**Introduce** yourself in English

Hi everybody, I am glad to be here for this interview.

My name is Wang Xin, my English name is Vic Wang, so you can call me Vic.

I come from Foshan, the city next to Guangzhou. I graduated from South China Agricultural University in 2008.

I have been engaged in software development and testing for over 13 years, including 5 years of automation testing, 3 years of test management, 2 years of full stack development and DevOps experience.

I am proficient in UI and API testing, included manually and automatically, and also have many experiences of mobile apps and performance test. I am familiar with many testing tools, like postman, fiddler, selenium and JMeter.

I have worked in multiple companies in roles such as a Test Engineer, Test Team Leader, and Test Development Engineer. Among them, I would like to focus on my experiences in my last company and the previous projects I have worked at HSBC.

In last company, I served as the head of automation testing for my department. I built a UI automation testing platform for the testing team. This platform was developed using Python's Django framework along with Selenium.

In HSBC, I initially started working on System API automation team in WPB. After working in that role for two years, I transfer to ET where I worked as a full-stack development engineer. In this role, I was involved in various tasks such as development, testing, deployment, evergreen, and some DevOps jobs. It allowed me to have a comprehensive understanding of the entire software development lifecycle, from coding to testing, deployment, and maintenance.

**Project Experience:**

**Introduce more about your latest project?**

Artemis is a financial risk public data search tool and Dive is a financial risk personal data search tool, both used by bank employees. They are programmed by java, BS structure and search the data from elastic search.Design the bottom-level search engine (Elasticsearch) upgrade and testing plan.

S/PAPI is the API between client and the bank’s backend system like mainframe. It implements business logic by combining multiple lower-level APIs and transforming the results.

**Can u introduce me one defect in this project?**

On the searching page, there are multiple search criteria, and each criterion has multiple options. Each criterion works fine when used individually. It is also fine to use multiple options for a single criterion or combine multiple criteria together. However, the search results do not meet the expectations when multiple criteria are combined with multiple options simultaneously.

When I captured the page's requests using Fiddler, I noticed that the query statement generated by the JavaScript code was missing some conditions compared to when it was used individually. After discussing with the development team, we discovered that this was due to the usage of some generic methods where the condition combinations were overwritten to the same variable, thus overriding the previous conditions.

**Can talk a bit detail on your responsibility project?**

**Whats the business about the system do**

**In your team, whats your roles and responsibilities as a tester**

Cooperate with developers and PO to clarify the requirements, design test scenarios and prepare test data.

Script the test code and execute, collect test reports and provide QA sign off.

Cooperate with other project team developers to check the code merge quality, provide cross entity sign off.

Cooperate with developers to conduct performance tests and review performance test reports.

**Language:**

What kind of programming language you use: Java/Python

比较关心的是框架的架构和使用到的技术栈(用的什么语言）

**Test Management/process：**

**How about other test management?**

**talk about the whole test process?**

The test process encompasses the entire lifecycle of testing activities, from test planning to test closure. Here's an overview of the typical test process:

1. Test Planning: In this initial phase, the testing objectives, scope, and strategy are defined. Test plans are created, including the identification of the testing techniques, resources, and timelines. Test requirements are analyzed, and the test environment is set up.

2. Test Design: Test cases are designed based on the requirements and specifications. Test data and test scenarios are created to cover various functional and non-functional aspects of the system. Test design techniques such as equivalence partitioning, boundary value analysis, and error guessing may be employed.

3. Test Execution: Test cases are executed based on the test plan and test design. Testers execute the test cases and record the actual results. Defects and issues encountered during testing are logged in a defect tracking system. Test execution can be performed manually or through automated test scripts.

4. Defect Reporting and Management: Defects identified during test execution are reported in a defect tracking system. Each defect is documented with information such as steps to reproduce, severity, priority, and assigned to the responsible team member. Defects are tracked, managed, and retested after fixes.

5. Test Reporting and Metrics: Test progress and results are documented in test reports. These reports provide a summary of test activities, test coverage, defects found, and overall system quality. Test metrics such as test coverage, defect density, and test effectiveness may be calculated to assess the quality of testing efforts.

