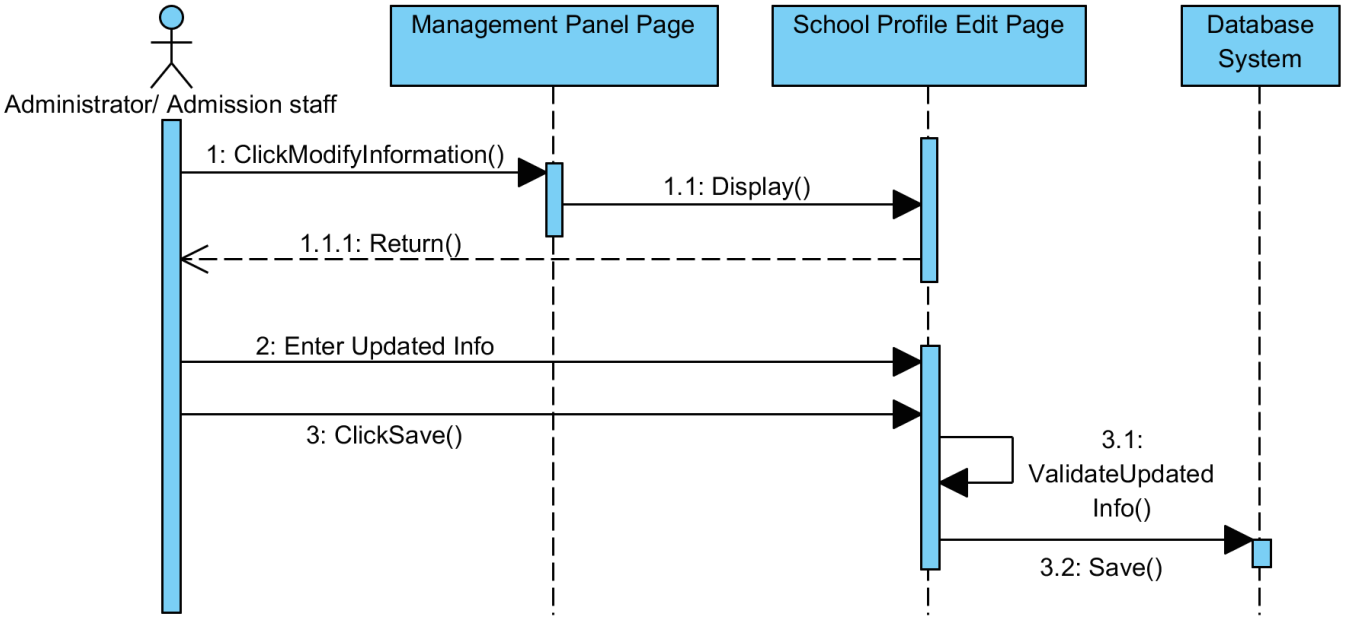


Figure 6 - Maintain School Profile Interaction Diagram - Modify School Profile Flow

