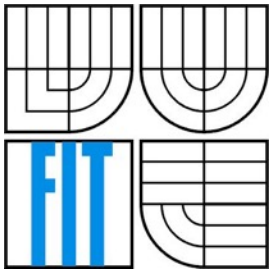
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

Fakulta informačních technologií
Faculty of Information Technology

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

Brno, 2016 Vladimír Alfery

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

ZOBRAZENÍ LETOVÝCH INFORMACÍ CESTUJÍCÍM

VISUALIZATION OF FLIGHT DATA TO PASSENGERS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — VLADIMÍR ALFERY
AUTHOR

VEDOUCÍ PRÁCE — Ing. PETER CHUDÝ, Ph.D. MBA
SUPERVISOR

BRNO 2016

Abstrakt

Cílem práce je umožnit pasažérům zobrazit aktuální letové informace, které vidí piloti v pilotní kabině. Tyto informace jsou zobrazitelné na elektronickém zařízení, připojeném k interní bezdrátové síti v letadle. Pro zaručení kompatibility se jedná o webovou aplikaci využívající CesiumJS knihovnu pro zobrazení virtuálního pohledu z kokpitu, která podporuje operační systémy Windows, OS X, iOS a Android.

Abstract

The goal of the thesis is to enable passengers to see the actual flight information, which the pilot sees from the cockpit. This information is viewable on electronic device connected to the aircraft internal wireless network. Web application which uses CesiumJS library to view terrain is used in order to ensure compatibility with the most nowadays devices with operating system Windows, OS X, iOS and Android.

Klíčová slova

Kokpit, letectví, zobrazení, CesiumJS, HTML, Python, Javascript, GoogleMaps, virtuální realita, design, grafika.

Keywords

Cockpit, airways, visualization, CesiumJS, HTML, Python, JavaScript, GoogleMaps, virtual reality, design, graphic.

Citace

Vladimír Alfery : Zobrazení letových informací pasažérům, bakalářská práce, Brno, FIT VUT v Brně, 2016

Zobrazení letových informací pasažérům

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Petera Chudého, Ph.D. MBA. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Vladimír Alfery
14.5.2016

Poděkování

Děkuji všem, kteří mi jakoukoliv formou pomohli v práci, připomínkami, nápady či v morálce. Jako prvnímu bych chtěl velmi poděkovat vedoucímu mé bakalářské práce Ing. Peteru Chudému, Ph.D. MBA, který mi poskytoval jak odborné informace, tak mi také pomáhal ve výběru správné cesty k úspěšnému dokončení práce. Nesmím zapomenout ani na podporu rodičů, rodiny a kolegů.

Obsah

Seznam zkratek

LCD – Liquid Crystal Display

PFD – Primary Flight Display

MFD – Multi-function display

API - Application Programming Interface

HTML – HyperText Markup Language

XML - Extensible Markup Language

FTP - File Transfer Protocol

HTTP - Hypertext Transfer Protocol

POP3 - Post Office Protocol

SMTP - Simple Mail Transfer Protocol

IMAP - Internet Message Access Protocol

SSH - Secure Shell

AIS – Indicated Air Speed

GS – Ground Speed

GPS - Global Positioning System

1 Úvod

Vývoj zobrazení letových informací na displejích v pilotním prostoru v posledních letech už nezaznamenává tak razantní proměny, jak tomu bylo v posledních desetiletích minulého století.

V dnešní době je již tolik platných nařízení a předpisů standardizujících zobrazení, že již není možno experimentovat s rozdílným rozložením ovládacích panelů. Existují však výjimky, snažící se zavést moderní zobrazení do avioniky a dokazují, že i moderní zobrazení může zaručit kvalitní, přehledný a srozumitelný systém.

