I'm not a robot

Here are Rest API interview questions and answers for fresher as well as experienced candidates to get their dream job. Rest API Interview Questions and Answers for Freshers1) Explain what is REST and RESTFUL?REST represents REpresentational State Transfer; it is a relatively new aspect of writing web API.RESTFUL is referred for web services written by applying REST architectural concept are called RESTful services, it focuses on system resources and how state of resource should be transported over HTTP protocol to different clients written in different language. In RESTFUL web service HTTP methods like GET, POST, PUT and DELETE can be used to perform CRUD operations. Free PDF Download: Rest API Interview Questions & Answers >> 2) Explain the architectural style for creating web API?The architectural style for creating web api areHTTP for client server communicationXML/JSON as formatting languageSimple URI as the address for the servicesStateless communication 3) Mention what tools are required to test your web API?SOAPUI tool for SOAP WS and Firefox poster plugin for RESTFUL services. 4) Mention what are the HTTP methods supported by REST?HTTP methods supported by REST are:GET: It requests a resource at the request URL. It should not contain a request body as it will be discarded. Maybe it can be cached locally or on the server.POST: It submits information to the service for processing; it should typically return the modified or new resourcePUT: At the request URL it update the resourceDELETE: At the request URL it removes the resourceOPTIONS: It indicates which techniques are supportedHEAD: About the request URL it returns meta informationLets take a look from a perspective of just a single record. Lets say there was an employee record with the employee number of 1.The following actions would have their respective meanings.POST This would not be applicable since we are fetching data of employee 1 which is already created.GET This would be used to get the details of the employee with Employee no as 1 using the RESTful web servicePUT This would be used to update the details of the employee with Employee no as 1 using the RESTful web serviceDELETE This is used to delete the details of the employee with Employee no as 1 5) Mention whether you can use GET request instead of PUT to create a resource?No, you are not supposed to use PUT for GET. GET operations should only have view rights, while PUT resource is used for updating a data. 6) Mention what are resources in a REST architecture?Resources are identified by logical URLs; it is the key element of a RESTful design. Unlike, SOAP web services in REST, you view the product data as a resource and this resource should contain all the required information. Rest API Interview Questions and Answers for Experienced7) Mention what is the difference between AJAX and REST?AJAXRESTIn Ajax, the request are sent to the server by using XMLHttpRequest objects. The response is used by the JavaScript code to dynamically alter the current page.REST have a URL structure and a request/response pattern the revolve around the use of resources.Ajax is a set of technology; it is a technique of dynamically updating parts of UI without having to reload the page.REST is a type of software architecture and a method for users to request data or information from servers.Ajax eliminates the interaction between the customer and server asynchronously.REST requires the interaction between the customer and server. 8) Mention some key characteristics of REST?Some key characteristics of REST includesREST is stateless, therefore the SERVER has no state (or session data)With a well-applied REST API, the server could be restarted between two calls as every data is passed to the serverWeb service mostly uses POST method to make operations, whereas REST uses GET to access resources 9) Mention what are the different application integration styles?The different integration styles includeShared databaseBatch file transferInvoking remote procedure (RPC)Swapping asynchronous messages over a message oriented middle-ware (MOM) 10) Explain how JAXB related to RESTful web API?JAXB stands for java arch for XML binding. 11) Mention what is the difference between PUT and POST?PUT puts a file or resource at a particular URI and exactly at that URI. If there is already a file or resource at that URI, PUT changes that file or resource. If there is no resource or file there, PUT makes onePOST sends data to a particular URI and expects the resource at that URI to deal with the request. The web server at this point can decide what to do with the data in the context of specified resourcePUT is idempotent meaning, invoking it any number of times will not have an impact on resources.However, POST is not idempotent, meaning if you invoke POST multiple times it keeps creating more resources 12) Mention which markup language can be used in restful web api?JSON and XML are the two markup language that can be used in restful web api 13) Mention what is the difference between RPC or document style web services? How you determine to which one to choose?In document style web services, we can transport an XML message as part of SOAP request which is not possible in RPC style web service. Document style web service is most appropriate in some application where XML message behaves as document and content of that document can alter and intention of web service does not rely on the content of XML message. 14) Mention what is JAX-WS and JAX-RS?Both JAX-WS and JAX-RS are libraries (APIs) for doing communication in various ways in Java. JAX-WS is a library that can be used to do SOAP communication in JAVA, and JAX-RS lets you do the REST communication in JAVA. 15) List out the tools or API for developing or testing web api?Testing tools for web services for REST APIs includesSpring REST web service using MVCJersey APICXFAxisRestlet, 16) Mention what is the difference between SOAP and REST?SOAPRESTSOAP is a protocol through which two computer communicates by sharing XML document.Rest is a service architecture and design for network-based software architectures.SOAP permits only XMLREST supports many different data formatsSOAP based reads cannot be cachedREST reads can be cachedSOAP is like custom desktop application, closely connected to the serverA REST client is more like a browser; it knows how to standardized methods and an application has to fit inside itSOAP is slower than RESTREST is faster than SOAPIt runs on HTTP but envelopes the messageIt uses the HTTP headers to hold meta informationThese interview questions will also help in your viva(orals) Here are API Testing interview questions and answers for freshers as well as experienced candidates to get their dream job. 1)What are the tools used for API testing? The tools used for various API testing are SoapUI Pro PostMan Alertsite API monitoring Free PDF Download: API Testing Interview Questions 2)What is API testing? API (Application Programming Interface) specifies how some software components should interact with other, in other words its a set of functions and procedures that allows the creation of applications which access the features or data of an application or operating system. Testing of these functions is known as API testing. 3)What are the common tests performed on APIs? The common tests performed on APIs Verification of the API whether it is updating any data structure Verify if the API does not return anything Based on input conditions, returned values from the APIs are checked Verification of the API whether it triggers some other event or calls another API 4)Mention the key difference between UI level testing and API testing? UI (User Interface) refers to testing graphical interface such as how user interacts with the applications, testing application elements like fonts, images, layouts etc. UI testing basically focuses on look and feel of an application. While, API enables communication between two separate software systems. A software system implementing an API contains functions or sub-routines that can be executed by another software system 5)Explain what is SOAP? API Testing Interview Questions SOAP-stands for Simple Object Access Protocol, and it is an XML based protocol for exchanging information between computers. 6)Explain what is REST API? It is a set of functions to which the developers performs requests and receive responses. In REST API interaction is made via HTTP protocol REST stands for Representational State Transfer, it is quickly becoming defacto standard for API creation. 7)Difference API and Unit Testing? API testing UNIT testing Unit testing is owned by development team API is mostly black box testing Unit testing is white box testing Full functionality of the system is considered in API testing as it will be used by the end-user (external developers who will use your API) Unit testing is done to verify whether each unit in isolation performs as expected or not API test are often run after the build is ready and authors do not have access to the source code For each of their module the developers are expected to build unit tests for each of their code modules and have to ensure that each module pass unit test before the code is included in a build 8)How to test APIs ? To test the APIs you should follow the following steps Select the suite in which you want to add the API test case Choose test development mode Develop test cases for the desired API methods Configure application control parameters Configure test conditions Configure method validation Execute API test View test reports Filter API test cases Sequence API test cases 9)Mention what the main areas to be taken in consideration while writing API document ? The key area to be considered when writing API documents are Source of the content Document plan or sketch Delivery layout Information required for each function in the document Automatic document creation programs 10)In API document explain how to document each function ?What are the tools used for documentation? Description: Small description about what a function does Syntax: Syntax about the parameter of the code, the sequence in which they occur, required and optional elements etc. Parameters: Functions parameters Error Messages: Syntax of error messages Example Code: Small snippet of code Related Links: Related functions Popular tools used for API documentations are JavaDoc (for Java code) Doxygen (for .Net code) 11)Explain API framework? API framework is self-explanatory. Values for test run and for holding the configurable parts, config file is used. Automated test cases must represent in parse-table format within config file. When testing API, it is not necessary to test each API so the config file have some section whose all API are activated for that specific run. 12)How does the API Builder work? API Builder is a PLSQL program consists of four SQL files For setting API parameters and starting the process one file is responsible Two files are created for temporary tables and Master package to create the outputted code Fourth file creates spooled output of the code into a file called output_script_.sql 13)Explain what is TestApi ? TestApi is a library of utility and test APIs that enables testers and developers to create testing tools and automated tests for .NET and Win32 application. It provides a set of common test building blocks, types, data-structure and algorithms. 14)What is Input injection and what are different ways of doing it ? Input Injection: It is the act of simulating user input, in several ways you can simulate user input. Direct Method Invocation Invocation using an accessibility interface Simulation using low-level input Simulation using a device driver Simulation using a robot 15)What are the main challenges of API testing? The main challenges in API testing is Parameter Selection Parameter Combination Call sequencing 16)What is API testing with runscope ? Runscope is a web application that provides backend services and easy to use interface for testing APIs. 17)Explain what are the principles of API test design? The principle for API test design are Setup : Create objects, start services, initialize data etc Execution: Steps to exercise API or scenario, also logging Verification: Oracles to evaluate execution outcome Reporting: Pass, failed or blocked Clean up: Pre-test state 18)What are the types of Bugs will API testing finds? The types of Bugs, API will find Missing or duplicate functionality Fails to handle error conditions gracefully Stress Reliability Security Unused flags Not implemented errors Inconsistent error handling Performance Multi-threading issues Improper errors 19)What are the tools used for API test automation? While testing Unit and API testing, both target source code, if an API method is using code based on .NET then the tool which is supporting should have .NET Automation tools for API testing can be used are NUnit for .NET JUnit for Java HP UFT Soap UI 20)Mention the steps for testing API ? API testing steps Select the test case that has to be fulfilled For API call develop a test case To meet the test case configure the API parameters Determine how will you validate a successful test Using programming language like PHP or .NET execute the API call Allow the API call to return the data to validate 21)What are the common protocols that are testing in API tesing ? These interview questions will also help in your viva(orals) The Postman is a popular API testing tool that helps the development team create, share, and test documents of API. The Postman tool provides a GUI interface to test an API and a command-line utility for experienced testers. This API testing tool also helps you to create the request as per the need of API. The Postman team has also released a Chrome extension that can be used to test the API from your browser.Postman Interview Questions for Freshers and ExperiencedThe list below covers all the Postman questions for freshers and professional-level candidates. This Postman interview questions guide will help you crack the interview and help you get your dream job for software testing.1) What is Postman?The Postman is a highly popular API testing tool that helps the development team create, share, and test an API. The Postman tool provides a GUI interface to the API and a command-line utility for experienced testers. 2) What is an HTTP request?An HTTP request is a program that the client makes to a name host located on a server. It works as a communication interface or a request-response protocol between a client and server. The primary use of the HTTP request is to access a resource on the server. To initiate the HTTP request, the client uses components of a URL (Uniform Resource Locator) that also includes the information needed to access the resource.Postman Interview Questions 3) State The Core Components of an HTTP Response?In Postman, every HTTP response contains four key elements.Response/Status Code- There are response code issues by a sever for clients request, as 404 means Page Not Found.HTTP Version- HTTP version name. For example, HTTP v2.2Response Header- It included information for the HTTP response message. For example. The content length, date, status, server type, etc.Response Body It contains the data which a client requested from the server. 4) Why do we use Postman?Here are some most prominent reasons for using Postman:It is free to use software which is helpful for API testingIt helps you to manage the end-to-end lifecycle of APIIt offers Runtime Service that helps manage API collections, workspaces, environments, and different examples.You can also integrate Postman with CI/CD tools like Circle CI, Jenkins, etc.It has a vast community forum that can easily address any technical issues you face while using the tool. 5) What is API?Application Programming Interface (API) is a software interface that enables two applications to interact with each other without any user intervention. It is a collection of software functions and procedures. API is defined as a code that helps two different software communicate and exchange data with each other. 6) What are the important authorization methods supported by Postman?Postman offers the following API request authorization options 1) API Key, 2) Oauth 1.0 and 2.0, 3) Basic auth, 4) Digest auth, 5) Bearer Token, 6) AWS Signature, 7) Hawk Authentication, 8) NTLM Authentication. 7) How do you log variable values in Postman?Postman allows to log the variable values in Postman in the console by using the following command:console.log(pm.variables.get("guru99")); 8) Postman is available as a native desktop app for?Postman API testing tool is currently available for Mac, Windows (32-bit / 64-bit), and Linux (32-bit / 64-bit) 9) What is the History tab in Postman?All the request you send in Postman appears under the History tab of the sidebar. It is very much similar to browser history, which you can clear whenever you want. 10) What is Basic Auth in Postman?In Postman, Basic Auth is an authorization method provided for HTTP user agents like web browsers to enter username and password. After entering the username and password that you can associate with the request. 11) What is a binary form in POST methods?Post binary firm is designed to send the information in a format that is impossible to enter manually. These options are used while sending larger files like CSV files, etc. 12) What is the main difference between Authorization and authentication?Here are a few differences between authorization and Authentication:Authorization is the act of allowing or permitting someone, whereas authentication is proving that something is genuine.Authentication always comes first, while authorization comes after authentication.Authorization is open to anyone with permission, whereas authentication requires you to have a password. 13) What is the Payload in Postman?The Payload of an API Module is the body of your request and response message. When making an API request, it contains the data you send to the server. You can send and receive Payload in various formats, for example, JSON or XML. 14) What is a collection in Postman?A collection in Postman enables you to group similar requests. It also allows you to systematically arrange the requests into folders. 15) What is a Pre-Request script?Pre-request scripts help you to execute JavaScript before a request runs. It allows you to accomplish pre-processing tasks like setting variable values, parameters, headers, and body data. 16) How the Postman variables are accessed?The Postman variables are always accessed by using the variable name:{{variable name}} 17) What is the meaning of the term environment in Postman?The environment in Postman is a set of key-value pairs. Postman allows you to build multiple environments and switch among them with a click of a button. 18) Why does Postman never accepts any other encoding apart from Base64?You can use base64 as it helps us transmit the data into the textual form and send it as HTML form data. Moreover, we must rely upon the same 64 characters in any encoding language. 19) Which kind of encoding does Postman accepts for authorization credentials?Postman only accepts Base64 encoding, which is provided inbuilt in Postman. Otherwise, it would benefit when you use 3rd party websites that help you to convert the credentials into base64. 20) What are the different scopes of an environment variable in Postman?Scope of a variable in Postman is defined as the boundaries it can access.Here are important scopes of an environment variable in Postman:Local Scope: This type of variable can be accessed only in the environment in which it was createdGlobal Scope: This type of variable can be accessed globally in any environment or no environment. 21) Is it possible to import local variables in Postman Monitors?Postman monitors allow you to import local variables but not global variables. 