**Department of Computer Science & Engineering**

**Software Engineering**

**Test Plan Document**

**UE22CS341A**

**Name: Sai Ganesh Name: Vignesh .N**

**SRN: PES1UG22CS505 SRN: PES1UG23CS836**

**Billing System for Department Store**

1. **Introduction**

**Project**: Billing System for Department Store

**Purpose**: The purpose of this testing document is to ensure that the Billing System for the department store meets all specified functional requirements, providing an accurate and efficient solution for billing, inventory management, and user operations. By conducting a series of structured manual tests, this document aims to identify and address potential issues early in the implementation phase, reducing errors and ensuring smooth operational performance. The ultimate goal is to enhance customer satisfaction and streamline daily processes for store staff.

**Scope**: This testing plan covers all critical modules and workflows within the billing system, ensuring seamless integration and robust functionality across the following areas:

* **Billing Module**: Includes functionalities like item scanning, cart management, discount application, tax calculations, payment processing, and receipt generation.
* **Inventory Management Module**: Manages inventory levels, ensures real-time updates for stock quantities post-purchase, and provides alerts for items with low stock.
* **User Management Module**: Manages customer and staff accounts, supports role-based access control, and ensures secure handling of customer data.

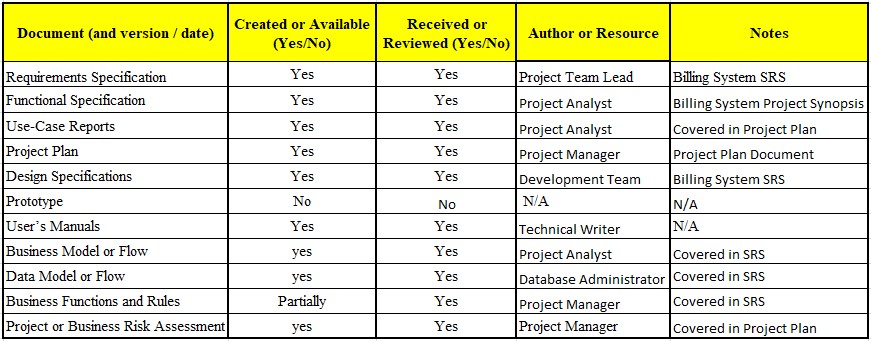
**Each module will be tested through:**

* **Unit Testing**: To validate individual functionalities (e.g., scanning items, applying discounts, calculating tax) and ensure they perform as expected under isolated conditions.
* **Integration Testing**: To test the interactions between different modules, such as linking the cart, inventory, and checkout processes to ensure smooth transitions across workflows.
* **System Testing**: To verify the entire billing workflow, ensuring that all components work together from start to finish, including edge cases and error-handling scenarios.

**Additional Objectives:**

* **Data Accuracy**: Ensures the correct calculation of item prices, taxes, and discounts, and verifies accurate updates to inventory and financial records after transactions.
* **Usability**: Confirms that the interface is user-friendly for both customers and store staff, with clear prompts, minimal errors, and ease of navigation through the billing and checkout process.
* **Security and Compliance**: Validates that the system complies with data security standards, protects customer data, and restricts access based on user roles.
* **Reliability and Performance**: Assesses system stability under varying loads, especially during peak hours, and ensures quick response times for scanning and processing payments.

**Expected Outcome**: By the end of this testing process, the billing system should operate reliably and efficiently, providing an intuitive user experience with accurate transaction handling, secure data management, and smooth integration across all modules. This testing process is designed to support the successful deployment of the billing system in a real-world department store environment, enhancing both customer service and operational efficiency.



