**KidiCode**

**Software Requirements Specification**

Version 1.0



**Group Id: S25PROJECT**

**Supervisor Name : Muhammad Hashir Khan**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Date (dd/mm/yyyy) | Version | Description | Author |
| 16/5/2025 | 1.0 | This Software Requirements Specification (SRS) defines the functional and non-functional requirements for **Kidicode**, a comprehensive web-based Learning Management System (LMS) designed to facilitate online education for multiple user roles including Admins, Instructors, Students, and Parents. The platform integrates course creation, interactive learning, assessments and subscription management into a unified, user-friendly environment. Kidicode enables instructors to create, edit, and manage courses with various difficulty levels and diverse question formats such as multiple-choice, coding challenges, and subjective assessments. Students can access courses, track their progress, complete assignments, and engage through features including points, badges, and leaderboards. Parents have access to dedicated portals to monitor their child’s learning progress. The system supports flexible user authentication methods, including email, phone, and social media sign-ins, ensuring secure access. It also includes free and paid courses with multiple subscription plans and seamless payment gateway integration. Additionally, Kidicode offers a company portfolio and blog section to showcase training programs, student success stories, and system functionalities. Developed following the VU Process Model with elements of the Waterfall and Spiral methodologies, this SRS document provides detailed use case diagrams illustrating core functionalities, usage scenario tables for all identified use cases, and a comprehensive work plan for project progress tracking. This document serves as a guide to ensure the development of a reliable, scalable, and engaging LMS platform tailored to meet modern educational needs. | Bc210423070 |

**Table of Contents**

1. [Scope (of the project)](file:/C:/Users/student/Desktop/ak.as.docx#scope) Page No**: 04**
2. [Functional Requirements Non Functional requirements](file:/C:/Users/student/Desktop/ak.as.docx#FRNFR) Page No: **05,06**
3. [Use Case Diagram](file:/C:/Users/student/Desktop/ak.as.docx#UCD) Page No: **07**
4. [Usage Scenarios](file:/C:/Users/student/Desktop/ak.as.docx#UCS)  Page No: **08**
5. [Adopted Methodology](file:/C:/Users/student/Desktop/ak.as.docx#Adopted)  Page No: **16**
6. [Work Plan (Use MS Project to create Schedule/Work Plan)](file:/C:/Users/student/Desktop/ak.as.docx#Gantt)\_\_\_\_\_\_\_\_\_\_Page No: **20**

**SRS Document**

1. ***Scope Of Project***

The main goal of this project is to design and develop **Kidicode**, a dynamic web-based Learning Management System (LMS) that supports seamless online education for diverse users including administrators, instructors, students, and parents. This system streamlines educational activities by providing a comprehensive platform for course creation, interactive learning, assessment management, and subscription handling. Kidicode enables instructors to upload and manage courses with varied difficulty levels while allowing students to access course materials, track their learning progress, and complete assignments and quizzes. Parents have a dedicated portal to monitor their child’s academic performance. The system supports multiple authentication methods such as email, phone, and social media, ensuring secure and flexible user access. The platform will offer user-friendly features including course browsing, enrollment in free and paid courses, subscription management with monthly or yearly plans, and integration with secure payment gateways. Administrators will manage users, content, analytics, and system settings through a centralized dashboard. The LMS will have a responsive design optimized for desktops, tablets, and mobile devices to provide an engaging and accessible learning experience on any platform. Kidicode supports distinct user roles: Admins, Instructors, Students, and Parents. Each role has specific privileges to access relevant functionalities. Core features include course management, quizzes and assessments, gamification elements like points and badges, real-time progress tracking, secure authentication, payment processing, and role-based access control.

