IT PROJECT 700

Analysis Phase

Table of content

1) Introduction

The analysis phase of the SDLC (Software Development Life Cycle) is the first phase in the software development process. It involves gathering and understanding the requirements of our Ticketing Web Based Application system to be developed. During the analysis phase, the following activities are Will be performed:

a) Requirements Gathering: The web-based Application is expected to provide clients with quotations and answer frequently asked question about the business
b) Requirement Analysis: According to our analysis, the requirement analysis of the web-based application shows that there will be no conflict or ambiguities.
c) Feasibility study: A feasibility study was conducted to assess the technical, economic, and operational feasibility of the proposed software system which is the Web Based Application. This has helped in determining whether the project should proceed to the next phase or not.
d) System modeling: Various modeling techniques, such as use case diagrams, activity diagrams, and data flow diagrams, were used to represent the system requirements and its behavior. These models help in visualizing and understanding the system.
e) Risk assessment: Potential risks and challenges associated with the software development project are identified and analyzed. This helps in developing strategies to mitigate or manage these risks.
f) Cost estimation: The cost of developing the software system is estimated based on the requirements and the complexity of the project. This helps in budgeting and resource allocation.
g) Stakeholder communication: Throughout the analysis phase, effective communication with the stakeholders is crucial. Regular meetings and discussions are held to ensure that the requirements are well-understood and any changes or clarifications are addressed.

The analysis phase sets the foundation for the subsequent phases of the SDLC, such as design, development, testing, and deployment. It is important to thoroughly analyze and document the requirements during this phase to ensure a successful software development project.

2) Information Gathering methodology(Observation, participatory, Interviews)

Observation

The information-gathering technique that is most effective in combining information from a variety of perspectives, in analysis phase, and in resolving discrepancies is Observation.

Observation: involves carefully watching and monitoring people, processes, and events in order to gather information and insights. This technique allows researchers to collect data on human behavior, interactions, and interactions with the environment without relying on self-reported information or interviews. By observing people in their natural environment, researchers can gain a deeper understanding of their behaviors, motivations, and decision-making processes. This information can then be used to build consensus and resolve discrepancies among different perspectives. For example, if there is a disagreement between two groups of stakeholders about a particular issue, observation can be used to gather data on how each group interacts with the issue and how they make decisions related to it. This information can then be used to build consensus and resolve discrepancies by providing a neutral and objective view of the situation. Observation is a powerful information-gathering technique that can be used to build consensus and resolve discrepancies by providing a comprehensive and unbiased view of the data.

Participatory analysis

Participatory analysis in the analysis phase refers to involving stakeholders or end-users in the process of analyzing data or information. It is a collaborative approach that aims to gather insights, perspectives, and knowledge from those who are directly affected by or have expertise in the subject matter.

Criteria for participatory analysis in the analysis phase may include:

1. Inclusion of stakeholders: The analysis process should involve a diverse group of stakeholders, including end-users, community members, experts, and relevant organizations.

2. Active participation: Stakeholders should actively contribute their knowledge, experiences, and perspectives during the analysis phase. This can be done through workshops, focus groups, interviews, surveys, or other participatory methods.

3. Shared decision-making: The analysis process should aim to reach consensus or shared understanding among stakeholders. Decision-making should be inclusive and consider the perspectives and needs of all participants.

4. Transparency and accountability: The analysis process should be transparent, with clear communication of methods, findings, and limitations. Stakeholders should have access to information and be able to provide feedback or challenge the analysis if needed.

5. Empowerment and capacity building: Participatory analysis should aim to empower stakeholders by building their capacity to understand and engage in the analysis process. This can include providing training, resources, or support to enhance their knowledge and skills.

An interview in the analysis phase refers to a process of gathering information and conducting discussions with stakeholders, subject matter experts, and users to understand their requirements, expectations, and concerns related to a project or system. It helps in identifying the needs and goals of the project and provides insights for further analysis and decision-making.

Interview

Attributes of an interview in the analysis phase:

1. Purpose: The main objective of the interview is to gather information and insights from stakeholders.

2. Participants: The interview involves the interviewer (typically a business analyst or project manager) and the interviewee (stakeholders, subject matter experts, users).

3. Questions: The interviewer prepares a set of questions to guide the interview and elicit relevant information.

The Questions

- What type of business?
- Who is the Project Manager?
- How will the system generate Quotations?
- How will the system solve real world Problems?
- How will the system be created?
- How much budget is required for this project?

