Hands-On Lab: Dynatrace Synthetic Monitoring - Browser Monitor & Waterfall Analysis

---

Lab Setup Prerequisites

- Dynatrace SaaS/Managed account or access to the **Dynatrace Demo Environment**.
- Admin or monitoring privileges to create synthetic monitors.
- A sample public or internal web URL for testing.

---

1. Accessing the Dynatrace Demo Environment

Objective: Learn how to navigate and access the pre-configured Dynatrace environment for lab practice.

Steps:

1. Go to: https://www.dynatrace.com
2. Click **Login** > Select **Dynatrace Demo Environment** (if available) or log in to your environment.
3. Explore the **Demo Smartscape**, **Sample Applications**, and **Dashboards**.
4. Familiarize yourself with:
    - **Applications**
    - **Infrastructure**
    - **Transactions & Services**
    - **Synthetic Monitoring**

Tip: Demo environments often have synthetic data to explore dashboards and analysis workflows without affecting real services.

---

2. Exploring the Dynatrace UI and Dashboard Navigation

Objective: Get comfortable navigating Dynatrace's web UI and finding key observability data.

Key UI Areas:

- **Global Navigation Menu (left sidebar)**: Access dashboards, Smartscape, services, apps, etc.
- **Top Navigation Bar**: Use for global search, environment settings, and management zones.
- **Dashboards**:
    - Navigate to **Dashboards > My Dashboards**
    - Create a **new custom dashboard**
    - Add tiles for synthetic monitors, availability, or response time

Exercise:

- Use the top search to locate "easyTravel".
- Find a web application and click into its dashboard.
- View **Top Web Requests**, **User Sessions**, and **Performance Metrics**.

---

3. Creating a Single-URL Browser Monitor

Objective: Create a basic synthetic monitor to test the uptime and load performance of a webpage.

Steps:

1. Navigate to: **Synthetic Monitoring > Create a Synthetic Monitor**
2. Select **Browser Monitor > Single-URL browser monitor**
3. Fill in the required fields:
    - **Monitor name**: Homepage Uptime Test
    - **URL**: https://www.example.com
    - **Frequency**: Every 5 minutes
    - **Locations**: Select 2-3 public locations
    - **Availability**: Enable alerting on downtime
4. Click **Create monitor**

Note: This will simulate a user visiting the page in a real browser (Chrome headless) and capture full loading metrics.

---

4. Analyzing the Waterfall Data and Identifying Load Times

Objective: Understand how to read and interpret **waterfall charts** to optimize page performance.

Steps:

1. Go to the newly created browser monitor.
2. Open **Execution Details** of any recent run.
3. Click on **Waterfall analysis**.

Waterfall Chart Breakdown:

- Displays **timeline of all resources loaded** (CSS, JS, images, etc.).
- Shows:
    - **DNS Resolution Time**
    - **Connection Time**

    - **SSL Handshake**
    - **TTFB (Time to First Byte)**
    - **Download Time**
- Metrics to focus on:
    - **Visually complete**
    - **DOM interactive**
    - **Total page load time**

Exercise:

- Identify the **slowest resource**.
- Check if it's blocked, redirected, or failing.
- Hover on timing bars to get a breakdown of each phase.

---

1. Configuring Multiple Synthetic Monitors (Browser + HTTP)

A. Creating a Browser Monitor (Single-URL)

Objective: Simulate real user interaction and page load metrics.

Steps:

1. Navigate to: Synthetic Monitoring > Create a Synthetic Monitor
2. Select: **Browser Monitor > Single-URL**
3. Configure:
    - **Monitor Name:** Browser_Uptime_Homepage
    - **URL:** https://example.com
    - **Frequency:** Every 5 mins
    - **Locations:** Choose 2+ locations
    - **Options:** Enable visual complete, TTFB, etc.
4. Click **Create Monitor**

B. Creating an HTTP Monitor (API/Endpoint)

Objective: Monitor REST API/HTTP services for availability and response validation.

Steps:

1. Go to: Synthetic Monitoring > Create a Synthetic Monitor
2. Select: **HTTP Monitor**

3. Configuration:
    - **Monitor Name:** HTTP_Status_API
    - **URL:** https://api.example.com/health
    - **Method:** GET
    - **Frequency:** Every 5 mins
    - **Validation Rules:**
        - Status code = 200
        - Response body contains keyword (optional)
4. Choose locations and create monitor

---

2. Setting Thresholds and Alerts

Objective: Configure performance baselines and enable proactive alerts.