6. Test Completion and Closure: When the testing objectives have been met, and the exit criteria defined in the test plan are satisfied, the test cycle is considered complete. A formal closure report is prepared, summarizing the testing activities, achievements, challenges, and lessons learned. Test artifacts, such as test cases, test data, and test scripts, may be archived for future reference.

7. Retesting and Regression Testing: After defects are fixed, retesting is performed to ensure that the reported issues have been resolved. Regression testing is conducted to ensure that fixes or changes have not introduced new defects or impacted existing functionality.

Throughout the test process, communication and collaboration among team members, including developers, testers, and stakeholders, play a crucial role. Testers work closely with developers to ensure defects are understood and resolved. Ongoing feedback and continuous improvement are encouraged to enhance the effectiveness and efficiency of the testing process.

怎么安排测试任务？

1. 确定测试目标和范围：首先，明确测试的目标和范围。了解系统的需求和预期功能，以及测试的关注点和约束条件。
2. 识别测试需求和功能：基于系统需求和功能规格，识别需要进行测试的各个模块、功能和特性。将系统的不同方面划分为独立的测试需求。
3. 确定测试优先级：评估测试需求的重要性和紧急程度，以确定测试的优先级。优先级可以基于业务价值、风险等因素进行确定。
4. 分配测试资源：根据测试需求和优先级，确定测试所需的资源，包括测试人员、测试环境、测试数据等。确保每个测试任务都有适当的资源支持。
5. 制定测试计划：根据测试目标、范围、优先级和可用资源，制定测试计划。确定测试活动的时间表、里程碑和关键任务。
6. 分配测试任务：将测试任务分配给测试团队的成员。考虑每个团队成员的专业领域、技能和可用时间，合理分配测试任务。
7. 设定时间和截止日期：为每个测试任务设定适当的时间估计，并设定截止日期。确保测试任务的时间安排合理，并与整体项目计划协调一致。
8. 建立任务跟踪和监控机制：跟踪测试任务的进度和状态，并及时采取措施解决任何延迟或问题。使用任务管理工具或跟踪系统来监控任务的执行。
9. 持续评估和调整：根据测试任务的进展和新的信息，持续评估测试任务的优先级和资源分配。根据需要进行调整，确保测试任务的有效执行。
10. 沟通和协作：测试任务的安排需要与项目团队和利益相关者进行充分的沟通和协作。确保测试任务的目标、进展和需求清晰地传达给相关方，并及时解决任何问题或疑虑。

defect的测试流程?

测defect，回归相关模块，视影响范围决定是否需要回归整个项目，入不需要可在发版或合并代码前再回归项目

你有独立测过一个项目吗 ? 会有时紧迫的情况吗 ？

**Test case**

How do you implement your test cases(testing)?

What a steps for a typical a test case would you?

一个典型的测试用例的步骤是什么？

1. Understand the test objective: Understand the purpose and goal of the test case.

了解测试目标：了解测试用例的目的和目标。

2. Identify test inputs: Determine the specific inputs or data that will be used for the test case.

确定测试输入：确定将用于测试用例的具体输入或数据。

3. Define expected outcomes: Clearly define the expected results or outcomes that should be observed when executing the test case.

定义预期结果：明确定义在执行测试用例时应观察到的预期结果或输出。

4. Set up preconditions: Establish any necessary preconditions or prerequisites that need to be in place before executing the test case.

设置前置条件：在执行测试用例之前，建立任何必要的前置条件或先决条件。

5. Execute the test case: Run the test case by following the defined steps and using the identified inputs.

执行测试用例：按照定义的步骤和使用确定的输入运行测试用例。

6. Compare actual and expected outcomes: Compare the actual results obtained from executing the test case with the expected outcomes defined earlier.

比较实际和预期结果：将从执行测试用例中获得的实际结果与之前定义的预期结果进行比较。

7. Analyze and report discrepancies: Identify any discrepancies or differences between the actual and expected outcomes, and report them for further investigation.