4. Structure: The interview follows a structured or semi-structured format, with predefined questions and room for open-ended discussions.

5. Documentation: The interviewer takes notes or records the interview to capture important details and requirements.

6. Analysis: The information gathered from the interview is analyzed to identify patterns, common themes, and requirements.

7. Follow-up: Depending on the complexity of the project, follow-up interviews or discussions may be conducted to clarify or gather additional information.

It is important to note that interviews in the analysis phase may vary in their approach and techniques based on the specific project, stakeholders, and requirements. The attributes listed above serve as a general rubric for making classifications and understanding the concept of an interview in the analysis phase.

3) Analysis of existing system

The purpose of studying the existing system during the analysis phase is to understand how it currently operates, identify its strengths and weaknesses, and determine what changes are necessary to meet the project's objectives. There are several reasons why studying the existing system is important:

1. Identify current business processes: By examining the existing system, analysts have identified the current business processes and workflows that are in place. Our baseline for understanding how the organization operates and how information flows through the system
2. Determine system requirements: Analyzing the existing system helped to identify the key requirements for the new system. This includes identifying the functionality that the new system must provide, as well as any constraints or limitations that must be taken into account.

3. Identify areas for improvement: By examining the existing system, analysts can identify areas where the system can be improved. This includes identifying inefficiencies, redundancies, and areas where manual processes can be automated such as Tax Invoice, Quotations and Travel logs.

4. Establish a baseline for measuring success: By studying the existing system, analysts has establish a baseline for measuring the success of the new system. This helps to ensure that the new system is meeting the organization's needs and that it is delivering the expected benefits.

4) Data Analysis (Data Integrity & Constraints)

Data Analysis

Data Analysis in the analysis phase refers to the process of examining and interpreting data to uncover patterns, trends, and insights. It involves applying various techniques and methods to transform raw data into meaningful information that can be used for decision-making.

Criteria for making classifications in data analysis include:

1. Data Quality: Assessing the accuracy, completeness, and reliability of the data.

2. Data Cleaning: Removing any inconsistencies, errors, or outliers in the data.

3. Data Transformation: Converting data into a suitable format for analysis, such as aggregating or summarizing.

4. Data Exploration: Exploring the data through visualization, descriptive statistics, and exploratory data analysis techniques.

5. Data Modeling: Applying statistical or machine learning models to analyze relationships and make predictions.

6. Data Interpretation: Drawing conclusions and making inferences based on the analysis results.

Each of these criteria plays a crucial role in the data analysis process, ensuring that the analysis is accurate, reliable, and actionable.

Data Intergrity

Data integrity in the analysis phase refers to the accuracy, consistency, and reliability of the data used for analysis. It ensures that the data is complete, valid, and free from errors or inconsistencies.

To maintain data integrity in the analysis phase, the following criteria or attributes should be considered:

1. Accuracy: The data should be correct and precise, without any errors or inaccuracies. It should reflect the actual values or information it represents.

2. Completeness: The data should be comprehensive and include all the necessary information required for analysis. It should not have any missing values or gaps.

3. Consistency: The data should be consistent across different sources, systems, or time periods. It should not have conflicting or contradictory information.

4. Validity: The data should be valid and relevant to the analysis being performed. It should meet the defined criteria or rules for inclusion in the analysis.

5. Reliability: The data should be reliable and trustworthy. It should be obtained from reliable sources and be free from bias or manipulation.

By ensuring these attributes, data integrity can be maintained in the analysis phase, leading to accurate and reliable insights and conclusions.

Data Constraints

Data constraints in the analysis phase refer to the limitations or restrictions on the data that is being analyzed. These constraints help ensure that the data is accurate, reliable, and suitable for analysis. Here are some common data constraints in the analysis phase:

1. Completeness: The data should be complete, meaning that it includes all the necessary information required for analysis. Missing or incomplete data can lead to inaccurate results.

2. Consistency: The data should be consistent, meaning that it is free from contradictions or discrepancies. Inconsistent data can lead to unreliable analysis.

3. Accuracy: The data should be accurate, meaning that it is free from errors or mistakes. Inaccurate data can lead to incorrect analysis and conclusions.

4. Validity: The data should be valid, meaning that it is relevant and applicable to the analysis being conducted. Invalid data can lead to irrelevant or misleading results.

