**PIZZAHUB**

**Software Requirements Specification**

Version 1.0



**Group Id: S25PROJECT31CA4**

**Supervisor Name : Muhammad Hashir Khan**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Date (dd/mm/yyyy) | Version | Description | Author |
| 16/5/2025 | 1.0 | This Software Requirements Specification (SRS) outlines the functional and non-functional requirements for PizzaHub, a web-based application tailored for pizza fast-food businesses, combining Point of Sale (POS) and online ordering functionalities into one unified platform. The system facilitates efficient order processing for both in-store and online customers, allowing the management of menu items such as pizzas, fast food, drinks, and more, while offering customers the convenience of browsing, ordering, and tracking from their own devices. It supports user account creation, menu browsing, secure order placement, and real-time order tracking. The platform enables administrators to manage product listings, handle in-store sales via the POS module, generate invoices, and access sales reports through a centralized dashboard. Based on the VU Process Model and following a hybrid of the Waterfall and Spiral development methodologies, this SRS includes use case details, functional specifications, system constraints, and development workflows to ensure the creation of a responsive, reliable, and user-centric PizzaHub system. | bc220200976 |

**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: **18**
6. [Work Plan (Use MS Project to create Schedule/Work Plan)](file:/C:/Users/student/Desktop/ak.as.docx#Gantt)\_\_\_\_\_\_\_\_\_\_Page No: **23**

**SRS DOCUMENT**

* 1. **Scope Of Project**

The primary objective of this project is to design and develop a dynamic and feature-rich web application, PizzaHub, which serves as an integrated platform for managing both online and in-store operations of a pizza fast-food business. This system aims to streamline daily business processes by providing an all-in-one solution that combines Point of Sale (POS) capabilities with an online ordering system, ensuring efficient handling of sales, inventory, customer interactions, and reporting. The platform will enable shop owners to manage product offerings such as pizzas, fast food items, and beverages, while allowing customers to conveniently browse the menu and place orders from their homes.

PizzaHub will support intuitive functionalities for customers to register accounts, explore categorized menus, add items to their cart, make secure online orders, and track their order statuses in real-time. Simultaneously, administrators will be able to access a secure dashboard to manage menu items, process walk-in orders using the POS interface, generate invoices, apply promotions, and monitor overall sales performance. The application will feature a responsive and user-friendly interface, optimized for desktops, tablets, and mobile devices to ensure a seamless user experience across all platforms.

The system will accommodate three key user roles: Guests, Registered Customers, and Administrators. Guest users can browse the menu and explore offerings, while registered customers can place and manage orders, view order history, and receive delivery updates. Administrators will oversee platform operations, manage products and customer accounts, track both online and POS orders, and generate comprehensive sales reports. The platform will include secure login, real-time order tracking, invoice generation, and responsive design. It will also support essential features such as search and filter, multiple payment options, and role-based access control.

This project aims to enhance operational efficiency for fast-food businesses while improving the overall customer experience through an accessible, scalable, and secure digital platform. The system will be developed using modern web technologies to ensure high performance, maintainability, and future scalability in a competitive food service market.

* 1. **Functional Requirements:-**

**1: Customer Module(Website):-**

* **User Registration/Login:**

Customers can create accounts and log in securely.

* **Menu Browsing:**

Customers can view the menu with categories (pizzas, fast food, drinks, etc.).

* **Online Ordering:**

Customers can add products to their cart and place orders online.

Customers can provide delivery/pickup details and payment information.

* **Order Tracking:**

Customers can view the status of their orders (e.g., pending, in progress, completed).

* **Order History:**

Customers can view their past orders and invoices.

**2: Admin Module: -**

* **Login/Logout**:

Admin can log in securely to access the POS and dashboard.

* **Product Management:**

Add, update, or delete products (pizzas, fast food, drinks, etc.).

Categorize products (e.g., Pizza, Fast Food, Drinks).

* **Order Management**:

Create and manage in-store orders through the POS.

Add multiple products to an order and calculate the total bill.

Apply discounts or promotions to orders.

