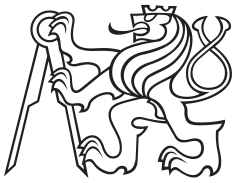
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra počítačové grafiky a interakce

Mobilní aplikace pro vzdálené ovládání chytrého termostatu se zónovým vytápěním

Šimon Kadlec

Vedoucí: Ing. Michal Vaňkát
Obor: Web a multimedia
Studijní program: Softwarové technologie a management
Květen 2017

Poděkování

Především chci poděkovat vedoucímu mé práce Ing. Michalu Vaňkátovi za podporu a trpělivost při psání této práce. Dále chci poděkovat Ing. Ivo Malému za poskytnuté konzultace. Také děkuji ČVUT, že mi je tak dobrou *alma mater*.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškerou použitou literaturu.

V Praze, 25. května 2017

Abstrakt

Tato práce pojednává o návrhu a implementaci mobilní aplikace pro vzdálené ovládání chytrého termostatu se zónovou regulací vytápění. Byly porovnány různé frameworky a na základě výsledku rešerše byl vybrán Ionic(v2), v němž byla aplikace naprogramována. Proběhlo testování návrhu uživatelského rozhraní testy bez uživatelů (heuristická evaluace, kognitivní průchod) i testy s uživateli. Výsledné uživatelské rozhraní bulo použito pro funkční prototyp aplikace.

Klíčová slova: hybridní, framework, termostat, ionic, mobile

Vedoucí: Ing. Michal Vaňkát
SE-VA technologies s.r.o.,
Fügnerovo náměstí 1808/3,
120 00 Praha 2

Abstract

This thesis deals with the design and implementation of multiplatform mobile application that can be used to remotely control a thermostat with zoned heating.For that purpose, there was conducted an evaluation of suitable frameworks and based upon that the most suitable framework Ionic (v2) was chosen to proceed with the implementation.The user interface design was tested with and without users using heuristic evaluation and cognitive walkthrough approaches.The resulting user interface made the base for the functional prototype of the mobile application.

Keywords: hybrid, framework, thermostat, ionic, mobile

Title translation: Mobile application to remotely control a thermostat with zoned heating

Obsah

Obrázky

Tabulky

Kapitola 1

Úvod

Chytré termostaty se zónovou regulací vytápění by v dnešní době, se zvyšujícími se nároky na úsporu energie, měly získávat stále významnější roli pro dosažení požadovaných úspor pomocí efektivní a účelné regulace vytápění. Firma SE-VA technologies s.r.o.[1], se sídlem Fügnerovo náměstí 1808/3, 120 00 Praha 2, se proto, po prozkoumání konkurenčního prostředí na trhu a po zajištění distribuční a odběratelské sítě, rozhodla pro vývoj svého vlastního termostatu se zónovým vytápěním.

Při vývoji jakéhokoliv produktu, je kladen velký důraz na požadavky zákazníků a jejich pohodlí při jeho používání. U regulačních zařízení obecně, chytrý termostat nevyjímaje, je velmi důležitá možnost jeho vzdáleného ovládání, typicky přes mobilní zařízení. Pro dosažení zákazníky požadované míry jednoduchosti a komfortu se jako nejvhodnější jeví mobilní aplikace. Jedním z předpokladů pro úspěch takového produktu na trhu je vhodný návrh uživatelského rozhraní. Dnes je trh daleko více nasycen zbožím, a díky tomu má zákazník více možností výběru. Proto je důležité, dříve opomíjenou věc - přívětivé uživatelské rozhraní, nepodcenit a věnovat jí velkou pozornost.

1.1 Cíl práce

Tato práce si klade za cíl realizovat mobilní aplikaci pro ovládání chytrého termostatu se zónovým vytápěním pomocí vhodně vybraného multiplatform-

[1] https://www.se-va.cz/

ního frameworku. Framework musí být zvolen tak, aby bylo možné pokrýt co nejvíce druhů chytrých mobilních zařízení s co nejmenšími náklady na vývoj.

Na začátek je důležité analyzovat práci s termostatem a dle toho definovat vhodné případy užití. Termostat zatím fyzicky neexistuje, proto definované případy užití nemusí být definitivní. Z logiky věci tedy není tato mobilní aplikace omezena požadavky již existujícího fyzického zařízení, nýbrž si klade za cíl definovat ideální požadavky na takové zařízení. Na základě výsledků této práce bude probíhat výběr či vývoj samotného fyzického zařízení a realizace jeho omezeného uživatelského rozhraní. Toto ovšem není cílem práce jako takové.

Práce se drží metodiky UCD (User Centered Design) a proto je její součástí návrh uživatelského rozhraní, jeho otestovaní testy bez uživatele - heuristické evaluace a kognitivního průchodu, a dále pak testy s uživateli. Následuje zapracování výstupu testů a vhodné upravení návrhu uživatelského rozhraní na jehož základě vznikne prototyp výsledné aplikace, který bude dále testován s uživateli na vzorových případech konfigurace.

1.2 Přínos aplikace

Na přínos realizované mobilní aplikace lze nahlížet z více rovin. Za prvé poskytne zadavateli cennou zpětnou vazbu od uživatelů pro samotný vývoj či výběr vhodného fyzického zařízení. Za druhé si řešitel osvojí práci s novými technologiemi v návrhu a vývoji mobilních aplikací v komerčním sektoru.

Kapitola 2

Analýza

Tato kapitola se věnuje analyzovaní práce s termostatem a definování případů užití. Také obsahuje rešerše existujících mobilních aplikací pro ovládání termostatů a nástrojů pro vývoj mobilních multiplatformních aplikací.

2.1 O využívaní termostatů

Ze studie [3], která analyzuje reálné užívání termostatů (i programovatelných) vyplývá, že většina uživatelů používá termostat pouze jako binární manuální vypínač topení, a že z programovatelných termostatů je skutečně naprogramovaných pouze zhruba 30%. Respondenti pro tuto studii si často stěžovali na špatné umístění termostatu v domě/bytě a z toho plynoucí mylné údaje z teplotních senzorů. Dalším z problémů byla složitost ovládání testovaných zařízení (byť někdy i jen zdánlivá). Největší procento termostatů bylo dle studie v režimu "hold", tedy nenaprogramovaných. Ačkoliv byla studie provedena na nereprezentativním vzorku respondentů, ukázala, čeho je dobré se vyvarovat a na co si dát pozor při návrhu aplikace.

2.2 Případy užití

Případy užití (UC - Use Cases) byly definovány za spolupráce se zadavatelem. Některé případy byly dodatečně přidány na základě podnětů z testování prototypu. Následuje seznam UC, ve formátu (číslo) - (rozhraní) - (uživatel) - (UC).

1. mobilní - nepřihlášený - Po startu aplikace mám možnost se přihlásit pomocí Facebook účtu.
2. mobilní - nepřihlášený - Po startu aplikace mám možnost se přihlásit pomocí Google účtu.
3. mobilní - nepřihlášený - Po startu aplikace mám možnost se přihlásit pomocí emailového účtu.
4. mobilní - přihlášený - Po přihlášení vidím hlavní menu s možností přidat termostat, v případě že už termostat mám propojený se svým účtem, zobrazí se mi seznam propojených termostatů (tlačítko na přidání stále zůstává).
5. mobilní - přihlášený - Možnost změnit jednotky stupňů C/F.(přidán na základě výsledků testování prototypu)
6. mobilní - přihlášený - Po zvolení termostatu vidím přehledovou obrazovku termostatu – tzv. dashboard.
7. mobilní - přihlášený - Na dashboardu vidím název termostatu a ovládací prvek pro nastavení parametrů termostatu.
8. mobilní - přihlášený - Na dashboardu vidím ovládací prvek a graf naměřených hodnot.
9. mobilní - přihlášený - Na dashboardu vidím aktuální měřenou a požadovanou teplotu, aktuální režim (komfort, útlum, ...) jako hlavní údaje zobrazené dominantně.
10. mobilní - přihlášený - Na dashboardu vidím grafické znázornění časového programu a ovládací prvek pro časové programy.
11. mobilní - přihlášený - Mám možnost nastavit časové programy na každý den v týdnu.
12. mobilní - přihlášený - Mám možnost zobrazit graf měřené a požadované hodnoty přes (téměř) celé okno, v grafu je vidět průběh měřené hodnoty a požadované hodnoty (jasně oddělené graficky).

13. mobilní - přihlášený, Mám možnost definovat režimy termostatu, na výběr je z několika režimů, ze kterých mohu vybírat.

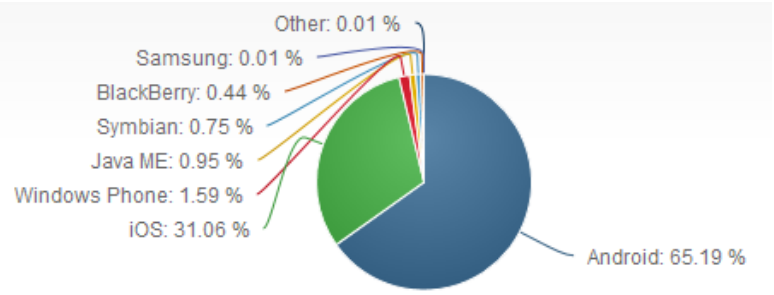
14. mobilní - přihlášený - Mám možnost zobrazení časového programu přehledu celého týdne.

15. mobilní - přihlášený - Zvolím-li konkrétní časové pole, zobrazí se nabídka s výběrem režimu, který bude v danou chvíli aktivován + přesné časové určení změny.

2.3 Obecně o přístupu k vývoji mobilních aplikacích

2.3.1 Mobilní platformy

V této podsekci se práce věnuje procentuálnímu zastoupení mobilních platforem na trhu smartphone a tablet PC. Porovnání, jak tomu bylo v květnu 2017, lze vidět v grafu 2.1 a tabulce 2.1 (převzato z [5]).



Obrázek 2.1: Procentuální zastoupení jednotlivých platforem na trhu.

Z tabulky vyplývá, že nejdůležitější jsou první dvě platformy s největším podílem na trhu, tzn. Android[1] a iOS[2]. Zde je základní shrnutí rozdílu v přístupu k vývoji a návrhu uživatelského rozhraní, podrobnějšímu porovnání se zde věnovat nebudeme. Následující tabulka 2.2 [1] shrnuje hlavní rozdíly mezi platformami. Zkratky použité v tabulce 2.2: VM (Virtual Machine), IDE (Integrated Development Environment), OS (Operační Systém).

[1] https://www.android.com/
[2] https://developer.apple.com/ios/

Operační systém	Podíl na trhu
Android	47.18%
iOS	42.60%
Java ME	3.35%
Symbian	3.32%
Windows Phone	2.44%
BlackBerry	0.97%
Kindle	0.07%
Samsung	0.04%

Tabulka 2.1: Procentuální zastoupení jednotlivých platforem na trhu.

	Android	iOS
OS pro vývoj	Více	Mac OS
Programovací jazyk	Java	ObjectiveC,Swift
UI	XML	Cocoa Touch
VM	Dalvik VM	žádný
IDE	Android Studio	XCode
Obchod	Google Play	App Store

Tabulka 2.2: Porovnání mobilních platforem.

2.3.2 Přístupy k vývoji mobilní aplikace

Rozdělování přístupu k vývoji mobilní aplikace do kategorií je obtížné, často nejednoznačné[1]. Pro účely této práce vystačí rozdělení přístupu k vývoji mobilních aplikací do tří (resp. čtyř) kategorií.

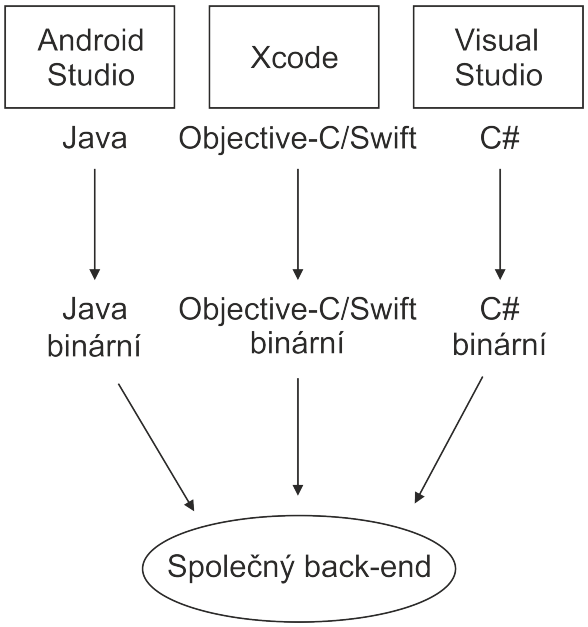
- Nativní aplikace
  - Čistě nativní (co platforma to jiný zdrojový kód)
  - Multiplatformně nativní (kompilace zdrojového kódu, jiného než nativního, do nativního pro každou platformu)
- Hybridní aplikace
- Mobilní web

Každý přístup vyhovuje jinému typu a použití aplikace. Před začátkem vývoje je dobré projít všechny možnosti a zvolit vhodný způsob.

