Part 2

1. *Explain the difference, in databases, between 'Having' and 'where' when it comes to a query. Develop one example for each one of this two cases and point out the difference.*

   In short, both "Where" and "Having" are used in SQL queries to filter records, the main difference lies in the fact that "Where" is based on individual rows, while "Having" is used to filter the results of grouping functions. Also, the "Where" clause should be before a Group by clause, whereas the "Having" clause is applied after the grouping has occurred.

   Example using Where:
   "SELECT name, code, age FROM Student WHERE age > 18; "
   (Query using Where clause retrieve the names, codes, and ages of students where the age is greater than 18)

   Example using Having:
   "SELECT name, code, AVG(GPA) AS averageGPA FROM Student GROUP BY class HAVING AVG(GPA) > 3.5;"
   (Query using Having clause to get average GPA for classes with an average GPA greater than 3.5)

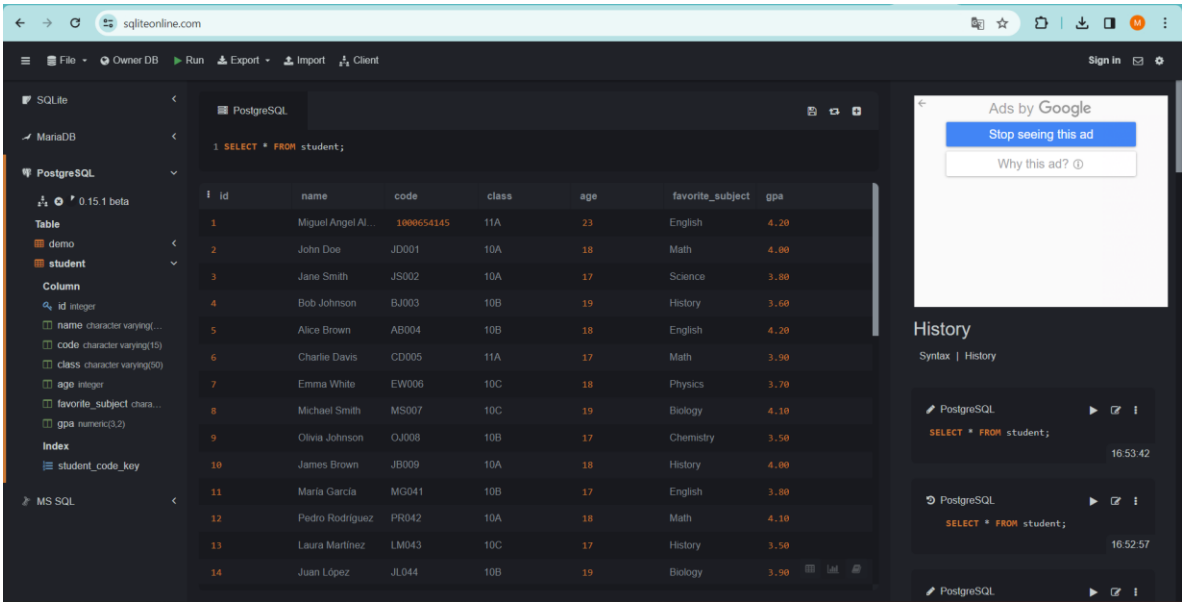
2. *Write a query to create a data table 'Student' with the following attributes in it: 'Name', 'Code', 'Class', 'Age', 'Favorite Subject', 'GPA' (5.0 scale).*

   "CREATE TABLE Student(ID SERIAL PRIMARY KEY, name VARCHAR(200) NOT NULL, code VARCHAR(15) NOT NULL UNIQUE, class VARCHAR(50), age INT NOT NULL, favorite_subject VARCHAR(100), GPA DECIMAL(3,2) NOT NULL);"

