Software Requirements Specification

For

< BSCS Admission Management System for GC University Lahore >

Version 1.0 approved

Prepared by < Hanya Khan - 6440>

<06-01-25>

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The **Software Requirements Specification (SRS)** document consists of all requirements regarding the **BSCS Admission Management System for Government College University (GCU)**, Lahore. The Admission Management System functions basically to automate the admission process for prospective BSCS students in a very friendly user- and efficient, and transparent manner.

The document has a very detailed description of functional requirements and non-functional requirements in terms of system user interface design, data management, security measures, and administrative functionalities. This SRS involves automation in different respects: filling of applications, eligibility verification, entry test score handling, merit calculation, and final admission confirming. Thus, it paints the complete picture of how the system would operate and coordinate between students and administrators.

1.2 Document Conventions

This document follows the IEEE Std 830-1998 standard for SRS documents. All requirements are stated in clear way and without ambiguity with "shall" statements for mandatory requirements.

1.3 Intended Audience and Reading Suggestions

For whom this document is made are the following:

- Developers: responsible for the designing and implementation of the system.
- Project Managers: manage the project to develop the system according to the required specifications.
- End-Users: prospective students, admission staff, and all who would be using the system.

The suggestion goes to the reader to initially read overview sections 1.1-4 to get a better understanding of purpose, scope, and context of the system.

1.4 Product Scope

The BSCS Admission Management System is a software solution designed to automate and streamline the admission procedure of Government College University (GCU), Lahore. At its core, the system aims to improve efficiency in the admission process, as well as transparency and results of admission. This complements the strategic goal of the University to go with practices that are newer and enhanced with technology.
The system offers the following benefits:

- Simplifies the application process for prospective students by enabling online submission and document uploads.
- Improves communication through real-time tracking of application statuses and prompt delivery of admission decisions.

- Reduces manual administrative tasks, allowing the admission staff to focus on critical reviews and decision-making.
- Automates the creation of report outputs for timely access to admissions information.

Features such as these will enable the aforementioned systems to support GCU Lahore's vision of high-quality education through technology to ensure smooth and transparent admission processes.

1.5 References

★ GCU Lahore Admission Policy 2022
★ BSCS Program Requirements 2022
★ IEEE Std 830-1998: Recommended Practice for Software Requirements Specifications
★ Pakistan Higher Education Commission (HEC) guidelines for admission systems

2. Overall Description

2.1 Product Perspective

BSCS Admission Automation System is the most recent remedy to automation in Admissions of the Bachelor's Degree Program in Computer Science at the School of Software Engineering. It supersedes manual and semi-automated approaches that have been typically employed, in accordance with procedures, exacting tolls in human errors, inefficiencies, and delays.

The present admission process is entirely manual, relying on paper forms, calculating eligibility and occasional merit scores by hand, and has been posting announcements on a wall for that. It is very time-consuming and requires much effort and chance. Consequently, transparency and coordination between the users and the administrators collapsed.

It is entirely automated in the entire admission process and has been made efficient, accurate, and transparent." It also improves the student experience by giving online access to eligibility checking, test score managing, merit list generation, and fee payment status tracking, making those processes free from manual intervention.

2.2 Product Function

The **BSCS Admission Automation System** has many features which are going to streamline and automate the admission process. Below is a high-level summary of the major functions that the system must be able to perform by the user or himself.

1. Student Registration and Application Management

- **Fill Admission Forms**: Students can fill out admission forms online, entering personal and academic details (e.g., Form
- Number, Name, Matric Marks, F.Sc. Marks, Address, Phone Number).
- **Submit Documents**: Students can upload supporting documents like scanned copies of their academic certificates.

- **Eligibility Verification**: The system checks if the student meets the eligibility criterion (minimum 50% in F.Sc.) and notifies the student about their eligibility status.
- **Receive Roll Number**: Eligible students are assigned roll numbers for the entry test when the entry test schedule is released on University official website.

2. Entry Test Management

- **Test Scheduling**: The system schedules the entry test for eligible students and provides details about the test location and timing.
- **Test Score Management**: Students' test are checked and their scores are entered into the system for the next step of admission.

3. Interview Management

- **Interview Scheduling**: After the entry test is done, the system arranges interviews for those accepted by the test who have completed the test.
- **Interview Results Management:**

  Then the interview will proceed, and results will be recorded by the system (score or assessment) for later inclusion in calculating the merit score.

4. Merit Calculation and Ranking

★ **Calculate Merit Scores**: The system calculates each student's final merit score based on the following weightage:
- **Matric**: 10%
- **F.Sc.**: 50%
- **Entry Test**: 30%
- **Interview**: 10%

★ **Generate Merit List**: The system ranks students according to their merit scores and generates the final merit list, displaying the top 200 candidates.

5. Admission Confirmation

- **Fee Payment Integration**: The system is integrated with an online payment gateway that allows students to pay the mandatory admission fee.
- **Form Attestation**: Students will submit attested copies of documents only once the merit list is out.
- **Confirm Admission**: After the student pays the fee and gets their documents attested, the system confirms the student's admission status and updates the records with admission status.

2.3 User Classes and Characteristics

1. Student Class

★ **Description**: Represents the prospective student applying for admission to the BSCS program. This user class interacts with the system primarily for application submission, eligibility checking, test participation, interview scheduling, merit list checking, and fee payment.

★ **Frequency of Use**: **High** during the application and admission process. They are active from the moment they fill the admission form until they confirm their admission.

★ **Primary Functions**:

- **Fill Admission Forms**: Complete personal, academic, and contact details.
- **Eligibility Check**: Ensure that they meet the minimum requirements (e.g., 50% in FSC).
- **Entry Test**: Participate in the entry test and check results.
- **Interview**: If eligible, attend the interview.
- **View Merit List**: Check their final rank based on the merit calculation.
- **Pay Fee**: Pay the admission fee once the merit list is published.
- **Submit Attested Documents**: After confirmation, submit original documents.

★ **Security and Privilege Levels**:

- **Low Privilege**: Only has access to their own data (e.g., form details, test results, and status).
- **Authentication**: Secure login and session management for each student.

★ **Technical Expertise**:

- **Basic to Intermediate**: Students may have varying levels of technical expertise, but the system should be intuitive enough for them to interact with it without much technical knowledge.

★ **Pertinent Requirements**:

- Clear guidance for form submission and eligibility feedback.
- Simple navigation for submitting documents and checking test results.
- Secure login, ensuring privacy and data security.

2. Admission Form Class

★ **Description**: Represents the structured form that students fill out during the admission process. This class handles the collection of essential information such as **personal details**, **academic marks**, **previous qualifications**, and **contact information**.

★ **Frequency of Use**: **Medium**. It is used at the beginning of the process when the student registers and fills in the required information.

★ **Primary Functions**:

- **Fill Form**: Students will fill out personal, academic, and contact details.
- **Form Validation**: Ensure all required fields are filled and meet eligibility criteria (F.Sc. percentage, etc.).
- **Submission**: Submit the filled form for eligibility check and processing.

★ **Security and Privilege Levels**:
- **Low Privilege**: Only access by students who are filling out or editing their form.

★ **Technical Expertise**:
- **Basic**: Needs to be very user-friendly as this is typically the student's first interaction with the system.

★ **Pertinent Requirements**:
- Form fields must be clearly labeled and validated.
- Error messages for missing or incorrect information.

3. Entry Test Class

★ **Description**: Represents the **entry test** that students must take to qualify for admission. The system will need to handle the scheduling, test administration, and scoring of the entry test.

★ **Frequency of Use**: **Medium to High** during the admission process, especially as the test date approaches.

★ **Primary Functions**:
- **Test Scheduling**: Schedule the test for eligible students.
- **Test Results**: Record and store entry test results.
- **Test Participation**: Provide an interface for students to take the test (if automated) or for administrators to manage scores.

