**Testing Report**

**1. Introduction**

The automotive repair industry is undergoing a digital transformation, with modern systems offering customers more efficient services, such as online booking, tracking of vehicle repair status, and instant quotes. For any such system to be effective, thorough testing is crucial to ensure that it performs correctly, securely, and efficiently under various conditions. This report provides a comprehensive analysis of the testing process of an automotive repair system, which includes validating core functionalities, assessing performance under load, ensuring security, and enhancing user experience.

The aim of this report is to document the results of functional, performance, and security testing, and provide recommendations for system improvements based on the findings. The system in question handles various service processes, including customer registration, service order management, pricing generation, and post-service satisfaction feedback.

**2. Testing Goals**

The primary goals of this testing are:

1. Functionality Validation: To ensure that all system modules, such as order processing, quotation generation, customer feedback, and payment handling, work as intended.

2. Performance Testing: To assess the system's behavior under different loads, ensuring that it can handle high user volumes without compromising performance.

3. Security Testing: To verify that customer data is adequately protected from unauthorized access and data breaches.

4. User Experience Testing: To evaluate how intuitive and user-friendly the interface is and identify areas for improvement.

5. Regression and Integration Testing: To ensure that newly introduced features do not disrupt existing functionalities and that the new modules integrate smoothly with the current system.

**3. Testing Strategy**

3.1 Functional Testing Strategy

The functional testing strategy focuses on ensuring that the core functionalities of the automotive repair system are working as expected. This includes verifying that each individual service module, from the customer’s initial request to the final satisfaction survey, operates smoothly. The key functional areas covered by the test cases are:

• Refund Request Submission: Customers submit refund requests for completed service orders.

• Membership Status Verification: The system checks whether the customer is a member and offers appropriate benefits (e.g., discounts for members).

• Towing Service Quote Generation: The system automatically generates quotes for towing services when customers request them.

• Deposit Collection: The system calculates and processes the deposit required for a specific repair service based on the estimated cost.

• Vehicle Diagnosis and Repair: The diagnostic and repair process, including recording the diagnosis results, generating a repair plan, and creating a service quote.

• Repair Completion and Customer Acceptance: Ensures that after the vehicle repair, the customer can review the work, provide feedback, and accept or request further actions.

• Customer Satisfaction Survey: A system-generated survey that captures customer feedback post-service.

• Form Validation Error Handling: Tests how the system handles user errors in form submissions, such as incomplete or invalid data.

3.2 Automation Testing Strategy

Automation testing plays a vital role in streamlining repetitive tests and verifying the functionality of the system quickly and efficiently. The automation strategy includes simulating real-world user interactions with the system to ensure that it behaves as expected. Key areas of automated testing include:

• Customer Reception Simulation: Verifies that the system correctly captures customer and vehicle information and offers membership options where applicable.

• Towing Service Request Automation: Simulates a towing service request, generating a quote and managing deposit payment.

• Vehicle Repair Process Automation: Simulates a vehicle arriving at the repair shop, followed by automatic diagnosis, repair plan generation, and quote creation.

• Customer Acceptance and Feedback Automation: Simulates customer vehicle inspection, automatically capturing feedback through satisfaction surveys, and processing refund requests.

By automating these test cases, we can ensure consistency in testing and reduce the time needed to perform manual tests.

3.3User Experience Testing Strategy

User experience (UX) is a crucial aspect of the automotive repair system because it directly impacts customer satisfaction and system usability. The UX testing strategy includes:

• Interface Usability Testing: Evaluating how easy and intuitive it is for users to navigate the system and access its features.

