# Testing

# Integration Testing for the Sprint:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test Case ID | Module Name | Test Scenario | Expected Output | Actual Output | Status | Tools Used | Commands Used |
| |  | | --- | | **IT001** | | Get My Donation | Login with correct credentials, verify otp, and then fetches the otp | User’s donations fetched successfully | User’s donations fetched successfully | **✅**Pass | Jest, Supertest | npm test donations.integration.test.js |
| **IT002** | Create Donation | User create donation with correct data | Donation created successfully | Donation created successfully | **✅**Pass | Jest, Supertest | npm test donations.integration.test.js |
| **IT003** | Create Donation | User create donation with incorrect data | Error message: " missing required fields  " | Error message displayed | **✅**Pass | Jest, Supertest | npm test donations.integration.test.js |
| **IT004** | Edit Donation | User edit donation with correct data | Donation edited successfully | Donation edited successfully | **✅**Pass | Jest, Supertest | npm test donations.integration.test.js |
| **IT005** | Edit Donation | User edit donation with incorrect data | Error message: " missing required fields" | Error message displayed | **✅**Pass | Jest, Supertest | npm test donations.integration.test.js |
| **IT006** | Delete Donation | User delete donation with correct id | Donation deleted successfully | Donation deleted successfully | **✅**Pass | Jest, Supertest | npm test donations.integration.test.js |
| **IT007** | Create AI Chat | User sends valid chat request | AI response returned successfully | AI response returned successfully | **✅**Pass | Jest, Supertest | npm test aichat.integration.test.js |
| **IT008** | Create AI Chat | User sends invalid role in chat request | Error message: "Invalid role" | Error message displayed | **✅**Pass | Jest, Supertest | npm test aichat.integration.test.js |
| **IT009** | Create AI Chat | User sends empty messages array | Error message: "Messages array cannot be empty" | Error message displayed | **✅Pass** | Jest, Supertest | npm test aichat.integration.test.js |
| **IT010** | Create AI Chat | User sends no messages field | Error message: "Messages field is required" | Error message displayed | **✅Pass** | Jest, Supertest | npm test aichat.integration.test.js |
| **IT011** | Create AI Chat | User sends message without content | Error message: "Message content is required" | Error message displayed | **✅Pass** | Jest, Supertest | npm test aichat.integration.test.js |
| **IT012** | Create AI Chat | User sends message without role | Error message: "Role is required” | Error message displayed | **✅Pass** | Jest, Supertest | npm test aichat.integration.test.js |
| **IT013** | Get Labs | User get labs | Fetches labs successfully | Fetches labs successfully | **✅Pass** | Jest, Supertest | npm test labs.integration.test.js |
| **IT014** | Create Labs | User create a lab with correct data | labs created successfully | labs created successfully | **✅Pass** | Jest, Supertest | npm test labs.integration.test.js |
| **IT015** | Create Labs | User create a lab with incorrect data | Error message: “missing labName and invalid date format” | Error message displayed | **✅Pass** | Jest, Supertest | npm test labs.integration.test.js |
| **IT016** | Create Labs | User create a lab without patientId | Error message: “patientId does not exist” | Error message displayed | **✅Pass** | Jest, Supertest | npm test labs.integration.test.js |
| **IT017** | Create Labs | User create a lab with invalid patientId | Error message: “invalid patientId” | Error message displayed | **✅Pass** | Jest, Supertest | npm test labs.integration.test.js |

Code:

import 'dotenv/config';

import express from 'express';

import 'express-async-errors';

import request from 'supertest';

import mongoose from 'mongoose';

import AIChatRouter from '../../routes/ai-chat.routes.js';

import { connectDB } from '../../config/database.js';

import { getSessionMiddleware } from '../../middleware/session-middleware.js';

let app;

const baseRoute = '/api/ai-chat';

beforeAll(async () => {

  await connectDB();

  app = express();

  app.use(express.json());

  app.use(getSessionMiddleware());

  app.use(baseRoute, AIChatRouter);

});

afterAll(async () => {

  await mongoose.connection.close();

});

describe('POST /api/ai-chat', () => {

  const endpoint = `${baseRoute}/`;

  const postAIChat = async (data) => {

    return await request(app).post(endpoint).send(data);

