DEPARTMENT OF COMPUTER ENGINEERING

NAME:-Shashwat Shah

SAPID:-60004220126

SOFTWARE TESTING & QUALITY ASSURANCE(STQA)

EXPERIMENT NO.01

AIM:- To make a test case verification document for a smart hotel management system.

THEORY:

Implementing Test-Driven Development (TDD) for Payment Processing in a Smart Hotel Management System

To effectively develop a robust and secure payment processing system for a Smart Hotel Management System (SHMS), adopting a Test-Driven Development (TDD) approach is crucial. This method ensures that the system is built around precise specifications, is thoroughly tested from the outset, and accommodates all potential payment scenarios. Below is a refined, step-by-step theory of how TDD can be implemented for this use case:

1. Identifying Payment Requirements

The first phase of TDD involves understanding the specific payment processing needs for a hotel management system. Key requirements include:

- **Integration with Payment Gateways**: Define the preferred payment providers (e.g., Stripe, PayPal, or regional solutions).
- **Diverse Payment Methods**: Ensure compatibility with various payment methods, such as credit cards, digital wallets, online bank transfers, and even cryptocurrencies.
- **Payment Flow**: Clarify when payments are processed—whether at booking, check-in, or checkout—and how partial payments, full payments, or deposits are handled.
- **Error Handling Protocols**: Specify how the system should respond to issues like transaction declines, network failures, or potential fraud detection.

2. Defining Payment Requirements

Once the payment requirements are identified, they should be clearly defined:

- **Secure Gateway Integration**: Payment gateways must be implemented securely to protect sensitive transaction data.
- **Support for Multiple Payment Methods**: The system should seamlessly handle various forms of payments.

- **Error Handling Strategies**: Outline how errors such as invalid input, timeouts, declined payments, and network issues will be handled to ensure smooth operations.

3. Writing Test Cases

In this phase, specific and thorough test cases are created to evaluate the payment system. These test cases should include:

- **Positive Scenarios**: For example, successful payments using valid credit cards or PayPal for booking a premium room.
- **Negative Scenarios**: For example, declined payments due to insufficient funds or incorrect card details.
- **Edge Cases**: Including large transactions, payment timeouts, and handling payment gateway unavailability.

4. Running Initial Tests (Red Phase)

The TDD process begins by running the initial tests, which should all fail since the payment functionality has not yet been implemented. This failure marks the "Red" phase, highlighting the gap between the current state and the desired system functionality.

5. Writing the Code (Green Phase)

In the "Green" phase, the actual payment functionality is implemented based on the test cases. This involves:

- **Payment Gateway Integration**: Implementing secure communication with chosen payment providers (e.g., Stripe, PayPal).
- **Multi-Method Payment Support**: Enabling the system to handle different payment types, such as credit cards, wallets, etc.
- **Security Compliance**: Ensuring that the payment processing follows industry security standards like PCI-DSS.
- **Handling Secure User Inputs**: Ensuring that all user data, including billing details, are handled securely.

Once the code is written, tests are rerun to ensure they pass successfully.

6. Refactoring Code (Refactor Phase)

After successful implementation, the code is refactored to enhance its efficiency and maintainability without compromising the functionality. The tests should still pass, ensuring that no regressions occur during the refactoring process.

7. Error Handling and Edge Case Testing

Additional test cases are developed to focus on specific error scenarios:

- **Unavailable Payment Gateway**: What happens when the payment gateway is temporarily down?
- **Incorrect Payment Information**: How does the system handle expired or invalid credit card details?
- **Excessive Transaction Amounts**: Testing how the system manages large bookings or transactions exceeding available funds.

Network-related issues like timeouts or connection failures must also be tested.

8. Automating the Tests

Automated testing is set up to ensure that payment-related tests are continuously executed as part of the development lifecycle. By integrating automated tests with continuous integration/continuous delivery (CI/CD) pipelines, payment functionality is consistently validated with each code update.

