**ŽILINSKÁ UNIVERZITA V ŽILINE**

**FAKULTA RIADENIA A INFORMATIKY**

**Vývoj aplikácií pre mobilné zariadenia**

Semestrálna práca

Pavol Adamík 5ZIY23

**Školský rok 2021/2022**

# Popis a analýza riešeného problému

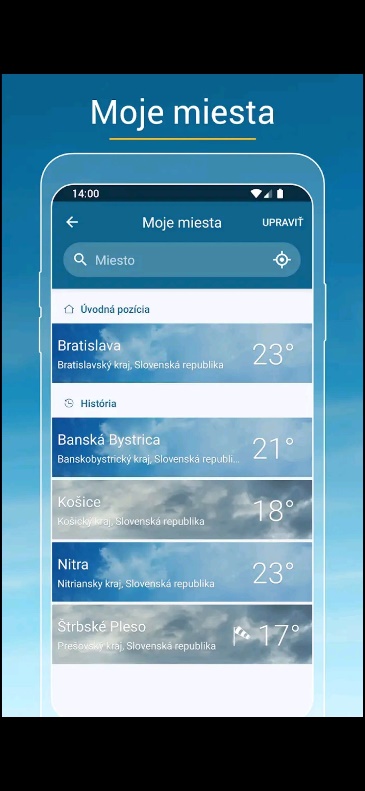
## Špecifikácia zadania

Za tému semestrálnej práce som si vybral vytvorenie aplikácie Weather / Počasie (ďalej len Weather). Respektíve dokončenie aplikácie a pridanie nových komponentov oproti tomu, čo sme sa naučili na iOS akadémii. Cieľom aplikácie je oboznámiť používateľa o počasí v aktuálnom meste kde sa práve nachádza, poprípade v meste ktoré chce vyhľadať na najbližších 7 dní, respektíve 48 hodín.

## Porovnanie iných aplikácií oproti mojej

Zo všetkých aplikácii, ktoré som našiel na google Play s podobným zameraním som sa rozhodol vybrať tieto dve aplikácie. *Počasie & Radar: výstrahy* (ďalej len Počasie A) a  *Počasie Slovensko* (ďalej len Počasie B).

Počasie A:



Obrázok, na ktorom je text, elektronika, snímka obrazovky, zobraziť

Automaticky generovaný popisObrázok, na ktorom je text, elektronika, zobraziť, snímka obrazovky

Automaticky generovaný popisPočasie B:

Obe aplikácie obsahujú základné prvky počasia ako je: predpoveď počasia na aktuálny deň, denná predpoveď, hodinová predpoveď, poloha, vyhľadávanie, detaily. Počasie A má oproti počasiu B aj meteoradar, kde si používateľ môže pozrieť oblačnosť, zrášky na konkrétnom mieste, graf 14 dňovej predpovede. Počasie B obsahuje navyše dátum, popis týždennej predpovede a pocitovú teplotu

Môj názor na Počasie A je taký, že je tam síce veľmi veľa funkcii, ale je zložitá na používanie, nie je až tak jednoducho spracovaná. Má v pozadí aktuálnu oblačnosť, čo býva často neprehľadné.

Počasie B je na druhú stranu moc jednoduché, teda aspoň môj prvotný dojem na jej dizajn bol taký. Myslel som si, že nejaké informácie tam chýbajú ale pre bežného používateľa sú podľa mňa dostačujúce. Nepáči sa mi akurát tá modrá farba, takže na také počasie by som sa nedokázal dlho pozerať.

Odlišnosti oproti mojej aplikácií:

Moja aplikácia neobsahuje:

* + - Nastavenia aplikácie
    - Animáciu stavu počasia (sneženie, pršanie, fúkanie atď.)
    - Kvalitu indexu vzduchu
    - Vlhkosť
    - Tlak
    - Uv index
    - Graf predpovedi počasia

Obsahuje navyše:

* Oproti Počasiu A dnešný dátum, pocitovú teplotu
* Jasnejší popis o aktuálnom dni
* Oproti Počasiu B názov aplikácie, veternosť v každej hodine
* Predpoveď počasia na 7 dní hneď na úvodnej obrazovke
* Informácie o výstrahách oproti počasiu A (aspoň som ich tam nenašiel)
* Možnosť zvoliť si čierne alebo biele pozadie
* Oddelená hodinová predpoveď
* Východ a západ slnka
* Preklad do slovenského jazyku

Ostatné prvky máme rovnaké, ale v inom prevedení.