5. Timeliness: The data should be timely, meaning that it is up-to-date and reflects the current state of the subject being analyzed. Outdated or stale data may not accurately represent the current situation.

6. Integrity: The data should have integrity, meaning that it is secure and protected from unauthorized access, modification, or deletion. Data integrity ensures the reliability and trustworthiness of the analysis.

These data constraints serve as criteria or rubrics for making classifications and decisions during the analysis phase. By adhering to these constraints, analysts can ensure that the data used for analysis is of high quality and can be relied upon for making informed decisions.

5) Weakness of the Current System

The current system is a manual system. Therefore it has the following constraints

- Cost are too high (Cost to travel to sites, costs to log in quotations and Tax Invoice Manually)
- Time, Time is also another constraints. You can only attend one customer at the certain time.

- Storage, You need a proper storage for all the quotations, reports, Tax Invoices and Important company documents.
- You need a clerk, for documents to be properly filed you will need a clerk on a daily basis.
- You need accessories such as printers and filling books. These helps with properly putting the documents together.

6) Analysis of the Proposed System (Functional Requirements)

List of Functional Requirements: -

a) Performance, Just as the automotive industry has key metrics to describe a car's performance, such as horsepower and torque, web application metrics work in a similar way – providing a measurable and comparable way to express how well the site is performing compared to others.

These are the following key Factors we have considered in perfomance

✓ Page Load Time
  This key web performance indicator measures the time it takes for a web page to fully load and be ready for interaction. This metric is crucial as it directly impacts user experience and engagement.

✓ Largest Contentful Paint (LCP)
  It measures the time it takes for the largest visible element on a web page to become fully rendered, providing insight into when the main content is ready for user interaction, with faster LCP times indicating a more responsive user experience.

✓ First Input Delay (FID)
  FID focuses on the responsiveness of a web page to user interactions, measuring the time between a user's action (like clicking a button) and the browser's ability to respond. A low FID score indicates a more interactive and user-friendly web application.

✓ Cumulative Layout Shift (CLS)
  CLS quantifies the visual stability of a web page by measuring how much elements shift or move unexpectedly during the page load. A low CLS score indicates a more visually consistent experience, minimizing user frustration caused by sudden layout changes.

✓ Time To First Byte (TTFB)
  TTFB is one of the main indicators that indicate a website's performance and measures the time it takes for the browser to receive

the first byte of data from the server. A lower TTFB indicates faster server response times and can contribute to overall faster page load times.

- ✓ Network Round Trip Time (RTT)
  It is the time it takes for a request to travel from the client to the server and back. RTT affects the overall latency and responsiveness of the web application.

b) System security and privacy, Web application security (also known as Web AppSec) is the idea of building websites to function as expected, even when they are under attack. The concept involves a collection of security controls engineered into a Web application to protect its assets from potentially malicious agents. Web applications, like all software, inevitably contain defects. Some of these defects constitute actual vulnerabilities that can be exploited, introducing risks to organizations. Web application security defends against such defects.

The Three Security system in place

- ✓ Static Application Security Test (SAST). This application security approach offers automated and manual testing techniques. It is best for identifying bugs without the need to execute applications in a production environment. It also enables developers to scan source code and systematically find and eliminate software security vulnerabilities.
- ✓ Penetration Test. This manual application security test is best for critical applications, especially those undergoing major changes. The assessment involves business logic and adversary-based testing to discover advanced attack scenarios.

- ✓ Runtime Application Self Protection (RASP). This evolving application security approach encompasses a number of technological techniques to instrument an application so that attacks can be monitored as they execute and, ideally, blocked in real time.

c) Error handling and recovery, Error handling and graceful error recovery play a crucial role in web application development. We have will follow these practices to handle errors on our web applications:

- ✓ Implement proper exception handling

Exception handling allows you to catch and handle any errors or exceptions that occur during the execution of your code. Try-catch blocks to catch specific exceptions and handle them accordingly. This prevents the application from crashing and provides a controlled way to deal with errors.

✓ Provide informative error messages

When an error occurs, it’s essential to provide users with clear and informative error messages. Avoid generic error messages like ‘An error occurred,’ as they do not provide any useful information. It will include specific details about the error and suggest possible solutions. This helps users understand what went wrong and how to fix it.

✓ Log errors for debugging

Logging errors is crucial for debugging and troubleshooting. Implement a logging mechanism that records errors along with relevant information like timestamps and stack traces. This allows us to analyze and debug the issues, even if they cannot be reproduced in real-time. Logging also helps in identifying recurring errors or patterns that need to be addressed.