• Error Handling Usability: Testing the clarity and helpfulness of error messages displayed when users submit incomplete or invalid information.4. Testing Implementation Details

**4.Functional Test Cases Implementation**

Functional testing is a critical part of ensuring that all business processes and system features work as intended. It covers the functionality of individual system modules as well as their interactions with one another. Below are the expanded test cases implemented for the automotive service system:

1. Refund Request Submission

* Precondition:
  + The customer must have a completed service order in the system. This ensures that a valid transaction exists, which can be associated with the refund request.
* Operation:
  + The customer submits a refund request form via the system, entering their name, phone number, order ID, and the reason for requesting a refund. This request is logged and sent to the relevant department for processing.
* Expected Outcome:
  + The system should:
    1. Capture and store the refund request data accurately.
    2. Trigger the refund process, which could include a financial transaction or notification to the customer service team.
    3. Notify the appropriate department or workflow to process the refund request in the background.
* Automation Script:
  + The script simulates the customer filling out the refund form. It automatically inputs the necessary fields, such as name, phone number, order ID, and reason for the refund.
  + The script ensures that the system stores the request correctly in the database and triggers the refund workflow (such as notifying customer service or initiating a payment process).
  + The script verifies that all the request data is logged, ensuring data consistency and traceability.

2. Membership Status Check

* Precondition:
  + The customer is either a member or a non-member in the system. The membership status is a key factor in determining the pricing or promotional benefits available to the customer.
* Operation:
  + The system checks the customer’s membership status based on their stored details. The system may access a customer database or membership registry to identify whether the customer is a member.
* Expected Outcome:
  + If the customer is a member, the system provides a 20% discount on services.
  + If the customer is not a member, the system prompts the customer with an option to enroll in the membership program, offering the benefits of membership (e.g., discounts, priority services).
* Automation Script:
  + The script tests two scenarios: one where the customer is a member and another where the customer is not a member.
  + For non-members, the script ensures that the system presents the membership enrollment option and clearly explains the benefits.
  + For members, the script verifies that the discount is automatically applied to the service quote or invoice.

3. Towing Service Quote Generation

* Precondition:
  + The customer’s vehicle has broken down and requires towing services. The system should be able to handle dynamic pricing based on the type of towing required (e.g., distance, vehicle size).
* Operation:
  + The customer requests towing services through the system interface, providing necessary details such as vehicle type, location, and distance to be covered.
* Expected Outcome:
  + The system generates an accurate towing quote based on the customer’s location, the type of service required, and the vehicle’s specifications (e.g., vehicle size, weight).
  + The system displays the quote to the customer, including all relevant pricing details (e.g., base price, additional charges).
* Automation Script:
  + The script simulates a customer’s request for towing services by inputting data like vehicle type, location, and distance towed.
  + The system should return a towing quote that matches predefined pricing rules based on input parameters.
  + The script checks that the final quote aligns with the system's pricing model and is presented clearly to the customer.

4. Deposit Collection and Vehicle Diagnosis

* Precondition:
  + The vehicle has undergone a diagnostic process, and the diagnosis results have been logged in the system. Based on the diagnostic report, the customer may need to pay a deposit for repair services.
* Operation:
  + The customer requests repair services based on the diagnostic results. The system calculates the required deposit, which may vary depending on the severity of the vehicle’s issues.
* Expected Outcome:
  + The system calculates the deposit amount based on the nature of the repair (e.g., major repairs or minor repairs).
  + The system generates payment instructions for the customer, ensuring that the correct amount is displayed, and prompts the customer to make the deposit.
* Automation Script:
  + The script simulates the process of vehicle diagnostics by inputting a variety of diagnostic reports, from minor to major issues.
  + For each diagnosis, the system should generate an accurate deposit requirement.
  + The script verifies that the correct amount is displayed, ensuring that the system follows the defined pricing algorithm.

5. Repair Completion and Customer Acceptance

* Precondition:
  + The vehicle repair has been completed, and the customer is ready to inspect the vehicle.
* Operation:
  + The customer inspects the repaired vehicle. If satisfied with the work, they accept the repair and retrieve the vehicle. If not satisfied, they request further repairs or a refund.
* Expected Outcome:
  + If the customer is satisfied, the system allows the customer to accept the vehicle and finalize the service transaction.
  + If the customer is not satisfied, the system triggers follow-up actions, such as initiating further repairs or offering a refund process.
* Automation Script:
  + The script simulates both customer satisfaction and dissatisfaction scenarios. In the case of satisfaction, the system should mark the vehicle as "accepted" and finalize the transaction.
  + If the customer is dissatisfied, the script verifies that the system prompts the appropriate follow-up actions, such as requesting additional repairs or processing a refund.