分析和报告差异：识别实际结果与预期结果之间的任何差异，并报告它们以进行进一步的调查。

8. Clean up and restore system state: If any changes were made during the test case execution, restore the system to its original state or clean up any temporary data.

清理和恢复系统状态：如果在测试用例执行过程中进行了任何更改，请将系统恢复到其原始状态或清理任何临时数据。

9. Retest if necessary: If discrepancies or issues were found, fix them

**UI/API test**

**API**

You just mention that you have experience UI automation testing and API testing. Which part do you do? mainly UI or API?

How about the API automation testing?How did you do it?

Would you identify what kind of cases using API?

选择API测试还是UI测试取决于你想要测试的具体方面和目标。下面是一些指导原则：

**选择API测试的情况**：

1集成测试：当你需要验证不同系统或组件之间的接口交互和数据传递时，API测试非常有用。它可以确保各个系统之间的通信正常，并验证数据的正确性和一致性。  
Integration testing: API testing is useful when you need to validate the interface interactions and data transfer between different systems or components. It ensures smooth communication between systems and verifies the correctness and consistency of data.

2性能测试：如果你需要评估系统的性能特征，如响应时间、吞吐量和资源利用率，API测试可以提供更准确和可控的性能指标。通过直接调用API，可以模拟并测量系统的负载和响应时间。  
Performance testing: If you need to evaluate performance characteristics of a system, such as response time, throughput, and resource utilization, API testing provides more accurate and controllable performance metrics. By directly calling APIs, you can simulate and measure the system's load and response time.

3自动化测试：API测试是自动化测试的理想选择。由于API通常具有清晰的输入和输出，通过编写自动化脚本来测试API可以更加方便和高效。  
Automation testing: API testing is an ideal choice for automation testing. Since APIs often have clear inputs and outputs, writing automated scripts to test APIs is convenient and efficient.

**选择UI测试的情况：**

1用户体验和界面验证：如果你关注用户界面的外观、布局和交互体验，UI测试是必要的。它可以确保用户界面的正确显示和响应用户的操作。  
User experience and interface validation: If you focus on the appearance, layout, and interactive experience of the user interface, UI testing is necessary. It ensures the correct display of the user interface and responsiveness to user actions.

2用户流程和导航：如果你想验证用户在应用程序或网站中的特定流程和导航路径是否正常工作，UI测试可以模拟用户的交互动作，确保正确的导航和流程。  
User flows and navigation: If you want to validate specific user flows and navigation paths in an application or website, UI testing can simulate user interactions to ensure proper navigation and flow.

3兼容性测试：UI测试可以检查应用程序在不同浏览器、操作系统和设备上的兼容性。通过模拟用户的界面操作，可以确保应用程序在各种环境下都能正常运行。  
Compatibility testing: UI testing can check the compatibility of an application across different browsers, operating systems, and devices. By emulating user interface actions, it ensures the application works correctly in various environments.

4可视化验证：UI测试可以验证图表、图形、图像等可视化元素的正确性和准确性。  
Visual validation: UI testing can verify the correctness and accuracy of visual elements such as charts, graphs, and images.

需要注意的是，API测试和UI测试并不是互斥的，而是可以相互补充的。在综合测试策略中，可能需要同时进行API测试和UI测试，以确保系统的全面测试覆盖。具体选择哪种测试方法还取决于你的测试目标、需求和资源限制。  
It's important to note that API testing and UI testing are not mutually exclusive but can complement each other. In a comprehensive testing strategy, you may need to perform both API testing and UI testing to ensure thorough test coverage of the system. The specific choice of testing methods also depends on your testing goals, requirements, and resource limitations.

**UI**

How do u do the testing in UI?

Ul automation, what kind of framework you are using ?

