**1. Objective**

This document outlines the test plan for the **Temperature Converter** application. The objective is to ensure accurate and consistent conversion between Celsius and Fahrenheit for all supported input values. The target audience includes **developers, testers, and end-users requiring accurate temperature conversion in software systems**.

**2. Scope**

**Features to be tested**:

* Conversion from Celsius to Fahrenheit
* Conversion from Fahrenheit to Celsius
* Accuracy of calculations
* Correct structure of SOAP response
* Error handling for invalid input

**Types of testing**:

* Manual Testing
* Automated API Testing
* Performance Testing
* Usability Testing (for any UI component using this API)

**Environments**:

* Browsers (for UI integration): Chrome, Firefox, Edge
* Operating Systems: Windows, macOS, Linux
* Device Types: Desktop, Tablet, Mobile (if integrated with UI)

**Evaluation criteria**:

* Correctness of output
* Response time under 2 seconds
* Zero critical bugs
* High unit and integration test coverage

**Team roles and responsibilities**:

* Test Lead: Overall test planning and coordination
* Testers: Create and execute test cases
* Developers: Implement and fix logic
* DevOps: API environment management
* Stakeholders: Review test deliverables

**3. Inclusions**

**Introduction**:  
Covers validation of SOAP-based temperature conversion API using predefined input-output pairs and boundary cases.

**Test Objectives**:

* Ensure conversion accuracy (formula-based validation)
* Handle invalid/missing inputs gracefully
* Validate XML response structure

**4. Exclusions**

* JSON or REST-based temperature services
* Localization or multi-language support
* Unit conversions other than temperature

**5. Test Environments**

**Operating Systems**:

* Windows 10, macOS Ventura, Ubuntu 22.04

**Browsers (UI context)**:

* Chrome, Firefox, Edge

**Devices**:

* Laptops, Tablets, Smartphones

**Network**:

* Wi-Fi, Wired

**Hardware/Software**:

* Min 4GB RAM, SOAP testing tool (e.g., SoapUI, Postman with SOAP extension)

**Security Protocols**:

* None for open access (otherwise Basic Auth if added later)

**Access Permissions**:

* Testers: Read/Write to test system
* Developers: Full environment access
* Stakeholders: Report access only

**6. Defect Reporting Procedure**

**Defect Criteria**:

* Incorrect conversions
* Invalid responses or crash
* Response delays or timeouts

**Reporting Steps**:

* Capture Request and Response
* Log issue in JIRA
* Attach screenshots and logs
* Mention severity and reproducibility

**Triage Process**:

* Critical: Wrong conversion output
* High: No response or invalid XML
* Medium: Slowness or UI formatting
* Low: Typos or minor UI glitches

**Tools**:

* JIRA, SoapUI, Excel (for reports)

**Metrics**:

* No. of defects, fix rate, open vs closed bugs

**7. Test Strategy**

**Step 1: Test Design**  
Techniques Used:

* **Equivalence Class Partitioning**: Valid & invalid values
* **Boundary Value Analysis**: 0°C, -273.15°C (absolute zero), 100°C
* **Decision Table**: Conversion logic under various inputs
* **Error Guessing**: Alphanumeric and null inputs
* **Use Case Testing**: Real-world temp entries like weather data

**Step 2: Test Execution**

* **Smoke Testing**: API availability and basic response
* **In-depth Testing**: Full validation using test cases
* **Cross Environment Testing**: Windows, Linux
* **Defect Logging**: JIRA + daily status updates

**Testing Types**:

* Smoke, Sanity, Regression, Retesting, Functional Testing, XML Validation

**Step 3: Best Practices**

* **Context Driven Testing**
* **Shift Left Testing**: Start API validation during development
* **End-to-End Scenarios**: UI to backend integration

**8. Test Schedule**

| **Task** | **Duration** | **Dates** |
| --- | --- | --- |
| Test Plan Preparation | 1 Day | Apr 24 |
| Test Case Design | 1 Day | Apr 25 |
| Test Execution | 2 Days | Apr 26 – Apr 27 |
| Bug Reporting/Review | Ongoing | Apr 26 – Apr 28 |
| Test Summary Report | 1 Day | Apr 29 |

**9. Test Deliverables**

* Test Plan Document
* Test Scenarios and Cases
* Execution Reports
* Defect Reports
* Final Test Summary Report

**10. Entry and Exit Criteria**

**Requirement Analysis**

* Entry: Approved API specs
* Exit: Complete understanding of functionality

**Test Execution**

* Entry: Signed-off test cases, stable API deployed
* Exit: All test cases executed, major bugs closed

**Test Closure**

* Entry: Final execution completed
* Exit: Summary report shared and reviewed

**11. Tools**

* SoapUI / Postman (SOAP support)
* JIRA
* Excel & Word
* Snipping Tool

**12. Risks and Mitigations**

| **Risk** | **Mitigation** |
| --- | --- |
| SOAP Service Unavailable | Use mock service or request fix from Dev |
| XML Structure Change | Add dynamic schema validation |
| Misinterpretation of Results | Use formula validation (C = (F-32)\*5/9) |

**13. Approvals**

**Documents for Client Approval**:

* Test Plan
* Test Scenarios and Cases
* Defect and Summary Reports