Software Requirements Specification

for

ERP on College Management System

Faculty of Computers and Artificial intelligence

‘

Contents

Introduction

Purpose

This document describes the software requirements specification (SRS) for the Collage Management System that provides the access and management of information of different modules in a collage-like Students, Teachers/librarian.] There are five users for this system

1. Admin (have full access to read and write of all modules in management system)
2. Teacher (have access limited to write and manage the student's marks, attendance, etc.).
3. Student (have access limited to view marks, attendance, etc.)
4. librarian (have access limited to write and view everything regarding books and library etc.)
5. Super Admin (can Add\delete college to the system)

Intended Audience and Reading Suggestions

This document is to be read by the development team, the project managers, testers, and documentation writers. The software engineer/Developer and project managers need to become intimately familiar with the SRS. Others involved need to review the document. Testers need an understanding of the system features to develop meaningful test cases and give useful feedback to the developers. The developers need to know the requirements of the software product they need to build.

This document is for general discussions on the implementation decisions regarding the College Management System. The user of the product should have the concepts of RDMS, SQL, interfaces, and classes.

Product Scope

As Colleges are growing day by day more and more, and also increasing the complexity of storing information of students and related to the college system, they face many related issues: attendance and fee of students, salary details of employees, etc.

This project is based on the educational institute system where this application gives maximum services in a single software product that is used by teacher and system administration

- Any college can use this system as it is not client centric
- All admission and examination related work for the student can be done using this
  system.
- Deliver Electronic Workplace

Resources

Drive link Contain all the diagrams for further information

Overall Description

Product perspective

The system will consist of a web portal. The web portal will be used to maintain reports regarding the performance of faculty and student. This software is to be used as a grading tool to analyze performance of student at various institute. These grades would be a part of academic curriculum of respective institute.

Product functions

1. The admin has the privilege to insert , delete and update information about Head of institute and various institute details. He can view the report of various Institutes and analyze their performance.

2. The teacher is the person who manages the courses and analyzes the student’s performance and reports to coordinator , teacher will interact with the students and gives a brief analysis of student’s performance to admin.

3. Librarian is the person who help students to find books which will help them in their study .

4. Student is the person who enrolls for the courses and gains knowledge through quizzes, assignments given by teacher .

2.3 Assumptions and dependencies

One assumption about the product is that it will always be used on operating system and browser that have enough performance. If the system does not have enough hardware resources available for the application, for example the users might have allocated them with other applications; there may be scenarios where the application does not work as intended or even at all.

User Types and Authorities

This management system is controlled by the teachers and system administrators. In this system, admin is the main user who has full access to the management system admin can view and modify all information that is stored in the database.

Admin can view and modify the student's records like student's profile, attendance, fee, results, and details of teachers and other employees in college, their personal information.

Teachers have access to view and modify the student's information like their attendance, marks of exams to generate the progress report of students.

Design and Implementation Constraints

During the implementation of the product, different challenges are faced. Choosing the interface for the management system was a paramount issue. Connecting the database with the application was a major problem

For connecting the database, we had to install MySQL and then we had to download the driver(software). The connection of the database that is created in MySQL with PHP

Operating Environment

The CMS is expected to be deployed in a real environment to manage the DBMS inside the college. The centralized database is used to store the information. The user only within the college (members of college staff/student) can use this management system. Users outside form the college cannot access the management system.

The database is used in different departments within a branch of the college. The database used to store the information is the centralized database. The software we have developed will be installed on different computer systems within a college and software will be connected to a centralized database through LAN within a college and then the user can interact with the system and can store the data and other users can get access the stored through a centralized database.

External Interface Requirements

User Interfaces

The website should work and be tested against IE, Firefox, Google Chrome, and Netscape.

Hardware Interfaces

There are no special hardware interface requirements

Software Requirements Specification for College Website

Software Interfaces

Software requirements of the system are very nominal, and no other special requirement is there

hence it is economically feasible. Also, PHP is open source and thus easily available free of cost.

Communications Interfaces

There are no special communication interface requirements

Functional requirements:

Admin:

- After successful login, system shall display administrative functions.
- Admin can change his bio data and profile if he needed it
- Super admin can add colleges into the system with their details. Also, can view/delete
  a college from the system.
- Admin can add/view student * add student detail into a system.
- Admin can add/view teacher * add teacher detail into a system
- Admin can add/view librarian *add librarian detail into a system
- Admin can be add/update a timetable for students. Information about the classes time with particular subject with day wise in this timetable. Admin
  can be update timetable branch wise and year wise.
- Admin can enter fee detail into a system. *paid/unpaid fee
- Admin can add event detail into a system.
- Admin can view and delete book request from database.
- Admin can see all the books detail from database.

Teacher:

- After successful login, user shall be able to continue navigating through the website and
  view school/college detailed information.
- Teacher can add assignment detail for the students.
- Teacher can add attendance detail into a database.
- Teacher can add student results details
- Teacher can view event detail from database.

Librarian:

- Librarian can login in his personal account.
- Librarian can add books into database for student use. * adding a single or multiple books
- Librarian can view added books
- Librarian can request a book
- Librarian can view student request for a book.
- Librarian can view detail of all issue books.
- Librarian can issue a book to students.
- Librarian can view detail of all return book.
- Librarian can view all the events.

Student:

- After successful login, user shall be able to continue navigating through the website and view school/college detailed information. *login with initial username and password given by the faculty

- Student shall be able to update and maintain their profile, such as changing password and
  personal details. * if student is going to update his personal information in student panel
  then the updating is also done or been performed in college panel and parents' panel.
- student can see his profile *if needed modification can be done for new password and
  profile picture etc.
- Attendance: this field informs about the attendance of student by semester wise till date it has been present .
- Marks: this field informs about the marks of student by semester wise till semester it has been present. It can also display the marks of first midterm and final exam of a particular semester.
  It show that student is achieved marks of a particular subject from total marks.
- Student can view book added by librarian
- Student can view all timetable detail
- Student can view all assignment detail added by teacher.
- Student can view all event details.

Nonfunctional requirements:

Usability

- System is very easy to understand and user friendly. -look and feel
- People with no training and no understanding of English shall be able to use the system by providing images as much as we can
- The system shall be easy to use by adult members (age 18 to 80) after 4 hours of training , with 2 errors per hour after the training
- Formal language should be used in the user interface
- System shall be interactive
- At least 80% of users polled after a 3-month usage period shall rate their satisfaction with the system at 7 and more on a scale of 1 to 10

Performance

- In this system user can connect any time whenever he/she wants.
  *the college management system shall be built upon the web development technique and put on the web server online. The system and the server must be capable of handling the Realtime error functionality occurs by the defined users.
- The system should be able to perform multiple tasks at once.
- The system shall handle up to 300 users simultaneously

Reliability

- The system is safety critical. If it moves out of normal operation mode, the requirement

to drop or down the server and fix it as soon as possible and open it again. This emergency
behavior shall not occur without reason.

- Mean time to failure : once every 30 days

Availability

- The system is saving the time for student and teacher too.
  When in normal operating conditions, request by a user for an online system shall be
  handled within 1 second. The site should load in 3 seconds when the number of simultaneous users are > 10000
- The system should be available so that the user can connect anytime he/she wants.

Security

- The system is secure.
  There shall be a strong security mechanism should be place in the server side of the
  system to keep unwanted users to hack or damage the system. However, all users of the system
  give and store the details of privacy related to personal information and many other. However,
  our system can be accessed online so we need very secured system as far as security is
  concerned.
- Only the users with the role “admin” can view the students and teacher data
- password requirements – length, special characters

- System shall distinguish between authorized and unauthorized users according to their roles
- Immunity: system shall be protected against hacks and attacks
- Integrity: prevention of unauthorized modification/deletion of data according to their roles

Maintainability

- There shall be design documents describing maintenance of the software and database
  used to save the user details as well as the daily updated and modification done in system. There
  shall be an access on the control system by the admin to maintained it properly at the front end as
  well as at back end.
- The system shall be able to be modified to cope with new rules added by the Admin Each Semester

Portability

- There is portability requirement as far as our system is concern because it is an online as
  well as offline (local server based) system so we can access it from anywhere through the
  internet connection. And we have to maintain the copy of stored data into our database.

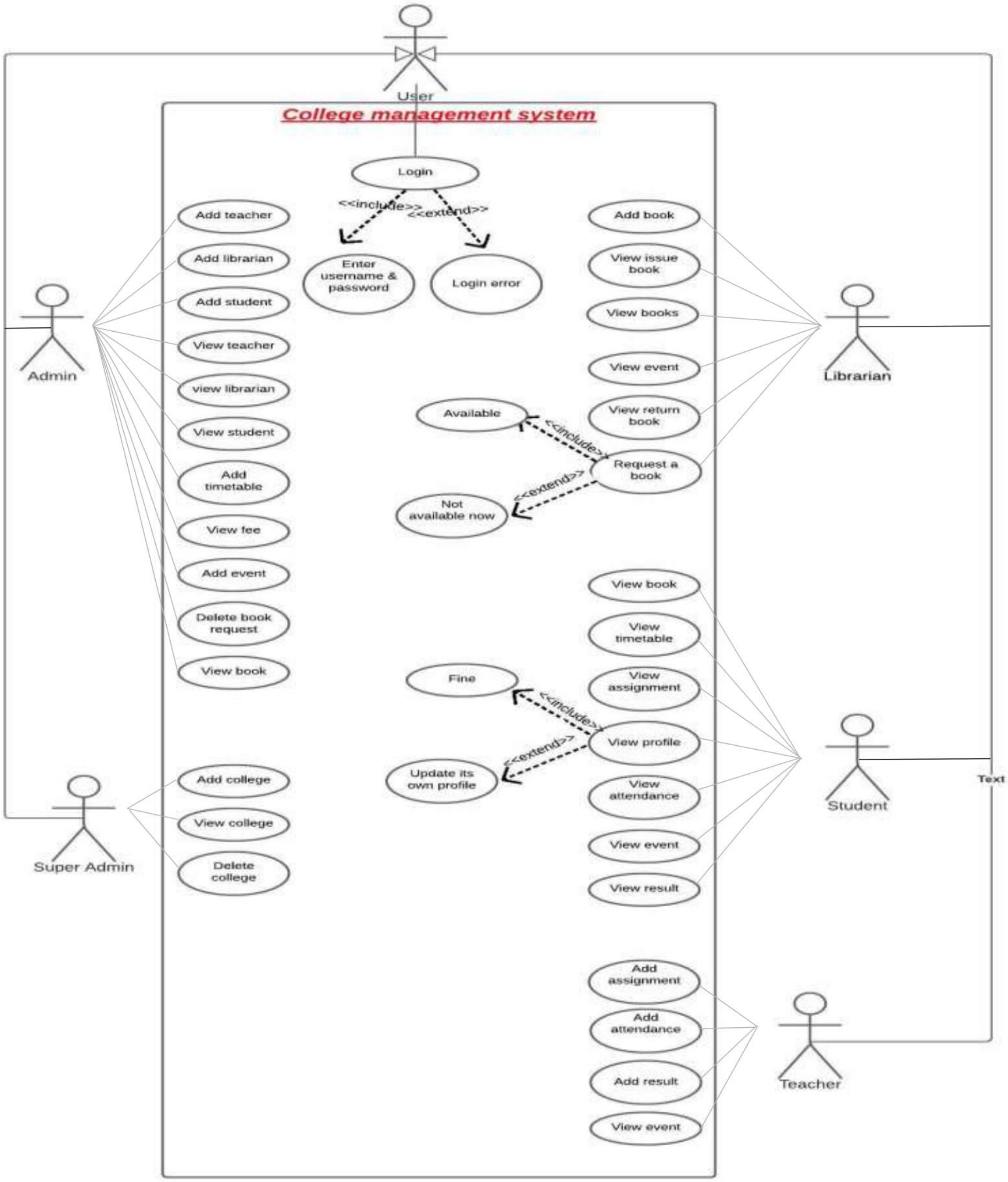
Accessibility

- The system shall be accessible by people with specific vision needs, to the extent that a user shall be able to:
  a. Display the whole user interface in a large font without truncating displayed text or other values.
- The system must meet web content accessibility guidelines WCAG3.

Robustness

- Time to restart after failure : one hour

Use-Case Diagram



Use case scenarios

(1) login:

Name/identifier:	**Login (login id, login password)**
Pre-login:	**Users log in by entering their ids and passwords, whether it is a (super admin, admin, teacher, librarian, or student).**
Post-login:	**After the user logs in successfully, each user can log in to his profile and do his work, for example the student can see his attendance and his assignment etc. the admin can add new teachers, students, librarians, and timetables.**
Login-check:	**Checks whether the user data is correct or not.** **If correct:** **It enables the user to log into the personal profile and exercise its validity according to the party logged in to the system.** **If not correct:** **The system shows him that its data are not identical.**

(2) add/delete universities:

Name or identifier:	**Add or delete universities.**
Pre-addition:	**The super admin can add/delete universities on the system.** **Verify the addition or deletion.** **It determines whether the university was added/deleted or not.**
Post-addition:	**The Universities are available on the system if the super admin added them, or they are wiped from the system if the super admin deleted them.**

(3) add teacher/librarian/student:

Name or identifier:	Add a teacher, librarian, or a student.
Pre-addition:	Admin can add teacher, student, or librarian.
Post-addition:	After adding users by the admin such as teacher, student, and librarian, they can use the system successfully, for example, the teacher can add student results and add assignments and attendance for students and students can view them.
Addition-check:	If users have been added successfully by the admin: a message appears that the operation was successful. If users are not added successfully: A message appears that the operation was unsuccessful.

(4) add results/assignments/attendance:

Name or identifier:	Add results, assignments, and attendance.
Pre-addition:	The teacher adds results, assignments, and attendance to the system.
Post-addition:	After adding results, assignments, attendance successfully on the system (students can view them).
Addition verification:	Determines whether the add operation was successful or not.

(5) add/delete books:

Name or identifier:	**Add or delete books.**
Pre-addition:	**The admin or librarian can add/delete a book from the system.**
Post-addition:	**Books are now available on the system. After adding it by admin or librarian, it can be viewed by students, and the librarian can return it to its place.**
Verifying adding or deleting the book:	**Determines whether the book was added or deleted successfully.**

(6) add time-table:

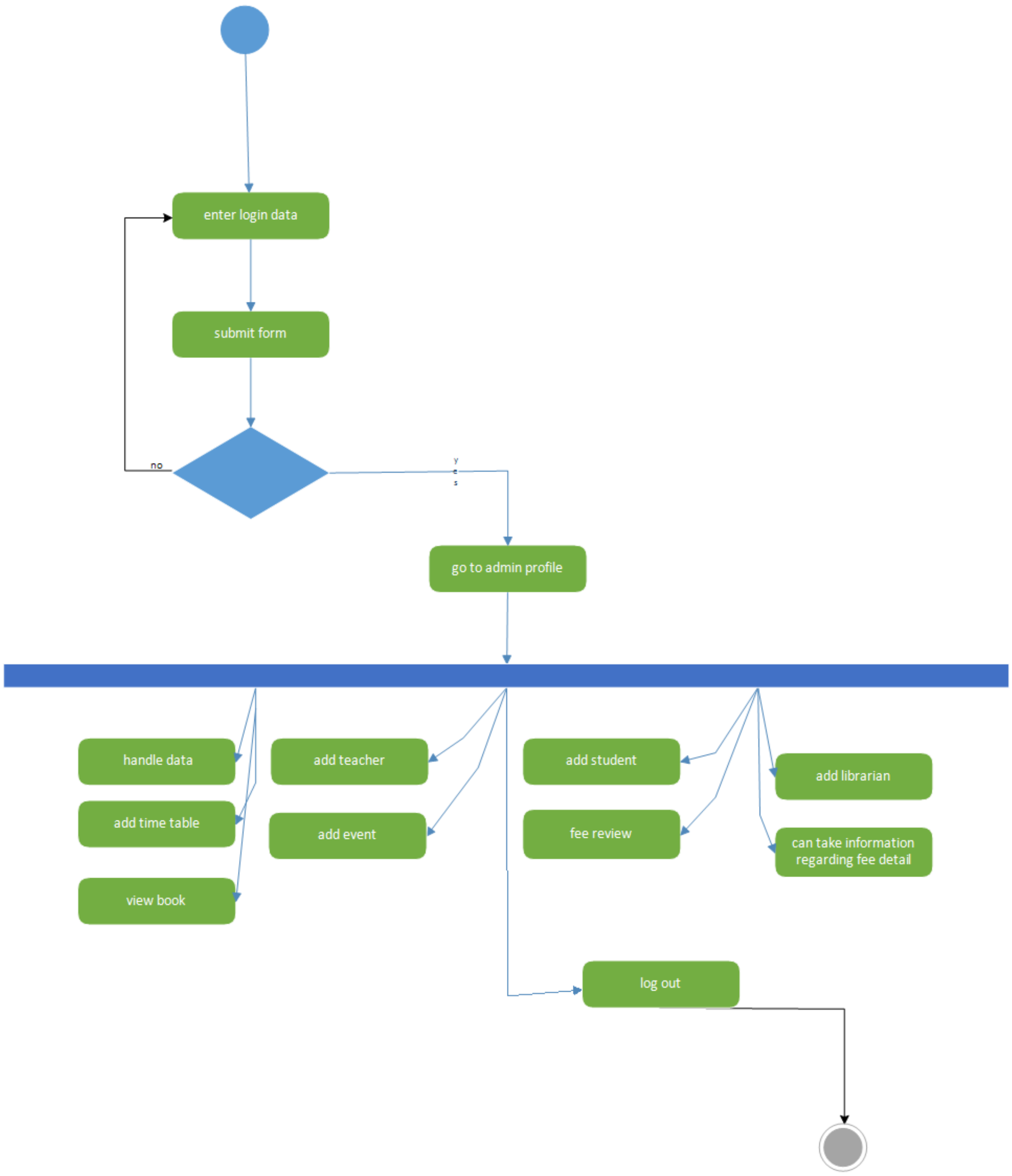
Name or identifier:	Add timetable
Pre-addition:	The admin can add a timetable.
Check the timetable extension:	Determines whether or not a timetable is added.
Post-addition:	Timetable is now available on the system and can be viewed by students.

(7) fee review:

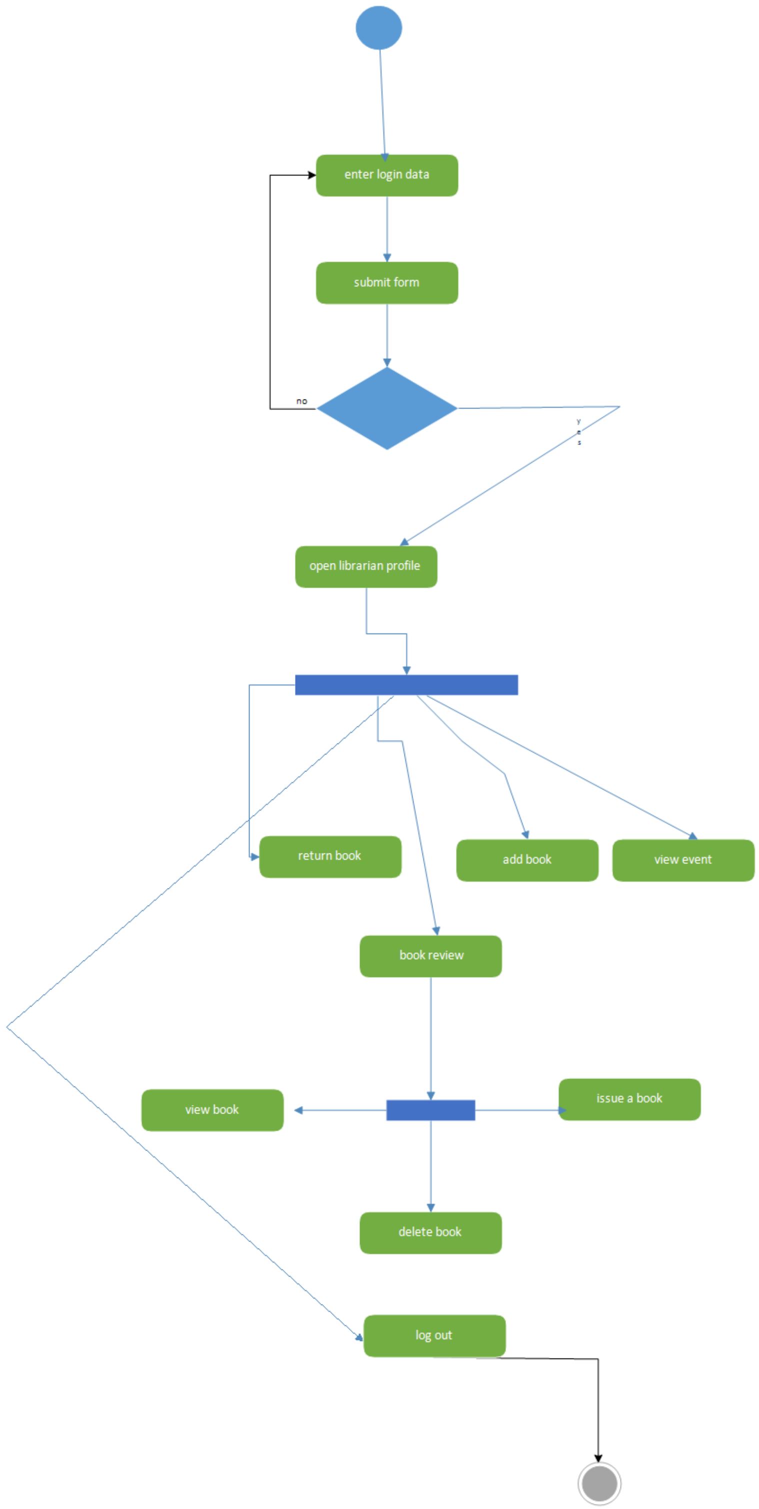
Name or identifier:	**Fee review.**
Pre-fee review:	**The admin can add fees to the system.**
Post-fee review:	**After the administrator adds the fees to the system, the admin can take information regarding the fee detail. The value of each university and the dates for its payment can also be determined, students can also view it.**
Check fees:	**Checks whether the fee was added successfully or not.**