**2. Test Case Template**

**1. Test Case: UT\_01**

|  |  |
| --- | --- |
| Test Case ID | UT\_01 |
| Name of the Module | Billing |
| Test Case Description | Verify the calculation of the total bill with multiple items. |
| Pre-conditions | System is connected to the inventory database. |
| Test Steps | 1. Add Item A (Price: $50, Quantity: 2). 2. Add Item B (Price: $30, Quantity: 1). 3. Click 'Calculate Total'. |
| Test Data | Item A, Item B, Quantities. |
| Expected Results | Total amount should be $130. |
| Actual Results | (To be filled after testing) |
| Test Result | (Pass/Fail) |

**2. Test Case: UT\_02**

|  |  |
| --- | --- |
| Test Case ID | UT\_02 |
| Name of the Module | User Management |
| Test Case Description | Verify login functionality with valid credentials. |
| Pre-conditions | User account is active. |
| Test Steps | 1. Navigate to the login page. 2. Enter valid username and password. 3. Click 'Login'. |
| Test Data | Username: cashier1, Password: pass123. |
| Expected Results | User should be logged in and redirected to the dashboard. |
| Actual Results | (To be filled after testing) |
| Test Result | (Pass/Fail) |

**3. Test Case: IT\_01**

|  |  |
| --- | --- |
| Test Case ID | IT\_01 |
| Name of the Module | Inventory Management |
| Test Case Description | Verify stock update after a sale. |
| Pre-conditions | Product has available stock. |
| Test Steps | 1. Sell 5 units of Product X. 2. Verify stock level in database. |
| Test Data | Product X, Initial Stock: 20 units, Quantity Sold: 5 units. |
| Expected Results | Stock should decrease to 15 units. |
| Actual Results | (To be filled after testing) |
| Test Result | (Pass/Fail) |

**4.** **Test Case: UT\_03**

|  |  |
| --- | --- |
| Test Case ID | UT\_03 |
| Name of the Module | Billing |
| Test Case Description | Verify behaviour when an invalid discount code is applied. |
| Pre-conditions | Discount feature is enabled. |
| Test Steps | 1. Add items to cart. 2. Apply discount code 'INVALIDCODE'. 3. Click 'Apply'. |
| Test Data | Discount Code: INVALIDCODE. |
| Expected Results | Error message "Invalid discount code" should be displayed. |
| Actual Results | (To be filled after testing) |
| Test Result | (Pass/Fail) |

**5.** **Test Case: ST\_01**

|  |  |
| --- | --- |
| Test Case ID | ST\_01 |
| Name of the Module | Reports |
| Test Case Description | Verify daily sales report generation. |
| Pre-conditions | Sales data is available for the day. |
| Test Steps | 1. Go to 'Reports' section. 2. Select 'Daily Sales Report'. 3. Click 'Generate'. |
| Test Data | Date: 2024-10-19. |
| Expected Results | Report should include all transactions for the selected date. |
| Actual Results | (To be filled after testing) |
| Test Result | (Pass/Fail) |

**6.** **Test Case: IT\_02**

|  |  |
| --- | --- |
| Test Case ID | IT\_02 |
| Name of the Module | Payment Processing |
| Test Case Description | Verify successful transaction with valid card details. |
| Pre-conditions | Payment gateway is accessible. |
| Test Steps | 1. Add items to cart. 2. Select 'Card Payment'. 3. Enter valid card details and submit. |
| Test Data | Card Number: 1234 5678 9123 4567, Expiry: 12/25, CVV: 123. |
| Expected Results | Transaction should be successful and receipt generated. |
| Actual Results | (To be filled after testing) |
| Test Result | (Pass/Fail) |

**7.** **Test Case: UT\_04**

|  |  |
| --- | --- |
| Test Case ID | UT\_04 |
| Name of the Module | User Management |
| Test Case Description | Verify failed login with invalid credentials. |
| Pre-conditions | User account exists. |
| Test Steps | 1. Navigate to login page. 2. Enter invalid username/password. 3. Click 'Login'. |
| Test Data | Username: wrongUser, Password: wrongPass. |
| Expected Results | Error message "Invalid username or password" should be displayed. |
| Actual Results | (To be filled after testing) |
| Test Result | (Pass/Fail) |

