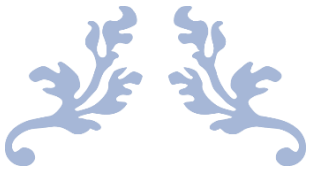

UNIVERSITY SYSTEM

Abdallah Ahmed Mohamed Awadallah
Mahmoud Fathy Sayed
Medhat Ashraf Alhaddad
Mohamed Sameh Abdelhakim
Nour E-Din Osama Mohamed
Youssef Khaled Hussein

Cairo, 2018

Table of Contents

Table of Figures

1. Introduction

This report shows the design and planning of a university system from point of view of program designer. The report includes the functional and non-functional requirements and many diagrams/models to help in understanding how the system works and other information which are all as shown in the table of contents.

2. Detailed description of functional and Non-Functional Requirements

1.1 Functional Requirements

- The system lets new students register.
- The system lets registered students enroll in courses, the number of courses depends on the GPA of the student, if a student has a GPA that is higher than 3.0 the student is offered six or seven courses depending on their choice, if the student has a GPA that is between 2.0 and 3.0, then the student is only allowed to enroll in six courses, if the student has a GPA that is lower than 2.0 then the student can only enroll in four courses.
- The system manages the schedules of the following:
  1. Courses and students such that no student has clashing course schedules
  2. Staff
  3. Classrooms such that no classroom is taken by two courses at the same time slot
  4. Exams.
- The system allows the teachers to put students' grades on the system, students have access to a section where they can see their grades

- The system allows the teachers to upload lectures to the system, students are able to download the files when they reach the lectures section.
- The system allows the teachers to upload assignments and projects and set their deadlines.
- The system allows students to add, drop and withdraw a course or more that they have previously enrolled in, with condition that the request made by the student has to be within a certain duration of the registration period.
- The system should also keep track of the available, borrowed, missing, and needed books from the university library.

1.2 Non-Functional Requirements

- The system's reliability (Mean Time to Failure) is more than 4 weeks.
- The system's availability (Mean Time to Recovery) is less than 1 hour.
- The system's response time is less than 0.5 seconds
- The servers used to host the website are Oracle – provided servers.
- The databases used are provided by Microsoft
- The peak-load-capacity is ¾ the number of registered students +1000

3. Stake Holders

- Students
- Professors

- Librarians
- Staff
- Admins
- University Dean
- Architect
- Designer
- Implementor
- Tester, integrator
- Manager
- Quality assurance people

4. Views and View Points

4.1 View

Chosen architecture view for this system is the module view.

4.1.1 Decomposition

Student Management: subsystem of university management.

Course Management: subsystem of university management.

Grade Management: subsystem of Course Management.

Submission Management: subsystem of Grade Management.

Library Management: subsystem of university management.

Schedule Management: subsystem of university management.

4.1.2 Uses

Schedule Management uses Student Management data to make schedules.

Submission Management uses Course Management data to manage uploaded solved and unsolved assignments and projects.

4.1.3 Layers

Submission Management uses course management to allow students' assignments and projects to be uploaded and graded.

Grades Management uses Course Management to assign grades to students according to their graded assignments.

University Management uses Course Management to show the students their grades.

4.1.4 Class

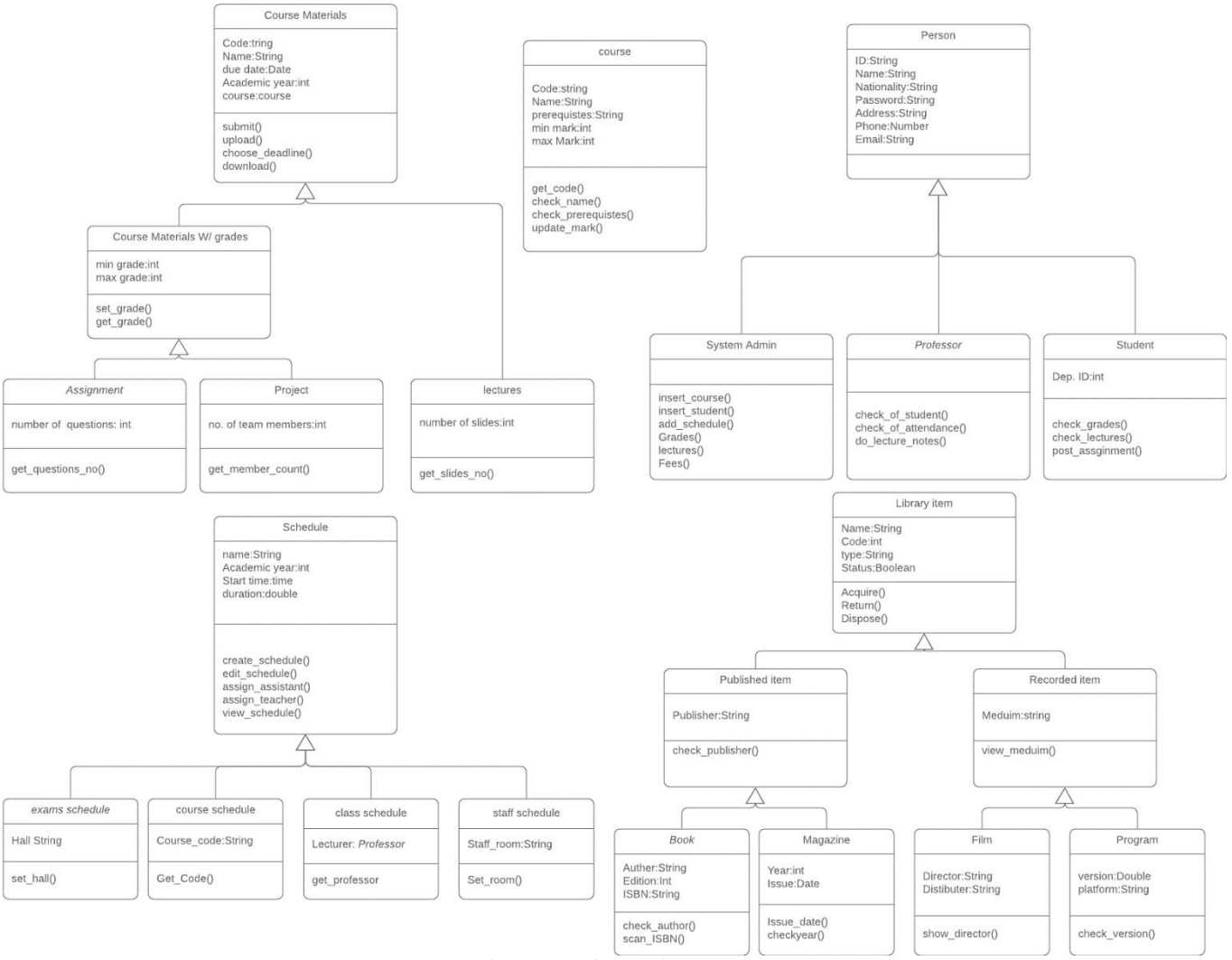

Figure 1: Class Diagram

4.2 View points

4.2.1 Logical View

- The system lets new students register.
- The system lets registered students enroll in courses.
- The system manages the schedules of the courses, staff, classrooms, exams.

- The system allows the teachers to put students' grades on the system.
- Students have access to a section where they can see their grades.
- The system allows the teachers to upload lectures to the system.
- Students are able to download the files when they reach the lectures section.
- The system allows the teachers to upload assignments and projects and set their deadlines.
- The system allows students to add, drop and withdraw a course or more that they have previously enrolled in.
- The system should also keep track of the available, borrowed, missing, and needed books from the university library.

The logical view of this system is the functional requirements provided for the project. The system shall provide these services for the staff and the students (end users) of the system. When providing the logical view we deal with the end user functionality and nothing related to the design of the system itself, as we are only showing a key abstraction of the system.

4.2.2 Process View

- The system's reliability (Mean Time to Failure) is more than 4 weeks.
- The system's availability (Mean Time to Recovery) is less than 1 hour.
- The system's response time is less than 0.5 seconds

- The peak-load-capacity is ¾ the number of registered students +1000
- The system lets students download the assignments that the teachers upload.
- The system keeps track of the available, borrowed, missing and needed books from the university library.

The process view shows some non-functional requirements such as the system's performance, concurrency, availability, fault tolerance and integrity, as well as some of the interactions between the processes and the tasks at runtime. The view allows us to see the what needs to happen inside the system to accomplish it's goals.

4.2.3 Implementation View

- The system consists of Courses, Library, Students & Staff Members

This view shows the software components that should be in the system and their interactions with the system. These modules are separated and are easily developed in small chunks by one or more developers. It is aimed for the programmers, as it makes the development process easier as the system is divided into little modules and mapped to the system's architecture rather than being one big complex module.

4.2.4 Deployment View

- The system's reliability is more than 4 weeks
- The system's availability is less than 1 hour
- The peak-load-capacity is ¾ the number of registered students +1000
- The system runs on a server provided by Oracle

- The databases of the university are provided by Microsoft

This view describes the environment into which the system will be deployed and is aimed for system engineers. This system will be deployed on servers provided by Oracle and the users shall connect to that server by their devices (Android, IOS, Windows). It takes into account the system's non-functional requirements such as system's availability, reliability and scalability.

4.2.5 Scenario View

- The system lets the user fill registration and then this registration form and either accepts it or prints an error message for the missing or wrong info.
- The students can log into the LMS and their information is checked whether or not this account is on the system and valid or not.
- The system allows the student to access a section in the website with the student's term work marks.
- The system allows the student to manage his courses where he can view, add or drop the courses after the registration which is always checked for availability according to the deadlines set on them.
- The system allows the user to view the library books and send borrow requests which is then verified by the library staff member and then is added to the student's borrow list and removed from the library.
- The system allows the teaching staff to upload the course materials, assignments, projects & grades to the system where the student can then view it from the course management section on website.

This view is the driver that helps designers discover architectural elements during the architectural design, and validate the architecture design on paper & as the starting point for the tests of an architectural prototype. This view illustrates the important use case scenarios, which shows how the other 4 views work together in the system. The system allows the students, teaching staff, library staff and administration staff

members to access certain functionalities in the system where the interactions between the actors happen on the system & discover the needed components for such a system.

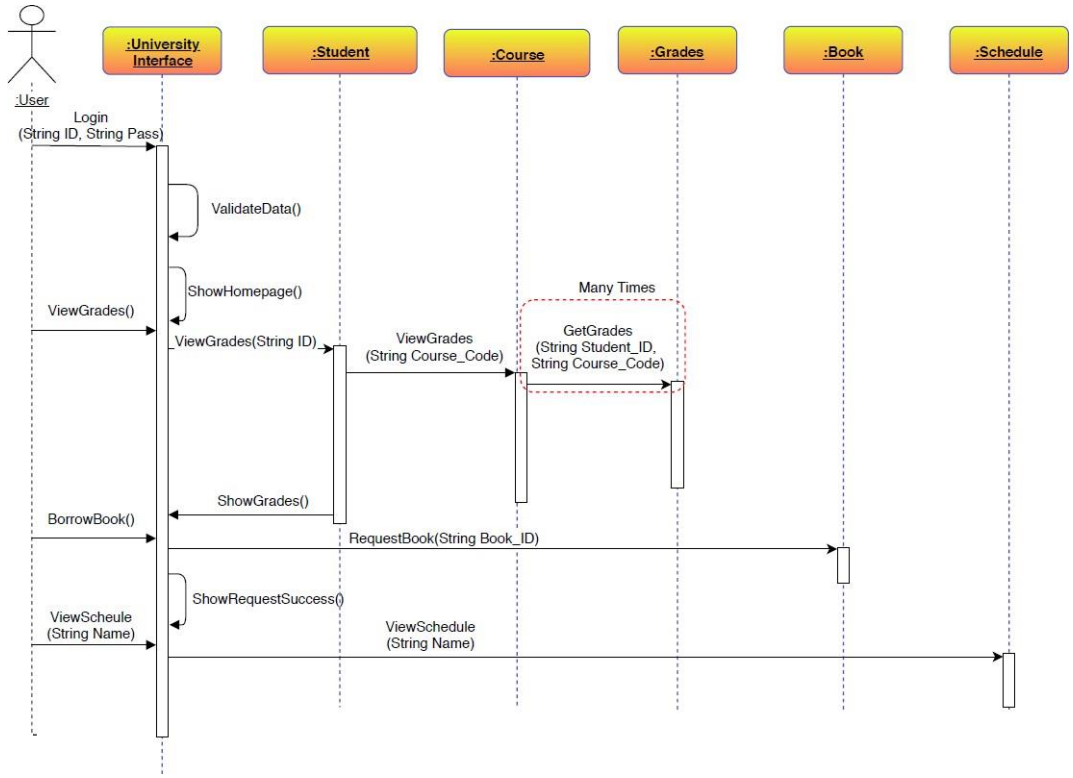

Figure 2: Sequence Diagram

5. Architecture Design

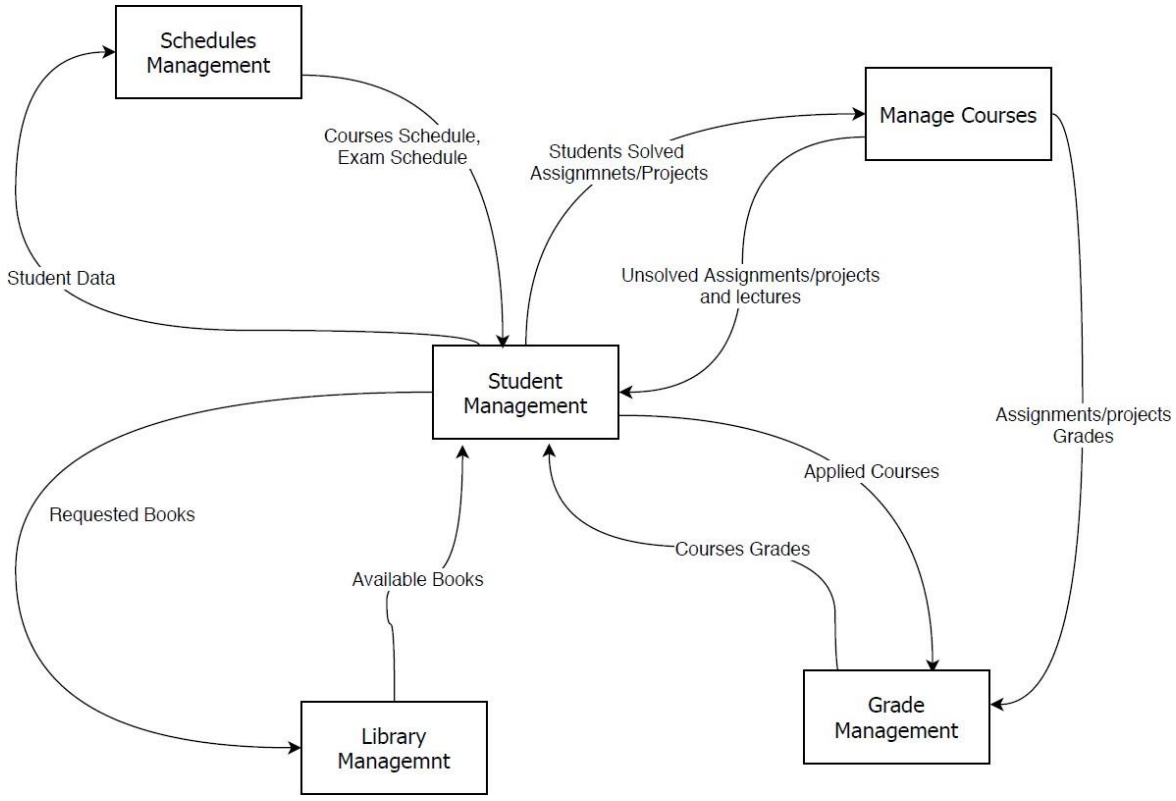

Figure 3: Architecture Model

6. Architecture Style

Chosen architecture style is **abstract data type style**.

Problem: identify and protect related bodies of information as students' data, professors' data, and staff's data, as no student should be allowed to view data of other students, also information of each course shouldn't be open to other courses. Data representations are likely to change.

Context: object-oriented methods which guide the design, object-oriented languages which provide the class-concept

Solution:

- system model: Each component has its own local data, they are hidden from other components and can only be accessed using connectors that are given to specific components only.
- components: managers (servers, objects)
- connectors: procedure call (message)

- control structure: single thread; control is decentralized

Variants: java, C++, C#, python and php.

Chosen language is php as it supports OOP and our system is a website so php is the most suitable language for this system, it is also free to use, and many browsers supports it as it is popular.

7. Decisions

Problem	Decision	Reasons	Alternatives
Which Programming Language to use for back-end?	php.	• php supports OOP which is the main principle used to code the website's functionality. • Free. • Supported by most browsers due to its popularity.	Python Java C++ C#
Which programming language to use for front-end?	Html, CSS and JavaScript.	JavaScript is supported by more browsers in comparison with WebAssembly.	Html, CSS and WebAssembly.
Which server to host on?	Oracle-provided servers.	A deal between Oracle and the university.	Amazon servers.
What type of database to use?	Microsoft provided servers.	A deal between Microsoft and the university.	MongoDB. SQLite.

