



Daffodil *International* **University**

Final SRS V-5

Course Code : CSE-236

Course Title : Software Project II

Submitted To

Mr. Golam Rabbany
Lecturer,
Department of CSE,
Daffodil International University

Submitted By

Md. Rakib Hasan

ID: 221-15-5225

Sec :61_ V

Department of CSE

Daffodil International University

Date of submission : 28-11-2023

Software Requirements Specification (SRS) for Fivestar Tech

Table of Contents

Software Requirements Specification (SRS) for Fivestar Tech.....	2
1. INTRODUCTION	6
1.1 PURPOSE	6
1.2 SCOPE	7
1.3 DEFINITIONS, ACRONYMS, AND ABBREVIATIONS.....	7
1.5 OVERVIEW	7
2. OVERALL DESCRIPTION	7
2.1 PRODUCT PERSPECTIVE.....	7
2.2 PRODUCT FEATURES.....	7
2.3 USER CLASSES AND CHARACTERISTICS	8
2.4 OPERATING ENVIRONMENT	8
2.5 DESIGN AND IMPLEMENTATION CONSTRAINTS.....	8
2.6 USER DOCUMENTATION	8
2.7 ASSUMPTIONS AND DEPENDENCIES.....	8
3.SPECIFIC REQUIREMENTS	8
3.1 EXTERNAL INTERFACE REQUIREMENTS.....	9
3.1.2 HARDWARE INTERFACES	9
3.1.3 SOFTWARE INTERFACES	9
3.1.4 COMMUNICATION INTERFACES	9
3.2 FUNCTIONAL REQUIREMENTS	9
3.2.1. USER REGISTRATION.....	9
3.2.2.PRODUCT BROWSING	9
3.2.3.SHOPPING CART	10
3.2.4.CHECKOUT AND PAYMENT	10
3.2.5.ORDER MANAGEMENT	10
3.2.6.CUSTOMER SUPPORT.....	10

3.2.7.ADMINISTRATOR FEATURES	10
3.3 NON-FUNCTIONAL REQUIREMENTS	10
4.Software Development Life Cycle (SDLC).....	12
4.1 Project Planning:	12
4.2 Feasibility Study:.....	13
4.3 System Design:	13
4.4 Implementation (Coding) :	13
4.5 Testing:	13
4.6 Deployment:.....	13
4.7 Maintenance and Support:.....	14
4.8 Documentation:.....	14
4.9 Training:.....	14
4.10 Monitoring and Optimization:.....	14
4.11The most important and popular SDLC models followed in the industry –.....	14
4.11.1.SDLC - Waterfall Model	15
4.11.2.SDLC - Iterative Model.....	16
4.11.3.SDLC - Spiral Model	18
4.11.4.SDLC - V-Model.....	20
4.11.5.SDLC - Big Bang Model	22
5.APPROPRIATE MODEL FOR MY PROJECT	24
SDLC - Waterfall Model SDLC -	24
6. Choosing Right Programming Language:.....	27
6.1.1 Understand Project Requirements:.....	27
6.1.2 Evaluate Developer Skills and Team Expertise:	27
6.1.3 Security Considerations:.....	27
6.1.4 Consider Cost and Licensing:.....	27
6.1.5 Future Trends and Industry Adoption:	27
6.2 Explore some popular programming languages	28
6.1.3 Which is the best programming language?	28
7. Software Testing:.....	29
7.1.1 Types of Testing:.....	29
7.1.2 Levels of Testing:	29
7.1.3 Testing Methods:.....	29

7.1.4 Testing Process:	30
7.1.5 Testing Tools:	30
7.1.6 Challenges in Software Testing:.....	30
7.1.7 Best Practices:	30
7.1.8 Agile Testing:	30
7.2 Testing Types:	30
7.3 Testing Techniques:	31
7.4 Testing Tactics:.....	31
7.5 Measurement in Software Testing	31
7.6 Collecting Test Data.....	32
7.7 Benefits of Measurement	32
7.8 Quality Assurance:	32
7.9 Risk Management:	32
8.Font end Language	33
8.1.HTML (HyperText Markup Language):	33
8.2.CSS (Cascading Style Sheets):.....	33
8.3.React:	33
8.4.Angular:.....	34
8.5.Vue.js:	34
8.6.Bootstrap:.....	34
8.7.Sass (Syntactically Awesome Stylesheets):	34
9.Back End Language	34
9.1.Node.js:	34
9.2.Python:	35
9.3.Ruby:	35
9.4.Java:.....	35
9.5.PHP:.....	35
9.6.Go (Golang):	35
10.Database	36
Relational Databases:	36
10.1.MySQL:	36
PostgreSQL:	36
10.2.NoSQL Databases:	36

MongoDB	36
Cassandra:	36
10.3.Graph Databases:	36
Neo4	36
10.4.In-Memory Databases:.....	36
Redis:.....	36
10.5.Columnar Databases:	36
10.6.Time-Series Databases:	36
InfluxDB:	36
11.Memory Size	37
11.1.Web Server Memory:	37
11.2.Application Server Memory:	37
11.3.Database Memory:.....	37
11.4.Caching:.....	37
11.5.Content Delivery:	37
11.6.Scaling:	37
11.7.Monitoring and Analytics:.....	38
11.8.Server Infrastructure:	38
12. Choosing the Right Technology Stack:	38
12.1. Understand Project Requirements:.....	38
12.2. Evaluate Developer Skills and Team Expertise:	38
12.3. Security Considerations:.....	39
12.4. Consider Cost and Licensing:.....	39
12.5. Future Trends and Industry Adoption:	39
12.6. Explore Some Popular Technologies:	39
12.7. Making the Decision:.....	39
12.8 Testing Types for Online Shopping Management:.....	39
12.9. Testing Techniques:	40
12.10. Testing Tactics:.....	40
12.11. Proposed Testing Process:.....	40
12.12 Measurement in Software Testing:	40
12.13 Collecting Test Data:	41
12.14 Benefits of Measurement:	41

12.15 Quality Assurance:.....	41
13.Language choosing:	42
13.1.HTML	42
13.2.CSS.....	42
13.3.JavaScript.....	43
14.The Technology Stack	44
14.1. The most popular programming languages for software development	44
15.Technical Process and Methodology:.....	46
16.Reporting Table:	46
16.1.Performance Requirements	46
16.2.Safety Requirements	46
16.3.Security Requirements	46
16.4.Software Quality Attributes.....	47
17.Deployment	47
17.1.Development Overview	47
17.2.Deployment Steps	47
17.3.Rollback Plan	48
17.4.Post-Deployment Activities	48
17.5.Deployment Resources	48
17.6.Considerations.....	49
17.7.Deployment Sign-Off.....	49
Contact Us.....	49

1. INTRODUCTION

1.1 PURPOSE

The purpose of this Software Requirements Specification (SRS) is to define the requirements for the Star Tech Online Shopping website, hereinafter referred to as "Star Tech." This document serves as a reference for the development and testing teams to create a high-quality online shopping platform.

1.2 SCOPE

The scope of this SRS includes the features, functionalities, and constraints of the Star Tech Online Shopping website available at [\[https://www.startech.com.bd/\]](https://www.startech.com.bd/)(<https://www.startech.com.bd/>).

1.3 DEFINITIONS, ACRONYMS, AND ABBREVIATIONS

- SRS: Software Requirements Specification
- URL: Uniform Resource Locator

1.4 REFERENCES

- [Star Tech Online Shopping](<https://www.startech.com.bd/>)

1.5 OVERVIEW

Star Tech is an online shopping platform that offers a wide range of electronics and technology-related products. This SRS document outlines the requirements necessary to maintain and enhance the website's functionality and user experience.

2. OVERALL DESCRIPTION

2.1 PRODUCT PERSPECTIVE

Star Tech is a standalone online shopping platform. It interfaces with various external payment gateways and logistics services to facilitate online transactions and product deliveries.

2.2 PRODUCT FEATURES

The key features of Star Tech include but are not limited to:

- User registration and authentication
- Browsing and searching for products
- Adding products to a shopping cart
- Online payment processing

- Order tracking and history
- Customer support and feedback

2.3 USER CLASSES AND CHARACTERISTICS

The primary user classes for Star Tech are:

1. Guests: Users who visit the website without logging in.
2. Registered Users: Users who create accounts on the website.
3. Administrators: Website administrators who manage products, users, and orders.

