TO-DO APP FEJLESZTÉSE MESTERSÉGES INTELLIGENCIÁVAL

A beadandómban egy olyan modern webalkalmazást készítettem, amely a teendők nyilvántartására szolgál, és amelyet egy mesterséges intelligenciával (ChatGPT) kiegészült chatfelülettel is bővítettem. A rendszer célja nem csupán a teendők egyszerű adminisztrációja, hanem az is, hogy a felhasználó egy intelligens segítséggel tudjon kommunikálni azokkal kapcsolatban. A projekt három fő komponensre bontható: a frontend felület, a C#-ban készült to-do backend, valamint a Python nyelvű AI backend.

A mesterséges intelligenciával való kommunikációért felelős backend részt Python nyelven valósítottam meg. Ennek felépítése viszonylag egyszerű, egyetlen fájlban helyezkedik el, és a korábban órán bemutatott példa alapján került kidolgozásra. Az alkalmazás egy HTTP POST végpontot biztosít, amely a C# backendből érkező adatokat fogadja, majd azokat egy olyan formára alakítja át, amely kompatibilis a ChatGPT API igényeivel.

A POST kérés során a rendszer három fő adatot kap: az eddigi üzenetek listáját, a felhasználó által választott modell nevét, valamint az aktuális teendők listáját. Ezekből építek fel egy olyan promptot, amely világos instrukciókat ad a mesterséges intelligenciának a kívánt viselkedésről. Az üzenetek előzményeinek megtartása elengedhetetlen, különben minden egyes kérdést kontextus nélküli, új megkeresésként értelmezne az AI, ami nem kívánatos viselkedéshez vezethet. Ugyanígy a teendők listája is kulcsfontosságú: az AI csak akkor tud érdemben reagálni, ha rendelkezésére állnak a releváns adatok. A válasz generálása után az eredmény visszaküldhető a C# backend felé.

A C#-ban megírt backend réteg felelős a teendők kezeléséért és tárolásáért, illetve kapcsolatot tart a Python backenddel. A projekt során külön modelleket hoztam létre a teendők és az üzenetek kezelésére, amelyek lehetővé teszik az adatok egységes és strukturált tárolását. A teendőkhöz egy különálló service osztályt is készítettem, amely az alap CRUD (Create, Read, Update, Delete) műveleteket valósítja meg, valamint a jelenlegi feladatokat egy JSON fájlba is kimenti, így biztosítva az adatok rendelkezésre állását.

A backend controller rétege biztosítja azokat a végpontokat, amelyeket a frontend hív meg, így valós időben frissíthető és követhető a felhasználói felületen megjelenő teendők listája. Külön figyelmet fordítottam az AI-val történő integrációra is: a megfelelő végpontban a controller lekéri az összes aktuális teendőt, valamint fogadja az üzeneteket, és ezeket együttesen továbbítja a Python backend felé. A válaszként kapott szöveget az alkalmazás visszaküldi a felhasználónak. Hiba esetén a rendszer egy egyszerű szöveget ad vissza: „Hiba történt a ChatGPT hívás során.”

A projekt frontend része React frameworkkel, TypeScript nyelven készült. Az alkalmazás felhasználói felülete egyszerű, de jól áttekinthető: középpontjában egy űrlap helyezkedik el, amellyel a felhasználó új teendőket tud hozzáadni az alatta elhelyezkedő listához. A feladatokhoz opcionálisan lejárati dátum is megadható, amellyel nyomon követhető, hogy melyik feladat meddig teljesítendő.

A felhasználói élmény növelése érdekében a rendszer toast értesítéseket jelenít meg sikeres műveletek (például hozzáadás, törlés vagy státuszváltás) esetén. A feladatokat külön jelölhetjük késznek is, de automatikus törlés nem történik – ez a felhasználóra van bízva.

A felületen található egy beépített chat ablak is, amely közvetlen kapcsolatot biztosít a ChatGPT-vel. Az üzeneteket egy tömbben tároljuk, ahol minden üzenethez hozzá van rendelve, hogy az a felhasználótól vagy az AI-tól származik. Ez az információ kulcsfontosságú a vizuális megjelenítés szempontjából, mivel ennek megfelelően kerülnek a stílusok alkalmazásra (pl. eltérő szín, igazítás).

Amikor egy új üzenetet küldünk, a teljes üzenetlista elküldésre kerül a C# backend felé, amely ezt továbbítja a Python oldalnak. A válasz feldolgozása ideje alatt egy „Gondolkodom…” felirat jelenik meg a felhasználó számára, így elkerülhető az az érzet, hogy az alkalmazás lefagyott volna.

Összességében a projekt során egy olyan, több rétegű alkalmazást készítettem, amelyben modern technológiák együttműködésével valósult meg egy egyszerű, de intelligens felhasználói élményt nyújtó rendszer. A különböző nyelven írt backendek (C# és Python) hatékonyan működnek együtt, míg a React-alapú frontend valós időben képes reagálni a backend változásaira. A mesterséges intelligencia integrálása kibővítette a felhasználói interakciókat, és egyúttal lehetőséget biztosított arra is, hogy mélyebben megismerkedjek az AI API-k gyakorlati használatával.