Table 1: Decisions

8. Components

I- composition component: login

specification: manage the login system

Component: login	
Requires interface:	login(string name, password pn) Doc: given name and password; validate then create session and login user
Provides interface:	Natural:permission_level(String name) Doc: given user name; check the permission level(student, teacher, staff)
Component Characteristics	
Meta-data:	username, hash password and permission level
black/gray:	Black, no need to manipulate data or methods.
Developed/reused:	Reused,Genreic and abstract login system with password and id
documentation:	component used to manage the login system
	number of interfaces (requires/provides) 1/1
Types of composition:	Hierarchical composition

II- composition component: Student

specification: manage student register, enroll, add, drop and withdraw

Component: register	
Requires interface:	

Student_registration(String cn)

Doc: given course name; register student to the this course after checking conditions

Provides interface:

Natural:number_of_courses()

Doc: return the number of courses this student registered

Boolean:Is_registered()

Doc: return true if this student is registered

Component Characteristics

Meta-data:	Student name, student id, course pointer
black/gray:	grey,we need to redevelop the output to communicate with other component
Developed/reused:	Reused with modification;Genreic, abstract of student in university; but modification needed
documentation:	manage student register
	number of interfaces (requires/provides) 1 / 2
Types of composition:	Additive composition

Component: enroll in course

Requires interface:

Student_enroll(String cn)

Doc: given the course name; enroll student to this course

Provides interface:

Boolean:enroll_statue(string cn)

Doc: given course name; return enroll statue

Component Characteristics

Meta-data:	Student name, student id, course pointer

black/gray:	grey,we need to redevelop the output to communicate with other component
Developed/reused:	Reused with modification; Generic, abstract of student in university; but modification needed.
documentation:	Manage student enroll.
	number of interfaces (requires/provides) 1 / 1
Types of composition:	Additive composition

Component: add drop withdraw	
Requires **interface:**	add_course(string cn) Doc:given course name; check then add this course to student drop_course(string cn) Doc:given course name; check then drop this course to student withdraw_course(string cn) Doc:given course name; check then withdraw this course to student
Provides **interface:**	String[]:courses_added() Doc:return all the courses added by student string[]:courses_dropped() Doc:return all the courses dropped by student string[]:courses_withdraw() Doc:return all the courses withdrawn by student
Component Characteristics	
Meta-data:	Student name, student id, course pointer

black/gray:	grey,we need to redevelop the output to communicate with other component
Developed/reused:	Reused with modification; Generic, abstract of student in university; but modification needed.
documentation:	Manage student add, drop and withdraw.
	number of interfaces (requires/provides) 3 / 3
Types of composition:	Additive composition

III- composition component: scheduling

specification: manage the (courses, class rooms, exams and staff) scheduling

Component: course scheduling	
Requires interface:	create_schedule(string[] cna, stamps tps) Doc:given courses and times; check clashes and create schedule edit_schedule(string cn, String cn, stamp tp) Doc:given old course, new course and time; check clashes and edit schedule assign_assistant(String name,alpnum ID, String cn) Doc:given assistant name and id; assign him to given course name assign_teacher(String name,alpnum ID) Doc:given teacher name and id; assign him to given course name
Provides interface:	Image:view_schedule() return Image of courses schedule
Component Characteristics	
Meta-data:	courses pointer array, date and time stamp, teacher name, assistant name
black/gray:	Black, no need to manipulate data or methods
Developed/reused:	Developed, Generic use, abstract of Scheduling in university,
documentation:	manage the courses scheduling
	number of interfaces (requires/provides) 4 / 1

Types of composition:	Additive composition

Component: class rooms scheduling

Requires

interface:

create_schedule(string[] crn, stamps tps)

Doc:given classrooms and times; check clashes and create schedule

edit_schedule(string crn, String nwcrn, stamp tp)

Doc:given old class room, new class room and time; check clashes and edit schedule

Provides

interface:

Image:view_schedule()

Doc:return Image of class room schedule

Component Characteristics

Meta-data:	courses pointer array, date and time stamp, teacher name, assistant name
black/gray:	Black, no need to manipulate data or methods
Developed/reused:	Developed, Generic use, abstract of Scheduling in university,
documentation:	manage the class rooms scheduling
	number of interfaces (requires/provides) 2 / 1
Types of composition:	Additive composition

Component: staff working scheduling

Requires

interface:

create_schedule(string[] srn, stamps tps)

Doc:given staff and times; check clashes and create schedule

edit_schedule(string srn, String nwsrn, stamp tp)

Doc:given old staff, new staff and time; check clashes and edit schedule

Provides interface:

Image:view_schedule()

Doc:return Image of staff schedule

Component Characteristics

Meta-data:	courses pointer array, date and time stamp, teacher name, assistant name
black/gray:	Black, no need to manipulate data or methods
Developed/reused:	Developed, Generic use, abstract of Scheduling in university,
documentation:	Manage the staff scheduling.
	number of interfaces (requires/provides) 2 / 1
Types of composition:	Additive composition

Component: exams working scheduling

Requires interface:

create_schedule(string[] exms, stamps tps)

Doc:given exams and times; check clashes and create schedule

assign_supervisor(String name,alphanum ID)

Doc:given teacher name and id; assign him to given exam name

assign_assistant(String name,alphanum ID)

Doc:given assistant name and id; assign him to given exam name

edit_schedule(string exm, String nwexm, stamp tp)

	Doc:given old course, new course and time; check clashes and edit schedule **Provides interface:** Image:view_schedule() return Image of exams schedule
Component Characteristics	
Meta-data:	courses pointer array, date and time stamp, teacher name, assistant name
black/gray:	Black, no need to manipulate data or methods
Developed/reused:	Developed, Generic use, abstract of Scheduling in university,
documentation:	Manage the staff scheduling.
	number of interfaces (requires/provides) 4 / 1
Types of composition:	Additive composition

IV- composition component: course

Specification: manage course (grades, assignments, projects, materials and att.)

Component: grades
Requires interface: add_grade(String grade; ptr sid) Doc:given grade and student id; assign grade to student **Provides interface:** String:retrieve_grade(ptr sid) Doc:given student id; return his course grade Image:show_grades() Doc:return all students grades for the course
Component Characteristics

Meta-data:	course name, course id, grade, date and time stamp (assignment, project, materials) files
black/gray:	Grey, we need to redevelop the output to communicate with other component
Developed/reused:	Reused, abstract of upload and download system; but modification needed
documentation:	Manage course grades.
	number of interfaces (requires/provides) 1 / 2
Types of composition:	Additive composition

Component: assignments

Requires

interface:

submit_assignment(ptr File)

Doc:given assignment file; submit student assignment

upload_assignment(ptr File)

Doc:given assignment file; upload assignment

choose_deadline(Stamp dt)

Doc:given date; assign deadline

Provides

interface:

File: download_assignment()

Doc:return assignment file

Component Characteristics

Meta-data:	course name, course id, grade, date and time stamp (assignment, project, materials) files
black/gray:	Grey, we need to redevelop the output to communicate with other component
Developed/reused:	Reused, abstract of upload and download system; but modification needed
documentation:	manage course assignments
	number of interfaces (requires/provides) 3 / 1

Types of composition:	Additive composition

Component: projects	
Requires **interface:**	submit_Project(ptr File) Doc:given project file; submit student project upload_Project(ptr File) Doc:given project file; upload project choose_deadline(Stamp dt) Doc:given date; assign deadline
Provides **interface:**	File: download_assignment() Doc:return project file
Component Characteristics	
Meta-data:	course name, course id, grade, date and time stamp (assignment, project, materials) files
black/gray:	Grey, we need to redevelop the output to communicate with other component
Developed/reused:	Reused, abstract of upload and download system; but modification needed
documentation:	manage course projects
	number of interfaces (requires/provides) 3 / 1
Types of composition:	Additive composition

Component: materials

Requires
interface:

add_course_materials(ptr File)

Doc:given file; add this material to the course

Provides
interface:

Files :review_course_material()

Doc:return the course materials files

Component Characteristics

Meta-data:	course name, course id, grade, date and time stamp (assignment, project, materials) files
black/gray:	Grey, we need to redevelop the output to communicate with other component
Developed/reused:	Reused, abstract of upload and download system; but modification needed
documentation:	manage course projects
	number of interfaces (requires/provides) 1 / 1
Types of composition:	Additive composition

V- composition component: library

specification: Manage library system

Component: library

Requires
interface:

Add_book(order pid)

Doc:given order id; add new book to DB

return_book(String bok)

Doc:given book name; remove borrow statue from book

Provides **interface:**	List:review_books() Doc: return list of available books review_book(string bok) Doc: given book name; return book information request_borrowing(String bok) Doc: given book name; add borrow statue for book Boolean:validate_borrow_limit() Doc: check if number of borrowed books is acceptable Boolean:validate_return_state() Doc: check if all borrowed books is returned
Component Characteristics	
Meta-data:	books name, books information, order id borrow statue, borrow_limit, return_statue
black/gray:	Black, no need to manipulate data or methods
Developed/reused:	Reused, Generic use, abstract of library system
documentation:	Manage library system
	number of interfaces (requires/provides) 1 / 1
Types of composition:	Hierarchical composition

