UNIT: SOFTWARE ENGINEERING (ICS 2302)

TITLE: GROUP ASSIGNMENT

|  |  |
| --- | --- |
| NAME | REG NO |
| MOSES TIACH MAYANG | SCT221-C004-0519/2022 |
| ABDULLAHI HASSAN MURSAL | SCT221-C004-0515/2022 |
| ALI MUSDAF IBRAHIM | SCT221-C004-0681/2022 |

QUESTION ONE

**Read carefully and understand the case study below to answer the two questions. Be as innovative as possible.**

 The railway reservation system functions as follows;

The passenger is required to fill in a reservation form giving detail of his journey. The counter clerk ensures whether the place is available. If so, entries are made in a register, tickets are prepared, amount is computed and cash is accepted.

A booking statement is prepared in triplicate format from the reservation register. One copy is retained as office copy, the other is pasted to the compartment and third is passed on to the train conductor. Besides booking statement, cash statement is prepared at the end of each shift.

Prepare System Require Specification and system specification for above system problem.

**INTRODUCTION**

The Railway Reservation System aims to streamline the booking process, ensuring efficiency and accuracy in ticket reservations

**FUNCTIONAL REQUIREMENTS**

i)**Reservation Form:**

1. The passengers must provide essential journey details.

2. Form should include passenger information, travel date and destination.

ii)**Availability Check:**

1.The system must check seat availability based on the provided details.

2. Real-time updates on seat availability should be displayed.

**iii)Reservation Register:**

1.A centralized reservation register to record booking details.

2.Entries include passenger details and journey information.

iv)**Amount Computation:**

Automatic calculation of the total fare based on the distance and class of travel.

v)**Cash Handling:**

1.Secure handling of cash transactions by the counter clerk.

2.Intergration with a secure payment gateway for online reservations.

vi)**Cash Statement:**

1.End of shift cash statement summarizing the day’s transactions.

2.Ensures accountability and transparency in financial transactions.

**NON-FUNCTIONAL REQUIREMENTS**

**i)Security:**

1.Secure user authentication for counter clerks.

2.Encryption of sensitive passenger and financial data.

ii)**Reliability:**

Regular backups of reservation and financial data.

iii)Performance:

1.Efficient response times for seat availability checks and ticket generation.

2.Scalabilty to handle peak booking periods.

**SYSTEM SPECIFICATION**

i)**Hardware Requirements:**

1.Serves with sufficient processing power and storage.

2.Point of Sale terminals for counter clerks.

**ii)Software Requirements:**

1.Database management system for reservation data storage.

2.Secure payment gateway integration.

3.User-friendly interface for counter clerks and passengers.

iii**)Interfaces:**

1.Intergation with train’s onboard system for conductor communication.

2.Application Programming interface for real-time seat availability updates.

iv)**Protocols:**

1.Secure communication protocols for online reservations.

2.Stanadard protocols for communication between the reservation system and the other railway systems.

Design a prototype for the case study

i)**User Interface:**

1.Clean and intuitive interface for counter clerks and passengers.

2.Input forms for passenger details, journey information and payment.

ii)**Dashboard:**

1.Counter clerk dashboard displaying real-time seat availability and reservation status.

2.Passenger dashboard for managing bookings and viewing ticket details.

iii**)Automation:**

1.Automated seat availability checks and real-time updates.

2.Instant ticket generation upon successful reservation.

iv**)Reports:**

1.Comprehensive reports for daily, weekly and monthly transactions.

2.Visual representations of seat occupancy and revenue.

v)Security Features:

1.User authentication for counter clerks.

2.Multiple payment options for passenger convenience.

vi)**Feedback Mechanism:**

1.Passenger feedback form for contentious improvement.

2.System logs for monitoring and troubleshooting.

vii)**Scalability**:

Design to accommodate future system enhancements and increased user load.

This prototype ensures a user-friendly, secure and efficient railway reservation system that meets the needs of both passengers and railway staff.

**ASSIGNMENT TWO**

Discuss the following software testing techniques (walkthroughs, reviews and inspections, dynamic testing, Traceability matrices. Debugging environments)

1.**WALKTHROUGHS:**

Walkthroughs are informal, collaborative and step by step reviews of a system or its components or processes to identify potential issues. This technique involves a thorough and detailed review of documents, code, or design specifications.

