**LAB ASSIGNMENT 5**

**Name: Sanket Hol**

**PRN:12310611**

**Class: TY AIML-A**

**Batch: 3**

**Roll No: 44**

**Experiment Number: 05**

**Title:**

Design and develop a responsive website to prepare one semester result of VIT students using  
REACT, Springboot and MySQL/ MongoDB/Oracle. Take any four subjects with MSE Marks (30%)  
ESE Marks (70%).

**Theory:**

Student result generation systems are widely used in educational institutions for automated marks calculation and grade processing. A modern implementation requires a **full-stack architecture**, typically involving:

**Frontend – React JS**

* Component-based structure
* Hooks for state management
* Axios for API communication
* Responsive UI with CSS/Bootstrap

**Backend – Spring Boot**

* REST API creation
* Model-View-Service architecture
* Database connectivity using Spring Data JPA
* Secure & scalable server-side logic

**Database – MySQL / MongoDB / Oracle**

Stores:

* Student Information
* Subject Marks (MSE + ESE)
* Final Calculated Results

**Result Calculation Logic**

For each subject:

* **MSE Weightage:** 30%
* **ESE Weightage:** 70%
* **Final Marks = (MSE × 0.3) + (ESE × 0.7)**
* Grade assignment based on total marks

This experiment demonstrates end-to-end development: frontend → backend → database → result computation → display on UI.

**Modules Designed:**

**1. Student Result Entry Page**

* Fields: Student Name, PRN, Semester
* Input fields for **four subjects** with MSE & ESE marks
* Validation for numeric ranges

**2. Result Calculation Engine (Backend)**

* Spring Boot service computes final weighted marks
* Stores values in database
* Returns final computed results to frontend

**3. Result Display Page (Frontend)**

* Displays per-subject marks breakdown
* Shows final marks & grade
* Responsive table layout
* Option to view/save results

**4. Database Schema**

Table: results

| **Field** | **Type** | **Description** |
| --- | --- | --- |
| id | PK | Auto-generated |
| student\_name | varchar | Student full name |
| prn | varchar | Unique PRN |
| subject\_name | varchar | Subject title |
| mse | int | MSE marks |
| ese | int | ESE marks |
| final\_marks | float | Weighted score |
| grade | char | A/B/C/D/F |

**Software & Technologies Used:**

* **Frontend:** React JS, Axios, Bootstrap
* **Backend:** Spring Boot, Spring Web, Spring Data JPA
* **Database:** MySQL / MongoDB / Oracle
* **Tools:** VS Code, IntelliJ/Eclipse, XAMPP/Workbench, Postman

**Procedure:**

1. **Backend (Spring Boot)**
2. Open Spring Initializr → Select:
   * Spring Web
   * Spring Data JPA
   * MySQL Driver
3. Generate & extract project.
4. Configure application.properties
   * DB URL
   * Username/Password
   * JPA settings
5. Create **Entity Class**
   * Result.java
6. Create **Repository Interface**
   * Extends JpaRepository
7. Create **Service Layer**
   * Method to compute (MSE × 0.3) + (ESE × 0.7)
8. Create **REST Controller**
   * /api/results/add → POST
   * /api/results/get/{prn} → GET
9. Test API using Postman.

