Test Case Specification Document for Microwave Radar Analysis Model Interaction

July 25, 2025

Overview

This document outlines the test cases for validating the detailed design of the Microwave Radar Analysis model interaction system. The system includes an Analysis Manager, frontend applications, various models (Absorption Loss Model, ECM Model, Radar Model), and a Data Logger, with data managed via PostgreSQL. The test cases ensure functionality, performance, and reliability as per the design document.

Test Scope

- **In-Scope**: All use cases, API interactions, data retrieval/storage, model computations, and non-functional requirements.
- **Out-of-Scope**: External system integrations beyond API endpoints, hardware-specific testing.

Test Environment

- **Hardware**: Standard server with 16GB RAM, 8-core CPU.
- **Software**:
  - Angular frontend, Java 21 (Spring Boot backend), PostgreSQL 16.
  - Curl for RESTful API testing .
  - Postgresql CLI or PgAdmin for DB validation
- **Network**: Localhost-based environment , fully offline capable; no internet dependency during execution

Test Cases

1. Valid Login Credentials

- **Test ID**: TC001
- **Description**: Ensure the user can log in with valid credentials.
- **Preconditions:** Frontend and backend are running. Database has at least one user (testuser, password123).

- **Test Steps:**
    1. Open the login page in the browser.
    2. Enter valid username and password.
    3. Click the "Login" button.
    4. Observe the response and redirection behavior.
- **Expected Result**: User is redirected to the main dashboard view.
- **Actual Result**: User successfully redirected to /dashboard after login.
- **Pass/Fail**: Pass
- **Priority:** High

2. Invalid Credentials

- **Test ID**: TC002
- **Description**: Ensure incorrect username or password leads to login error.
- **Preconditions:** Frontend and backend running.
- **Test Steps:**
    1. Open the login page in the browser.
    2. Enter the wrong username or wrong password.
    3. Click the "Login" button.
    4. Observe the response and redirection behavior.
- **Expected Result**: Login fails. A user-friendly error message is shown like "Invalid credentials". No redirect happens.
- **Actual Result**: Message displayed: "Invalid credentials.".
- **Pass/Fail**: Pass
- **Priority:** High

3. Empty Fields During Authentication

- **Test ID**: TC003
- **Description**: Ensure login fails when username/password fields are empty.
- **Preconditions:** Frontend input validation in place.
- **Test Steps:**
    1. Open the login page in the browser.

2. Frontend input validation in place.
3. Click the "Login" button.
4. Observe the response and redirection behavior.

- **Expected Result**: No request sent to backend. UI shows “Please select a username and enter password.” message.
- **Actual Result**: Message displayed: "Please select a username and enter password.".
- **Pass/Fail**: Pass
- **Priority:** Medium

4. Backend Down During Authentication

- **Test ID**: TC004
- **Description**: Ensure frontend gracefully handles if login API is unreachable.
- **Preconditions:** Frontend running, Backend turned off or unreachable.
- **Test Steps:**
  1. Open the login page in the browser.
  2. Enter valid credentials and click login.
  3. Click the "Login" button.
  4. Observe the response and redirection behavior.
- **Expected Result**: Frontend catches error and shows: “Login service unavailable. Try again later.”
- **Actual Result**: Message displayed: “Login service unavailable. Try again later.”
- **Pass/Fail**: Pass
- **Priority:** Medium

5. Analysis Manager Not Running

- **Test ID**: TC005
- **Description**: Ensure that the frontend properly handles the case where the Analysis Manager microservice is not available.
- **Preconditions:** Frontend is running. Analysis Manager is not running.
- **Test Steps:**
  1. Open the frontend dashboard.
  2. Configure any scenario with valid inputs.
  3. Submit the request.

4. Observe the log section of the dashboard.

- **Expected Result**: No request sent to backend. UI shows “Unable to connect to the analysis model” message.
- **Actual Result**: Message displayed: "Unable to connect to the analysis model".
- **Pass/Fail**: Pass
- **Priority:** High

6. Submit Analysis Request to Analysis Manager

- **Test ID**: TC006
- **Description**: Verify that the frontend successfully sends an analysis request to the Analysis Manager service and receives a valid request ID.
- **Preconditions:** Frontend, Analysis Manager and Absorption Loss Model are running
- **Test Steps:**
    1. Open the frontend dashboard in a browser.
    2. Input parameters associated with scenario ID SC001.
    3. Submit the request.
    4. Open the browser's developer console or network tab.
    5. Inspect the HTTP response and look for a returned request ID.
- **Expected Result**:
    1. HTTP Status: 200 OK;
    2. JSON Response: { "requestId": <numeric_id> }
- **Actual Result**:
    1. HTTP Status: 200 OK;
    2. JSON Response: { "requestId": 39 }
- **Pass/Fail**: Pass
- **Priority**: High

7. Model Microservice Not Running

- **Test ID**: TC007
- **Description**: Ensure that the Analysis Manager detects and reports the absence of the required model microservice (e.g., Absorption Loss Model) when a request is made.
- **Preconditions:** Frontend and Analysis Manager are running while target model service

(e.g., AbsorptionLossService) is not running.