2.2.1.3. Use Filter System

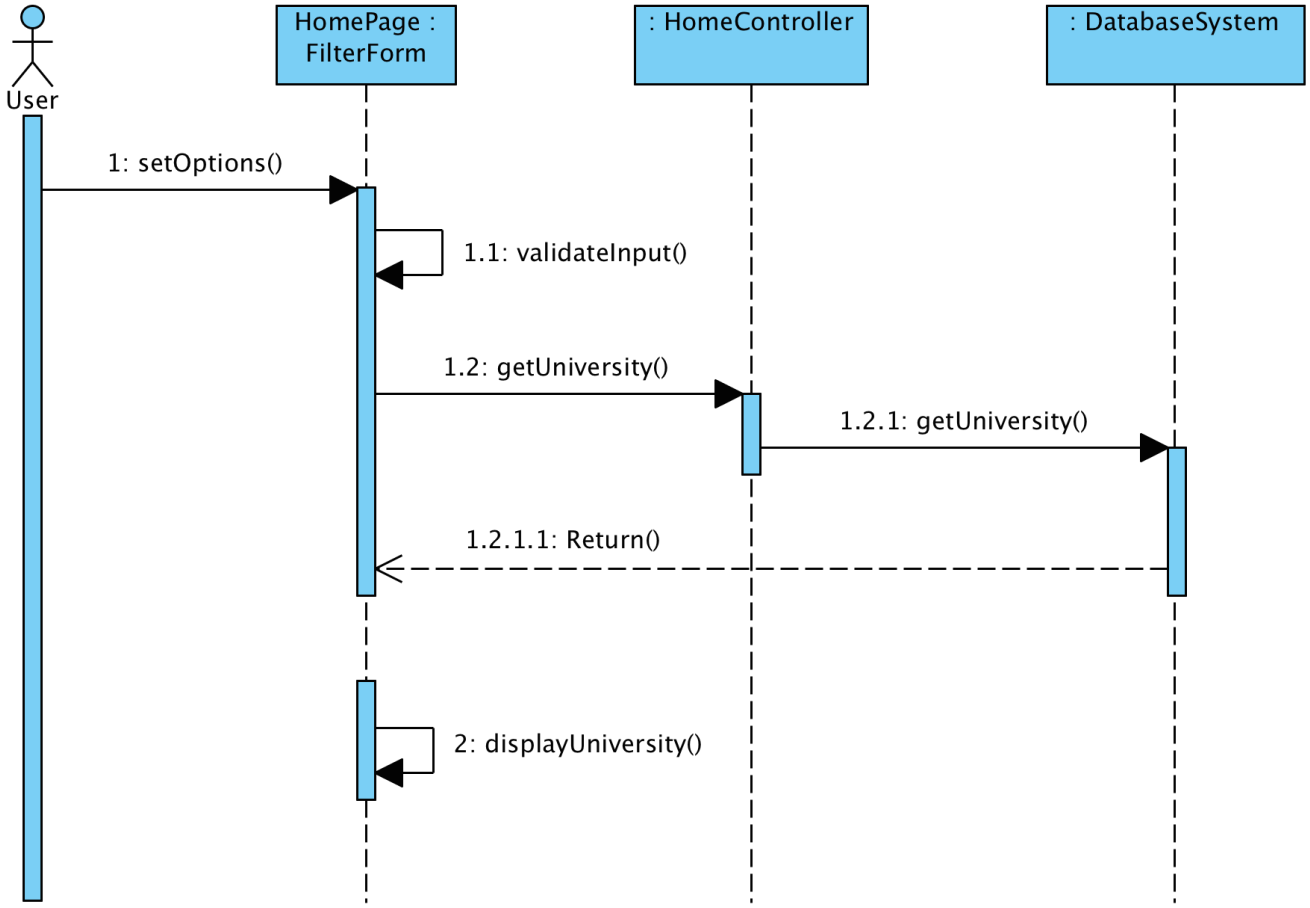


Figure 7 - Use Filter System Interaction Diagram - Basic Flow

2.2.1.4. View School Information

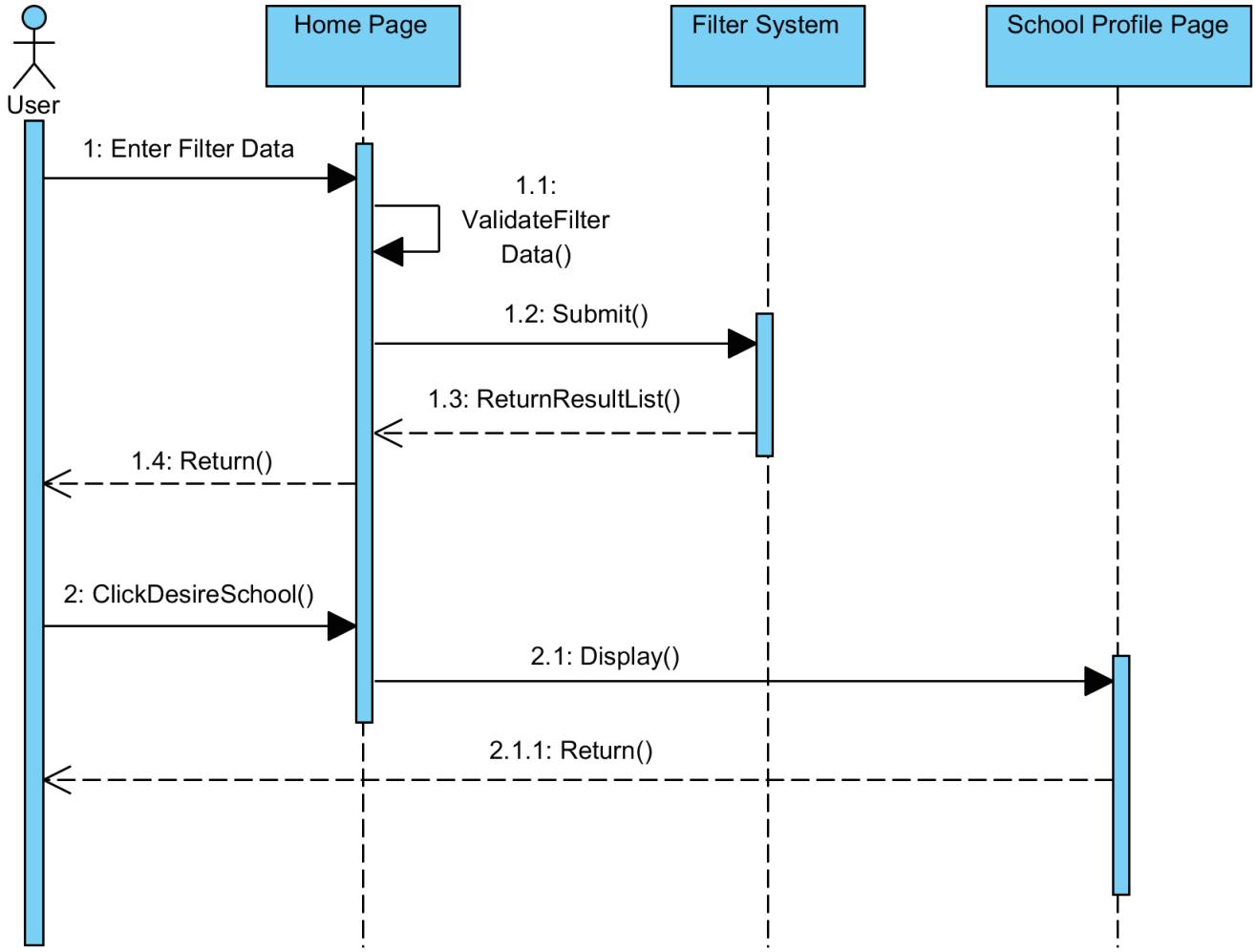


Figure 8 - View School Information Interaction Diagram - Basic Flow

2.2.1.5. Register School Account

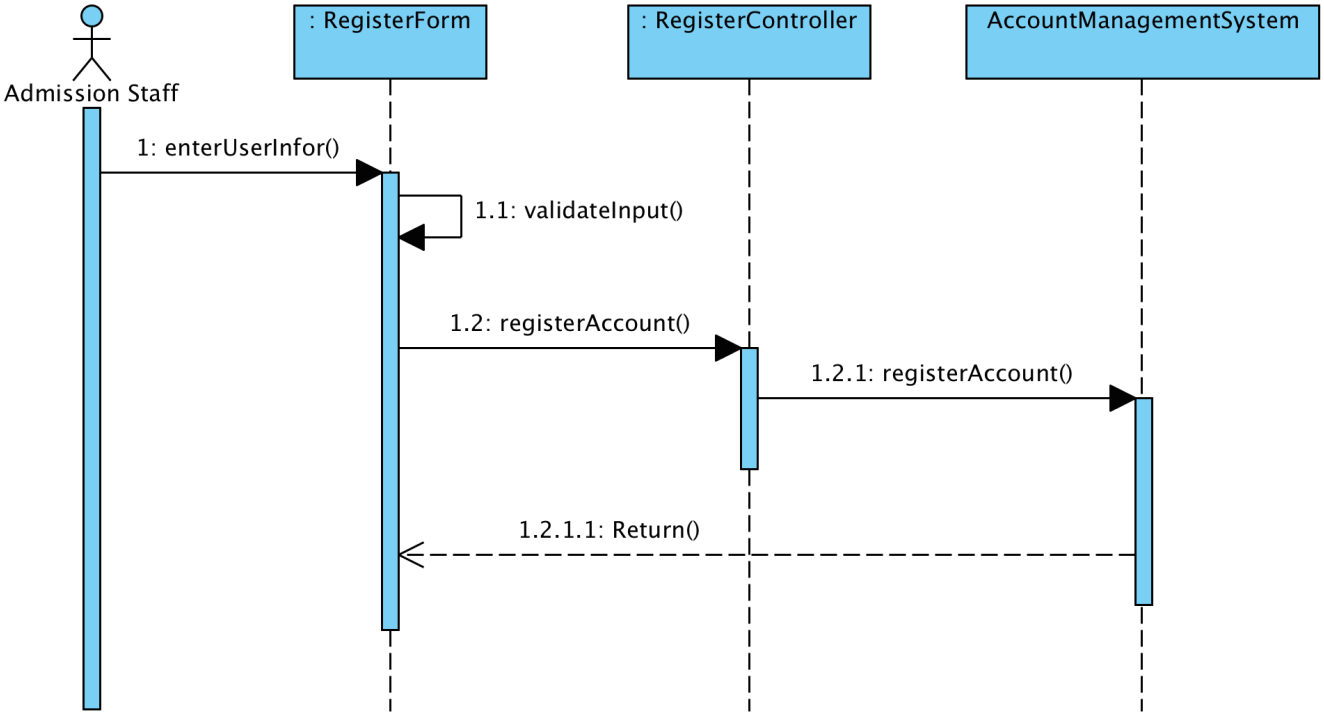


Figure 9 - Register School Account Interaction Diagram - Basic Flow

2.2.1.6. Maintain School Account

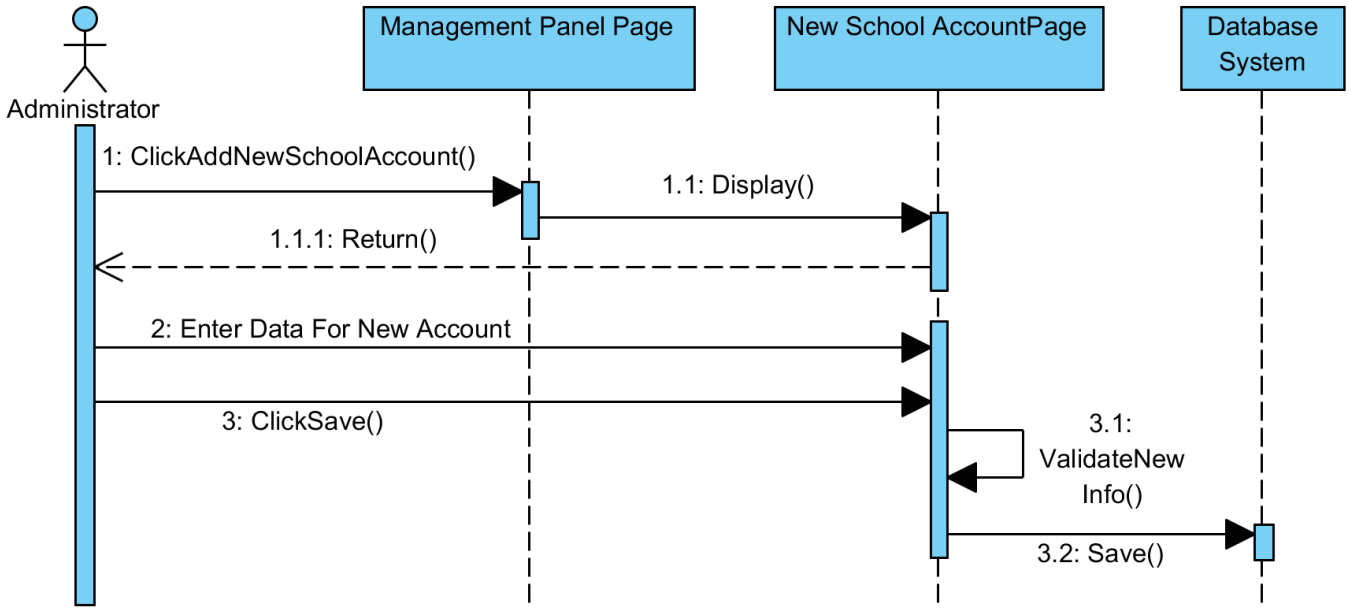


Figure 10 - Maintain School Account Interaction Diagram - Basic Flow

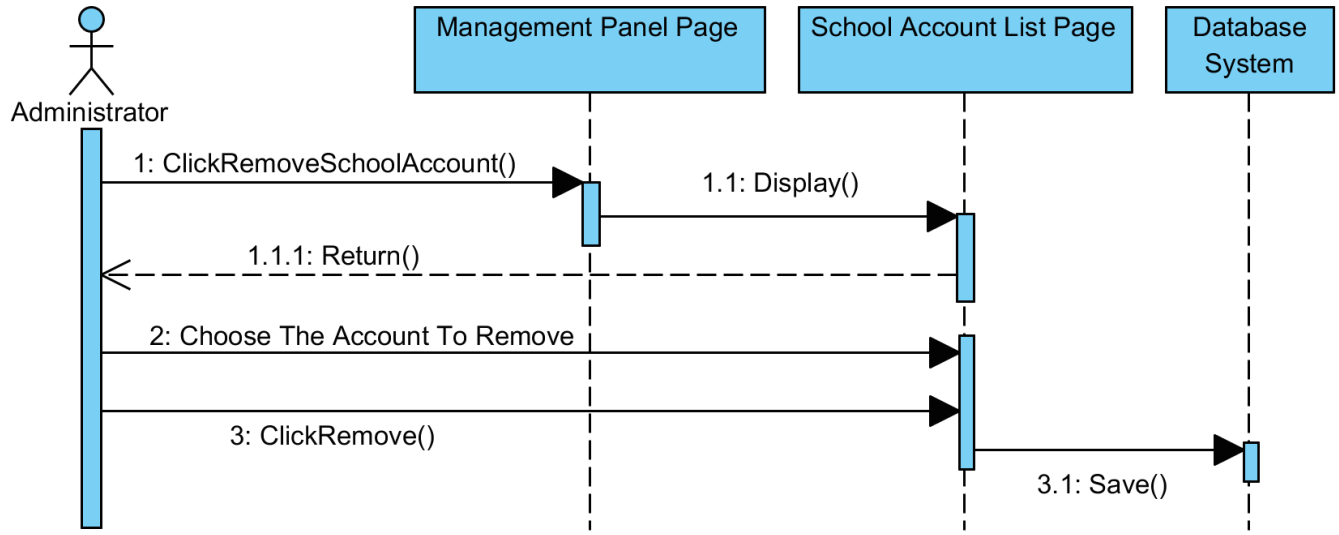


Figure 11 - Maintain School Account Interaction Diagram - Remove Account Flow

2.2.1.7. Use Rating System

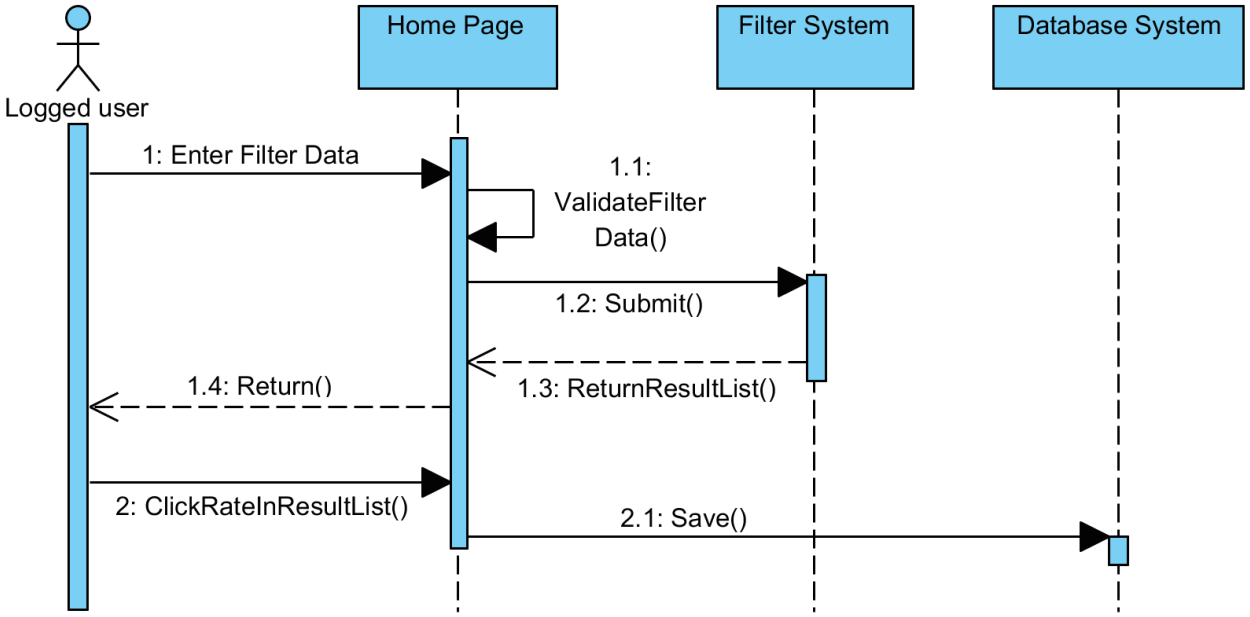


Figure 12 - Use Rating System Interaction Diagram -
Rating in University page Flow

