Peter James Mavisi-SCT221-C004-0518/2022

Brian Kibet ng’eno-sct221-c004-0603/2022

Shadrack dut-sct221-c004-0406/2022

**Assignment 1;**

1. Prepare system requirement specification

**Introduction**

-The railway reservation system is an online application for assisting user or railway manager in managing reservation in a railway department, the system will provide basic set of features to conduct reservation or cancellation ,train scheduling timing, search for available reservation and

-Manage checking/check out processes.

**Purpose**

-The railways reservation system would have the following key goals:-

* Provide a graphical user interface for user to interface with the backend database
* Screens to to search for a train /seats based on user driven parameters
* Minimize overall time

**Scope of project**

-Main scope and deliverables of the project would be fast, easy and efficient working.

-Easy way for cancellation and reservation of the tickets.

-Get exact information about trains, available seats, charges hence faster communication.

**PRODUCT FUNCTIONS:**

-It will have two users interface layers, first one is user interface and the second one is Train Management Interface.

GUEST INTERFACE:

* View\_Train\_Info
* Search Train
* Create Account
* Customer Login
* Purchase Ticket
* Purchase History

TRAIN MANAGEMENT INTERFACE:

* Login Function
* Staff Module
* Reservation Module
* Ticket Module
* Train Module
* Cost Module
* Availability Module
* Branch Module

2.5 OPERATING ENVIRONMENT:

* Linux-8
* Windows-7
* Windows-8
* Windows-10
* Mac Os

2.6 OPERATIONS: ฀

* Book at ticket counter from 6 am to 11 pm anytime. ฀
* Book well before 50 days of travel. ฀
* One-form for 4 people only. ฀
* To spare time and long lines of people at the booking counter, online reservation is a great tool. 2.7 User Interfaces:
* The system will give an assistance (clarification) link from each showed HTML page to disclose how to utilize that page. The Web page of RRS will allow total route, train choice, ticket reservation and ticket/train data queries using input devices be it keyboard, mouse or touch screen in Smart-phones.

Hardware Interfaces:

* Nothing special Interface is required.

Different Functionalities in the system

• View Train Details:

The system will help users or guests to see timings, PNR number, Train Name, Journey time etc. of trains.

• Find Trains:

The system will have a search function. User can request train based on starting and ending station from the database and the needful would be provided accordingly

• Sign In: When the user has already registered him/her just needs to enter the username and password which he/she has created during registration process not exist.

• Book Ticket:

- The system will allow client to buy the train ticket from user interface. The system will show all the train info. The framework will show the date time and cost of the ticket.

-the framework will have confirm booking button. At the instant when confirm booking button is pressed/clicked, the framework will provoke client to login or enlist account.

-If customer haven’t login prompt customer to login and create account. The system will also ask the details such as Train no, Starting destination, Arrival Destination, Booking Quota (General OR Total), Booking date etc.

• Transaction History:

- The customer can view all the transaction history from this function. All the payment details like any transaction failed, information about the money etc. would be recorded here and whenever the user requests for it would be sent to the user.

• Staff Login:

- The system will allow only staff username and password to access the system. Access to this module is decided by account-type and branch/department address.

- When the staff has been logged in the system admin will have given the permission to the staff to perform CRUD operations. CRUD means Create, Read, Update and Delete.

Railway administration will have to login first to get to this module. The system admin has the permission to add staff, update staff and erase staff. The system will request the following details to be entered for the record:

