Automation Test Examination

Name: Rothy Yorn

1. Automation testing is a software testing technique where special tools and scripts are used to automatically execute test cases instead of doing them manually. It helps verify that the software behaves as expected
2. API Testing Tools are used to check if APIs work correctly, return the right data
   - **Postman:** Send requests, check responses, automate tests
   - **SoapUI:** Functional, security, and load testing for REST & SOAP APIs
   - **Insomnia:** Lightweight tool for sending REST requests
   - **JMeter:** Performance/load testing for APIs
   - **Rest Assured:** Java library for automated API tests
   - **Katalon Studio:** GUI + scripting for API and web testing
3. API (Application Programming Interface) is a contract that defines how two software components interact. It provides endpoints, methods, and data formats so that applications can exchange information in a structured and consistent way. For example, when a mobile app fetches weather data from a server, it does so through an API
4. I would automate a test when it is repetitive, time-consuming to run manually, or critical to the business, such as regression or smoke tests. I also consider automation when the application is stable and the tests need to run frequently across different environments or data sets
5. The default success code in HTTP is 200, which means the request was successfully received, understood, and processed by the server. Other 2xx codes like 201 Created or 204 No Content also indicate success, but 200 is the standard default
6. Test Scenario API [Login]

ID	Test Cases	Test data	Expected
TC001	Login Successful	Username: Demo Password: Demo	Should be **Login Successful**
TC002	Login Invalid Username	Username: Invalid Password: Demo	Should be response message **Invalid username or password**
TC003	Login Invalid Password	Username: Demo Password: invalid	Should be response message **Invalid username or password**
TC004	Login Invalid Username And Password	Username: invalid Password: invalid	Should be response message **Invalid username or password**
TC005	Login Missing Username Field	Password: Demo	Should be response message **Username is required**
TC006	Login Missing Password Field	Password: Demo	Should be response message **Password is required**
TC007	Login With Long Username/Password	Username: 500 chars Password: 500 chars	Should be response message **Input exceeds maximum length**
TC008	Login With Blank Body	Bank Request Body	Should be response message **Username and password are required**

TC009	Login With Expired Password	Username: Demo Password: Old Password	Should be response message **Password expired**
TC010	Login With User status Inactive	Username: user inactive Password: Demo	Should be response message **User account is inactive**
TC011	Login SQL Injection Attempt	Username: “demo” OR ‘1’=’1’ Password: Demo	Should be response message **Invalid username or password**
TC012	Login With Wrong Content-Type header	Invalid content-type in header	Should be request **Unsupported content type**
TC013	Login Invalid Username or Password Multiple Times	Try invalid username or password 5times	The system should be **Block the user**
TC014	Login Rate Limit	Send 100login request in 1min	The system should lock the request and response message **Too many login attempts, please try again**
TC015	Login Attempt Concurrent Request	Send multiple login request in parallel with same credentials	API should handle correctly return the consistent response
TC016	Login When Server Issue (Down)	Try to login when server issue	Should be response **Server temporarily unavailable**

7. Load Testing is a type of performance testing that checks how a system behaves under expected or higher-than-normal user loads It helps determine whether the application can handle concurrent users, transactions, or data volume without slowing down or failing
8. Some disadvantages of manual testing
    - **Takes a lot of time:** Testing everything by hand is slow
    - **Cannot test everything:** Hard to check all cases, especially for big apps
    - **Repetitive work:** Doing the same tests again and again can be boring
    - **Slow for repeated testing:** Repeating tests for updates (regression) is inefficient
    - **Feedback is slower** : Bugs may be found late, causing delays
    - **Costly for big projects**: Need more testers, which costs more
    - **Hard to reproduce bugs**: Sometimes it’s difficult to follow the exact steps to find a bug again