**8.** **Test Case: ST\_02**

|  |  |
| --- | --- |
| Test Case ID | ST\_02 |
| Name of the Module | Billing |
| Test Case Description | Verify invoice generation with detailed breakdown. |
| Pre-conditions | System is operational, Items are in cart. |
| Test Steps | 1. Add items to cart. 2. Click 'Generate Invoice'. 3. Check invoice details. |
| Test Data | Item A, Item B. |
| Expected Results | Invoice should include item details, quantity, price, discounts, and total amount. |
| Actual Results | (To be filled after testing) |
| Test Result | (Pass/Fail) |

**10.** **Test Case: UT\_05**

|  |  |
| --- | --- |
| Test Case ID | UT\_05 |
| Name of the Module | Inventory Management |
| Test Case Description | Verify alert for low stock level. |
| Pre-conditions | Stock threshold is set. |
| Test Steps | 1. Reduce stock of Product Y below threshold. 2. Check for alert. |
| Test Data | Product Y, Threshold: 5 units. |
| Expected Results | Alert "Low stock for Product Y" should be displayed. |
| Actual Results | (To be filled after testing) |
| Test Result | (Pass/Fail) |

**11. Test Case: IT\_03**

|  |  |
| --- | --- |
| Test Case ID | IT\_03 |
| Name of the Module | **Payment Processing** |
| Test Case Description | **Verify transaction failure with expired card.** |
| Pre-conditions | **Payment gateway is accessible.** |
| Test Steps | **1. Add items to cart. 2. Select 'Card Payment'. 3. Enter expired card details and submit.** |
| Test Data | **Card Number: 9876 5432 1098 7654, Expiry: 12/22, CVV: 321.** |
| Expected Results | **Transaction should be declined with message "Card expired."** |
| Actual Results | **(To be filled after testing)** |
| Test Result | **(Pass/Fail)** |

**Conclusion**

The testing phase for the Billing System for the department store is crucial in verifying that all functionalities work as intended and that the system meets the required standards for performance, accuracy, and usability. Through unit, integration, and system testing, we have systematically validated each core module, including Billing, Inventory Management, and User Management. This testing ensures that all components work cohesively to provide a seamless, secure, and efficient experience for users.

In summary, this testing process has significantly enhanced the system’s quality and readiness for use, ultimately contributing to improved operational efficiency, customer satisfaction, and overall business performance. Future maintenance and periodic testing are recommended to adapt the system to evolving needs and to ensure continued optimal performance.

**Test Deliverables :**

**The following deliverables will be produced as part of the testing process:**

1. **Software Requirements Specification (SRS)**: The document outlining the functional and non-functional requirements of the Billing System, which will serve as the baseline for the tests.
2. **Project Synopsis**: A summary of the project's objectives, scope, and expected outcomes, used to align testing with the overall project goals.
3. **Project Plan and Architecture Design**: Documentation detailing the project timeline, roles, responsibilities, and the system’s high-level design. This will be used to validate the system’s architecture against its intended design.
4. **Test Cases**: Documentation of all test scenarios, covering both functional and nonfunctional requirements (such as performance), and including detailed steps for executing each test.
5. **Test Summary Report**: A comprehensive report summarizing the results of all tests performed, identifying any issues, and providing recommendations for further action.

**Environmental Needs**

* 1. **Hardware:**

|  |  |
| --- | --- |
| **Machine** | **Configuration** |
| Server | 4 CPU, 16GB RAM, 500GB HDD |
| Client | POS Terminal, 8GB RAM |

* 1. ***Software:***

|  |  |  |
| --- | --- | --- |
| **Machine** | **OS** | **Software** |
| Server | Linux/Windows | MySQL, Java Runtime Environment |
| Client | Windows | Java Swing application, Printer |

* 1. ***Tools:***
* **Version Control**: Git
* **Task Tracking**: Jira or Trello

1. **Staffing**

***5.1 Responsibilities:***

* **Test Plan**: Your Name
* **Test Case**: Test Lead
* **Manual Testing**: Test Team
* **Automated Testing**: Automation Engineer

***5.2 Training:***

* Training will be provided on test automation tools and specific use cases for testing the billing system.