✓ Design for graceful error recovery

Planned for graceful error recovery by designing the application to handle errors gracefully. This means providing fallback options or alternative solutions whenever possible. For example, if a database query fails, it should display cached data instead of throwing an error. By providing default values or alternative workflows to keep the user engaged even when errors occur. This can significantly improve user experience and prevent frustration.

d) Data entry and management

One of the most significant advancements in remote entry is the process of entering data on a form accessed on the Web. This method has become a popular way to collect data because access to the internet has expanded dramatically, allowing data to be entered directly into a central database. It also provides less dependency on specific types of equipment for entering data. Web-based methods allow for instant editing checks as responses are entered, and, if desired, allows for many of the traditional techniques for inputting responses such as textboxes, dropdowns, checkboxes or other styles that are available through web programming without additional software installed on the client other than a web browser.

We will use Hyper Text Markup Language (HTML) to develop web forms that will be displayed through client browsers and allowed data to be entered and submitted to a server. The JavaScript programming language is used to allow data to be validated before it was submitted. Enhancements to HTML through Dynamic HTML (DHTML) and Cascading Style Sheets (CSS) gave us the ability to better control the appearance of the web forms and graphical images. Server side languages (such as Java, or Php) will be used to produce Web applications on the server that execute at the time of connection to the web site. These languages are used to build server applications that more extensively examine the data for errors when it is submitted by the remote computer from the customer

7) Non-Functional Requirements

List of some non-functional requirements: -

a) Scalability requirements

scalability is an application's ability to handle more customers or users than at the beginning of its work. Scalability is one of the most crucial factors when it comes to web application development because that's exactly what provides our users with a good user experience when they double or even triple. Scalability is responsible for handling the traffic, responding accurately, and reacting to the growing number of requests. Creating an incredible application, the last thing we want from it is to flop and lose our users on the way. Unfortunately, it might happen when an application's user base grows too fast. When web application's architecture wasn't designed for seamlessly handling explosive growth, its hardware infrastructure might collapse.

Three principle that our applications will consider to make sure that it is scalability

➢ Efficiency

Failure will be the only option if the system you're working on cannot cope with all the loads and perform functions and tasks in every condition.

➢ Fewer dependencies

When making some changes, they aren't supposed to affect the code crucially. Also, it is essential to expand the software's functionality, using assets you have already created to improve code reusability.

➢ Testability

Testing is supposed to be easy, so we can find bugs and fix them effortlessly.

b) Cost and resource requirements.

The application will need the following:

- Free visual studio code application
- Free XAMP application
- Afrihost Web hosting portal (Fees Vary)
- Knowledge of the following languages (HTML, Javascript, PHP, JAVA, CSS, MYSQL)

c) Availability

Software availability refers to the ability of users to access and use software without experiencing frequent disruptions, downtime, or errors. High software availability is crucial for ensuring user satisfaction, business continuity, and reliable service delivery. Here are five key aspects of software availability:

- ✧ High Availability (HA): High availability refers to the ability of a software system to remain operational and accessible even in the face of hardware or software failures. HA is achieved through redundancy and fault-tolerant design, ensuring that if one component fails, another takes over seamlessly. This minimizes downtime and ensures continuous service availability.
- ✧ Fault Tolerance: Fault tolerance is the ability of a software system to continue functioning properly even when one or more components fail. Fault-tolerant systems are designed to detect and recover from failures automatically, without impacting the overall availability of the software. This is achieved through techniques such as redundancy, error detection, and error recovery mechanisms.
- ✧ Disaster Recovery (DR): Disaster recovery refers to the processes and procedures in place to restore software functionality after a major disruption, such as a natural disaster or a cyber-attack. DR plans include backup and restoration strategies, data replication, and failover mechanisms to ensure that the software can be quickly recovered and made available to users.
- ✧ Load Balancing: Load balancing is the distribution of workloads across multiple servers or resources to optimize performance and prevent overloading. By evenly distributing the workload, load balancing ensures that no single server or resource becomes a bottleneck, thereby improving the availability and responsiveness of the software. Load balancing can be achieved through various techniques, such as round-robin, least-connections, or dynamic load balancing algorithms.
- ✧ Monitoring and Alerting: Monitoring and alerting systems continuously monitor the health and performance of the software and its underlying infrastructure. These systems detect anomalies, errors, or performance degradation in real-time and trigger alerts to notify administrators or support teams. By proactively identifying and addressing issues, monitoring and alerting systems help maintain high software availability by minimizing downtime and ensuring prompt resolution of problems.