Ukážka obrázkov z mojej aplikácie:

Obrázok, na ktorom je text

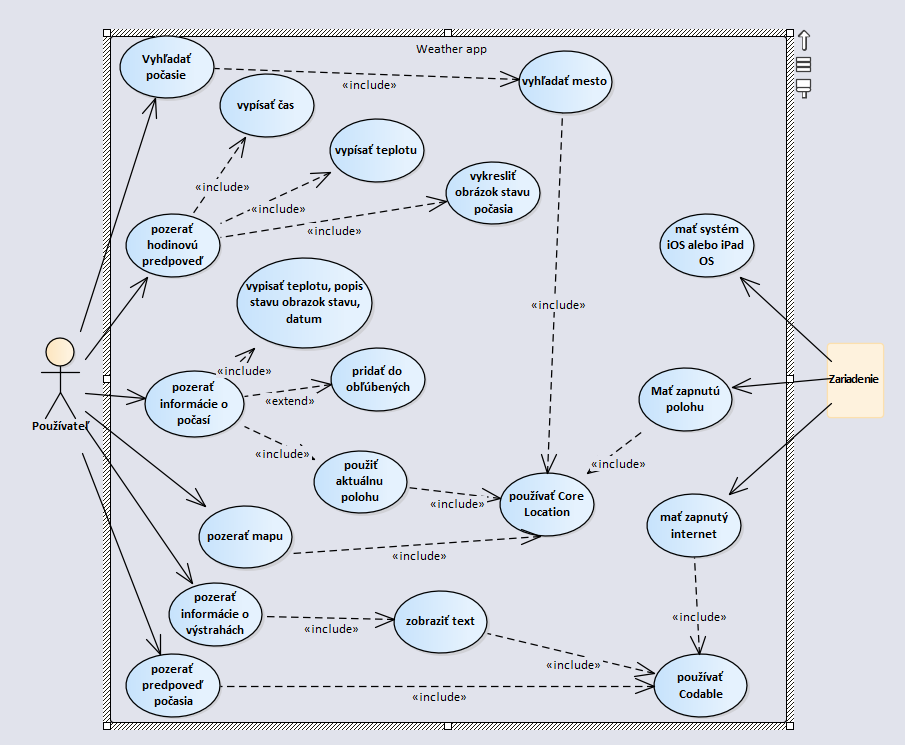
Automaticky generovaný popisObrázok, na ktorom je text, monitor, obrazovka, čierne

Automaticky generovaný popisObrázok, na ktorom je text, monitor, čierne, snímka obrazovky

