UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Web Applications A.Y. 2021-2022

Homework 1 – Server-side Design and Development

Master Degree in Computer Engineering
Master Degree in Cybersecurity
Master Degree in ICT for Internet and Multimedia

Deadline: 22 April, 2022

Group TORE	Project We are implementing a Learning Management System (LMS)	
Last Name	**First Name**	**Badge Number**
Ahmed	Saad	2044003
Bahrami	Sepide	2043887
Hansen	Marit Fredrikke	2062124
Rao	Abdul Moeed	2044017
Rehman	Abdul	2048675
Sohail	Mohammad Muzammil	2043886
Soomro	Dodo Khan	2051143
Sultanova	Aliia	2041422
Zabalawi	Iyad	2041407

1 Objectives

TORE is a Learning Management System (LMS). It's purpose to empower departments with training and development for the learners, to provide ways of professor-student communication, sharing the information and making the studying process easier. All courses are offered through the platform. We would like to represent an application because audiences take most naturally to an LMS when it adopts modern solutions that people are used to working with. More traditional modes like paper tests and in-person instructor-led training just don't hit the mark for modern learners these days. Our team will include crucial part for any LMS as application management of profiles, signing up, logging in, enrolling/quitting a course, uploading files, the ability to register courses and to share course materials, to make announcements.

2 Main Functionalities

TORE- Learning Management System will allow students and teachers to interact with each other in an easy way to share study material and enquiries creating a central platform. Major functionalities will be as following

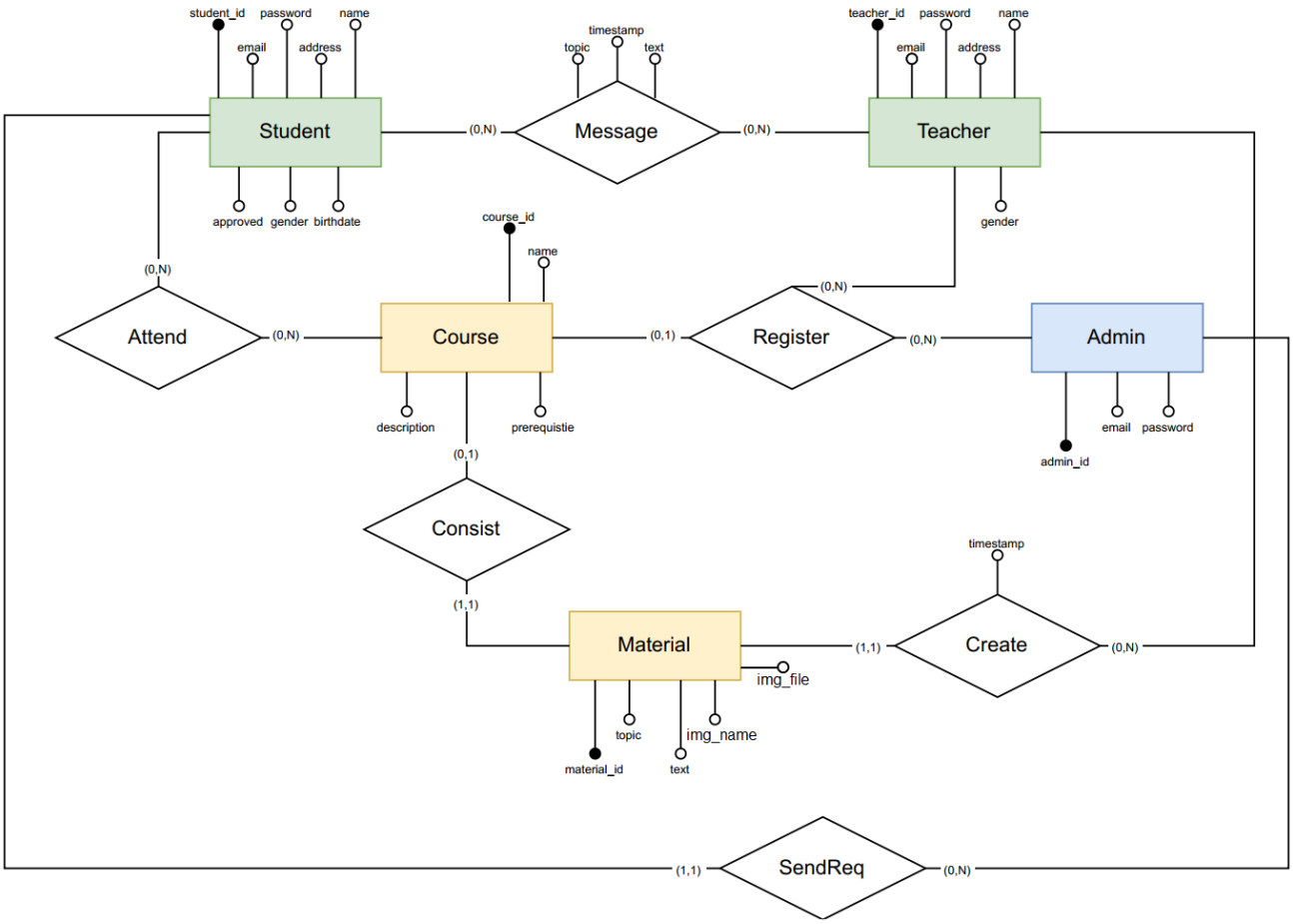
1. **Main Area**. Accessible to everyone interested in knowing about the system and will be used for registering new users and signing up existing users.
2. **Admin Area**. This area will only be accessible to users with admin rights. It will hold following functionalities.
    - Approval of all users registering for the system.
    - Maintenance of profiles according to type of account.
    - Maintenance of Database.
    - Assignment of tasks to teacher accounts.
    - Maintenance of subject allotment to teachers.
3. **Faculty Area**. This area will be restricted for the use of teachers only and will perform following
    - Each teacher will be able to access his own specified subjects.
    - Teacher will have the authority to manage students and course contents.
    - Management of student affairs including enrolment.
    - Replying to the comments/ questions made by the students.
4. **Students Area**. This area will be accessible to all users registered as students and will offer following functionalities
    - Enrolment to subjects/ courses.
    - Viewing contents shared by teachers of each course.
    - Submitting questions/ commenting on course contents.

Major roles of different users will be following

1. Admin. Responsible for over all management and approval of all users registering.
2. Teachers. Will be able to deal with subjects allotted by admin and enrol students.
3. Students. Can register and enrol into subjects offered after approval from admin.

3 Data Logic Layer

3.1 Entity-Relationship Schema



The ER schema contains 5 main entities:

- Admin: This entity consists of the system admin and its primary key is an integer with auto-increment. Also, it has email(username) and password for logging into system as the admin user.
- Student: This entity is another type of the available users in the system. The primary key is an integer with auto-increment and in addition, there are some attributes used for logging into system (email, password) and personal information (name, gender, birth-date, address) for registering in the system. There is another Boolean attribute named as 'approved' which its value is set to true after the admin accepts the student's registration request.
- Course: This entity is the collection of courses available in the system for the students to enrol in. The primary key is an integer with auto-increment and the other attributes are used to save some information about each course.
- Teacher: Another type of users is the teacher entity which are registered (signed up) into system by the admin. Then one or more courses can be assigned to them.

- Material: Each course may have some materials (homeworks, announcements, etc). Therefore, they are all kept by having an integer auto-increment id for each material and their topic, text, and attachments if available are saved. Attachment can only be of maximum one image per each material posted.

3.2 Other Information

The n-ary relationship between Admin, Teacher and Course allows the admin to register teachers in the database and to assign one or more courses to each of the teachers. Thus, there would be no conflicts since admin does the assignment task.
Also each student, after filling in the registration form must wait to be accepted by the Admin (approved=true) and then can log into the system, enrol in courses, send messages to teachers and see the course materials available.

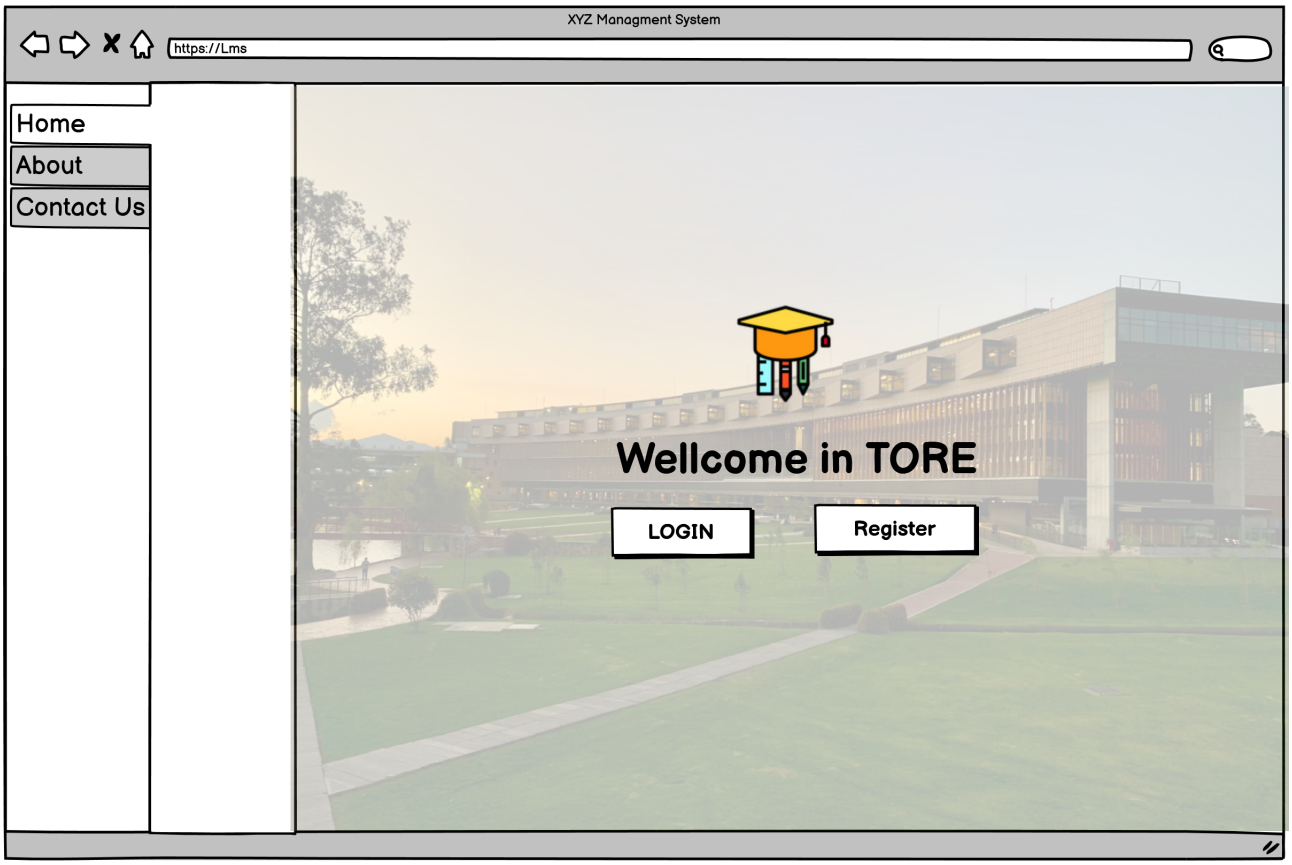
4 Presentation Logic Layer

The website will be divided into the following pages:

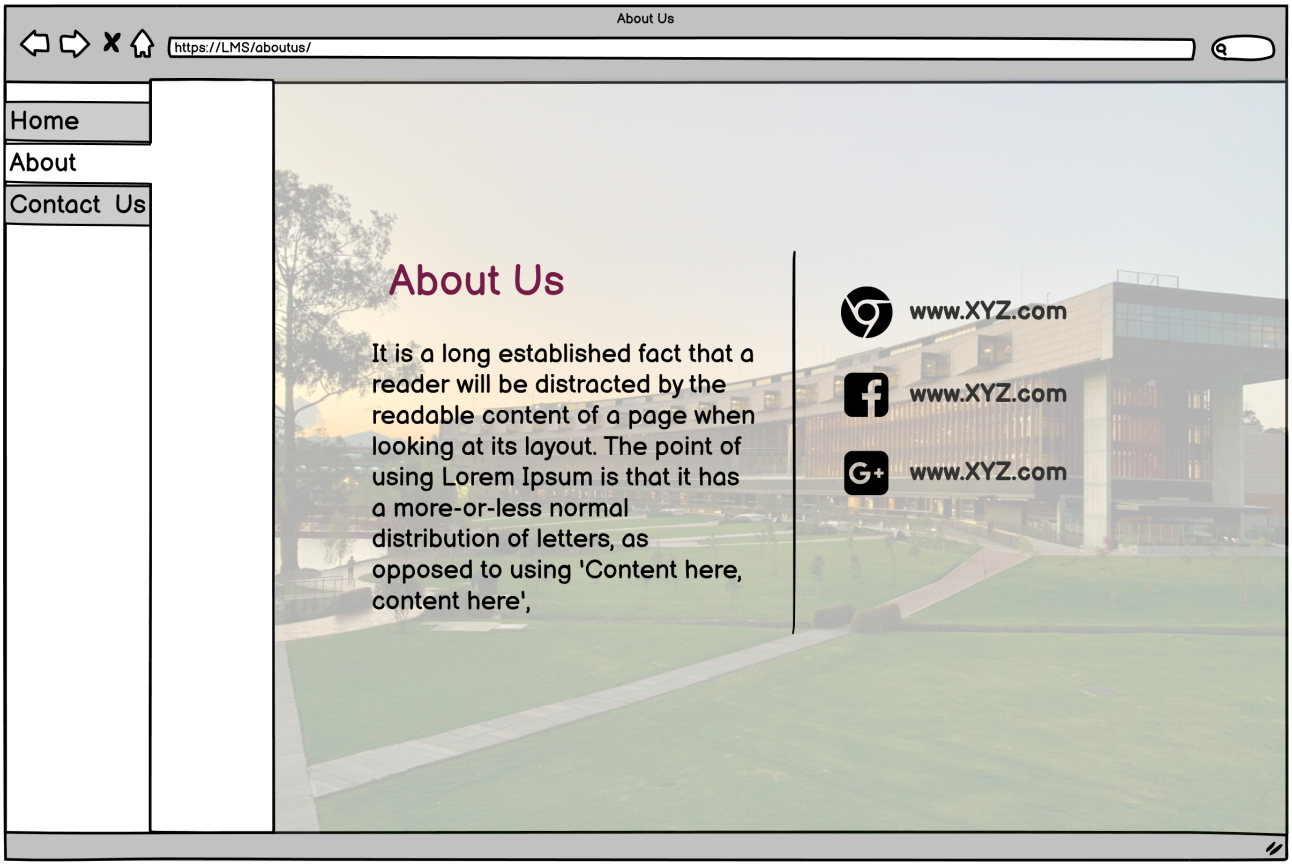
- Homepage: allows users to login or register to the system through LOGIN and REGISTER buttons, respectively and contains area with the sections AboutUs and ContactUs. Developed via jsp.
- Login page: allows the users login to the LMS. Developed via jsp.
- Student Registration page: allows students to register in the LMS. Developed via jsp.
- Administration page: allows to administrate users' accounts, register and add new teachers to the system. Written in HTML.
- Teacher page: allows teachers to administrate the courses' and students' details. written in HTML with the support of javascript.
- List students page: allows to manage the enrolled courses. Written in HTML.

4.1 Homepage

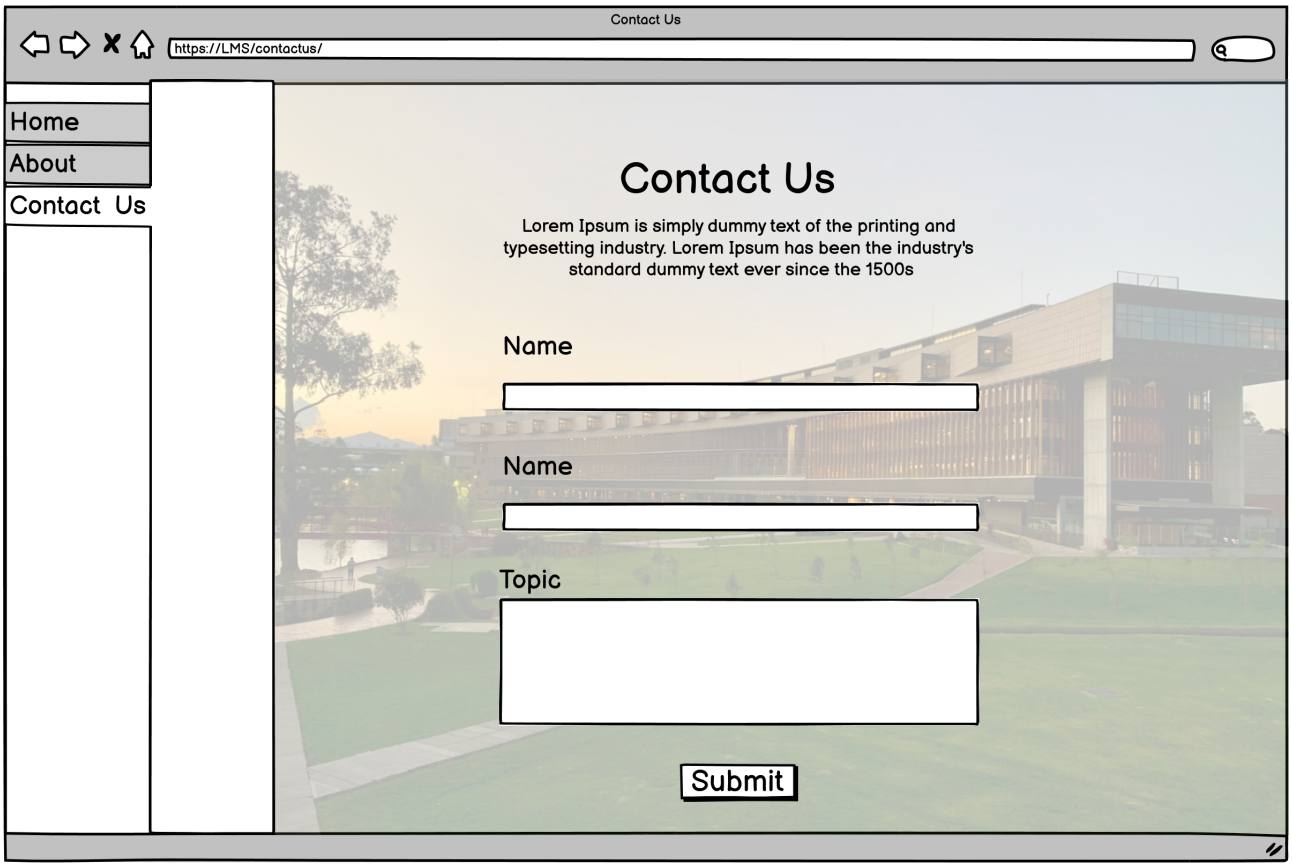
The homepage contains the LOGIN button for the users who has already registered and using the system and the Register button for new users. Pressing the REGISTER button the user will go to the student registration page. The left-side area of the homepage will contain the information sections as About Us and Contact Us. There will be the description of the TORE project, contact links for our social media pages and the contact form for sending us messages.



The about us page contains the information about the TORE system and all useful links to contact with us.