Table 2: Components Table

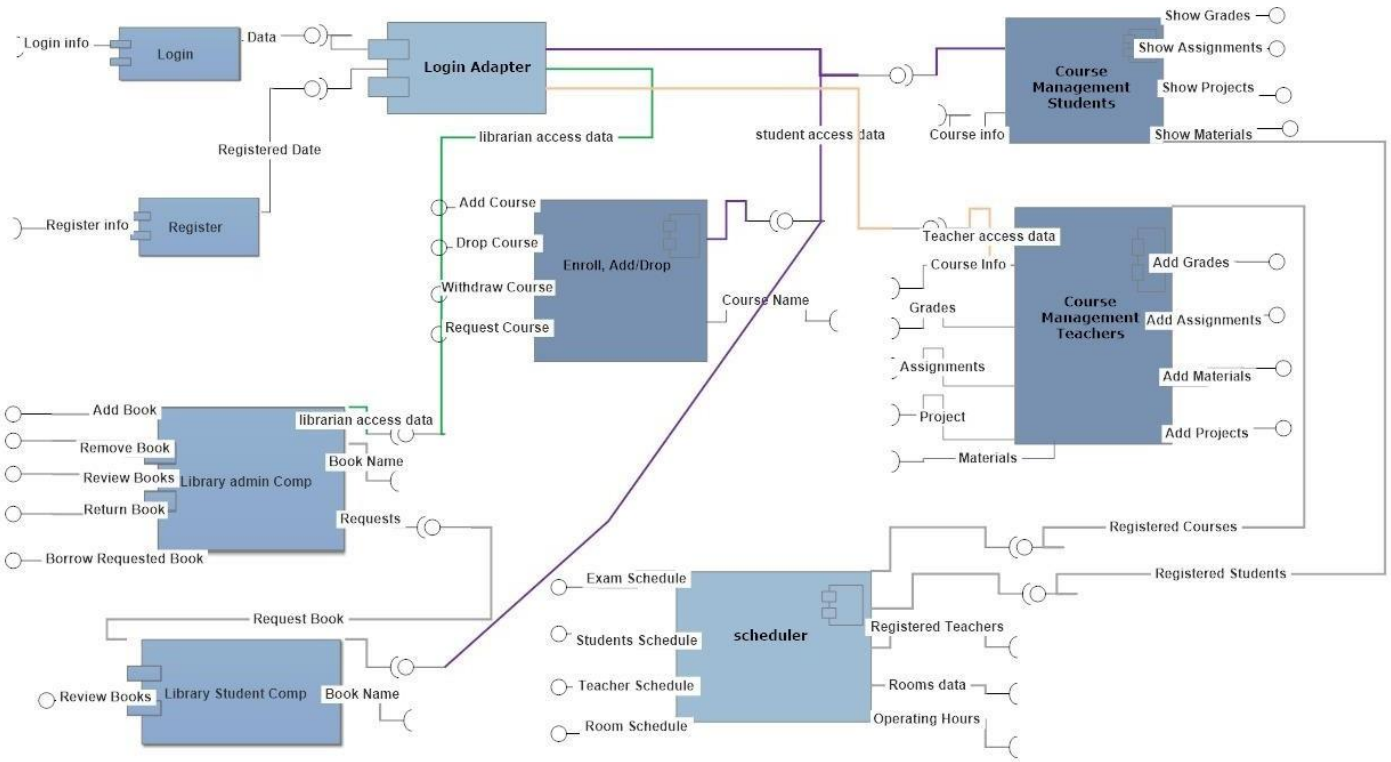

Figure 4: Component Diagram

9. Data Flow Diagram

9.1 Context Diagram

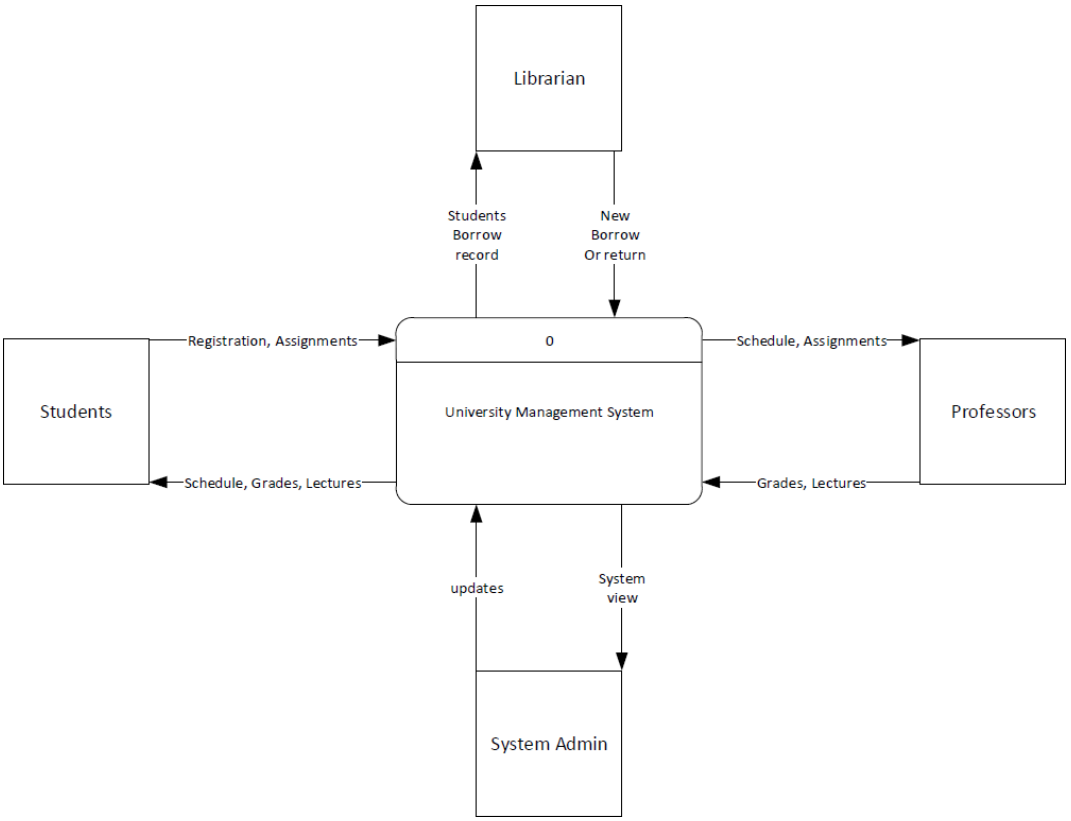

Figure 5: DFD Context Diagram

9.2 Level 0

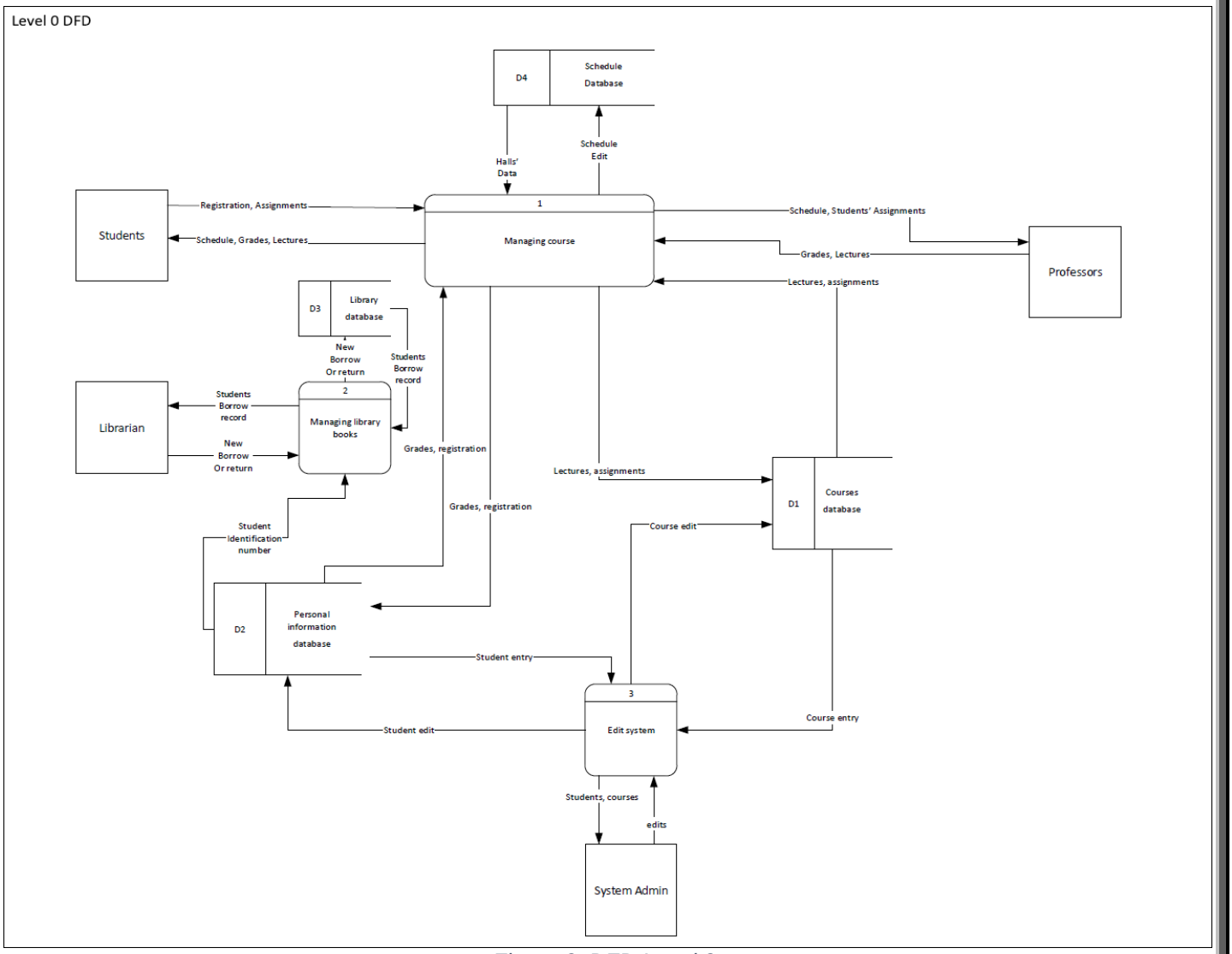

Figure 6: DFD Level 0

9.3 Level 1

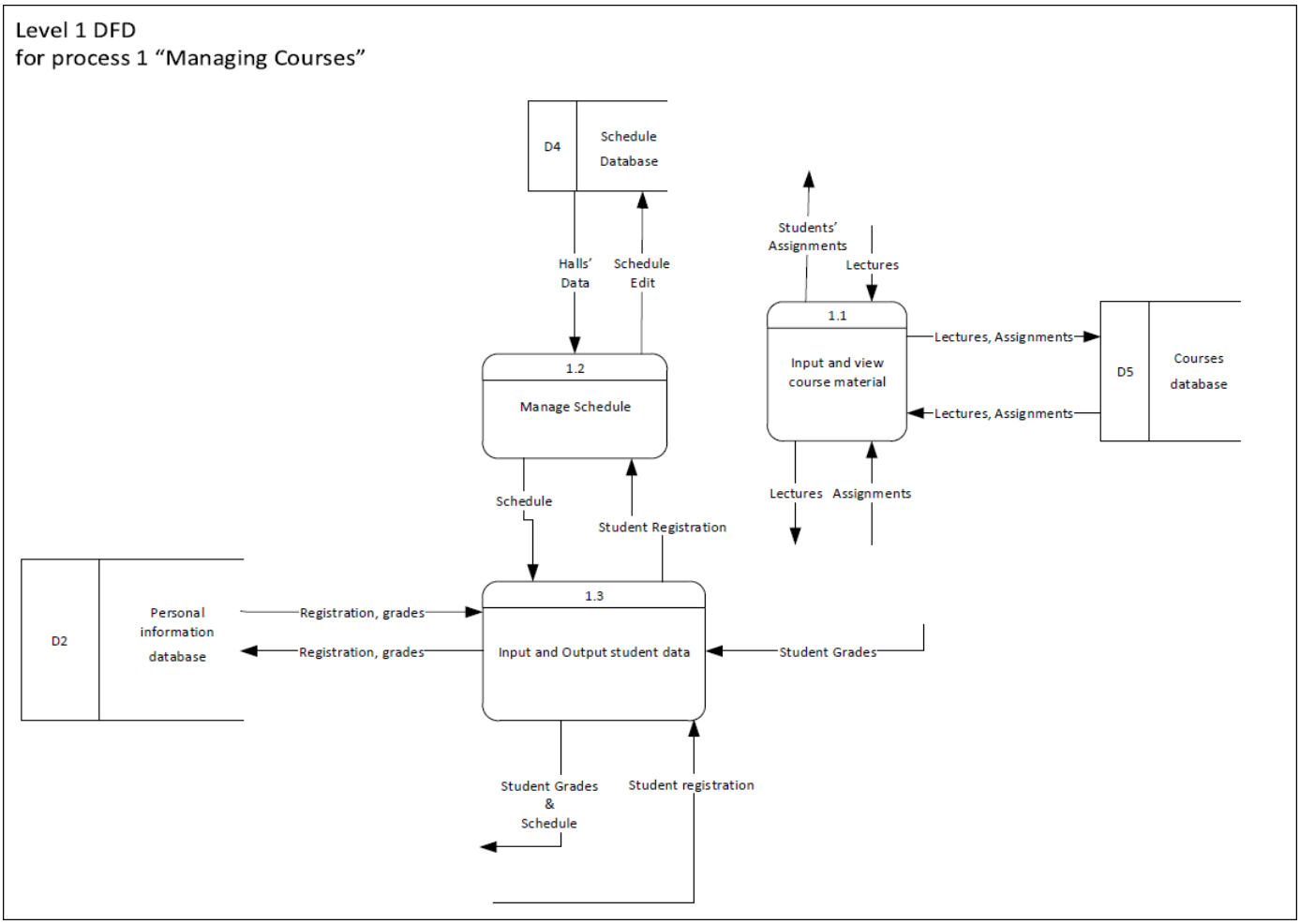

Figure 7: DFD Level 1

9.4 Level 2

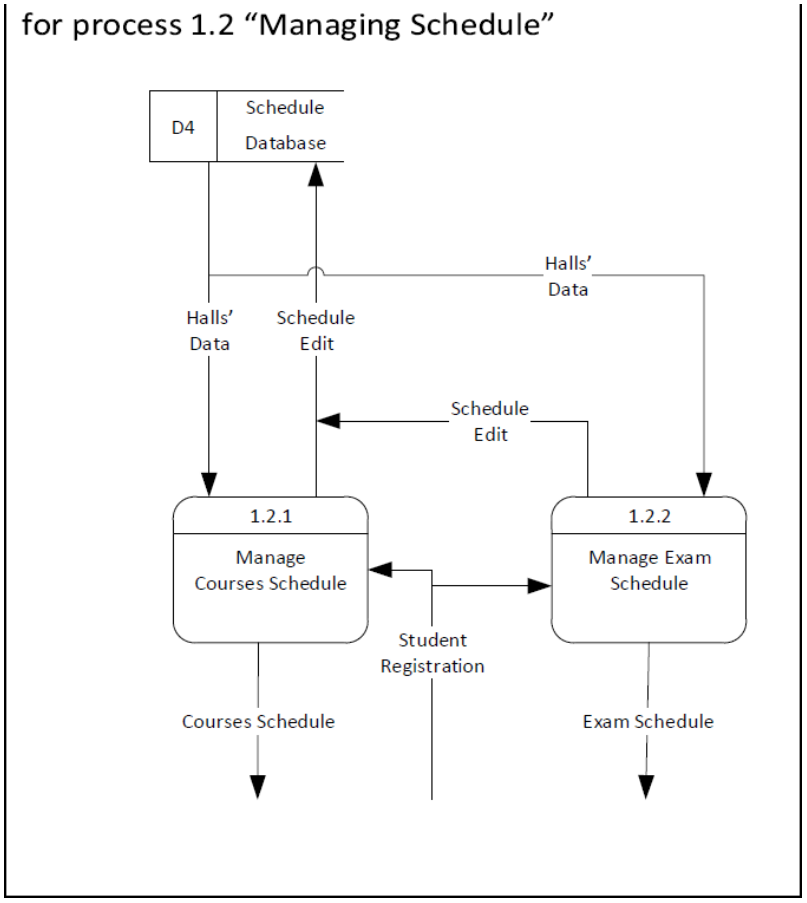

Figure 8: DFD Level 2

10. Models

10.1 Context Model

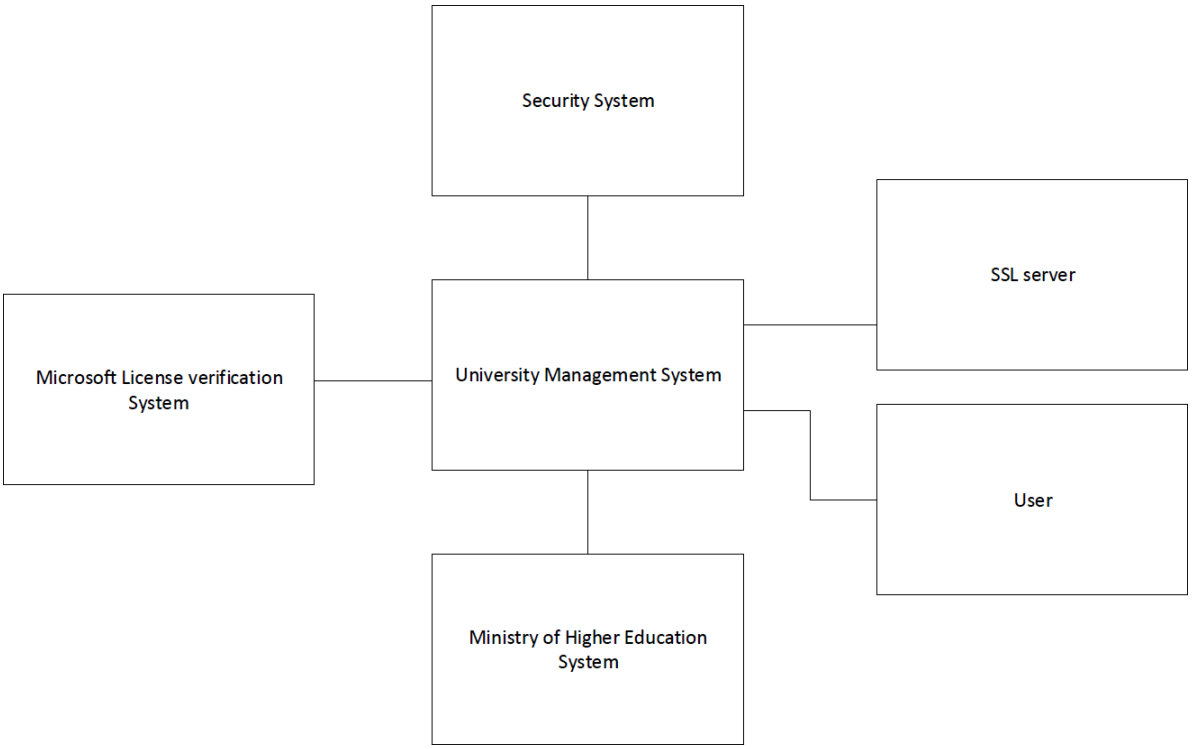

Figure 9: Context Model

10.2 Behaviour Model

10.2.1 State Diagram

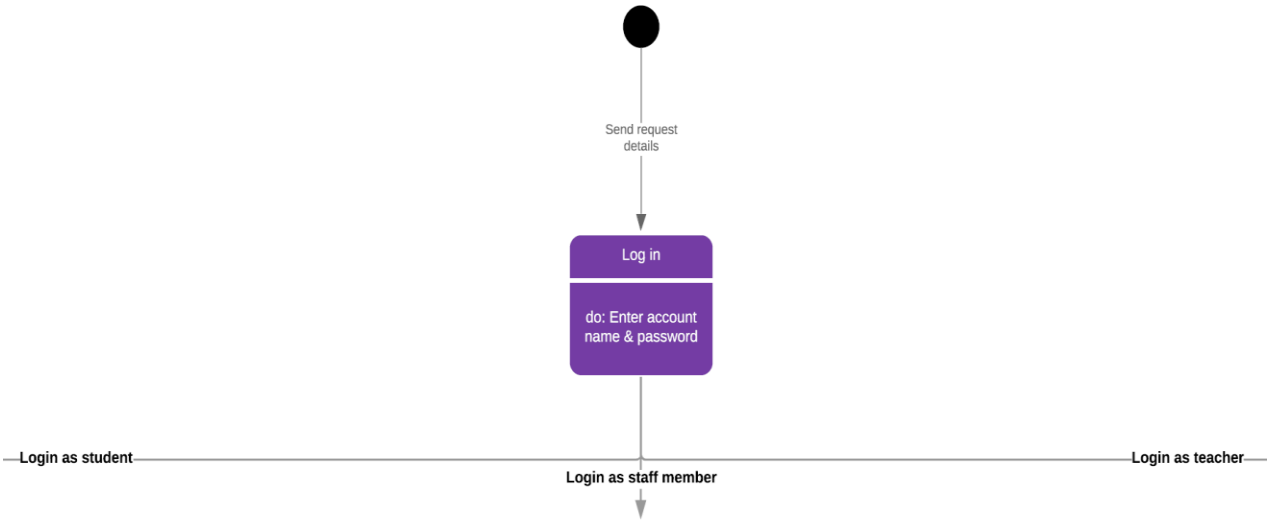

Figure 10: State Diagram (Start)

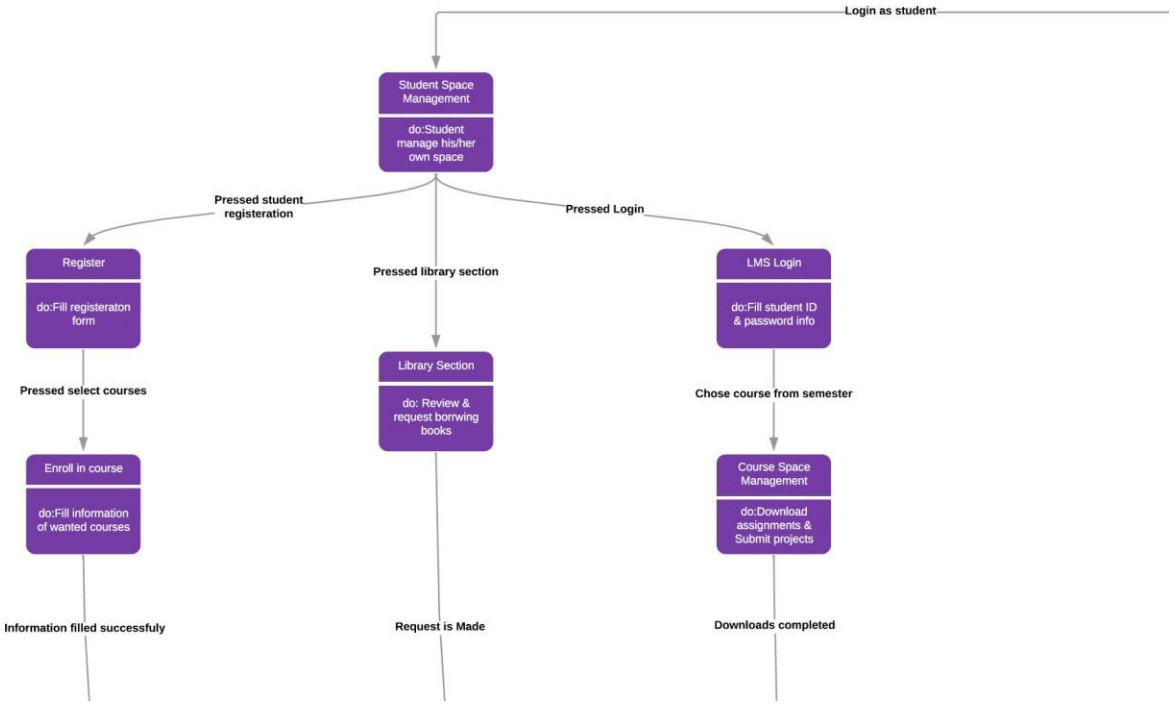

Figure 11: State Diagram (Login as Student)

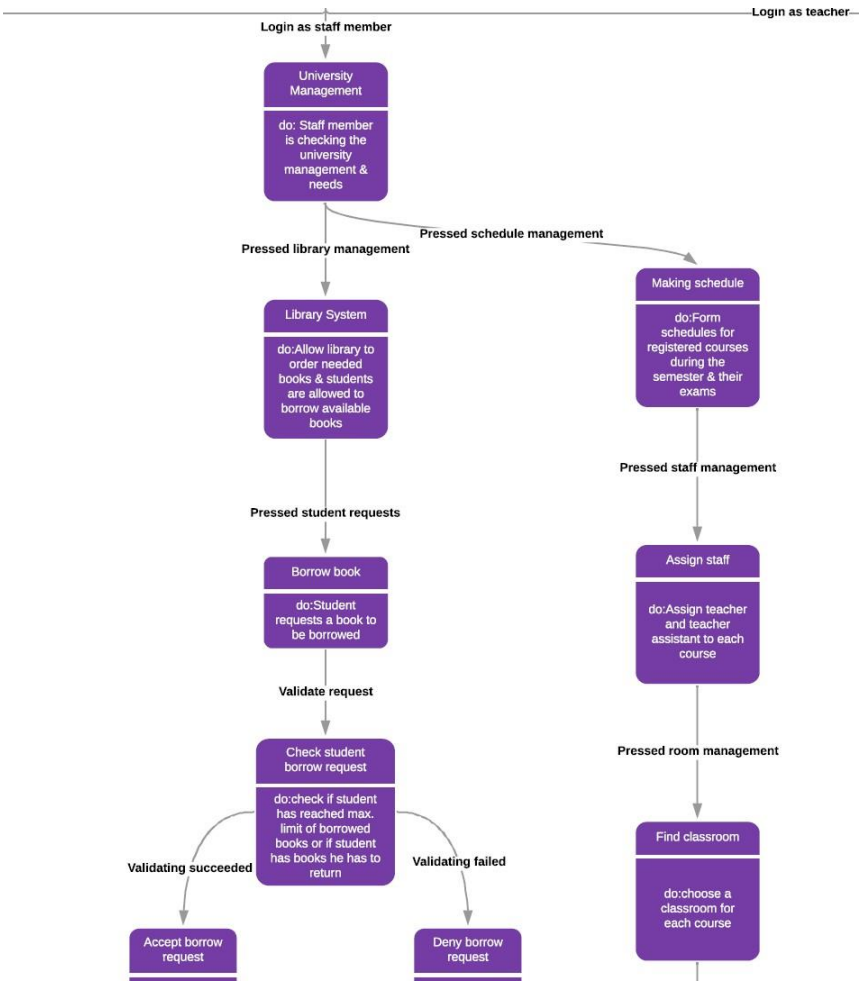

Figure 12: State Diagram (Login as Staff) 1

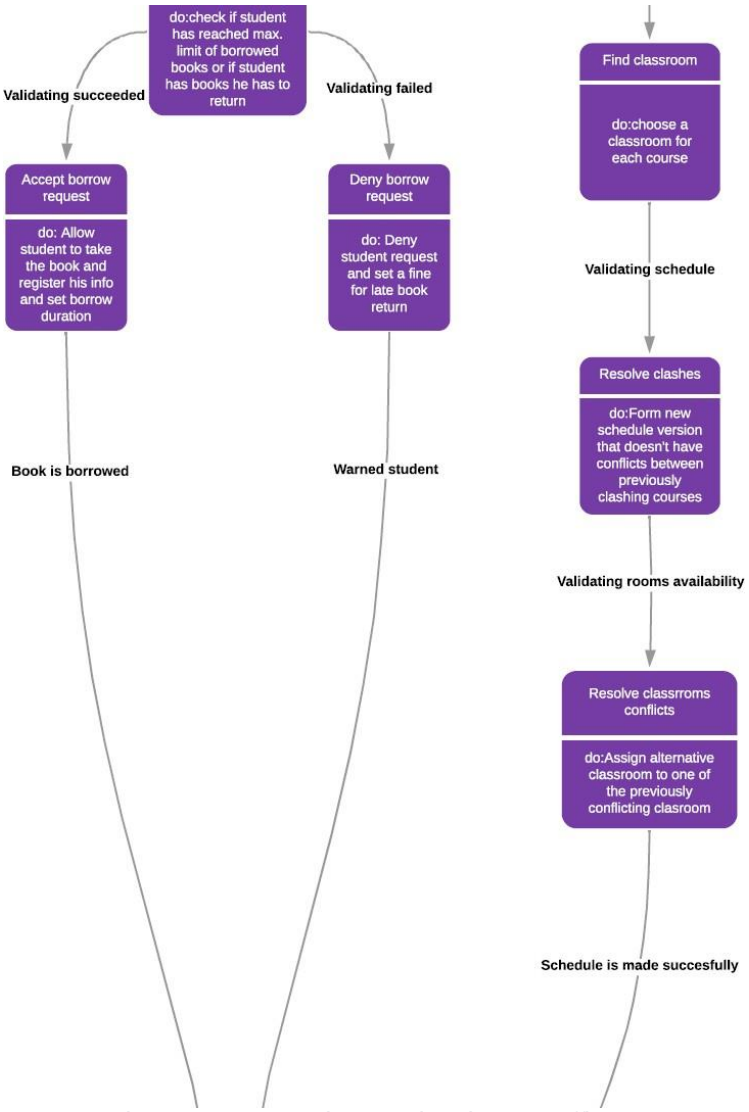

Figure 13: State Diagram (Login as Staff) 2

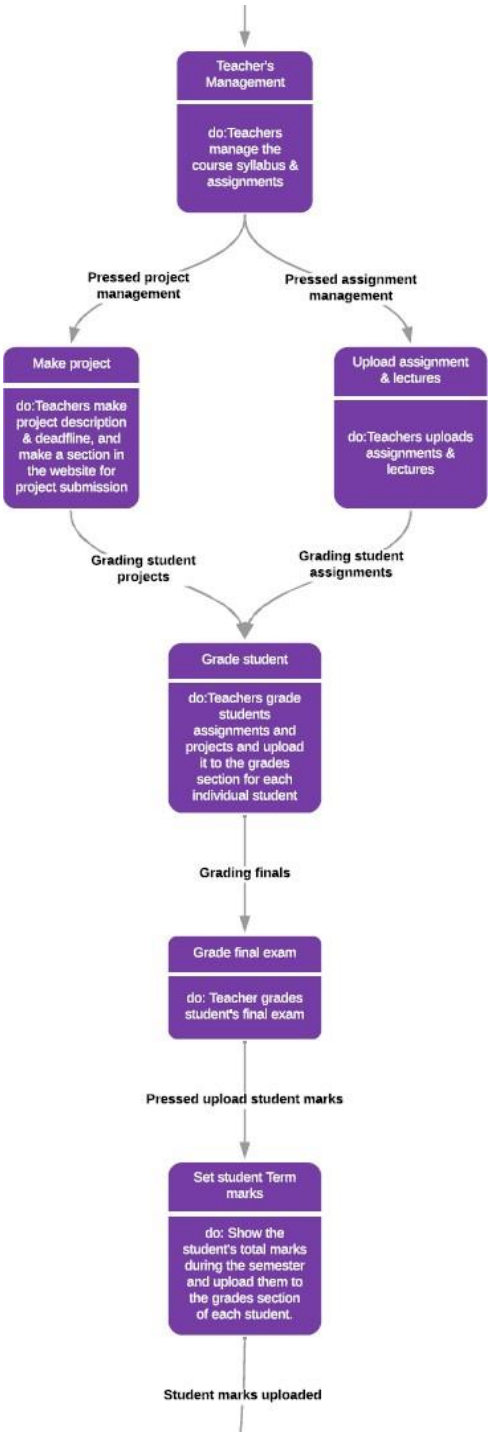

Figure 14: State Diagram (Login as Teacher)

