Haladási napló -

couchsurfing alkalmazás fejlesztése iOS platformon

**Szeptember 9-15.**

* Létrehoztam a GitHub repo-t, amelyben majd láthatóak lesznek a kódok. A repo linkje: <https://github.com/belbebe/couchsurfing_ios>
* Alkalmazásfejlesztői tapasztalat híján elkezdtem egy online kurzust, amely iOS fejlesztéssel foglalkozik
* A justinmind program segítségével elkezdtem az alkalmazás képernyő képeinek a megtervezését
* Létrehoztam egy nem végleges kliens oldali specifikációt, amelyet tovább szeretnék részletezni, illetve szükség esetén módosítok rajta, ez elérhető a következő linken:

<https://docs.google.com/document/d/1MG6L1ZPzAI2ftvTS2guLQ37nQHWmXCmHf22piukXJcw/edit?usp=sharing>

**Szeptember 16-22.**

* Átgondoltam az adatmodellt, és a csütörtöki konzultációs időpontig el is készítem a sematikus ábrázolását
* Részletesebben átgondoltam a kliens oldali specifikációt, ezt is frissítem majd a megfelelő dokumentumban - frissítés megtörtént
* Folytattam az online iOS app fejlesztésről szóló kurzust
* VIPER architektúrával való megismerkedés

**Október 28.**

* Megnéztem a REST API-kkal kapcsolatos videókat a kapcsolódó előadásokról (általános REST API-s videó, Swagger-rel kapcsolatos videó)

**Március 9.**

* View elkészítésének elkezdése
* Ütemterv jelen állás szerint:
  + UI elkészítése, adott esetben bizonyos funkciók implementálása: 5-6. hét
  + Backend elkészítése: 6. héttől 3-4 hét, lehetőleg minél hamarabb

**Március 16.**

* Foglalási oldal majdnem teljesen kész, egyéni igények jelzésével probléma (hogyan lehet kicsi multiselect list-et létrehozni), keresés gomb létrehozása még nem történt meg
* Működéshez szükséges „@Static” tagváltozók (property-k) létrehozása, ezekben a változókban történik az olyan adatok eltárolása perzisztens módon, mint például a textfield-ekből bevitt szövegek („source of truth”). Ahogy a user módosítja az ezekhez a változókhoz tartozó binding változók értékét, a @Static-kel ellátott változók értéke is dinamikusan változik, majd ezen változók alapján történik a View rebuild-elése.
* A Foglalási View összerakása során igény szerint alkalmaztam Vstack-eket és Hstack-eket, ezekben helyeztem el a különböző UI komponenseket, amelyekből létrehoztam a UI-t. Az egyes komponensekhez modifier-eket is hozzáadtam, ezzel létrehozva egy többé-kevésbé egységes kinézetet. A stack-eknél alkalmaztam a „spacing” paramétert, hogy az egyes komponensek egymástól elkülönítve jelenjenek meg.
* A felhasznált komponensek között user inputot lehetővé tevő komponensek is megtalálhatóak, pl. TextField komponensek, illetve DatePicker komponensek. A fizetési módot egy Picker komponenssel (<https://stackoverflow.com/questions/56513339/is-there-a-way-to-create-a-dropdown-menu-button-in-swiftui>) valósítottam meg, amelyben megtalálhatóak a fizetési lehetőségek, a user ezekből tud majd igény szerint választani.
* Az ár kategória meghatározásánál a TextField-ek inputjait szűröm, hogy csak numerikus értékek legyenek eltárolva a megfelelő @Static változókban. Ehhez működőképes megoldást a <https://stackoverflow.com/questions/58733003/swiftui-how-to-create-textfield-that-only-accepts-numbers> linken találtam.
  + Itt meg kell valósítani a keyboard eltűnését, illetve a begépelt érték hozzárendelését az adott változóhoz! <https://www.cometchat.com/tutorials/how-to-dismiss-ios-keyboard-swift>

**Március 23.**

* Belefutottam egy olyan hibába, hogy a Vstack, Hstack, stb konténerek csak 10 db View elemet (vagy abból származtatott komponenst) tudnak eltárolni -> megoldás: azoknak az elemeknek, amelyek már nem férnének bele az adott konténerbe, létrehozok egy „Group” objektumot, majd ezt az objektumot rakom a konténer utolsó szabad helyére. Azokat a View objektumokat, amelyeket hozzá szeretnék adni a konténerhez, egyszerűen csak belerakom a Group objektumba, és a kapott hiba megszűnik.
* Hozzáadtam a képernyőhöz egy „List” objektumot, amelyben az egyéb kívánt keresési paramétereket lehet majd kiválasztani (nem-dohányzó, állatbarát hely, stb).

<https://developer.apple.com/documentation/swiftui/list>

* + A „List” objektummal felmerült egy olyan probléma, hogy az egész View-ban alkalmazott ScrollView és a List olyan módon nem voltak kompatibilisek egymással, hogy a ScrollView a dimenzióit a benne található komponensektől várja, míg a List az őt befoglaló konténertől. Emiatt a List-nek nem volt definiálva méret, ami miatt nem lett renderelve a UI-ra. Erre megoldás, hogy definiáltam egy környezeti változót, amelyben megadtam a List egy sorának minimum magasságát, amelyet egy „frame” modifier-ben átadtam a List-nek.

<https://stackoverflow.com/questions/61437905/swiftui-list-is-not-showing-any-items>

* + További probléma, volt hogy a „NavigationView”-t, amelyet a „List”-hez használtam, nem ismerte fel a fordító, így teljesen ki kellett vinni a ScrollView-n kívülre.
  + A „List” elemeinek kiválasztása az „Edit” gomb megnyomásával történik. A kiválasztás egy „@State” változóban lesz eltárolva, ezáltal megvalósítva a többszörös kiválasztást, azonban a „Done” gomb megnyomásával eltűnik a kiválasztás a UI-n, de a változóban megőrzödik, újbóli „Edit” gomb megnyomással ismét megjelenik.
* Ezen felül hozzáadtam a Foglalás képernyőhöz egy gombot is, amellyel majd a keresést végre lehet hajtani. A gomb design-ja egyelőre meglehetősen kezdetleges, ezen majd javítok, hogy legyen kerete, háttérszín és a betűk színe is megfelelő legyen.

Feladat: „EgyebIgeny” osztály kiszervezése külön Swift fájlba, az egyes tab-oknak megfelelő view-k kiszervezése külön View fájlokba

* Segítség: <https://www.hackingwithswift.com/quick-start/swiftui/adding-tabview-and-tabitem>