3. *Insert at least 40 records in the last table with close to real data.*

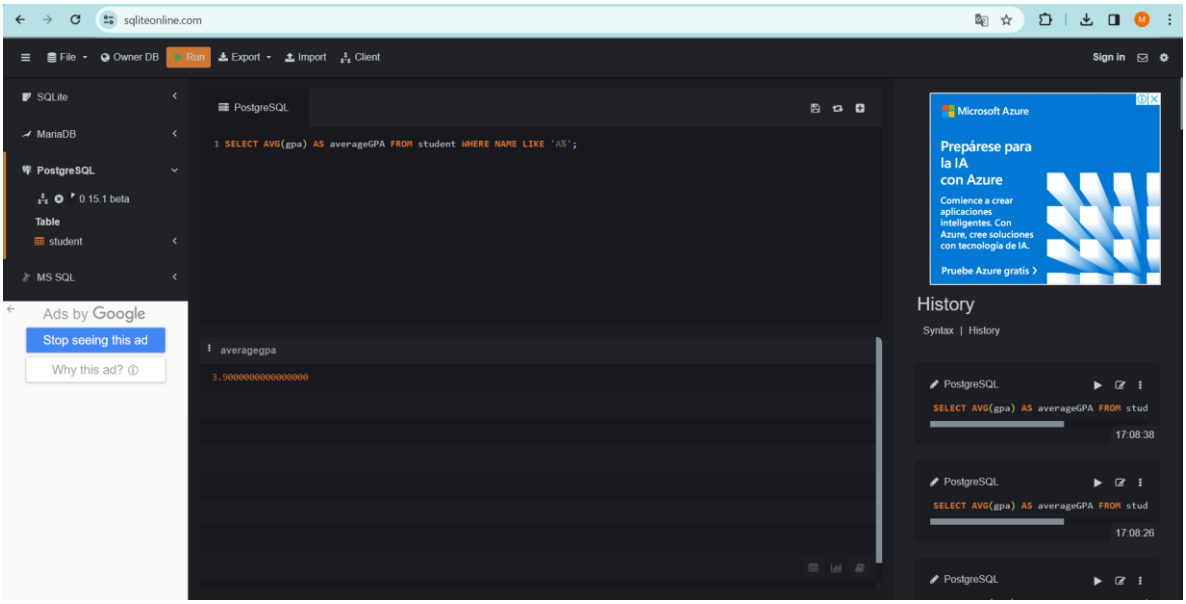
   "INSERT INTO Student(name, code, class, age, favorite_subject, GPA) VALUES
   ('Miguel Angel Alvarez', '1000654145', '11A', 23, 'English', 4.2),
   ('John Doe', 'JD001', '10A', 18, 'Math', 4.0),
   ('Jane Smith', 'JS002', '10A', 17, 'Science', 3.8),
   ('Bob Johnson', 'BJ003', '10B', 19, 'History', 3.6),
   ('Alice Brown', 'AB004', '10B', 18, 'English', 4.2),
   ('Charlie Davis', 'CD005', '11A', 17, 'Math', 3.9),
   ('Emma White', 'EW006', '10C', 18, 'Physics', 3.7),
   ('Michael Smith', 'MS007', '10C', 19, 'Biology', 4.1),
   ('Olivia Johnson', 'OJ008', '10B', 17, 'Chemistry', 3.5),
   ('James Brown', 'JB009', '10A', 18, 'History', 4.0),
   ('María García', 'MG041', '10B', 17, 'English', 3.8),
   ('Pedro Rodríguez', 'PR042', '10A', 18, 'Math', 4.1),
   ('Laura Martínez', 'LM043', '10C', 17, 'History', 3.5),
   ('Juan López', 'JL044', '10B', 19, 'Biology', 3.9),
   ('Ana Sánchez', 'AS045', '10A', 18, 'History', 4.2),
   ('Carlos González', 'CG046', '10C', 17, 'Chemistry', 3.7),

('Isabel Fernández', 'IF047', '10B', 19, 'Physics', 4.0),
('Antonio Ruiz', 'AR048', '10A', 18, 'Chemistry', 3.6),
('Sofía Díaz', 'SD049', '10C', 17, 'English', 4.0),
('Miguel Torres', 'MT050', '10B', 19, 'Chemistry', 3.8),
('Giuseppe Moretti', 'GM051', '10B', 17, 'Math', 3.8),
('Isabella Romano', 'IR052', '10A', 18, 'Science', 4.1),
('Alessia Ferrari', 'AF053', '10C', 17, 'History', 3.5),
('Marco Conti', 'MC054', '10B', 19, 'Biology', 3.9),
('Sophia Ricci', 'SR055', '10A', 18, 'Art', 4.2),
('Luca Rossini', 'LR056', '10C', 17, 'Chemistry', 3.7),
('Valentina De Luca', 'VDL057', '10B', 19, 'Physics', 4.0),
('Leonardo Bianchi', 'LB058', '10A', 18, 'Physical Education', 3.6),
('Aria Russo', 'AR059', '10C', 17, 'English', 4.0),
('Lorenzo Greco', 'LG060', '10B', 19, 'Art', 3.8),
('Gabriel Silva', 'GS061', '10B', 17, 'Math', 4.8),
('Sophia Oliveira', 'SO062', '10A', 18, 'Science', 4.1),
('Lucas Pereira', 'LP063', '10C', 17, 'History', 2.5),
('Isabella Costa', 'IC064', '10B', 19, 'Biology', 3.9),
('Enzo Santos', 'ES065', '10A', 18, 'Art', 4.2),
('Lara Almeida', 'LA066', '10C', 17, 'Chemistry', 2.7),
('Matheus Rocha', 'MR067', '10B', 19, 'Physics', 4.0),
('Julia Cardoso', 'JC068', '10A', 18, 'Physical Education', 2.6),
('Pedro Lima', 'PL069', '10C', 17, 'English', 4.0),
('Mariana Santos', 'MS070', '10B', 19, 'Math', 2.8);