* **Invoice Generation:**

Generate and print invoices for completed orders

Invoices include order ID, product list, total amount, taxes, discounts, and date/time.

* **Sales Reporting**:

View daily, weekly, and monthly sales reports.

Track most sold products, total revenue, and average order value.

**3. Admin Dashboard (Website):-**

* **Order Management:**

Admin can view and manage both online and in-store orders.

Admin can update order status (e.g., pending, in progress, completed).

* **Sales Reporting:**

Admin can view sales reports for both online and in-store orders.

* **User Management:**

Admin can view and manage customer accounts. 4.

**4. Shared Features:-**

* **Responsive Design:**

The website and POS are accessible on desktops, tablets, and mobile devices.

* **Search and Filter:**

Customers and admin can search and filter products by name, category, or price.

* 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. Usability:**

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

**3. Reliability:**

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

**4. Maintainability:**

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

**5. Compliance:**

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

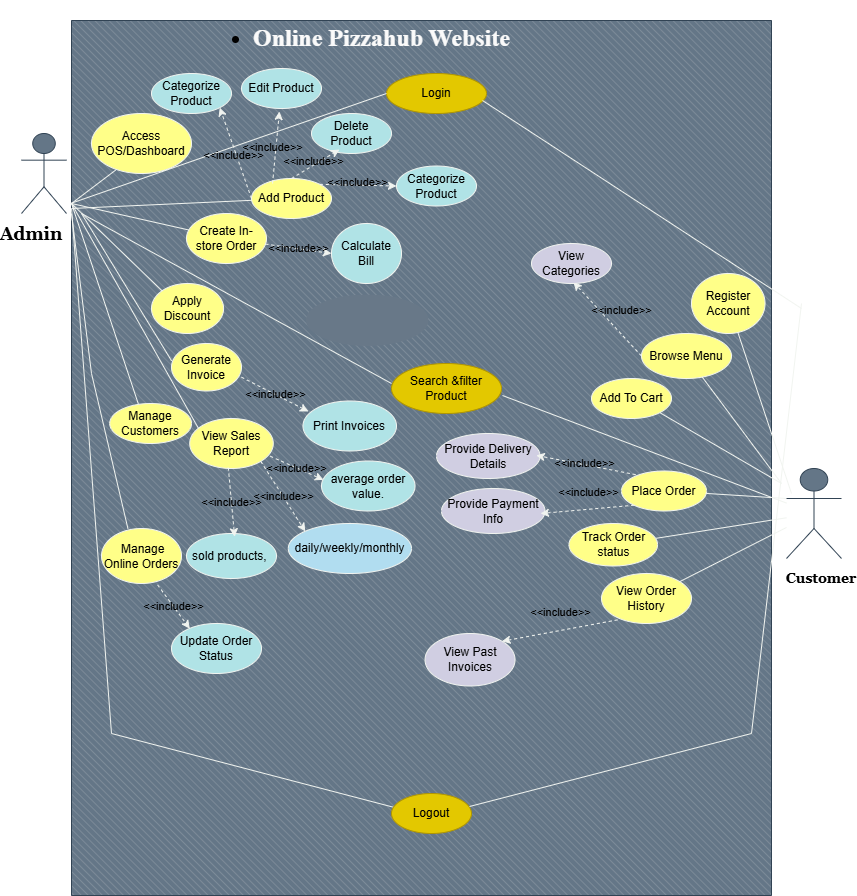
**6.Performance:**

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

**7. Scalability:**

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

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



1. **Usage Scanerios:-**

|  |  |
| --- | --- |
| Use Case | Register |
| FR-ID | FR-001 |
| Actors | Customer (Guest) |
| Description | Allows a new customer to create an account on the PizzaHub website. |
| Alternative Path | - User cancels registration.- System temporarily unavailable. |
| Pre Conditions | - User is on the registration page.- System is available. |
| Action | 1. User navigates to the Register page  2. User fills in required details (username, email, password, etc.).  3. User submits the registration form  4. System validates input.  5. System creates a new account  6. System shows confirmation message. |
| Post Conditions | - New customer account is created.- User can now log in. |
| Author | Bc220200976 |
| Exception | Invalid data input, username/email already exists. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Login |
| FR-ID | FR-002 |
| Actors | Customer, Admin |
| Description | Allows registered users to securely log in. |
| Alternative Path | - Incorrect credentials.- Account locked. |
| Pre Conditions | - User must be registered.- System is available. |
| Action | 1. User navigates to the login page  .2. User enters username/email and password  .3. User submits login form  .4. System validates credentials.  5. If valid, system creates user session  .6. User is redirected to the dashboard/homepage. |
| Post Conditions | - User session is active.- User gains access to authorized features. |
| Author | Bc220200976 |
| Exception | Wrong password, system error. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Browse Menu |
| FR-ID | FR-003 |
| Actors | Customer |
| Description | Allows customers to view the menu items grouped by categories. |
| Alternative Path | - Network failure. |
| Pre Conditions | - User is logged in or guest browsing.- System is available. |
| Action | 1. User selects the "Menu" option.  2. System fetches categories (Pizza, Fast Food, Drinks)  .3. System displays menu items with categories.  4. User browses items. |
| Post Conditions | - Menu is displayed successfully. |
| Author | Bc220200976 |
| Exception | Data fetch failure. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | View the menu with categories |
| FR-ID | FR-004 |
| Actors | Customer |
