**TC01:** Verify Required Headers are Present

17 February 2025

10:40 AM

**Description:** Ensure that required headers (e.g., Content-Type, Authorization, Accept) are included in the request.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a request without the required headers.
2. Verify the response status code (should be 400 or 401 depending on the missing header).
3. Send a request with all required headers.
4. Verify the response status code is 200 (OK).

**Expected results:** Missing required headers should result in an error, and all required headers should be present for a successful response

=========================================================================================

**TC02:** Verify Correct Content-Type Header

17 February 2025

10:40 AM

**Description:** Ensure the Content-Type header is set correctly for different request payloads (e.g., application/json, application/xml)

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a request with a valid Content-Type (e.g., application/json).
2. Verify the response status code is 200.
3. Send a request with an invalid Content-Type (e.g., text/plain).
4. Verify the response status code is 415 (Unsupported Media Type).

**Expected results:** API should return a success code for valid Content-Type and an error code for invalid Content-Type

=========================================================================================

**TC03:** Verify Authorization Header for Authentication

17 February 2025

10:40 AM

**Description:** Test if the Authorization header is properly handled for authentication (e.g., Bearer <token>).

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a request without the Authorization header.
2. Verify the response status code is 401 (Unauthorized).
3. Send a request with an invalid token in the Authorization header.
4. Verify the response status code is 403 (Forbidden).
5. Send a request with a valid token in the Authorization header.
6. Verify the response status code is 200 (OK).

**Expected results:** The API should handle valid, invalid, and missing tokens correctly and return the appropriate status codes.

=========================================================================================

**TC04:** Verify Accept Header for Response Format

17 February 2025

10:40 AM

**Description:** Ensure that the "Accept" header is respected to return the correct response format (e.g., application/json, application/xml).

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a request with the Accept header set to application/json.
2. Verify that the response is in JSON format.
3. Send a request with the Accept header set to application/xml.
4. Verify that the response is in XML format.
5. Send a request with an unsupported Accept header (e.g., text/html).
6. Verify that the response status code is 406 (Not Acceptable).

**Expected results:** The response format must be the same as the Accept header specified format

=========================================================================================

**TC05:** Verify Cache-Control Header

17 February 2025

10:40 AM

**Description:** Ensure the Cache-Control header is correctly set for caching behavior.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a request to the API.
2. Verify the response contains the correct Cache-Control header (e.g., no-cache, private, public).
3. Send a follow-up request and check if the Cache-Control header is applied properly in the response.

**Expected results:** The Cache-Control header should match the expected caching rules.

=========================================================================================

**TC06:** Verify API Response Status Code

17 February 2025

10:40 AM

**Description:** This test ensures that the API responds status is correct for each API Method (GET, POST, PUT, DELETE)

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a GET, POST, PUT, DELETE request to the API endpoint.
2. Check the status code returned.
3. Verify that the status code is as expected.

**Expected results:** HTTP method with recommended response code

1. GET - HTTP response code 200 (OK)
2. POST - HTTP response code 201 (Created)
3. PUT - HTTP response code 201 (Created)
4. DELETE - HTTP response code 200 (OK)

=========================================================================================

**TC07:** Verify that the API response is in the expected format (e.g., JSON, XML)

17 February 2025

10:40 AM

**Description:** This test verifies that the API response is in the correct format, such as JSON or XML.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a valid request (e.g., GET) to the API endpoint.
2. Capture the Content-Type header from the response.
3. Verify that the Content-Type header matches the expected format (e.g., application/json or application/xml).
4. Optionally, inspect the response body to confirm it matches the expected format (e.g., valid JSON or XML).

**Expected results:** The Content-Type header should reflect the correct response format (e.g., application/json), and the response body should be formatted correctly.

=========================================================================================

**TC08:** Verify that the API response contains all the expected fields

17 February 2025

10:40 AM

**Description:** This test ensures that the API returns all the fields expected in the response body.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a valid request to the API endpoint.
2. Capture the response body.
3. Compare the response with the API documentation or expected field list.
4. Verify that all expected fields (e.g., id, name, status, created) are present in the response body.