Walkthroughs are primarily used for knowledge transfer, finding defects, and ensuring that the system meets its requirements. It's a collaborative approach involving developers, testers, and other stakeholders.

**2.** REVIEWS **AND INSPECTIONS:**

Reviews and inspections are formal activities where a group of people examines the work product to find defects and improve quality.

These techniques aim to identify and fix defects early in the development process. They also serve as a means of knowledge sharing, ensuring that the software aligns with organizational standards and requirements.

**3.Dynamic Testing:**

Dynamic testing involves the execution of the software to evaluate its behaviour during runtime. It includes various testing types such as unit testing, integration testing, system testing, and acceptance testing.

Dynamic testing aims to validate the functional and non-functional aspects of the software. It helps identify defects related to the system's behaviour, performance, security, and other quality attributes.

**4.Traceability Matrices:**

Traceability matrices are tables that link requirements to test cases, ensuring that each requirement is covered by one or more test cases. They provide a means to trace the development and testing process.

Traceability matrices help ensure comprehensive test coverage by mapping requirements to test cases. They also assist in impact analysis, change management, and verifying that all requirements are addressed during testing.

**5. Debugging Environments:**

Debugging is the process of identifying and fixing errors or defects in the software. Debugging environments provide tools and facilities to assist developers in locating and resolving issues in the code.

Debugging environments are crucial for developers during the software development phase. They help identify the root cause of defects, understand the program flow, and ensure the correctness of the code.

Discuss Software Quality Evaluation **–** Problems, Software standards, Certification, Software Tools support for Systems Engineering

**Software Quality Evaluation-Problems**

**Subjectivity**: Evaluating software quality can be subjective and dependent on individual perspectives. Different stakeholders may have varied opinions on what constitutes quality.

**Changing Requirements**: Software quality evaluation becomes challenging when requirements change frequently, leading to the need for continuous reassessment.

**Complexity:** With the increasing complexity of software systems, evaluating all aspects of quality, including functionality, performance, security, and usability, becomes intricate.

**Measuring Non-functional Requirements**: Non-functional requirements, such as maintainability and reliability, are harder to quantify and measure objectively.

**SOFTWARE STANDARDS**

Software standards are guidelines and specifications established to ensure consistency, interoperability, and quality in software development and related processes. These standards are created and maintained by various organizations and industry bodies to provide a common framework for developing, testing, and maintaining software. Adhering to these standards helps organizations produce reliable, secure, and maintainable software products.

**EXAMPLES**

1. **ISO/IEC 25000-** This standard provides a framework for software product quality, defining a set of quality characteristics such as functionality, reliability, usability, efficiency, maintainability, and portability. It includes models for quality management systems and evaluation processes.
2. **ISO/IEC 9126-** This standard, now part of ISO/IEC 25000, specifies a model for the quality of software products, focusing on characteristics like functionality, reliability, usability, efficiency, maintainability, and portability.
3. **Capability Maturity Model Integration (CMMI)-** Developed by the Software Engineering Institute, CMMI provides a set of best practices for process improvement in software development and maintenance. It has five maturity levels, each representing a different stage of process maturity.
4. **IEEE Standards-** The Institute of Electrical and Electronics Engineers (IEEE) has developed numerous standards related to software engineering, covering topics such as software requirements, design, testing, and maintenance.

**CERTICATION**

Certification in software engineering is a formal recognition that an individual or an organization has met a set of predefined standards and possesses a certain level of knowledge, skills, and competencies in the field of software development and engineering. These certifications serve various purposes, including demonstrating professional expertise, ensuring adherence to industry standards, and enhancing career opportunities.

**Types of Certifications**

**1.Vendor-Specific Certifications-** These certifications are tailored to specific technologies or platforms offered by particular vendors.

**2.** **Process and Methodology Certifications-** Certifications that emphasize methodologies, frameworks, and project management practices in software development.

**Software Tools Support for Systems Engineering**

**Requirements Management Tools:** Tools like IBM Engineering Requirements Management DOORS help manage and trace requirements throughout the development process.

**Modeling Tools**: Tools like IBM Engineering Systems Design Rhapsody and Enterprise Architect support modeling of systems and software architecture.

