

# PERFORMANCE TESTING

DATE	31.10.2025
TEAM ID	NM2025TMID04135
PROJECT NAME	Calculating family expenses using service now
MAXIMUM MARK	4 Marks

## 1. Understanding the Scenario

You likely have a ServiceNow app or module that:

- Takes inputs like income, expenses (rent, food, transport, etc.)
- Calculates totals or summaries (monthly, yearly, etc.)
- Stores results in records or displays them on a dashboard.

Your goal is to ensure the performance (speed, scalability, reliability) of this process — especially under load.

---

## 2. Performance Testing Objectives

Typical goals:

- Measure response time for calculations (e.g., expense summary load time).
- Determine system behavior under load (e.g., 100 users calculating at once).

- Find bottlenecks in scripts, workflows, or integrations (e.g., with tables, APIs).
- Validate ServiceNow instance performance and server response consistency.

### 3. Performance Testing Types

Type	Description	Example
Load Testing	Normal expected usage	50 concurrent users calculating expenses
Stress Testing	Beyond normal limits	200 users submitting data simultaneously
Spike Testing	Sudden surge of requests	0 → 100 users in 10 seconds
Endurance Testing	Long-term stability	Continuous calculations for 2 hours
Scalability Testing	System growth capacity	Increase users gradually until slowdowns occur

Since ServiceNow is web-based and API-driven, you can use:

Tool	Use Case
ServiceNow ATF (Automated Test Framework)	Functional testing (limited for load)
JMeter	Load & stress testing of UI or APIs
LoadRunner	Enterprise-grade load testing
Postman + Newman	Simple API-based load tests
K6 or Locust	Lightweight script-based performance testing (modern choice)

## 5. Approach Example Using JMeter

Identify endpoints

Example:

POST

`https://<instance>.service-now.com/api/x_family_expense/calculate`

Payload might include:

```
{
  "income": 6000,
```

```
"rent": 1500,  
  
"food": 800,  
  
"utilities": 200  
}
```

## **1. Create JMeter Test Plan**

- **Add Thread Group (e.g., 50 users, 10 iterations).**
- **Add HTTP Request Sampler (the API endpoint).**
- **Add HTTP Header Manager (with Auth token).**
- **Add View Results Tree, Summary Report, and Aggregate Graph.**

## **2. Run the test**

### **Measure:**

- **Average response time**
- **90th percentile response**
- **Error rate**
- **Throughput**

## **3. Analyze results**

- **Identify slow scripts (Business Rules, Script Includes).**
- **Check ServiceNow logs for slow transactions.**
- **Use Performance Analytics to monitor response times.**

## 6. Key Metrics to Capture

Metric	Target Example
Response Time	< 2 seconds
Error Rate	< 1%
Throughput	20 requests/sec
CPU / Memory Usage	< 80% utilization
Database Query Time	< 300 ms per transaction

## 7. Reporting

Your performance test report should include:

- Test objectives and scope
- Test environment details (ServiceNow instance, version, hardware)
- Load model (user count, duration)
- Test results (tables + graphs)

- **Bottlenecks and recommendations**

## **8. Optimization Tips**

- **Minimize unnecessary GlideRecord queries.**
- **Use async calls where possible (GlideAjax).**
- **Avoid long-running Business Rules.**
- **Cache frequent lookups using GlideCache.**
- **Monitor with ServiceNow Performance Dashboard.**