  };

  it('should return a valid AI response for a valid chat request', async () => {

    const postData = {

      messages: [{ role: 'user', message: 'What are the lab timings?' }],

    };

    const res = await postAIChat(postData);

    expect(res.statusCode).toBe(200);

    expect(res.body).toEqual(

      expect.objectContaining({

        data: expect.objectContaining({

          message: expect.any(String),

        }),

      })

    );

  });

  it('should fail with invalid message schema', async () => {

    const postData = {

      messages: [{ role: 'invalid-role', message: 'What are the lab timings?' }],

    };

    const res = await postAIChat(postData);

    expect(res.statusCode).toBe(400);

    expect(res.body).toEqual(

      expect.objectContaining({

        message: expect.any(String),

      })

    );

  });

  it('should fail when messages array is empty', async () => {

    const postData = {

      messages: [],

    };

    const res = await postAIChat(postData);

    expect(res.statusCode).toBe(400);

    expect(res.body).toEqual(

      expect.objectContaining({

        message: expect.any(String),

      })

    );

  });

  it('should fail when no messages field is provided', async () => {

    const postData = {};

    const res = await postAIChat(postData);

    expect(res.statusCode).toBe(400);

    expect(res.body).toEqual(

      expect.objectContaining({

        message: expect.any(String),

      })

    );

  });

  it('should fail when message content is missing', async () => {

    const postData = {

      messages: [{ role: 'user' }],

    };

    const res = await postAIChat(postData);

    expect(res.statusCode).toBe(400);

    expect(res.body).toEqual(

      expect.objectContaining({

        message: expect.any(String),

      })

    );

  });

  it('should fail when role is missing in the message', async () => {

    const postData = {

      messages: [{ message: 'What are the lab timings?' }],

    };

    const res = await postAIChat(postData);

    expect(res.statusCode).toBe(400);

    expect(res.body).toEqual(

      expect.objectContaining({

        message: expect.any(String),

      })

    );

  });

});

Result:



**System Testing for the Sprint:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test Case ID | Module Name | Test Scenario | Expected Output | Actual Output | Status | Tools Used | Commands Used |
| |  | | --- | | **ST001** | | Patient Login | Login with correct credentials, verify otp | Patient Logged in successfully | Patient Logged in successfully | **✅**Pass | Puppeteer | node .\system-tests\ feedback.system.js |
| **ST002** | Patient gives Feedback | Login with correct credentials, verify otp and then give Feedback | Feedback given successfully | Feedback given successfully | **✅**Pass | Puppeteer | node .\system-tests\ feedback.system.js |
| **ST003** | Patient Edit Profile | User edit his profile | Profile edited successfully | Profile edited successfully | **✅**Pass | Puppeteer | node .\system-tests\patient-edit-form.system.js |
| **ST004** | Patient Matches Tab | User directs towards the Matches Tab | Matches Tab opened successfully | Matches Tab opened successfully | **✅**Pass | Puppeteer | node .\system-tests\patient-matches-tab.system.js |
| **ST005** | Patient Clinical Trials | User goes to clinical trial and inquire doctor info | Info shown successfully | Info shown successfully | **✅**Pass | Puppeteer | node .\system-tests\patient-clinical-trials-tab.system.js |
| **ST006** | Patient Clinical Trials | User goes to clinical trial and then clicks match button | Match tab appear successfully | Match tab appear successfully | **✅**Pass | Puppeteer | node .\system-tests\patient-clinical-trials-tab.system.js |
| **ST007** | Filter in Matches Tab | User Filter out the Matched based on age, gender and disease | Filtered successfully | Filtered successfully | **✅**Pass | Puppeteer | node .\system-tests\filter-matches-tab.system.js |
| **ST008** | Doctor Login | Login with correct credentials, verify otp | Doctor Logged in successfully | Doctor Logged in successfully | **✅**Pass | Puppeteer | node .\system-tests\ \_doctor-login |
| **ST009** | Doctor Matches Tab | User directs towards the Matches Tab | Matches Tab opened successfully | Matches Tab opened successfully | **✅**Pass | Puppeteer | node .\system-tests\doctor-matches-tab.system.js |
| **ST010** | Doctor’s Patient Tab | User goes to patient tab and inquire patient info | Info shown successfully | Info shown successfully | **✅**Pass | Puppeteer | node .\system-tests\doctor-patient-tab.system copy.js |
| **ST011** | Doctor’s Patient Tab | User goes to patient tab and then clicks match button | Match tab appear successfully | Match tab appear successfully | **✅**Pass | Puppeteer | node .\system-tests\doctor-patient-tab.system copy.js |
| **ST012** | Doctor: Create Lab Appointment | User goes to lab form, fills the form and create lab | Lab appointment created successfully | Lab appointment created successfully | **✅**Pass | Puppeteer | node .\system-tests\create-lab-appointment.system.js |
| **ST013** | Doctor: View My Lab Appointments | User goes to my lab tabs | Lab rows are displayed if doctor has them | Lab rows are displayed if doctor has them | **✅**Pass | Puppeteer | node system-tests/my-labs.system.js |
| **ST013** | Doctor: View My Lab Appointments | User goes to my lab tabs | “No labs found” is displayed if doctor does not have them | “No labs found” is displayed if doctor does not have them | **✅**Pass | Puppeteer | node system-tests/my-labs.system.js |
| **ST014** | AI Chat box | User goes to ai chat box | User is greeted with welcome message | User is greeted with welcome message | **✅**Pass | Puppeteer | node system-tests/ai-chatbox.system.js |
| **ST015** | AI Chat box | User goes to ai chat box | User sends a message and wait for response | User sends a message and wait for response | **✅**Pass | Puppeteer | node system-tests/ai-chatbox.system.js |
| **ST015** | AI Chat box | User goes to ai chat box | AI bot responds to user message | AI bot responds to user message | **✅**Pass | Puppeteer | node system-tests/ai-chatbox.system.js |
| **ST016** | Donor: Create Donation Listing | User goes to create donation form and fill the form | Donation created successfully | Donation created successfully | **✅**Pass | Puppeteer | node system-tests/create-donation.system.js |
| **ST017** | Donor: View My Donations | User goes to my donations tab | Lab rows are displayed if donor has donations | Lab rows are displayed if donor has donations | **✅**Pass | Puppeteer | node system-tests/my-donations.system.js |
| **ST018** | Donor: View My Donations | User goes to my donations tab | “No donations found” is displayed if donor has no donations | “No donations found” is displayed if donor has no donations | **✅**Pass | Puppeteer | node system-tests/my-donations.system.js |
| **ST019** | Donor: View all donations | User goes to donations tab | Donations cards are displayed with button to contact the donor. When button is clicked a page opens with donor contact info | Donations cards are displayed with button to contact the donor. When button is clicked a page opens with donor contact info | **✅**Pass | Puppeteer | node system-tests/all-donations.system.js |
| **ST020** | Donor: view all donations | User goes to donations tab | “No donations available” message is displayed if there are no items for donation. | “No donations available” message is displayed if there are no items for donation. | **✅**Pass | Puppeteer | node system-tests/all-donations.system.js |
| **ST021** | Donor: Delete Donation | Users deletes a donation | User goes to my donations page, wait for donations to load, click to delete a notification. He is redirected to confirm the delete operation. On confirmation the donation is deleted. | User goes to my donations page, wait for donations to load, click to delete a notification. He is redirected to confirm the delete operation. On confirmation the donation is deleted. | **✅**Pass | Puppeteer | node system-tests/delete-donation.system.js |
| **ST022** | Donor: Edit a donation | User wants to edit a donation | User goes to my donations page, wait for donations to load, click to edit a notification. Edit donation page opens to edit the donation. User updates data and submit to update the donation | User goes to my donations page, wait for donations to load, click to edit a notification. Edit donation page opens to edit the donation. User updates data and submit to update the donation | **✅**Pass | Puppeteer | node system-tests/edit-donation.system.js |