For Browser Monitor

1. Open the monitor > Go to **Thresholds**
2. Set:
    - **Visually complete time:** Warn if > 3s, Fail if > 5s
    - **Availability threshold:** Alert if > 1 failure in 10 runs

For HTTP Monitor

1. Under **Validation**, set:
    - Expected **Status Code** (200)
    - Expected response **time threshold** (e.g., 2s)
2. Navigate to **Anomaly Detection > Custom events for alerting**
3. Create alert for:
    - Failure rate increase
    - Response time degradation

Notification Setup (Optional)

- Navigate to **Settings > Alerting > Problem Notifications**
- Add email, Slack, webhook, or service integrations (e.g., OpsGenie, PagerDuty)

---

3. Testing for SSL Certificate Validity and Uptime

Objective: Ensure SSL certificates are valid and notify before expiry.

Enable SSL Certificate Monitoring:

1. Go to your **HTTP/Browser Monitor**
2. Under **Advanced options** (in HTTP monitor), enable:
    - **Check SSL certificate validity**
    - Set alert if certificate expires in **< 30 days**
3. Dynatrace will auto-check:
    - **Expiration date**
    - **Common name (CN) match**
    - **Trusted issuer**

Tip: You can also view SSL certificate details in the HTTP response headers tab of the monitor run.

---

4. Analyzing Monitoring Results and Identifying Failures

Objective: Troubleshoot failed runs and identify root causes.

Steps:

1. Navigate to: Synthetic > Monitor list
2. Select a monitor > View **Recent executions**
3. Click on a **Failed run**:
    - Check **Waterfall View** (for Browser monitor)
    - View **HTTP response codes** and **headers** (for HTTP monitor)
4. Use the **Failure Reason** to:
    - Identify DNS issues, timeouts, SSL failures, or broken links
    - Check response body content mismatches or latency spikes

Use:

- **Reports** for trend analysis over time
- **Dashboards** to visualize failure patterns and monitor health

---

1. Building a Multi-Step Clickpath Monitor

Objective: Create a synthetic monitor that mimics real-user navigation across multiple pages.

Steps:

1. Navigate to Synthetic Monitoring > Create a Synthetic Monitor
2. Select **Browser Monitor > Clickpath browser monitor**

3. Click **Record Clickpath** – a recorder window will open.

Recording a Sample Journey:

1. **Navigate** to: https://demo.easytravel.com
2. Click on **Login**
3. Enter username and password (demo/demo123)
4. Click **Sign In**
5. Navigate to a **product page** or **booking section**
6. Click **Book Now** or submit a form
7. Click **Finish Recording** and save

Tip: Keep monitor names like Clickpath_UserJourney_Login_Book

---

2. Simulating a User Journey with Form Submissions

Objective: Simulate actual interactions like search, form input, and submission.

Example Use Case:

- A user logs in → Searches for a product → Submits a form

Dynatrace Actions to Record:

- **Input Field Entry** (username, password, search term)
- **Button Clicks**
- **Dropdown Selections**
- **Form Submission**

Parameters:

- You can **inject variables** (e.g., search terms) for dynamic testing.

---

3. Adding Validations and Assertions

Objective: Ensure the monitored flow returns correct content and status.

Steps to Add Assertions:

After finishing the recording:

1. Click **Edit** on any step
2. Add **Validations**:
    - **Text Validation**: Ensure the page contains expected text like “Welcome back” or “Booking confirmed”

- **Element Exists**: Assert presence of DOM elements like buttons or messages
- **Status Code**: Expect 200 OK on each step

3. Set **failure conditions** if validation fails

Tip: Add validation **after form submission** to ensure success message or page is reached

---

4. Analyzing Test Results and Fixing Errors

Objective: Understand execution results, debug failures, and improve test stability.

