## S E - I C O U R S E P R O J E C T ( P H A S E 2 C O V E R S H E E T )

**Discussions Scheduled for Week 11 or 12** *(Specific dates TBA later).*

* Print 1 copy of this cover sheet and attach it to a printed copy of the documentation *(SRS, … etc.)*. You must submit softcopies of all your documents *(as PDFs)*; details will be announced later.
* Please write all your names in Arabic.
* Please make sure that your students’ IDs are correct.
* Handwritten Signatures for the attendance of all team members should be filled in before the discussion.
* Please attend the discussion on time *(announced separately)*, late teams will lose 3 grades.

# Project Name:

**Team Information *(typed not handwritten, except for the attendance signature)*:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **ID**  **[Ordered by ID]** | **Full Name [In Arabic]** | **Attendance**  **[Handwritten Signature]** | **Final Grade** |
| **1** | 20208012 | احمد طارق عبدالرحمن منير |  |  |
| **2** | 20208088 | حسام الدين خالد محمد فتح الله |  |  |
| **3** | 20208100 | رحمه الدريني الدريني محمد |  |  |
| **4** | 20208121 | ريم ابراهيم شوقي ابراهيم |  |  |
| **5** | 20208197 | محمد سعيد وزيري احمد |  |  |

# Grading Criteria:

|  |  |  |  |
| --- | --- | --- | --- |
| **Items** | | **Grade** | **Notes** |
| **System Architecture –** *including applied Architectural Pattern(s)* | **2** |  |  |
| **Sequence Diagram(s)** *including* ***System Sequence Diagrams (SSDs)*** | **3** |  |  |
| **Collaboration/Communication Diagram(s)** | **2** |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Class Diagram** *(3 versions)*   1. **An initial version based on the requirements and Use- Case/Activity diagrams.** 2. **An intermediate version based on the interaction diagrams.** 3. **A final version, after applying the design pattern(s) and**   **any other modifications.** | **4.5** |  |  |
| **Package Diagram(s)** | **2** |  |  |
| **3 Mandatory Design Patterns Applied** *(Including a typed description)* | **4.5** |  |  |
| **Object Diagrams** *(Including object diagrams that illustrate the preconditions and the post-conditions of selected functions)* | **2** |  |  |

**N.B.** .. **You must update and resubmit the initial part of the documentation submitted in phase 1** (including the Functional / Non-Functional requirements, Use-case Diagrams & Descriptions, Activity Diagrams, .. etc.).

## Teaching-Assistant’s Signature:

**20**

**Software Requirements Specification**

**For**

**Exam Cell Automation**

**Version 1.0 approved.**

**Prepared by:**

-Rahma El-Dreny El-Dreny Mohamed.

-Reem Ibrahim Shawky Ibrahim.

-Hossam El-Dien Khaled Mohamed.

-Ahmed Tarek Abdelrahman Monir.

-Mohamed Saeed Wazery Ahmed.

**Helwan University – Faculty of Computer and Information – Medical Informatics department.**

**May 2021.**

**Table of Contents**  **ii**

**1.** **Introduction** **1**

1.1 Purpose 1

1.2 Intended Audience and Reading Suggestions 1

1.3 Product Scope 1

1.4 References 1

**2.** **Overall Description** **2**

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

**3.** **System Features** **4**

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

**4.** **Other Non-functional Requirements** **4**

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

**Appendix: Analysis Models** **5**

# 1. Introduction

This document contains information about intent to have its existing internet web site redesigned and implemented in a responsive, flexible and interactive design; such that it requires least number of user actions/interaction to get the content delivered and presented.

## 1.1 Purpose

The project aims to bring in a centralized system that will ensure the activities in the context of an examination that can be effectively managed.

## 1.2 Intended Audience and Reading Suggestions

This project is a prototype for exam cell automation system and it is restricted within the college premises. This has been implemented under the guidance of college professors. This project is useful for the college management team and as well as to the students.

## 1.3 Product Scope

The purpose of the online exam cell automation system is to ease exam management and to create a convenient and easy-to-use application for students, trying to get their academic records. The system is based on a relational database with college system and students. We will have a database server supporting hundreds of students and their related information.

## 1.4 References

https://drive.google.com/drive/folders/1GdWnofSGJ5gsgNA2IzOk6guFFgPF1K3r

*2*

## 2. Overall description

2.1Product perspective

the centralized exam cell system manages the following

* Exam details: that includes the exam’s duration, exam’s date, exam’s hall
* Subject’s info: that includes the subject’s credit hours, Subject’s ID
* Student info: that includes student ID, email addresses, phone numbers, hall tickets
* Mark sheet: that is specified for every student and contain the following info: number of subjects the student had registered, student’s grade in every subject, student’s GPA and semester number

## 2.2 Product Functions

The student can register himself into the system by providing his info, after logging in he can view his enrolled subjects, can view and edit his info, can view his mark sheet that contains his educational career info and can view or print his hall ticket

On the other hand, the admin can register any student into the system by providing the student’s info, after logging in the admin can view all the enrolled student and view or edit their info, can view all the student’s mark sheets.

## 2.3 User Classes and Characteristics

The system will support two types of user privileges, Student and Admin. Students will have access to students' functions, and the admins have access to both students and exam management functions.

The student should be able to do the following functions:

* Make a new subject enrollment
* View mark sheet
* View and edit his info
* View and print his hall ticket
* Register into the system
* Login

The admin should have following management functionalities:

* Student's privileges
* View all the registered students
* View/edit student’s info
* Generation marksheets
* View all/specific student’s marksheet
* Editing a specific marksheet

2.4 Operating Enviromint

Operating environment for the exam management system is as listed below.

* Local database
* Operating system: Windows.
* database:SQL+database.

# 3. System Features

* **DESCRIPTION and PRIORITY**

The exam cell automation system maintains information students, their registered courses and academic records. Of course, this project has a high priority because it is very difficult to deal with that number of students and their records at the same time at the college.

* **STIMULUS/RESPONSE SEQUENCES**
  + Displays student’s ticket hall.
  + Displays a detailed list of academic records.
  + Edit student’s information.
* **FUNCTIONAL REQUIREMENTS**
  + Student should register to enroll the system.
  + Student shall Log into the system.
  + Student shall view his ticket hall and print it.
  + Student shall view his mark sheet and print it.
  + Students shall edit his personal information.
  + System should make login verification
  + System should make registration verification.
  + System should send an e-mail to the student to confirm his registration and the student’s ticket hall will be attached in the mail.
  + System should send all enrolled students to the admin.
  + Admin shall log into the system.
  + Admin shall register students into the system.
  + Admin should view all enrolled students.
  + Admin shall edit student information.
  + Admin should enter grades in the mark sheet.
  + Admin shall view all mark sheets or view one mark sheet only, print all mark sheets separately or print only one mark sheet.
  + Admin should generate one mark sheet for each student.

# 4. Other Non-functional Requirements

## 4.1 Performance Requirements

-Boost's enterprise accessibility.

-Faster exam registration.

-Improved accuracy of student data.

4.2 Usability & Humanity REQS

- Easy result generation.

- Better convenience for students.

## 4.3 Security Requirements

- System prevents unauthorized access

## 4.5 Software Quality Attributes

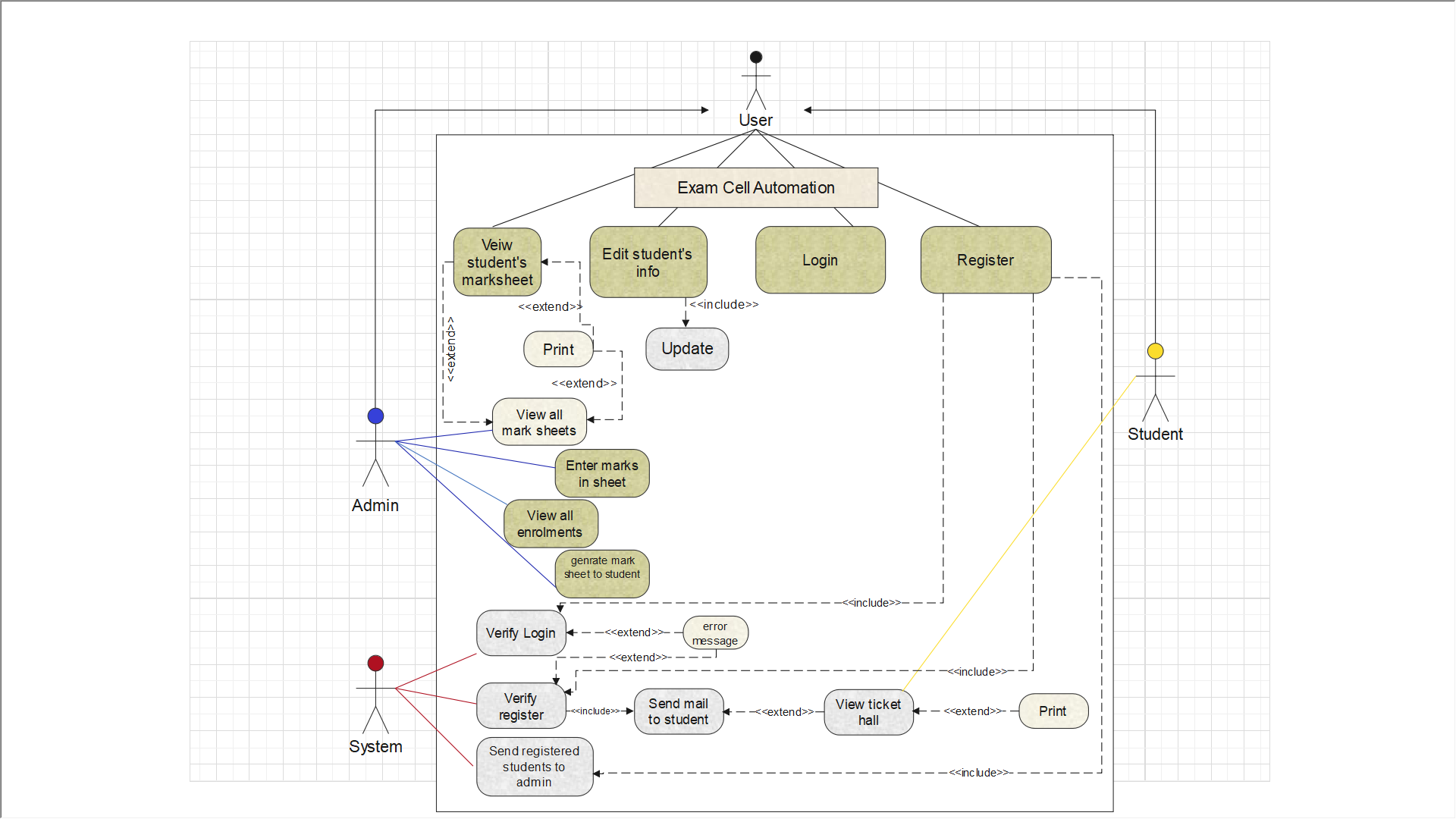
-There will be total three to six semesters each semester contains maximum seven subjects.

