Veeva Systems
QA Engineer - Technical Screening Questionnaire

Name: Varun Vora
Date: 8th July 2024

1) Please self-evaluate the following skills sets with the level definition below:

SKILLS	LEVEL
Agile Methodologies	**5**
Restful API/JSON Functional Testing	**5**
Testing on Web/Cloud-based Application	**4**
Develop Automated Test Scripts	**4**
Testing Database with Queries	**3**

LEVEL	1	2	3	4	5
(1 -5)	(No Experience)	(Knowledge)	(can perform with help)	(can perform independently)	(can lead others to perform)

2) Please describe some of the tools/methods that you have used to troubleshooting a Web/Cloud-based application.

In my previous roles, I have utilized various tools and methods to effectively troubleshoot and enhance web/cloud-based applications, with a particular focus on testing and problem resolution. Here are some key examples:

1. **Automated Testing**:
    - **Postman and ReadyAPI**: I used these tools extensively for automating API testing, creating over 250+ test cases to ensure thorough coverage. This approach not only reduced testing time but also enhanced regression testing, quickly identifying and addressing issues before they impacted the production environment.
    - **JMeter**: For performance testing, I leveraged Apache JMeter to simulate high load conditions and identify bottlenecks. By doing so, I was able to optimize system performance, achieving a 25% improvement in responsiveness under heavy loads.
2. **Unit and Integration Testing**:

- **JUnit and Mockito**: I developed 100+ unit test cases using JUnit and Mockito, significantly increasing code coverage by 20%. This rigorous testing helped in identifying and resolving defects early in the development cycle, ensuring higher code quality and stability.

3. **Continuous Integration and Continuous Deployment (CI/CD)**:
   - **Jenkins and GitLab CI/CD**: Implemented automated CI/CD pipelines that integrated testing into the build process. This setup enabled continuous testing and immediate feedback on code changes, facilitating quick troubleshooting and resolution of any issues. The automation led to a 40% reduction in deployment times, enhancing overall operational efficiency.
4. **Scripting and Automation**:
   - **Bash and Groovy Scripts**: Developed scripts to automate various aspects of the software development and testing process. For example, I automated the migration of workloads from SGE to LSF, which improved job scheduling and resource management, and reduced manual intervention.
   - Developed and utilized Groovy scripts to automate various CI/CD tasks within Jenkins. These scripts helped streamline the build and deployment processes, allowing for more efficient and error-free operations.
5. **Cloud-Based Troubleshooting**:
   - **AWS, Azure, and GCP**: Managed and troubleshooted cloud infrastructure on AWS, Azure, and GCP. This included handling issues related to EC2 instances, Lambda functions, API Gateways, and more. By using tools like AWS CloudWatch and Azure Monitor, I ensured the smooth operation of cloud services and quickly resolved any incidents that arose.
6. **Monitoring and Logging**:
   - **Prometheus and Grafana**: Utilized these tools for real-time monitoring and visualization of application performance. By setting up detailed dashboards, I could proactively identify and troubleshoot issues such as performance degradation and resource bottlenecks.
   - **Jira and Confluence**: Maintained meticulous documentation of test results and issues using Jira and Confluence. This systematic approach ensured that all problems were tracked, analyzed, and resolved efficiently, providing a clear history of issue resolution.

These tools and methodologies have equipped me with a robust skill set for effectively troubleshooting and optimizing web/cloud-based applications, ensuring they operate smoothly and efficiently.

3) Please describe an example where a customer reports a defect in Production, but you cannot reproduce it consistently. How would you report the issue

Scenario: While working at Infosys for Dish Network, I was part of the Real Time Replenishment (RTR) team, responsible for managing credit and debit card transactions and handling premium customer cards. During our routine testing in the production environment, we discovered that account balances were not being tracked correctly, leading to occasional transaction failures. These issues were handled

through our AWS Lambda functions, Step Functions, and API Gateways. However, the issue could not be reproduced consistently.

