### \*\*Chapter 8: Testing and Debugging Your Cashfree Payment Integration\*\*

In this chapter, we’ll dive into testing and debugging your Cashfree payment integration. Testing ensures that the payment flows work smoothly in various scenarios before going live. We will cover setting up a sandbox environment, simulating different payment statuses, and common debugging strategies.

---

#### \*\*Topic 1: Using the Cashfree Sandbox Environment\*\*

Cashfree provides a \*\*sandbox environment\*\* for testing your integration without making real transactions. This allows you to safely test all payment flows and error handling.

---

##### \*\*Step 1: Setting Up Sandbox Credentials\*\*

Cashfree provides separate credentials for the sandbox environment. Make sure you are using the \*\*sandbox API keys\*\* and base URL.

1. \*\*Add Sandbox Credentials:\*\*

Configure the sandbox environment in your `application.properties` or `application.yml`:

```properties

cashfree.sandbox.appId=YOUR\_SANDBOX\_APP\_ID

cashfree.sandbox.secretKey=YOUR\_SANDBOX\_SECRET\_KEY

cashfree.sandbox.baseUrl=https://test.cashfree.com

```

And load them in your configuration class:

```java

@Value("${cashfree.sandbox.appId}")

private String sandboxAppId;

@Value("${cashfree.sandbox.secretKey}")

private String sandboxSecretKey;

@Value("${cashfree.sandbox.baseUrl}")

private String sandboxBaseUrl;

```

2. \*\*Switch Between Sandbox and Production:\*\*

Use a property to switch between the sandbox and production environment, based on whether the application is running in test mode or live.

```java

public String getBaseUrl(boolean isSandbox) {

return isSandbox ? sandboxBaseUrl : productionBaseUrl;

}

```

##### \*\*Step 2: Generate Payment Tokens in Sandbox\*\*

When generating payment tokens in the sandbox, you can test both successful and failed payment scenarios.

- Successful payment response from sandbox:

```json

{

"status": "OK",

"message": "Token generated",

"cftoken": "test\_payment\_token"

}

```

- Failed payment simulation (e.g., insufficient funds):

```json

{

"status": "ERROR",

"message": "Payment failed due to insufficient funds",

"code": "402"

}

```

---

#### \*\*Topic 2: Simulating Payment Flows\*\*

You need to test various payment statuses and scenarios to ensure your application handles all possible outcomes, including success, failure, and refunds.

---

##### \*\*Step 1: Simulating Different Payment Statuses\*\*

In the sandbox, you can simulate different payment statuses such as:

- \*\*SUCCESS\*\*: Payment was successful.

- \*\*FAILED\*\*: Payment failed.

- \*\*PENDING\*\*: Payment is still pending and will be updated later.

1. \*\*Handling Payment Status in Webhooks:\*\*

Test how your application processes different payment statuses using webhook events. Make sure your webhook handler can process various scenarios.

```java

@PostMapping("/webhook")

public ResponseEntity<String> handleWebhook(@RequestBody Map<String, String> payload) {

String orderStatus = payload.get("txStatus");

switch (orderStatus) {

case "SUCCESS":

// Handle successful payment

break;

case "FAILED":

// Handle failed payment

break;

case "PENDING":

// Handle pending payment

break;

default:

// Unknown status

return ResponseEntity.status(HttpStatus.BAD\_REQUEST).body("Unknown status");

}

return ResponseEntity.ok("Webhook processed");

}

```

2. \*\*Simulating in Sandbox:\*\*

When using the Cashfree sandbox, simulate success or failure by using specific test cards or payment methods provided by Cashfree. This lets you verify how your application behaves under different payment conditions.

##### \*\*Step 2: Test Card and UPI IDs\*\*

Cashfree provides test card numbers and UPI IDs for sandbox testing.

- \*\*Sample Test Card:\*\*

| Card Type | Card Number | CVV | Expiry Date |

|------------------|--------------------|------|---------------|

| VISA (Success) | 4111 1111 1111 1111 | 123 | 12/26 |

| VISA (Failure) | 5105 1051 0510 5100 | 123 | 12/26 |

- \*\*Sample Test UPI IDs:\*\*

- UPI Success: `testsuccess@upi`

- UPI Failure: `testfail@upi`

These allow you to simulate both successful and failed payments when testing your integration.

---

#### \*\*Topic 3: Debugging Common Issues\*\*

Payment integrations can involve multiple error sources, such as API failures, network issues, or configuration problems. Below are some common issues you might face and how to debug them.