2.4 OPERATING ENVIRONMENT

Star Tech operates in a web-based environment, accessible through standard web browsers. It is compatible with common operating systems such as Windows, macOS, and Linux.

2.5 DESIGN AND IMPLEMENTATION CONSTRAINTS

Star Tech is designed to be responsive and accessible on various devices, including smartphones, tablets, and desktop computers. It must adhere to web development best practices and be compatible with modern web browsers.

2.6 USER DOCUMENTATION

User documentation, including a user guide and FAQs, will be provided on the website to assist users in navigating and using the platform effectively.

2.7 ASSUMPTIONS AND DEPENDENCIES

The successful operation of Star Tech depends on external services for payment processing and order fulfillment. It assumes a stable internet connection for users.

3.SPECIFIC REQUIREMENTS

3.1 EXTERNAL INTERFACE REQUIREMENTS

3.1.2 HARDWARE INTERFACES

- Star Tech shall not require any specific hardware interfaces beyond standard input devices (e.g., keyboard, mouse) and display devices (e.g., monitor, screen).

3.1.3 SOFTWARE INTERFACES

- Star Tech shall integrate with external payment gateways for processing payments securely.
- Integration with logistics and shipping services may be required to track and fulfill orders.

3.1.4 COMMUNICATION INTERFACES

- The website shall use secure communication protocols (e.g., HTTPS) to protect user data during transmission.

3.2 FUNCTIONAL REQUIREMENTS

The functional requirements specify the interactions and capabilities of the Star Tech website, including but not limited to:

3.2.1. USER REGISTRATION

- Users can create accounts with valid email addresses and passwords.
- User profile information can be updated.

3.2.2. PRODUCT BROWSING

- Users can browse products by categories and search for specific items.
- Detailed product pages include descriptions, images, and prices.

3.2.3.SHOPPING CART

- Users can add and remove items from their shopping carts.
- Cart contents are saved for registered users.

3.2.4.CHECKOUT AND PAYMENT

- Users can proceed to checkout from their shopping carts.
- Payment options include credit/debit cards and other supported methods.

3.2.5.ORDER MANAGEMENT

- Users can view order status and history.
- Registered users can track orders.

3.2.6.CUSTOMER SUPPORT

- Users can contact customer support for inquiries and assistance.

3.2.7.ADMINISTRATOR FEATURES

- Administrators can manage product listings and inventory.
- User accounts can be managed, including suspending or banning users if necessary.

3.3 NON-FUNCTIONAL REQUIREMENTS

3.3.1 PERFORMANCE REQUIREMENTS

- The website shall load product pages within 2 seconds on average.
- Payment processing shall be completed within 10 seconds.
- The system shall support concurrent users without significant performance degradation.

3.3.2 SECURITY REQUIREMENTS

- User passwords shall be securely hashed and stored.
- Payment information shall be encrypted during transmission.
- Access to sensitive administrator functions shall require authentication and authorization.

3.3.3 RELIABILITY REQUIREMENTS

- The website shall have a system for regular backups and disaster recovery.
- It shall have an uptime of at least 99.9% per month.

3.3.4 USABILITY AND USER EXPERIENCE REQUIREMENTS

- The website shall be intuitive and user-friendly.
- It shall be accessible to users with disabilities in compliance with accessibility standards.

3.3.5 LEGAL AND COMPLIANCE REQUIREMENTS

- The website shall comply with applicable data protection and privacy laws.
- Terms and conditions, as well as a privacy policy, shall be accessible to users.

This Software Requirements Specification serves as a foundation for the development and testing of the Star Tech Online Shopping website. It will be subject to updates and revisions as necessary throughout the development process.

Contact

Md. Rakib Hasan

Cell: 01518696023

E-mail: rakibhase@gmail.com

By Post

Khagan, Birulia, Savar, Dhaka-1206

We look forward to hearing from you soon!

Client Signature

Kabir Khan

Managing Director

Fivestar Tech

4. Software Development Life Cycle (SDLC)

The SDLC is a process used by software developers to design, develop, test, and deploy software. Here are the key stages:

4.1 Project Planning:

Define the scope, objectives, and requirements of the online shopping management system.

Identify the target audience and market.

Create a project plan outlining tasks, timelines, and resource requirements.

Define the budget and allocate resources.

4.2 Feasibility Study:

Conduct a feasibility study to ensure the project is viable.

Evaluate technical, operational, and economic factors.

Identify potential risks and propose mitigation strategies.

4.3 System Design:

Create a detailed system design based on the requirements.

Design the architecture, database structure, and user interfaces.

Specify hardware and software requirements.

Plan for security measures to protect user data.

4.4 Implementation (Coding) :

Develop the online shopping management system according to the design specifications.

Write code in accordance with coding standards.

Conduct regular code reviews to ensure quality.

4.5 Testing:

Perform unit testing to check the functionality of individual components.

Conduct integration testing to ensure different modules work together.

Perform system testing to validate the entire system against requirements.

Identify and fix bugs or issues.

4.6 Deployment:

Deploy the online shopping management system in a controlled environment.

Monitor the system for any performance issues or errors.

Implement a rollback plan in case of deployment failures.

4.7 Maintenance and Support:

Provide ongoing maintenance and support to address any issues or bugs that arise post-deployment.

Implement updates and improvements based on user feedback.

Monitor system performance and security regularly.

4.8 Documentation:

Create comprehensive documentation for the online shopping management system, including user manuals and technical documentation.

Document the code to facilitate future maintenance and updates.

4.9 Training:

Provide training to end-users and support staff on how to use the system.

Create training materials and documentation.

4.10 Monitoring and Optimization:

Implement monitoring tools to track system performance and user activity.

Optimize the system based on usage patterns and feedback.

Plan for future updates and enhancements.

4.11 The most important and popular SDLC models followed in the industry –

i. Waterfall Model

ii. Iterative Model

iii. Spiral Model

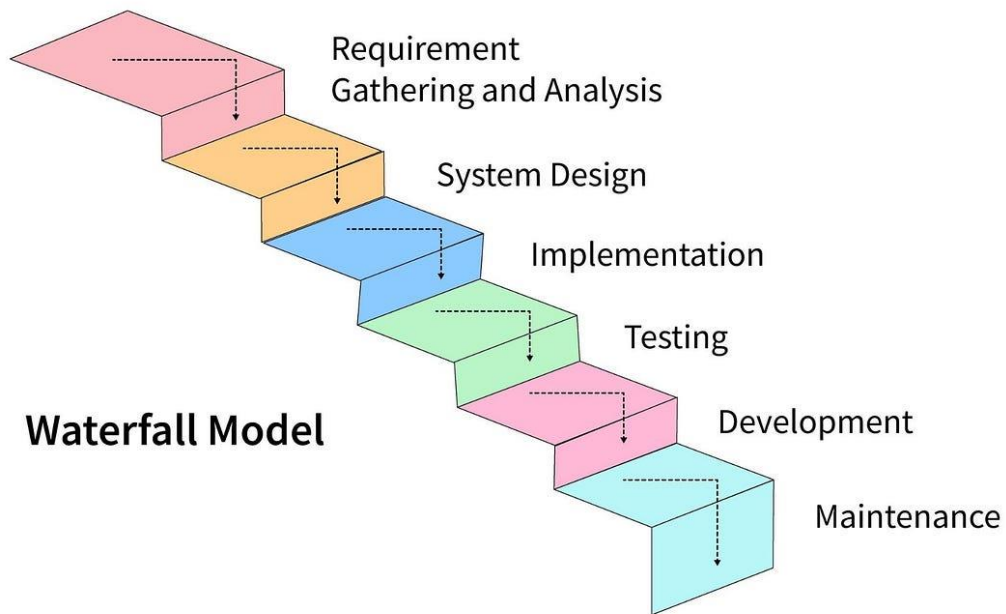
iv. V-Model

v. Big Bang Model

4.11.1.SDLC - Waterfall Model

The Waterfall Model was the first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use.

Waterfall Model – Design:



Waterfall Model – Application:

- Product definition is stable.
- Technology is understood and is not dynamic.

- There are no ambiguous requirements.
- The project is short.