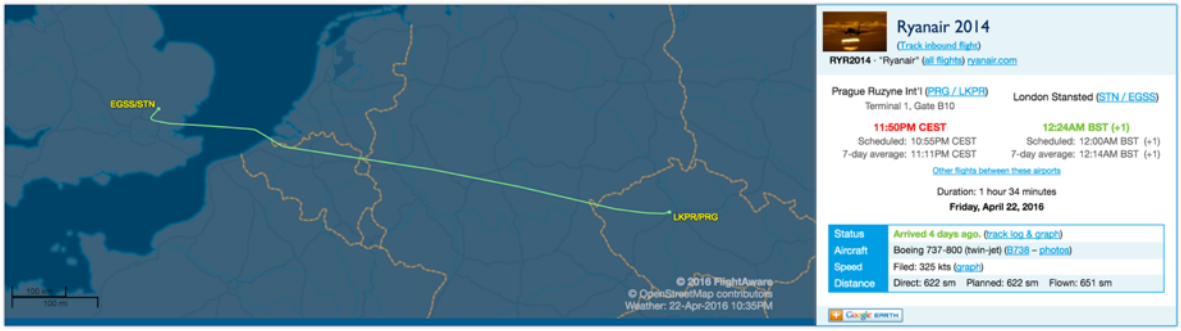
V posledních letech se mění znění leteckých nařízení ohledně zákazu používání elektronických zařízení během letů, což otevírá nový prostor na trhu s možností informovanosti cestujících a zobrazení zajímavých informací přímo během letu. Velké letecké společnosti již dnes nabízejí služby, které ukazují aktuální letové informace na integrovaných zobrazovacích zařízeních, ale stále se jedná pouze o interní zobrazení dat s vysokými pořizovacími náklady.

Cílem mé práce je umožnit takovéto informace zobrazit leteckým nadšencům na zařízeních, které mají cestující již u sebe, tudíž bez dalších přidaných nákladů. Pro kvalitní práci je nutné pro ni vytyčit jasné cíle, mantinely a prostředky. K tomuto stavu se lze dostat za pomoci studia dosavadních řešení, kritického pohledu na vlastní návrh a konzultací řešení s ostatními. Po tomto průzkumu je nutno zvolit správné prostředky na implementaci s ohledem na získané informace. Výsledná aplikace by měla být minimálně konkurenceschopná s přínosem nových pohledů na problém a na celkové možné řešení.

2 Současné trendy vizualizace letových veličin

Typickým představitelem zobrazení letových informací v letadlech je systém Garmin G1000 (Obrázek 2.1), který se skládá ze dvou LCD displejů PFD a MFD. Tento systém je určen pro umístění v pilotním prostoru a proto není možno takto zobrazené informace předat cestujícím. V poslední době je snaha zobrazovat základní letové informace cestujícím. V tomto odvětví je již několik služeb, které se zabývají zobrazením letových dat, ale vždy se jedná pouze o zobrazení trasy bez aktuálních technických údajů s nevalnou hodnotou pro technické nadšence.

Obrázek 2.1: Garmin G1000 v letounu Cessna 172.



Obrázek 2.2: Náhled zobrazení letu v flightaware.com.