---

##### \*\*Step 1: Debugging Failed Payment Token Generation\*\*

Sometimes, the payment token generation request might fail due to incorrect configuration or API issues.

- \*\*Check API Response:\*\*

Log the response from Cashfree's API to see if there are any errors or failure messages.

```java

if (response.getStatusCode() != HttpStatus.OK) {

System.out.println("Failed to generate payment token: " + response.getBody());

}

```

- \*\*Validate Credentials:\*\*

Ensure that the correct app ID and secret key are being used and are valid for the environment (sandbox vs production).

##### \*\*Step 2: Handling Webhook Failures\*\*

Webhooks might not be received due to various reasons, such as network errors, incorrect webhook URLs, or signature verification failures.

1. \*\*Webhook Signature Verification Failure:\*\*

- Make sure that the webhook signature is being properly verified using the HMAC SHA256 method.

- Log the received signature and payload to help diagnose signature mismatch issues.

2. \*\*Webhook Not Being Triggered:\*\*

- Ensure that the webhook URL is publicly accessible and uses HTTPS.

- Check if the webhook URL is correctly configured in the Cashfree dashboard.

3. \*\*Handling Timeouts:\*\*

- Webhooks should be processed within a reasonable time frame. Make sure your webhook handling logic is efficient to avoid timeouts.

---

##### \*\*Step 3: Debugging Payment Status Issues\*\*

Sometimes, payment status may not be updated properly, or the payment may remain pending for a long time.

- \*\*Check Payment Status API:\*\*

Use Cashfree's payment status API to manually check the status of a payment if the webhook was missed or if the status is not updated properly.

```java

public String checkPaymentStatus(String orderId) {

String url = cashfreeConfig.getBaseUrl() + "/api/v1/order/info/status?orderId=" + orderId;

HttpHeaders headers = new HttpHeaders();

headers.set("x-client-id", cashfreeConfig.getAppId());

headers.set("x-client-secret", cashfreeConfig.getSecretKey());

HttpEntity<String> request = new HttpEntity<>(headers);

ResponseEntity<Map> response = restTemplate.exchange(url, HttpMethod.GET, request, Map.class);

if (response.getStatusCode() == HttpStatus.OK) {

return response.getBody().get("txStatus").toString();

} else {

throw new RuntimeException("Failed to fetch payment status");

}

}

```

- \*\*Handling Pending Payments:\*\*

Some payment methods may take time to confirm (e.g., Net Banking). Ensure that your application can handle pending payments and update the status later when the payment is confirmed.

---

#### \*\*Topic 4: Logging and Monitoring\*\*

To debug and maintain the health of your payment system, ensure you have proper logging and monitoring in place.

---

##### \*\*Step 1: Enable Detailed Logging\*\*

Enable detailed logging in your application to track the flow of requests and responses to/from Cashfree. This will help in diagnosing issues when something goes wrong.

1. \*\*Log API Requests and Responses:\*\*

Log both outgoing requests and incoming responses from Cashfree's API for easier debugging.

```java

System.out.println("Request Payload: " + requestBody);

System.out.println("Response from Cashfree: " + response.getBody());

```

2. \*\*Log Webhook Payloads:\*\*

Log the payload received from the webhook to keep a record of all payment status updates.

```java

@PostMapping("/webhook")

public ResponseEntity<String> handleWebhook(@RequestBody Map<String, String> payload) {

System.out.println("Received webhook: " + payload);

// Process webhook

}

```

##### \*\*Step 2: Set Up Monitoring\*\*

Monitor your payment system in production to ensure smooth operation.

- Use tools like \*\*Prometheus\*\* and \*\*Grafana\*\* to monitor API response times, success rates, and error rates.

- Set up alerts for critical failures, such as payment token generation failure or webhook processing issues.

---

#### \*\*What We’ve Covered So Far:\*\*

1. \*\*Sandbox Environment:\*\* How to test your integration using Cashfree’s sandbox environment.

2. \*\*Simulating Payment Flows:\*\* Testing different payment scenarios such as success, failure, and pending statuses.

3. \*\*Debugging Common Issues:\*\* How to troubleshoot common issues like failed payment token generation and webhook failures.

4. \*\*Logging and Monitoring:\*\* Using logs and monitoring tools to keep track of your payment system.

---

\*\*Next Step:\*\* In the \*\*Final Chapter\*\*, we’ll go over \*\*going live\*\* with your Cashfree integration, which includes moving from sandbox to production and ensuring compliance with Cashfree's guidelines for a successful launch.

Let me know if you’re ready to move on!