Waterfall Model – Advantages:

- Simple and easy to understand and use
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented

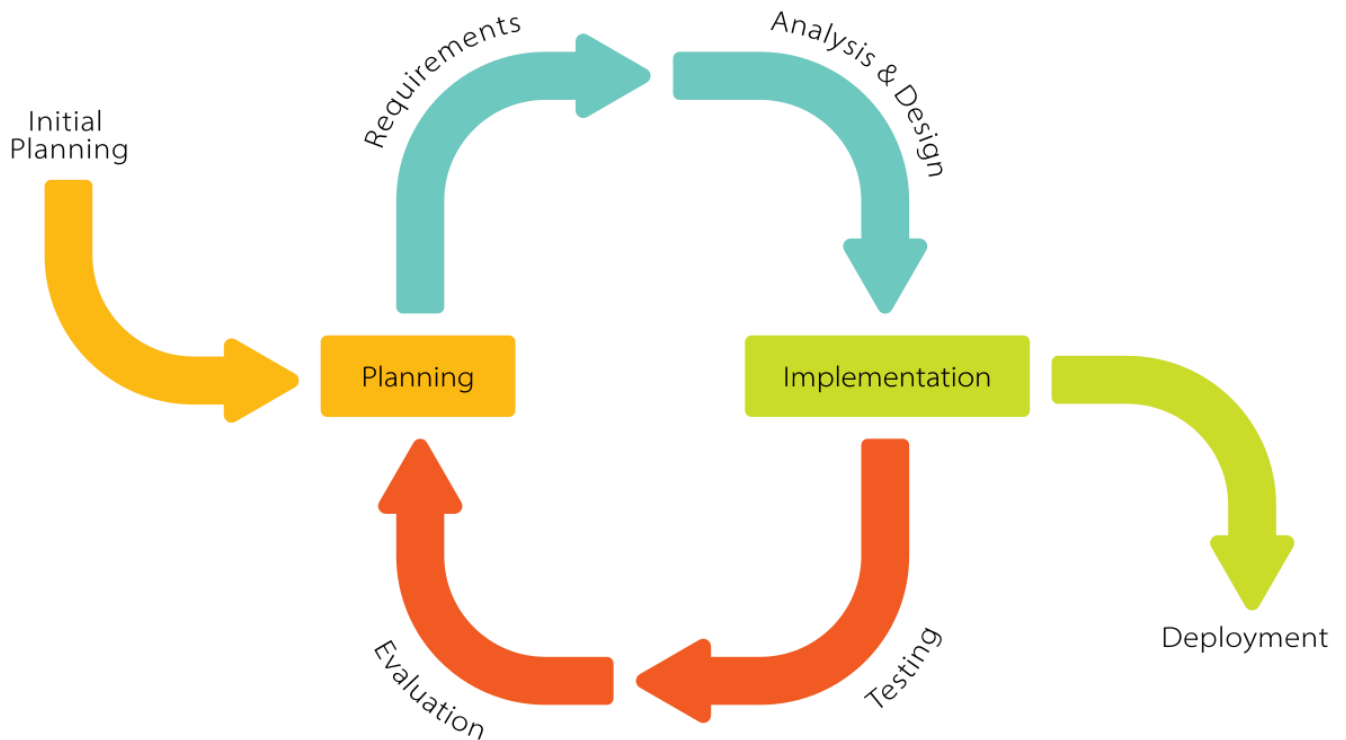
Waterfall Model – Disadvantages:

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.

4.11.2.SDLC - Iterative Model

In the Iterative model, iterative process starts with a simple implementation of a small set of the software requirements.

Iterative Model – Design:



Iterative Model - Application

- Requirements of the complete system are clearly defined and understood.
- Major requirements must be defined; however, some functionalities or requested enhancements may evolve with time.
- There is a time to the market constraint.

The advantages of the Iterative:

- Results are obtained early and periodically.
- Parallel development can be planned.
- Progress can be measured.
- Less costly to change the scope/requirements.
- Testing and debugging during smaller iteration is easy.
- Risks are identified and resolved during iteration.

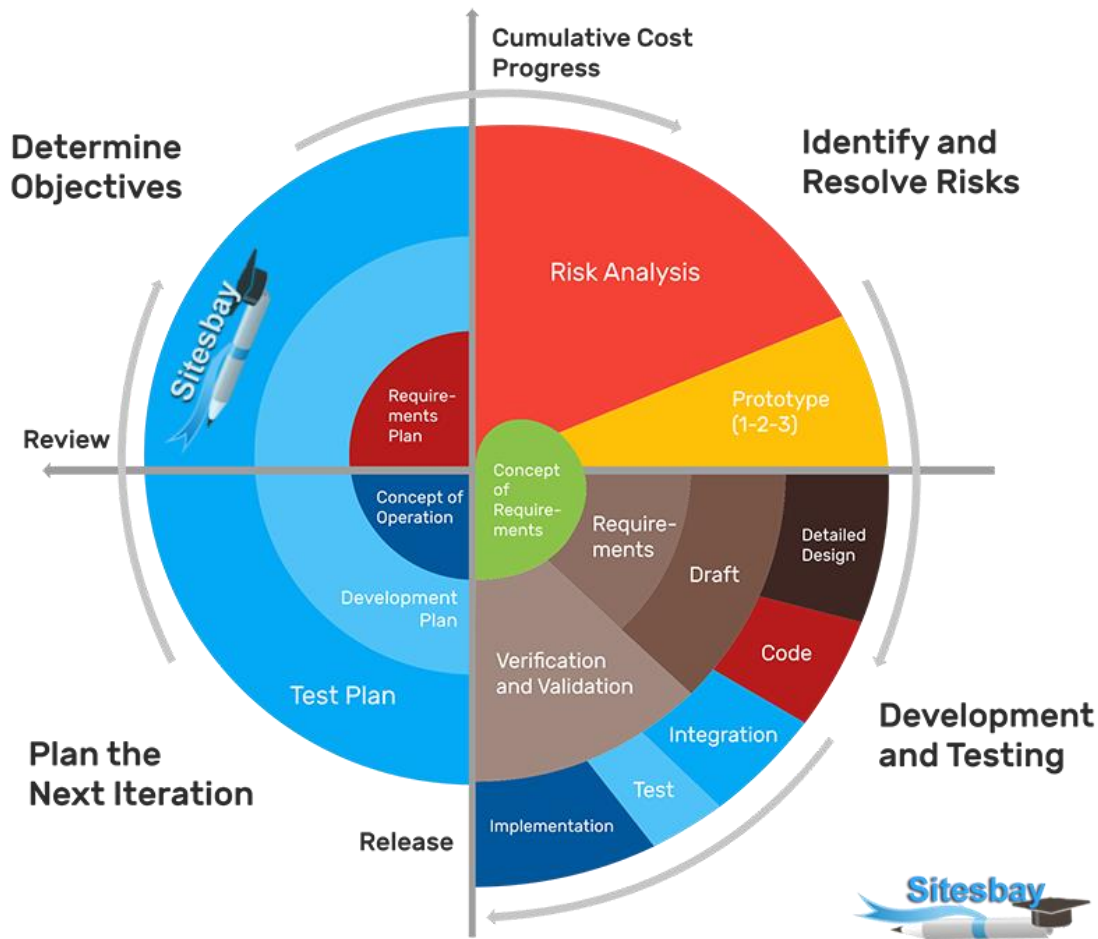
The disadvantages of the Iterative:

- More resources may be required.
- Although cost of change is lesser, but it is not very suitable for changing requirements.
- More management attention is required

4.11.3.SDLC - Spiral Model

The spiral model combines the idea of iterative development with the systematic, controlled aspects of the waterfall model. This Spiral model is a combination of iterative development process model and sequential linear development model.

Spiral Model – Design:



Spiral Model Application

- When there is a budget constraint and risk evaluation is important.
- For medium to high-risk projects.
- Long-term project commitment because of potential changes to economic priorities as the requirements change with time.
- Customer is not sure of their requirements which is usually the case.

The advantages of the Spiral SDLC:

- Changing requirements can be accommodated.

- Allows extensive use of prototypes.
- Requirements can be captured more accurately.
- Users see the system early.