This project aims to enhance the efficiency and accessibility of online education by delivering a reliable, scalable, and secure platform built using modern web technologies. It will facilitate seamless integration with third-party tools such as SCORM/xAPI for learning tracking and popular payment gateways. Ongoing updates and maintenance will ensure the system evolves with educational needs and technology trends. Ultimately, Kidicode aspires to provide a robust foundation for modern e-learning environments, empowering all stakeholders to achieve their educational goals effectively. The system’s modular architecture will allow for easy future enhancements and customization to meet evolving educational demands. User feedback mechanisms will be incorporated to continuously improve the learning experience. By combining innovative features with a user-centric design, LearnKit aims to set a new standard for online education platforms.

* 1. ***Functional Requirements:-***
* **User Roles & Authentication**

1. Admin Panel: Manage content, users, and analytics.

2. Instructor Dashboard: Upload courses, monitor student progress.

3. Student Dashboard: Access courses, track progress, complete assignments.

4. Parent Portal: Monitor child’s learning progress.

5. User Authentication: Sign-up/login via email, phone, or social media.

* **Course Management & Interactive Learning Features**

1. Course Creation & Management: Create, edit, delete courses; assign difficulty levels (Beginner, Intermediate, Advanced).

2. Quizzes & Assessments: MCQs, coding challenges, and subjective questions.

3. Gamification: Points, badges, and leaderboards for engagement.

4. SCORM/xAPI Integration: Ensure compatibility for tracking learning experiences.

* **Payment & Subscription**

1. Free & Paid Courses

2. Subscription Plans: Monthly/Yearly access.

3. Payment Gateway Integration

* **Company Portfolio & Blog**

1. Showcase training programs, achievements, and student success stories.

2. Provide an overview of the LMS and its functionalities.

* 1. ***Non Functional Requirements:-***

**1. Security:**

This includes requirements related to the protection of the system and its data from unauthorized access, as well as the ability to detect and recover from security breaches.

**2. Maintainability:**

This includes requirements related to the ease of maintaining the system, including testing, debugging, and modifying the system.

**3. Reliability:**

This includes requirements related to the system’s ability to function correctly and consistently under normal and abnormal conditions.

**4. Scalability:**

The system must be able to handle a 20% increase in users and transactions without a significant decrease in performance.

**5. Compliance:**

This includes requirements related to adherence to laws, regulations, industry standards, or company policies.