8) Data Model for the proposed system

A web-based application data model represents the structure and organization of data within a web application. Here's a general outline for our application

1. Identify Entities: Determine the main concepts or objects relevant to your application (Users, Products, Quotations, Tax Invoices).

2. Define Attributes: Identify the properties or characteristics of each entity (User: name, email, password; Product: title, description, price).

3. Establish Relationships: Define how entities interact or relate to each other (e.g., Users make Orders, Orders contain Products).

4. Determine Data Types: Assign data types to each attribute (name: string, price: decimal).

5. Consider Data Constraints: Define rules for data validation and integrity (unique email addresses, required fields).

Data model for an Web based application:

Entities:

- Users
- Products
- Quotation
- Tax Invoice

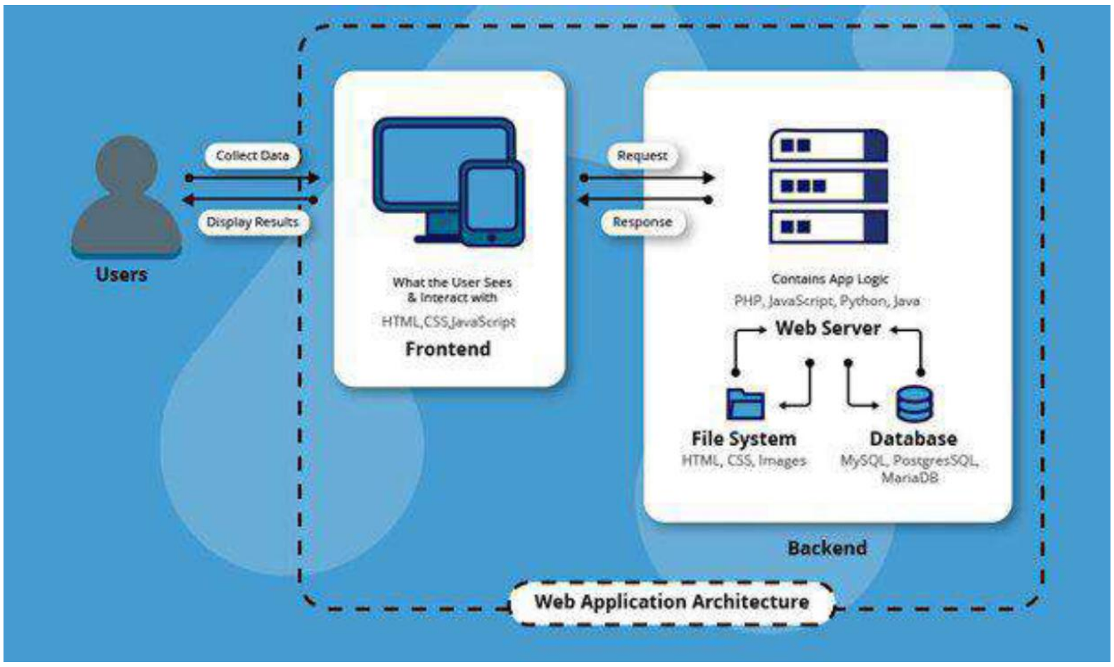
Attributes:

- Users:
    - id (int, primary key)
    - name (string)
    - email (string, unique)
    - password (string)
- Products:
    - id (int, primary key)
    - title (string)
    - description (text)
    - price (decimal)
- Orders:
    - id (int, primary key)
    - user_id (int, foreign key referencing Users)
    - order_date (datetime)
    - total (decimal)
- Payments:

- id (int, primary key)
- order_id (int, foreign key referencing Orders)
- payment_date (datetime)
- amount (decimal)

Relationships:

- A User can make many Orders (one-to-many).
- An Quotation is made by one User (many-to-one).
- An Quotation contains many Products (many-to-many).
- A Tax Invoice is associated with one Quotation (many-to-one).


Users
Collect Data
Display Results
What the User Sees & Interact with
HTML,CSS,JavaScript
Frontend
Request
Response
Contains App Logic
PHP, JavaScript, Python, Java
Web Server
File System
HTML, CSS, Images
Database
MySQL, PostgresSQL, MariaDB
Backend
Web Application Architecture