2. What is API testing?

API stands for Application Programing Interface. It’s basiclly allow software system to communicate with each other. API testing is about veryfing it’s functionalty, performance, security and reliability so it may work as expected.   
There is Functional testing, where we test that the API return the expected responses for valid requests. There is performance testing. In this case what we are looking for its respond time, and how it behaves under various loads. Security testing is where we try to very if its secure, manly focusing on acces control and data encryption. There is reliabilty Testing where we test its stability over time. And finaly there is validation of response data. In this we ches if it returns the proper formated data. In my opinion adn many more according to google and ChatGPT, one of the best testing tool is Postman, where we can ping different servers for differnet response codes.

Question 2

3. What is the difference between regression and smoke testing?

**Regression testing**: It is used to authenticate a code change in the software does not impact the existing functonality of the product. Test cases are re-executed to check the previous functionality of the application is working fine, and the new changes have not produced any bugs. Can be performed on a new build when there is a significant cahnge in the original functionality. For example:

* new functionality is added
* there is a Change Requirement
* a defect is fixed
* the res is a performance issue to fix
* in case of enviroment change

**Smoke testing:** It checks if the core functionality is working after a new build or update so we can check quickly if the software is stable enough for further testing. It is done in a small set on a critical functionality

Question 13

What is Agile?

Agile is a process of implementing guiding principles that allow teams and businesses to rapidly develop products that have value to their business stakeholders. **It aligns software development with the needs of customers while breaking down barriers between traditionallly isolated teams**. Large emphasis what the client wants and what can be done

Agile manifesto:

* **Individuals and interactions** over processes and tools
* **Working software** over comprehensive documentation

User story is during the consultation process, you parse the work to be done into functional increments. It is written by the perspective of an end user. Each user story contribute to the value of the product and help define its purpose. It is usually written by stakeholder, that incule clients, managers, developers or the users them selves. It is not equal to system requirements

Most common user tempalte is the the 5 W’s:

* who – for example as a user
* what – for example what I want to acces via my mobile
* why – because I might not always have a computer
* where
* want

Acceptance Criteria are **the conditons that must be met for the product to be acceptable from a user perspective.** These will be defined as a set of statements with well-established pass/fail results for both functional and non-functional requirements. Any variations will be taken into account. It helps prevent misscommunication

Well written Acceptance Criteria:

* Acceptance criteria are the basis for the testing systems
* Criteria are written by clients, product owners, and/or development team
* Development team reviews acceptance criteria before writing tests

What is TDD?

Test-driven Development. Tests are written before writing the actual code. Write only enough code to pass the test, which leads to simpler and more reliable code. Key steps are the following:  
Red phase where we write a test specific to the feature or function, before we write the code. It describe the desired behavior and small piece of functionality. It will obviously fail beacuse the code does not exist yet, but it will show the perspective of the test. Green phase where we write just enough code to make the thest pass. It focuses on to pass the test with the simplest solotion, so it stops any unnecessary complexity. The Refactor phase is when the code pass the test the code is improved in structure, readability and perfomance, without changing its behavior. After the refactoring the code should still pass.

Where does the TDD fall short:

* In the TDD process, developers want to know what to test, how much to test, and how to understand failing tests –what interface should be tested? how much should be tested?
* Without clear guidelines as to what should be tested and how, confusion and misunderstandings are common
* Presenting TDD in a different light through BDD allows the development process to avoid a number of pitfalls

Question 3

4. What is BDD?

So, What is BDD?:

* Behaviour Driven Development
* BDD takes principles from both **TDD and Agile and aims to present the needs of both business analysts and developers into a single framework**
* It emphasizes collaboration between developers, testers, and non-technical stakeholders. It’s a communication bridge between technical team members. and non-technical
* **Business needs are defined in code and are testable**
* Writing test with expressive names, usually in a form of sentance, that declared what a test was cheking

BDD is not for discovering what product to build. Product owners should now what the end product is

BDD is for building the correct, reliable product that end users will love using

Starting points for BDD:

* Product owners already understand what product they desire
* Some, if not most, acceptance criteria have beed defined
* User stories have already been written