★ **Security and Privilege Levels**:
- **Medium Privilege**: Students can only access their own test results, while administrators have full access to scores and test schedules.

★ **Technical Expertise**:
- **Intermediate**: Administrators will need to manage test scheduling, results, and notifications.

★ **Pertinent Requirements**:
- Secure test data handling.
- Integration with other admission process modules to use test scores in merit calculation.

4. Administrator Class

★ **Description**: Represents the **admission administrators** responsible for overseeing the application process, reviewing forms, verifying eligibility, and generating the merit list.

★ **Frequency of Use**: **High** throughout the admission cycle. Administrators will interact with the system regularly to manage students' applications, entry test results, and merit list creation.

★ **Primary Functions**:

- **Process Applications**: Review and verify student application forms.
- **Eligibility Check**: Ensure that all students meet the eligibility criteria before advancing.
- **Test Score Management**: Review and input test scores for students.
- **Interview Scheduling**: Schedule interviews for shortlisted candidates.

- **Generate Merit List**: Based on the predefined weightage (Matric: 10%, F.Sc.: 50%, Test: 30%, Interview: 10%).
- **Approve Admissions**: Confirm the final admission of students after reviewing the merit list and fee payment.

★ **Security and Privilege Levels**:

- **High Privilege**: Access to all data, including sensitive student information, eligibility verification, and merit list creation.
- **Role-Based Access**: Some administrators may have limited access (e.g., only to forms, not test scores).

★ **Technical Expertise**:

- **Intermediate to High**: Administrators will need to understand how to manage student data, process forms, and generate reports.

★ **Pertinent Requirements**:

- Clear guidelines for reviewing and approving applications.
- Ability to manage large volumes of student data securely.

5. Treasurer Officer Class

★ **Description**: Represents the **Treasurer Officer**, who is responsible for managing the fee payment process and ensuring that students pay the required fees to confirm their admission.

★ **Frequency of Use**: **Medium**, primarily focused on the final stages of the admission process after the merit list is published.

★ **Primary Functions**:

- **Fee Payment Management**: Oversee the payment process, including ensuring fees are paid.
- **Payment Confirmation**: Verify that the payment is complete before confirming admission.
- **Financial Reports**: Generate reports related to the financial status of students and overall fee collection.

★ **Security and Privilege Levels**:

- **Medium to High Privilege**: Access to payment details and confirmation statuses, but not to sensitive academic information like test scores.

★ **Technical Expertise**:

- **Basic to Intermediate**: Familiarity with payment systems and reports.

★ **Pertinent Requirements**:

- Integration with payment gateways.
- Ability to confirm payments and update student admission status.

6. Merit List Class

★ **Description**: This class represents the **Merit List** that is generated once all test scores, interviews, and other requirements have been completed.

★ **Frequency of Use**: **Medium to Low**. The merit list is generated once at the end of the process, but it is accessed frequently by both administrators and students once it is published.

★ **Primary Functions**:

- **Generate Merit List**: Calculate the merit scores based on predefined weightages.
- **Display Ranking**: List top students based on merit for the final admission process.
- **Notify Students**: Send out merit list notifications to students.

★ **Security and Privilege Levels**:
- **High Privilege**: Only administrators and top-level management will generate and approve the merit list.

★ **Technical Expertise**:
- **Intermediate to High**: Knowledge of how merit is calculated and how it impacts admission decisions.

★ **Pertinent Requirements**:
- Accurate calculation of merit.
- Ability to handle large numbers of applicants and display results clearly.

7. Interview Class

★ **Description**: Represents the **Interview** process for shortlisted students, which is used to evaluate candidates further after the entry test.

★ **Frequency of Use**: **Medium**. The interview occurs after the entry test, and the interview results affect the final merit score.

★ **Primary Functions**:
- **Schedule Interviews**: Set up interview schedules for shortlisted students.
- **Evaluate Students**: Capture and store interview results (scores, assessments).
- **Impact Merit List**: Ensure the interview results are included in the final merit score calculation.

★ **Security and Privilege Levels**:
- **Medium Privilege**: Only administrators or interview panel members will access the interview results.

★ **Technical Expertise**:
- **Intermediate**: Administrators will need to manage scheduling and scoring efficiently.

★ **Pertinent Requirements**:
- Efficient scheduling and management of interviews.
- Secure handling of interview data.

8. Fee Payment Class

- ★ **Description**: This class represents the **fee payment** process that students need to go through to confirm their admission.
- ★ **Frequency of Use**: **Medium to Low**. The fee payment process is only relevant once the merit list is published and students are ready to confirm their admission.
- ★ **Primary Functions**:
  - **Initiate Payment**: Start the payment process via an integrated payment gateway.
  - **Confirm Payment**: Ensure the payment is received before confirming admission.
  - **Record Transaction**: Store payment records in the system for reporting.
- ★ **Security and Privilege Levels**:

  Medium Privilege: Students and treasurer officers will interact with the payment system.
- ★ **Technical Expertise**:
  - **Basic**: Requires integration with secure payment systems.
- ★ **Pertinent Requirements**:
  - Integration with payment gateways.
  - Secure transactions and record keeping.

2.4 Operating Environment

The **BSCS Admission Automation System** is designed to operate in a modern, flexible, and scalable computing environment. Below is a description of the **hardware platform**, **operating system**, **software components**, and **dependencies** that the system will require to run efficiently.

1. Hardware Platform

- ★ The system will be hosted on both client-side (for users like that of students and administrators) and server-side (for system administrators and database management). Its design is web-based, meaning it will mainly run in all modern web browsers hence it will not require any specific hardware more than average user devices and server resources.

- ★ **Client-Side (Student & Administrator Devices)**:
  - **Desktop/Laptop**: Any modern desktop or laptop with a working internet connection and a modern browser.
  - **Mobile Devices**: Support for smartphones and tablets (Android and iOS) to allow access to the admission system on the go.
  - **Minimum Requirements**:
    - Processor: Any modern CPU (Intel i3 or equivalent).
    - RAM: Minimum 4GB.
    - Storage: Any modern SSD or HDD with at least 10GB of free space.
    - Network: Stable internet connection (at least 2 Mbps for smooth operation).
- ★ **Server-Side (Hosting and Database Server)**:
  - **Server Hardware**:
    - Processor: Quad-Core Processor or higher (Intel Xeon, AMD Ryzen, or equivalent).
    - RAM: Minimum 8GB (for small to medium-scale deployments, scalable as per load).

    - Storage: At least 100GB SSD or scalable cloud storage, depending on the number of students.
    - Backup Storage: Cloud-based or external backups to prevent data loss.
    - Network: High-speed internet connectivity (dedicated bandwidth).

2. Operating System

The system will primarily be **web-based** and accessible through any modern web browser. However, the underlying operating system of both the server and client-side devices needs to be specified:

- **Client-Side (Student & Administrator Devices)**:
  - **Operating Systems Supported**:
    - **Windows**: Windows 10 or higher.
    - **Mac OS**: macOS 10.14 (Mojave) or higher.
    - **Linux**: Any modern distribution (Ubuntu, Fedora, etc.).
    - **Mobile Devices**:
      - **Android**: Version 5.0 (Lollipop) or higher.
      - **iOS**: Version 11.0 or higher.
- **Server-Side**:
  - **Operating Systems**: The server will run on a robust, secure, and scalable operating system.
    - **Linux** (Preferred): Ubuntu Server 20.04 LTS, CentOS 8, or Red Hat Enterprise Linux (RHEL).
    - **Windows Server**: Windows Server 2016 or higher.
    - **Cloud Environment**:
      - If hosted on the cloud (AWS, Google Cloud, or Azure), the underlying OS will depend on the chosen service, typically Linux-based servers (Ubuntu) or Windows-based virtual machines.
    - **Containerization**: If containerized (using Docker, Kubernetes), the host OS can vary, but common choices are **Ubuntu** or **CentOS**.