Dle zadání má být výsledná aplikace multiplatformní, pro úplnost budou rozebrány všechny výše zmíněné kategorie.

Nativní aplikace

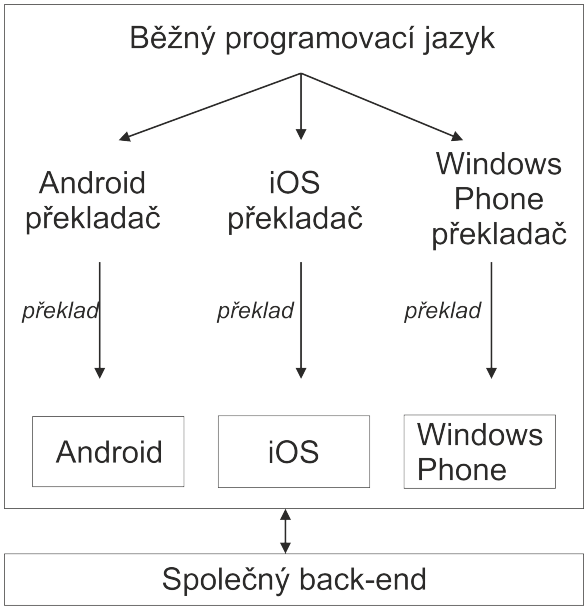
V současnosti je tento způsob nejrozšířenější. Důvodem je především, že tento přístup je doporučován samotnými vývojáři jednotlivých platforem (Google, Apple). Multiplatformní aplikace pro ně nejsou prioritou, i když v poslední době můžeme zaznamenat v tomto směru snahu i od některých vývojářů platforem, Googlu a Microsoftu (Xamarin[4]).



Obrázek 2.2: Nativní přístup.

Čistě nativní přístup (obrázek 2.2) přináší největší možnosti (funkční a při návrhu uživatelského rozhraní) a nejlepší výkon[2]. Ovšem za cenu řešení problémů na každé platformě zvlášť. Další možností je právě multiplatformně-nativní vývoj (obrázek 3.1). Výsledné aplikace jsou nativní a ušetří čas při vývoji, protože zhruba 70 % kódu je společných a zbylých 30 % je opět závislých na platformě. Mezi takové patří například NativeScript[3], CodenameOne[4], Xamarin[5].

[3]https://www.nativescript.org/
[4]https://www.codenameone.com/
[5]https://www.xamarin.com/



Obrázek 2.3: Multiplatformně-nativní přístup.

Výhody nativního vývoje.

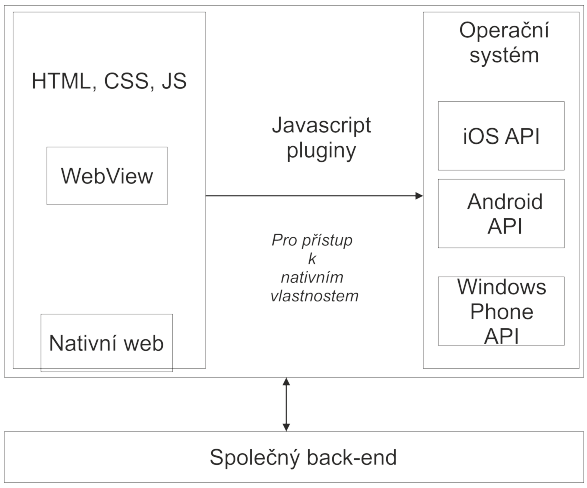
- Výkon – oproti hybridní aplikaci [2] nebo mobilní web aplikaci jsou psány ve vhodnějších jazycích pro dané platformy.
- Podpora nejnovějších API (Application Programming Interface) – možnost využití všech API telefonu.
- Nativní UI (User Interface) prvky – UI prvky jsou pro vývojáře připraveny přímo pro danou platformu a nemusí tak řešit jejich kompatibilitu.

Nevýhody (čistě) nativního vývoje.

- Vysoká cena – nutnost programovat, testovat a udržovat každou platformu zvlášť.
- Nároky na programátora – vývojář musí znát více programovacích jazyků pro různé platformy a více nativních SDK (Software Development Kit), nebo je zapotřebí větší tým vývojářů.
- Náročné opravy – na každé platformě se provádí zvlášť.

Hybridní aplikace

Každá běžná platforma obsahuje tzv. WebView, neboli instanci webového prohlížeče. Tohoto využívají hybridní aplikace – jejich základem je právě WebView, které zobrazuje pro telefony optimalizovanou webovou aplikaci, strukturu můžeme vidět na obrázku 2.4. Dříve byl hlavním nedostatkem hybridních aplikací nedostatečný výkon WebViews nebo nedostupnost systémových API při jejich volání z WebView. Dnes se i na WebView dá udělat aplikace s dobrým UX (User Experience), tak že ji z uživatelského pohledu nelze rozlišit od nativní.



Obrázek 2.4: Hybridní přístup.

Android a iOS se od sebe koncepcí uživatelského rozhraní výrazně neliší. Existují frameworky, které své UI prvky automaticky přizpůsobuje tzv. guidelines[6] dané platformy. Díky tomu stejná aplikace na iOS a Android bude vypadat různě (komponenty budou přizpůsobené platformě).

Velmi často využívaným řešením pro vývoj hybridních mobilních aplikací je framework Cordova[7]. Jde o framework v nativním kódu platformy, který typicky využívá WebView zvětšený na celou obrazovku, kde běží webová aplikace. Tuto aplikaci lze propojit s API telefonu (fotoaparát, GPS, gyroskop, atd.) pomocí pluginů, které zajišťují komunikaci mezi WebView (Cordovou) a platformou.

Výhody hybridního vývoje.

[6] Doporučený postup
[7] https://cordova.apache.org/

- Multiplatformnost – hlavní přednost tohoto přístupu. Zpravidla levnější vývoj aplikace.
- Snadné testování – 90 % práce lze simulovat v prohlížeči. Odpadá často zdlouhavé nasazování aplikace.

Nevýhody hybridního vývoje.

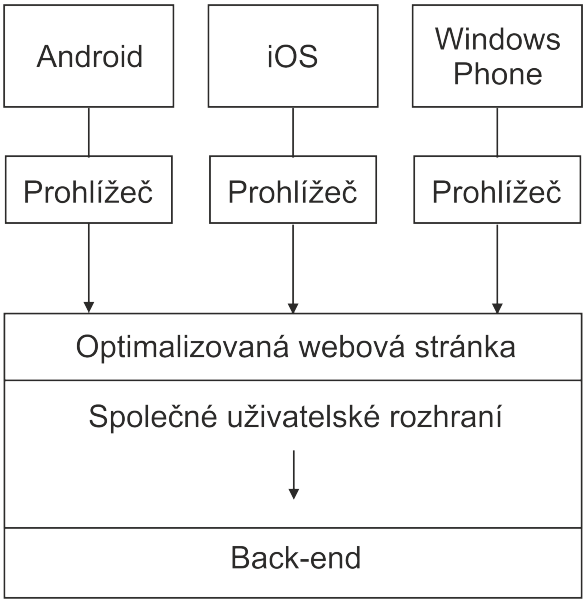
- Vyšší hardwarové požadavky – stále jde o webovou aplikaci běžící v prohlížeči, to znamená hardwarová omezení. Tento nedostatek se minimalizuje se vzrůstajícím výkonem koncových zařízení a prohlížečů.
- Nedostupnost API – neustálý vývoj nových API, pro přístup k novým funkcím telefonu. Může se proto stát, že nově dostupná API nejsou určitým multiplatformním frameworkem podporována.
- Absence nativních UI prvků – častá absence UI prvků, na které jsou uživatelé jednotlivých platforem zvyklí. Napodobení rozhraní nativních aplikací není jednoduché.

Mobilní web

Pokud by mobilní aplikace měla kopírovat funkcionalitu z již běžící webové aplikace, je potřeba zjistit, zda není snazší udělat optimalizovaný mobilní web. Mobilní web se hodí spíše pro malé a nenáročné aplikace, bez požadavků na speciální funkčnost. Předpokladem je i uživatelův přístup na internetu. Když nestačí pouhé přestylizování, lze vytvořit mobilní web jako samostatnou aplikaci a zobrazovat ji pouze na zařízeních s menším displejem. Tento přístup se využívá při vývoji větších a složitějších webových aplikací. Strukturu lze vidět na obrázku 2.5.

Výhody webového vývoje.

- Rychlost vývoje – u jednoduchých existujících webů často stačí změna CSS (Cascading Style Sheets).
- Univerzálnost – není potřeba měnit kód ve více aplikacích.
- Není nutné instalovat aplikaci do telefonu.
- Rychlá distribuce aplikace.



Obrázek 2.5: Webový přístup.

Nevýhody webového vývoje.

- Horší uživatelská zkušenost – bez dostatečných knihoven se web těžce přizpůsobuje, aby z něj měl uživatel podobný dojem jako z nativní nebo hybridní aplikace.
- Absence napojení na API – omezená možnost napojení na API se tu nabízí (GPS).
- Chybí nativní UI prvky – vše se musí nastylovat, nebo se musí použít knihovny, které požadované UI prvky implementují.
- Nutnost připojení k internetu a fyzická absence aplikace v telefonu.

Srovnání pro a proti jednotlivých přístupů

Příklady typů aplikací, které se hodí nebo nehodí pro jednotlivé přístupy.

- Nativní aplikace
  - Vhodné pro: složité animace, 2D/3D hry, aplikace určené pro jednu platformu, aplikace vyžadují méně používaná API.

  - Nevhodné pro: vývojářsky jednoduché aplikace, prototypování, multiplatformní aplikace.

- Hybridní aplikace
  - Vhodné pro: 2D hry, proto-typování, aplikace s důrazem na nízkou cenu, aplikace pro stahování dat z API, multiplatformní aplikace.
  - Nevhodné pro: 3D hry, složité animace, rozšířená realita, výpočetně náročné aplikace.

- Mobilní web
  - Vhodné pro: jednoduché aplikace s důrazem na nízkou cenu, aplikace nevyžadující častý přístup k API, aplikace, které nemusejí dodržovat guidelines, aplikace, které jsou založené na webové verzi.
  - Nevhodné pro: načítání kontaktů z telefonu, také všechno co hybridní aplikace.

Na internetu[8] lze nalézt i jednoduché analýzy přístupů k vývoji mobilních aplikací z hlediska ceny, doby vývoje a spotřeby personálních zdrojů k vývoji. Ceny vývoje u nativních aplikací jsou v průměru jednou až dvakrát vyšší než u zbylých dvou přístupů, často je to způsobené větším počtem lidí potřebných k vývoji. Podle kritéria délky vývoje jasně vítězí mobilní web, ale výrazně zaostává, když má vyvíjená aplikace přistupovat k API.

2.4 Analýza existujících řešení aplikací pro vzdálené ovládání termostatu

Na trhu v současné době lze najít mnoho firem, které nabízejí termostaty se zónovým vytápěním a mají pro svůj produkt i mobilní aplikace, nabízející tak pohodlnější ovládání. Následuje rešerše dostupných aplikací, pro mobilní platformu Android - Google Play. Takových aplikací je více a mnohé z nich mají, na základě negativního uživatelského hodnocení, nešťastně navržené UI. V rámci rešerše byly vybrány aplikace, které po nainstalovaní nabízejí demo program, a je možné si je otestovat. Jedná se o aplikace iT600[9], Netatmo thermostat[10] a eModul[11]. Po vyzkoušení těchto aplikací bylo možné určit co je u jednotlivých aplikacích z uživatelského hlediska povedené a co nepovedené. Následně bylo možné se z toho při návrhu rozhraní a definovaní UC poučit.

