Android alapú szoftverfejlesztés

Önálló laboratórium – Bárányos Donát

Tartalom

Feladatleírás

A feladat egy tetszőleges, kellő komplexitású android-alapú alkalmazás elkészítése saját ötlet alapján. Az én választásom egy modern Jetpack Compose-on alapuló turisztikai segédalkalmazásra esett, melyet a jövőben folytatni tervezek és a féléves munka ennek az alapját képezi.

Tervezési fázis

A projekt első lépései a tervezéssel teltek. Megterveztem az alapvető funkciókat, mint például a felhasználókezelés, belépés, regisztráció, authentikáció, az almenük, képernyők számát, illetve kinézetét.

A tervezéshez a Figma nevű desgin eszközt használtam, a mai trendeknek megfelelően. A folyamat több lépésből állt, ezek a következőek:

Koncepció és Kutatás:

Minden projekt a koncepcióalkotással és a kutatással kezdődik. Fontos megérteni az igényeket és elvárásokat. A Figma segít a koncepciók vizualizálásában moodboardok, vázlatok és prototípusok készítésével.

Drótvázak (Wireframe-ek) és Elrendezés:

A drótvázak elkészítése a vizuális hierarchia, a tartalom elrendezése és a felhasználói interakciók megtervezésének alapjait fekteti le. A Figma vektoros rajzeszközöket és sablonokat kínál a gyors és hatékony drótvázoláshoz. Könnyedén lehet iterálni és finomítani a drótvázakat a visszajelzések alapján.

Vizuális Tervezés:

A vizuális tervezés a színek, a tipográfia, az ikonok és a képi elemek használatával adja meg az UI arcát és hangulatát. A Figma gazdag eszközkészletet biztosít a stílusok létrehozásához, szerkesztéséhez és alkalmazásához az egész projekten belül. Ezen felül én a Google Material Design 3-as desgin keretrendszer használata mellett döntöttem, ami segít egységes, letisztult felületet tervezni és implementálni.

Prototípuskészítés és Tesztelés:

Figmán belül lehetőség van arra, hogy a képernyőterveken egyes pontokat megjelöljünk, melyeknek később navigációs feladta lesz és ezáltal egy kezdeti prototípust tudjunk készíteni. Külön hasznos a több navigációs munkamenet (workflow), mely segítségével akár csak kisebb folyamatok szimulációját is be lehet mutatni.

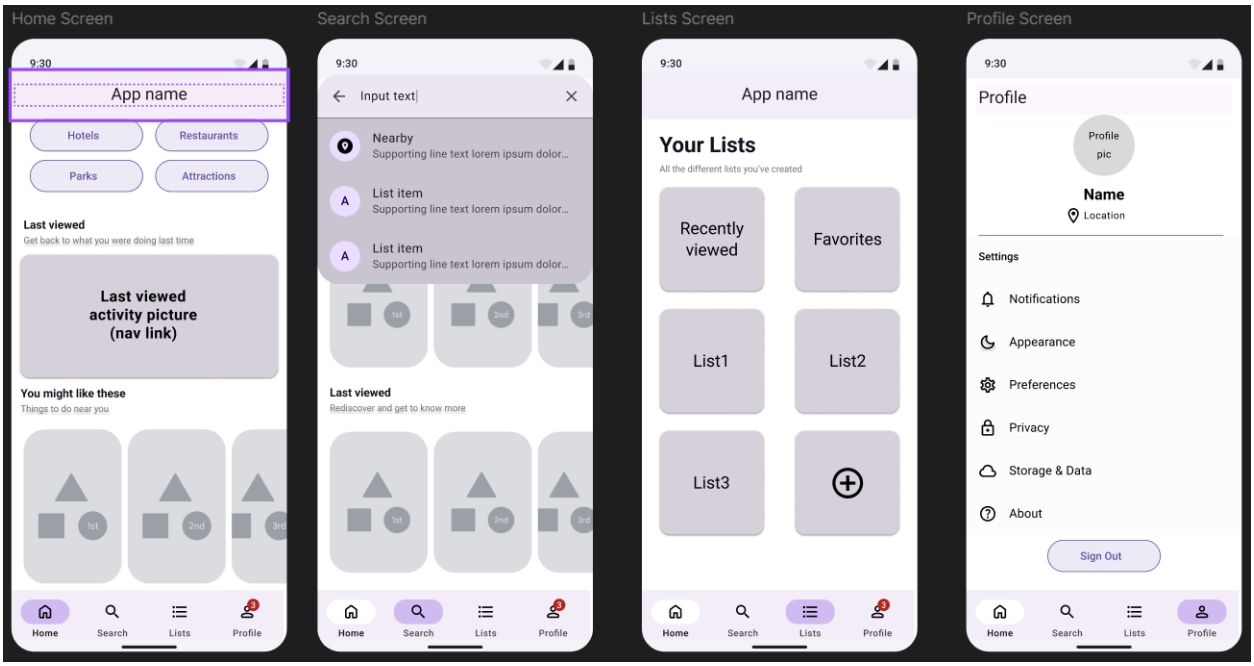
Iteráció és Finomítás:

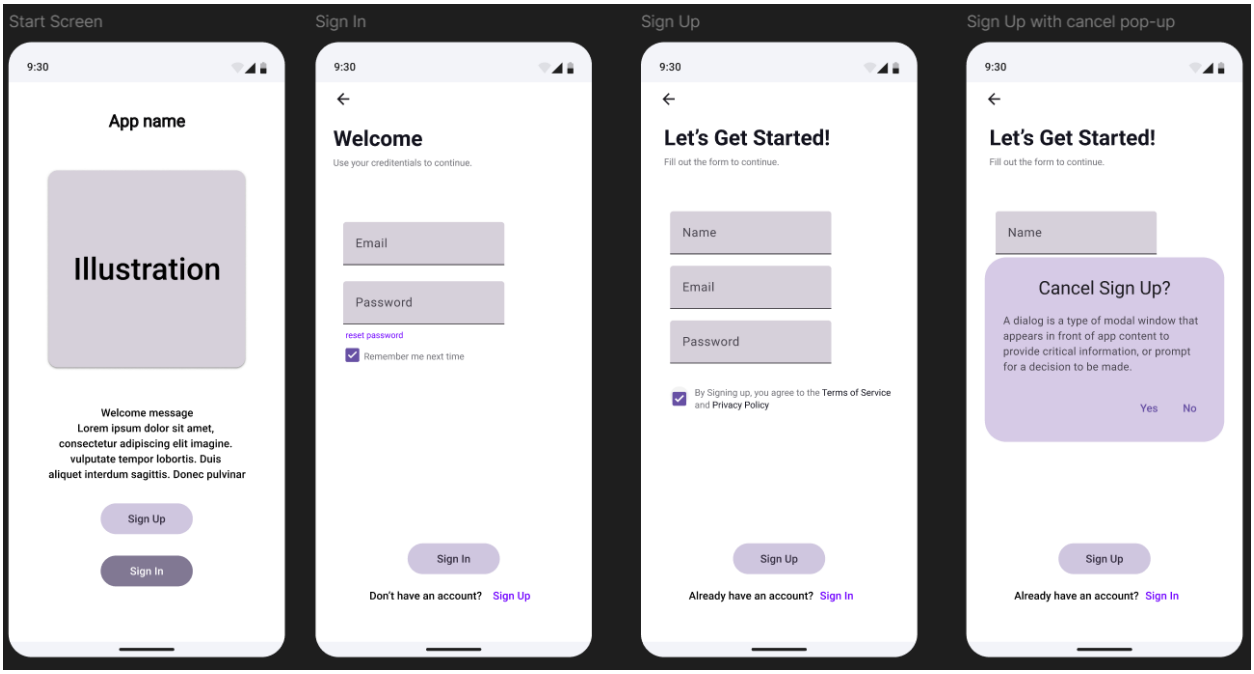
A UI/UX design folyamata nem lineáris. A design-t finomítani kell a visszajelzések, igények alapján. Emellett, kihasználva a Figma verziókövetési adottságait, egyes kijelzőkterveknél, volt, hogy visszatértem korábbi verziókhoz.

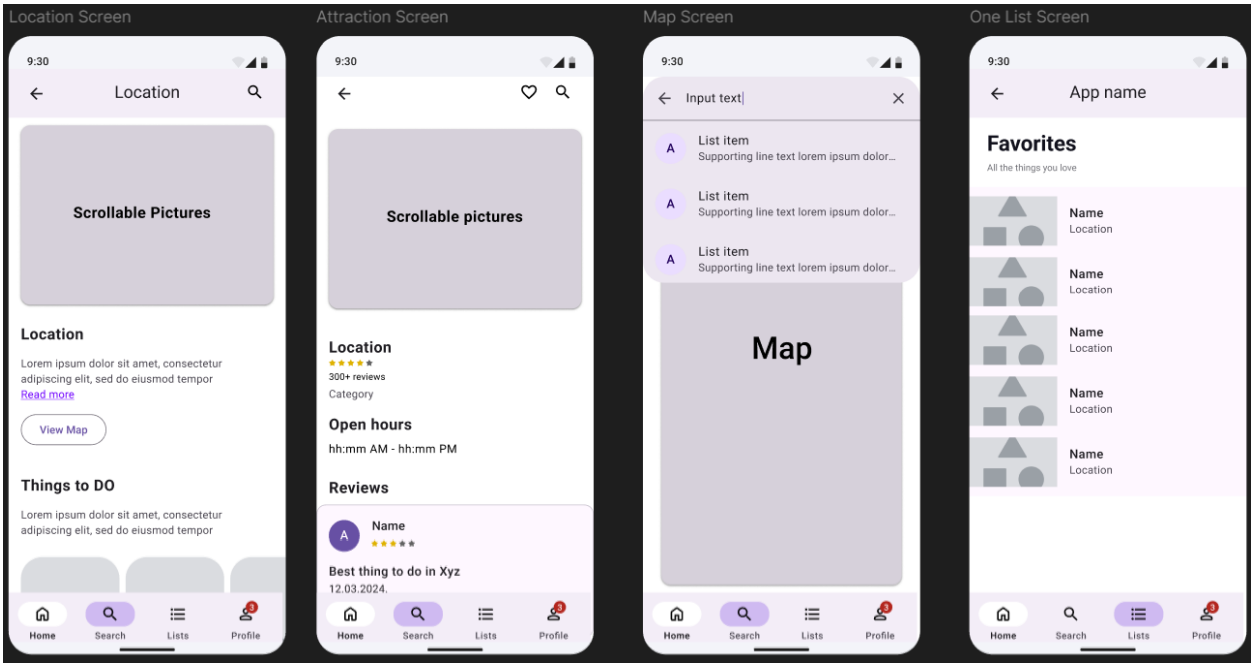
A tervezés eredménye:

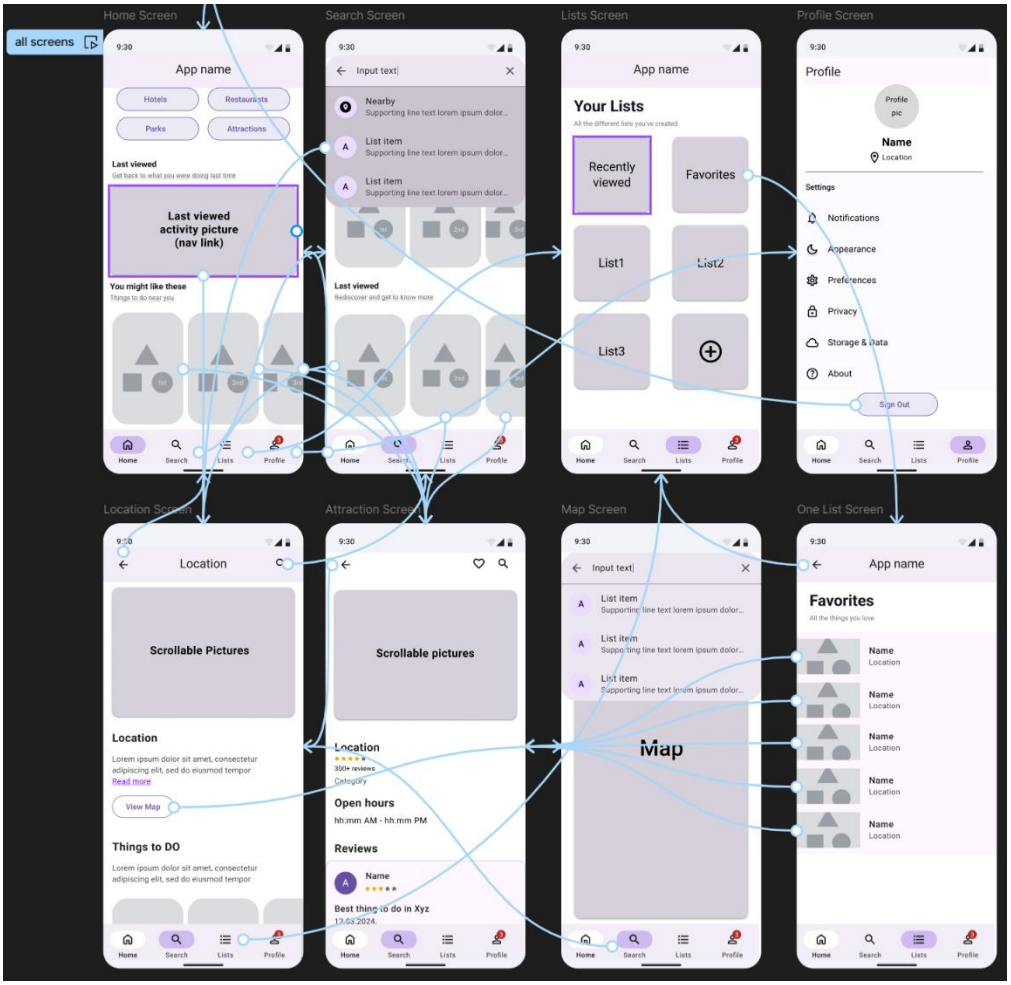
A Figma segítségével az alkalmazásban használt komponensek egyes részeit leíró kód keletkezett, ami megkönnyítette az implementációt, azon felül, hogy a tervezési folyamat önmagában segített vizualizálni az alkalmazást és a fejlesztés irányát.

Képek a Figmából:






Location Screen
Attraction Screen
Map Screen
One List Screen
9:30
Location
Scrollable Pictures
Location
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor
Read more
View Map
Things to DO
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor
Home
Search
Lists
Profile
Scrollable pictures
Location
300+ reviews
Category
Open hours
hh:mm AM - hh:mm PM
Reviews
Name
Best thing to do in Xyz
12.03.2024.
Input text
List item
Supporting line text lorem ipsum dolor...
Map
App name
Favorites
All the things you love
Name
Location


Home Screen
Search Screen
Lists Screen
Profile Screen
all screens
App name
Hotels
Restaurants
Parks
Attractions
Last viewed
Get back to what you were doing last time
Last viewed activity picture (nav link)
You might like these
Things to do near you
Input text
Nearby
Supporting line text lorem ipsum dolor...
List item
Last viewed
Rediscover and get to know more
Your Lists
All the different lists you've created
Recently viewed
Favorites
List1
List2
List3
Profile
Profile pic
Name
Location
Settings
Notifications
Appearance
Preferences
Privacy
Storage & Data
About
Sign Out
Location Screen
Attraction Screen
Map Screen
One List Screen
Location
Scrollable Pictures
Location
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor
Read more
View Map
Things to DO
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor
Scrollable pictures
Location
300+ reviews
Category
Open hours
hh:mm AM - hh:mm PM
Reviews
Name
Best thing to do in Xyz
17.01.2024.
Map
Favorites
All the things you love
Name
Location

Technológiai Stack

A technológiai stack a szoftverfejlesztési projektben használt eszközök, keretrendszerek és programozási nyelvek összességét jelenti. A megfelelő technológiai stack kiválasztása kulcsfontosságú a projekt sikeréhez, mivel befolyásolja a fejlesztési folyamat hatékonyságát, a kód minőségét és a végső termék funkcionalitását.

Ebben a bekezdésben bemutatom az önálló laboratórium project keretein belül használt technológiai stacket, kiemelve a főbb komponenseket és azok szerepét a projekt megvalósításában.

Programozási nyelvek:

A Kotlin a Google által preferált programozási nyelv az Android-alkalmazások fejlesztéséhez. Modern, statikusan típusú nyelv, amely objektumorientált és funkcionális programozási paradigmákat is támogat. A Kotlin hatékony, megbízható és könnyen karbantartható, illetve segítségével natív alkalmazás készíthető. Ideális választássá téve komplex mobilalkalmazások fejlesztéséhez.

Keretrendszerek és Eszközök:

Jetpack Compose: A Jetpack Compose a Google deklaratív UI keretrendszere az Android-alkalmazásokhoz. Lehetővé teszi a fejlesztők számára, hogy a UI-t deklaratív módon építsék fel, elkerülve a bonyolult XML-elrendezéseket. Valamint lehetőséget ad a komponens alapú fejlesztésre, a modern trendeknek megfelelően.

Firebase: A Firebase a Google által kínált felhőalapú platform, amely számos szolgáltatást nyújt mobilalkalmazások fejlesztéséhez, beleértve az adatbázisokat, a hitelesítést, a tárolást és az analitikát. A Firebase skálázható, megbízható és könnyen integrálható, ideális választássá téve a mobilalkalmazások háttérfunkcióinak megvalósításához.