10.2.2 State Diagram Table

State	Description
Log in	The user logs in to the system with his account
Student Space Management	Student Is managing his files and assignments
Register	Students are filling their information for registering
Enroll In Course	Students are choosing the courses they want to take this semester

LMS Login	Student is logging in the LMS to obtain the courses files
Course Space Management	Student is downloading assignments and uploads projects for grading
Library Section	Student views the books online and sends a request to borrow books
University Management	Staff members managing the university files and website sections
Making Schedule	Staff members filling the system with info to create schedules
Assign Staff	Staff members are assigning the teaching staff to the offered courses
Find Classroom	Staff members are assigning classrooms for the lectures and sections
Resolve Clashes	Staff members making new version of schedules that has no conflicts
Resolve Classroom Conflicts	Staff members are re-assigning classrooms that are occupied by another lecture/section at the same time
Library Systems	Library staff allows students requests for borrowing books and check if the library needs any new books
Borrow Book	Library staff manages students requests for borrowing books
Check Student Borrow request	Library staff checks the availability of the requested book, the student information and any outstanding books that the student has to return
Accept Borrow Request	Library staff accepts the borrow request and adds it to the account
Deny Borrow Request	Library staff rejects the borrow request and sets fine for the outstanding books that hasn't returned yet
Teacher's Management	Teaching staff manages their profiles and personal space on website
Upload assignments & lectures	Teaching staff uploads the assignments and the lectures for the specified courses and manages the course folders on the website
Make Projects	Teaching staff set project description & deadline on the website
Grade Student	Teaching staff sets the students marks for assignments & projects
Grade final exam	Teaching staff sets the students marks for their final exams
Set Student Term Marks	Teaching staff sets the overall grade of the students for this semester

Table 3: State Diagram State Description

Stimulus	Description
Login as student	The user has entered a student ID and password
Pressed Login	The student has pressed the LMS login button
Pressed library section	The student has pressed the library section button
Pressed student registration	The student has pressed the registration button
Chose course from semester	The student has pressed the course link
Pressed select courses	The student has pressed the select courses button
Downloads completed	The student downloads were completed
Request is made	The student request for borrowing book made successfully
Information filled successfully	The student filled all needed info for registration successfully
Login as staff member	The user has entered a staff member ID and password
Pressed schedule management	The staff member has pressed schedule management button
Pressed library management	The staff member has pressed the library management button
Pressed staff management	The staff member has pressed the staff management button
Pressed room management	The staff member has pressed the room management button
Validating schedule	The schedule are being validated for no more conflicts
Validating rooms availability	The rooms are being validated for no conflicting occupancies
Schedule is made successfully	The schedule & rooms has been verified successfully
Pressed student request	The library staff has pressed student requests button
Validate request	The borrow request are being validated
Validating succeeded	The borrow request has been validated and accepted
Validating failed	The borrow request has been denied
Book is borrowed	The book has been added to the student account
Warned student	The student has been warned by a fee for late books
Login as teacher	The user has entered a teacher ID and password
Pressed assignment management	The teaching staff has pressed assignment management button
Pressed project management	The teaching staff has pressed project management button
Grading student assignment	The students assignments are sent to be graded
Grading student projects	The students projects are sent to be graded
Grading finals	The students final exams are sent to be graded
Pressed upload student marks	The teaching staff has pressed the upload student marks button

Student marks uploaded	The teaching staff has finished uploading the students marks

Table 4: State Diagram Stimuli

10.3 Semantic Model

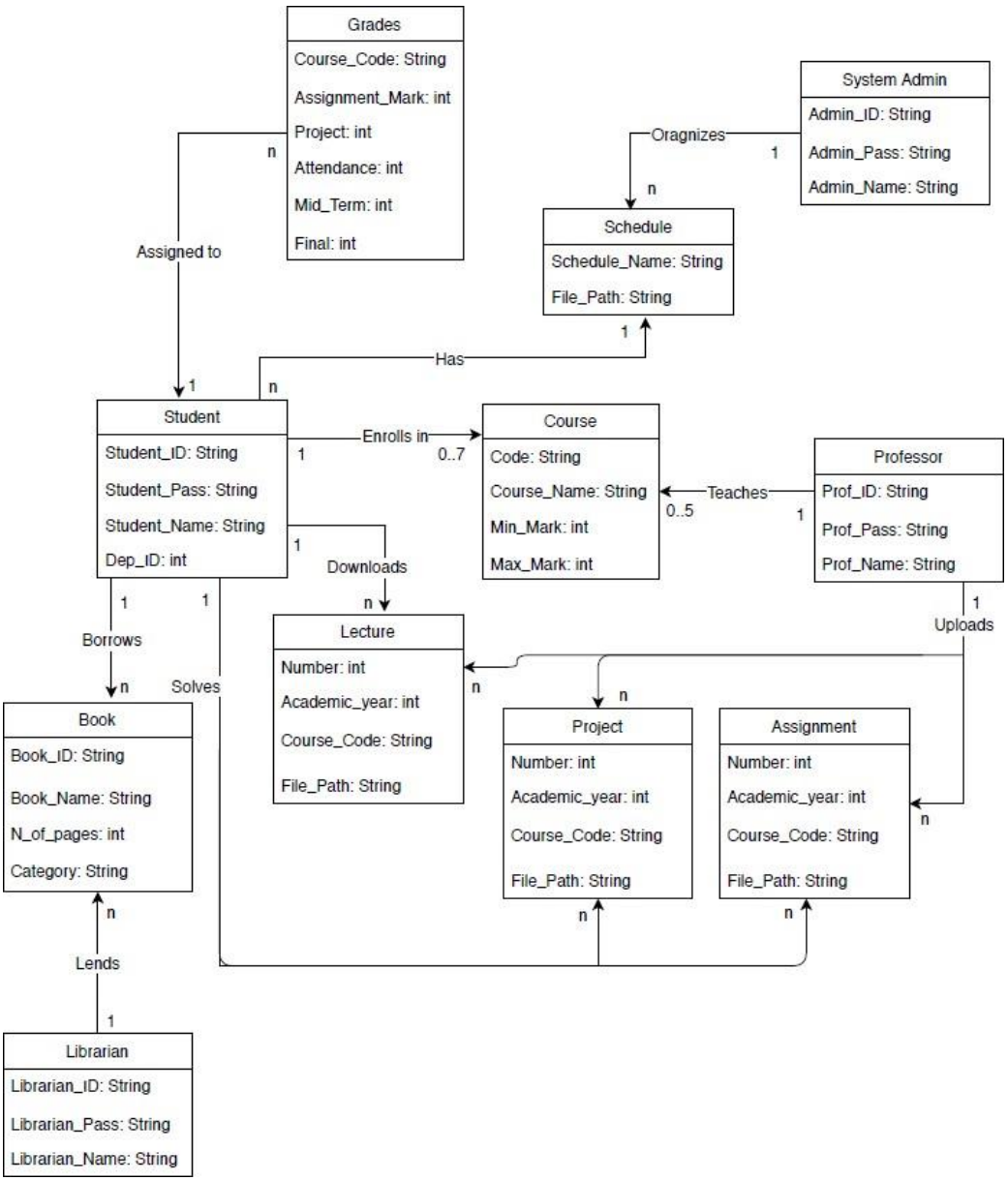

Figure 15: Semantic Model

10.3.1 Data Dictionary

Name	Description	Type	Date
Student	Details of student who enrolled in this university	Entity	23.12.2018
Student_ID	ID of the student, which is unique for person student in the university	Attribute	24.12.2018
Student_Pass	Password of student which is used to login in to the system	Attribute	24.12.2018
Student_Name	Name of the student	Attribute	24.12.2018
Dep_ID	The ID of the department that the student is in	Attribute	24.12.2018
Enrolls in	A 1:many relationship between Student and Course, as students enrolls in courses	Relation	24.12.2018
Downloads	A 1:many relationship between Student and Lecture, as students download lectures to study them.	Relation	24.12.2018
Borrows	A 1:many relationship between Student and Book, as students borrow books from library.	Relation	24.12.2018
Solves	A 1:many relationship between Student, Project and Assignment, as students solve assignments and projects that are to be submitted.	Relation	24.12.2018
Has	A many:1 relationship between Student and Schedule, as all students have 1 schedule for the courses, midterms and finals	Relation	24.12.2018
Course	Details of the courses available in the university	Entity	24.12.2018
Code	Code of a course, which is unique for every course	Attribute	24.12.2018
Course_Name	The name of the course	Attribute	24.12.2018
Min_Mark	Minimum mark that a student can get in the course or he will fail it	Attribute	24.12.2018
Max_Mark	The maximum mark a student can get in a course	Attribute	24.12.2018
Teaches	A 1:many relationship between Professor and Course, as many courses can be taught by 1 professor	Relation	24.12.2018
Professor	The details of a professor which works in the university	Entity	24.12.2018
Prof_ID	ID of the professor, which is unique for every person in the university	Attribute	24.12.2018
Prof_Name	Name of the professor	Attribute	24.12.2018
Prof_Pass	Password of professor which is used to login in to the system	Attribute	24.12.2018

Uploads	A 1:many relationship between Professor, Project, Assignment and Lecture, as professors upload all that's needed by the student on the system for students to study them and solve problems on them	Relation	24.12.2018
Assignment	Details of the assignments uploaded for students	Entity	24.12.2018
Project	Details of the projects uploaded for students	Entity	24.12.2018
Lecture	Details of the lectures uploaded for students	Entity	24.12.2018
Number	It shows the number of the uploaded assignment, project or lecture	Attribute	24.12.2018
Academic_year	It shows which academic year this assignment, project or lecture belongs to	Attribute	24.12.2018
Course_Code	It shows the course that the uploaded assignment, project or lecture belongs to	Attribute	22.12.2018
Grades	Details of all the grades that are assigned to students	Entity	22.12.2018
Assigned to	A many:1 relationship between Grades and Student, as students get their grades of all courses assigned to them.	Relation	24.12.2018
Assignment_Mark	The grades of the assignments	Attribute	22.12.2018
Project	The grades of the project	Attribute	22.12.2018
Attendance	The grades of the attendance	Attribute	22.12.2018
Mid_Term	The grades of the midterm exam	Attribute	22.12.2018
Final	The grades of the final exam	Attribute	22.12.2018
System Admin	Details of the system admin (Staff) that can register students, edit their schedules	Entity	22.12.2018
Admin_ID	ID of the admin, which is unique for every person in the university	Attribute	22.12.2018
Admin_Pass	Password of admin which is used to login in to the system	Attribute	23.12.2018
Admin_Name	Name of the admin	Attribute	22.12.2018
Organizes	A 1:many relationship between System Admin and Schedule, as the admin can change schedules in case of clashes.	Relation	24.12.2018
Schedule	Details of the schedules in the university as midterm, finals, and course schedules	Entity	23.12.2018
Schedule_Name	Name of the schedule to identify it	Attribute	23.12.2018
File_Path	Shows the path of the pdf file to be shown for students when they open it, used for lectures, assignments, projects and schedules	Attribute	23.12.2018
Librarian	Details of the librarian working in the library of the university	Entity	23.12.2018
Librarian_ID	ID of the librarian, which is unique for every person in the university	Attribute	23.12.2018

Librarian_Pass	Password of librarian which is used to login in to the system	Attribute	23.12.2018
Librarian_Name	Name of the librarian	Attribute	23.12.2018
Lends	A 1:many relationship between Librarian and Book, as the librarian can lend books to students that request books.	Relation	24.12.2018
Book	Details of the books which are found in the library	Entity	23.12.2018
Book_ID	ID of the book to differentiate it from other books	Attribute	23.12.2018
Book_Name	Name of the book	Attribute	23.12.2018
N_of_pages	Number of pages of the book	Attribute	23.12.2018
Category	Category of the book	Attribute	23.12.2018

Table 5: Data Dictionary

10.4 Object Model

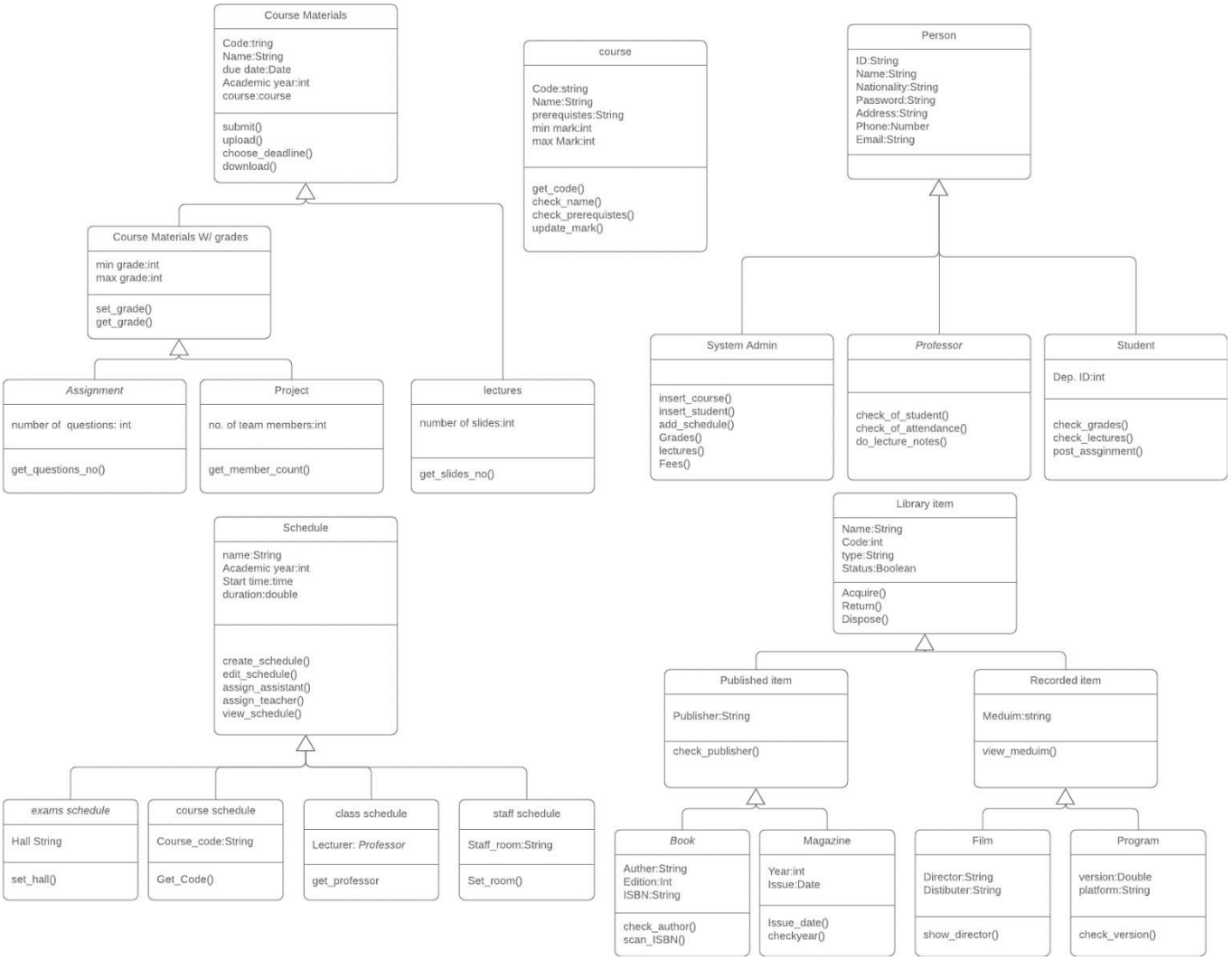

Figure 16: Object Model

10.5 Process Model

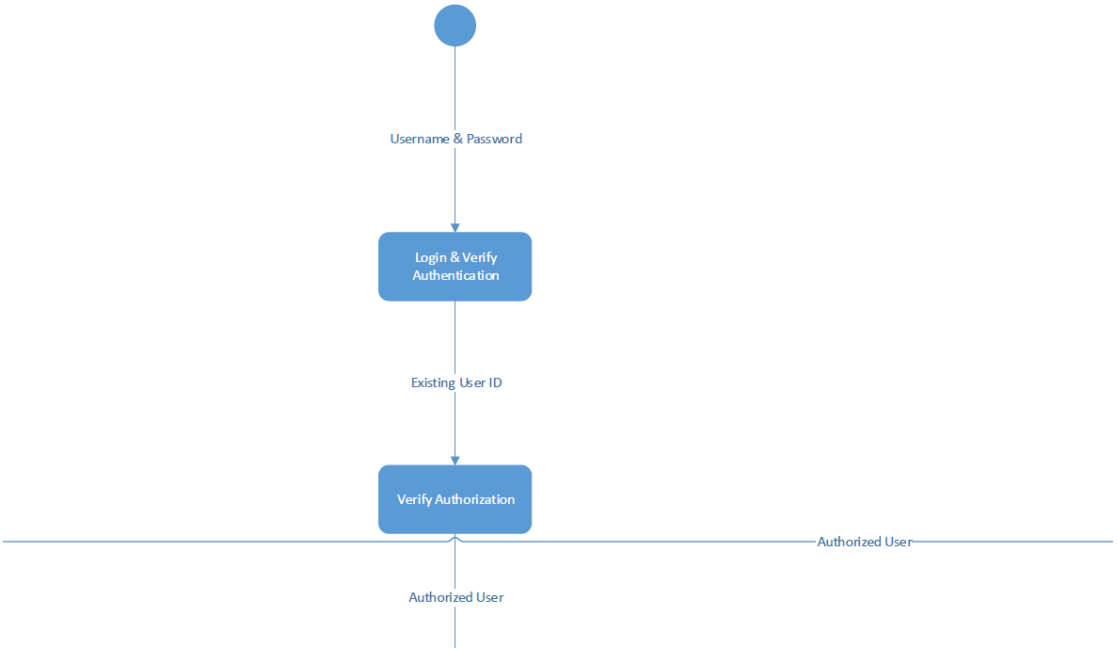

Figure 17: Process Model (Start)