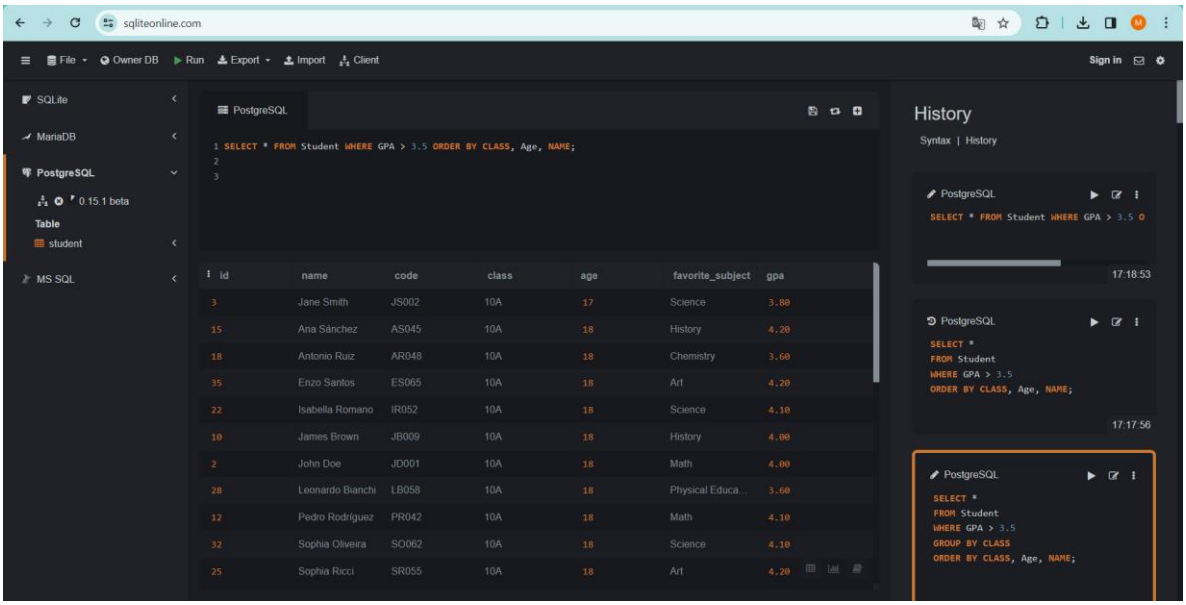
4. *Write a query to get the average of the GPA from all the students whose name starts with 'A'.*

   "SELECT AVG(GPA) AS averageGPA FROM Student WHERE name LIKE 'A%';"