Activity Diagram

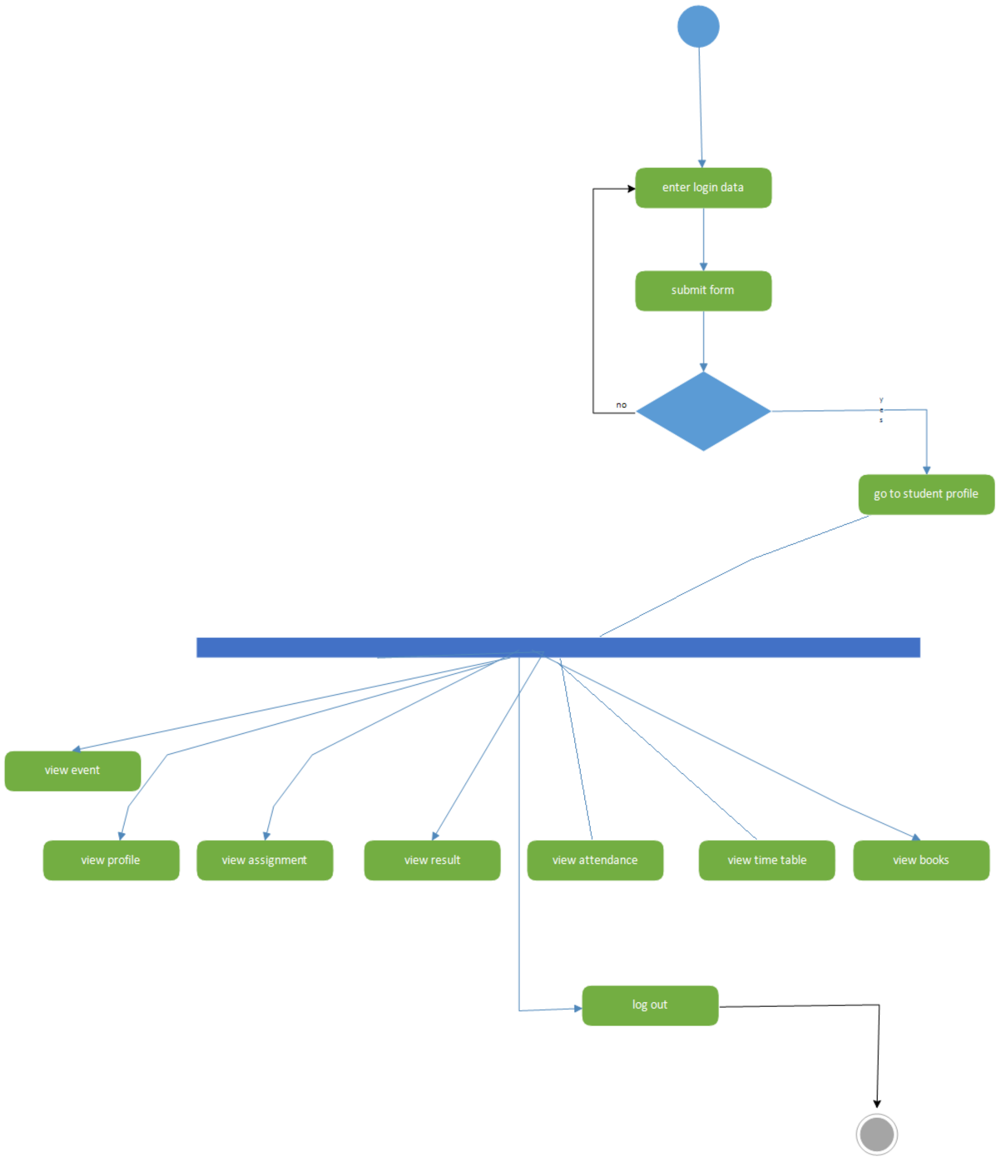
Admin:



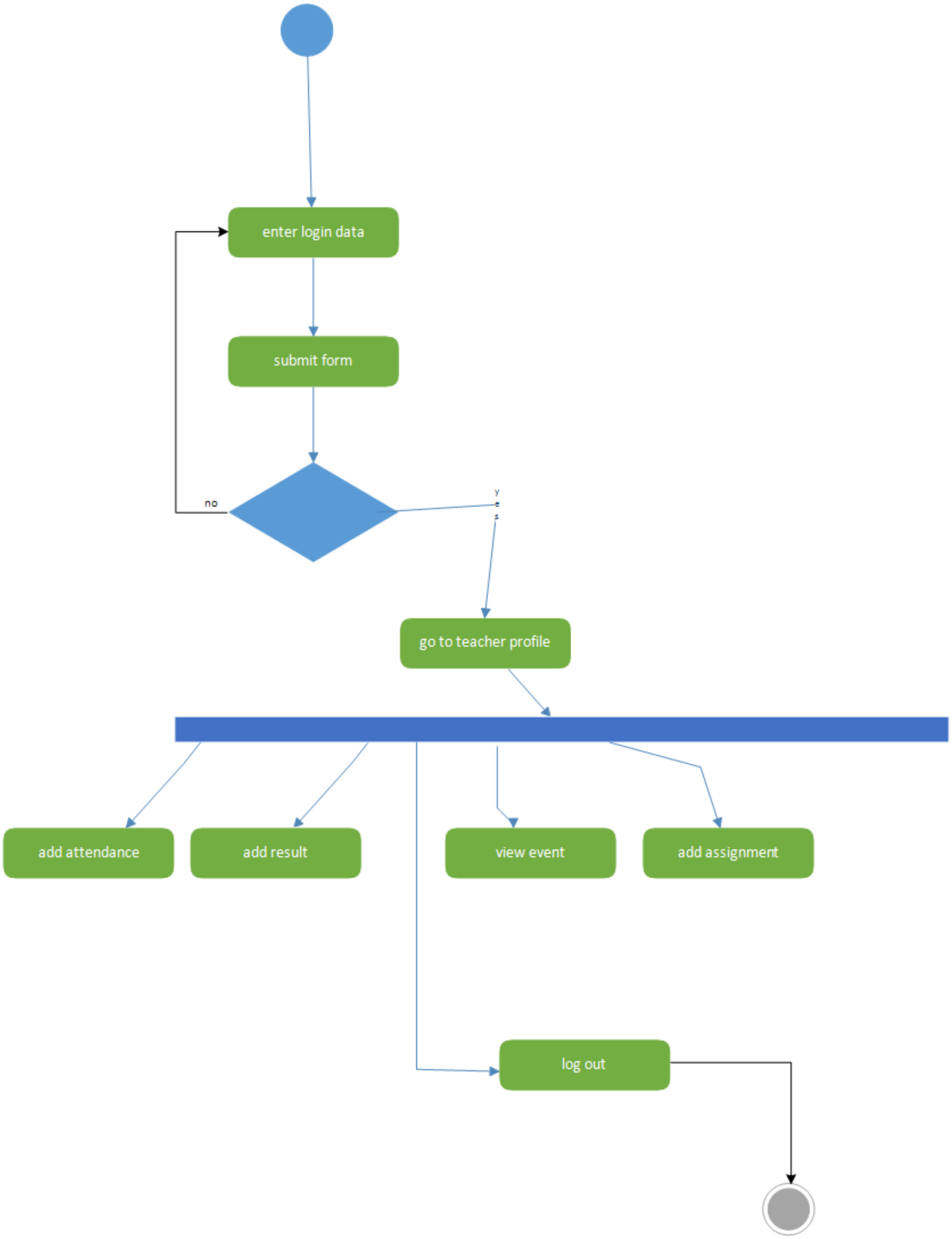
Librarian:


enter login data
submit form
no
yes
open librarian profile
return book
add book
view event
book review
view book
issue a book
delete book
log out

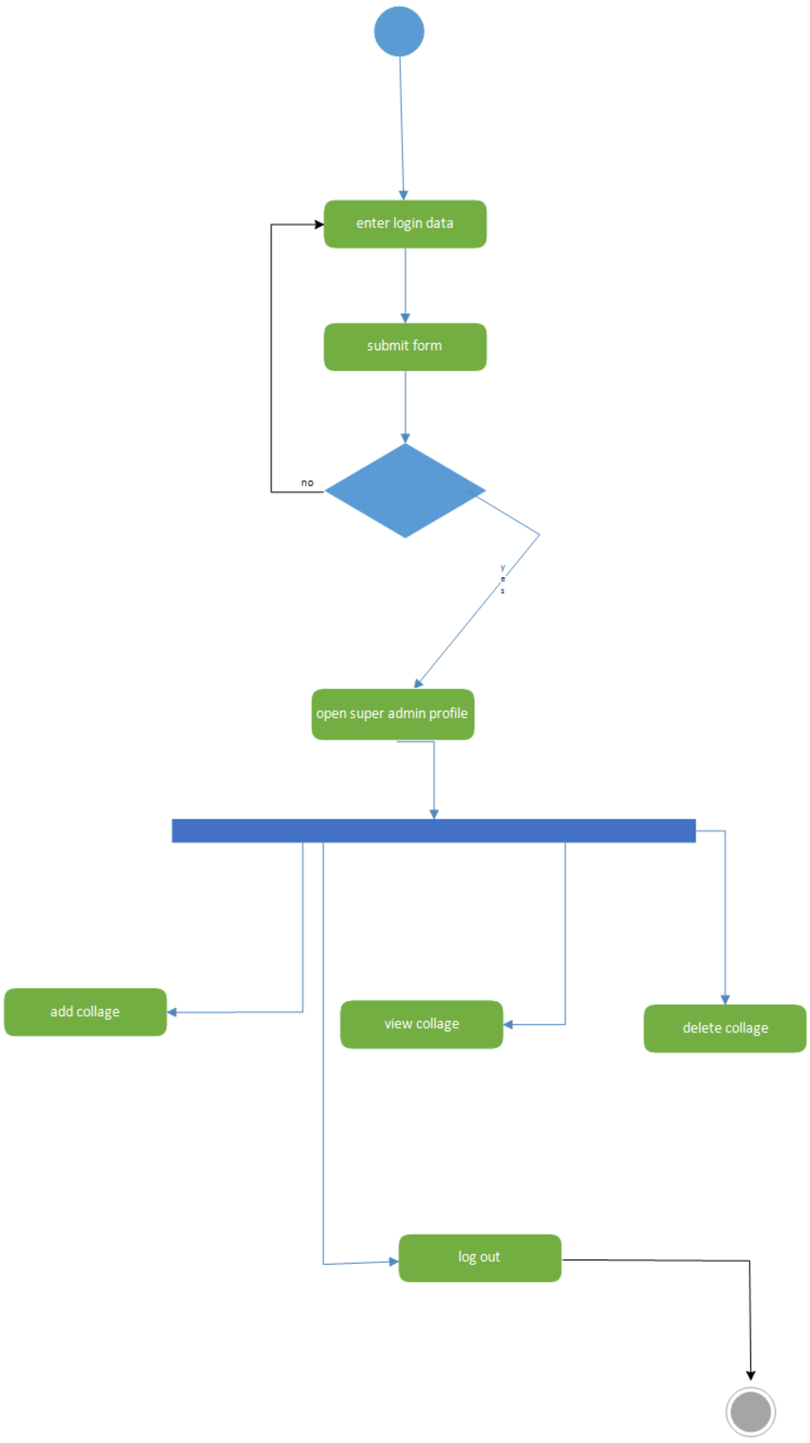
Student:


enter login data
submit form
no
y
e
s
go to student profile
view event
view profile
view assignment
view result
view attendance
view time table
view books
log out

Teacher:

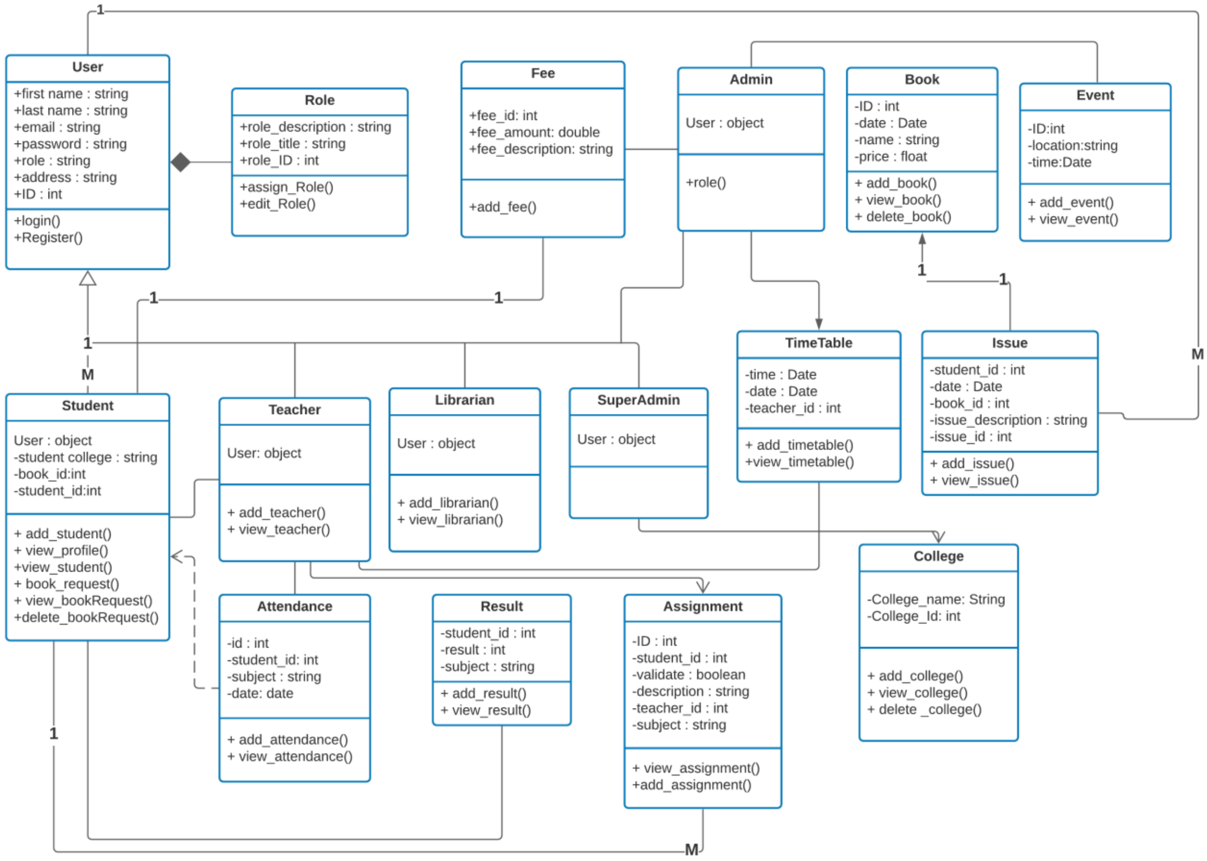

enter login data
submit form
no
yes
go to teacher profile
add attendance
add result
view event
add assignment
log out

Super Admin:

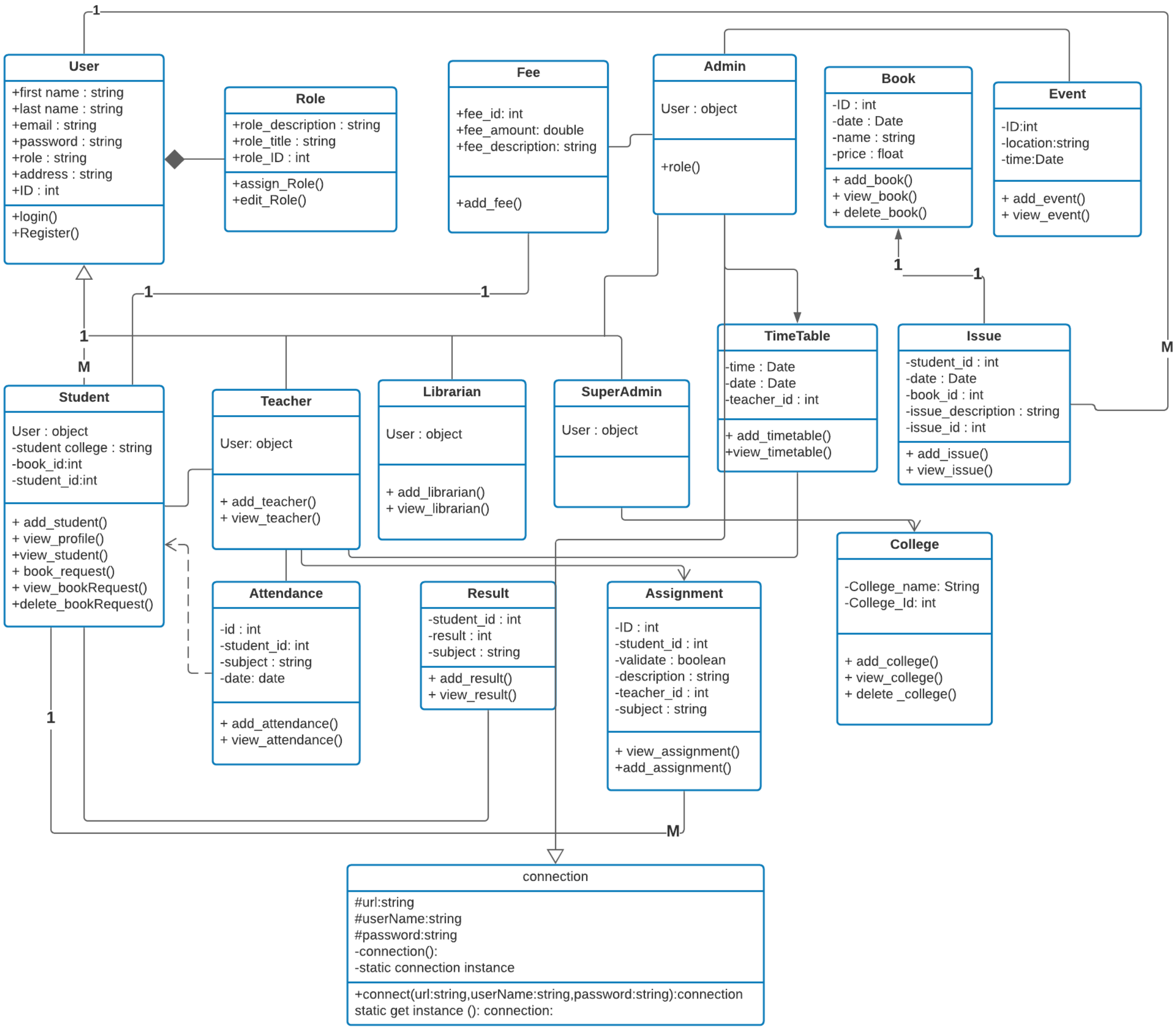

enter login data
submit form
no
yes
open super admin profile
add collage
view collage
delete collage
log out

Class Diagram

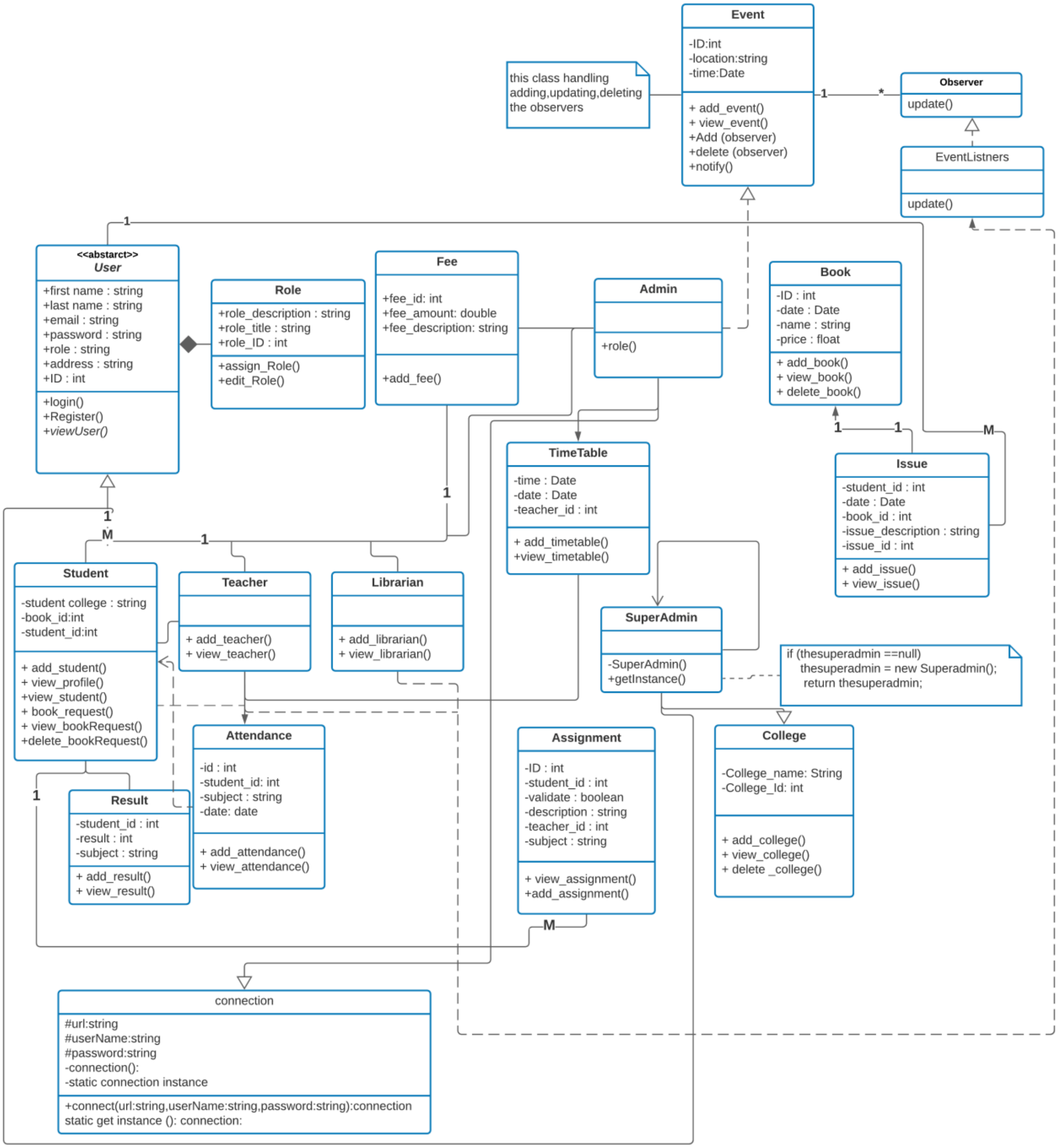
1.1.1 An initial version based on the requirements and Use-Case/Activity diagrams.



1.1.2 An intermediate version based on the interaction diagrams.



1.1.3 A final version, after applying the design pattern(s) and any other modifications.



Design Pattern Applied

Singleton pattern

Context:
It is very common to find classes for which only one Instance should exist(singleton).

Examples: Super admin class

Problem:
How do you ensure that it is never possible to create more than one instance of a singleton class. AND provide a global point of access to it.

Forces:

1. The use of a public constructor cannot guarantee that no more than one instance will be created.
2. The singleton instance must also be accessible to all classes that require it; therefore, it must often be public.

Solution:

1. Have the constructor private to ensure that no

   Other class will be able to create an instance of the

   class singleton.
2. Define a public static method, The first time this method is called, it creates the single instance of the class “singleton” and stores a reference to that object in a static private variable.

Abstraction – occurrence

Context:
In a user model you find a set of related objects “occurrences”; the members of such a set share common information but also differ from each other in important ways
Example: Teacher, librarian, Student

Problem:
What is the best way to represent such sets of occurrences?

Forces:

You want to represent the members of each set of occurrences without duplicating the common information

Solution:

Create an “abstraction” class that contains the common data

Then create an “occurrence” class representing the occurrences of the abstraction

Connect these classes with a one-to-many association

Observer / publish-subscriber

Context

When partitioning a system into individual classes you want the coupling between then to be loose, so you have the flexibility to vary them independently

Problem

A mechanism is needed to ensure that when the state of an object changes related objects are updated to keep them in step

Forces

The different parts of a system have to kept in step with one another without being too tightly coupled

Solution

One object has the role of the subject / publisher and one or more other objects the role of observer/subscribers. The observers register themselves with the subject, & if the state of the subject changes the observers are notified & can the update themselves.