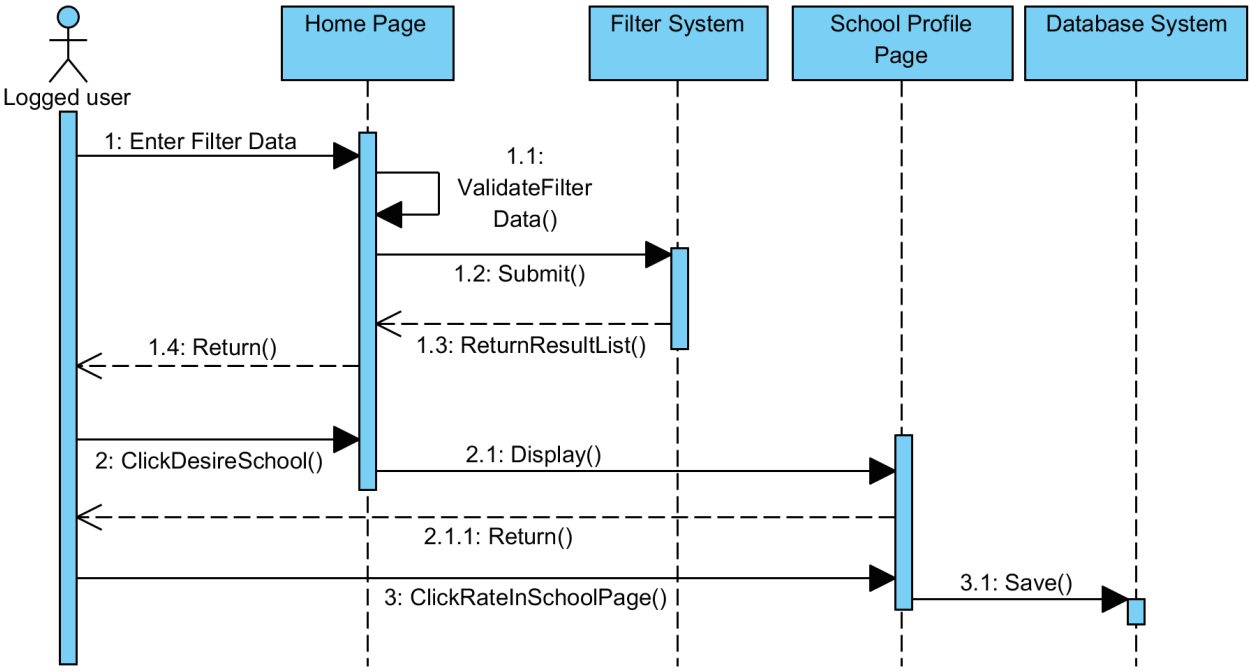


Figure 13 - Use Rating System Interaction Diagram -
Rating in Home page Flow

2.2.1.8. Post/Edit/Delete Comment

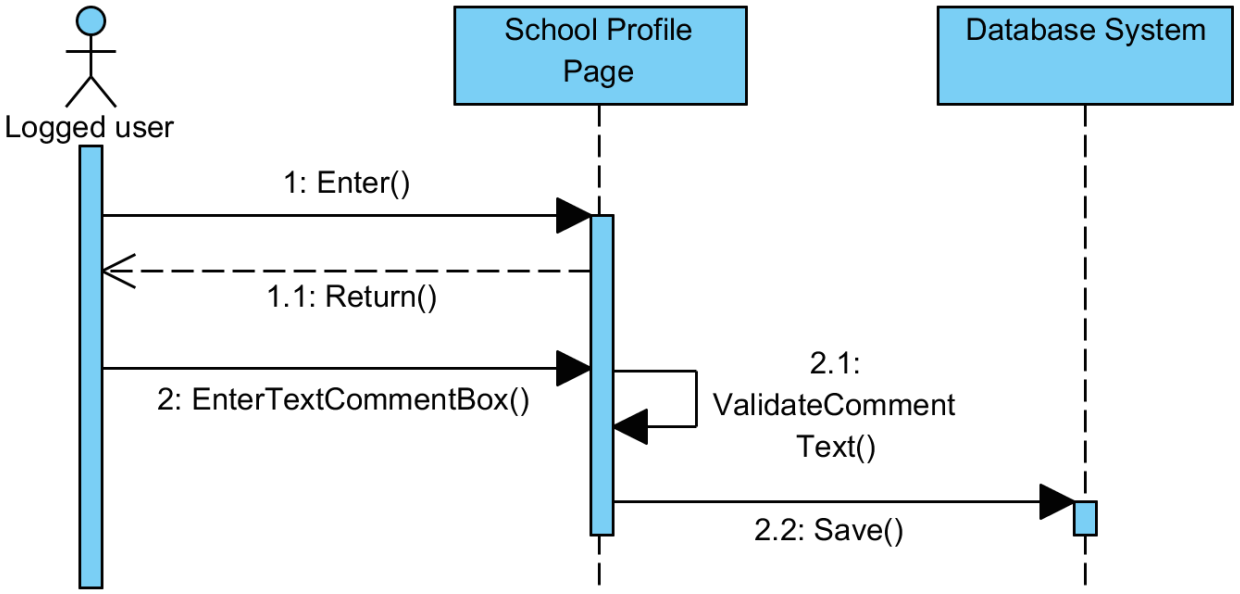


Figure 14 - Post Comment Interaction Diagram

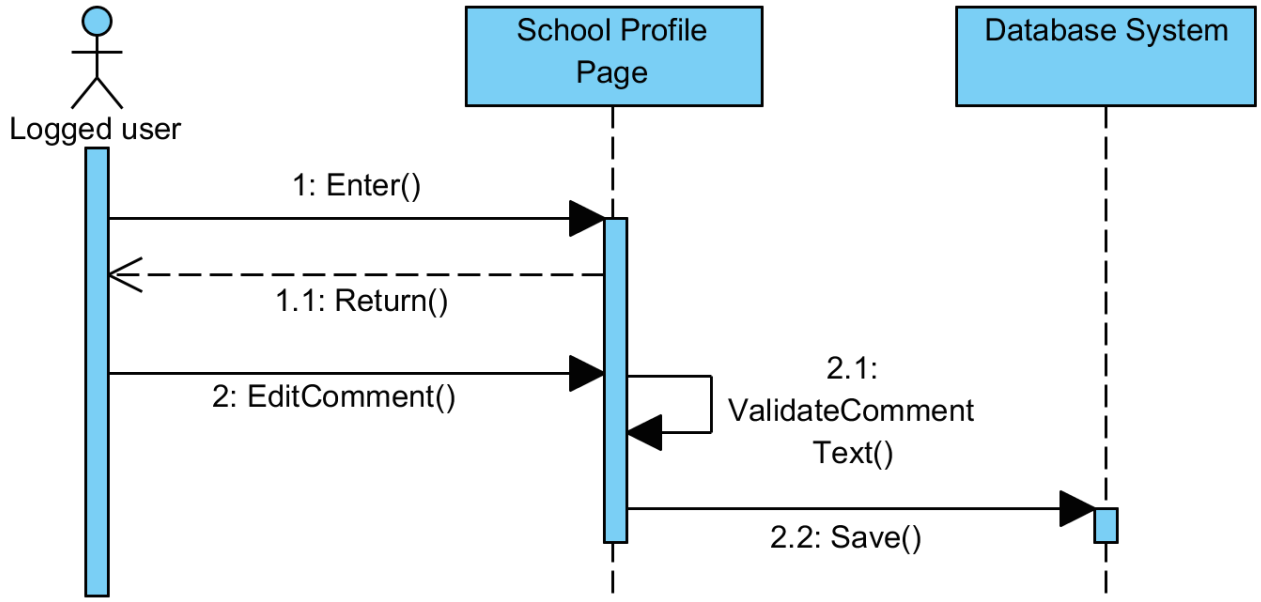


Figure 15 - Edit Comment Interaction Diagram

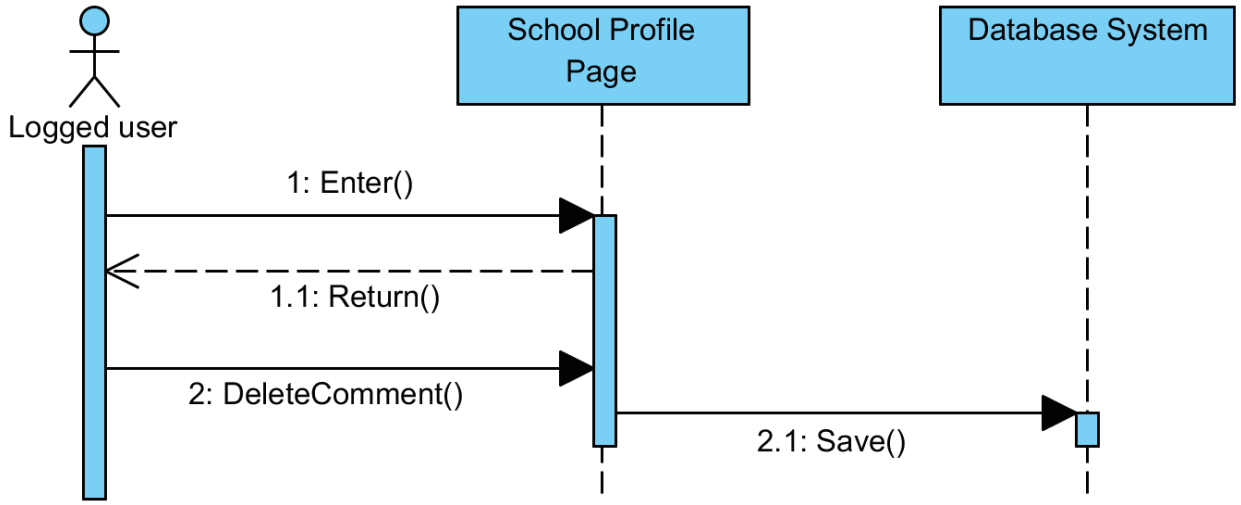


Figure 16 - Delete Comment Interaction Diagram

2.2.2. Use-case Realization View of Participating Class (VOPCs)

2.2.2.1. Login

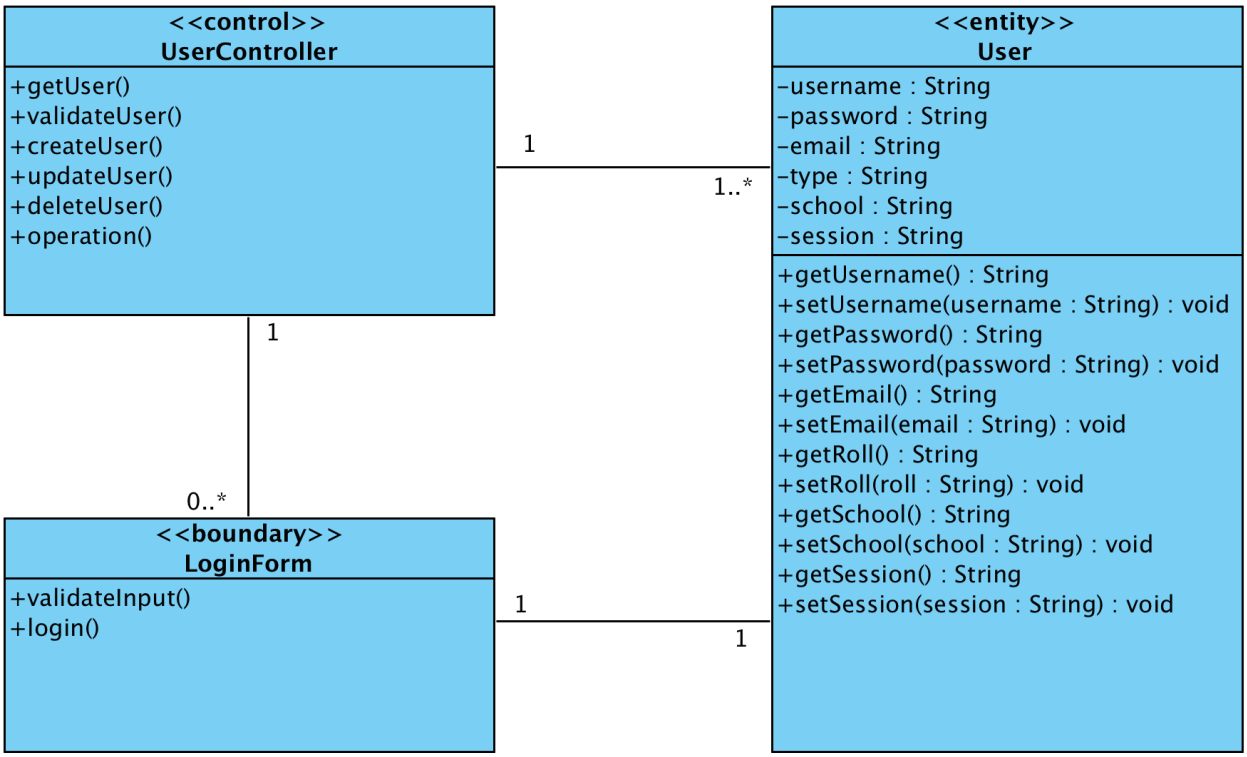


Figure 17 - Login VOPC Diagram

2.2.2.2. Post/Edit/Delete Comment

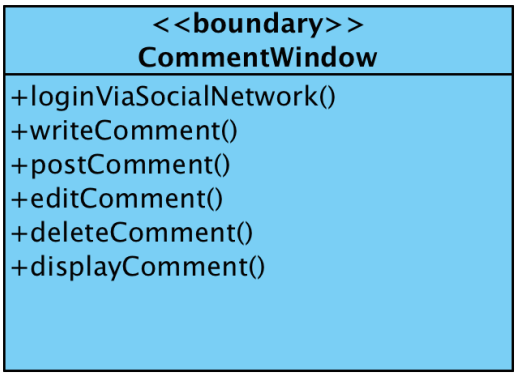


Figure 18 - Post/Edit/Delete VOPC Diagram

2.2.2.3. View School Information

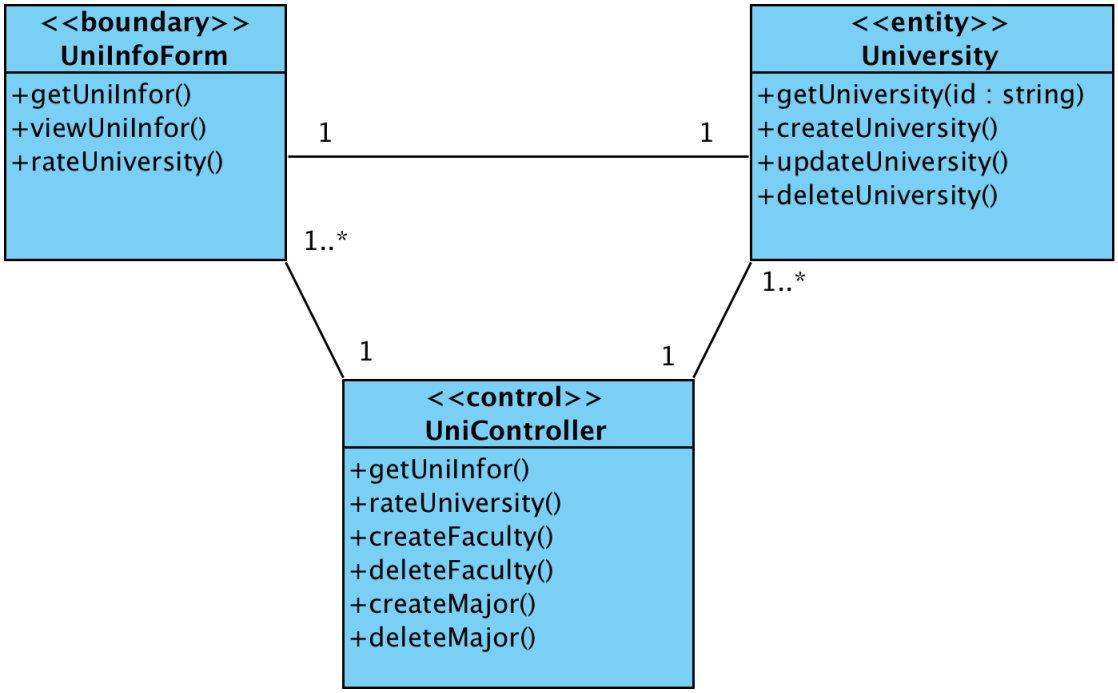


Figure 19 - View School Information VOPC Diagram

2.2.2.4. Maintain School Account

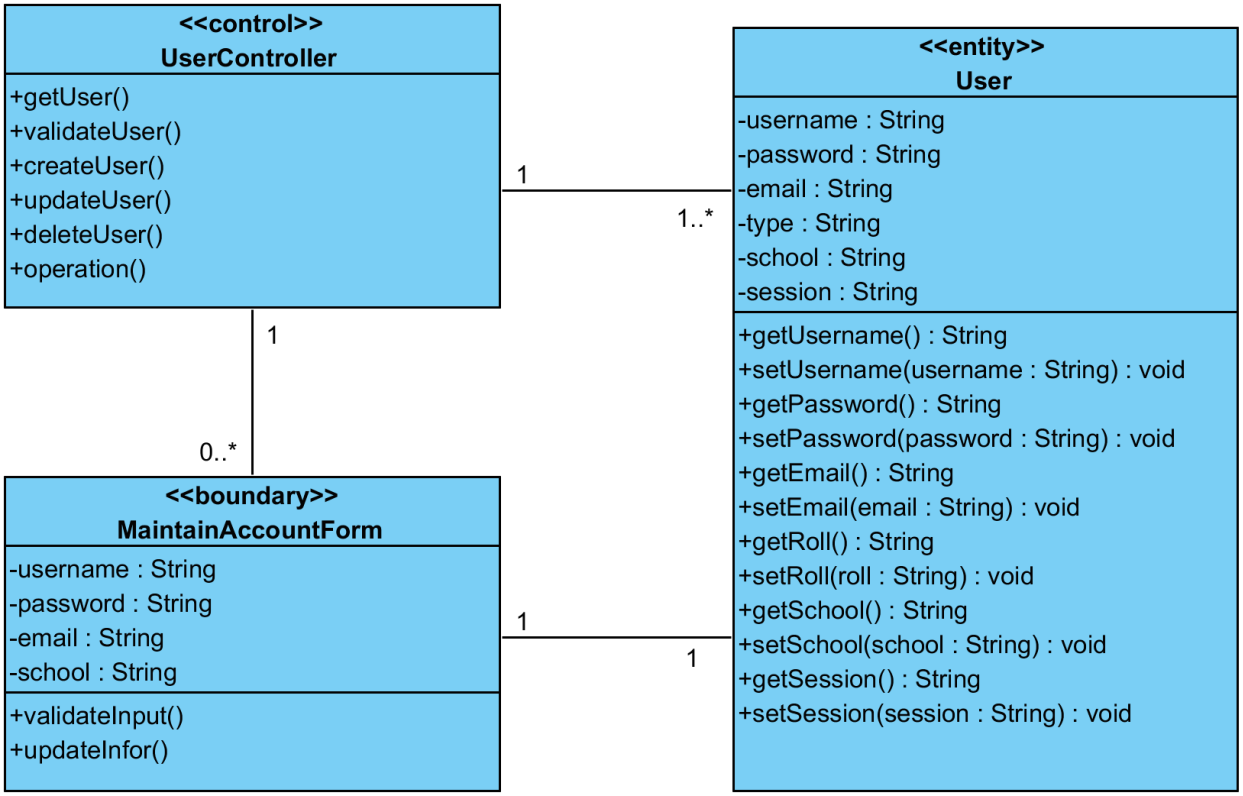


Figure 20 - Maintain School Account VOPC Diagram