| Description | Customer views detailed menu with categorized items. |
| Alternative Path | Show all categories as default, Show popular items instead, Show message: "No items available", Redirect to home or suggest other categories |
| Pre Conditions | Menu must be loaded. |
| Action | 1. User selects menu categories  .2. System filters and shows items by category. |
| Post Conditions | Items filtered by category. |
| Author | Bc220200976 |
| Exception | Category not found, No items in selected category, Database fetch error, User not authorized to view (if restricted) |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Add to Cart |
| FR-ID | FR-005 |
| Actors | Customer |
| Description | Allows customers to add items to their shopping cart. |
| Alternative Path | - Item out of stock. |
| Pre Conditions | - User must be logged in.- Items available. |
| Action | 1. User selects an item.  2. User clicks "Add to Cart".  3. System adds item to user's cart.  4. User sees updated cart. |
| Post Conditions | Cart updated with selected items. |
| Author | Bc220200976 |
| Exception | Item unavailable. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Search and Filter Products |
| FR-ID | FR-06 |
| Actors | Admin, Customer |
| Description | Enables searching and filtering of products. |
| Alternative Path | No results found. |
| Pre Conditions | Product database available. |
| Action | 1. User enters search query2. System filters product list3. Displays relevant results. |
| Post Conditions | Products shown based on filters. |
| Author | Bc220200976 |
| Exception | Invalid search query. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Place Order |
| FR-ID | FR-007 |
| Actors | Customer |
| Description | Finalizes the purchase of items. |
| Alternative Path | Edit cart before placing order, Guest user registers before ordering |
| Pre Conditions | Checkout completed. |
| Action | 1. User confirms order details.  2. System saves order.  3. Order confirmation sent to user. |
| Post Conditions | Order recorded. |
| Author | Bc220200976 |
| Exception | Incomplete order details, Payment failure, Item not available |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Provide Payment Information |
| FR-ID | FR-008 |
| Actors | Customer |
| Description | Allows customers to provide payment and delivery details. |
| Alternative Path | - Payment failure. |
| Pre Conditions | Cart must have items. |
| Action | 1. User reviews cart.  2. User clicks checkout.  3. User provides payment info.  4. User confirms order.  5. System processes order. |
| Post Conditions | Order created. |
| Author | Bc220200976 |
| Exception | Payment declined. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Provide Delivery Details |
| FR-ID | FR-009 |
| Actors | Admin |
| Description | Allows admin to enter or update delivery details. |
| Alternative Path | Address invalid. |
| Pre Conditions | Order must exist. |
| Action | 1. Admin enters delivery info. 2. System validates info. 3. Info is saved. |
| Post Conditions | Delivery info saved. |
| Author | Bc220200976 |
| Exception | Validation failed. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Track Order Status |
| FR-ID | FR-010 |
| Actors | Customer |
| Description | Customer can check the status of their placed orders. |
| Alternative Path | Prompt login for tracking, Track via email link with limited info |
| Pre Conditions | Order must exist. |
| Action | 1. User navigates to “Track Order”.  2. User selects an order.  3. System displays order status. |
| Post Conditions | Status displayed. |
| Author | Bc220200976 |
| Exception | Invalid Order ID, Server or network error |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | View Order History |