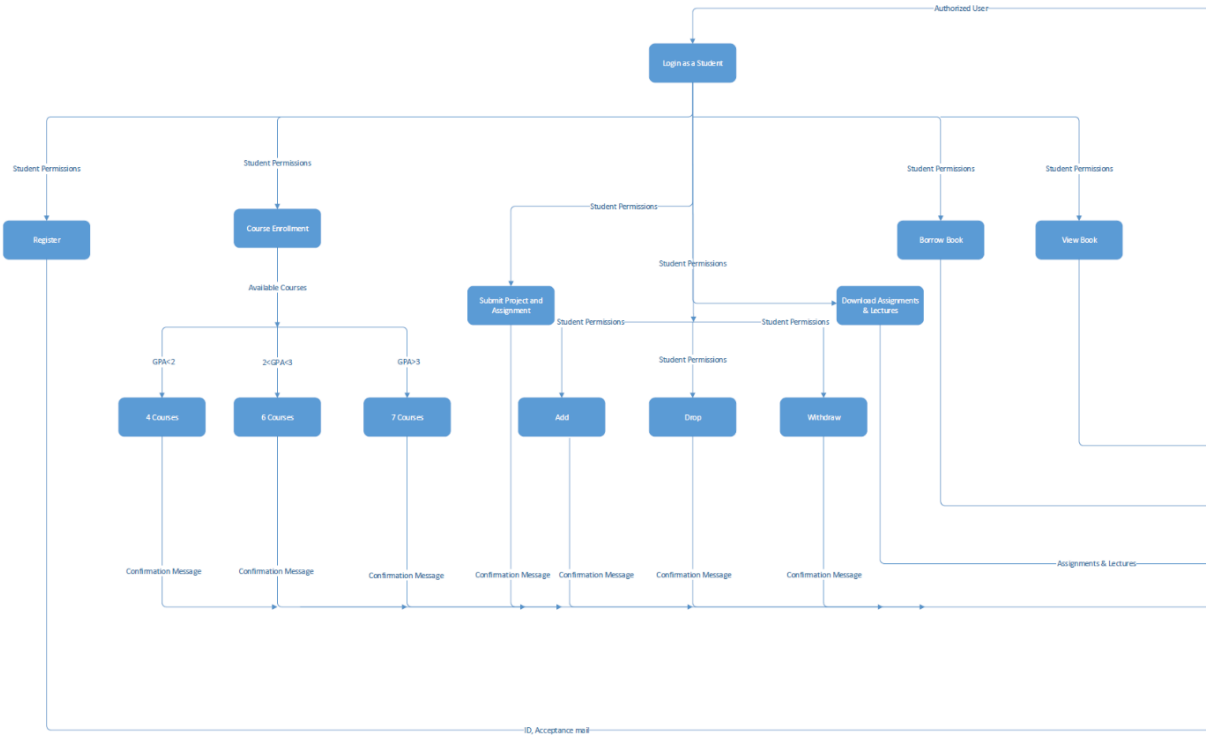

Figure 18: Process Model (Login as Student)

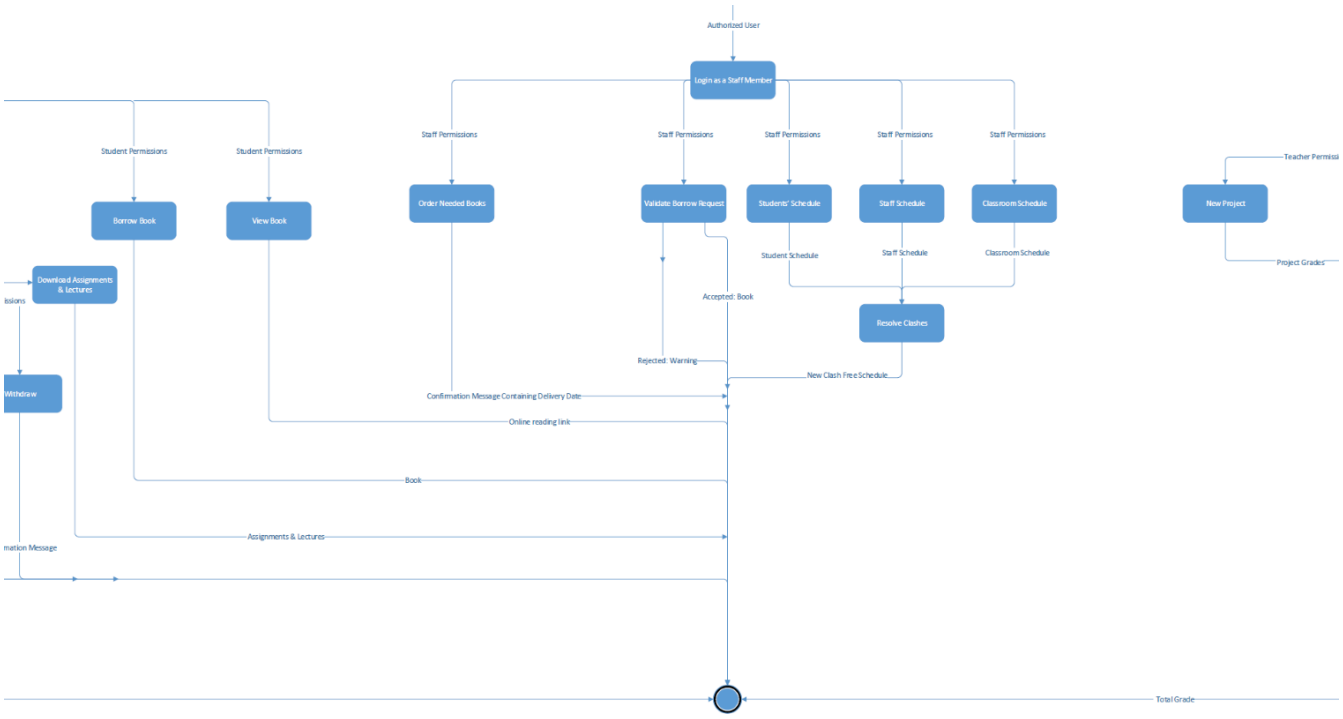

Figure 19: Process Model (Login as Staff)

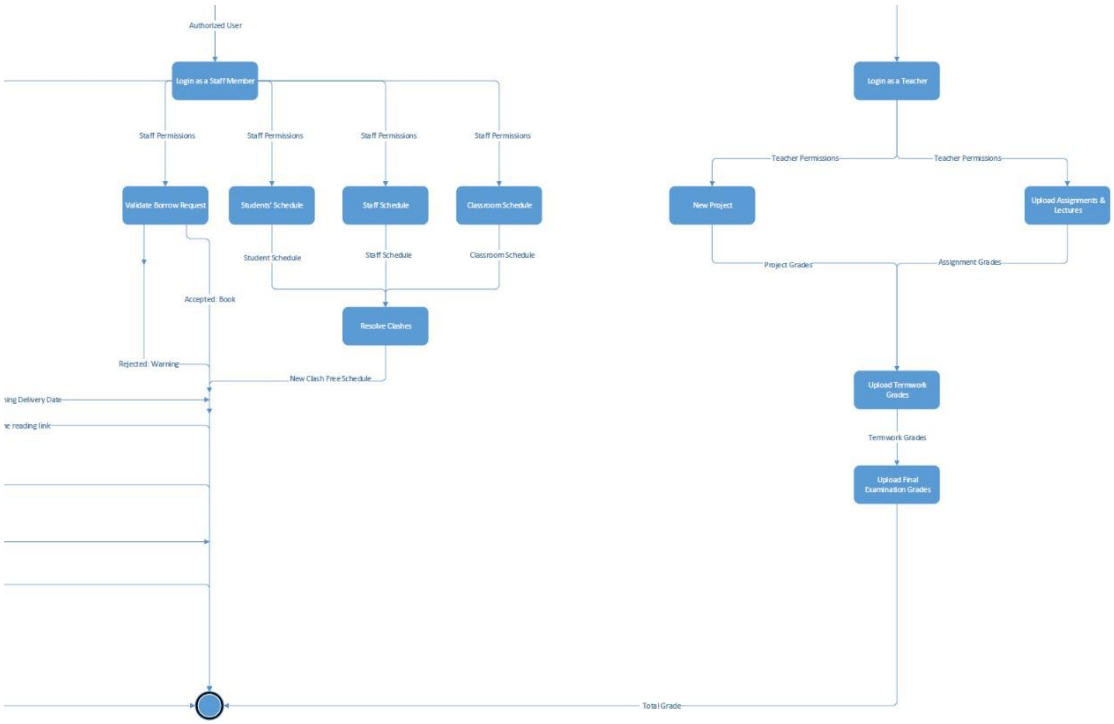

Figure 20: Process Model (Login as Teacher)

11. Analyzing Functional Points

11.1 Function Points Calculation

Measurement parameter	Count	Simple	Average	Complex	
Number of user inputs	19	3	6	9	114
Number of user outputs	18	5	10	15	180
Number of user inquiries	27	6	12	15	324
Number of files	250	3	4	5	1000
Number of ext. interfaces	31	8	12	15	372

Table 6: UFC Calculations

Count Total: 345

UFC: 1990

11.2 Complexity

Complexity factor:Fi	value=0	value=1	value=2	value=3	value=4	value=5	Fi
Backup and recovery	0	0	0	0	1	0	4
Data communication	0	0	0	0	1	0	4
Distributed processing systems	0	0	0	0	0	1	5
Is performance critical?	0	0	0	0	1	0	4
Existing operating environment	0	0	0	0	0	1	5
On-line data entry	0	0	0	0	0	1	5
Input transaction built over multiple screens	0	0	0	0	1	0	4
Master files updated on-line	0	0	0	0	1	0	4
Complexity of inputs, outputs, files, inquiries	0	0	1	0	0	0	2
Complexity of processing	0	0	1	0	0	0	2
Code design for re-use	0	0	0	0	1	0	4
Are conversion/installation included in design?	0	0	0	1	0	0	3
Multiple installations	1	0	0	0	0	0	0
Application designed to facilitate change by user	0	0	0	0	0	1	5

Table 7: Complexity Calculation

Sigma(F)=51

Complexity Adjustment Factor= 0.65 + 0.01*Sigma(F)=1.16

FP= UFC * Complexity Adjustment Factor = 1990 * 1.16 = 2308

Assuming:

- Estimated FP= 2308
- Organisation average productivity (similar project type) = 50 FP/p-m
- Burdened labour rate = 12000 $/p-m

Then:

- Estimated Effort=2308/50=(46.16)=47 p-m
- Cost per FP=12000/47= 255 $/m-p
- Project Cost= 12000*47=564000 $

11.3 LOC Approach

Functions	Estimated LOC	LOC/pm	$/LOC	Cost	Effort(Months)
Login	380	52	17	6460	15.6
Book Borrowing	600	78	11	6600	7.8
Book Viewing	360	46	11	3960	7.2
New Student Registration	680	89	14	9520	8.9
Schedule Management	1350	134	19	25650	21.4
Grading	510	72	13	6630	8.4
Assignments	280	63	15	4200	9.7
Projects	300	67	15	4500	10.1
Course Registration	710	91	18	12780	12.5
Add/Drop/Withdraw Course	910	106	18	16380	19.4
Totals	6080			96680	121.0

Table 8: LOC Calculation

Assuming:

- Estimated project LOC=6080
- Organisational productivity (similar project type) = 135 LOC/p-m
- Burdened labour rate = 12000 $/p-m

Then:

- Effort=6080/130 = 45 p-m
- Cost per LOC = 12000/130 = (92.3) = 93 $
- Project Total Cost= 12000*45 = 540000 $

12. Cost Analysis

12.1 Estimation by analogy

Blackboard is a software company located in the USA. It is superior in the LMS market and they always go the extra mile with their products. The company has been keeping a low profile with its cost estimations. The only available record of their cost per customer is an article by Michael Feldstein that can be found here. According to the Michael the company's software costs $250,000. After adjusting for inflation the cost before profit becomes **$312,547**.

12.2 Pricing to Win

This project is targeting Ain Shams university to serve around 100,000 students. According to this article the expected budget for those kind of numbers are around $1.5 per user per year or $150,000 per year. If the Contract period is 5 years, then the announced price should be **$750,000**

13. User Guide

13.1 Login

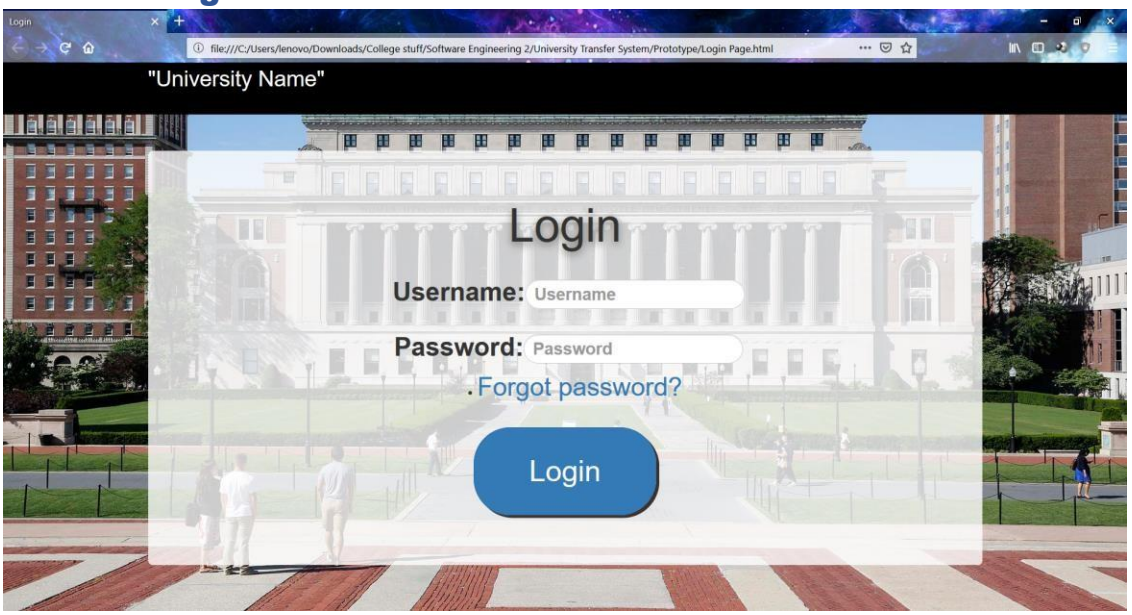

Figure 21: Login Page

Users of the website enter their username which is determined by their IDs and password which then opens different pages depending on the username

entered, there are 4 options after the login which are, student page, professor page, librarian page and system admin page. The page always shows the name of the user at the top right.

13.1.1 Student

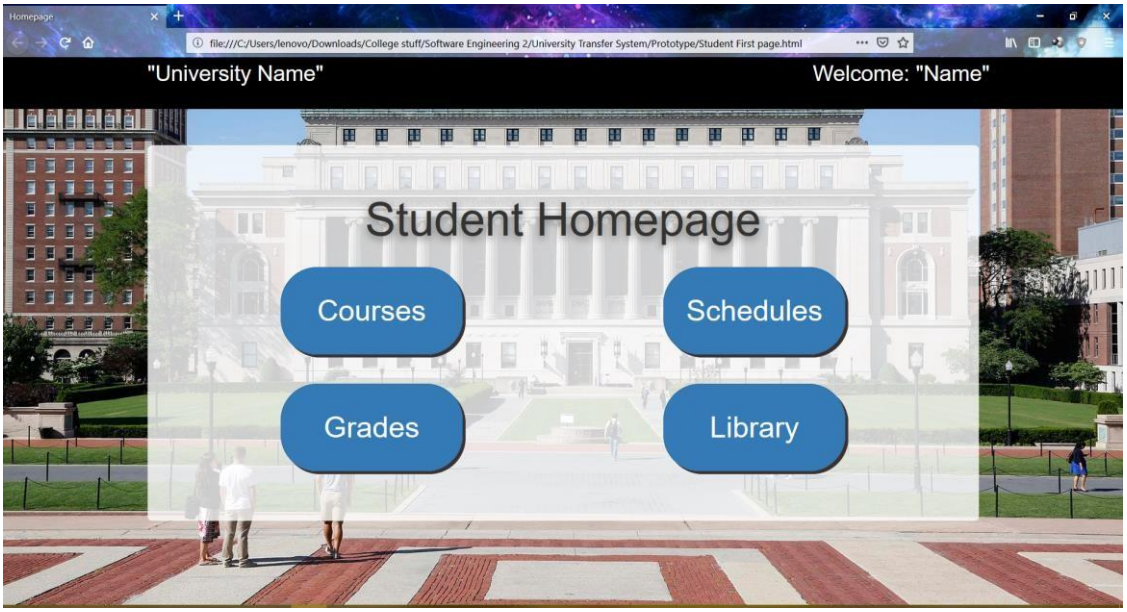

Figure 22: Student Homepage

The student has 4 options

- Courses: which allows the student to register new courses at the beginning of the semester, add or drop courses, view his current courses, withdraw from

courses and manage his courses to upload/download assignments, lectures and projects as shown

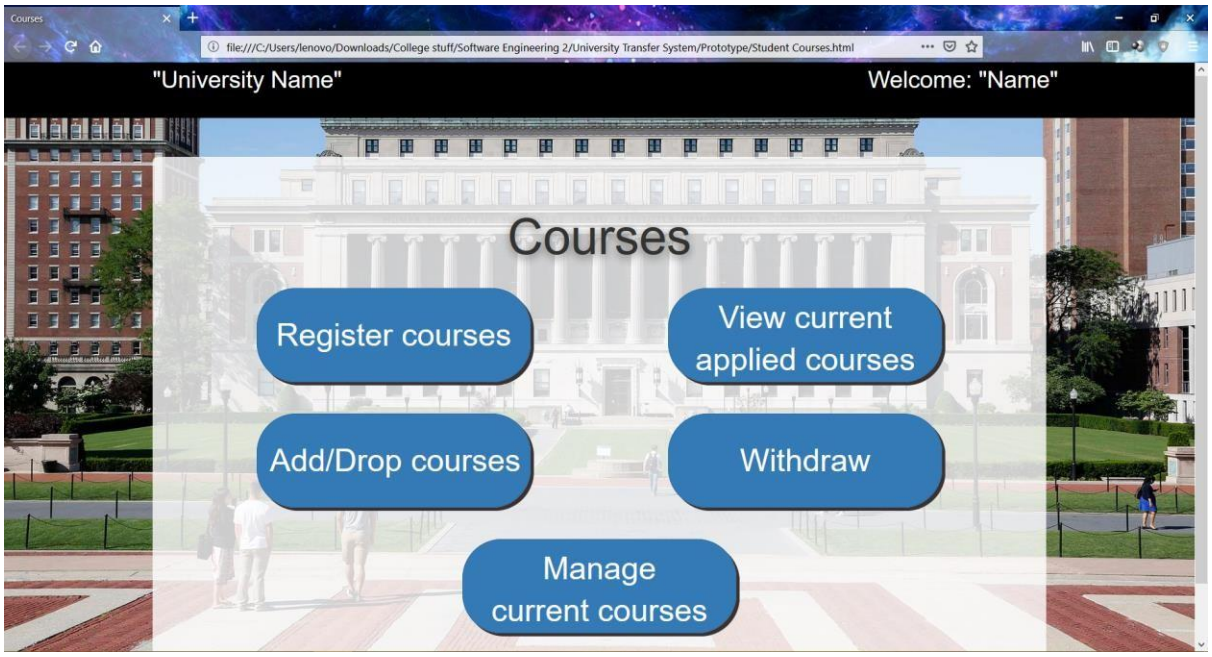

Figure 23: Student Course Management Page

- Schedules: Allow students to view academic calendar, courses schedule, exam schedules, and TAs and professors office hours. Each button in the page opens a pdf of the picked schedule.