How to Report the Issue:

1. **Initial Investigation:**
    - **Gather Details:** I started by gathering all relevant details about the transactions where the issue occurred, such as the type of cards involved, the transaction amounts, the timestamps, and any error messages received.
    - **Check Logs:** I examined the logs from AWS CloudWatch for the Lambda functions, Step Functions, and API Gateways to identify any patterns or errors coinciding with the transaction failures. This helped in identifying potential common factors.
2. **Reproduce the Issue:**
    - **Simulate Transactions:** I attempted to replicate the transactions in a controlled environment using the same types of premium customer cards and transaction amounts. Despite multiple attempts, the issue did not manifest consistently.
    - **Edge Cases:** I considered various edge cases and conditions such as network latency, server load, and concurrent transactions that might affect the outcome.
3. **Isolate Variables:**
    - **Feature Toggle:** I used feature toggles to selectively enable and disable parts of the Lambda functions and Step Functions to isolate the problematic module or functionality.
    - **Environment Comparison:** Compared the configurations between the production, testing, and development environments to identify any discrepancies that could influence the behavior of the system.
4. **Collaborate with Team:**
    - **Discuss with Team Members:** I brought the issue to the attention of my team during our daily stand-up meetings. We brainstormed potential causes and shared similar experiences that might provide insights into the problem.
    - **Expert Consultation:** Consulted with senior developers and architects to leverage their experience and insights in troubleshooting complex issues.
5. **Enhanced Monitoring and Logging:**
    - **Increase Logging:** Enhanced the logging around the account balance tracking and transaction processing code to capture more detailed information when the issue occurs. This included adding additional log statements and ensuring all exceptions were properly logged.
    - **Set Up Alerts:** Configured alerts in AWS CloudWatch to notify the team immediately if the issue reoccurred under specific conditions.
6. **Reporting the Issue:**
    - **Document Findings:** Documented all findings meticulously, including the steps taken to reproduce the issue, the conditions under which it occurred, and the results of our simulations.

- **Detailed Report:** Created a detailed issue report in our tracking system (e.g., Jira) with all the gathered information, including logs, screenshots, and any error messages. This report was assigned to the relevant team members for further investigation.
- **Continuous Monitoring:** Implemented continuous monitoring to capture additional data if the issue reoccurred, providing more context for troubleshooting.

By following these steps, I ensured a thorough investigation and transparent communication within the team. Even though the issue could not be consistently reproduced, the comprehensive reporting and continuous monitoring facilitated a more targeted approach to identifying and resolving the problem.

4) Please provide specific details of your experience with the following technologies and how you used them to develop/execute manual or automated testing.

i) Java (e.g. Eclipse, IntelliJ, etc)
ii) REST/JSON based Web Services
iii) Scripting Languages (e.g. JavaScript, Groovy, Python)
iv) Test Framework (e.g. TestNG, Selenium, JUnit, etc)
v) Continuous Integration/Continuous Testing (e.g. Jenkins, Teamcity)
vi) Relational Database

i) Experience with Java:

Eclipse IDE:

- **Synopsys**: During my internship at Synopsys, I extensively used Eclipse IDE for developing and maintaining internal software tools. My work involved handling Java projects that were managed with Gradle. I used Eclipse's features such as code completion, debugging, and project management to streamline development.
- **Infosys**: At Infosys, I also used Eclipse IDE for various tasks, including developing automated testing scripts and maintaining application code. I was involved in handling Java projects that were managed by Maven. Eclipse's robust plugin ecosystem allowed me to integrate other tools such as Jenkins seamlessly.

Key Projects:

1. **Automated Testing Scripts**:
   - Developed automated API testing scripts using Java in Eclipse.
   - Integrated these scripts with Maven for dependency management and build automation.
   - Used Eclipse's debugging tools to troubleshoot and resolve issues in the testing scripts.
2. **CI/CD Integration**:
   - Implemented CI/CD pipelines using Jenkins, with Java projects managed in Eclipse.
   - Configured Maven builds and test executions within Jenkins, leveraging Eclipse's configuration capabilities to manage and test the integration.

ii) Experience with REST/JSON based Web Services:

- At Infosys, I worked on several projects that involved integrating and testing RESTful web services. This included both internal and client-facing applications.
- Automated API testing using tools like Postman and ReadyAPI, covering over 100 test cases. This automation reduced testing time and improved regression coverage.
- Conducted performance testing using Apache JMeter to identify bottlenecks in the web services, optimizing the code to improve response times under high loads by 25%.

Key Projects:

1. **API Development and Testing:**
    - Developed RESTful APIs to handle transactions and manage customer data for premium cards at Infosys.
    - Automated the testing of these APIs to ensure they correctly processed requests and returned accurate JSON responses.
2. **Performance Optimization:**
    - Used JMeter to simulate high load conditions on APIs and identify performance issues.
    - Optimized API endpoints to handle larger volumes of requests efficiently, ensuring stable performance during peak usage times**.**

iii) Experience with Scripting Languages
Groovy:

- **Synopsys**: At Synopsys, I leveraged Groovy for developing internal software tools and automating tasks related to IP design stages. I used Groovy scripts extensively for automating software QA checks and tests against files generated from semiconductor chips. This scripting helped enhance efficiency and accelerate project delivery.
- **Infosys**: I also used Groovy for writing scripts to automate parts of the CI/CD pipeline within Jenkins. These scripts streamlined the build and deployment processes, ensuring more efficient and error-free operations.

