Exam Cell Automation

for

<Project>

Version 1.0 approved

Prepared by <author>

<organization>

<date created>

Table of Contents

Table of Contents ii

1. Introduction 1

1.1 Purpose 1

1.2 Product Scope 1

1.3 Actors included 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Constraints 2

3. System Requirements Specification 3

3.1 User Functional requirements 3

3.2 System Functional requirements 3

4. Non-Functional requirements 4

4.1 Performance 4

4.2 Safety and Security 4

4.3 Reliability 4

4.4 Usability 4

5. Use Case Description 5

6. System design 6

6.1 Use Case Diagram 6

6.1.1 Generalized Use Case Diagram 6

6.1.2 Described Use Case Diagram 6

6.2 Activity diagram 6

6.3 Database Specification (ERD, Tables) 6

6.4 Class Diagram 6

1. Introduction

1.1 Purpose

The purpose of SRS is to determine both the functional and the non-functional requirements of exam cell automation and it also provides an overall description about the system with UML analysis models.

1.2 Product Scope

It offers hall tickets for the students and websites that showcase the mark sheets.

1.3 Actors included in the exam cell automation.

Admin	Someone who is responsible for managing the system.
Student	Someone who registers in the system and views marksheets and hall tickets.

2. Overall Description

2.1 Product Perspective

The software described in this SRS is the software for Exam cell automation

2.2 Product Functions

- Admin can view who has enrolled into the system and can see all the new enrollment on his login.

- Students can register by providing their Personal Details like Name, Address, Phone No, etc., and a photo, to the enroll themselves into the System.

- Students can use their credentials provided by admin to login into the system.

- System allows admin to generate and print mark sheet of examination for every student.

- System allows registered students to view and modify/update the personal details like Phone number, email-id, etc.

- System mails the link of soft copy to every student who have registered after creating the hall ticket.

2.3 User Classes and Characteristics

Admin can approve authorized students after registration and add them to the system.

2.4 Constraints

- Only authorized users can access the system.
- The student has a maximum limit of 7 subjects for each semester.
- The students cannot graduate if they spent less 3 semesters.
- The students can spend 6 semesters at max.

3. System requirement specification

3.1 User Functional requirements

the exam cell automation system shall allow only registered students to receive their login details, view and print hall tickets and update their personal details.

3.2 System Functional requirements

1. Students can register up to seven subjects for each semester

2. there is from three to six semesters in total to register for them

3. Students can complete register only after providing their personal information.

4. Student have to pass subjects of each semester to can up to the next semester

5. System allow only registered students to benefit from system features on login with id and password provided by admin to prevent unauthorized/malicious access to the system.

6. Admin can’t add multiple students with the same id (each student has a unique id) and only

7. Registered students should be added/recorded into the system.

8. Students can update their personal details only after login to the system and the hall ticket which can be shown and printed after login and also from the received link.

9. Admin is allowed to create a mark sheet for each student (who are already recorded in the system/DB) , add marks to mark sheet and print mark sheet for every student separately .

10. Students can view their mark sheet after login and only after the admin create a mark sheet for them and add grades for all subjects they have registered for.

4. Non-Functional requirements: -

4.1 Performance

1- The system should be compatible with all modern browsers.

2- The system should response the operational messages to the users within 2 seconds

3- The system should be reliable.

4.2 Safety and Security

1. Hashing technology should be used to handle the secure login for users.

2. uninterrupted access to data & functionality by authorized users

3. Protection of data from unauthorized access

4.3 Reliability

1. Providing catching of exceptions so that unintended results do not occur such as system crashes or data validation failures.

4.4 Usability

1. The system should have a user-friendly interface.

2. The product shall be easy to use by members of the public without training.

5 Use Case Description:-

Identifier and name : exam cell automation

Initiator : student

Goal : student get examination hall ticket for the registered subjects to take the exams

Precondition : The student should Provide personal details and subject

Postcondition : student get hall ticket and login details (id and password) to login to the system

Assumption : initiator is using web browser to perform the use case. The student isn't already known to the system.

Main success scenario:

1- System allows students to register for subject.

2- Students starts providing their personal details.

3- student register from 3 to 6 semester and maximum 7 subjects for each semester.

4- admin adds new registrations to the system.

5- the system creates hall tickets.

6- the system mails the students with link contains login information and hall ticket.

7- the student login with id and password provided in the received the Email.

8- student can update their personal details or register for another subject.

9- student can view and print hall ticket.

10- admin select student, create mark sheets and add marks.

11- students can view their mark sheets.

Extensions:

2.a student already registered before with the same personal details

2.a.1 the system print error message that the student already registered

7.a student enter invalid id or password

7.a.1 system print Error message to re-enter id and password or use forget password option

7.a.2 student enter a valid id and password or reset the password

8.a student register more than seven subjects

8.a.1 system print error message that student cannot register for more than seven subjects for each semester

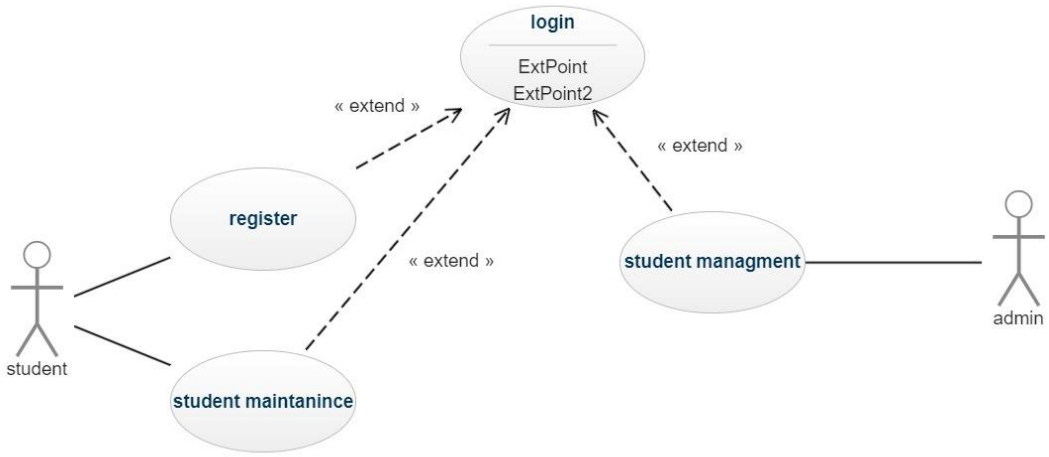
11.a student view mark sheet which is not created by the admin yet.