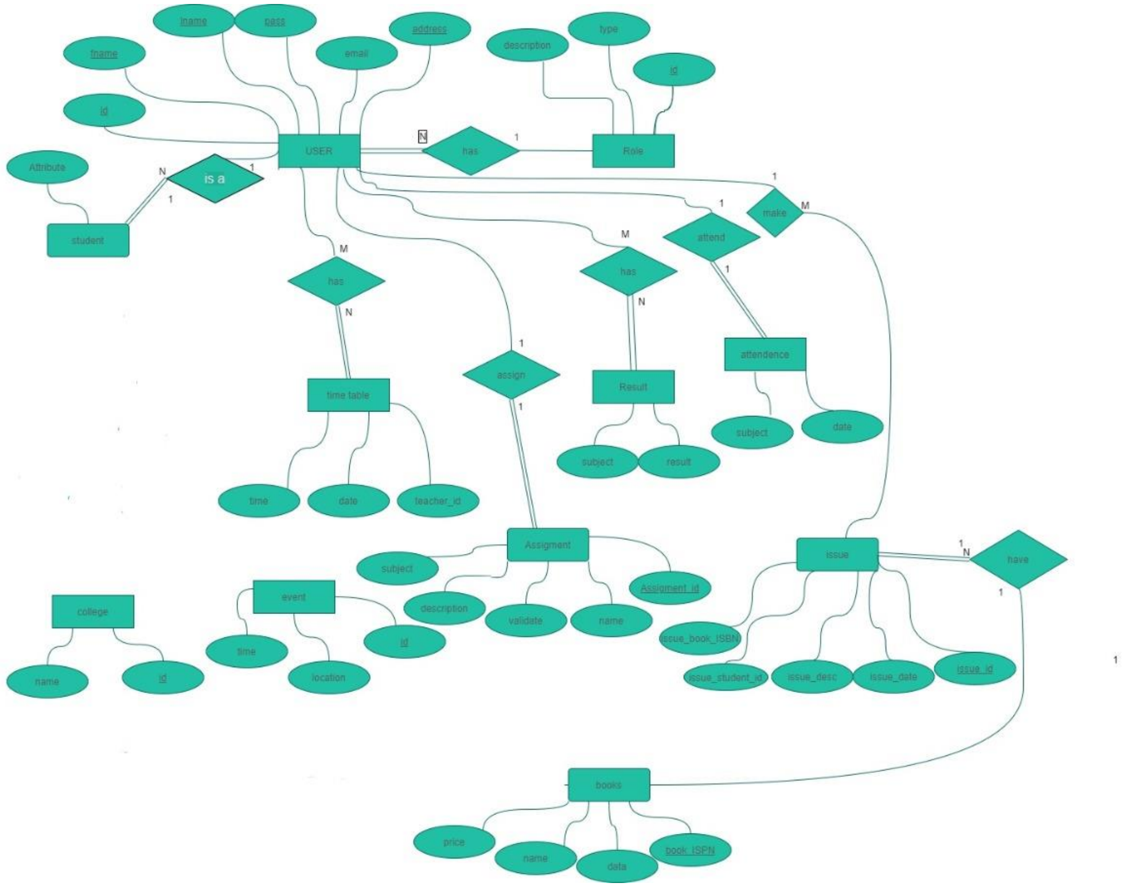
We use:

The pull model where the subject simple notifies the observers that there have been changes, and it is the responsibility of the observers to find out the details they need to update themselves.

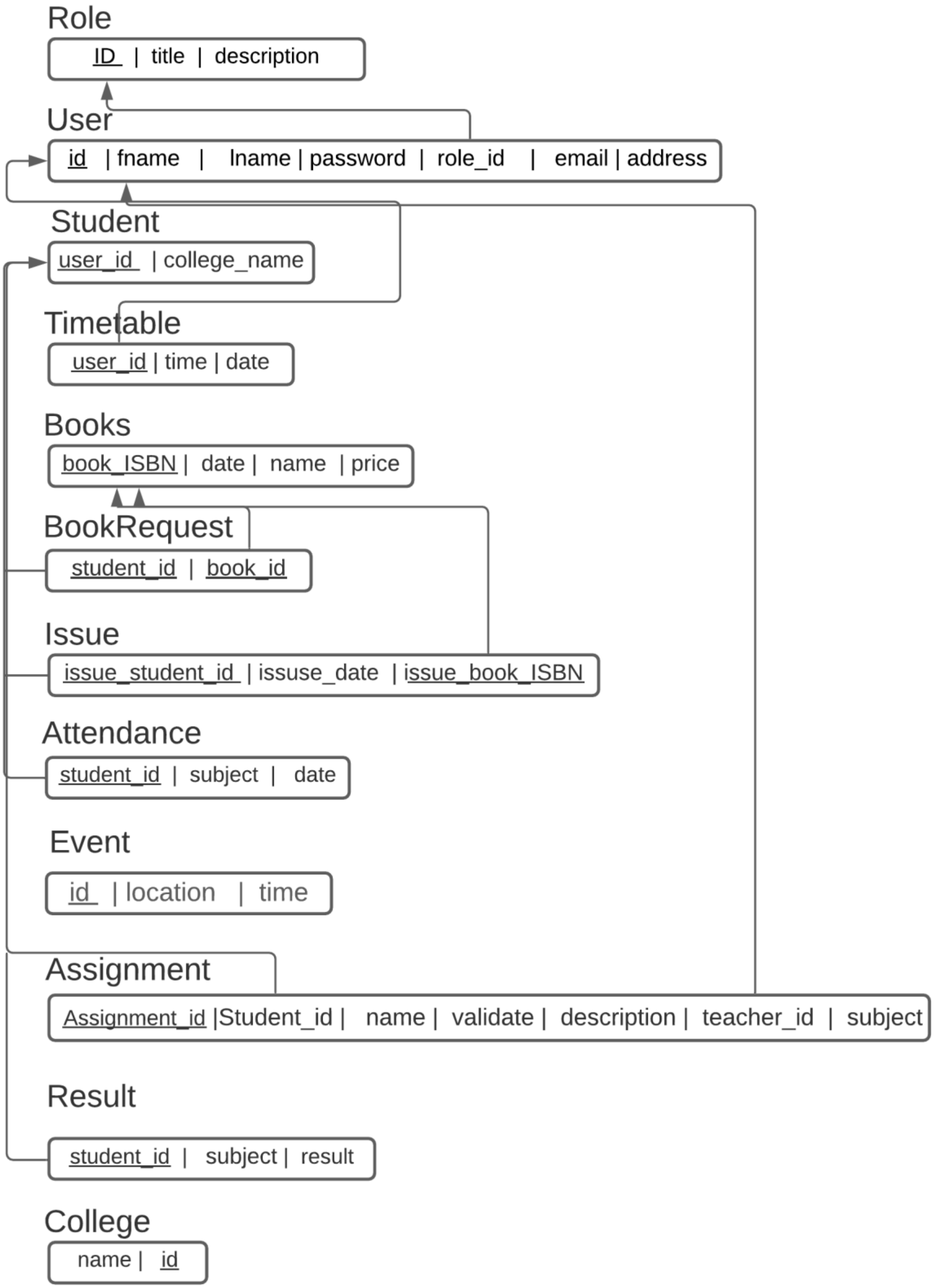
Example

The relationship between the view and the model (Event) in a mvc design can be realized by applying the observer pattern. The view registers with the model (Event) and is notified every time the model's state changes, allowing it to update itself to reflect the change.

ER Diagram:


lname
pass
address
fname
email
description
type
id
id
USER
N
has
1
Role
Attribute
N
is a
1
1
student
1
make
M
1
attend
M
M
has
has
N
1
N
1
attendence
assign
Result
time table
1
subject
date
subject
result
time
date
teacher_id
Assigment
subject
Assigment_id
description
validate
name
1
N
issue
have
1
college
event
time
id
location
name
id
issue_book_ISBN
issue_student_id
issue_desc
issue_date
issue_id
1
books
price
name
data
book_ISPN

Relational Data Model:



Role

ID | title | description

User

id | fname | lname | password | role_id | email | address

Student

user_id | college_name

Timetable

user_id | time | date

Books

book_ISBN | date | name | price

BookRequest

student_id | book_id

Issue

issue_student_id | issuse_date | issue_book_ISBN

Attendance

student_id | subject | date

Event

id | location | time

Assignment

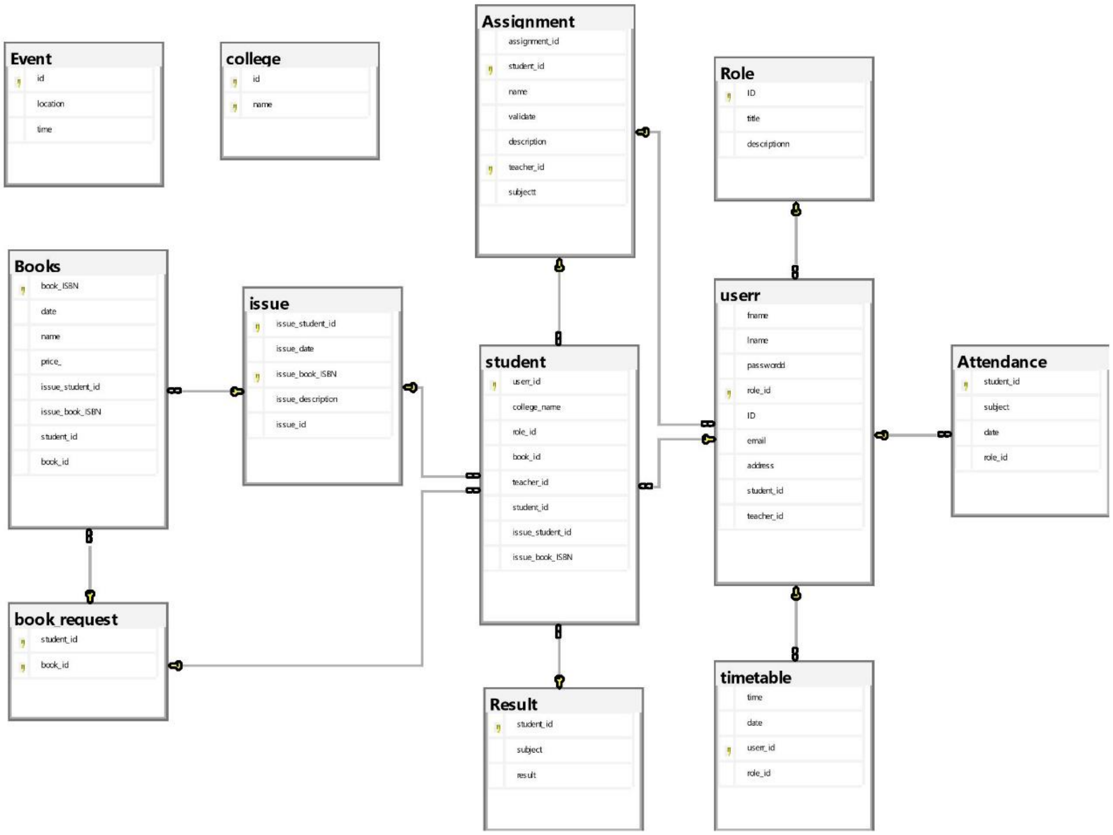
Assignment_id |Student_id | name | validate | description | teacher_id | subject

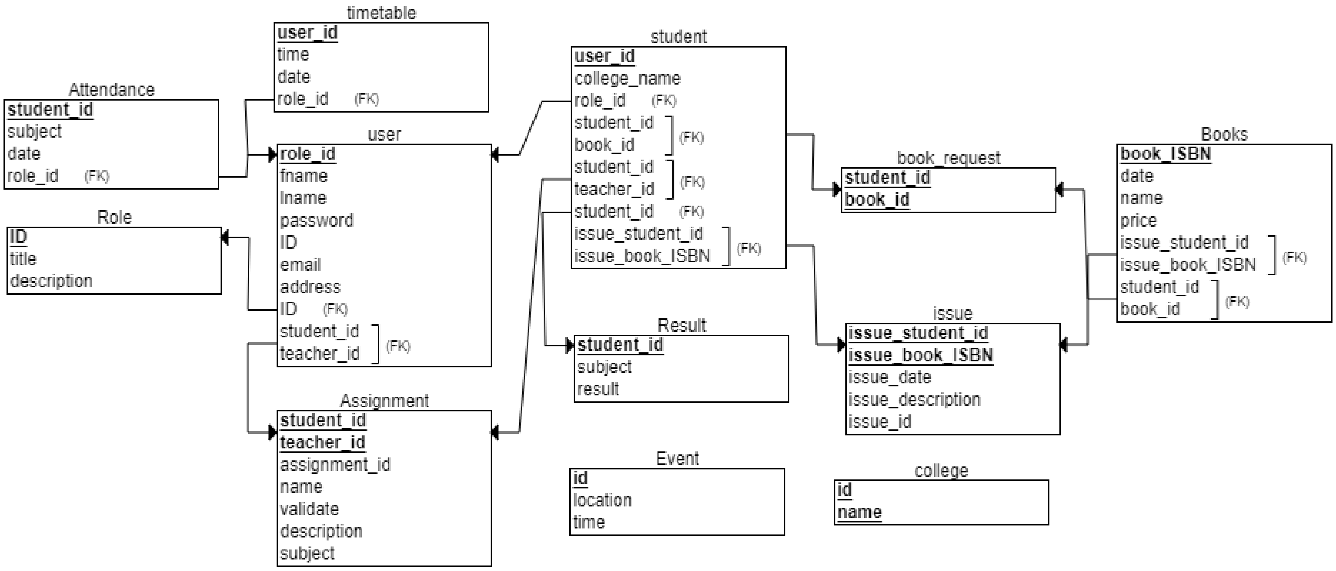
Result

student_id | subject | result

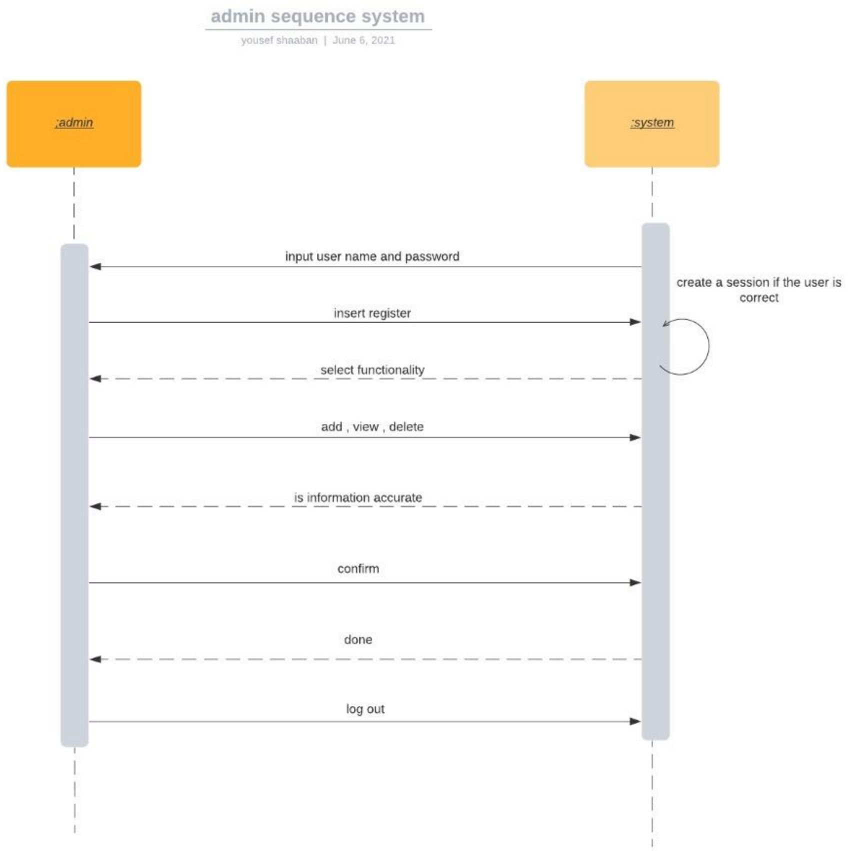
College

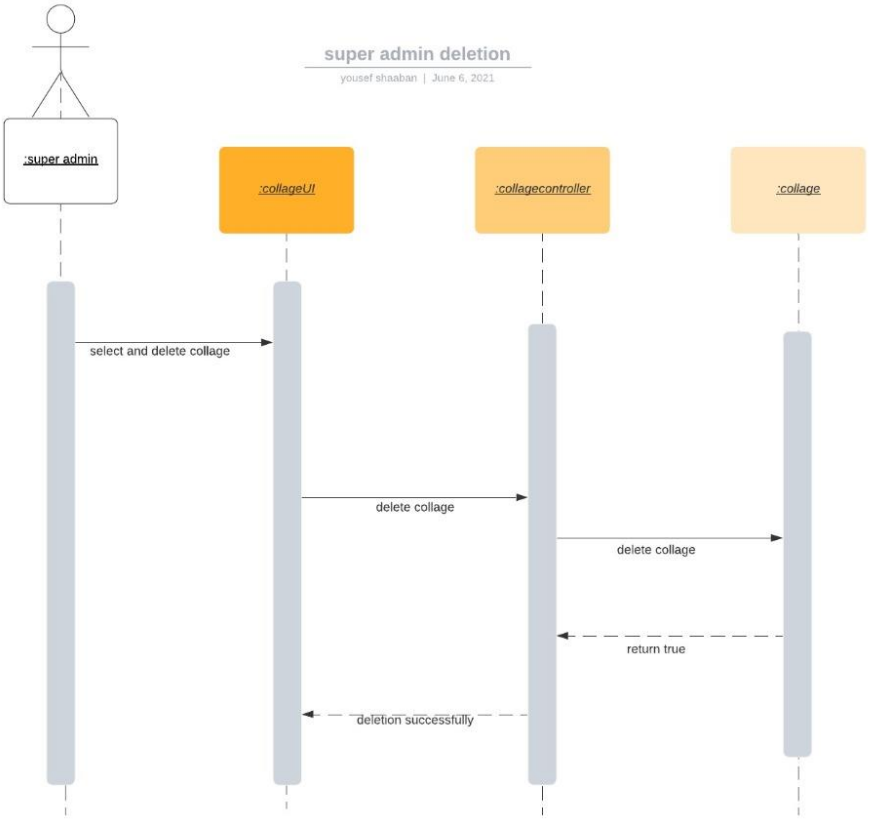
name | id


Event
id
location
time
college
id
name
Assignment
assignment_id
student_id
name
validate
description
teacher_id
subjectt
Role
ID
title
descriptionn
Books
book_ISBN
date
name
price_
issue_student_id
issue_book_ISBN
student_id
book_id
issue
issue_student_id
issue_date
issue_book_ISBN
issue_description
issue_id
student
userr_id
college_name
role_id
book_id
teacher_id
student_id
issue_student_id
issue_book_ISBN
userr
fname
lname
passwordd
role_id
ID
email
address
student_id
teacher_id
Attendance
student_id
subject
date
role_id
book_request
student_id
book_id
Result
student_id
subject
result
timetable
time
date
userr_id
role_id


timetable
user_id
time
date
role_id (FK)
Attendance
student_id
subject
date
role_id (FK)
user
role_id
fname
lname
password
ID
email
address
ID (FK)
student_id
teacher_id (FK)
Role
ID
title
description
Assignment
student_id
teacher_id
assignment_id
name
validate
description
subject
student
user_id
college_name
role_id (FK)
student_id
book_id (FK)
student_id
teacher_id (FK)
student_id (FK)
issue_student_id
issue_book_ISBN (FK)
Result
student_id
subject
result
Event
id
location
time
book_request
student_id
book_id
issue
issue_student_id
issue_book_ISBN
issue_date
issue_description
issue_id
college
id
name
Books
book_ISBN
date
name
price
issue_student_id
issue_book_ISBN (FK)
student_id
book_id (FK)

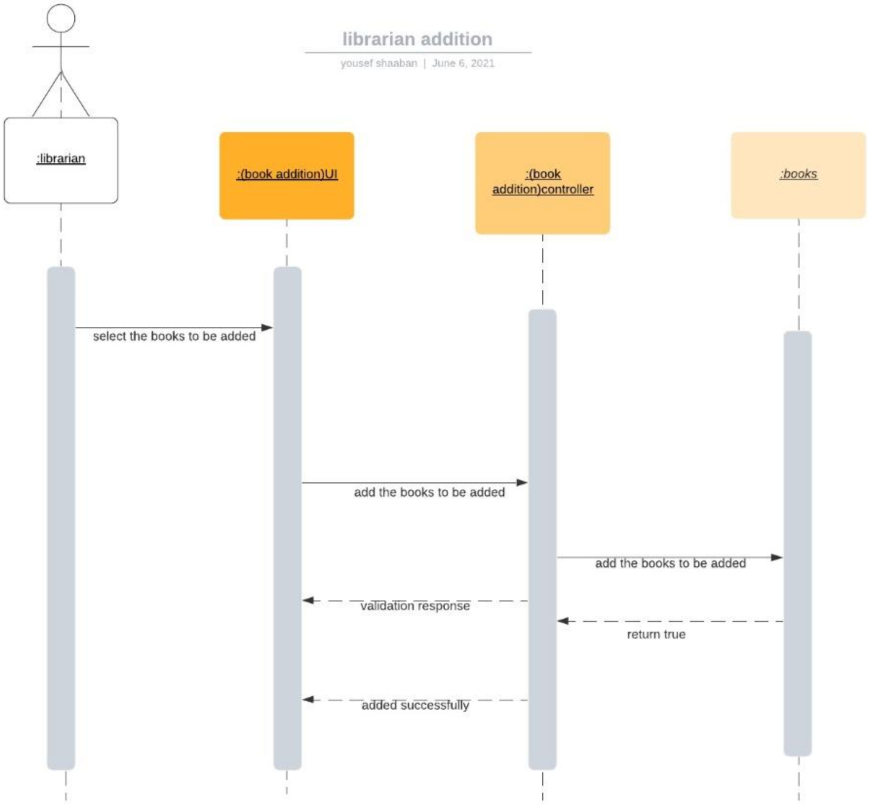
Sequence Diagram

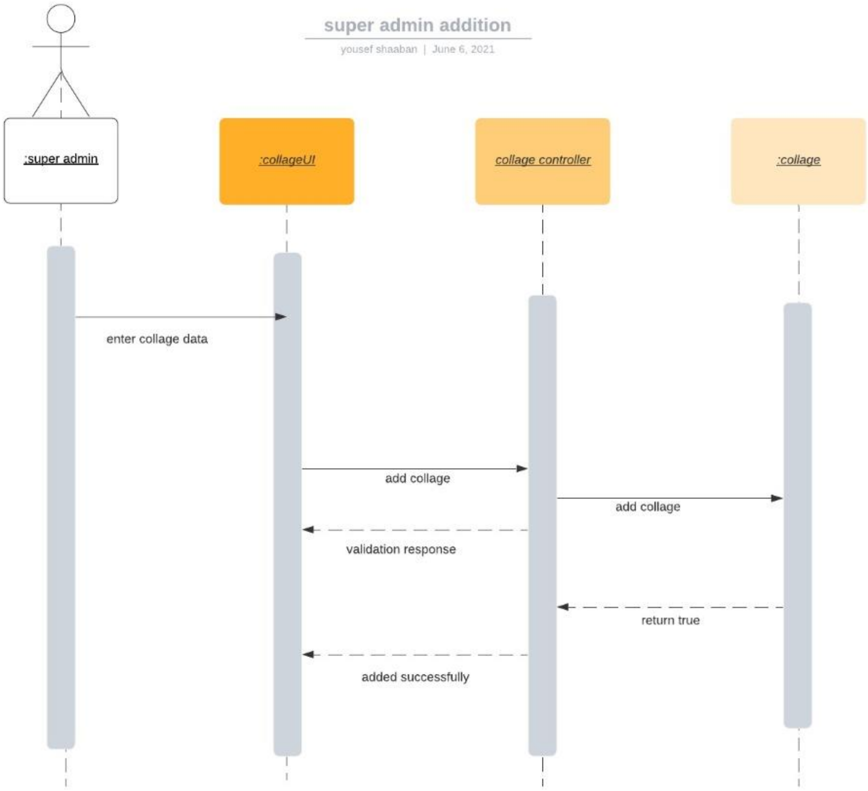




super admin deletion

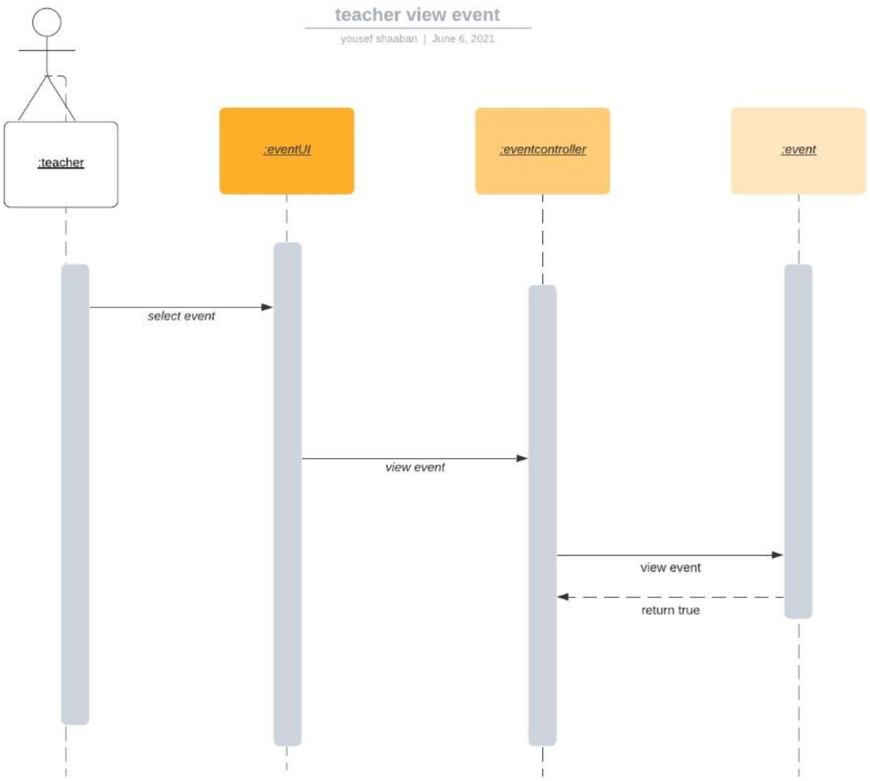
yousef shaaban | June 6, 2021


librarian addition
yousef shaaban | June 6, 2021
:librarian
:(book addition)UI
:(book addition)controller
:books
select the books to be added
add the books to be added
add the books to be added
validation response
return true
added successfully