9. Integration Testing

The payment system must integrate seamlessly with other components of the hotel management system. Integration testing should cover:

- **Booking and Payment Flow**: Ensure the booking process correctly triggers payments and updates the system's records.
- **Room Availability and Guest Records**: Verify that room availability and guest data are accurately updated after a payment is successful.
- **Inventory Management**: Ensure the system appropriately handles booking, cancellations, and room inventory updates in relation to payments.
- **Order Processing and Checkout**: Test how the payment system integrates with checkout operations, ensuring smooth transitions.

10. Security Testing

A robust security strategy is essential to protect payment transactions and sensitive data. Key areas to test include:

- **Encryption**: Ensure that data transmission between the hotel system and the payment gateway is encrypted.
- **Compliance**: Verify that the system complies with PCI-DSS and other relevant security standards.
- **Vulnerability Scans**: Regular scans should be performed to detect and mitigate risks like SQL injection or cross-site scripting.
- **Fraud Prevention**: Test how the system responds to fraudulent transactions and ensure mechanisms are in place to prevent them.

11. Documentation

Finally, comprehensive documentation of the payment testing process is necessary. This should include:

- **Detailed Test Cases**: Document each test scenario, including expected outcomes.
- **Payment Methods and Gateways**: Outline the specific payment methods and gateway solutions used in the system.
- **Testing Environments**: Note any specific configurations required for testing (e.g., sandbox accounts, test servers).
- **Security Testing Results**: Document findings from security tests and any vulnerabilities that were identified and resolved.

TEST CASE SCENARIO 2 - FEATURES

Test	Test Objective	Precondition	Steps	Test data	Expected result	Postcondition	Pass / Fail
TC_001	Ensure that a user cannot access the system using incorrect login details.	None	Open the login page.	Invalid username and incorrect password	Error message should appear stating that login credentials are incorrect.	Error message displayed indicating invalid credentials	Pass
			Enter an incorrect username and password.				
			Click the "Login" button.				

TC_002	Confirm that a user can successfully log in using correct credentials.	The user must have an active account in the system.	Open the login page.	Correct username and password	The system should successfully log in the user and navigate to the main dashboard or home screen	User successfully logged in and redirected to the home page.	Pass
			Input a valid username.				
			Input the correct password				
			Click the "Sign In" button.				
TC_003	Validate "Remember Password" option retains login.	Active user account required.	Open login page	Correct username & password	User stays logged in after reopening the browser.	User remained logged in.	Pass
			Select "Remember Me."				
			Enter valid credentials.				
			Click "Login."				
			Close and reopen the browser.				
TC_004	Ensure the user can log out successfully.	Users must be logged in.	Click the "Logout" button or link.	User is redirected to the login page.	Logged-in user session	User is redirected to the login page.	Pass

TEST CASE SCENARIO 2 - FEATURES

Test Case ID	Use Case	Test Objective	Precondition	Steps	Test Data	Expected Result	Pass/Fail
TC_MOB_1_1	Initiate Mobile Recharge (US_MOB_01)	Verify successful mobile recharge functionality.	User has a registered account with sufficient balance.	1. User opens the mobile service provider app.	Valid phone number, Valid recharge amount (e.g., $10)	Successful recharge confirmation message with details (amount, phone number).	Pass
				2. User navigates to the "Mobile Recharge" section.			
				3. User enters their phone number.			
				4. User selects a valid recharge amount.			
				5. User confirms the recharge.			
TC_MOB_1_2	Initiate Mobile Recharge (US_MOB_01)	Verify error handling for invalid recharge amount.	User has a registered account with sufficient balance.	1. User opens the mobile service provider app.	Valid phone number, Invalid recharge amount (e.g., -10)	Error message indicating invalid recharge amount.	Pass
				2. User navigates to the "Mobile Recharge" section.			
				3. User enters their phone number.			
				4. User enters an			

				invalid recharge amount (e.g., -10).			
				5. User confirms the recharge.			
TC_BILL_2_1	Initiate Bill Payment (US_BILL_02)	Verify successful bill payment using UPI with sufficient balance.	User has a registered account with linked utility services and a valid UPI account with sufficient balance.	1. User opens the mobile service provider app.	Valid UPI address, Sufficient balance in UPI account	Successful bill payment confirmation message with details (amount, service, date).	Pass
				2. User navigates to the "Bill Payment" section			
				3. User selects the specific utility service and billing period.			
				4. User selects UPI as the payment method.			
				5. User enters their UPI PIN.			
TC_LOAN_1_1	Browse Loans (US_LOAN_01)	Verify user can browse available loan options.	User has a registered account.	1. User opens the mobile service provider app.	N/A	List of available loan options with details (amount range, interest rate, terms) is displayed.	Pass