11.a.1 system print error message that results are not published yet

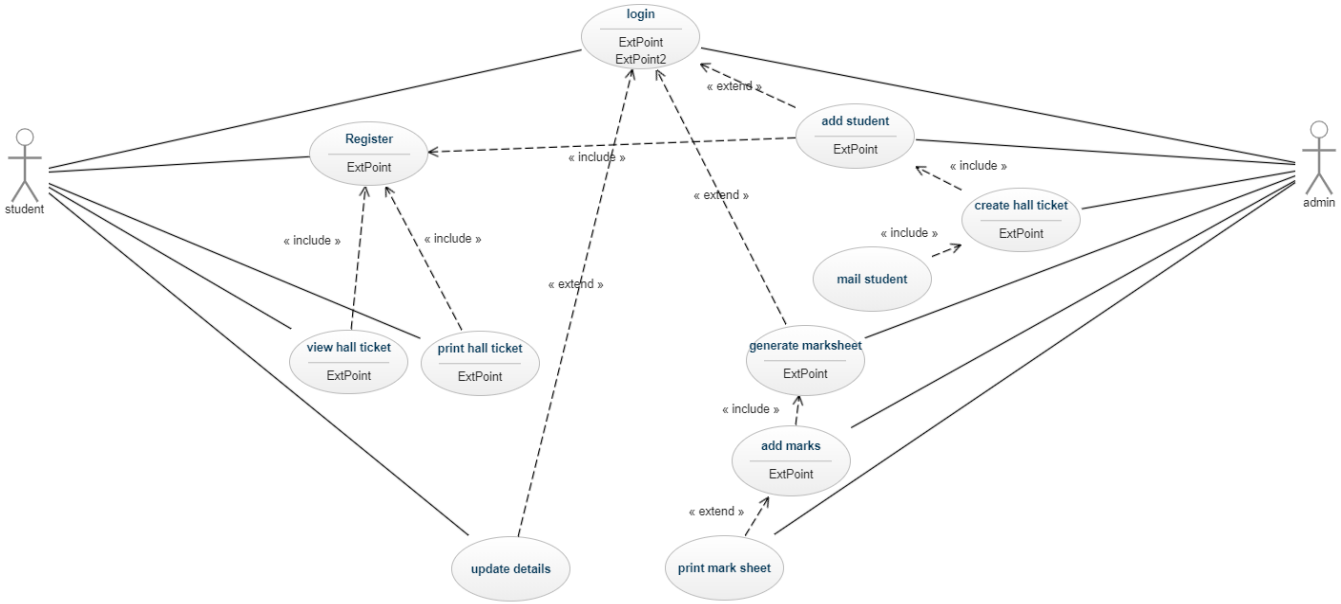
6. System Design

6.1 UML Diagrams

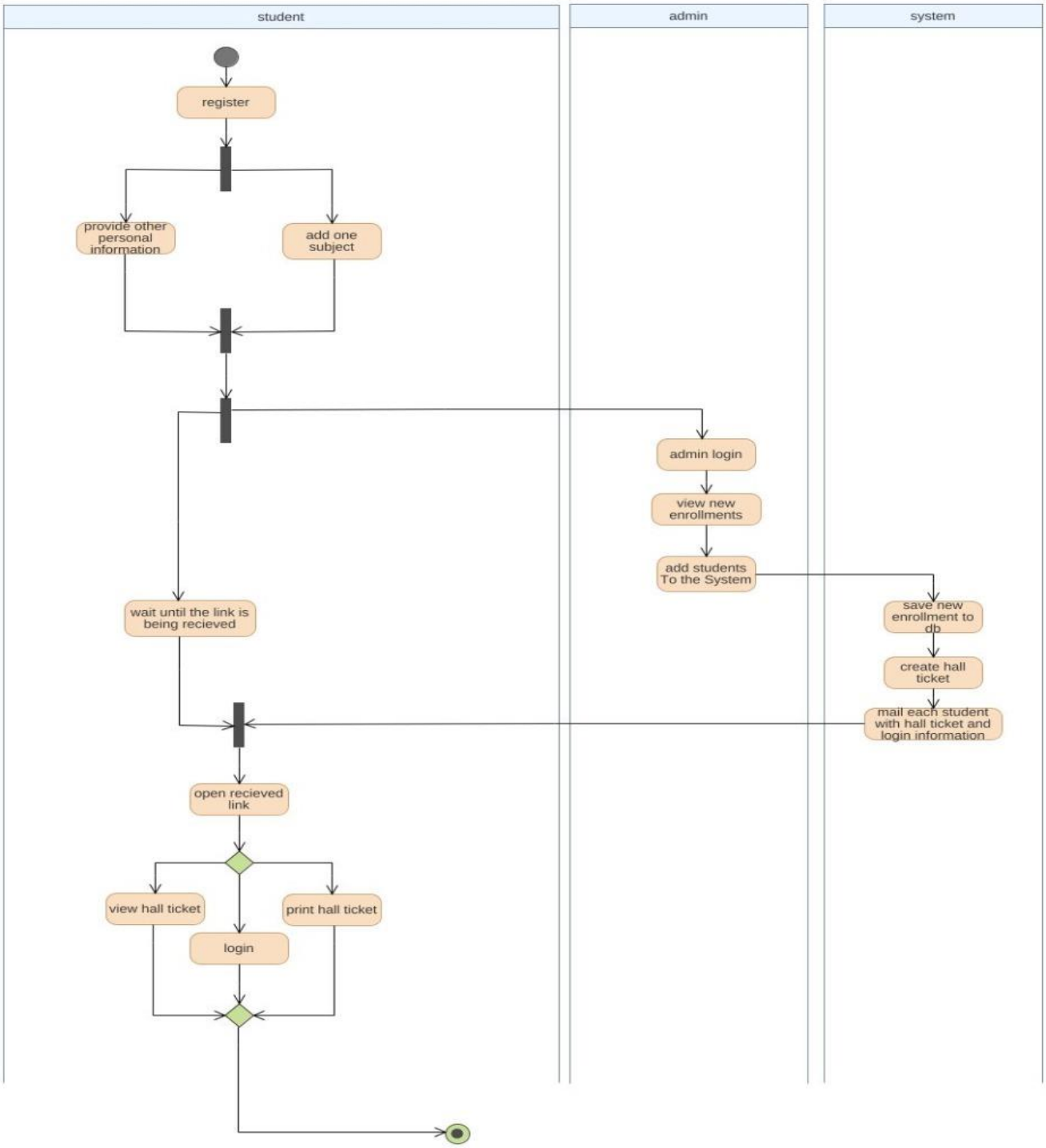
1) **Generalized use case Diagram of exam cell automation**

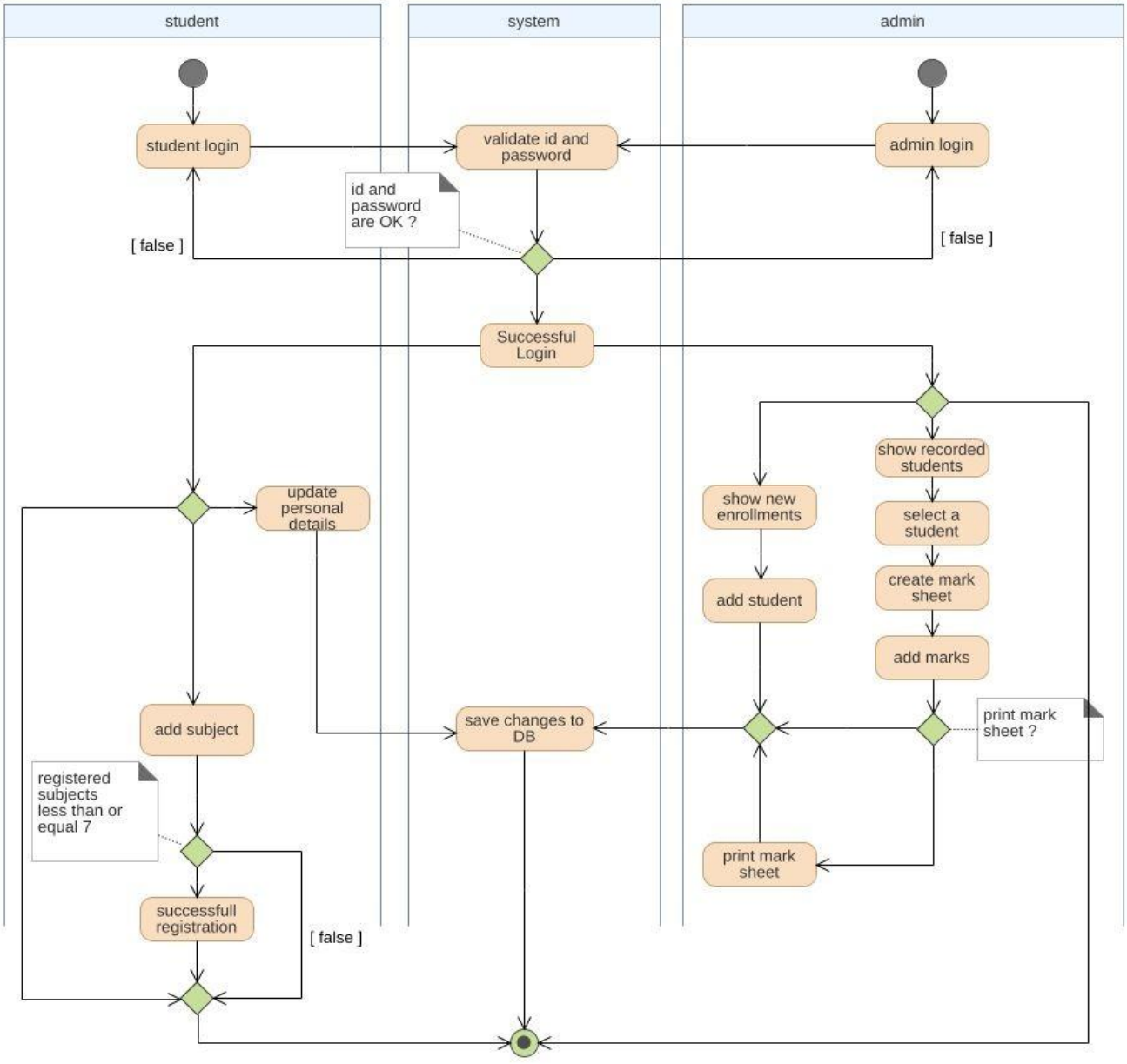


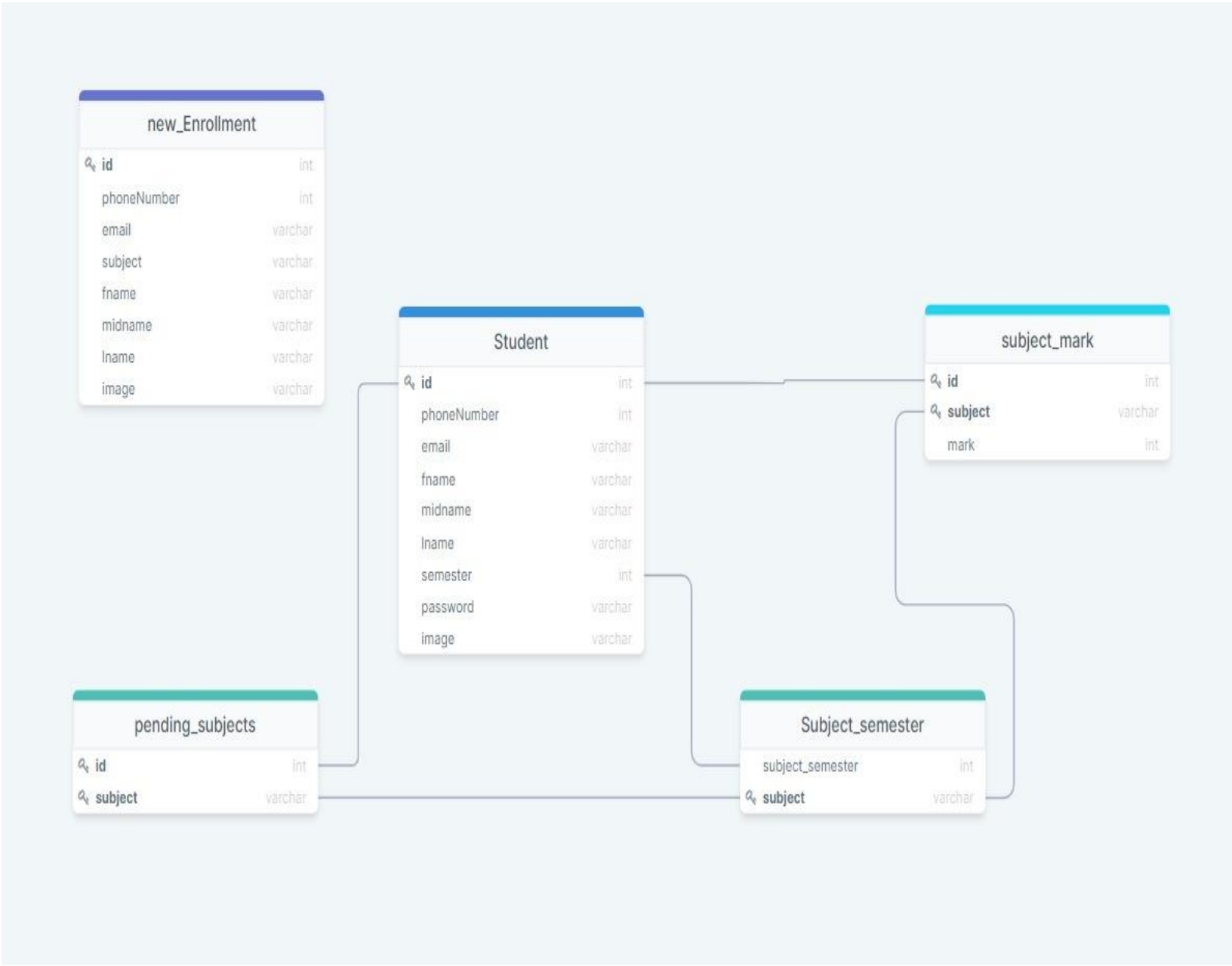
2) **Described use case Diagram of exam cell automation**