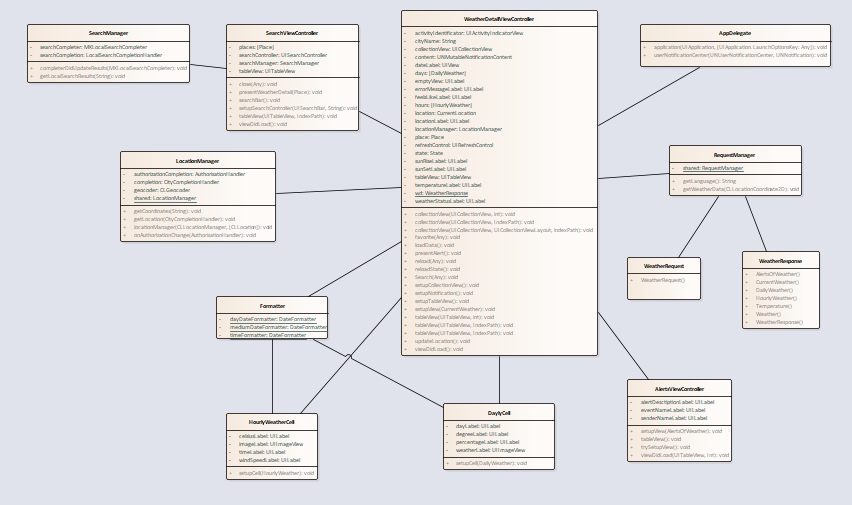
Automaticky generovaný popis

# Návrh riešenia problému

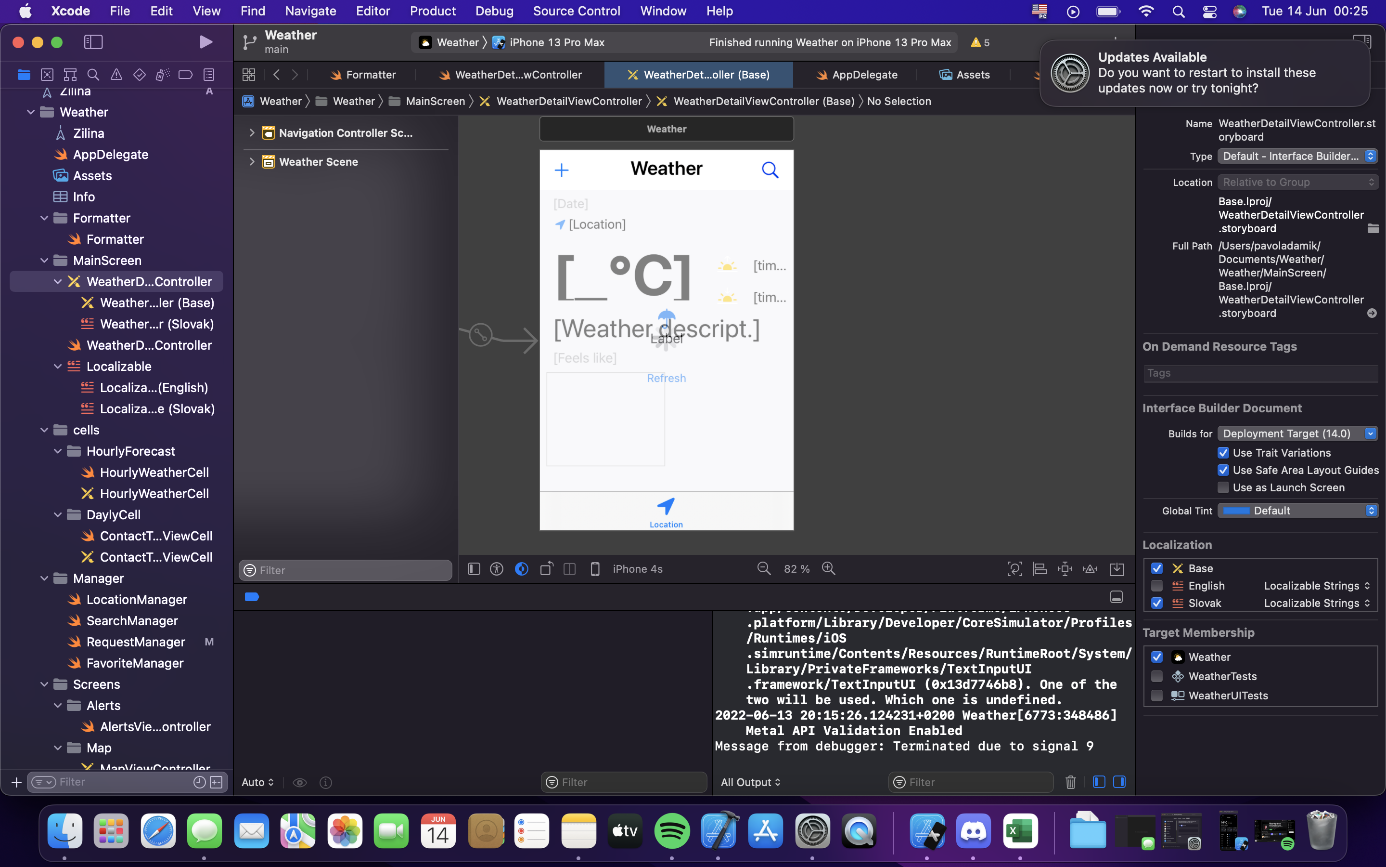
## Krátka analýza



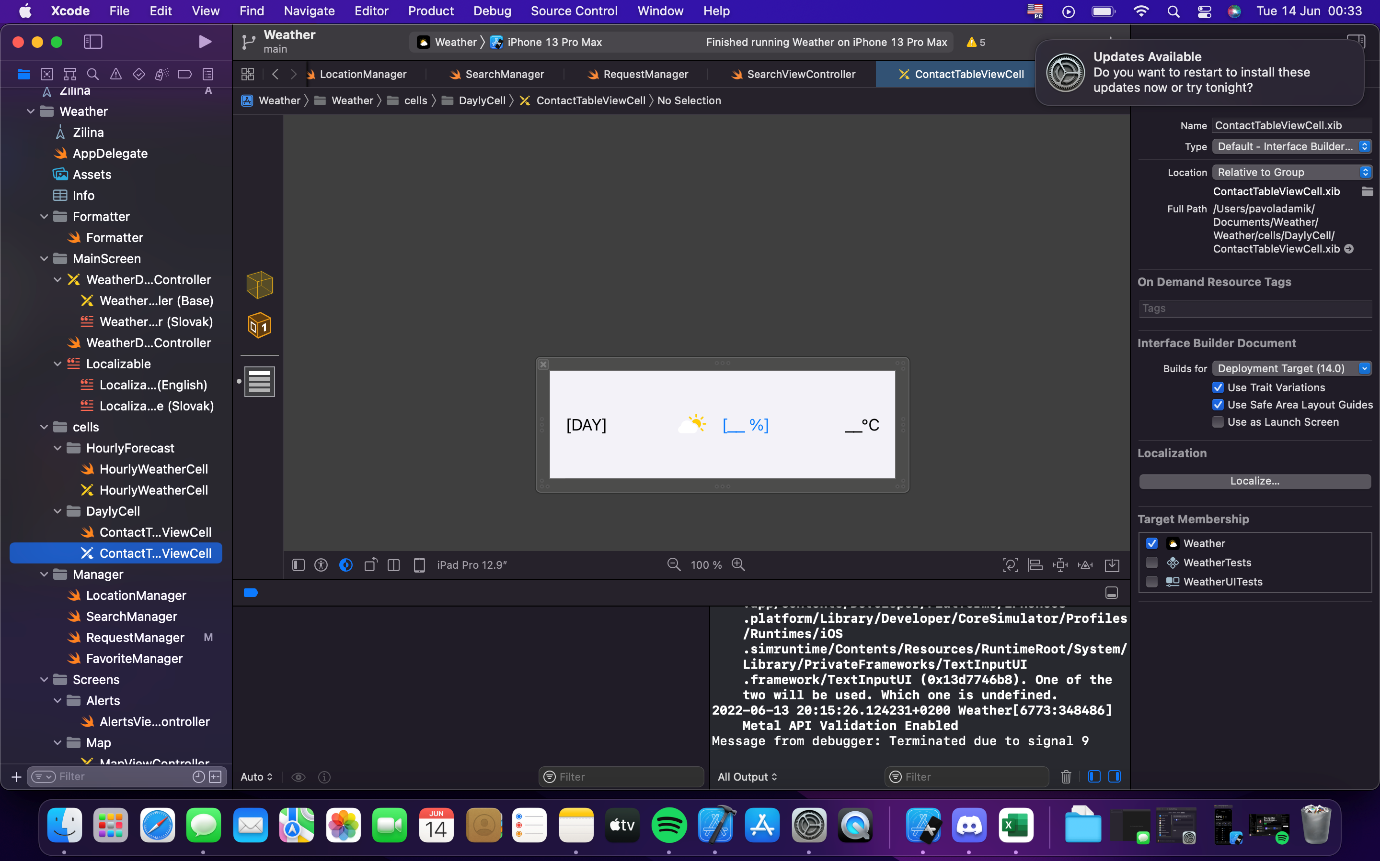
## Návrh aplikácie



# Popis implementácie



Vzhľadom na to, že rozhranie(interface) a dizajn sú v iOS aplikáciách výrazne dôležité, tak som sa zo začiatku zameral na UI (user interface). Keďže to je vec, ktorú používatelia vidia a to „predáva“. Aplikácia je tvorená pre všetky veľkosti iPhone a iPad pomocou funkcie AutoLayout. Po vyriešení grafickej stránky (viď obrázok vedľa) som sa zameral na správne načítanie dát a na správne formátovanie. Pri týchto veciach som riešil otázku UIKitu, Navigation Baru, Viewu, Labelov, Stack Viewu, Constraintov a AutoLayoutu.