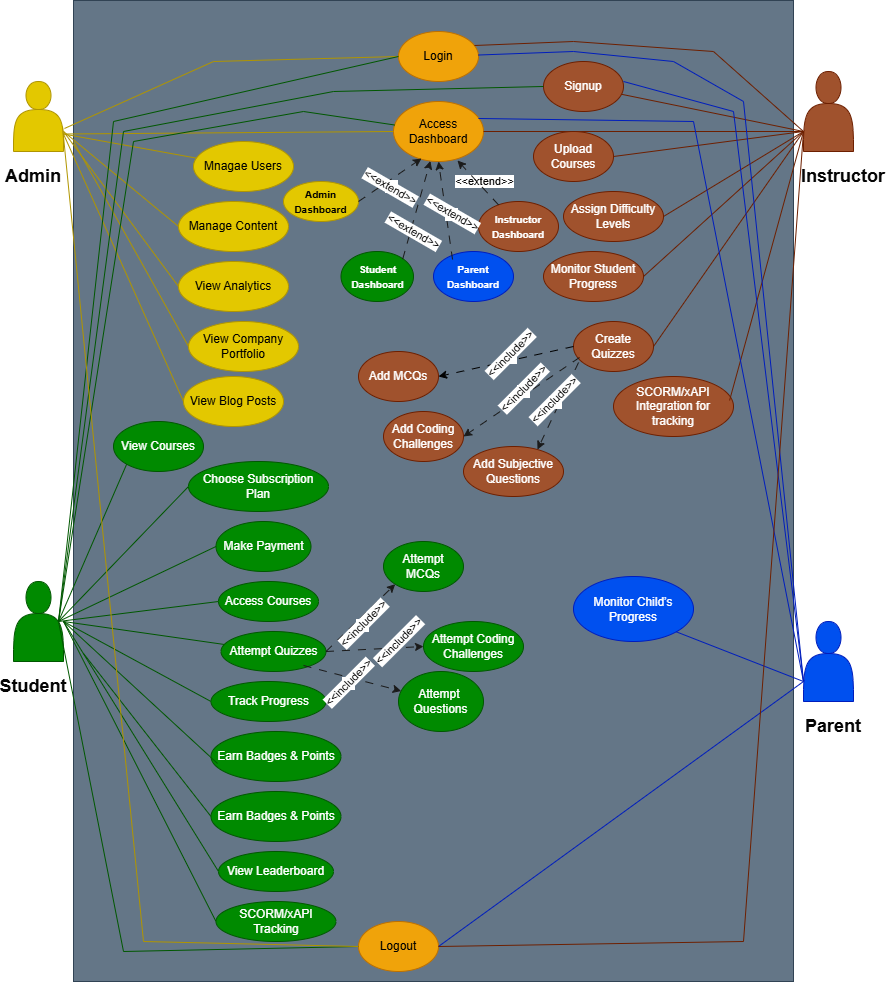
**6.Performance:**

The system must respond to user inputs within 2 seconds and handle a minimum of 100 concurrent users.

**7. Usability:**

This includes requirements related to the ease of use and understandability of the system for the end-users.

1. ***Use Case Diagram(s):-***



1. ***Usage Scanerios:-***

# Signup

|  |  |
| --- | --- |
| Use Case | Signup |
| FR-ID | FR-001 |
| Actors | Student,instructor,parent |
| Description | User can register in via multiple methods. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. User opens signup page. 2. User chooses method: email, phone, or social media. 3. User enters credentials and submits. 4. System validate and registers the user. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Login

|  |  |
| --- | --- |
| Use Case | Login |
| FR-ID | FR-002 |
| Actors | Student,instructor, parent, Admin |
| Description | User can register in via multiple methods. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. User opens login page. 2. User chooses method: email, phone, or social media. 3. User enters credentials and submits. 4. Select role 5. System verify and login the user. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Access Dashboard

|  |  |
| --- | --- |
| Use Case | Access Dashboard |
| FR-ID | FR-003 |
| Actors | Student, Instructor, Admin, Parent |
| Description | User accesses their respective dashboard. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. User logs in successfully. 2. System detects role. 3. System redirects user to respective dashboard. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# View Courses

|  |  |
| --- | --- |
| Use Case | View Courses |
| FR-ID | FR-004 |
| Actors | Student |
| Description | Student can browse available courses. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. User navigates to course listing. 2. System displays all available courses. 3. User selects a course to view details. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Choose Subscription Plan

|  |  |
| --- | --- |
| Use Case | Choose Subscription Plan |
| FR-ID | FR-005 |
| Actors | Student |
| Description | User selects a subscription plan. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. User goes to subscription section. 2. User selects Monthly or Yearly plan. 3. User confirms the selection. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Make Payment

|  |  |
| --- | --- |
| Use Case | Make Payment |
| FR-ID | FR-006 |
| Actors | Student, Parent |
| Description | User completes payment via integrated gateways. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. User selects course or plan. 2. ystem redirects to payment gateway. 3. User enters payment details. 4. System processes payment and updates status. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Monitor Child’s Progress

|  |  |
| --- | --- |
| Use Case | Monitor Child’s Progress |
| FR-ID | FR-007 |
| Actors | Parent |
| Description | Parent tracks child’s performance. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Parent logs in to Parent Portal. 2. Parent selects child profile. 3. System displays child’s progress metrics. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Access Courses

|  |  |
| --- | --- |
| Use Case | Access Courses |
| FR-ID | FR-008 |
| Actors | Student |
| Description | Student accesses enrolled course content. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Student logs in and accesses dashboard. 2. Student clicks on enrolled course. 3. System loads course content. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Track Progress