6. Customer Satisfaction Survey

* Precondition:
  + The service has been completed, and the customer is ready to provide feedback regarding their experience.
* Operation:
  + After the service is completed, the system prompts the customer to fill out a satisfaction survey regarding the quality of the service provided.
* Expected Outcome:
  + The system records the customer’s responses and stores them for future analysis.
  + The survey data is utilized to improve the service and address any issues raised by the customer.
* Automation Script:
  + The script simulates the customer filling out the satisfaction survey after service completion.
  + The system should correctly log all responses in the backend, and negative feedback should be flagged for review by the customer service team.
  + The system should store feedback securely and make it accessible for future improvements in services.

7. Form Validation Error Handling

* Precondition:
  + The customer submits a form with missing or incorrect information. This could include missing required fields or providing incorrect data formats (e.g., an invalid email address or phone number).
* Operation:
  + The customer attempts to submit a form with missing or invalid information. The system should intercept the submission and provide clear, actionable feedback.
* Expected Outcome:
  + The system should:
    1. Prevent form submission until all required fields are filled out correctly.
    2. Display an error message that clearly explains which fields are missing or incorrect.
    3. Ensure the form cannot be submitted until the errors are corrected.
* Automation Script:
  + The script tests various invalid form submissions, such as missing fields (e.g., no name or email) and incorrect formats (e.g., invalid phone numbers or emails).
  + It verifies that the system provides clear error messages and prevents submission until the issues are resolved.
  + The script also ensures that the system’s error messages are meaningful and guide the user to make the necessary corrections.

**5. Test result analysis**

1. Functional test results

Refund request form submission: The system can correctly receive and process refund requests and start the refund process. However, in the form field verification, the system failed to prompt customers to fill in the required items in time, which needs to be optimized.

Member status check: The system can provide correct discount information according to the customer's membership status, and performs well.

Towing service quotation generation: The quotation generation is accurate, and the system can feedback the quotation information in time.

Repair and acceptance process: After the vehicle repair is completed, the customer acceptance process is smooth, but if the customer is not satisfied, the subsequent processing process response is slightly slow, which needs to be optimized.

Customer satisfaction survey: The system can successfully record customer feedback, and subsequent service improvements have a basis.

2. Performance test results

Concurrency performance: When simulating 100 concurrent users, the system response time increases significantly. When more than 200 concurrent users, the response time reaches an unacceptable level, and the system needs to be optimized (such as introducing load balancing, etc.).

Data processing capability: When processing large amounts of data, the query response time is slow, especially when querying historical orders, the system performance needs to be improved.

3. User experience test results

Interface design: Users generally believe that the system interface is simple and easy to use, but some advanced functions are not intuitive enough. It is recommended to add help documents or guidance.

Error prompts: The system can correctly prompt users of operation errors and provide reasonable solutions.

4. System enhancement suggestions

Function improvement suggestions:

Enhance the refund form verification logic to ensure that customers fill in all required items before submission.

Optimize the process after customer acceptance and shorten the response time when dissatisfied.

Performance optimization suggestions:

Load balancing optimization for concurrent requests to improve the system's response speed under high concurrency.

Optimize query algorithms, especially data processing speed under large data volumes.

Security improvement suggestions:

Strengthen security protection measures in the payment process to avoid data leakage risks.

Enhance user permission management and refine permission control for different roles.

User experience optimization suggestions:

Improve the interactive design of advanced functions to make them more intuitive and easy to understand.

Optimize error prompt information to ensure that users can quickly understand and solve problems.

**7. Conclusion**

This test covers the main functions and processes of the automobile maintenance system. Through the test, we found the improvement space and performance bottlenecks of some functions. Based on the analysis of the test results, we put forward system optimization suggestions, hoping to further improve the stability, performance and user experience of the system in the subsequent development process.