2.2.2.5. Maintain School Profile

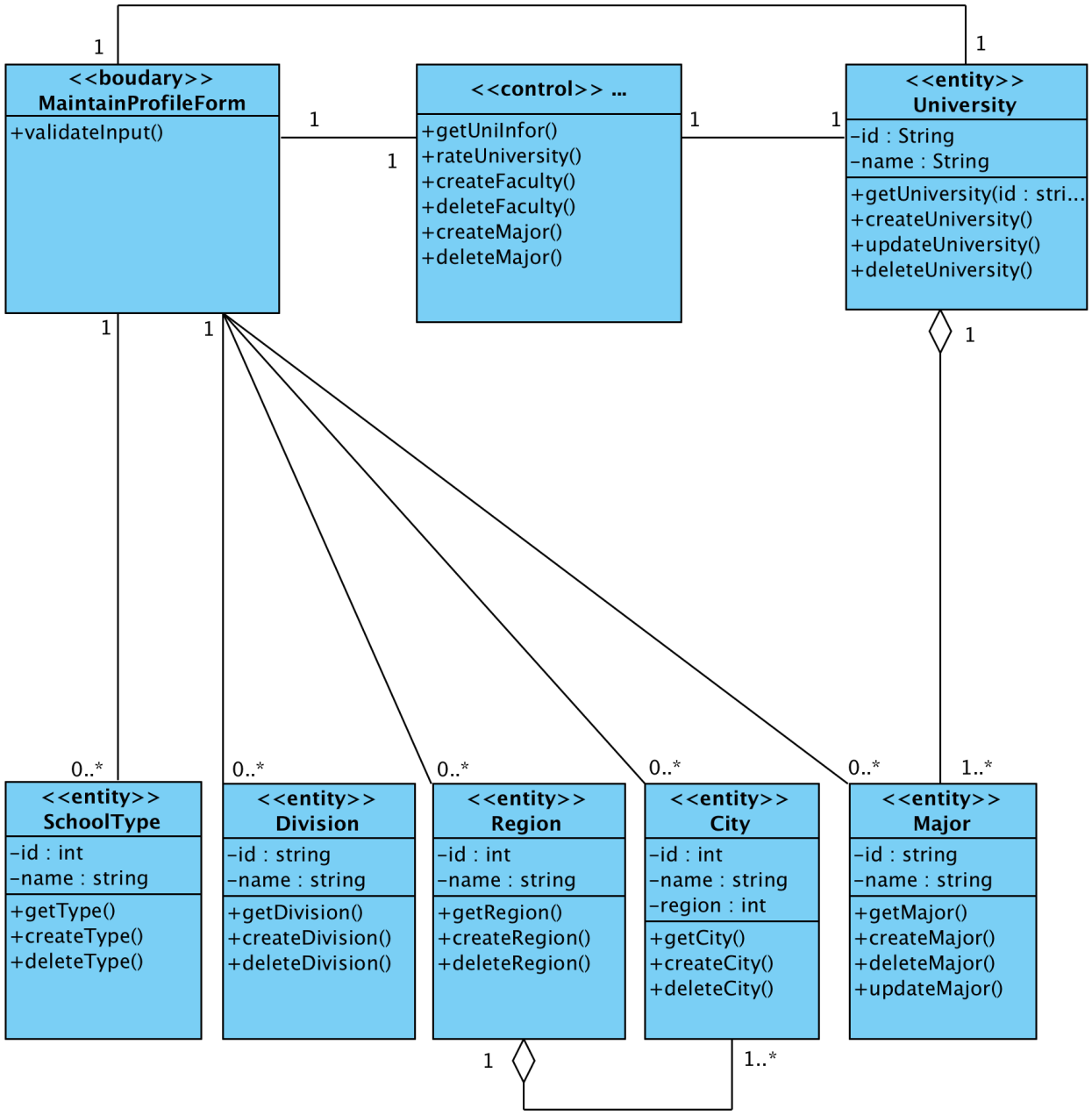


Figure 21 - Maintain School Profile VOPC Diagram

2.2.2.6. Use Filter System

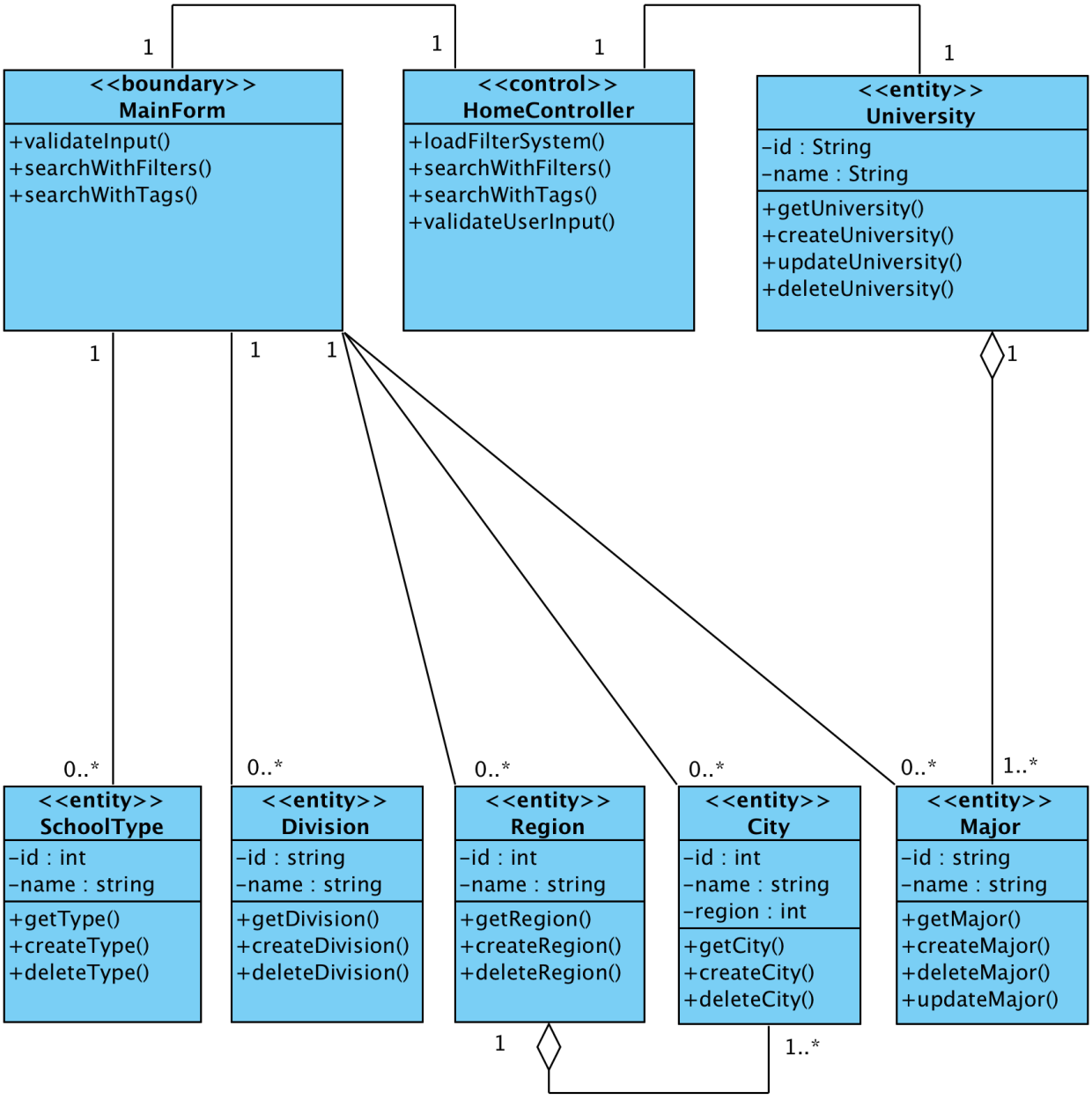


Figure 22 - Use Filter System VOPC

2.2.2.7. Use Rating System

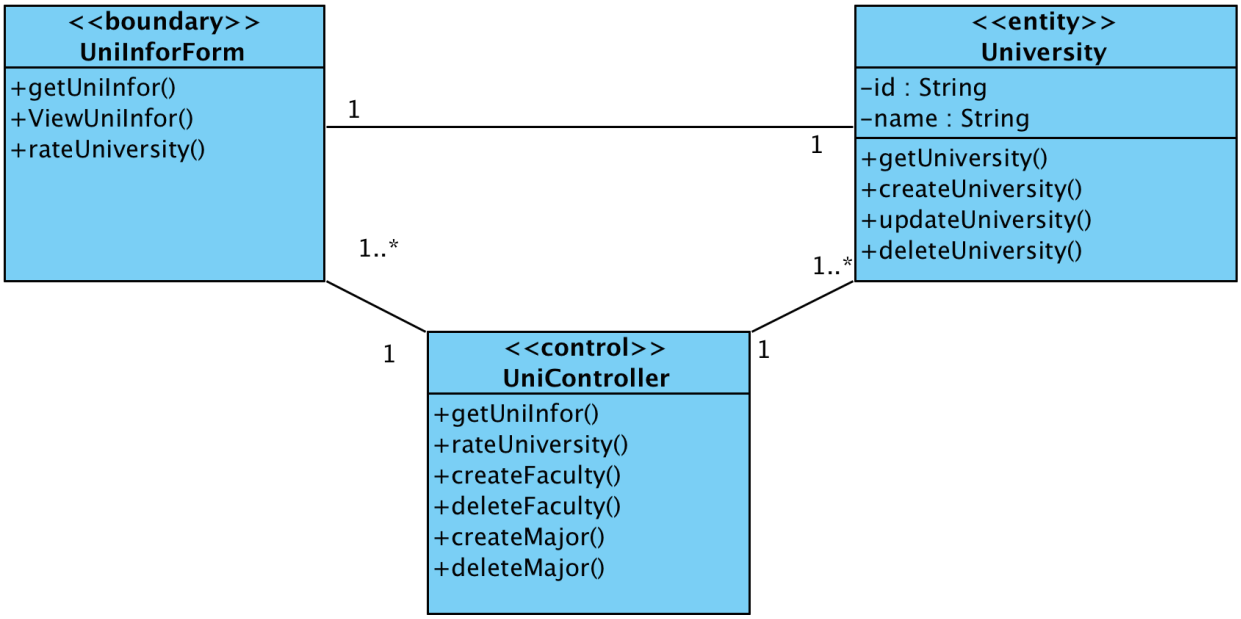


Figure 23 - Use Rating System VOPC

2.2.2.8. Register School Account

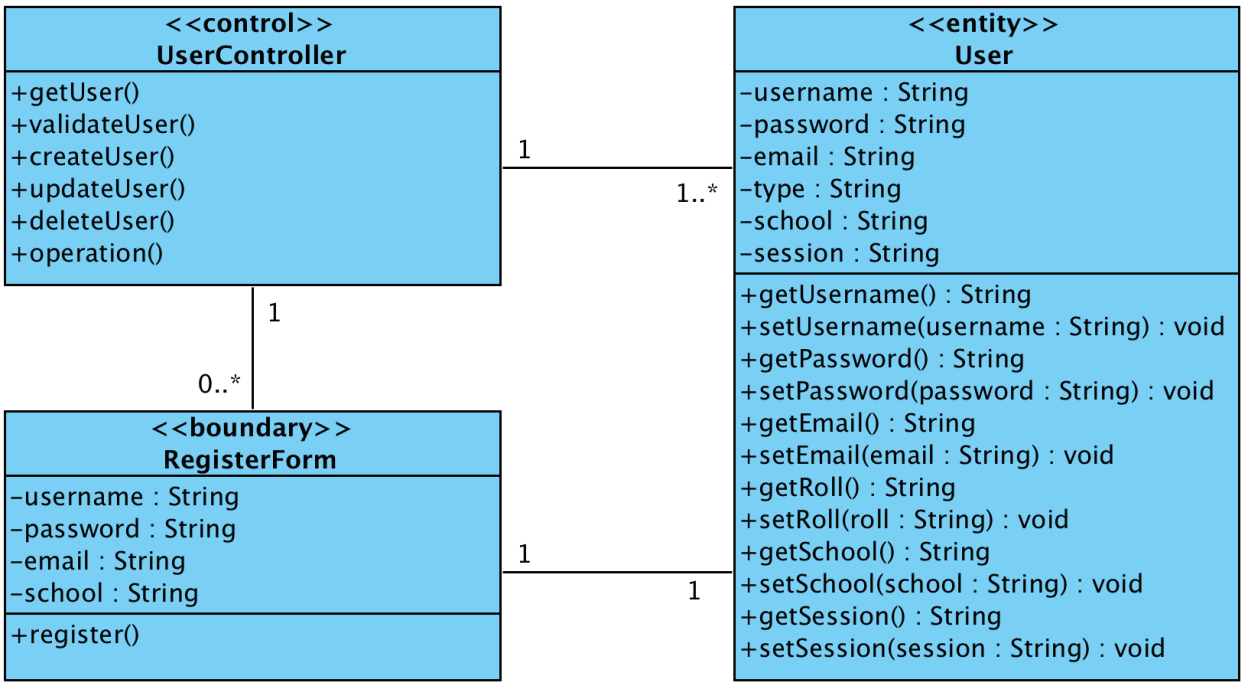


Figure 24 - Register School Account VOPC

2.2.3. Analysis mechanism

Analysis Class	**Analysis Mechanism(s)**
LoginForm	Error Management
User	Persistence, Security
UserController	Error Management, Resource Management, Security
CommentWindow	Error Management, Security
UniInforForm	Error Management
University	Persistence, Security
UniController	Error Management, Resource Management, Security
MaintainAccountForm	Error Management, Security
MaintainProfileForm	Error Management, Security