2.5 Design and Implementation Constraints

The BSCS Admission Automation System faces various design and implementation constraints which affect its architecture and way of development. To begin with, the system must accommodate relevant regulatory policies regarding data privacy and security. In other words, the system must comply with laws concerning data protection, such as the **General Data Protection Regulation (GDPR)** where applicable or local regulations such as **Pakistan's Data Protection Bill.** The sensitive personal data concerning students' names, their scores, and other academic information must be securely stored, transmitted, and managed according to such regulations. In addition to this, the system should further provide the necessary mechanisms for allowing the user to access, change, or erase their data according to privacy laws. More importantly, strict **role-based access control (RBAC)** is to be implemented to ensure that sensitive information is accessed by only authorized personnel.

From a **hardware perspective**, the system should have the capacity to accommodate a large number of student records, especially at peak periods when merit lists would be released. The server **infrastructure** must provide an adequate configuration of at least **8GB RAM** to allow concurrent access by multiple users without degrading performance during high traffic. The system's scalability should also allow for additional storage in line with the increase in the user base and records. Data loss cannot be afforded under any circumstance, more so with important documents, test scores, and payment records. Backup strategies need to be in place in this regard. Client-side, the system should be able to run on a wide range of devices such as desktops, laptops and mobile devices, and should only require a minimal hardware requirement of 2GB RAM.

The emerging system, therefore, has a limitation technologically, which prompts use of very popular web technologies. The front end would use the latest **JavaScript framework**, such as **React.js** or **Vue.js**, to provide energetic, responsive user interfaces across various devices and sizes. The back end would thus comprise of **Node.js** or Java with Spring Boot, developing a strong, scalable environment for business logic and API calls. The system will, however, interact with a relational database such as MySQL or PostgreSQL for structured data storage.
Database queries must be efficient, especially over a large number of records. Payment processing will also be integrated with a third party payment gateway, say PayPal or Stripe; thus, the standards of Payment Card Industry Data Security Standard (PCI DSS) are to be followed for the handling of all financial transactions.

Security considerations underscore, however, the complete design of the system. All communications between a user and the server must be encrypted via secure sockets layer (SSL) or transport layer security (TLS), and any sensitive data, such as passwords, must be encrypted in the database. The system shall maintain multi-factor authentication (MFA) for privileged users such as administrators. Role-based access rights, which restrict the use of sensitive data according to the user's roles, will also be created. Finally, the system will have those common security threats: SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). In addition to that, very thorough logging and auditing for security will be essential to observe prospective vulnerabilities and to ensure correction of several best practices in security.

The **system** must operate using the **HTTP/HTTPS protocol**, with all sensitive data being transmitted over **HTTPS** to maintain confidentiality and integrity. **RESTful APIs** will be used to enable communication between the frontend and backend, as well as for integrating with third-party services like email/SMS notifications and the payment gateway. These APIs will follow standard conventions, using **JSON** for data exchange to ensure interoperability across platforms. Finally, the system's design must adhere to established **coding standards** and **best practices** to ensure maintainability and ease of future updates. The codebase should be modular, with a focus on reusability and separation of concerns, enabling easy maintenance and potential feature enhancements. Developers must follow **Test-Driven Development (TDD)** principles to ensure the system is robust, and include unit and integration tests to cover core functionalities. Furthermore, **documentation** of the system architecture, code, and APIs will be essential for ongoing maintenance and future development efforts. The system should also include **automated backup** mechanisms to safeguard critical data, with regular backups performed to minimize data loss risks.

2.6 User Documentation

The **BSCS Admission Automation System** will be accompanied by a comprehensive set of user documentation components to support users in effectively navigating and utilizing the system. The core documentation will include a **User Manual**, which provides detailed, step-by-step instructions for both students and administrators. For students, the manual will guide them through the process

of filling out the admission form, checking eligibility, viewing test results, and making payments. For administrators, it will explain how to manage student records, generate roll numbers, handle test scores, and process merit lists. This manual will be available in **PDF format** for easy downloading and printing, and an **HTML version** will be accessible through the system's website. In addition to the manual, the system will feature an **Online Help** section, integrated directly within the application to provide users with context-sensitive assistance. This help system will allow users to quickly find solutions to common issues, such as troubleshooting login problems or payment issues. The **Online Help** will be accessible via a help icon within the system and will be presented in **HTML format**, ensuring it can be accessed easily by users at any time while interacting with the system.
To further assist users, **Tutorials** will be provided to help students and administrators familiarize themselves with the system's key features. These tutorials will include **interactive HTML guides** and **video tutorials**, which will be hosted on the university's website or integrated within the system. The video tutorials will serve as a visual walkthrough of key tasks, such as submitting an admission form or generating a merit list, catering to users who prefer visual learning. The system will also include a **Quick Start Guide**, a concise document designed to help both students and administrators perform essential tasks quickly. This guide will provide high-level instructions on how to begin using the system and will be available in **PDF format** for download or as a printed copy.

2.7 Assumptions and Dependencies

<This is not applicable in our project.>

3. External Interface Requirements

3.1 User Interfaces:

The BSCS Admission Management System will offer easy-to-use, intuitive interfaces for students and administrators to engage with the system.The logical characteristics of each interface are described as follows:

1. Student Interface

★ **Purpose:**

Enables students to submit applications, check eligibility, view roll numbers, and review merit list results.

★ **Logical Characteristics:**

❖ **Sample Screens:**

- Admission Form Submission Screen.
- Eligibility Check Result Page.
- Entry Test Roll Number Page.
- Merit List Display Screen.

❖ **Screen Layout Standards:**

- Consistent layout across all pages.
- A navigation bar located at the top for easy access to key functions like “Home,” “Submit Form,” “View Eligibility,” and “Merit List.”
- Clearly labeled form fields for personal and academic information should be organized logically (for example, "Personal Details," "Academic Details").

★ **Standard Buttons and Functions:**

- Submit, Reset, and Cancel buttons for forms.
- Help button at the bottom-right corner for user guidance.
- Standard confirmation messages for successful submissions or alerts for errors (e.g., “Please enter all required fields!”).

★ **Keyboard Shortcuts:**

- Tab to navigate between form fields.
- Ctrl + S for quick form submission.

★ **Error Message Display Standards:**

- Error messages displayed in red text under the relevant field (e.g., “Marks must be a number between 0 and 1100”).
- Pop-up dialog for critical errors (e.g., “Form submission failed due to server error!”).

2. **Administrator Interface**

★ **Purpose:**

Provides tools for managing student applications, verifying eligibility, assigning roll numbers, entering test scores, generating merit lists, and finalizing admissions.

★ **Logical Characteristics:**

★ **Sample Screens:**

- Dashboard for application statistics.
- List of eligible/ineligible students.
- Test Score Entry Screen.
- Merit List Generation and Review Page.

★ **Screen Layout Standards:**

- Left-side navigation panel for quick access to key administrative tasks.
- Table format for data display (e.g., student records, test scores).
- Filters and sorting options for quick data retrieval.

★ **Standard Buttons and Functions:**

- "Approve," "Reject," "Save," and "Generate Merit List" buttons.
- Export functionality (e.g., exporting the merit list as PDF or Excel).

- **Keyboard Shortcuts:**
    - Ctrl + F to search for student records.
    - Ctrl + G to generate a merit list.

- **Error Message Display Standards:**
    - Validation messages for incorrect entries (e.g., "Test score cannot exceed 100!").
    - Status alerts (e.g., "Merit list generation completed successfully!").

3. Common Interface Elements

Both student and administrator interfaces will adhere to the following standards:

- **Consistency:** Maintain consistent fonts, colors, and design layouts throughout all screens.
- **Accessibility:** Ensure that the system is accessible for users with disabilities (e.g., compatibility with screen readers, high-contrast themes).
- **Responsive Design:** Ensure interfaces are tailored for optimal use across desktops, tablets, and mobile devices.
- **Security:** User login screens for secure access, with roles assigned to students and administrators.