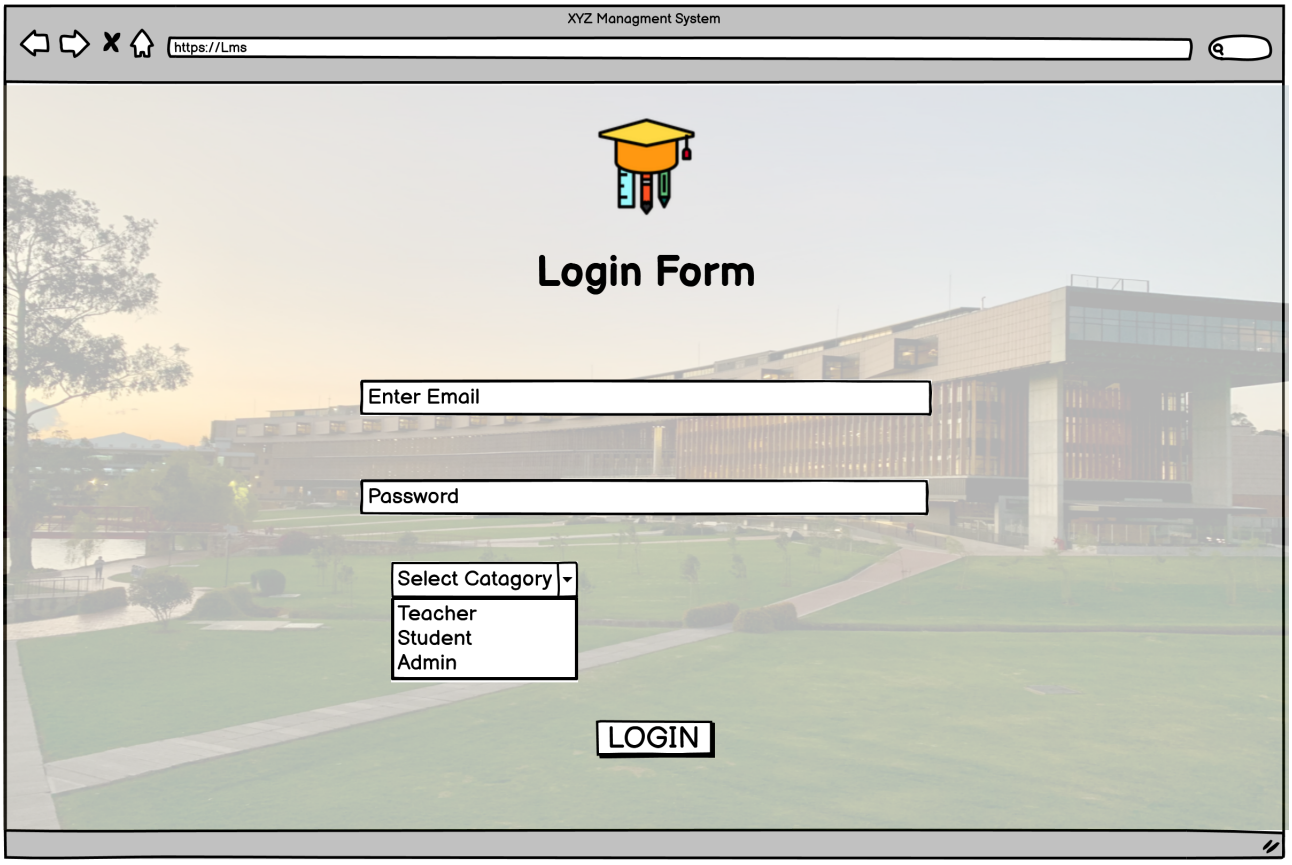


The contact us page contains a form for manual filling in the name of the user, the subject of the message and the description of the topic. By clicking the "Submit" button, the message is send.



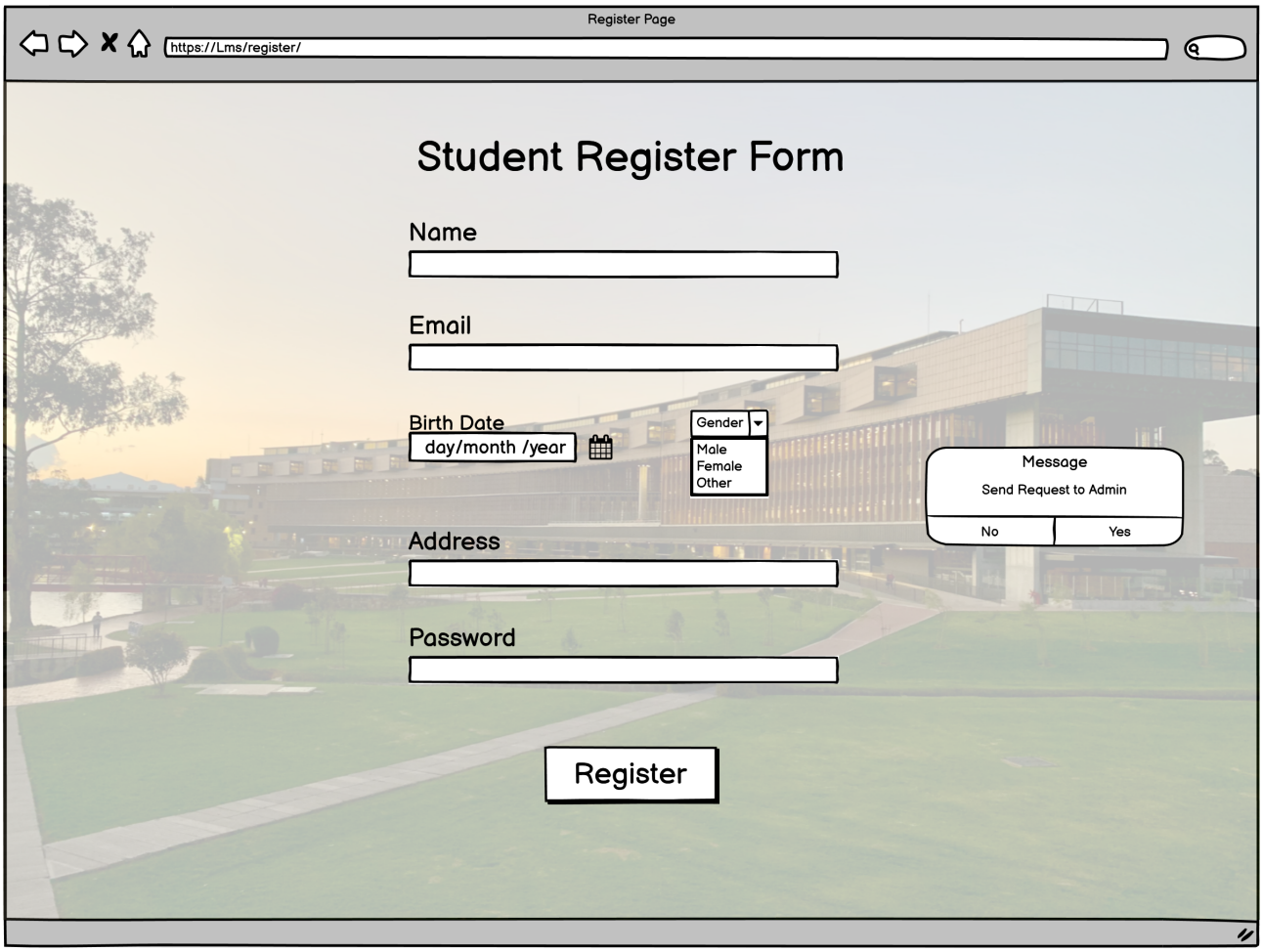
4.2 Login Page

The login page contains the form used to having an access to users own page inside the system. There are two input fields for e-mail address and the password, respectively. The user can select his/her status at the category field as teacher, student or admin. By pressing the LOGIN button the user will go to the personal page with all the information concerning his/her courses that are in progress or have already been finished.



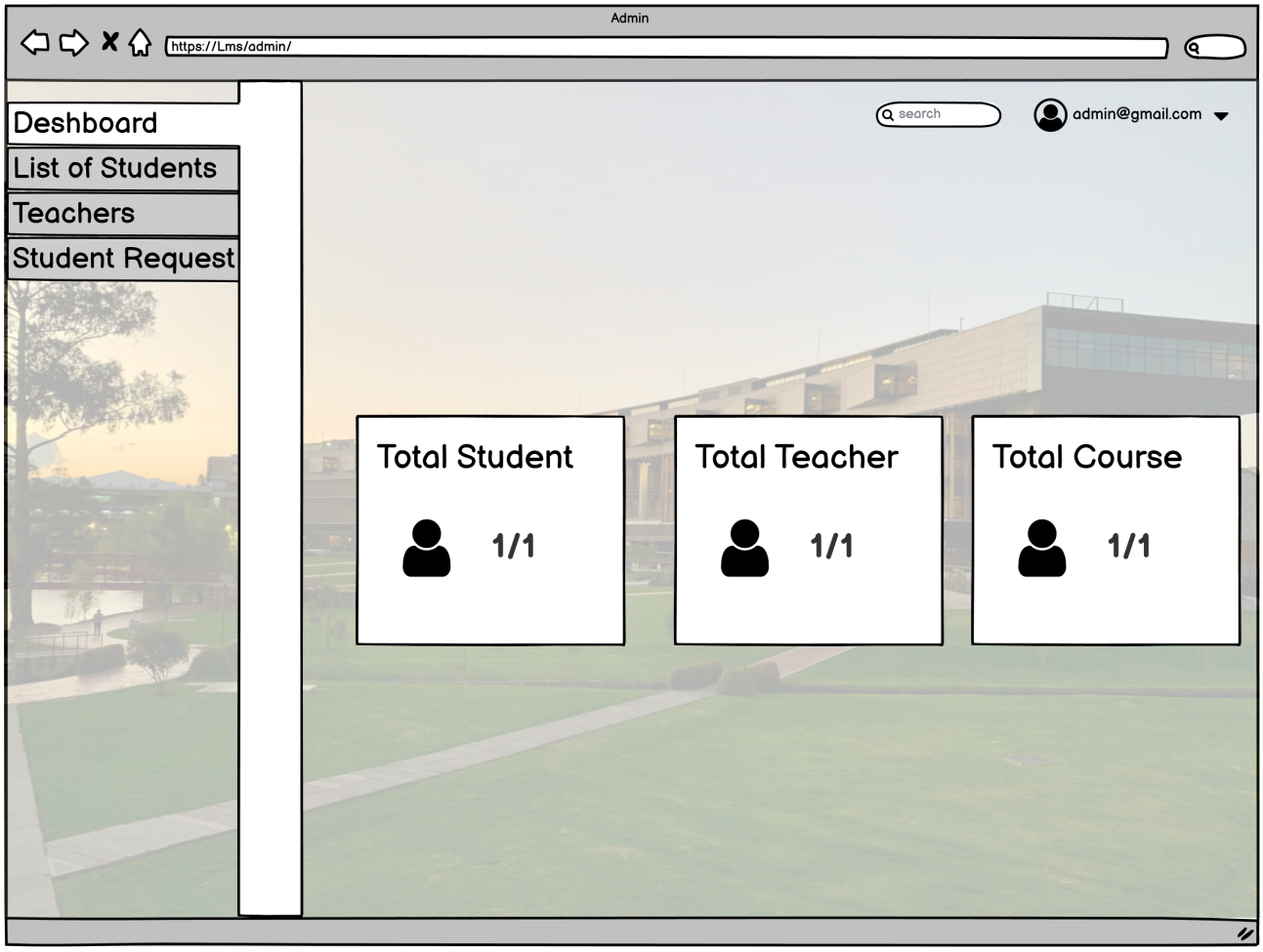
4.3 Student registration page

The registration page contains a form for manual filling in the name, e-mail, address and password. The Gender field gives the option to choose between three variants and at the Birth Date field the date can be selected. After the form is filled in the user submitting it and sending the register request to admin by clicking the Register button. Getting the register request from user the admin gives the access to the portal or decline it. If the register request is approved for the user he/she will have the ability to login to the portal.