SchoolType	Persistence, Security
Division	Persistence, Security
Region	Persistence, Security
City	Persistence, Security
Major	Persistence, Security
MainForm	Error Management, Security
HomeController	Error Management, Resource Management, Security
RegisterForm	Error Management, Reporting

3. Design

3.1. Identify Design Elements

3.1.1. Subsystem Context Diagram

3.1.1.1. Maintain University Subsystem

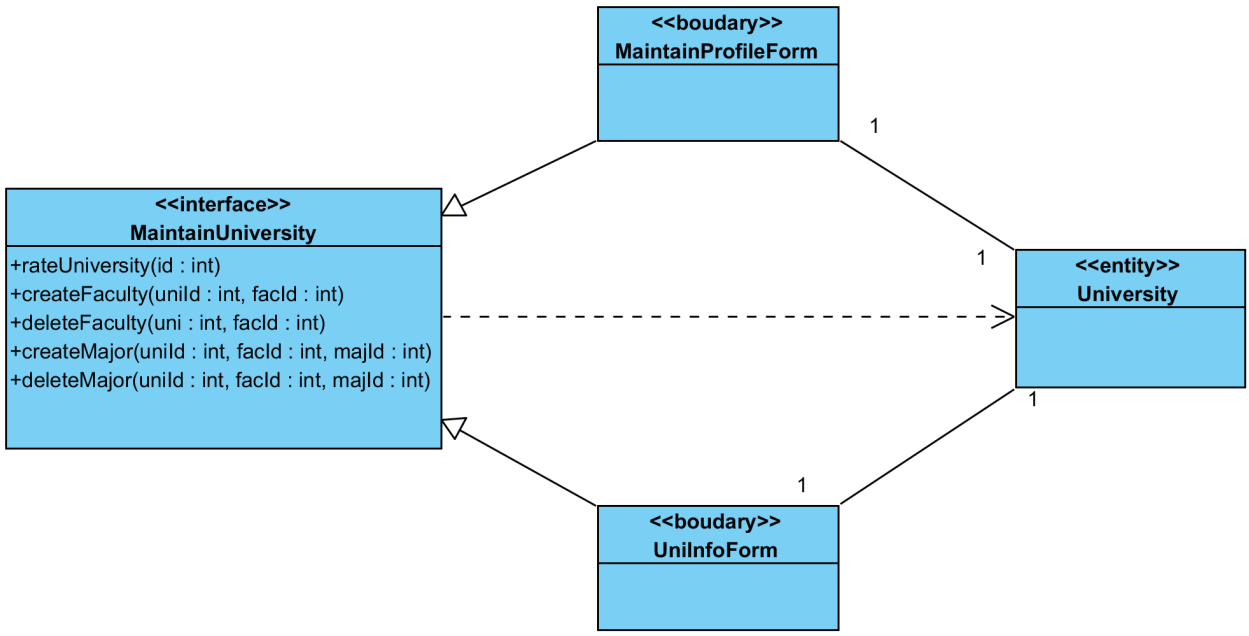


Figure 25 - Maintain University Subsystem

Subsystem Interface Descriptions

MaintainUniversity: encapsulates the action of changing school Faculty, Major and rating.

rateUniversity: change the school rating with given school id.

createFaculty: make a new faculty for a school with given id.

deleteFaculty: remove a faculty of a school with given id.

createMajor: make a new major for a faculty of a school with given id.

deleteMajor: remove a major of a faculty of a school with given id.

