

## **Student Inquiry Assistance Portal for NIIT Pvt Ltd**

### **Problem Statement:**

In the rapidly evolving landscape of online education, the process of discovering, inquiring about, and obtaining information on available courses can be cumbersome for prospective students. Current systems may lack a centralized and interactive platform that seamlessly connects students with detailed course information and provides a straightforward mechanism for submitting inquiries.

Prospective students face challenges in navigating through a plethora of online courses, obtaining comprehensive details, and engaging in a streamlined inquiry process. The absence of a dedicated system for submitting inquiries and providing feedback may lead to frustration and a less-than-optimal experience for both students and administrators.

The Online Course Inquiry and Information System aim to address these challenges by providing a robust and user-friendly platform. For prospective students, the system offers a seamless interface to create user profiles, explore courses, and submit inquiries. Detailed course information, including descriptions, instructors, duration, and pricing, is readily accessible, streamlining the decision-making process for students.

### **Users of the System:**

**Authentication and Authorization: Role based authentication.**

#### **Role:**

1. Admin
2. Student
3. OfficeStaff

Admin should have permission for CRUD operation.

#### **Role based Menu Options**

##### **Admin:**

- Register
- Login
- Post Course details
- Edit Course Details
- Delete Course Details
- View All Courses
- View All Enquiries
- Change Status of Enquiry
- View/update/delete Admission details.
- View Payment History
- Logout

##### **Student:**

- Register
- Login
- View All Courses
- Check the Status of Enquiry
- Delete Enquiry
- Admission for courses
- Make Payment for the course
- View Payment History
- Logout

**OfficeStaff**

- Register
- Login
- View All Enquiries
- Reply the enquiries
- Close the Enquiry
- View Payment History
- View Admission details
- Check the balance amount to be paid
- Logout

**Functional Requirements:****Prospective Students:**

1. Create a user profile to submit course inquiries.
2. Browse and search for available online courses.
3. View detailed information about courses, including course descriptions, instructors, duration, and pricing.
4. Submit inquiries for specific courses, including questions or requests for additional information.
5. Provide feedback and reviews on the courses and the inquiry process.
6. Student can enroll the course.
7. Payment Process: The student can pay the exam fees, and other payment in online.

**Admin:**

1. Add, update, or remove course listings, including course details and availability.
2. Monitor and respond to inquiries submitted by prospective students.
3. Send notifications and updates to students regarding their inquiries.
4. View, update and delete admission.
5. Collect and review feedback and reviews submitted by students.

**Office Staff**

1. View the student enquiries and respond to the messages
2. Close the student enquiry
3. Check the balance amount to be paid

**Non-Functional Requirements:**

1. Usability: The user interface should be intuitive and user-friendly, with responsive design for mobile and desktop users.
2. Availability: The system should be available 24/7 with minimal downtime for maintenance.
3. Logging and Auditing: Support logging and auditing of system activities for monitoring and troubleshooting.

**Application Flow:****Student side:**

The application flow for the portal begins with user registration, where prospective students create accounts by providing personal information. Upon logging in, users access the user dashboard and view the available courses with payment details. The student can enquire about the course timing, course details, syllabus doubts etc.

The student can submit maximum of 5 enquiry per day. The student has limitation of submitting the enquiry. The student can fill the course admission form and enroll in it. The student can do the payment and view all the payment history.

**Admin side:**

The administrative flow within the portal begins with administrators accessing the admin dashboard, providing a comprehensive overview of courses offered with payment details. The admin can view the list of student's profiles.

The admin can view the enquiries submitted by the students. For each enquiry change the status and reply back to the student enquiry. The admin can view, delete and edit the admissions.

The admin can view the payment history.

**OfficeStaff SideTop of FormBottom of Form:**

The office staff flow within the portal begins with staff dashboard, providing a comprehensive overview of students and enquiries. The staff will reply to the student enquiries and close the enquiry by changing the status.

The staff also view the payment history and check the balance amount to be paid by the student.