22) What is the Postman execution order for a collection?For all the requests in a collection, the scripts will execute in the following given order:Step 1) A pre-request script associated with a collection will run before every request.Step 2) A pre-request script associated with a folder will run before every request in a specific folder.Step 3) A test script associated with a collection will run after every request.Step 4) A test script associated with a folder will run after the request in the specific folder. 23) Can you have two global scope variables with the same name in Postman?No, the global scope never has duplicate/same names, while variables having local scope can have the same name in various environments. 24) How do you access postman variables?You can log the variable values in the console by using the command:console. Log (pm. Variables. 25) What is the use of the collection in Postman?In Postman, a collection is used to group similar requests. It helps you to arrange requests systematically into folders. 26) How can you use POSTMAN to generate random numbers in a specific range?In your Pre-request script define your variable with:pm.globals.set('randomNum', Math.floor(Math.random() * 5));Then in your URL call your variable in the URL like so:Output:{{randomNum}} 27) How do you remove local variables?Local variables are automatically removed once the tests have been executed. 28) What is Postman Collection runners?Postman contains a collection runner that is useful for automating API testing. It helps visualize details of each iteration and test results. A postman collection runner is also used for Data-driven testing. 29) What do you mean by postman monitors?The postman monitor feature is used for running collections until the users specified time. However, the user must be logged in to their account to use this feature. 30) Why saving your work in the Postman cloud is not advisable?You should not save your work in Postman as your business details do not remain confidential. Moreover, saving your on-Postman cloud may cause a security breach as it requires sign-in. Therefore, saving your work in the Postman cloud is not advisable. 31) What are the standard rules of an API test design?Here are the key principles of an API test design:Setup: Create objects, start services, and initialize data.Execution: Apply API or the scenario, including loggingVerification: It is use for evaluating the result of the executionReporting: Indicates Pass, failed, or blocked statusClean up: Pre-test state 32) What is the Team workspace in Postman?A Team workspace is a synergetic environment for users where many programmers may develop APIs, provides feedback, and collaborate on the same pools of requests. It also helps to synchronize and collaborates all the teams work in one place. 33) What is the 301-status code?301 status code represents a permanent redirect from one website page to another. It tells the search engines that the old page has become outdated, and the engine has to index the new page or URL. 34) What is status code 201?Status code 201 is created only when a resource is successfully created using a PUT or POST request. It returns a link to a newly created one with the help of the location header. 35) What is the procedure to remove local variables?The local variables can be automatically removed when you have executed and completed the tests. 36) What is Pre-Request Script in Postman?In Postman, a pre-request script is a set of rules that runs before executing a request. 37) How are Query Parameters different from Path Variables?In Postman, Path Variables are used to identify specific resources, and Query Parameters are used to sort or filter the resources. 38) What are the main drawbacks of Postman?Here are some cons/ disadvantages of using Postman:Postman cannot process more than 1000 API requests.It is not easy to manage the collections and requests in a massive size project.Postman is not an ideal API tool for workspace management in the form of code as there can be lots of code duplication while handing the dynamic API requests. 39) What are some of the JS libraries available in Postman?Some JS libraries available in Postman are1) Lodash2) Moment3) GUID APIs have taken over the world! And REST (Representational State Transfer) has emerged as the undisputed leader for building versatile and scalable APIs. As an aspiring developer, a strong grasp of REST principles is a must-have skill for your toolbox. This guide will help you ace REST API interviews by developing an expert-level understanding of what makes REST tick! I have compiled 34 genuine questions ranging from the basics of what REST actually means all the way to advanced topics like security, performance testing and best practices. By the time you work through these, you will have the knowledge and confidence to build and deploy production-grade REST APIs confidently. Lets get started!A Brief History of RESTBut before we jump into the questions, let me give you a quick primer on REST. While interviewers wont ask you to recite history, having context helps connect dots faster.So heres a super-short background! In 2000, Roy Fielding introduced the architectural principles behind REST in his famous dissertation while working on HTTP protocol standards. REST was a general architectural style for distributed systems, not just limited to web APIs.However, developers found REST to be an excellent fit for creating web services. Over the years, REST defeated heavier alternative technologies like SOAP and XML-RPC to gain widespread adoption. Its simplicity, versatility and scalability made it the natural choice for modern APIs.Fun fact Every time you use mobile or web apps that get data from the internet, there is likely a REST API powering it behind the scenes! Beginner REST Interview QuestionsLets start from the basics:Q1. What is REST API?REST or RESTful API stands for Representational State Transfer Application Programming Interface.It is an architectural style for designing loosely coupled web services focused on resources and standard HTTP methods. REST APIs allow different systems to exchange data seamlessly over HTTP protocol for tasks like data retrieval, state updates, search, transaction processing and more. They can be consumed by a wide range of clients like web apps, mobile apps, other APIs etc.For example, here is what a simple REST call looks like:GET /users/123Host: example.comThis will retrieve the details of user with ID 123 from the server.Q2. What are the key principles of REST APIs?The guiding principles of REST include:Client-Server Separation of concerns between UI and dataStatelessness No client context on server between requestsCacheability Responses must allow caching for performanceUniform Interface Standard methods to manipulate resources Layered System Components can be independently deployed These principles make REST scalable, fast and flexible.Q3. What are the advantages of using REST APIs?Some major benefits of REST APIs are:Use lighter JSON for data instead of heavy XML in SOAP Built for scalability by being statelessBetter separation of client and server logicImproved performance and efficiency General purpose interface for diverse appsEnhanced portability across languagesAllow easy integration with other systemsStandards-based using HTTP, JSON, URIs As you can see, REST solves many problems of traditional heavy RPC architectures conveniently.Intermediate REST Interview QuestionsNow that you know the basics, lets step things up!Q4. What are the most common HTTP methods used in REST APIs?The most ubiquitous HTTP methods used in REST APIs as standard operations on resources are:GET Retrieve a resourcePOST Create a new resource PUT Update an existing resourceDELETE Remove a resourcePATCH Partial update of resourceThese verbs constitute the uniform interface for manipulating resources in a REST system.Q5. How do CRUD operations map to HTTP methods?The acronym CRUD stands for: Create resourcesRead or retrieve resourcesUpdate existing resourcesDelete resourcesHere is how CRUD maps to the common HTTP methods:CRUDHTTP MethodCreatePOSTReadGETUpdatePUT/PATCHDeleteDELETEKeeping CRUD operations mapped to standardized HTTP methods is an important aspect of the uniform REST interface.Q6. What status codes are most common in REST API responses?While there are over 70 HTTP status codes defined, here are the most widely used:200 OK The standard success response201 Created After a POST creating a new resource204 No Content When no response body is sent back 400 Bad Request Invalid client request, fix and resubmit401 Unauthorized Authentication failure403 Forbidden User lacks valid permissions404 Not Found Invalid resource URI or deleted resource500 Internal Server Error The developers nemesis! Make sure you are familiar with these common codes.Q7. What does statelessness mean in context of REST APIs?Statelessness is a key constraint of REST APIs. It means the server will not store any client session state or context between requests. Each request from the client must contain all the necessary information required to process it independently without needing any server side sessions.Statelessness has performance and scalability advantages:Reduces server loadSimplifies implementationImproves reliability with tighter failure isolation Allows better horizontal scalingTypically, state data is held on client side in technologies like application state, cookies or server-side caching instead.Q8. Why is having a uniform interface important in REST?The uniform interface principle requires REST APIs use standard conventions and declarative formats like HTTP, JSON, XML consistently across components and languages. This brings many benefits:Standard methods and formats regardless of device, app typeAvoids tight coupling between client and server Enables independent evolution of connected systems Improves API discoverability, visibility and portabilitySimplifies integration between elementsBy standardizing around widely adopted formats, REST tackles complexity in modern distributed systems effectively.Q9. What are resources and resource identifiers in REST?Resources are fundamental entities managed by a REST API. They represent an object or data the API consumer requires. Resources have associated data fields or attributes that comprise the state information.Examples of resources: Resource identifiers (IDs) uniquely identify each resource or object on the server. URLs and URNs are typical resource IDs in REST.For instance: 1234 is the resource ID for that particular user. Resource IDs should be unique, persistent and immutable.Q10. What are representations in REST architecture?Representations capture the current state of a resource. When clients interact with a resource, they are actually exchanging representations of resource state instead of the resource itself.For example, a GET request to retrieve a user may return a JSON representation capturing latest details of that user without sending the actual user object.Common representation formats are:JSON (Javascript Serialized Format)XML (Extensible Markup Language) Plain textSo in summary, representations are transient snapshots of resource state in standard transmittable formats.Advanced REST API Interview QuestionsLets up the ante more with some advanced level questions:Q11. What does Idempotence mean in REST APIs?Idempotence is the principle that identical requests with identical parameters always produces the same end result, irrespective of the number of execution attempts. So the effects of an idempotent operation are independent of the number of times it is executed.For example, RESTful reads or GET requests are idempotent. Calling the same GET multiple times produces the same result. However, POST increments every execution.Idempotence allows easier handling of failures, consistency in case of retries etc. Q12. What are safe and unsafe operations in REST APIs?Safe methods are idempotent Calling them multiple times produces same resultUnsafe methods may modify or mutate state and are non-idempotentMethodSafetyIdempotentGETSafeYesPOSTUnsafeNoPUTUnsafeYesDELETEUnsafeYesSafe operations are cache friendly since same input always returns same output.Q13. How to implement API versioning for a REST API?There are a few common patterns used to handle API versions:Include version number in the URI (E.g. v1/users) Pass API version as a request parameter (E.g.GET /users?api=v1) Create a custom request header like Api-Version Handling based on content type like Accept headersOf these, putting the version number right in the URL keeps implementation clean. This also allows easy developer discovery of the API variations.Q14. How do you implement caching in a REST API?Some ways to leverage caching are: Set Headers like Cache-Control, ETag and Expires to enable cachingUse GET for cacheable read operations only Make sure responses contain no client specific stateCache wherever possible clients, CDNs, proxies, serversInvalidate cache when resources are modified Enabling caching is critical to improve throughput and lower latency for REST applications at scale.Q15. What are the different security options for REST APIs?Stateless nature of REST permits easy adoption of diverse security models:Basic Auth Username/password over HTTPS as cred in headers API Key Random unique API key to allow accessOAuth 2.0 Delegated authorization standard using access tokensJSON Web Token (JWT) Compact encoded token with user claims Choosing the optimal approach depends on the API use case, security needs and audience. Q16. What is HATEOAS principle in REST?HATEOAS stands for Hypermedia As The Engine Of Application State.It is a constraint of REST that champions hypermedia formats like HTML with links over rigid formats like JSON when designing public APIs. The progression of application flows by navigating state transitions via links.For example, a HATEOAS style REST API response after adding an order may contain not just order details but also related links about order delivery status, payment process etc. to guide further interactions.This enables loose coupling between client and services.Q17. How does Richardson Maturity Model describe levels of REST adoption?Proposed by Leonard Richardson, this model describes three levels of API adoption of core REST principles:Level 0 Uses HTTP as transport protocol without leveraging its methods and capabilitiesLevel 1 Resources with proper URIs and HTTP verbs to manipulate resourcesLevel 2 HTTP verbs applied correctly including error handling Level 3 Hypermedia controls with state transitions via hyperlinksAs expected, most modern REST APIs try to operate at Level 2 with elements of Level 3 via HATEOAS. This model gives a quick way to analyze REST capabilities.REST API Testing Interview QuestionsLets now explore some questions from the quality assurance domain:Q18. How is testing a REST API different from testing a SOAP API?While basic testing methodologies apply equally to REST and SOAP APIs, some key differences exist around messaging formats, tools and processes: Transport HTTP vs SOAPMessage formats JSON vs XMLPayload complexity REST is simplerTools Postman vs SoapUIAuthentication More options with REST However, fundamentals around requirements analysis, test case design and automated checks apply uniformly.Q19. What makes a good end-to-end test suite for a REST API?A high-quality E2E test suite for REST API should cover: All flows Happy paths to error pathsSecurity Auth, rate limiting, permissions etc. Invalid/malformed input data checks API specifications compliance testingAutomating repetitive test casesRealistic test data including images, special chars etc.Idempotency enforcementCustom metrics collection and reporting The extent of coverage would depend on the context, risk and development lifecycle stage. Q20. How can you load test a REST API and analyze its performance?When load testing a REST API, some key focus areas are:Understand and establish performance baselines first Simulate realistic usage patterns Stress test for concurrent connections Identify failure points connections, data errorsCheck for throttling, traffic shaping defensesIsolate subsystem bottlenecks DB, cache, networkMonitor memory, CPU usage spikes Analyze for any data drift or inconsistenciesTools like JMeter, k6, Locust can assist with automated API load generation and testing.Monitoring stack with metrics like response time graphs, error rates, throughput etc. provide meaningful performance visibility. REST API Design Interview QuestionsLets round things off with some common RESTful design interview questions:Q21. How would you optimize response payloads to improve efficiency?Some techniques to optimize response payload performance in APIs:Support response compression using gzip Set size limits, pagination for large datasetsAllow sorting, filtering to reduce transfersInclude only requested fields, avoid overfetching Allow embedding related data to reduce chattiness Cache wherever possible client side, CDNsUse efficient serialization formats like JSON Fine-tuning responses based on client device capabilities and use cases is vital forREST API optimization.Q22. Discuss some best practices for error handling in REST APIsSome vital best practices around error handling in REST APIs:Use correct HTTP status codes that match failure semanticsInclude error details in body for debuggability Handle both functional and operational errorsMake error response structure consistent Avoid leaking sensitive internals in responsesLog failures on servers for diagnosisReturn validation failures promptly with clear actionable adviceFail fast to aid rapid recoveryWell designed error flows and resiliency is essential for robust production APIs. Q23. How would you design REST API endpoints for a multi-tenant application?For building a scalable multi-tenant REST API, some good practices are:Include tenant ID as part of endpoint URIs for isolationEx: /api/tenants/{tenantId}/ordersCan also use dedicated domains per tenant for looser couplingAdd endpoints for cross-tenant queries in analytics use cases Implement request rate limiting per tenant Enable gathering per tenant metricsHandle tenant lifecycle in the API appropriately Building multi-tenancy right into the API contract aids scalability for SaaS applications massively.Q24. What factors influence your choice between synchronous vs asynchronous REST APIs?Some key considerations towards sync vs async REST API: Sync APIsWorks well for real-time UIs requiring fast feedback loops Caller blocked until response returned Good for simple request-response casesAsync APIs Useful for long running operations where blocking is undesiredEnables event-driven architectures push vs pull Decouples caller from handler Complicates coordination and error handlingGenerally, opt for async APIs where scalability needs outweigh responsiveness requirements or to enable non-blocking flows.ConclusionWow, those were quite a lot of questions! With this exhaustive primer spanning beginner to advanced concepts, you now have all the ammunition required to demolish any REST API interview. Good understanding of resources, URIs, caching, statelessness, error handling, asynchronous processing among other concepts will highlight both depth and breadth in your knowledge. If you are an aspiring REST API developer, possessing mastery over these areas will hugely improve your skills in analyzing requirements, designing usable contracts and developing robust production-ready interfaces consumers will love!So go dazzle your future employers with your new found REST prowess. Best wishes! In this article, weve compiled a list of Rest Assured API testing interview questions and provided detailed answers for fresher for experienced to help you prepare effectively. API testing is a crucial part of modern software development and quality assurance processes. When it comes to testing RESTful APIs in Java-based projects, Rest Assured is a popular and powerful tool of choice. If youre preparing for an interview for an API testing role or a position that involves using Rest Assured, youll likely encounter questions that assess your knowledge and skills in Rest Assured API testing. Rest Assured API Interview Questions and Answers 1. What is Rest Assured, and why is it used for API testing? Answer: Rest Assured is a Java-based library for automating and testing RESTful APIs. It simplifies the process of sending HTTP requests, validating responses, and performing assertions on API endpoints. Rest Assured is widely use for API testing because it provides a fluent and expressive API that makes writing tests easy and readable. 