PRACTICALS

OF

SYSTEM ANALYSIS AND DESIGN

SUBMITTED TO THE
DSEU Dwarka Campus
(Delhi Skill and Entrepreneurship University)

In Partial Fulfilment of the Requirements
For the award of

Diploma in Computer Engineering

SUBMITTED BY

NEETU(10621341)

Delhi Skill and Entrepreneurship University
Govt. of NCT of Delhi

UNDER THE GUIDANCE OF
SHIVANI ANAND

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
DSEU DWARKA CAMPUS,
Sector 9, Dwarka, New Delhi ,2024

1. **Identifying User and System Requirements: They should document functional and non-functional requirements.**

FUNCTIONAL REQUIREMENTS:

1.User Authentication:

Users (administrators, teachers, students, parents) must be able to securely log in using unique credentials.

Forgot password/reset password functionality should be available.

2.Student Profile Management:

Ability to create, update, and delete student profiles with information such as name, address, contact details, emergency contacts, etc.

Capture additional details like medical information, special needs, and academic history.

3.Class and Course Management:

Create and manage courses with details such as course code, title, description, prerequisites, and instructor information.

Allow students to enroll in courses and drop courses within specified deadlines.

4.Attendance Tracking:

Record and track student attendance for each class session.

Provide options for manual entry, barcode scanning, or integration with biometric systems.

5.Grade Management:

Input and manage student grades for assignments, quizzes, exams, and other assessments.

Calculate GPA and provide grade reports for students and parents to view.

6.Class Scheduling:

Generate class schedules for each semester/term based on course offerings and instructor availability.

Allow for adjustments to schedules and room assignments as needed.

NON-FUNCTIONAL REQUIREMENTS:

1.Performance:

The system should be responsive and able to handle concurrent users without significant degradation in performance.

Response times for actions like login, grade entry, and attendance tracking should be minimal.

2.Security:

Implement robust authentication mechanisms to ensure only authorized users can access sensitive student data.

Encrypt data transmission and storage to protect against unauthorized access and data breaches.

3.Scalability:

The system should be scalable to accommodate increasing numbers of students, courses, and users over time.

Ensure that performance remains consistent even as the system grows in size and complexity.

4.Reliability:

The system should be reliable and available for use during school hours and critical periods like registration and exam times.

Implement backup and disaster recovery procedures to minimize data loss in the event of system failures.

2.Requirement Specification: This document outlines in detail the agreed-upon functionalities, features, and constraints of the software or system being developed. Produce the same based on above requirements identified in 1.

Requirement Specification Document: Student Information System

1. Introduction:

The Student Information System (SIS) is a comprehensive software solution designed to efficiently manage and streamline the administrative and academic processes related to students in educational institutions. This document outlines the detailed functionalities, features, and constraints of the SIS.

2. Functional Requirements:

2.1 User Authentication:

Description: Users must be able to securely log in to the system using unique credentials.

Details:
The system shall provide a login interface with fields for username and password.
Users shall be authenticated against stored credentials in the system database.
Forgot password/reset password functionality shall be available, requiring users to provide additional verification information.
2.2 Student Profile Management:

Description: The system shall allow administrators to create, update, and delete student profiles with relevant information.
Details:
Student profiles shall include fields for name, address, contact details, emergency contacts, medical information, special needs, and academic history.
Administrators shall have privileges to edit and manage student profiles, ensuring accurate and up-to-date information.

2.3 Class and Course Management:

Description: The system shall facilitate the creation, management, and enrollment in courses.
Details:
Courses shall have attributes such as course code, title, description, prerequisites, and instructor information.
Students shall be able to enroll in courses and drop courses within specified deadlines.
2.4 Attendance Tracking:

Description: The system shall allow teachers to record and track student attendance for each class session.
Details:
Attendance records shall be maintained for each student, linked to specific class sessions.
Options for manual entry, barcode scanning, or integration with biometric systems shall be available for recording attendance.