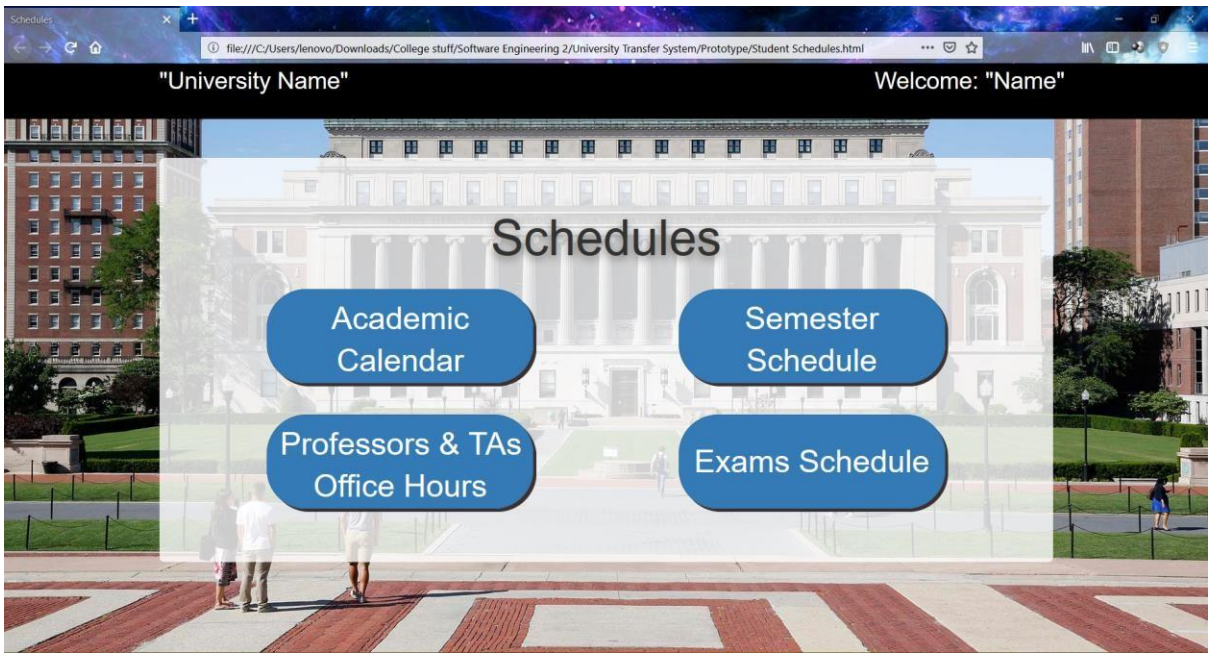

Figure 24: Student Schedule Management

- Grades: Which has 2 options, previous terms transcript shows a pdf of the final grades of all past semesters and the accumulative GPA, and current term grades show all the term work marks which are updated by the professors and shows the final grade when the semester is over.

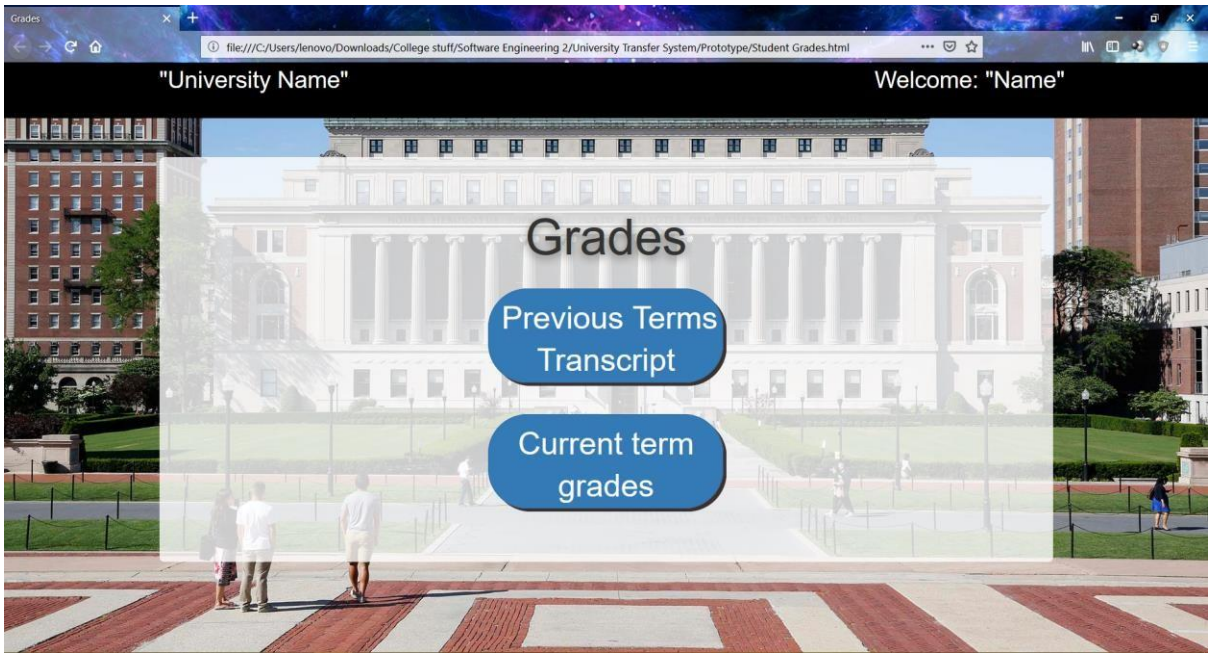

Figure 25: Student Grades

Semester grades are shown in a table as shown below, it shows which semester the student is in right now, with all the data of all coruses.

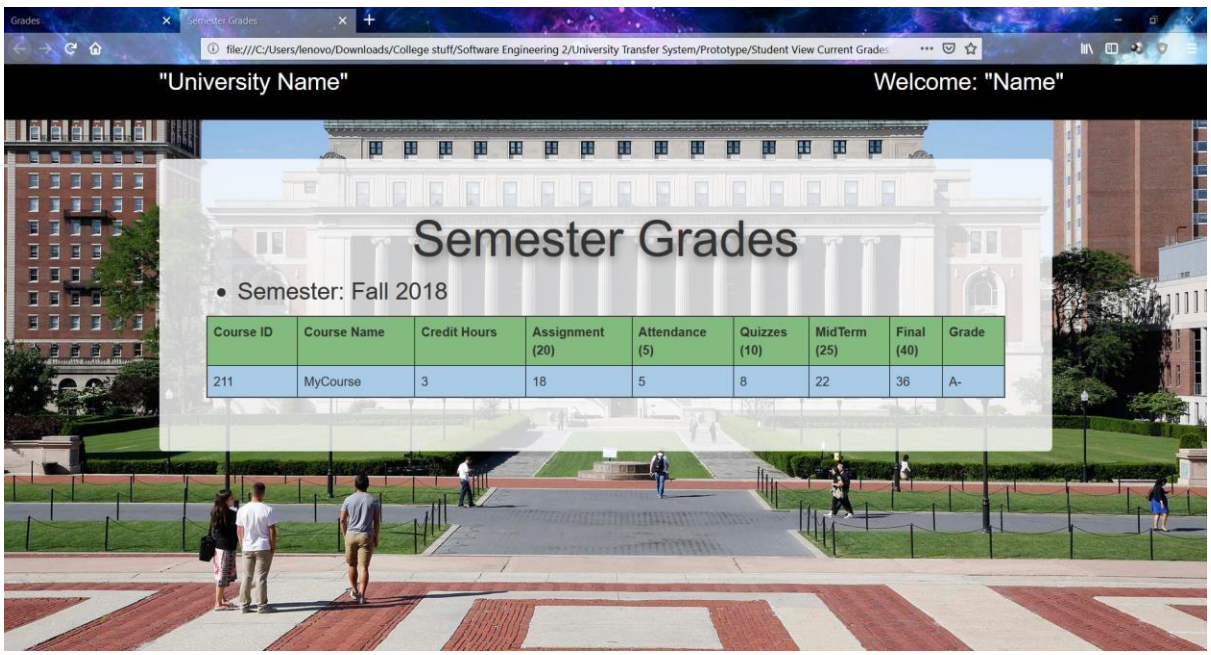

Figure 26: Student View Semester Grades

- Library: which shows the student all the books that are borrowed and requested, it also shows the due time of borrowed books to be returned and the availability of the requested books as shown.

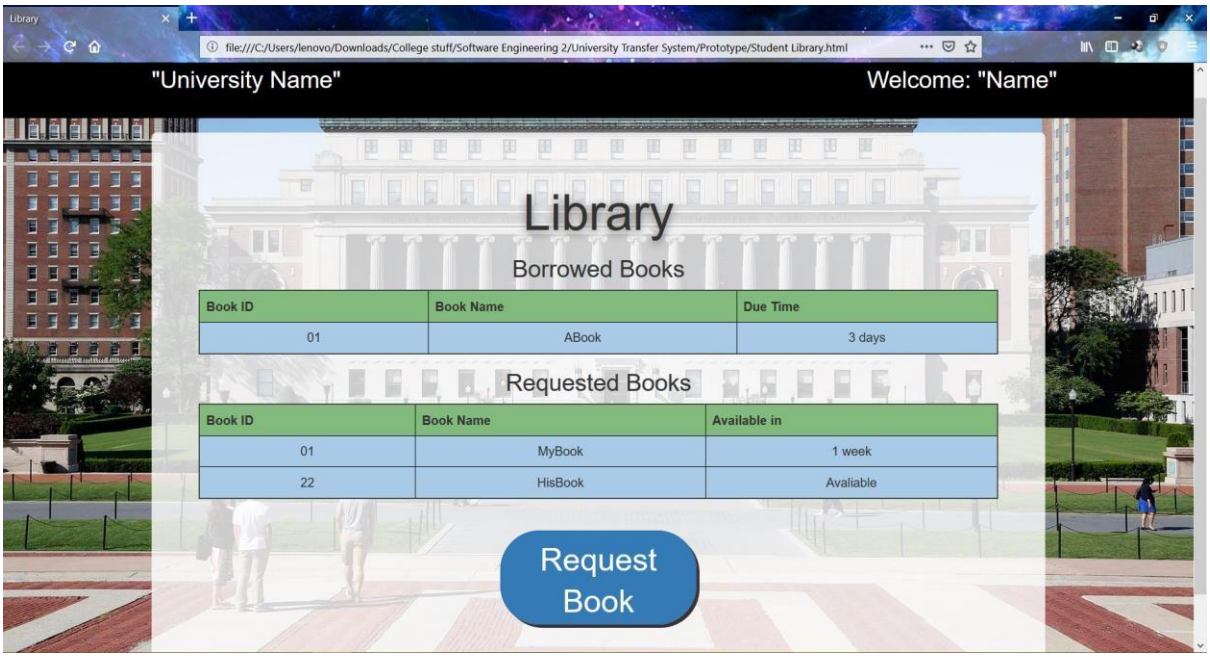

Figure 27: Student Library

The student should first request a book before borrowing it from the request book button, then he has to enter the details of the book he wants to borrow as shown below, after the student submits the data the book gets added to the requested books table.

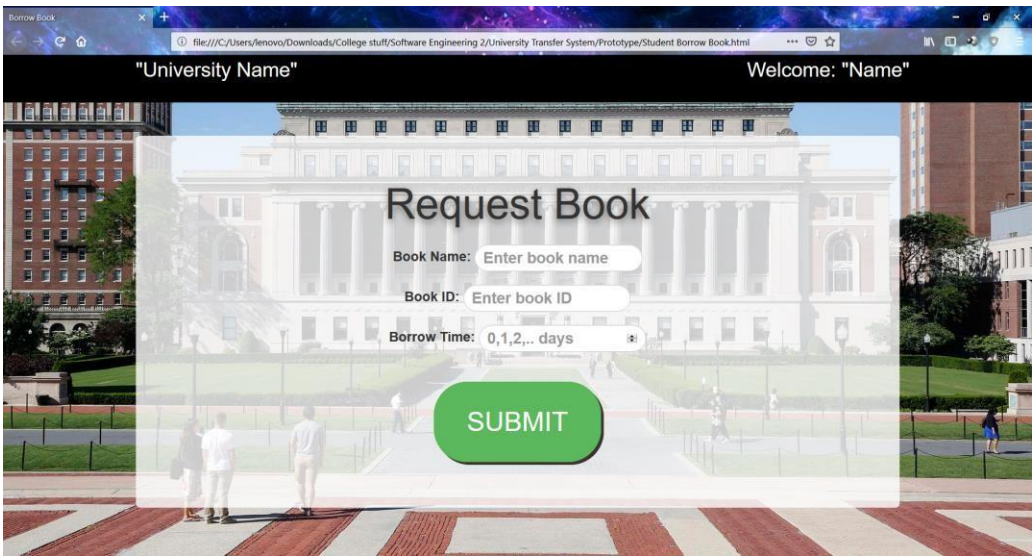

Figure 28: Student Request Book

Courses

The courses page has 5 options

- Register courses: Which allows the student to choose courses to register using the “Take” checkbox for the current semester, the student should choose courses based on the GPA, for example if the student picks 5 courses while his GPA is less than 2.0 the system will refuse. The student can take new courses or improve the grades of old courses if the student wants to.

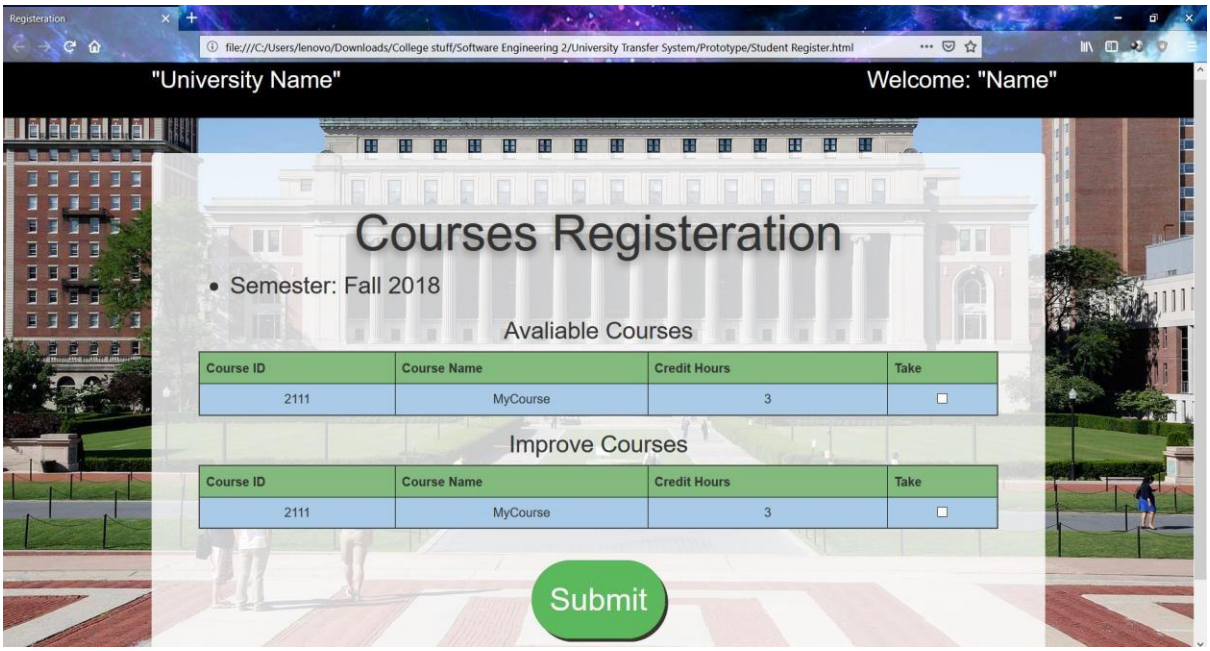

Figure 29: Course Registration

- View current applied courses: Shows the current semester and all registered courses and their data as shown.

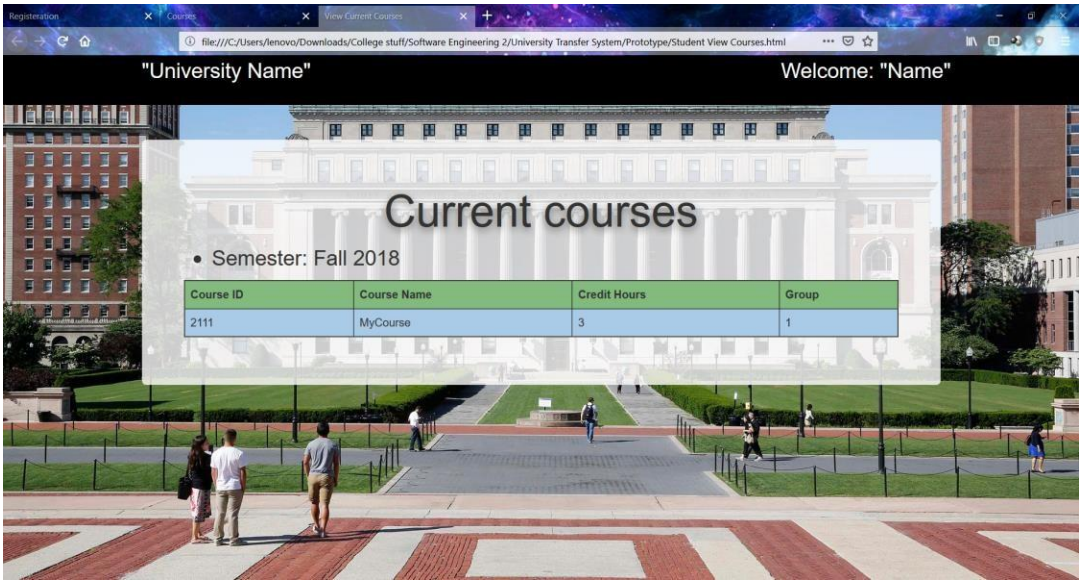

Figure 30: View Current Semester Courses

- Add/drop courses: Allows student to add or drop courses from this semester using the checkbox as shown according to the minimum and maximum number of courses the student can take per semester. (Dropped courses are not added to semester fees)

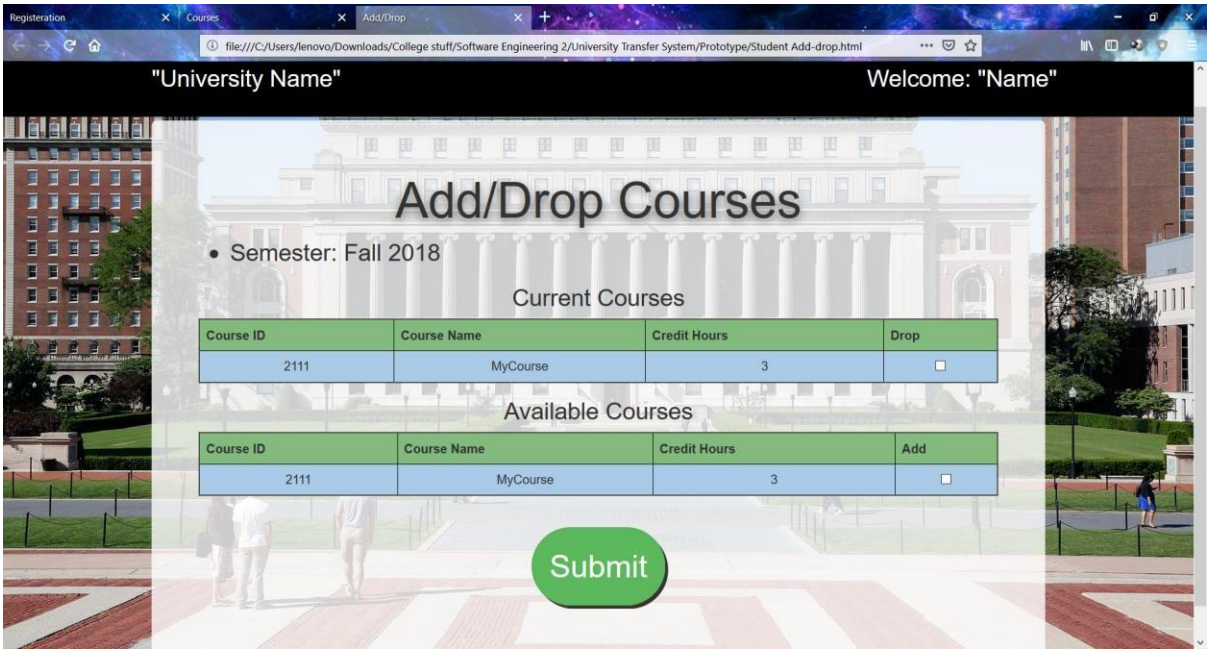

Figure 31: Add/Drop Courses

- Withdraw: Allows student to choose a course to remove from current semester. (but he still has to pay its fees)