Finding our Acceptance Criteria:

* Developers or QA seek clarity from business stakeholders to define acceptance criteria for an application
* This occurs from conversations with stakeholders based on concrete examples dreived from the user story
* The best way to finding is to ask the stakeholder to provide a specific example
* **Names, Context, Situational** awarness what we need

We need a way to translate our acceptance criteria into executable test, that can verify the behaviour of our system and create confidence that we are building the right thing in the beginning even if these requirements might cahnge later on

The specification language for Cucumber, Gherkin is a business-readable, domain-specific language that describes your software’s bahavior, but not how that behavior is being impelemented. Gerkin serves two purposes: to automate testing as well as acting as documentation for the application.

Gherkin Syntax:

* **Feature:** Feature name. – Some terse, yet descriptive, text of what is desired
* **Scenario:** Some determinable business situation. Single concrete example of a user story, about how a system should behave
* **Given:** Describes the context or precondition for the scenario
* **When:** Some event or action by the actor. The person or an event acting on a system
* **Then:** Some testable expected outcome to the scenario

Question 5

6. What is a test pyramid?

Testing pyramid shows the different levels of testing. The base of the pyramid is **Unit test**: Focusing on individual components of functions of the software. Fast, reliable and provide immidiate feedback. The majority of test should consist of this. The next in the line is **Integration tests**: Howdifferent components of the modules work together providing insight of the systems internals. Fewer test than unit test, but still essential. And finally its **End-to-End** testing:

Its about testing from top to bottom every component of the application, to test if it work properly. From the UI to the back-end. It simulates real user scenarios and ensure the system meets the requirements. It’s however slow and and depend on many external factors. It should be the fewest test.

Question 6

7. You have 200 test cases. You need to do regression. But another 50 automated and 50 manual tests tops up. From these 100 new ones, 50% are with the highest importance. How many test cases would you include in the regression testing? Including the first 200 ones.

Question 7

8. It’s Monday morning. You have to finish a lot of tests until Wednesday at 16:00 when the software will be released in production. You know it’s impossible to do all of them until then. What do you do?

read more

[**1**](https://www.glassdoor.com/Interview/8-It%E2%80%99s-Monday-morning-You-have-to-finish-a-lot-of-tests-until-Wednesday-at-16-00-when-the-software-will-be-released-in-pr-QTN_3031112.htm)

Question 8

9. How do you report defects? what’s the status of a defect report? from new to…? say all states.

If an error occours we shall make a precise and exact documentation about it. First the summary: In wich enviroment the problem came to us. Then what version did we use: what version of the webpage, browser etc.,Steps to reproduce: how did we achived the error. Description of the actual result: what does the screen showes us, we can prove it with screenshots. Expected result: what should have happend, if the everything worked out fine

Question 9

10. What are the most important steps in the manual testing?

Question 10

11. What’s a constructor? an overload? an overwrite? (override - dang it englishj)

Constructor: a special function or method in a class that is called when an object or a class is created to initialize the object’s state. Overloading: Defining multiple functions or methods with the same name but dfifferent parameters. Override: A subclass that has a function or other implemantation that is already defined in its parent class. In thos case the parent class implementation gets overwritten

Question 11

12. In a web page, for e.g. Gmail, how do you test the login page?

Question 12

13. In a web page with two text boxes and a button, how do you test?

Question 13

14. What is Agile? Difference between Agile and Waterfall.

Question 14

15. One developer doesn’t agree with you, how do you handle it? He says the defect it’s not a defect, but it’s a feature. How do you handle this situation?

read more

Question 15

16. What are your three strengths/weakness points?

Question 16

17. Have you worked with Jira, Junit?

Question 17

18. How would you write a manual test case? which are the stages?

Question 18

19. SQL questions. Do you have experience in database testing?

Question 23

24. What is CI/CD?

Continous Integration: Frequently integrating small changes into the main branch. The code is commited, the automated tests are run. Helps with detecting early bugs, making it easier to fix.  
Continious Deployment: If code cahnges pass the automated testing, it will be automatically deployed tp production. No need for manual intervention wich yields faster delivery.