| FR-ID | FR-011 |
| Actors | Customer |
| Description | Allows customers to see their past orders. |
| Alternative Path | Apply filters for results, Export order history |
| Pre Conditions | User logged in. |
| Action | 1. User navigates to order history.  2. System fetches user orders.  3. Orders displayed with details. |
| Post Conditions | History displayed. |
| Author | Bc220200976 |
| Exception | User not logged in, No order records found |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | View Past Invoices |
| FR-ID | FR-012 |
| Actors | Admin |
| Description | Admin can view all past invoices. |
| Alternative Path | No invoices found. |
| Pre Conditions | Invoices must exist. |
| Action | 1. Admin opens invoice section. 2. System displays invoice list. |
| Post Conditions | Past invoices displayed. |
| Author | Bc220200976 |
| Exception | Database connection lost. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Manage Product |
| FR-ID | FR-013 |
| Actors | Admin |
| Description | Allows admin to add, edit, delete, and categorize products. |
| Alternative Path | Invalid product data. |
| Pre Conditions | Admin must be logged in. |
| Action | 1. Admin opens product management section. 2. Admin adds/edits/deletes product details. 3. Admin categorizes the product. 4. System validates and saves changes. |
| Post Conditions | Product is added/updated/deleted and categorized. |
| Author | Bc220200976 |
| Exception | Validation error or database failure. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Access POS/Dashboard |
| FR-ID | FR-014 |
| Actors | Admin |
| Description | Allows admin to access the POS and main dashboard. |
| Alternative Path | POS module not loaded. |
| Pre Conditions | Admin must be logged in. |
| Action | 1. Admin clicks POS/Dashboard. 2. System loads the POS interface. |
| Post Conditions | POS Dashboard is accessible. |
| Author | Bc220200976 |
| Exception | System crash or loading error. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Create In-store Order |
| FR-ID | FR-015 |
| Actors | Admin |
| Description | Allows admin to place orders for in-store customers. |
| Alternative Path | Out of stock. |
| Pre Conditions | Product must be available. |
| Action | 1. Admin selects product. 2. Admin adds customer info. 3. Admin confirms order. |
| Post Conditions | Order placed successfully. |
| Author | Bc220200976 |
| Exception | Product unavailable. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Calculate Bill |
| FR-ID | FR-016 |
| Actors | System |
| Description | Calculates the final bill based on cart items, discounts, and taxes. |
| Alternative Path | Calculation error due to invalid data. |
| Pre Conditions | Cart must have at least one item. |
| Action | 1. System fetches all cart items2. Calculates item total3. Applies discount4. Adds applicable taxes5. Displays total amount. |
| Post Conditions | Final amount ready for checkout. |
| Author | Bc220200976 |
| Exception | Invalid product data or calculation failure. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Apply Discount |
| FR-ID | FR-017 |
| Actors | Admin |
| Description | Allows admin to apply discounts on customer orders. |
| Alternative Path | Invalid discount. |
| Pre Conditions | Order must exist. |
| Action | 1. Admin selects order. 2. Admin enters discount. 3. System updates bill. |
| Post Conditions | Discount applied. |
| Author | Bc220200976 |
| Exception | Invalid discount format. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Generate Invoice |
| FR-ID | FR-018 |
| Actors | Admin |
| Description | Generate invoices after order confirmation. |
| Alternative Path | Invoice not generated. |
| Pre Conditions | Order must be placed. |