[8] http://janvaclavik.cz/jak-vyvijet-mobilni-aplikace/
[9] https://play.google.com/store/apps/details?id=com.computime.it600
[10] https://play.google.com/store/apps/details?id=com.netatmo.thermostat
[11] https://play.google.com/store/apps/details?id=pl.techsterowniki.emodul

2.5 Výběr vhodného frameworku pro vývoj aplikace

Hlavní kritériem při výběru bylo, že se musí jednat o hybridní framework, pro snadný vývoj na více platformách současně. Frameworků, které tento požadavek splňují, je mnoho. Proto jsou z tohoto množství vybrány ty, které jsou zmíněny v odborných článcích [1] [2] a jsou také často zmiňované a doporučované v internetových diskuzích a článcích[6]. Jde o dva frameworky hybridního typu - Ionic[12] a Onsen[13], a čtyři multiplatformně nativního typu - Xamarin, Titanium[14], RhoMobile[15] a Codename One. Následující podkapitoly a tabulky porovnávají jednotlivé možnosti, které frameworky nabízí (čerpáno ze stránek jednotlivých frameworků). Některá kritéria výběru jsou podrobněji popsána dále.

2.5.1 Porovnání licence

Softwarová licence je v informatice právní nástroj, který umožňuje používat nebo redistribuovat software, který je chráněn zákonem. V České republice se jedná o Autorský zákon. Je několik typů, pro tuto práci jsou relevantní hlavně MIT Licence[16], GNU Licence[17], Apache Licence[18] a iniciativa, která je sdružuje OSS[19] (Open-Source software). Porovnání typů licencí u vybraných frameworků, tabulka 2.3.

- MIT Licence – Software uvolněný pod touto licencí je možné použít jak v proprietárním software (s podmínkou, že text licence MIT musí být dodáván spolu s daným software), tak i s GPL (General Public License) licencovaným software (díky tomu, že GPL explicitně povoluje kombinaci s licencí MIT).
- GNU GPL – GPL je nejpopulárnějším a dobře známým příkladem silně copyleft licence, která vyžaduje, aby byla odvozená díla dostupná pod toutéž licencí. V rámci této filosofie je řečeno, že poskytuje uživatelům počítačového programu práva svobodného softwaru a používá copyleft

[12] http://ionicframework.com/
[13] https://onsen.io/
[14] http://www.appcelerator.org/titanium
[15] http://docs.rhomobile.com/en/5.4/home
[16] https://opensource.org/licenses/MIT
[17] https://www.gnu.org/licenses/licenses.en.html
[18] https://www.apache.org/licenses/
[19] https://opensource.org/licenses

k zajištění, aby byly tyto svobody ochráněny, i když je dílo změněno nebo k něčemu přidáno. Toto je rozdíl oproti permisivním licencím svobodného softwaru, jejímž typickým případem jsou BSD (Berkeley Software Distribution) licence.

- Apache Licence - Požaduje po uživateli zachování autorství (copyright) a tzv. disclaimer, tedy zřeknutí se odpovědnosti. Jako ostatní licence tohoto druhu, i Apache umožňuje uživateli svobodné užívání softwaru k různým účelům: distribuci, upravování, následné redistribuci upravené verze softwaru, apod. To vše je možné, aniž by došlo k porušení licenčních práv.

- OSS– Jde o počítačový software s otevřeným zdrojovým kódem. Otevřenost zde znamená jak technickou dostupnost kódu, tak legální dostupnost – licenci software, která umožňuje, při dodržení jistých podmínek (ku příkladu GNU licence), uživatelům zdrojový kód využívat, například prohlížet a upravovat.

Framework	Typ licence
Ionic	MIT
Onsen	OSS(Apache2.0)
Xamarin	OSS (MIT)
Titanium	OSS(Apache2.0)
RhoMobile	MIT
CodenameOne	OSS(GPLv2)

Tabulka 2.3: Porovnání licencí.

2.5.2 Programovací jazyk

Programovací jazyk je kritérium důležité hlavně pro cílovou platformu a pro vývojáře aplikace. Je velkou výhodou, když je vývojář s jazykem daného frameworku již seznámen. Hybridní aplikace se nejčastěji programují pomocí JS[20] (JavaScript)/TS[21] (TypeScript), HTML5 (HyperText Markup Language) a CSS, u multiplatformně nativních frameworků je škála programovacích jazyků podstatně rozmanitější, tabulka 2.4.

[20] https://www.javascript.com/
[21] https://www.typescriptlang.org/

Framework	Programovací jazyk
Ionic	JS, HTML5, CSS
Onsen	JS, HTML5, CSS
Xamarin	C#
Titanium	JS
RhoMobile	JS, HTML5, CSS, Ruby
CodenameOne	Java

Tabulka 2.4: Porovnání programovacích jazyků.

2.5.3 Porovnání nabízených služeb

Porovnání toho, co nabízejí jednotlivé frameworky zdarma v základní verzi a toho, co nabízejí za poplatek v nejlevnějším nabízeném balíčku služeb. Tabulky 2.5 a 2.6.

Framework	Zdarma
Ionic	CLI; ionicCreator(omezená funkčnost), developer licence (1 projekt, veřejné projekty, základní exportování)
Onsen	zdarma
Xamarin	Xamarin SDK, OSS development, pro malé týmy
Titanium	Appcelerator Studio IDE, liveview, api builder, historie uživatelských statistik 1 měsíc
RhoMobile	RhoStudio, Rhodes (lokální kompilace)
CodenameOne	podpora komunity, kompilace do Androidu, iOS, Windows Phone, BlackBerry

Tabulka 2.5: Porovnání vlastností nabízených zdarma.

Framework	Nejlevnější nabízené balíčky	Cena
Ionic	ionic Creator – neomezeně projektů, soukromé projekty + další nástroje	24$/měsíc (účtováno ročně)
Onsen	zdarma	
Xamarin	Xamarin Studio Professional a SDK, žádné omezení používání, přístup k bonusovému obsahu Xamarin University	cena na míru
Titanium	app designer, app preview, historie uživatelských statistik 3 měsíce	99$/měsíc (účtováno ročně)
RhoMobile	RhoElements, kompilace v cloudu (Rhodes/RE), Visual Studio plug-in, Sync (RhoConnect) – 1000 zařízení, Push 1000 zařízení, App Management – 1000 zařízení	299$/měsíc
CodenameOne	neomezený počet aplikací, přístup k nativnímu kódu	19$/měsíc

Tabulka 2.6: Porovnání zpoplatněných vlastností.

2.5.4 Platformy podporované výslednou aplikací

Jak už je výše zmíněno, pro tuto práci je důležitá podpora platforem Android a iOS. Podpora pro ostatní platformy může být výhodou. Tabulka 2.7.

2.5.5 Doporučené IDE pro vývoj aplikací

Některé frameworky nabízejí vlastní IDE přizpůsobené pro vývoj aplikací v daném frameworku. Jiné nabízejí pluginy do stávajících vývojových prostředí. Vhodně zvolené vývojové prostředí může vývojáři usnadnit mnoho práce, proto se vyplatí tuto volbu nepodcenit. Tabulka 2.8.

Framework	Podporované platformy
Ionic	Android, iOS, BlackBerry 10, Ubuntu, Windows Phone (8, 8.1, 10)
Onsen	Android, iOS, BlackBerry 10, Ubuntu, Windows Phone (8, 8.1, 10)
Xamarin	Android, iOS, Windows Phone, Windows Store apps
Titanium	iOS, Android, Windows
RhoMobile	Android 1.6+, iOS 3.0+, Windows Mobile 6.1 Professional, Windows Mobile 6.0 Standard, BlackBerry, Symbian
CodenameOne	Android, iOS, Windows Phone, BlackBerry

Tabulka 2.7: Porovnání podporovaných platforem.

Framework	Doporučené IDE
Ionic	-
Onsen	Monaca IDE
Xamarin	Xamarin Studio (Mac), VisualStudio (Windows)
Titanium	Titanium IDE
RhoMobile	Xcode, Eclipse, RhoStudio IDE
CodenameOne	Eclipse, NetBeans, IntelliJ/IDEA

Tabulka 2.8: Porovnání IDE.

2.5.6 Doplňující informace

Informace pro které nejsou potřebné samostatné tabulky - zda je nutné SDK pro jednotlivé platformy a zda je výstupem i desktopová aplikace. Tabulka 2.9.

2.5.7 Subjektivní hodnocení frameworků

Subjektivní hodnocení jednotlivých frameworků je napsáno na základě výše zmíněných kritérií a podle požadavků na funkčnost aplikace. Při hodnocení je přihlédnuto i k objektivním výsledkům testů, bohužel data nejsou dostupná pro všechny porovnávané frameworky[2]. Systém hodnocení je zvolen jako

Framework	Nutnost SDK platformy	Desktopová aplikace
Ionic	ne	ano
Onsen	ne	ano
Xamarin	ano (pro Android – Windows, Linux; pro iOS - Mac)	ne
Titanium	ano (pro Android – Windows, Linux; pro iOS - Mac)	ne
RhoMobile	ano (pro Android – Windows, Linux; pro iOS - Mac)	ne
CodenameOne	ano (pro Android – Windows, Linux; pro iOS - Mac)	ne

Tabulka 2.9: Porovnání doplňujících vlastností.

ve škole, čím nižší známka tím lepší hodnocení. Rozsah hodnot je 0 – 4. Frameworky jsou hodnoceny na základě 4 nejdůležitějších kategorií. Tabulka 2.10.

Framework	Licence	Jazyk	Cena	Platformy	Typ aplikace	Celkem
Ionic	0	1	1	1	0	3
Onsen	0	1	2	1	0	4
Xamarin	0	2	1	2	1	5
Titanium	0	2	0	2	1	5
RhoMobile	0	2	2	0	1	5
CodenameOne	0	1	1	2	1	5

Tabulka 2.10: Subjektivní hodnocení frameworků.

2.5.8 Finální výběr

Po provedení rešerše nástrojů pro vývoj multiplatformních aplikací, s přihlédnutím k možnému vývoji desktopové aplikace a po konzultaci se zadavatelem práce bylo rozhodnuto, že nevhodnějším nástrojem pro vývoj aplikace na ovládání chytrého termostatu se zónovým vytápěním je framework Ionic(v2). Z porovnávaných frameworků vyniká nabízenými UI prvky, které jsou jako nativní prvky platforem. Díky jeho propojení s Cordova nabízí i dobré propojení s nativními funkcemi zařízení.

Kapitola 3

Návrh aplikace a uživatelského rozhraní

Tato kapitola se věnuje návrhu uživatelského rozhraní, jeho následnému testování a vyhodnocení testů ve formě nového návrhu. Tento návrh se již stal základem ze kterého se vychází při implementaci. První návrh uživatelského rozhraní vznikl ve formě tzv. mockupů, tedy jednoduchých prototypů, na základě definovaných případů užití. Při návrhu byly dodržovány Material Design guidelines[1] navrhnuté firmou Google pro webové aplikace a platformu Android.

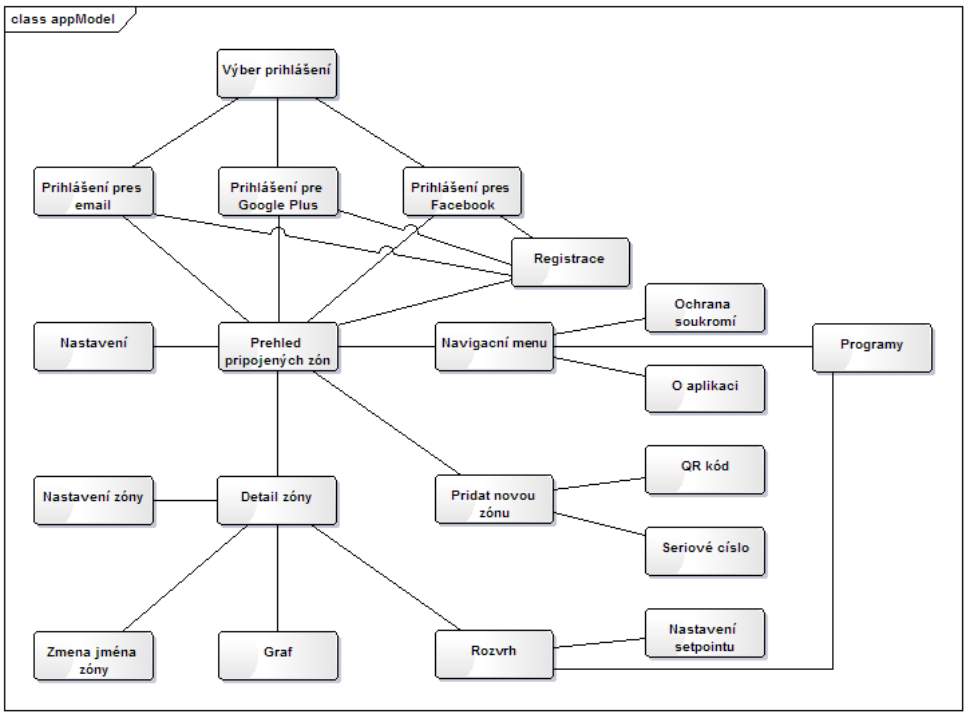
3.1 Logická struktura aplikace

Třídní model reprezentující propojení stránek a tím i základní strukturu aplikace.