- **Test Steps:**
    1. Open the frontend dashboard.
    2. Input parameters associated with scenario ID SC001.
    3. Submit the request.
    4. Observe the log section of the dashboard.
- **Expected Result**: No request sent to backend. UI shows “Unable to connect to the absorption-loss model” message.
- **Actual Result**: Message displayed: "Unable to connect to the absorption-loss model".
- **Pass/Fail**: Pass
- **Priority:** High

8. Validate Data Retrieval from Absorption Loss Model

- **Test ID**: TC008
- **Description**: Ensure that the Analysis Manager correctly delegates the analysis request to the Absorption Loss Model and retrieves the expected dataset.
- **Preconditions:** Frontend, Analysis Manage and Absorption Loss Model are running
- **Test Steps:**
    1. Open the frontend dashboard in a browser.
    2. Input parameters associated with scenario ID SC001.
    3. Submit the request.
    4. Observe the Output section on the dashboard.
    5. (Optional) Open the browser’s developer console or network tab and observe the JSON response from the system.
- **Expected Result**: The output section displays a table with the following values:
  ldash <numeric_value>
- **Actual Result**: A table containing:
  ldash 0.1451
- **Pass/Fail**: Pass
- **Priority**: High

9. Validate Data Retrieval from ECM Model

- **Test ID**: TC009

- **Description**: Ensure that the Analysis Manager correctly delegates the analysis request to the ECM Model and receives the expected output dataset.
- **Preconditions**: Frontend, Analysis Manage and ECM Model are running
- **Test Steps:**
  1. Open the frontend dashboard in a browser.
  2. Input parameters associated with scenario ID SC002.
  3. Submit the request.
  4. Observe the Output section on the dashboard.
  5. (Optional) Open the browser's developer console or network tab and observe the JSON response from the system.
- **Expected Result**: The output section displays a table with the following value:
  jra <numeric_value>
- **Actual Result**: A table containing:
  jra -85.6758
- **Pass/Fail**: Pass
- **Priority**: High

10. Validate Data Retrieval from Radar Model

- **Test ID**: TC0010
- **Description**: Ensure that the Analysis Manager correctly delegates the analysis request to the Radar Model and receives the expected output dataset.
- **Preconditions:** Frontend, Analysis Manage and Radar Model are running
- **Test Steps:**
  1. Open the frontend dashboard in a browser.
  2. Input parameters associated with scenario ID SC003.
  3. Submit the request.
  4. Observe the Output section on the dashboard.
  5. (Optional) Open the browser's developer console or network tab and observe the JSON response from the system.
- **Expected Result**: The output section displays a table with the following values:
  pd <numeric_value>
  sreceiveRadar <numeric_value>
  siradar <numeric_value>
- **Actual Result**: A table containing:
  pd 0.0001
  sreceiveRadar -100.0000
  siradar -100.0000

- **Pass/Fail**: Pass
- **Priority**: High

11. View Single Output on Cesium

- **Test ID**: TC0011
- **Description**: Ensure that running a model displays the expected output (correctly on the Cesium map.
- **Preconditions**: All components are up and running
- **Test Steps:**
    1. Open the frontend dashboard in a browser.
    2. Input parameters associated with scenario ID SC001.
    3. Submit the request.
    4. Observe map rendering (3D domes, lines, or markers) based on response data.
- **Expected Result**:
    1. Cesium map renders successfully.
    2. Visual output matches model response (e.g., correct domes for target/emitter positions).
    3. No console or rendering errors.
- **Actual Result**:
    1. Cesium map rendered successfully.
    2. Visual output matched model response
    3. No console or rendering errors.
- **Pass/Fail**: Pass
- **Priority**: High

12. Verify Historical Output Persistence and Manual SQL Retrieval

- **Test ID**: TC0012
- **Description**: Ensure that after a successful computation, the output is correctly persisted in the PostgreSQL database and can be manually retrieved using a SQL join query between input parameters and results.
- **Preconditions**: A successful Absorption Loss computation has already been executed and the PostgreSQL database is running and accessible.
- Test Steps:
    1. Connect to the database