Android Studio: Az Android Studio a Google hivatalos integrált fejlesztési környezete (IDE) az Android-alkalmazások fejlesztéséhez. Teljes funkcionalitást kínál a kódolástól a tesztelésig és a telepítésig. Az Android Studio intuitív felülettel és beépített emulátorral rendelkezik, amelyet én is használtam a teszteléshez, illetve a bemutatáshoz.

Figma: A Figma egy népszerű tervezői platform, amelyet a felhasználói felületek (UI) és felhasználói élmények (UX) tervezéséhez használnak. Intuitív felületet, vektoros

rajzeszközöket és prototípuskészítési funkciókat kínál, lehetővé téve a tervezők számára, hogy hatékonyan vizualizálják és teszteljék az UI koncepciókat.

Adatbázis:

Firebase Firestore: A Firebase Firestore egy NoSQL dokumentum-adatbázis, amely skálázható, valós idejű adathozzáférést biztosít. Ideális választás mobilalkalmazásokhoz, amelyek dinamikus adatbázis-lekérdezéseket és valós idejű szinkronizálást igényelnek.

Hitelesítés:

Firebase Authentication: A Firebase Authentication egy szolgáltatás, amely a felhasználók bejelentkezésének és hitelesítésének kezelését automatizálja a mobilalkalmazásokban. Támogatja a jelszóalapú hitelesítést, a közösségi média bejelentkezéseket és a biometrikus hitelesítést. Melyek közül a választásom a jelszó-email párosra esett, de a jövőben ezt tervezem bővíteni.

Az alkalmazás alapjainak implementálása

Az alkalmazásnak több nagyobb eleme van:

- Home (kezdőképernyő)
- Keresés
- Listák
- Profil
- Város
- Attrakció

Az első folyamat amit elkészítettem az a felhasználó belépése és regisztrációja, ez a következőképpen néz ki:

Belépés (Sign In)

1. A felhasználó beírja az e-mail címét és jelszavát.
2. Az alkalmazás elküldi a bejelentkezési adatokat a Firebase Authenticationnek.
3. A Firebase Authentication ellenőrzi a bejelentkezési adatokat.

4. Ha a bejelentkezési adatok helyesek, a Firebase Authentication visszaad egy hitelesítési tokent.
5. Az alkalmazás tárolja a hitelesítési tokent a helyi tárhelyen.

Regisztráció (Sign Up)

1. A felhasználó beírja az e-mail címét, jelszavát és a nevét, ezek szükségesek jelenleg.
2. Az alkalmazás elküldi a regisztrációs adatokat a Firebase Authenticationnek.
3. A Firebase Authentication létrehoz egy új felhasználói fiókot és visszaad egy hitelesítési tokent.
4. Az alkalmazás tárolja a hitelesítési tokent a helyi tárhelyen.

Home képernyő

Leírás:

A Home képernyő az alkalmazás egyik fő képernyője. Különböző szekciókat jelenít meg, amelyek segítségével a felhasználók felfedezhetik a különböző funkciókat és lehetőségeket.

Komponensek:

- **TopAppBar (Felső App Sáv):** A középen az alkalmazás nevét jeleníti meg.
- **Szálloda & Étterem Gombok:** Két OutlinedButton gomb a Szállodák és Éttermek szekciók gyors eléréséhez, ez az aktuális városra vonatkozik.
- **Parkok & Látnivalók Gombok:** További két OutlinedButton gomb a Parkok és Látnivalók szekciókhoz, ez az aktuális városra vonatkozik.
- **Utoljára Megtekintve Szekció:** Fejléc és leírás szöveg az "Utoljára Megtekintve" szekcióhoz.
- **Utoljára Megtekintve Kártya:** Egy kattintható kártya, amely helyőrzőt jelenít meg a felhasználó által utoljára megtekintett tevékenységhez. A kártyára kattintva a felhasználót egy adott képernyőre kellene navigálni.
- **Programlehetőségek a közelben:** Cím a "Programlehetőségek a közelben" szekcióhoz.
- **Helyszínlista:** Vízszintes sor, amely kártyákat tartalmaz a közeli helyszínek megjelenítésére. Minden kártya egy helyőrző szöveget jelenít meg (további megvalósítás szükséges).

Navigáció:

- Az "Utoljára Megtekintve" kártyára kattintva a felhasználót egy adott attrakció vagy város képernyőjére kell navigálni.
- A "Helyszínlista" egy elemére kattintva a felhasználót a konkrét helyszín részleteit tartalmazó képernyőre lehet navigálni.

Függőségek:

- androidx.compose.foundation: Alapvető UI elemeket biztosít, mint például Column (Oszlop), Row (Sor), Box (Doboz), Card (Kártya) stb.
- androidx.compose.material3: Material Design 3 komponenseket biztosít, mint például TopAppBar, OutlinedButton, Text (Szöveg), Icon (Ikon) stb.
- androidx.compose.material.icons.outlined: Előre definiált ikonokat biztosít látnivalókhoz, szállodákhoz, éttermekhez stb.
- androidx.navigation: (ebben a fájlban nem használják közvetlenül) Navigációs lehetőségeket biztosít a képernyők között.

Tennivalók:

- Meg kell valósítani a Szálloda, Étterem, Parkok és Látnivalók gombok műveleteit.
- Logikát kell beépíteni a felhasználó által utoljára megtekintett tevékenység megjelenítéséhez és a konkrét képernyőre való navigáláshoz.
- A "Helyszínlista" megvalósítása valós adatokkal és a részletek képernyőire való navigációval.

Keresés

Leírás:

A Keresés képernyő lehetővé teszi a felhasználóknak, hogy helyszíneket keressenek, valamint megtekinthetik a kedvenc és a legutóbb megtekintett helyszíneket.