2. How do you set up Rest Assured in a Java project, and what are the necessary dependencies? Answer: To set up Rest Assured in a Java project: Add the Rest Assured dependency to your projects build file (e.g., Gradle or Maven). Import necessary classes from the Rest Assured library in your Java code. For Gradle, the dependency might look like this: dependencies {implementation 'io.rest-assured:rest-assured:4.4.0'} For Maven, add this to your pom.xml: io.rest-assured rest-assured 4.4.0 3. How do you send a GET request using Rest Assured, and how do you validate the response? Answer: To send a GET request using Rest Assured: import io.restassured.RestAssured;import io.restassured.response.Response; Response response = RestAssured.get("); // To validate the response status code (e.g., 200 OK):response.then().statusCode(200); // To validate response body (e.g., JSON):response.then().body("key", equalTo("expectedValue")); You can use various methods provided by Rest Assureds Response object to validate different aspects of the response, such as headers, cookies, and more. 4. Explain the purpose of Rest Assureds given-when-then structure in API testing. Answer: Rest Assured promotes a clear and expressive way of writing API tests using the given-when-then structure: Given: This is where you set up the initial conditions for the test, such as headers, request parameters, or authentication. When: This is where you perform the actual HTTP request (e.g., GET, POST, PUT). Then: In this section, you define assertions and validations on the response to ensure it meets the expected criteria. This structure improves the readability of tests and separates concerns, making it easier to understand the purpose of each test step. 5. How can you pass query parameters in a Rest Assured GET request? Answer: You can pass query parameters in a Rest Assured GET request using the param method: Response response = RestAssured.given().param("paramName", "paramValue").when().get("); You can chain multiple 'param' methods to add multiple query parameters. 6. Explain how to send a POST request with a JSON payload using Rest Assured. Answer: To send a POST request with a JSON payload using Rest Assured: import io.restassured.RestAssured;import io.restassured.http.ContentType; String requestBody = "{\"key\": \"value\"}"; Response response = RestAssured.given().contentType(ContentType.JSON) // Set content type to JSON.body(requestBody) // Set the request body.when().post("); // Validate the response as needed Ensure that you set the Content-Type header to JSON and provide the JSON payload in the request body. 7. How do you handle authentication in Rest Assured? Answer: Rest Assured provides methods to handle various types of authentication, including Basic Authentication and OAuth. For Basic Authentication: import io.restassured.RestAssured;import io.restassured.authentication.BasicAuthScheme; BasicAuthScheme authScheme = new BasicAuthScheme();authScheme.setUserName("username");authScheme.setPassword("password"); Response response = RestAssured.given().auth().basic("username", "password").when().get("); // Validate the response For OAuth, you would set up the appropriate OAuth scheme and provide the necessary tokens. 8. What is request specification in Rest Assured, and how does it simplify API testing? Answer: A request specification in Rest Assured is a reusable configuration that allows you to define common settings and headers for API requests. It simplifies API testing by eliminating the need to repeatedly specify the same settings for multiple requests. You can create a request specification once and reuse it across multiple tests. 9. How do you extract data from a JSON response in Rest Assured? Answer: You can extract data from a JSON response in Rest Assured using the jsonPath method: import io.restassured.RestAssured; String responseBody = RestAssured.get(").then().extract().response().asString(); String extractedValue = RestAssured.given().param("paramName", "paramValue").when().get(").then().extract().path("key"); You can use 'jsonPath' to navigate the JSON structure and extract specific values. 10. What is Rest Assureds response specification, and why is it valuable in API testing? Answer: A response specification in Rest Assured is a reusable configuration that defines common expectations and validations for API responses. It is valuable in API testing because it allows you to define expected status codes, response content types, and other response-related criteria once and reuse them across multiple tests. This promotes consistency and reduces redundancy in test code. 11. How can you handle dynamic values in API responses, such as timestamps or unique identifiers, when writing Rest Assured tests? Answer: To handle dynamic values in API responses, you can use Rest Assureds flexible matching mechanisms, such as: Ignoring specific fields during response validation. Using wildcard matches for dynamic or unpredictable data. Extracting dynamic values and storing them in variables for later use in assertions. 12. Explain how you can perform API testing for endpoints that require multipart file uploads in Rest Assured. Answer: To perform API testing for endpoints that require multipart file uploads in Rest Assured: import io.restassured.RestAssured;import io.restassured.http.ContentType;import org.testng.annotations.Test; import java.io.File; public class FileUploadTest { @Testpublic void testFileUpload() { File fileToUpload = new File("path/to/your/file.txt"); RestAssured .given() .multiPart("file", fileToUpload, "text/plain") .when() .post(") .then() .statusCode(200);} } You can use the multiPart method to attach files to the request. 13. How do you handle API testing scenarios that involve handling cookies in Rest Assured? Answer: Rest Assured provides methods to handle cookies in API testing. To handle cookies: Use the cookie method to set and retrieve cookies in the request and response. Perform cookie-related validations using assertions to ensure cookies are correctly set and received. 14. What is Rest Assureds logging feature, and how can it assist in debugging API tests? Answer: Rest Assureds logging feature allows you to log request and response details, including headers, bodies, and other relevant information. It assists in debugging API tests by providing a clear view of the communication between your tests and the API server. You can enable logging at various levels, such as ALL', HEADERS, or BODY, to capture the desired level of detail. 15. Explain how you can perform API testing for pagination, where you need to retrieve multiple pages of data using Rest Assured. Answer: To perform API testing for pagination in Rest Assured: Send the initial request to fetch the first page of data. Extract information about pagination, such as next page URLs or tokens, from the response. Use a loop to automate subsequent requests, following the pagination information until all data is retrieved and tested. You can use variables to keep track of the next page URLs or tokens and update them in each iteration. 16. How can you handle timeouts in Rest Assured requests, and why is it important? Answer: You can handle timeouts in Rest Assured requests using the timeout method to specify the maximum time allowed for a request to complete. Timeout handling is important in API testing to ensure that tests do not hang indefinitely and that response times are within acceptable limits. RestAssured.given().urlEncodingEnabled(false) // Disable URL encoding for the path parameter.pathParam("paramName", "paramValue").when().get(" paramName}").then().statusCode(200); 17. What is Rest Assureds support for BDD (Behavior-Driven Development), and how does it improve test readability? Answer: Rest Assured provides support for BDD-style testing using natural language constructs like given(), when(), and then(). This improves test readability by making test code more closely resemble plain English sentences, making it easier for non-technical stakeholders to understand the test scenarios. given().param("paramName", "paramValue")when().get(")then().statusCode(200); 18. How do you handle dynamic authentication tokens that change frequently in API testing with Rest Assured? Answer: To handle dynamic authentication tokens in Rest Assured: Extract the token from the response using jsonPath or other methods. Store the token in a variable. Use the stored token in subsequent requests for authentication. String token = RestAssured.given().param("username", "user").param("password", "pass").when().post(").then().statusCode(200).extract().path("token"); RestAssured.given().header("Authorization", "Bearer " + token).when().get(").then().statusCode(200); 19. Explain the purpose of Rest Assured filters, and how can they be used in API testing? Answer: Rest Assured filters use to intercept and manipulate requests and responses in API testing. They are valuable for scenarios where you need to customize or log the request/response details or perform pre-processing/post-processing tasks. Filters can be added to Rest Assured to perform actions before and after requests are sent and received. 20. How can you parameterize Rest Assured tests to run with different sets of data? Answer: You can parameterize Rest Assured tests using TestNG or JUnit data-driven testing features. For example, in TestNG, you can use the @DataProvider' annotation to provide different sets of data to a test method, enabling multiple test executions with varying input data. @DataProvider(name = "testData")public Object[][] testData() {return new Object[][] {{"param1", "value1"},{"param2", "value2"},// Add more data sets as needed};} @Test(dataProvider = "testData")public void testApi(String param, String value) {RestAssured.given().param(param, value).when().get(").then().statusCode(200);} 21. How do you handle API testing scenarios that involve handling different HTTP methods (e.g., GET, POST, PUT, DELETE) in Rest Assured? Answer: To handle different HTTP methods in Rest Assured: Use the corresponding method (get(), post(), put(), delete(), etc.) to specify the desired HTTP method for the request. Define request parameters, headers, and the request body as needed for each specific method. Perform assertions and validations based on the expected behavior for that method. 22. What is the purpose of Rest Assureds expect() method, and how can it be used in API testing? Answer: Rest Assureds expect() method is used to specify expectations for API responses in a more fluent and readable way. It allows you to define assertions and validations directly within the test flow, improving test clarity and conciseness. expect().statusCode(200).body("key", equalTo("expectedValue")).header("Content-Type", "application/json").when().get("); 23. How can you simulate error scenarios and test error handling in Rest Assured API tests? Answer: To simulate error scenarios and test error handling in Rest Assured: Send requests that expect to produce error responses (e.g., invalid input, unauthorized access). Use assertions to validate the correct status codes, error messages, and response structures as defined in the APIs documentation. RestAssured.given().param("invalidParam", "invalidValue").when().get(").then().statusCode(400).body("error.message", equalTo("Invalid parameter")); 24. How can you handle testing scenarios that involve handling different content types (e.g., JSON, XML) in Rest Assured? Answer: To handle different content types in Rest Assured: Set the Content-Type header to specify the desired content type for request payloads (if applicable). Use Rest Assureds contentType() method to specify the expected content type in response validations. Customize request and response handling based on the content type, such as parsing XML or JSON responses accordingly. RestAssured.given().contentType(ContentType.JSON).body("{ \"key\": \"value\" }").when().post(").then().contentType(ContentType.JSON).statusCode(201); 25. Explain how you would handle API testing for endpoints that require authentication tokens to be included in the request header in Rest Assured. Answer: To handle API testing for endpoints that require authentication tokens in the request header: Extract the authentication token (e.g., JWT, OAuth token) from a previous request or another source. Set the Authorization header with the token value in the request using the header() method. String token = "your-auth-token"; RestAssured.given().header("Authorization", "Bearer " + token).when().get(").then().statusCode(200); 26. What is the purpose of Rest Assureds given().filter() method, and how can it be used in API testing? Answer: Rest Assureds given()'.'filter() method is used to apply custom filters to the request or response. Filters allow you to perform tasks such as logging, modifying headers, or handling specific scenarios (e.g., authentication) before or after the request is sent. This is particularly useful for customizing request and response behavior in a flexible manner. RestAssured.given().filter(requestFilter) // Apply a custom request filter.when().get(").then().statusCode(200); 27. How can you handle API testing scenarios that involve testing file downloads in Rest Assured? Answer: To handle API testing scenarios that involve testing file downloads in Rest Assured: Send a request to the API endpoint that triggers the file download. Validate the response status code (e.g., 200 OK). Use the response to access and verify the downloaded file content or its properties. Response response = RestAssured.when().get("); response.then().statusCode(200); // Validate the downloaded file content, size, or metadata as needed 28. What is Rest Assureds support for handling redirects in API testing, and how can it be used? Answer: Rest Assured supports handling redirects by default. When you send a request, Rest Assured will automatically follow redirects and return the final response. You can use the statusCode() method to ensure that the final response has the expected status code, which includes following any redirects that occurred. RestAssured.when().get(").then().statusCode(200); // Ensure that the final response is 200 OK 29. How can you perform API testing for endpoints that require sending cookies in Rest Assured requests? Answer: To perform API testing for endpoints that require sending cookies in Rest Assured requests: Use the cookie method to set cookies in the request. Validate the presence and values of cookies in the response using assertions. RestAssured.given().cookie("sessionId", "your-cookie-value").when().get(").then().statusCode(200).cookie("responseCookieName", "expectedCookieValue"); 30. Explain the purpose of Rest Assureds log() method, and how can it assist in debugging API tests? Answer: Rest Assureds log() method is used to log request and response details, including headers, bodies, and other relevant information. It assists in debugging API tests by providing a detailed log of the interactions between your tests and the API server. You can use different logging levels (ALL, HEADERS, BODY, etc.) to control the level of detail in the logs. RestAssured.given().log().all() // Log all request and response details.when().get(").then().statusCode(200); 31. How can you handle testing scenarios where API responses include binary data, such as images or PDF files? Answer: To handle testing scenarios where API responses include binary data, such as images or PDF files: Retrieve the binary data as a byte array or input stream from the response. Validate the content, size, or other properties of the binary data as needed. byte[] binaryData = RestAssured.when().get(").then().statusCode(200).extract().asByteArray(); // Validate binary data properties or save it to a file for further inspection 32. How do you handle API testing for endpoints that require sending custom request headers in Rest Assured? Answer: To handle API testing for endpoints that require sending custom request headers in Rest Assured: Use the header()' method to set custom headers in the request. Ensure that the headers are correctly set by inspecting the response. RestAssured.given().header("CustomHeader", "headerValue").when().get(").then().statusCode(200).header("ResponseHeader", "expectedValue"); 33. What is Rest Assureds with() method, and how can it be used to enhance API testing? Answer: Rest Assureds with() method allows you to configure and enhance Rest Assureds behavior by providing various options and settings. You can use it to specify configurations such as authentication schemes, custom filters, and more to tailor Rest Assureds behavior to your specific testing needs. RestAssured.with().authentication(authScheme) // Configure authentication scheme.filter(requestFilter) // Apply a custom request filter.get(").then().statusCode(200); 34. Explain how you can perform API testing for endpoints that require sending form data in Rest Assured requests. Answer: To perform API testing for endpoints that require sending form data in Rest Assured requests: Use the formParam() method to set form parameters in the request. Ensure that the form parameters are correctly sent and processed by the API. RestAssured.given().formParam("param1", "value1").formParam("param2", "value2").when().post(").then().statusCode(200); 35. How do you handle testing scenarios that involve validating response time and performance in Rest Assured? Answer: To handle testing scenarios that involve validating response time and performance in Rest Assured: Use Rest Assureds timing assertions, such as time() and timeIn(), to measure response times. Set acceptable response time thresholds and validate that the actual response times meet these thresholds. RestAssured.given().when().get(").then().statusCode(200).time(lessThan(5000)); // Ensure response time is less than 5 seconds 36. What is the purpose of Rest Assureds 'filter()' method, and how can it be used to customize API requests and responses? Answer: Rest Assureds filter() method is used to apply custom filters to API requests and responses. Filters allow you to customize and manipulate requests and responses at various stages of the HTTP interaction, such as before sending the request or after receiving the response. This is valuable for tasks like logging, modifying headers, or handling specific scenarios (e.g., authentication). RestAssured.given().filter(requestFilter) // Apply a custom request filter.when().get(").then().statusCode(200); 37. How do you handle API testing scenarios that involve working with XML responses in Rest Assured? Answer: To handle API testing scenarios involving XML responses in Rest Assured: Set the Content-Type header to specify that the request expects XML responses. Use XML-specific validation methods and assertions provided by Rest Assured to validate XML content and structure. RestAssured.given().contentType(ContentType.XML).when().get(").then().statusCode(200).body("rootElement.childElement", equalTo("expectedValue")); 38. Explain how you can perform API testing for endpoints that require sending binary data (e.g., file uploads) in Rest Assured requests. Answer: To perform API testing for endpoints that require sending binary data (e.g., file uploads) in Rest Assured requests: Use the multiPart() method to attach binary files to the request. Ensure that the file is correctly attached and processed by the API. import io.restassured.RestAssured;import io.restassured.http.ContentType; RestAssured.given().multiPart("fileParamName", new File("path/to/file.pdf"), ContentType.PDF).when().post(").then().statusCode(200); 39. How can you handle API testing scenarios that involve sending and validating request payloads with nested JSON structures in Rest Assured? Answer: To handle API testing scenarios with nested JSON payloads in Rest Assured: Build a nested JSON object or structure in your test code. Use the body() method to set the JSON payload in the request. Validate the response using JSON path expressions to access and assert values within the nested structure. import io.restassured.RestAssured; String requestBody = "{" +"\"key1\": \"value1\"," +"\"key2\": {" +"\"nestedKey\": \"nestedValue\"" +"}" +"}"; RestAssured.given().contentType(ContentType.JSON).body(requestBody).when().post(").then().statusCode(200).body("key2.nestedKey", equalTo("nestedValue")); 40. What is Rest Assureds support for handling timeouts, and how can you specify timeout settings for API requests? Answer: Rest Assured allows you to specify timeout settings for API requests using the timeout() method. You can set the maximum time allowed for a request to complete, both for connecting to the server and for receiving a response. This is important for ensuring that tests do not hang indefinitely due to slow responses. RestAssured.given().timeout(5000) // Set a timeout of 5 seconds for the request.when().get(").then().statusCode(200); Conclusion Rest Assured API testing interview questions and answers should further strengthen your knowledge in this domain. Keep in mind that practical experience and hands-on testing with real APIs are essential for mastering Rest Assured and excelling in API testing interviews. Good luck with your preparations! Here are Rest API interview questions and answers for fresher as well as experienced