The staff will send the reminder message about the balance to be paid.

**Abstract**

The Online Course Inquiry and Information System is a dynamic web-based application designed to facilitate the exploration and inquiry process for prospective students interested in online courses.

Catering to the needs of both administrators and students, this platform offers a user-friendly interface that enables students to browse available courses, submit inquiries, and receive relevant information.

Administrators, in turn, have the capability to manage course listings, respond to inquiries, and collect valuable feedback to enhance the overall user experience.

**Mandatory Modules:****Modules of the Application:****ADMIN:**

Ø Register

Ø Login

Ø Dashboard

- Add course details
- View, edit and delete course details
- View enquiry details
- View Payment details

**STUDENT:**

Ø Register

Ø Login

Ø Dashboard

- View course details
- Add enquiry details (5 enquiries per day)
- View enquiry details
- Make Payment

**OFFICESTAFF:**

Ø Register

Ø Login

Ø Dashboard

- View payment details
- View admission details

## **Technology Stack**

Front End  
React, HTML, CSS

Back End  
Java, Spring Boot and MySQL

Authentication  
JWT for User Authentication

### **Application assumptions:**

1. The login page should be the first page rendered when the application loads.
2. Manual routing should be restricted by using Auth Guard by implementing the can Activate interface.
3. Logging out must again redirect to the login page.
4. To navigate to the admin side, you can store a user type as admin in the database with a username and password as admin.
5. Design forgot password and forgot email buttons in login page.

## **Validation**

### **Client-Side Validation:**

Implement client-side validation using HTML5 attributes and JavaScript to validate user input before making API requests.

Provide immediate feedback to users for invalid input, such as displaying error messages near the input fields.

1. Basic email validation should be performed.
2. Basic mobile number validation should be performed.
3. Basic password should be performed

### **Server-Side Validation:**

Implement server-side validation in the controllers to ensure data integrity.

Validate user input and API responses to prevent unexpected or malicious data from affecting the application.

Return appropriate validation error messages to the user interface for any validation failures.

### **Exception Handling**

Implement exception handling mechanisms in the controllers to gracefully handle errors and exceptions.

Define custom exception classes for different error scenarios, such as API communication errors or database errors.

Log exceptions for debugging purposes while presenting user-friendly error messages to users.

Record all the exceptions and errors handled store in separate table “**ErrorLogs**”.

### **Error Pages:**

Create custom error pages for different HTTP status codes (e.g., **404** Not Found, **500** Internal Server Error) to provide a consistent and user-friendly error experience.

Ensure that error pages contain helpful information and guidance for users.

Thus, create a reliable and user-friendly web application that not only meets user expectations but also provides a robust and secure experience, even when faced with unexpected situations.

## Project Tasks:

### 1. API Endpoints:

|                        |                                      |
|------------------------|--------------------------------------|
| auth-controller ^      |                                      |
| POST                   | /auth/register v                     |
| POST                   | /auth/login v                        |
| enquiry-controller ^   |                                      |
| POST                   | /api/student/makepayment v           |
| POST                   | /api/student/enquiry v               |
| GET                    | /api/student/userid/{userId} v       |
| GET                    | /api/student/courses v               |
| DELETE                 | /api/student/{id} v                  |
| admission-controller ^ |                                      |
| POST                   | /student/admissions/addadmission v   |
| GET                    | /student/admissions/{admissionId} v  |
| course-controller ^    |                                      |
| PUT                    | /api/updateadmission/{admissionId} v |
| PUT                    | /api/course/{id} v                   |
| DELETE                 | /api/course/{id} v                   |
| GET                    | /api/course v                        |
| POST                   | /api/course v                        |
| GET                    | /api/student/course v                |
| GET                    | /api/payments/{id} v                 |
| GET                    | /api/enquiry v                       |
| GET                    | /api/admissions v                    |
| DELETE                 | /api/admission/{admissionId} v       |