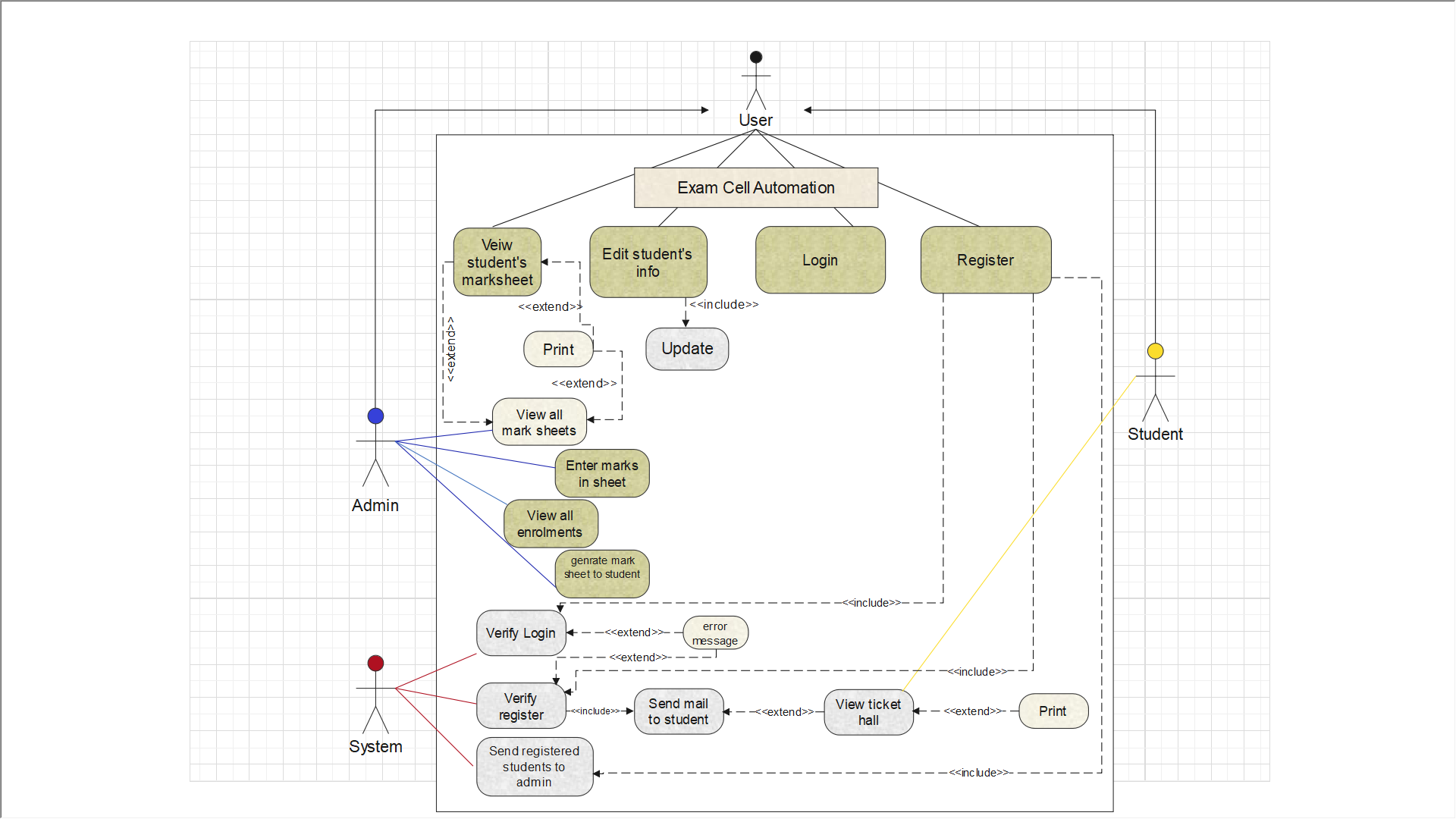
**Appendix: Analysis Models**

* Activity Diagram:

Diagram

Description automatically generated

Use Case Diagram:



* Use case Scenarios:

1. Register use case:

|  |  |
| --- | --- |
| Use case name | Registration. |
| Actors | Student, Admin. |
| Precondition | User should enter all information are needed to be registered in the system. |
| Basic flow | 1. Open registration page. 2. Enter all needed information. 3. Click on submit button. 4. System verify registration information <included>. |
| Alternative flow | If the information is not valid:-  -the user is promoted to re-enter information again.  If email exists in database:-  -prompt user to enter new email. |
| Post condition | If the case completed successfully :  -account is created successfully.  -user received an email from the system confirms his registration <included>.  -send registered students to admin <included>. |

2. Send mail to students use case:

|  |  |
| --- | --- |
| Use case name | Send mail to students. |
| Actors | System. |
| Precondition | -User should register to the system.  -System verify his registration. |
| Basic flow | 1. System checks user’s information. 2. Auto send a mail to the user to confirm his registration, <included> student’s ticket hall. |
| Alternative flow | If the user does not receive an email: -  -the user should register again. |
| Post condition | If the case completed successfully:  -user received an email from the system confirms his registration and his ticket hall. |

3. Send registered students to admin use case:

|  |  |
| --- | --- |
| Use case name | Send registered students to admin. |
| Actors | System. |
| Precondition | User should register to the system. |
| Basic flow | 1. After the system checked registered students and confirmed their registration. 2. Auto send all enrolments to the admin. |
| Alternative flow |  |
| Post condition | If the case completed successfully:  -Admin receives all enrolled students. |

4. View ticket hall use case:

|  |  |
| --- | --- |
| Use case name | View ticket hall. |
| Actors | Student. |
| Precondition | System sends an email to the student. |
| Basic flow | 1. Open the mail. 2. Press view ticket hall. 3. Click print button (optional). |
| Alternative flow |  |
| Post condition | If the case completed successfully:  -student will have his ticket hall and he can print it or not (optional). |

5. Print ticket hall use case:

|  |  |
| --- | --- |
| Use case name | Print ticket hall. |
| Actors | Student. |
| Precondition | Student views ticket hall. |
| Basic flow | 1. View ticket hall. 2. Click print button. |
| Alternative flow | Student may not print the ticket hall. |
| Post condition | If the case completed successfully:  -student will have his ticket hall printed successfully. |

6. Login use case:

|  |  |
| --- | --- |
| Use case name | Login. |
| Actors | Admin, student. |
| Precondition | User should enter username and password. |
| Basic flow | 1. Open login page.  2. Enter username and password.  3. Click on login button.  4. System verify login information <included>. |
| Alternative flow | System will check username and password, if the username or password is incorrect: -  -the user is promoted to re-enter them again. |
| Post condition | If the case completed successfully:  -the user can access the website and log into his account. |

7. View all enrolments use case:

|  |  |
| --- | --- |
| Use case name | View all enrolments. |
| Actors | Admin. |
| Precondition | -System sends all enrolled students to admin.  -Admin should log into his account. |
| Basic flow | 1. Admin opens his login page. 2. Click view all enrolments. |
| Alternative flow |  |
| Post condition | If the case completed successfully:  -Admin can access all information of all enrolled students. |

8. Edit student’s information use case:

|  |  |
| --- | --- |
| Use case name | Edit student’s information. |
| Actors | Admin, students. |
| Precondition | User should log into the system. |
| Basic flow | 1. Click edit button. 2. Edit information. 3. Click update button. |
| Alternative flow |  |
| Post condition | If the case completed successfully :  -Student’s information will be edited and updated. |