3. Non-Functional Requirements:

3.1 Performance:

Description: The system shall be responsive and performant under varying loads.
Details:

Response times for critical actions (e.g., login, grade entry) shall be minimal, even under peak usage.
The system shall be capable of handling concurrent users without significant degradation in performance.
3.2 Security:

Description: The system shall implement robust security measures to protect sensitive data.
Details:
Authentication mechanisms shall ensure only authorized users can access the system.
Data transmission and storage shall be encrypted to prevent unauthorized access and data breaches.

3.Design Document: Create a design document illustrating the flow of activities and decision points.

The SIS architecture comprises three main components:

User Interface: Provides interfaces for administrators, teachers, students, and parents to interact with the system.
Application Logic: Implements business logic, data processing, and communication between components.
Database: Stores student information, course data, attendance records, grades, and other relevant data.
3. Activity Flow:

3.1 User Authentication:

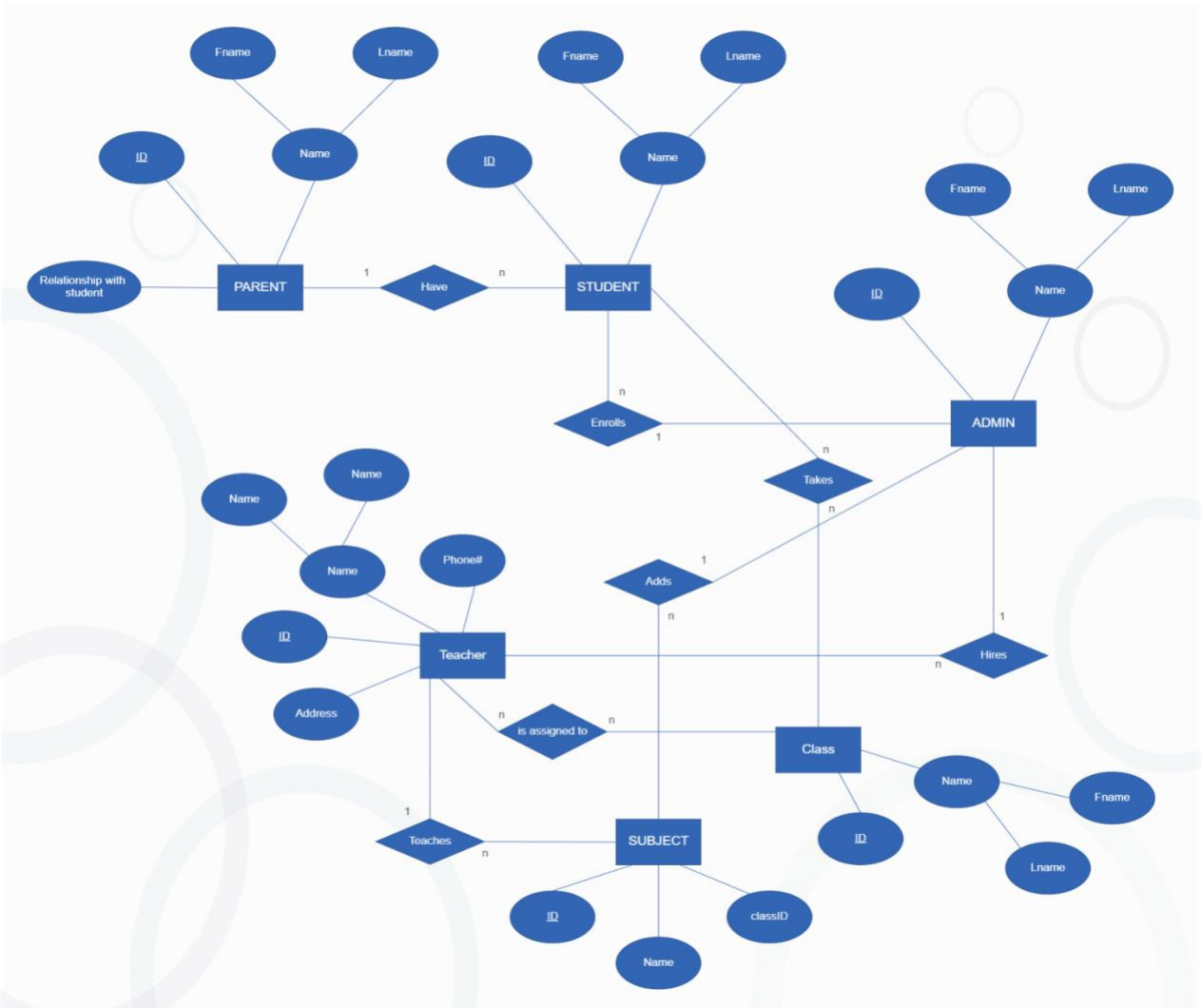
User accesses the login page.
User provides credentials (username, password).
System verifies credentials against the database.
If authentication is successful, user is directed to the dashboard.
If authentication fails, user is prompted to re-enter credentials or reset password.
3.2 Student Profile Management:

Administrator accesses the student management section.
Administrator selects option to create, update, or delete student profiles.
System presents form to input or modify student information.

Administrator submits the form.
System validates input and updates the database.
Confirmation message is displayed.
3.3 Class and Course Management:

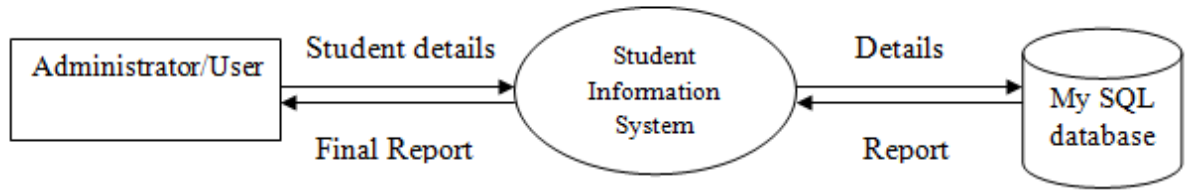
Administrator accesses the course management section.
Administrator selects option to create, update, or delete courses.
System presents form to input or modify course details.
Administrator submits the form.
System validates input and updates the database.
Confirmation message is displayed.

4.Entity-Relationship Diagram (ERD): Design an ERD including entities, attributes, and relationships.