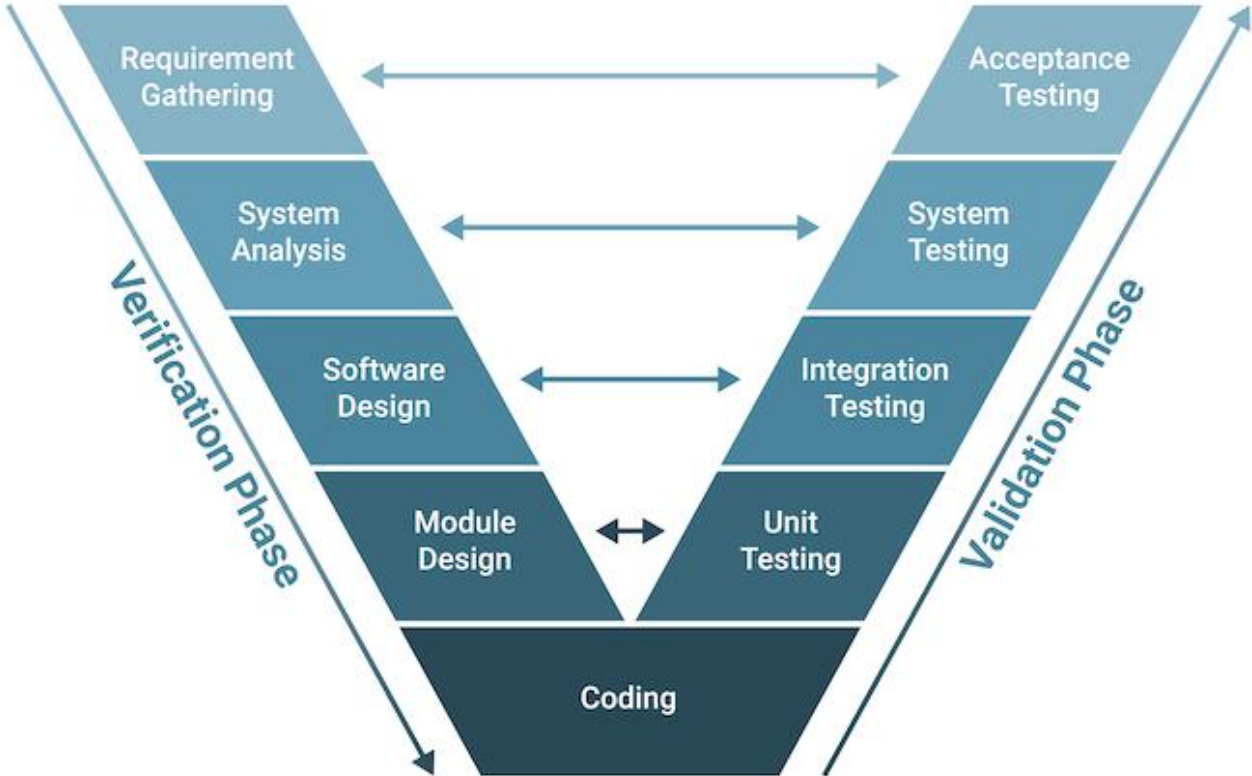
The disadvantages:

- Management is more complex.
- End of the project may not be known early.
- Not suitable for small or low risk projects and could be expensive for small projects.
- Process is complex.
- Spiral may go on indefinitely.

4.11.4.SDLC - V-Model

The V-model is an SDLC model where execution of processes happens in a sequential manner in a V-shape. It is also known as Verification and Validation model.

V-Model – Design:



V- Model – Application

- Requirements are well defined, clearly documented and fixed.
- Product definition is stable.
- Technology is not dynamic and is well understood by the project team.
- There are no ambiguous or undefined requirements.
- The project is short.

The advantages:

- This is a highly-disciplined model and Phases are completed one at a time.
- Works well for smaller projects where requirements are very well understood.

- Simple and easy to understand and use.
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.

The disadvantages:

- High risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects

4.11.5.SDLC - Big Bang Model

The Big Bang model is an SDLC model where we do not follow any specific process.



Big Bang Model – Design and Application

The Big Bang Model comprises of focusing all the possible resources in the software development and coding, with very little or no planning. The requirements are understood and implemented as they come. Any changes required may or may not need to revamp the complete software.

The advantages

- This is a very simple model
- Little or no planning required
- Easy to manage
- Very few resources required
- Gives flexibility to developers

- It is a good learning aid for new comers.

The disadvantages

- Very High risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Can turn out to be very expensive if requirements

5.APPROPRIATE MODEL FOR MY PROJECT

SDLC - Waterfall Model

SDLC - Waterfall Model SDLC -

There are many reasons for use this model .one of the best reasons it is easy to use and it is helpful for small project

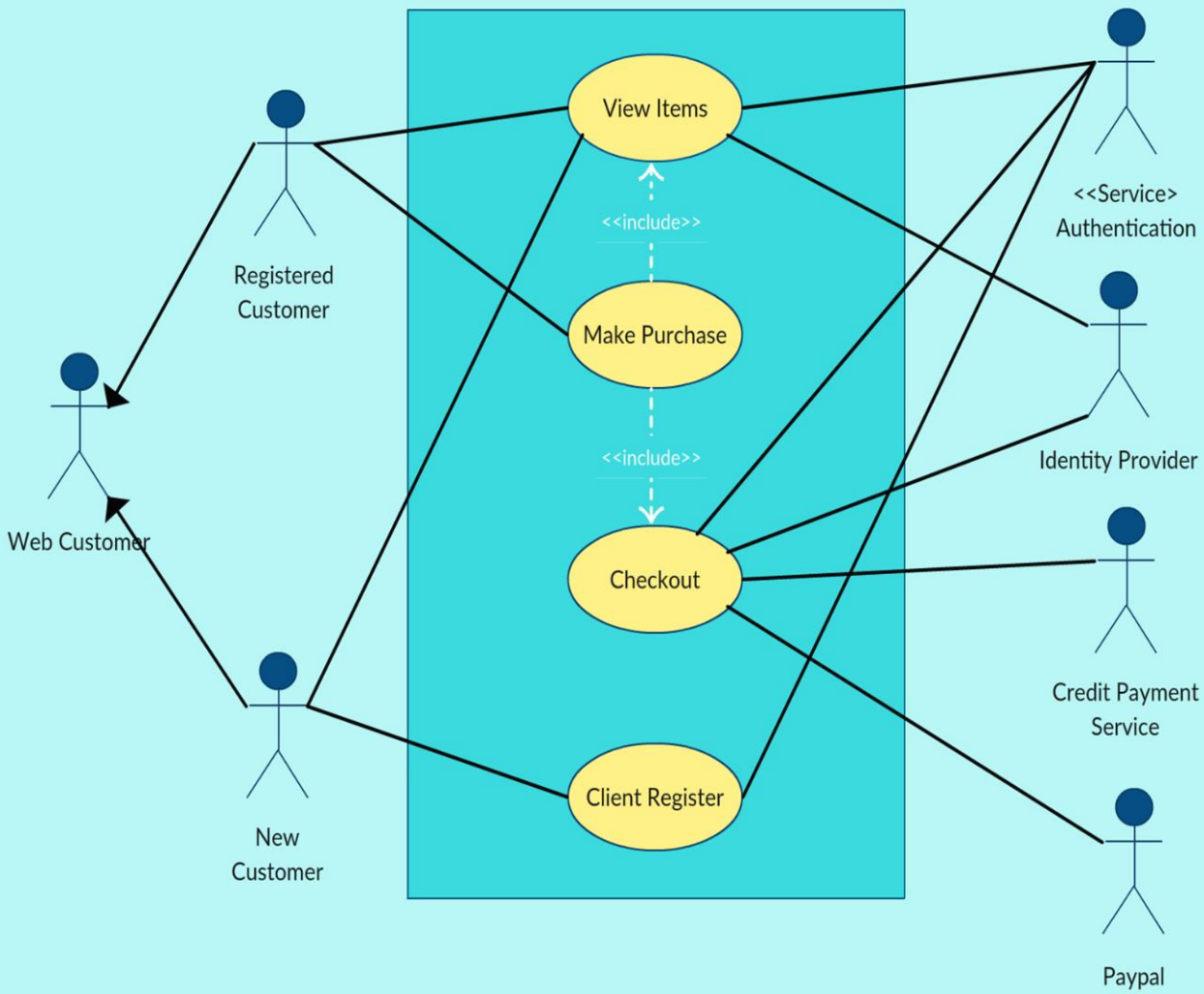
Some Advantage of this mode which is properly follow my project this are given below.