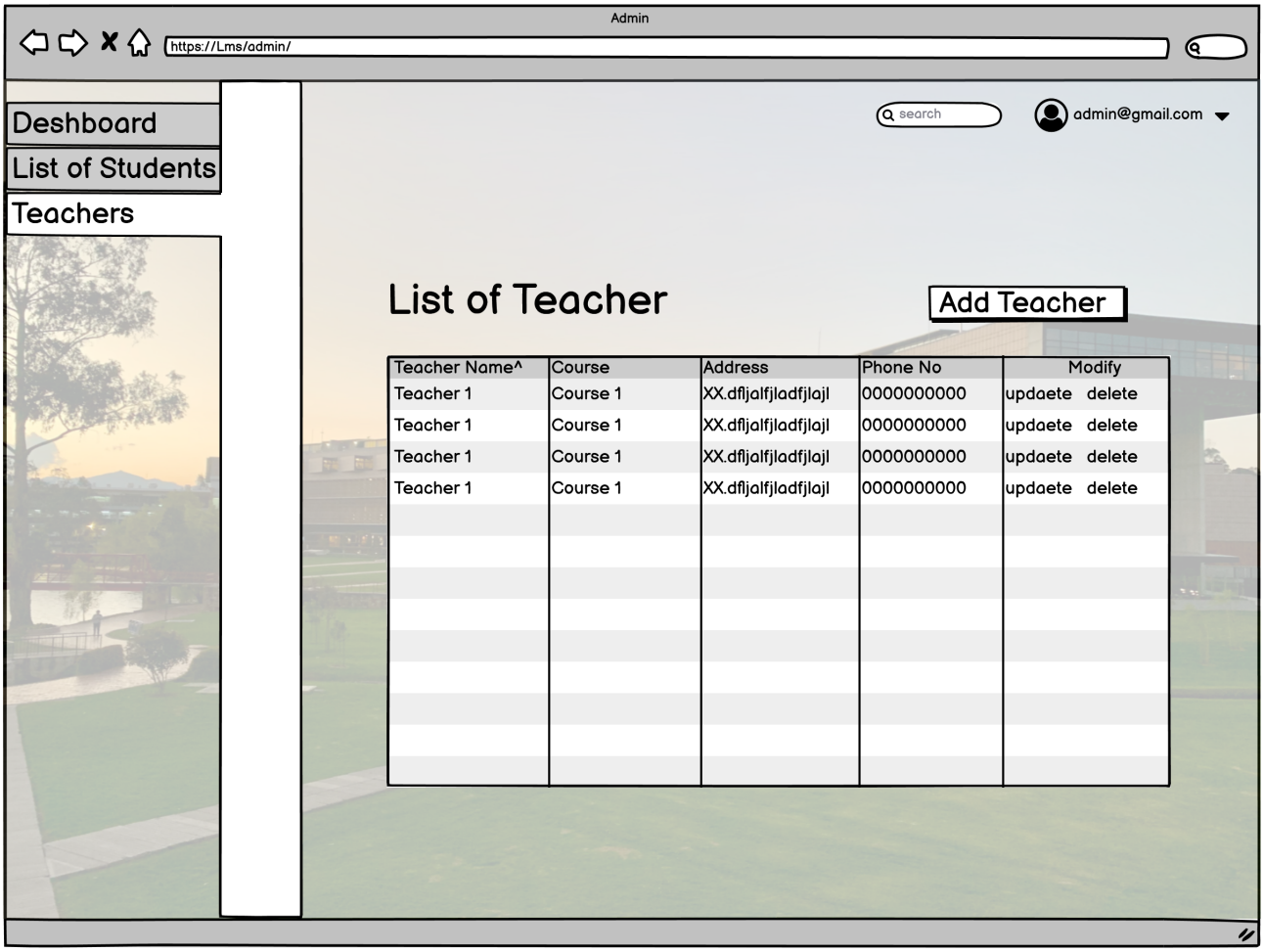


4.4 Admin Page

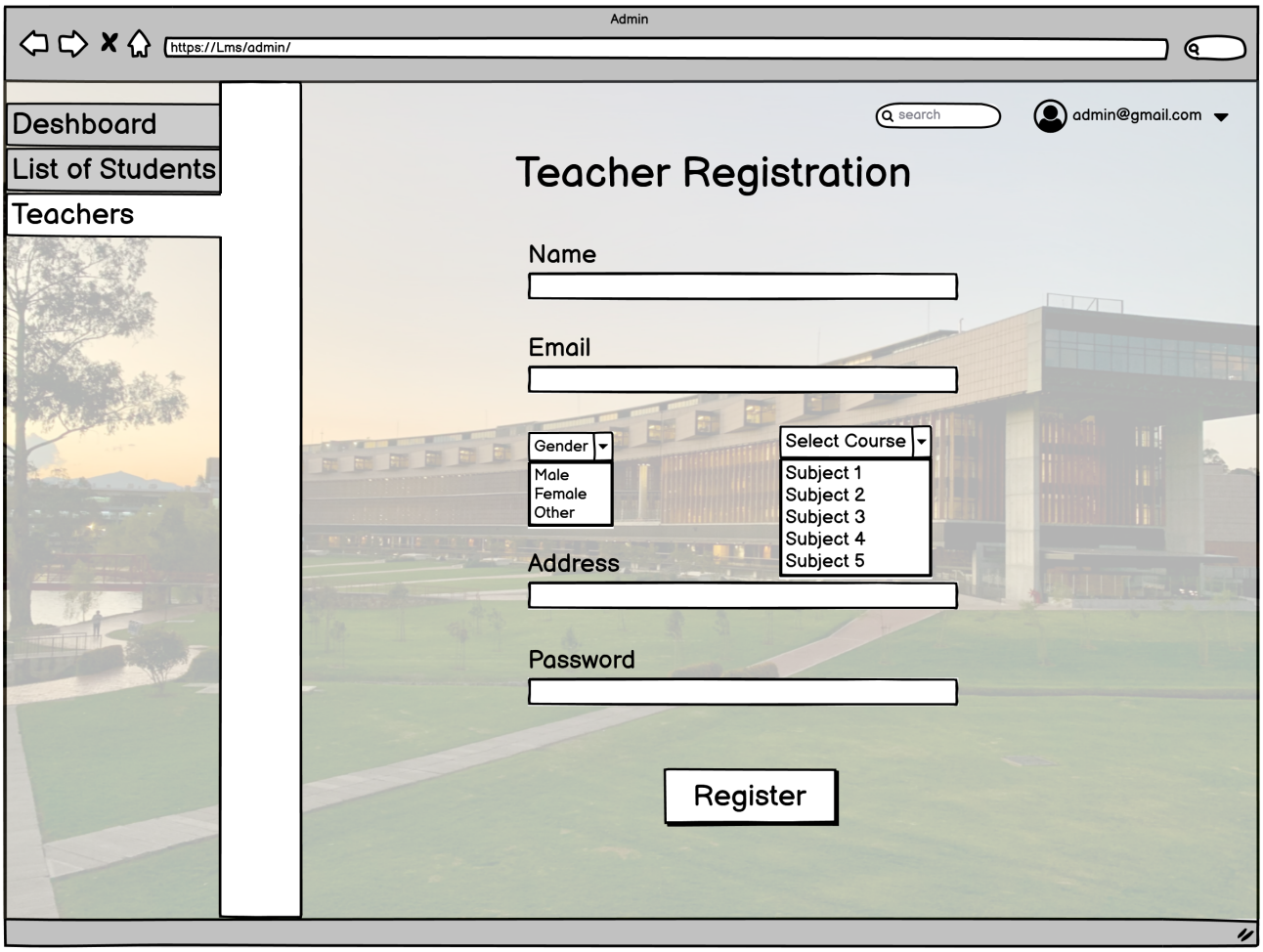
The admin page is divided into four sections: "dashboard", "List of Students", "Teachers" and "Student Request". The main area of the dashboard section contains the total number of users and courses details. The all requests form registered students at the system are located at the "Student Request" section and admin can give the access to the portal or reject the request.



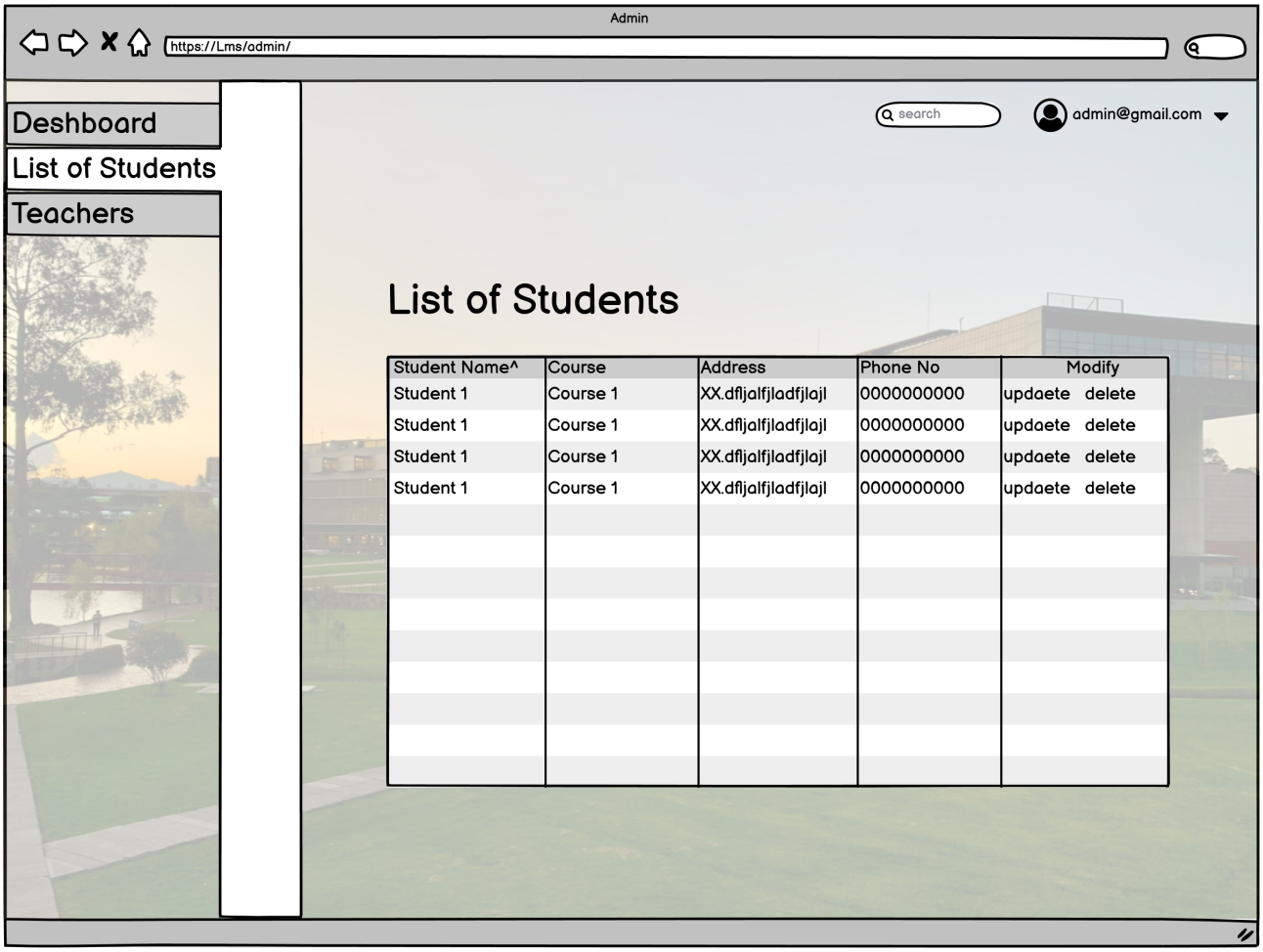
The admin page allows to register new teachers going to "Teachers" section. It contains a table with all the information about each teacher and also there is a "Modify" column where admin can update or delete data. Clicking on the button "update" admin updates the user details, or the button "delete" admin deletes the user from the database, revoking their possibility to login and therefore to access the courses. Clicking to the "add teacher" button the admin is going to registration page for adding a new teacher to the system.