3.2 První kolo návrhu uživatelského rozhraní

Bez předchozích zkušeností v oblasti návrhu uživatelského rozhraní byl začátek práce zajímavější. Inspirací byly všechny ostatní aplikace, které byly k dispozici. Hlavním zdrojem byly aplikace od Google, protože u nich lze předpokládat přívětivé uživatelské rozhraní.

[1] https://material.io/guidelines/



Obrázek 3.1: Model aplikace.

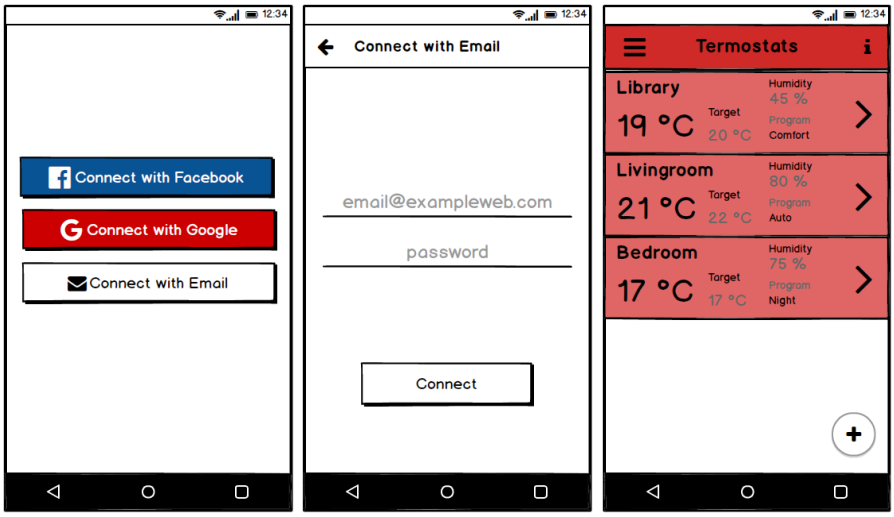
3.2.1 Mockupy

Pro výrobu mockupů (modelu) byla využita aplikace Balsamiq Mockups[2]. Ta umožňuje jako jeden z výstupů PDF dokument propojený odkazy, na kterém byly následně prováděny testy s uživateli i bez uživatelů. První návrh uživatelského rozhraní (obrázky 3.2, 3.3, 3.4, 3.5, 3.6).

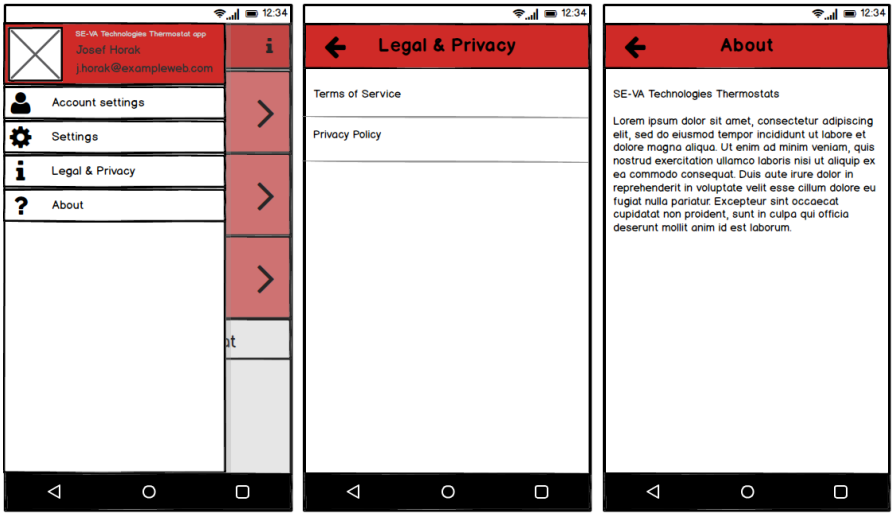
3.2.2 Testování

Tato kapitola se věnuje testování prvního návrhu uživatelského rozhraní. Návrh byl otestován kognitivním průchodem. Poté byl otestován dvěma experty a na závěr byl otestován se třemi uživateli.

[2]https://balsamiq.com/



Obrázek 3.2: Výběr přihlášení do aplikace, přihlášení do aplikace přes email, hlavní obrazovka.

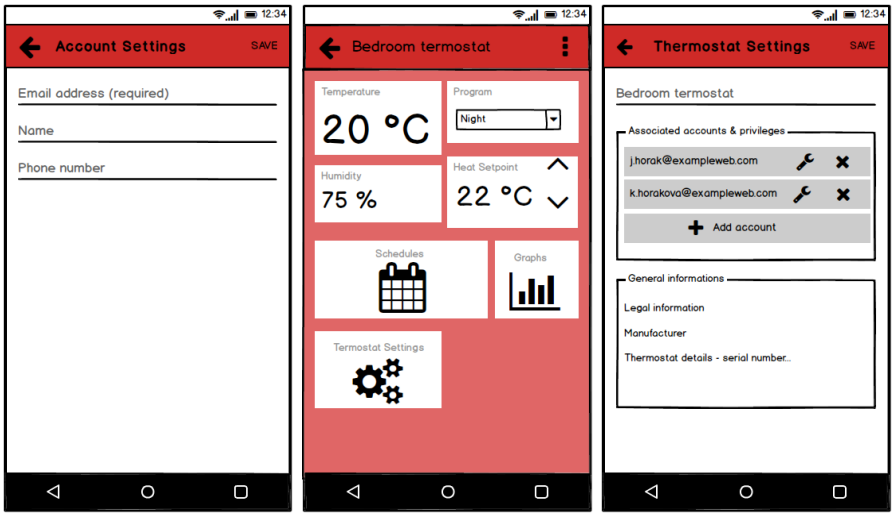


Obrázek 3.3: Navigační menu, soukromí a právní informace, o aplikaci.

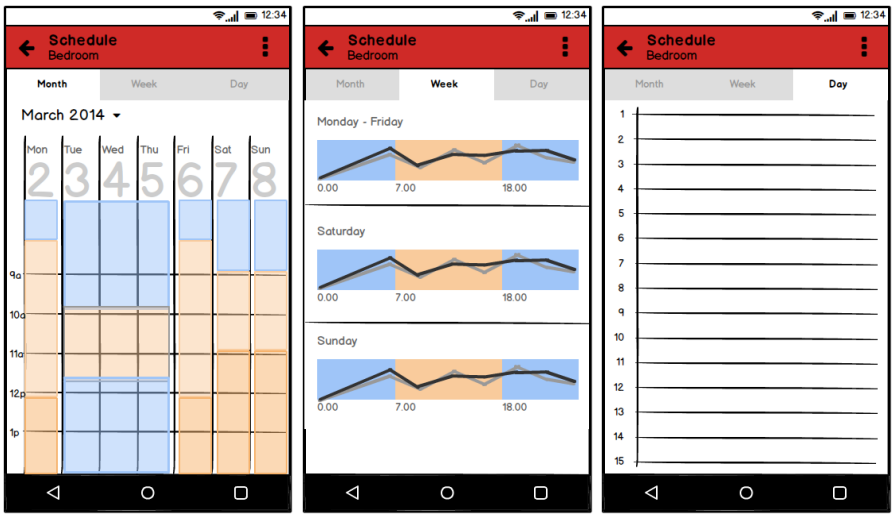
Testování bez uživatelů - kognitivní průchod

Kognitivní průchod je metoda testování uživatelského rozhraní, při které je simulován uživatel z cílové skupiny. Cílem této metody je zjistit, zda je uživatel schopen zvládnout daný scénář, případně kde a jak se od optimálního (předpokládaného) průchodu odchyluje. V každém kroku si pozorovatel (testující) klade tyto otázky:

- Q0: Čeho chce uživatel dosáhnout?



Obrázek 3.4: Nastavení účtu, detail zóny, nastavení zóny.



Obrázek 3.5: Rozvrh vytápění - měsíc, týden, den.

- Q1: Bude uživateli zřejmé co udělat?
- Q2: Spojí si uživatel správně popisek akce se svým cílem?
- Q3: Dostane uživatel smysluplnou odezvu?

Přehled testovaných funkcí.

1. Možnost změnit jméno a telefonní číslo asociované s účtem. Jedná se o poměrně osobní aplikaci, proto by měla poskytovat základní možnost nastavení osobních údajů. Očekává se bezproblémová funkčnost.

Obrázek 3.6: Graf historie vytápění zóny, výběr data.

2. Zvýšení/snížení teploty v libovolné místnosti. Jde o aplikaci k ovládání chytrého zónového termostatu, tudíž jde o jednu z hlavních funkcí. Její funkčnost by měla být bezproblémová a na první pohled zřejmá.

3. Změnit režim vytápění v libovolné místnosti. Aplikace by měla mít možnost nastavit režim vytápění (např. komfort, útlum, ...) a také vytvářet vlastní režimy.

4. Prohlížení historie naměřených hodnot. Uživatel by měl mít možnost nahlížet do historie, aby mohl porovnávat naměřené hodnoty s nastavenými a podle toho mohl přizpůsobit nastavení vytápěcích plánů.

Test.

1. Možnost změnit jméno a telefonní číslo asociované s účtem.

   - Krok 1: V menu kliknout na "Account Settings". Tabulka 3.1

Otázka	Odpověď	Popis	Doporučení
Q1	ANO		
Q2	ANO		
Q3	ANO	Otevře se obrazovka s poli pro změnu údajů.	

Tabulka 3.1: Funkce 1, krok 1.

   - Krok 2: Upravit údaje. Tabulka 3.2.
   - Krok 3: Uložit údaje. Tabulka 3.3.

Otázka	Odpověď	Popis	Doporučení
Q1	ANO		
Q2	ANO		
Q3	ANO	Pokud nevyplní povinné údaje, bude upozorněn.	

Tabulka 3.2: Funkce 1, krok 2.

Otázka	Odpověď	Popis	Doporučení
Q1	ANO		
Q2	ANO		
Q3	ANO	Údaje jsou uloženy a uživatel vrácen na, hlavní stránku.	

Tabulka 3.3: Funkce 1, krok 3.

2. Zvýšení/snížení teploty v libovolné místnosti.
   - Krok 1: Dostat se na detail místnosti. Tabulka 3.4

Otázka	Odpověď	Popis	Doporučení
Q1	ANO		
Q2	ANO		
Q3	ANO	Otevře se obrazovka s detailem místnosti.	

Tabulka 3.4: Funkce 2, krok 1.

   - Krok 2: Upravit teplotu. Tabulka 3.5.

Otázka	Odpověď	Popis	Doporučení
Q1	ANO		
Q2	ANO		
Q3	ANO	Změní se teplota.	

Tabulka 3.5: Funkce 2, krok 2.

3. Změnit režim vytápění v libovolné místnosti.
   - Krok 1: Dostat se na detail místnosti. Tabulka 3.6
   - Krok 2: Vybrat požadovaný režim. Tabulka 3.7.
4. Prohlížení historie naměřených hodnot.
   - Krok 1: Dostat se na detail místnosti. Tabulka 3.8
   - Krok 2: Dostat se do historie naměřených hodnot. Tabulka 3.9.

Otázka	Odpověď	Popis	Doporučení
Q1	ANO		
Q2	ANO		
Q3	ANO	Otevře se obrazovka s detailem místnosti.	

Tabulka 3.6: Funkce 3, krok 1.

Otázka	Odpověď	Popis	Doporučení
Q1	ANO		
Q2	ANO		
Q3	ANO	Hodnota programu bude nastavena na zvolený.	

Tabulka 3.7: Funkce 3, krok 2.

Otázka	Odpověď	Popis	Doporučení
Q1	ANO		
Q2	ANO		
Q3	ANO	Otevře se obrazovka s detailem místnosti.	