3.1.1.2. Maintain Filter Subsystem

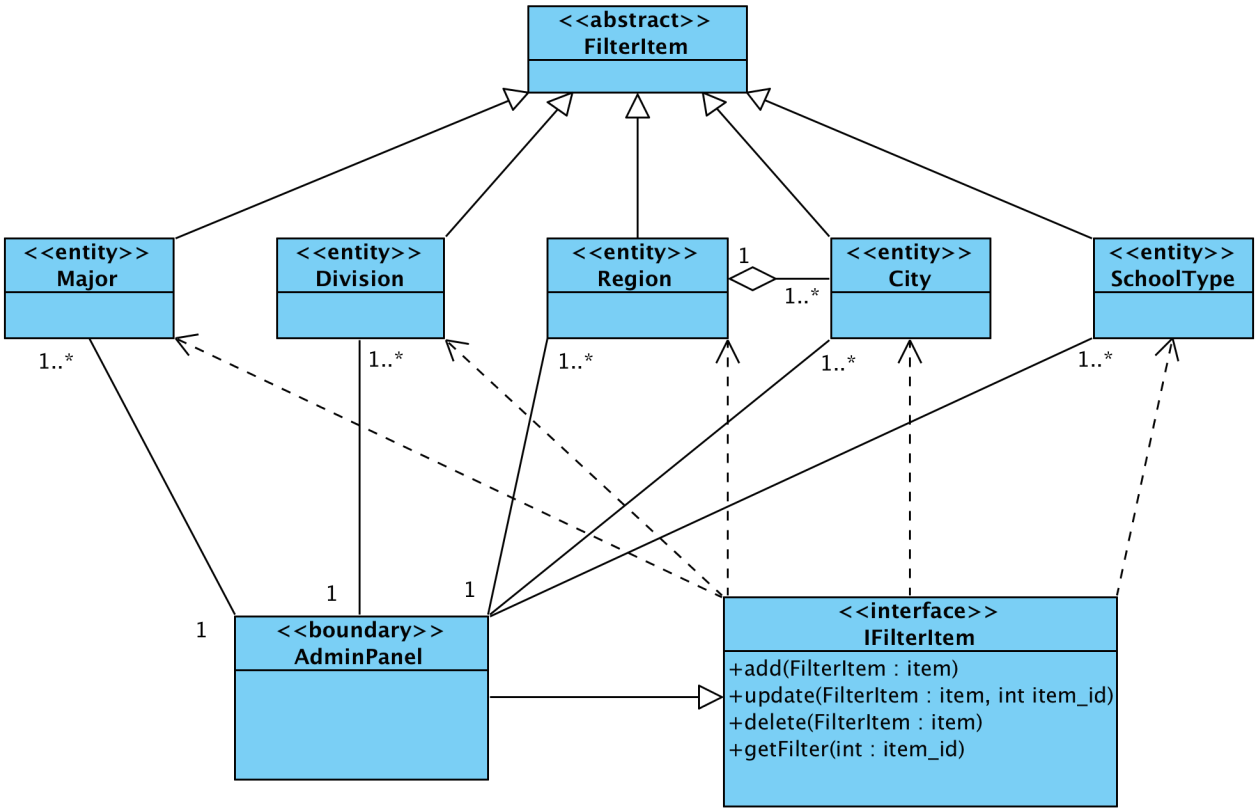


Figure 26 - Maintain Filter Subsystem

Subsystem Interface Descriptions

IFilterItem: encapsulates the action of add, update, delete or get a filter item.

Data: may be a Major, a Division, a Region, a City or a SchoolType.

add: add an item to the filter.

update: change the item in the filter to the given new item.

delete: remove an item from the filter with given item_id.

getFilter: get an item from the filter with given item_id.

3.1.2. Analysis Class to Design Element map

Analysis Class	Design Element
LoginForm	LoginForm
User	User
CommentWindow	ShowCommentForm
	AddCommentForm
	EditCommentForm
	DeleteCommentForm
UniInforForm	UniInforForm Subsystem
	IMaintainUniversity Interface
University	University
	Faculty
	Major
	News
MaintainAccountForm	MaintainAccountForm
MaintainProfileForm	MaintainProfileForm Subsystem
	IMaintainUniversity Interface
SchoolType	SchoolType
	FilterItem abstract
Division	Division
Region	Region
City	City
Major	Major
MainForm	MainForm
RegisterForm	RegisterForm

3.1.3. Design Element to Package map

Design Element	"Owning" package
LoginForm	Controller::GUI Controller
ShowCommentForm	
AddCommentForm	
EditCommentForm	
DeleteCommentForm	
MaintainAccountForm	
MainForm	
RegisterForm	
MaintainProfileForm	
UniInforForm	
User	Controller::Database Controller
University	
Faculty	
Major	
News	
Division	
City	
Region	
SchoolType	
IFilterItem	
IMaintainUniversity	