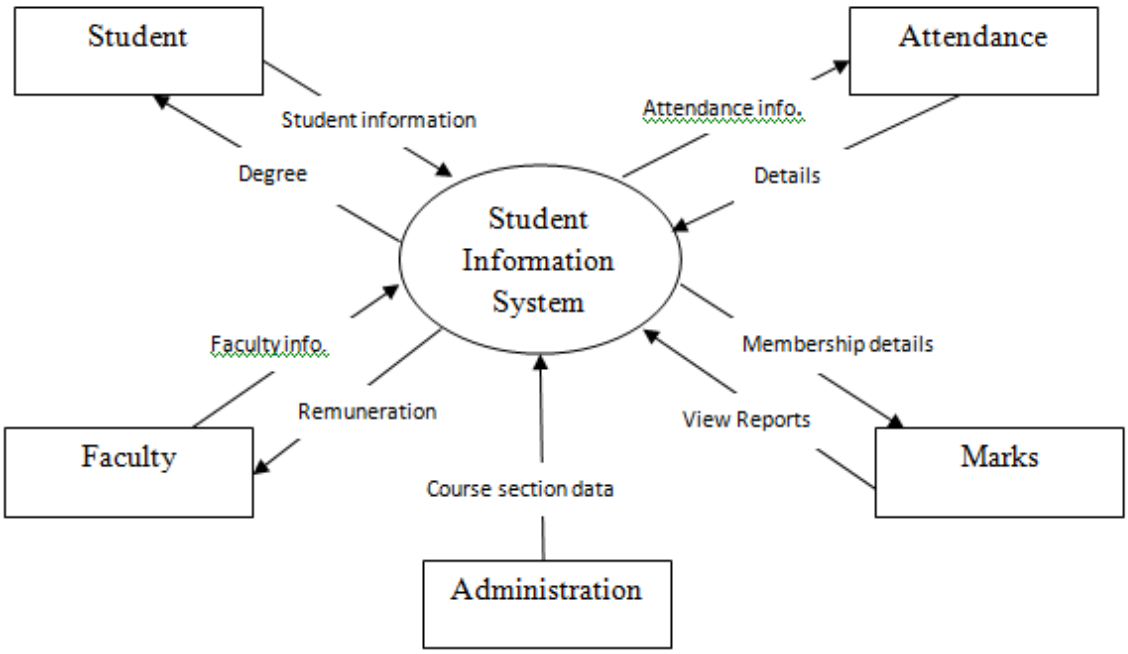


5..Data Flow Diagrams (DFD): Develop a level 0 DFD outlining major processes and data flows. Then, have them decompose it into more detailed level 1 DFDs.

Level 0 DFD :



Level 1 DFD:



6.System Architecture Design: Propose a high-level system architecture, including hardware components, software modules, and interfaces.

High-Level System Architecture Design for Student Information System:

1. Hardware Components:

- Server: Hosts the central database and application logic.
- Client Devices: Computers, laptops, tablets, or smartphones used by administrators, teachers, students, and parents to access the system.

2. Software Modules:

User Interface:

- Web Interface: Provides a user-friendly interface accessible via web browsers on client devices.
- Mobile App Interface: Optionally, a mobile application for convenient access on smartphones and tablets.

Application Logic:

- Backend Server: Implements business logic, data processing, and communication with the database.
- Authentication Module: Manages user authentication and authorization.
- Student Management Module: Handles operations related to student profiles.
- Course Management Module: Manages courses, class schedules, and enrollment.
- Attendance Tracking Module: Records and manages student attendance.
- Grade Management Module: Manages student grades, calculates GPA, and generates reports.
- Communication Module: Facilitates communication between users through messaging, announcements, and notifications.
- Integration Module: Integrates with external systems or services for additional functionalities (e.g., payment processing, learning management systems).

Database:

- Relational Database Management System (RDBMS): Stores student information, course data, attendance records, grades, and communication logs.
- Database Server: Hosts the RDBMS and manages data storage and retrieval.

3. Interfaces:

- User Interface:
- Interfaces with the web interface or mobile app interface for users to interact with the system.
- Application Programming Interfaces (APIs):
- Provides interfaces for communication between the frontend (user interface) and backend (application logic).
- RESTful APIs can be used for communication between client devices and the backend server.

Database Interface:

- Allows the application logic to communicate with the database server for data retrieval, storage, and manipulation.
- SQL or ORM (Object-Relational Mapping) frameworks can be used to interact with the database.

4. Communication Protocols:

- HTTP/HTTPS: For communication between client devices and the web server hosting the user interface.
- API Protocols (e.g., REST, SOAP): For communication between the frontend and backend modules via APIs.
- TCP/IP: For communication between server components within the system architecture.
- SMTP, SMS Gateways: For sending email notifications and SMS alerts through the communication module.

7.Project Plan: Create a project plan including tasks, timelines, resources, and risk management strategies. e,g Gantt Chart

Project Plan for Student Information System Implementation:

1. Project Scope:

- Develop a comprehensive Student Information System to manage student profiles, courses, attendance, grades, communication, and scheduling.