2. Execute:

   SELECT ap.*, ao.l_dash, wc.weather_type, wc.lat_min, wc.lat_max, wc.long_min, wc.long_max, wc.height_min, wc.height_max, wc.whole_globe FROM atmospheric_parameters ap JOIN atmospheric_outputs ao ON ap.id = ao.parameter_id LEFT JOIN weather_condition wc ON wc.parameters_id = ap.id WHERE ap.id = <request_id>;

- **Expected Result**:
  1. A record is returned with:
     - All input fields
     - A valid computed value for l_dash
  2. No SQL or join errors; result is correctly structured.
- **Actual Result**:

  id | freqj | freq_op | freqr | heightj | heightr | heightt | latj | latr | latt | longj | longr | longt | rfr | sfr | vis_fog | l_dash | weather_type | lat_min | lat_max | long_min | long_max | height_min | height_max | whole_globe

  ----+------------+------------+------------+---------+---------+---------+------+------+------+-------+-------+-------+-----+-----+---------+--------------------+--------------+---------+---------+----------+----------+------------+------------+-------------

  1 | 1000000000 | 1200000000 | 1200000000 | 2000 | 500 | 2000 | 34.1 | 34.2 | 34.1 | 69.1 | 69.2 | 69.1 | 10 | 5 | 3 | 0.14511131538810446 | clear | | | | | | | t

  (1 row)
- **Pass/Fail**: Pass
- **Priority**: High

13. Download Output

- **Test ID**: TC0013
- **Description**: Validate that the user can successfully download the absorption loss output file (CSV and pdf) generated after computation.
- **Preconditions**: All components are up and running:
- **Test Steps:**
  1. Open the frontend dashboard in a browser.
  2. Input parameters associated with scenario ID SC001.
  3. Submit the request.
  4. Click “Export PDF” and “Export CSV”

5. Open the downloaded files and observe the content

- **Expected Result**:
  1. File download completes successfully without corruption.
  2. File contains all necessary fields
  3. Data of all runs in the session displayed properly
- **Actual Result**:
  1. Download completed successfully without corruption.
  2. The file included all expected fields with correct headers and values.
  3. Data of all runs in the session was displayed properly.
- **Pass/Fail**: Pass
- **Priority**: Low

Non-Functional Test Cases

1. Offline Operation Verification

- **Test ID**: TC007=14
- **Description**: Ensure the entire system (frontend, backend, database, models) operates correctly in a fully offline environment, without initiating any external network connections during execution.
- **Preconditions**: All components are up and running:
  1. Frontend
  2. Backend microservices (e.g., Absorption Loss Service, ECM Model, Radar Model)
  3. PostgreSQL database
  4. Required models are locally deployed and available
- **Test Steps:**
  1. Disconnect the test environment from the internet (or simulate offline mode).
  2. Start all backend microservices, frontend server, and database.
  3. On the command line, execute: sudo netstat -tunp | grep ESTABLISHED
  4. Confirm there are no active connections to external IP addresses.
  5. Open the frontend in a browser and load the dashboard.
  6. Open the browser's Developer Tools → Network tab.
  7. Reload the frontend application and observe:
     - No requests are made to external URLs (e.g., CDNs, analytics, fonts).
     - All JS/CSS/fonts load from local or loopback (127.0.0.1).
  8. Submit a sample request (e.g., absorption model input).
  9. Confirm model response and frontend display works normally.
- **Expected Result**:

4. All HTTP/HTTPS/TCP connections remain local (127.0.0.1 or localhost).
5. No failed requests to remote hosts (e.g., no 404/503s to external domains).
6. The application continues to function completely in offline mode, including model execution and UI rendering.

- **Actual Result**:
  4. Netstat confirms all connections are local (127.0.0.1).
  5. No external requests logged in the browser.
  6. The system operated fully without failure or degraded functionality in offline mode.
- **Pass/Fail**: Pass
- **Priority**: High

2. Verify Performance (Data Retrieval)

- **Test ID**: TC0015
- **Description**: Validate that historical data retrieval from the PostgreSQL database is performant and completes in under 1 second.
- **Preconditions:** A successful Absorption Loss computation has already been executed and the PostgreSQL database is running and accessible.
- **Test Steps:**
  1. Connect to the database
  2. Execute: \timing
  3. Execute:

     SELECT ap.*, ao.l_dash FROM absorption loss_parameters ap
     JOIN absorption loss_outputs ao ON ap.id = ao.parameter_id
     WHERE ap.id =request ID ;
  4. Record response time.
  5. Repeat the test multiple times to confirm consistency.
- **Expected Result**: Query completes in under 1 second.
- **Actual Result**: Query executed in: 0.471 ms ~ 0.725 ms across 100 runs.
- **Pass/Fail**: Pass
- **Priority**: High