3.1.4. Architectural Components

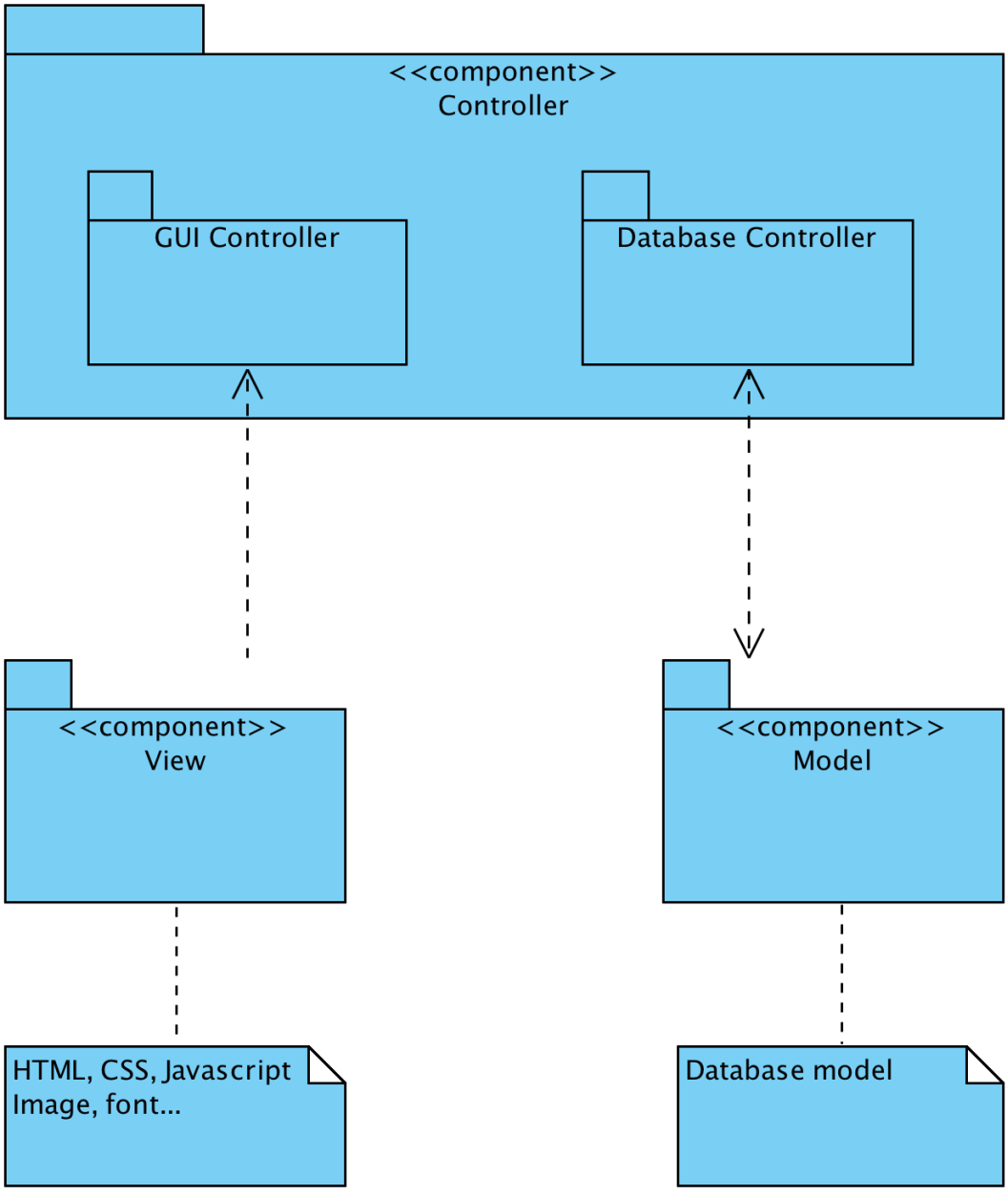


Figure 27 - Architectural Components

3.2. Describe the Run-time Architecture

This part describes the Restaurant Manager System Concurrency

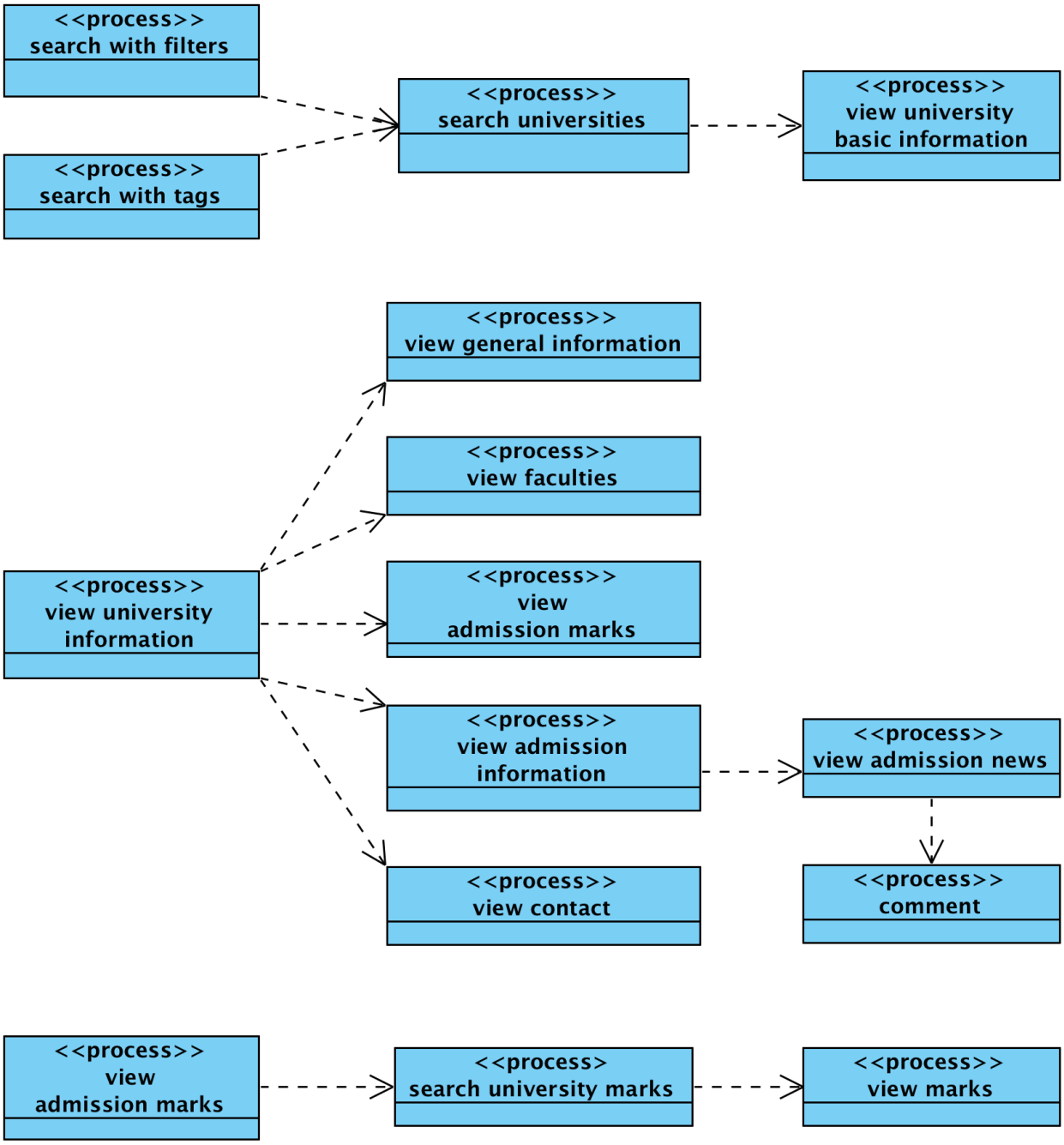


Figure 28 - Process Model

3.3. Describe Distribution

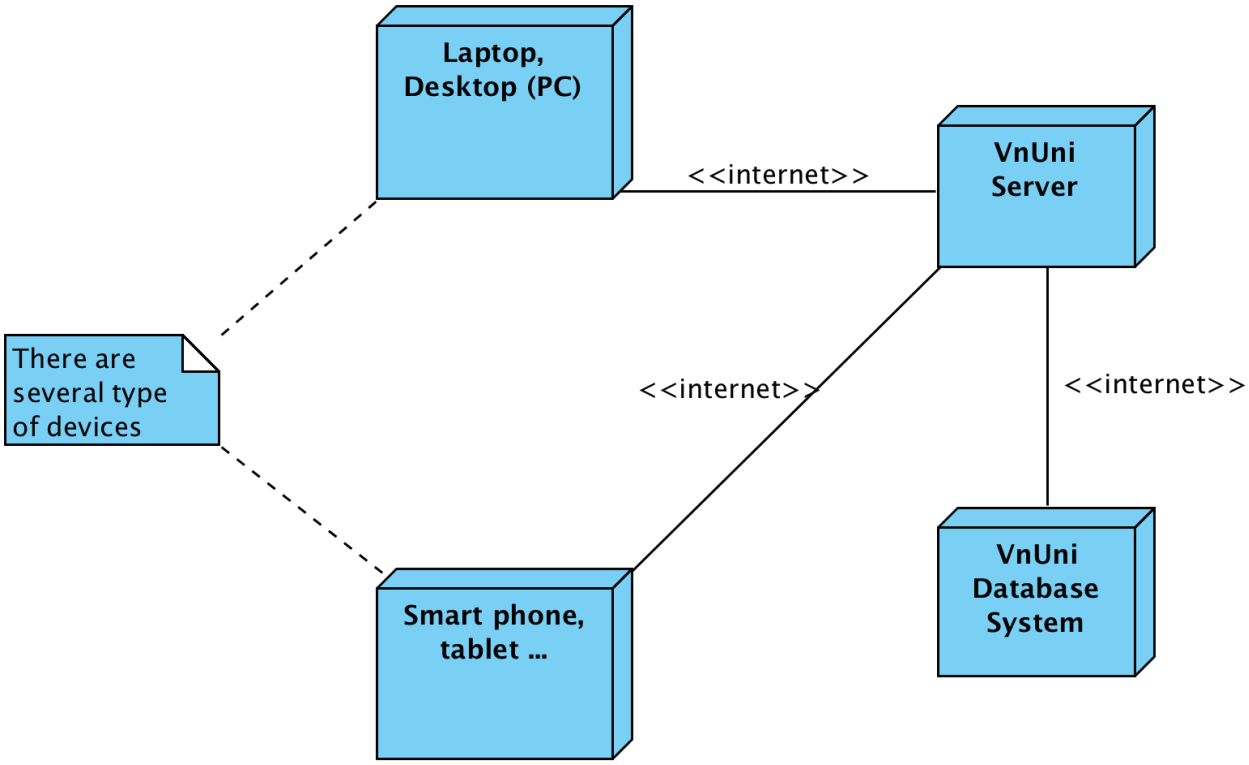


Figure 29 - Deployment Model

3.4. Use-case Design

This part describes the use case in the consideration of every analysis mechanism that were defined before.

3.5. Subsystem Design

None.

3.6. Class Design

3.6.1. Describe each class or interface

This part describes the all class structure and their description also the total class diagram and their dependency.

3.6.1.1. LoginForm

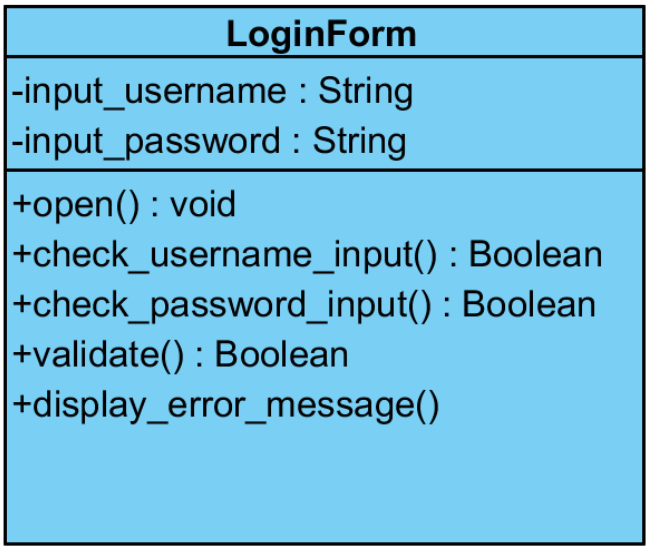


Figure 30 - LoginForm Class

3.6.1.2. CommentWindow

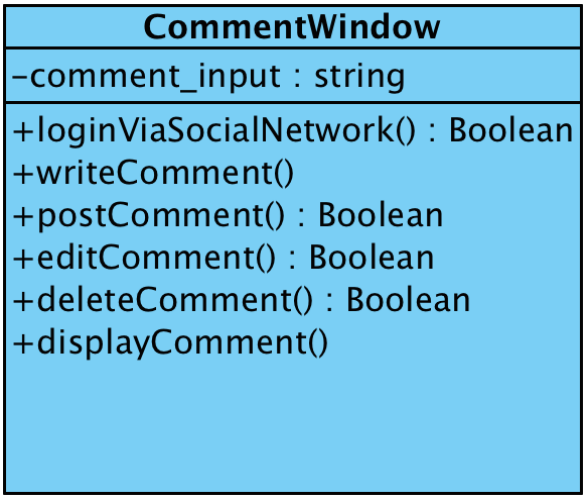


Figure 31 - CommentWindow Class

3.6.1.3. ShowCommentForm

This class inherits the class "CommentWindow" but the input form is different. So all function will be overridden.

3.6.1.4. AddCommentForm

This class inherits the class "CommentWindow" but the input form is different. So all function will be overridden.

3.6.1.5. EditCommentForm

This class inherits the class "CommentWindow" but the input form is different. So all function will be overridden.

3.6.1.6. DeleteCommentForm

This class inherits the class "CommentWindow" but the input form is different. So all function will be overridden.