candidates to get their dream job. Rest API Interview Questions and Answers for Freshers1) Explain what is REST and RESTFUL?REST represents REpresentational State Transfer; it is a relatively new aspect of writing web API.RESTFUL is referred for web services written by applying REST architectural concept are called RESTful services, it focuses on system resources and how state of resource should be transported over HTTP protocol to different clients written in different language. In RESTFUL web service HTTP methods like GET, POST, PUT and DELETE can be used to perform CRUD operations. Free PDF Download: Rest API Interview Questions & Answers >> 2) Explain the architectural style for creating web API?The architectural style for creating web api areHTTP for client server communicationXML/JSON as formatting languageSimple URI as the address for the servicesStateless communication 3) Mention what tools are required to test your web API?SOAPUI tool for SOAP WS and Firefox poster plugin for RESTFUL services. 4) Mention what are the HTTP methods supported by REST?HTTP methods supported by REST are:GET: It requests a resource at the request URL. It should not contain a request body as it will be discarded. Maybe it can be cached locally or on the server.POST: It submits information to the service for processing; it should typically return the modified or new resourcePUT: At the request URL it update the resourceDELETE: At the request URL it removes the resourceOPTIONS: It indicates which techniques are supportedHEAD: About the request URL it returns meta informationLets take a look from a perspective of just a single record. Lets say there was an employee record with the employee number of 1.The following actions would have their respective meanings.POST This would not be applicable since we are fetching data of employee 1 which is already created.GET This would be used to get the details of the employee with Employee no as 1 using the RESTful web servicePUT This would be used to update the details of the employee with Employee no as 1 using the RESTful web serviceDELETE This is used to delete the details of the employee with Employee no as 1 5) Mention whether you can use GET request instead of PUT to create a resource?No, you are not supposed to use PUT for GET. GET operations should only have view rights, while PUT resource is used for updating a data. 6) Mention what are resources in a REST architecture?Resources are identified by logical URLs; it is the key element of a RESTful design. Unlike, SOAP web services in REST, you view the product data as a resource and this resource should contain all the required information. Rest API Interview Questions and Answers for Experienced7) Mention what is the difference between AJAX and REST?AJAXRESTIn Ajax, the request are sent to the server by using XMLHttpRequest objects. The response is used by the JavaScript code to dynamically alter the current page.REST have a URL structure and a request/response pattern the revolve around the use of resources.Ajax is a set of technology; it is a technique of dynamically updating parts of UI without having to reload the page.REST is a type of software architecture and a method for users to request data or information from servers.Ajax eliminates the interaction between the customer and server asynchronously.REST requires the interaction between the customer and server. 8) Mention some key characteristics of REST?Some key characteristics of REST includesREST is stateless, therefore the SERVER has no state (or session data)With a well-applied REST API, the server could be restarted between two calls as every data is passed to the serverWeb service mostly uses POST method to make operations, whereas REST uses GET to access resources 9) Mention what are the different application integration styles?The different integration styles includeShared databaseBatch file transferInvoking remote procedure (RPC)Swapping asynchronous messages over a message oriented middle-ware (MOM) 10) Explain how JAXB related to RESTful web API?JAXB stands for java arch for XML binding. 11) Mention what is the difference between PUT and POST?PUT puts a file or resource at a particular URI and exactly at that URI. If there is already a file or resource at that URI, PUT changes that file or resource. If there is no resource or file there, PUT makes onePOST sends data to a particular URI and expects the resource at that URI to deal with the request. The web server at this point can decide what to do with the data in the context of specified resourcePUT is idempotent meaning, invoking it any number of times will not have an impact on resources.However, POST is not idempotent, meaning if you invoke POST multiple times it keeps creating more resources 12) Mention which markup language can be used in restful web api?JSON and XML are the two markup language that can be used in restful web api 13) Mention what is the difference between RPC or document style web services? How you determine to which one to choose?In document style web services, we can transport an XML message as part of SOAP request which is not possible in RPC style web service. Document style web service is most appropriate in some application where XML message behaves as document and content of that document can alter and intention of web service does not rely on the content of XML message. 14) Mention what is JAX-WS and JAX-RS?Both JAX-WS and JAX-RS are libraries (APIs) for doing communication in various ways in Java. JAX-WS is a library that can be used to do SOAP communication in JAVA, and JAX-RS lets you do the REST communication in JAVA. 15) List out the tools or API for developing or testing web api?Testing tools for web services for REST APIs includesSpring REST web service using MVCJersey APICXFAxisRestlet, 16) Mention what is the difference between SOAP and REST?SOAPRESTSOAP is a protocol through which two computer communicates by sharing XML document.Rest is a service architecture and design for network-based software architectures.SOAP permits only XMLREST supports many different data formatsSOAP based reads cannot be cachedREST reads can be cachedSOAP is like custom desktop application, closely connected to the serverA REST client is more like a browser; it knows how to standardized methods and an application has to fit inside itSOAP is slower than RESTREST is faster than SOAPIt runs on HTTP but envelopes the messageIt uses the HTTP headers to hold meta informationThese interview questions will also help in your viva(orals)Whether youre a current or aspiring web developer, you should be familiar with APIs and how they work, especially REST APIs if youve worked with any software integration, youve likely engaged with one. In this post, well go through 25 important REST API questions you should know how to answer when interviewing for a web developer position, including some broad questions and others that may require more in-depth, technical responses. So, lets get started.Table of Contents Beginner Interview Questions with Answers1. What is REST?Answer: REST stands for Representational State Transfer, and is an architectural style based on the Hypertext Transfer Protocol (HTTP) for developing web-based applications.REST outlines several guidelines that web services must follow to be considered RESTful. These guidelines ensure that requests and resources are sent easily and efficiently between client and server using standardized HTTP methods.2. What is a REST API?Answer: An application programming interface (API) is a software-to-software interface that allows otherwise separate applications to interact and share data. For example, a news website could leverage the X (Twitter) API to automatically find relevant tweets and include them in news articles.A REST API, also called a RESTful API, is an API that follows REST principles. In a REST API, all data is treated as resources, each one represented by a unique uniform resource identifier (URI). For example, the X API makes each tweet an available resource that can be retrieved by clients. Clients can also use Xs API to post tweets and perform other actions on the site.For a more detailed explanation, check out this video from IBM: 3. Whats a real-world example of a REST API?Answer: Here are some examples of REST APIs in use:X allows publishing sites to pull information like tweets, users, tweet streams, and so on. Developers can also write programs to post tweets through the API instead of the website interface.Airlines expose their flight times and prices through APIs so travel and ticketing sites can use them.To display weather information, weather apps harness public APIs that share weather data.Public transportation services usually make their data public in real-time via APIs so that mapping and navigation apps (like Google Maps) can use them.In turn, Google Maps hosts several APIs that make its mapping data available to developers. Developers leverage these APIs to place dynamic maps on their websites or in their apps.4. Which protocol do REST APIs use?Answer: REST APIs use the HTTP protocol to communicate with clients. This allows REST APIs to be easily deployed over the internet, since HTTP is the same protocol that is used to deliver web pages to client browsers.5. What is a resource?Answer: In REST, every accessible piece of content on the server is labeled as a resource. A resource is an object with a type, associated data, a relationship with other resources on the server, and a list of methods that can be used with it. For example, a resource could be an HTML or text file, a data file, an image or video, or an executable code file.A resource is identified with a URI. Clients access resources by including their URIs in HTTP requests.6. What is a URI?Answer: URI stands for uniform resource identifier. In REST, a URI is a string that identifies a resource on a web server. Each resource has its own unique URI, which, when included in an HTTP request, allows clients to target that resource and perform actions on it. The process of targeting a resource with its URI is called addressing.The format of a URI is as follows:////Intermediate Interview Questions with Answers7. What is messaging in the context of REST?Answer: In REST, messaging refers to the back-and-forth communication between the client and API. An interaction always starts with the client messaging the API with an HTTP request. The API processes this request. It then sends back an HTTP response that gives the status of the request and any resources the client asked for.8. Which HTTP request methods are supported by REST?Answer: An HTTP request method indicates which action the client wants the API to perform on a resource. The four primary HTTP request methods in REST are:GET. Requests a resource from the server. (Note that GET cannot modify server resources, as it is a read-only method.)POST. Creates a new resource on the server.PUT. Updates an existing resource on the server.DELETE. Removes a resource from the server.Additionally, two less common HTTP requests you should also know are:HEAD. Requests meta-information about a resource. This request is similar to GET, but the response does not include a response body.OPTIONS. Retrieves a list of possible methods for a resource.9. What is the difference between the POST method and the PUT method?Answer: POST and PUT are similar but not exactly the same. POST is for creating a resource on the server, whereas PUT is for replacing a resource at a specific URI with another resource. If you use PUT at a URI that already has an associated resource, PUT will replace that resource. If there is no resource at the specified URI, PUT creates one.Additionally, PUT is idempotent, which means that calling it multiple times will only result in one resource. This is because each call replaces the existing resource (or creates a new one if there is nothing to replace).POST is not idempotent. If you call POST 10 times, youll end up with 10 different resources on the server, each with its own URI. This also means that POST responses are cacheable, whereas PUT responses are not.10. What is CRUD?Answer: CRUD stands for Create, Read, Update, Delete. These are the four basic actions that can be performed on databases through a REST API. Each action corresponds to an HTTP request method:Create = POSTRead = GETUpdate = PUTDelete = DELETEIts not the most elegant of acronyms, but it works.11. What are the main parts of an HTTP request?Answer: HTTP requests are sent by the client to the API. They request data or perform some action on the server. There are five main components of an HTTP request in REST:Start line: Indicates the intended action of the request and includes:A request method that indicates the HTTP request method to be performed on the resource (i.e., GET, POST, PUT, DELETE).A URI that identifies the requested resource on the server.The HTTP version being used, which signals which version the API should respond with.HTTP request header. Lists metadata about the request, such as the user agent, file formats the client will accept, the format of the request body, language, caching preferences, etc.HTTP request body. Contains any data associated with the request. This is only necessary if the request is to modify data on the server with the POST or PUT methods.12. What are the main parts of an HTTP response?Answer: HTTP responses are sent by the API to the client. They inform the client that the requested action was (or was not) completed and to deliver any requested resources. There are four main components of an HTTP response:HTTP version. The version of HTTP used.Status line. Indicates the status of the request with an HTTP response status code.HTTP response header. Lists metadata about the response, such as the date, server, user agent, file formats of the returned resources, caching information, etc.HTTP response body. Contains the resource data that was requested by the client and is also called the payload.13. What are some common HTTP response status codes you might see when working with a REST API?Answer: HTTP response status codes tell the client the result of the requested action (GET, POST, etc.). Some common codes youll see in HTTP responses are:200 OK. The request succeeded.201 Created. The request succeeded, and a resource was created.400 Bad Request. The request was not fulfilled due to an error in the request, such as a typo or missing data.401 Unauthorized. The request was not fulfilled because the client is not authenticated or authorized to access the requested resource.403 Forbidden. The request was not fulfilled because the client is authenticated but not authorized to access the requested resource.404 Not Found. The request was not fulfilled because the server could not locate the requested resource.500 Internal Server Error. The request was not fulfilled due to an unexpected problem with the server. (See also: 500 Internal Server Errors: What They Are & How to Fix Them)502 Bad Gateway. The request was not fulfilled due to an invalid response from an upstream server.503 Service Unavailable. The server was unable to process the request due to maintenance, overloading, or another temporary interference.14. Which markup languages are primarily used to represent resources in REST APIs? Answer: In REST APIs, XML (extensible markup language) and JSON (JavaScript Object Notation) are the two most common languages for representing resources.15. What are the principles of REST?Answer: REST APIs must adhere to five requirements:Client-server decoupling. The client and server can only interact in a series of requests and responses. Only clients can make requests, and only servers can send responses. This simple principle allows both parties to operate independently of each other.Uniform interface. All communications between the client and server must follow the same protocol. For REST, this protocol is HTTP. A uniform interface simplifies integrations because every application uses the same language to request and send data.Stateless. In stateless communication, the server does not store any information about past requests/responses. Each request and response contains all the information needed to complete the interaction. Stateless communication reduces server load, saves memory, and improves performance. It also eliminates the possibility of a failed request caused by missing data.Layered system. Layers are servers that sit between the client and the API server. These additional servers perform various functions, like identifying spam and improving performance (See also: What Is a CDN?). In REST, layers are modular and can be added and removed without affecting the messages between the client and the API server.Cacheable. Server responses indicate whether or not the resource is cacheable, so that clients can cache any resources to improve performance.Additionally, REST includes one optional condition:Code on demand. An API can send executable computer code to clients in its response. This lets the client application run the code in its own back end.16. What does it mean for an API to be stateless?Answer: Statelessness is one of the key principles of REST architecture. In stateless communication, the server does not store any information about previous communications. In other words, the client and server do not know each others state.Every request and response is a new interaction, and each request includes everything the server needs to give a successful response.17. What is caching?Answer: Caching is the method of temporarily storing a copy of a server response in a location (like computer memory) in order to retrieve it more quickly in the future.When working with REST APIs, Caching improves server performance by reducing the work the server has to do to fulfill the request. Caching also makes applications that use the API run faster since they dont need to send a new request every time they need a resource.The cache duration of a resource (i.e., how long the resource can be cached by the client before the resource is retrieved again) is specified in the HTTP response header with the Cache-Control field.18. What is payload?Answer: Payload refers to the data in the body of the HTTP request and/or response messages in GET or POST requests.For example, if you request a specific tweet from the X API, the payload comprises the document containing the tweet text and any associated files for rendering the tweet on a page.Payload can also be included in the HTTP request with the POST method. If you want to post a tweet through Xs API, the tweet text that you send in your POST request is the payload.Statelessness simplifies client-server interactions because the server does not rely on past requests to process future requests and, thus, does not need to consume space and resources storing data from these requests.19. What are some benefits