| Action | 1. Admin selects order. 2. Clicks generate invoice. 3. System creates invoice. |
| Post Conditions | Invoice created. |
| Author | Bc220200976 |
| Exception | System error while generating. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Print Invoices |
| FR-ID | FR-019 |
| Actors | Admin |
| Description | Enables admin to print order invoices. |
| Alternative Path | Printer not connected or invoice generation failed. |
| Pre Conditions | Invoice must exist. |
| Action | 1. Admin opens invoice section2. Selects invoice3. Clicks print4. System sends data to printer. |
| Post Conditions | Invoice is printed. |
| Author | Bc220200976 |
| Exception | Printer error, invoice missing. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | View Sales Report |
| FR-ID | FR-020 |
| Actors | Admin |
| Description | Admin views overall sales reports. |
| Alternative Path | No sales data available. |
| Pre Conditions | Admin logged in. |
| Action | 1. Admin selects report type2. System fetches data3. Displays sales report. |
| Post Conditions | Report viewed. |
| Author | Bc220200976 |
| Exception | Data retrieval error. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Average Order Value Report |
| FR-ID | FR-021 |
| Actors | Admin |
| Description | Calculates average order value from order history. |
| Alternative Path | No orders found. |
| Pre Conditions | At least one completed order. |
| Action | 1. Admin navigates to reports2. System calculates total value and order count3. Computes average4. Displays result. |
| Post Conditions | Average order value report generated. |
| Author | Bc220200976 |
| Exception | Division by zero error. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Daily/Weekly/Monthly Sales Reports |
| FR-ID | FR-022 |
| Actors | Admin |
| Description | Generate time-specific sales reports. |
| Alternative Path | Time range not selected. |
| Pre Conditions | Completed orders in selected range. |
| Action | 1. Admin selects range (daily/weekly/monthly)2. System filters data3. Displays report. |
| Post Conditions | Specific sales report shown. |
| Author | Bc220200976 |
| Exception | Report not found. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | View Sold Products |
| FR-ID | FR-023 |
| Actors | Admin |
| Description | Shows list of sold products and quantities. |
| Alternative Path | Product data not found. |
| Pre Conditions | Orders placed. |
| Action | 1. Admin opens product report2. System compiles sold item list3. Shows quantities and sales. |
| Post Conditions | Admin views sold product data. |
| Author | Bc220200976 |
| Exception | Data fetch failure. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Manage Online Orders |
| FR-ID | FR-024 |
| Actors | Admin |
| Description | Admin manages all orders.Includes: Update Order Status, Apply Discounts, Generate Invoices. |
| Alternative Path | Item Out of stock. |
| Pre Conditions | Admin logged in. |
| Action | 1. Admin views orders.  2. Admin updates status or applies discounts.  3. Admin generates invoices.  4. System updates records accordingly. |
| Post Conditions | Orders processed. |
| Author | Bc220200976 |
| Exception | Product unavailable. |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Update Order Status |
| FR-ID | FR-025 |
| Actors | Admin |
| Description | Admin updates the progress/status of an order. |
| Alternative Path | Update multiple orders at once, Revert to previous status |
| Pre Conditions | Order exists. |
| Action | 1. Admin selects order.  2. Admin changes status.  3. System saves update. |
| Post Conditions | Status changed. |
| Author | Bc220200976 |
| Exception | Invalid order ID or status, Unauthorized access attempt |
| Modification History | Version V1.1 |