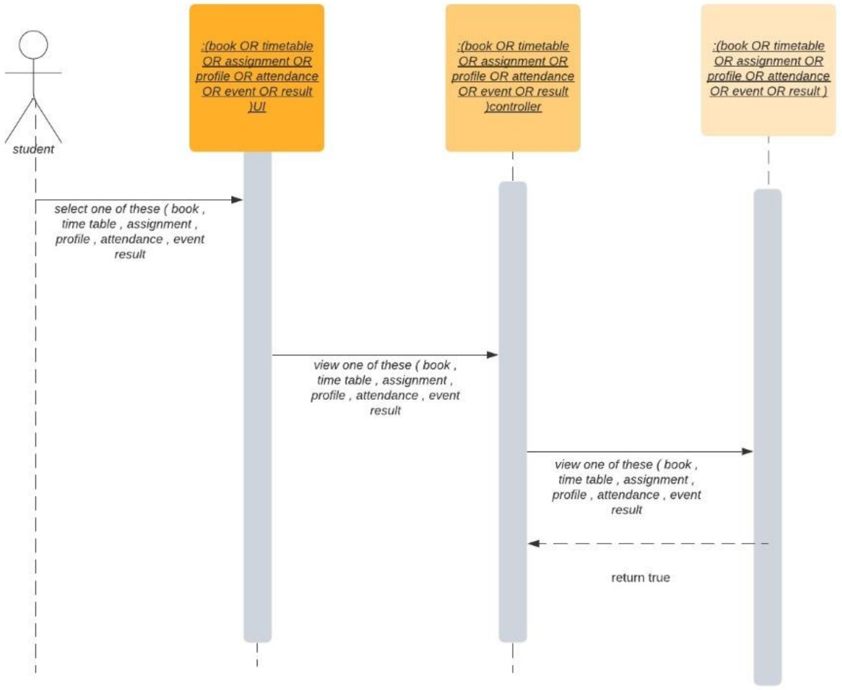
super admin addition

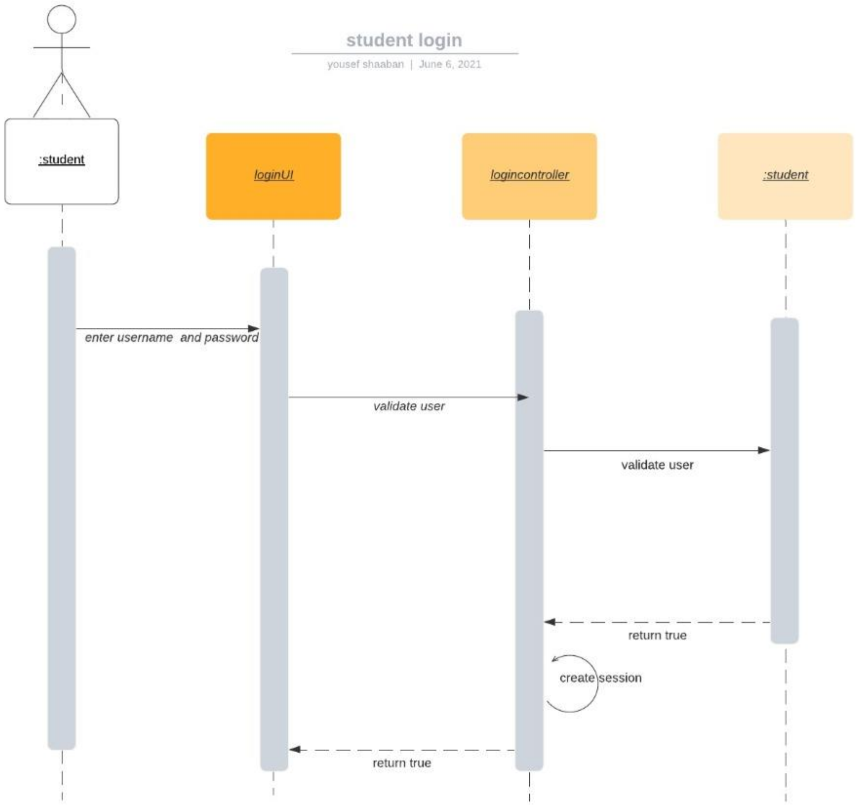
yousef shaaban | June 6, 2021

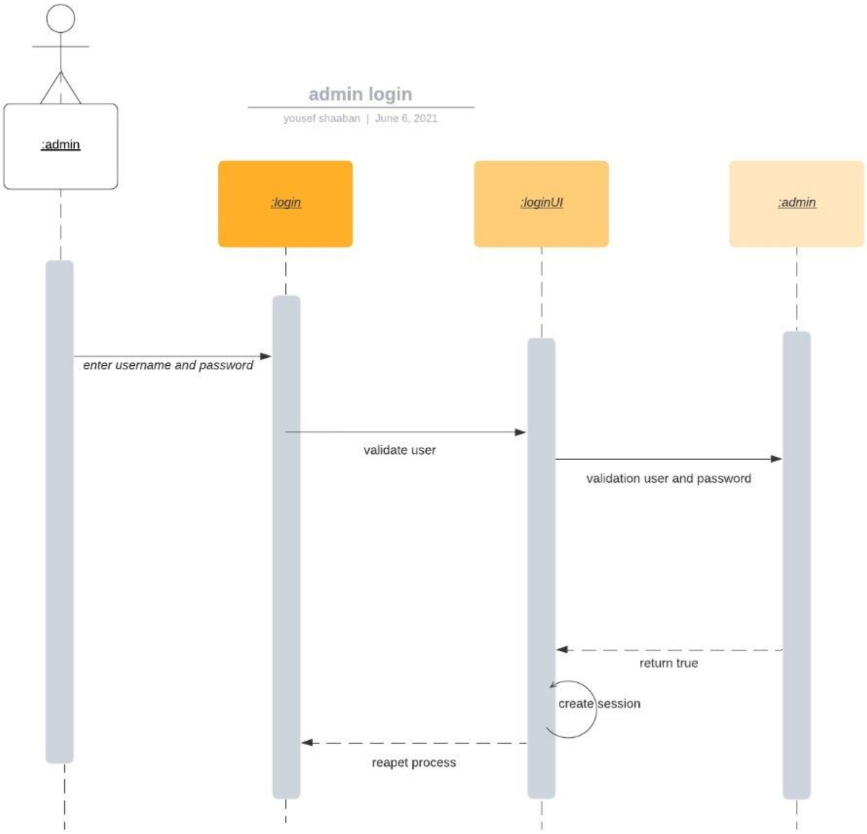

teacher view event
yousef shaaban | June 6, 2021
:teacher
:eventUI
:eventcontroller
:event
select event
view event
view event
return true

student 2

yousef shaaban | June 6, 2021

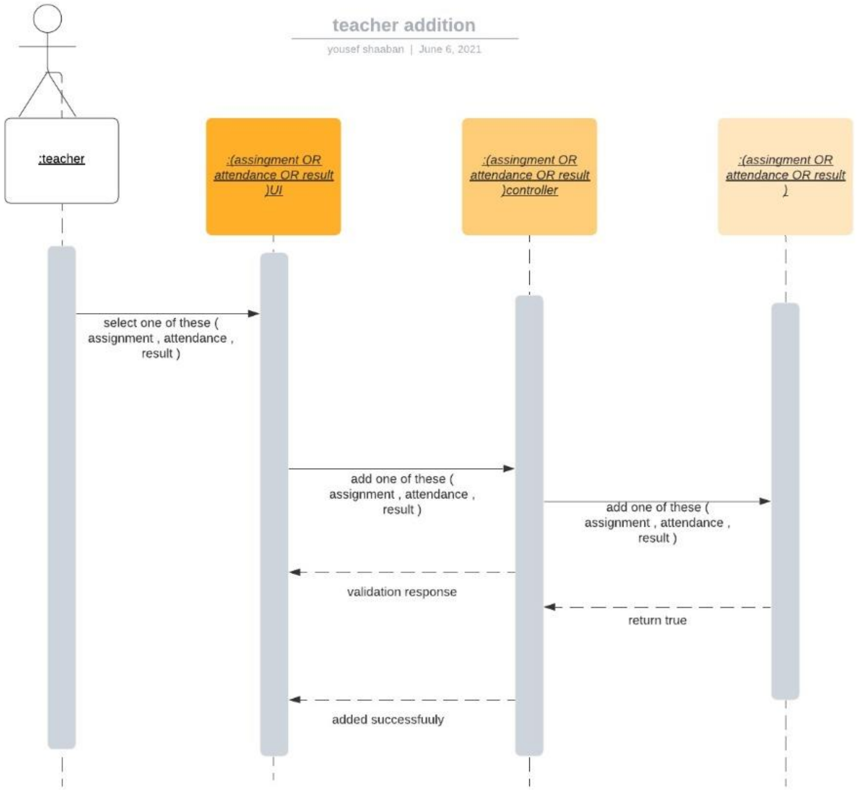

student
:(book OR timetable OR assignment OR profile OR attendance OR event OR result)UI
:(book OR timetable OR assignment OR profile OR attendance OR event OR result)controller
:(book OR timetable OR assignment OR profile OR attendance OR event OR result)
select one of these (book , time table , assignment , profile , attendance , event result
view one of these (book , time table , assignment , profile , attendance , event result
view one of these (book , time table , assignment , profile , attendance , event result
return true

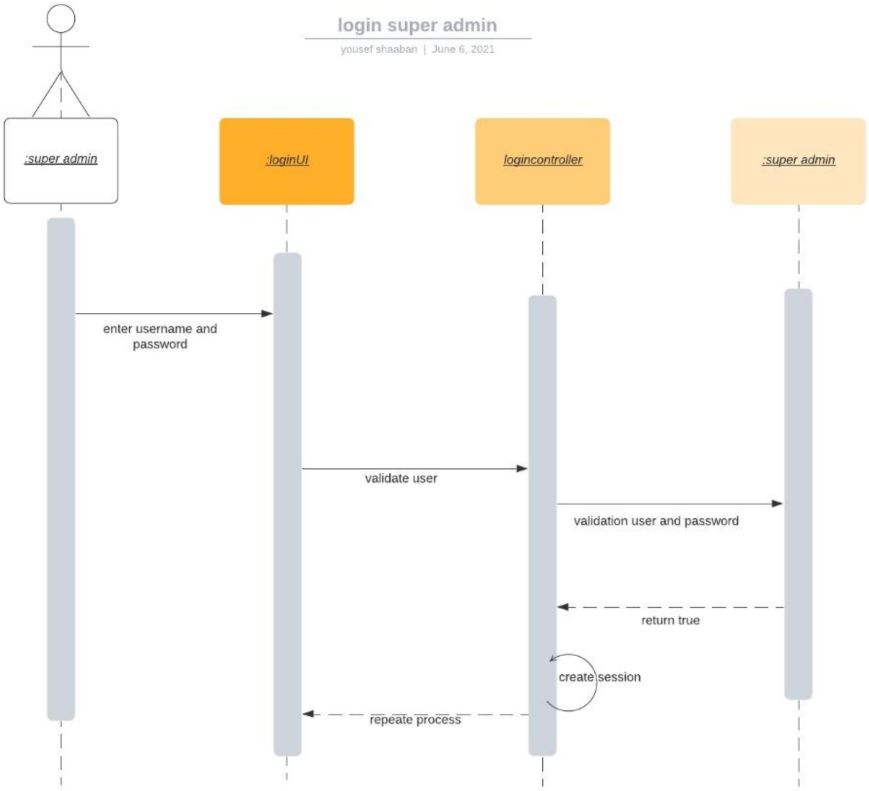

student login
yousef shaaban | June 6, 2021
:student
loginUI
logincontroller
:student
enter username and password
validate user
validate user
return true
create session
return true

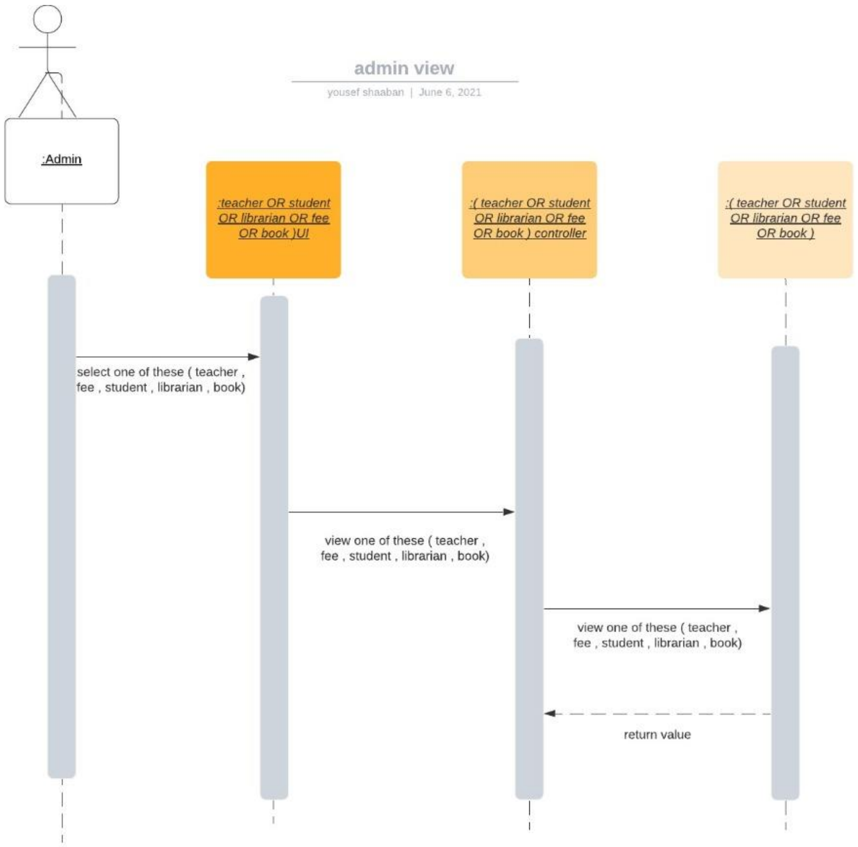


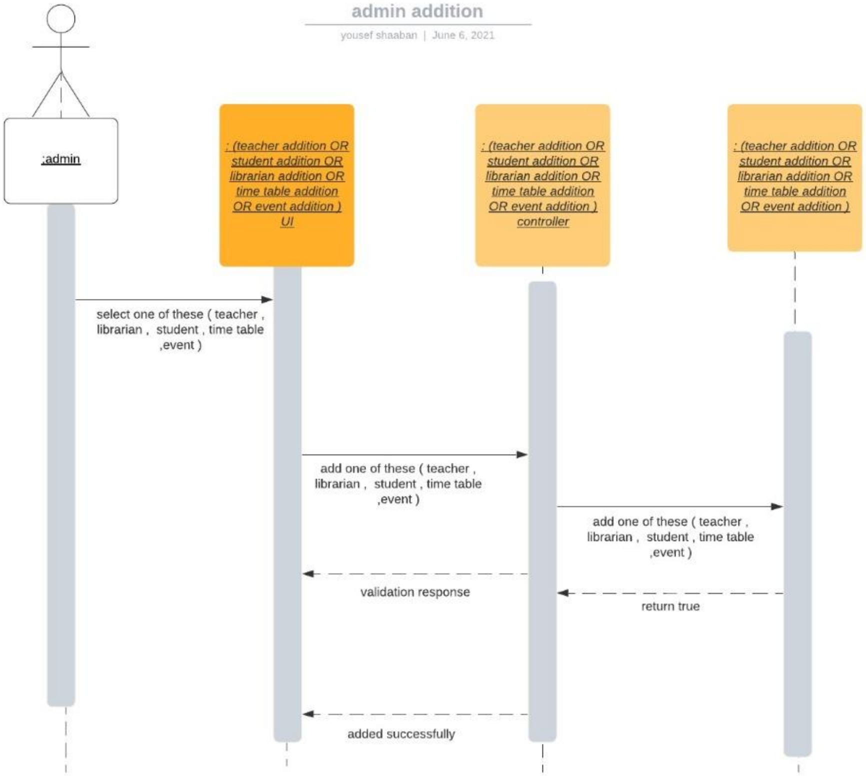
admin login

yousef shaaban | June 6, 2021


teacher addition
yousef shaaban | June 6, 2021
:teacher
:(assingment OR attendance OR result)UI
:(assingment OR attendance OR result)controller
:(assingment OR attendance OR result)
select one of these (assignment , attendance , result)
add one of these (assignment , attendance , result)
add one of these (assignment , attendance , result)
validation response
return true
added successfuuly


login super admin
yousef shaaban | June 6, 2021
:super admin
:loginUI
logincontroller
:super admin
enter username and password
validate user
validation user and password
return true
create session
repeate process

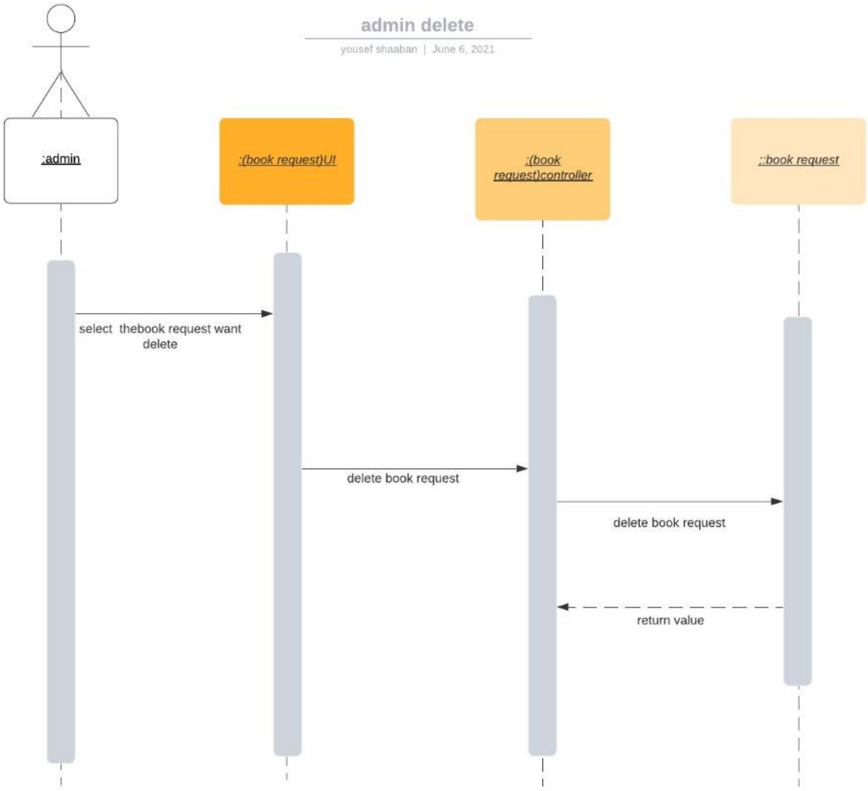

admin view
yousef shaaban | June 6, 2021
:Admin
:teacher OR student OR librarian OR fee OR book)UI
:(teacher OR student OR librarian OR fee OR book) controller
:(teacher OR student OR librarian OR fee OR book)
select one of these (teacher , fee , student , librarian , book)
view one of these (teacher , fee , student , librarian , book)
view one of these (teacher , fee , student , librarian , book)
return value