of REST?Answer: REST is by far the most common method for building web APIs. Here are some reasons why:REST is based around HTTP and fits within the existing infrastructure of the web, making it easy to implement by web applications.REST uses simple web technologies like XML and JSON, making it easy to learn.Because REST communications are stateless, the client and server are decoupled. This means that integrations are scalable and easy to build and manage over time.The REST architecture is flexible enough to adapt to a huge variety of use cases.REST is a lightweight architecture. Applications built with REST are generally faster than those built with other types of APIs.REST is easy to test in the browser with an API testing tool.20. What is the difference between REST and AJAX?Answer: Asynchronous JavaScript, or AJAX, is a set of web development techniques used in web applications. At its core, AJAX allows a web page to make requests to a server and update the page interface without needing to refresh.An AJAX client might use REST APIs with its requests, but AJAX doesnt have to work with REST APIs exclusively. REST APIs can communicate with any client, whether the client uses AJAX or not.Unlike REST, which uses HTTP requests and responses for messaging, AJAX sends its requests to the server with the XMLHttpRequest object that is built into JavaScript. Server responses are executed by the pages JavaScript code to alter the page content.Expert Interview Questions with Answers21. What are some main characteristics of REST?Answer: If youre asked to point out the main features that distinguish REST APIs from others, here are some points to touch on:REST uses the HTTP protocol for communication.REST makes server resources available via URIs. Each resource has a unique URI.REST is stateless, meaning the server does not store information about past communications with clients.REST uses GET to retrieve resources from a server, whereas other web service methods use POST.22. What are some drawbacks of REST?Answer: While statelessness is a benefit of REST, it can sometimes be a disadvantage. REST does not preserve state. In other words, the server does not keep records of past interactions. If preserving state is necessary, that responsibility falls on the client.Additionally, REST is less strict with its security measures than SOAP, so developers need to be cautious and only work with APIs from legitimate, reputable providers. It also makes REST a poor choice for sending confidential information between servers and clients.23. How do you test APIs?Answer: There are many software tools designed for testing RESTful APIs Postman, JMeter, and Katalon Studio are a few. The testing process usually involves sending various requests from your testing tool and monitoring how your API responds. Many testing tools also support automated testing, allowing you to run many different scenarios quickly.To learn more about how API testing works and what kinds of tests you can run, see our guide to API testing.And, check out this post to brush up on some API testing interview questions.24. What is the difference between REST and SOAP?Answer: REST and SOAP (Simple Object Access Protocol) are two different approaches to building APIs. Here are the key differences between them:SOAP is a strict protocol for building secure APIs. REST is not a protocol it is an architectural style dictated by a set of guidelines (see question 15).REST APIs are simpler to build, more lightweight, and generally faster than SOAP APIs.SOAP APIs are considered more secure than REST APIs, though REST APIs can still implement safety features to make them reasonably secure.REST allows caching of responses, whereas SOAP does not.SOAP encodes data in XML format. REST allows you to encode data in any format, though XML and JSON are the most popular.25. How do you keep REST APIs secure?Answer: REST APIs do not employ as strict security measures as SOAP APIs and, therefore, should not be used to send or retrieve sensitive information. However, good REST APIs still implement safety measures for secure and reliable data transfers.Authentication and authorization. All requests to the API should be authenticated and authorized. Authentication is the process of verifying the identity of the client, and authorization is confirming that the client has permission to access the requested resources.Validation. After authentication and authorization, requests still need to be scanned for potentially malicious code before the API gives access to its resources. Otherwise, a server will be vulnerable to an injection attack.Encryption. TLS/SSL encryption secures the connection between client and server and prevents attackers from intercepting requests and responses.Rate-limiting. Rate-limiting methods like quotas and throttling prevent brute-force attacks like DDoS that attempt to slow or crash the server.No sensitive information in URIs. Protected information (e.g., username, password, or authentication token) should not be visible in the URI of a resource.For more in-depth explanations, see our post on API security.Now you do the REST.REST APIs are so common today that any software developer should be at least familiar with their concepts and, ideally, have some experience coding applications with them.If you can, strengthen your responses by recounting times when youve worked with a REST API. It shows that you not only have the knowledge, but you also have real-world familiarity with this powerful technology. Good luck!Are you prepared to excel in your API testing interview? This article serves as your comprehensive guide to mastering common API testing interview questions. We cover key topics including API concepts, API testing tools, Postman, and Rest Assured, providing insightful answers to help you stand out.Whether youre a fresher or an experienced professional, these interview questions empower you to showcase your expertise and leave a lasting impression.Dont hesitate; immerse yourself in this guide and elevate your API testing interview preparation. Your success is just a read away! This resource is tailored to support your journey, ensuring youre well-prepared to tackle any API testing interview that comes your way.Lastly, stay confident and composed during your interview. Your knowledge and preparation will shine through, making a strong impression on your potential employers. Best of luck in your API testing interview youve got thisHere are most frequently Asked API Interview Questions For Freshers:API testing is the process of validating the functionality, reliability, performance, and security of application programming interfaces (APIs). It ensures that APIs perform as intended and interact seamlessly with other software components.API testing is an integral part of modern software development problems and solutionsAPI testing is crucial as it helps detect defects early in the development lifecycle, ensures the stability of software applications, and enhances the user experience.SOAP (Simple Object Access Protocol) is a protocol that uses XML for message exchange and relies heavily on request-response interactions.REST (Representational State Transfer), on the other hand, is an architectural style that uses various HTTP methods and focuses on resource-based interactions.RESTful APIs are generally more lightweight, easier to implement, and widely adopted for web services.Authentication in API testing is often managed through various methods such as Basic Authentication (username/password), API keys, OAuth, or tokens. These mechanisms validate the identity of the user or application accessing the API.Testers simulate these authentication methods to ensure secure access and data protection.An API request consists of an HTTP method (GET, POST, PUT, DELETE), endpoint URL, headers (including authentication), parameters, and request body (if applicable).These components collectively define the action, data, and scope of the API call.Status codes indicate the outcome of an API request. For instance, a 200 status code signifies a successful request, while a 404 status code indicates that the resource wasnt found.Status codes help developers and testers understand the outcome of an API call and take appropriate actions.Data security in API testing involves verifying that sensitive information is transmitted securely using encryption protocols (e.g., HTTPS).Testers also validate that access controls and authorization mechanisms are in place to prevent unauthorized access to data.Endpoint testing involves testing individual API endpoints (URLs) to ensure they respond correctly to specific HTTP methods and return the expected data. It ensures that each endpoint behaves as intended and handles different scenarios effectively.Parameterization involves substituting dynamic values (variables) in API requests. Testers achieve this by using data-driven testing approaches, test data from external sources (such as spreadsheets or databases), or by using environment-specific values.API versioning is the practice of maintaining multiple versions of an API to support backward compatibility while introducing new features.Testers verify that API versions are correctly specified in requests and that changes between versions dont adversely affect existing functionality.The payload is the data sent in the request body of an API call. It could be in various formats like JSON, XML, or other data structures. Testers ensure that payloads are correctly structured, valid, and include all required information for the API call to succeed.11. What are Postman collections and environments?Postman collections are sets of saved API requests that can be organized, executed, and shared.Environments store variables and configurations that can be used across multiple requests within a collection, facilitating parameterization and testing in different scenarios.12. How do you automate API tests in Postman?API tests can be automated in Postman using test scripts. These scripts are written in JavaScript and executed after making an API call.They enable assertions to validate responses, manipulate data, and perform complex validations during automated testing.13. Explain the purpose of Pre-request Scripts and Tests in Postman.Pre-request Scripts are executed before sending a request and can be used to set dynamic values or perform calculations on variables.Tests are executed after receiving a response and help validate the response data against expected values, status codes, and more.Api documentation helps developers and testers understand how to interact with the API correctly and facilitates seamless integration.Optimizing API documentation provides comprehensive details about an APIs endpoints, methods, parameters, response formats, authentication requirements, and usage examples.Logging requests and responses during API validation failures is crucial for debugging. In tools like Postman, you can use the log() function to log detailed information about both the request and response, aiding in identifying the cause of failures.Here are some tricky API testing interview questions along with their answers:To handle dynamic values, you can use data validation techniques like regular expressions or custom validation scripts. You can also capture the dynamic values during the test execution and use them for assertions in subsequent requests.OAuth 2.0 defines several authorization flows, including Authorization Code, Implicit, Resource Owner Password Credentials, and Client Credentials.Each flow is used in different scenarios, such as web applications, single-page applications, mobile apps, and server-to-server communication.To simulate delayed responses, you can introduce a delay using tools like WireMock or Postmans setNextRequest feature. This helps in testing how your application handles slow API responses.A mock server simulates the behavior of real APIs during testing. Its beneficial for parallel development, enabling teams to work on different modules independently.Mock servers ensure that testing isnt dependent on the availability of actual APIs.One approach is to use environment variables or configuration files to store sensitive credentials.During testing, these variables can be accessed to populate the authentication details dynamically.Statelessness means that each API request from a client must contain all the information needed to understand and process the request.The server does not store any client state between requests. This simplifies scalability, as servers can process requests independently.To ensure data integrity, you can perform API tests that involve database operations in a controlled test environment, such as a test database.After testing, rollback or clean up the test data to keep the environment consistent.23. Explain the concept of content negotiation in API testing.Content negotiation allows clients to specify the format in which they want to receive the response (e.g., JSON or XML).The server examines the requests Accept header and provides the response in the requested format if available.Caching is the process of storing a copy of frequently used data to reduce the load on the server.It can impact testing as cached responses might not reflect real-time data changes. Testing caching involves strategies like cache disabling or expiration validation.Load testing APIs involves sending a large number of concurrent requests to assess performance and scalability.Challenges include handling concurrent connections, simulating real-world usage, and monitoring server response times to detect bottlenecks.These tricky questions are designed to test your deeper understanding of API testing concepts and scenarios.Postman API Interview Questions and AnswersHere are 15 common Postman API testing interview questions along with answers1. What is Postman, and how is it used in API testing? Postman is a popular collaboration platform for API development that simplifies the process of creating, testing, and documenting APIs. It provides a user-friendly interface to send API requests, view responses, and validate the behavior of APIs.2. What are Postman collections, and how are they useful?Postman collections are groups of related API requests organized for efficient testing and collaboration.They allow you to save, organize, and execute a sequence of API requests as a single entity. Collections enhance productivity by enabling the creation of test suites and workflows.3. How do you parameterize requests in Postman?Parameterization in Postman involves using variables to replace dynamic values in API requests.Variables can be defined at the collection, environment, or request level. They allow you to reuse and customize data across multiple requests or scenarios.4. What are Postman environments, and how do they work?Postman environments store variables and values that can be used across different requests within a collection.They facilitate testing in various environments (such as development, testing, production) and help avoid hardcoding values.Postman provides various methods for extracting data from API responses, such as using JSONPath or XPath expressions. You can use the pm.response.json() function to access JSON data and apply filters to extract specific values.6. How do you automate tests in Postman?Postman allows you to automate tests using scripting. You can write test scripts in JavaScript using the Postman scripting sandbox.These scripts can validate responses, check status codes, and perform complex assertions.7. Explain the purpose of Pre-request Scripts and Tests in Postman.Pre-request Scripts are executed before sending a request and are used to set up variables or perform calculations.Tests are executed after receiving a response and help validate the response data against expected values, headers, and status codes.8. How can you handle authentication in Postman?Postman supports various authentication methods, including Basic Authentication, Bearer Token, and OAuth 2.0.You can configure authentication settings in the request headers or use Postmans built-in authorization mechanisms.9. What is the Postman Runner, and how can you use it for testing?The Postman Runner is a feature that allows you to execute a collection of requests in a sequence or with iterations.Its useful for running multiple test scenarios, load testing, and verifying the functionality of APIs.You can share Postman collections with others by generating a shareable link or by exporting the collection as a JSON file.Additionally, you can collaborate using the Team feature in Postman, which allows team members to work on the same collection simultaneously.11. What are Postman monitors, and how do they benefit API testing?Postman monitors automate the process of running collections at scheduled intervals.They are useful for continuous integration and regression testing. Monitors allow teams to detect issues early in the development cycle and ensure that APIs are functioning as expected over time.Dynamic values can be handled using environment variables, scripting, and regular expressions.Postmans scripting capabilities enable you to extract dynamic values from responses and update variables for subsequent requests.13. Can you explain the concept of chaining requests in Postman?Chaining requests in Postman involves using the response data from one request as input for another request within the same collection. This is achieved using test scripts to extract data and set it as variables, which can then be used in subsequent requests.14. What is the difference between Postman and Newman?Postman is the graphical user interface for creating and testing APIs, while Newman is the command-line tool that allows you to run Postman collections in an automated, headless mode.Newman is often used for integrating Postman tests into CI/CD pipelines.15. How do you handle cookies and sessions in Postman?Postman supports cookies and sessions by automatically managing and sending cookies received in API responses.You can access cookies using the pm.cookies object in test scripts. For maintaining sessions, you can extract session tokens from responses and use them in subsequent requests.Here are some of the frequently asked API interview questions and answers for rest assured:Rest Assured is a Java library that provides a domain-specific language (DSL) for writing robust and maintainable tests for RESTful APIs. It simplifies the process of sending HTTP requests, receiving responses, and validating API behavior.Both api testing tools, Rest Assured allows for programmatic API testing within Java projects, while Postman is a graphical tool for manually testing APIs. Rest Assured is often preferred for automated testing and integration into larger test frameworks.Rest Assured supports various authentication methods such as Basic, Digest, OAuth, and more. You can use the .auth() method to specify the authentication credentials or tokens required to access protected resources.Rest Assured provides a fluent API for response validation. You can use methods like .statusCode(), .body(), .header(), and .jsonPath() to check the expected status codes, response content, headers, and specific JSON values.Method chaining is a technique used in Rest Assured where multiple method calls are chained together in a single line. Each method returns an object that allows you to call subsequent methods on it, making the code concise and readable.In Rest Assured, given() is used to set up the request specification, when() is used to specify the HTTP method and the API endpoint, and then() is used to define the assertions on the response.You can use the .header() method to add headers to the request, and the .header() method in conjunction with .extract().headers() to extract and validate response headers.The extract() method in Rest Assured is used to extract different parts of the response, such as the response body, headers, cookies, and more. It allows you to validate and manipulate these extracted values.You can use the .pathParam(), .queryParam(), and .formParam() methods to parameterize API requests. These methods allow you to replace placeholders in the request URL or body