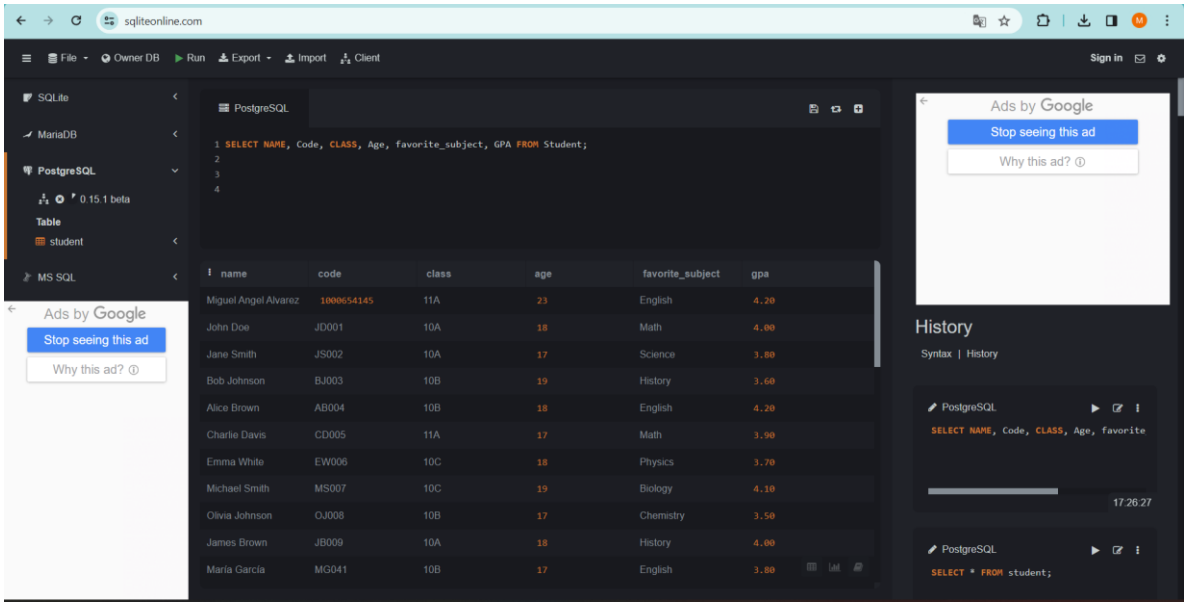
5. *Write a query to get the list of students that are in the same class, have a GPA higher than 3.5/5.0 and order them by Age and Name.*

   "SELECT * FROM Student WHERE GPA > 3.5 ORDER BY class, age, name;"



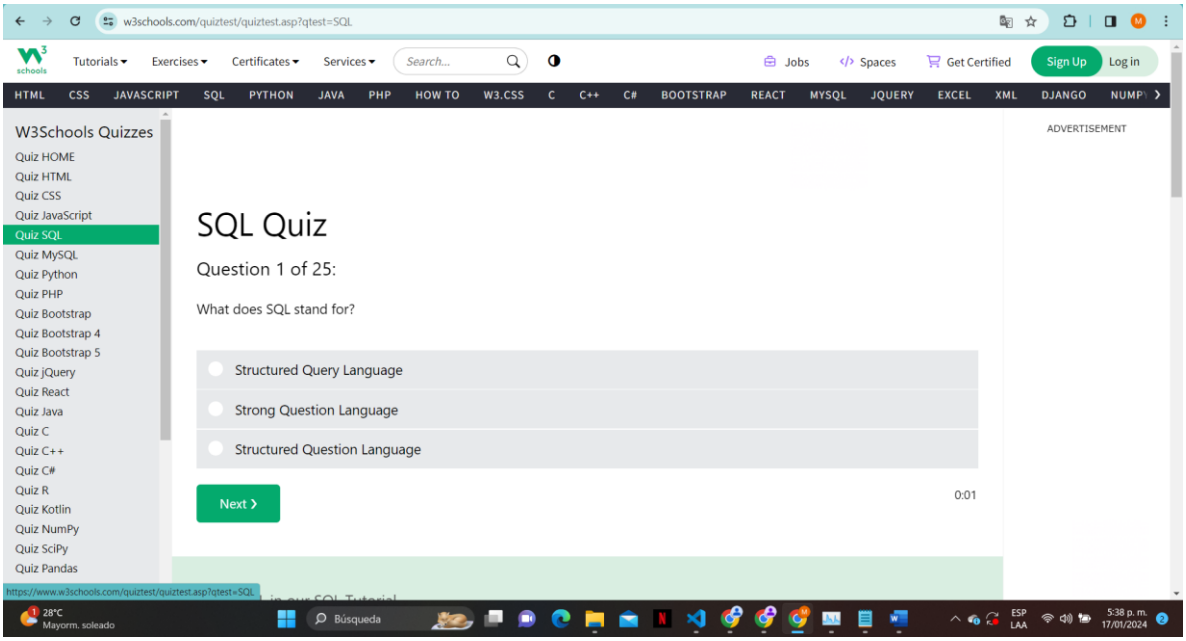
6. *Write a query to get the list of all the students with 'Name', 'Code', 'Class', 'Age', 'Favorite Subject', 'GPA'.*