**Performance test:**

做性能测试会关注哪些指标？

怎么分析性能测试结果？

**压力测试：**

看简历上写到用jmeter 做过压测，能说说怎么做压测吗？

压力测试是用于评估系统在负载条件下的性能和稳定性的测试过程。下面是一个典型的压力测试流程：

确定测试目标：明确压力测试的目标，例如确定要测试的系统、特定的性能指标和负载条件。

定义测试方案：制定详细的测试计划，定义测试的范围、策略和方法。确定要使用的工具和技术，以及测试所需的硬件、网络和环境配置。

设计测试场景：根据系统的使用情况和预期负载，设计不同的测试场景。确定要模拟的用户行为、请求类型、并发用户数和负载模式等。

准备测试环境：配置测试环境，包括硬件设备、网络设置和服务器资源。确保测试环境与生产环境的相似性，并进行必要的性能调优和优化。

设置性能基线：在进行压力测试之前，收集系统的基准性能数据，作为后续测试结果的参照。这可以帮助评估系统在负载条件下的改进和变化。

执行压力测试：根据设计的测试场景和负载模式，使用压力测试工具模拟并生成负载。逐步增加负载，观察系统的性能指标和行为，记录测试结果。

监控和分析：在测试执行期间，监控系统的性能指标和资源使用情况。收集和分析测试日志、错误报告和性能数据，以识别性能瓶颈和潜在的问题。

性能优化：根据测试结果和分析，识别系统的瓶颈和性能问题。进行必要的优化和改进，如调整配置、优化代码或增加资源。

结果评估和报告：根据测试结果和性能指标，评估系统的性能和稳定性。撰写测试报告，总结测试过程、结果和建议。

迭代测试和验证：根据测试结果和报告中的建议，进行必要的优化和改进。重复进行压力测试，以验证改进措施的有效性和系统的性能提升。

压力测试是一个迭代的过程，需要根据测试结果进行反馈和调整。通过不断优化和改进，确保系统在负载条件下具有良好的性能和稳定性。

**Postman：**

用postman 做什么？

postman 里的collection怎么管理?

**Database/Test data:**

What’s your database?

How do you maintain test data?

who provide these Redis?

Did you write your own code to reset the data environment or does it already exist in the framework? Have you ever done encapsulation code? How to write it?

Is your test data hardcored in your scripts? Where are the expected values written?

**Automation Framework/Script:**

Do u have any other automation framework in this project?

Have you experienced **Pytest** ?

pytest-html 报告  
pytest-xdist 分布式和多线程  
pytest-ordering 执行顺序  
pytest-rerunfailures 失败重跑  
allure-pytest 更好的报告

How do you write automation scripts? How to compare data consistency?

How u review script with testing department

Automation/usually use in which process?

Explain more about what kind of functions you automated for this project

Are you using test automation framework, when do test automation, who will maintain this test automation framework, if there is different pods, do they use automation or specific

说一下你们的自动化测试框架？

Selenium：

如何定位selenium的元素？

单双斜杠的区别?

定位元素问题：没有ID，class随机变化，出现的层级也变化，有时候在table有时候在html,但你知道文本，文本知道text内容，怎么通过文本定位元素呢？

自动化的具体工作，功能的实现，怎么写的

自动化问的比较细致，具体功能点怎么实现，还有写过哪些工具，如何实现的，具体代码

根据候选人的简历问自动化相关内容

自动化平台搭建：

你怎么搭建自动化测试平台，为什么这么搭建，有什么考虑，怎么实现的？

你们搭建是多少人参与？你在里面负责哪些模块？

* 你在里面实现了哪些功能？哪些工具类？
* 在你项目里，说到截图我问一下，我要截全屏的图，网页超过屏幕大小，我要截整个网页，怎么做？
* 这个是你之前实现的，还是现有的？
* Report的部分是你全部负责的？怎么分工呢？
* 如果你开箱即用的话不好用，你会怎么做？比如某个dev你给他高亮？
* 我问的不是展示的图片的高亮，而是高亮report某一段文字，怎么改呢？
* 你们平台除了selenium，有接入Appium

**Testing Point**:

How about the check point could you verify in the automation testing? Which type of verification is done? How do you prepare test data to achieve these checks?

**Platform/Tools:**

linux or windows?

How about git?

you familiar with middle ware

Do u have monitoring system?