**B. Frontend (React JS)**

1. Create React App:

npx create-react-app frontend

1. Install Axios & Bootstrap:

npm install axios bootstrap

1. Pages Created:

* Result Entry Form
* Marks Table
* Final Result Page

1. Axios Configuration:

* Base API URL
* POST request to submit marks
* GET request to display result

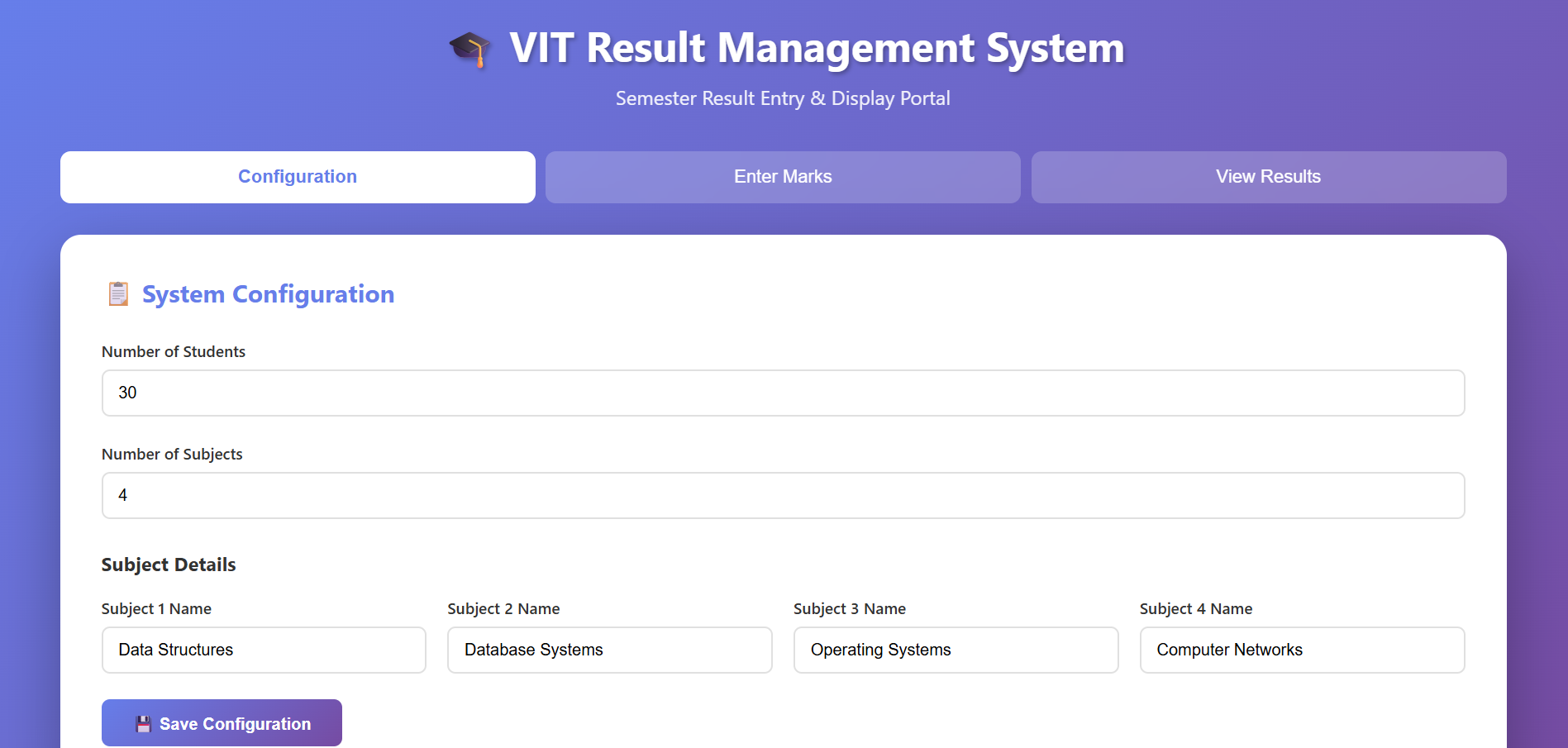
1. Implement UI:

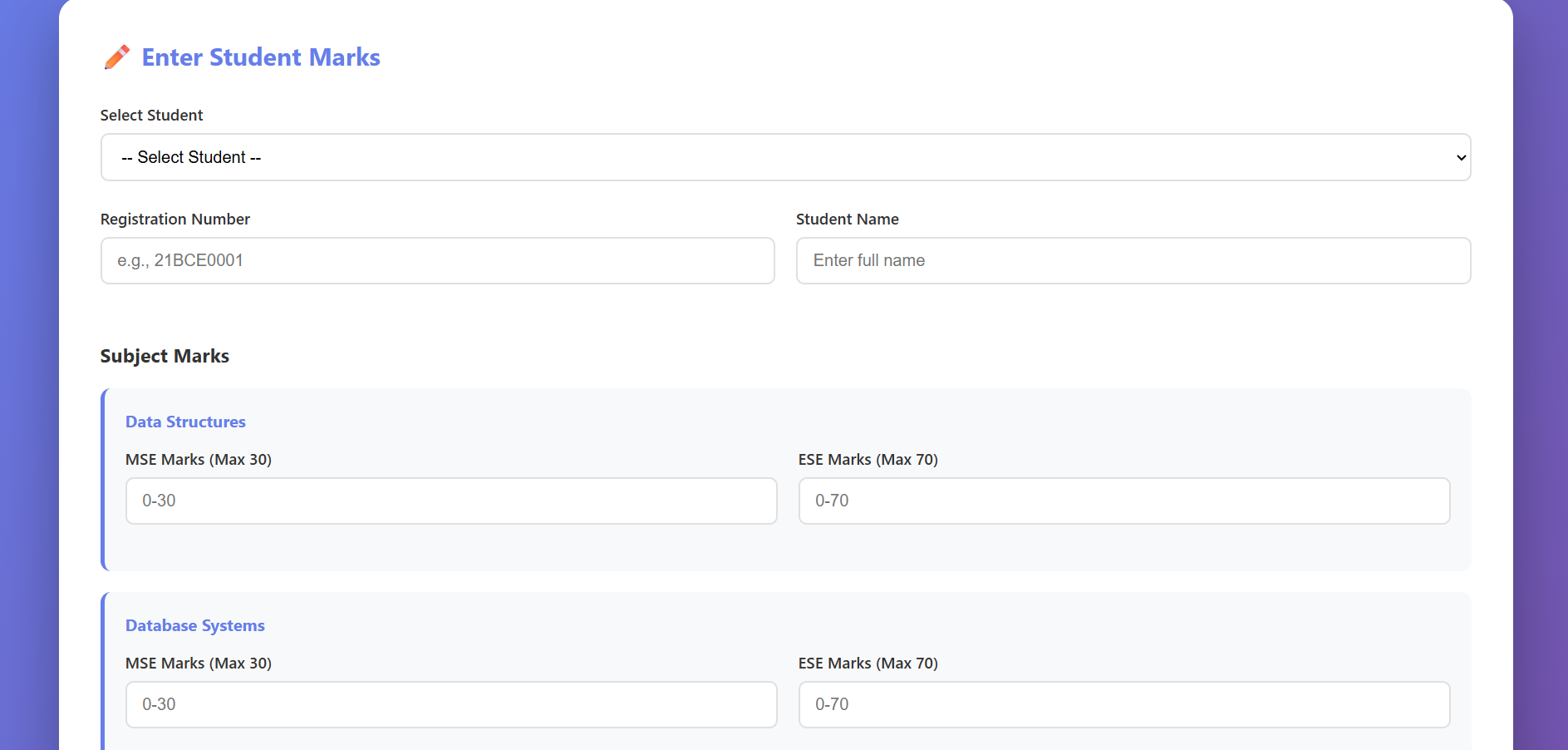
* Four subject fields
* MSE/ESE inputs
* Submit button
* Responsive design