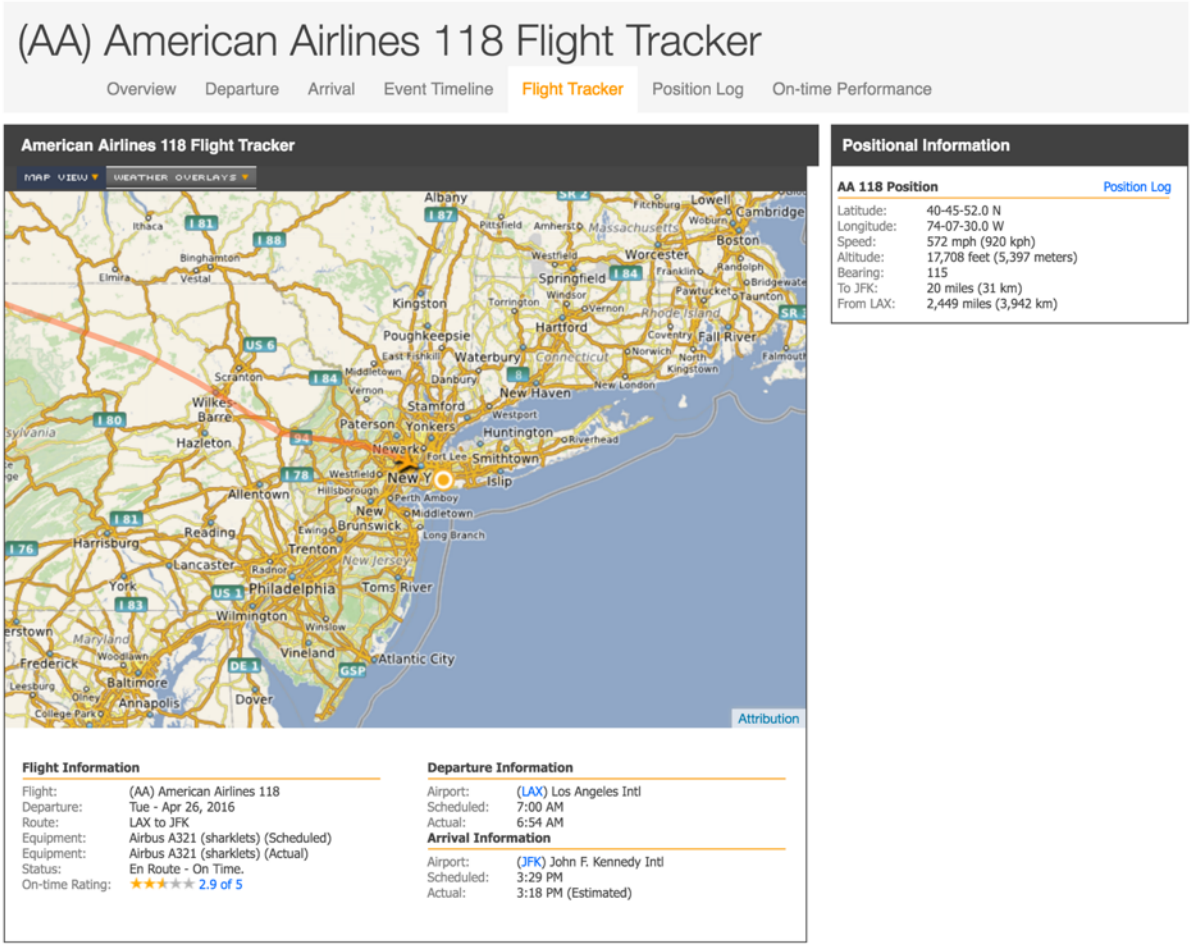
Jednoduchým zobrazovačem informací o již vykonaných letech je systém **Flightaware.com** (Obrázek 2.2) s možností vyhledávání informací o zpožděních a terminálech pomocí aerolinky a čísla letu, popřípadě pomocí míst odletu a příletu.

Výhody řešení:

- pro běžné uživatele naprosto dostačující,
- základní informace na jednom místě,
- možnost dohledání letu pomocí vyhledávače.

Nevýhody řešení:

- chybovost informací,
- neaktuálnost,
- není možnost simulace dat s posunem času,
- grafické zpracování.



Obrázek 2.3: Náhled zobrazení letu v www.flightstats.com.