Tabulka 3.8: Funkce 4, krok 1.

Otázka	Odpověď	Popis	Doporučení
Q1	ANO		
Q2	ANO		
Q3	ANO	Otevře se obrazovka historie.	

Tabulka 3.9: Funkce 4, krok 2.

- Krok 3: Prohlížení/vybírání konkrétního časového úseku z historie. Tabulka 3.10.

Otázka	Odpověď	Popis	Doporučení
Q1	ANO		
Q2	ANO		
Q3	ANO	Uživatel na obrazovce vidí požadované informace.	

Tabulka 3.10: Funkce 4, krok 3.

Shrnutí testování kognitivním průchodem. Test kognitivním průchodem uživatelského rozhraní aplikace pro ovládání termostatu byl proveden autorem návrhu. Lze tedy předpokládat, že výsledky budou poněkud zkreslené.

Testování bez uživatelů - heuristická evaluace

Heuristická evaluace je metoda založená na sadě pravidel (heuristik), která jsou předem definována. Aplikace je poté otestována experty, kteří jsou s problematikou aplikace obeznámeni, zda tato pravidla dodržuje nebo zda se dle nich chová. Pro testování heuristickou evaluací byly zvoleny heuristiky od Jaacoba Nielsona[3]. Návrh uživatelského rozhraní aplikace byl otestován dvěma experty. Expert č. 1 – vývojář mobilních aplikací, pohybuje se v oboru několik let. Expert č. 2 – zabývá se přímo testováním uživatelských rozhraní, má zkušenosti i v oblasti návrhu UI a vývoje aplikací pro termostaty.

Sada použitých heuristik:

1. Viditelnost stavu systému (Visibility of system status).
2. Propojení systému a reálného světa (Match between system and the real world).
3. Uživatelská svoboda a kontrola (User control and freedom).
4. Standardizace a konzistence (Consistency and standards).
5. Prevence chyb (Error prevention).
6. Rozpoznání namísto vzpomínání (Recognition rather than recall).
7. Flexibilní a efektivní použití (Flexibility and efficiency of use).
8. Estetický a minimalistický (Aesthetic and minimalist design).
9. Pomoc uživatelů pochopit, poznat a vzpamatovat se z chyb (Help users recognize, diagnose, and recover from errors).
10. Nápověda a návody (Help and documentation).

U nálezů jsou vždy uvedeny priority v číselné podobě 0-4, kde hodnoty vyjadřují následující:

[3] http://www.useit.com/papers/heuristic/heuristic$_l$*ist.html*

- 0 – Není problémem použitelnosti.
- 1 – Kosmetický problém.
- 2 – Malý problém použitelnosti.
- 3 – Problém použitelnosti, důležité odstranit.
- 4 – Závažný problém, musí být odstraněn.

Přehled nálezů – Expert 1.

1.
   - Popis problému: Obrazovka „detail místnosti“ je velmi nepřehledná, u většiny panelů není jasné k čemu, který slouží; příliš mnoho panelů, nejsou jasně vidět důležité informace.
   - Priorita: 4.
   - Narušuje heuristiky: všechny (výrazně: 1, 3, 6, 7).
   - Doporučení: Kompletně předělat tuto obrazovku. Zvýraznit důležité informace – hlavně teplotu. Pole „Settings“ přesunout do kontextového popup menu. Pole „Graphs“ nahradit přímo grafem (po kliknutí se zvětší na celou stránku).
2.
   - Popis problému: Není jasné, jak fungují možnosti „Schedule“ a „Program“. Po rozkliknutí pole „Schedule“ není jasné, jak by mělo fungovat následné upravování plánu.
   - Priorita: 4.
   - Narušuje heuristiky: 1, 4, 6, 7, 9 a 10.
   - Doporučení: Změnit časové „rozlišení“, poskytnout uživateli „menší volnost“. Důkladně promyslet funkčnost těchto vlastností.

Přehled nálezů – Expert 2.

1.
   - Popis problému: Obrazovka „detail místnosti“ je nepřehledná. Moc možností – dvojí nastavování („Heat Setpoint“, „Program“). Chybí možnost přepínat mezi jednotkami stupňů (F a C).
   - Priorita: 4.
   - Narušuje heuristiky: všechny (výrazně: 1, 3, 6, 7).
   - Doporučení: Předělat tuto obrazovku. Zvýraznit důležité informace – hlavně teplotu. Omezit možnosti nastavování. Přidat přepínání mezi jednotkami stupňů.

2. Popis problému: Mnoho překlepů.
   - Priorita: 0.
   - Narušuje heuristiky: 2, 4 a 8.
   - Doporučení: Opravit překlepy, poté nechat ještě někoho zkontrolovat.
3. Popis problému: Hlavní obrazovka by neměla mít titulek „Thermostats“, nepoužívá se v tomto typů aplikací.
   - Priorita: 1.
   - Narušuje heuristiky: 2, 4 a 8.
   - Doporučení: Opravit na „Zones“, jde o zónový termostat.
4. Popis problému: Celá aplikace – mnoho obrazovek – nepřehledné.
   - Priorita: 3.
   - Narušuje heuristiky: 1, 6, 7 a 8.
   - Doporučení: Aplikace tohoto typu by měli mít 3 hlavní obrazovky, aby se udržela přehlednost a uživatel se neztratil.
5. Popis problému: Obrazovka „Schedule“, nepřehledná. Očekávání, že se s grafem dá hýbat a tím nastavovat hodnoty.
   - Priorita: 3.
   - Narušuje heuristiky: 1, 4, 6, 7, 9 a 10.
   - Doporučení: Řádně promyslet, jaké jsou funkční požadavky.

Shrnutí heuristické evaluace. Po přezkoumání nálezů získaných od expertů je jasné, že nalezené problémy je nutné odstranit. Předpoklad, že výsledky testování kognitivním průchodem byly zkreslené, se tedy potvrdil.

Testování s uživateli

Tato část dokumentu obsahuje rozbor testování návrhu uživatelského rozhraní mobilní aplikace pro ovládání chytrého termostatu zónovou regulací vytápění. Test s uživatelem by měl odhalit, jestli je návrh uživatelského rozhraní aplikace správný, co se týká přehlednosti, intuitivnosti a grafického zpracování. Na rozdíl od testu bez uživatele budou v tomto případě výsledkem data ověřená a potvrzená reálnými, potenciálními uživateli.

Výběr účastníků. V tomto stádiu vývoje je nutné otestovat přehlednost a intuitivnost uživatelského rozhraní. Pro účely testu nebyl zapotřebí screener (dotazník pro výběr vhodných účastníků testů) a jediným kritériem při výběru účastníků bylo, zda používají chytrý telefon s dotykovým displejem.

Pre-test dotazník. Účastníci byli požádáni o vyplnění krátkého dotazníku před testem, aby nasbírané informace měli nějakou relevantní hodnotu. Za tímto účelem jim byly položeny následující čtyři otázky.

1. Jaký operační systém (mobilní) používáte?
   a. Android
   b. iOS
   c. Jiný - jaký?
2. Používal/a jste někdy chytrý termostat, popřípadě aplikaci na jeho ovládání?
   a. Ano - jakou?
   b. Ne
3. Jak hodnotíte své zkušenosti s technikou?
   a. Běžný uživatel
   b. Pokročilý uživatel
   c. Zkušený uživatel (vývojář, technik, ...)
4. Souhlasíte se zaznamenáváním průběhu testu a následným využitím těchto informací při vypracování bakalářské práce (ČVUT – FEL - Katedra počítačové grafiky a interakce)?
   a. Ano
   b. Ne

Post-test dotazník. Pro zkvalitnění zpětné vazby byly po konci testu účastníkům položeny dvě doplňující otázky.

1. Co byste vylepšili/změnili, co se vám nelíbilo?
2. Zaujalo vás na aplikaci něco pozitivního?

Test. Samotný test probíhal na telefonu s OS Android, na kterém byl nahrán návrh uživatelského rozhraní aplikace ve formě mockupu v PDF verzi (viz ukázkový průchod). Průběh testu byl zaznamenáván na papír (příhodnější a rychlejší, než jiný typ záznamu - audio, video). Účastníkům bylo předloženo šest potenciálních scénářů a bylo sledováno, jak jimi projdou.

1. Přihlaste se do aplikace pomocí emailového účtu. Poté si změňte osobní údaje, jméno a telefonní číslo, svého účtu.
2. (přihlášený uživatel) Jste v místnosti "Bedroom“ a je vám zima, chcete proto v místnosti zvýšit teplotu o 2 C.
3. (přihlášený uživatel) Chystáte se na víkend pryč, změňte režim vytápění v libovolné místnosti ze stávajícího na „Away“.
4. (přihlášený uživatel) Už jste týden mimo domov a zajímá vás, jaké hodnoty zaznamenal termostat včera v poledne, tedy 27. 12. 2017 ve 12.00.
5. (přihlášený uživatel) Zajímají vás informace o nakládání s vašimi osobními daty a jaké jsou podmínky používání této služby.
6. (přihlášený uživatel) Máte problém s termostatem a aplikací, chcete proto kontaktovat výrobce termostatu, zjistěte tedy kontaktní informace o výrobci a o verzi aplikace.

Ukázkový průchod.

1. Scénář č.1. Obrázky: přihlášení pře email 3.2, nastavení účtu 3.4.
2. Scénář č. 2 a č. 3. Shodné obrázky pro oba scénáře: detail zóny 3.4.
3. Scénář č. 4. Obrázky: graf historie 3.6.
4. Scénář č. 5 a č. 6. Obrázky: soukromí a právní informace, o aplikaci 3.6.

Stručný záznam z testů.

- Účastník č. 1
  - Pre-test dotazník:
    1. Android

2. Ne
3. Běžný uživatel
4. Ano

- Průchod scénáři:
  1. Bez problémů.
  2. Dokončí scénář úspěšně, ale poté znejistí, zda ho provedl správně. Nejistota mezi poli „Temperature“ a „Heat Setpoint“.
  3. Místo na pole „Program“ kliká na „Schedule“.
  4. Kliká na pole „Schedule“ – pro kliká novou obrazovku. Vrací se zpět a kliká na pole „Graphs“. Vybere správné datum a končí.
  5. Bez problémů.
  6. Kliká na ikonu „i“, poté přes slide out menu na „About“.
- Post-test dotazník:
  1. Nerozumí rozdílu mezi „Graphs“ a „Schedule“ – spojil by do jednoho „kalendáře“. Nelíbí se způsob zvyšování teploty přes pole „Heat Setpoint“, chtěl by to mít přes kliknutí na pole „Temperature“.
  2. Nic nevyniklo.

- Účastník č. 2
  - Pre-test dotazník:
    1. iOS, dříve Android
    2. Ne
    3. Zkušený uživatel
    4. Ano
  - Průchod scénáři:
    1. Bez problémů.
    2. Nemůže se dostat do detailu místnosti, kliká na celé pole. Po chvíli na to přijde. Zbytek bez problémů.
    3. Kliká správně na pole „Program“, ale váhal.
    4. Bez problémů.
    5. Bez problémů.
    6. Bez problémů.
  - Post-test dotazník:
    1. Nekonzistence aplikace, nefungují základní principy ovládání. Nelíbí se pole „Schedule“ a co se pod ním skrývá.
    2. Nic nevyniklo.
- Účastník č. 3
  - Pre-test dotazník:
    1. Android

2. Ne
3. Pokročilý uživatel
4. Ano

- Průchod scénáři:
  1. Bez problémů.
  2. Bez problémů.
  3. Nerozhodný mezi polem „Schedule“ a „Program“. Vybírá si „Schedule“.
  4. Bez problémů.
  5. Bez problémů.
  6. Bez problémů.
- Post-test dotazník:
  1. Překlepy/chyby v angličtině. Zmatečná obrazovka detail místnosti. Složitá a nejasná funkce „Schedule“.
  2. Nic nevyniklo.

Problémy vzniklé při testování. Žádné větší problémy nenastaly. Občas, když se kliklo mimo tlačítko, tak se účastník dostal na nežádoucí stránku – důsledek prototypu ve formátu PDF.