Steps:

1. Navigate to: Synthetic > Monitor list > Clickpath Monitor
2. View **Recent Executions**
3. Click on **Failed execution**
    - View **Screenshots** at each step
    - Review **Console logs** and **Network timings**
    - Check for:
        - Missing elements
        - Timeouts
        - Redirect failures
        - Validation mismatches
4. Re-record or **edit steps** to improve reliability (e.g., increase wait time, reselect elements)

Debugging Tips:

Issue	Resolution
Element not found	Use Wait for element, reselect with precise CSS path
Timeout	Increase default wait time or add manual wait
Login fails	Validate credentials or simulate via API
Step skipped	Ensure clicks/forms are not dependent on animations

---

Advanced HTTP Monitor Setup & API Testing in Dynatrace

Lab Objective

By the end of this lab, you will:

- Create and execute an HTTP monitor in Dynatrace
- Chain multiple API calls to simulate real scenarios
- Use authentication tokens in headers
- Validate the response payload and monitor the response time

---

1. Creating an HTTP Monitor

1. **Select Monitor Type:**
    - Choose **"HTTP Monitor"** as the type.
2. **Define Basic Settings:**
    - **Name:** API_HealthCheck_Monitor
    - **Type:** HTTP
    - **Interval:** Every 5 minutes
    - **Locations:** Select synthetic locations (e.g., Frankfurt, Mumbai, Singapore)
3. **Add the first request (e.g., login):**
    - **Method:** POST
    - **URL:** https://example.com/api/login
    - **Headers:**

pgsql

CopyEdit

Content-Type: application/json

    - **Body:**

json

CopyEdit

{

"username": "user01",

"password": "mypassword"

}

---

2. Performing API Testing with Chained Requests

1. **Add the second request (e.g., Fetch user data):**
    - **Method:** GET

- **URL:** https://example.com/api/user/profile
- **Add Header:**
  - Authorization: Bearer {{login.response.body.token}}
- This uses a **chained variable** from the first request's response.

2. **Configure Variable Extraction (Chaining Logic):**
   - On the login request, define:
     - **Variable Name:** token
     - **Extraction Type:** JSON path
     - **JSON path:** $.token
   - This extracts the token dynamically and passes it to the next request.

---

3. Using Authentication Tokens

- Ensure that you store the **token from the login API** as a variable.
- Pass it using:

css

CopyEdit

Header: Authorization: Bearer {{token}}

This simulates **real-world API usage** where every secure endpoint is token-protected.

---

4. Verifying Response Payloads and Response Times

1. **Add Validations:**
   - Navigate to **Validations > Response content match**
   - Define:
     - Check if the response **contains** specific keywords (e.g., "user_id": or "status":"success")
     - Optionally, check **HTTP status code = 200**
2. **Add Performance Validation:**
   - Set maximum **response time threshold**, e.g., 2 seconds.
   - If the response time exceeds this, Dynatrace will raise an alert.

Result Monitoring and Debugging

- Open the monitor after deployment and view:
    - **Execution timeline**
    - **Step-by-step breakdown**
    - **Captured token values**
    - **Validation results**
    - **Failure reasons if any**
- You can also **analyze response headers, bodies, and error messages** in detail.

Troubleshooting Synthetic Monitor Failures in Dynatrace

This lab will walk you through the steps to **troubleshoot synthetic monitor failures**, perform **root cause analysis using Dynatrace Davis AI**, and **leverage logs to detect third-party issues**.

Lab Setup and Access

- Log in to the **Dynatrace SaaS portal**
- Navigate to Synthetic > Monitor Overview
- Select a **failed HTTP/Browser monitor**

---

1. Troubleshooting Synthetic Monitor Failures

Step-by-Step:

1. **Open the Failing Monitor**
    - Go to: Synthetic > Monitors
    - Filter for **Failed status**
    - Click on any recent failed execution
2. **Analyze Execution Breakdown**
    - Check:
        - **Which step failed?** (Login step, API call, redirection)
        - **Error type:** Timeout, DNS error, SSL certificate issue, unexpected response, etc.
        - **HTTP Status Code** (e.g., 500, 403, 404)
3. **Waterfall & Request Details**
    - For browser monitors:
        - Analyze **waterfall visualization** for slow-loading assets or broken links

- For HTTP monitors:
    - Review **headers, payloads, and response times**
    - Identify missing tokens, failed auth, or incorrect status codes

2. Performing Root Cause Analysis with Dynatrace Davis AI

Dynatrace Davis uses **AI-driven root cause analysis** to automatically correlate the synthetic failure with backend issues.

Step-by-Step:

1. **Access the Problem Card:**
    - Navigate to Problems > Open Problems
    - Locate the **problem related to your synthetic monitor**
2. **Review Davis AI Insights:**
    - Check:
        - **Root cause entity:** Is it a backend service? Host? Third-party call?
        - **Timeline view:** When did the anomaly begin?
        - **Impact analysis:** What services/users are affected?
3. **Dependency Mapping:**
    - Use **Smartscape** to identify affected downstream or upstream services
    - Davis AI might flag issues like:
        - Increased failure rate on backend service
        - Slowness in database response
        - Failed external HTTP request (e.g., payment API)

3. Using Logs to Identify Third-Party Issues

Logs can help identify **external dependency failures**, especially when synthetic monitors fail intermittently.