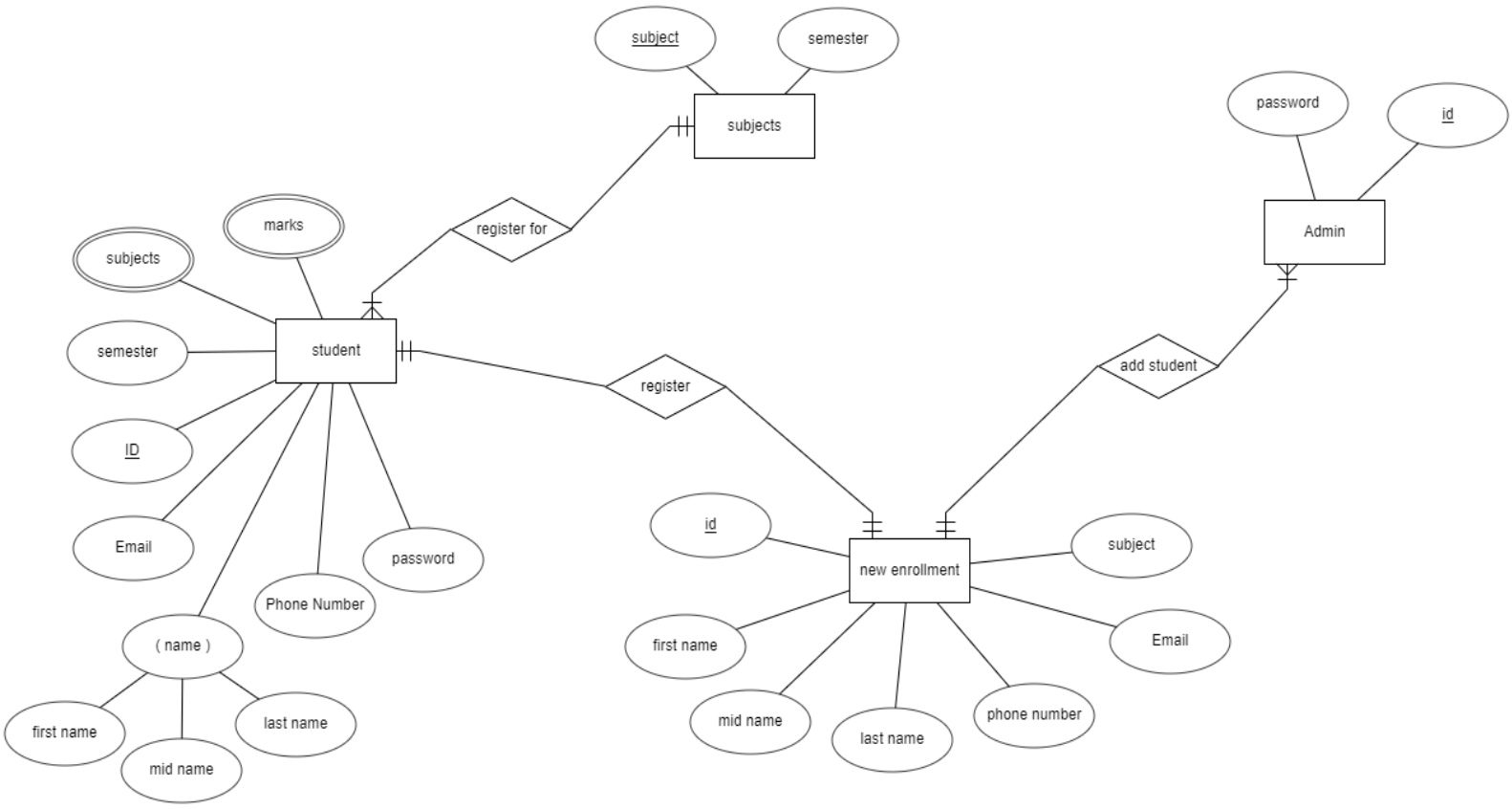
2) Activity diagrams



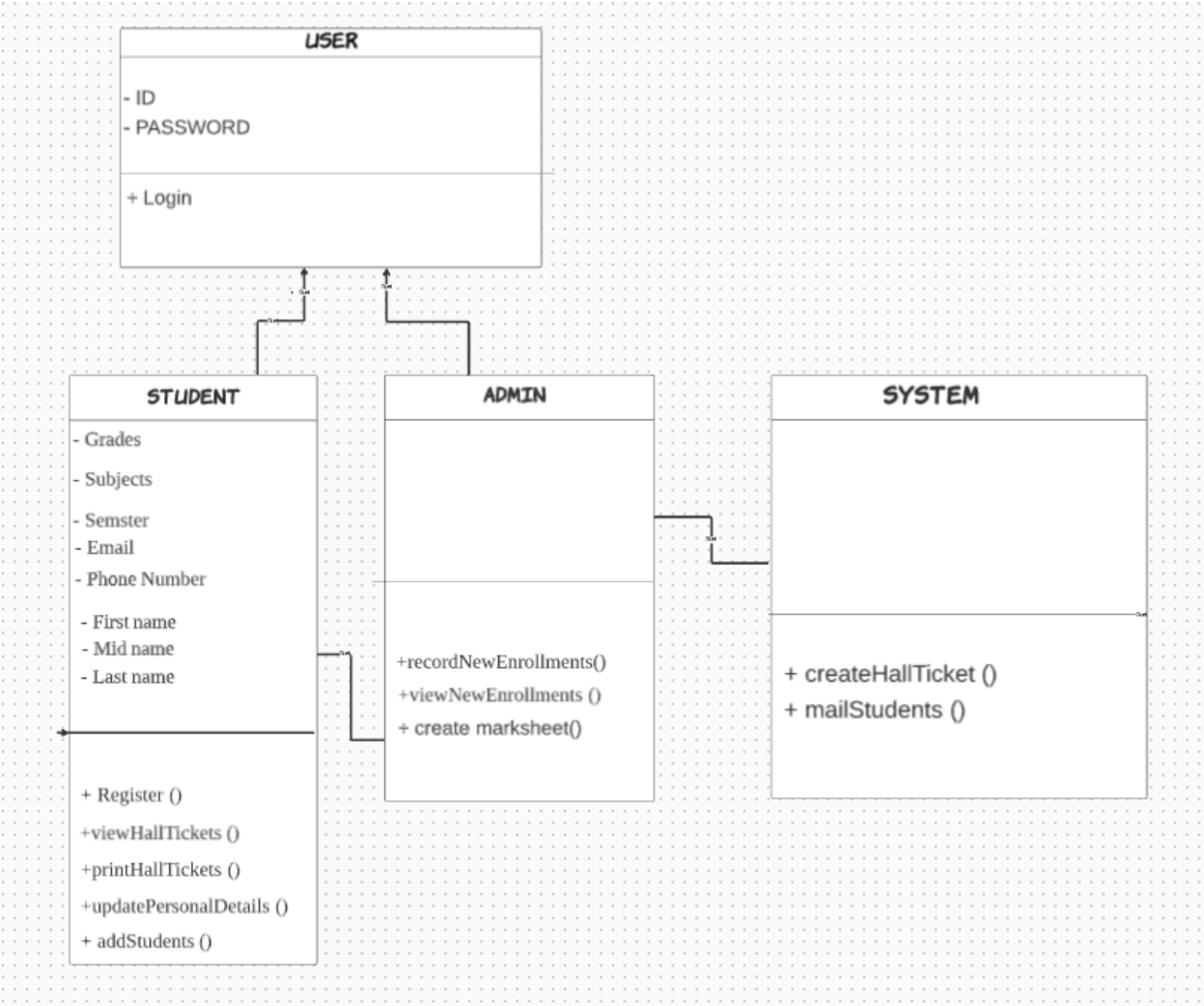

student
system
admin
student login
validate id and password
admin login
id and password are OK ?
[false]
[false]
Successful Login
show recorded students
update personal details
show new enrollments
select a student
add student
create mark sheet
add marks
add subject
save changes to DB
print mark sheet ?
registered subjects less than or equal 7
print mark sheet
successfull registration
[false]