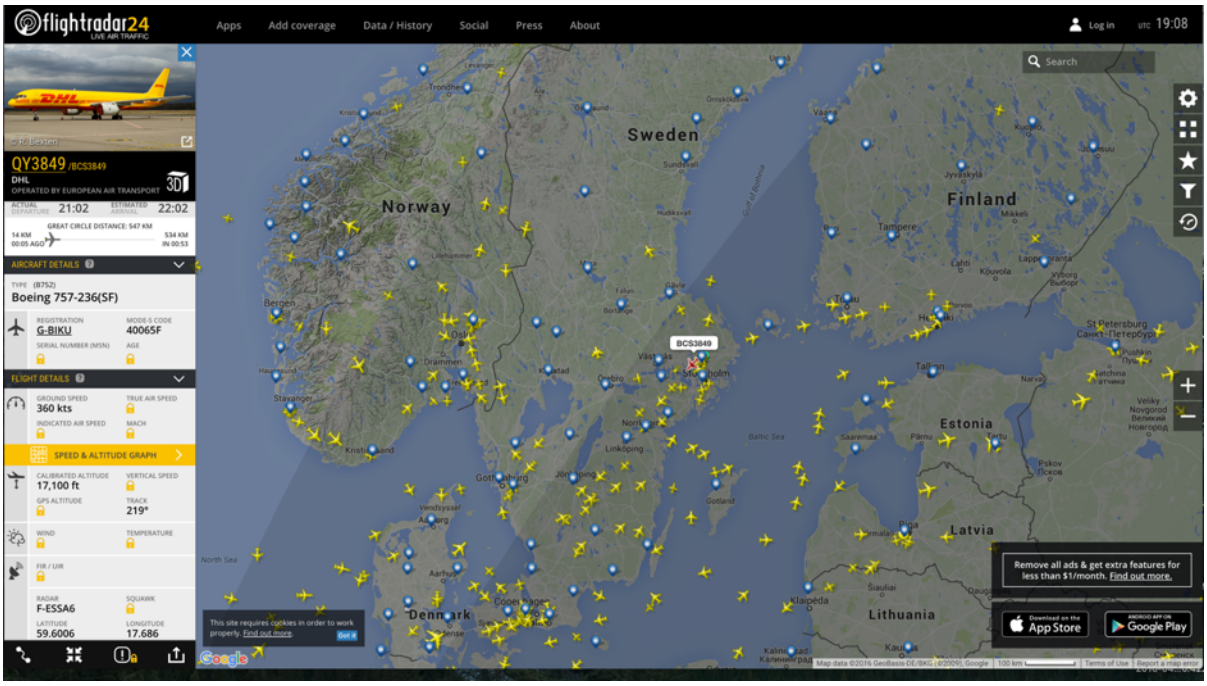
Pokročilým webovým systémem, který se zabývá zobrazením dokonce i aktuálních dat, je server **Flightstats.com** (Obrázek 2.3). Tento systém poskytuje aktuální data o probíhajících letech včetně informace o aktuální poloze v omezené míře.

Výhody řešení:

- kvalitně zpracovaný design,
- agregátor dat z mnoha zdrojů dat,
- informace o počasí.

Nevýhody řešení:

- velká prodleva v informacích o aktuální poloze,
- chybějící technické údaje.



Obrázek 2.4: Náhled zobrazení dat na www.flightradar24.com.

Nejdůmyslnějším systémem v tomto oboru je **Flightradar24.com** (Obrázek 2.4). Jedná se webový systém zobrazující aktuální letecký provoz včetně technických údajů v placené verzi.

Výhody řešení:

- uhlazený design,
- dostupné informace,
- vysoká vypovídající hodnota údajů,
- kompatibilita s mobilními zařízeními (aplikace).

Nevýhody řešení:

- velké množství dat bez možnosti filtrace.

3 Analytický rozbor

Před samotnou realizací je potřeba provést průzkum již realizovaných projektů, zabývajících se zobrazením letových informací. Tímto průzkumem je možno eliminovat značnou část slepých cest k realizaci a získat hmotnou představu, jakým směrem chceme daný projekt vést. Důležité je také projít větší množství dostupných prostředků pro realizaci a pro finální výběr vhodných nástrojů a vývojových prostředí. Dosavadní řešení jsou však značně odlišná od samotného cíle práce.