- Simple and easy to understand and use
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented

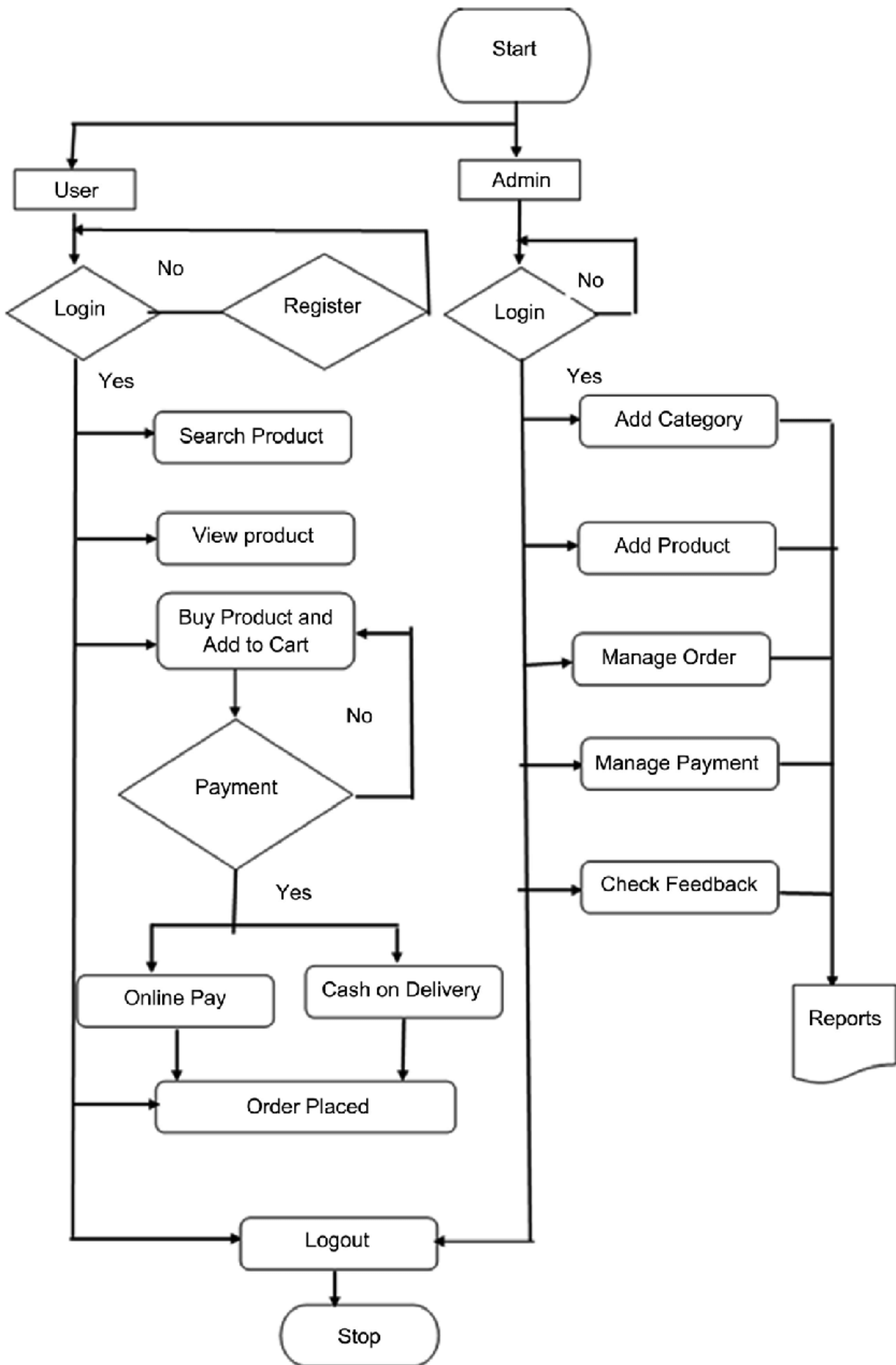
Therefore, I want to use this model for my project.

USE CASE DIAGRAM

Online Shopping System



FLOWCHART DIAGRAM



6. Choosing Right Programming Language:

Choosing the best programming language for web development depends on various factors, including project requirements, team expertise, scalability needs, and personal preferences. Here's a step-by-step guide to help you select the most suitable programming language for your web development project:

6.1.1 Understand Project Requirements:

- **Define the Project Scope:** Clearly outline the goals and features of your web application. Consider whether it's a content-based site, an e-commerce platform, a real-time application, or something else. Different projects may benefit from different languages.

- **Consider Front-End and Back-End Needs:** Identify the level of interactivity required on the client side (front-end) and the server-side (backend). Some languages are better suited for specific tasks, so understanding these requirements is crucial.

6.1.2 Evaluate Developer Skills and Team Expertise:

- **Assess Existing Skills:** Consider the expertise of your development team. If your team is already proficient in a particular language, it might be beneficial to stick with that language to leverage existing skills and knowledge.

- **Consider Learning Curve:** If you're starting with a new language, assess the learning curve. Some languages are known for their ease of learning, while others might require more time and effort.

6.1.3 Security Considerations:

- **Assess Security Features:** Different languages have varying levels of builtin security features. Understand the security requirements of your project and choose a language with a strong emphasis on security. For example, languages like Java and C# are known for their security features.

6.1.4 Consider Cost and Licensing:

- **Licensing:** Be aware of the licensing requirements of the language and any associated frameworks. Some languages have open-source licensing, while others may have commercial licenses.

6.1.5 Future Trends and Industry Adoption:

- Industry Trends: Stay informed about current industry trends. While it's essential to choose a language based on your project's needs, considering emerging trends can be valuable for future-proofing your application.

6.2 Explore some popular programming languages

Learning a new web development language requires both time and resources. As a result, it's crucial to think about which languages provide the quickest routes to mastery. Fortunately, programmers are frequently upfront about some languages being easier to learn than others. The following are among the simplest to understand and are among the best programming languages for web development.

1. HTML
2. Python
3. JavaScript
4. PHP
5. Java

6.1.3 Which is the best programming language?

There isn't a universally "best" programming language because the suitability of a language depends on various factors, including the nature of the project, the development team's expertise, and the specific requirements of the task at hand. Different programming languages are designed for different purposes, and each has its strengths and weaknesses. Here are a few popular programming languages, each known for specific use cases:

I choose Javascript programming language cause:

Client-Side Web Development:

- In-Browser Execution: JavaScript is the primary language for client-side web development. It runs directly in web browsers, allowing developers to create dynamic and interactive user interfaces. This capability is crucial for enhancing user experience on websites and web applications.

Versatility:

- Full-Stack Development: JavaScript is not limited to the client side. With the advent of Node.js, developers can use JavaScript for server-side programming as well. This enables full-stack development, where the same language (JavaScript) is used for both front-end and back-end development.

Community Support:

- **Large and Active Community:** JavaScript has one of the largest and most active developer communities. This means a wealth of resources, tutorials, and third-party libraries are available, making it easier for developers to find solutions to problems and stay updated on best practices.

Dynamic Typing and Prototypal Inheritance:

- **Dynamic Typing:** JavaScript is a dynamically-typed language, which means that variables are not bound to a specific data type at compile time. This flexibility can simplify coding and make it more adaptable to changes.

7. Software Testing:

Software testing is a crucial process in the development of software to ensure that it meets the specified requirements, functions correctly, and is free of defects or bugs. The primary goal of software testing is to identify and fix any issues before the software is released to end-users. Here are some key aspects of software testing:

7.1.1 Types of Testing:

- **Manual Testing:** This involves testers manually executing test cases without using any automation tools.
- **Automated Testing:** Involves the use of testing tools and scripts to automate the execution of test cases.