Komponensek:

- **Keresősáv:** A felhasználók itt adhatják meg a keresett helyszín nevét.
- **Kedvencek:** Szakasz a felhasználó kedvenc helyszíneivel.

- **Utoljára Megtekintve:** Szakasz a felhasználó által legutóbb megtekintett helyszínekkel.

Keresősáv:

- A Keresősáv egy TextField komponens, amely lehetővé teszi a felhasználó számára a keresőkifejezés beírását.
- A felhasználó a billentyűzet Enter billentyűjének megnyomásával indíthatja el a keresést.
- A keresés jelenleg három előre rögzített helyszínre (Budapest, Bécs, Róma) működik. A megadott helyszín szövegének pontosan egyeznie kell ("Budapest\n", "Bécs\n", "Róma\n") a navigációhoz.

Kedvencek:

- A Kedvencek szekció jelenleg nincs megvalósítva.
- A tervek szerint ez a szekció a felhasználó kedvenc helyszíneit fogja megjeleníteni.

Utoljára Megtekintve:

- Az Utoljára Megtekintve szekció három kártyát jelenít meg helyőrző szövegekkel.
- Jelenleg három előre definiált hely van kódolva: Budapest, Bécs és Róma.
- A kártyára kattintás a felhasználót a megadott helyszínre (pl.: "Budapest") navigálja.

Navigáció:

- A Keresősáv Enter billentyűjének lenyomásával vagy az Utoljára Megtekintve kártyára kattintással a felhasználó a megadott helyszínre navigál (feltételezve a pontos szövegegyezést a keresésnél).

Függőségek:

- androidx.compose.foundation: Alapvető UI elemeket biztosít, mint például Column (Oszlop), Row (Sor), Box (Doboz), Card (Kártya) stb.
- androidx.compose.material3: Material Design 3 komponenseket biztosít, mint például Text (Szöveg), TextField (Szövegbeviteli mező), Button (Gomb) stb.
- androidx.compose.runtime: A runtime komponenseket biztosít, mint például a collectAsState, amely az állapot változásainak figyelésére szolgál.

- androidx.navigation: (ebben a fájlban nem használják közvetlenül) Navigációs lehetőségeket biztosít a képernyők között.
- com.example.myapplication.Model.SearchViewModel: A keresés logikáját és állapotát kezelő ViewModel.

Tennivalók:

- A Kedvencek szekció megvalósítása.
- A helyszínlista dinamikus betöltése a valós adatokból.
- Egy görgethető LazyRow komponensbe legyenek ágyazva a kártyák.

Listák

Leírás:

A Listák képernyő lehetővé teszi a felhasználók számára, hogy létrehozzák és kezeljék a listáikat. A listák olyan gyűjtőhelyek, amelyek különböző elemeket tartalmazhatnak.

Komponensek:

- **Felső App Sáv:** Az alkalmazás nevét jeleníti meg.
- **Lista Létrehozása Szekció:** Szakasz a listák kezeléséhez.
    - Szöveg: "A Listáid"
    - Leírás: "Az összes általad létrehozott lista"
    - Új Lista gomb: Lehetővé teszi a felhasználó számára új lista létrehozását (a művelet.
- **Lista Elemek:** A létrehozott listák vizuális megjelenítése.
    - Minden listaelem egy doboz, amely a lista nevét jeleníti meg.
    - A Boxra kattintás a listával kapcsolatos további műveletekhez vezet.

Funkciók:

- Új lista létrehozása.
- Lista megnyitása további részletek megtekintéséhez.

Megvalósítás:

- A képernyő egy Column elrendezést használ, amely függőlegesen helyezi el a komponenseket.
- A TopAppBar komponens megjeleníti az alkalmazás nevét a képernyő tetején.
- A "Lista Létrehozása Szekció" két Text komponensből és egy Button komponensből áll.
- A "Lista Elemek" egy oszlop, amely több Box elemet tartalmaz. Minden Box egy listát reprezentál.
- A Box elemek tartalmaznak egy Text komponenst a lista neve megjelenítéséhez, valamint egy kattintható területet további műveletekhez (még nincs megvalósítva).

Függőségek:

- androidx.compose.foundation: Alapvető UI elemeket biztosít, mint például Column (Oszlop), Row (Sor), Box (Doboz) stb.
- androidx.compose.material3: Material Design 3 komponenseket biztosít, mint például TopAppBar, Text, Button stb.
- androidx.compose.material.icons.outlined: Ikonokat biztosít

Tennivalók:

- A listák megnyitásának és a további műveletek kezelésének megvalósítása.

Profil

Leírás:

A Profil képernyő a felhasználó fiókjával kapcsolatos beállításokat és információkat jeleníti meg. A képernyő két fő szakaszra osztható:

- **Felhasználói Profil:** A felhasználó nevét, profilképét és tartózkodási helyét jeleníti meg.
- **Profil Beállítások:** Listaelemekből áll, amelyek különböző profilbeállításokhoz navigálnak.

Komponensek:

- **Felső App Sáv:** A képernyő tetején található, és a "Profil" címet jeleníti meg.

- **Profilkép:** A felhasználó profilképét megjeleníti kör alakú kivágással.
- **Felhasználó neve és tartózkodási helye:** A felhasználó nevét cím szövegstílusban, a tartózkodási helyét pedig a szövegtörzs szövegstílusban jeleníti meg.
- **Szerkesztés Profil gomb vagy Bejelentkezés:**
    - Ha a felhasználó be van jelentkezve, a gomb szövege "Profil Szerkesztése".
    - Ha a felhasználó nincs bejelentkezve, a gomb szövege "Bejelentkezés". A gomb megnyomása a "bejelentkezés" képernyőre navigál.
- **Profil Beállítások Lista:** Több elemből álló lista, amelyek a felhasználó különféle beállítási lehetőségeihez vezetnek. Minden elem egy címet és egy ikont tartalmaz.