Požadavky na aplikaci

Pro úspěšnou realizaci projektu a následné uvedení na trh je potřeba nejprve získat relevantní informace o požadavcích na aplikaci. V praxi je možno použití více způsobů:

- studiem dosavadních řešení,
- diskuzí s potenciálními uživateli,
- konzultací se zadavatelem,
- studiem zobrazitelných veličin.

Kombinací výše uvedených postupů je vysoká pravděpodobnost správného návrhu aplikace od samého začátku bez nutnosti zásadních korektur během implementace a následného testování.

Při **studiu dosavadních řešení** je možno získat představu o námi chtěném řešení s jasným vytyčením hlavních znaků, které chceme v naší aplikaci využít, a naopak s jasným vyřazením funkcí a disfunkcí, které v naší aplikaci v žádném případě nechceme. Z takto získaných dat nám vyjde hodnotná informace, zdali náš projekt nebude ve výsledku značnou kopií již některého z realizovaných řešení. V takovém případě je nutno zvážit zda má cenu vykonávat úsilí ve vývoji již objeveného. Na druhou stranu může vzniknout hodnotný seznam, určující kostru celého projektu a prioritních funkcí, které chceme v našem systému aplikovat.

Diskuze s potencionálními uživateli patří k nejhodnotnějším informacím v ohledu na požadované vlastnosti. Váha zjištěných výsledků záleží na vybrání vhodných respondentů a jednoznačně na počtu respondentů, jenž se výzkumu účastní. Velmi detailní popis návrhu od jednoho respondenta má nulovou vypovídající hodnotu v porovnání s hodnotných nástinem požadavků od padesáti uživatelů, kteří o danou problematiku jeví vážný zájem.

Pokud se jedná o projekt zadaný společností, je potřeba přiklánět velkou váhu **konzultaci návrhu řešení se zadavatelem**. Takto zjištěné informace mohou být v mnoha případech směrodatné a mohou udávat téměř jasný směr vývoje. Pokud je zadavatelem předložený návrh v rozporu s ostatními informacemi, získanými jinými cestami, je potřeba dostat všechny dané rozlišnosti do funkční a realizovatelné podoby.

Technologie dostupné pro realizaci

Cílem projektu je zobrazit dané informace na zařízeních cestujících. Je tedy kladen velký důraz na kompatibilitu pokud možno na většině dnes dostupných zařízeních. Tento požadavek vylučuje použití vývojových prostředí a programovacích jazyků, které jsou určeny výhradně na daný typ zařízení či systém.

Řešením je využití schopností a nástrojů, které mají integrována všechna novodobá elektronická zařízení. Vytvoření webové aplikace na straně uživatele, spustitelné v prohlížeči, se tedy ukázalo jako nejlepší řešení [1].

Programovací jazyky na straně klienta

Základním prvkem celého projektu na straně uživatele je použití značkovacího jazyka **HTML** (HyperText Markup Language) [2], který je určen k zobrazení informací ve webových prohlížečích. Jazyk klade důraz na rozložení a formu za možnosti zobrazení bloků, obrázků, odkazů a textu. Před příchodem verze HTML 5 byl plánován přechod na jazyk XHTML, který měl zjednodušit vývoj za pomoci kompatibility s XML formátem dat. Výsledkem HTML souboru je základní kostra aplikace bez definované funkčnosti prvků a grafické formy.

Dalším stavebním prostředkem je použití jazyku **CSS** (Cascading Style Sheets) [3]. Využití kaskádových stylů umožňuje oddělení strukturální složky (HTML) webové stránky a grafickou část, která je definována právě v CSS šablonách. Toto rozdělení umožňuje snadnou manipulaci a editaci kódu bez velkých problémů. Potíž může nastat při implementaci řešení, která se v různých prohlížečích chovají odlišně. Je tedy dobré se těmto problematickým situacím pokud možno vyhnout, jinak bude potřeba trávit mnoho času při optimalizaci na různé prohlížeče. Oddělení grafické a strukturované složky umožňuje nalinkováním jiného stylu změnit kompletní vzhled dané stránky, tak také naopak již nadefinovaný vzhled použít na více stránkách.