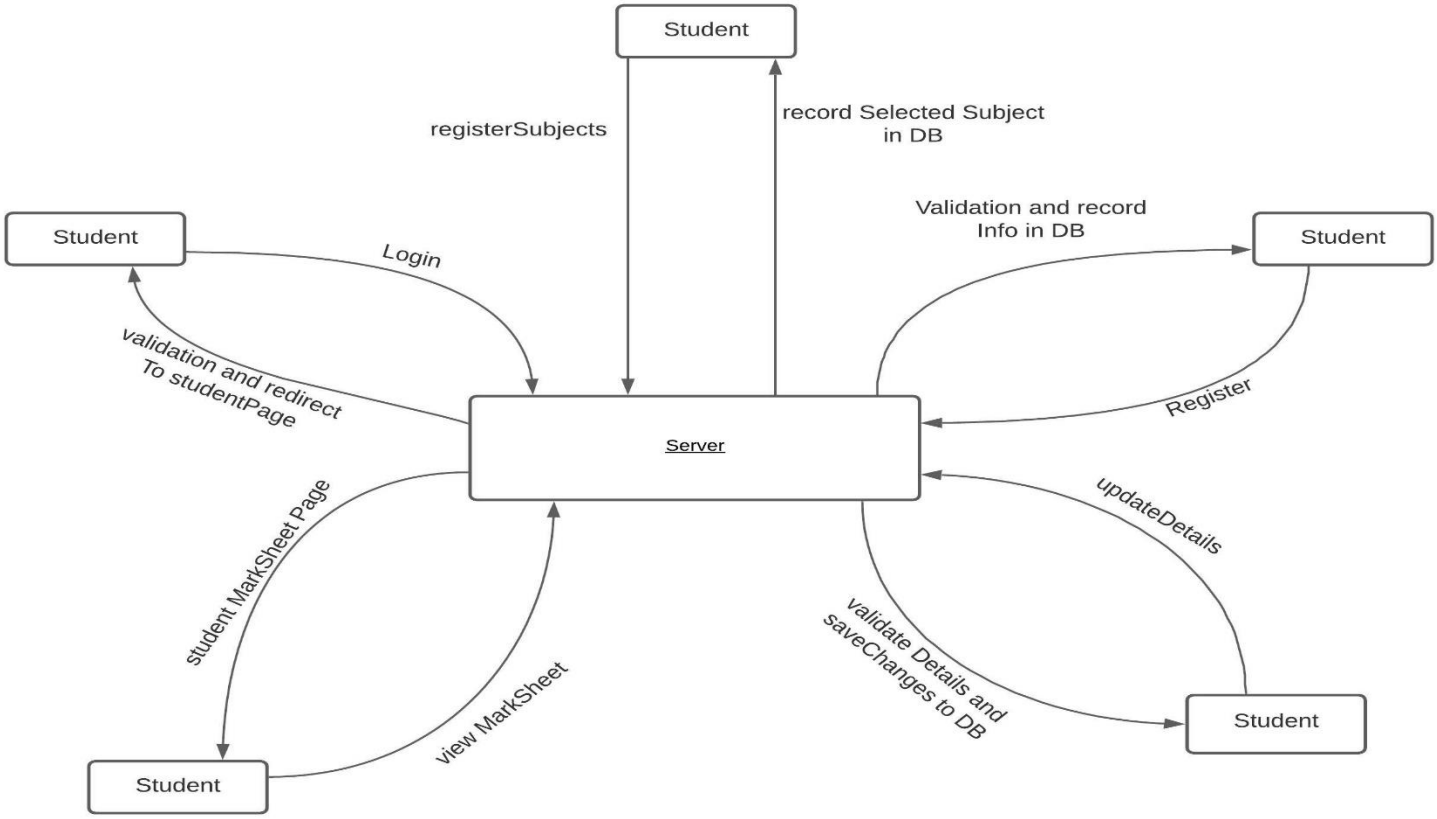
3) **Database specification (ERD, Tables)**

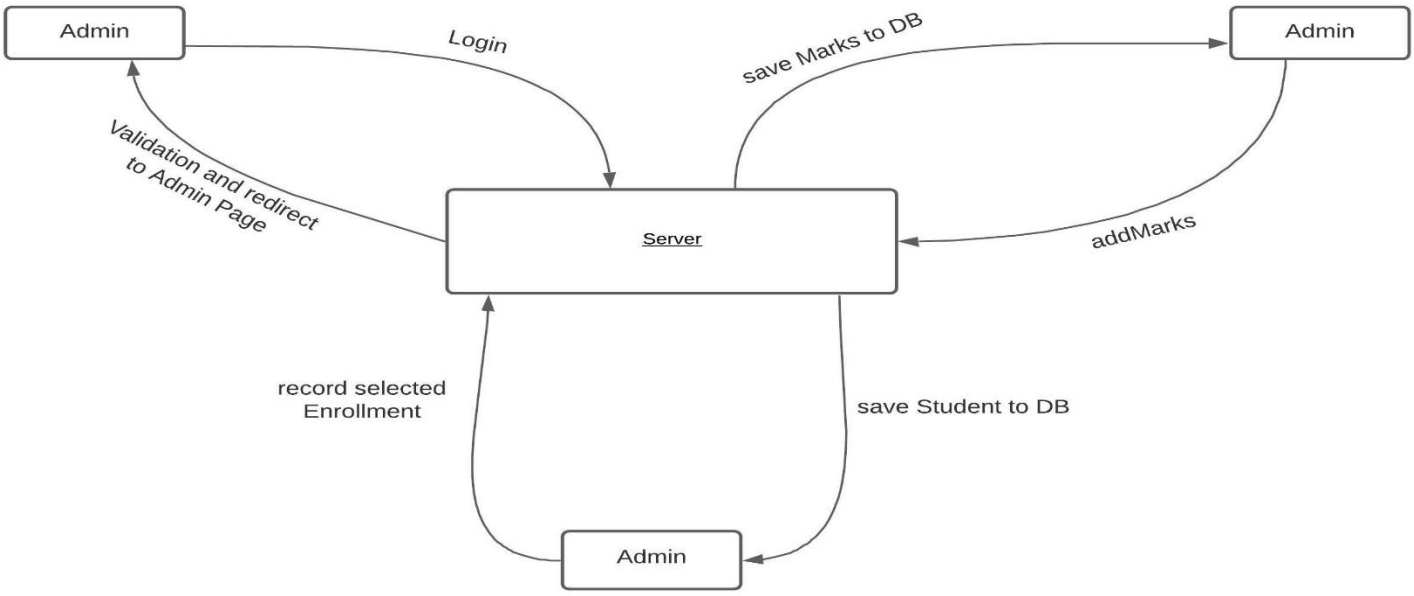

subject
semester
subjects
password
id
Admin
marks
register for
subjects
semester
student
register
add student
ID
Email
password
Phone Number
(name)
first name
mid name
last name
id
new enrollment
subject
first name
Email
mid name
last name
phone number