## OfficeStaff

/admin/course/getPayment/\*  
/admin/course/getalladmissions

## Backend

### Primary Model

```
class User
{
    String email
    Long userId
    String password
    String username
    String mobileNumber
    String userRole
}

class Course
{
    Integer CourseID
    String CourseName
    String Description
    String Duration
    Integer FeesAmount

    @ManyToMany
    List<Student> students

    @OneToMany
    List<Enquiry> enquiries
}

class Student
{
    studentId
    studentName
    studentEmailId
    @ManyToMany
    List<Course> courses

    @OneToMany
    List<Enquiry> enquiries
    @OneToOne
    User user
}
```

```

class Enquiry
{
    Integer enquiryID
    DateTime enquiryDate
    String title
    String description
    String enquiryType

    @ManyToOne
    Student student
}

class Payment
{
    Long paymentId;
    String status; //paid=true/false
    Double amountPaid;
    Date paymentDate;
    String modeOfPayment;

    @ManyToOne
    Student student;
}

```

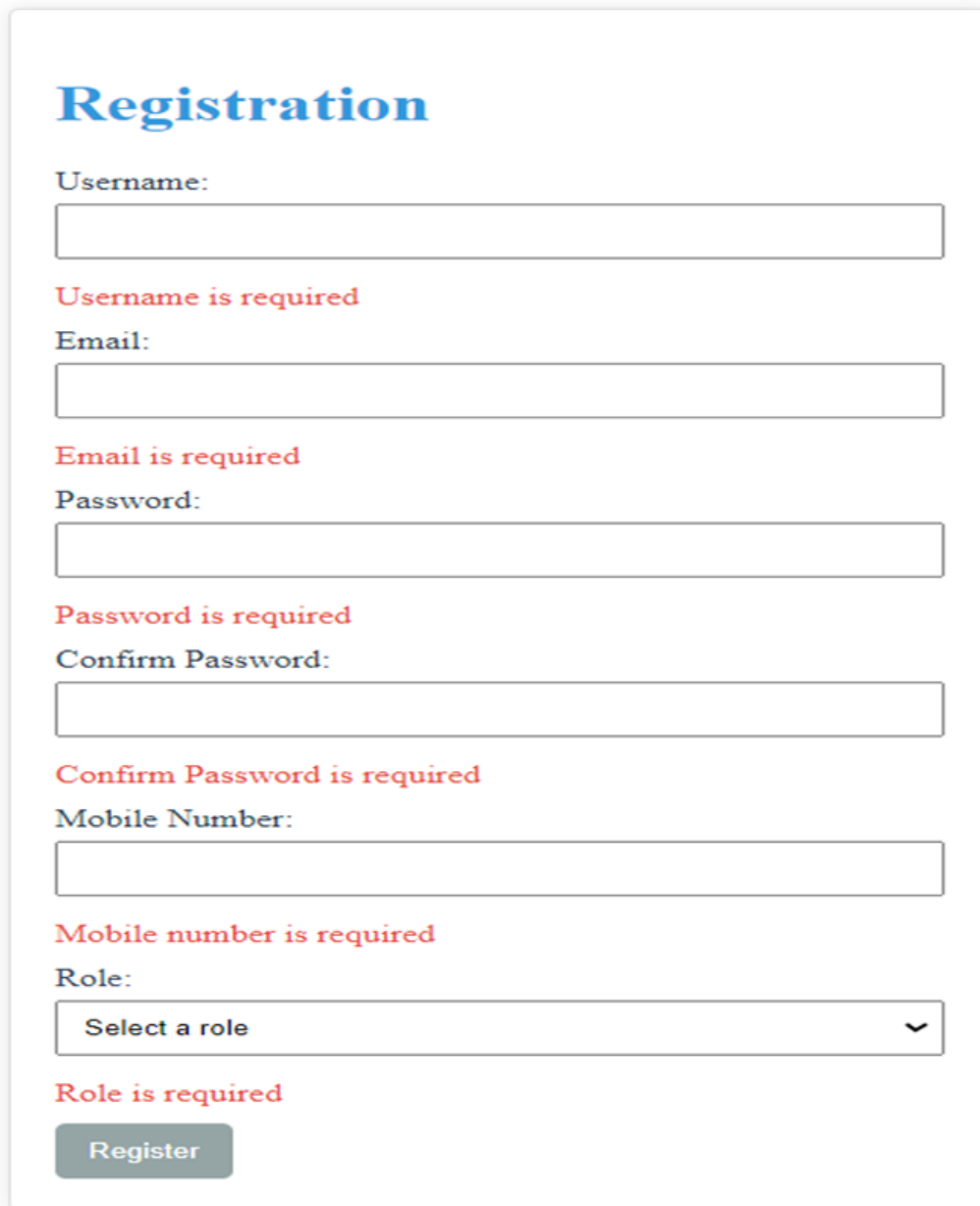
### Frontend Sample Screenshots:

#### Home page



Registration and login forms for admin, student and office staff.

Below Screenshot display a Registration form with validation for each field.



The screenshot shows a registration form titled "Registration" in blue. It contains several input fields, each followed by a red validation message: "Username is required", "Email is required", "Password is required", "Confirm Password is required", and "Mobile number is required". The "Role" field is a dropdown menu with the text "Select a role" and a downward arrow, followed by the red message "Role is required". At the bottom is a grey "Register" button.

**Registration**

Username:  
  
Username is required

Email:  
  
Email is required

Password:  
  
Password is required

Confirm Password:  
  
Confirm Password is required

Mobile Number:  
  
Mobile number is required

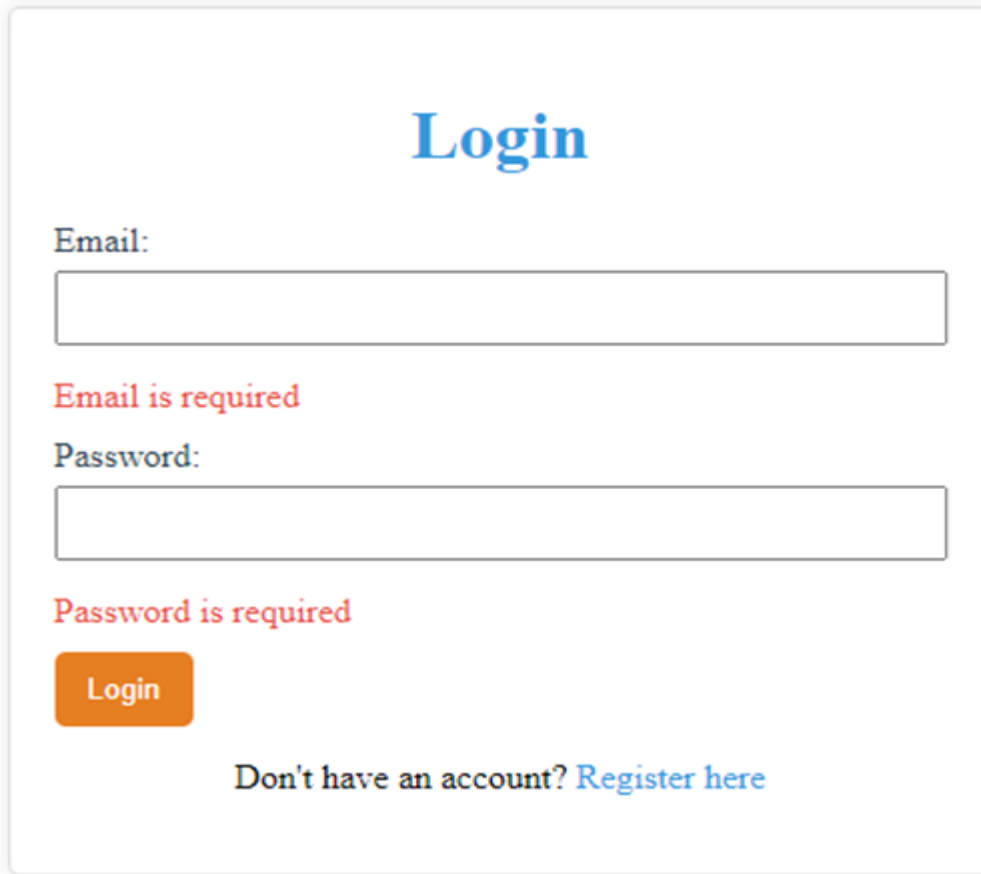
Role:  
  
