The University of Azad Jammu and Kashmir

Lab Task # 05

Course Instructor: Engr. Sidra Rafique

Semester: Fall-2024

Session: 2022-2026

Submission Date: Nov 22,2024

Code: SE-3102

Course Name: SC&D

Submitted By: Group-I

R. No: 2022-SE-03

R. No: 2022-SE-06

R. No: 2022-SE-10

R. No: 2022-SE-16

R. No: 2022-SE-22

R. No: 2022-SE-28

R. No: 2022-SE-34

Project Name: Easy Leave

Table of Contents

Software Requirements Specification (SRS)

Project Title:

Easy Leave Management System

---

1. Introduction

1.1 Purpose

The purpose of this document is to define the functional and non-functional requirements for the **Easy Leave Management System**. This system automates leave application workflows, approval processes, notifications, and record maintenance for organizations or educational institutions.

1.2 Scope

The Easy Leave Management System is an intranet-based application that:

- Provides a centralized platform for leave management.
- Enables faculty and staff to apply for, approve, or cancel leaves efficiently.
- Automates leave balance updates, notifications, and report generation.
- Reduces paperwork, minimizes errors, and ensures data security.

1.3 Definitions, Acronyms, and Abbreviations

- **HOD:** Head of Department.
- **HR:** Human Resources.
- **CL:** Casual Leave.
- **CCL:** Compensatory Casual Leave.

1.4 References

- Organizational HR policies regarding leave.
- UML diagrams and software engineering principles.

---

2. Overall Description

2.1 Product Perspective

The Easy Leave Management System replaces the manual leave management process with a digital platform. It maintains leave records, processes applications, and automates approval workflows.

2.2 Product Features

- User login and password management.
- Leave balance inquiries.
- Leave application, withdrawal, and cancellation.
- Notifications via email for all leave-related actions.
- Leave approval or rejection by supervisors.
- Automated leave approvals for unprocessed requests older than two weeks.

2.3 User Classes and Characteristics

1. **Staff Members (Teaching and Non-Teaching):**
    - Apply for leaves and view leave history.
2. **HOD/Manager:**
    - Approve or reject leave applications.
3. **Administrator:**
    - Maintain leave records and manage user accounts.

2.4 Operating Environment

- Web-based application accessible through browsers (Chrome, Firefox, Edge).
- Database server for storing and retrieving leave records.

2.5 Constraints

- The system must operate within an organizational intranet.
- Data storage should not exceed 1 million transactions.

2.6 Assumptions and Dependencies

- All users will have access to the intranet.
- Leave policies are predefined and stored in the database.

---

3. Functional Requirements

3.1 Login System

- Users can log in using unique credentials.
- Passwords can be reset or changed.

3.2 Leave Application

- Users can apply for leave specifying:

    - From and to dates.
    - Reason for leave.
    - Address during leave.
    - Supervisor’s email ID.

3.3 Leave Approval

- Supervisors can approve or reject leave applications.
- Notifications are sent for each action.

3.4 Leave Cancellation

- Users can cancel approved leaves, subject to supervisor approval.

3.5 Notifications

- Automatic emails are triggered for every action (application, approval, rejection, etc.).

3.6 Leave Balance and History

- Users can view their current leave balance and leave history.

3.7 Automated Processes

- Automatic leave crediting as per policy.
- Auto-approval for leave applications older than two weeks.

---

4. Non-Functional Requirements

4.1 Security

- Role-based access control.
- Firewall protection for the intranet system.

4.2 Performance

- The system should handle multiple concurrent users.

4.3 Scalability

- Capable of handling increased data volume and user count.

4.4 Data Integrity

- Regular backups for data protection.

4.5 Usability

- User-friendly interface for all roles (staff, HOD, admin).

---

5. System Models

5.1 Use Case Diagram

- Illustrates user interactions (e.g., staff applying for leave, HOD approving applications).

5.2 Sequence Diagram

- Describes the sequence of events for leave application and approval.

5.3 Activity Diagram

- Maps out the workflow of applying for, approving, and cancelling leaves.

---

6. Data Dictionary

Field Name	Type	Constraints
staffID	Number	Primary key
name	Varchar	Not null
deptID	Number	Foreign key
leaveType	Varchar	CL, CCL, etc.
startDate	Date	Not null
endDate	Date	Not null
status	Char	Pending/Approved/Rejected

---

7. Prototype

The prototype will include:

- **Login Page**: Staff and admin login interface.
- **Dashboard**: Displays leave balance, leave history, and application forms.
- **Application Workflow**: Leave request submission and approval processes.