|  |  |
| --- | --- |
| Use Case | Track Progress |
| FR-ID | FR-009 |
| Actors | Student |
| Description | Student views progress statistics. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Student selects a course. 2. System displays progress bar and analytics. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Attempt Quizzes

|  |  |
| --- | --- |
| Use Case | Attempt Quizzes |
| FR-ID | FR-010 |
| Actors | Student |
| Description | Student attempts quizzes associated with courses. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Student opens course quizzes. 2. Student selects a quiz and begins. 3. System records answers and submits quiz. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Earn Badges & Points

|  |  |
| --- | --- |
| Use Case | Earn Badges & Points |
| FR-ID | FR-011 |
| Actors | Student |
| Description | Gamification elements rewarding user. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Student completes course activities. 2. System awards points or badges automatically. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# View Leaderboard

|  |  |
| --- | --- |
| Use Case | View Leaderboard |
| FR-ID | FR-012 |
| Actors | Student |
| Description | Students can view ranking positions. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Student opens leaderboard section. 2. system displays ranking of all students. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# SCORM/xAPI Tracking

|  |  |
| --- | --- |
| Use Case | SCORM/xAPI Tracking |
| FR-ID | FR-013 |
| Actors | Student, Instructor |
| Description | Tracks learning experience via SCORM/xAPI. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Student launches SCORM/xAPI content. 2. System tracks activity and completion status. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Upload Courses

|  |  |
| --- | --- |
| Use Case | Upload Courses |
| FR-ID | FR-014 |
| Actors | Instructor |
| Description | Instructor uploads new course content. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Instructor accesses upload section. 2. Instructor fills course form and uploads content. 3. System stores and lists the course. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Edit/Delete Courses

|  |  |
| --- | --- |
| Use Case | Edit/Delete Courses |
| FR-ID | FR-015 |
| Actors | Instructor |
| Description | Instructor modifies or removes course content. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Instructor accesses course list. 2. Instructor selects course to edit or delete. 3. System performs the update or deletion. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Assign Difficulty Levels

|  |  |
| --- | --- |
| Use Case | Assign Difficulty Levels |
| FR-ID | FR-016 |
| Actors | Instructor |
| Description | Instructor sets difficulty level for courses. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Instructor selects course. 2. Instructor chooses a difficulty level. 3. System updates course settings. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Monitor Student Progress

|  |  |
| --- | --- |
| Use Case | Monitor Student Progress |
| FR-ID | FR-017 |
| Actors | Instructor |
| Description | Instructor monitors enrolled student activities. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Instructor views enrolled students. 2. System displays performance and progress details. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Create Quizzes

|  |  |
| --- | --- |
| Use Case | Create Quizzes |
| FR-ID | FR-018 |
| Actors | Instructor |
| Description | Instructor creates quizzes for courses. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Instructor opens quiz creation module. 2. Instructor adds questions and options. 3. System saves the quiz and links to course. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Manage Users

|  |  |
| --- | --- |
| Use Case | Manage Users |
| FR-ID | FR-019 |
| Actors | Admin |
| Description | Admin manages user accounts. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Admin opens user management panel. 2. Admin views, edits, or deletes user records. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# Manage Content

|  |  |
| --- | --- |
| Use Case | Manage Content |
| FR-ID | FR-020 |
| Actors | Admin |
| Description | Admin manages platform content. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Admin accesses content panel. 2. Admin adds/edits training programs or blogs. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# View Analytics

|  |  |
| --- | --- |
| Use Case | View Analytics |
| FR-ID | FR-021 |
| Actors | Admin |
| Description | Admin reviews platform-wide analytics. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. Admin opens analytics dashboard. 2. System displays user stats, engagement, and reports. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# View Company Portfolio

|  |  |
| --- | --- |
| Use Case | View Company Portfolio |
| FR-ID | FR-022 |
| Actors | Guest, Admin |
| Description | View organization details and programs. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. User clicks on portfolio link. 2. System shows organizational achievements and programs. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