3. Verify Performance (Model Computation)

- **Test ID**: TC0016
- **Description**: Ensure model computation completes in under 1 second via Analysis Manager

- **Preconditions**: All components are up and running:
    1. Frontend
    2. Backend microservices (e.g., Absorption Loss Service, ECM Model, Radar Model)
    3. PostgreSQL database
- **Test Steps:**
    1. Send a POST request to the Analysis Manager endpoint:
    2. POST http://localhost:8084/api/analysis/request
    3. Include a valid JSON payload in the request body (e.g., sample radar/ECM parameters).
    4. Record the total response time using curl.
    5. Repeat the test multiple times to confirm consistency.
- **Expected Result**: Total response time for the model computation must be less than 1 second.
- **Actual Result**: Average time observed: 0.347456 s ~ 0.47342 s across 100 runs.
- **Pass/Fail**: Pass
- **Priority**: High

4. Verify API Success Rate Reliability

- **Test ID**: TC017
- **Description**: Verify that the Analysis Manager's /api/analysis/request endpoint maintains at least a 99% success rate when handling a burst of 100 consecutive requests per model.
- **Preconditions**: All components are up and running:
    1. Frontend
    2. Backend microservices (e.g., Absorption Loss Service, ECM Model, Radar Model)
    3. PostgreSQL database
- **Test Steps:**
    1. For each model (Radar, ECM, Absorption), send 100 consecutive POST requests to: http://localhost:8084/api/analysis/request
    2. Use a valid JSON payload in each request.
    3. Log HTTP status codes for all 100 responses.
    4. Compute success rate using the formula:
    5. Success Rate (%) = (Number of HTTP 200 responses / Total Requests) × 100
- **Expected Result**: Success Rate: ≥ 99% for each model

- **Actual Result**:
    1. Radar: 100% (100/100)
    2. ECM: 100% (100/100)
    3. Absorption: 100% (100/100)
- **Pass/Fail**: Pass
- **Priority**: High

Test Execution Schedule

- **Start Date**: July 17, 2025
- **End Date**: July 24, 2025
- **Time**: 09:00 AM - 06:00 PM IST daily, starting from 08:57 PM IST on July 16, 2025, after initial setup.

Pass/Fail Criteria

- All high-priority test cases must pass.
- ≥ 90% of medium-priority test cases must pass.
- No critical failures in non-functional tests.

Assumptions

- Test environment mirrors production.
- All dependencies are pre-installed

Glossary

- **API**: Application Programming Interface.
- **SC001:** Scenario ID used to identify a specific test scenario in the Analysis Manager system. It corresponds to a predefined absorption loss model dataset and associated simulation inputs.

  {"rfr": 10, "sfr": 5, "visFog": 3, "freqR": 1200000000, "freqJ": 1000000000, "freqOp": 1200000000, "latR": 34.2, "longR": 69.2, "heightR": 500, "latT": 34.1, "longT": 69.1, "heightT": 2000, "latJ": 34.1, "longJ": 69.1, "heightJ": 2000, "weatherConditions": [{ "weatherType": "clear", "wholeGlobe": true}]}
- **C002:** Scenario ID used to identify a specific test scenario in the Analysis Manager

system. It corresponds to a predefined ECM model dataset and associated simulation inputs.

{"pJ": 100, "gJ": 20, "lJ": 10, "azimuthJ": 35, "elevationJ": 24, "beamwidthAzJ": 25, "beamwidthElJ": 25, "bJ": 500000000, "freqJ": 1, "latJ": 34.1, "longJ": 69.1, "heightJ": 2000, "jammerType": "Responsive", "jammerSubType": "Self - protection", "antennaType": "Gain Pattern", "antennaSubType": "Omnidirectional", "latR": 34.2, "longR": 69.2, "heightR": 500, "bR": 200000000, "freqR": 1.2}

- **C003:** Scenario ID used to identify a specific test scenario in the Analysis Manager system. It corresponds to a predefined radar model dataset and associated simulation inputs.

{"gr": 8.3, "lo": 3.7, "polarizationR": "Linear Horizontal", "polarizationJ": "Linear 45", "antennaType": "Gain Pattern", "antennaSubType": "Omnidirectional", "freqR_radar": 12400000000, "pr_radar": 15200000, "br_radar": 56700000, "fr_radar": 6.2, "k_radar": 1.38e-23, "t_radar": 290, "BMWEl_R_radar": -32, "BMWAz_R_radar": 67, "elLimit_radar": 65, "latT_radar": 22.4, "longT_radar": 88.3, "heightT_radar": 150, "pfa_radar": 0.0000012}