4) Class Diagram

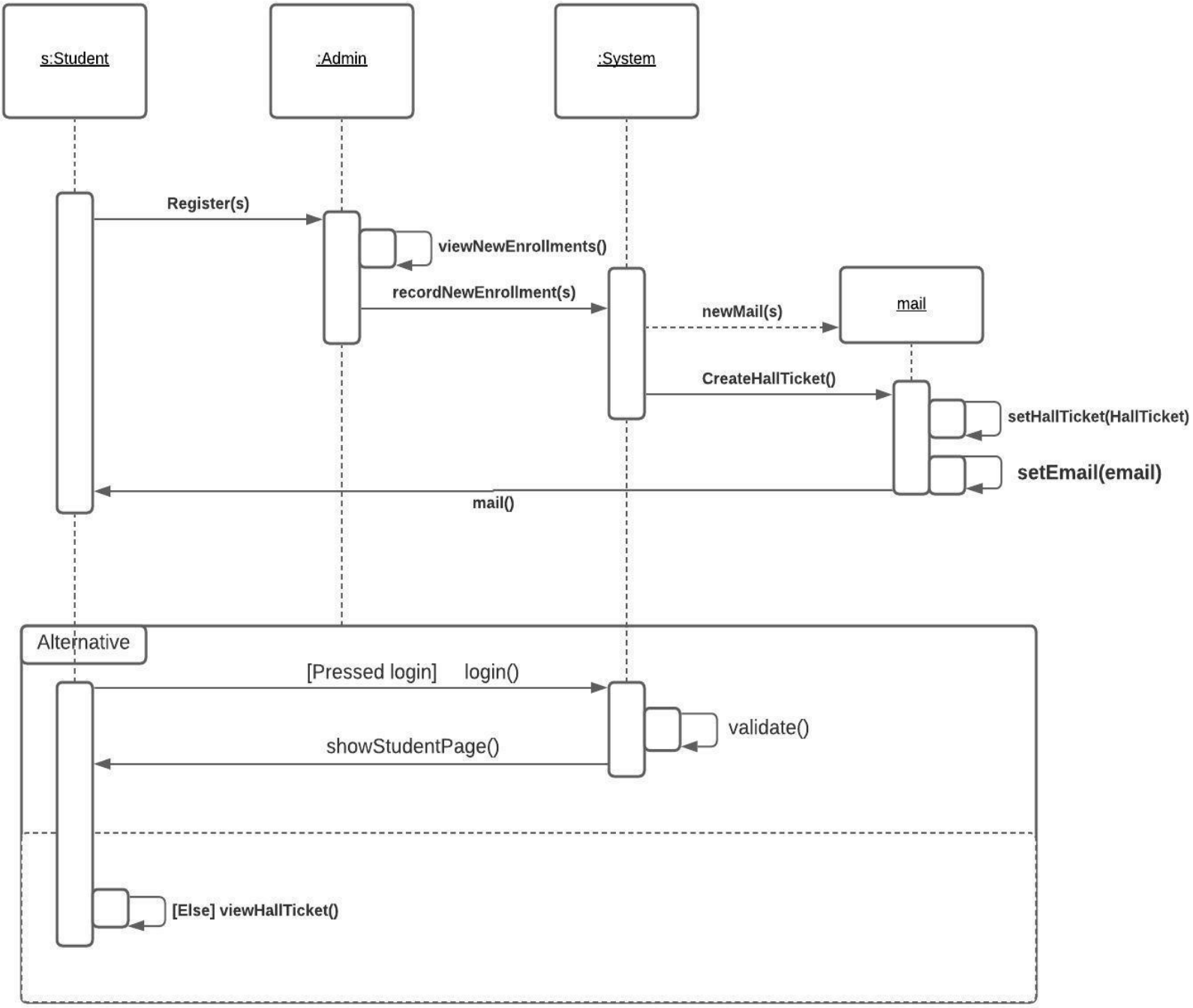


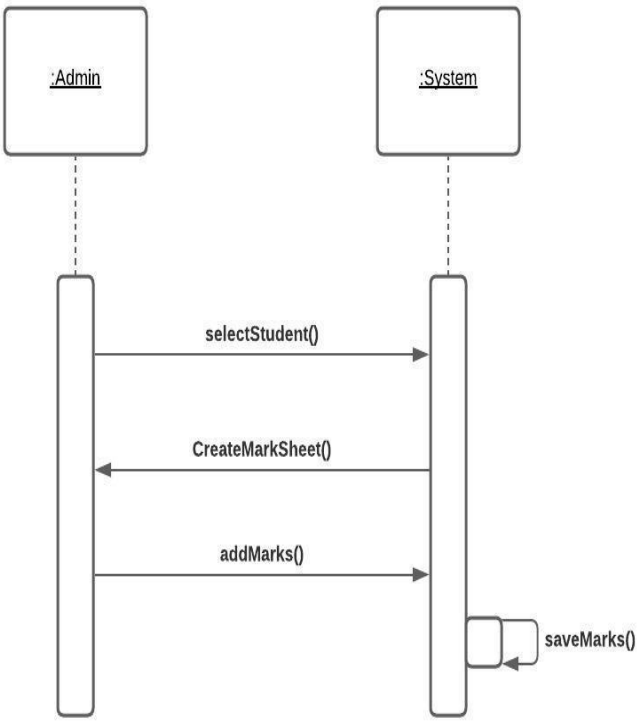
System Arcitecture

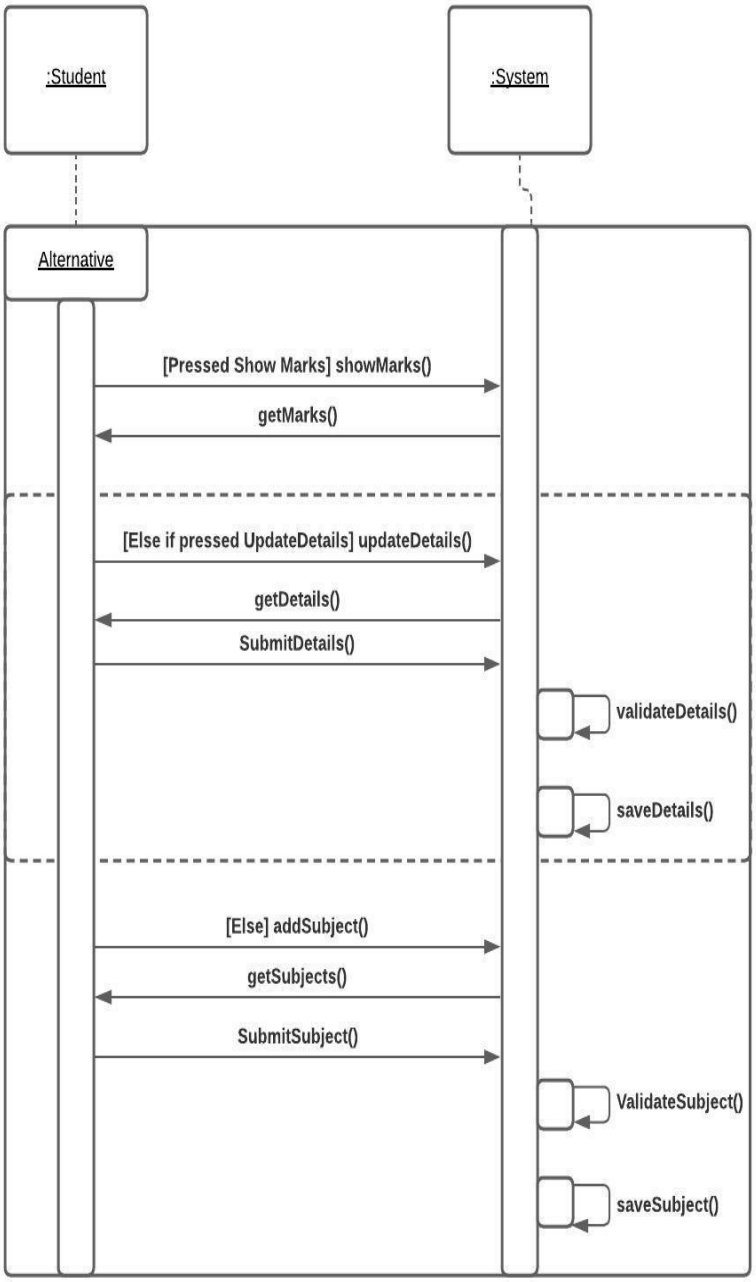




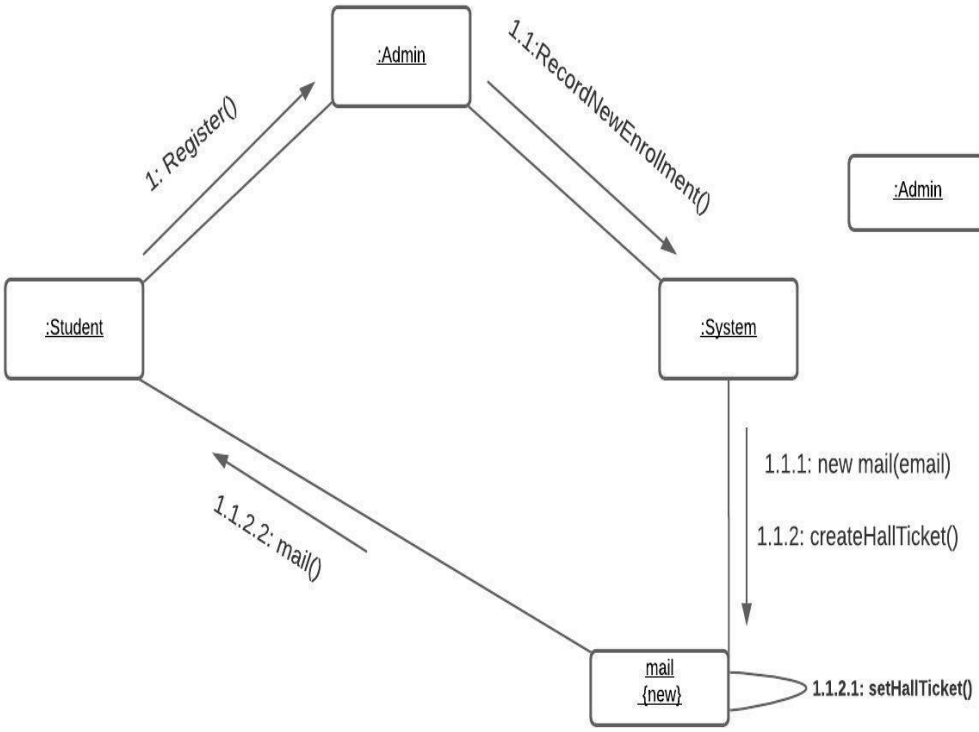
Sequence Diagram

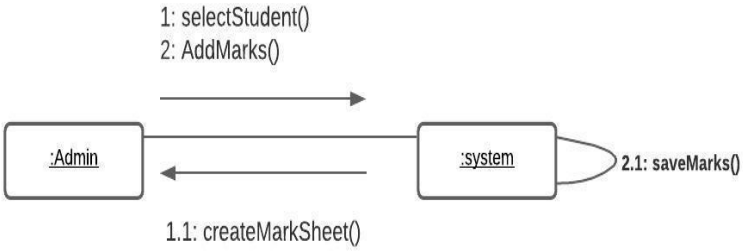


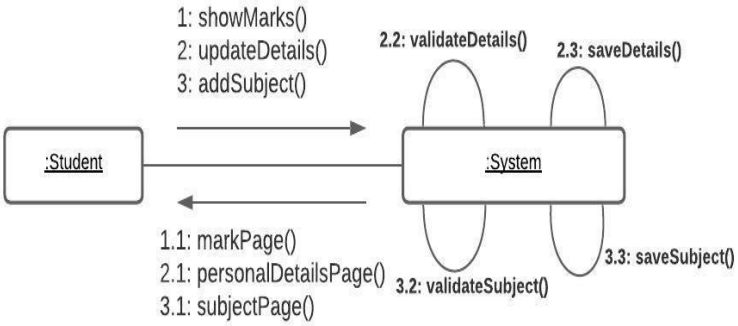

:Admin
:System
selectStudent()
CreateMarkSheet()
addMarks()
saveMarks()


:Student
:System
Alternative
[Pressed Show Marks] showMarks()
getMarks()
[Else if pressed UpdateDetails] updateDetails()
getDetails()
SubmitDetails()
validateDetails()
saveDetails()
[Else] addSubject()
getSubjects()
SubmitSubject()
ValidateSubject()
saveSubject()