Následne som pomocou UITableView a UITableViewCell vytvoril cellu, ktorú opäť trebalo správne zobraziť a naplniť platnými dátami.

Po dokončení tejto úlohy som riešil otázku lokácie. Aby aplikácia mohla pracovať so špecifickými koordinátami, musel som neimportovať CoreLocation a následne som mohol pracovať s CLLocation Manažérom a CLGeocoderom. Na to aby som mohol pracovať s používateľovou lokáciou, musel som spraviť tabuľku, ktorá sa používateľa spýta, či súhlasí s zapnutím polohy čím nám dáva informáciu, že môžeme pracovať s jeho lokáciou. K práci s touto lokáciou som si vytvoril triedu LocationManager, ktorá má na starosti otázky týkajúce lokácie. Vďaka tomu som ďalej mohol pracovať s informáciami týkajúcimi sa konkrétnej lokácie resp. mesta.

Následne som si vytvoril SearchController, ktorý pracuje taktiež s lokáciou. Pre prácu s vyhľadávaním som si spravil Search manažéra, ktorý sa stará o metódy ako: „počúvanie“ na klávesnicu a následné dekódovanie týchto dát. Resp. opačná operácia ako sme mali pri získavaní polohy z koordinátou. Doplňovanie slov ktoré začnem písať, nastavenie stavov objektu po vyhľadaní konkrétneho mesta.

Na to aby som mohol pracovať s konkrétnymi API dátami, musel som si neimplementovať Swift Package Manažéra - Alamofire, ktorý za mňa rieši veľa prípadov. Musel som sa zaregistrovať na stránku <https://openweathermap.org/api/one-call-api> odkiaľ čerpám dáta. Dáta sú obmedzené, čiže oproti prvotnému návrhu som mohol spraviť len predpoveď na 7 dní (nie na 14) a nemohol som použiť radar, ktorého použitie by bolo drahé : ). Pri práci s API dátami, som si ich musel najskôr dekódovať v JSON-e a následne som vytváral triedy ako WeatherResponse a WeatherRequest a na to, aby som si uložil môj unikátny API kľúč a spustil dekódovanie, tak som si vytvoril Request manažéra, s ktorým sa dá ďalej pracovať. Respektíve, on mi umožní pracovať už s dekódovanými dátami, ktoré potrebujem.

Obrázok, na ktorom je text, snímka obrazovky, monitor

Automaticky generovaný popisNásledne som si vytvoril HourlyCellu, ktorá sa líši od Dayly tým, že sa nepohybuje vertikálne ale horizontálne. Na to aby som mohol pracovať s takouto cellou, musel som pracovať už nie s tableView ale s collectionView, s ktorým sa pracovalo dosť podobne ako s tableView.

Po práci s API som si vytvoril TabBar, kde som si určil, ktorá obrazovka bude hlavná a ktoré ďalšie obrazovky chcem vytvoriť. Rozhodol som sa vytvoriť obrazovku, kde sa budú zobrazovať výstrahy, ich popis a autor, aby som sa naučil ako funguje ťahanie textov z API. Po úspešnom dokončení, som sa rozhodol preložiť aplikáciu do slovenčiny, načo som si vytváral nové String Files, kde som si vygeneroval konkrétne slová, resp. zakódované slová, ktoré som potom preložil. Nakreslil som si logo aplikácie, ktoré som následne naformátoval a spravil z neho ikonu aplikácie.

Ako poslednú vec som odstraňoval duplicity v kóde, ošetroval rôzne stavy, týkajúce sa padania aplikácie, autorizácie, povolenia, tabuľky pri nejakej systémovej chybe, animácie a podobne.

# Zdroje

<https://openweathermap.org/api/one-call-api>

goodrequest

https://www.hackingwithswift.com/100

<https://cocoacasts.com/ud-5-how-to-store-a-custom-object-in-user-defaults-in-swift>

https://www.wtfautolayout.com/