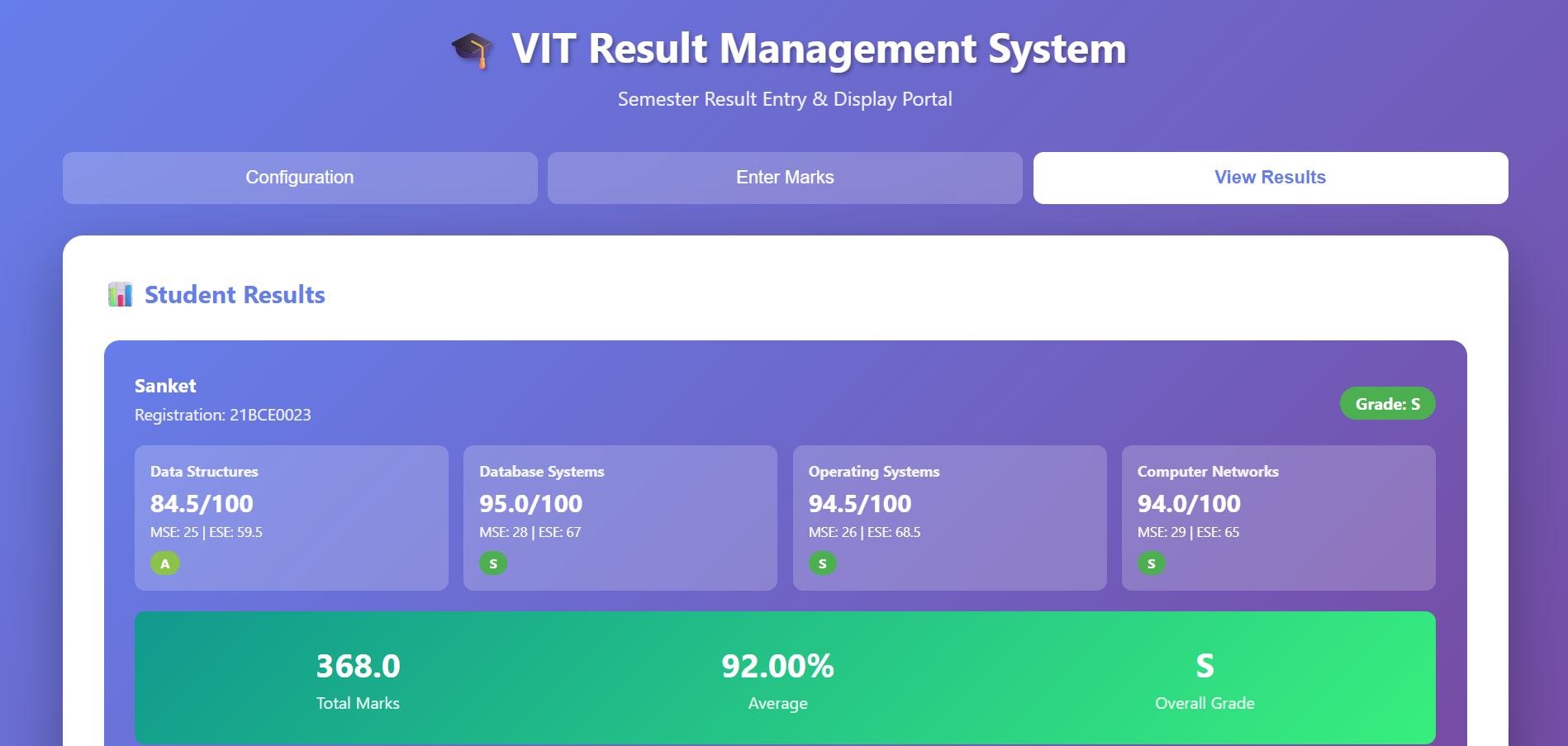
**C. Integration**

* Connect frontend & backend using Axios
* Verify marks calculations
* Check DB entries
* Ensure UI updates dynamically

**Sample Output**







**Conclusion:**

The developed application successfully computes semester results for VIT students using weighted MSE and ESE marks. The project integrates:

* **React** for interactive and responsive UI
* **Spring Boot** for server-side REST APIs
* **MySQL/MongoDB/Oracle** for persistent storage

The system automates result generation, reduces manual calculation errors, and demonstrates complete full-stack development skills. This experiment strengthens understanding of:

* API design
* Data flow across frontend → backend → database
* Weighted score computation
* Responsive web application development