Step-by-Step:

1. **Navigate to Logs:**
    - Go to Logs and Events
    - Filter logs for affected **services or hosts** at the time of failure
2. **Search Log Patterns:**
    - Use keywords like:

bash

CopyEdit

error OR fail OR timeout OR exception

    - Filter for relevant timestamps and monitor execution IDs

3. **Correlate with Third-Party APIs:**
    - Identify patterns such as:
        - ConnectionRefused or TimeoutException to external URLs
        - SSLHandshakeFailed due to expired 3rd party cert
        - 403/429 errors from rate-limited APIs
4. **Visual Correlation:**
    - Open Distributed traces if enabled
    - Drill down into PurePath associated with the synthetic test
    - Locate the external call span and response status

Private Synthetic Location Deployment, Alerts & Reporting in Dynatrace

---

Lab Objective

This lab covers:

- Setting up a **Private Synthetic Location (PSL)** using **ActiveGate**
- Configuring **custom alert rules** and **thresholds**
- Generating and exporting **monitoring reports** for stakeholder visibility

---

Pre-Requisites

- Access to Dynatrace environment with admin rights
- A VM or server (Linux/Windows) to host the **Environment ActiveGate**
- Internet access or proxy for ActiveGate to connect to Dynatrace
- Docker (for containerized PSL agents, if needed)

---

1. Deploying a Private Synthetic Location

Step 1: Install Environment ActiveGate

1. Navigate to:
   Settings > Deployment Status > ActiveGates
2. Download the installer for your OS (Linux/Windows)
3. Run the installer with admin privileges on your target machine

Linux example:

bash

CopyEdit

sudo ./Dynatrace-ActiveGate-Linux.sh

4. Verify ActiveGate status:
   - Go to Deployment Status > ActiveGates
   - Confirm status is "Connected"

Step 2: Configure the Private Synthetic Location

1. Go to:
   Settings > Synthetic monitoring > Private synthetic locations
2. Click **"Set up a new private location"**
3. Give it a **name** and assign the newly installed **ActiveGate**
4. (Optional) Enable **Docker**-based execution for HTTP/Browser monitors
5. Save and test the connection

Now you can assign synthetic monitors to this private location

2. Configuring Custom Alerts and Thresholds

Step 1: Set Thresholds in a Synthetic Monitor

1. Go to:
   Synthetic > Monitors
2. Select a monitor (e.g., HTTP, Browser, or Clickpath)
3. Edit → **"Failure criteria"**
4. Add thresholds:
   - Response time (e.g., > 5s)
   - HTTP status code validation
   - SSL certificate expiration

Step 2: Create Custom Alert Profiles

1. Navigate to:
   Settings > Alerting > Alerting profiles
2. Click **“Add alerting profile”**
3. Define:
    - Affected entity (e.g., Synthetic Monitors)
    - Problem severity
    - Time windows for alerting

Step 3: Setup Notification Integration (e.g., Email, Slack)

1. Go to:
   Settings > Integration > Problem notifications
2. Choose an integration (e.g., Email, Teams, Slack)
3. Set filters to trigger only for synthetic-related alerts

---

3. Generating and Exporting Monitoring Reports

Option 1: PDF Reports from Dashboards

1. Go to Dashboards
2. Create a dashboard or open an existing one with:
    - Synthetic monitor charts
    - SLA summaries
    - Uptime heatmaps
3. Click **“Export > PDF”**
4. Schedule it to be sent via email:
    - Export > Schedule PDF report
    - Choose recipients and frequency (daily/weekly)

Option 2: Custom Data Export via API

1. Use the Dynatrace **Metrics API** to pull data:
    - Endpoint:

bash

CopyEdit

GET /api/v2/metrics/query

    - Example for synthetic availability metric:

bash

CopyEdit

```
curl -X GET \
'https://<tenant>/api/v2/metrics/query?metric=synthetic.browser.availability.location&resolution=Inf&from=now-1h&to=now' \
-H 'Authorization: Api-Token <token>'
```

2. Save the data as JSON/CSV and visualize in Excel or BI tools