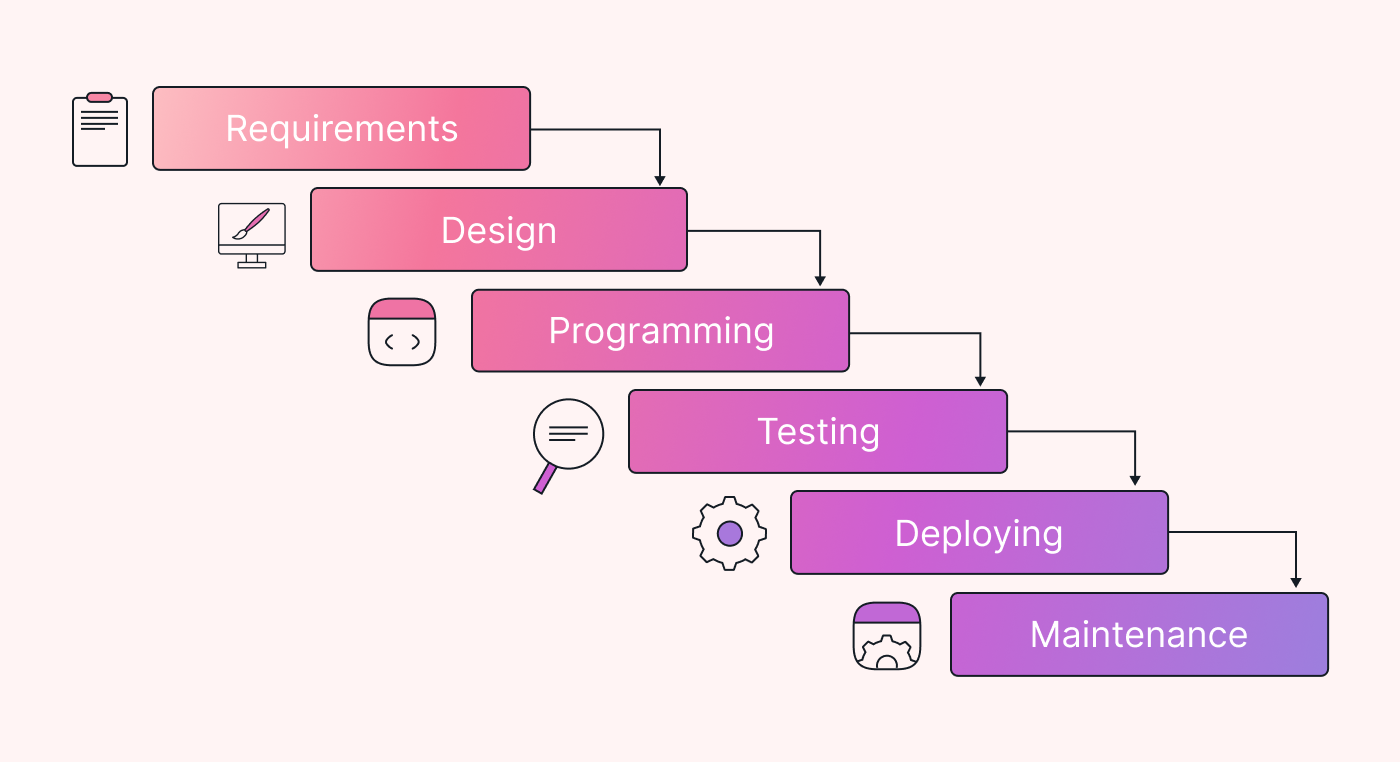
|  |  |
| --- | --- |
| Use Case | Manage Customers |
| FR-ID | FR-026 |
| Actors | Admin |
| Description | Admin manages customer accounts. |
| Alternative Path | Disable account instead of delete, Reset customer password |
| Pre Conditions | Admin logged in. |
| Action | 1. Admin views customers.  2. Admin updates or deletes accounts. |
| Post Conditions | Customer data updated. |
| Author | Bc220200976 |
| Exception | Customer not found, Duplicate email during update |
| Modification History | Version V1.1 |

