Homework exercise

1. To effectively test the new microservice developed by the backend team in our GUI-SERVER system, I would ask for the following information:

Relevant Endpoint URLs: This information will be necessary to send requests and validate responses.

Http request methods: Understand which HTTP request methods you will use for each endpoint, such as GET, POST, PUT, DELETE, Patch etc.

Request Payloads: Realize which structure and format of the request payloads required for each endpoint.

Error Handling: Ask about the expected error responses and their formats. Identify the error codes, error messages, and any additional information provided by the microservice when errors occur.

Authorization server: The layer that separates the client and the resource owner. This server provides the access token to the client or the user-agent through

which the client or the user-agent gains access to the resources on the server.

Request Headers: Request headers play a major role in making sure that API works as per the technical and functional aspects.

Testing Environment:. It's important to have a separate testing environment that copies the production environment’s data, allowing you to test without impacting real user data or services.

.

1. Send a request to an invalid endpoint and validate that the response returns an appropriate error code (404 Not Found).
2. Verify that a GET request to the specified endpoint returns the expected data.

Ensure the response contains the correct status code and data format

1. Test the API by omitting required fields in the request payload and verify that the service returns the expected error response indicating the missing fields.
2. Send a request with an incorrect payload format and confirm that the API responds with the appropriate error code and error message.
3. Test the API's create endpoint by sending a request to create new data and validate that the response indicates successful creation.
4. Verify that the API can successfully update existing data by sending a request with the updated information and confirming that the response reflects the changes.
5. Verify that the Authorization server properly authenticates valid credentials and returns the expected access token or authorization code.
6. Verify that the API correctly responds with the requested response format specified in the Accept header, such as JSON or XML.
7. Confirm the availability of a separate testing environment that mirrors the production setup, ensuring that testing activities do not affect real user data or services.
8. Verify that the testing environment is properly isolated from the production environment, ensuring there is no data or functionality overlap.