**✅ Step-by-Step Guide to Performance Testing in ServiceNow**

**1. 🎯 Define Performance Testing Objectives**

| **Goal** | **Why It Matters** |
| --- | --- |
| Validate app response time | Ensure quick form submissions and dashboard loading |
| Test automation triggers | Verify Flow Designer flows (e.g., budget alerts) run without lag |
| Measure concurrent usage | See how the app handles multiple users entering expenses at once |
| Identify bottlenecks | Find delays in table queries, notifications, or scripts |

**2. 📊 Identify What to Test**

| **Area** | **Examples** |
| --- | --- |
| **UI Performance** | Loading speed of forms, dashboards, portal pages |
| **Backend Logic** | Flow Designer execution, script includes, business rules |
| **Data Load** | Handling large amounts of expense data (e.g., 10,000 records) |
| **Notification Delays** | Time between trigger and user alert |

**3. 🧪 Test Scenarios**

| **Test Case** | **Description** |
| --- | --- |
| **Form Load Test** | Measure time to open the expense form under different loads |
| **Record Creation Load Test** | Create 100, 1,000, or 10,000 expense records via script |
| **Budget Alert Trigger Time** | Measure time between reaching limit and alert email/SMS |
| **Concurrent Submission Test** | Simulate 10 users submitting expenses at once |
| **Monthly Report Generation Test** | Measure time to generate/report data for one month |

**4. 🛠 Tools You Can Use**

**✅ *ServiceNow Native Tools:***

* **Performance Analytics** (if licensed) – Real-time performance tracking
* **Transaction Quota Monitoring** – See system usage per user/session
* **System Logs** – Track script execution time
* **Scheduled Job Logs** – Monitor performance of background jobs

**✅ *External Tools (for load simulation)*:**

* **JMeter or LoadRunner** – Simulate user load via API
* **Postman/Newman** – Test REST APIs (if app exposes any)
* **Browser Dev Tools** – Measure UI load time and payload size

**5. 🧠 Test Data Preparation**

* Use **scripts or imports** to generate dummy data:
  + 1000+ expense records
  + 12+ budget categories
  + Multiple users (family members) using different roles
* Use GlideRecord scripts or ServiceNow **CSV imports**

**6. 📈 Metrics to Track**

| **Metric** | **Target/Threshold** |
| --- | --- |
| Form load time | < 2 seconds |
| Record submission time | < 1 second |
| Flow trigger time | < 2 seconds |
| Monthly report time | < 5 seconds |
| Concurrent user support | 10–20 users (for family-level) |

**7. 📋 Sample Test Plan Table**

| **Test Case** | **Description** | **Tool** | **Expected Result** |
| --- | --- | --- | --- |
| TC1 | Submit 100 expense entries in 1 minute | Scripted test | No errors, <1 sec per insert |
| TC2 | Trigger budget alert when limit hit | Manual + Flow Logs | Alert within 2 seconds |
| TC3 | Load dashboard with 1000 records | Browser dev tools | Loads in <3 seconds |
| TC4 | Run monthly report with 12 categories | Scheduled Job | Completes in <5 seconds |
| TC5 | Simulate 10 users submitting at once | JMeter | System handles without crash |

**8. 🧾 Report Findings**

Include:

* Summary of test cases run
* Screenshots of key metrics (load time, flow execution time)
* Bottlenecks identified (if any)
* Suggested optimizations (e.g., script improvements, indexing)

**🧩 Bonus: Common Optimization Tips in ServiceNow**

* Use **GlideAggregate** instead of looping through records for calculations
* Add **indexes** on frequently filtered fields (e.g., date, user, category)
* Optimize **Flow Designer** by reducing nested logic or frequent triggers
* Use **lazy loading** for widgets or dashboard data
* Enable **performance logging** during tests

**📄 Deliverables of Performance Testing**

* **Performance Test Plan (document or spreadsheet)**
* **Execution logs or screenshots**
* **Summary Report** with observations and recommendations