3.2 Hardware Interfaces

<This section is **not applicable** to the BSCS Admission Management System as it is a software-based application designed to run on standard personal computers or laptops. The system does not interact directly with any specialized hardware components and only requires basic hardware like a computer, keyboard, and monitor for its operation.>

3.3 Software Interfaces

The BSCS Admission Management System will use basic and widely understood technologies to ensure compatibility and ease of use for students and administrators. The software components and their roles are explained below:

1. Operating System Compatibility

- **Supported Operating Systems:**
    - Microsoft Windows (7 and above).
    - Linux (Ubuntu or similar distributions).

The system must work smoothly on common operating systems used by students and administrators.

2. Database Management System (DBMS)

- ★ **Database:** Microsoft Access or MySQL (basic version).
  To store student records, test scores, eligibility results, and merit lists.
- ★ **Data Items:**
  - **Incoming Data:** Student personal and academic details, test scores, and interview marks.
  - **Outgoing Data:** Eligibility status, roll numbers, and ranked merit lists.
- ★ **Communication:**
  - Data will be stored and retrieved using simple SQL commands such as SELECT, INSERT, and UPDATE.

3. Software for Development

- ★ **Languages Used:**
  - Front-end: Basic HTML for interface design.
  - Back-end: C++ or Java for processing logic and database connectivity.

(To make the system simple to develop and use without needing advanced frameworks or libraries.)

- ★ **Tools:**
  - Any text editor like Notepad++ or an IDE like Code::Blocks, Dev-C++, or Eclipse.

4. User Interface

- ★ **Technology Used:**
  - Basic CLI (Command Line Interface) for administrators (using C++ or Java).
  - Simple GUI (Graphical User Interface) using Java Swing for students and administrators.

  (To allow users to interact with the system through basic forms and menu options.)
- ★ **Data Sharing:** The GUI will take input from students (e.g., application forms) and store it in the database.

5. File Handling

- ★ **Purpose:** For storing and retrieving data in cases where a database is not feasible (e.g., during testing).
- ★ **Implementation:**
  - Text files (.txt) or CSV files will store student details, test scores, and merit lists.
  - File operations like reading and writing will be performed using C++ or Java libraries.

6. Data Communication

- ★ **Type of Communication:**
  - The system will process data locally, meaning all data exchanges happen within the program and the database or files on the same computer.
- ★ **Services Provided:**
  - Form submission.
  - Roll number generation.
  - Display of eligibility results and merit lists.

7. Notifications

- ★ **Method:**
  - Basic alerts or messages shown on the display after task completion (e.g., “Form submitted successfully,” or “You do not meet the eligibility criteria”)
- ★ **.Implementation:**
  - Displayed messages utilizing cout (C++) or System.out.println (Java).

8. Tools and Utilities

- ★ **Development Tools:**
  - Code::Blocks (for C++ programming).
  - Eclipse IDE (for Java programming).
- ★ **Version Control:**
  - Manual file saving or simple version tracking with backups.

9. Data Sharing and Constraints

- ★ **Shared Data:**
  - Student records, test scores, and merit calculations will be managed by a single database or file.
- ★ **Constraints:**
  - Proper input validation in C++ or Java to avoid incorrect data entries.
  - Simple sorting algorithms (e.g., bubble sort or selection sort) to rank students by merit.

3.4 Communications Interfaces

The BSCS Admission Management System includes basic communication functions to ensure smooth interaction between users (students and administrators) and the system. The communication requirements are described below:

1. Communication Functions

- ★ **Email Integration :**
  - The system use email to notify students about their eligibility status, roll numbers, and merit list results.
  - **Message Formatting:** A simple text-based email containing relevant information (e.g., “You are eligible for the test. Your roll number is 12345.”).
  - **Implementation:** Can be handled manually by administrators using a basic email client like Gmail or Outlook.
- ★ **Web-Based Access (Optional):**
  - If implemented, students and administrators can access the system via a web browser.
  - **Protocol Used:** HTTP.
  - **Purpose:** Allows remote access for filling admission forms, viewing eligibility, and checking merit lists.
- ★ **Local Communication:**

- For small-scale use, the system operates on a single computer or within a local network without requiring internet connectivity.
- **Method:** Data will be communicated directly between the application and the database or files on the same machine.

2. Communication Standards

- ★ **File Transfer (Optional):**
  - FTP (File Transfer Protocol) can be used for securely transferring data (e.g., merit list files) between local computers.
- ★ **Network Protocols:**
  - If a network is required, basic TCP/IP communication can be used for connecting multiple systems (e.g., client and server setup).

3. Communication Security

- ★ **Encryption:**
  - If email or web-based communication is used, sensitive data (e.g., roll numbers, test scores) will be encrypted using basic encryption algorithms (e.g., AES) to protect privacy.
- ★ **Authentication:**
  - A login system with username and password is used to secure access to the system.

4. Data Transfer Rates

- Since this is a small-scale system, communication will involve minimal data transfer (e.g., text-based messages or forms). A basic internet connection or local network will suffice.

5. Synchronization Mechanisms

- ★ **For Single-User System:**
  - No synchronization is required as all operations are performed locally.
- ★ **For Multi-User System:**
  - Basic synchronization of data is handled through regular database updates (e.g., when a student submits a form, the database is immediately updated).

6. Communication Channels

- ★ **Student Communication:**
  - Students will interact with the system through the interface, either locally (offline) or via a browser (if web-based).
- ★ **Administrator Communication:**
  - Administrators will receive notifications about student submissions and manage the database.

This communication setup ensures that the system remains simple, cost-effective, and easy to deploy while supporting essential communication needs. Advanced technologies can be introduced later if required.

4. System Features

4.1 Admission Form Management

4.1.1 Description and Priority

- **Description:** This feature allows students to fill out and submit their admission forms online, providing required personal and academic details.
- **Priority:** High
- **Priority Ratings:**
  - Benefit: 9
  - Penalty: 8
  - Cost: 5
  - Risk: 4

4.1.2 Stimulus/Response Sequences

- **Stimulus:** The student accesses the admission portal and submits their application form.
- **Response:** Upon submission of the data, the system checks its validity and saves it in a database before corresponding messages are shown when the submission is successful or when errors exist in the inputs-rejected messages.

4.1.3 Functional Requirements

- **REQ-1:** The system must allow students to enter Form Number, Name, Matric Marks, and F.Sc. Marks.
- **REQ-2:** The system must validate that all required fields are filled and inputs are in the correct format.
- **REQ-3:** The system must store validated form data in the database.
- **REQ-4:** The system must display an error message for incomplete or incorrect forms.

4.2 Eligibility Verification

4.2.1 Description and Priority

- **Description:** This feature checks the eligibility of students based on their F.Sc. marks (minimum 50%).
- **Priority:** High
- **Priority Ratings:**
  - Benefit: 9
  - Penalty: 9
  - Cost: 4
  - Risk: 3

4.2.2 Stimulus/Response Sequences

- **Stimulus:** The student submits their academic details during form submission.
- **Response:** The system calculates eligibility and notifies the student of their status. If eligible, a roll number for the entry test is generated.

4.2.3 Functional Requirements

- **REQ-1:** The system must calculate eligibility based on F.Sc. marks provided by the student.
- **REQ-2:** The system must generate a roll number for students who meet the eligibility criteria.
- **REQ-3:** The system must notify ineligible students with a message explaining the reason.