a. Staff Username

B. password

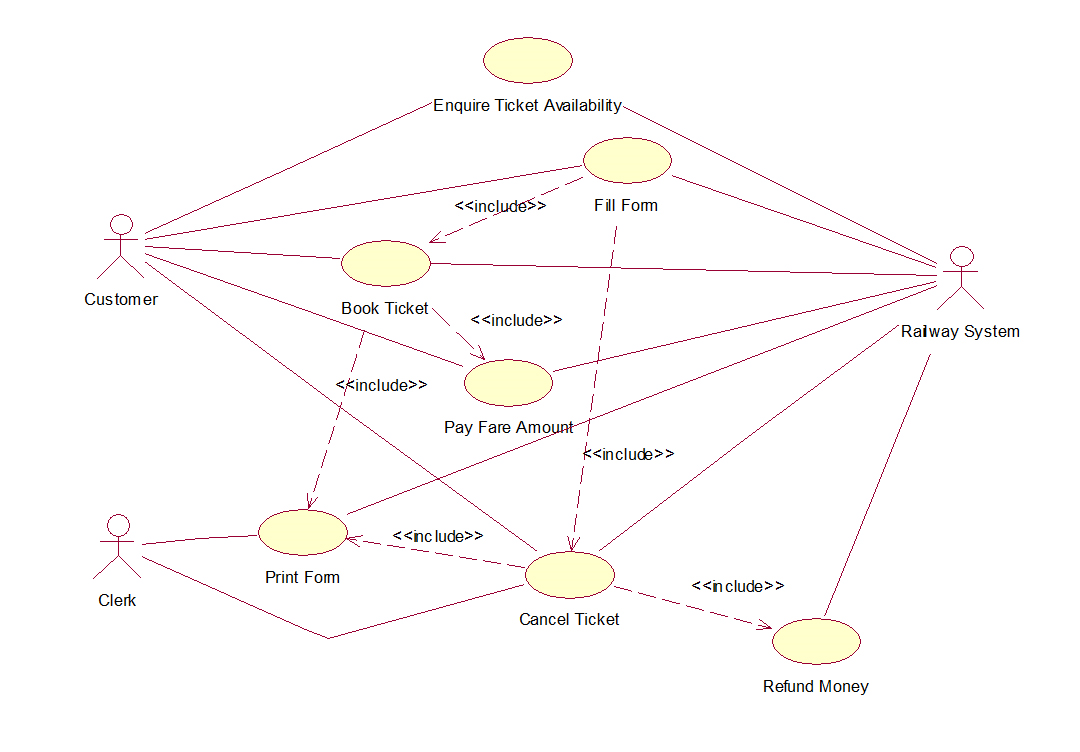
c. Account

d. First Name

• Reservation Module:

- The system will permit counter representative, station director and station manager of the specific station to get to this module. They ought to login first. The framework will restrict the admittance to branch level in sequential manner

2. Design a prototype for the above case study.



**Assignment 2.**

**Qns.1**

1. Walkthroughs: A walkthrough is a type of informal review where the author of a software artifact (such as a requirement specification, design document, or code) presents it to a group of peers or stakeholders for feedback and comments. The purpose of a walkthrough is to improve the quality of the artifact, identify any errors or inconsistencies, and ensure that it meets the objectives and standards. A walkthrough is usually conducted in an informal and collaborative manner, without following a strict agenda or checklist. The participants can ask questions, make suggestions, and point out any issues during the presentation. The author can then incorporate the feedback and revise the artifact accordingly.

- Reviews and inspections: A review or inspection is a type of formal review where a software artifact is examined by a group of qualified reviewers or inspectors for defects, compliance, and improvement opportunities. The purpose of a review or inspection is to ensure that the artifact meets the quality criteria, conforms to the specifications and standards, and satisfies the requirements and expectations. A review or inspection is usually conducted in a structured and systematic manner, following a predefined process and checklist. The participants have to prepare for the review or inspection by reading the artifact beforehand, identifying any potential defects, and documenting their findings. The review or inspection session involves discussing the findings, resolving any disagreements, and making recommendations for corrective actions. The author has to follow up on the recommendations and verify that the defects have been fixed.

- Dynamic testing: Dynamic testing is a type of software testing where the software is executed with various inputs and outputs to check its behavior and functionality. The purpose of dynamic testing is to verify that the software performs as expected, meets the functional requirements, and handles different scenarios and situations. Dynamic testing can be performed at different levels of software development, such as unit testing, integration testing, system testing, and acceptance testing. Dynamic testing can also be classified into different types based on the test objectives, such as functional testing, non-functional testing, regression testing, performance testing, security testing, usability testing, etc.

- Traceability matrices: A traceability matrix is a document that maps and traces the relationship between different software artifacts, such as requirements, design documents, test cases, code modules, etc. The purpose of a traceability matrix is to ensure that all the artifacts are consistent and aligned with each other, that all the requirements are covered by the design and test cases, that all the test cases are linked to the requirements and code modules, that all the code modules are derived from the design documents and test cases, etc. A traceability matrix helps to track the changes and impact of any modifications in any artifact, to measure the completeness and quality of the software development process, and to facilitate communication and collaboration among different stakeholders.

- Debugging environments: A debugging environment is a set of tools and techniques that help software developers to find and fix errors or bugs in their code. A debugging environment typically consists of a debugger

**Qns 2.**

Software quality evaluation is a critical process in software development that aims to assess the overall quality of a software product. It involves identifying and measuring various attributes of the software, such as functionality, reliability, efficiency, maintainability, and usability. However, there are several problems associated with software quality evaluation that need to be addressed to ensure effective and reliable results.