Přehled nálezů. Popis priorit nálezů:

- **Priorita:** nízká. **Popis:** jde hlavně o menší problémy, které uživateli vadí, ale i přes ně dokáže systém správně používat bez větší frustrace.
- **Priorita:** střední. **Popis:** problémy které nejsou kritické, nicméně ovlivňují ovládání/chápání systému uživatelem. Uživatel u nich přichází o čas a začíná se objevovat frustrace. Tím pádem při větším počtu těchto problémů se uživatel může začít porozhlížet po konkurenčních systémech.
- **Priorita:** vysoká. **Popis:** tyto problémy jsou kritické, zásadně zasahují do schopnosti uživatele používat systém a měli by být co nejdříve odstraněny.

1. Nález - „Temperature“ x „Heat Setpoint“
   - Priorita: Vysoká
   - Popis: Od všech účastníků byla získána zpětná vazba, že není úplně jasný rozdíl mezi „Temperature“ a „Heat Setpoint“.

- Doporučení: Vizuálně rozlišit, popřípadě sloučit.

2. Nález - obrzovka detailu místnosti
   - Priorita: Vysoká
   - Popis: Od všech účastníků byla získána zpětná vazba, že obrazovka detailu je nepřehledná a jednotlivá pole jsou matoucí a některá přebytečná.
   - Doporučení: Předělat návrh detailu místnosti, výrazně zjednodušit.
3. Nález - „Schedule“ x „Program“
   - Priorita: Střední
   - Popis: Není jasný rozdíl mezi funkcí „Program“ a „Schedule“.
   - Doporučení: Přejmenovat, vymyslet tak, aby nebylo matoucí a funkce byly jasné.

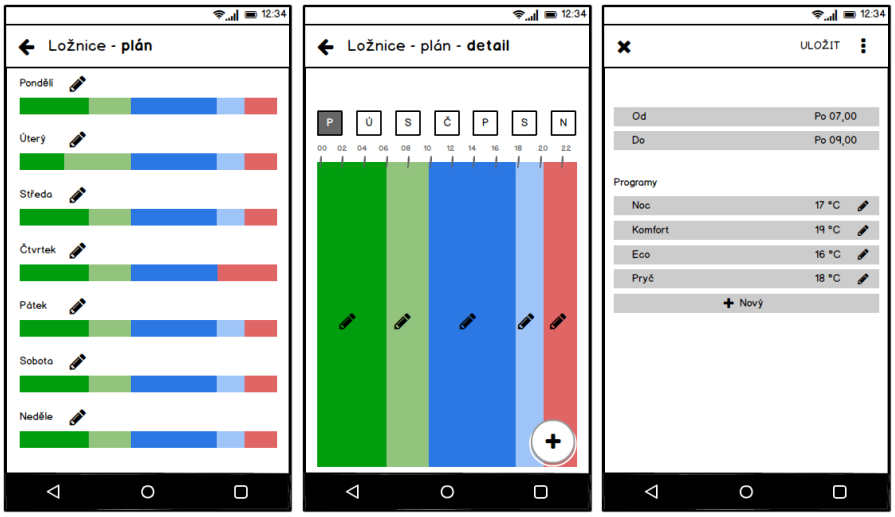
Shrnutí výsledků testů s uživateli. Z vetší části se potvrdily výsledky, které byly získány při heuristické evaluaci, tj. první prototyp uživatelského rozhraní je nutné zjednodušit a odstranit nejasnosti v uživatelském rozhraní.

3.3 Druhé kolo návrhu uživatelského rozhraní

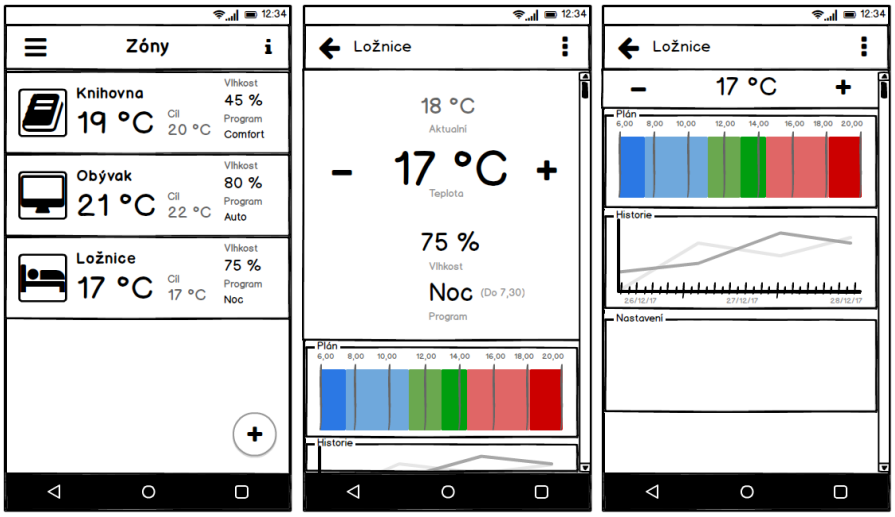
Na základě získané zpětné vazby na první verzi prototypu a zpracování výstupů testovaní byla vytvořena druhá verze návrhu uživatelského rozhraní, která odstraňuje předchozí nedostatky. A která je základem pro realizaci aplikace.

3.3.1 Změny v návrhu

Na obrázku 3.7 je nová verze obrazovky s rozvrhů vytápění. Zjednodušené UI umožňuje lepší orientaci a je intuitivnější. Obrázek 3.8 - detail zóny - představuje novou verzi obrazovky, jsou odebrané nejasné popisky, celkové zjednodušení uživatelského rozhraní, pozornost směřována hlavně na změnu požadované teploty.



Obrázek 3.7: Rozvrh vytápění, detail dne, nastavení změny programu.



Obrázek 3.8: Hlavní obrazovka, detail zóny, detail zóny po posunu.

Kapitola 4

Implementace

Tato kapitola se věnuje implementační části práce. Popisuje využité nástroje a technologie potřebné k implementaci aplikace v Ionic frameworku.

4.1 Použité nástroje

4.1.1 Ionic(v2)

Ionic je SDK pomocí něhož lze vyvíjet mobilní hybridní aplikace. Nejdůležitější součástí Ionic SDK je Ionic framework. Obsahuje ale další zajímavé nástroje - CLI (Command Line Interface), prototypovací nástroj Ionic Creator. Usnadněním pro vývojáře je absence nutnosti instalovat SDK jednotlivých platforem, vyvíjenou aplikaci lze testovat přímo ve webovém prohlížeči.

Ionic framework je postaven na dvou frameworcích AngularJS[1] a Apache Cordova[2]. Díky Cordově může Ionic nabídnout přístup k velké části dostupných služeb mobilního telefonu. Fungování služeb se pak neliší od použití v nativních aplikacích. Ionic má i nemalou komunitu mezi vývojáři a dobře zpracovanou dokumentaci.

[1] https://angular.io/
[2] https://cordova.apache.org/

Ionic také nabízí podporu všech hlavních platforem Android 4.4 a výš a iOS 7 a výš, jako bonus můžeme pro naši aplikaci brát podporu Windows Phone.

4.1.2 AngularJS a Apache Cordova

AngularJS 2

Jak už jména naznačuje jedná se webový javascriptový MVC (Model View Controller) framework. Angular aplikace jsou tvořeny z komponent. Komponenta je kombinace HTML šablony její třídy, která ovládá část obrazovky. Mezi jeho přednosti patří "Two Way Data-Binding"[3](obrázek 4.1), direktivy a znovupoužitelnost komponent.

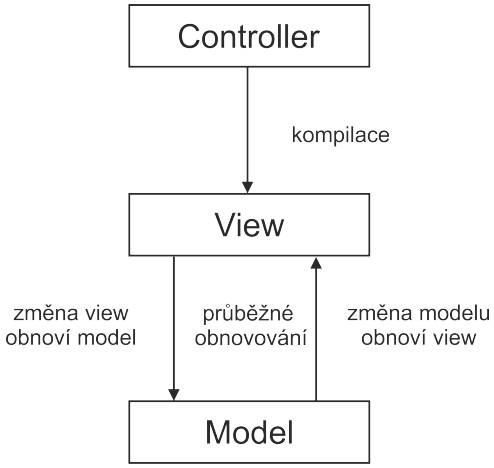
Angular pracuje s pojmem direktivy:

1. **Komponenty** - direktivy se šablonou.
2. **Strukturální direktivy** - manipulují s DOM (Document Object Model). (např. ngFor, ngIf)
3. **Direktivy atributů** - mění vzhled nebo chování jiných elementů, komponentů nebo direktiv.(např. ngStyle)

Apache Cordova

Apache Cordova je open-source framework pro vývoj hybridních mobilních aplikací. Dovoluje použití standardních webových technologií(HTML5, CSS3, JS). Aplikace se spouští ve wrapperu[4] dané platformy a tím umožňuje přístup k nativním službám zařízení (senzory, data, atd.). Tento přístup, aplikace k nativnímu API umožníme instalací pluginů do aplikace. Cordova má v základu sadu pluginů, které nabízí podporu základních služeb(kontakty, kamera, atd.). Navíc lze využít i pluginů třetích stran, které škálu podporovaných služeb jednotlivých zařízení zvětšují.

[3]Volně přeloženo jako "dvoucestná synchronizace dat"
[4]"Most"mezi dvěma různými rozhraními.



Obrázek 4.1: "Two Way Data-Binding"v AngularJS.

4.1.3 Spring Boot

Spring Boot[5] projekt je postaven na Sprig[6] frameworku a knihovnách třetích stran. Umožňuje vytvoření automaticky konfigurované spring aplikace, která následně běží např. na Tomcat[7] serveru. Také odpadá potřeba konfigurovat XML (eXtensible Markup Language).

Pro tuto práci byl zvolen, pro svoji přiměřenou jednoduchost, aby sloužil jako server poskytující aplikaci data(protože skutečná data nejsou k dispozici). Na serveru je implementováno REST[8] API (REpresentational State Transfer), které nabízí aplikaci přístup k datům na serveru. Komunikace mezi aplikací a serverem je zprostředkována HTTP protokolem.

4.1.4 Další využité technologie

REST API

REST API je pro rozhraní orientované na data umožňující provádět nad nimi CRUD (Create, Read, Update, Delete) operace. Je bezstavové a umožňuje

[5] http://projects.spring.io/spring-boot/
[6] https://spring.io/
[7] Java HTTP web server
[8] https://tools.ietf.org/html/rfc6690

paralelní zpracování obsahu. Autorizace je řešena pomocí OAuth 2.0[9]. Obecně se rozděluje do 4 úrovní:

1. **Nultá úroveň** - přenos dat a pro něj zvolený protokol, dnes se nejvíce využívá HTTP.
2. **První úroveň** - zdroje. Každý zdroj má právě jeden koncový bod, na kterém je dostupný (např. GET /users vrátí objekt všech uživatelů, GET /users/1/zones vrátí všechny zóny uživatele s id 1).
3. **Druhá úroveň** - HTTP metody. Vyjadřují akci která se má vykonat(GET, POST, PUT, DELETE, PATCH).
4. **Třetí úroveň** - HATEOAS[10]. Jde o to, že by REST API mělo být řízeno odkazy.

JSON

JSON[11] (JavaScript Object Notation) je formát pro výměnu dat nezávislí na počítačové platformě. Je založen na dvou strukturách kolekce párů název/hodnota nebo seřazený seznam hodnot. Jde o univerzální datové struktury, které moderní programovací jazyky v nějaké formě podporují. V této práci budeme pro přenos dat využívat formát JSON, protože umožňuje snadné zpracování Javou i TS.

OAuth 2.0

OAuth je autorizační framework, který se často používá pro zabezpečení RESTových webových služeb. Výhoda kterou poskytuje je, že uživatel může poskytnout klientské aplikaci přístup k jeho datům v nějaké službě, aniž by té aplikaci musel vyzradit své přístupové údaje do služby. Také umožňuje podrobně vymezit pravomoci jednotlivých klientských aplikací a sledovat využívání poskytnutých privilegií.

[9] https://tools.ietf.org/html/rfc6749
[10] Hypertext As The Engine Of Application State
[11] https://tools.ietf.org/html/rfc4627

4.2 Struktura aplikace

V Ionic projektu jsou s každou stránkou přímo spojeny tři soubory:
`pageName.ts` - pro definici funkcí stránky,
`pageName.html` - struktura elementů obsažených na stránce a
`pageName.scss` (SCSS - Sassy CSS) - pro definici stylů elementů. Celou strukturu aplikace lze nalézt v příloze A.