4.3 Test Score Management

4.3.1 Description and Priority

- ★ **Description:** This feature allows administrators to input and manage entry test scores for eligible candidates.
- ★ **Priority:** Medium
- ★ **Priority Ratings:**
  - Benefit: 8
  - Penalty: 6
  - Cost: 5
  - Risk: 4

4.3.2 Stimulus/Response Sequences

- ★ **Stimulus:** The administrator uploads or inputs test scores into the system.
- ★ **Response:** The system updates each candidate's profile with the test score and confirms successful data entry.

4.3.3 Functional Requirements

- ★ **REQ-1:** The system must allow administrators to upload test scores for each student.
- ★ **REQ-2:** The system must associate test scores with the respective roll numbers.
- ★ **REQ-3:** The system must validate test score data for completeness and accuracy.

4.4 Merit List Generation

4.4.1 Description and Priority

- ★ **Description:** This feature calculates merit scores using a predefined weightage and generates a ranked list of top 200 (as there are only 200 seats available) candidates.
- ★ **Priority:** High
- ★ **Priority Ratings:**
  - Benefit: 10
  - Penalty: 9
  - Cost: 6
  - Risk: 3

4.4.2 Stimulus/Response Sequences

- ★ **Stimulus:** The administrator initiates the merit list generation process.
- ★ **Response:** The system calculates merit scores for all candidates, ranks them, and generates a list of the top 200 candidates.

4.4.3 Functional Requirements

- ★ **REQ-1:** The system must calculate merit scores using the formula:
  - o Matric (10%) + F.Sc. (50%) + Test (30%) + Interview (10%).
- ★ **REQ-2:** The system must sort candidates based on merit scores in descending order.
- ★ **REQ-3:** The system must generate and display a ranked list of the top 200 candidates.
- ★ **REQ-4:** The system must export the merit list in a printable format.

4.5 Fee and Admission Confirmation

4.5.1 Description and Priority

- ★ **Description:** This feature tracks students' submission of attested forms and fee payments to confirm their admission.
- ★ **Priority:** Medium
- ★ **Priority Ratings:**
  - Benefit: 7
  - Penalty: 8
  - Cost: 5
  - Risk: 4

4.5.2 Stimulus/Response Sequences

- ★ **Stimulus:** A student submits their attested documents and pays the required fee.
- ★ **Response:** The system verifies the documents, records the payment, and updates the admission status.

4.5.3 Functional Requirements

- ★ **REQ-1:** The system must allow administrators to verify uploaded documents.
- ★ **REQ-2:** The system must record and update fee payment status for each student.
- ★ **REQ-3:** The system must notify students of successful admission confirmation.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

1. **Response Time:**
   - The system should process and validate submitted admission forms within **2 seconds** to provide immediate feedback to students.
   - Eligibility verification and roll number generation should take no more than **3 seconds** after form submission.
2. **Merit List Generation:**
   - The system must calculate merit scores for all applicants and generate the ranked list of top 200 candidates within **10 seconds** after the process is initiated by the administrator.
3. **Data Retrieval:**
   - The system should retrieve student data, test scores, and merit results from the database in less than **1 second** per query to ensure smooth interaction for both students and administrators.
4. **Concurrent Users:**
   - The system should support at least **100 concurrent users** during peak admission periods, ensuring uninterrupted access to the portal for form submission, result checking, and merit list viewing.
5. **Data Upload and Processing:**
   - Administrators should be able to upload test scores and other bulk data files (e.g., CSV) containing up to **10,000 records**, and the system must process and update the database within **15 seconds**.
6. **Server Uptime:**
   - The system must maintain **99.9% uptime** during the admission cycle to ensure continuous availability for users.
7. **Error Handling:**

- Any system error or failure must display a meaningful error message to the user and log the issue for troubleshooting within **5 seconds** of detection.

8. **Notification Delivery:**
    - Messages or notifications about eligibility constraint, roll number, and merit list results must be sent to a student within one minute of the event trigger.

These performance requirements ensure the system is optimized for peak load periods, thus providing timely responses to users and continuing to give a seamless experience for the student as well as the administrator.

5.2 Safety Requirements

1. **Data Integrity:**
    - So the system must safeguard student data from loss or damage while undergoing processing, storage, or retrieval. Regular database backups must be done daily in order to avoid the loss of data stemming from hardware or software failures.

    - Database must not incur any loss or damage in student data like personal details, academic records, and test scores through processing, storage, and retrieval. Especially for hardware or software faults, daily backups should be done in order to protect from the data loss.
2. **Access Control:**
    - Confidential data such as test scores and merit calculations should only be accessible to authorized staff members such as administrators. Access should be restricted by a user authenticating via a username and password.
3. **Error Prevention:**
    - The system must validate all user inputs (e.g., marks, personal details) to prevent the submission of incorrect or incomplete data that could compromise the admission process.
4. **System Crash Recovery:**
    - The system must include recovery mechanisms to restore the last saved state in case of an unexpected shutdown or crash to avoid loss of progress in form submissions or data updates.
5. **Compliance with Privacy Regulations:**
    - The system must comply with relevant data protection laws, such as **Pakistan's Personal Data Protection Bill**, to ensure the privacy and security of students' personal information.
6. **Fraud Prevention:**
    - Measures should be in place to prevent fraudulent activities, such as the submission of fake documents. For example, administrators should have the ability to verify uploaded documents manually or through external checks if needed.
7. **Secure Communication:**
    - All communication between users and the system (e.g., form submissions, result retrieval) must be encrypted using **HTTPS** to protect against data interception.
8. **Physical Notice Board Display Safeguards:**

- Display material on the notice board must strictly contain necessary details especially - roll numbers, ranks-in-order for safety and avoidance of disclosing sensitive information to the members of the public.

Requirements relating to safety ensure the protection of both students' and the institution's information from abuse and overall integrity and reliability of the admission process.

5.3 Security Requirements

1. **User Authentication:**
    - The system must implement user authentication for students (e.g., login credentials) to access and submit admission forms. Administrators must have role-based access with higher privileges.
2. **Data Encryption:**
    - All privacy-sensitive information exchanged between the user and the system, that is the user's personal information and academic records, should be encrypted using the SSL/TLS protocols to avoid interception.
3. **Database Security:**
    - Definitely, That database must require access control of heavy security. Only authorized administrators will be permitted to view, modify or even delete records.
4. **Protection Against Unauthorized Access:**
    - In order to prevent brute force attacks, the system will provide account locking after a fixed number of unsuccessful login attempts.
5. **Data Privacy Compliance:**
    - The system shall comply with relevant data protection laws (for instance, **Personal Data Protection Bill of Pakistan)** pertaining to the security handling and processing of student data.
6. **Activity Logging and Monitoring:**
    - The system must maintain a log of all administrative actions (e.g., data uploads, merit list generation) for auditing and troubleshooting purposes.
7. **Secure Document Uploads:**
    - If the system allows document uploads (e.g., attested forms), it must validate file types and sizes to prevent malicious file uploads.
8. **Backup and Disaster Recovery:**
    - Regular and institutionalised back-up of all data should be done, and a disaster recovery plan to restore data in case of hardware or software failure must be implemented.

This section shows commitment towards data protection of users as well as the integrity of the system, and adds to the credibility and reliability of the system as a whole. It also mitigates the risk of possible breaches or unauthorized access to data.
In case these aspects are not within the scope of your requirements; for example, if you are dealing solely with the feasibility and not with the implementation in the real world, you can do away with this section. However, it does lend some respectability to the project, letting everyone know that you are aware of the modern-day software needs.

