**API Task**

1. **API Tests**

**1.  Authenticate User Tests**

|  |  |
| --- | --- |
| **ID** | **Test Case** |
| AU01 | Valid credentials |
| AU02 | Invalid email format |
| AU03 | Non-existent email |
| AU04 | Wrong password |
| AU05 | Empty email field |
| AU06 | Empty password field |

**2.  Create User Tests**

|  |  |
| --- | --- |
| **ID** | **Test Case** |
| CU01 | Valid user data |
| CU02 | Missing name |
| CU03 | Missing email |
| CU04 | Missing password |
| CU05 | Invalid email format |
| CU06 | Email already exists |
| CU07 | Password too short |
| CU08 | Special characters in name |

**3. Get User Tests**

|  |  |
| --- | --- |
| **ID** | **Test Case** |
| GU01 | Valid token |
| GU02 | Invalid token |
| GU03 | Missing token |

**4. Patch User Tests**

|  |  |
| --- | --- |
| **ID** | **Test Case** |
| PU01 | Update all fields with valid data |
| PU02 | Update only name |
| PU03 | Update only email |
| PU04 | Update only password |
| PU05 | Update with invalid email format |
| PU06 | Update with email that already exists |
| PU07 | Update with invalid token |
| PU08 | Empty request body |

**5. Delete User Tests**

|  |  |
| --- | --- |
| **ID** | **Test Case** |
| DU01 | Valid token |
| DU02 | Invalid token |
| DU03 | Missing token |

**6.  Delete All Users Tests**

|  |  |
| --- | --- |
| **ID** | **Test Case** |
| DA01 | Valid admin key |
| DA02 | Invalid admin key |
| DA03 | Missing admin key |

1. **Bug Report**

**Authentication Endpoint**

**Bug #1: Authentication API returns 401 instead of 400 for invalid email format**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-001 |
| **Test Case ID** | AU02 - Invalid email format |
| **Description** | The authentication API endpoint `/api/v1/auth` returns HTTP status code 401 (Unauthorized) instead of 400 (Bad Request) when an invalid email format is provided. |
| **Expected Result** | - When a user provides an invalid email format, the API should return HTTP status code 400 (Bad Request)  - The response should include an appropriate error message about incorrect email or password |
| **Actual Result** | 401 Unauthorized status code |
| **Steps to Reproduce** | 1. Send POST request to `/api/v1/auth`  2. Include invalid email format in request body  3. Include valid password in request body  4. Observe the HTTP status code returned |
| **Severity** | Medium |
| **Priority** | Medium |

**Bug #2: Authentication API returns 401 instead of 400 for empty email field**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-002 |
| **Test Case ID** | AU05 - Empty email field |
| **Description** | The authentication API endpoint `/api/v1/auth` returns HTTP status code 401 (Unauthorized) instead of 400 (Bad Request) when an empty email field is provided. |
| **Expected Result** | - When a user provides an empty email field, the API should return HTTP status code 400 (Bad Request)  - The response should indicate a client-side error for missing required field |
| **Actual Result** | 401 Unauthorized status code |
| **Steps to Reproduce** | 1. Send POST request to `/api/v1/auth`  2. Include empty email field in request body  3. Include valid password in request body  4. Observe the HTTP status code returned |
| **Severity** | Medium |
| **Priority** | Medium |

**Bug #3: Authentication API returns 401 instead of 400 for empty password field**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-003 |
| **Test Case ID** | AU06 - Empty password field |
| **Description** | When attempting to authenticate with an empty password field, the API returns a 401 Unauthorized status code instead of the expected 400 Bad Request. |
| **Expected Result** | - When a user provides an empty password field, the API should return HTTP status code 400 (Bad Request)  - The response should indicate a client-side error for missing required field |
| **Actual Result** | 401 Unauthorized status code |
| **Steps to Reproduce** | 1. Send POST request to `/api/v1/auth`  2. Include valid email in request body  3. Include empty password field in request body  4. Observe the HTTP status code returned |
| **Severity** | Medium |
| **Priority** | Medium |

**Create User End Point**

**Bug#4: User API inconsistently returns 200 or 401 instead of 400 for missing name field**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-004 |
| **Test Case ID** | CU02 - Missing name |
| **Description** | When attempting to create a user without the name field, the API returns inconsistent status codes across different runs. It initially returns 200 OK (implying success) and later returns 401 Unauthorized. The expected behaviour is a 400 Bad Request indicating that the name is required. |
| **Expected Result** | - HTTP 400 Bad Request - Error message stating that name is a required field |
| **Actual Result** | - First run: 200 OK - Later runs: 401 Unauthorized Both are incorrect |
| **Steps to Reproduce** | 1. Send POST request to `/api/v1/users `  2. Include valid email in request body  3. Include valid password field in request body  4. Observe varying HTTP status code responses across runs returned |
| **Severity** | High |
| **Priority** | High |