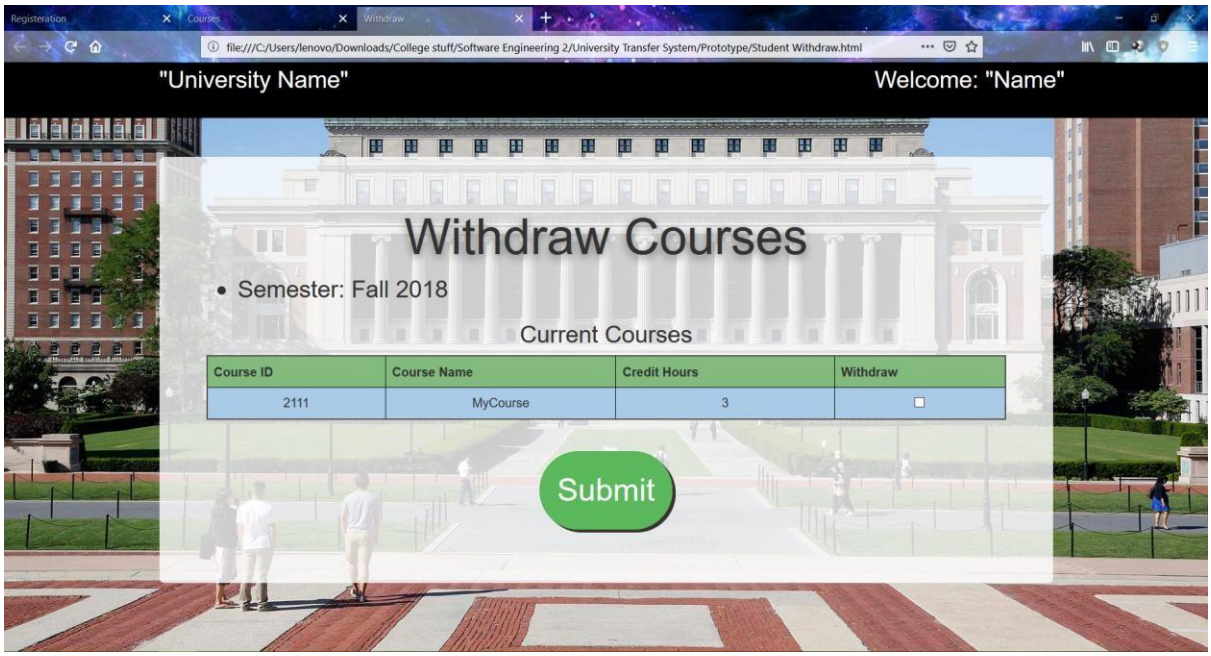

Figure 32: Withdraw Courses

- Manage current courses: Allows student to manage his courses, so it opens a page which shows all registered courses for this semester as shown.

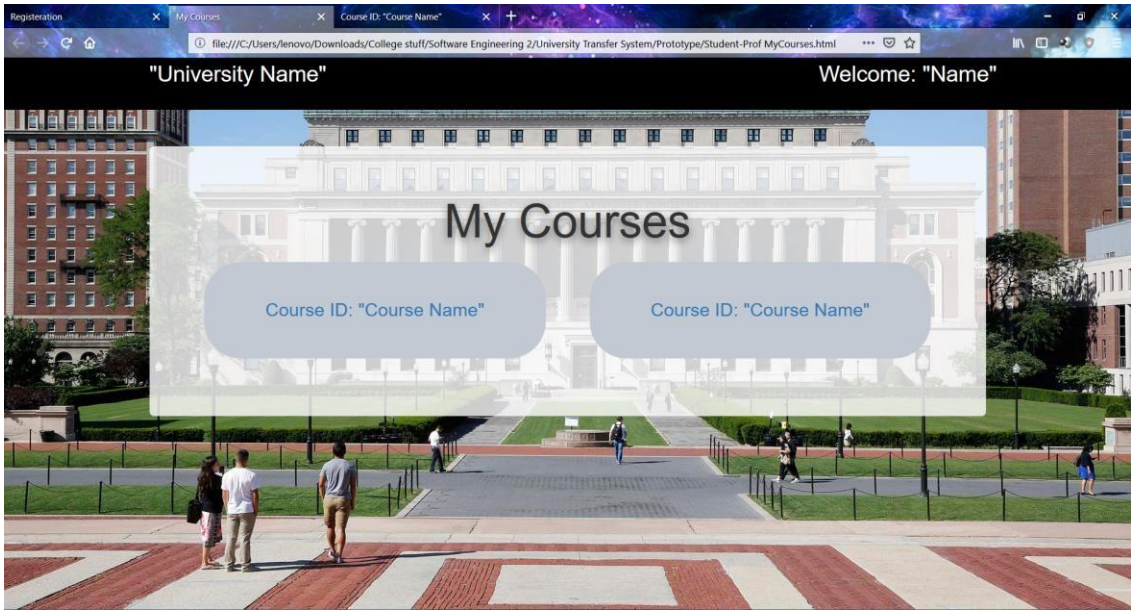

Figure 33: Student Course Management

Each course has 4 options to show which are assignments, projects, lectures, and others as shown.

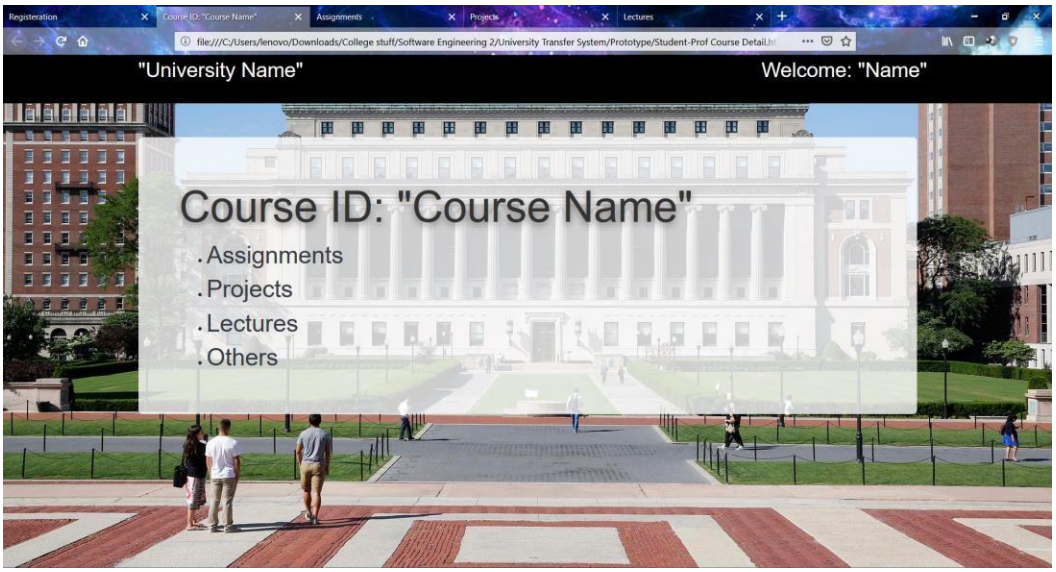

Figure 34: Course Page

All options contain files to be downloaded according to picked option, for example if a student opens lectures the student can download any lecture that the professor has uploaded. Assignments and projects should be uploaded by the student to be graded by the professor, so the assignments and projects pages show the next assignment/project to be uploaded and gives student a button to upload from as shown. (To the left are the assignments to be done which are uploaded by the professor, on the right the student should upload his/her solution, the same is done for projects)

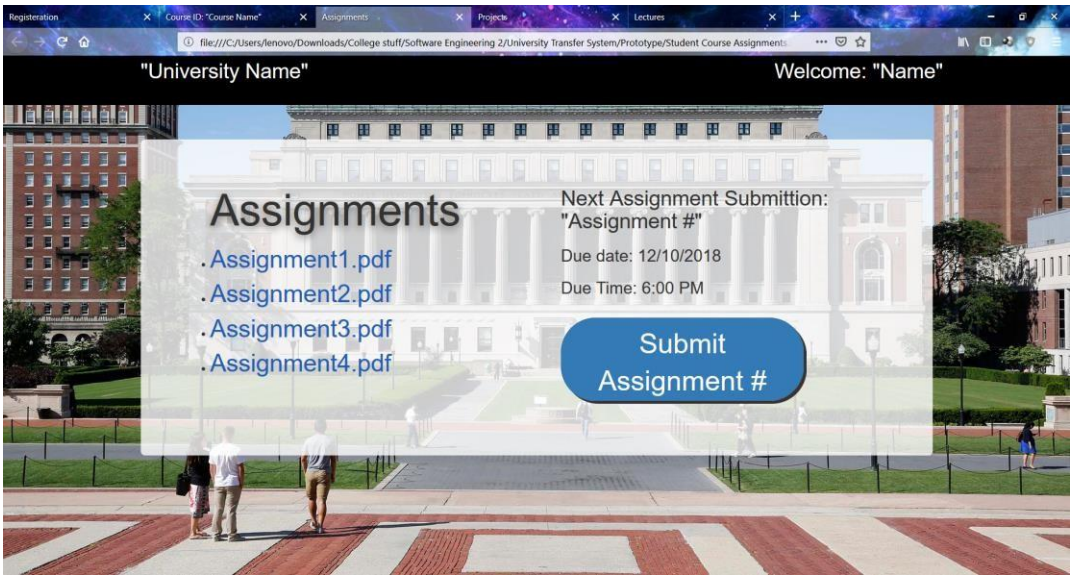

Figure 35: Course Assignments Page

The “others” folder contain any files that the professor wants to share with the students for further references.

13.2 Professor

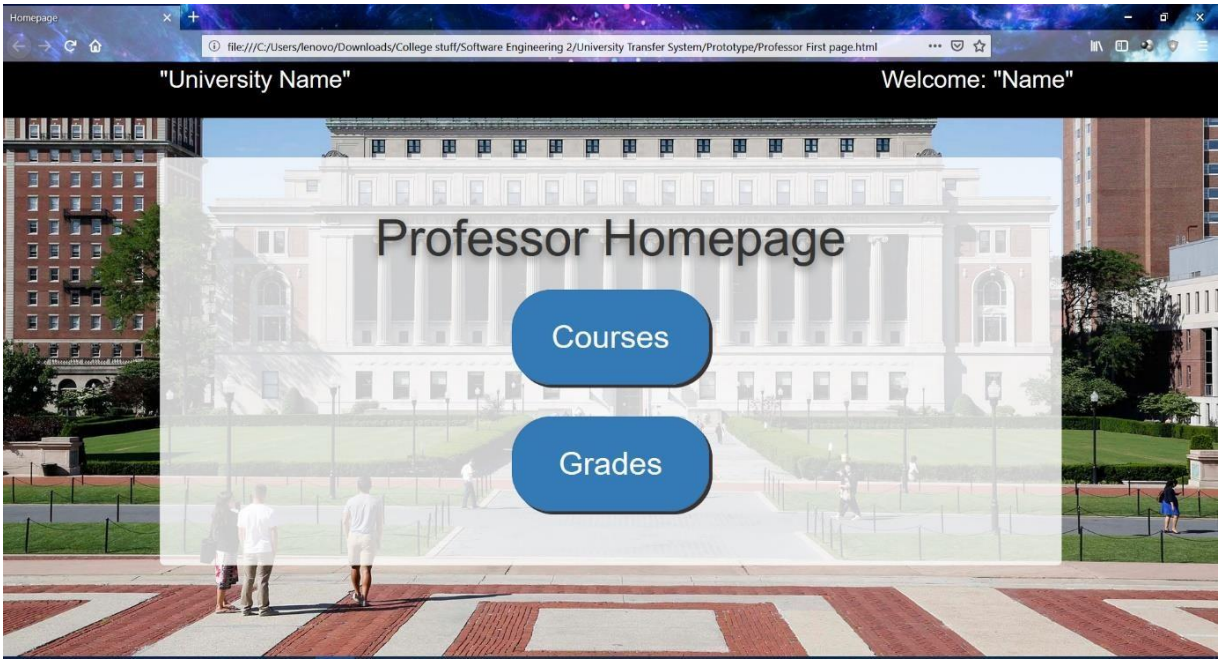

Figure 36: Professor Homepage

Professor has 2 options, managing the courses he teaches from the courses button, and managing students’ grades from grades button.

13.2.1 Courses

The professor has the same interface as students when they manage their courses and has the same folders, but inside the folder the professor has the option to upload assignments, lectures, and projects as shown.

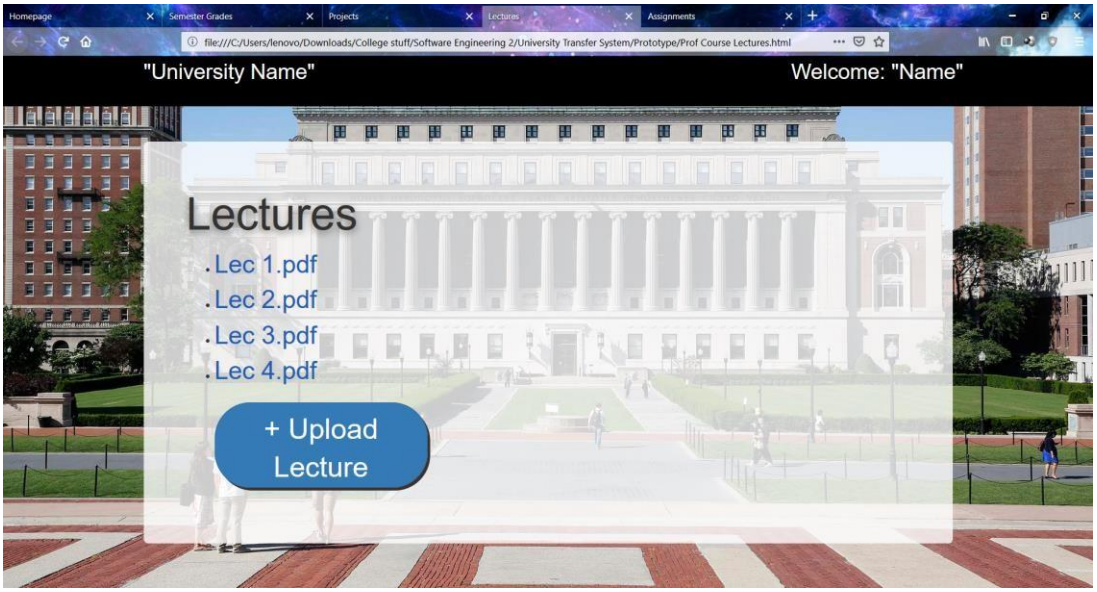

Figure 37: Professor Lectures Page

When uploading a lecture, the professor only browses it from the computer and it's uploaded but when uploading projects and assignment (after pressing the upload project/assignment button)

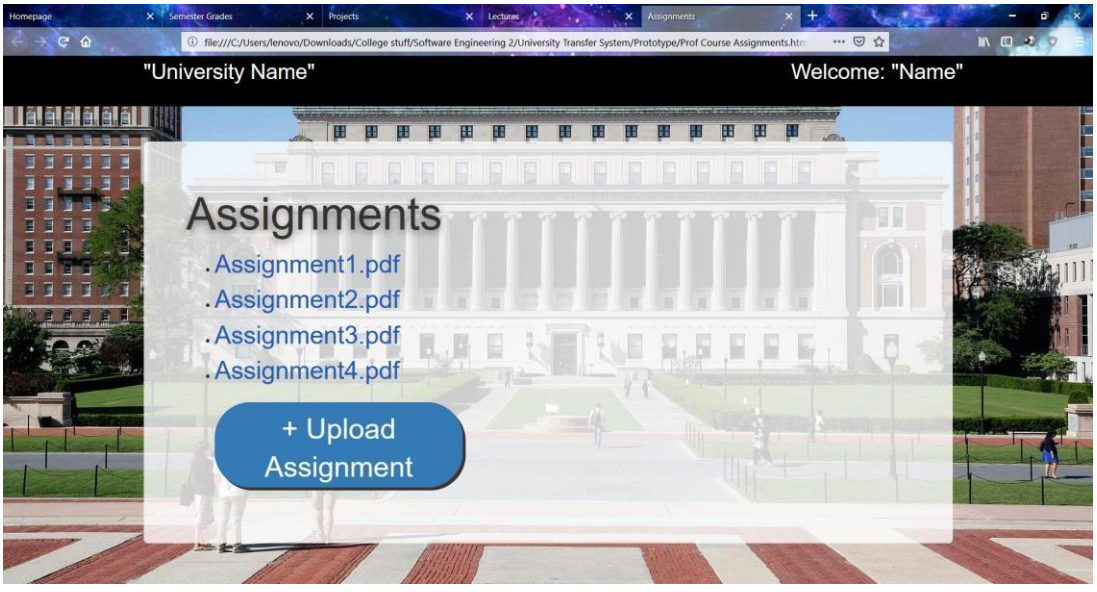

Figure 38: Professor Assignments Page

The professor is taken to another page to put the deadline for uploading the project or assignment as shown.

