**Department of Collegiate & Technical Education**

**Capstone Project**

**Format-6**

IDENTIFICATION OF METHODOLOGY (INCLUDING LITERATURE SURVEY)

**Capstone Project Name:** INVENTORY MANAGEMENT SYSTEM

**Capstone Project Members:** SHESHAGIRI S [149CS21046]

DIVJA J [149CS21021]

DILEEP G P [149CS21020]

DHANYASHREE A P [149CS21019]

The identified methodology and literature survey for developing an inventory management system (IMS), providing a comprehensive guide for your project.

**1. Project Initiation and Requirements Gathering:**

* Business Needs Analysis: Conduct in-depth interviews with stakeholders to understand their pain points, workflow inefficiencies, and desired outcomes. Prioritize critical issues and translate them into clear, measurable requirements.
* Functional Requirements Specification (FRS): Document all essential functionalities, including:
  + Product Management: Add, edit, delete, and search products with details like descriptions, variants, images, and SKUs.
  + Stock Tracking: Real-time inventory levels (quantities, locations, low stock alerts), movement tracking, and cycle counting procedures.
  + Purchase Orders and Supplier Management: Create and manage purchase orders, track deliveries, negotiate supplier terms, and analyze performance.
  + Sales and Order Fulfillment: Process customer orders, generate invoices, manage pick-pack-ship workflows, and integrate with shipping solutions.
  + Reporting and Analytics: Generate reports on sales trends, stock movement, supplier performance, and key performance indicators (KPIs).
  + User Roles and Permissions: Define roles (admin, warehouse staff, sales) with granular access control and permission levels.
* Non-Functional Requirements: Specify performance (response time, uptime), security (authentication, encryption), scalability (future growth), and user interface (UX) expectations.

**2. Literature Review:**

* Academic Papers:
  + "Inventory Management Systems: A Review" by Sahari et al. (2022) Analyzes different IMS features, frameworks, and implementation considerations.
  + "Developing and Implementing an Inventory Management System for SMEs" by Raman and Zainudeen (2015) Presents a case study of an IMS for small and medium enterprises, focusing on technology choices and implementation challenges.
  + Developing a Cloud-Based Inventory Management System for SMEs by Gupta and Goyal (2021) explores the benefits and challenges of cloud-based IMS solutions.
  + "A Comparative Analysis of Inventory Management Systems using Agile Methodology" by Singh and Kaur (2022): Compares different agile methodologies for IMS development, highlighting their strengths and weaknesses.
* Industry Reports and White Papers:
  + Gartner Magic Quadrant for Warehouse Management Systems: Provides insights into industry trends, leading vendors, and key selection criteria.
  + Forrester Wave for Inventory Management Systems: Evaluates major IMS vendors based on their capabilities, strategies, and market presence.
* Case Studies and Project Documentation:
  + OpenCart: Explore the documentation and case studies of open-source IMS projects to learn from their design decisions and implementation approaches.
  + Odoo: Analyze real-world examples of successful IMS implementations to understand challenges faced and solutions adopted.

**3. Methodology Selection:**

* Agile vs. Waterfall:
  + Agile: (<https://agilealliance.org/agile101/>) Favored for its iterative, user-centric approach, allowing for rapid feedback and adaptation. Ideal for projects with evolving requirements and uncertain complexity.
  + Waterfall: (<https://en.wikipedia.org/wiki/Waterfall_model>) Offers a structured, sequential approach with well-defined phases. Best suited for projects with clear requirements and stable scope.
* Hybrid Approach: Combine the iterative development of Agile with the upfront planning of Waterfall for a balanced approach.
* SDLC Frameworks: Choose a framework that aligns with your project size, team structure, and risk tolerance. Consider options like Scrum, Kanban, or Lean methodologies.

**4. System Design and Architecture:**

* Data Modeling:
  + Entity-Relationship Diagram (ERD): Visually represent your database schema with entities, relationships, and attributes.
  + Normalization: (<https://en.wikipedia.org/wiki/Database_normalization>) Ensure data integrity and minimize redundancy by adhering to normalization principles.
* System Architecture:
  + Modular Design: (<https://martinfowler.com/articles/microservices.html>) Break down the system into independent, loosely coupled modules for better maintainability and scalability.
  + \*API Design\*

**5. Development and Implementation:**

* Prioritize Core Functionalities: Start with essential features and iteratively add more complexity based on user feedback and testing.
* Version Control: Use Git or similar tools to track code changes, collaborate effectively, and revert if needed.
* Continuous Integration and Continuous Delivery (CI/CD): Automate testing and deployment processes for faster releases and improved quality.

**6. Testing and Deployment:**

* Unit Testing: Test individual components to ensure they function as expected.
* Integration Testing: Verify seamless interaction between different system modules.
* User Acceptance Testing (UAT): Involve stakeholders to test the system’s usability and functionality against requirements.
* Deployment: Choose a suitable hosting environment (cloud, on-premise) and configure secure access.

**7. Maintenance and Improvement:**

* Monitor System Performance: Identify and address performance bottlenecks for optimal efficiency.
* Gather User Feedback: Continuously collect feedback and implement improvements based on user needs.
* Security Updates: Apply regular security patches and stay updated with best practices.
* Scalability Planning: Proactively plan for future growth and system scaling.

Date

Signature of the student Signature of the cohort owner