# About Myself

I am [**My Name**]. I have been working in **software testing industry** about two plus years and **experienced** in **Manual as well as Automated testing** **of web and client/ server applications.** As a software test engineer, I was **involved in all stages of software development life cycle** in multiple projects of different company that I previously worked with and have **deeper understanding of the development processes such as waterfall and agile methodology.**

I have experience work with different operating systems such as Windows &

UNIX. I am expert in working with HP tools **HP ALM / Quality Center, HP Unified Functional Testing, formerly Quick Test Professional (QTP), LoadRunner**. I have in depth knowledge of **SQL, PL/SQL**. I have extensive experience in **Smoke, Functional, Positive, Negative, Boundary, Integration, Regression, Database, User acceptance, System, GUI, End-to-end, Performance, Stress, Volume Testing**.

I am expert in **creating documents** like **Test Plan, Test Case, Test Script, Test**

**Scenarios, Requirement Assessment Report, Test Analysis Report, Test Matrix, Requirement Traceability Matrix, Executive Summary, Level of Effort, Meeting Minute, Post Mortem Report, User Manuals** etc.

I have ability to work with **minimum documentation** and **minimum supervision**. I have worked in different software environments and can easily fit into new environments. I have ability to work **independently** and **use** **sound judgment**, with a **minimum amount of supervision**. **I am able to effectively manage multiple assignments simultaneously. I am well experienced in working with people from different background providing excellent team work and determined to become maximum productive person for the company**.

# About Last Project

**What did you do in your last project?**

In my last project (**Project Name from Your resume**), the application was a webbased application developed in Java platform also there was client/server application. As a Sr. Tester/QA Person, I wrote **Test Plans, Test Cases** from the requirement documents and **Use Cases**. I performed **Smoke Testing, Functional Testing, Backend Testing, Black Box Testing, Integration Testing, Regression Testing and UAT** (User Acceptance Testing).As a Sr. Tester/QA I have experience to set up **automation frame work environment “Starting from scratch”**, also I have participated in Performance, Load and Stress Testing by using HP Load Runner. I attended several **Triage meeting** also walkthrough meetings for requirement reviews and provided feedback to the Business Analysts. Moreover, I was involving the backend testing, which required writing SQL queries directly to the database. Besides these, I have experience to write defects using **HP ALM**, Once the defects were fixed, retested them and if the **PASS**, closed the defect. If the defect was **not passing** on QA testing then reopened the defect.

In my previous projects, I had to learn about the application by reading documentation, use cases, design documents, design mockup, and business rules. In my last project with (Project Name from Your Resume) I was involved in requirement assessment after receiving the requirements and design documents. I did the requirement assessment based on:

* Is the requirement specific, explicit, and detailed?
* Is the requirement consistent with other requirements?
* Is the requirement factually correct?
* Is the requirement objectively testable?
* Is the requirement unambiguous?
* Is the requirement testable in available test environment?
* Does the requirement define all references?
* In the requirement complete?
* Does the requirement specify an exception condition?

A good requirement should be clear, complete, detailed, cohesive, and testable and agreed to by all players.

I created requirement assessment based on SMART criterion. SMART criteria are as follow:

* Specific (precise, detailed definite)
* Measurable (quantifiable, computable)
* Attainable or Achievable (within reach, possible, reasonable)
* Realistic (practical, sensible, rational)
* Timely (appropriate, apt, well-timed, suitable)

After getting the final requirement:

* + I exported them to the requirement section of **HP ALM**.
  + Convert requirements to tests.
  + Developed test cases or design steps for tests under Test Plan section  Create templates for common steps like log in / log out.
  + Performed requirement coverage – linking between requirements and test cases (So that you could get exact report like how many requirements passed or fails)
  + Attached necessary screen shots docs with the test cases
  + Create test set on test lab section
  + Import tests in the test set
  + Execute and run tests under test lab
  + Logs or documents defects using add defects icon; Filled up all required fields like assign by, assign to, severity, priority, reproducible, version, summery, steps to reproduce the defects etc. (talk about defect life cycle)
  + Track, monitor, search, filter defects
  + Generate graphs and reports for defects, requirement coverage etc.
  + Bug Life Cycle\*

While done with the manual testing, I worked to find the right candidate for regression testing by risk analysis. Regression testing ensures that a change or fix has not caused faults to appear in unchanged parts of the system.

1. **Regression (Fix)**: Whenever defects are fixed and deployed to system test, the test scripts directly related to those defects will be run. This effort will only cover selected data paths within the application. The effort and resources will be identified by calculating the time required to run the test scripts.
2. **Regression (Risk):** During defect fixes the development team may unintentionally introduce new errors into the code. The Test team will conduct an impact and risk analysis for the defects and identify additional test scripts that should be run based on code complexity, defect density, and code priority. The impact analysis provides the number of test scripts and the level of effort to run the test scripts for the current build as well as for the previous builds.