with actual values.Rest Assured provides powerful features like JsonPath and XMLPath to extract and validate dynamic values from API responses.These tools allow you to navigate and manipulate the JSON or XML structure effectively.The .given().spec() method allows you to reuse a predefined request specification in multiple API requests. It helps in maintaining consistency and reducing redundancy in request setup.Rest Assured provides the .time() method that allows you to assert the response time of an API call. You can specify the maximum acceptable response time and time unit to validate the APIs performance.13. What is the difference between assertThat() and assertThat().body() in Rest Assured?assertThat() is used for general assertions on the response, while assertThat().body() is used specifically to validate the response body content, such as JSON or XML attributes and values.You can use the .assertThrows() method to validate that a specific exception is thrown during an API call. This is useful for checking error responses and ensuring proper error handling.Rest Assured supports file uploads through the .multiPart() method, which allows you to attach files to a request. This is particularly useful when testing APIs that require file uploads, such as image or document uploads.For freshers, API testing interviews generally focus on evaluating your foundational knowledge, problem-solving skills, and your ability to grasp essential concepts.The interviewers understand that you may not have extensive experience, so theyre more interested in gauging your enthusiasm, willingness to learn, and potential to grow within the roleExperienced candidates face a more in-depth evaluation of their API testing expertise, along with a focus on their practical experience, qualities and understanding of QA lead roles and responsibilities, if youre applying for a managerial position and contributions to previous projects.Here are few general tips to know before your interview:Expect questions that test your understanding of API basics, such as HTTP methods, status codes, request and response structures, and the purpose of APIs in software development.You might be presented with hypothetical scenarios to assess how you approach testing challenges. Focus on your thought process, how you identify test cases, and your approach to validating API responses.Interviewers often inquire about your testing approach. Be prepared to explain how youd design test cases, what factors you consider, and how you ensure thorough coverage of API functionalities.If you have automation experience, anticipate questions about the tools and frameworks youve used, your approach to building automated tests, and how you manage test data.Be ready to discuss past projects where youve applied API testing. Highlight challenges you faced, how you overcame them, and the impact of your testing on the overall projects success.Mastering fundamental API testing interview questions and answers empowers you to confidently navigate the interview process and leave a lasting impression on potential employers. These tools play a crucial role in ensuring the functionality, reliability, and security of applications relying on API interactions.As you step into your API testing interview, remember that preparation is your strongest ally. Dedicate yourself to understanding the intricacies of API testing and commit to continuous learning; this dedication will undoubtedly shine through.Embrace the opportunity with confidence, showcasing not only your technical prowess but also your problem-solving skills and adaptability. Approach each question thoughtfully, drawing on your knowledge and experiences.Best of luck! May your interview be a stepping stone to a successful and fulfilling career in API testing, where youll contribute significantly to the development of robust and reliable software systems Prior to the year 2000, Application Program Interface, which is now commonly called APIs, were designed to be secure which were also very complex to develop, harder to develop, and even harder to maintain. They were not meant to be accessible. In 2000, a group of researchers headed by Roy Fielding came up with the idea of REST (REpresentational State Transfer) which brought out the true power and potential of APIs. The purpose of REST was to create standards for communication between two servers that are located anywhere in the world. They came up with various principles, constraints, and properties that constituted a resource-oriented architecture, client-server-based architecture, and interface uniformity that required no state preservation. These were easily cacheable, simple to develop, and could be implemented by means of Hypertext Transfer Protocol(HTTP). The rules seemed many, but the purpose was universal. The onset of REST became a true game-changer for the API landscape as the APIs developed under REST used less bandwidth, were simple to develop and the communication was supported via the internet thereby not requiring the servers to be connected physically.RESTful APIs have also given birth to various trends like cloud computing and microservices-based architecture. They have made communication and computing over the internet seem easy. Hence, it is important for any developer to know what REST is, how it works, what are its features and how can you develop services in a secure manner to go with the trend. Many companies prefer developers with REST knowledge as they can help them develop products that are scalable, easy to maintain and make their products reach out to the world due to the power of the internet.In the below section, we will see what are the most commonly asked questions on RESTful web services during an interview and some questions on the JAX-RS library, and some on RESTful web services implemented using the Spring MVC framework. RESTful web services are services that follow REST architecture. REST stands for Representational State Transfer and uses HTTP protocol (web protocol) for implementation. These services are lightweight, provide maintainability, scalability, support communication among multiple applications that are developed using different programming languages. They provide means of accessing resources present at server required for the client via the web browser by means of request headers, request body, response body, status codes, etc. Create a free personalised study plan Create a FREE custom study plan Get into your dream companies with expert guidance Get into your dream companies with expert.. Real-Life Problems Prep for Target Roles Custom Plan Duration The technique of sending a message from the REST client to the REST server in the form of an HTTP request and the server responding back with the response as HTTP Response is called Messaging. The messages contained constitute the data and the metadata about the message. The disadvantages are:As the services follow the idea of statelessness, it is not possible to maintain sessions. (Session simulation responsibility lies on the client-side to pass the session id)REST does not impose security restrictions inherently. It inherits the security measures of the protocols implementing it. Hence, care must be chosen to implement security measures like integrating SSL/TLS based authentications, etc. You can download a PDF version of Rest Api Interview Questions. Download PDF Download PDF HTTP Methods are also known as HTTP Verbs. They form a major portion of uniform interface restriction followed by the REST that specifies what action has to be followed to get the requested resource. Below are some examples of HTTP Methods:GET: This is used for fetching details from the server and is basically a read-only operation.POST: This method is used for the creation of new resources on the server.PUT: This method is used to update the old/existing resource on the server or to replace the resource.DELETE: This method is used to delete the resource on the server.PATCH: This is used for modifying the resource on the server.OPTIONS: This fetches the list of supported options of resources present on the server.The POST, GET, PUT, DELETE corresponds to the create, read, update, delete operations which are most commonly called CRUD Operations.GET, HEAD, OPTIONS are safe and idempotent methods whereas PUT and DELETE methods are only idempotent. POST and PATCH methods are neither safe nor idempotent. These are the standard codes that refer to the predefined status of the task at the server. Following are the status codes formats available:1xx - represents informational responses2xx - represents successful responses3xx - represents redirects4xx - represents client errors5xx - represents server errorsMost commonly used status codes are:200 - success/OK201 - CREATED - used in POST or PUT methods.304 - NOT MODIFIED - used in conditional GET requests to reduce the bandwidth use of the network. Here, the body of the response sent should be empty.400 - BAD REQUEST - This can be due to validation errors or missing input data.401- UNAUTHORIZED - This is returned when there is no valid authentication credentials sent along with the request.403 - FORBIDDEN - sent when the user does not have access (or is forbidden) to the resource.404 - NOT FOUND - Resource method is not available.500 - INTERNAL SERVER ERROR - server threw some exceptions while running the method.502 - BAD GATEWAY - Server was not able to get the response from another upstream server. As the name itself stands (JAX-RS= Java API for RESTful Web Services) is a Java-based specification defined by JEE for the implementation of RESTful services. The JAX-RS library makes usage of annotations from Java 5 onwards to simplify the process of web services development. The latest version is 3.0 which was released in June 2020. This specification also provides necessary support to create REST clients. The REST architecture is designed in such a way that the client state is not maintained on the server. This is known as statelessness. The context is provided by the client to the server using which the server processes the clients request. The session on the server is identified by the session identifier sent by the client. Advance your career with Mock Assessments Refine your coding skills with Mock Assessments Real-world coding challenges for top company interviews Real-world coding challenges for top companies Real-Life Problems Detailed reports Every RESTful web service has the following features:The service is based on the Client-Server model.The service uses HTTP Protocol for fetching data/resources, query execution, or any other functions.The medium of communication between the client and server is called Messaging.Resources are accessible to the service by means of URIs.It follows the statelessness concept where the client request and response are not dependent on others and thereby provides total assurance of getting the required data.These services also use the concept of caching to minimize the server calls for the same type of repeated requests.These services can also use SOAP services as implementation protocol to REST architectural pattern. Uniform Resource Identifier is the full form of URI which is used for identifying each resource of the REST architecture. URI is of the format:////There are 2 types of URI:URN: Uniform Resource Name identifies the resource by means of a name that is both unique and persistent.URN doesnt always specify where to locate the resource on the internet. They are used as templates that are used by other parsers to identify the resource.These follow the urn scheme and usually prefixed with urn:. Examples includeurn:isbn:1234567890 is used for identification of book based on the ISBN number in a library application.urn:mpeg:mpeg7:schema:2001 is the default namespace rules for metadata of MPEG-7 video.Whenever a URN identifies a document, they are easily translated into a URL by using resolver after which the document can be downloaded.URL: Uniform Resource Locator has the information regarding fetching of a resource from its location.Examples include: //sampleServer.com/sampleFile.zipfile:///home/interviewbit/sampleFile.txtURLs start with a protocol (like ftp, http etc) and they have the information of the network hostname (sampleServer.com) and the path to the document(/samplePage.html). It can also have query parameters. Every content in the REST architecture is considered a resource. The resource is analogous to the object in the object-oriented programming world. They can either be represented as text files, HTML pages, images, or any other dynamic data.The REST Server provides access to these resources whereas the REST client consumes (accesses and modifies) these resources. Every resource is identified globally by means of a URI. JAX-RS applications have the root resource classes packaged in a war file. There are 2 means of configuring JAX-RS applications.Use @ApplicationPath annotation in a subclass of javax.ws.rs.core.Application that is packaged in the WAR file.Use the tag inside the web.xml of the WAR. web.xml is the deployment descriptor of the application where the mappings to the servlets can be defined. They are the runtime annotations in the JAX-RS library that are applied to Java methods. They correspond to the HTTP request methods that the clients want to make. They are @GET, @POST, @PUT, @DELETE, @HEAD.Usage Example:import javax.ws.rs.Path; public class InterviewBitService { public String getRESTQuestions() { } } Yes. the JAX-RS Client API provides a method called Invocation.Builder.async() that is used for constructing client requests that need to be executed asynchronously. Invoking a request asynchronously does the task of returning the control to the caller by returning with datatype java.util.concurrent.Future whose type is set to return the service call type. Future objects are used because they have the required methods to check whether the asynchronous calls have been completed and if yes, then retrieve the responses. They also provide the flexibility to cancel the request invocations and also check if the cancellation has been successful.Let us understand this with the help of a random example. We know that the Future interface from the java.util.concurrent has the below functions available:package java.util.concurrent;public interface Future { boolean cancel(boolean mayInterruptIfRunning); boolean isCancelled(); boolean isDone(); V get() throws InterruptedException, ExecutionException; V get(long timeout, TimeUnit unit) throws InterruptedException, ExecutionException, TimeoutException;}Let us consider we have this function below which is used for processing 2 Ids parallelly.public void processIds(String userId1, String questionId){ Client client = ClientBuilder.newClient(); Future futureResponse1 = client.target(" +userId).request().async().get(); Future futureResponse2 = client.target(" +questionId).request().async().get(Question.class); Response response1 = futureResponse1.get(); User userObject = response1.readEntity(User.class); try { Question question = futureResponse2.get(2, TimeUnit.SECONDS); } catch (TimeoutException timeoutException) { } return;}In the above example, we see that there are 2 separate requests getting executed parallelly. For the first future object, we await the javax.ws.rs.core.Response indefinitely using the get() method until we get the response. For the second future object, we wait for the response only for 2 seconds and if we do not get within 2 seconds, then the get() method throws TimeoutException. We can also use the isDone() method or isCancelled() method to find out whether the executors have completed or cancelled. @Path - This specifies the relative URI path to the REST resource.@GET - This is a request method designator which is corresponding to the HTTP GET requests. They process GET requests.@POST - This is a request method designator which is corresponding to the HTTP POST requests. They process POST requests.@PUT - This is a request method designator which is corresponding to the HTTP PUT requests. They process PUT requests.@DELETE - This is a request method designator which is corresponding to the HTTP DELETE requests. They process DELETE requests.@HEAD - This is a request method designator which is corresponding to the HTTP HEAD requests. They process HEAD requests.@PathParam - This is the URI path parameter that helps developers to extract the parameters from the URI and use them in the resource class/methods.@QueryParam - This is the URI query parameter that helps developers extract the query parameters from the URI and use them in the resource class/methods.@Produces - This specifies what MIME media types of the resource representations are produced and sent to the client as a response.@Consumes - This specifies which MIME media types of the resource representations are accepted or consumed by the server from the client. A resource class is nothing but a Java class that uses JAX-RS provided annotations for implementing web resources.They are the POJOs that are annotated either with @Path or have at least one method annotated with @Path, @GET, @POST, @DELETE, @PUT, etc.Example:import javax.ws.rs.Path; public class InterviewBitService { } JAX-RS stands for Java API for RESTful Web services. They are nothing but a set of Java-based APIs that are provided in the Java EE which is useful in the implementation and development of RESTful web services.Features of JAX-RS are:POJO-based: The APIs in the JAX-RS is based on a certain set of annotations, classes, and interfaces that are used with POJO (Plain Old Java Object) to expose the services as web services.HTTP-based: The JAX-RS APIs are designed using HTTP as their base protocol. They support the HTTP usage patterns and they provide the corresponding mapping between the HTTP actions and the API classes.Format Independent: They can be used to work with a wide range of data types that are supported by the HTTP body content.Container Independent: The APIs can be deployed in the Java EE container or a servlet container such as Tomcat or they can also be plugged into JAX-WS (Java API for XML-based web services) providers. The RestTemplate is the main class meant for the client-side access for Spring-based RESTful services. The communication to the server is accomplished using the REST constraints. This is similar to other template classes such as JdbcTemplate, HibernateTemplate, etc provided by Spring. The RestTemplate provides high-level implementation details for the HTTP Methods like GET, POST, PUT, etc, and gives the methods to communicate using the URI template, URI path params, request/response types, request object, etc as part of arguments.Commonly used annotations like @GetMapping, @PostMapping, @PutMapping, etc are provided by this class from Spring 4.3. Prior to that, Spring provided (and still provides) @RequestMapping annotation to indicate what methods were being used. The annotation is used for mapping requests to specific handler classes or methods.In spring, all the incoming web request routing is handled by Dispatcher Servlet. When it gets the request, it determines which controller is meant for processing the request by means of request handlers. The Dispatcher Servlet scans all the classes annotated with @Controller. The process of routing requests depends on @RequestMapping annotations that are declared inside the controller classes and their methods. @Controller@RestControllerMostly used traditional Spring MVC service.Represents RESTful web service in Spring.It is mostly used in Spring MVC service where model data needs to rendered using view.It is used in case of RESTful web service that returns object values bound to response body.If response values need to be converted through HttpMessageConverters and sent via response object, extra annotation @ResponseBody needs to be used on the class or the method handlers.The default behavior of the @RestController needs to be written on the response body because it is the combination of @Controller and @ResponseBody.@Controller provides control and flexibility over how the response needs to be sent.