**SOFTWARE DEVELOPMENT LIFECYCLE (SDLC)**

The SDLC is a well-defined process of constructing software solutions that will enable the realization of a structured procedure from the initiation phase up to the deployment phase. An Explanation of how a Requisition Management System can be developed using this SDLC stage is presented below.

The first phase is Requirement Gathering & Analysis in which the stakeholders (HR, finance, and procurement dept.) work together to determine the system requirements. This stage assists in defining major requirements for the system which should include user registration, submission of requisitions and approval processes, and report generation. The documented requirements are further reviewed in terms of their feasibility to establish whether it is possible to build the system within specified financial and temporal resources.

Following is System Design where there is a description of the architecture of the Requisition Management System. This phase involves drawing High-Level Design (HLD), which outlines how the structure of a system and how users are going to interact with the system and Low-Level Design (LLD) which describes how a particular part of the system is going to function within its own module. Another component that the database schema includes is the requisition details, approval status, and user data; prototypes of user interface, either in form or dashboard, are also developed for the purpose of feedback.

Next is the Implementation / Coding phase where the developers put together the system as the design determines. Essential sub-components such as the user module, the requisition form and Approval processes are built in a way that a version control system is used during the developmental process and also to incorporate error handling mechanisms. To keep up with the quality of codes in the system there are periodic code review in order to keep up with the best practices.

When the system has been coded then moves to the Testing phase as another phase. Here, Unit Testing confirms that every function or feature works as required, for instance, the user login feature or submitting a requisition while Integration Testing investigates if the several components integrate well. System testing recreates real life scenarios while User Acceptance Testing (UAT) allows users to ensure that a system will work as per the business needs. Any bug or problem is resolved here with the help of codes and software.

The next step that follows after the testing is the Deployment where it is implemented for organizational usage. This could be done in the form of a pilot test to a small number of people before going for full implementation. Once it has steadied the free version of the system is released to other users with regular tracking for performance and bugs.

Last but not the least is the Maintenance which helps the organization to keep the system running effectively and updated once it has been deployed. This phase also continues with having new versions to provide new features, addressing possible security holes and Operational bugs encountered. Feedback from the user is received in a continuous basis to make improvements to the system.

Here’s how the SDLC applies to the Requisition System Assesment\_PartB: Here’s how the SDLC applies to the Requisition System project:

**1. Planning**: Think of this as laying the foundation. The team figures out what the project is about, what it needs to achieve, and the resources and time required.

**2. Requirement Analysis**

**• Goal:** Know what is required from the system.

Acquiring requirements of the requisition system like: recording information about the staffs, handling of the requisition request, estimations of the costs, approval or rejection of the request and providing the statistics.

For the Requisition System:

Requirements would include a method of entering staff information and items by which costs are related. A system through which all the requisitions in the course of an organization can be captured and retrieved. A method for automatically green flagging or pending flagging requisitions beyond, or below, a particular cost ceiling.

**3. System Design**

• **Goal**: Organise within the design the system architecture and layout, and how the specified requirements will be met.

• Activities: Communally, the structures for the requisitions, approval processes, and database structure if necessary in addition to the flow, the user interface if it is a web or desktop application, and the logic for the requisitions and approval processes. Explain on how it will deal with staff details, how it will take requests, and how approvals will be done.

For the Requisition System:

• Establish a class known as RequisitionSystem, to respond to user information input, performance of calculations, and granting of approvals. Decide whether to use a basic data structure such as a simple list or an array in memory (similar to the Python list) or persist it to the external storage system such as a database or a file. Decide on the format to be used to indicate the status of requisitions, whether it is approved or still pending, and how the information on the staff is to be stored.

**4. Implementation (Coding)**

**• Goal:** Now it is time to write the actual code according to specified design documents.

Some of the acts that should be performed include; translation of the design into code. Ensure that coding standards and best practices are followed as each function is developed and test the function.

**For the Requisition System:**

• Design the class RequisitionSystem that shall have methods that will manage staff info, process requisition, and generate stats.

• Take an evaluation of how the sistem is capable of handling multiple requests from different users or where requisitions should be pending or not approved.

**5. Deployment**

**• Goal:** Put it out into the field and make it usable.

Since the system is to be used, it should be fitted in the environment of use. Provide documents (handbook) and if need be orient the users.

**For the Requisition System:**

• If the system is standalone, build it as a script that users will be able to execute.

• If it is a web or a desktop application, then upload it on a server or desktop.

• If it’s a document it should be made available to the appropriate employees/staff and other interested parties.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**6. Maintenance**

• Goal: Ensure the software is always active and at the same time apply for updates as may be required.

Address any issues that occur after the software has been deployed in an organization. Sometimes it is necessary to add new modifications or improvements to the software (for example creating a connection with a database or including a login).

**For the Requisition System:**

• They may need more options such as export requisitions or work with approval processes or have the possibility to save data permanently. Add-ons may involve changes in the approval procedure, or the criteria used to set auto-approval for an order.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Requisition System Problem Solution**

In this requisition system, SDLC assists by providing a framework or procedures to follow hence coming up with well developed software solution that is competent. It is a step-by-step approach to planning and executing the work and makes sure none of the vital activities like, for example, requirement gathering, system designing or testing, can be omitted.

To further improve the current solution: To further improve the current solution:

1. Database Integration: Storing requisitions in the memory did not happen which would have been beneficial but using a database rather than memory helps in persisting the data sessions.

2. User Interface: There is always a way to at least make a web or desktop GUI as simple as ‘enter information of requisition to order from staff’, which should be helpful in increasing user experience.

3. Role-Based Access Control: Since different roles (for instance admin role, an approver role, or requester role) would be set, only the authorized users can approve or

Key SDLC Phases for Requisition Management System: Key SDLC Phases for Requisition Management System:

1. Requirement Gathering & Analysis: Gather stakeholder demands & Conduct the study and check feasibility.

2. System Design: Design and scope the system, proposing a detailed plan at a more abstract level, as well as at a more detailed level, with the help of sketches, accurately delineating how data is processed in the system and how the system’s interface looks.

3. Implementation / Coding: Further the system’s modules based on coding standards as well as employing version control.

4. Testing: In this stage, unit, integration, system, and user acceptance testing will be done to check on system quality.

5. Deployment: Stage-by-stage implementation of the system in the production environment is possible in this approach with ideal performance.

6. Maintenance: Help users continuously, correct mistakes, and replenish and improve the application by feedback of everyday users.

This SDLC approach assures the company has the sound and scalable solution to requisitioning from its submission, approval to tracking.