9. Enter mark sheet use case:

|  |  |
| --- | --- |
| Use case name | Enter mark sheets. |
| Actors | Admin. |
| Precondition | Student should have an exam sheet. |
| Basic flow | 1. Open mark sheet. 2. Enter student’s information and grades. 3. Enter save button. |
| Alternative flow | If student does not have an exam sheet:-  -will be counted as absence. |
| Post condition | If the case completed successfully :  -Every student has one mark sheet, admin has access to all mark sheets. |

10. Generate mark sheets use case:

|  |  |
| --- | --- |
| Use case name | Generate mark sheets. |
| Actors | Admin. |
| Precondition | Admin should have entered all mark sheets. |
| Basic flow | 1. Generate mark sheets to each student separately to his account. |
| Alternative flow |  |
| Post condition | If the case completed successfully :  -Each student will have his mark sheet . |

11. View all mark sheets use case:

|  |  |
| --- | --- |
| Use case name | View mark sheets. |
| Actors | Admin. |
| Precondition | -Admin should have entered all mark sheets. |
| Basic flow | 1. View all mark sheets. 2. Press one mark sheet specific to view it. 3. Click print all mark sheets (each one separately) (optional). 4. Click print mark sheet (for specific mark sheet) (optional). |
| Alternative flow |  |
| Post condition | If the case completed successfully :  -admin can view all mark sheets and can print them or not (optional). |

12. View mark sheet use case:

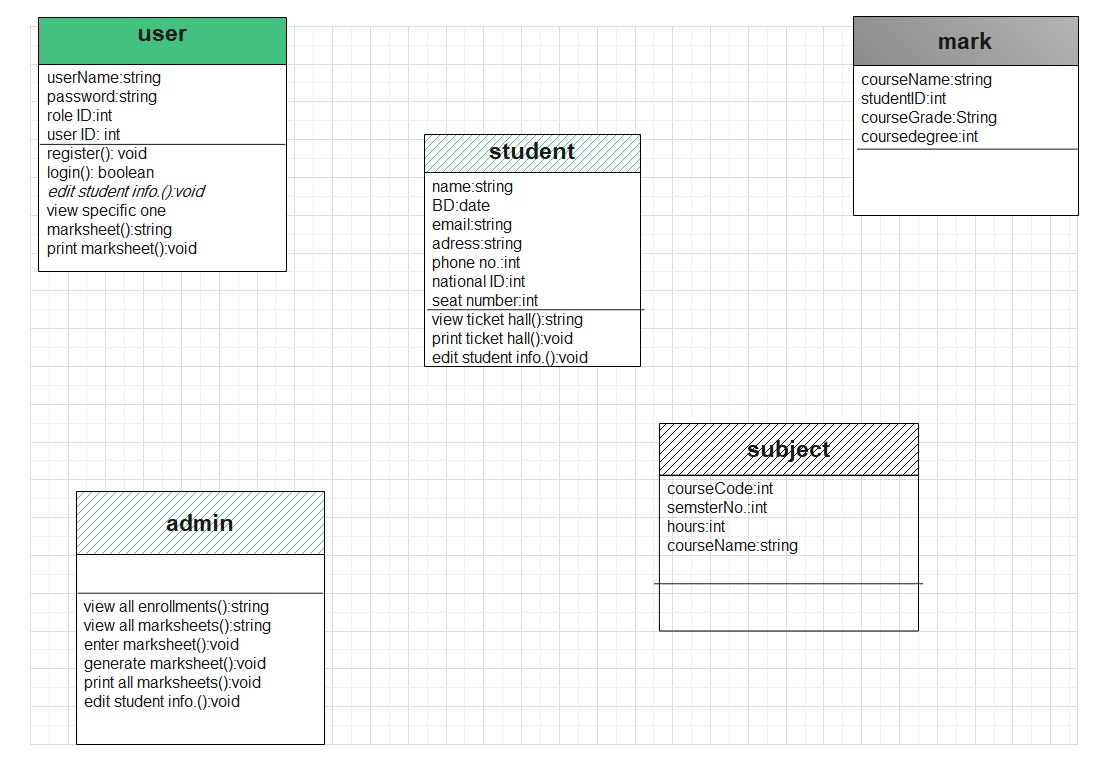
|  |  |
| --- | --- |
| Use case name | View mark sheet. |
| Actors | Admin, students. |
| Precondition | -Admin should have entered all mark sheets.  -Each students received his mark sheet. |
| Basic flow | 1. View mark sheet. 2. Click print mark sheet (optional). |
| Alternative flow | If student can’t found his mark sheets:-  -student should mail the admin by his problem. |
| Post condition | If the case completed successfully :  -user can view mark sheet and can print it or not (optional). |

13. Print mark sheet use case:

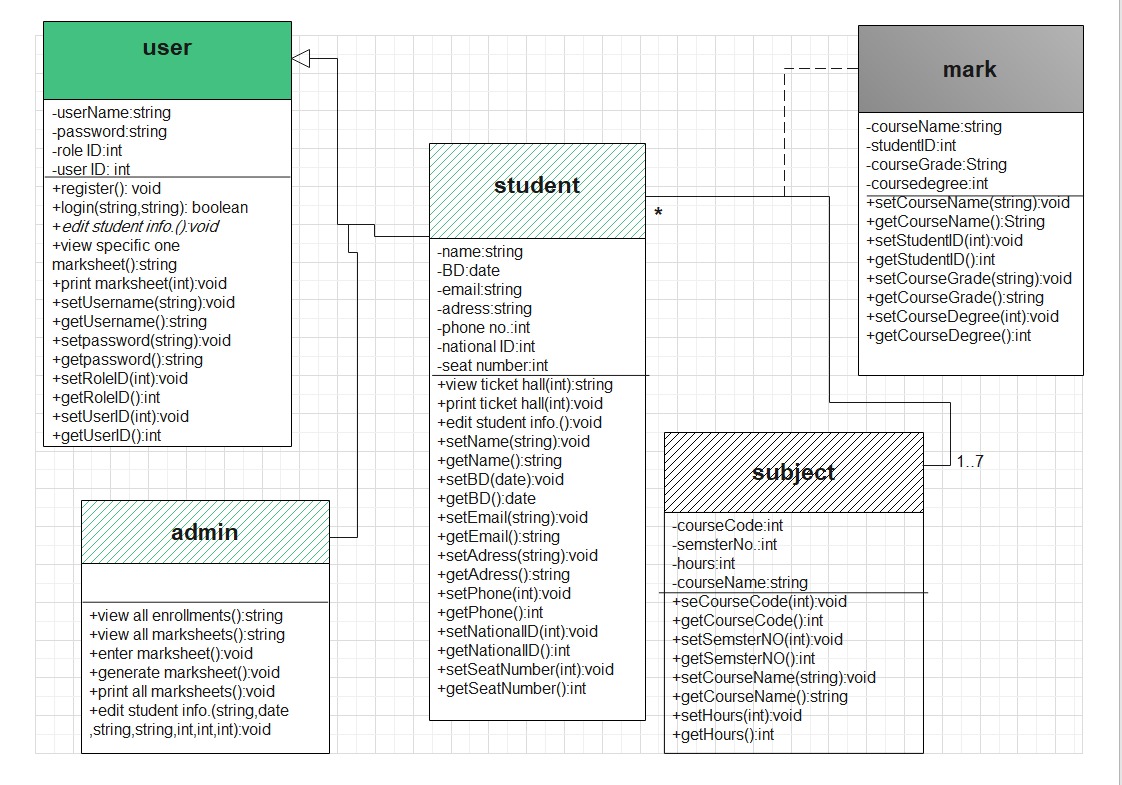
|  |  |
| --- | --- |
| Use case name | Print mark sheet. |
| Actors | Admin, Student. |
| Precondition | User views the mark sheet. |
| Basic flow | 1. View mark sheet. 2. Click print button. |
| Alternative flow | User may not print the ticket hall. |
| Post condition | If the case completed successfully :  -user will have the mark sheet printed successfully.  -if there is more than one mark sheet it will be printed successfully. |

* Class Diagram:

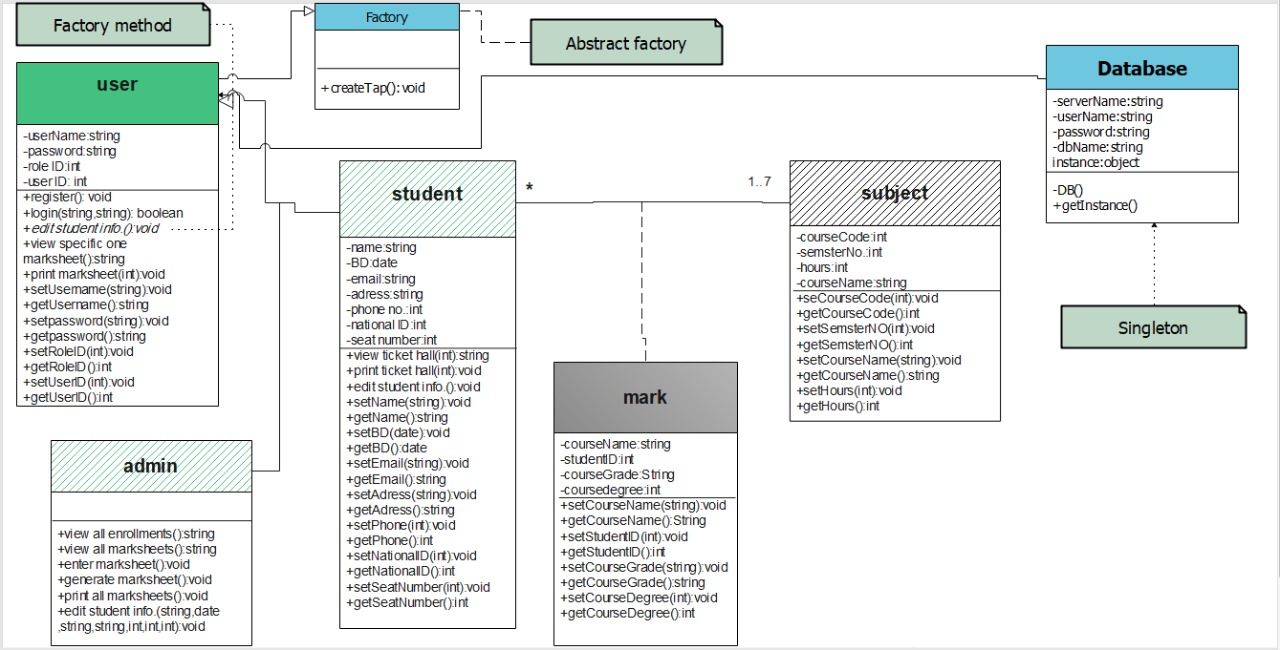
Version 1:



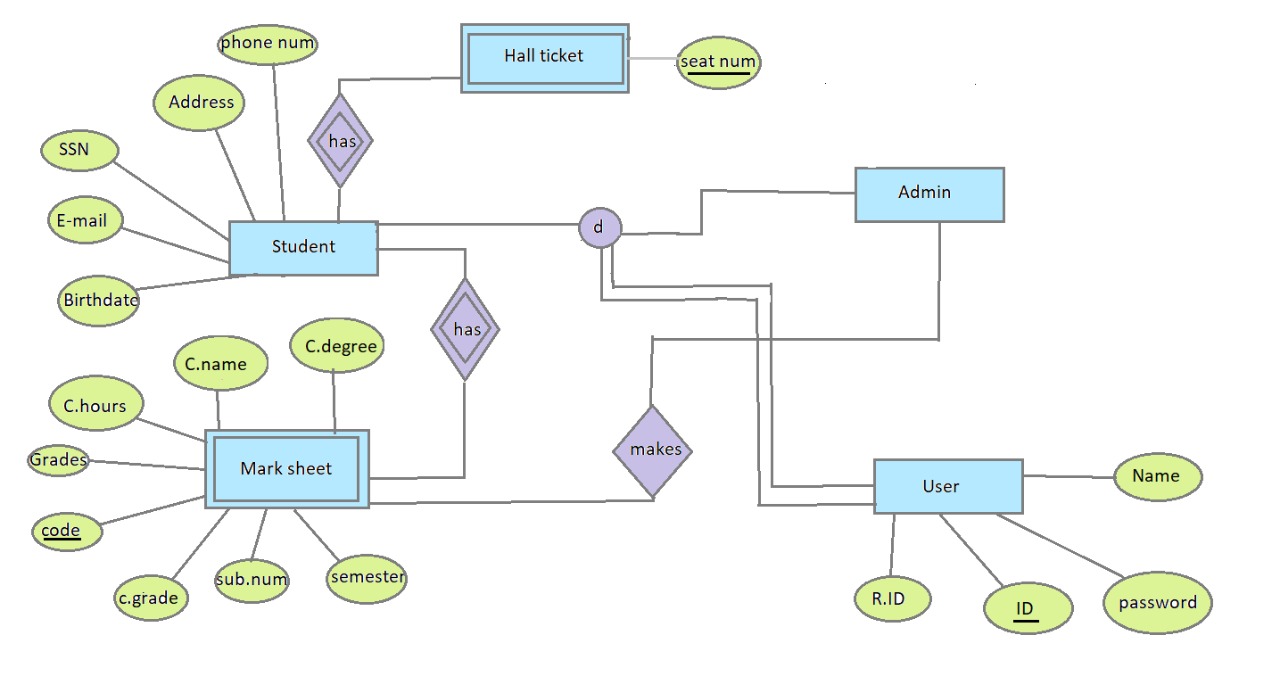
Version 2:



Version 3:



* Entity Relationship Diagram:



* Relational Model:

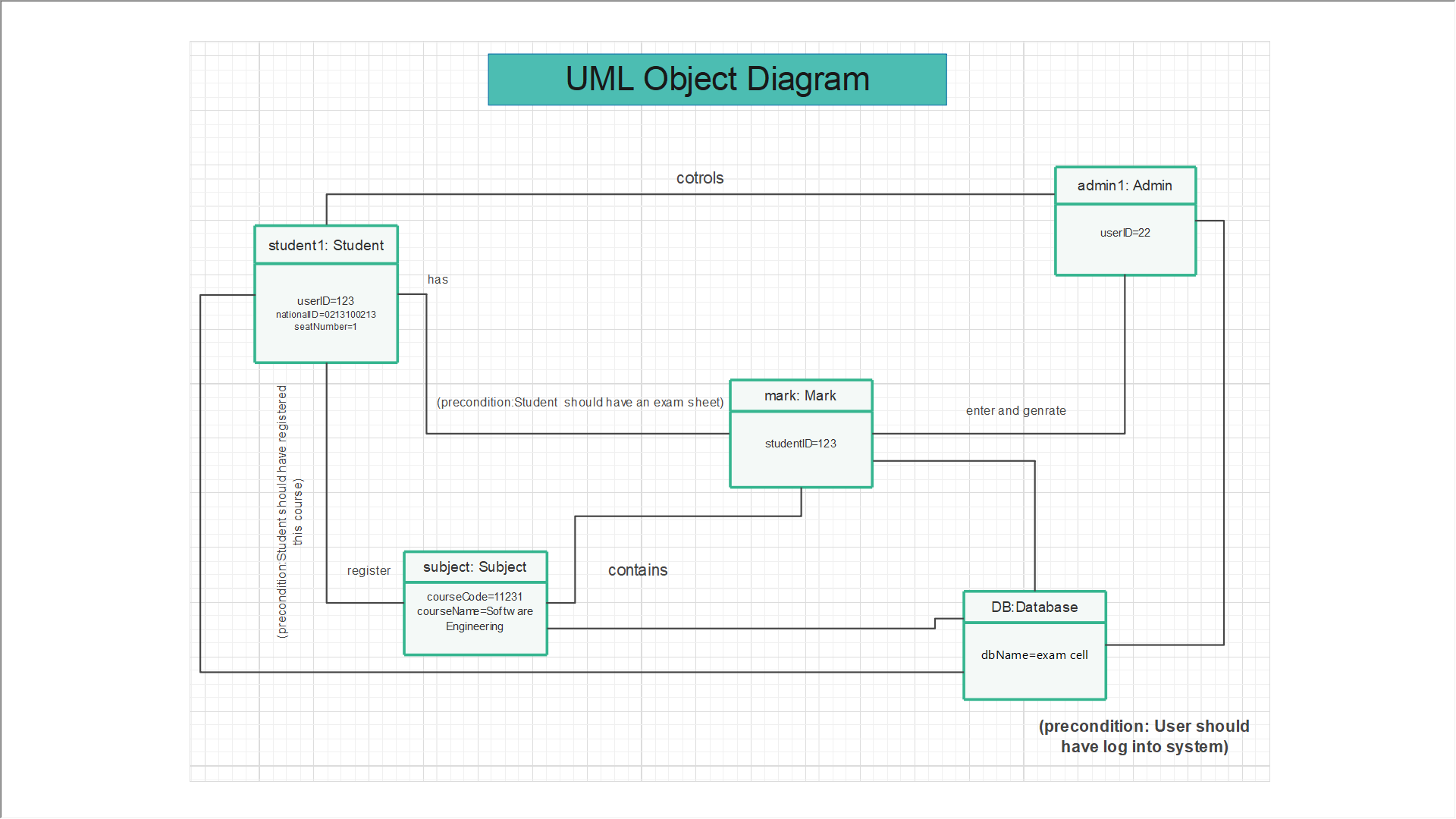
Diagram

Description automatically generated

* Package Diagram:

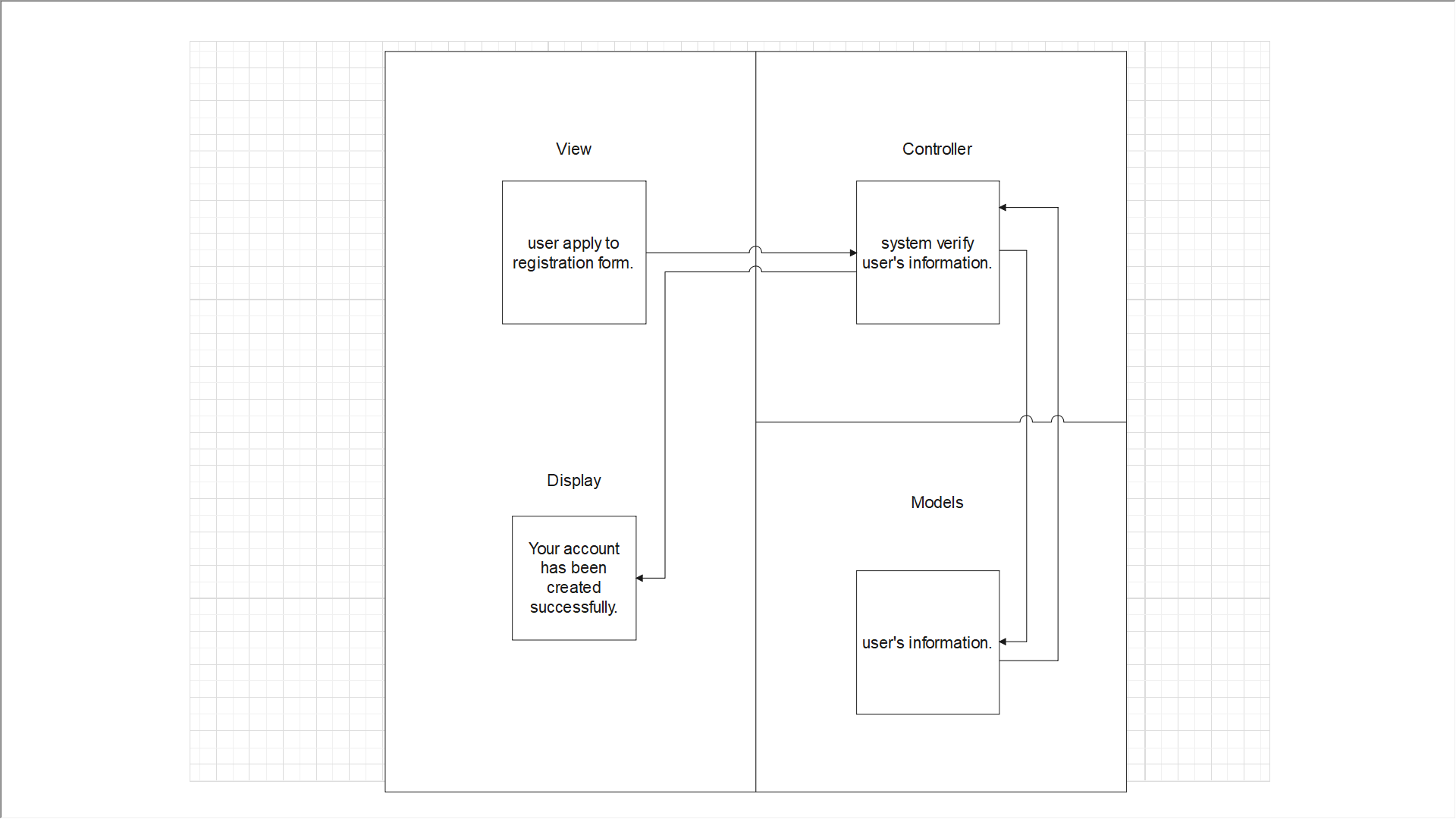
## 

* Object Diagram:

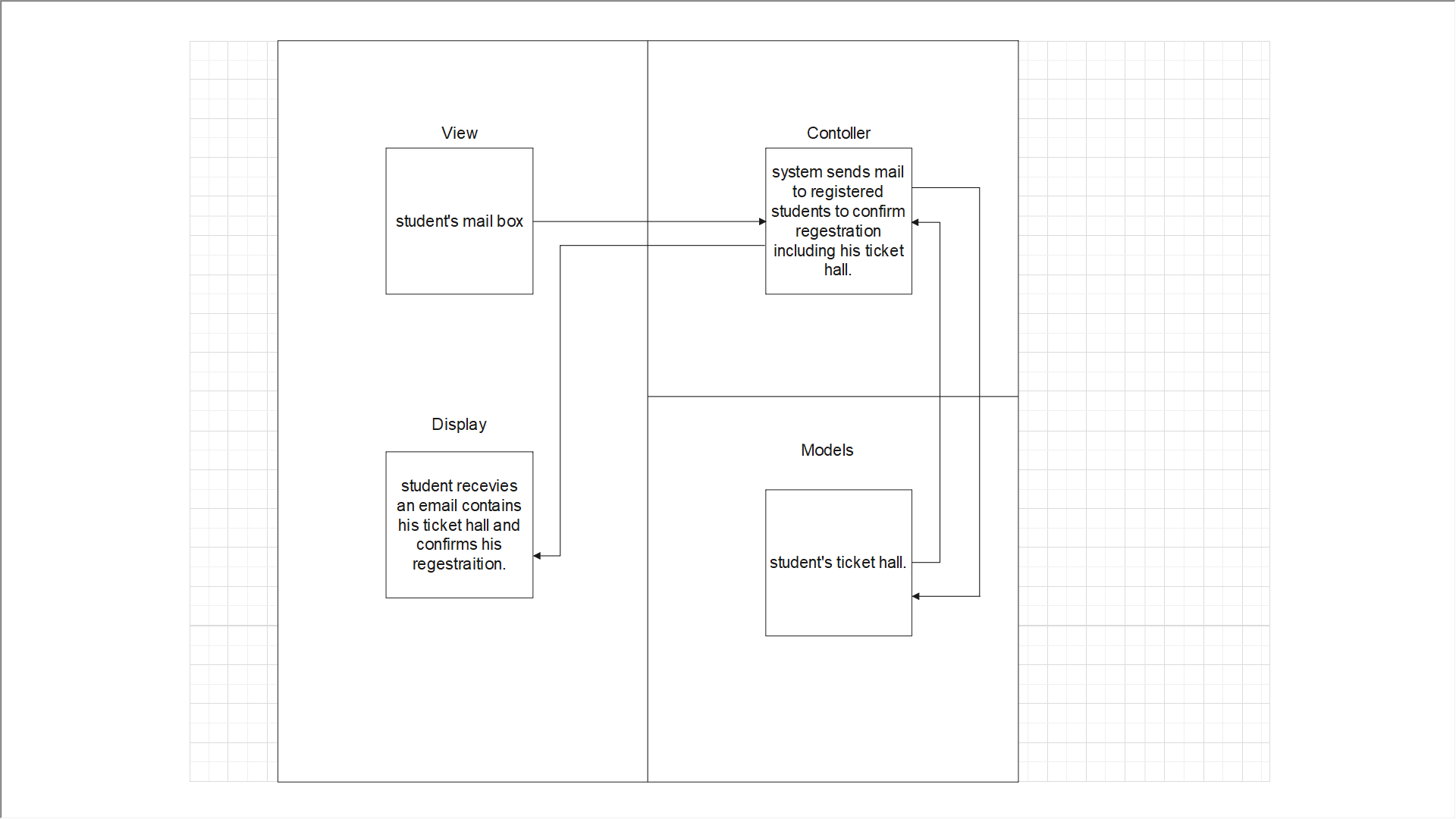


* Architecture Design (MVC):

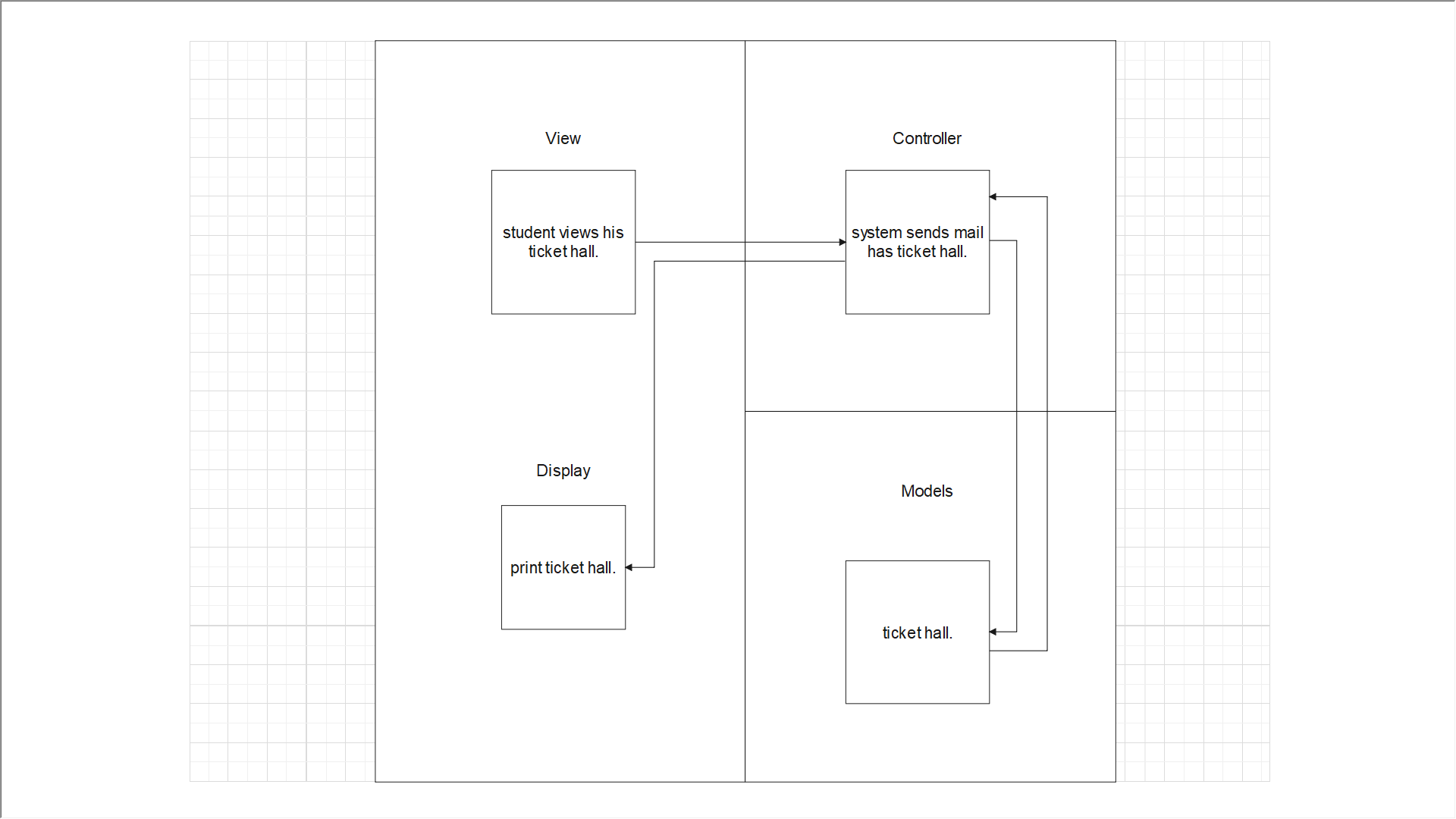
Register:



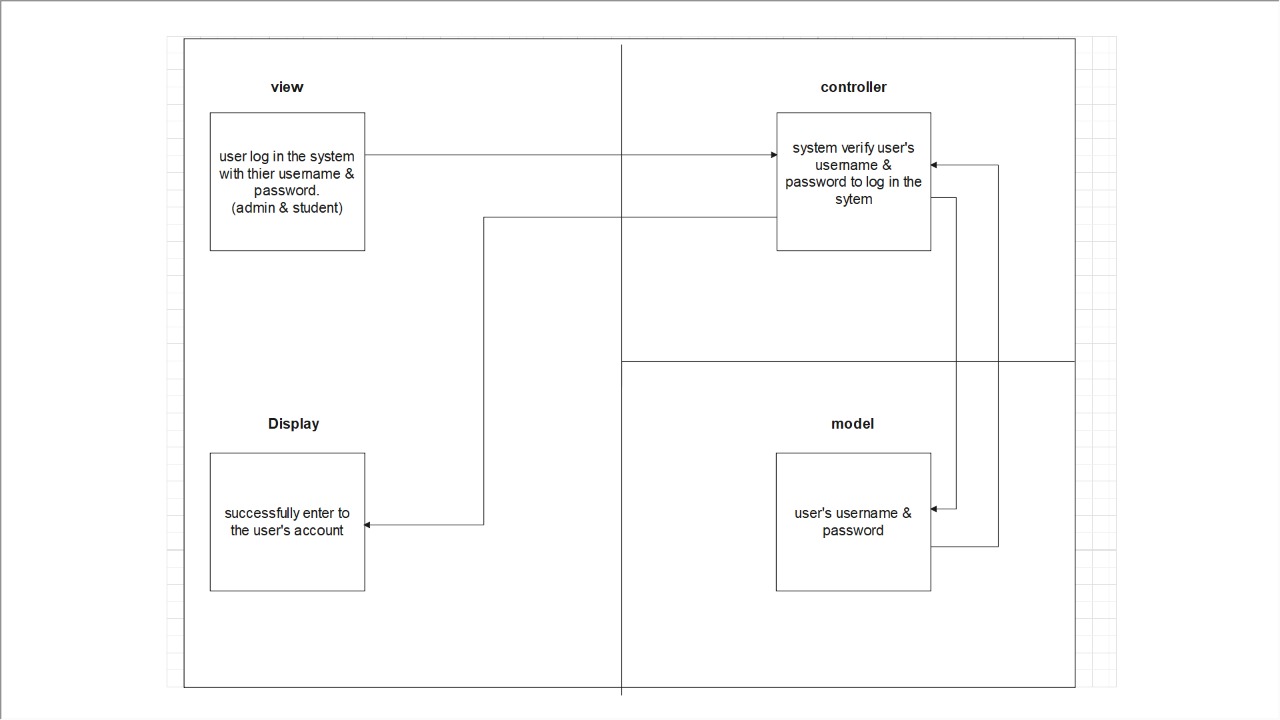
System sends mail to registered students:



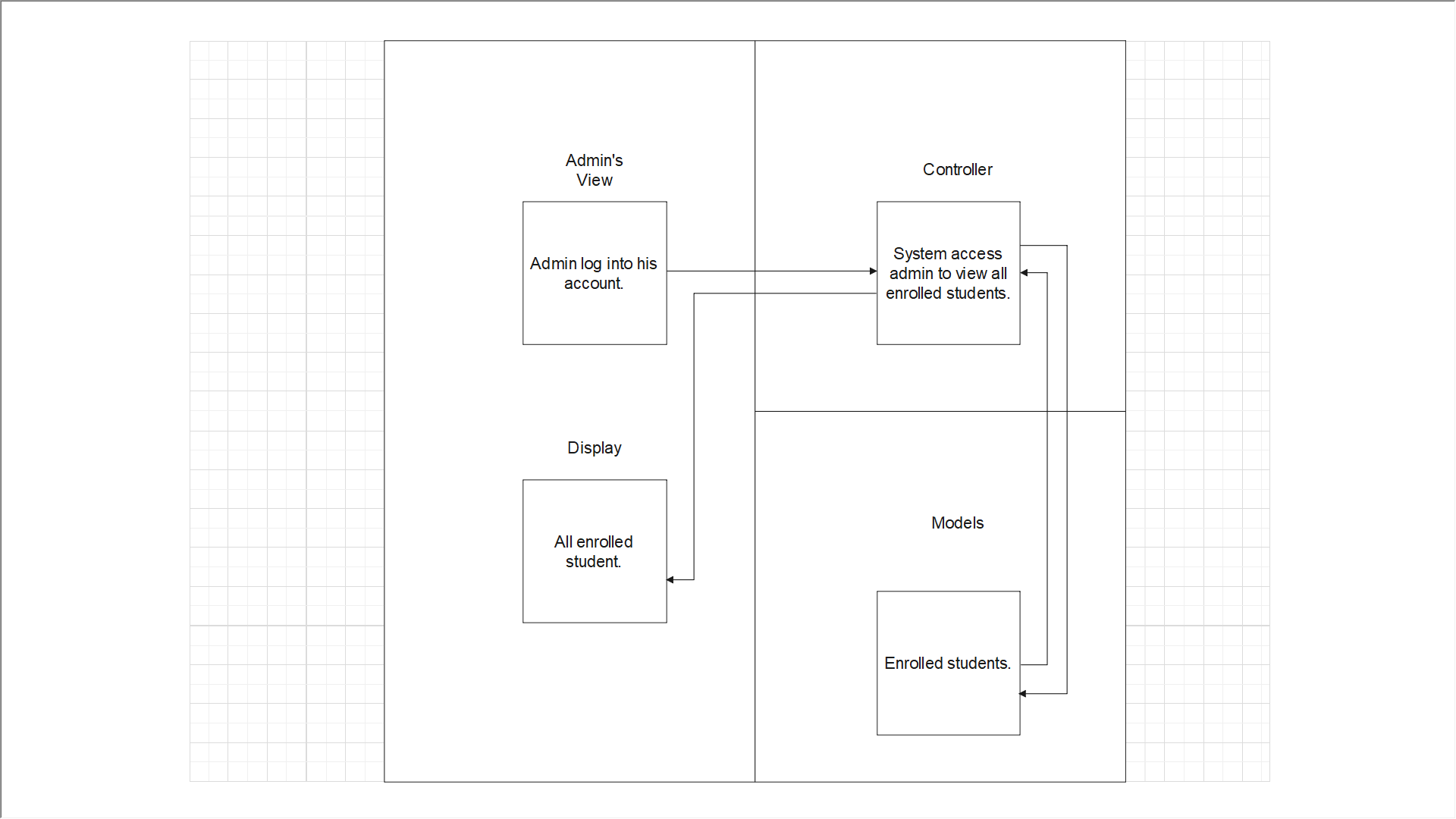
Student receives mail from system:



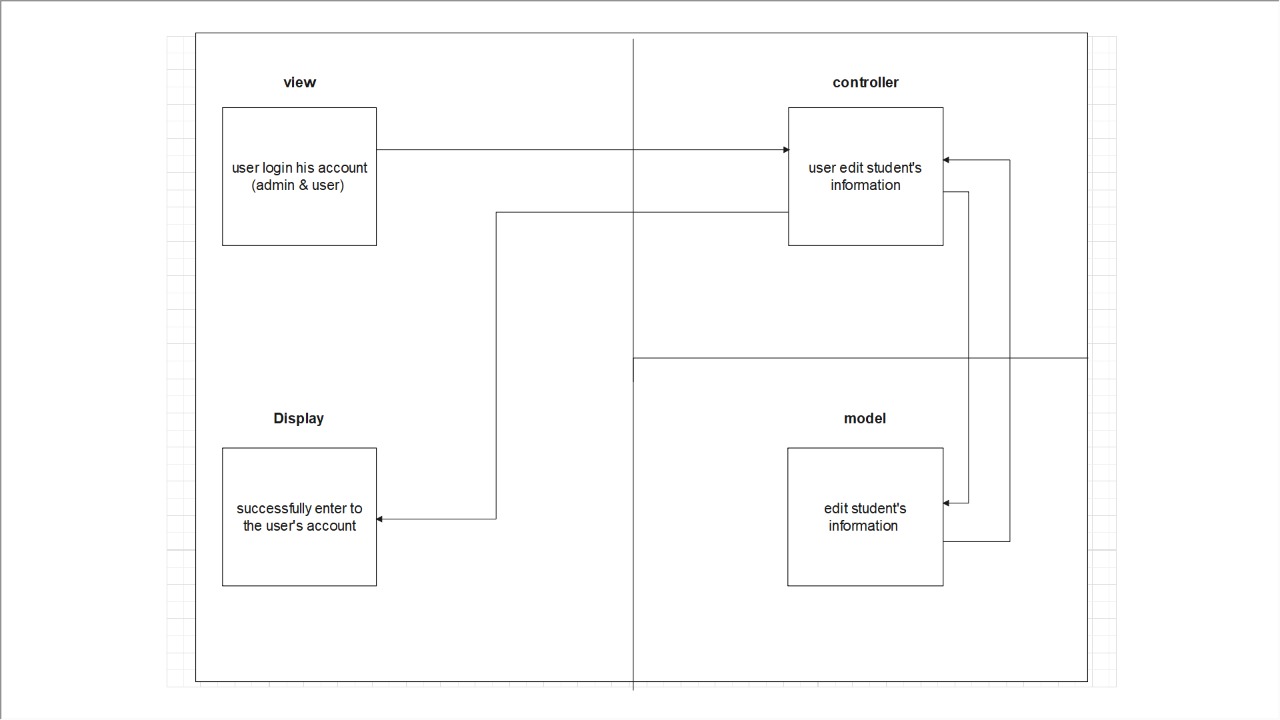
Login:



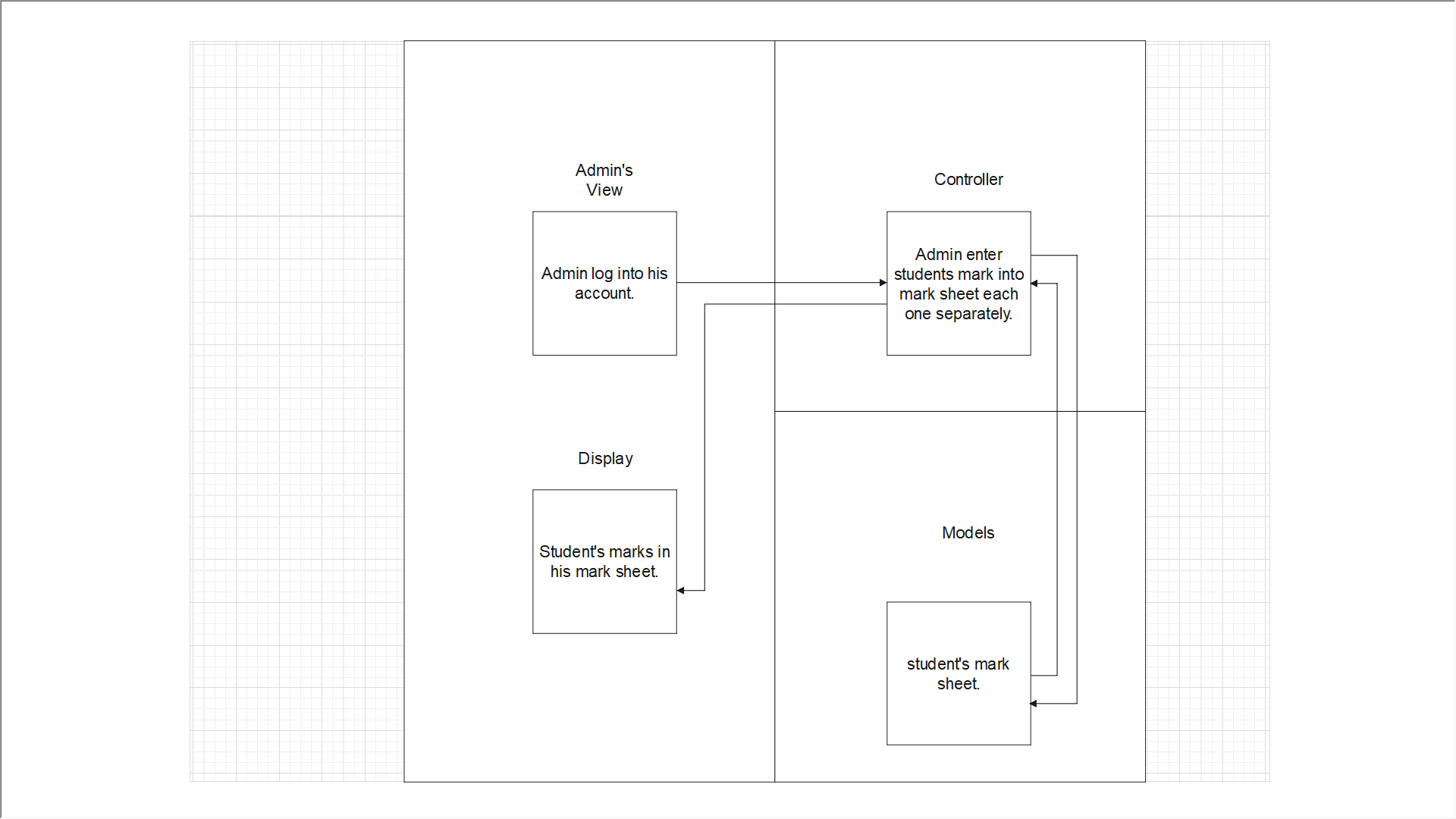
Admin access enrolled students:



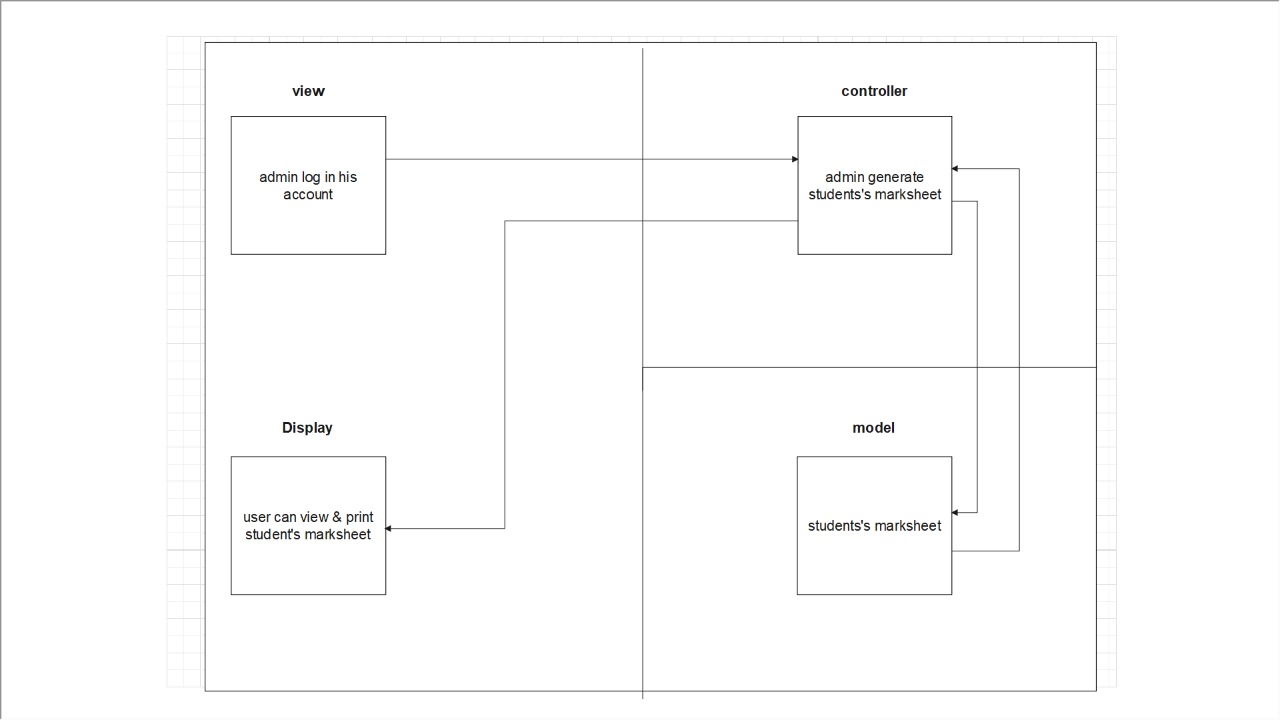
Edit student’s information:



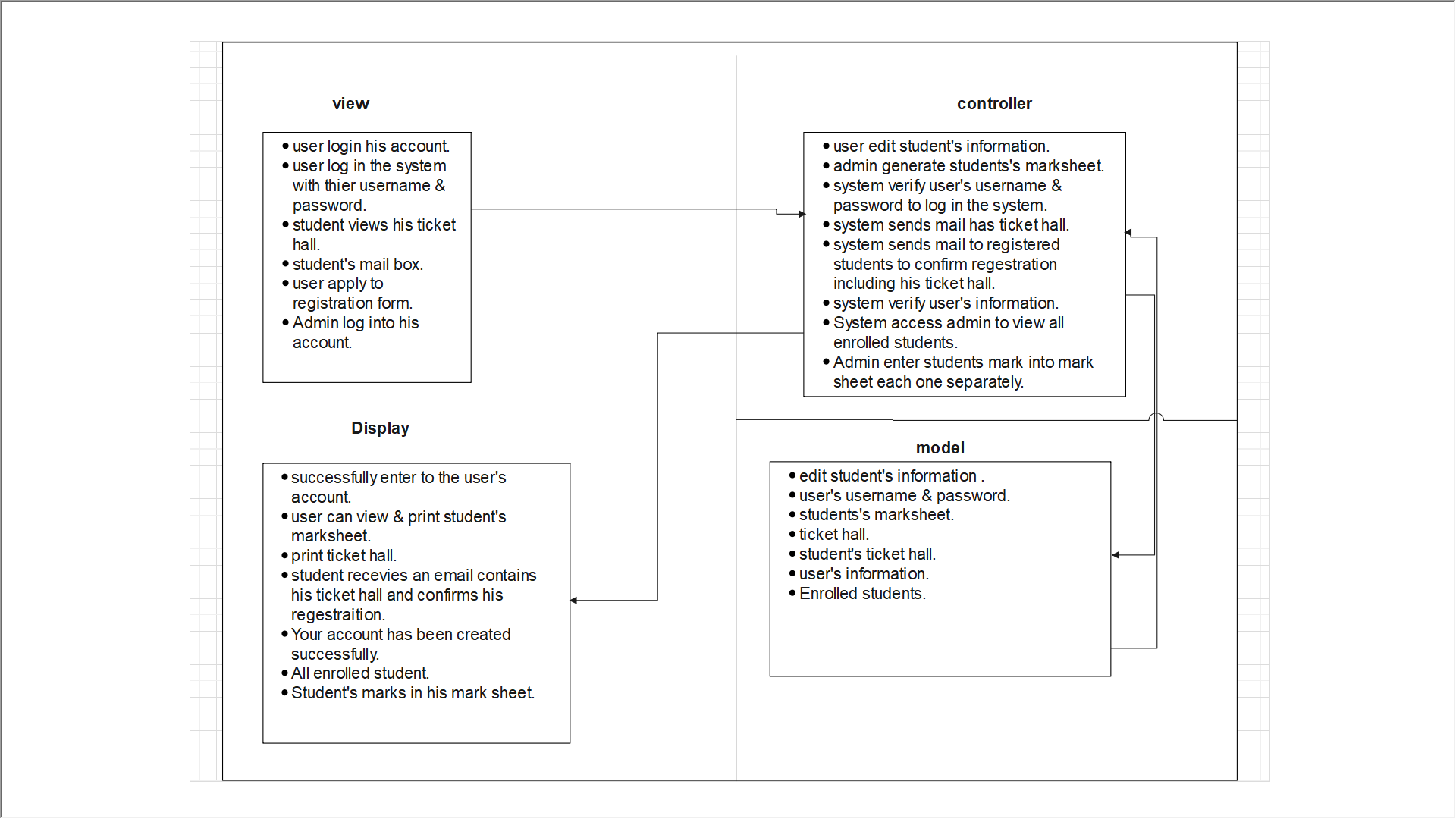
Admin enters marks in mark sheet:



Admin generate mark sheets:



Collected architecture design:



* Collaboration diagram:

Chart

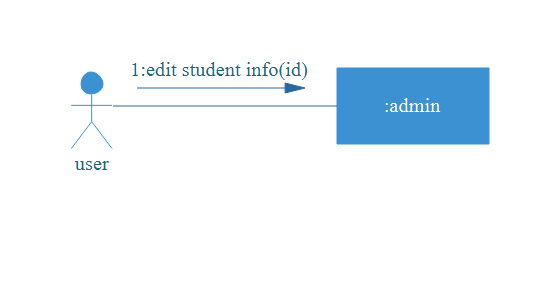
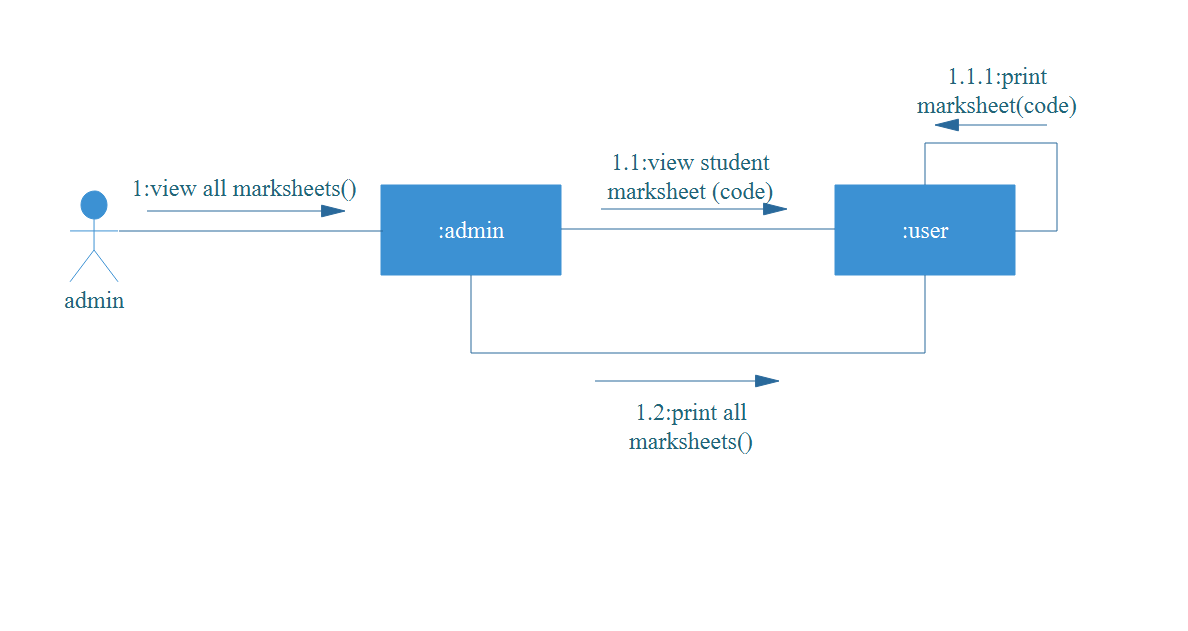
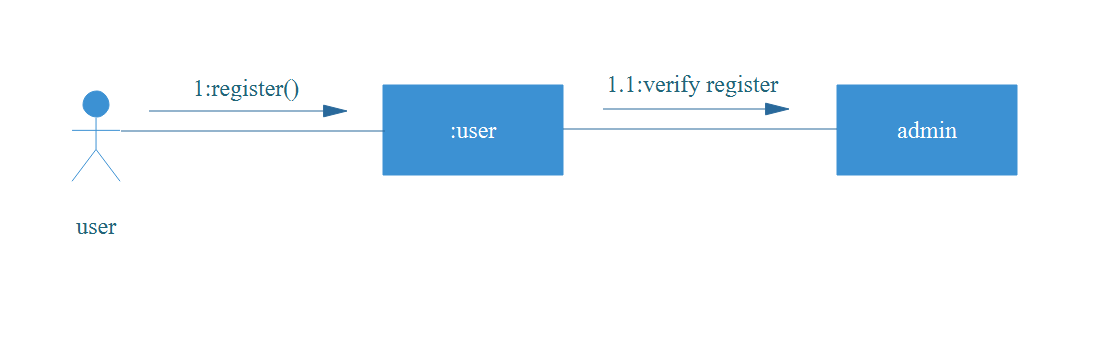
Description automatically generated

Chart

Description automatically generated with low confidenceChart, waterfall chart

Description automatically generatedChart, waterfall chart

Description automatically generatedWaterfall chart

Description automatically generated

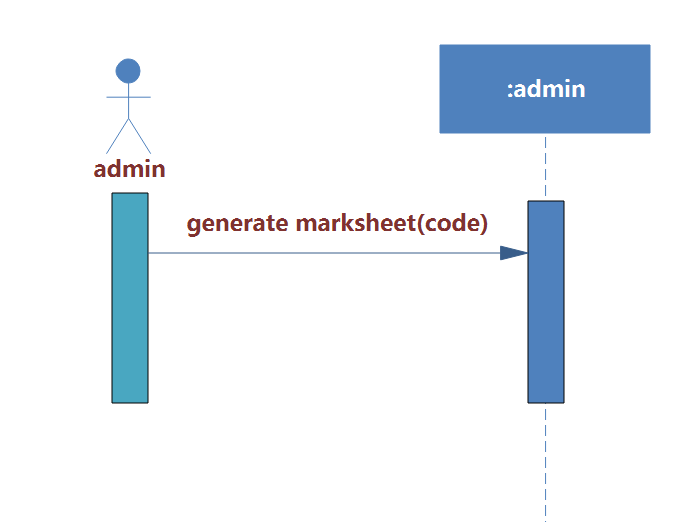
**•** Sequence diagram:

Diagram

Description automatically generatedDiagram

Description automatically generated

Chart, waterfall chart

Description automatically generatedChart

Description automatically generatedDiagram, waterfall chart

Description automatically generatedChart

Description automatically generatedChart

Description automatically generated

System Sequence diagram:

A picture containing diagram

Description automatically generated

A picture containing diagram

Description automatically generated

Chart

Description automatically generated

Chart

Description automatically generated

Chart

Description automatically generatedGraphical user interface

Description automatically generated with low confidenceDiagram

Description automatically generated

Chart

Description automatically generated