7.1.2 Levels of Testing:

- **Unit Testing:** Testing individual units or components of the software to ensure they work as intended.
- **Integration Testing:** Verifying that different units or components work together when integrated.
- **System Testing:** Testing the entire system to ensure it meets the specified requirements.
- **Acceptance Testing:** Ensuring that the software meets the user's acceptance criteria.

7.1.3 Testing Methods:

- **Black Box Testing:** Testing without knowledge of the internal workings of the system. Focus is on inputs and outputs.
- **White Box Testing:** Testing with knowledge of the internal workings of the system. Focus is on code structure and logic.
- **Gray Box Testing:** A combination of black box and white box testing.

7.1.4 Testing Process:

- Test Planning: Defining the testing approach, scope, resources, and schedule.
- Test Design: Creating test cases based on requirements and specifications.
- Test Execution: Running the test cases and capturing the results.
- Defect Tracking: Identifying and logging defects or issues. • Test Reporting: Communicating the test results and status to stakeholders.

7.1.5 Testing Tools:

- Test Management Tools: Help manage and organize test cases, test execution, and test results.
- Automation Testing Tools: Assist in automating the execution of test cases to improve efficiency.
- Performance Testing Tools: Evaluate the performance and scalability of the software.
-

7.1.6 Challenges in Software Testing:

- Incomplete Requirements: Lack of clear and complete requirements can make testing challenging.
- Changing Requirements: Changes in requirements during the development process can impact testing.
- Time and Budget Constraints: Limited time and budget can affect the thoroughness of testing.

7.1.7 Best Practices:

- Early Testing: Start testing as early as possible in the development life cycle.
- Test in Realistic Environment: Test the software in an environment that closely resembles the production environment.
- Regression Testing: Perform regression testing to ensure that new changes do not negatively impact existing functionality.

7.1.8 Agile Testing:

In agile development, testing is often integrated into the development process with frequent and iterative testing cycles.

7.2 Testing Types:

Functional Testing:

- Unit Testing: Testing individual units or components in isolation.
- Integration Testing: Testing the interaction between integrated components or systems.
- System Testing: Evaluating the entire system's functionality as a whole.
- Acceptance Testing: Confirming that the system meets user acceptance criteria.

Non-functional Testing:

- Performance Testing: Assessing the system's responsiveness, scalability, and speed.
- Security Testing: Identifying vulnerabilities and ensuring data integrity and confidentiality.
- Usability Testing: Evaluating the user interface and overall user experience.
- Reliability Testing: Checking the system's ability to perform consistently under various conditions.

7.3 Testing Techniques:

Static Testing:

- Code Review: Manual examination of the source code to identify issues.
- Static Analysis: Automated analysis of the code without executing it.

Dynamic Testing:

- Dynamic Analysis: Analyzing the software during execution.
- Dynamic Testing Tools: Tools that assess the software as it runs.

7.4 Testing Tactics:

Regression Testing:

Ensuring that new changes don't negatively impact existing functionality.

Exploratory Testing:

Simultaneously designing and executing test cases based on real-time analysis.

Propose Testing Process

Test Planning:

Objective:

- Define the overall goals and scope of the testing effort.
- Identify resources, including personnel and testing tools.
- Establish a timeline for testing activities.

Activities:

- Develop a Test Plan document outlining the testing strategy.
- Identify testing environments and configurations.
- Define entry and exit criteria for each testing phase.
- Allocate responsibilities to team members.

7.5 Measurement in Software Testing

Test Coverage:

Objective:

- Measure the extent to which the source code or requirements have been tested.

Metrics:

- Statement Coverage: Percentage of code statements executed by tests.
- Branch Coverage: Percentage of decision points exercised by tests.
- Path Coverage: Percentage of code paths tested.

7.6 Collecting Test Data

Understand Requirements:

Objective:

- Gain a clear understanding of the testing requirements to determine the types of data needed.

Activities:

- Review functional and non-functional requirements.
- Identify data dependencies and constraints specified in the requirements.

7.7 Benefits of Measurement

Assessment of Progress:

Objective:

- Evaluate the progress of the project or specific activities within it.

Benefits:

- Enables tracking of project milestones and completion of tasks.
- Provides a quantitative measure of progress against timelines.

7.8 Quality Assurance:

Objective:

- Assess and ensure the quality of the software or product.

Benefits:

- Identifies defects and issues early in the development lifecycle.
- Supports proactive measures to improve software quality.

7.9 Risk Management:

Objective:

- Identify and mitigate potential risks in the project.

Benefits:

- Helps in early detection of high-risk areas.
- Facilitates informed decision-making to address risks promptly.

Effective software testing is essential for delivering high-quality software that meets user expectations and operates reliably in various environments. It is an ongoing process that evolves with the development life cycle and requires collaboration between developers and testers.

modify this information for a tech online shopping managements

ChatGPT

Certainly! Let's tailor the information for a tech online shopping management system.

8. Font end Language

Choosing a front-end language for a technical website depends on various factors such as your team's expertise, project requirements, and your target audience. Here are some commonly used front-end languages for building technical websites:

8.1. HTML (HyperText Markup Language):

Description: HTML is the standard markup language for creating the structure of web pages.

Use Case: Essential for creating the basic structure of your website, defining headings, paragraphs, lists, links, and other elements.

8.2. CSS (Cascading Style Sheets):

Description: CSS is used for styling and layout. It allows you to control the presentation of HTML elements.

Use Case: Crucial for making your website visually appealing and responsive. It handles aspects like colors, fonts, spacing, and layout.

JavaScript:

Description: JavaScript is a versatile scripting language used to add interactivity and dynamic features to web pages.

Use Case: Essential for creating interactive elements, handling user input, and making your website more dynamic. It's often used in conjunction with HTML and CSS.

8.3. React:

Description: A JavaScript library for building user interfaces, developed by Facebook.

Use Case: Ideal for building complex, dynamic, and interactive user interfaces. React is component-based, making it easy to manage and update different parts of your application independently.

8.4.Angular:

Description: A TypeScript-based framework developed by Google for building web applications.

Use Case: Suitable for building large-scale, feature-rich applications. It provides a comprehensive structure for organizing code and offers tools for building robust, maintainable web applications.

8.5.Vue.js:

Description: A progressive JavaScript framework for building user interfaces.

Use Case: Known for its simplicity and ease of integration. Vue.js is suitable for both small-scale projects and more complex applications.

8.6.Bootstrap:

Description: A front-end framework that simplifies the design process using pre-built components and a responsive grid system.

Use Case: Great for quickly prototyping and developing responsive, mobile-first websites. It's particularly useful if you want a consistent look and feel without spending too much time on design.

8.7.Sass (Syntactically Awesome Stylesheets):

Description: An extension of CSS that adds features like variables, nesting, and mixins.

Use Case: Improves the maintainability and organization of your stylesheets. Particularly useful for larger projects where complex styling is involved.

9.Back End Language

9.1.Node.js:

Description: JavaScript runtime built on Chrome's V8 JavaScript engine. It allows you to use JavaScript for server-side scripting.

Use Case: Particularly well-suited for building scalable network applications. It's commonly used for real-time applications and APIs.

9.2.Python:

Description: A versatile and readable programming language with a large ecosystem of libraries and frameworks.

Use Case: Widely used for web development. Popular frameworks like Django and Flask make it easy to build robust and scalable web applications.

9.3.Ruby:

Description: A dynamic, object-oriented programming language with a focus on simplicity and productivity.

Use Case: Often used with the Ruby on Rails framework, which provides a convention over configuration approach, making it efficient for building web applications.

9.4.Java:

Description: A widely-used, class-based, object-oriented programming language.

Use Case: Java is known for its portability and scalability. It's commonly used in enterprise-level applications. Spring Boot is a popular framework for building Java-based web applications.

9.5.PHP:

Description: A server-side scripting language designed for web development.

Use Case: Widely used for building dynamic websites. PHP is often embedded in HTML and works well with databases. Popular frameworks include Laravel and Symfony.

9.6.Go (Golang):

Description: A statically typed language developed by Google, known for its efficiency and simplicity.

Use Case: Go is designed for building scalable and performant applications. It's becoming increasingly popular for building web servers and APIs.

10.Database

Relational Databases:

10.1.MySQL: An open-source relational database management system. It is known for its reliability, performance, and ease of use.

PostgreSQL: Another open-source RDBMS that is highly extensible and supports a wide range of data types. It's known for its advanced features and standards compliance.