Collaboration Diagram

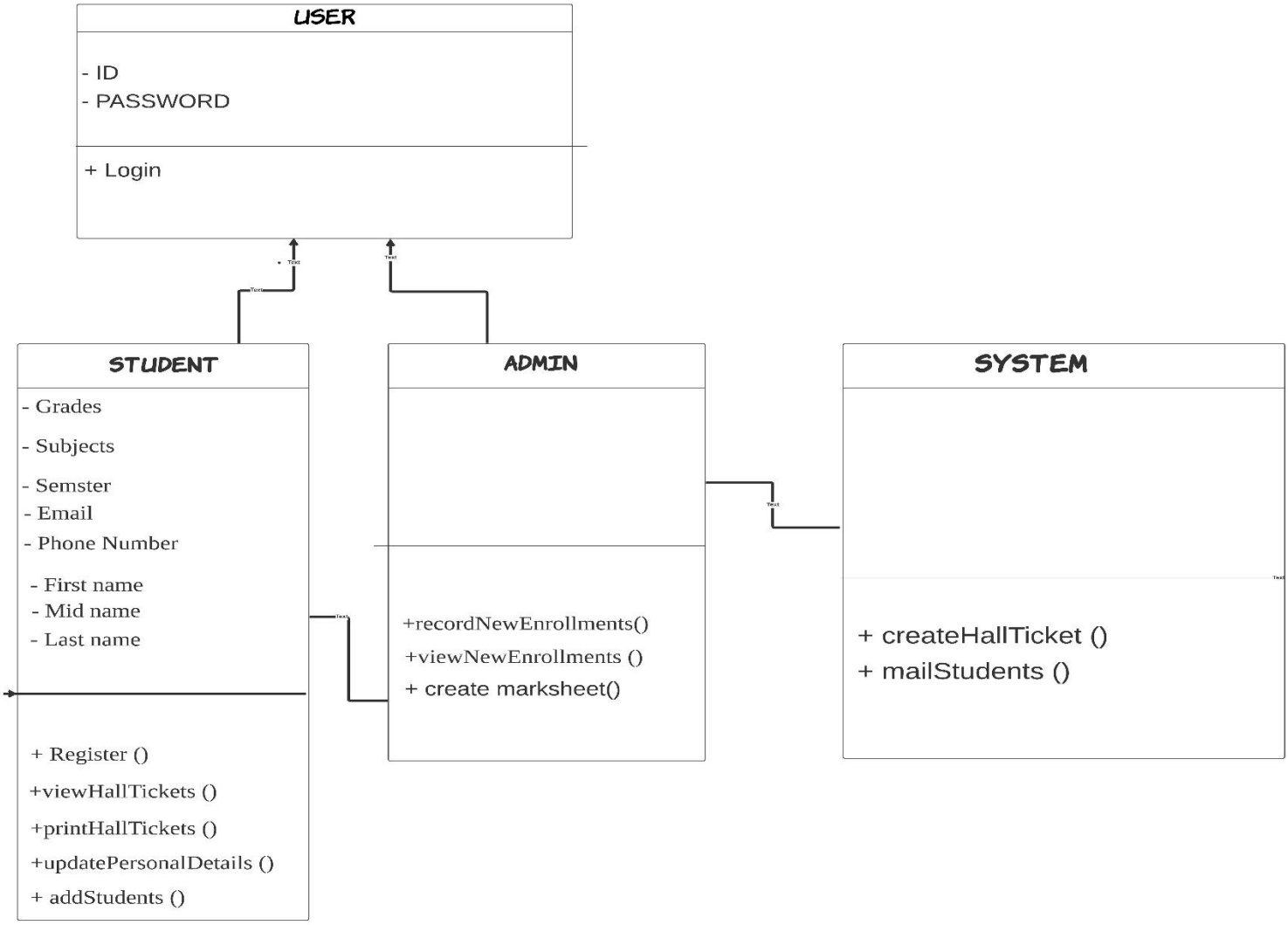




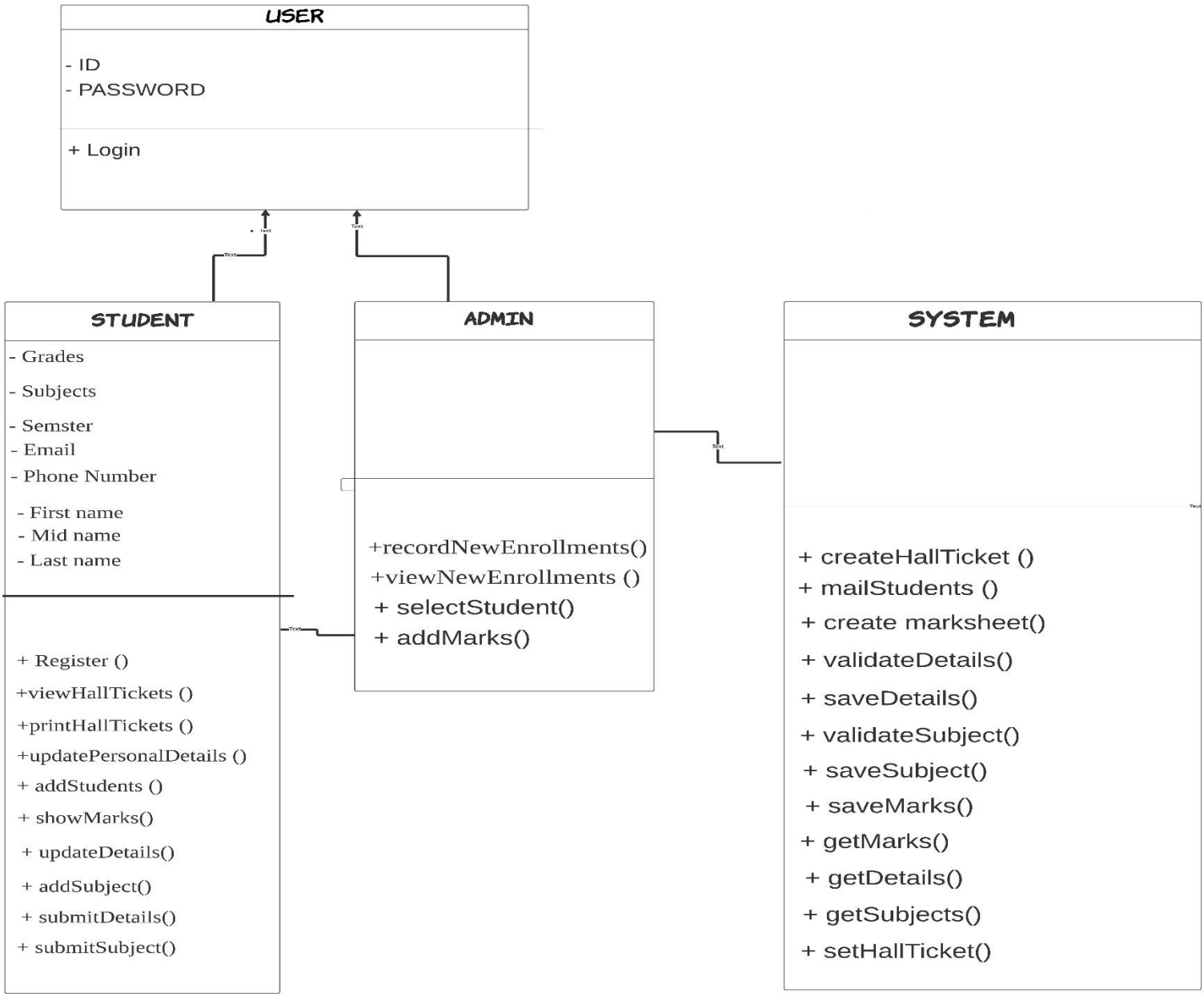


Class Diagram

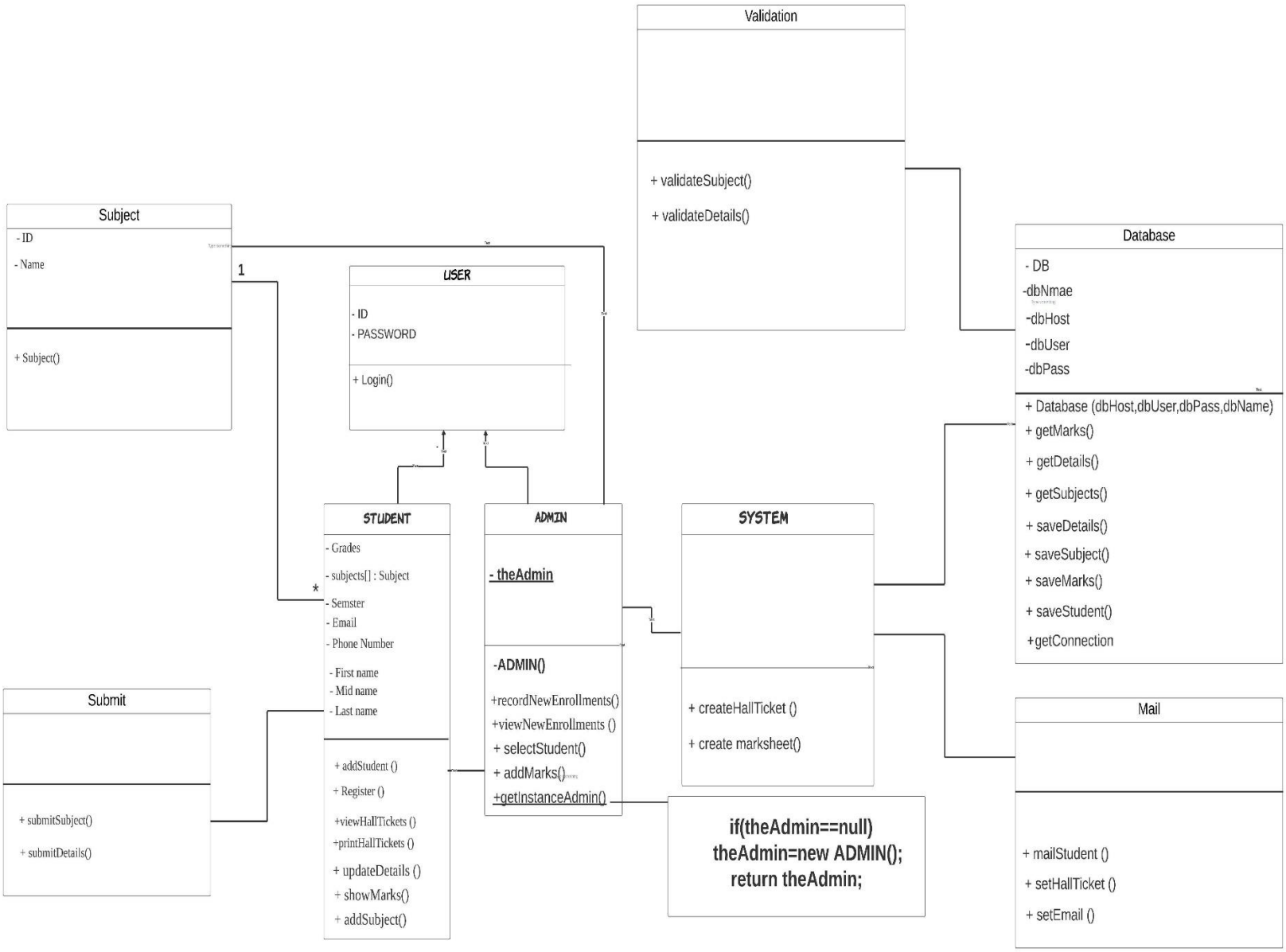
1- Initial Version



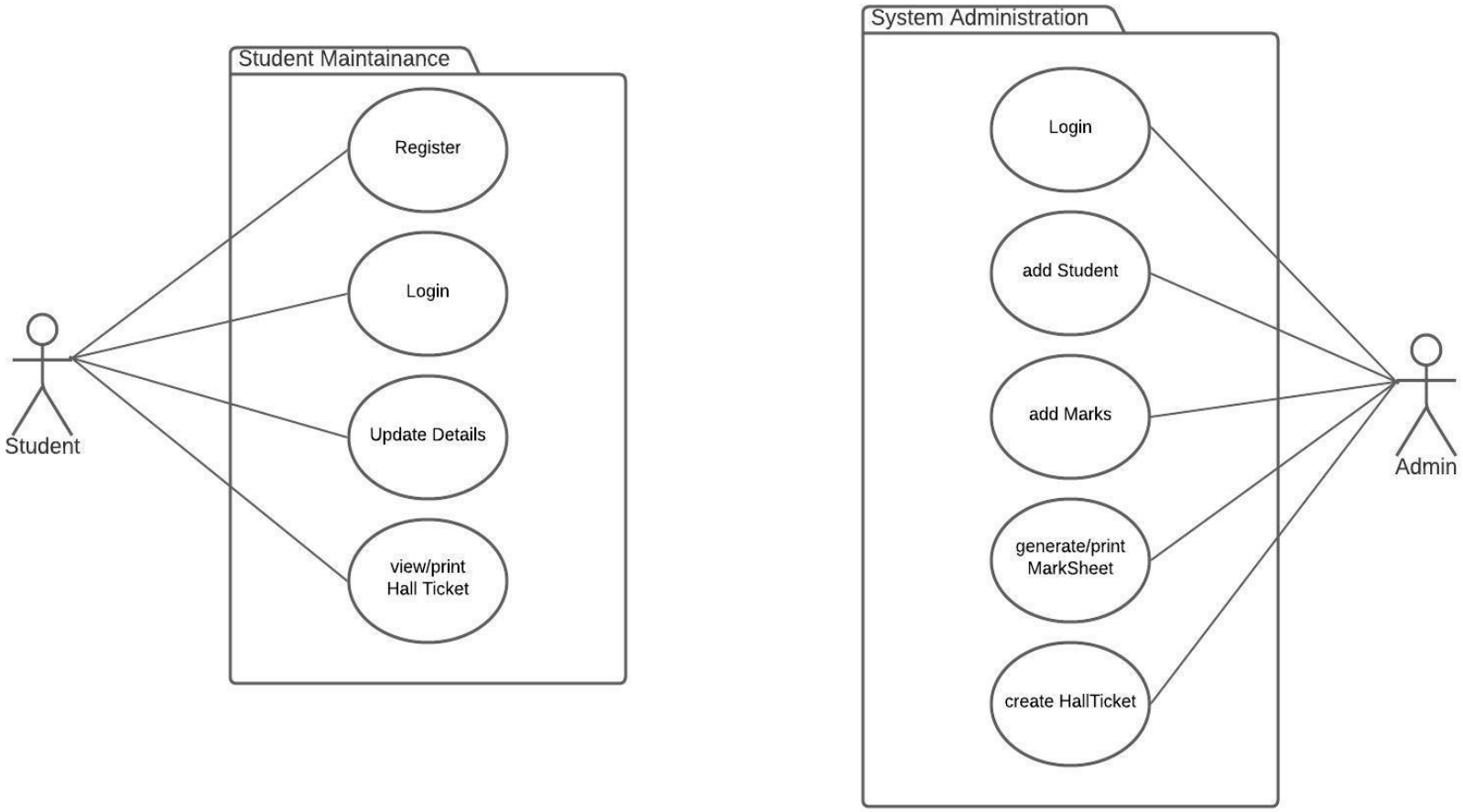
2- Intermediate Version



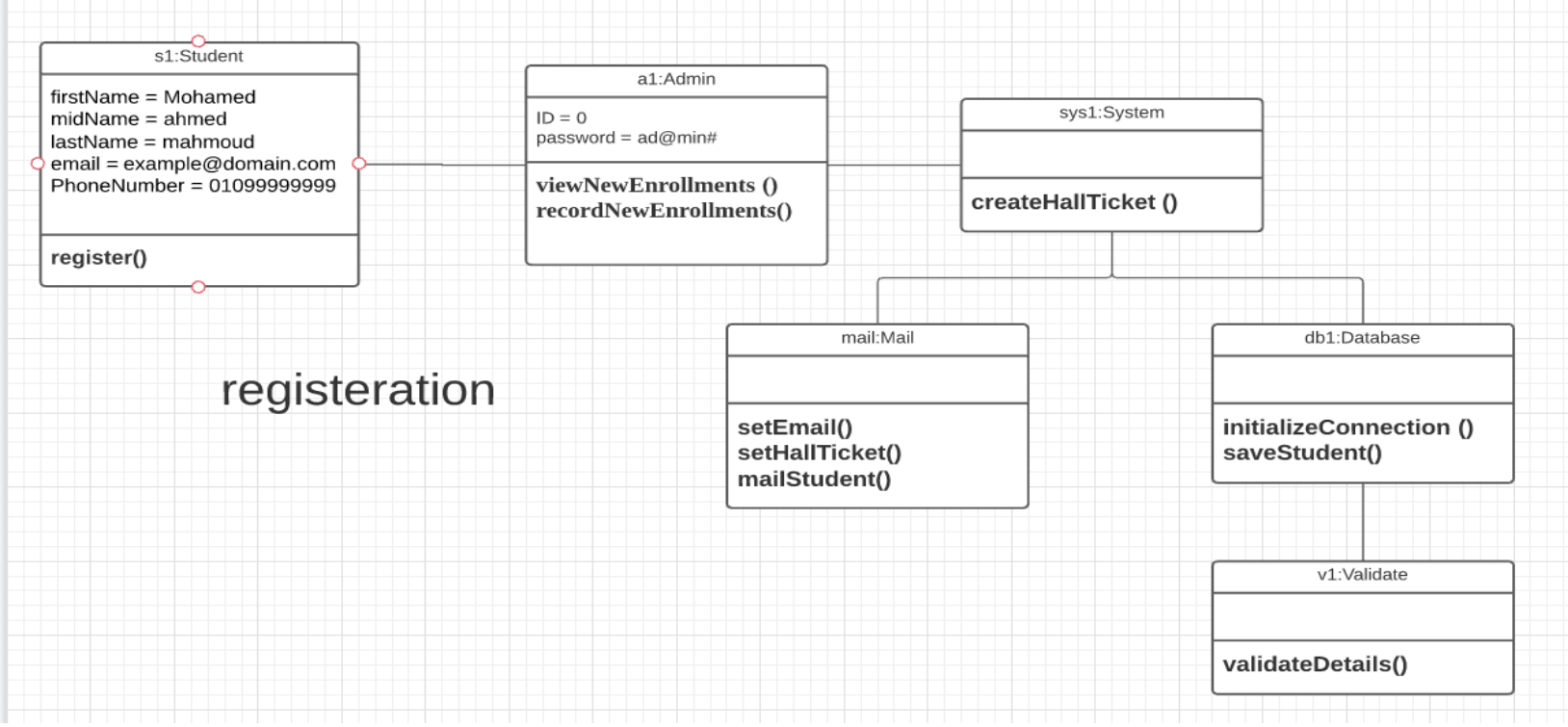
3- Final Version



Package Diagram

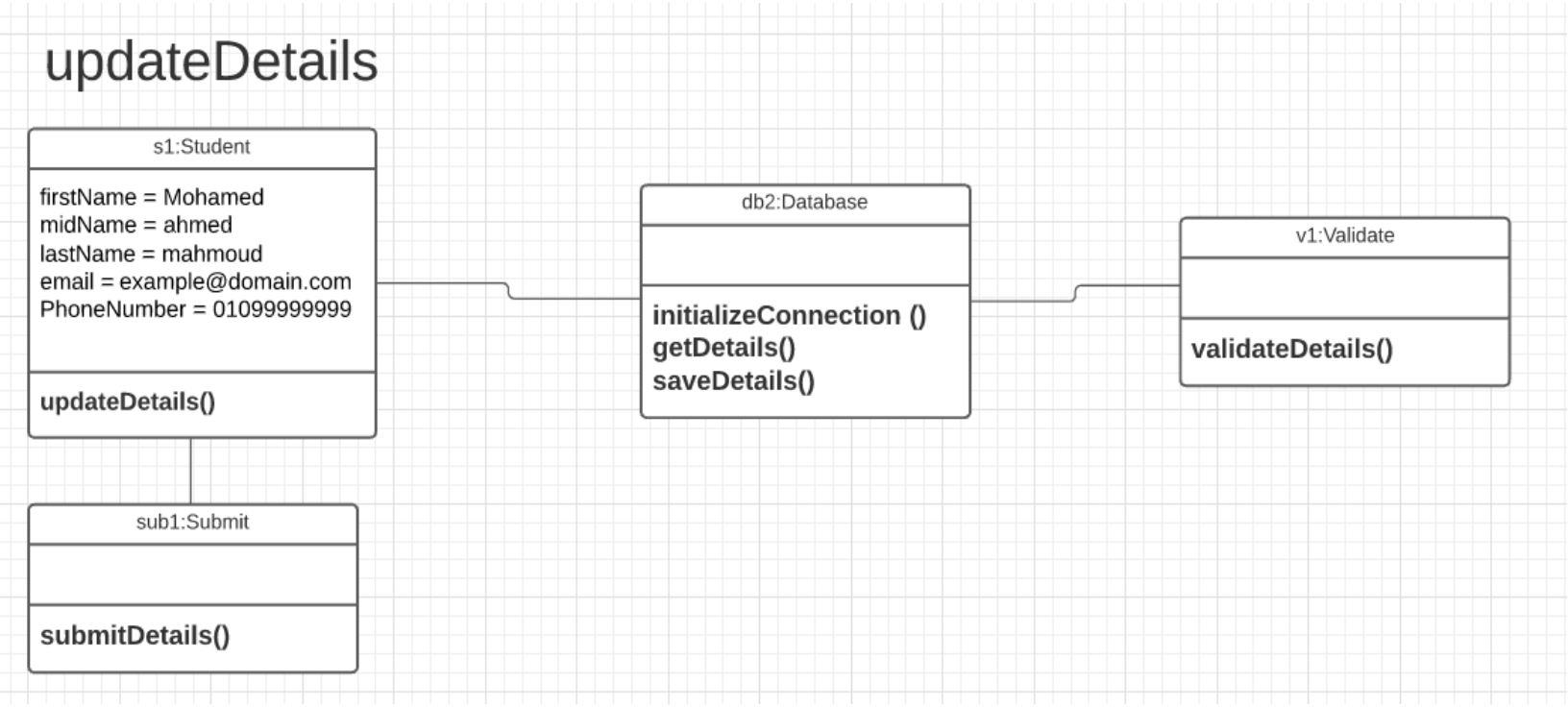


Object Diagram



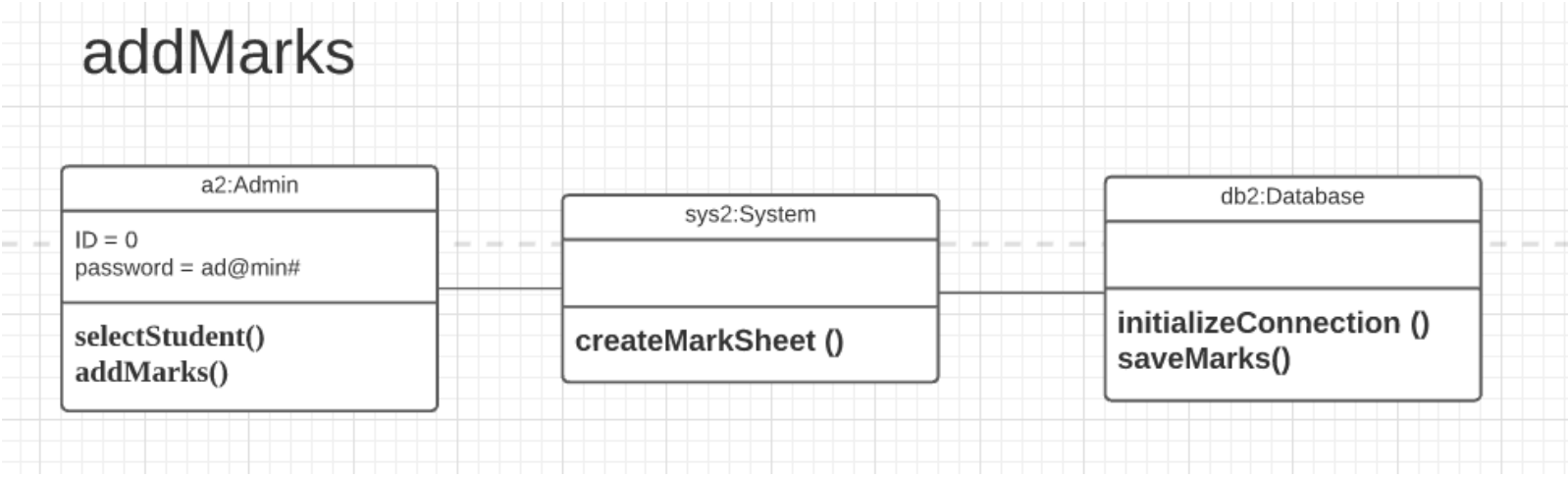
Precondition : the student should provide personal details and select subject

Postcondition : student get hall ticket and login details (id and password)



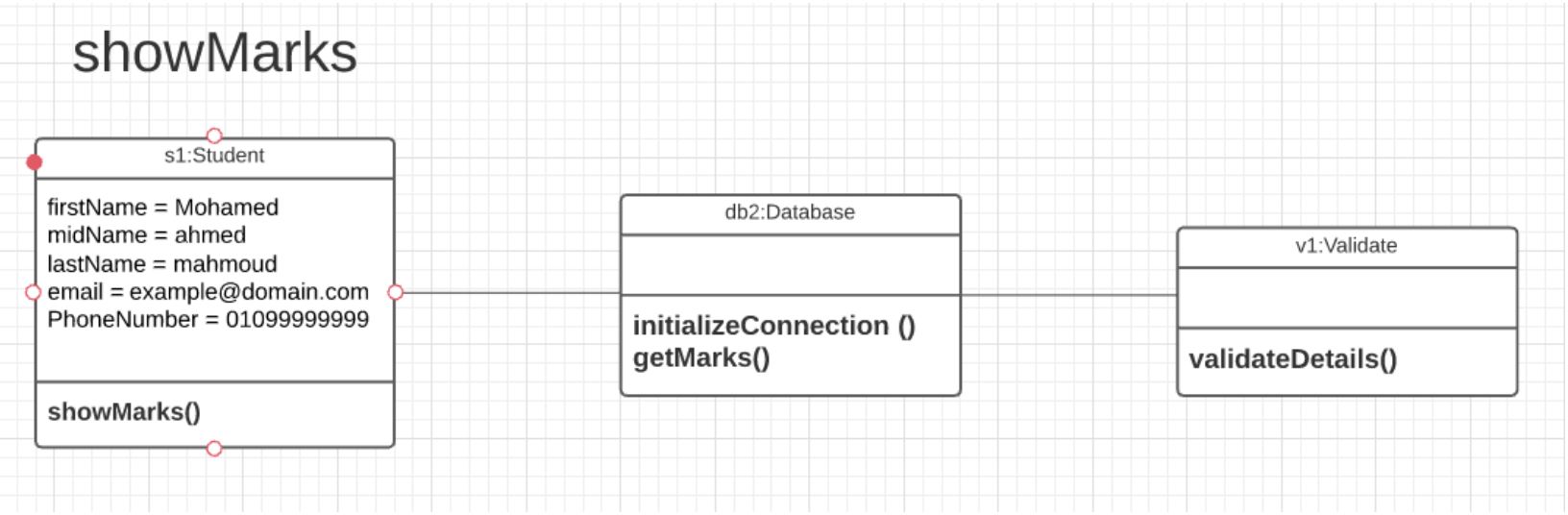
Precondition: admin should login and exam should be taken by the student

Postcondition: subject marks are set to the selected student.



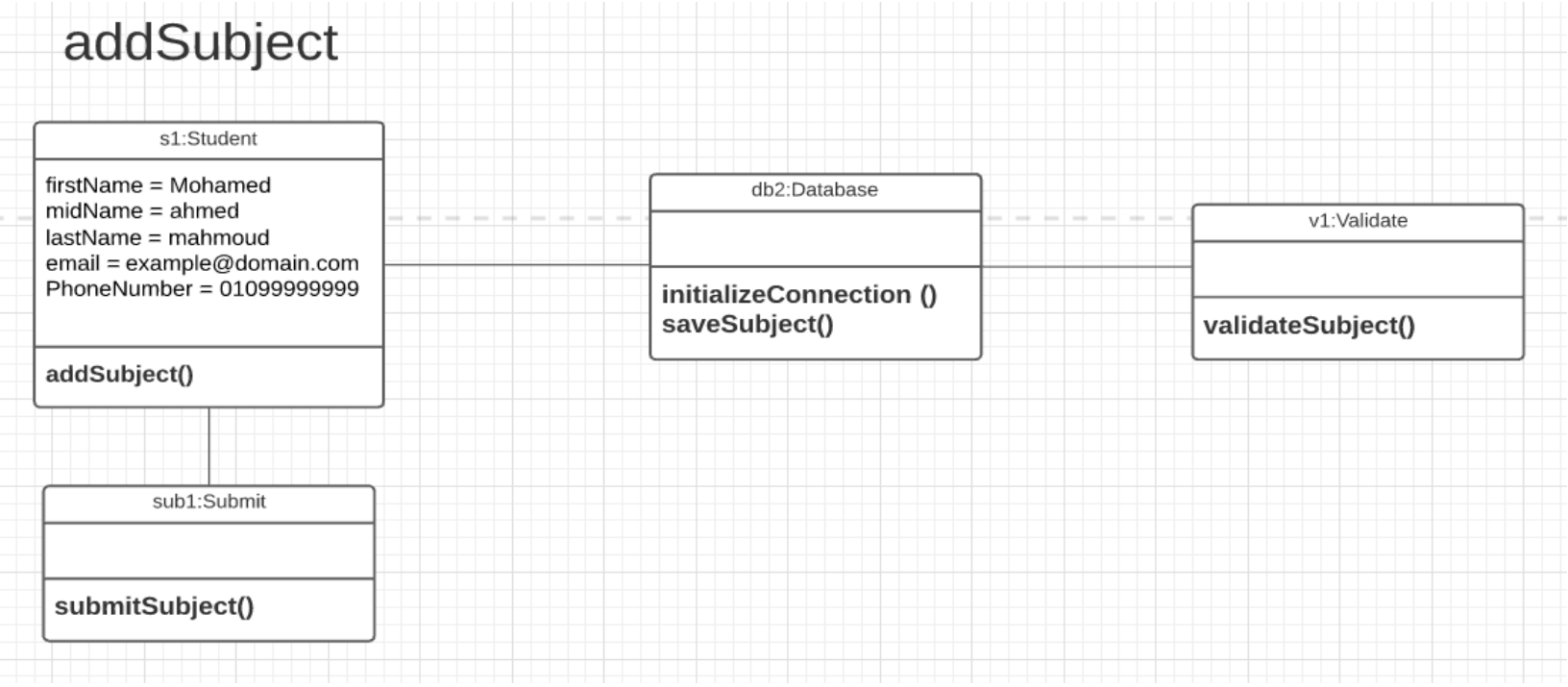
Precondition: student should login, press updateDetails button and provide the new details

Postcondition: student's personal information is updated.



Precondition: student should login, press addSubject button and select subject