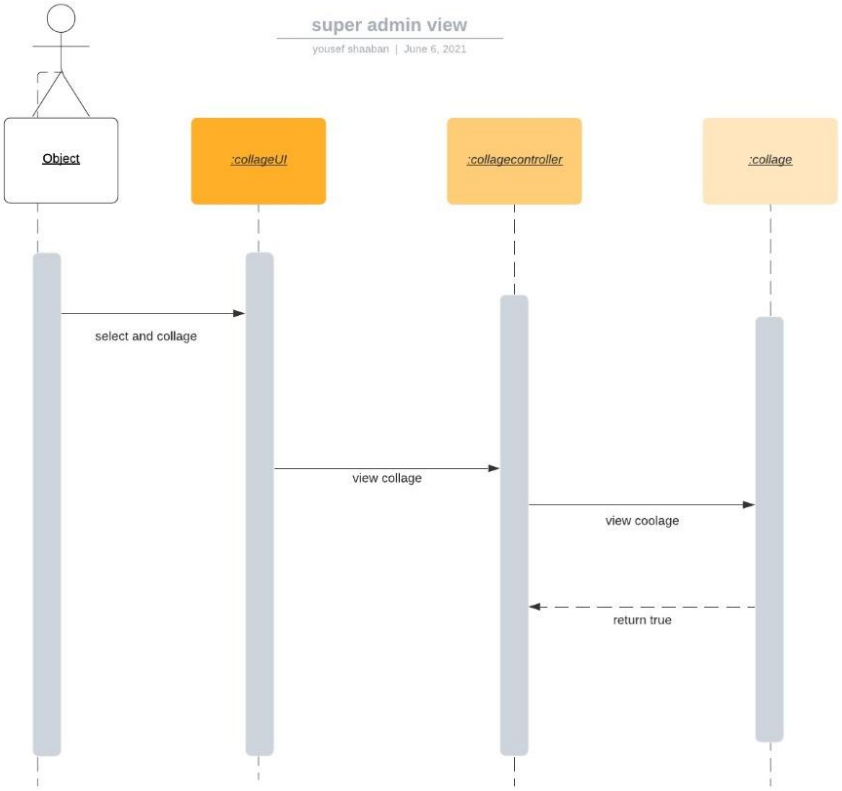
admin addition

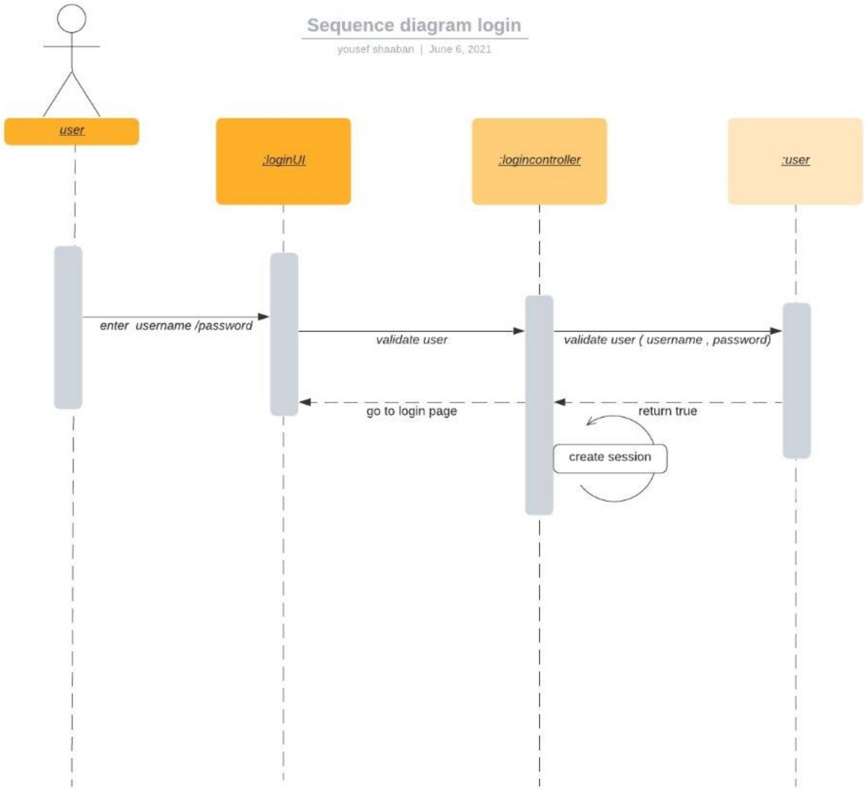
yousef shaaban | June 6, 2021

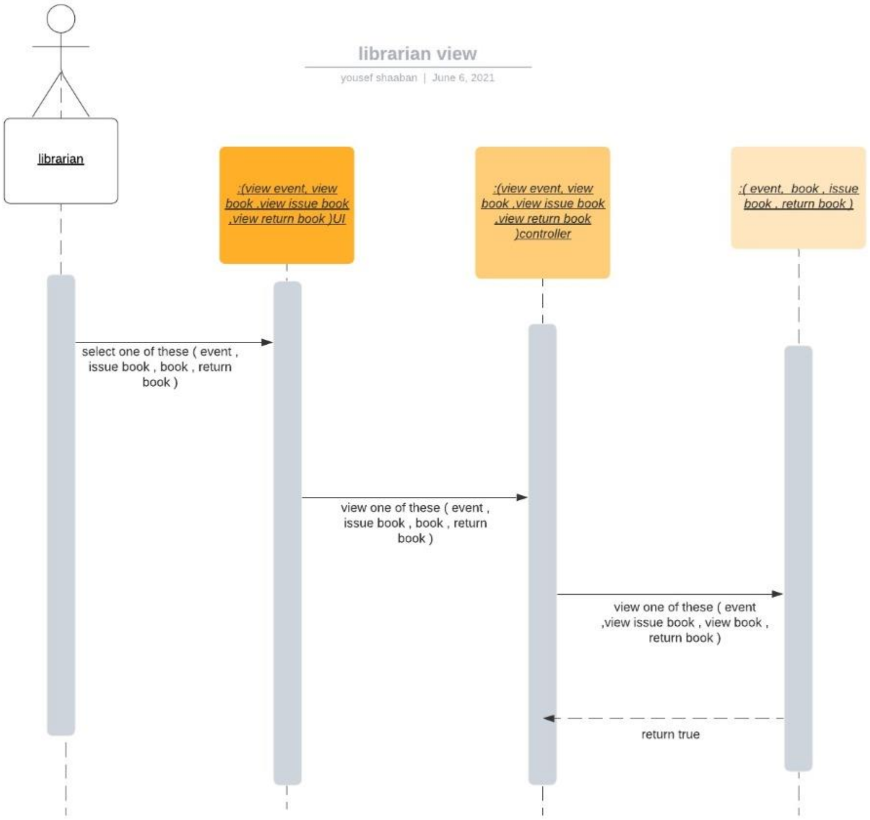
:admin
: (teacher addition OR student addition OR librarian addition OR time table addition OR event addition) UI
: (teacher addition OR student addition OR librarian addition OR time table addition OR event addition) controller
: (teacher addition OR student addition OR librarian addition OR time table addition OR event addition)
select one of these (teacher , librarian , student , time table ,event)
add one of these (teacher , librarian , student , time table ,event)
add one of these (teacher , librarian , student , time table ,event)
validation response
return true
added successfully


admin delete
yousef shaaban | June 6, 2021
:admin
:(book request)UI
:(book request)controller
::book request
select thebook request want delete
delete book request
delete book request
return value

SSS


super admin view
yousef shaaban | June 6, 2021
Object
:collageUI
:collagecontroller
:collage
select and collage
view collage
view coolage
return true

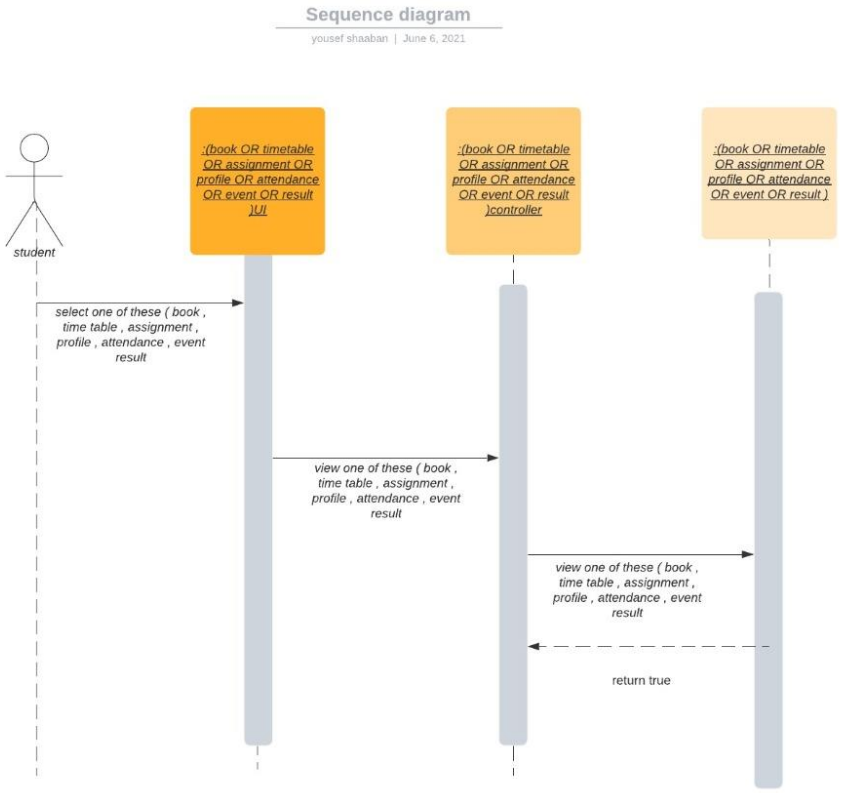

Sequence diagram login
yousef shaaban | June 6, 2021
user
:loginUI
:logincontroller
:user
enter username /password
validate user
validate user (username , password)
go to login page
return true
create session

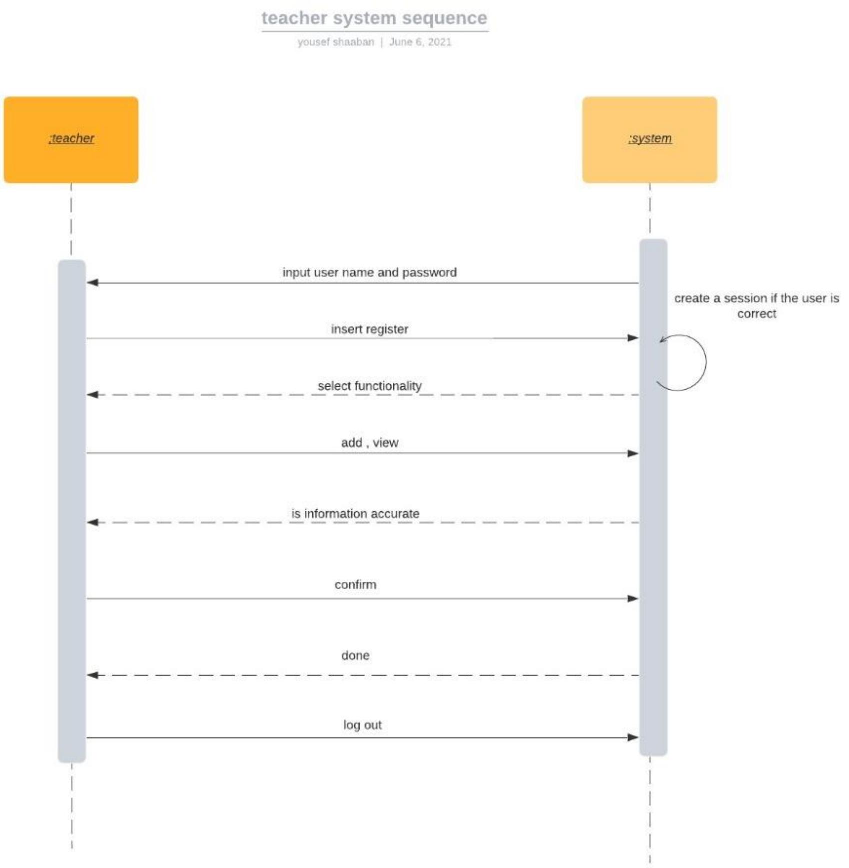


librarian view

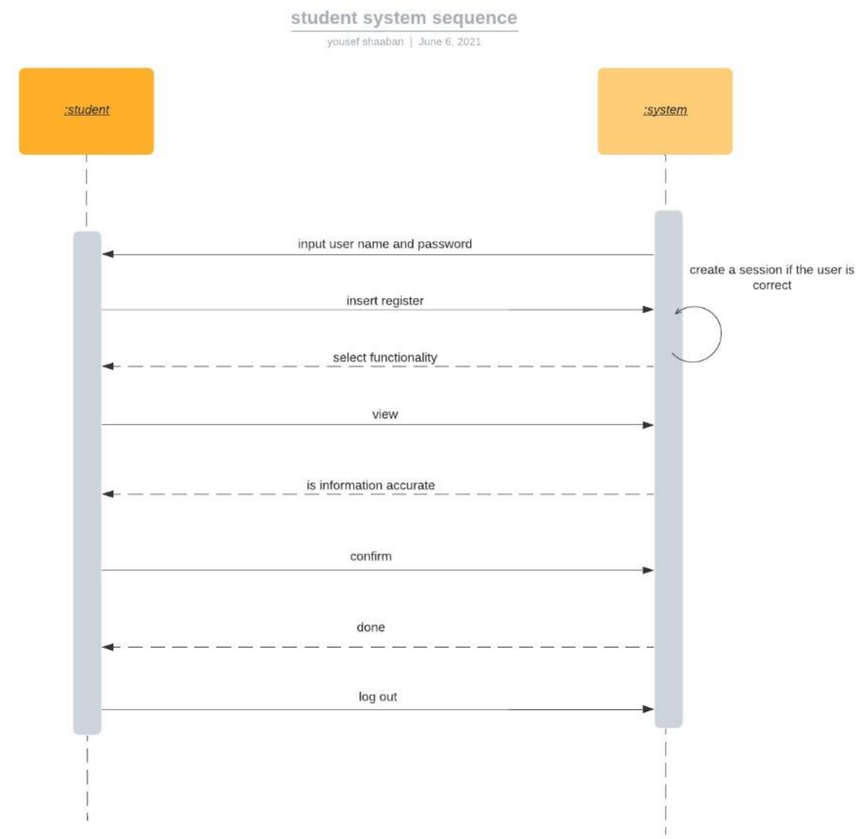
yousef shaaban | June 6, 2021

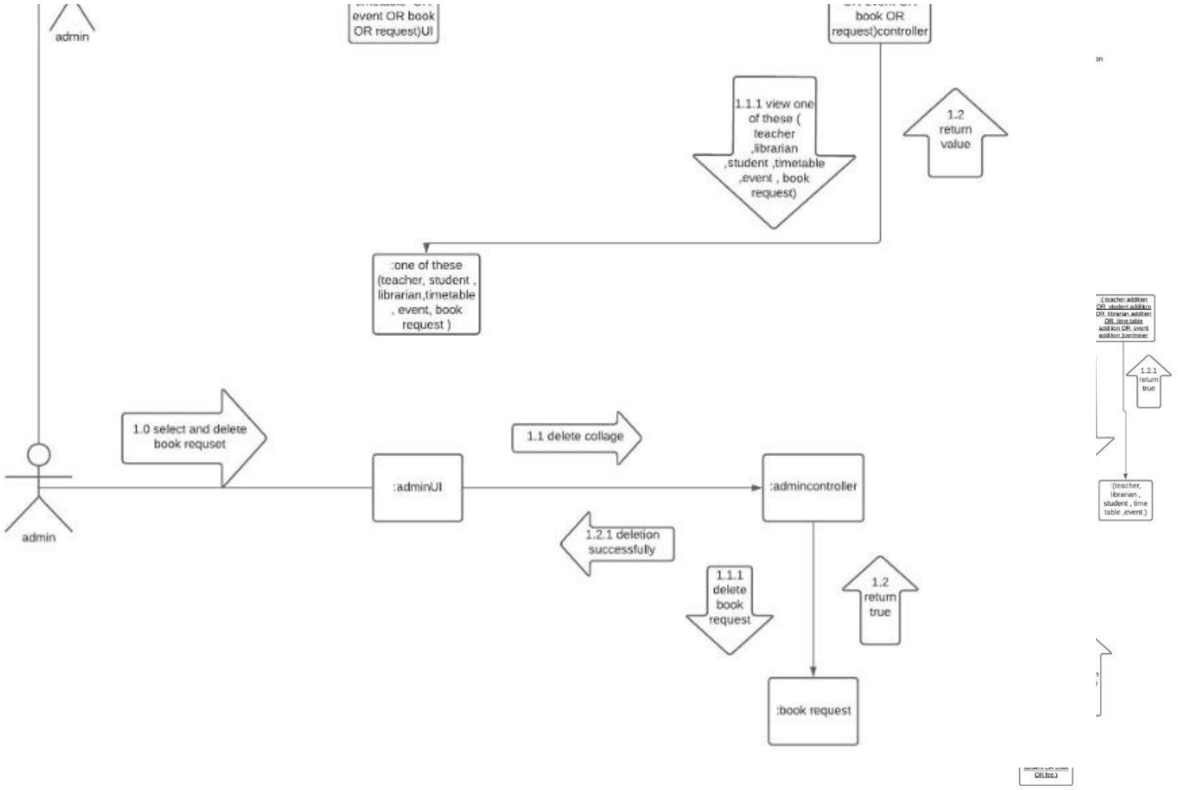
librarian
:(view event, view book ,view issue book ,view return book)UI
:(view event, view book ,view issue book ,view return book)controller
:(event, book , issue book , return book)
select one of these (event , issue book , book , return book)
view one of these (event , issue book , book , return book)
view one of these (event ,view issue book , view book , return book)
return true


Sequence diagram
yousef shaaban | June 6, 2021
:(book OR timetable OR assignment OR profile OR attendance OR event OR result)UI
:(book OR timetable OR assignment OR profile OR attendance OR event OR result)controller
:(book OR timetable OR assignment OR profile OR attendance OR event OR result)
student
select one of these (book , time table , assignment , profile , attendance , event result
view one of these (book , time table , assignment , profile , attendance , event result
view one of these (book , time table , assignment , profile , attendance , event result
return true

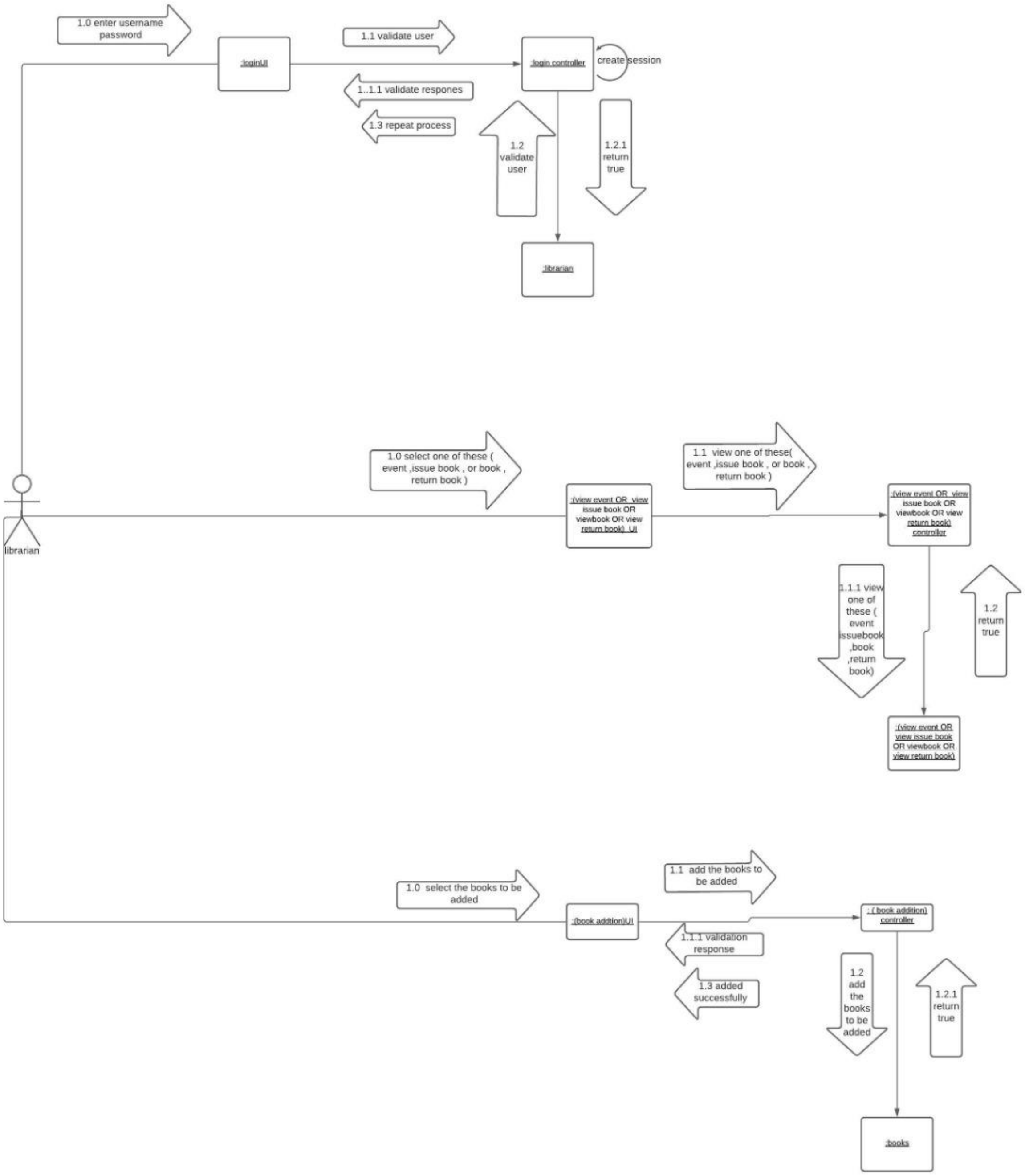


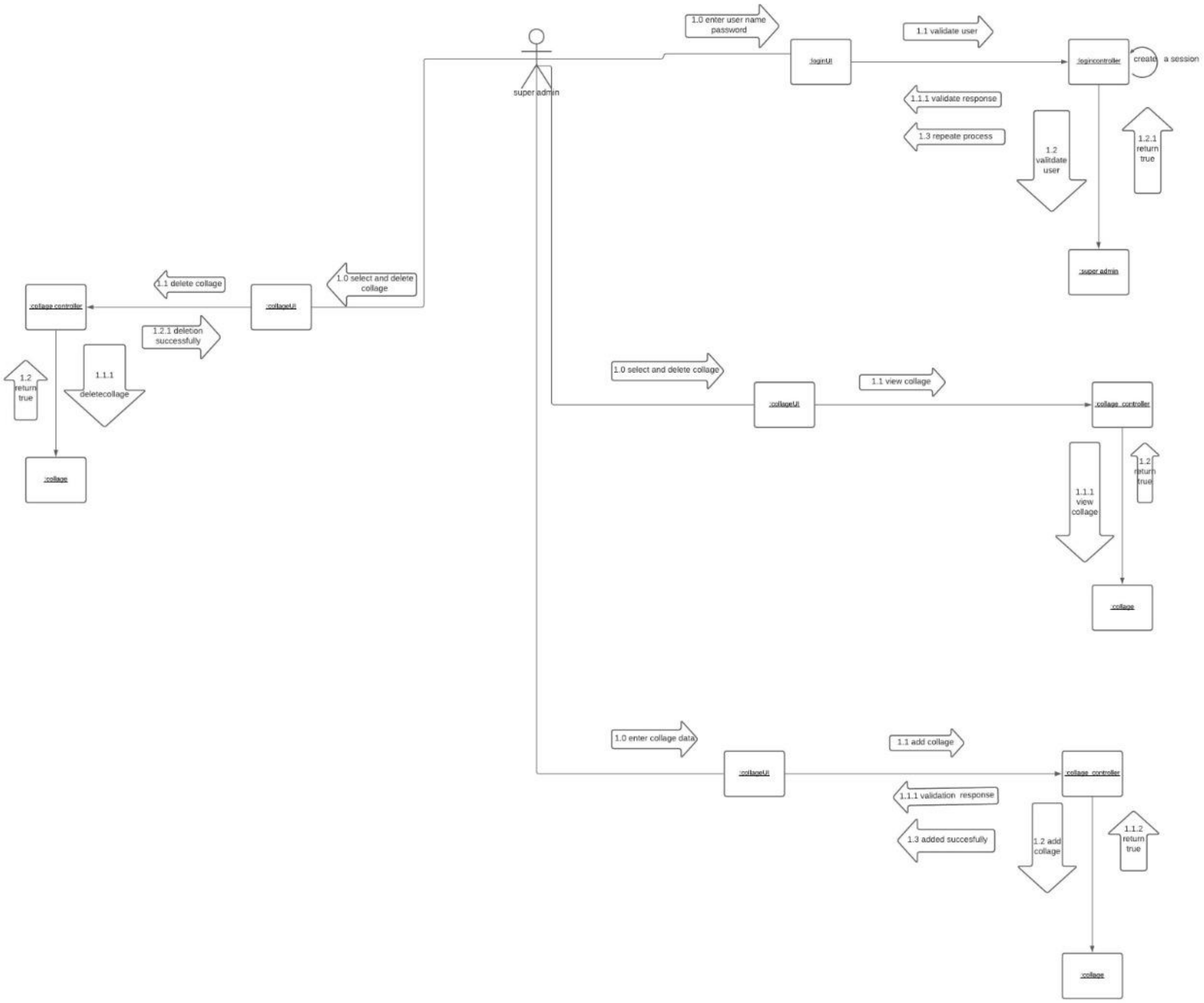
System Sequence Diagram

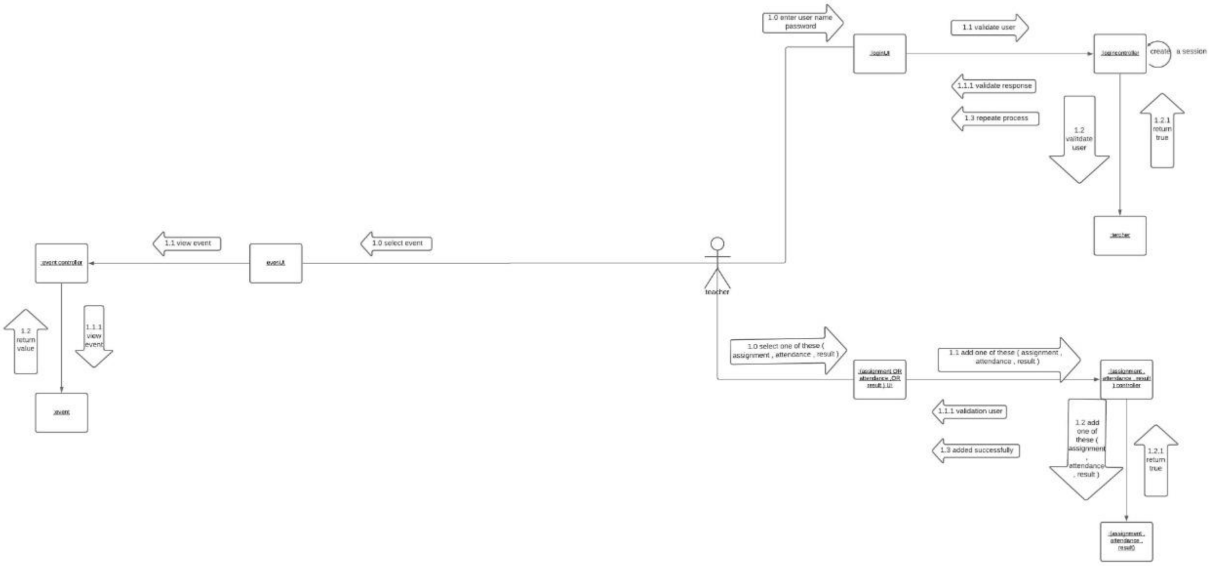

student system sequence
yousef shaaban | June 6, 2021
:student
:system
input user name and password
create a session if the user is correct
insert register
select functionality
view
is information accurate
confirm
done
log out