10.2.NoSQL Databases:

MongoDB: A popular document-oriented NoSQL database. It stores data in flexible, JSON-like documents, making it suitable for projects with rapidly changing schemas or large amounts of unstructured data.

Cassandra: A distributed NoSQL database designed for handling large amounts of data across many commodity servers without any single point of failure. It's particularly well-suited for time-series data.

10.3.Graph Databases:

Neo4j: A graph database that is optimized for storing and querying highly connected data. It's ideal for applications with complex relationships between entities.

10.4.In-Memory Databases:

Redis: An open-source, in-memory data structure store. It can be used as a database, cache, and message broker. Redis is known for its speed and versatility.

10.5.Columnar Databases:

Apache Cassandra: While often categorized as a NoSQL database, Cassandra is specifically designed for handling large amounts of columnar data. It's distributed and highly scalable.

10.6.Time-Series Databases:

InfluxDB: A scalable and high-performance time-series database. It's suitable for storing and querying time-series data generated by IoT devices, monitoring systems, and other applications.

Search Engines:

Elasticsearch: Although primarily used for full-text search, Elasticsearch can also function as a NoSQL document store. It's commonly used for applications that require powerful search capabilities.

11.Memory Size

11.1.Web Server Memory:

The memory required by your web server (e.g., Apache, Nginx) depends on the number of concurrent connections, the size of static files, and the complexity of dynamic content generation.

11.2.Application Server Memory:

If your website uses a server-side application framework (e.g., Node.js, Django, Ruby on Rails), the memory requirements will depend on the size and complexity of your application, as well as the number of simultaneous users.

11.3.Database Memory:

The memory requirements for your database server depend on the size of your database, the complexity of queries, and the number of concurrent database connections. In-memory databases may have higher memory requirements but can provide faster access.

11.4.Caching:

If you implement caching mechanisms (e.g., Redis, Memcached) to optimize performance, you'll need to allocate memory for caching purposes. The amount of memory will depend on the size and frequency of cached data.

11.5.Content Delivery:

If you use a Content Delivery Network (CDN) to deliver static content, the memory requirements on your server may be reduced, but you'll need to consider the CDN costs and configuration.

11.6.Scoring:

Consider whether your website needs to scale horizontally (adding more servers) or vertically (upgrading server resources). Vertical scaling may involve increasing the memory on a single server, while horizontal scaling may require distributing the load across multiple servers.

11.7. Monitoring and Analytics:

Implement monitoring tools to track memory usage over time. This will help you identify patterns, optimize resource allocation, and plan for scaling as your website grows.

11.8. Server Infrastructure:

Cloud service providers often offer different types of virtual machines or containers with varying amounts of memory. Choose a configuration that aligns with your estimated requirements and allows for future growth.

12. Choosing the Right Technology Stack:

12.1. Understand Project Requirements:

Define the E-commerce Scope: Clearly outline the goals and features of your online shopping platform. Consider whether it's a multi-vendor marketplace, a single-vendor store, or a combination of both.

Consider Front-End and Back-End Needs: Identify the interactivity required on the client side (front-end) and the server-side (backend). Different languages and frameworks are optimized for specific e-commerce tasks.

12.2. Evaluate Developer Skills and Team Expertise:

Assess Existing Skills: Consider the expertise of your development team. If they are proficient in a particular language or framework, it might be advantageous to leverage their existing skills.

Consider Learning Curve: Assess the learning curve of new technologies. Opt for languages and frameworks that align with your team's capabilities and project timelines.

12.3. Security Considerations:

Evaluate Security Features: Given the sensitive nature of online shopping, prioritize languages and frameworks with strong built-in security features. Consider the security track record of technologies; for example, languages like Python and frameworks like Django are known for their security.

12.4. Consider Cost and Licensing:

Licensing: Be aware of the licensing requirements of the language and any associated frameworks. Opt for technologies with open-source licensing to manage costs effectively.

12.5. Future Trends and Industry Adoption:

Industry Trends: Stay informed about current trends in e-commerce technology. While meeting project needs is crucial, considering emerging trends can future-proof your online shopping platform.

12.6. Explore Some Popular Technologies:

Learning a new tech stack requires time and resources. Consider technologies that provide a balance between effectiveness and ease of mastery. Here are some suitable for online shopping:

HTML/CSS/JavaScript: Essential for front-end development.

Python/Django or Flask: A robust and secure choice for backend development.

React or Vue.js: JavaScript libraries for building dynamic user interfaces.

Node.js: Enables JavaScript on the server-side for full-stack development.

SQL/NoSQL Databases: Depending on your data requirements.

12.7. Making the Decision:

Selecting the best technology stack for your online shopping platform depends on the specific requirements, team expertise, and project goals. Each technology has its strengths and weaknesses.

12.8 Testing Types for Online Shopping Management:

Functional Testing:

Unit Testing: Test individual components.

Integration Testing: Check interaction between integrated components.

System Testing: Assess the entire system's functionality.

Acceptance Testing: Confirm the system meets user acceptance criteria.

Non-functional Testing:

Performance Testing: Evaluate responsiveness, scalability, and speed.
Security Testing: Identify vulnerabilities and ensure data integrity.
Usability Testing: Evaluate user interface and overall user experience.
Reliability Testing: Check consistent performance under various conditions.

12.9. Testing Techniques:

Static Testing:

Code Review: Manual examination of source code.
Static Analysis: Automated analysis of code without execution.
Dynamic Testing:

Dynamic Analysis: Analyzing software during execution.
Dynamic Testing Tools: Tools that assess software during runtime.

12.10. Testing Tactics:

Regression Testing: Ensure new changes don't negatively impact existing functionality.
Exploratory Testing: Simultaneously design and execute test cases based on real-time analysis.

12.11. Proposed Testing Process:

Test Planning:

Objective:
Define overall testing goals and scope.
Identify resources and testing tools.
Establish a timeline for testing activities.

Activities:

Develop a Test Plan document outlining the testing strategy.
Identify testing environments and configurations.
Define entry and exit criteria for each testing phase.
Allocate responsibilities to team members.

12.12 Measurement in Software Testing:

Test Coverage:

Objective:

Measure the extent to which the online shopping platform has been tested.

Metrics:

Statement Coverage: Percentage of code statements executed by tests.

Branch Coverage: Percentage of decision points exercised by tests.

Path Coverage: Percentage of code paths tested.

12.13 Collecting Test Data:

Understand Requirements:

Objective:

Gain a clear understanding of the testing requirements.

Identify data dependencies and constraints specified in the requirements.

12.14 Benefits of Measurement:

Assessment of Progress:

Objective:

Evaluate the progress of the online shopping platform.

Benefits:

Enables tracking of project milestones and completion of tasks.

Provides a quantitative measure of progress against timelines.

12.15 Quality Assurance:

Objective:

Assess and ensure the quality of the online shopping platform.

Benefits:

Identifies defects and issues early in the development lifecycle.

Supports proactive measures to improve software quality.

13.Language choosing:

HTML,CSS,and Javascript files make up everything you see on a website. As a web developer, these are the most basic tools I used . They are the languages that I used for build your websites.

13.1.HTML

The basic programming language for web creation is HTML (HyperText Markup Language). It contains the essential elements of a website, such as words, titles, and paragraphs.