As we all know, retesting a whole application is never possible because of budget, time, and resource constraint. So we need to select / choose the right candidates for regression testing so that there are no adverse effects in any part of the application. Sometimes bug hides within the bug. So if we do regression testing we can eliminate more bugs.

I selected the right candidates for regression testing by considering:

* Which functionality is most important to the project intended?
* Which functionality is most visible to the user?
* Which functionality has the largest safety impact?
* Which functionality has the largest financial impact on users?
* Which aspects of the application are most important to the customer?
* Which parts of the code are most complex, and thus most subject to errors?
* Which part of the application were developed in rush and panic mode?
* Which part of the requirements and design are unclear or poorly thought out?
* What do the developers think are the highest-risk aspects of the application?
* What kinds of problems would cause the most customer service complaints?
* What kinds of tests could easily cover multiple functionalities?

**Regression** testing can be done manually but if possible, **automated** test scripts are used to run regression testing. The regression test scripts are performed for each cycle within a build as well as for previous builds.

The final cycle of regression testing before production covers the test scripts that are not touched in any of the regression cycles to ensure all the test scripts are run at least twice before production deployment.

As soon as the right candidates are selected, I consolidated all requirements to multiple scenarios, and developed the scripts using, **HP Unified Functional Testing, formerly Quick Test Professional (QTP).** After developing the script, I did script enhancement by:

* Inserting **checkpoints** into the test
* **Parameterize** tests
* Creating **Output Values**
* Using **Regular expression**
* Creating **multiple actions**, and **reused functions**
* **Debugging** Scripts
* **Synchronizing** tests
* Using managing **Recovery Scenarios**, and **Optional steps**
* Creating **virtual objects**
* Configuring **Smart Identification**
* Working with **Descriptive Programming**
* Working with User defined Functions---**created function** for link so that I could find any desired link and click on it, created function for edit box so that I could find any desired edit box and enter a value on it, created function for button so that I could find any desired button and click on it. I have created a function to close all browsers before and after executing the script. I have created function to customize results in Excel sheet, attach the result file, and send the result to the team members using MS Outlook.
* Creating dictionary objects
* Using environmental variables
* Integrating **UFT/QTP with HP ALM/ Quality Center**

I was involved in **Performance testing** after I was done with **regression** testing which included both functional testing and GUI testing. I performed performance testing by using **LoadRunner** that load tests applications by emulating an environment in which multiple users work concurrently. **LoadRunner** replaces real users with **Virtual users**. I was involved picking up the business processes that we will do performance testing on. The performance testing candidates were picked for the following business processes:

* **Mission critical**
* **Heavy throughput and**
* **Dynamic content**

As soon as the business processes are picked, I used the standard performance testing life cycle, which follows the following:

* • **Gathering requirements**
* **• Creating test plan**

Once the test environment is available I started to validate the test environment, Start scripting, and script enhancement)

* **Inserting transaction points**
* **Inserting Comments**
* **Creating Checkpoints**
* **Rendezvous points**
* **Parameterize the Scripts**
* **Configuring Run time Settings**
* **Sending Output Message**
* **Correlating the scripts**
* **Creating Custom Vuser Scripts**
* **Test execution (stand-alone mode)**
* **Create scenarios using the LR Controller: both Manual and Goal oriented Scenarios.**
* **Establish Monitors to monitor the test**
* **Analyze the test**

I was also involved writing **performance test plan**. A performance test plan contains executive summary that describe the objective of the test, strategy, transactions, purpose of the test, and the scope of the test that lists the application functions to be tested. The Organization and **Responsibilities** section describe roles and responsibilities of individuals.

The **Production** **Environment** **Description** section contains the architecture and hardware specification, and expected increase in load volumes for long term and short term. **Test Environment Description** section contains environment architectures for all environments used and describes server hardware specifications. The plan describes **assumption, constraints, risks, and contingencies**.

The Test Approach section describes details on methodology, and description of each test steps, and execution. The performance test plan also lists **Monitoring Criteria** with the tools to be used, and the environment elements to be monitored. The plan describes **test entry and exit criteria**, **pass/fail criteria**, **application response time, hardware utilization, network utilization criteria**. The **Test Execution section** contains script description with run time settings. The Project Schedule section lists the major milestone dates for the project.

The **Test Cycle** sections describes the **execution cycles and specify the number of Vusers** for Baseline, **Stress, and endurance testing cycles** and the **goal of each cycles**. The plan also contains acronyms and definitions, **deliverables**, and **appendices**.