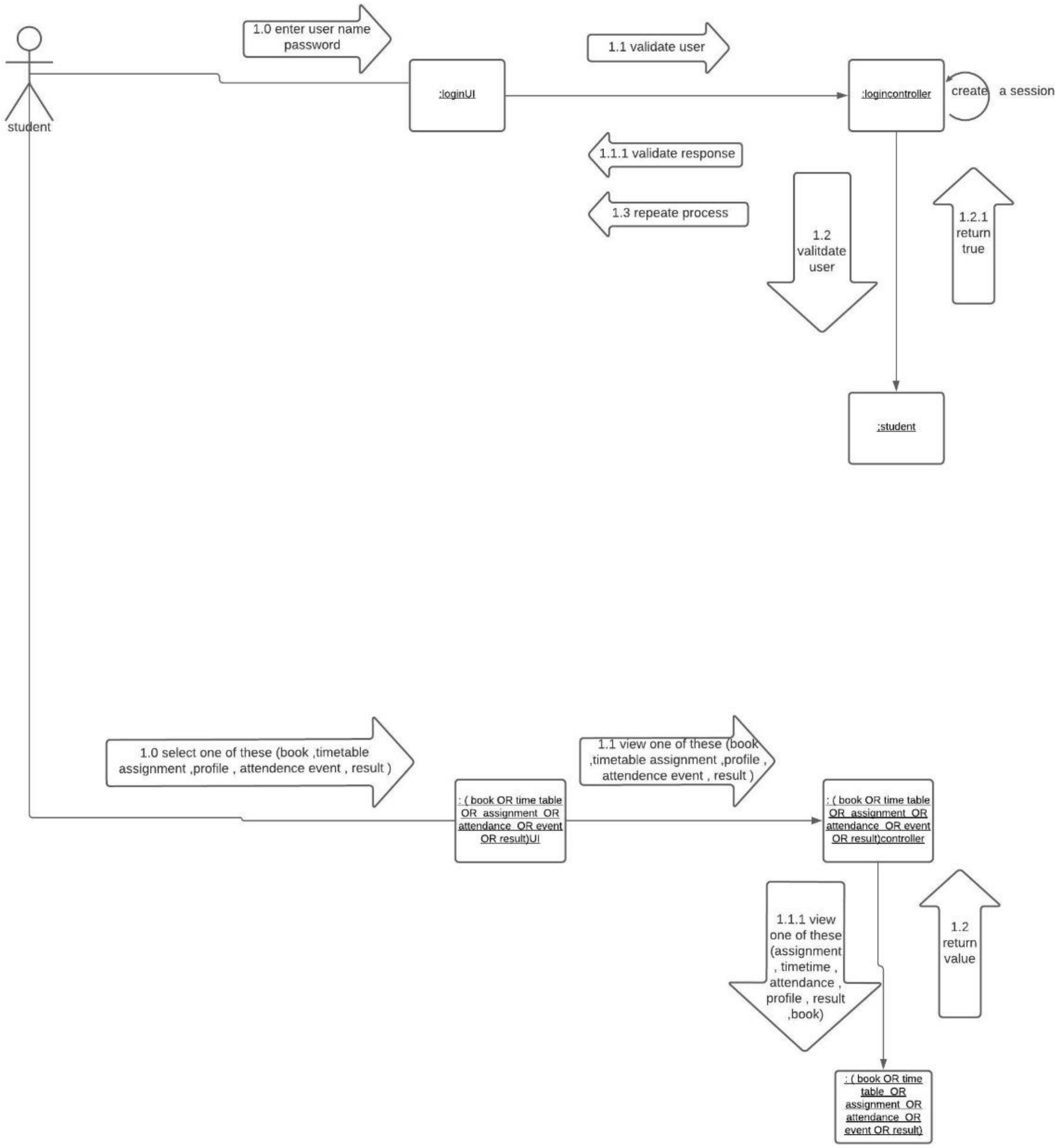

admin
event OR book OR request)UI
book OR request)controller
1.1.1 view one of these (teacher ,librarian ,student ,timetable ,event , book request)
1.2 return value
:one of these (teacher, student , librarian,timetable , event, book request)
:(teacher addition OR student addition OR librarian addition OR time table addition OR event addition)controller
1.2.1 return true
:(teacher, librarian , student , time table ,event)
1.0 select and delete book requset
1.1 delete collage
admin
:adminUI
:admincontroller
1.2.1 deletion successfully
1.1.1 delete book request
1.2 return true
:book request

Communication Diagram

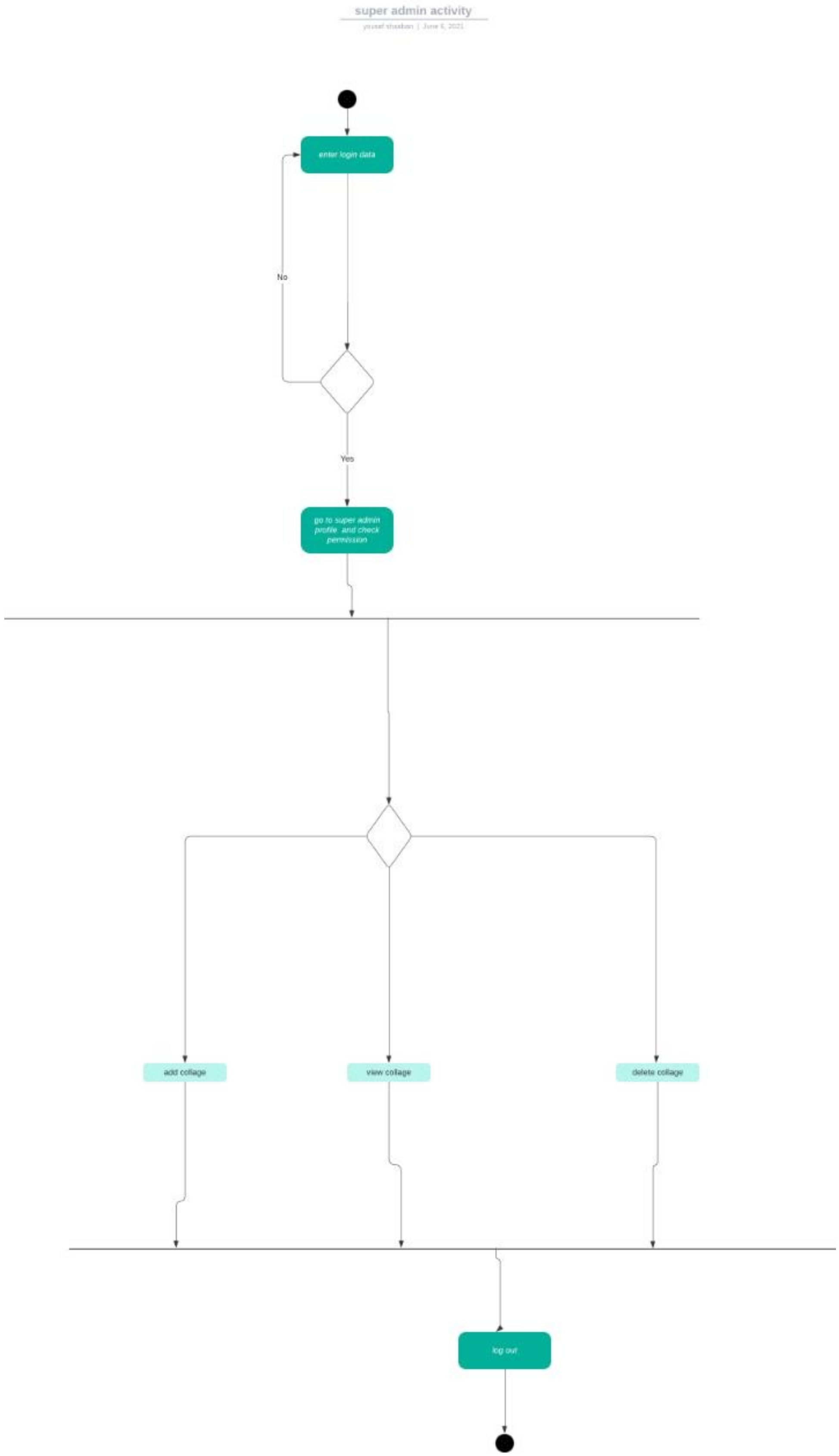

1.0 enter username password
:loginUI
1.1 validate user
:login controller
create session
1..1.1 validate respones
1.3 repeat process
1.2 validate user
1.2.1 return true
:librarian
librarian
1.0 select one of these (event ,issue book , or book , return book)
:(view event OR view issue book OR viewbook OR view return book) UI
1.1 view one of these(event ,issue book , or book , return book)
:(view event OR view issue book OR viewbook OR view return book) controller
1.1.1 view one of these (event issuebook ,book ,return book)
1.2 return true
:(view event OR view issue book OR viewbook OR view return book)
1.0 select the books to be added
:(book addtion)UI
1.1 add the books to be added
: (book addition) controller
1.1.1 validation response
1.3 added successfully
1.2 add the books to be added
1.2.1 return true
:books

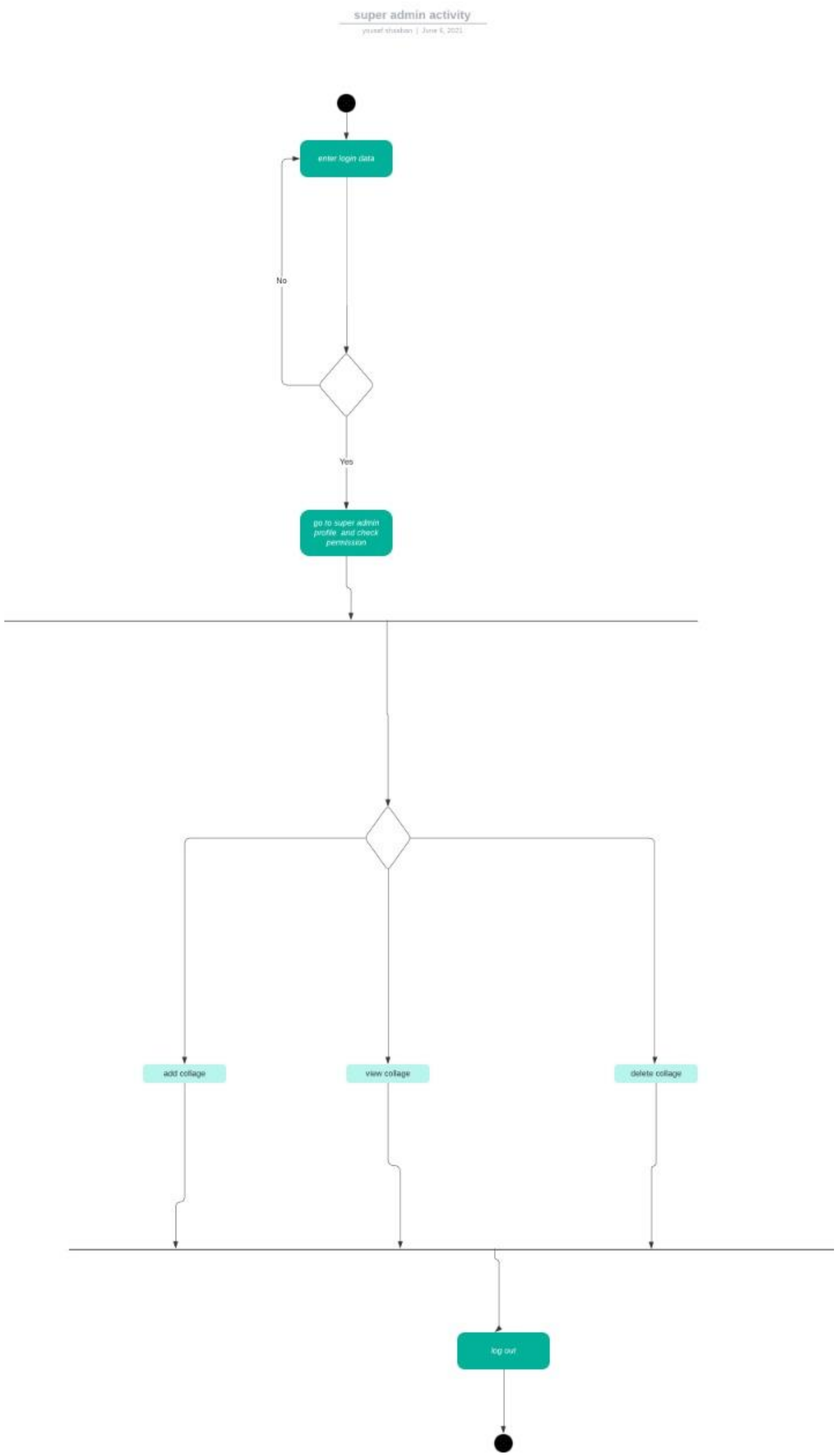

1.0 enter user name password
1.1 validate user
:loginUI
:logincontroller
create a session
super admin
1.1.1 validate response
1.3 repeate process
1.2 valitdate user
1.2.1 return true
:super admin
1.1 delete collage
1.0 select and delete collage
:collage controller
:collageUI
1.2.1 deletion successfully
1.2 return true
1.1.1 deletecollage
:collage
1.0 select and delete collage
1.1 view collage
:collageUI
:collage controller
1.1.1 view collage
1.2 return true
:collage
1.0 enter collage data
1.1 add collage
:collageUI
:collage controller
1.1.1 validation response
1.3 added succesfully
1.2 add collage
1.1.2 return true
:collage

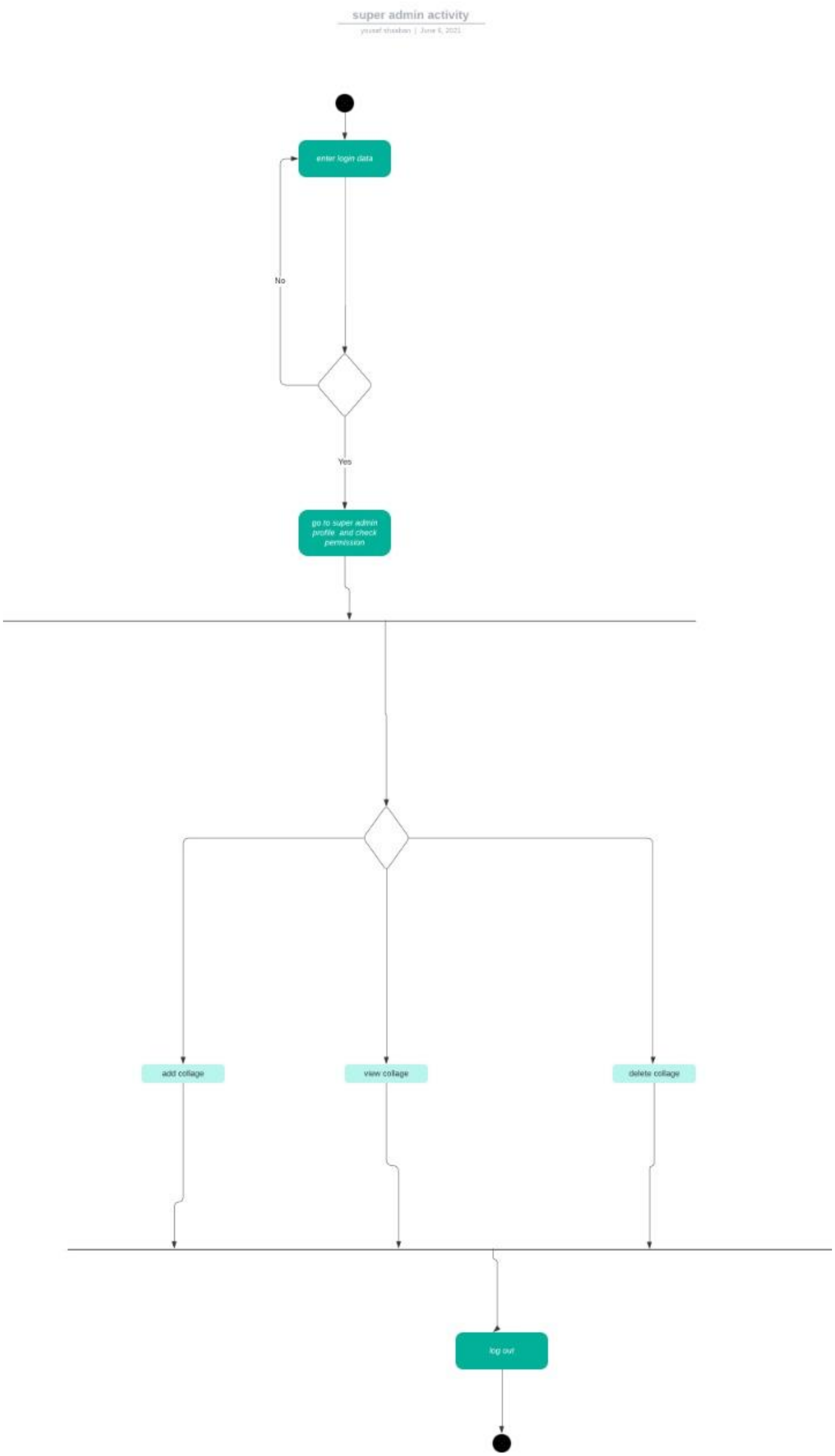

1.0 enter user name password
1.1 validate user
:loginUI
:logincontroller
create a session
1.1.1 validate response
1.3 repeate process
1.2 valitdate user
1.2.1 return true
:teacher
:event controller
1.1 view event
eventUI
1.0 select event
teacher
1.2 return value
1.1.1 view event
:event
1.0 select one of these (assignment , attendance , result)
:(assignment OR attendance ,OR result) UI
1.1 add one of these (assignment , attendance , result)
:(assignment , attendance , result) controller
1.1.1 validation user
1.3 added successfully
1.2 add one of these (assignment , attendance , result)
1.2.1 return true
:(assignment , attendance , result)


student
1.0 enter user name password
:loginUI
1.1 validate user
:logincontroller
create a session
1.1.1 validate response
1.3 repeate process
1.2 valitdate user
1.2.1 return true
:student
1.0 select one of these (book ,timetable assignment ,profile , attendence event , result)
: (book OR time table OR assignment OR attendance OR event OR result)UI
1.1 view one of these (book ,timetable assignment ,profile , attendence event , result)
: (book OR time table OR assignment OR attendance OR event OR result)controller
1.1.1 view one of these (assignment , timetime , attendance , profile , result ,book)
1.2 return value
: (book OR time table OR assignment OR attendance OR event OR result)

Activity Diagram Updated Version

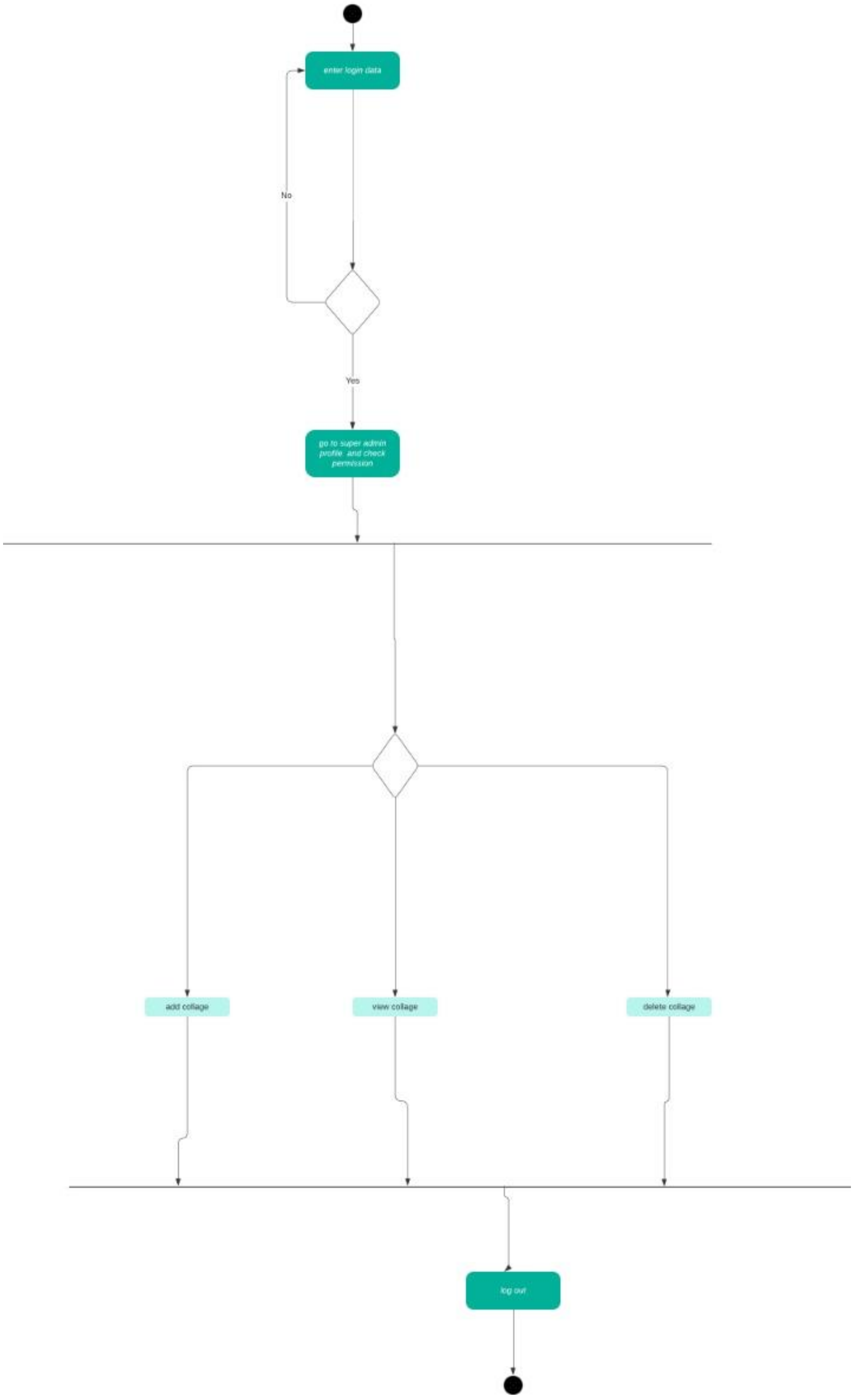

super admin activity
yousef shaaban | June 6, 2021
enter login data
No
Yes
go to super admin profile and check permission
add collage
view collage
delete collage
log out