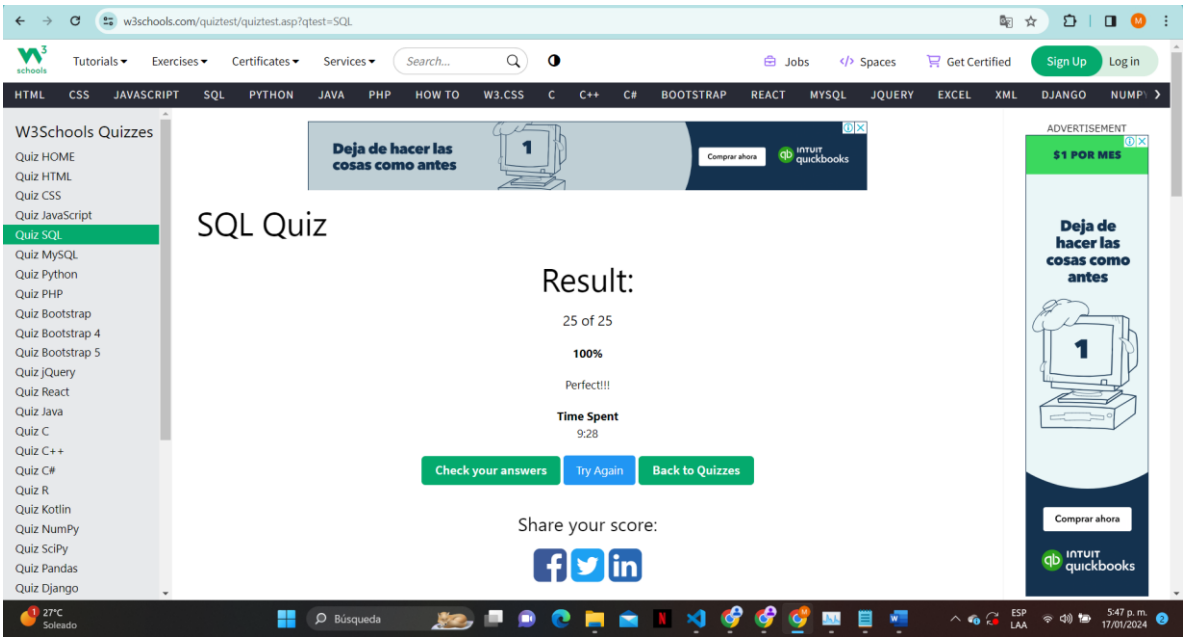
   "SELECT name, code, class, age, favorite_subject, GPA FROM Student;"



(It is not correct to use "SELECT * FROM Student" because I have an ID column)
Workspace: https://sqliteonline.com/

7. *Take the following 25 question quiz about SQL, please include a screenshot about the results and time it took to take the test.*
   http://www.w3schools.com/quiztest/quiztest.asp?qtest=SQL





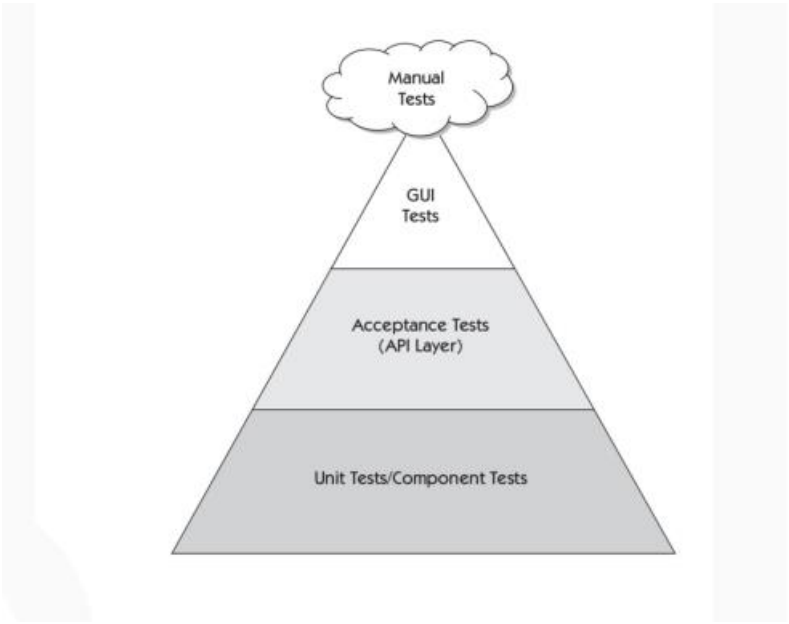
Part 3

1. *What is the difference between a unit test, an acceptance test, an integration test and an end-to-end test?*