2. Tasks:

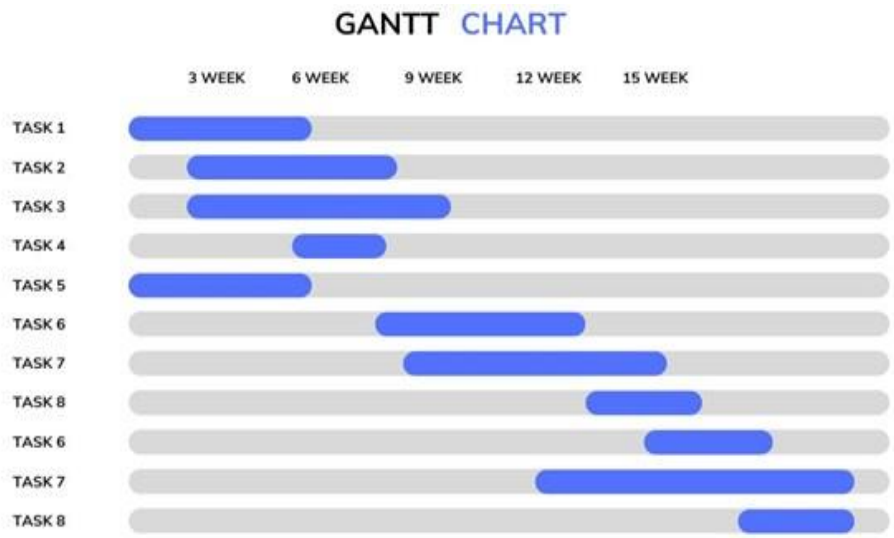
- Requirement Analysis and Documentation
- System Design and Architecture
- Database Design and Implementation
- User Interface Development
- Application Logic Development
- Integration of Modules
- Testing and Quality Assurance
- Deployment and User Training
- Maintenance and Support

3. Timelines:

- Requirement Analysis: 2 weeks
- System Design: 3 weeks
- Database Implementation: 2 weeks
- UI Development: 4 weeks
- Application Logic Development: 6 weeks
- Integration: 2 weeks
- Testing: 4 weeks
- Deployment: 1 week
- User Training: 1 week
- Maintenance: Ongoing

4. Resources:

- Project Manager
- Developers (Backend, Frontend)
- Database Administrator
- UI/UX Designer
- Quality Assurance Engineer
- Technical Support Staff
- Training Facilitators



8.User Interface Design: Create User friendly interfaces aligned with the systems requirement and objectives.

User Interface Design for Student Information System:

1. Login Page:

- Clean and intuitive login interface with fields for username and password.
- Option for users to reset password or recover account.
- Logo and branding elements to establish identity.

2. Dashboard:

- Personalized dashboard for each user role (administrator, teacher, student, parent).
- Overview of important information such as upcoming assignments, grades, attendance, and announcements.
- Quick access to commonly used features and modules.

3. Student Profile Management:

- Form-based interface for creating, updating, and viewing student profiles.
- Clearly labeled fields for student information including name, address, contact details, medical information, and special needs.
- Option to upload profile pictures for visual identification.

4. Course Management:

- Interface for administrators to create, edit, and manage courses.
- Form fields for course details such as course code, title, description, prerequisites, and instructor information.
- Ability to view enrolled students and manage course rosters.

5. Attendance Tracking:

- User-friendly interface for teachers to record and track student attendance.
- List of enrolled students with checkboxes to mark attendance for each session.
- Visual indicators for attendance status (present, absent, late).

9.Prototyping: Build a prototype of a software

Building a prototype for a software like a Student Information System involves creating a simplified version of the system to demonstrate key functionalities and gather feedback from stakeholders. Here's a simplified guide to building a prototype:

1. Define Objectives:

Clearly define the objectives and goals of the prototype. What are the key features and functionalities that need to be demonstrated?

2. Identify Key Features:

Based on the requirements and user needs, identify the key features and functionalities that will be included in the prototype. Focus on essential elements such as login, student profile management, course management, attendance tracking, grade management, etc.

