A P I

1. What is API
   1. API is stands for Application Programming Interface
   2. It is a connector between 2 services
   3. It is an interface between different software program or service
   4. It is a connector between BE and FE
2. What is API testing
   1. Testing the product about what work and how the system behave when we do something
   2. Testing to understand how the BE work to see what the FE show
3. How to approach API testing using exploratory.
   1. Know about the type of API, its implementation. Read the document before starting
   2. What Exploratory testing
      1. Test case are not created in advance but rather checking the system directly on the fly
      2. All is about discovery, investigation, learning by doing.
      3. For Exploratory, the functionalities are checked in an ad-hoc manner (without preparation. For Normal testing, functionalities are checked in a structural manner.
      4. Create initial explore plan
      5. HTTP codes and error codes also negative tc
      6. Understand the endpoint, so you know what api request and its response to test
4. API test case

* HTTP status code.
* Valid Response payload
* Response Time
* Parse the Response data
* Valid Response headers.
* Negative Test Cases response
* Security Related Test Cases
* Schema validation
* Field Type
* Mandatory Fields
* Chaining Request verification.
* Verification of APIs with Data parameters.
* End to End CRUD flows
* Database Integrity Test Cases
* File Upload Test Case

1. How to discover what request happen on the browser
   1. User inspect element - Network - check on the response Name - RIght click - Copy as Curl
   2. User charles proxy for Mac
   3. Use Fiddler for Win
2. Skills needed for API testing
   1. For API testing using postman
      1. Learn JS a bit
      2. Assertion and Schema validation concept
      3. Chai JS basic
      4. BDD = Behavior Driven Development (kontrak). Testing untuk memastikan apa yang diharapkan stakeholder bekerja sesuai harapan mereka.
         1. ***Context (Starting State), Event (*Apa yang *User* lakukan), *Outcome* (Apa yang diharapkan)**
   2. Tools
      1. Chrome dev tool (inspect element)
      2. Jenkin CI/CD tools
      3. Docker/Kubernekes
      4. Web Proxy tools
         1. Fiddler
         2. Charles
   3. Json/XML Parsing
   4. HTTP Method basics
   5. Able to make test scenario, test case, test suit, efficient in manual testing
   6. Exploratory testing
   7. Request, Response, Header, Auth, Cookies, Session, Chaining APIS
   8. API types: REST, SOAP, RPC, HTTP status code
   9. Database skills at least SQL basics
3. Common API bugs:
   1. Wrong error codes
   2. Missing Keys
   3. Empty post or Update are not handled properly
4. API testcase
   1. Validate the keys with the Min. and Max. range of APIs (length of input for ex)
   2. Key verification. If we have JSON, XML APIs we should verify it’s all the keys are coming
   3. Have a test case to do XML, JSON Schema validation
   4. Verify the Parse the Response data
   5. Verify the JSON Schema validation, verify the field type, verify the mandatory field
   6. Valid response header and negative test cases response
   7. Verify how the APIs error codes handled
   8. Verify the response HTTP status code
   9. Valid response payload
   10. Chaining request verification
   11. Verification of APIs with Data parameters
   12. End to End CRUD flows
   13. Database integrity test
   14. File upload test case
5. API security testing using burp suite
   1. Secure call with HTTPs
   2. Avoid using basic Auth (Use advance like JWT, Oauth1,2)
   3. Auth token should have Expiry
   4. Input validation test (sanitize the data from APIs before putting to DB)
   5. User privilege escalation test: For example, access/refresh token of one user shouldn’t be accepted by another user
   6. Validate user input to avoid common vulnerabilities
   7. Use of Quotas and Throttling: Limit requests (Throttling) to avoid DDos / brute force attacks
   8. Detect Insecure API Calls With Sniffers
   9. Use HSTS header with SSL to avoid SSL strip attack
   10. Validate content-type on request Accept header
   11. Don’t use any sensitive data (credentials, password, security, tokens, or API keys) in the URL, but use standard Auth header
   12. <https://owasp.org/www-project-api-security/>
   13. <https://github.com/shieldfy/API-Security-Checklist/>
6. Types of performance testing API
   1. Baseline testing: how system performs under expected or normal load and creates a baseline with which other types of tests can be compared
   2. Stress testing: Find the load volume where the system actually breaks or is close to breaking
   3. Load testing: In the increasing the load see how the system behaves under higher load
   4. Soak testing: run load test or even baseline test over a long period of time and see how the target environment handles system resource and if it work properly
   5. Scalability testing: to check weather your system scales appropriately to changing load
7. Why do performance testing?
   1. To determine the overall performance of a system under specific circumstances
   2. Two types:
      1. Functional test: To ensure that the API returns the desired output for a given input
      2. Load test: In this multiple clients hit the same server at the same time, and number of clients can drastically affect the behavior of the API
   3. Non optimized code is making a request to a database that is slow in responding
   4. Processing of payload (request and response bodies to multiple clients)
   5. Check the impact of a large amount of XML / JSON processing on the server side

