**Indian Institute of Information Technology Pune**

**NAME:** Deepesh Patil

**GROUP:** 2

**MIS:** 112215055

**YEAR/SEM:** 3rd Year/5th Sem

**TASK-3:**

**Preparation of Software Configuration Management and Risk Management-related documents.**

**1. Software Configuration Management (SCM) Document**

**1.1 Introduction**

* **Purpose:** The purpose of this document is to define the processes, procedures, and tools used to manage and control the configurations of the Online Course Reservation System throughout its development and maintenance lifecycle.
* **Scope:** This document applies to all phases of the software development lifecycle, including requirements gathering, design, development, testing, deployment, and maintenance.

**1.2 Configuration Management Plan**

**1.2.1 Configuration Identification**

* **Components:**
  + Source Code (Frontend, Backend)
  + Database Schemas
  + Configuration Files (Server settings, environment variables)
  + Documentation (SRS, Design Documents, User Manuals)
  + Test Cases and Scripts
* **Naming Conventions:**
  + Source code files: module\_name\_version.extension (e.g., course\_reservation\_v1.0.py)
  + Documents: doc\_type\_project\_name\_version.extension (e.g., SRS\_CourseReservation\_v1.0.docx)

**1.2.2 Version Control**

* **Version Control System (VCS):**
  + Tool: Git
  + Repository: Hosted on GitHub/GitLab/Bitbucket
* **Branching Strategy:**
  + main branch for stable releases
  + development branch for ongoing development
  + Feature branches for specific functionalities (e.g., feature-user-auth)

**1.2.3 Configuration Control**

* **Change Request Process:**
  + All change requests must be documented and approved by the configuration control board (CCB) before implementation.
* **Impact Analysis:**
  + Assess the impact of the proposed changes on existing configurations, timelines, and resources.
* **Change Implementation:**
  + Changes are implemented in the development environment and undergo testing before merging into the main branch.

**1.2.4 Configuration Status Accounting**

* **Documentation:**
  + Maintain a configuration status report that includes the current version, change history, and status of each configuration item.
* **Tracking Tools:**
  + Use of issue tracking systems like Jira or Trello for monitoring change requests and configuration statuses.

**1.2.5 Configuration Audits**

* **Audit Schedule:**
  + Regular audits (e.g., quarterly) to ensure compliance with the SCM process.
* **Audit Process:**
  + Review and verify the configuration items, change requests, and version histories against the SCM plan.

**1.3 Roles and Responsibilities**

* **Configuration Manager:** Oversees the SCM process, approves changes, and manages the configuration repository.
* **Development Team:** Implements changes, follows version control practices, and documents modifications.
* **Quality Assurance (QA) Team:** Verifies that the configurations comply with the specifications and conducts audits.

**1.4 Tools and Technologies**

* **Version Control:** Git
* **Issue Tracking:** Jira/Trello
* **Build Automation:** Jenkins/Travis CI
* **Documentation:** Confluence/Google Docs

**2. Risk Management Document**

**2.1 Introduction**

* **Purpose:** The purpose of this document is to identify, assess, and manage risks associated with the development and deployment of the Online Course Reservation System.
* **Scope:** This document covers risks in project management, software development, deployment, and maintenance phases.

**2.2 Risk Identification**

* **Technical Risks:**
  + **Integration Issues:** Challenges in integrating the system with third-party payment gateways or external APIs.
  + **Security Vulnerabilities:** Potential security flaws in user authentication, data storage, and payment processing.
  + **Performance Bottlenecks:** The system may not handle high traffic or large datasets efficiently.
* **Project Management Risks:**
  + **Scope Creep:** Uncontrolled changes or continuous growth in project scope leading to delays and budget overruns.
  + **Resource Availability:** Lack of availability of skilled developers, testers, or other critical resources.
  + **Timeline Slippage:** Delays in meeting project milestones or deadlines.
* **Operational Risks:**
  + **Data Loss:** Risk of losing data due to software bugs, database corruption, or hardware failure.
  + **User Adoption:** Low adoption rate by the target audience due to usability issues or lack of awareness.
* **External Risks:**
  + **Compliance Issues:** Non-compliance with data protection regulations like GDPR or PCI-DSS standards.
  + **Vendor Dependence:** Reliance on third-party services for critical functionalities like payment processing.

**2.3 Risk Assessment**

| **Risk** | **Probability** | **Impact** | **Priority** | **Mitigation Strategy** |
| --- | --- | --- | --- | --- |
| Integration Issues | Medium | High | High | Early integration testing, modular design |
| Security Vulnerabilities | High | High | High | Regular security audits, use of secure coding practices |
| Performance Bottlenecks | Medium | Medium | Medium | Performance testing, scalable architecture |
| Scope Creep | Medium | High | High | Strict change management, clear requirement documentation |
| Resource Availability | Low | High | Medium | Cross-training team members, resource planning |
| Timeline Slippage | Medium | Medium | Medium | Agile methodology, regular progress reviews |
| Data Loss | Low | High | High | Regular backups, redundancy |
| User Adoption | Medium | Medium | Medium | User training, feedback loops |
| Compliance Issues | Medium | High | High | Legal consultation, regular compliance checks |
| Vendor Dependence | Medium | Medium | Medium | Multiple vendor strategy, contingency planning |

**2.4 Risk Mitigation Strategies**

* **Integration Issues:** Conduct early integration testing and use a modular design approach to isolate and resolve integration problems.
* **Security Vulnerabilities:** Implement secure coding practices, conduct regular security audits, and ensure all sensitive data is encrypted.
* **Performance Bottlenecks:** Perform load testing and optimize code for performance, implement a scalable architecture.
* **Scope Creep:** Implement strict change management processes and maintain clear and detailed requirement documentation.
* **Resource Availability:** Ensure backup resources are available and provide cross-training to mitigate the impact of resource unavailability.
* **Timeline Slippage:** Use Agile development methodologies with regular sprints and progress reviews to stay on track.
* **Data Loss:** Implement regular data backups, use redundancy in storage, and test disaster recovery plans.
* **User Adoption:** Conduct user training sessions, gather feedback during development, and make iterative improvements.
* **Compliance Issues:** Consult legal experts on compliance matters and conduct regular reviews to ensure adherence to regulations.
* **Vendor Dependence:** Develop contingency plans and have alternative vendors ready to mitigate dependency risks.

**2.5 Risk Monitoring and Reporting**

* **Monitoring:** Regularly review risk status and mitigation strategies during project meetings.
* **Reporting:** Document all identified risks, their status, and actions taken in a risk register. Update stakeholders regularly.

**2.6 Roles and Responsibilities**

* **Project Manager:** Oversees risk management activities, ensures risk mitigation strategies are implemented.
* **Risk Management Team:** Identifies, assesses, and monitors risks throughout the project lifecycle.
* **Development Team:** Implements risk mitigation strategies in the development process.
* **QA Team:** Tests the system for vulnerabilities and ensures compliance with security and performance standards.