				2. User navigates to the "Loans" section.			
TC_LOAN_2_1	Apply for Loans (US_LOAN_02)	Verify user can initiate the loan application process.	User has a registered account and meets eligibility criteria for a specific loan.	1. User opens the mobile service provider app.	Valid user information meeting loan eligibility criteria	User is directed to the loan application form to complete further details.	Pass
				2. User navigates to the "Loans" section.			
				3. User selects a specific loan product.			
				4. User initiates the application process by filling out required information.			
TC_TIC_1_1	Browse Tickets for Movies (US_TIC_01)	Verify user can browse available movie tickets for a particular showtime.	User has a registered account.	1. User opens the mobile service provider app	Movie title, Show date & time	List of available seats for the selected showtime is displayed.	Pass
				2. User navigates to the "Tickets" section (Movies)			
				3. User selects a desired movie.			
				4. User selects a			

				specific showtime.			
TC_TIC_2_1	Book Tickets (US_TIC_02)	Verify successful movie ticket booking with valid payment.	User has a registered account with a valid payment method (e.g., credit card) and sufficient balance.	1. User opens the mobile service provider app.	Valid payment method, Sufficient balance	Booking confirmation message with ticket details (movie, showtime, seats) is displayed.	Pass
				2. User navigates to the "Tickets" section (Movies).			
				3. User selects a desired movie			
				4. User selects a specific showtime.			
				5. User selects available seats.			
				6. User confirms the booking and completes the payment.			
TC_TIC_2_2	Book Tickets (US_TIC_02)		User has a registered account with a valid payment method.	1. User opens the mobile service provider app.	N/A	Error message indicating seat selection conflict and suggesting alternative seats.	Pass
				2. User navigates to the "Tickets"			

				section (Movies).			
				3. User selects a desired movie.			
				4. User selects a specific showtime.			
				5. User attempts to select already booked seats.			
TC_INS_1_1	View Insurance (US_INS_01)	Verify user can view their existing insurance details or browse available plans.	User has a registered account.	1. User opens the mobile service provider app.	N/A	- If user has existing insurance : User's insurance details are displayed.	Pass
				2. User navigates to the "Insurance " section.			
TC_INS_2_1	Confirm Insurance Plans (US_INS_02)	Verify user can confirm enrollment in a chosen insurance plan.	User has a registered account and selects a plan they are eligible for.	1. User opens the mobile service provider app.	Valid user information meeting plan eligibility criteria		Pass
				2. User navigates to the "Insurance " section			
				3. User selects the option to browse available plans (if no existing plan).			
				4. User selects a			

				specific insurance plan			
				5. User confirms enrollment by following the plan's enrollment process.			
TC_MOB_1_3	Initiate Mobile Recharge (US_MOB_01)	Verify successful mobile recharge functionality.	User has a registered account with sufficient balance.	1. User opens the mobile service provider app.	Valid phone number, Valid recharge amount (e.g., $10)	Error message: System overload, please try again later.	Fail
				2. User navigates to the "Mobile Recharge" section.			
				3. User enters their phone number.			
				4. User selects a valid recharge amount.			
				5. User confirms the recharge.			
TC_TIC_2_3	Book Tickets (US_TIC_02)	Verify successful movie ticket booking with valid payment.	User has a registered account with a valid payment method (e.g., credit card) and sufficient balance.	1. User opens the mobile service provider app.	Valid payment method, Sufficient balance	Booking confirmation message, but the seats are actually not reserved due to a system glitch.	Fail
				(Follow steps 2-5 from			

				TC_TIC_2_1)			
				6. User confirms the booking and completes the payment.			