3.6.1.7. MaintainAccountForm

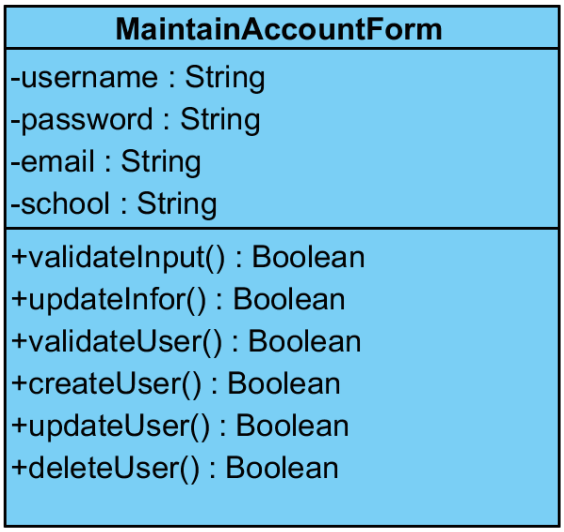


Figure 31 - MaintainAccountForm Class

3.6.1.8. UniInforForm

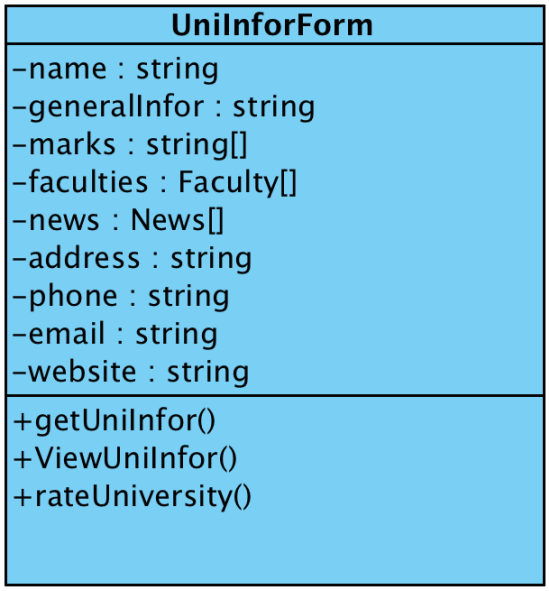


Figure 31 - UniInforForm Class

3.6.1.9. MaintainProfileForm

3.6.1.10. MainForm

3.6.1.11. AdminPanel

3.6.1.12. RegisterForm

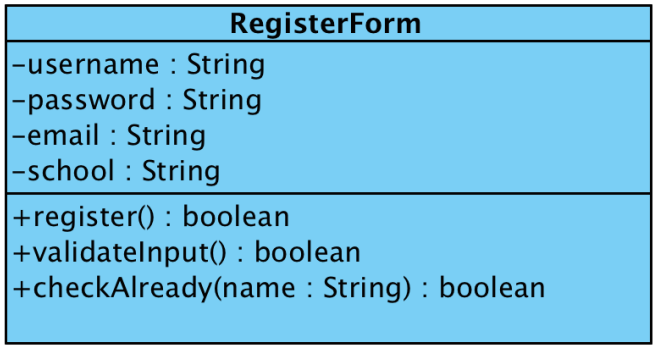


Figure 00 - RegisterForm Class

3.6.1.13. User

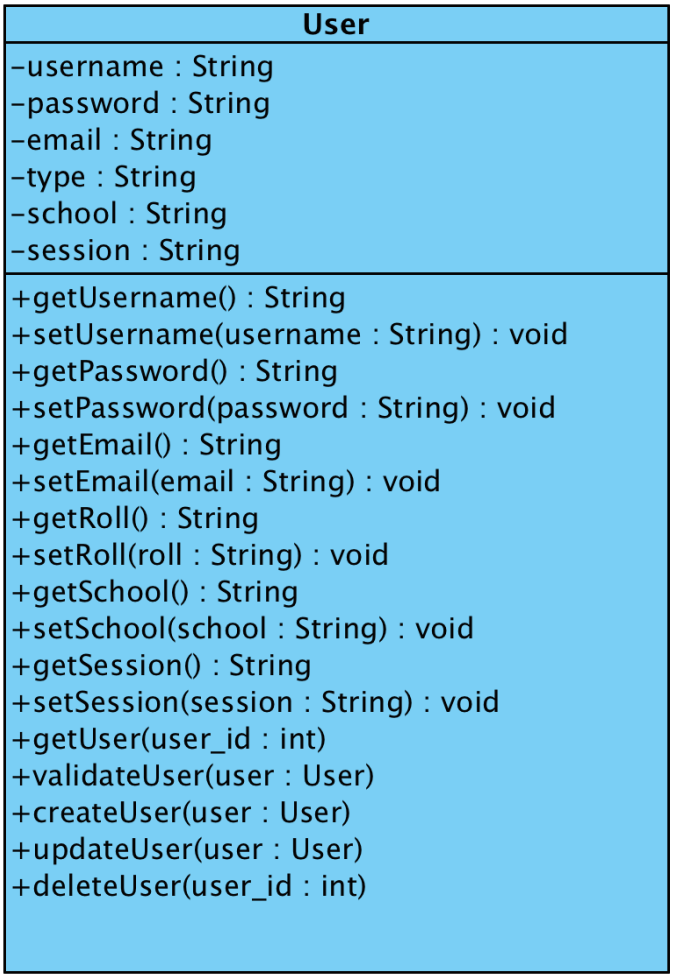


Figure 00 - User Class

3.6.1.14. University

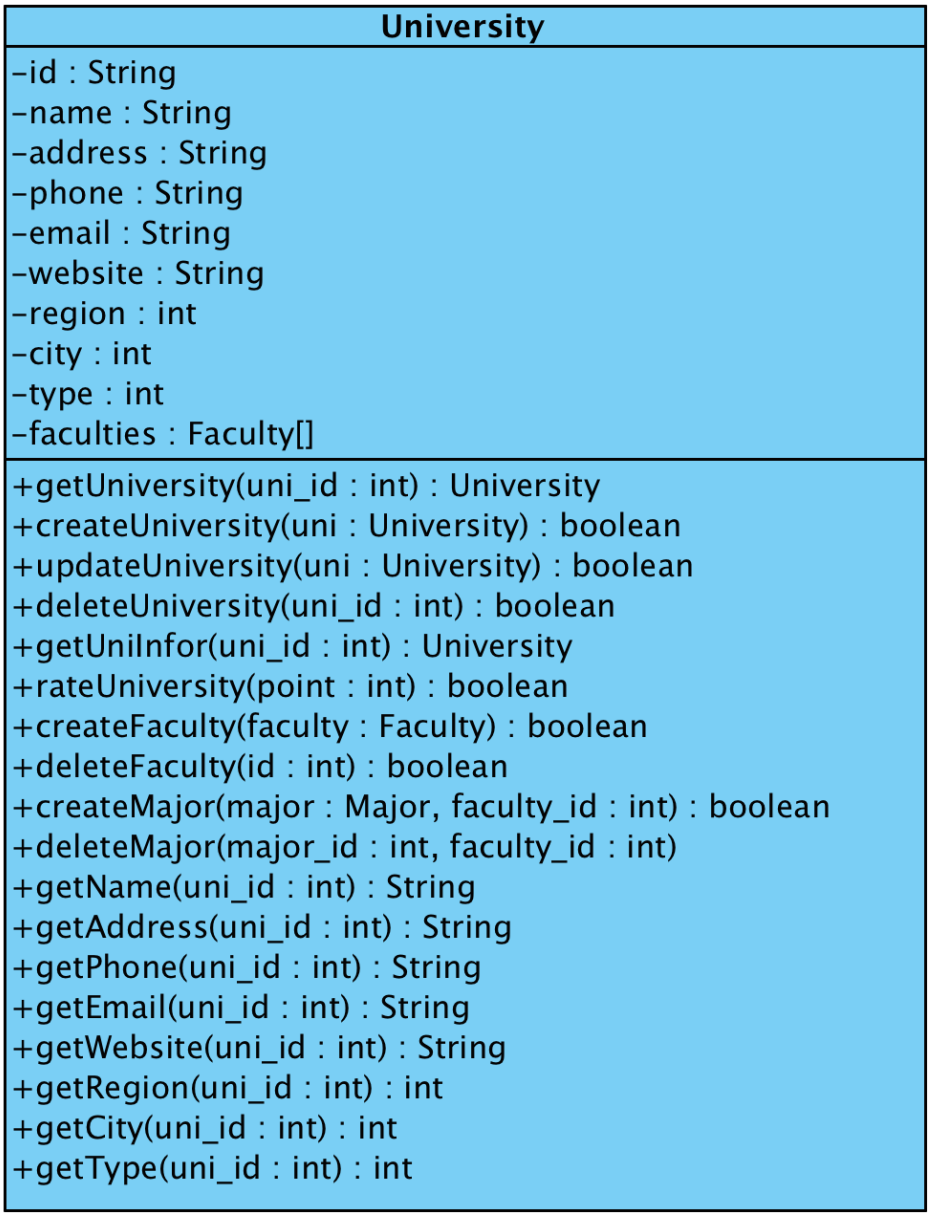


Figure 00 - University Class

3.6.1.15. Faculty

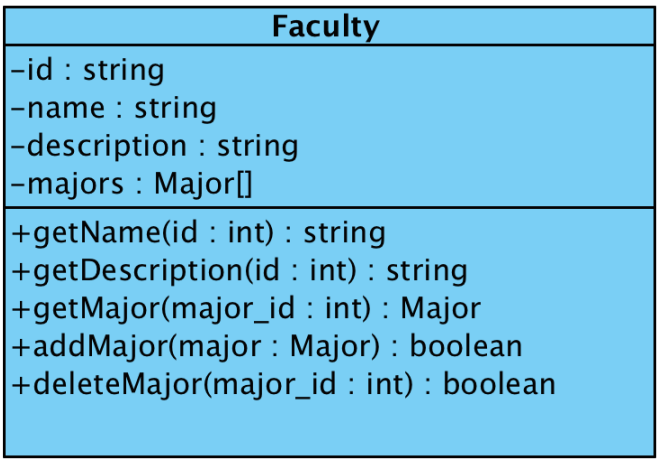


Figure 00 - Faculty Class

3.6.1.16. Major

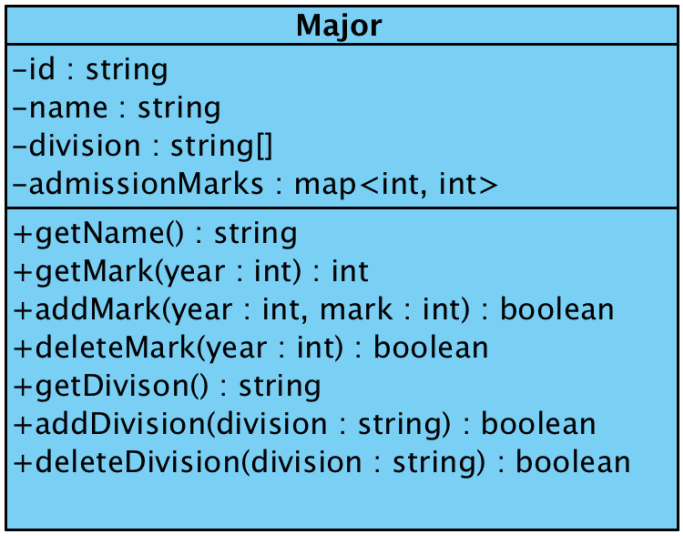


Figure 00 - Major Class

3.6.1.17. Division

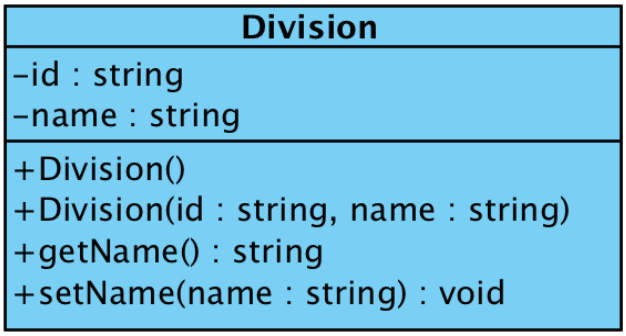


Figure 00 - Division Class

3.6.1.18. Region

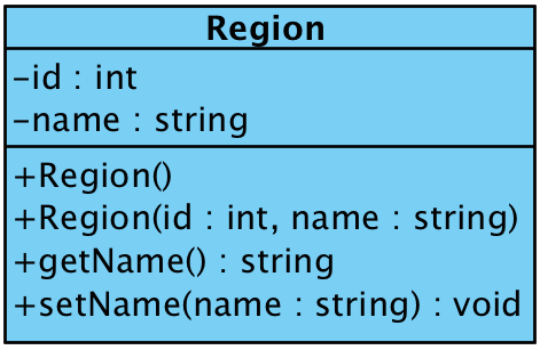


Figure 00 - Region Class

3.6.1.19. SchoolType

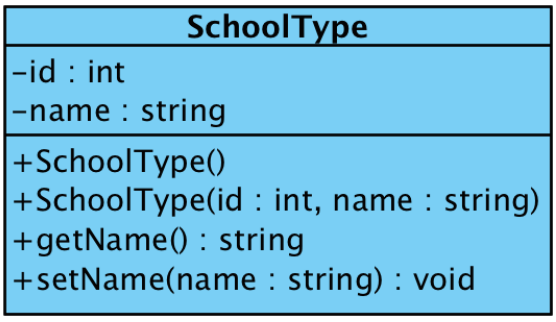


Figure 00 - SchoolType Class

3.6.1.20. City

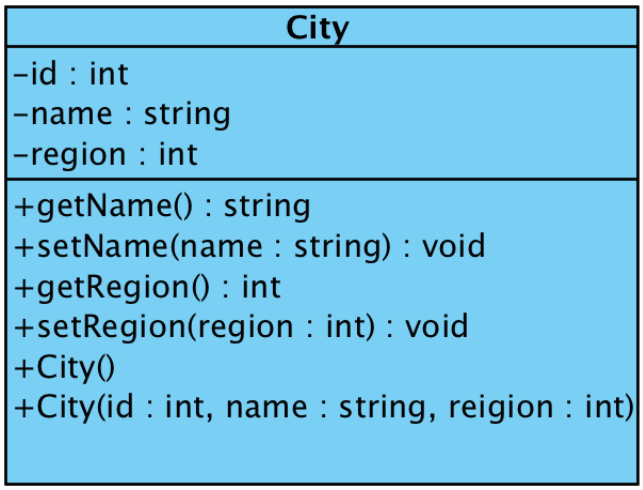


Figure 00 - City Class

3.6.1.21. News

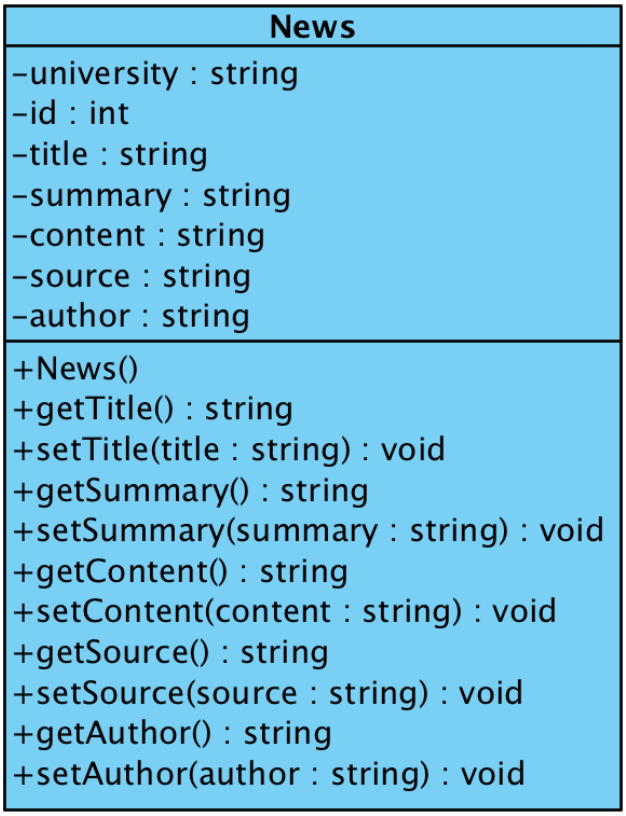


Figure 00 - News Class

3.6.1.22. IFilterItem

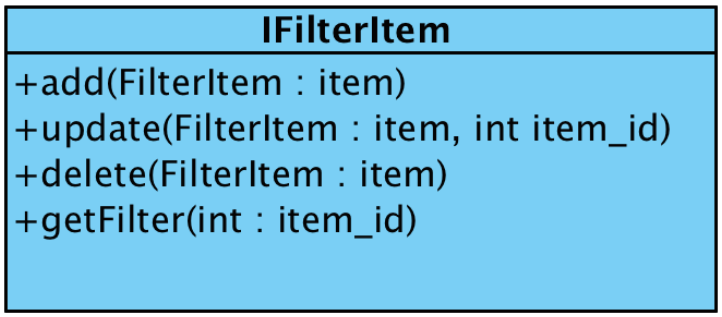


Figure 00 - IFilterItem Interface

3.6.1.23. IMaintainUniversity

3.6.2. Class diagram in total

Appendix A - Figures