5.4 Software Quality Attributes

1. **Usability:**
   - The system must provide an intuitive interface for students and administrators, ensuring that users can complete tasks (e.g., form submission or merit list generation) without prior training.
   - Forms should include clear instructions and error prompts to guide users through the process.
2. **Reliability:**
   - The system must maintain accurate data storage and retrieval, ensuring zero loss of information during operations.
   - It should perform without failure for at least 99% of the time during the admission cycle.
3. **Maintainability:**
   - The system codebase should be modular and well-documented, allowing developers to easily update or modify specific features, such as weightage criteria or eligibility rules.
4. **Flexibility:**
   - The system should allow administrators to adjust merit calculation weightage (e.g., F.Sc. or test scores) without requiring significant software changes.
5. **Scalability:**
   - The system must be scalable to handle a growing number of applicants and concurrent users as the university's intake increases.
6. **Testability:**
   - The system would include logging and debugging tools for testing components isolatedly, say eligibility verification or merit score calculations.
7. **Portability:**
   - The program would have the capability of running on the widely used Windows 10 and Linux operating systems, making it accessible to almost everyone.
8. **Security:**
   - Ensure robust access controls and encryption mechanisms to safeguard user data.
9. **Correctness:**
   - All calculations, such as merit score generation, must be mathematically accurate and verifiable against manual methods.
10. **Availability:**
    - The system must remain operational during peak admission times with minimal downtime

5.5 Business Rules

1. **Eligibility Criteria:**
   - ★ Only the students having at least a 50% score in F.Sc. can further apply.
2. **Role-Based Access Control:**
   - ★ Students can only access the admission form and their personal data.
   - ★ Administrators are responsible for updating test scores, generating merit lists, and verifying submitted documents.
3. **Merit Calculation Rules:**
   - ★ The final merit score must be calculated using the following weightage:
     - ▪ **Matric Marks:** 10%

- **F.Sc. Marks:** 50%
- **Entry Test Scores:** 30%
- **Interview Performance:** 10%

4. **Application Submission Deadline:**
   - ★ The students will submit their applications on or before the submission deadline; late applications will not be accepted.
5. **Document Verification:**
   - ★ Students must provide attested copies of their documents (e.g., academic certificates) during the final admission process.
6. **Fee Submission:**
   - ★ Only students listed in the final merit list can proceed with fee submission. Admissions are confirmed upon payment of the required fee.
7. **Merit List Rules:**
   - ★ The merit list must include only the top **200 candidates**, ranked by their calculated merit scores.
8. **Transparency Requirements:**
   - ★ This makes all processes, ranging from the eligibility verification and merit score calculations, automated but also logged and auditable for their transparency purposes.

Appendix A: Glossary

This glossary provides definitions and explanations of terms, acronyms, and abbreviations used in this SRS to ensure proper interpretation of the document.

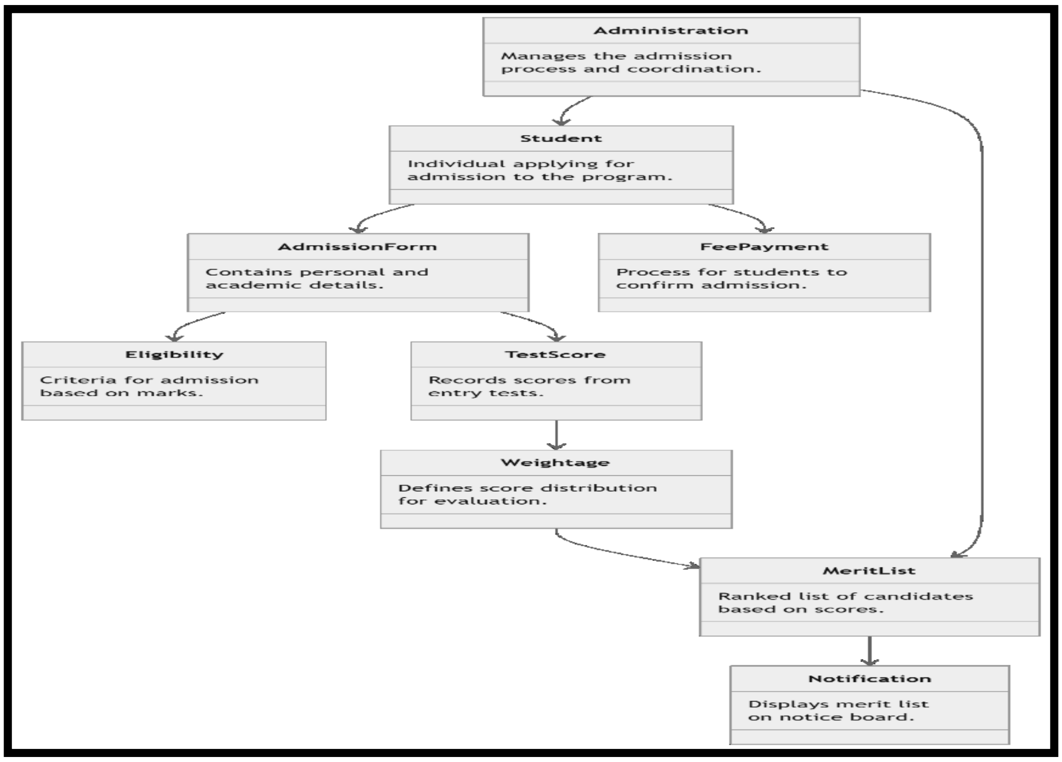
- ★ **Administrator:**
  The personnel responsible for managing the admission system, including data input, test score updates, and merit list generation.
- ★ **Applicant:**
  A student who applies for admission to the BSCS program using the online admission system.
- ★ **BSCS:**
  Bachelor of Science in Computer Science – The undergraduate program offered by the School of Software Engineering, GC University Lahore.
- ★ **Eligibility Criteria:**
  The minimum requirements (e.g., 50% marks in F.Sc.) that a student must meet to proceed in the admission process.
- ★ **F.Sc.:**
  Fellowship of Science – A pre-university qualification in Pakistan, equivalent to high school or intermediate level.
- ★ **GUI:**
  Graphical User Interface – The visual elements of the system that allow users to interact with it, such as forms, buttons, and menus.

- ★ **Merit List:**
  A ranked list of applicants generated based on their calculated merit scores, determining eligibility for admission.
- ★ **Merit Score:**
  The final score calculated using predefined weightage for academic marks (Matric and F.Sc.), entry test scores, and interview performance.
- ★ **Roll Number:**
  A unique identifier assigned to eligible applicants for the entry test.
- ★ **SSL/TLS:**
  Secure Sockets Layer / Transport Layer Security – Protocols used to encrypt data transferred between the system and its users.
- ★ **System:**
  The BSCS Admission Automation System developed for GC University Lahore to streamline the admission process.
- ★ **Test Scores:**
  The marks obtained by an applicant in the university's entry test, which contribute to their merit score.