**Unit Testing for the Sprint:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test Case ID | Module Name | Test Scenario | Expected Output | Actual Output | Status | Tools Used | Commands Used |
| |  | | --- | | **UT001** | | Post AI Chat | Request with correct data | successful response | successful response | **✅**Pass | Jest | npm run test |
| |  | | --- | | **UT002** | | Post AI Chat | Request with empty messages | Throw Error ” Messages array must contain at least one message“ | Throw Error ” Messages array must contain at least one message“ | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT003** | | No OpenAI Choices | Valid message, but OpenAI returns no choices | HttpError thrown | HttpError thrown | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT004** | | postPatientRegisteration - Notify Matching Doctors | Valid patient data and doctors with matching specialization | Creates notifications for both doctor and patient | Creates notifications for both doctor and patient | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT005** | | postPatientRegisteration -Disease Not Found | Valid patient data but diseaseId is invalid | Handles error and does not send notifications | Handles error gracefully and does not send notifications | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT006** | | postPatientRegisteration - No Matching Doctors | Valid patient data but doctors do not match specialization | No notifications sent to doctors or patient | No notifications sent to doctors or patient | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT007** | | createLabAppointment - Successful Creation | Valid appointment body and authenticated doctor | Saves the appointment and sends notification to patient | Saves the appointment and sends notification to patient | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT008** | | createLabAppointment - Patient Not Found | Valid appointment body but patientId not in DB | Throws 404 HttpError with NotFoundError | Throws 404 HttpError with NotFoundError | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT009** | | createLabAppointment - Invalid Appointment Date | Body with invalid date format for appointmentDateTime | Zod error mentioning Invalid date format | Zod error mentioning Invalid date format | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT010** | | getAllLabs - Return Labs for Doctor | user login by type 'doctor' | Labs for the doctor should be fetched | Labs for the doctor should be fetched | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT011** | | getAllLabs - Unauthorized Access by Patient | user login by type 'patient' | Should throw HttpError with 403 status | throw HttpError with 403 status | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT012** | | createDonationController - Successful Donation Creation | Valid donation request and authenticated user | Should save the donation and return a success response | save the donation and return a success response | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT013** | | createDonationController - Missing Required Field | Donation request with empty 'equipmentName' | Should throw ZodError for validation failure | throw ZodError for validation failure | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT014** | | createDonationController - Invalid Years of Use | createDonationController is called | throw ZodError for invalid number | throw ZodError for invalid number | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT015** | | createDonationController - Handles Optional Fields | Donation request with optional fields omitted | create donation and return a valid success response | create donation and return a valid success response | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT016** | | deleteDonationController - Donation Already Deleted | Donation does not exist in DB | return success response with null data | return success response with null data | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT017** | | deleteDonationController - User Not Authorized | Authenticated user is not the donation owner | throw HttpError with status 403 | throw HttpError with status 403 | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT018** | | deleteDonationController - Successful Deletion | Authenticated user is the donation owner | Should delete donation and return success response | mockDeleteOne called, notification sent, ApiSuccessResponse | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT019** | | getAllDonationsController - Fetch All Donations | Query type='all' and is-taken='all' | Should return all donation entries | return all donation entries | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT020** | | getAllDonationsController - Fetch Donations By Current User | Query type='by-me' and is-taken='all' | Should return donations made by authenticated user | return donations made by authenticated user | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT021** | | getAllDonationsController - Fetch Not Taken & Not By User | Query type='not-by-me' and is-taken='false' | Should return available donations not made by the user | return available donations not made by the user | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT022** | | getAllDonationsController - Invalid Query Parameters | Query with type='invalid' | Should throw validation error from Zod schema | Should throw validation error from Zod schema | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT023** | | updateDonationController - Successful Update | Valid donation ID and update payload | Should update donation and notify user | update donation and notify user | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT024** | | updateDonationController - Donation Not Found | Donation ID that doesn't exist | Should throw 404 HttpError | throw 404 HttpError | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT025** | | updateDonationController - Unauthorized Update Attempt | Donation not owned by user | Should throw 403 HttpError | throw 403 HttpError | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT026** | | updateDonationController - Missing Required Field | Empty required field (equipmentType) | Should throw ZodError for schema validation | throw ZodError for schema validation | **✅Pass** | Jest | npm run test |
| |  | | --- | | **UT027** | | updateDonationController - Invalid yearsOfUse | yearsOfUse is 0 (non-positive) | Should throw ZodError | throw ZodError | **✅Pass** | Jest | npm run test |