# View Blog Posts

|  |  |
| --- | --- |
| Use Case | View Blog Posts |
| FR-ID | FR-023 |
| Actors | Guest, Student |
| Description | View latest posts related to LMS. |
| Alternative Path | - |
| Pre Conditions | User authenticated where needed. |
| Action | 1. User opens blog section. 2. System lists blog articles and allows reading. |
| Post Conditions | System updates or displays result. |
| Author | Bc210423070 |
| Exception | System error or invalid input. |
| Modification History | V 1.1 |

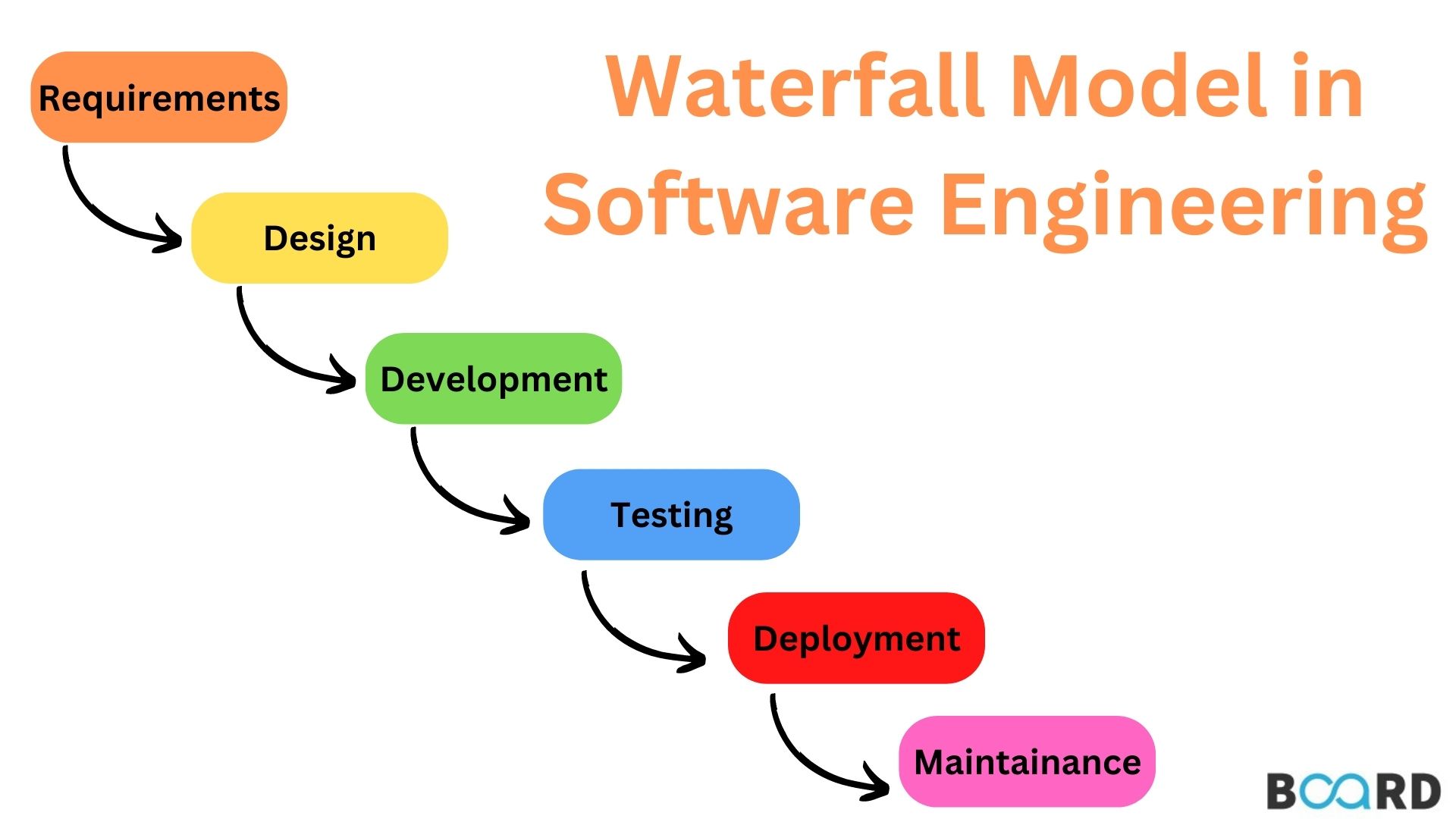
# Logout

|  |  |
| --- | --- |
| Use Case | Logout |
| FR-ID | FR-024 |
| Actors | Student,instructor, parent, Admin |
| Description | User logs out from the system to end the current session |
| Alternative Path | User session already expired. |
| Pre Conditions | User must be logged in. |
| Action | 1. User clicks on the logout button or link.  2. System terminates the user session.  3. User is redirected to the login or homepage. |
| Post Conditions | User session is ended and user is logged out. |
| Author | Bc210423070 |
| Exception | Logout failed due to server error. |
| Modification History | V 1.1 |

1. ***Adopted Methodologies:-***

**1.Waterfall Model:**

The Waterfall model is a linear, sequential software development process where each phase is completed before moving on to the next one, with no overlap or iteration. It follows a rigid structure, requiring fixed and well-defined requirements, and is suitable for projects with complexity and risk.



The six stages above are as follows:

**I. Requirement Analysis and Definition:**

What- The systems services, constraints and goals are established by consultation with system users. They are then defined in detail and serve as a system specification.