Postcondition: student's selected subject is added



Precondition: student should login, press showMarks button

Postcondition: student is directed to his mark sheet.

Design Pattern

1- DESIGN PATTERN DESCRIPTION

• Name: Singleton

• Context: only one instance should exist (Admin)

• Problem to ensure that it is never possible to create more than one instance of the Admin class ?

• Forces: The use of a public constructor cannot guarantee that no more than one instance of Admin class will be created.

The Admin class instance must also be accessible to all classes that require it, therefore it must often be public.

• Solution: The constructor should be private to make sure that no other class will be able to create an instance of the Admin class.

The method should be a static method and stores a reference to that object in a static private variable (the Admin)

2-DESIGN PATTERN DESCRIPTION

• Name: Immutable

• Context: The state of the Database class never changes after creation

• Problem: to create Database class and establish connection of database once without changing.

• Forces: nothing should allow unauthorized modification of the Database class for connection in database.

• Solution: Make sure that the constructor of the database class is the only place where instances of variables are set for making a connection only once.

Instance methods which access properties must not change instance variables.

3- DESIGN PATTERN DESCRIPTION

• Name: Abstraction-Occurrence Pattern

• Context: the members of Student class share common information but also differ from each other in important ways

• Problem: Best method to represent the Student class.

• Forces: Representing the members of student class without duplicating the common information.

• Solution:

1- Create an subject class that contains the common data repeated in the student class instances

2- Then create an student class representing the occurrences of the subject class.

3- Connect these classes with a one-to-many association.

• Anti-Pattern: Using single class leads to duplication of information

Inheritance leads to the same result