Car Diagnostic Expert System Test Report

The tests will be performed to investigate the performance, robustness, and scalability of the expert system for a wide range of load conditions and inputs. Tests are divided into three main categories: Load Tests, System Behavior Tests, and Resilience Tests.

Burden Tests

- Test 1: High Fact Load

Objective: To evaluate the performance of the system when multiple facts are injected into it simultaneously.

Rationale: To ensure that the system can handle large volumes of facts without affecting its performance.

Steps:

Create 1000 different symptoms related to car problems.

Inject the facts into the inference engine.

Measure the execution time.

Expected Results:

The system should process the facts in less than 10 seconds.

No errors or crashes should occur during the processing.

The system should exhibit overall stability in handling the load.

- Test 2: High Fact Load with Concurrency Diagnostics

Objective: The performance of the system when running multiple diagnostics concurrently.

Rationale: To ensure the system can support multiple queries running at the same time without degrading performance.

Step:

5000 Symptoms created

100 parallel diagnostic requests started

Response times for every diagnosis measured

Expected Results:

Response time should remain consistent and not exceed 5 seconds per diagnosis.

The system should respond to all queries without failures.

- Test 3: Incremental Data Scalability Test

Objective: The system shall be benchmarked on the reaction of increasing the number of facts incrementally.

Rationale: The objective is to test the scalability of the system.

Steps:

Start with 100 facts and increase up to 5000 facts.

Measure the execution time of each set of increased facts.

Expected Results:

Execution time shall increase predictably and shall not take more than 20 seconds with 5000 facts.

The system should keep running smoothly, without any errors, up until a maximum load is reached.

System Performance Tests

- Test 1: Complex Combinations of Facts and Rules

Objective: The purpose of this test is to gauge the system's efficiency in handling complex combinations of facts.

Rationale: To confirm that the inference engine accurately solves combinations of symptoms and fires the related rules.

Steps:

Prepare the most complicated sets of facts, such as "car won't start" diagnosis with multiple potential causes.

Run the inference engine.

Test if the correct rules fire up and the diagnoses are concluded.

Expected Outputs:

The engine should consider each combination and deliver good diagnostics.

Execution time should be minimal (under 5 seconds).

- Test 2: Partial Input Diagnostics

Objective: To test the system’s behavior with incomplete or partial fact input.

Rationale: So that one can be certain that the system can operate when the user does not key in all symptoms.

Procedure:

Only inject partial symptoms, like "car will not start" but do not inject any details as to noises or otherwise. Run the diagnosis. See if the system can catch the partial input and provide suggestions.

Expected Results The system provides a list of probable diagnostics to further help it if needed. The system does not fail or give any wrong results. Test 3: Rules with Priority Evaluation

Objective: To test the priority handling of the system, which rule will fire first.

Rationale: One wants to check that when there are overlapping rules, only the most critical ones will fire first.

Steps:

Create several rules of different priorities such as overheat versus no fuel diagnosis.

Start up an engine cycle, then check which of the two rules is the first to fire.

Expected Results:

The system must be able to handle the hierarchy of the rules correctly by firing the high-priority rules first.

Robustness Testing

- Test 1: Incomplete or Wrong Input

Objective: Understand how erroneous or incomplete input is handled by the system.

Rationale: The system should, therefore, be tested for strength and stability so that it can support incorrect input without failure.

Steps:

Insert bad facts-for instance, introduce an irrelevant symptom or one formatted incorrectly.

Run the diagnostic.

Test error handling.

Expected Results:

This must respond with a friendly error message and fail completely.

It should keep functioning normally after taking care of the error.

- Test 2: System Crash Resistance Objective

This test will check the system's resilience against sudden crashes or failures. Rationale: To ensure that the system recovers itself without losing any data or malfunctioning in operation. Steps: Simulate a system crash - forcing it to shut down the database service or the inference engine. Restart the Service. Check that the system resumes its previous state and functions correctly. Expected Results: The system should automatically recover without losing any important data.

The system recovery should not have any errors.

- Test 3: High Run Frequency Test

Objective: To test the performance of the system when the frequency of consecutive executions is very high.

User testing roadmap

For the tests to be performed by the end users of the application, the following tasks will be performed to correctly collect the data and to be able to take adequate statistical samples:

1. The users will be given one or more test cases to perform on the expert system.
2. The users will use the application as they would normally use it, and the researcher will note down the symptoms of discomfort that they notice as well as the results of the system.
3. Test users will be surveyed about their overall experience with the application, how satisfied they were with the response, what they would improve, etc.
4. As a method of verification, a final interview will be conducted to better cement the information provided.

Process deliverables: survey, data collected, discomfort collected, interviews.

A group of 2 to 3 people that meet the characteristics of being a target audience will be used as test cases.

On the other hand, a similar test will be carried out, but with the expert where the issue of experience with the application will be obviated and the focus will be on visualizing and verifying if it is functionally valid.

Deliverables of the process: interview