1. Subjectivity: Evaluating software quality is often subjective and depends on individual perspectives and biases. Different stakeholders may have different criteria for judging quality, leading to varying evaluations. This subjectivity makes it challenging to establish consistent and objective measures of software quality.

2. Complexity: Software systems are becoming increasingly complex, making it difficult to assess their quality comprehensively. With the rise of distributed systems, cloud computing, and machine learning, evaluating quality attributes like scalability, security, and performance becomes more complex and requires sophisticated evaluation techniques.

3. Lack of standards: Although various software quality standards exist, such as ISO/IEC 25010 (formerly known as ISO/IEC 9126) and the Capability Maturity Model Integration (CMMI), there is no universally accepted standard for software quality evaluation. This lack of standardization makes it challenging to compare and benchmark different software products effectively.

To address these problems and promote consistent software quality evaluation, several software standards have been developed. These standards define guidelines, processes, and metrics to assess software quality. Some prominent standards include:

1. ISO/IEC 25010: This standard provides a framework for evaluating the quality characteristics of software. It defines a set of quality attributes such as functionality, reliability, usability, efficiency, maintainability, and portability.

2. Capability Maturity Model Integration (CMMI): CMMI is a process improvement framework that helps organizations improve their software development and management processes. It provides a set of best practices for achieving higher levels of maturity in software development, which indirectly impacts software quality.

Certification plays an essential role in software quality evaluation. It provides a formal recognition that a software product or process meets specific quality standards. Certification programs, such as the International Software Testing Qualifications Board (ISTQB) certification, validate the competence of software testers, ensuring that they possess the necessary skills and knowledge to evaluate software quality effectively.

Software tools also play a significant role in supporting software quality evaluation, particularly in the field of systems engineering. These tools aid in automating quality evaluation processes, managing requirements, performing system modeling and simulation, conducting code analysis, and executing various testing activities. Examples of such tools include IBM Rational DOORS, Spark Systems Enterprise Architect, and HP ALM (Application Lifecycle Management).

**Qns3.**

CASE (Computer-Aided Software Engineering) tools play a crucial role in web engineering techniques and processes by providing software developers and engineers with automated support for various activities throughout the web development lifecycle. These tools assist in tasks such as requirements gathering, analysis and design, code generation, testing, and maintenance. They help streamline the web development process, improve productivity, and ensure adherence to standards and guidelines. Let's discuss the role of CASE tools in web engineering techniques and process, as well as their impact on standards and guidelines:

1. Requirements Gathering and Analysis:

- CASE tools offer features for capturing, documenting, and analyzing requirements in a structured manner.

- They facilitate the creation of use case diagrams, entity-relationship diagrams, and other visual models to represent system requirements.

- These tools support collaboration among stakeholders, enable requirement traceability, and help ensure that all requirements are properly addressed.

2. Design and Modeling:

- CASE tools provide graphical modeling capabilities to create visual representations of the web application's architecture, components, and interactions.

- They support the creation of various design diagrams such as class diagrams, sequence diagrams, and activity diagrams.

- These tools enable designers to visualize the system's structure and behavior, helping to identify potential design flaws or inconsistencies.

3. Code Generation:

- CASE tools assist in automating the process of generating code from design models, reducing manual effort and minimizing errors.

- They can generate skeleton code, templates, or even complete code based on predefined design patterns and coding standards.

- These tools ensure consistency in coding practices and promote adherence to coding guidelines and best practices.

4. Testing and Debugging:

- CASE tools provide functionality for test case generation, test execution, and result analysis.

- They assist in automating the testing process, reducing manual effort and improving test coverage.

- These tools enable developers to identify and debug issues by providing features like code tracing, breakpoints, and error logging.

5. Maintenance and Evolution:

- CASE tools offer features for code refactoring, which helps improve the structure and maintainability of the codebase.

- They support version control, allowing developers to track changes, collaborate, and manage code repositories.

- These tools assist in impact analysis, helping developers understand the consequences of making changes to the system and ensuring that modifications do not introduce new issues.

6. Standards and Guidelines:

- CASE tools often incorporate industry standards and guidelines into their functionalities.

- They provide templates and wizards that enforce compliance with coding standards, design patterns, and architectural principles.

- These tools help ensure that web applications are developed following best practices, resulting in maintainable, scalable, and secure systems.

7. Collaboration and Documentation:

- CASE tools support collaboration among team members by providing shared repositories, version control systems, and communication channels.

- They facilitate documentation generation, allowing developers to automatically generate documentation from design models, code comments, and other artifacts.

- These tools promote effective communication, knowledge sharing.