Role is required

Login form:

On clicking "Login button" with empty field , validated error message is displayed as shown in the



below screenshots



A login form with a white background and a light gray border. At the top, the word "Login" is written in a large, blue, serif font. Below it, the label "Email:" is in a blue, serif font, followed by a white rectangular input field. Under the input field, the text "Email is required" is displayed in a red, serif font. Below this, the label "Password:" is in a blue, serif font, followed by another white rectangular input field. Under the second input field, the text "Password is required" is displayed in a red, serif font. At the bottom left, there is an orange rectangular button with the word "Login" in white, sans-serif font. At the bottom center, the text "Don't have an account? Register here" is displayed, with "Register here" in a blue, serif font and underlined.

On Successful login , dashboard screen is displayed as per the role of login account

### Admin Side Dashboard

Admin can add , view the course created and able to view the payment details and student enquiry details.



An admin dashboard with an orange header bar. The header bar contains the text "Student Enquiry System" on the left and "Home Admin Logout" on the right. Below the header bar, there are four colored rectangular buttons arranged in a 2x2 grid. The top-left button is light blue and contains the text "Add Course". The top-right button is light green and contains the text "View Courses". The bottom-left button is yellow and contains the text "View Enquiries". The bottom-right button is light purple and contains the text "View Payment".

Admin can add the course details:

Add Course Details

Course Name:

Enter Course Name

\*Course Name is required.

Description:

Enter Description

\*Description is required.

Duration:

Enter Duration

\*Duration is required.

Fees Amount:

0

Add Course

Admin can view all the courses and have access to edit/delete the courses:

| Course Name                  | Description   | Duration | Fees Amount | Actions               |
|------------------------------|---|----------|-------------|-----------------------|
| Introduction to Programming  | A beginner-friendly course on programming concepts.                 | 8 weeks  | 10000       | <div>EditDelete</div> |
| Web Development Fundamentals | Learn the basics of web development and design.                     | 10 weeks | 15000       | <div>EditDelete</div> |
| Data Science Essentials      | Explore fundamental concepts of data science and analytics.         | 12 weeks | 20000       | <div>EditDelete</div> |
| Graphic Design Essentials    | Explore the principles of graphic design and visual communication.  | 5 weeks  | 8000        | <div>EditDelete</div> |
| Digital Marketing Strategies | Learn the fundamentals of digital marketing and online advertising. | 10 weeks | 12500       | <div>EditDelete</div> |

Admin can view the list of student enquiries:

| Course Name             | Description   | Email          | Enquiry Type |
|-------------------------|---|----------------|--------------|
| Data Science Essentials | I need full details of this course, kindly contact to my email. | john@gmail.com | general      |

Admin can view, update and delete the list of student admissions:

| Admission ID | User ID | Course Name                  | Status              | Action            |
|--------------|---------|------------------------------|---------------------|-------------------|
| 1            | 2       | Introduction to Programming  | <div>Pending</div>  | <div>Delete</div> |
| 2            | 2       | Data Science Essentials      | <div>Accepted</div> | <div>Delete</div> |
| 3            | 4       | Digital Marketing Strategies | <div>Pending</div>  | <div>Delete</div> |

Admin can view the payment history:

## Payment History

| Payment ID | Student ID | Admission ID | Course ID | Status | Total Amount | Payment Date | Mode of Payment |
|------------|------------|--------------|-----------|--------|--------------|--------------|-----------------|
| 1          | 2          | 2            | 3         | Paid   | 20000        | 2022-01-01   | Card            |