3. Choose Tools and Technologies:

Select appropriate tools and technologies for prototyping. Depending on your preferences and expertise, you can use prototyping tools like Adobe XD, Sketch, Figma, or even simple HTML/CSS for basic prototypes.

4. Design User Interface:

Design the user interface of the prototype. Create wireframes or mockups for each screen or interface component using the selected prototyping tool. Focus on creating a simple and intuitive design that showcases key functionalities.

5. Implement Functionality:

Begin implementing the functionality of the prototype. Use placeholder data or sample data to simulate real interactions. Start with basic functionalities such as login authentication, student profile creation, and course management.

6. Iterate and Refine:

Iterate on the prototype based on feedback from stakeholders. Gather input from users, developers, and other stakeholders to identify areas for improvement and refinement. Make necessary adjustments to the design and functionality to enhance usability and effectiveness.

7. Test and Gather Feedback:

Conduct usability testing with target users to evaluate the prototype. Gather feedback on usability, functionality, and overall user experience. Use feedback to identify areas for further refinement and improvement.

8. Document and Present:

Document the findings, feedback, and revisions made during the prototyping process. Prepare a presentation or report to present the prototype to stakeholders, highlighting key features, functionalities, and improvements.

9. Plan for Next Steps:

Based on feedback and insights gathered from the prototype, plan for the next steps in the development process. Determine whether further iterations of the prototype are needed or if it's ready to move into full-scale development

10.System Testing Plan: Create a comprehensive test plan covering unit testing, integration testing, and system testing. They should identify test cases and expected outcomes.

System Testing Plan for Student Information System:

1. Introduction:

The system testing plan outlines the testing approach for the Student Information System to ensure its functionality, reliability, and performance meet the requirements and expectations of stakeholders.

2. Testing Types:

a. Unit Testing: Testing individual components/modules in isolation.

b. Integration Testing: Testing the interaction between integrated modules.

c. System Testing: Testing the system as a whole to validate end-to-end functionality.

3. Test Environment:

Test environment should replicate the production environment as closely as possible, including hardware, software, and network configurations.

a. Unit Testing:

Login Authentication:

Test Case 1: Verify valid login credentials.

Expected Outcome: User successfully logs in.

Test Case 2: Verify invalid login credentials.

Expected Outcome: Authentication failure with appropriate error message.

Student Profile Management:

Test Case 1: Verify creation of a new student profile.

Expected Outcome: New student profile is created successfully.

Test Case 2: Verify updating existing student profile.

Expected Outcome: Student profile information is updated correctly.

b. Integration Testing:

Integration between User Authentication and Student Profile Management:

Test Case: Verify that user authentication is required before accessing student profile management functionalities.

Expected Outcome: User is prompted to login before accessing student profile management features.

Integration between Course Management and Grade Management:

Test Case: Verify that course details are correctly linked with grade management functionalities.

Expected Outcome: Grades are correctly associated with respective courses.

c. System Testing:

End-to-End Functionality:

Test Case 1: Verify end-to-end flow from login to accessing student profile, course management, attendance tracking, grade management, and communication tools.

Expected Outcome: User can navigate through different functionalities seamlessly.

Data Integrity and Security:

Test Case 1: Verify data encryption for sensitive information such as passwords and grades.

Expected Outcome: Sensitive data is encrypted and securely stored.

Test Case 2: Verify role-based access control for different user roles (administrator, teacher, student, parent).

Expected Outcome: Users can only access functionalities appropriate for their role.

Test Execution:

Execute test cases systematically, documenting results and any deviations from expected outcomes.

6. Defect Tracking and Resolution:

Document and track any defects or issues encountered during testing.

Prioritize defects based on severity and impact on system functionality.

Work with development team to address and resolve defects in a timely manner.

7. Test Completion and Sign-Off:

Complete testing activities once all test cases have been executed and defects resolved.

Obtain sign-off from stakeholders indicating acceptance of system functionality and readiness for deployment.