4.2.1 Přihlášení

Výběr přihlášení

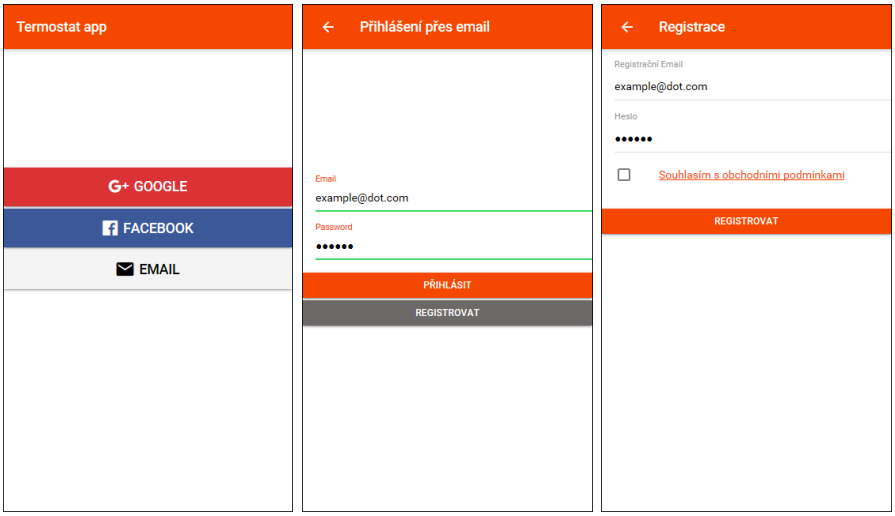
Úvodní stránka (obrázek 4.2) dává na výběr mezi přihlášením do aplikace prostřednictvím jedné ze tří možností:

1. Facebook účet.
2. Google Plus účet.
3. Emailová adresa.

Aplikace má implementované přihlášení přes email, implementace zbylých možností přihlášení je naplánována až s vývojem prototypu termostatu.

Přihlášení přes email

Po vyplnění validních přihlašovacích údajů (obrázek 4.2) se lze přihlásit do aplikace. Autorizace je implementována za pomoci OAuth 2.0 frameworku, který je v aplikaci implementován v provideru. Implementace možnosti registrace do aplikace (obrázek 4.2) je naplánována až s vývojem prototypu termostatu a definování kroků vedoucích k registraci do aplikace.



Obrázek 4.2: Výběr přihlášení do aplikace, přihlášení přes email, registrace.

4.2.2 Přehled zón

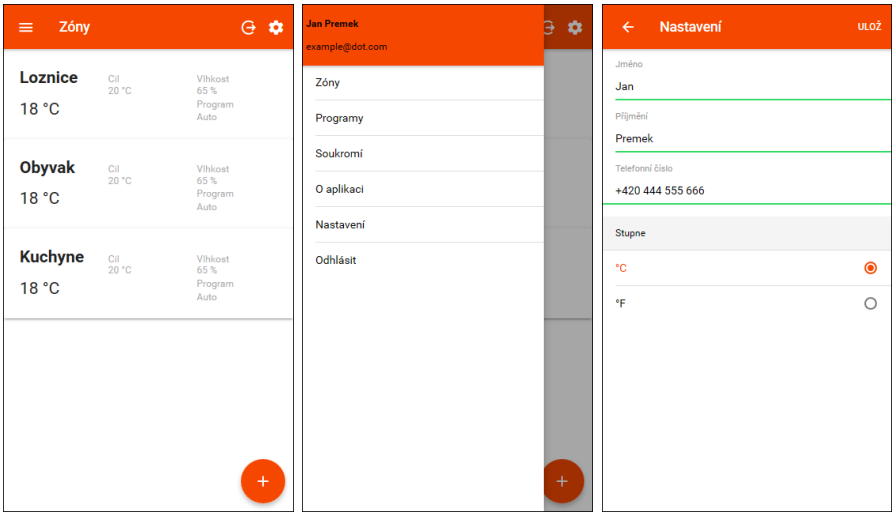
Na této stránce (obrázek 4.3), se nachází přehled zón asociovaných s přihlášeným účtem. Z ní se přímo můžeme dostat do detailu jednotlivých zón, nastavení aplikace a přihlášeného účtu (obrázek 4.3), přidání nové zóny (obrázek 4.4) k účtu a také do navigačního menu (obrázek 4.3).

Nastavení aplikace a přihlášeného účtu

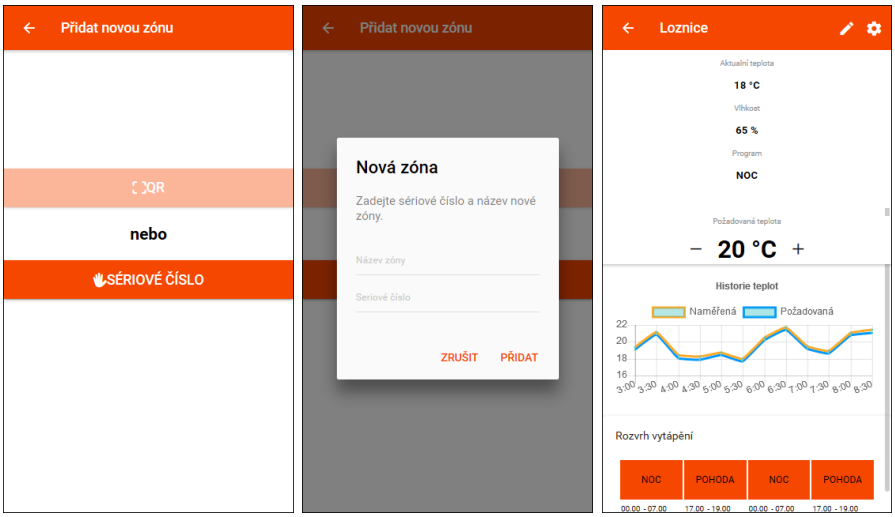
Na této stránce máme možnost změnit jednotky stupňů v celé aplikaci (Fahrenheit/Celsius). Dále jméno, příjmení a telefonní číslo přihlášeného uživatele.

Přidání nové zóny

Přidání nové zóny je možné dvěma způsoby. Buď pomocí sériového čísla nebo pomocí naskenování QR (Quick Response) kódu. Implementována je část přidání zóny pomocí sériového čísla. Implementace přidání pomocí QR kódu bude je naplánována až s vývojem prototypu termostatu.



Obrázek 4.3: Přehled zón, navigační menu, nastavení.



Obrázek 4.4: Přidání nové zóny, detail vybrané zóny.

4.2.3 Detail zóny

Tato stránka (obrázek 4.4) obsahuje nejpodstatnější informace v aplikaci - nastavení vytápění v zóně, historii vytápění v zóně a změnu aktuálně požadované teploty v zóně. tato stránka prošla v procesu návrhu největšími změnami, aby uživatelsky přívětivá. Odtud se můžeme dostat na graf historie naměřených hodnot (obrázek 4.5), na změnu jména zóny (obrázek 4.5), na nastavení možností zóny (obrázek 4.5) a na rozvrh plánování vytápění zóny (obrázek 4.6).

Změna jména zóny

Jednoduchá stránka umožňující změnu jména zóny.

Nastavení zóny

Umožňuje spravovat účty přidružené k zóně, také zobrazuje informace o fyzickém zařízení (termostatu) v zóně.

Graf historie naměřených hodnot

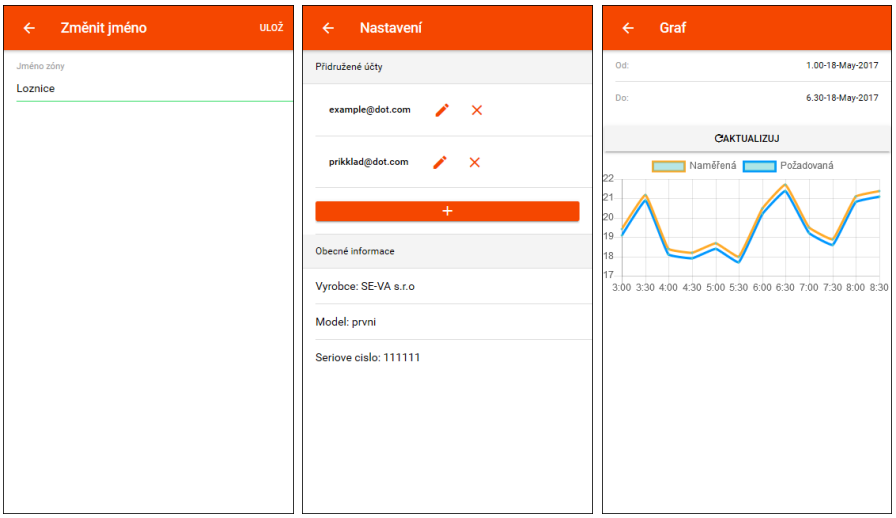
Tato stránka umožňuje zobrazit si požadovaný úsek historie vytápění zóny. Graf obsahuje dvě křivky, jednu pro skutečně naměřené hodnoty a druhou pro požadované hodnoty.

Rozvrh vytápění zóny

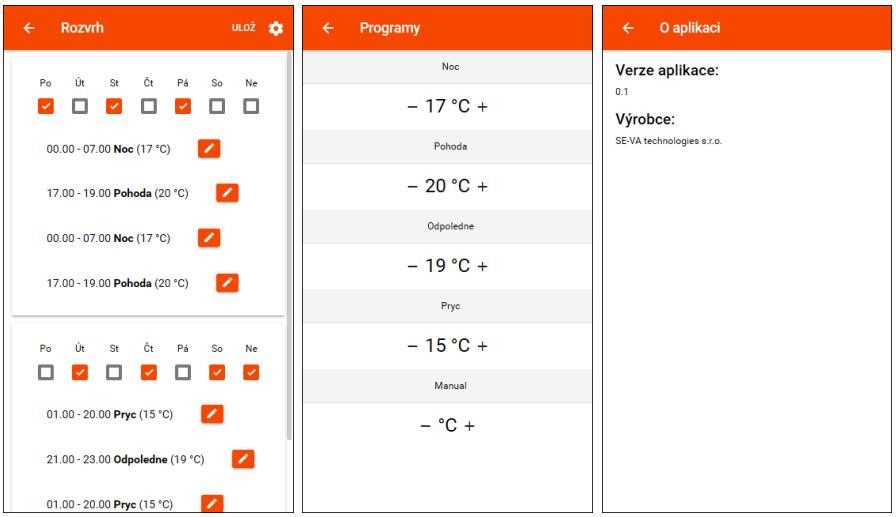
Stránka umožňující spravovat rozvrh vytápění zóny. Lze si předvolit dva typy rozvrhů pro vytápění zóny během týdne. Každý rozvrh obsahuje 4 možnosti změny programu. Z této stránky se také dostaneme na správu programů (obrázek 4.6).

4.2.4 Navigační menu

Navigační menu aplikace(obrázek 4.3) umožňující rychlejší orientaci v aplikaci.



Obrázek 4.5: Změna jména zóny, nastavení zóny, graf historie naměřených hodnot.



Obrázek 4.6: Rozvrhy vytápění, nastavení programů, informace o aplikaci.

Kapitola 5

Testování

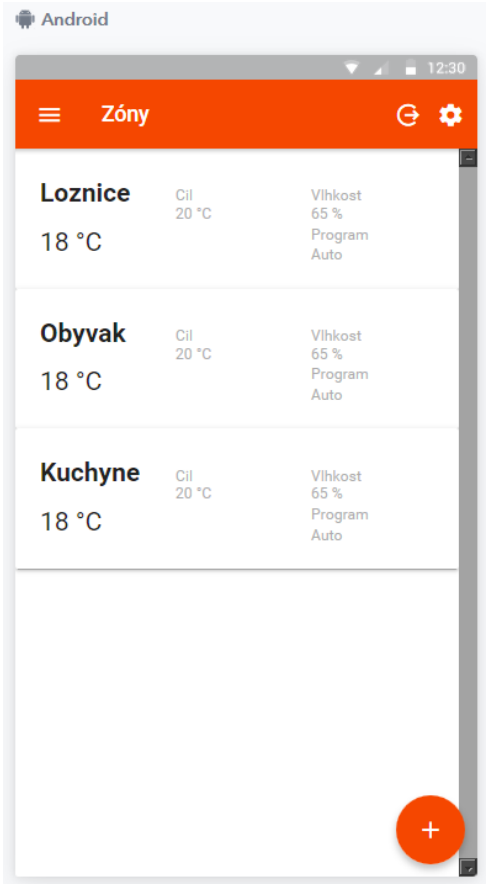
Tato kapitola se věnuje testování implementované aplikace se vzorovými konfiguracemi. Testované konfigurace jsou řešeny pomocí skutečných dat na backend-serveru. Aplikaci otestujeme testy použitelnosti.