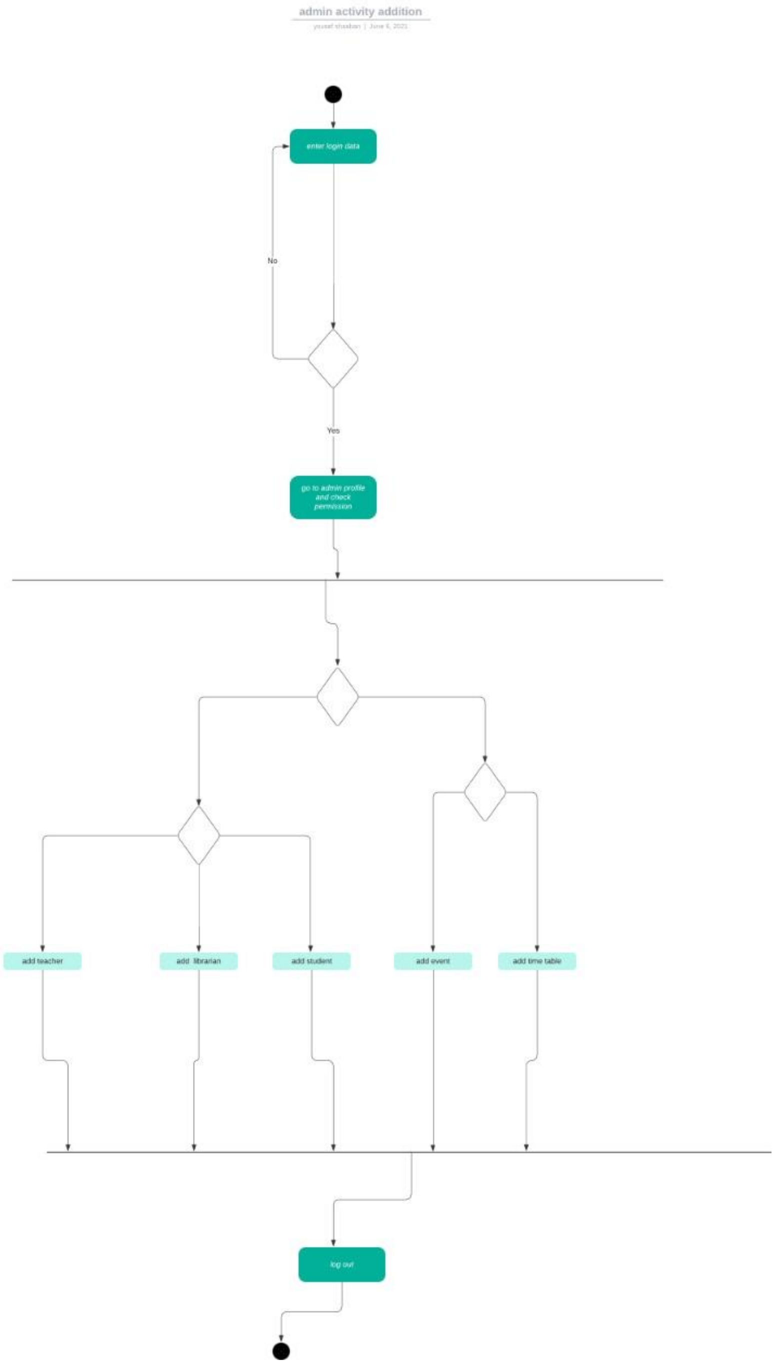

super admin activity
yousef shaaban | June 6, 2021
enter login data
No
Yes
go to super admin profile and check permission
add collage
view collage
delete collage
log out


super admin activity
yousef shaaban | June 6, 2021
enter login data
No
Yes
go to super admin profile and check permission
add collage
view collage
delete collage
log out

super admin activity

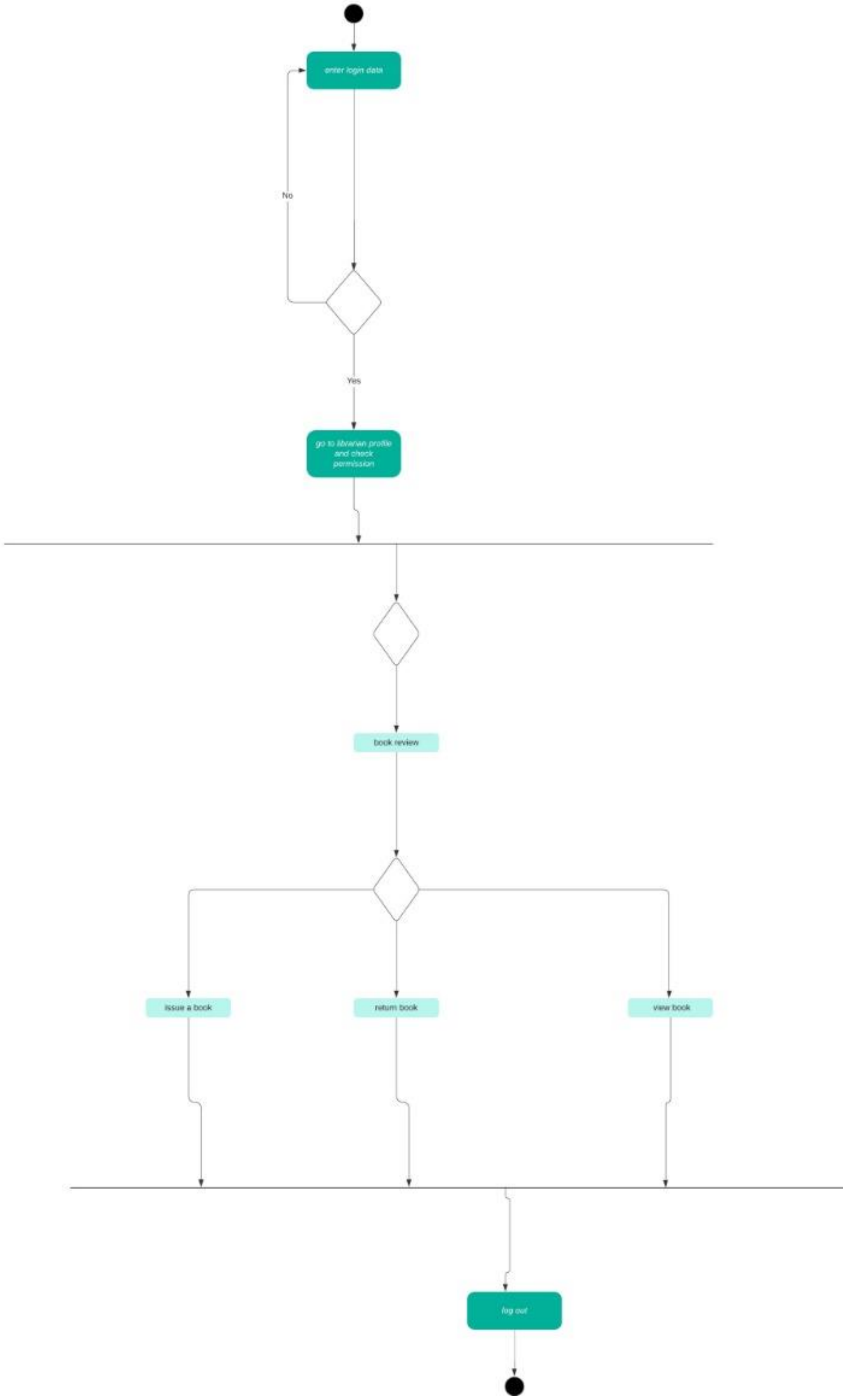
yousef shaaban | June 6, 2021

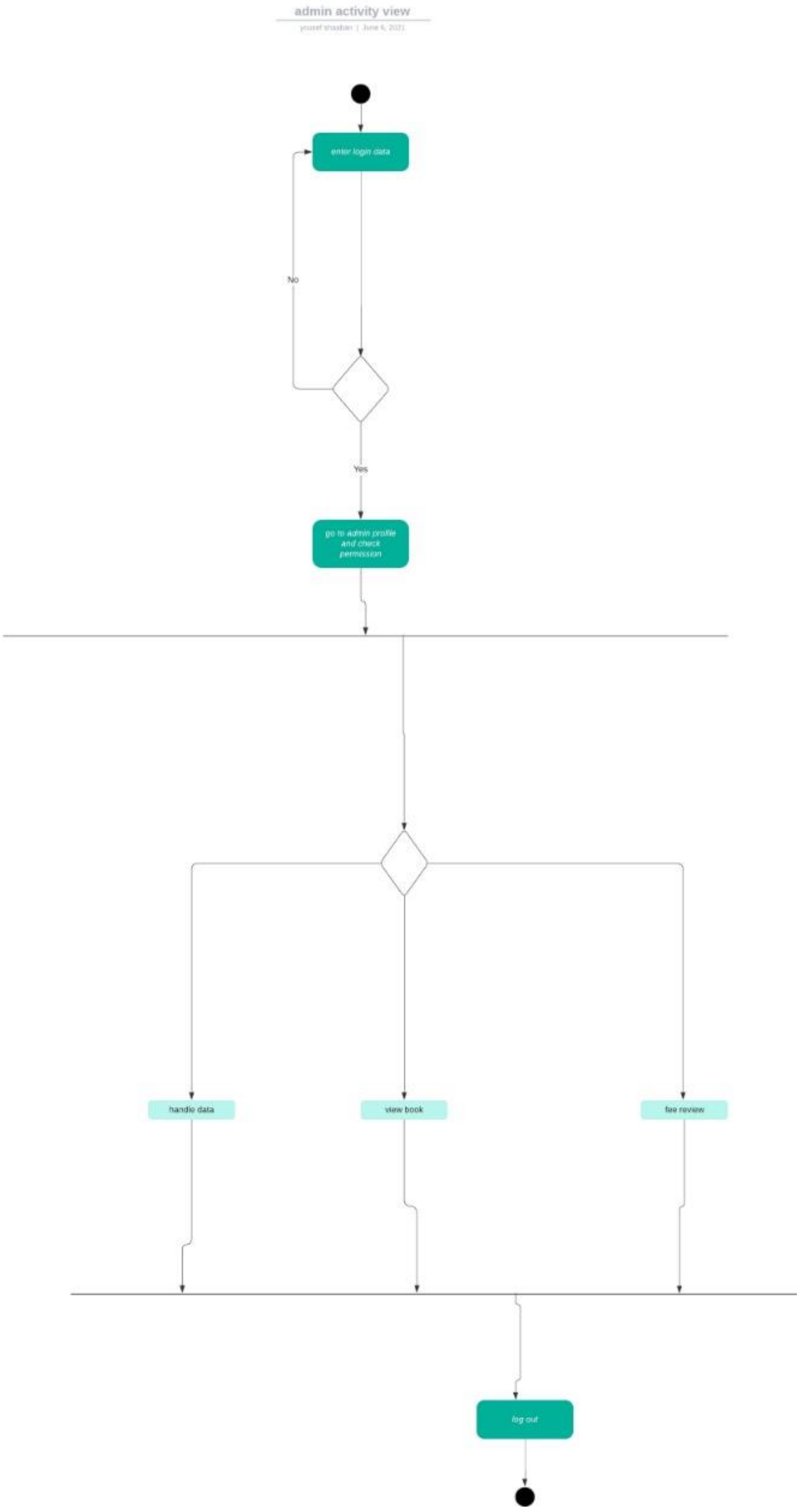


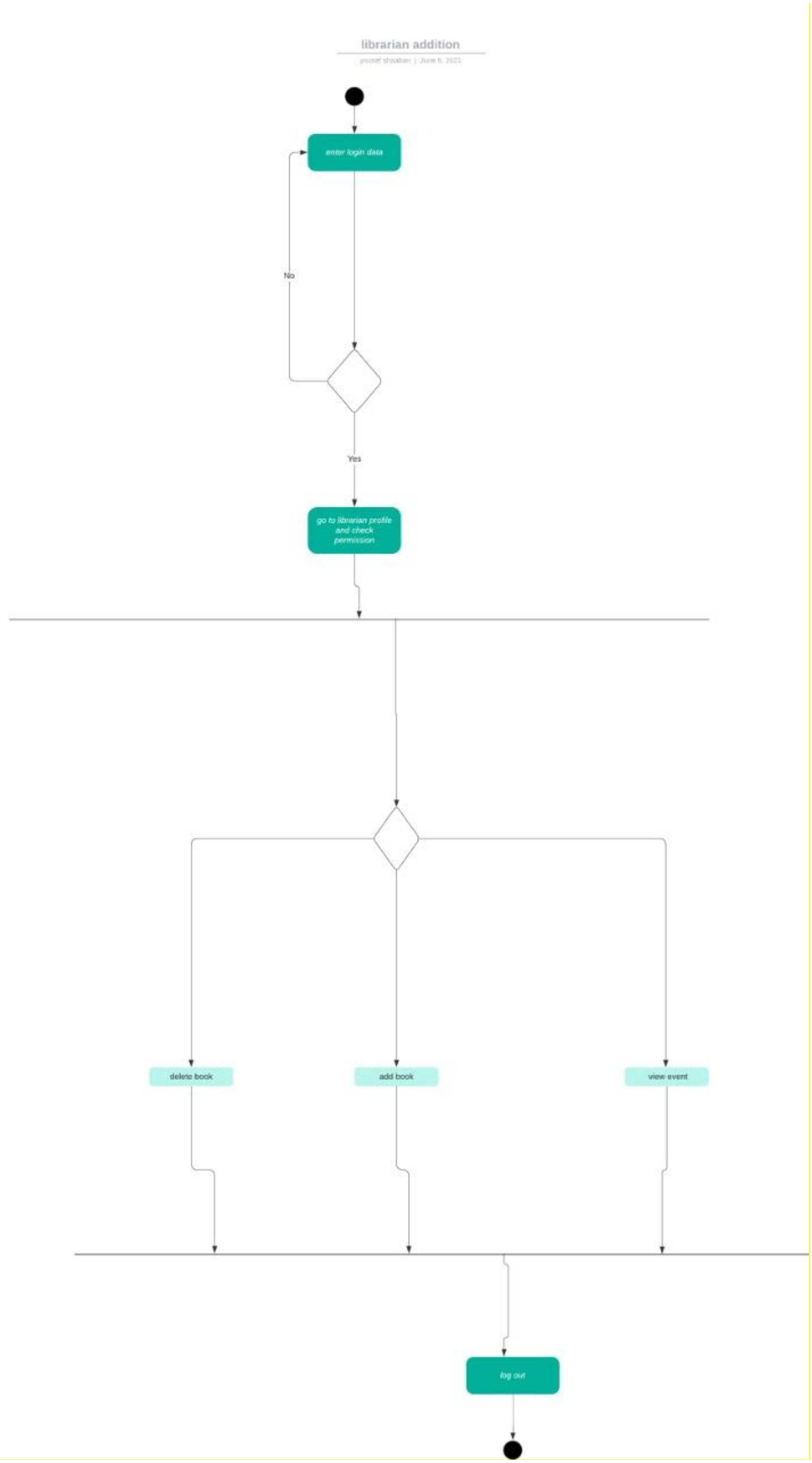

admin activity addition
yousef shaaban | June 6, 2021
enter login data
No
Yes
go to admin profile and check permission
add teacher
add librarian
add student
add event
add time table
log out

librarian activity review

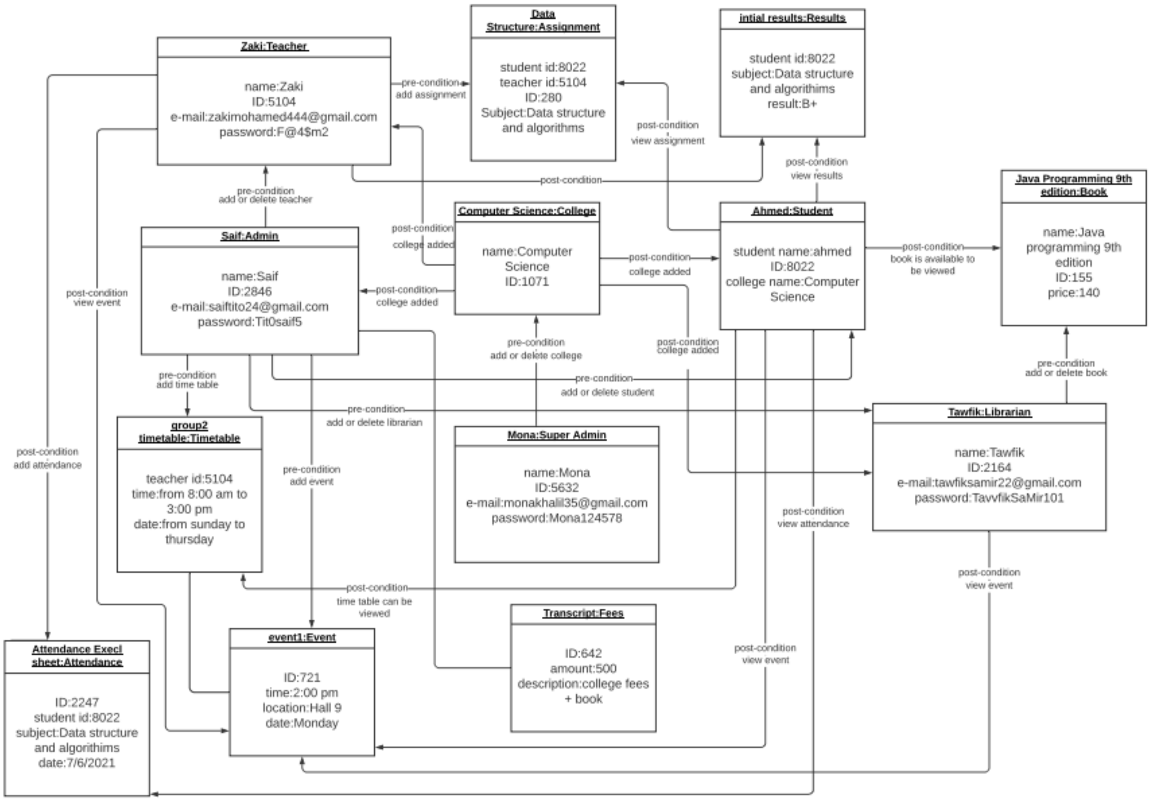
yousef droubian | June 6, 2021



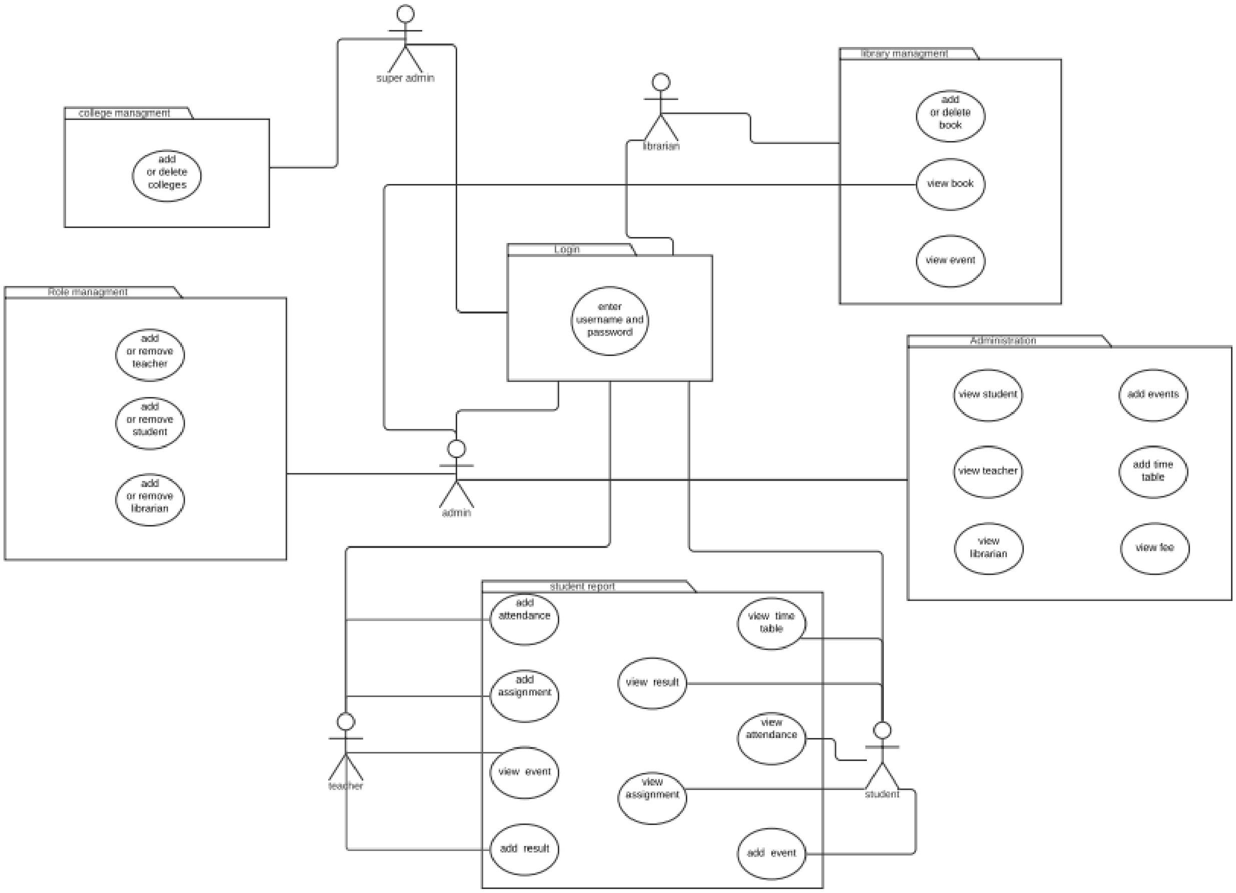

admin activity view
yousef shaaban | June 6, 2021
enter login data
No
Yes
go to admin profile and check permission
handle data
view book
fee review
log out


librarian addition
yousef shaaban | June 6, 2021
enter login data
No
Yes
go to librarian profile and check permission
delete book
add book
view event
log out

