# Django vizsga - elméleti feladatok

1. Mi az MVT, és hogyan valósul meg Django-ban? (5 pont)  
   Model-View-Template architektúra, ahol a:  
   - Model felel az adatok kezeléséért, ez a logikai adatstruktúra az applikáció mögött egy adatbázis áltat reprezentálva.  
   - View Python funkciók gyűjtménye, amely http requesteket fogad response-okat ad vissza, ahol a response lehet egy HTML dokumentum vagy egy átirányitás, XML dokumentum...  
   - Template a statikus része, a kívánt HTML kimenet beillesztett dinamikus tartalommal.
2. Sorolj fel legalább 3 field típust és szükséges / opcionális paramétereiket! (3 pont)  
   - IntegerField (nincs paraméter)  
   - CharField (kötelező: max\_length/kivéve Postgres)  
   - EmailField (opcionális: max\_length)Ű  
   - TimeField (opcionális: auto\_now, auto\_now\_add)
3. Hogyan kezeli egy django modell a kapcsolódó modelleket törlés esetén?   
   Sorolj fel legalább 3 lehetséges opciót! (3 pont)  
   - SetNull: a kapcsolódó objektum törlése esetén Null-ra állítja a kapcsolódó mezők értékét  
   - Cascade: törli a kapcsolódó child modelleket  
   - Protect: nem engedi törölni a kapcsolódó modellt
4. Hogyan épül fel egy alap Django projekt és app? Milyen fájlokból áll, és mi a szerepük? Sorolj fel legalább hármat! (3 pont)  
   - manage.py: parancssoros eszköz (futtatás, migrációk, szerverindítás).  
   -sajat\_projekt/settings.py: a projekt **beállításai** (adatbázis, appok, időzóna stb.).  
   - views.py: **Nézetek** – itt dől el, mit küld vissza egy kérésre a szerver (HTML, JSON stb.).  
   - urls.py: **App URL-jeinek** kezelése (manuálisan kell létrehozni).
5. Mi a 3 fő alkotórésze a Django template nyelvének (a html kódon kívül)?   
   Írj mindegyikre legalább 1 példát! (3 pont)  
   - template-változók: változók vagy objektum mezők megjelenítése {{ user.username }}  
   - template-tag-ek: vezérlés, logika {% if %}{% endif %}  
   -template-filterek: érték formázása megjalanítéskor {{ nev|upper }}
6. Milyen HTTP metódusokat kezelhet egy Django view? Írd le, melyik mire való! (4 pont)  
   - GET: adatok lekérése, az érték megjelenik az URL-ben  
   - POST: adatok küldése a szerverre, az érték nem jelenik meg az URL-ben  
   - PUT: egy meglévő elem teljes frissítése  
   - PATCH: egy meglévő elem részleges frissítése (akár csak egy mező)  
   - DELETE: egy elem törlése
7. Írj legalább 3 HTTP státuszkódot és jelentését! (3 pont)  
   - 404: nem található (hibás url vagy törölt cél esetén)  
   - 403: jogosulatlan kérés (a felhasználó nem rendelkezik megfelelő engedélyekkel az eléréshez)  
   - 201: létrehozva (API-knál jellemző, sikeres objektumlétrehozás esetén)
8. Mik a REST api alapelvei? Sorolj fel legalább 3-at, és röviden magyarázd meg, melyik mit jelent! (5 pont)  
   - Erőforrás alapúság: Az URL-ek erőforrásokat jelölnek, nem műveleteket  
   - HTTP metódusok használaza: A CRUD műveleteket HTTP metódusokkal fejezzük ki  
   - Állapotmentesség: A szerver nem emlékszik a korábbi kérésekre
9. Hogyan kezeli a Django a statikus fájlokat? Mit kell beállítani a használatukhoz? (3 pont)  
   Külön kezeli a dinamikus tartalomtól, használatukhoz a statikus fájlok mappájának beállítása szükséges, melyhez a mappát először létre kell hozni (static). A settings.py fájlban a static URL beállítása szükséges, a template-ekben pedig a {% load static %} tag elhelyezése.
10. Sorolj fel legalább 3 management parancsot, és hogy melyiket mire használjuk! (3 pont)  
    - runserver: elindítja a beépített fejlesztői szervert  
    - makemessages: létrehozza a nyelvi fordítási fájlokat (.po) az alkalmazás többnyelvűségéhez  
    - makemigrations: változásokat keres a models.py-ban és letrehozza a megfelelő adatbázismigrációkat.