@RestController annotation has no such flexibility and writes all the results to the response body. @PathVariable annotation is used for passing the parameter with the URL that is required to get the data. Spring MVC provides support for URL customization for data retrieval using @PathVariable annotation. Yes. Spring MVC needs to be on the classpath of the application while developing RESTful web services using Spring. This is because, the Spring MVC provides the necessary annotations like @RestController, @RequestBody, @PathVariable, etc. Hence the spring-mvc.jar needs to be on the classpath or the corresponding Maven entry in the pom.xml. HttpMessageConverter is a strategic interface that specified a converter for conversion between HTTP Requests and responses. Spring REST uses the HttpMessageConverter for converting responses to various data formats like JSON, XML, etc. Spring makes use of the Accept header for determining the type of content the client expects. Based on this, Spring would find the registered message converter interface that is capable of this conversion. We have seen what are the most commonly asked questions on RESTful web services during an interview. REST APIs have become a very important tool in the software industry. Developing RESTful web services that are scalable and easily maintainable is considered an art. As the industry trends increase, the REST architecture would become more concrete and the demand for developers who know the development of RESTful web services would increase steadily.References:To learn more about REST, you can refer to the below 2 links: //www.restapitutorial.com/ 1. What protocol does REST follow? 2. Is it possible to maintain sessions in REST on the server-side? 3. HTTP Code 200 represents which among the following? 4. Which of the below does the task of binding the parameters passed to the HTTP method to HTTP Header while using JAX RS API? 5. What does the status code 302 represent? 6. What method should be used to obtain a list of supported operations in REST services? 7. What constraint is not a strict requirement for a service to be called a RESTful web service? 8. Which among the below directives belonging to the Cache-Control header of HTTP response provide information to the server that the resources have to be revalidated if max-age has crossed? 9. What category do 1xx HTTP status codes belong to? 10. Which component of HTTP response has the metadata for stating the type of response message is in the form of key-value pairs? 11. Which of the following options are true for REST Services? 12. What category does the 5xx HTTP code belong to? 13. Which directive of the Cache-control header in the HTTP Response tells that the resource cannot be cached? Here are Rest API interview questions and answers for fresher as well as experienced candidates to get their dream job. Rest API Interview Questions and Answers for Freshers1) Explain what is REST and RESTFUL?REST represents REpresentational State Transfer; it is a relatively new aspect of writing web API.RESTFUL is referred for web services written by applying REST architectural concept are called RESTful services, it focuses on system resources and how state of resource should be transported over HTTP protocol to different clients written in different language. In RESTFUL web service HTTP methods like GET, POST, PUT and DELETE can be used to perform CRUD operations. Free PDF Download: Rest API Interview Questions & Answers >> 2) Explain the architectural style for creating web API?The architectural style for creating web api areHTTP for client server communicationXML/JSON as formatting languageSimple URI as the address for the servicesStateless communication 3) Mention what tools are required to test your web API?SOAPUI tool for SOAP WS and Firefox poster plugin for RESTFUL services. 4) Mention what are the HTTP methods supported by REST?HTTP methods supported by REST are:GET: It requests a resource at the request URL. It should not contain a request body as it will be discarded. Maybe it can be cached locally or on the server.POST: It submits information to the service for processing; it should typically return the modified or new resourcePUT: At the request URL it update the resourceDELETE: At the request URL it removes the resourceOPTIONS: It indicates which techniques are supportedHEAD: About the request URL it returns meta informationLets take a look from a perspective of just a single record. Lets say there was an employee record with the employee number of 1.The following actions would have their respective meanings.POST This would not be applicable since we are fetching data of employee 1 which is already created.GET This would be used to get the details of the employee with Employee no as 1 using the RESTful web servicePUT This would be used to update the details of the employee with Employee no as 1 using the RESTful web serviceDELETE This is used to delete the details of the employee with Employee no as 1 5) Mention whether you can use GET request instead of PUT to create a resource?No, you are not supposed to use PUT for GET. GET operations should only have view rights, while PUT resource is used for updating a data. 6) Mention what are resources in a REST architecture?Resources are identified by logical URLs; it is the key element of a RESTful design. Unlike, SOAP web services in REST, you view the product data as a resource and this resource should contain all the required information. Rest API Interview Questions and Answers for Experienced7) Mention what is the difference between AJAX and REST?AJAXRESTIn Ajax, the request are sent to the server by using XMLHttpRequest objects. The response is used by the JavaScript code to dynamically alter the current page.REST have a URL structure and a request/response pattern the revolve around the use of resources.Ajax is a set of technology; it is a technique of dynamically updating parts of UI without having to reload the page.REST is a type of software architecture and a method for users to request data or information from servers.Ajax eliminates the interaction between the customer and server asynchronously.REST requires the interaction between the customer and server. 8) Mention some key characteristics of REST?Some key characteristics of REST includesREST is stateless, therefore the SERVER has no state (or session data)With a well-applied REST API, the server could be restarted between two calls as every data is passed to the serverWeb service mostly uses POST method to make operations, whereas REST uses GET to access resources 9) Mention what are the different application integration styles?The different integration styles includeShared databaseBatch file transferInvoking remote procedure (RPC)Swapping asynchronous messages over a message oriented middle-ware (MOM) 10) Explain how JAXB related to RESTful web API?JAXB stands for java arch for XML binding. 11) Mention what is the difference between PUT and POST?PUT puts a file or resource at a particular URI and exactly at that URI. If there is already a file or resource at that URI, PUT changes that file or resource. If there is no resource or file there, PUT makes onePOST sends data to a particular URI and expects the resource at that URI to deal with the request. The web server at this point can decide what to do with the data in the context of specified resourcePUT is idempotent meaning, invoking it any number of times will not have an impact on resources.However, POST is not idempotent, meaning if you invoke POST multiple times it keeps creating more resources 12) Mention which markup language can be used in restful web api?JSON and XML are the two markup language that can be used in restful web api 13) Mention what is the difference between RPC or document style web services? How you determine to which one to choose?In document style web services, we can transport an XML message as part of SOAP request which is not possible in RPC style web service. Document style web service is most appropriate in some application where XML message behaves as document and content of that document can alter and intention of web service does not rely on the content of XML message. 14) Mention what is JAX-WS and JAX-RS?Both JAX-WS and JAX-RS are libraries (APIs) for doing communication in various ways in Java. JAX-WS is a library that can be used to do SOAP communication in JAVA, and JAX-RS lets you do the REST communication in JAVA. 15) List out the tools or API for developing or testing web api?Testing tools for web services for REST APIs includesSpring REST web service using MVCJersey APICXFAxisRestlet, 16) Mention what is the difference between SOAP and REST?SOAPRESTSOAP is a protocol through which two computer communicates by sharing XML document.Rest is a service architecture and design for network-based software architectures.SOAP permits only XMLREST supports many different data formatsSOAP based reads cannot be cachedREST reads can be cachedSOAP is like custom desktop application, closely connected to the serverA REST client is more like a browser; it knows how to standardized methods and an application has to fit inside itSOAP is slower than RESTREST is faster than SOAPIt runs on HTTP but envelopes the messageIt uses the HTTP headers to hold meta informationThese interview questions will also help in your viva(orals) API testingensures that APIs function correctly infunctionality, performance, security, and reliability. UnlikeUI testing, which focuses on appearance, API testing verifies backend operations. With APIs being crucial in modernsoftware, mastering API testing is essential for QA professionals. Companies likeGoogle, Facebook, and Amazon prioritizeAPI testing for high software standards.This interview guide covers the most askedAPI Testing Interview Questionsforbeginners and experienced professionals(3, 5, and 8 years). It includesREST vs. SOAP, testing tools, authentication, performance testing, andPostmanexamples to help you excel in API testing interviewsAPI Testing Interview Questions For Freshers These questions focus on foundational concepts to help freshers build a strong understanding of API testing basics.1. What is API testing, and why is it important?API Testing verifies the functionality, performance, and security of application programming interfaces (APIs) by sending requests and validating responses. It is important because APIs enable communication between systems, and testing ensures the reliability and correctness of the software product.API testing processRead More - API Testing.2. What is an API and Types of APIs?AnApplication Programming Interface (API)is useful for communication between different software systems. It acts as an interface between two applications and allows two software systems to communicate with one another.APIs (Application Programming Interfaces) allow different software systems to communicate with each other.Web APIs: A Web API (also known as Web Service) is one of the most common types of APIs. It works over the internet and follows HTTP protocols. Web APIs are open-source, meaning they can be accessed by multiple clients like phones, tablets, or computers to exchange data.Local APIs: Local APIs allow applications within the same system to communicate with each other. Developers use them for middleware servicessoftware that connects different applications.Program APIs: Program APIs help one software program interact with another, even if they are not on the same system. They often use Remote Procedural Calls (RPCs) to make a remote program appear as if its running locally.Read More - API.3. What is the difference between REST and SOAP APIs?REST: Lightweight, uses HTTP methods (GET, POST), stateless, and returns JSON/XML.SOAP: Protocol-based, uses XML, and supports complex operations with WS-Security.SOAP APIREST APIRelies on SOAP (Simple Object Access Protocol)Relies on REST (Representational State Transfer) architecture using HTTP.Transports data in standard XML format.Generally transports data in JSON. It is based on URI. Because REST follows a stateless model, REST does not enforce message format as XML or JSON etc.Because it is XML based and relies on SOAP, it works with WSDLIt works with GET, POST, PUT, DELETERead More - SOAP API Vs. REST API.4. Mention Common HTTP Methods used in API Testing. Here are the Common HTTP Methods used in API Testing: POST: It is used to create new data or resources on an API. GET: It is used to retrieve data or resources from an API. PUT: It is used to update existing data or resources on an API. PATCH: It is used to update existing data or resources from an API partially. DELETE: It is used to delete data or resources from an API. Read More - HTTP Methods of API Testing.5. What are the Different Types of API Testing?Here are the different types of API testing :Functional Testing: Checks if the API functions correctly according to the requirements.Load Testing: Tests how the API performs under heavy load or many simultaneous requests to check its stability and responsiveness.Security Testing: Ensures the API is protected against threats like unauthorized access, injection attacks, and data leaks.Reliability Testing: Verifies that the API can handle requests consistently over time without failures.Validation Testing: Confirms that the API returns the expected results and follows the correct data format and protocols.Error Handling Testing: Checks how the API responds to invalid inputs or unexpected conditions, making sure it handles errors gracefully.Interoperability Testing: Ensures the API works well across different devices, platforms, or systems that interact with it.Documentation Testing: Reviews the API documentation to ensure it is accurate, complete, and easy to understand for developers.6. What is the role of Postman in API testing?Postman is a tool for designing, testing, and automating API requests. It allows sending requests, validating responses, and scripting tests.Read More - Postman for API

Development.7. List the differences between API and Web Service.Here is theDifferences between Web Services and Web API: ParametersAPIWeb ServiceDefinitionAPI is an Application Programming Interface that acts as an interface between two applications.Web services are a type of API that must be accessed through a network connection.Protocols SupportIt provides support for HTTP/s protocol.It provides support for HTTP protocol.XML SupportAPI supports XML and JSON.Web service supports XML.Read More - Web Services Vs. Web API.8. What protocols can be tested using API Testing?API testing can be used to test various protocols that facilitate communication between software systems. Some commonly tested protocols include:HTTP/HTTPS:Used for RESTful APIs, ensuring secure and structured web communication.SOAP (Simple Object Access Protocol):A protocol that uses XML-based messaging for web services.REST (Representational State Transfer):A widely used web service architecture over HTTP.GraphQL:A query language for APIs that allows fetching specific data efficiently.WebSockets:For real-time communication between a client and a server9. Tell me Advantages and Disadvantages of API Testing.Advantages:Faster bug fixes:API testing works quickly to provide a solution, thus it helps to diagnose problems earlier in the development of the system allowing for efficient bug detection and removal.Reduced costs:API testing provides quick release of results, easier to maintain, thus reducing the cost of testing.Universal language support:API testing provides support for many different languages like JavaScript, Ruby, Python, and PHP. Formats like XML, and JSON are supported during API testing.Dis-advantages: Technical skills:API testing requires skilled and trained testersTime-consuming:API testing is time-consuming as it requires time to develop and execute test scripts for APIs.Limited documentation:Limited documentation of the API makes it difficult for the testers to understand how the API should behave in response to a particular input.10. List the differences between API Testing and Unit Testing.Here is theDifferences between API Testing and Unit Testing:ParametersAPI TestingUnit TestingDefinitionAPI testing is used to test the API to ensure that it meets the expectations for functionality, performance, security, etc.Unit Testingis used to test each unit and to ensure that each unit performs as expected or not.Carried outIt is carried out by QA team.It is carried out by developers.Type testingIt is mostly Black-box testing.It is White-box testing.Read More - API Testing Vs. Unit Testing.11. What is the difference between functional and non-functional API testing?Functional: Tests API logic and behavior (e.g., correct response data).Non-functional: Tests performance, security, and scalability (e.g., response time).12. What is an endpoint in API testing?An endpoint is a specific URL or route in an API that performs a function, accepting requests and returning responses.13. What is authentication in API testing, its purpose, and common methods?Authentication verifies the identity of a user or system to access protected API resources, ensuring security and preventing unauthorized access. Its purpose is to safeguard sensitive data and enforce access control. Common methods include: API Keys: Simple key in headers (e.g., X-API-Key: abc123).OAuth: Token-based delegation (e.g., OAuth 2.0 for user authorization).JWT: JSON Web Tokens for secure, stateless access.Basic Auth: Base64-encoded username:password in headers.14. What is a test case for API testing, and what does it include?A test case outlines steps to validate an API, including: Test ID, description, endpoint, method, request data, expected response, and status code.Read More - Test Case For API Testing.15. What is the difference between a GET and POST request?GET: Retrieves data, idempotent, parameters in URL.POST: Creates data, non-idempotent, data in request body.Read More - GET Vs. POST Request.Headers provide metadata about the request/response, such as content type, authorization, or caching instructions. 17. What is a mock API, and when is it used?A mock API simulates real API behavior for testing when the actual API is unavailable, using tools like WireMock or Mockoon.18. What is Negative Testing in the context of APIs?Negative testing validates API behavior with invalid inputs, such as incorrect data or unauthorized requests, to ensure proper error handling.Read More - Negative testing. 19. What is the purpose of authentication in API testing? API authentication is a combination of technology or process that verifies the identities of users who want access to an API. It involves the use of software protocol to verify identity of the requester before granting access to protected resources. Authentication verifies the identity of a user or system. Common methods include API keys, OAuth, JWT (JSON Web Tokens), and Basic Auth.Read More - API Authentication.20. What is a query parameter in API testing?Query parameters are key-value pairs in the URL to filter or modify a request, appended after a ?.These questions focus on practical automation, scripting, and intermediate concepts for testers.21. What is API mocking and Why is used ? API mocking is practice of simulating the behavior of an API endpoint during testing without actually invoking the real API. It is beneficial during the development stage. Request Headers: Provide metadata about the request, such as authentication (Authorization), content type (Content-Type), and caching control.Response Headers: Contain metadata about the server response, including status info, content type, and security policies.Here are the reasons why it is used: Mock APIs are useful when applications have dependencies with external APIs. Mock APIs are useful for testers to plan and validate test executions and for developers to do unit testing and identify the initial development stages. Mock APIs are useful in situations where the complete API needs to be made available for consumer testing before committing it to development stage22. What is the purpose of HTTP status codes in API testing?HTTP status codes indicate the result of an API request, helping testers understand whether the request was successful, failed, or requires action.Indicate Response Status:Shows if the request succeeded (200 OK), failed (400 Bad Request), or needs authentication (401 Unauthorized).Aid Debugging:Helps identify issues like server errors (500 Internal Server Error).Improve API Communication:Ensures clear communication between client and server.Enhance Automated Testing:Used in test validations to verify expected API behavior.Read More -HTTP status codes. 