Profil Beállítások Elemek:

- Értesítések (ikon: harang)
- Megjelenés (ikon: Sötét téma)
- Beállítások (ikon: Fogaskerék)
- Adatvédelem (ikon: Lakát)
- Tárolás és Adatok (ikon: Tároló)
- Névjegy (ikon: Információ)
- Kijelentkezés (ikon: Kijelentkezés)

Funkciók:

- Profil szerkesztése (még nincs megvalósítva).
- Kijelentkezés a fiókból.

Megvalósítás:

- A képernyő egy Column elrendezést használ, amely függőlegesen helyezi el a komponenseket.
- A TopAppBar komponens megjeleníti a képernyő címet.
- A felhasználó profiljának adatai Text és Image komponensekkel vannak megjelenítve.
- A "Szerkesztés Profil" vagy "Bejelentkezés" gomb egy Button komponens.

- A Profil beállítások listája egy Column elrendezésen belül ListItem komponenseket használ.

Függőségek:

- androidx.compose.foundation: Alapvető UI elemeket biztosít, mint például Column (Oszlop), Row (Sor), Image (Kép) stb.
- androidx.compose.material3: Material Design 3 komponenseket biztosít, mint például TopAppBar, Text, Button, ListItem stb.
- androidx.compose.material.icons.outlined: Ikonokat biztosít.
- androidx.compose.material.icons.filled: Ikonokat biztosít.
- androidx.compose.material.icons.automirrored: Ikonokat biztosít.
- com.example.myapplication.Model.ProfileViewModel: A Profil képernyőhöz tartozó ViewModel, amely a felhasználó bejelentkezési állapotát kezeli.
- com.google.firebase.auth.FirebaseAuth: A Firebase Authentication használatához szükséges.

Tennivalók:

- Profil szerkesztése funkció megvalósítása.
- A Profil beállítások elemekre való kattintás kezelése, a megfelelő képernyőkhöz történő navigációval.

Belépés (Sign In)

Leírás:

A Bejelentkezés képernyő lehetővé teszi a felhasználóknak, hogy meglévő fiókkal bejelentkezzenek az alkalmazásba.

Komponensek:

- **Cím:** "Bejelentkezés" szöveg a képernyő tetején.
- **Email mező:** Az email cím megadására szolgáló szövegbeviteli mező.
- **Jelszó mező:** A jelszó megadására szolgáló szövegbeviteli mező.
    - A jelszó mező alapértelmezetten elrejti a beírt karaktereket.

    - A mező mellett található egy ikonra kattintva a jelszó láthatóvá tehető.
- **"Remember me next time" Jelölőnégyzet:** (még nincs megvalósítva) A bejelöléssel a rendszer megjegyzi a felhasználót a következő kijelentkezésig.
- **"Elfelejtett jelszó" Szöveg:** Kattintással a jelszó visszaállítási folyamatra navigál.
- **Bejelentkezés gomb:** A bejelentkezési folyamatot indítja el.
    - Bejelentkezés közben a gombon egy körkörös folyamatjelző jelenik meg.
- **Regisztráció:** Szöveges link a Regisztráció képernyőre való navigáláshoz.

Funkciók:

- Bejelentkezés email címmel és jelszóval.
- A "Remember me next time" jelölőnégyzet működésének megvalósítása.
- Az "Elfelejtett jelszó" szövegre kattintás kezelése és a jelszó visszaállítási folyamatra való navigálás.

Állapot:

- **passwordVisible:** Egy Boolean érték, amely meghatározza, hogy a jelszó látható-e a beviteli mezőben.

Események:

- **Bejelentkezés gomb megnyomása:**
    - A gomb megnyomása elindítja a bejelentkezési folyamatot a SignInViewModel segítségével.
    - Bejelentkezés közben a gombon egy körkörös folyamatjelző jelenik meg, és az isLoading állapot true értéket vesz fel.
    - Sikeres bejelentkezés esetén a navigáció a "home" képernyőre történik, és az isLoading állapot visszaáll false értékre.

Függőségek:

- androidx.compose.foundation: Alapvető UI elemeket biztosít, mint például Column (Oszlop), Row (Sor), Text (Szöveg) stb.
- androidx.compose.material3: Material Design 3 komponenseket biztosít, mint például OutlinedTextField (Szövegbeviteli mező), Button (Gomb), IconButton (Ikon gomb) stb.

- androidx.compose.runtime: Állapot kezelését biztosító függvényeket tartalmaz, mint például remember és mutableStateOf.
- androidx.compose.material.icons.filled: Ikonokat biztosít.
- androidx.navigation.NavController: A képernyők közötti navigációhoz szükséges.
- com.example.myapplication.Components.CheckBoxComponent: A "Remember me next time" jelölőnégyzetet megjelenítő komponens.
- com.example.myapplication.Model.SignInViewModel: A Bejelentkezés képernyőhöz tartozó ViewModel, amely a bejelentkezési folyamatot és az állapot kezelését végzi.
- kotlinx.coroutines: Aszinkron műveletek kezeléséhez szükséges függvényeket biztosít.

Regisztráció (Sign Up)

Leírás:

A Regisztráció képernyő lehetővé teszi az új felhasználóknak, hogy létrehozzák a fiókjukat az alkalmazásban.

Komponensek:

- **Cím:** "Let's Get Started!" szöveg a képernyő tetején.
- **Név mező:** A felhasználó nevének megadására szolgáló szövegbeviteli mező.
- **Email mező:** Az email cím megadására szolgáló szövegbeviteli mező.
- **Jelszó mező:** A jelszó megadására szolgáló szövegbeviteli mező.
    - A jelszó mező alapértelmezetten elrejti a beírt karaktereket.
    - A mező mellett található egy ikonra kattintva a jelszó láthatóvá tehető.
- **"Be kell tartania a Szolgáltatási Feltételeket és az Adatvédelmi Tájékoztatót" Jelölőnégyzet:** A felhasználó a regisztrációval elfogadja a feltételeket.
- **Regisztráció gomb:** A regisztrációs folyamatot indítja el.
    - Regisztráció közben a gombon egy körkörös folyamatjelző jelenik meg.
- **Bejelentkezés:** Szöveges link a Bejelentkezés képernyőre való navigáláshoz.