**II. System and Software Design:**

How – The system design process partitions the requirements to either hardware of software systems. It establishes and overall system architecture. Software design involves fundamental system abstractions and their relationships

**III.Implementation and Unit Testing: -**

How – During this stage the software design is realized as a set of programs or program units. Unit testing involves verifying that each unit meets its specifications.

**IV. Integration and system testing:**

The individual program unit or programs are integrated and tested as a complete system to ensure that the software requirements have been met. After testing, the software system is delivered to the customer.

**V.Deployment:-**

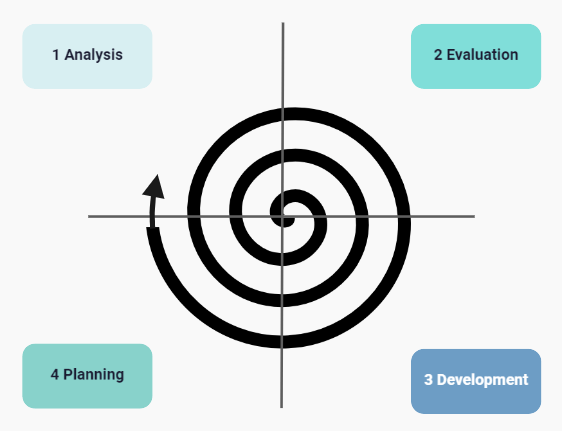
In the Deployment phase, the focus is on ensuring a smooth transition of the software from development to production, and making it available to end-users. This phase involves final testing, packaging, distribution, installation, configuration, user training, and documentation. Once the software is successfully deployed, it is handed over to the maintenance team for ongoing support and maintenance.

**VI. Operation and Maintenance:**

Normally this is the longest phase of the software life cycle. The system is installed and put into practical use. Maintenance involves correcting errors which were not discovered in earlier stages of the life-cycle, improving the implementation of system units and enhancing the system’s services as new requirements are discovered

**2.Spiral Modal:**

The Spiral Model is a [**Software Development Life Cycle (SDLC)**](https://www.geeksforgeeks.org/software-development-life-cycle-sdlc/) model that provides a systematic and iterative approach to software development. In its diagrammatic representation, looks like a spiral with many loops. The exact number of loops of the spiral is unknown and can vary from project to project. Each loop of the spiral is called a **Phase of the**software development.