Question 24

25. What is try/catch method?

It’s about hadling exceptions or errors that might happend durning the code execution. It allows to try a block of code and catch any errors that may occur, handeling it without crashing the entire program. There is three blocks in this scenario. **Try** block that contaisn the code that my error out. The next is the **catch** block. It will be executed if an error happens in the try block, allowing the user to respond to it. And aftert that the **finaly** block that is optional. It will run regadles if and error occurred or not. Mainly used for cleaning up task, like closing files and connections

Question 25

26. Did you have asserts in your latest tests? more asserts in one test or just one assert?

Question 26

27. Severity and priority. What is the difference and who should handle it?

Severity is about the impact of a defect on the application. It can be Trivial – Minor – Major – Critical  
Priority refers to the urgency of fixing the defect. It descibes how quicly the issue needs to be resolved. Its levels goes by low – medium and high. The severity is determinded by the testers and devs. The priority is determined by the owners and stakeholders

Question 27

28. What is static testing?

No code execution just examining the code, the documents for potential issues. It helps with early bug detections. It involves formal inspections where a team reviews the requirements and its completion. Walkthroughs where the code author presents the work for a team. And requriemets reviews for where the documentation is examined to ensure clarity, compeltness and feasibility

Question 28

29. What is functional/non-functional testing? give examples

Functional: What the system does, Does it behaves accordingly, Testing the functions that the user uses.   
Non-functional testing: Performance, Quality, Stress test, Security, User friendlines, Reliability under expected usage

Question 29

30. What is a test plan?

Test Plan Id:Unique id for the doc. Introduction:Overview of the project, of its purpose, the scope, backgorund info. Scope of Testing: description of features, functions, and modules that will be tested. Test objectives: Specific goal to ensuring functionality, performance, security and usability. Test strategy:Plan the approach to the testing, and determining the types of testing.Test Criteria:Entry criteria – specific conditions that must be met before testing begins, Exit criteria – conditions that must be met before testing is finsihed. Test Enviroment: details about hardware, software, network configurations and other reosources. Test Schedule:timelines and milestones.Resource Requirements: Persons and its roles and responsibilites, as well as any tools that needed. Risk Assessment: Analysis of potential risks and strategies for mitigating those. Test Deliverable: List of documents and artifacts that will be produced durning the testing process. Approval and Sign-off: Approve of the test plan

test types

Unit testing: for individual components

Integration Testing: how different modules or components for together

Functional Testing: The application behaves according to requirements

End to End : test the entire application as a real suer

Regression testing: Testing if recent changes affected existing functionality

Performance Test: Responsivnes, stability, scalability, application under heavy load

Security testing: Identify vulnerablitis and security flaws

UI testing:Test about he graphical user interface  
API testing: test that focus on interfaces between different software components

Smole testing: check the basic functionality

Acceptance testing: test about the application meets business requirements

OOP principles, testing approaches

OOP Principles:

Encapsulation: building data and methods that operate on the data into a single unit. With that acces to critical data is protected

Abstraction:Hiding complx impelementation details and olny show the essential features

Inheritance:A new calss inherits properties and behaviors

Polymorphism:Objects treated as isntances of their parent class. Allowing methods to behave differently based on the object

OOP and Testing

Unit Testing and Encapsulation:Unit testing ensures each calss or method behaves as expacted, no data leakage

Inheritance and Testing: Making sure that inherited methods work as expected, especially if overriden

Polymorphism and Testing:Objects with different parent classes exhibit the expected behavior

Variables:

* Three types of variables: var, let and const
* We need to understand waht scope is. Scope is visibility and accessibility of variables, functions and objects at various parts of the code:
* Global scope – variables can be accessed from anywhere in js code
* Function scope – variables are created when the function starts and deleted when the fucntion is completed
* Block scope – variables declared inside curly brackets lock and can’t be accessed from outside of the block.
* Var – global variable, recommended to aboid
* Let – use let when you need to reassing a bariable or when a variable is only needed within a block
* Const - use cosnt for vaiables taht should not chane their value after being assigned