23. What is the purpose of the request and response body in API testing?The request body is used to send and receive data such as input parameters, or data to create/ update resources via the REST API. The response body is the data API send to the client.Request Body: Contains data sent by the client to the API (e.g., user credentials, form data, JSON payload). Example: Sending user details in a POST request.Response Body: Contains data returned by the API, such as success messages, error details, or requested information. Example: Returning user profile details in a GET request.Read More - HTTP Request & Responses.24. What is API security testing, and why is it important?API security testingis the process of testing the vulnerabilities in the API. This is done through penetration testing or manual scanning of the APIs. API security testing is important:Cloud-based applications rely on APIs to exchange data and interact with each other. Any security vulnerabilities can have far-reaching consequences.Online operations and businesses rely on APIs to integrate different systems and services. This causes potential security risks.Organizations rely on security solutions that are built for web apps to detect and secure from API threats. Such solutions cannot detect unique vulnerabilities and gaps in the APIs.Read More - Security Testing.25. What is Boundary Value Analysis (BVA) in API testing?BVA tests edge cases of input ranges (e.g., min/max values) to find defects at boundaries.Read More - BVA.26. What is API versioning, and why is it important in API testing? API versioning involves specifying a version number in the API endpoint or headers to ensure the backward compatibility. It manages the changes in the API over time. It ensures the correct version of the API is being tested. It ensures that the changes in the API do not break the existing client applications. 27. What tools could be used for API testing? Katalon: This is easy-to-use tool that supports REST, SOAP requests, and SSL client certificates. It also enables test import from Swagger, Postman, WADL, and WSDL. JMeter: Apache JMeter is an open-source, 100% Java application that is created for performance testing. This tool enables automatic working with CSV files. REST-assured: REST-assured is an open-source Java domain-specific language that enables testing REST services. It supports POST, GET, PUT, DELETE, PATCH, and HEAD requests. SoapUI: SoapUI is an automated testing tools for Soap and REST APIs. It is an open-source tool that allows to create tests effortlessly with drag and drop, point-and-click. Postman: Postman is an application that is used for API testing. It is a standalone platform that is used to build, test, design, modify, and document APIs. 28. When writing API document, what must be considered? Key Considerations When Writing API DocumentationClear API Overview: Explain API purpose, features, and use cases.Endpoint Details: List all endpoints (GET /users, POST /orders) with descriptions.Request & Response Format: Provide JSON/XML examples with expected parameters.Authentication & Security: Explain how to use API keys, OAuth, or JWT for access.Error Handling: Document status codes (400 Bad Request, 500 Internal Server Error) and responses.Rate Limits & Throttling: Define request limits and handling of 429 Too Many Requests.29: How do you test GraphQL APIs compared to REST APIs?GraphQL Testing: Focuses on queries, mutations, and schema validation. Use tools like Postman or Apollo Client to send queries (e.g., { user(id: 1) { name } }) and verify response structures. Test for over-fetching/under-fetching and schema compliance.REST Testing: Tests fixed endpoints (e.g., GET /users/1) with predictable responses. Focus on HTTP methods and status codes.30. How do you ensure the security of APIs in your tests? To secure APIs during testing, follow these best practices:Authentication & Authorization: Verify security mechanisms like: OAuth 2.0, JWT (JSON Web Token), API Keys, Basic AuthInput Validation & Sanitization: Prevent SQL Injection andXSS attacks by validating input data.Secure Headers & Encryption: Use HTTPS (TLS/SSL) and set security headers like: Strict-Transport-Security, Content-Security-PolicyRate Limiting & Throttling: Prevent DDoS attacks by limiting API request rates.Token Expiry & Refresh Mechanism: Ensure JWTs expire and implement refresh tokens securely.Access Control (RBAC): Restrict API access based on user roles and permissions.Logging & Monitoring: Track API activity using logging tools for anomaly detection.Automated Security Testing: Use tools likeOWASPZAP, Postman Security Tests, Burp Suite to identify vulnerabilities.31. What are the different types of error responses in API testing? Error responses in API testing are categorized using HTTP status codes to indicate different types of failures. Here are the different types: Validation errors: Validation errors occur when the API request does not meet the validation criteria defined by the API. HTTP error status codes: These can be client-side status codes or server-side status codes. There are standard HTTP status codes like 4xx or 5xx that indicate errors in the API request or response. Custom error messages: These are the customer error messages that are returned by the API in case of errors or exceptions. 32. How do you handle error responses in your API tests? Handling Error Responses in API Testing:Validate HTTP Status Codes: Ensure correct 4xx (client errors) and 5xx (server errors) are returned.Check Error Messages: Verify meaningful, user-friendly error messages.Test Edge Cases: Send invalid inputs, missing parameters, and malformed requests.Implement Retry Mechanism: Retry requests for temporary failures (5xx errors).Monitor & Log Errors: Use tools like Postman, JMeter to track failures.Respect Rate Limits (429): Implement delays or follow Retry-After headers.Secure Error Handling: Ensure no sensitive data is leaked in error responses.33. What is input validation? Input validation is the process of verifying user input to ensure it is correct, secure, and meets expected criteria before processing. It helps prevent errors, data corruption, and security vulnerabilities like SQL Injection and XSS attacks. Input validation can be performed on both client-side and server-side for better security and data integrity. 34. Why input validation is important in API testing? Input validation is important in API testing as it prevents malicious data from entering the system. It is important to prevent injection attacks and other malicious activities. This can be done manually or using automated tools. 35. What is cross-site request forgery (CSRF)? Cross-site request forgery is an attack that forces user top execute unwanted action in the web application in which they are authorized to perform actions. This attack exploits the trust the web application has in an authenticated user. Read More - Cross-site request forgery.API Testing Interview Questions for Experienced These questions focus on Experienced concepts to help build a strong understanding of API testing with advance concepts.36. How CSRF can be prevented in API testing? CSRF tokens: CSRF can be prevented by using CSRF tokens. These are the random tokens that need to be unique per user session and should be of large random value to make it difficult to guess. SameSite cookies: SameSite is a browser security mechanism that determines when a website's cookies are included in the requests originating from other websites. Referer-based validation: HTTP referer header can be used to defend against CSRF attacks by verifying the request originated from the application's own domain. 37. What is API contract testing? API contract testing is a type of testing that aims to monitor the API conversation that takes place between the API consumer and the API producer. It can be performed using tools like Postman, Swagger, etc. 38. Why API contract testing important? API contract testing ensures that APIs follow a defined structure (contract) between the client and server, preventing breaking changes.Ensures Consistency: Verifies API responses match the expected format (JSON, XML).Detects Breaking Changes: Prevents issues when updating APIs.Improves Reliability: Ensures backward compatibility across services.Speeds Up Development: Allows frontend and backend teams to work independently.Enhances API Stability: Reduces failures in integrations by enforcing standards. 39. What is API performance testing? API Performance Testing evaluates an APIs speed, scalability, and reliability under different loads and conditions. It ensures that the API responds efficiently and remains stable under varying traffic levels.Response Time: Measures how quickly the API responds.Throughput: Checks the number of requests handled per second.Load Testing: Simulates high user traffic to check performance.Stress Testing: Determines API behavior under extreme conditions.Spike Testing: Evaluates API response to sudden traffic surges.40. Why API performance testing important? API performance testing ensures that an API is fast, scalable, and reliable under different workloads. API performance testing helps to improve the API's overall performance and stability. It helps to provide insight into the API's overall performance thus helping to identify the areas of strength and weakness. It helps to identify the performance issues and determine the impact of changes. It helps to ensure that the API is flexible and can handle the demands of the real world. 41. What is API monitoring? API monitoring is the process of continuously tracking an APIs availability, performance, and functionality to detect issues in real-time.Uptime & Downtime: Ensures the API is always accessible.Response Time: Tracks API speed and latency.Error Rates: Detects failed requests or unexpected responses.Traffic & Load: Monitors API usage patterns.Security Threats: Identifies potential vulnerabilities.Read More - API monitoring.42. Why API monitoring important? API monitoring ensures that APIs remain reliable, fast, and secure by tracking their performance in real-time. They provide measurements of how long a routine takes to execute, how often it is called, and how much of total time is spent in executing the transaction. It is important to ensure availability, performance, and security of the APIs. It helps to guarantee a dependable and effective user experience. It helps to track the availability of the critical APIs. 43. What is API virtualization? API virtualization is the process of using a tool that creates a virtual copy of the API mirroring all of the specifications of the production API and using this virtual copy for testing. Allows early testing before the actual API is built.Reduces dependency on live systems by simulating unavailable APIs.Improves development speed by enabling teams to work in parallel.Supports performance testing under various conditions.44. Why API virtualization is used in API testing? It is used in API testing as it allows for testing much earlier in the development process, removing the key bottlenecks that would otherwise delay production. It is helpful to isolate dependencies, simulate responses, and ensure consistent behavior. 45. What is the purpose of API documentation in API testing? API documentation is a technical document that describes the API in detail. It includes instructions on how to effectively use and integrate API and provides updates regarding API's lifecycle such as new versions. It is primary resource explaining what is possible with the API and how to get started with the API. 46. How do you prioritize API test cases for regression testing? Critical Functionality First: Focus on core APIs that impact business operations.Frequently Used APIs: Test APIs that are accessed most often by users.Recently Changed or Updated APIs: Prioritize APIs modified in recent updates.APIs with High Bug History: Retest APIs that previously had defects.Security and Authentication APIs: Ensure no vulnerabilities are introduced.Performance-Critical APIs: Validate APIs affecting system speed and scalability.Dependent APIs: Test APIs that interact with multiple services or components. 47. How to handle versioning in API testing? Test Multiple Versions: Ensure backward compatibility by testing old and new API versions.Use Version-Specific Endpoints: Validate API calls with versioned URLs (e.g., /v1/users vs. /v2/users).Check Header-Based Versioning: Verify API responses when versioning is done via headers (Accept: application/vnd.api.v2+json).Compare Responses: Ensure consistency between different versions while testing improvements.Validate Deprecated Features: Test error handling for removed or updated functionalities.Automate Version-Based Tests: Maintain separate test suites for each API version. 48. List the challenges faced while performing API testing. Building irrelevant tests: Building tests without considering how the APIs will be consumed may be quicker in short term. Initial setup of API testing: Setting up an API testing setup requires certain level of expertise and dedication among the team members. Not including API dependencies: Failure to include API dependencies as a part of API testing strategy can be a critical API testing challenge. Not validating data: It might be possible that API tests pass successfully but APIs are not returning the correct data in their responses. 49. What are the different bugs that can be found in API testing? Different Bugs Found in API Testing: Duplicate or missing functionality. Improper messaging. Multi-threading issues. Security and performance issues. Reliability issues. 50. Is it possible to hack API while testing? Yes, it is possible to hack API while testing as requests are being sent over the internet which mostly follows HTTP protocol which is a text-based protocol. Hence, it is important to perform security testing of the APIs to ensure safer systems. Common API Vulnerabilities Exploited:Broken Authentication: Weak tokens or missing authentication.SQL Injection: Malicious queries to access or modify databases.Cross-Site Scripting (XSS): Injecting scripts into API responses.Insecure Endpoints: Exposed sensitive data due to improper access controls.Rate Limiting Bypass: Exploiting APIs by sending excessive requests.51. How do you perform API Load Testing? API load testingis a type of performance testing that is done to check the application's capability to perform under various user loads. It is done by simulating many users hitting the API at the same time to identify if the application is capable of handling the load by maintaining the consistency in the response times and not impacting the functionality. Read More - API Load Testing.52. What is the test environment of API? The API test environment is a setup where APIs are tested under controlled conditions before deployment. It includes:Server & Database: A dedicated test server and database to mimic production.API Endpoints: Staging or sandbox URLs separate from live systems.Authentication & Security Setup: API keys, OAuth, JWT tokens for access control.Mock Services & Virtualization: Simulated dependencies for testing APIs in isolation.Testing Tools: Tools like Postman, JMeter, SoapUI for automation and performance testing.Logging & Monitoring: Tracks API requests, responses, and failures.Practical API Testing Interview QuestionsBelow are intermediate-level, hands-on API testing questions with detailed solutions, code snippets, and outputs, tailored for SDETs and testers preparing for interviews. 1. Write a RestAssured script to validate a POST API for creating a user, including response schema and status code.Steps:Set up RestAssured with the base URI.Send a POST request with a JSON payload containing name and job.Validate the status code and response fields.Use a JSON schema to ensure response structure.Extract and print the user ID. Java import io.restassured.RestAssured;import io.restassured.module.jsv.JsonSchemaValidator;import io.restassured.response.Response;import org.testng.Assert;import org.testng.annotations.Test; public class PostApiTest { @Test public void testPostUser() { // Set base URI RestAssured.baseURI = " ; // Define request payload String requestBody = "{\"name\": \"Alice\", \"job\": \"Engineer\"}"; // Send POST request and validate Response response = RestAssured.given() .header("Content-Type", "application/json") .body(requestBody) .when() .post("/users") .then() .statusCode(201) .body("name", org.hamcrest.Matchers.equalTo("Alice")) .body("job", org.hamcrest.Matchers.equalTo("Engineer")) .body(JsonSchemaValidator.matchesJsonSchemaInClasspath("user-schema.json")) .extract() .response(); // Extract and print user ID String userId = response.jsonPath().getString("id"); System.out.println("Created User ID: " + userId); // Additional assertion for non-null ID Assert.assertNotNull(userId, "User ID is null"); }}Schema File (user-schema.json in src/test/resources):{ "type": "object", "properties": { "name": { "type": "string" }, "job": { "type": "string" }, "id": { "type": "string" }, "createdAt": { "type": "string" } }, "required": ["name", "job", "id", "createdAt"]}Output: Created User ID: 123Test passes if status code is 201, response fields match, and schema is valid.2. Write a Postman script to chain a GET and POST request, passing data dynamically.Steps:Send a GET request to retrieve user data.Store the user ID in an environment variable.Use the ID in a POST request payload.Validate the POST response.Code (Postman Pre-request Script for GET):pm.sendRequest({ url: ' , method: 'GET'}, function (err, response) { if (!err) { pm.environment.set("userId", response.json().data.id); }});POST Request Setup:URL: POSTBody (raw JSON): {"title": "New Post", "userId": {{userId}}}Test Script (for POST):pm.test("POST request with chained userId", function () { pm.response.to.have.status(201); var jsonData = pm.response.json(); pm.expect(jsonData.userId).to.eql(parseInt(pm.environment.get("userId"))); console.log("Post created with userId: " + jsonData.userId);});Output: Post created with userId: 2Test passes if status is 201 and userId matches.Write a RestAssured script to test a DELETE API and verify resource removal.Steps:Send a DELETE request.Validate the status code (204).Send a GET request to confirm the resource is gone (expecting 404 or empty response). Java import io.restassured.RestAssured;import org.testng.annotations.Test; public class DeleteApiTest { @Test public void testDeleteUser() { RestAssured.baseURI = " ; // Send DELETE request RestAssured.given() .when() .delete("/users/2") .then() .statusCode(204) .log().all(); // Verify resource is deleted (ReqRes is mock, so GET may still return data) RestAssured.given() .when() .get("/users/2") .then() .statusCode(200); // Mock API limitation, ideally expect 404 }}Output: [DELETE Response]: 204 No ContentTest passes if DELETE returns 204. Note: ReqRes is a mock API, so GET may not reflect deletion; in real scenarios, expect 404.4. Write a Postman script to validate API response time and handle rate limiting.Steps:Send a GET request and validate response time.Simulate rate limiting with repeated requests in a loop.Check for 429 status (mock API may not enforce limits).Test Script:pm.test("Response time is under 500ms", function () { pm.expect(pm.response.responseTime).to.be.below(500); console.log("Response time: " + pm.response.responseTime + "ms");});// Simulate rate limiting (mock API may not return 429)for (let i = 0; i < 60; i++) { pm.sendRequest({ url: ' method: 'GET' }, function (err, res) { if (res && res.code === 429) { pm.test("Rate limit detected", function () { pm.expect(res.code).to.equal(429); }); } });}Output: Response time: 230msTest passes if response time is