Appendix B: Analysis Models

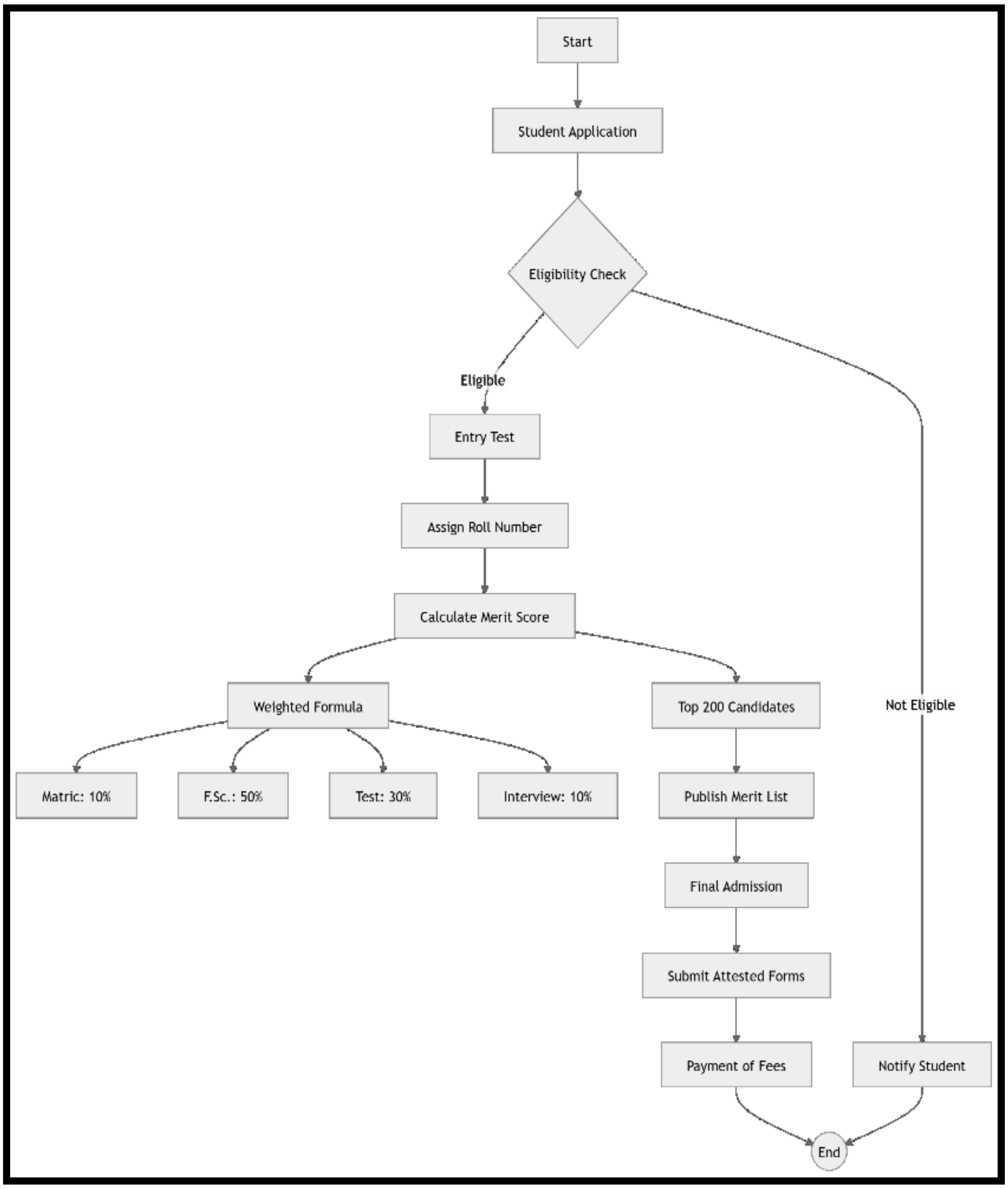
1.General Class Diagram

Represents the structure of the BSCS Admission Management System, including key entities like Administration, Student, AdmissionForm, FeePayment, Eligibility, TestScore, MeritList, and their relationships


Administration
Manages the admission process and coordination.
Student
Individual applying for admission to the program.
AdmissionForm
Contains personal and academic details.
FeePayment
Process for students to confirm admission.
Eligibility
Criteria for admission based on marks.
TestScore
Records scores from entry tests.
Weightage
Defines score distribution for evaluation.
MeritList
Ranked list of candidates based on scores.
Notification
Displays merit list on notice board.

2. Flowchart Diagram

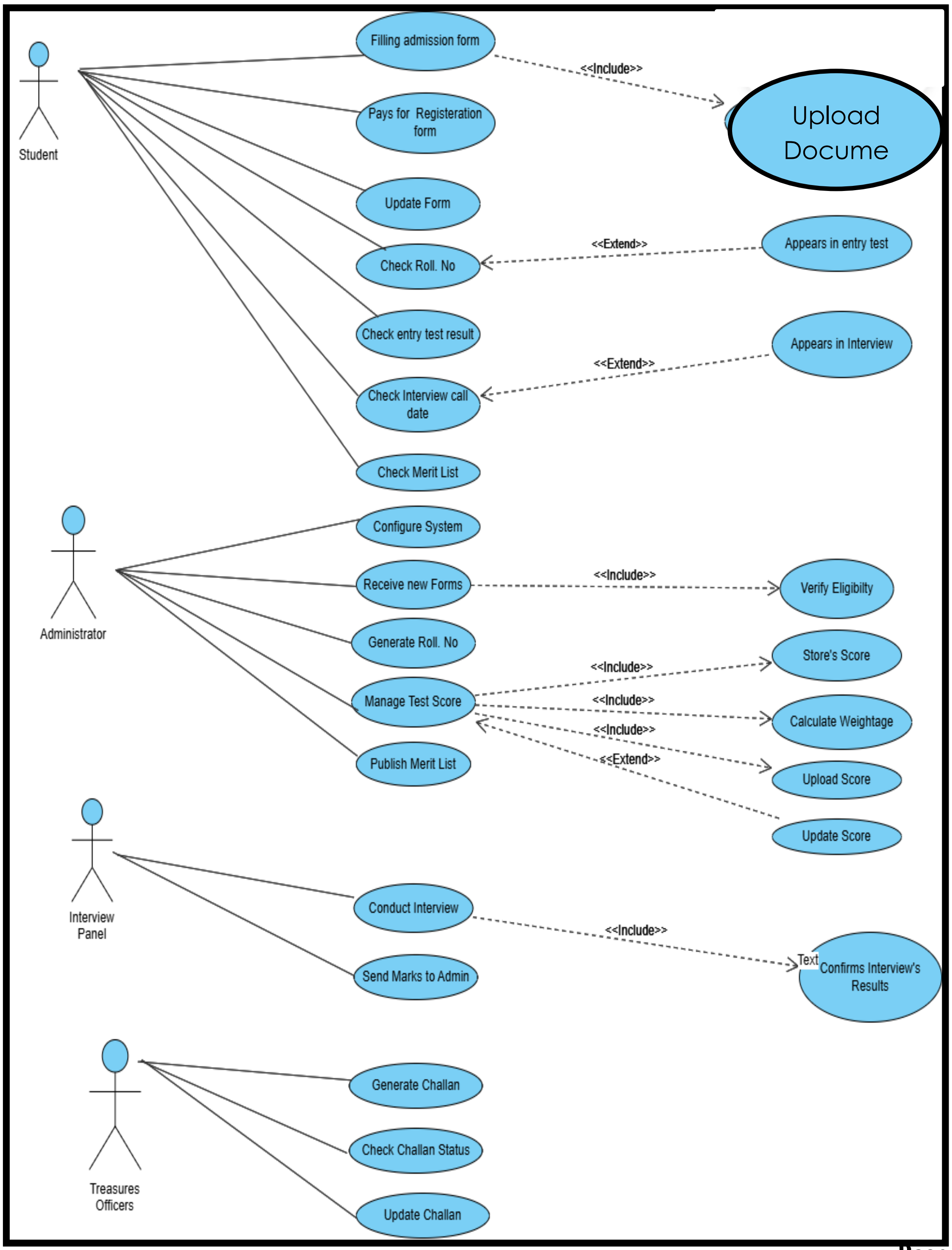
Depicts the step-by-step process flow of the BSCS admission process, starting from student application submission, eligibility check, merit score calculation using predefined weightage, merit list generation, and securing admission.



