Testing Instructions for the Ayurveda Project

This document outlines how to run tests for the Ayurveda project. It covers backend testing, frontend testing, end-to-end (E2E) testing, and running the full test suite via the run_tests.sh script.

Table of Contents

Introduction

The Ayurveda project includes a comprehensive testing framework that covers:

- **Backend Tests:** API endpoints and business logic (located in `ayurveda/tests/backend`).
- **Frontend Tests:** React component tests using Jest and React Testing Library (located in `ayurveda/tests/frontend`).
- **End-to-End Tests:** User flows using Cypress (configuration and tests in `cypress/e2e`).

This guide explains how to execute each type of test as well as how to run them all together using the provided `run_tests.sh` script.

Prerequisites

- **Software:** Node.js (v14 or higher), npm, Python 3.8+, and a package manager like pip or conda.
- **Dependencies:** Make sure you have installed all project dependencies by following instructions in the README.

For backend:

```
pip install -r requirements.txt
```

For frontend:

```
cd frontend
npm install
```

For Cypress (if E2E tests haven't been set up yet):

```
npm install --save-dev cypress
```

Running Backend Tests

Backend tests are written using pytest and are located in `ayurveda/tests/backend`.

Run all backend tests with:

```
pytest ./tests/backend
```

You can also run tests with a coverage report:

```
pytest --cov=.
```

Running Frontend Tests

Frontend tests are executed via npm using Jest and React Testing Library. They are located in `ayurveda/tests/frontend`.

Run the tests by navigating to the frontend directory:

```
cd frontend
npm test
```

For a coverage report, run:

```
npm test -- --coverage
```

Running End-to-End Tests

E2E testing for user flows is handled by Cypress. A sample test file is generated (if not already present) in `cypress/e2e/user_flows.cy.js` via the `run_tests.sh` script.

To run E2E tests headlessly:

```
npx cypress run
```

Alternatively, you can open the Cypress interactive test runner:

```
npx cypress open
```

Running All Tests Using run_tests.sh

The `run_tests.sh` script automates the testing process by:

1. Setting up environment variables for user-based testing scenarios.
2. Running backend tests.
3. Running frontend tests.
4. Generating and executing Cypress E2E tests.

Run the full test suite from the project root using:

```
sh run_tests.sh
```

If any tests fail, the script will exit with a non-zero status code.

Environment Variables and User-Based Testing

The following environment variables are used to simulate user interactions and can be adjusted as needed:

- `CHAT_RELATED_QUESTION`: e.g., "What Ayurvedic remedies help with digestion issues?"
- `CHAT_UNRELATED_QUESTION`: e.g., "What is the latest iPhone model?"
- `DOSHA_QUIZ_SELECTIONS`: e.g., "thin,dry,dry,variable,irregular,warm,light,anxious,fast,hyperactive"
- `WEATHER_CITY`: e.g., "Mumbai"
- `WEATHER_COUNTRY`: e.g., "India"
- `FOOD_RECOMMENDATION_QUERY`: e.g., "Recommend Ayurvedic foods for pitta dosha"

These variables are primarily set in the `run_tests.sh` script but can be overridden in your environment if necessary.

Troubleshooting

- **Test Failures:** Ensure that all dependencies are installed and that you are running the correct versions of Node.js, npm, and Python.
- **Environment Variables:** Verify that the environment variables are correctly set before running the tests. You can echo these variables to ensure they have the expected values.
- **Port Conflicts:** Make sure that no other application is using the ports required by the backend (typically 8080) or the frontend (typically 3000).

- **Cypress Issues:** If Cypress tests fail due to element selectors, verify that the data-cy attributes in your components match those referenced in `cypress/e2e/user_flows.cy.js`.

For further assistance, please refer to the README.md and RUN.md files for additional context about project setup.

Additional Notes

- The testing strategy ensures that all key features of the Ayurveda project (chat, dosha quiz, weather feature, and food recommendation) are properly validated.
- When contributing new features, please add and update tests accordingly to maintain high code quality.

Happy Testing!