5.1 Testování s uživateli

Při testování bude postupováno obdobně jako v případě testování PDF prototypu v sekci 3.2.2. Samotné testování bude probíhat ve webovém prohlížeči (obrázek 5.1).

5.1.1 Výběr účastníků

Cílem je opět otestovat přehlednost a intuitivnost uživatelského rozhraní prototypu aplikace. Žádná speciální kritéria proto pro výběr účastníků nejsou.



Obrázek 5.1: Testování aplikace v prohlížeči.

5.1.2 Pre-test dotazník

Účastníci byli opět požádáni o vyplnění krátkého dotazníku před testem.

1. Jaký operační systém (mobilní) používáte?
   a. Android
   b. iOS
   c. Jiný - jaký?
2. Používal/a jste někdy chytrý termostat, popřípadě aplikaci na jeho ovládání?
   a. Ano - jakou?
   b. Ne
3. Jak hodnotíte své zkušenosti s technikou?

a. Běžný uživatel
b. Pokročilý uživatel
c. Zkušený uživatel (vývojář, technik, ...)

4. Souhlasíte se zaznamenáváním průběhu testu a následným využitím těchto informací při vypracování bakalářské práce (ČVUT – FEL - Katedra počítačové grafiky a interakce)?

a. Ano
b. Ne

5.1.3 Post-test dotazník

Pro zkvalitnění zpětné vazby byly po konci testu účastníkům položeny dvě doplňující otázky.

1. Co byste vylepšili/změnili, co se vám nelíbilo?
2. Zaujalo vás na aplikaci něco pozitivního?

5.1.4 Testované scénáře

Scénáře jsou zjednodušenou verzí scénářů u testování PDF prototypu.

1. Přihlaste se do aplikace pomocí emailového účtu. Poté si změňte osobní údaje, jméno a telefonní číslo, svého účtu.
2. (přihlášený uživatel) Jste v místnosti "Loznice“ a je vám zima, chcete proto v místnosti zvýšit teplotu o 2 C.
3. (přihlášený uživatel) Už jste týden mimo domov a zajímá vás, jaké hodnoty zaznamenal termostat včera v poledne.
4. (přihlášený uživatel) Máte problém s termostatem a aplikací, chcete proto kontaktovat výrobce termostatu, zjistěte tedy kontaktní informace o výrobci a o verzi aplikace.

5.1.5 Stručný záznam z testů

- Účastník č. 1
  - Pre-test dotazník:
    1. Android
    2. Ne
    3. Pokročilý uživatel
    4. Ano
  - Průchod scénáři:
    1. Bez problémů.
    2. Bez problémů.
    3. Není jasné, že graf na detailu zóny lze rozkliknout. Po rozkliknutí bez problémů
    4. Bez problémů.
  - Post-test dotazník:
    1. Nic.
    2. Na obrazovce detailu termostatu zasouvací header.
- Účastník č. 2
  - Pre-test dotazník:
    1. iOS
    2. Ne
    3. Běžný uživatel
    4. Ano
  - Průchod scénáři:
    1. Bez problémů.
    2. Bez problémů.
    3. Začátek bez problému, pak problém s grafem - neví jak vyčíst naměřené hodnoty.
    4. Bez problémů.
  - Post-test dotazník:
    1. Špatně ovladatelný graf. Přidat možnost nastavení vzhledu aplikace (barvy).
    2. Nic nevyniklo.

5.1.6 Problémy vzniklé při testování

Jediný problém nastal při prvním scénáři prvního účastníka - aplikace spadla důsledkem chyby v HTML šabloně. Bylo rychle opraveno a zbytek testů bez problémů.

5.1.7 Přehled nálezů

Popis priorit nálezů:

- **Priorita:** nízká. **Popis:** jde hlavně o menší problémy, které uživateli vadí, ale i přes ně dokáže systém správně používat bez větší frustrace.
- **Priorita:** střední. **Popis:** problémy které nejsou kritické, nicméně ovlivňují ovládání/chápání systému uživatelem. Uživatel u nich přichází o čas a začíná se objevovat frustrace. Tím pádem při větším počtu těchto problémů se uživatel může začít porozhlížet po konkurenčních systémech.
- **Priorita:** vysoká. **Popis:** tyto problémy jsou kritické, zásadně zasahují do schopnosti uživatele používat systém a měli by být co nejdříve odstraněny.

1. Nález - Graf
   - Priorita: Střední.
   - Popis: Od jednoho účastníka stížnost na funkčnost grafu.
   - Doporučení: Nalézt jiné řešení.

5.1.8 Shrnutí výsledků testů s uživateli

Při testování se neobjevily žádné závažné nedostatky. Z toho lze soudit, že provedené změny v návrhu UI byly správné. V budoucnu je cíl zvýšit počet a komplexnost testovaných scénářů. Také začít vybírat účastníky testů na základě screeneru.

Kapitola 6

Závěr

V průběhu řešení zadání této práce byly provedeny rešerše nástrojů pro vývoj multiplatformních mobilních aplikací, návrh a otestování více verzí uživatelského rozhraní a v neposlední řadě i samotná implementace mobilní aplikace s použitím zvoleného frameworku Ionic.

Rešerše nástrojů pro vývoj mobilních multiplatformních aplikací poskytla mnoho nových informací. Výběr vhodného frameworku pro implementaci výsledné aplikace byl proveden na základe porovnání jednotlivě hodnocených vlastností a jeho vhodnosti pro realizovanou mobilní aplikaci. Důležitou vlastností zvoleného frameworku je, že nabízí UI prvky s nativním vzhledem, přizpůsobující se platformám Android a iOS.

Rešerše již existujících aplikací ukázala, čeho se při návrhu a implementaci vyvarovat a dala potřebný náhled do problematiky mobilních aplikací pro ovládání chytrých termostatů. Testování s uživateli poskytlo zpětnou vazbu pro úvodní návrh uživatelského rozhraní a tím i podněty pro jeho přepracování do podoby druhého prototypu s odstraněnými nedostatky. Tento druhý návrh reflektoval více požadavky potenciálních uživatelů[3] a stal se podkladem pro vývoj prototypu aplikace.

Implementační část práce byla časově nejnáročnější. Nejprve byly vytvořeny jednotlivé stránky aplikace a následně implementována jejich logika. První verze prototypu aplikace existovala ve verzi statického webu, tzn. data v ní byla napevno naprogramována. Pro druhou verzi jsem již byl napsán jednoduchý backend-server, s kterým aplikace komunikuje přes REST API s daty ve formátu JSON. Aplikace je tedy schopna obousměrné komunikovat se

serverem a využívá framework OAuth 2.0 pro základní autorizaci uživatele.

Testování výsledného implementovaného prototypu přineslo pozitivní výsledek, ve formě nízkého množství stížností na funkčnost.

6.1 Budoucí vývoj

Na základě poznatků z této práce budou definovány požadavky na fyzické zařízení termostatu. Implementovaná aplikace, především pak poznatky z oblasti UX, poslouží jako základ pro produkční mobilní aplikaci.

Příloha A

Instalační instrukce a struktura aplikace

A.1 Instalační instrukce

Nutnou prerekvizitou pro spuštění projektu je nainstalovaný Ionic framework[1] a Apache Cordova. Před otevřením aplikace je dobré, ve složce projektu, spustit příkaz
`npm install`, který stáhne všechny potřebné moduly, které by mohly chybět. Pro otevření aplikace v prohlížeči, stačí ve složce projektu spustit příkaz `ionic serve`.

Jednoduchý backend-server pro testovací účely, běží na adrese `https://dev.se-va.cz:9443/thermostat/`. Příklad REST API endpointu: `https://dev.se-va.cz:9443/thermostat/rest/v1/users/1`.

A.2 Struktura souborů a složek v projektu

Následuje struktura složky
`./src`, v které jsou všechny důležité soubory.
| declarations.d.ts
| index.html

[1] http://ionicframework.com/getting-started/

```
|       manifest.json
|       service-worker.js
|
+--app
|       app.component.ts
|       app.html
|       app.module.ts
|       app.scss
|       Constants.ts
|       main.ts
|
+--assets
|       +--icon
|       favicon.ico
|
+--components
|       +--expandable-header
|       expandable-header.html
|       expandable-header.scss
|        expandable-header.ts
|
+--domain
|       OAuth2AccessToken.ts
|       SimpleZone.ts
|
+--pages
|       +--about
|       |       about.html
|       |       about.scss
|       |       about.ts
|       |
|       +--add-new-zone
|       |       add-new-zone.html
|       |       add-new-zone.scss
|       |       add-new-zone.ts
|       |
|       +--dashboard
|       |       dashboard.html
|       |       dashboard.scss
|       |       dashboard.ts
|       |
|       +--degrees-setting
|       |       degrees-setting.html
|       |       degrees-setting.scss
|       |       degrees-setting.ts
|       |
```

```
|       +—edit-programs
|       |       edit-programs.html
|       |       edit-programs.scss
|       |       edit-programs.ts
|       |
|       +—edit-schedule-setpoint
|       |       edit-schedule-setpoint.html
|       |       edit-schedule-setpoint.scss
|       |       edit-schedule-setpoint.ts
|       |
|       +—edit-zone-name
|       |       edit-zone-name.html
|       |       edit-zone-name.scss
|       |       edit-zone-name.ts
|       |
|       +—graph
|       |       graph.html
|       |       graph.scss
|       |       graph.ts
|       |
|       +—home
|       |       home.html
|       |       home.scss
|       |       home.ts
|       |
|       +—login-email
|       |       login-email.html
|       |       login-email.scss
|       |       login-email.ts
|       |
|       +—overview
|       |       overview.html
|       |       overview.scss
|       |       overview.ts
|       |
|       +—privacy
|       |       privacy.html
|       |       privacy.scss
|       |       privacy.ts
|       |
|       +—registeremail
|       |       registeremail.html
|       |       registeremail.scss
|       |       registeremail.ts
|       |
|       +—schedule
```

```
|       |       schedule.html
|       |       schedule.scss
|       |       schedule.ts
|       |
|       +--schedules
|       |       schedules.html
|       |       schedules.scss
|       |       schedules.ts
|       |
|       +--zonesetting
|               zonesetting.html
|               zonesetting.scss
|               zonesetting.ts
|
+--providers
|       |       storageUtils.ts
|       |
|       +--global-var
|       |       global-var.ts
|       |
|       +--user-data
|               user-data.ts
|
+--theme
variables.scss
```

Příloha B

Literatura

[1] LATIF, Mounaim, Younes LAKHRISSI, El Habib NFAOUI a Najia ESSBAI. Cross platform approach for mobile application development: A survey. In: 2016 International Conference on Information Technology for Organizations Development (IT4OD) [online]. IEEE, 2016, s. 1-5 [cit. 2017-05-22]. DOI: 10.1109/IT4OD.2016.7479278. ISBN 978-1-4673-7689-1. Dostupné z: http://ieeexplore.ieee.org/document/7479278/

[2] M. Willocx, J. Vossaert and V. Naessens, "Comparing Performance Parameters of Mobile App Development Strategies,"2016 IEEE/ACM International Conference on Mobile Software Engineering and Systems (MOBILESoft), Austin, TX, 2016, pp. 38-47 [cit. 2017-05-22]. DOI: 10.1109/MobileSoft.2016.028. ISBN: 978-1-4503-4178-3. Dostupné z: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=arnumber=7832966isnumber=7832858

[3] Aragon, C., Hurwitz, B., Peffer, T. and Pritoni, M., 2010. How People acutally use Thermostats. In ACEEE Summer Study on Energy Efficiency in Buildings, pp. 193-206 [cit. 2017-05-22]. Dostupné z: http://people.ischool.berkeley.edu/ dhawal/files/aceee.pdf

[4] Jonathan Peppers:Xamarin Cross-platform Application Development, Packt Publishing, Ebook ISBN: 978-1-84969-847-4 | ISBN 10: 1-84969-847-3

[5] Operating System Market Share. Market Share Statistics for Internet Technologies [online]. [cit. 2017-05-22]. Dostupné z: https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=8qpcustomd=1qpstick=1

[6] Top 10 Cross-Platform Mobile Development Tools. Hongkiat [online]. [cit. 2017-05-22]. Dostupné z: http://www.hongkiat.com/blog/cross-mobile-platform-framework-wora/