---

8. Appendices

A. References

- Organizational leave policies.
- Online resources on software engineering methodologies.

B. Glossary

- **HOD:** Head of the Department.
- **HR:** Human Resources.

Pre-Lab Questions

1. Describe various phases of a software project.

A software project generally follows these phases:

1. **Problem Analysis and Planning:**
    - Study the problem thoroughly.
    - Define the project scope, objectives, and constraints.
    - Prepare an infrastructure plan.
2. **Software Requirement Analysis:**
    - Identify functional and non-functional requirements.
    - Break the project into phases or modules and list deliverables.
3. **System Design:**
    - Develop high-level and detailed-level designs like data flow diagrams (DFDs), activity diagrams, and class diagrams.
4. **Implementation (Coding):**
    - Convert the design into executable code.
5. **Testing:**
    - Perform unit testing, integration testing, and system testing to identify and resolve defects.
6. **Deployment and Maintenance:**
    - Deploy the product to the user environment.
    - Provide updates and resolve user-reported issues.

---

2. Explain various process models.

1. **Waterfall Model:**
    - Sequential process, where each phase must be completed before moving to the next. Suitable for well-defined projects.
2. **Spiral Model:**
    - Combines iterative development with systematic risk analysis. Useful for large and complex projects.

3. **Prototype Model:**
    - A prototype is developed to understand user requirements better. Ideal for projects where requirements are unclear.
4. **Agile Model:**
    - Iterative and incremental development with customer collaboration. Best for projects requiring frequent changes.
5. **V-Model:**
    - Verification and validation occur parallelly. Testing happens at each phase of development.

Lab Assignment

1. Analyze at which type of situations which process model can be used in a project.

1. **Waterfall Model:**
    - Use when the requirements are stable, and the project scope is well-defined.
2. **Prototype Model:**
    - Best for projects where requirements are unclear, like the Easy Leave system, to understand and refine requirements.
3. **Agile Model:**
    - Suitable for dynamic projects with changing requirements, ensuring customer feedback.
4. **Spiral Model:**
    - Ideal for large-scale projects with high risks, as risk evaluation happens iteratively.
5. **V-Model:**
    - Appropriate for projects where quality and validation are critical, like healthcare software.

2. Prepare Software Specification Document (SRS) for the given project.

Software Requirement Specification (SRS):

- **Title:** Easy Leave Management System

- **Objective:** Automate leave management for organizations or colleges with centralized database support.
- **Scope:**
  - Reduce paperwork and ensure systematic leave record maintenance.
  - Provide functionalities for leave application, approval, cancellation, and notifications.
- **Functional Requirements:**
  - Login system with password change option.
  - Leave balance and leave history query.
  - Leave application and withdrawal features.
  - Approval and rejection of leave by supervisors.
  - Automatic leave approval after 2 weeks.
  - Email notifications for all leave-related actions.
- **Non-Functional Requirements:**
  - **Security:** Role-based access and firewall protection.
  - **Scalability:** Efficient operation even with a large number of users.
- **Data Modeling:**
  - Use relational database tables like **StaffDetails**, **LeaveDetails**, **LeaveInfo**, and others.
- **UML Diagrams:**
  - Use case diagram, sequence diagram, and activity diagrams for different roles (Employee, HOD, Admin).

Post-Lab Questions

1. Explain various phases of a software project with brief description.

1. **Requirement Analysis:** Understand and document user needs.
2. **Planning:** Define objectives, scope, and resource allocation.
3. **Design:** Create blueprints such as class diagrams and DFDs.
4. **Implementation:** Develop the application based on designs.

5. **Testing:** Identify and resolve defects using test cases.
6. **Deployment:** Install the software in the intended environment.
7. **Maintenance:** Provide updates and resolve issues reported by users.

2. Explain how design can be constructed from analysis.

- **Analysis Output:** Functional requirements and data flows from the requirement analysis phase.
- **Transition to Design:**
    - Create DFDs, use case diagrams, and activity diagrams based on requirements.
    - Define the system architecture and design class diagrams, sequence diagrams, and interface layouts.

3. Describe the coding and testing process in a software project.

- **Coding:**
    - Use the design documents to write code for each module.
    - Follow coding standards and guidelines for consistency.
- **Testing:**
    - **Unit Testing:** Test individual components for functionality.
    - **Integration Testing:** Ensure modules work together.
    - **System Testing:** Validate the system as a whole against requirements.
    - **Acceptance Testing:** Verify the system meets user needs before deployment.