**Expected results:** The response data is accordance with the expected schema fields

=========================================================================================

**TC09:** Verify that the API response contains the correct data for each field

17 February 2025

10:40 AM

**Description:** This test ensures that the data returned for each field is accurate and matches expectations.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a valid request to the API endpoint.
2. Capture the response body.
3. Verify that the values in the response match the expected values (e.g., if the id field should be an integer, the name field should contain a string, etc.).
4. Optionally, compare the data with sample or reference data, if available.

**Expected results:** The values for each field in the response should be accurate and consistent with the expected data.

=========================================================================================

**TC10:** Verify that the API response time is within acceptable limits

17 February 2025

10:40 AM

**Description:** This test ensures that the API response time is within the specified limits (e.g., ≤ 3 seconds).

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a valid request to the API endpoint.
2. Measure the response time using a tool or API client (e.g., Postman, cURL).
3. Compare the response time to the predefined acceptable limit (e.g., ≤ 3 seconds).

**Expected results:** The response time should be within the acceptable limits (e.g., ≤ 3 seconds).

=========================================================================================

**TC11:** Verify that the API request parameters are correctly passed to the API

17 February 2025

10:40 AM

**Description:** This test ensures that query parameters, path parameters, and body parameters are correctly passed and processed by the API.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a GET or POST request with query parameters (e.g., /items?page=2&size=10) or path parameters (e.g., /items/{id}).
2. Capture the response.
3. Check if the response correctly reflects the parameters passed in the request (e.g., the returned items should match the requested page and size).

**Expected results:** The API should correctly handle the request parameters and return the expected results, such as pagination or filtering.

=========================================================================================

**TC12:** Verify that the API request method is correct (e.g. GET, POST, PUT, DELETE)

17 February 2025

10:40 AM

**Description:** This test ensures that the API handles different HTTP methods appropriately (e.g., GET for retrieval, POST for creation).

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a valid request using the specified HTTP method (e.g., GET, POST, PUT, DELETE).
2. Capture the response.
3. Verify that the API responds as expected for the method (e.g., a GET request retrieves data, a POST request creates a new resource, etc.).

**Expected results:** The API should correctly handle the request method and return the appropriate status and data.

=========================================================================================

**TC13:** Verify that the API endpoint URL is correct

17 February 2025

10:40 AM

**Description:** This test verifies that the API is reachable at the specified endpoint and responds as expected.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Verify the API endpoint URL (e.g., /items/{id} or /users).
2. Send a valid request to the API endpoint.
3. Check the status code and the response body to ensure the correct data is returned.

**Expected results:** The API should return the expected response and data for the correct endpoint URL.

=========================================================================================

**TC14:** Verify that the API returns an error message if the request is malformed

17 February 2025

10:40 AM

**Description:** This test ensures that the API responds with a meaningful error message when a request is malformed (e.g., missing required parameters, invalid JSON).

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a malformed request (e.g., missing required query parameter or invalid JSON body).
2. Capture the response status code and body.
3. Verify that the API responds with an appropriate error status (e.g., 400 Bad Request) and a meaningful error message describing the issue.

**Expected results:** The API should return an error with a 400 Bad Request status code (or equivalent) and an error message explaining what is wrong with the request.

=========================================================================================

**TC15:** Verify that the API returns an error message if the authentication fails

17 February 2025

10:40 AM

**Description:** This test verifies that the API returns an appropriate error when authentication fails (e.g., due to invalid credentials or missing authentication tokens).

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a request to the API endpoint without providing valid authentication credentials (e.g., missing API key or invalid token).
2. Capture the response.
3. Verify that the response status code is 401 Unauthorized (or the relevant error code for failed authentication).
4. Check the response body for an error message indicating that authentication has failed (e.g., "Authentication failed" or "Invalid credentials").

