**11. Tesztelés**

1. Mi a célja az unit tesztelésnek?
   * **Egyes komponensek és funkciók tesztelése külön-külön** ✅
   * Az egész alkalmazás tesztelése
   * A felhasználói felület tesztelése
2. Melyik eszköz használható JavaScript unit tesztelésére?
   * Mocha
   * **Jest** ✅
   * Selenium
3. Mi a célja az integrációs tesztelésnek?
   * Az egyes funkciók tesztelése izoláltan
   * **A különböző rendszerelemek együttműködésének tesztelése** ✅
   * A felhasználói felület tesztelése
4. Mi a célja az end-to-end (E2E) tesztelésnek?
   * A kód minőségének ellenőrzése
   * A teljes alkalmazás működésének tesztelése a felhasználói interakcióval
   * **A teljes folyamat automatizálása a felhasználói szempontból** ✅
5. Melyik eszköz nem használható E2E tesztelésre?
   * Cypress
   * **Jest** ✅
   * Selenium
6. Mi a különbség a unit tesztelés és az integrációs tesztelés között?
   * Az integrációs tesztelés kisebb egységeket tesztel, míg az unit tesztelés az egész rendszert
   * **Az integrációs tesztelés az egyes modulok közötti kapcsolatokat ellenőrzi** ✅
   * Az unit tesztelés csak a felhasználói felületet ellenőrzi
7. Melyik tesztelési technika ellenőrzi az alkalmazás különböző rétegeinek együttműködését?
   * **Integrációs tesztelés** ✅
   * Unit tesztelés
   * E2E tesztelés
8. Milyen típusú tesztelést végezhetünk a felhasználói felületen?
   * **End-to-end tesztelés** ✅
   * Unit tesztelés
   * Integrációs tesztelés
9. Mi az automatizált tesztelés előnye?
   * **Gyorsabb tesztelés és kevesebb emberi hiba** ✅
   * Az alkalmazás gyorsabb működése
   * Több tesztelési környezet szükséges
10. Mi az a tesztelési stratégia?
    * **A tesztelési eljárások és gyakorlatok meghatározása** ✅
    * A felhasználói felület tervezése
    * A tesztelés eszközeinek kiválasztása
11. Melyik eszközt használják a JavaScript alapú unit teszteléshez?
    * **Mocha** ✅
    * Selenium
    * Apache JMeter
12. Melyik eszköz a leggyakrabban használt E2E teszteléshez a webes alkalmazásokban?
    * Jest
    * **Cypress** ✅
    * Postman
13. Mi a célja a CI/CD tesztelésnek?
    * A kód futtatása a fejlesztő gépén
    * **Automatizált tesztelés az integráció és deploy előtt** ✅
    * A tesztelési környezet manuális beállítása
14. Mi a különbség a kódlefedettség és a hibakeresés között?
    * A kódlefedettség a kód helyességét ellenőrzi, a hibakeresés az alkalmazás futását követi
    * A kódlefedettség azt mutatja, hogy mely kódrészeket tesztelték, a hibakeresés segít a program hibáiban
    * **A kódlefedettség mutatja, hogy a kód hány százalékát tesztelték, a hibakeresés segít azonosítani a problémákat** ✅
15. Mi az a hibakeresési eszköz?
    * **Olyan eszköz, amely segít a hibák megtalálásában és javításában** ✅
    * Olyan eszköz, amely automatikusan teszteli a kódot
    * Olyan eszköz, amely a tesztelési környezetet kezeli
16. Mi a célja a kódlefedettség mérésének?
    * A tesztelési környezet dokumentálása
    * **A tesztelési hatékonyság és a lefedett kódrészletek mérése** ✅
    * A tesztelők munkájának nyomon követése
17. Milyen típusú tesztelés vizsgálja a felhasználói interakciókat és azok hatásait?
    * **End-to-end tesztelés** ✅
    * Unit tesztelés
    * Biztonsági tesztelés
18. Mi a tesztelés legfontosabb célja?
    * **A hibák és problémák azonosítása az alkalmazásban** ✅
    * A felhasználói élmény növelése
    * Az alkalmazás sebességének növelése
19. Mi az automatizált tesztelés hátránya?
    * Nincs
    * **Magas kezdeti költség és erőforrásigény** ✅
    * Csak webes alkalmazásokhoz használható
20. Melyik tesztelési típus segít megelőzni, hogy új kód bevezetése hibákat okozzon a már meglévő rendszerben?
    * **Regression tesztelés** ✅
    * Unit tesztelés
    * Stressz tesztelés

**12. Tesztelés még 20 teszt**

* A komponensek közötti interakciók tesztelése
* **Az alkalmazás funkcionalitásának ellenőrzése** ✅
* A felhasználói élmény tesztelése

**12.6. Melyik nem tesztelési típus?**

* Unit tesztelés
* **Backend tesztelés** ✅
* End-to-end tesztelés

**12.7. Melyik tesztelési eszköz használható automatizált webes teszteléshez?**

* **Selenium** ✅
* Mocha
* JUnit

**12.8. Melyik nem egy tesztelési technika?**

* **Code review** ✅
* Unit tesztelés
* End-to-end tesztelés

**12.9. Melyik eszköz NEM alkalmas unit tesztelésre?**

* Mocha
* JUnit
* **Selenium** ✅

**12.10. Mi a CI/CD célja a tesztelésben?**

* **Automatizált build és tesztelési folyamatok biztosítása** ✅
* Tesztadatok generálása
* Tesztelési dokumentáció készítése

**12.11. Melyik tesztelési típus teszteli az egész alkalmazás működését?**

* **End-to-end tesztelés** ✅
* Unit tesztelés
* Integrációs tesztelés

**12.12. Mi az a hibakeresési technika a tesztelésben?**

* A tesztelési dokumentáció készítése
* **A problémák forrásának azonosítása a kód vagy a teszt során** ✅
* Kód refaktorálása

**12.13. Melyik eszköz használható frontend teszteléshez React alkalmazások esetén?**

* **Jest** ✅
* TypeORM
* Sequelize

**12.14. Mi a különbség a unit és az integrációs tesztelés között?**

* Az integrációs tesztelés egyesíti a tesztelt komponenseket
* **A unit tesztelés egy komponens önálló működését ellenőrzi** ✅
* Az integrációs tesztelés ellenőrzi az egész alkalmazást

**12.15. Melyik nem tesztelési szint?**

* **Test-Driven Development (TDD)** ✅
* Unit tesztelés
* End-to-end tesztelés

**12.16. Mi az a "mocking" tesztelés során?**

* **A tesztelt objektumok helyettesítése mesterséges adatokkal** ✅
* Kód hibáinak megkeresése
* Tesztadatok exportálása

**12.17. Mi a célja a kódlefedettség mérésének?**

* **A tesztelt kód arányának nyomon követése** ✅
* A kód olvashatóságának javítása
* A hibák javítása

**12.18. Mi a tesztelési stratégiák lényege?**

* **A különböző tesztelési technikák és eszközök kombinálása** ✅
* A kód automatikus generálása
* A tesztelési jelentések készítése

**12.19. Melyik eszköz nem teszteléshez való?**

* **Notepad++** ✅
* Jest
* Mocha

**12.20. Mi a legjobb gyakorlat a tesztelés során?**

* Tesztelés kizárólag manuálisan
* **Automatizált tesztelés és folyamatos integráció használata** ✅
* Tesztelés kizárólag az alkalmazás végső verzióján