**Testing Tools:** Automation tools such as Selenium and JUnit help in creating and executing test cases, ensuring that the software meets specified requirements.

**Configuration Management Tools**: Tools like Git and SVN assist in managing versions and configurations of software systems.

**Challenges and Considerations**

**1.Interoperability of Tools:**

Ensuring that different tools used in systems engineering can seamlessly integrate and share data is a challenge.

**2.Cost and Resource Constraints:**

Acquiring and maintaining sophisticated tools can be expensive, and organizations need to consider budgetary constraints.

**3.Evolution of Standards:**

Software standards evolve, and organizations must adapt to the latest best practices to maintain or achieve certification.

**4.Balancing Functional and Non-functional Requirements:**

Striking a balance between evaluating functional requirements and non-functional aspects like usability and performance is crucial for comprehensive quality assessment.

**5.User-Centric Approach:**

Ensuring that software quality evaluation incorporates a user-centric approach to meet user expectations and satisfaction.

**Group Three:**  DiscussCASE tools as used in Web engineering techniques and process, standards and guidelines.

**Web Engineering Techniques and CASE Tools**

1.**Requirements Modeling-**Use of CASE tools for creating visual models like UML diagrams to represent web application requirements.

Tools-Enterprise Architect, IBM Rational Rose.

2.**Design and Prototyping-** CASE tools assist in designing the user interface and creating prototypes.

**2.Coding and Implementation-** Integrated Development Environments (IDEs) serve as CASE tools for coding, providing features like syntax highlighting, debugging, and version control.

Tools-Visual Studio Code

3.**Database Design-** CASE tools help design and model the database structure for web applications.

Tools-Oracle SQL Developer Data Modeler, MySQL Workbench.

4.**Testing-** CASE tools support test case generation, execution, and management.

**Challenges and Considerations**

**1.Tool Integration-**Ensuring seamless integration among various CASE tools to maintain consistency throughout the development lifecycle.

**2.Learning Curve-**CASE tools often have a learning curve, and teams need to invest time in training to use them effectively.

**3.Cost and Licensing-**Considerations around the cost of acquiring and maintaining licenses for CASE tools.

**Adaptability-**Ensuring that CASE tools can adapt to changes in web technologies and development practices.

Discuss Process Improvement (PI):- Quality and process standards and guidelines

Process Improvement (PI) in software development refers to the ongoing effort to enhance and optimize the processes involved in creating and delivering software products.

**Quality and Process Standards**

1. **ISO/IEC 9000 Series-** The ISO 9000 series provides a set of international standards for quality management systems. ISO 9001:2015, in particular, is widely used for software development process improvement.
2. **Capability Maturity Model Integration -**Developed by the Software Engineering Institute , CMMI provides a framework for process improvement. It defines maturity levels and key process areas.
3. **ISO/IEC 12207-**This standard focuses on the software life cycle processes, providing a framework for the development, acquisition, supply, operation, maintenance, and disposal of software systems.
4. IEEE Software Engineering Standards-The Institute of Electrical and Electronics Engineers (IEEE) has developed numerous standards covering various aspects of software engineering, including requirements, design, testing, and maintenance.

**Guidelines for Process Improvement**

1.**Define Clear Objectives:**

Clearly articulate the goals and objectives of the process improvement initiative to align with organizational objectives.

2.**Involve Stakeholders:**

Include key stakeholders from various departments and levels to ensure diverse perspectives and foster collaboration.

**3.Measure Performance:**

Use relevant metrics to assess the performance of existing processes and track improvements over time.

5**.Continuous Monitoring and Feedback:**

Implement continuous monitoring mechanisms and gather feedback to identify areas for improvement and adjust strategies accordingly.

**Challenges and Considerations:**

1.**Resistance to Change:**

Teams may resist changes to established processes. Effective communication and change management strategies are crucial.

**2.Resource Constraints:**

Limited resources, including time and budget, can impact the scope and pace of process improvement initiatives.

**3.Balancing Flexibility and Control:**

Achieving a balance between flexibility and control is essential, especially in dynamic environments.

**4.Sustainability:**

Maintaining the momentum of process improvement over the long term requires sustained commitment from the organization.

**5.Cultural Alignment:**

Ensuring that process improvements align with the organizational culture and values is critical for successful implementation