**Expected results:** The response should have a 401 Unauthorized status code, and the error message should indicate authentication failure.

=========================================================================================

**TC16:** Verify that the API returns an error message if the request payload is missing

17 February 2025

10:40 AM

**Description:** This test ensures that the API responds with an error when the request body (payload) is missing or incomplete, if it is required.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a request (e.g., POST or PUT) without including the required payload in the request body.
2. Capture the response.
3. Verify that the response status code is 400 Bad Request.
4. Check the response body for an appropriate error message indicating that the payload is missing (e.g., "Request payload is required").

**Expected results:** The response should have a 400 Bad Request status code and an error message indicating that the payload is missing.

=========================================================================================

**TC17:** Verify that the API correctly handles cookies and returns the correct HTTP status code

17 February 2025

10:40 AM

**Description:** This test ensures that the API correctly handles cookies (e.g., setting and retrieving cookies) and returns the correct HTTP status code.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a valid request to the API endpoint, ensuring that the server sets a cookie (e.g., authentication token or session ID).
2. Capture the response and check the Set-Cookie header to verify that the cookie is set correctly.
3. Send another request with the captured cookie in the Cookie header.
4. Verify that the API correctly returns a valid response with an appropriate status code (e.g., 200 OK) when the cookie is sent with the request.

**Expected results:** The API should correctly set the cookie in the response and handle it appropriately in subsequent requests, returning a 200 OK status code or another relevant status.

=========================================================================================

**TC18:** Verify that the API correctly handles caching and returns the correct HTTP status code

17 February 2025

10:40 AM

**Description:** This test checks whether the API correctly handles caching, such as using cache headers (e.g., Cache-Control, ETag) to manage responses.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a request to the API and capture the response headers.
2. Verify that caching-related headers (e.g., Cache-Control, ETag) are present in the response.
3. Send the same request again and check if the response is served from the cache by checking the Cache-Control or X-Cache headers (depending on the server).
4. Ensure that the response status code is 200 OK for cache hits or 304 Not Modified if the content is unchanged.

**Expected results:** The API should correctly include cache-related headers and return the correct status code based on caching behavior (200 OK or 304 Not Modified).

=========================================================================================

**TC19:** Verify that the API correctly handles cross-site scripting (XSS) attacks and returns the correct HTTP status code

17 February 2025

10:40 AM

**Description:** This test ensures that the API is resistant to cross-site scripting (XSS) attacks and returns a correct error code or sanitized output if such an attack is detected.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a request with malicious payload containing XSS code in one of the input fields (e.g., <script>alert('XSS')</script>).
2. Capture the response.
3. Verify that the API does not execute the script and instead returns a sanitized response.
4. Ensure that the API returns a status code of 400 Bad Request or another appropriate error code for invalid input.
5. Check the response body to ensure the XSS attempt is blocked, and any malicious input is sanitized or rejected.

**Expected results:** The API should return a 400 Bad Request or 422 Un-processable Entity status code with a message indicating that the input is invalid, and it should not execute any malicious scripts.

=========================================================================================

**TC20:** Verify that the API correctly handles SQL injection attacks and returns the correct HTTP status code

17 February 2025

10:40 AM

**Description:** This test ensures that the API is secure against SQL injection attacks and returns the correct error code when such attacks are attempted.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a request with a payload designed to test for SQL injection, such as username=' OR 1=1 -- in a query parameter or request body.
2. Capture the response.
3. Verify that the API does not execute any harmful SQL queries and returns a secure error message or status code.
4. Ensure that the API returns a 400 Bad Request, 422 Un-processable Entity, or similar status code indicating that the input is invalid or the query was malformed.
5. Check the response body to confirm that the API does not expose any database error messages or details that could help an attacker.

**Expected results:** The API should return a 400 Bad Request or similar error status code, indicating the request was malformed, and should never execute any SQL injection queries.