The Teacher registration page contains a form for manual filling in the name, e-mail, address and password. The Gender field gives the option to choose between three variants and at the Course field the subject can be selected from the database. After the form is filled in the admin submits it by clicking the Register button.

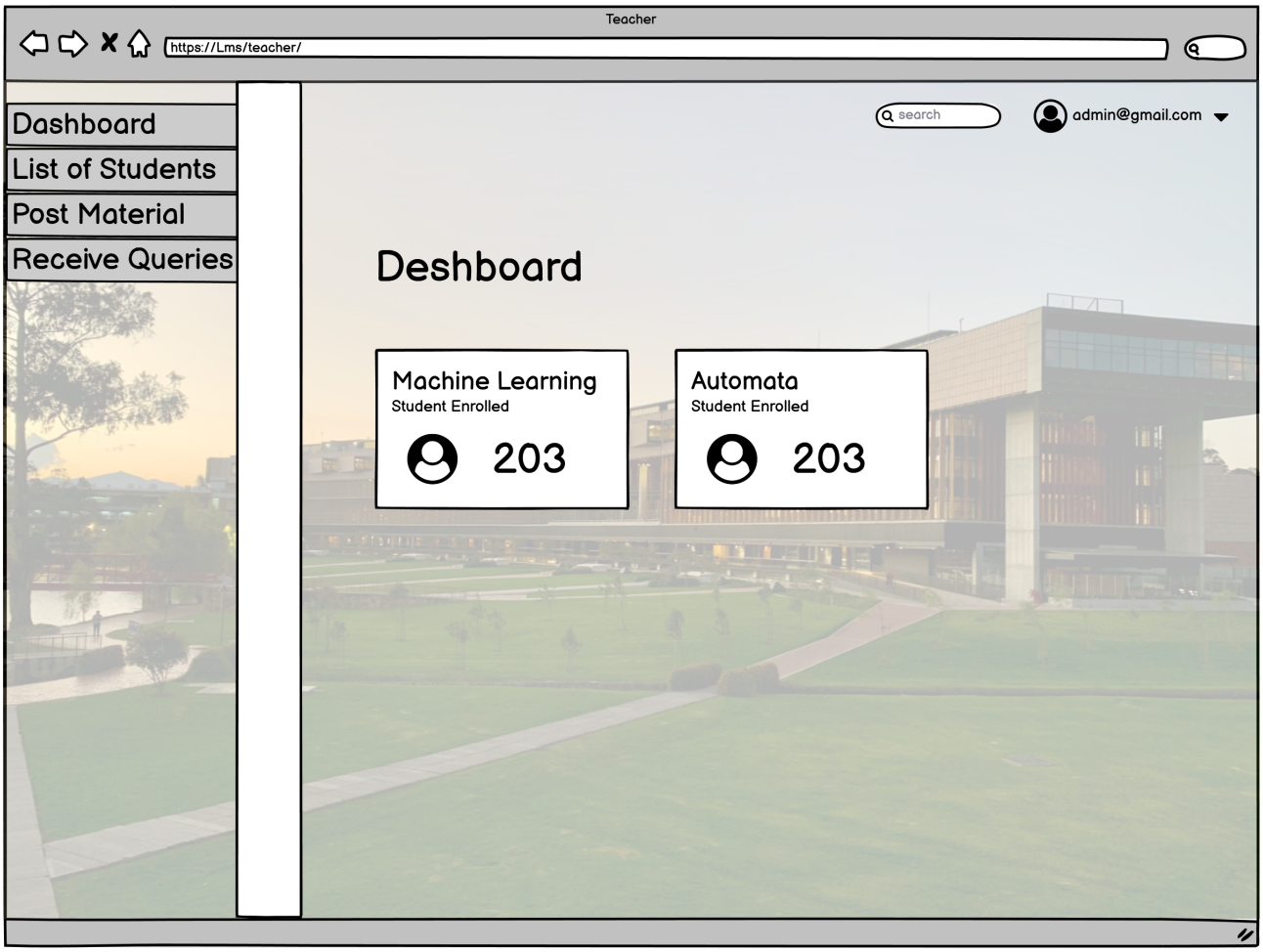


The "List of Students" section contains a table with all the information about each student and also there is a "Modify" column where admin can update or delete data. Clicking on the button "update" admin updates the user details, or the button "delete" admin deletes the user from the database, revoking their possibility to login and therefore to access the courses.