**Bug#5: User API inconsistently returns 200 or 401 instead of 400 for missing email field**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-005 |
| **Test Case ID** | CU03 - Missing email |
| **Description** | When registering a user with no email, the API returns either 200 OK or 401 Unauthorized depending on the test run. It should consistently return 400 Bad Request. |
| **Expected Result** | - HTTP 400 Bad Request - Error message stating that email is a required field |
| **Actual Result** | - First run: 200 OK - Later runs: 401 Unauthorized Both are incorrect |
| **Steps to Reproduce** | 1. Send POST request to `/api/v1/users `  2. Include valid name in request body  3. Include valid password field in request body  4. Observe varying HTTP status code responses across runs returned |
| **Severity** | High |
| **Priority** | High |

**Bug#6: User API returns 200 or 401 instead of 400 for missing password field**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-006 |
| **Test Case ID** | CU04 - Missing password |
| **Description** | Submitting a registration request without a password returns 200 OK in some cases and 401 Unauthorized in others. The API should reject the request with 400 Bad Request. |
| **Expected Result** | - HTTP 400 Bad Request - Error message stating that password is a required field |
| **Actual Result** | - First run: 200 OK - Later runs: 401 Unauthorized Both are incorrect |
| **Steps to Reproduce** | 1. Send POST request to `/api/v1/users `  2. Include valid name in request body  3. Include valid email field in request body  4. Observe varying HTTP status code responses across runs returned |
| **Severity** | High |
| **Priority** | High |

**Bug#7: User API inconsistently returns 200 or 401 instead of 400 for missing password field**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-007 |
| **Test Case ID** | CU05 - Invalid email format |
| **Description** | Submitting a request with an invalid email format returns 200 OK or 401 Unauthorized in different test runs. Expected behaviour is 400 Bad Request. |
| **Expected Result** | - HTTP 400 Bad Request  - Message indicating invalid email format |
| **Actual Result** | - First run: 200 OK - Later runs: 401 Unauthorized Both are incorrect |
| **Steps to Reproduce** | 1. Send POST request to `/api/v1/users `  2. Include valid name in request body  3. Include valid password field in request body  4. Include an invalid email field in request body  5. Observe varying HTTP status code responses across runs returned |
| **Severity** | High |
| **Priority** | High |

**Bug#8: User API returns 401 instead of 409 for existing email address**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-008 |
| **Description** | When trying to register with an email that already exists, the API returns 401 Unauthorized instead of the expected 409 Conflict. This makes it appear that authentication is required for signup, which is incorrect. |
| **Expected Result** | - HTTP 409 Conflict  - Message: "Email already exists" or equivalent |
| **Actual Result** | 401 Unauthorized status code |
| **Steps to Reproduce** | 1. Register a user normally 2. Try registering again with the same email 3. Observe the status code returned |
| **Severity** | Medium |
| **Priority** | Medium |

**Bug#9: User API inconsistently returns 200 or 401 instead of 400 for short password**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-009 |
| **Description** | Providing a password shorter than the required minimum (e.g., "123") leads to either 200 OK or 401 Unauthorized. The expected result is a 400 Bad Request with a clear validation error. |
| **Expected Result** | - HTTP 400 Bad Request - Error message about password length |
| **Actual Result** | - First run: 200 OK - Later runs: 401 Unauthorized Both are incorrect |
| **Steps to Reproduce** | 1. Send POST request to `/api/v1/users `  2. Include valid name in request body  3. Include valid email field in request body  4. Include short password field in request body  5. Observe varying HTTP status code responses across runs returned |
| **Severity** | High |
| **Priority** | High |

**Get Users**

**Bug#11: Get User API returns 403 instead of 401 for missing token**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-011 |
| **Test Case ID** | GU03 - Missing token |
| **Description** | When attempting to get user information without providing an authentication token, the API returns a 403 Forbidden status code instead of the expected 401 Unauthorized. |
| **Expected Result** | - HTTP 401 Unauthorized  - Error message containing "missing token" or "unauthorized" |
| **Actual Result** | 403 Forbidden status code |
| **Steps to Reproduce** | 1. Send GET request to `/api/v1/users`  2. Do not include any authentication token in headers  3. Observe HTTP status code response |
| **Severity** | Medium |
| **Priority** | Medium |

**PATCH USER BY TOKEN**

**Bug#12: Update User API returns 403 instead of 200 for update only name operation**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-012 |
| **Test Case ID** | PU02 - Update only name |
| **Description** | When attempting to update only the name field using a valid token, the API returns a 403 Forbidden instead of the expected 200 OK. |
| **Expected Result** | - HTTP 200 OK - Message: "User updated with success!" |
| **Actual Result** | 403 Forbidden status code |
| **Steps to Reproduce** | 1. Send PATCH request to /api/v1/users 2. Add valid Authorization token 3. Send {name: "Bebo”} in body 4. Observe status code |
| **Severity** | High |
| **Priority** | High |