=========================================================================================

**TC21:** Verify that the API correctly handles CSRF attacks and returns the correct HTTP status code

23 April 2025

10:04 PM

**Description:** This test ensures that the API is secure against CSRF injection attacks and returns the correct error code when such attacks are attempted.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a request with a payload designed to test for CSRF injection, such as <img src=x onerror=alert('CSRF')> in the request body.
2. Capture the response.
3. Verify that the API does not execute any harmful CSRF and returns a secure error message or status code.
4. Ensure that the API returns a 400 Bad Request, 422 Un-processable Entity, or similar status code indicating that the input is invalid or the query was malformed.
5. Check the response body to confirm that the API does not expose any database error messages or details that could help an attacker.

**Expected results:** The API should return a 400 Bad Request or similar error status code, indicating the request was malformed, and should never execute any CSRF

=========================================================================================

**TC22:** Schema validation for successful response

17 February 2025

10:40 AM

**Description:** This test case validates the schema of the response returned when the API request is successful. The schema must match the expected structure for a successful request, including proper field types, required fields, and data constraints.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a valid API request (e.g., GET /api/resource).
2. Verify that the HTTP status code returned is 200 (OK).
3. Validate that the response body matches the expected JSON schema, which could include fields such as:
   * status (string, expected value: "success")
   * data (array or object, depending on the API)
   * message (string, optional)
4. Ensure that all required fields are present and that the types of fields (e.g., string, integer, array) match the expected schema.
5. If any field is missing or has an incorrect type, mark the test as failed.

**Expected results:** The response body matches the expected JSON schema for a successful request. Status code is 200/201. All fields are present with correct types and values.

=========================================================================================

**TC23:** Schema validation for unauthorised request

17 February 2025

10:40 AM

**Description:** This test case checks the schema of the response when the API request is unauthorized (e.g., missing or invalid authentication token). It ensures that the structure of the response for an unauthorized request adheres to the expected error response format.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send an API request without a valid authentication token (e.g., GET /api/resource with missing or invalid token).
2. Verify that the HTTP status code returned is 401 Unauthorized or 403 Forbidden.
3. Validate that the response body matches the expected error schema:
   * status (string, expected value: "error")
   * error\_code (integer, e.g., 401 for unauthorized)
   * message (string, e.g., "Unauthorized access")
4. Ensure that all required error fields are present and the field types are correct.
5. If the response schema deviates from the expected error structure, mark the test as failed.

**Expected results:** The response body matches the expected JSON schema for an unauthorized request. Status code is 401 or 403. All fields are present with the correct types and values.

=========================================================================================

**TC24:** Schema validation for a valid payload

17 February 2025

10:40 AM

**Description:** This test case ensures that the schema of the payload being sent in the API request is validated. It checks whether the payload conforms to the expected format, including required fields, data types, and constraints.

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a request with a payload (e.g., POST /api/resource with JSON data).
2. Ensure the payload is structured correctly according to the API documentation, including required fields and correct data types. For example:
   * username (string, required)
   * email (string, valid email format)
   * password (string, required)
   * age (integer, optional)
3. If the API returns an error due to an invalid payload (e.g., missing fields or incorrect data types), the test should capture the error response and validate the schema for the error response.
4. Ensure that the payload does not violate any additional constraints (e.g., string length limits, valid email format, etc.).

**Expected results:** The payload is correctly structured, and all required fields are provided with the correct types. If the payload is invalid, the response should contain an error message with the correct schema.

=========================================================================================

**TC25: Verify that the API returns a 403 status code for unauthorised request**

25 April 2025

00:15 AM

**Description:** Ensure that an unauthorised request is not processed and a correct status code and response message is returned

**Pre-condition:** Key preconditions that should be met

* The API server should be up and running.
* The endpoint is accessible.
* A valid authorization token is available.
* The service account has the correct authorization and permissions to access the endpoints.

**Test steps:**

1. Send a request with a payload (e.g., POST /api/resource with JSON data).
2. Ensure the the Bearer token is invalid
3. If the API returns an error due to an invalid authorisation

**Expected results:** A 403 status code for unauthorised request is returned.