The distinction among each type of test is closely related to the layers, size of the architecture or its modularity, as well as the roles of individuals responsible for conducting test. First, a unit test is focused on individual components (such as functions or methods) and is typically carried out by a developer during the software development cycle. On the other hand, an acceptance test centers around the functional requirements requested by clients, and the person in charge is typically the end user. Otherwise, an integration test is involves testing two or more components and how they are interconnected. In this case, the responsibility may fall on both developers and testers, or at least one of them. Finally, an end-to-end test evaluates a transaction or process from start to finish within an application, often referred to as a use case. The responsibility for E2E test lies entirely with the tester.

2. *Could you explain Cohn's automation pyramid?*

Cohn’s automation pyramid suggests that the majority of test should be focused on the lower layers of the pyramid. Additionally, more scripts need to be created for the lower level to automate the test. For example, unit tests or component tests require more automated test (and, in general, more tests) than GUI tests. Furthermore, Manual tests are a crucial part of the pyramid because they express that it is not possible to automate 100% of the test, and the existence of manual testing is always necessary.



3. *Could you explain the difference between a black box testing and a white box testing?*

   The main difference between black box and white box testing is relate to their focus. Black box testing does not concentrate on the code of an application; instead, it assesses functionalities, behavior, and features. Moreover, the objective is to ensure that the software functions as expected from the user's perspective. On the other hand, white box testing involves the entire structure of the software, including code lines, architecture, and components. Its purpose is to verify the integrity and adherence to good practices within the application code.

4. *What is the purpose of an exploratory test and when is it useful to run them?*

   The purpose of an exploratory test is to conduct testing while actively interacting with and gaining knowledge about the software application, hence its name, which referring to explore. During exploratory testing, testers aim to uncover bugs, investigate their causes, and sometimes identify related issues. It can often be the initial type of test performed by testers when the development team releases their changes. Unlike scripted tests, exploratory testing needs to be more spontaneous, and it relies on the tester's domain knowledge to navigate through the application effectively.

5. *Mention at least 5 test design techniques and explain them briefly.*

a) **Use Case-Based testing:** Use cases specify the interactions between software elements and actors, detailing the activities actors can perform with the elements, and the conditions before and after executing any scenario.

b) **Equivalence partitioning:** Main idea involves dividing data into parts, processing each part in the same way. Additionally, it commonly categorizes partitions into valid and invalid values, representing accepted and rejected values by the software component.
c) **State transition testing:** It focuses on changing input conditions or states when an event occurs, and events are given through the transitions between different states.
d) **Boundary value analysis**: It is an extension of the equivalence partitioning. But this technique focuses on values at the boundaries of numeric or sequential data. It identifies minimum and maximum values, which would indicate that values outside the minimum and maximum range are considered invalid.
e) **Decision table:** Business rules define the functions and behavior of the system. Decision table help identify conditions and actions (input and output) as either true or false, providing a structured approach to testing complex rule-based systems.

6. What is the purpose of the following types of tests?

a) Functional test: Its intended purpose is to verify that the application functions as required based on user-defined specifications. It encompasses all the software's components, including features, UI, databases, etc. Functional testing contains various of the types of tests such as unit test, smoke test, integration, GUI, and regression test.

b) Performance test: Performance testing is conducted to validate the speed, scalability, responsiveness, and overall stability of the application under different conditions (expected and unexpected workloads). It helps identifying clogging or potentials software performance failures.

c) Security test: Security testing assesses vulnerabilities and weaknesses in the software. Security testing includes testing for potential data breaches, security risk, threats, and unauthorized access. Its purpose is to be prepared and resilient against malicious attacks.

d) Usability test: Usability testing overall user experience in their interaction with the software. Its purpose is keeping the application under the principle of user-friendliness, and it focuses on how easily users can interact with the application, as well as it helps enhance user satisfaction.

e) API test: This type of testing is performed to validate the reliability, functionality, availability, and performance of applications programming interfaces (APIs). Its purpose is to validate that APIs can communicate with each other and with all software system in general.

f) Unit Test: Unit testing as I mentioned above in other parts of the document, it is likely that testing of individual units or components of a software system in isolation. Its purpose is to validate that each unit component performs as designed. It is the major exponent in the early identification of bugs in the development process.