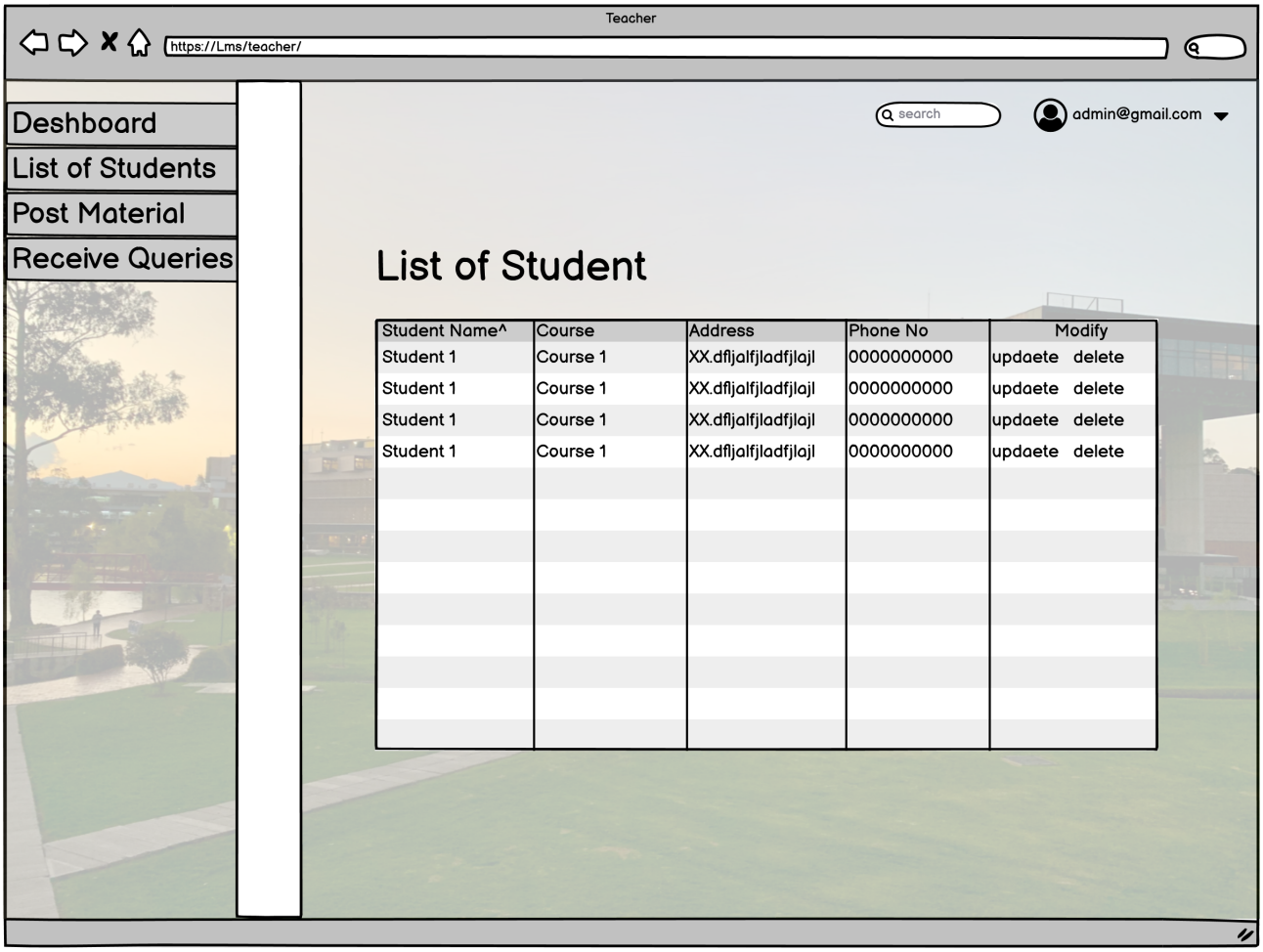


4.5 Teacher Page

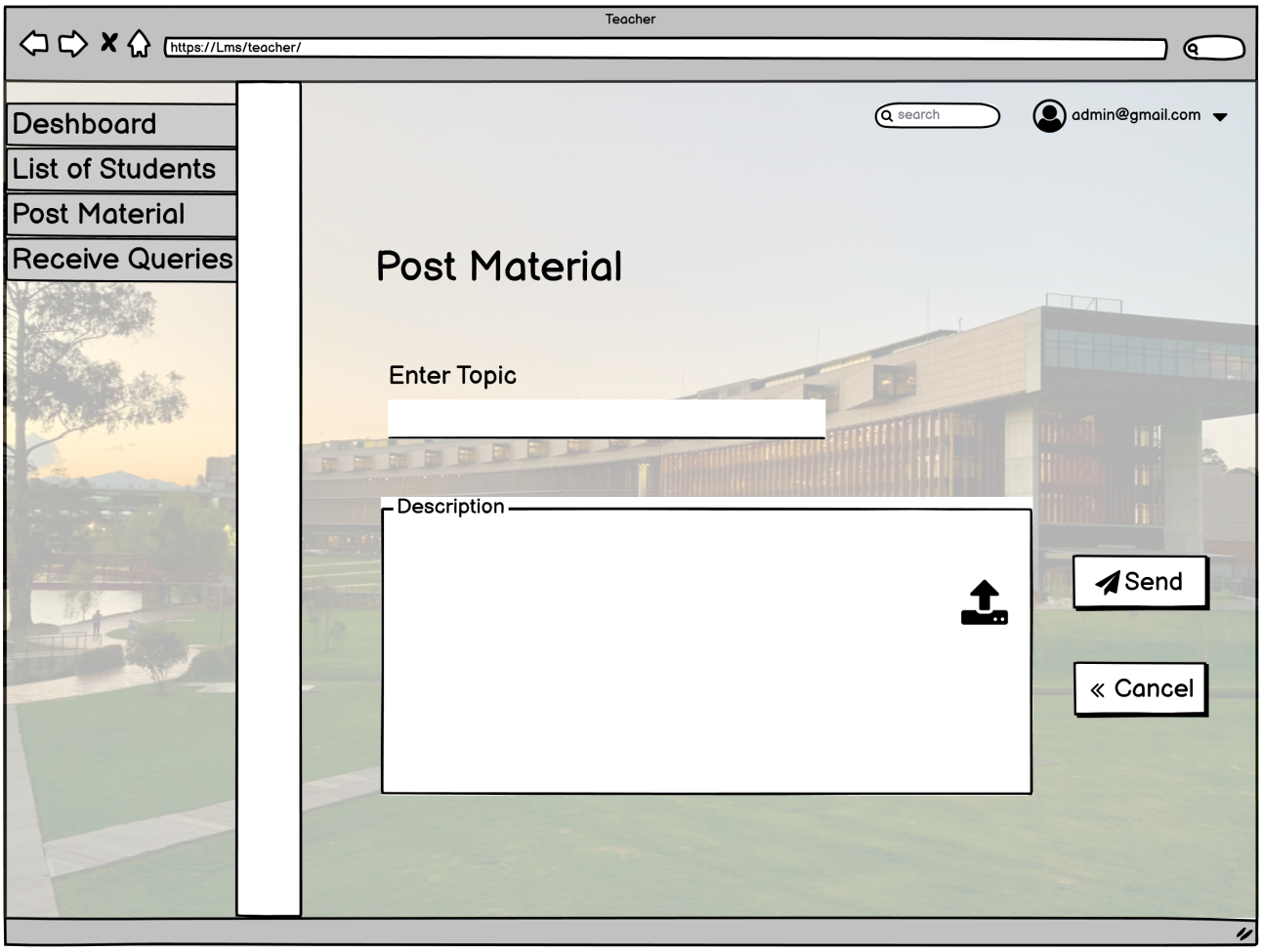
The Teacher page is divided into four sections: "dashboard", "List of Students", "Post Materials" and "Receive Queries". The main area of the dashboard section contains the total number of students enrolled for each course details.



The "List of Students" section contains a table with all the information about each student and also there is a "Modify" column where teacher can update or delete data. Clicking on the button "update" teacher updates the user details, or the button "delete" teacher deletes the user from the database, revoking their possibility to login and therefore to access the courses.

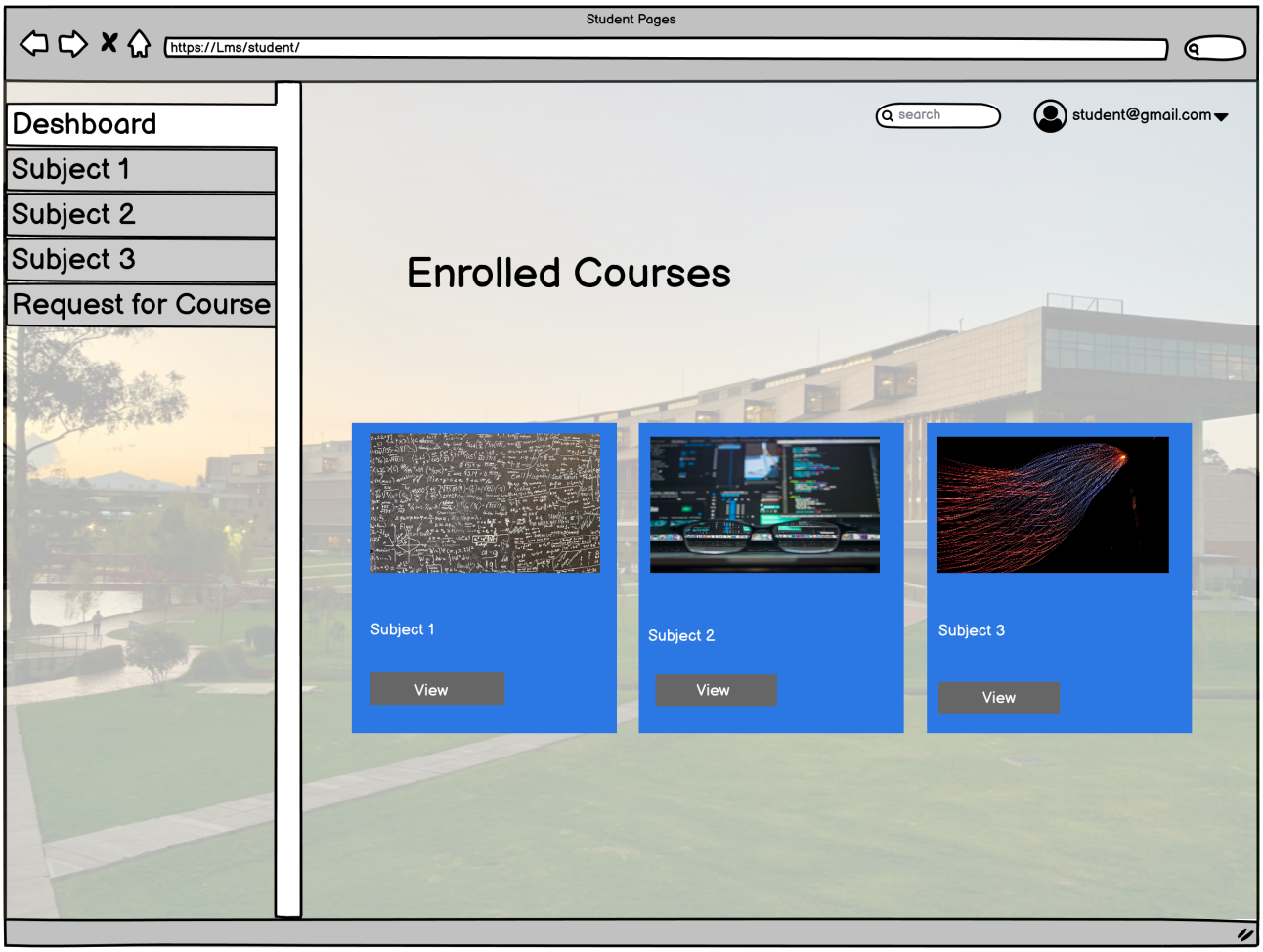


The Post Material page contains a form for manual filling in the topic and description with uploading the necessary material. After the form is filled in the teacher posts it by clicking the "Send" button.