Funkciók:

- Regisztráció név, email cím és jelszó megadásával.
- A felhasználó által megadott adatok ellenőrzése.
- Sikeres regisztráció esetén a navigáció a "home" képernyőre történik.

Állapot:

- **passwordVisible:** Egy Boolean érték, amely meghatározza, hogy a jelszó látható-e a beviteli mezőben.

Események:

- **Regisztráció gomb megnyomása:**
    - A gomb megnyomása elindítja a regisztrációs folyamatot a SignUpViewModel segítségével.
    - Regisztráció közben a gombon egy körkörös folyamatjelző jelenik meg, és az isLoading állapot true értéket vesz fel.
    - Sikeres regisztráció esetén a navigáció a "home" képernyőre történik, és az isLoading állapot visszaáll false értékre.

Függőségek:

- androidx.compose.foundation: Alapvető UI elemeket biztosít, mint például Column (Oszlop), Row (Sor), Text (Szöveg) stb.
- androidx.compose.material3: Material Design 3 komponenseket biztosít, mint például OutlinedTextField (Szövegbeviteli mező), Button (Gomb), IconButton (Ikon gomb) stb.
- androidx.compose.runtime: Állapot kezelését biztosító függvényeket tartalmaz, mint például remember és mutableStateOf.
- androidx.compose.material.icons.filled: Ikonokat biztosít.
- androidx.compose.ui.text.input.KeyboardOptions: Beállítja a billentyűzet típusát (ebben az esetben jelszó).
- androidx.compose.ui.text.input.PasswordVisualTransformation: Elrejti a beírt jelszó karaktereket.
- androidx.compose.ui.text.input.VisualTransformation: A beviteli mező megjelenítését szabályozza.

- androidx.navigation.NavController: A képernyők közötti navigációhoz szükséges.
- com.example.myapplication.Components.CheckBoxComponent: A "Be kell tartania a Szolgáltatási Feltételeket és az Adatvédelmi Tájékoztatót" jelölőnégyzetet megjelenítő komponens.
- com.example.myapplication.Model.SignUpViewModel: A Regisztráció képernyőhöz tartozó ViewModel, amely a regisztrációs folyamatot és az állapot kezelését végzi.
- kotlinx.coroutines: Aszinkron műveletek kezeléséhez szükséges függvényeket biztosít.

Budapest (város példa)

Leírás:

A Budapest képernyő az alkalmazás felhasználói számára részletes információkat nyújt Budapest városáról, valamint látnivalókról, éttermekről és parkokról.

Komponensek:

- **Felső App Sáv (TopAppBar):**
  - Cím: "Budapest" szöveg középen igazítva.
  - Jobb oldali ikongomb: a keresési képernyőre navigál.
- **Kép:** Budapest helyettesítő képe.
- **Cím:** "Budapest" főcím.
- **Leírás:** Rövid leírás Budapest városáról.
- **"See on Map" Gomb:** A Budapest térképre való navigálást kezdeményezi.
- **"Things to DO" Fecím:** Alsócímet jelenít meg a látnivalóknak.
- **Látnivaló kártyák:** Sorban elhelyezett három kártya, amelyek mindegyike egy budapesti látnivalót, éttermet vagy parkot képvisel.
  - Minden kártya tartalmaz egy képet és a hely nevét.
  - A kártyákra kattintva a felhasználó a kiválasztott helyre navigálhat.

Funkciók:

- Budapest városának és látnivalóinak bemutatása.

- A "See on Map" gomb megnyomása a Budapest térképre való navigálás (még nincs megvalósítva).
- A látnivaló kártyákra kattintás a kiválasztott helyre történő navigálás.

Navigáció:

- A jobb felső sarokban lévő kereső ikonra kattintva a felhasználó a keresési képernyőre navigálhat.
- A "See on Map" gombra kattintva a felhasználó a Budapest térképre navigál.
- A látnivaló kártyákra kattintva a felhasználó a kiválasztott budapesti látnivaló, étterem vagy park részletei képernyőre navigál.

AttractionBp (Egy város látnivalójának a példája)

Leírás:

A képernyő részletes információkat nyújt a kiválasztott budapesti látnivalóról. A képernyőn megtalálhatóak a látványosság képei, átlagos értékelése, értékelés száma, nyitvatartási ideje, valamint a hozzá tartozó vélemények listája.

Komponensek:

- **Felső App Sáv (TopAppBar):**
    - Cím: A kiválasztott látnivaló neve.
    - Jobb oldali ikonok:
        - Keresés ikon: a keresési képernyőre navigál.
        - Kedvenc ikon: a Kedvencek listához kapcsolja a látnivalót.
- **Kép:** A látnivaló képe (helyettesítő képpel jelenik meg).
- **Értékelés szekció:**
    - Csillag ikon sárga színben a pontszámot és a vélemények számát megjelenítve.
- **Nyitvatartási idő:** A látnivaló nyitvatartási ideje.
- **Vélemények:**
    - "Vélemények" címke.

- A felhasználók véleményeinek listája.

Funkciók:

- A kiválasztott budapesti látnivaló részleteinek megjelenítése.

Navigáció:

- A jobb felső sarokban lévő kereső ikonra kattintva a felhasználó a keresési képernyőre navigálhat.
- A jobb felső sarokban lévő kedvenc ikonra kattintva a felhasználó a Kedvencek listához adhatja a látnivalót (még nincs megvalósítva).

Adatmodell:

- Location: A látnivaló adatait tárolja (cím, átlagos értékelés, értékelések száma, nyitvatartási idő, képek, vélemények).
- Review: A vélemények adatait tárolja (szerző, szöveg, értékelés).