## STUDENT SIDE:

### Student Dashboard:

Student Enquiry System

HomeStudentLogout

View Courses

Add Enquiry

View Enquiries

The student can see the list of courses and can apply admission:

| Course Name                  | Description   | Duration | Fees Amount | Actions                         |
|------------------------------|---|----------|-------------|---------------------------------|
| Introduction to Programming  | A beginner-friendly course on programming concepts.                 | 8 weeks  | 10000       | <a href="#">Apply Admission</a> |
| Web Development Fundamentals | Learn the basics of web development and design.                     | 10 weeks | 15000       | <a href="#">Apply Admission</a> |
| Data Science Essentials      | Explore fundamental concepts of data science and analytics.         | 12 weeks | 20000       | <a href="#">Apply Admission</a> |
| Graphic Design Essentials    | Explore the principles of graphic design and visual communication.  | 5 weeks  | 8000        | <a href="#">Apply Admission</a> |
| Digital Marketing Strategies | Learn the fundamentals of digital marketing and online advertising. | 10 weeks | 12500       | <a href="#">Apply Admission</a> |

Student can apply for enquiry ,as shown in below screenshot.

### Add Enquiry Details

Enquiry Date:

dd - mm - yyyy

\*Enquiry Date is required

Course Name:

\*Course Name is required

Description:

\*Description is required

Email ID:

\*Email ID is required

Enquiry Type:

\*Enquiry Type is required

Submit

The student can do the payment by entering details in admission form:

Admission Form

Student ID:

2

Course Name:

Data Science Essentials

Total Amount:

20000

Mode of Payment:

Select a mode of payment

Payment Date:

dd - mm - yyyy

Submit Payment

Student Id will be prepopulated in the above form.

The student can view his admission details:

| Admission ID | Course Name                 | Status   |
|--------------|-----------------------------|----------|
| 1            | Introduction to Programming | Pending  |
| 2            | Data Science Essentials     | Accepted |

Office Staff:

The office staff can view admissions and payment history of students

| Admission ID | Student ID | Course Name                 | Status   |
|--------------|------------|-----------------------------|----------|
| 1            | 2          | Introduction to Programming | Pending  |
| 2            | 2          | Data Science Essentials     | Accepted |

**Note:**

- You should use NotFound(), NoContent(), BadRequest(), CreatedAtAction() to handle the HTTP status code as return values for the Controller methods as mentioned.
- Don't delete any files in a project environment.

**Other Important Key factors in the application:**

- Should use Custom Exceptions mandatory
- Tables should have proper relationship and keys
- Frontend Application should be menu driven.
- Proper Menu / Navigation for corresponding role
- Client-side Validations and server-side validations are mandatory
- Error should be handled
- Follow best programmer practice while developing
- Provide proper Naming Conventions

**Platform Prerequisites (Do's and Don'ts):**

1. The react app should run in port 8081.
2. The springapp should run in port 8080.

**HOW TO RUN THE PROJECT :****FRONTEND:****Step 1:**

Open the terminal

Use "nvm use 14" command to change node version to 14

**Step 1:**

Use "cd reactapp" command to go inside the reactapp folder

Install Node Modules - "npm install"

**Step 2:**

Write the code inside src folder

Create the necessary components

**Step 3:**

Click the run test case button to run the test cases

**Note :**

- Click PORT 8081 to view the result / output
- If any error persists while running the app , delete the node modules and reinstall them

## **BACKEND:**

**API endpoint:**  
8080

### **Platform Guidelines:**

To run the command use **Terminal** in the platform.

### **Spring Boot:**

Navigate to the springapp directory => **cd springapp**

To start/run the application '**mvn spring-boot:run**'

Click on the Run Test Case button to pass all the test cases

To Connect Database Open Terminal

Cmd:**mysql -u root --protocol=tcp -p**

Password: **examly**