**Pact Fundamentals Notes**

Contract testing mainly focuses on:

* Verifying the interactions between separate services, ensuring that they adhere to a predefined agreement
* Validating that a service provider meets the expectations of its consumers
* Enabling independent development by confirming that services communicate correctly, even as they evolve

From a validation perspective, checking:

* The JSON hierarchical structure e.g. objects within objects, and arrays of objects
* Each field contains the expected data type
* Business rules that were enforced such as the minimum and maximum values, or a string must match a specific pattern

The difference between **Consumer-Driven Tests** and **Provider Contract Tests**:

**Consumer-Driven Tests**

**Focus**: Ensure the consumer generates the correct expectations (requests and responses) for the provider.

* **Output**: Produces a **Pact file** that acts as a contract between the consumer and the provider.
* **Where It Runs**: On the **consumer service**.
* **Example**: A test checks that GET /getCourseName/Appium (request by the consumer) expects {name: "appium"} as the response.

**Provider Contract Tests**

* **Focus**: Ensure the provider fulfills the contract defined by the consumer in the Pact file.
* **Input**: Uses the Pact file generated by the consumer.
* **Where It Runs**: On the **provider service**.
* **Example**: Validates that the provider's GET /getCourseName/Appium endpoint returns {name: "appium"} as per the consumer’s expectations.

**Key Difference:**

* **CDTs**: Validate the consumer's expectations and generate contracts.
* **Provider Tests**: Validate the provider against the consumer’s contract.

The provided code shows **provider contract testing**, verifying that the "CoursesCatalogue" provider complies with Pact file expectations using @State and @PactFolder.