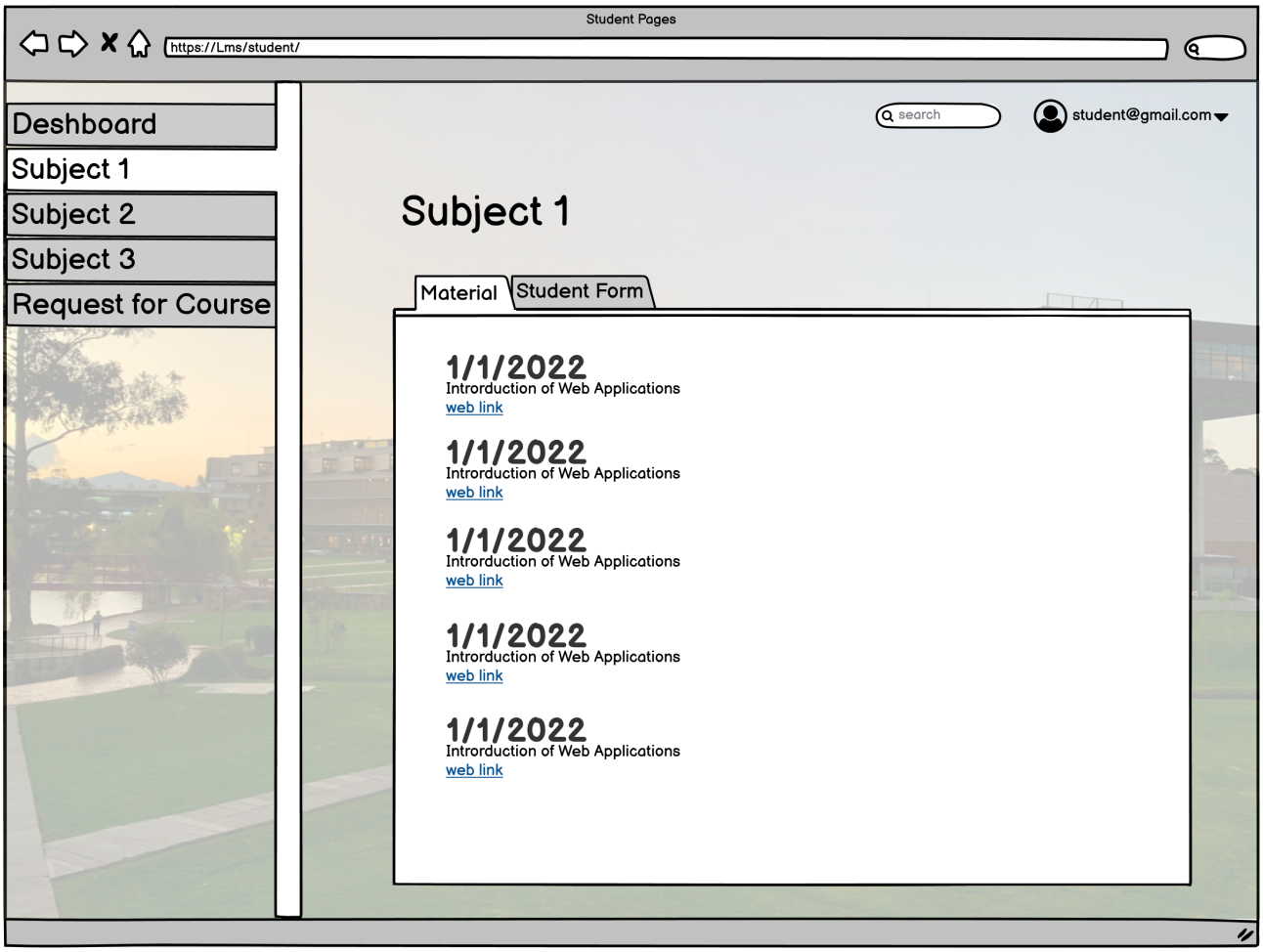


4.6 Students Page

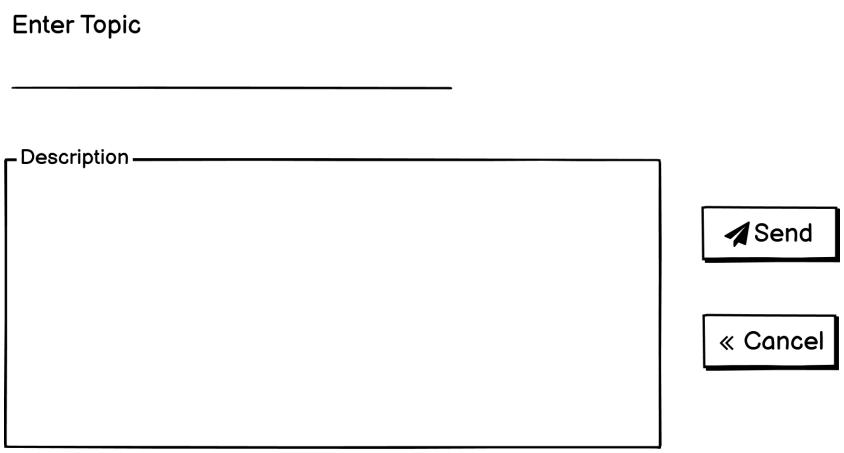
The students page is divided into three sections: "dashboard", "Subjects" and "Request for Course". The main area of the dashboard section contains the total number of enrolled courses details. By clicking to the "view" button under each course the details and materials about the course is shown.



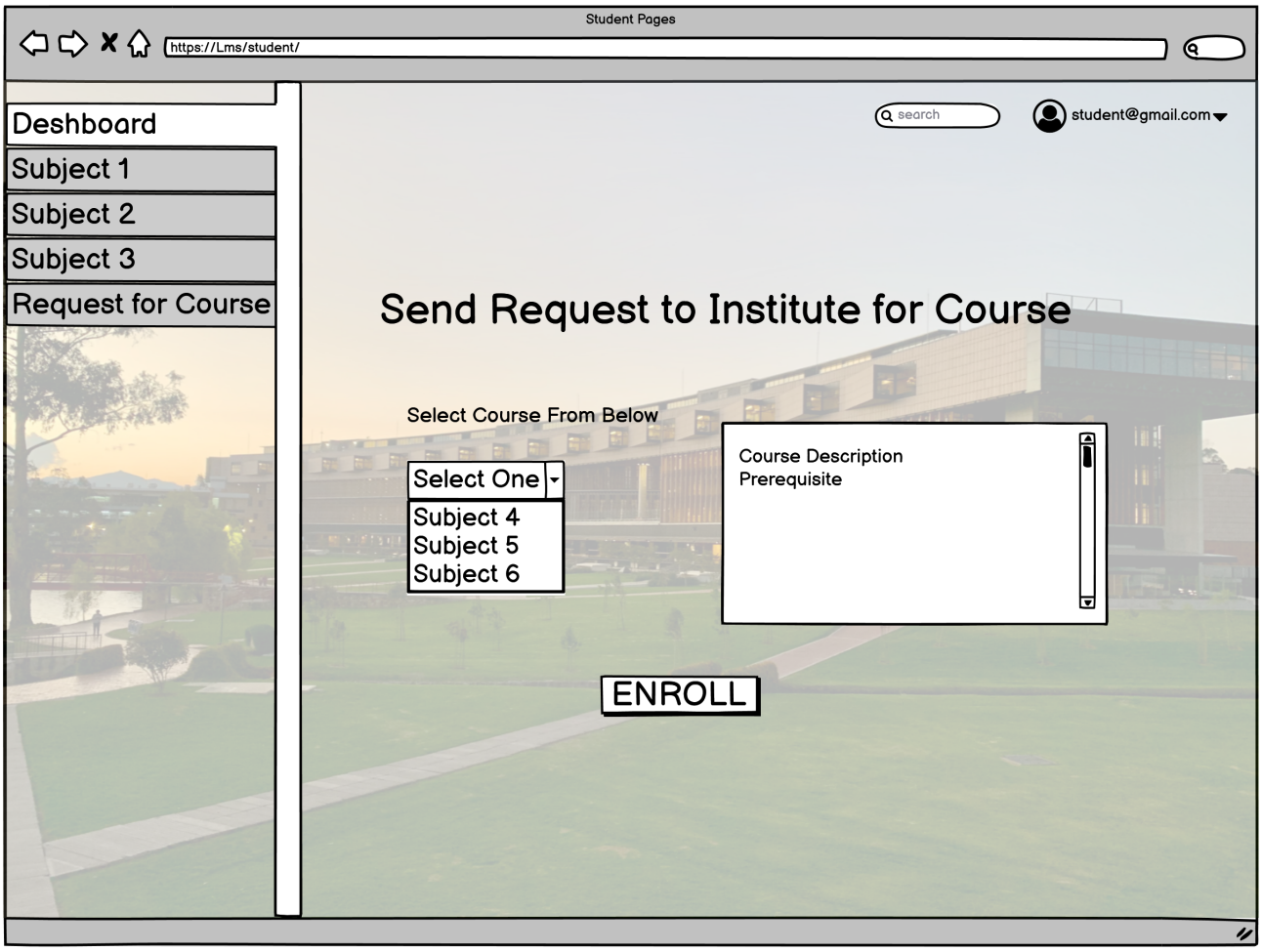
On the subjects page there are the material folder introducing all the information about particular course with dates and necessary links.



Clicking to the "Student Form" folder there is a form for filling in manually by the student. The topic and the description of the message should be written and send by clicking the "send" button.

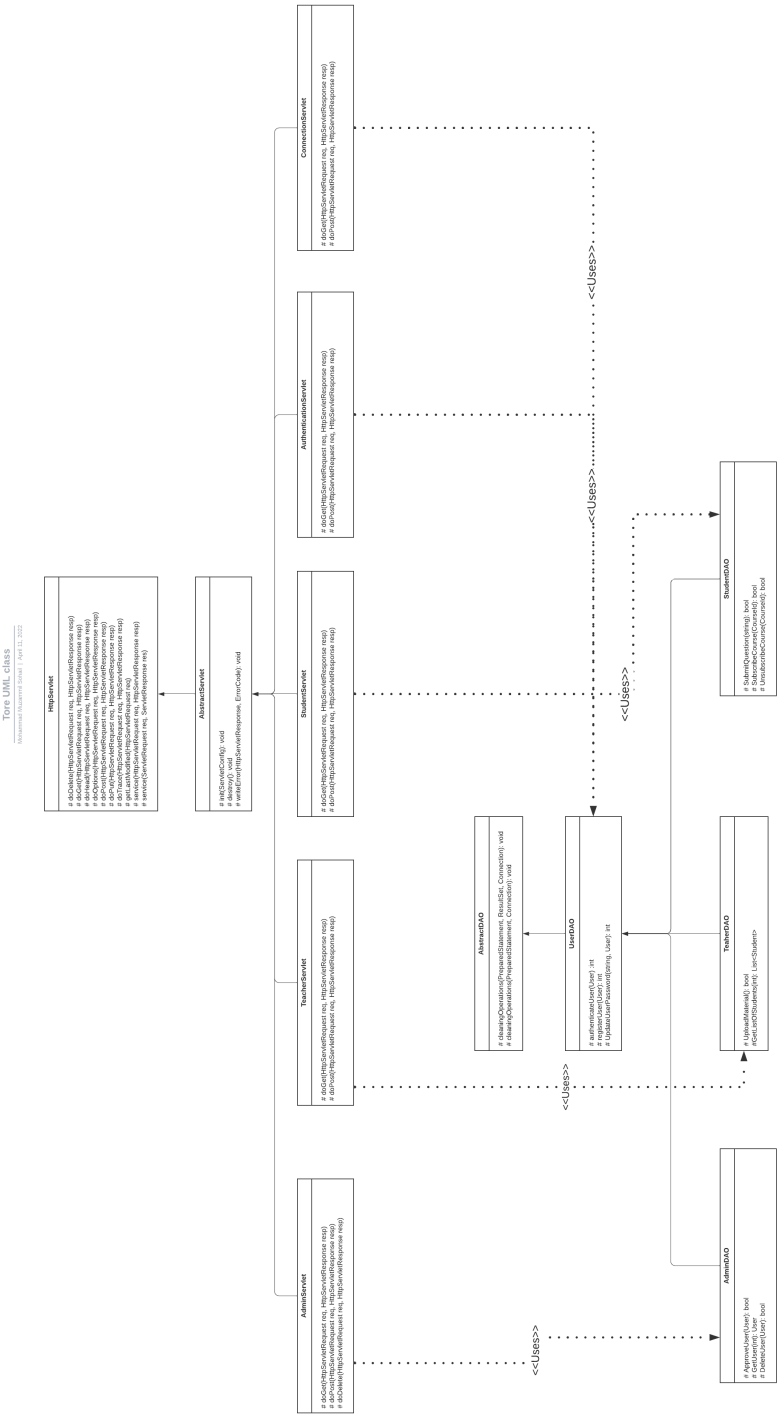


The Request for course part is for sending the request in order to enroll to the course. There are the "select" button for choosing the course the student want to be enrolled in and the chosen course description prerequisite.