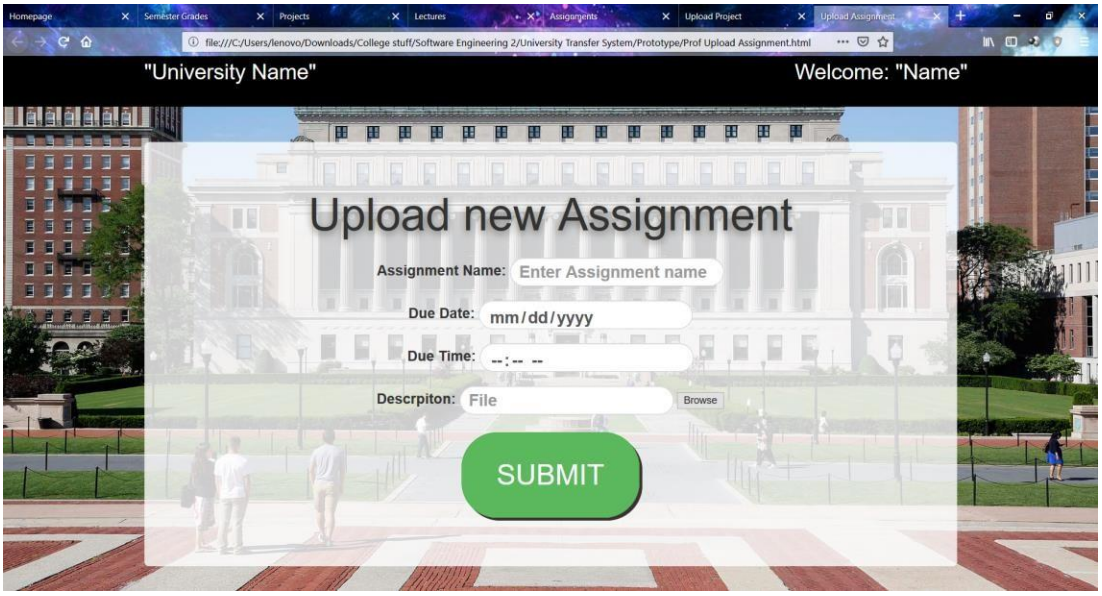

Figure 39: Professor Upload Assignment

13.2.2 Grades

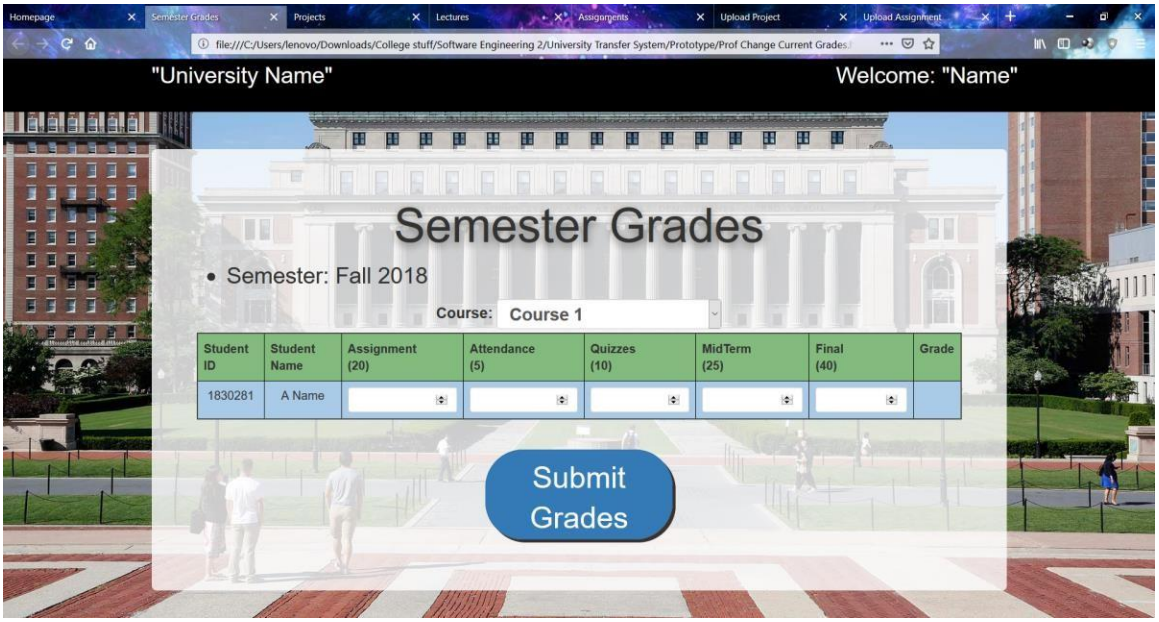

Figure 40: Professor change grades

If a professor teaches more than a course, he chooses the course he wants to change the grades in from the dropdown menu, then a table with all students in this course will be shown, the professor writes all the grades of the students then submits them from the submit button.

13.3 Librarian

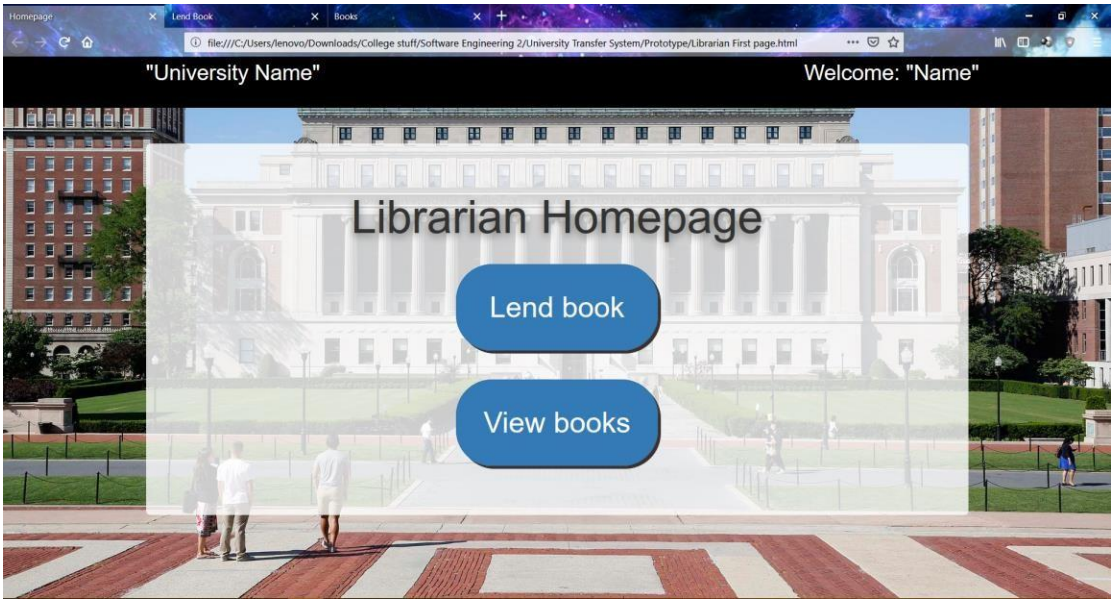

Figure 41: Librarian Homepage

The librarian can view the state of all books and all requested books using the view books button, which brings the librarian tables that contain all these data as shown below.

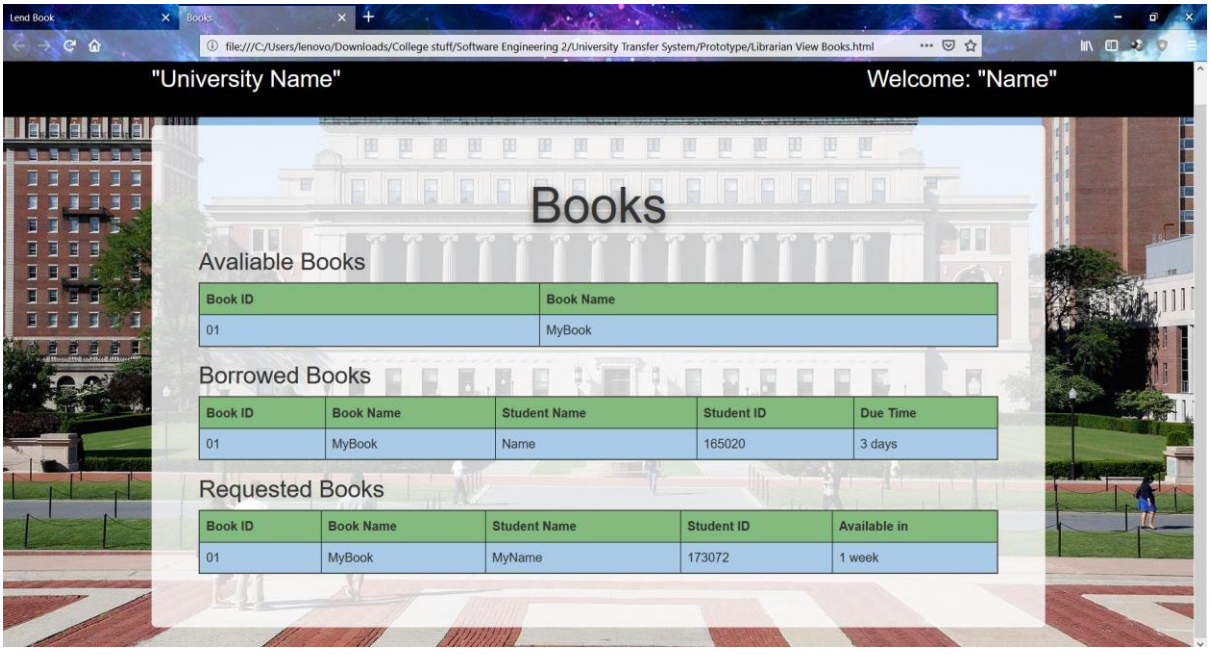

Figure 42: Librarian View Books

The librarian can also lend students books from the library and sets the due time for the students to return these books from the lend book button as shown.

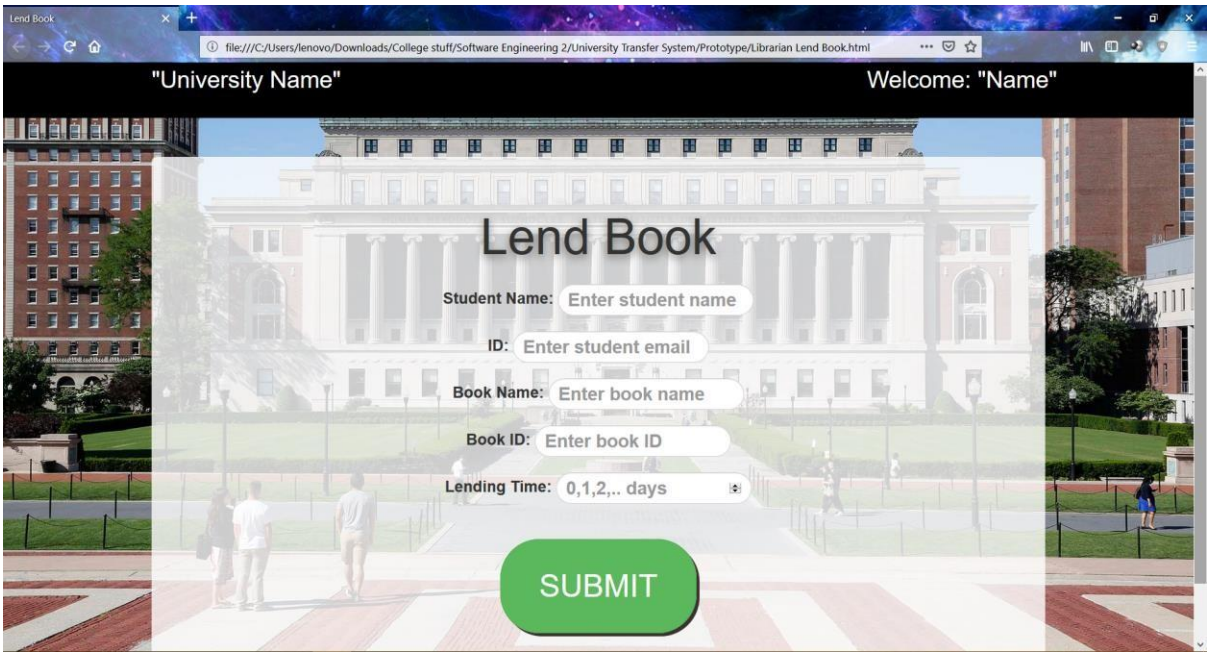

Figure 43: Librarian Lend Book

13.4 System Admin

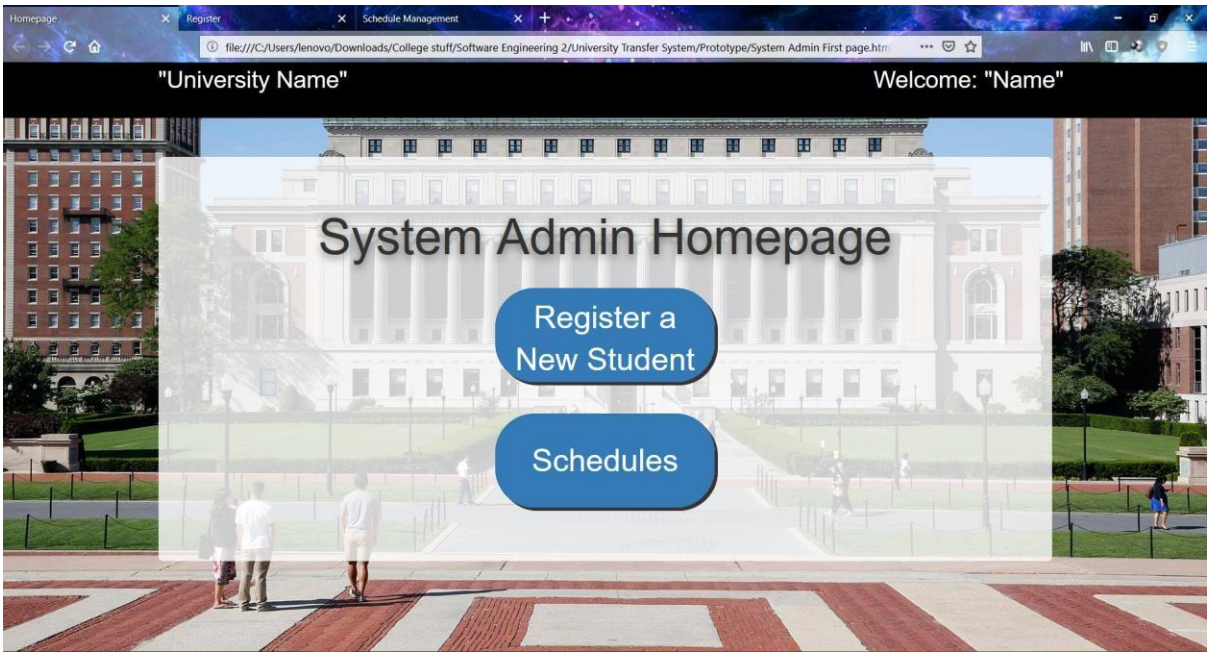

Figure 44: System Admin Homepage

System admins can register new students from the first button, which shows the admin the student registration form shown below. A new student is added to the database after the admin presses the submit button.

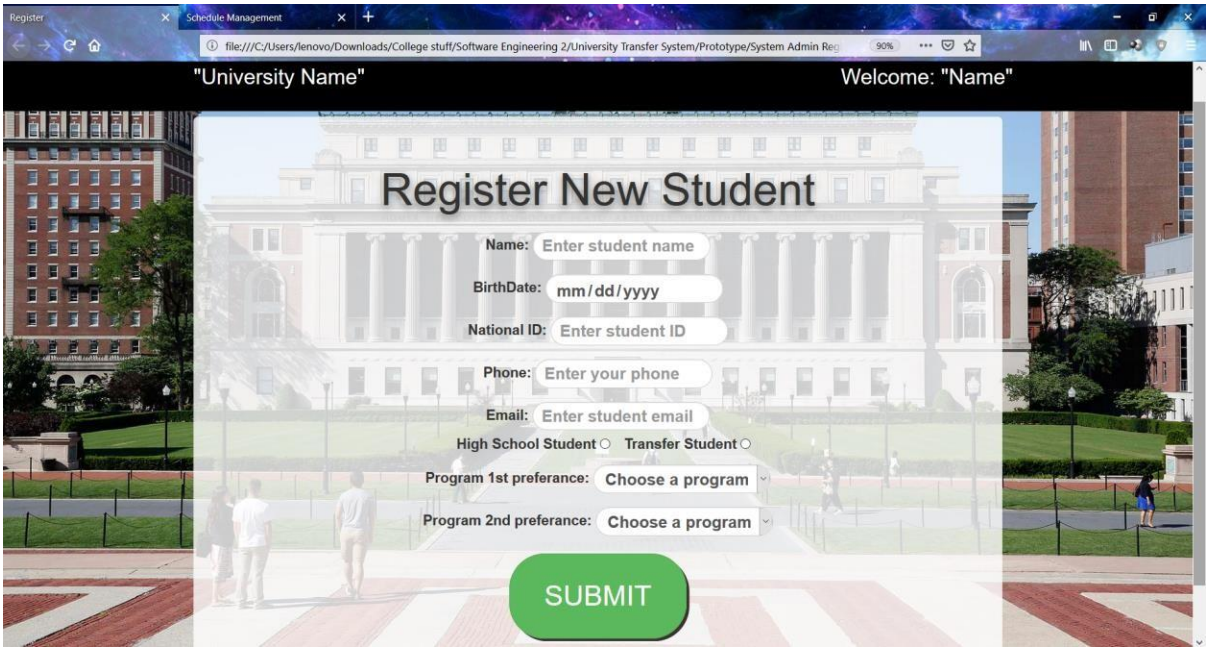

Figure 45: Admin Register New Student

The system admin can also update any schedule. The schedules are made at specific times using all the data of students and professors then are uploaded to the site as pdfs, the admin can view this pdf and can make changes to the schedule using an external system and upload the new one instead of this. The admin chooses the schedule he wants to view/update using the dropdown menu.

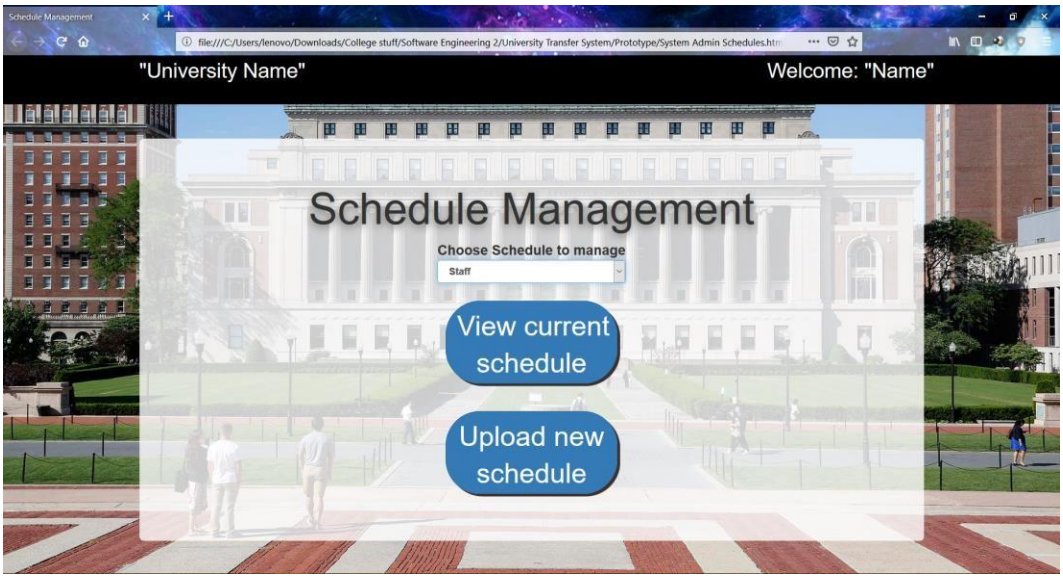

Figure 46: Admin Schedule Management

Not all options in the site are available simultaneously (as registration and withdraw both open in different times in the semester) but all of them exist at the same time, buttons that cannot work yet takes the user to the same page they are in.