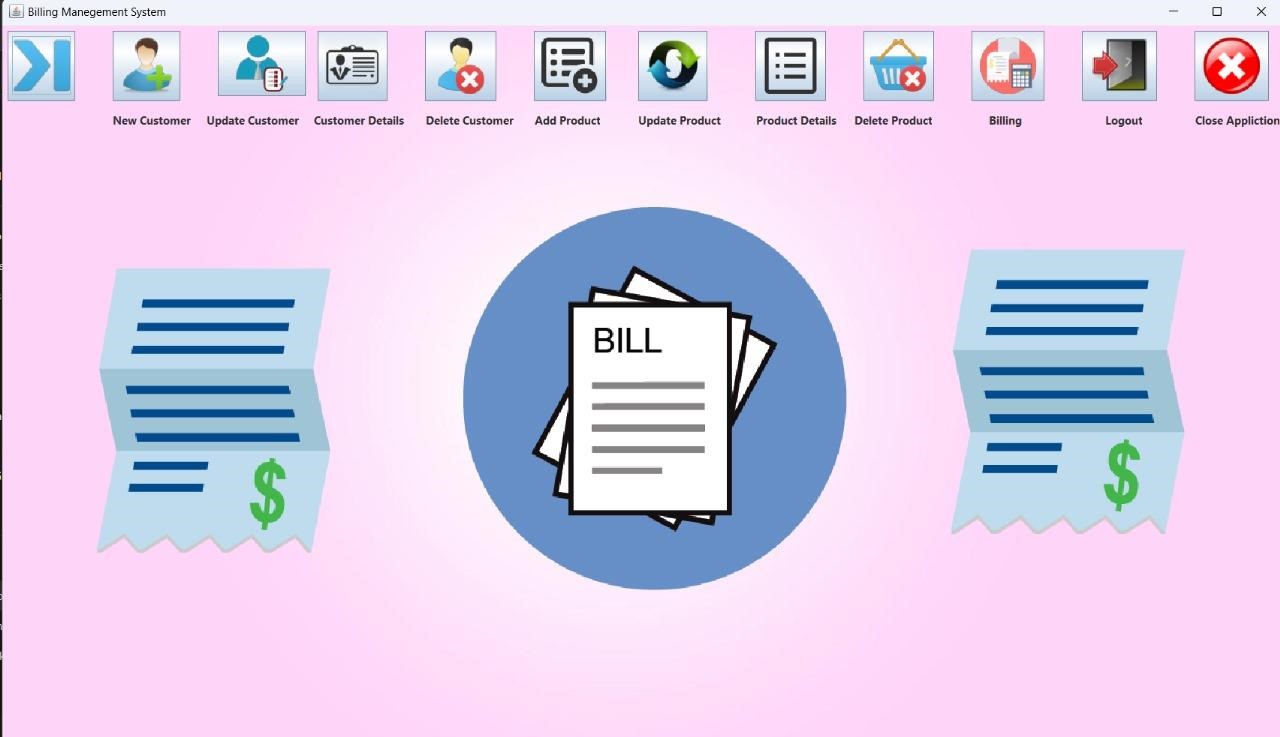
**6. Dependencies/Risks**

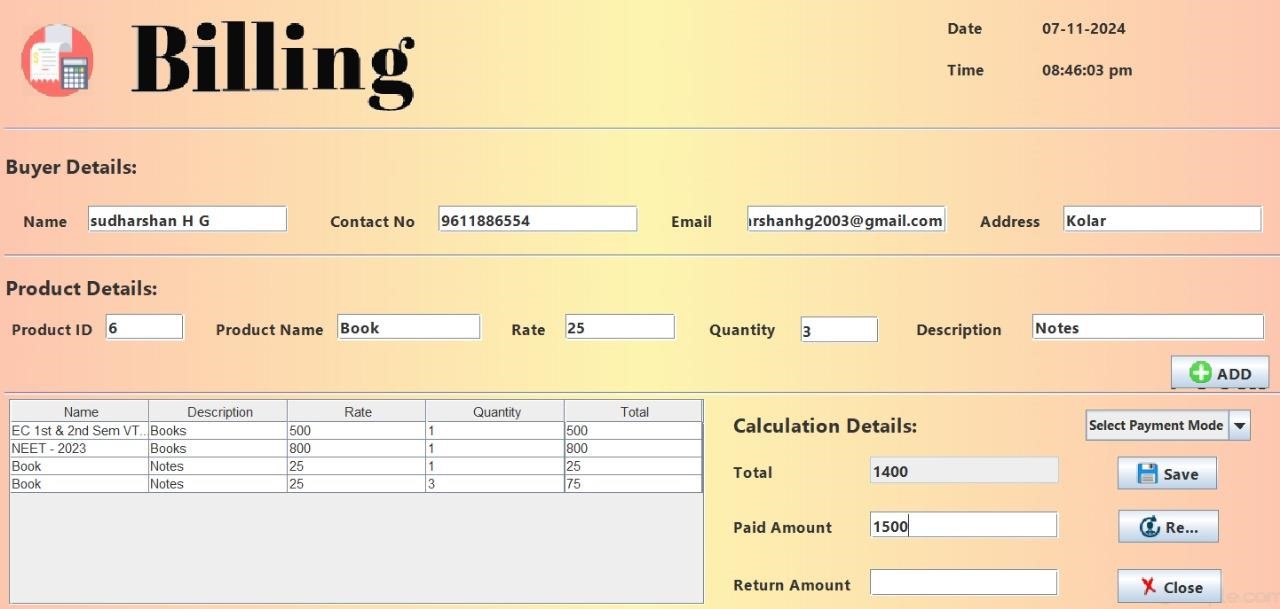
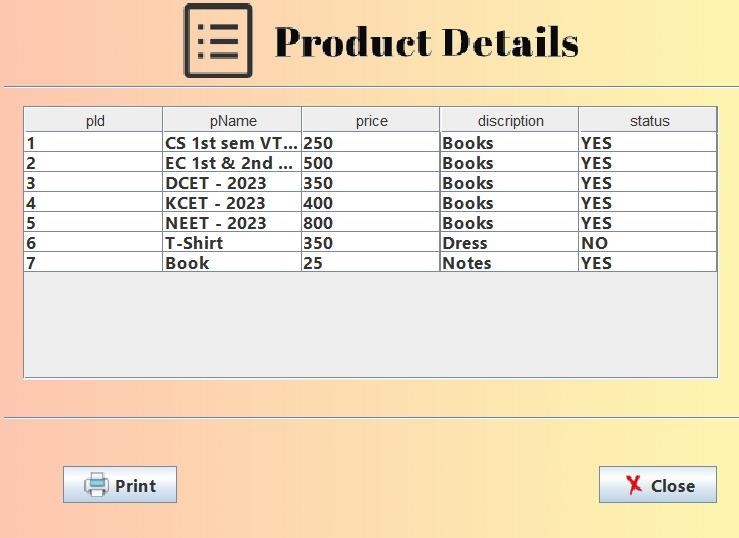
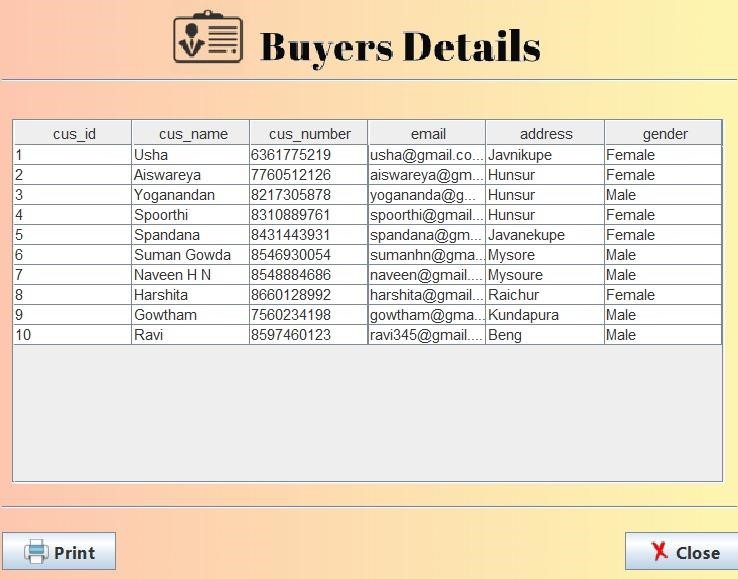
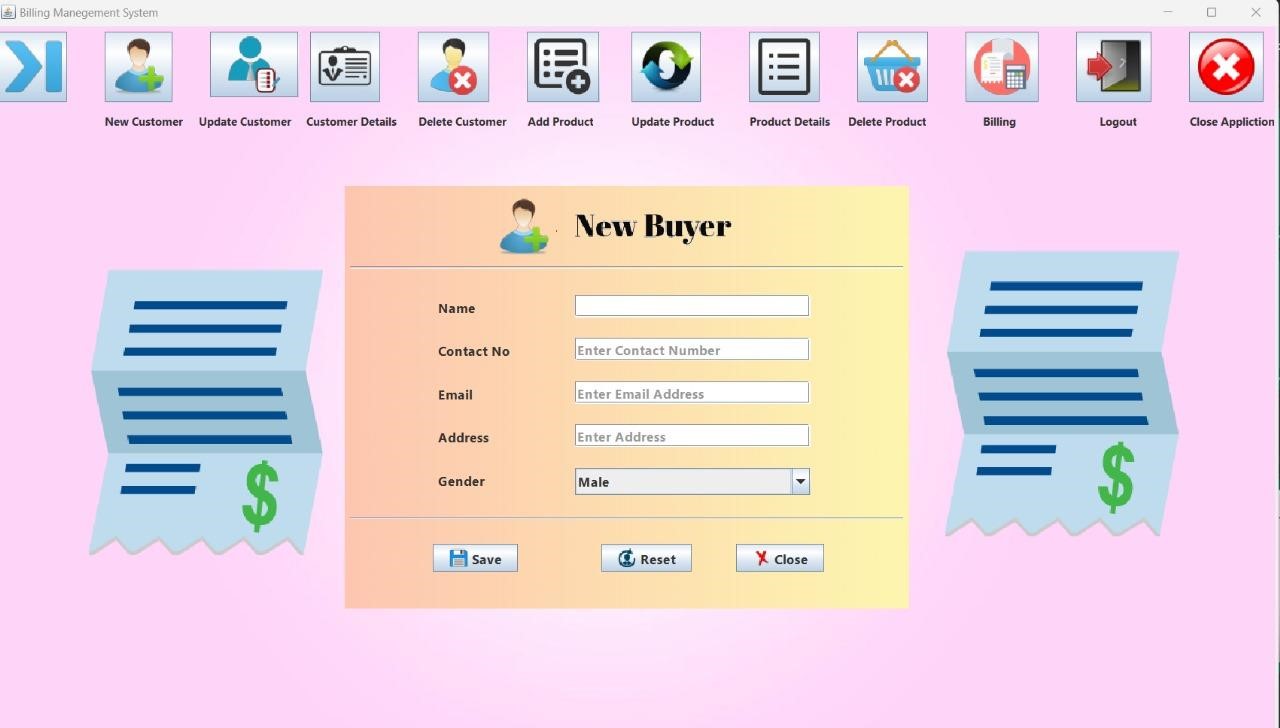
* 1. **Dependencies**:
* System components must be integrated on time for testing.
* Database backups must be scheduled regularly.
  1. **Risks**:
* Performance issues with high loads.
* Security vulnerabilities in data handling.

### **7. Schedule and Milestones**

|  |  |
| --- | --- |
| **Testing Type** | **Schedule Date** |
| Unit Testing | October 10, 2024 |
| Integration Testing | October 15, 2024 |
| System Testing | October 20, 2024 |
| Acceptance Testing | October 27, 2024 |

**SCREENSHOT OF OUTPUT**





**DATABASE OUTPUT SCREENSHOT**