**Bug#13: Update User API returns 403 instead of 200 for update only email operation**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-013 |
| **Test Case ID** | PU03 - Update only email |
| **Description** | When attempting to update only the email field using a valid token, the API returns a 403 Forbidden instead of the expected 200 OK. |
| **Expected Result** | - HTTP 200 OK - Message: "User updated with success!" |
| **Actual Result** | 403 Forbidden status code |
| **Steps to Reproduce** | 1. Send PATCH request to /api/v1/users 2. Add valid Authorization token 3. Send {name: "bebo@gmail.com”} in body 4. Observe status code |
| **Severity** | High |
| **Priority** | High |

**Bug#14: Update User API returns 403 instead of 200 for update only password operation**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-014 |
| **Test Case ID** | PU04 - Update only password |
| **Description** | Sending a valid password update results in a 403 Forbidden, even though the token and body are valid. |
| **Expected Result** | - HTTP 200 OK - Confirmation message about update |
| **Actual Result** | 403 Forbidden status code |
| **Steps to Reproduce** | 1. Send PATCH request to /api/v1/users 2. Add valid Authorization token 3. Send {password: "Habibapassword123" } in body 4. Observe status code |
| **Severity** | High |
| **Priority** | High |

**Bug#15: Update User API returns 403 instead of 400 for update with invalid format**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-015 |
| **Test Case ID** | PU05 - Invalid email format |
| **Description** | When providing an improperly formatted email during update, the API returns 403 Forbidden instead of 400 Bad Request. |
| **Expected Result** | - HTTP 400 Bad Request - Error message indicating invalid email format |
| **Actual Result** | 403 Forbidden status code |
| **Steps to Reproduce** | 1. Send PATCH request to /api/v1/users 2. Add valid Authorization token 3. Send {email: "not-an-email"} in body 4. Observe status code |
| **Severity** | Medium |
| **Priority** | Medium |

**Bug#16: Update User API returns 403 instead of 409 when update a user's email to one that already exists**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-016 |
| **Test Case ID** | PU06 - Email already exists |
| **Description** | When attempting to update a user's email to one that already exists in the system, the API returns a 403 Forbidden status code instead of the expected 409 Conflict. |
| **Expected Result** | - HTTP 409 Conflict - Error message: "Email already exists" |
| **Actual Result** | 403 Forbidden status code |
| **Steps to Reproduce** | |  | | --- | |  |  |  | | --- | | 1. Use a valid token 2. Attempt to update to an already-used email 3. Observe the status code | |
| **Severity** | High |
| **Priority** | High |

**Bug#17: Update User API returns 403 instead of 400 for update with invalid format**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-017 |
| **Test Case ID** | PU08 - Empty request body |
| **Description** | Sending an empty body during user update results in 403 Forbidden. The expected response is 400 Bad Request for missing update fields. |
| **Expected Result** | - HTTP 400 Bad Request - Error: "No fields to update" or similar |
| **Actual Result** | 403 Forbidden status code |
| **Steps to Reproduce** | 1. Send PATCH request to /api/v1/users 2. Add valid Authorization token 3. Send {} in body 4. Observe status code |
| **Severity** | Medium |
| **Priority** | Medium |

**Delete All Users**

**Bug#18: Delete All Users API returns 403 instead of 401 for missing admin key**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-018 |
| **Test Case ID** | DA03 - Missing admin key |
| **Description** | When a request is made to delete all users without providing the required admin key, the API responds with a 403 Forbidden status. Since the request lacks authentication credentials, it should return 401 Unauthorized. |
| **Expected Result** | - HTTP 401 Unauthorized  - Message indicating missing or required admin authentication |
| **Actual Result** | 403 Forbidden status code |
| **Steps to Reproduce** | 1. Send a DELETE request to /api/v1/all-users without any authentication headers 2. Observe the returned status code |
| **Severity** | Medium |
| **Priority** | Medium |

**Delete Users End Point**

**Bug#19: Delete User API returns 403 instead of 401 for missing token**

|  |  |
| --- | --- |
| **Field** | **Value** |
| **Bug ID** | BUG-019 |
| **Test Case ID** | DU03 - Missing token |
| **Description** | When attempting to delete a user without providing an authentication token, the API returns 403 Forbidden. The expected behaviour is a 401 Unauthorized since no credentials were supplied. |
| **Expected Result** | - HTTP 401 Unauthorized  - Message indicating that authentication is required |
| **Actual Result** | 403 Forbidden status code |
| **Severity** | Medium |
| **Priority** | Medium |