**Yang dilakukan API Testing :**

- memverifikasi fungsionalitas dan mengekspos kegagalan.

- memverifikasi panggilan dengan dua atau lebih parameter pada URL.

- memverifikasi behavior API dari eksternal environment.

- memverifikasi sequence of API calls.

- memverifikasi return value berdasarkan condition input.

- memverikasi API tidak mengembalikan nilai apapun.

- memverifikasi API triggers some other event or calls another API.

- memverifikasi struktur data.

- memverifikasi status kode HTTP benar.

- memverifikasi response JSON.

- memverifikasi response header.

- memverifikasi basic performance sanity.

**Tujuan API Testing :**

- menguji functionality, reliability, performance, and security dari programming interfaces.

- mencegah ditemukannya bug dari expected behavior software yang menyimpang.

- memastikan sesuai dengan spesifikasi requirement.

- mencegah regresi antara code merges and releases.

- continuous testing memastikan dapat diakses ketika software release.

**Mengapa harus melakukan API testing?**

- Language independent : independen / tidak terikat pada Bahasa pemrograman yang digunakan untuk mengembangkan aplikasi, karena menguji pertukaran data melalui XML dan JSON.

- GUI independent : independen / tidak terikat pada Graphical User Interface (GUI) atau yang sering disebut interface, karena menguji fungsionalitas dan seberapa kuat dari beban yang diberikan.

- Improved test coverage : meningkatkan test coverage dengan automation test termasuk positive test & negative test.

- Enables faster release : memungkinkan release aplikasi lebih cepat.

Bagaimana melakukan API testing?

- API specification review : mendokumentasikan requirement API testing yang dapat menjawab :

1. Apa tujuan API?
2. Bagaimana workflow aplikasi?
3. Integrasi mana yang didukung oleh API?
4. Apa sajakah fitur API?

- Setting up test environment : menyiapkan set parameter yang diperlukan, seperti konfigurasi database dan server untuk requirement aplikasi.

- Combining application data : menggabungkan data aplikasi dengan test API.

- Deciding what API for : menentukan tipe API testing yang akan dilakukan.

- Test execution & reporting : melakukan pengujian, mendokumentasikan pengujian dan hasil pengujian.

**Tipe API Testing :**

- Functionality Testing : menguji functionalitas API .

- Validation Testing : menguji API dari verifikasi aspek produk, behaviour, dan efisiensi.

- Reliability Testing : menguji koneksi dan hasil API yang konsisten.

- Load Testing : menguji respon API dibawah beban kondisi normal dan tinggi (bagian dari Performance Testing).

- Exploratory Testing : menguji API tidak sesuai rules aplikasi.

- Security Testing : menguji API dari external threats.

- Adhoc Testing : menguji API tanpa rencana dan dokumentasi.

- Usability Testing : menguji API dari segi user friendly.

- Dynamic Testing : menguji API dari analisis behaviour code, melibatkan nilai input dan output. Bagian dari validasi dan verifikasi, Tekniknya terdiri dari Functional Testing dan Nonfunctional Testing. Levelnya terdiri dari Unit Testing, Integration Testing, System Testing, Dan Acceptance Testing.

- Performance Testing : menguji API dari beban dan kecepatan internet.

- Data Driven Testing : menguji API dari data output yang dihasilkan.

- Regression Testing : menguji API dari keseluruhan aplikasi terhadap perubahan code aplikasi.

- UI Testing : menguji API dari tampilannya.

- Penetration Testing : menguji API dari perspektif penyerang keamanan dan integritas system, aplikasi, jaringan, dan database. Termasuk teknik Black Box Testing.

- Fuzz Testing : menguji API dari skenario terburuk.

- End to end Testing : menguji API dari flow aplikasi dari awal hingga akhir apakah sudah sesuai ekspektasi.