|  |  |
| --- | --- |
| Use Case | Logout |
| FR-ID | FR-027 |
| Actors | Customer, Admin |
| Description | Ends user session securely. |
| Alternative Path | Cancel logout, Auto logout after inactivity |
| Pre Conditions | User logged in. |
| Action | 1. User clicks logout  .2. System terminates session  .3. User redirected to homepage. |
| Post Conditions | Session ended. |
| Author | Bc220200976 |
| Exception | Session expired, Server error during logout |
| Modification History | Version V1.1 |

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:-**

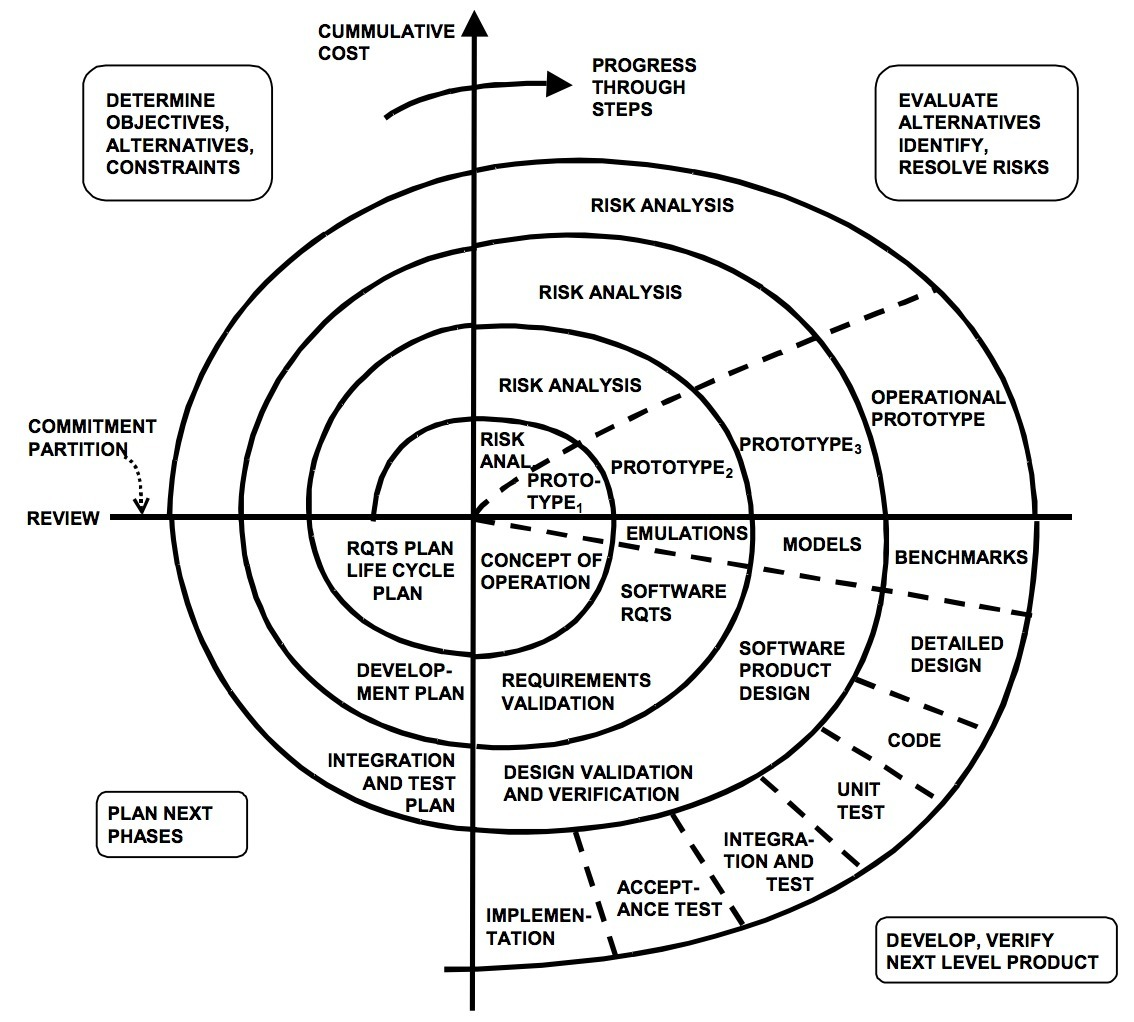
In the Deployment phase, the focus is on ensuring a smooth transition of the software from development to production, and making it available to end-users. This phase involves final testing, packaging, distribution, installation, configuration, user training, and documentation. Once the software is successfully 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:-**