**I. Objectives determination and identify alternative solutions:**

In this first quadrant, the main goal is to gather as much information as possible from the customers. This includes understanding their needs, expectations, and any constraints they may have. Once the objectives are clearly defined, the team starts brainstorming different solutions that could meet these objectives. These solutions are then evaluated based on their feasibility, cost, time, and alignment with the customer’s requirements. The best solutions are shortlisted for further analysis in the next quadrant.

**II. Identify and resolve Risks:**

The second quadrant is all about risk management. The team evaluates the shortlisted solutions from the first quadrant and identifies potential risks associated with each one. These risks could be technical, financial, operational, or even related to the market or customer preferences. Once the risks are identified, the team develops strategies to mitigate them. This could involve modifying the solution, developing contingency plans, or even discarding the solution if the risks are too high. The end result of this quadrant is a prototype of the chosen solution, which is then tested and refined in the next quadrant.

Develop the next version of the Product:

**III. The third quadrant is where the actual development happens.**

Based on the prototype from the second quadrant, the team starts building the features of the product. This involves coding, testing, and debugging. The team also verifies that the product meets the objectives defined in the first quadrant and that the risks identified in the second quadrant have been effectively mitigated. By the end of this quadrant, a new version of the product is ready for review.

**IV. Review and plan for the next Phase:**

The fourth and final quadrant involves reviewing the product with the customers. They evaluate the product to ensure it meets their needs and provides value. The team collects feedback and uses it to improve the product in the next phase. This quadrant also involves planning for the next phase of the Spiral Model, which could involve scaling the product, adding new features, or even starting a new project.

**Choosen Methodology:-**

I will choose vu process Model for my project which is the combination of waterfall and spiral model.

**3.VU Process Model:**

A structured methodology for process management and improvement, comprising four sequential stages.

**I. Vision (Definition and Goal-Setting)**

Define the desired future state of the process (Vision Statement)

Establish clear goals and objectives (SMART criteria)

Identify key performance indicators (KPIs) to measure success

Define the scope and boundaries of the process

Identify stakeholders and their requirements

**II. Understanding (Current State Assessment)**

Analyze the current state of the process (As-Is process mapping)

Identify strengths, weaknesses, opportunities, and threats (SWOT analysis)

Gather data and feedback from stakeholders (surveys, interviews, etc.)

Identify pain points and areas for improvement

Document the current process (process mapping, procedures, etc.)

**III. Upgrading (Process Improvement and Implementation)**

Design and implement process improvements (To-Be process mapping)

Develop new procedures and standards

Train and support stakeholders

Develop a change management plan

Implement the new process (transition from As-Is to To-Be)

**IV. Validation (Monitoring and Evaluation)**

Monitor and evaluate the improved process

Measure KPIs and adjust as needed

Continuously review and refine the process to ensure sustainability

Celebrate successes and recognize improvements



Reasons For Choosing:-

**I.Structured Approach:**

The VU Process Model emphasizes a structured approach to software development. It ensures that you follow a systematic sequence of steps, from requirements gathering to testing and deployment.

**II. Early Specifications and Design:**

Before writing a single line of code, the VU Process Model focuses on creating detailed specifications and design documentation. This minimizes wastage of effort and time and reduces the risk of schedule slippage or unmet customer expectations1.

**III. Risk Management:**

Like the Spiral Model, the VU Process Model incorporates risk management. It allows you to identify and address potential risks early in the project. By iteratively assessing risks, you can make informed decisions and adjust your approach as needed.

**IV. Adaptability:**

The VU Process Model allows for flexibility. You can adapt it to suit your project’s specific needs. For instance, if certain requirements change during development, you can adjust the design and specifications accordingly.

**V. Clear Documentation:**

The model encourages comprehensive documentation at each stage. This documentation serves as a valuable reference for team members, stakeholders, and future maintenance.

**VI. Gantt Chart Planning:**

The VU Process Model recommends creating a Gantt chart to visualize project activities and milestones. This helps manage project timelines effectively

1. **Work Plan:-**