Megjegyzés:

- A Kedvencek lista funkció jelenleg még nincs megvalósítva.

Példa Adatok:

```
val location = Location(
  title = "Lánchíd",
  averageRating = 4,
  numReviews = 100,
  openHours = "Hétfő - Vasárnap: 10:00 - 18:00",
  images = listOf(), // Képek listája
  reviews = listOf(
    Review(author = "John Doe", text = "Gyönyörű híd!", rating = 5),
    Review(author = "Jane Smith", text = "Sétálni is jó volt rajta.", rating = 4),
    // További vélemények
  ))
```

Locations osztály

Adatmodellek:

- **Location (Helyszín):** A budapesti helyszín (park, étterem, látnivaló) adatait tárolja.
- **Review (Vélemény):** A felhasználói vélemények adatait tárolja.

Location (Helyszín):

- title (Cím): String - A helyszín neve.
- imageId (Kép azonosító): Int - A helyszín képének azonosítója (jövőbeni használatra, még nincs implementálva).
- averageRating (Átlagos értékelés): Int - A helyszín átlagos értékelése (1-től 5-ig).
- numReviews (Értékelések száma): Int - A helyszínhez tartozó vélemények száma.
- openHours (Nyitvatartási idő): String - A helyszín nyitvatartási ideje.
- reviews (Vélemények): List<Review> - A helyszínhez tartozó vélemények listája.

Review (Vélemény):

- author (Szerző): String - A vélemény szerzőjének neve.
- text (Szöveg): String - A vélemény szövege.
- rating (Értékelés): Int - A felhasználó által adott pontszám (1-től 5-ig).

generateSampleLocations (Minta Helyszínek Generálása):

Ez a függvény minta helyszíneket generál Budapest számára (park, étterem, látnivaló). A generált helyszínek adatait a Location osztály példányai tárolják.

generateSampleReviews (Minta Vélemények Generálása):

Ez a függvény egy megadott számú minta véleményt generál. A generált véleményeket a Review osztály példányai tárolják.

Navigációs logika

Függvény: Navigation

Leírás:

A Navigation függvény az alkalmazás alján elhelyezkedő navigációs sávot hozza létre. A navigációs sáv négy főbb képernyőre navigál:

- **Főoldal (Home):** Az alkalmazás alapértelmezett képernyője.
- **Keresés (Search):** A keresési képernyőre navigál.
- **Listák (Lists):** A listák képernyőre navigál.
- **Profil (Profile):** A felhasználó profiljához kapcsolódó képernyőre navigál.

Paraméterek:

- navController: A NavHostController példány felelős a képernyők közötti navigációért.

Adatmodell:

- NavbarItem (Navigációs Sáv Elem): Tárolja az egyes navigációs elemek adatait, mint a célhely (route), a megjelenő felirat (label) és az ikon.

Működés:

1. A Navigation függvény megkapja a navController példányt.
2. A currentBackStackEntryAsState segítségével lekérdezi az aktuálisan megjelenített képernyő útvonalát (route).
3. A navigationItems lista tartalmazza a navigációs sávon megjelenő elemek adatait (NavbarItem).
4. A NavigationBar komponens megjeleníti a navigációs sávot.
5. A NavigationBarItem komponensek segítségével kerülnek definiálásra az egyes navigációs elemek.
    - Az selected paraméter meghatározza, hogy az adott elem ki van-e választva.
    - Az icon paraméter az elemhez tartozó ikont definiálja.
    - A label paraméter az elemhez tartozó feliratot definiálja.
    - Az onClick paraméter felelős a navigációért. A kiválasztott elemre kattintva a navController.navigate függvénnyel történik a navigálás a megadott útvonalra (item.route).

Függvény: TravelApp

Leírás:

A TravelApp függvény az utazási alkalmazás fő képernyőjét hozza létre. Ez a képernyő az alkalmazás alapvető szerkezetét definiálja, beleértve a navigációs sávot és a megjelenített tartalmat.

Komponensek:

- **Scaffold:** A Material 3 keretrendszer komponense, amely az alkalmazás alapvető elrendezését biztosítja.
- **Navigation:** Az alkalmazás alján elhelyezkedő navigációs sávot megjelenítő komponens. (lásd dokumentációját fent)
- **NavHost:** A Jetpack Compose navigációs komponense, amely kezeli a képernyők közötti navigációt.

Navigáció:

- A navigációs sávon kívüli navigációs függvények a többi képernyőre való navigálásra szolgálnak, vagy a Budapesthez tartozó részletes oldalak (a bemutatáshoz szükséges):
    - **Bejelentkezés (SignIn):** Bejelentkezési képernyő.
    - **Regisztráció (SignUp):** Regisztrációs képernyő.
    - **Budapest:** Budapest város képernyője.
        - **Budapest látnivalmak (attractionBP):** Kijelölt budapesti látnivaló részletei.
        - **Budapest éttermek (restaurantBP):** Kijelölt budapesti étterem részletei.
        - **Budapest parkok (parkBP):** Kijelölt budapesti park részletei.
    - **Bécs (Vienna):** Bécs város képernyője.
    - **Róma (Rome):** Róma város képernyője.
    - **Térkép (Map):** Térkép képernyő. (Jelenleg nincs implementálva)

Adatmodell:

- A generateSampleLocations függvény által generált minta adatokat használja a Budapest képernyő és részképernyői.

Jövőbeli tervek

A projekt célja az volt, hogy egy biztos alapot adjon a szakdolgozathoz. Éppen ezért vannak olyan feature-ök amik nem készültek el és a következő félévben ezeket tervezem implementálni, amellett, hogy egy saját backend-et készítek, melyet feltöltök valós adatokkal. Ezen kívül természetesen a felület még nem tökéletes, ezt is tervezem tovább csiszolni. Összességében elégedett vagyok a féléves eredménnyel, sikerült szerintem az alkalmazást olyan szintre eljuttatni, hogy az bővíthető és folytatható legyen.