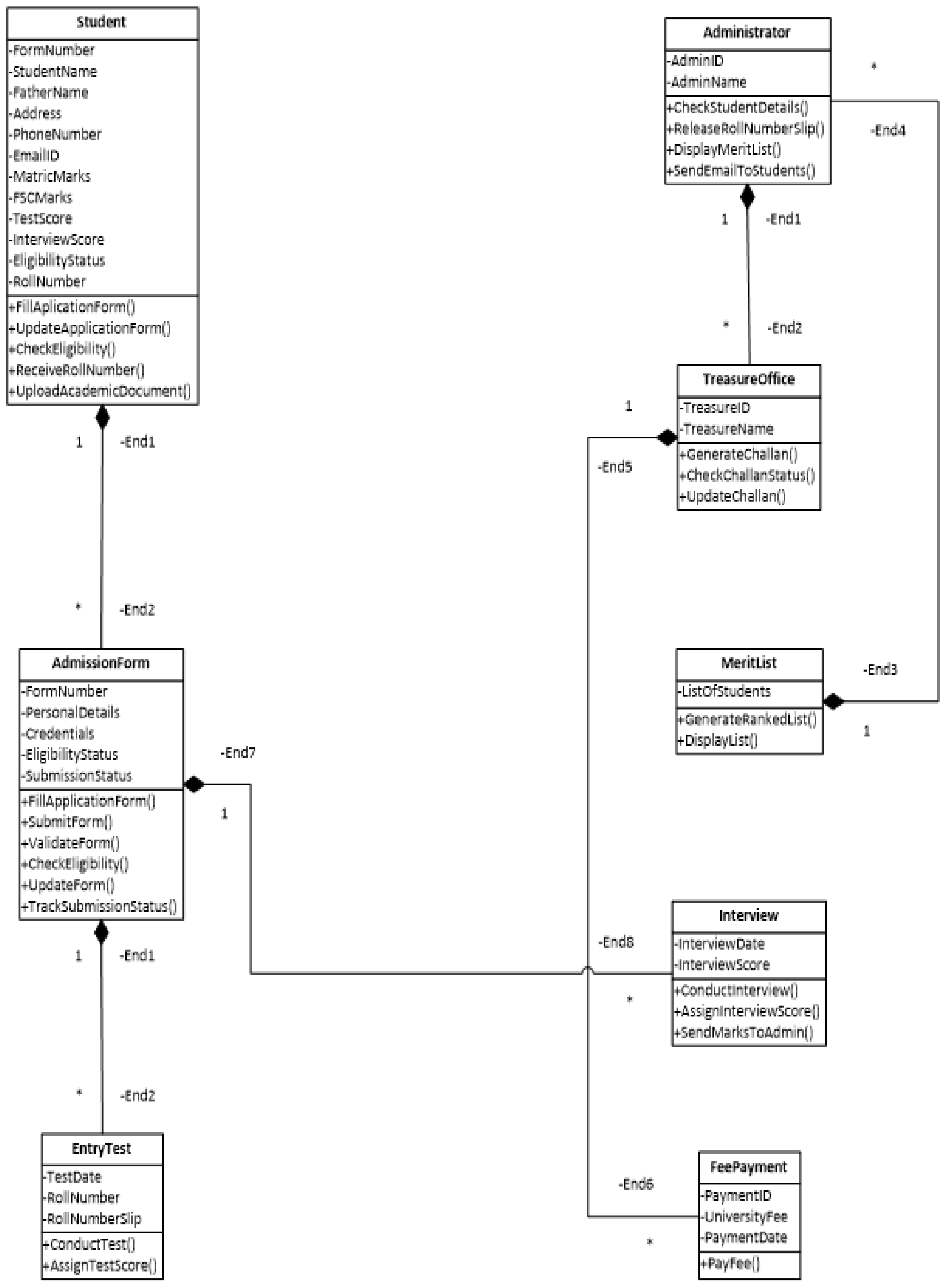
UML: Unified Modeling Language

The following document outlines the **Unified Modeling Language (UML)** representation for the **BSCS Admission System**. It includes:

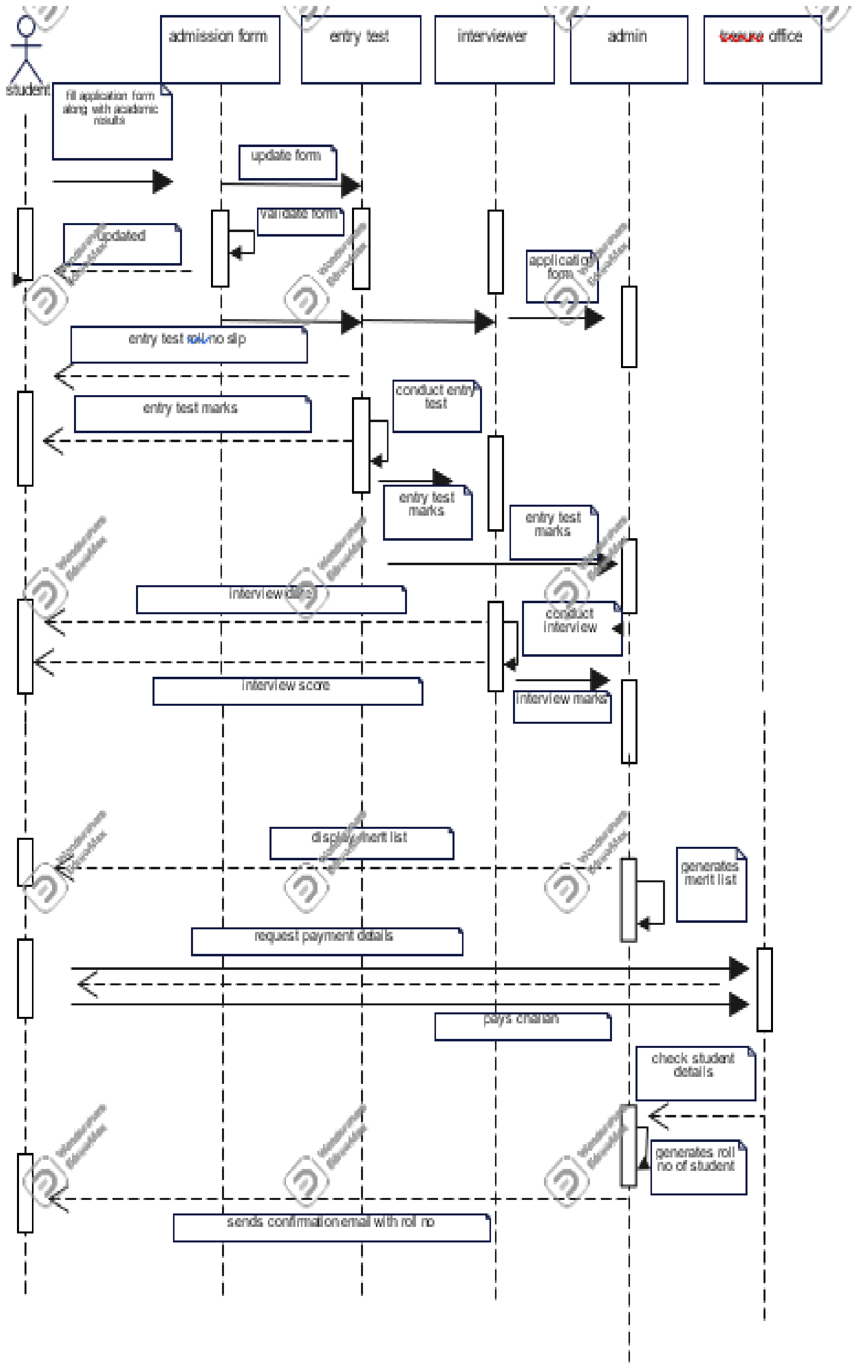
1. **Use Case Diagrams** to illustrate the interactions between users (actors) and the system, showcasing the functional requirements of the admission process.



2. **Class Diagrams :** to represent the structural design of the system, including its classes, attributes, methods, and relationships, providing a comprehensive view of the system's architecture.



3. **Sequence Diagrams** to depict the dynamic behavior of the system by modeling the flow of messages and interactions between objects during specific processes in the admission system.


student
admission form
entry test
interviewer
admin
~~tesure~~ office
fill application form along with academic results
update form
validate form
updated
application form
entry test ~~roll~~ no slip
conduct entry test
entry test marks
entry test marks
entry test marks
interview date
conduct interview
interview score
interview marks
display merit list
generates merit list
request payment details
pays challan
check student details
generates roll no of student
sends confirmation email with roll no