5 Business Logic Layer

5.1 Class Diagram

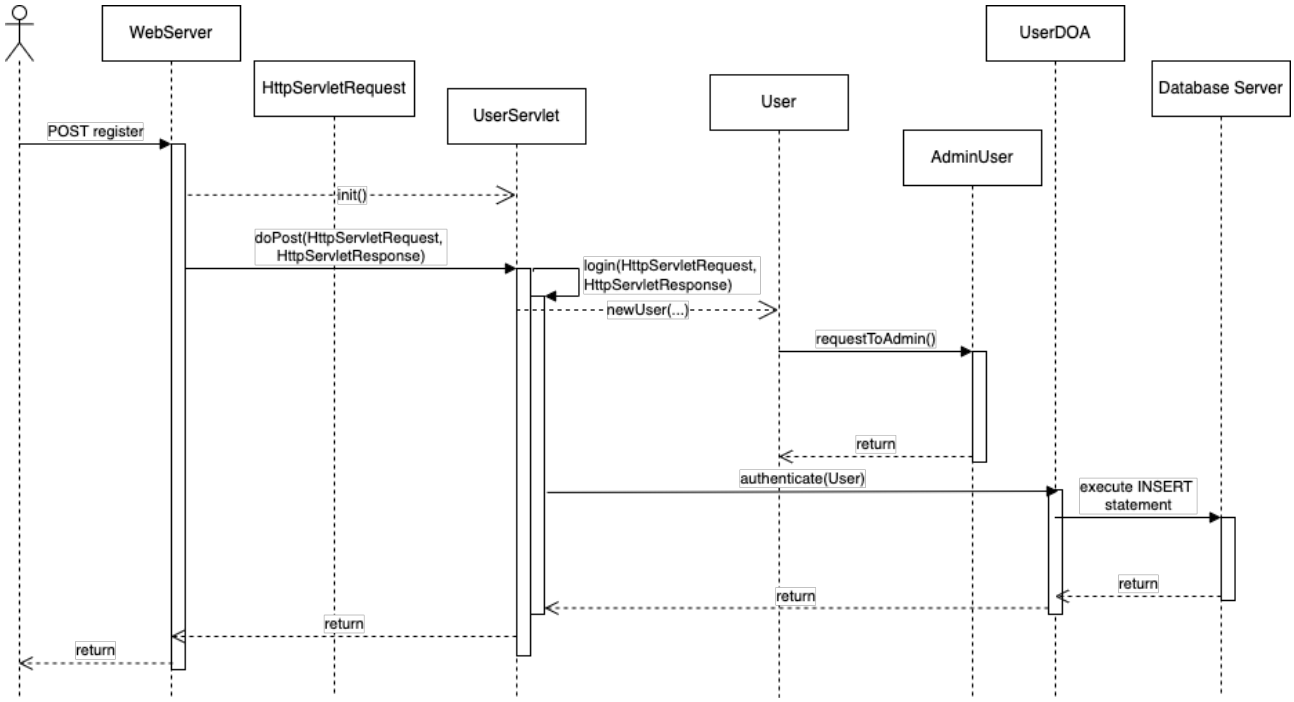


The class diagram contains (some of) the classes used to handle functionalities of the Admin, Teacher and Student in our system. AdminServlet, TecherServlet and StudentServlet are subclasses of the Java's HttpServlet

class. They are used to request functionalities accessible by the repective user. The Connection Servlet is used to make connections to the database and consequently exchange data from it. The Authentication Servlet is used to authenticate each of the respective users.

The AdminDAO, TeacherDAO and StudentDAO are subclasses of UserDAO because some of the functionalities are common in each of the users. Theses DAO classes are used to obtain access to the desired type of user in the system.

5.2 Sequence Diagram



The sequence diagram shows how the registration of a new user. The project uses the MVC pattern in the development. User, AdminUser, UserDOA, and DatabaseServer are the models. WebServer and HttpServletRequest interact as views, while UserServlet is the controller. The sequence starts with a user that sends a POST request for registration to the WebServer. A UserServlet is then initialised and executes methods for doPost() and login(). The user makes a call to the User model with newUser() and the User model asks for a request to the model AdminUser to approve the new user. After the approval of the new user, the UserServlet sends an authenticate(User) to UserDOA for the model to INSERT the new user data into the DatabaseServer model.

5.3 REST API Summary

The list of the rest API for this homework are still "in progress" because we have to decide according to the frontend where is better to use a REST call or not. Anyway, there are some resources that are just available with REST that are reported in the next table.

The rest API is studied to experiment with multiple different approaches to the REST paradigm. More in detail, endpoints associated with rides and devices are developed in a more traditional way, with the entirety of the information needed to satisfy the request directly in the URI. Other endpoints require additional information (such as the type of operation required) as additional parameters in the request.

Part of the URIs are filtered through different filters. S indicates that only users with the role of student can

access to the endpoint, T indicates that only T can access the endpoint, A that only administrators can request the specified URI to the server.

URI	Method	Description	Filter
admin/	POST	Go to the administrator section. Only admin can see access the infomation	A
admin/dashboard/	POST	Go to the administrator dashboard. Only admin login.	A
login/	POST	Go to login section for teacher and students only	S and T
register/	POST	Go to registration section for teacher and students only	S and T
home/	POST	Go to homepage	S and T
about-us/	POST	Go to about us	S and T
contact-us/	POST	Go to contact us	S and T
(user)/dashboard/	GET	Go to the administrator dashboard. Student and teacher can see access the infomation.	S and T
(user)/search/(course-name)	GET	Search for the course	S and T
(user)/(course-id)	GET	Go to the course for information	S and T
(user)/(course-id)/enroll	POST	Enroll in a course	S
(user)/(course-id)/unenroll	POST	Un-enroll in a course	S
teacher/(course-id)/post	GET	Post material for a course	T
student/(teacher-id)/message	POST	Message a teacher	T

Table 2: Describe in this table your REST API

5.4 REST Error Codes

The list of the rest API for this homework are still "in progress" because we have to decide according to the front-end where is better to use a REST call or not. The list of errors defined in the application. Application specific errors have the application error which follows a progressive numeration starting from -100. METHOD NOT ALLOWED errors are identified with the error code -500. Internal errors, which correspond to crashes, servlet exceptions, or problems with the input/output streams are identified with the Error Code -999.

Error Code	HTTP Status Code	Description
- 100	WRONG_FORMAT	Wrong format of the request
- 102	EMPTY_INPUT_FIELDS	One or more input fields are empty
- 105	WRONG_CREDENTIALS	Submitted credentials are wrong
- 109	UNRECOGNIZED_ROLE	Unrecognized role of a user
- 113	WRONG_REST_FORMAT	Wrong rest request format
- 116	USER_NOT_FOUND	User not found
- 120	BADLY_FORMATTED_JSON	The input json is in the wrong format
- 200	OPERATION_UNKNOWN	User Operation unknown
- 600	METHOD_NOT_ALLOWED	The function/method is not allowed
- 998	SQL_ERROR	SQL/or Database Error
- 999	INTERNAL_ERROR	Internal Error

- 500	SERVER_ERROR	Connection or Server Error

Table 3: Describe in this table your REST API Errors

5.5 REST API Details

In our project, we have 10 different resource types that our web application handles during its functioning. Following are resources.

Homepage content

The homepage contains an aggregation of different pieces of information. In particular, the Homepage has login and register pages, which allow users to log in themselves and register as well.

- URL: `Homepage/overview`
- Method: `GET`
- URL Parameters: `No Parameters are required in the URL`
- Data Parameters: `No Data is passed when requiring this method`
- Success Response: `Code: 200`

Login Page

- URL: `Homepage/LMS/register`
- Method: `POST`
- URL Parameters:
  - Name: {string}
  - email = {string}
  - birth_date = {string}
  - gender = {string}
  - address = {string}
  - password = {string}
- Success Response: `Code: 200`
  Content: The user is redirected to the login page.
- Error Responses:
  - Code: 400 BAD_REQUEST
    Content: `{\error": {\code": -102, \message" : \One or more input fields are empty."}}`
    When: some of the parameters are missing or are empty string.
  - Code: 409 CONFLICT
    Content: `{\error": {\code": -107, \message" : \Email Already Used."}}`
    When: The user is trying to get register with the same email address that is already present in database.

  - Code: 400 $BAD_REQUEST$
    $Content$: {\code": -108, \message" : \Password is Weak."}}
    $When$: $EnteredPasswordisweak$.
- Code: 500 $INTERNAL_SERVER_ERROR$
  $content$: {\code": -999, \message" : \Internal Error."}}
  $When$: $ifthereisaSQLExceptionoraNamingException, thiserrorisreturned$.

Admin Page

It contains the full details of Student and Teacher information in the database, there is a page that contains information about students and their enrolled courses, and there is also a separate page for Teacher's' information and their teaching courses.

1.View Students:

Admin can View all registered student.

- **URL:** `/admin/student-list`
- **Method:** `GET`
- **Data Parameters:** `No Data is passed when requiring this method.`
- **Success Response:** `Code: 200`
  Upon success, the servlet returns a JSON object with the required information.

  Content:
  {
  "data":
  [{
  "Name": "XYZ"
  "email":"xyz@gmail.com"
  "address": \Padova, PD 35010"
  "Courses List": \Machine Learning, Foundation of Database, Search Engine, Web Application"
  }]
  }
- **Error Responses:** `Code: 500 INTERNAL_SERVER_ERROR`
  Content: {\error": {\code": -999, \message" : \Internal Error."}}
  When: if there is a SQLException or a NamingException, this error is returned.

2. View Courses:

Admin can View all courses.

- **URL:** `/admin/courses`
- **Method:** `GET`
- **Data Parameters:** `No Data is passed when requiring this method.`

- **Success Response:** `Code: 200`
  Upon success, the servlet returns a JSON object with the required information.

  Content:
  {
  "data":
  [{
  `"Name": "XYZ"`
  `"email":"xyz@gmail.com"`
  `"address": \Padova, PD 35010"`
  "Courses List": "Machine Learning, Foundation of Database, Search Engine, Web Application"
  }]
  }

- **Error Responses:** `Code: 500 INTERNAL_SERVER_ERROR`
  Content: `{\error": {\code": -999, \message" : \Internal Error."}}`
  When: if there is a SQLException or a NamingException, this error is returned.

6 Group Members Contribution

Ahmed Saad Implemented Admin Panel and write the Presententation Layer with complete demo diagrams of every page.

Bahrami Sepide ER Diagram and designed the database.

Hansen Marit Fredrikke Implemeneted Teacher Panel and Sequence Diagram.

Rao Abdul Moeed Implemented the Logins of the application and write Main-functionalities.

Rehman Abdul Written REST API Details and checked the spelling mistakes.

Sohail Mohammad Muzammil Implemented Class Diagram and some functionalities in Student Panel

Soomro Dodo Khan Implemented Student Panel and write REST API Summary and Errors.

Sultanova Aliia Written Presentation Layer, check the Latex and SQL.

Zabalawi Iyad Written REST API Details and workflow.