13.2.CSS

CSS (Cascading Style Sheets) is a style sheet that describes how HTML components appear on a page. CSS is used to manage your website's appearance, style, and formatting, including RGB values, border colors, background pictures, and more.



```
body {  
  font: x-small;  
  background: #  
  color: black;  
  margin: 0;  
  padding: 0;
```

13.3. JavaScript

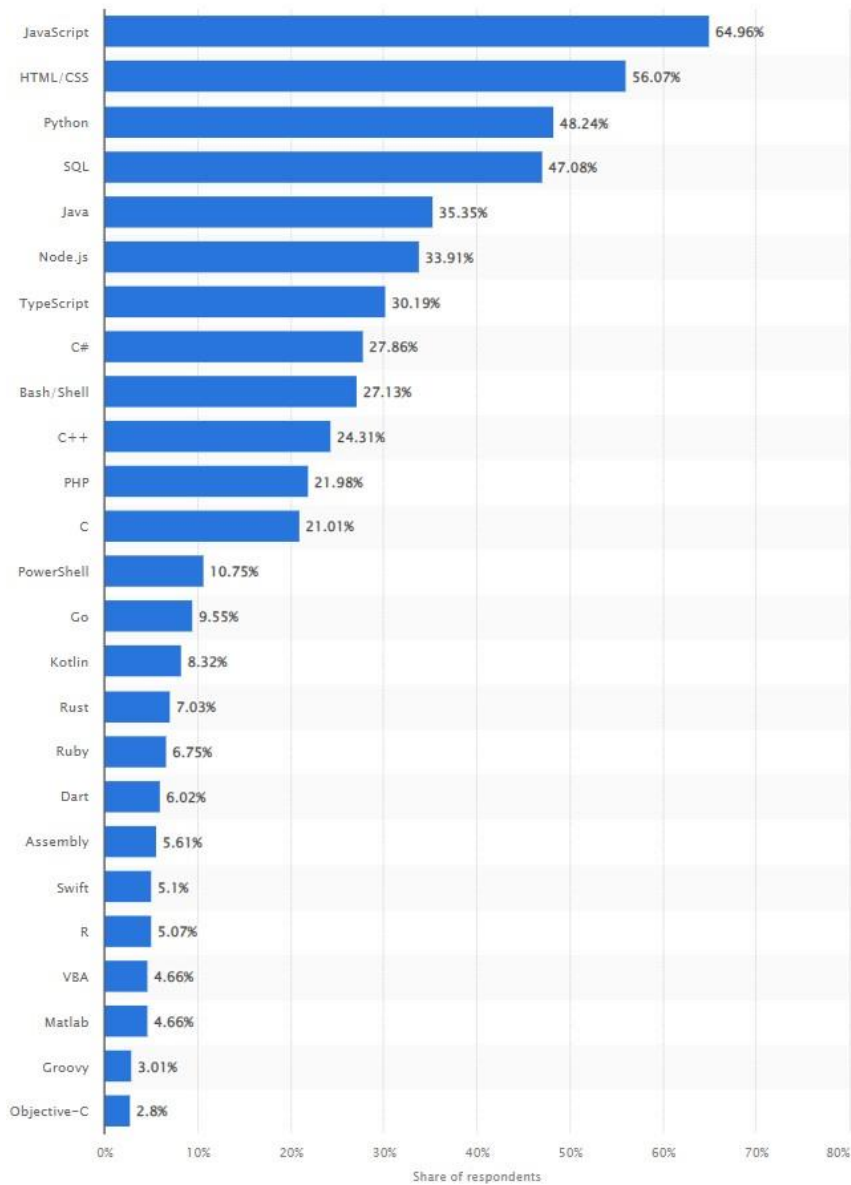
JavaScript is a scripting language that allows you to control the behavior of your website. If you're serious about web programming, you should begin by learning the fundamentals of JavaScript. It is one of the most widely used programming languages in the world, with a low entry barrier and instant results based on your code's success.



14.The Technology Stack

So now we know the fundamentals of HTML, CSS, and JavaScript. Following that, we'll look at the frameworks and libraries that make up the technological stack. A tech stack is a collection of software, apps, programming languages, and tools that work together to create a website. Different tech stacks will be used by companies and web developers depending on their specific goals and needs.

14.1. The most popular programming languages for software development



15. Technical Process and Methodology:



16. Reporting Table:

16.1. Performance Requirements

There is no performance requirement in this system because the server request and response are dependent on the end user internet connection.

16.2. Safety Requirements

The database may get crushed at any certain time due to virus or operating system failure. There for it is required to take the database backup so that the database is not lost. A proper UPS/ Inverter facility should be there in case of power supply failure.

16.3. Security Requirements

- ❖ The system will use a secure database.
- ❖ Normal users can just read information, but they cannot edit or modify

anything except their personal and some other information.

- ❖ The system will have several types of users and every user has access constraints.

16.4. Software Quality Attributes

- ❖ There may be multiple admins creating the project, all of them will have the right to create changes to the system. But the members or other users cannot make changes.
- ❖ The project should be open source.
- ❖ The quality of the database is maintained in such a way that it can be very user-friendly to all the users of the database.
- ❖ The user be able to easily download and install the system.

17. Deployment

17.1. Development Overview

The deployment of the AI Financial Advisor involves the systematic transition of the developed software from the testing and development environment to a production environment. This section outlines the deployment process, including the steps, resources, and considerations for a successful release.

17.2. Deployment Steps

- ❖ I Final Testing: Conduct a final round of testing to ensure that all features meet the specified requirements and that any identified issues have been addressed.
- ❖ I Data Migration: If applicable, migrate necessary data from the testing environment to the production environment to ensure continuity of user data.
- ❖ I Infrastructure Setup: Prepare the production infrastructure, including servers, databases, and any required third-party services. Ensure that the infrastructure meets the specified performance and security standards.
- ❖ I Security Review: Perform a comprehensive security review to identify and address any potential vulnerabilities. Ensure that encryption and access control measures are in place to protect user data.

- ❖ I Deployment of Components: Deploy the AI Financial Advisor software components, including the frontend, backend, and any supporting services, to the production environment.
- ❖ I Configuration Management: Update configuration settings to reflect the production environment. Configure settings such as database connections, API keys, and environment-specific variables.
- ❖ I Monitoring Setup: Implement monitoring tools to track system performance, identify potential issues, and ensure that the system is operating within defined parameters.
- ❖ I User Communication: Notify users of the upcoming deployment and any potential downtime. Provide clear instructions on any actions they may need to take, such as updating their applications.
- ❖ I Rollout Strategy: Depending on the scale and complexity of the deployment, consider a phased rollout to manage potential risks. Monitor the system closely during and after deployment.

17.3. Rollback Plan

In the event of unexpected issues during deployment, a rollback plan will be initiated. The rollback plan includes:

- ❖ Immediate Investigation: Identify the cause of the issue and assess its impact on users and the system.
- ❖ Isolation of New Features: If possible, isolate the newly deployed features or components causing the issue.
- ❖ Reversion to Previous Version: Roll back the deployment to the previous stable version.
- ❖ Communication: Notify users of the rollback, provide an explanation for the decision, and assure them that the team is actively addressing the issue.

17.4. Post-Deployment Activities

I Monitoring and Optimization: Continuously monitor the production environment for performance, security, and user engagement. Optimize configurations based on real-world usage.

I User Support: Be prepared to address user inquiries, issues, and feedback promptly. Provide support through designated channels such as chat, email, or a support ticket system.

I Documentation Update: Update relevant documentation, including user guides and support materials, to reflect any changes or enhancements introduced during deployment.

17.5. Deployment Resources

- ❖ I Deployment Team: The team responsible for overseeing and executing the deployment

process, including developers, system administrators, and DevOps engineers.

- ❖ **I User Support Team:** A dedicated team to handle user inquiries and provide support during and after deployment.
- ❖ **Communication Channels:** Establish clear communication channels for notifying users about the deployment, potential downtime, and any actions they need to take.

17.6.Considerations

- ❖ **Backup and Recovery:** Ensure that robust backup and recovery mechanisms are in place to safeguard against data loss during deployment.
- ❖ **Scalability:** Confirm that the production environment is scalable to handle increased user loads and data volumes.
- ❖ **Regulatory Compliance:** Validate that the deployed system complies with relevant financial regulations and data protection standards.
- ❖ **User Training:** If new features or changes impact user interactions, provide training materials or sessions to familiarize users with the updates.

17.7.Deployment Sign-Off

The deployment process will be considered complete when the following criteria are met:

- ❖ Successful deployment of all components to the production environment.
- ❖ Verification of data migration and integrity.
- ❖ Monitoring tools indicate normal system behavior.
- ❖ User support channels are operational and responsive.

Contact Us

You can get in touch with us in any of the below ways:

By Phone:

+8801518696023

Flex-IT / Office

By Email

contact@rakibit.com

On our website www.rakib-it.com

Address:

Rakib-IT Team,

Khagan

Birulia, Savar,Dhaka1207

We look forward to hearing from you soon!

Agreement Signed By:

.....
Client Signature Order Provider Signature Authority Signature
Md. Rakib Hasan (CEO)
Rakib-IT