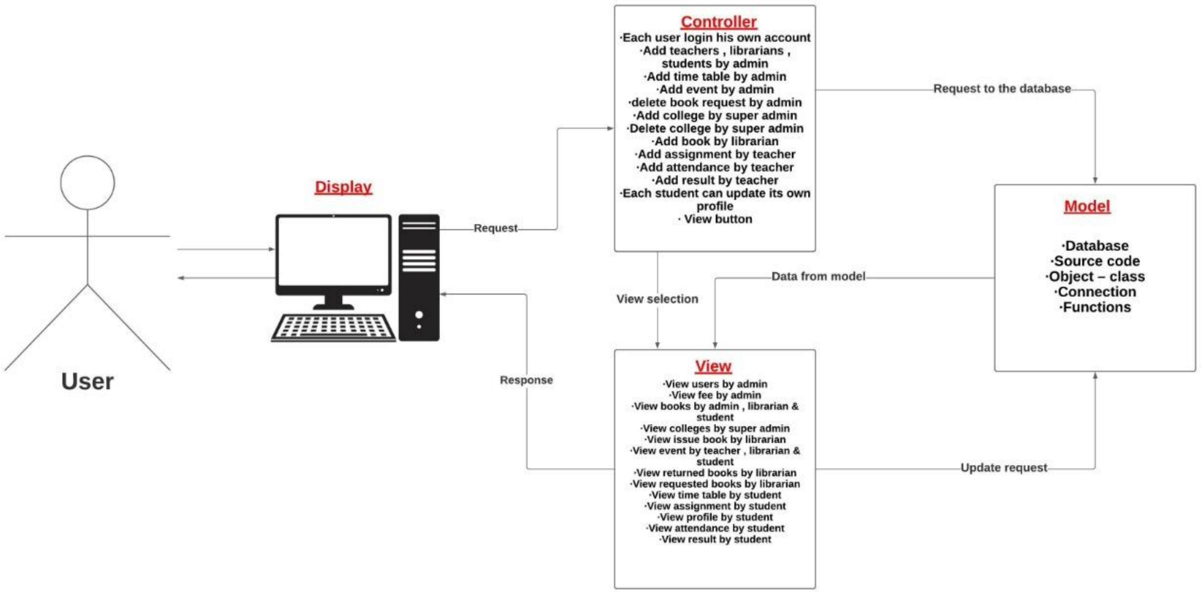
Object Diagram

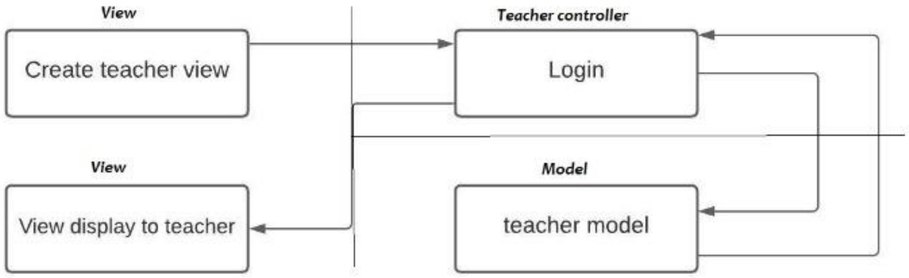


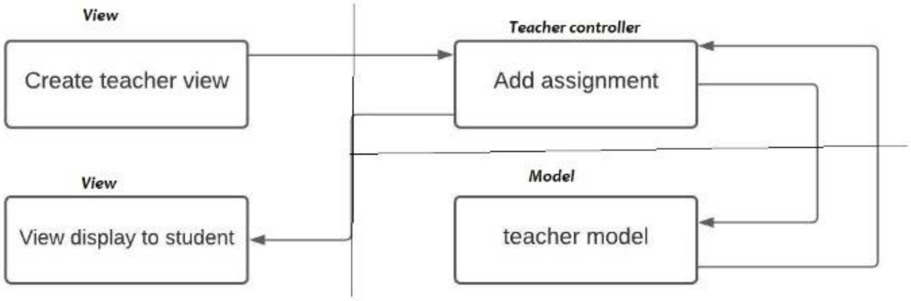
Package Diagram

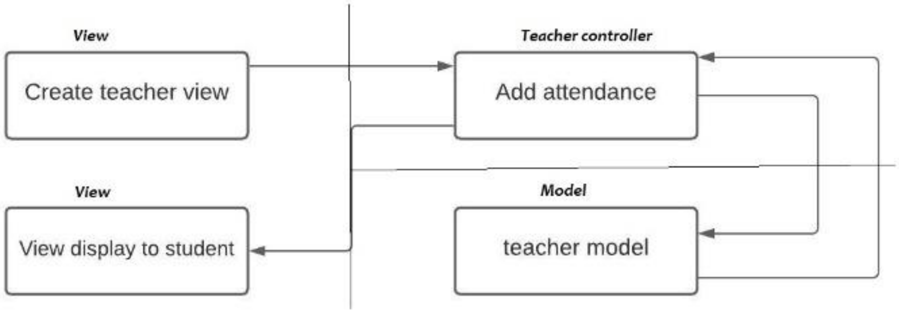


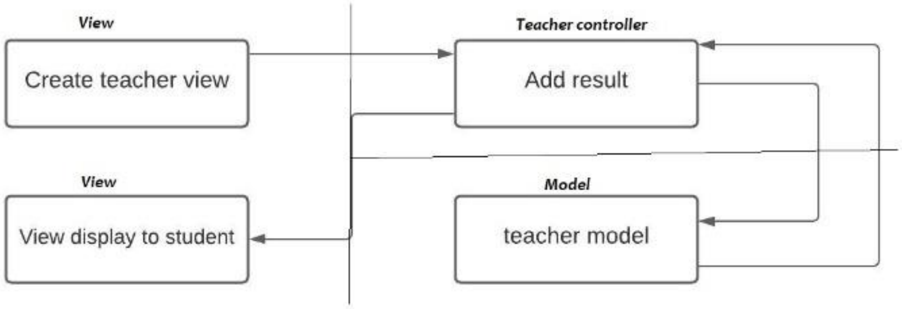
Architecture Diagram

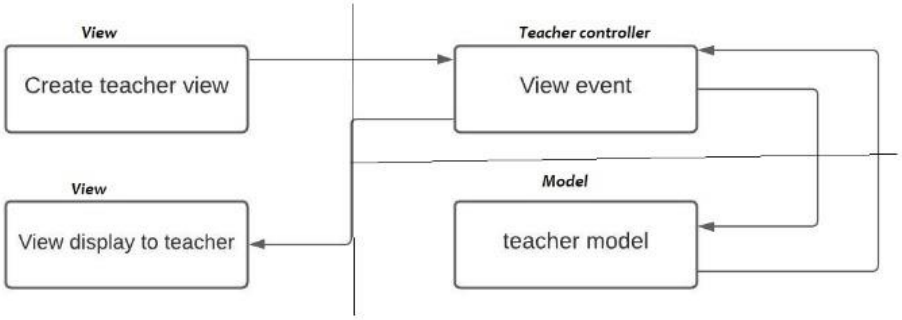

User
Display
Request
Response
Controller
·Each user login his own account
·Add teachers , librarians , students by admin
·Add time table by admin
·Add event by admin
·delete book request by admin
·Add college by super admin
·Delete college by super admin
·Add book by librarian
·Add assignment by teacher
·Add attendance by teacher
·Add result by teacher
·Each student can update its own profile
· View button
Request to the database
View selection
Data from model
Model
·Database
·Source code
·Object – class
·Connection
·Functions
View
·View users by admin
·View fee by admin
·View books by admin , librarian & student
·View colleges by super admin
·View issue book by librarian
·View event by teacher , librarian & student
·View returned books by librarian
·View requested books by librarian
·View time table by student
·View assignment by student
·View profile by student
·View attendance by student
·View result by student
Update request

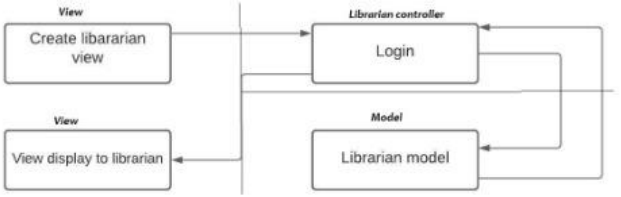

View
Create teacher view
Teacher controller
Login
View
View display to teacher
Model
teacher model

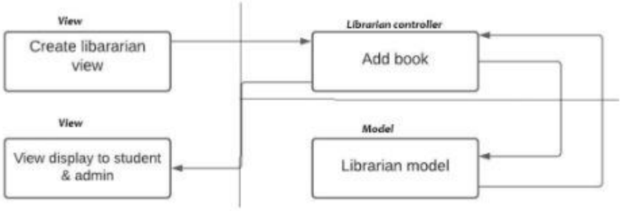

View
Create teacher view
Teacher controller
Add assignment
View
View display to student
Model
teacher model

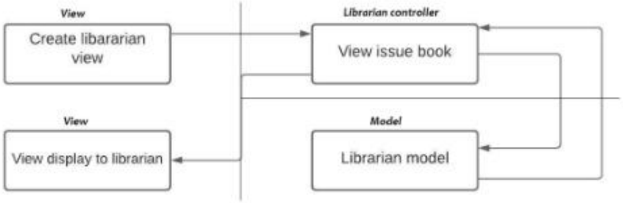

View
Create teacher view
Teacher controller
Add attendance
View
View display to student
Model
teacher model

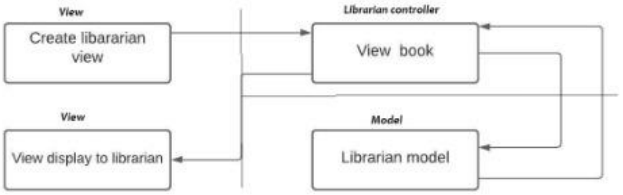

View
Create teacher view
Teacher controller
Add result
View
View display to student
Model
teacher model

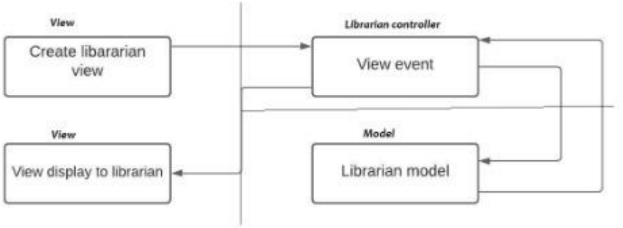

View
Create teacher view
Teacher controller
View event
View
View display to teacher
Model
teacher model

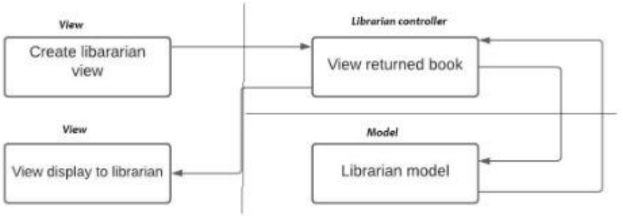

View
Create libararian view
Librarian controller
Login
View
View display to librarian
Model
Librarian model

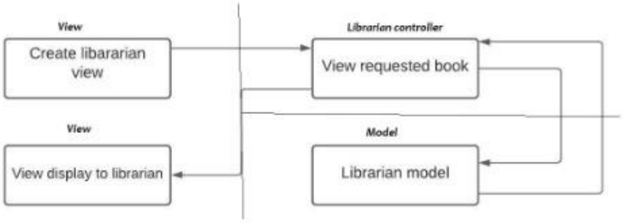

View
Create libararian view
Librarian controller
Add book
View
View display to student & admin
Model
Librarian model

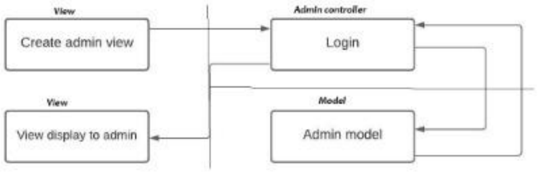

View
Create libararian view
Librarian controller
View issue book
View
View display to librarian
Model
Librarian model

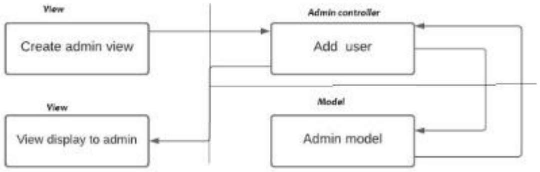

View
Create libararian view
Librarian controller
View book
View
View display to librarian
Model
Librarian model

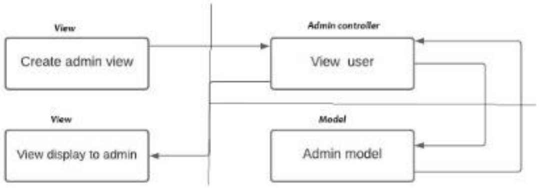

View
Create libararian view
Librarian controller
View event
View
View display to librarian
Model
Librarian model

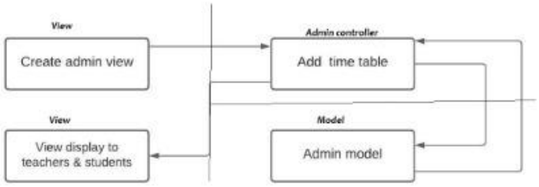

View
Create libararian view
Librarian controller
View returned book
View
View display to librarian
Model
Librarian model

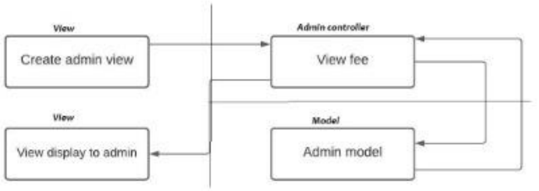

View
Create libararian view
Librarian controller
View requested book
View
View display to librarian
Model
Librarian model

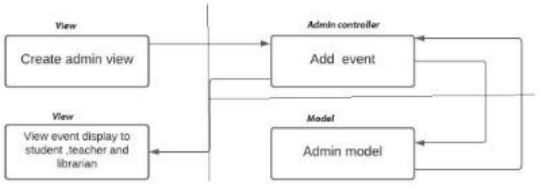

View
Create admin view
Admin controller
Login
View
View display to admin
Model
Admin model

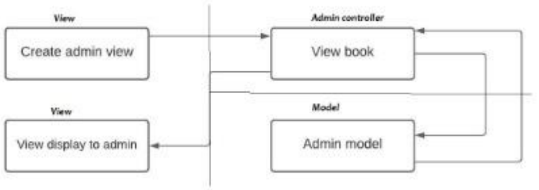

View
Create admin view
Admin controller
Add user
View
View display to admin
Model
Admin model

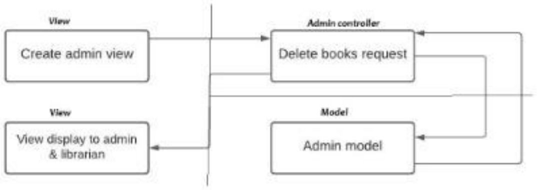

View
Create admin view
Admin controller
View user
View
View display to admin
Model
Admin model

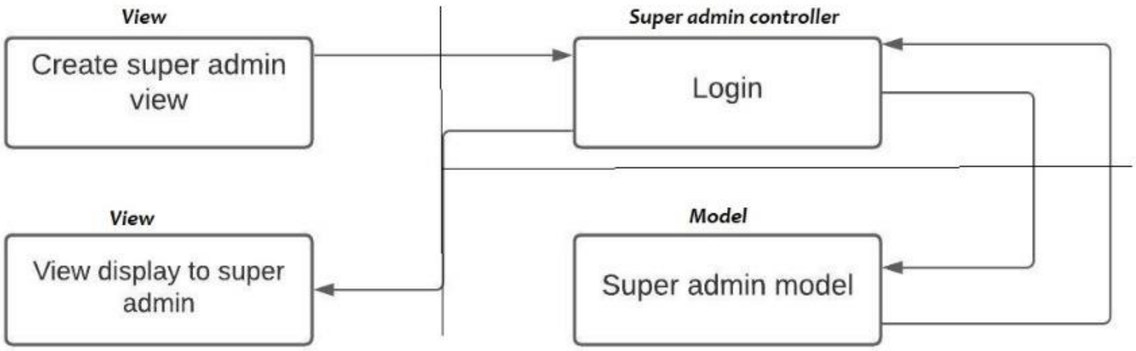

View
Create admin view
Admin controller
Add time table
View
View display to teachers & students
Model
Admin model

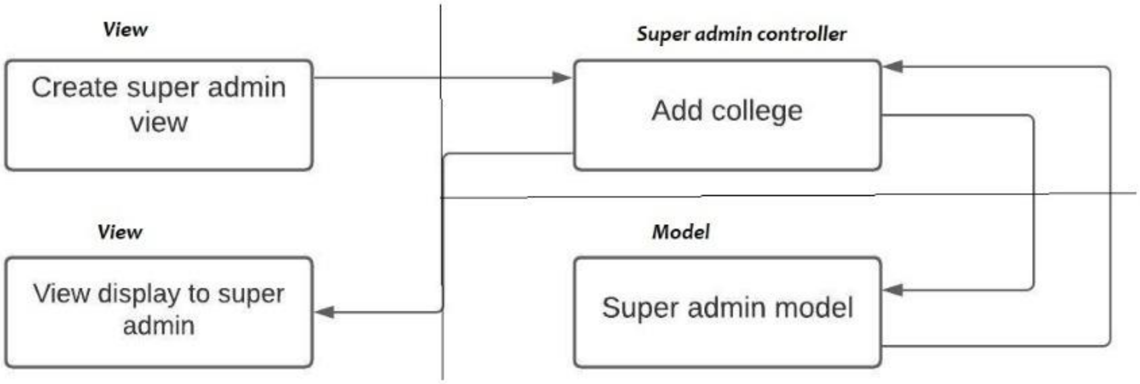

View
Create admin view
Admin controller
View fee
View
View display to admin
Model
Admin model

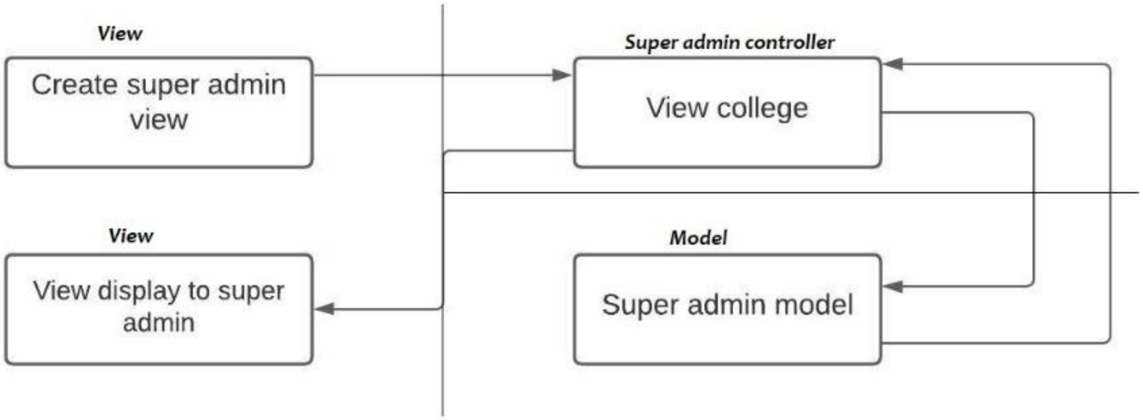

View
Create admin view
Admin controller
Add event
View
View event display to student ,teacher and librarian
Model
Admin model

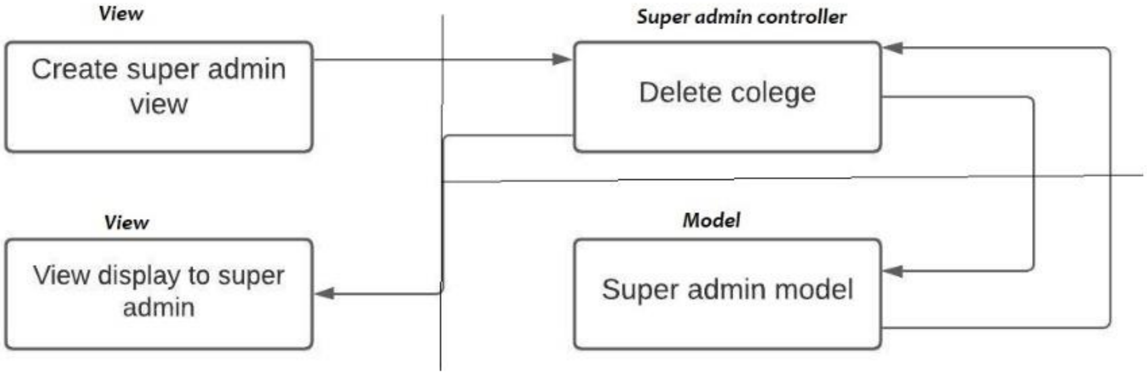

View
Create admin view
Admin controller
View book
View
View display to admin
Model
Admin model

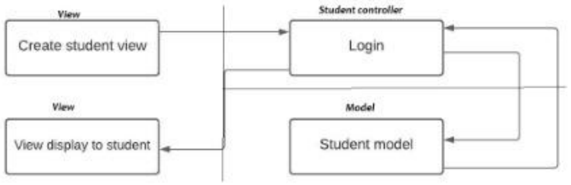

View
Create admin view
Admin controller
Delete books request
View
View display to admin & librarian
Model
Admin model

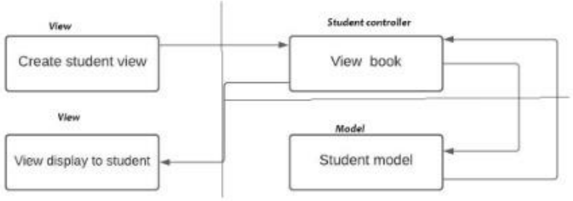

View
Create super admin view
View
View display to super admin
Super admin controller
Login
Model
Super admin model

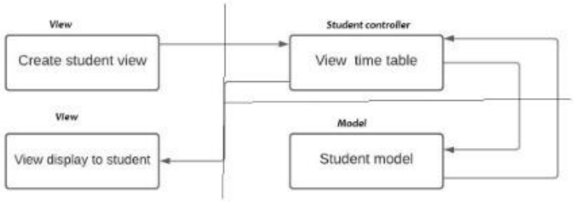

View
Create super admin view
View
View display to super admin
Super admin controller
Add college
Model
Super admin model

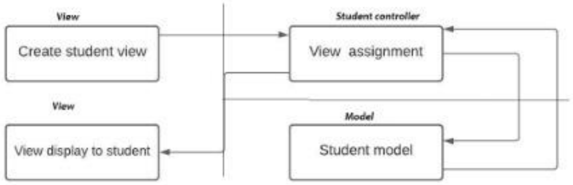

View
Create super admin view
View
View display to super admin
Super admin controller
View college
Model
Super admin model

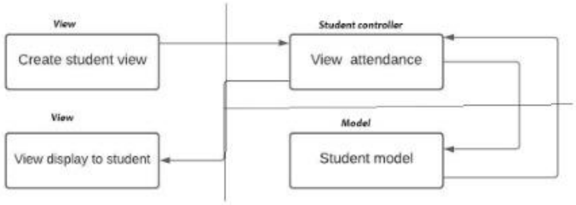

View
Create super admin view
View
View display to super admin
Super admin controller
Delete colege
Model
Super admin model

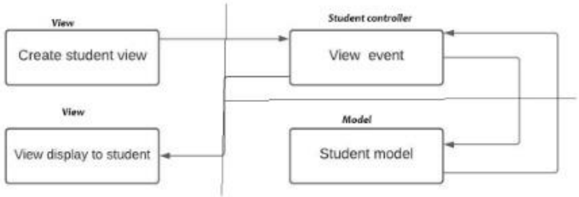

View
Create student view
View
View display to student
Student controller
Login
Model
Student model

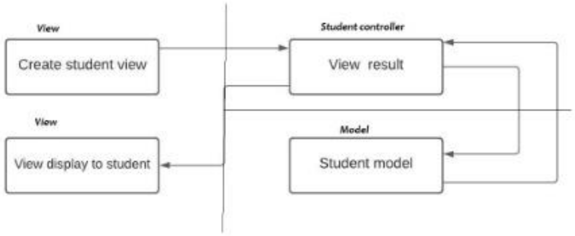

View
Create student view
View
View display to student
Student controller
View book
Model
Student model

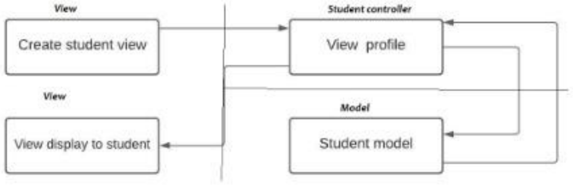

View
Create student view
View
View display to student
Student controller
View time table
Model
Student model


View
Create student view
View
View display to student
Student controller
View assignment
Model
Student model


View
Create student view
View
View display to student
Student controller
View attendance
Model
Student model


View
Create student view
View
View display to student
Student controller
View event
Model
Student model


View
Create student view
View
View display to student
Student controller
View result
Model
Student model


View
Create student view
View
View display to student
Student controller
View profile
Model
Student model