Python:

- **University Projects and Real-Time Stock Market Data ETL Pipeline:**
    - Developed Python scripts for an ETL pipeline, which involved producing and consuming real-time stock market data from Kafka topics.
    - Utilized Python to implement scripts that interacted with the Kafka cluster, ensuring the scalability and reliability of the data pipeline through Kubernetes orchestration.
- **MLOps Comparative Analysis:**
    - Used Python, alongside libraries like TensorFlow and Keras, to implement machine learning models and automate the training, evaluation, and deployment workflows.

Key Projects:

1. **Internal Tool Development with Groovy:**
    - Created Groovy scripts to automate QA processes and improve the efficiency of software development at Synopsys. These scripts handled various tasks such as file validation, data manipulation, and workflow automation.

2. **CI/CD Automation with Groovy:**
    - Developed Groovy scripts for Jenkins to automate the CI/CD pipeline. These scripts managed build processes, ran automated tests, and handled deployments, improving overall deployment efficiency by 40%.
    - **ETL Pipeline with Python:**
    - Implemented an ETL pipeline for real-time stock market data using Python scripts to produce and consume data from Kafka topics. This project involved automating data processing tasks and ensuring the robustness of the pipeline through effective scripting.

iv) Experience with Test Frameworks

- **Synopsys**: At Synopsys, I utilized **Spock Framework** extensively for unit testing Groovy applications. This involved writing numerous unit test cases to validate the functionality of internal software tools. Spock helped ensure high code quality.
- **Infosys**:
    - At Infosys, I contributed significantly to increasing code coverage by developing **JUnit and Mockito** test cases for various components of the applications. This practice not only improved the reliability of the code but also made the applications more robust by identifying and fixing bugs early in the development cycle.
    - Used **ReadyAPI and Postman** for manual and automated API testing. This tool allowed me to create complex test scenarios for RESTful services, ensuring the APIs performed correctly under various conditions.
    - Conducted performance testing with **Apache JMeter** to identify bottlenecks and optimize the system's responsiveness
- **Selenium and Cucumber:** Although I do not have professional experience with Selenium, I have extensively studied and practiced it in academic projects. The below project given by Veeva has helped me to gain a lot of knowledge on both these technologies and. Integrating both these technologies allowed me to create readable and understandable test scenarios, ensuring comprehensive testing.

v) Experience with Continuous Integration/Continuous Testing:

- At Infosys, I collaborated with client teams to design and implement automated CI/CD pipelines using **Jenkins**. These pipelines automated the build, test, and deployment processes, resulting in a 40% reduction in deployment times.
- I also used **GitLab CI/CD** for continuous integration and deployment of applications. By building and managing Docker images through GitLab CI/CD and GitLab Container Registry, I achieved a 30% increase in deployment efficiency. This ensured consistent deployment across different environments and improved operational performance.

vi) Experience with Relational Database

I do not have professional experience working with testing relational databases. However, I have a basic understanding of it. My coursework included learning about SQL, database design, and the use of relational database management systems (RDBMS) such as MySQL and PostgreSQL.

5) Create an automation framework in Java using Cucumber and Selenium WebDriver to test a web application's search functionality (Ex: E-commerce website such as Amazon, Etsy etc.,)

The framework should include feature files, step definitions, and page objects.
Write test scenarios to cover different search scenarios such as valid/invalid search terms, search suggestions, and search results pagination
On top of the above-mentioned scenarios, please feel free to include more scenarios that you can think of to improve search functionality coverage.
Provide a report of the test results in a readable format.
Please make sure to come up with a variety of search scenarios, not just happy path

I have successfully implemented an automation framework in Java using Cucumber and Selenium WebDriver to test the search functionality of Amazon website. The framework includes feature files, step definitions, and page objects. It covers a variety of search scenarios to ensure comprehensive testing of the search functionality.

The test execution results are captured and reported in a Cucumber HTML Report. The attached report includes information on the test scenarios executed.

I am attaching a Github Link for the repository VeevaEcommerceTest
https://github.com/varunvora2402/VeevaEcommerceTest