Jazykem vykonávajícím funkčnost dané stránky na straně klienta je programovací jazyk **JavaScript**. Tento jazyk umožnuje dynamickou úpravu prvků stránky, interakci s uživatelem, vytváření animací a také umožňuje interakci se zdroji dat na straně serveru. Jedná se o multiplatformní, objektově-orientovaný programovací jazyk, který je ideální pro projekty určené pro více systémů.

Programovací jazyky na straně serveru

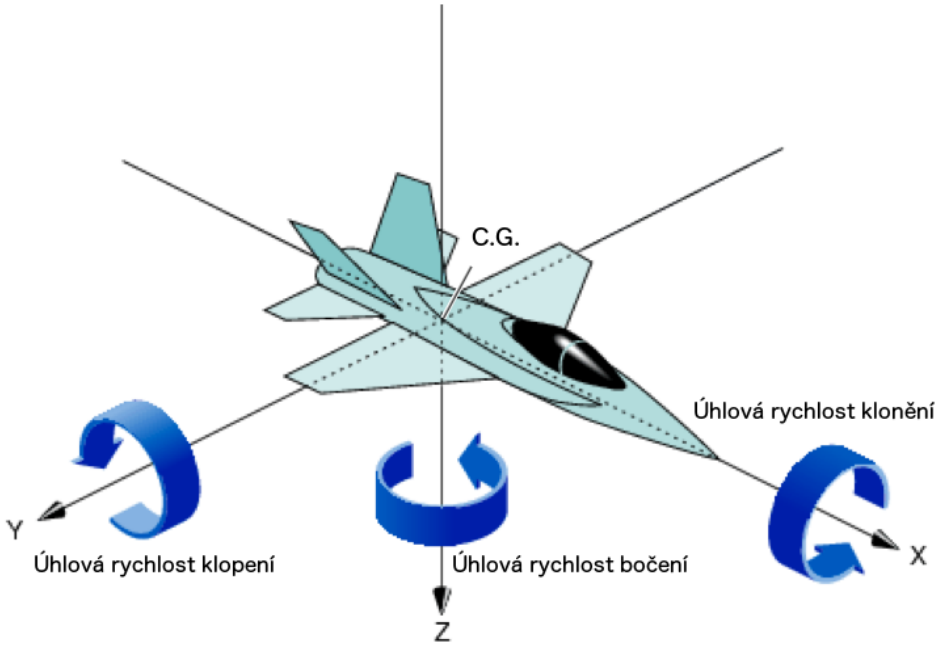
Hlavním prostředkem na straně serveru je skriptovací jazyk **Python** ve verzi 2.7 [4], který je využíván hlavně kvůli jednoduchému napojení na více komunikačním kanálů. Samotná realizace daných funkcí je v tomto jazyce velice přirozená a implementačně přívětivá.. Jedná se o jazyk s dynamickou kontrolou datových typů s podporou různých programovacích paradigmat, včetně objektově orientovaného, imperativního, procedurálního nebo funkcionálního. Python je vyvíjen jako open source projekt, který je multiplatformní. Ve většině distribucí systému Linux je Python součástí základní instalace. Samotný programovací jazyk Python je implementován v jazyce C (označován jako CPython). Aplikace, napsané v Pythonu, se vyznačují rychlostí a spolehlivostí, jelikož většina základních knihoven je implementována v jazyce C. Tím je i zaručena vysoká kompatibilita s tímto jazykem. Můžeme se setkat se službami, které jsou kompletně implementovány ve skriptovacím

jazyku Python, avšak jeden kritický modul projektu je implementován v jazyce C. Python balíčky již podporují většinu komunikačních protokolů, například FTP, HTTP, TELNET, POP3, SMTP, IMAP, SSH, které umožňují implementaci komunikačních nástrojů mezi různými prostředími. Python části projektu poskytne propojení mezi získanými daty ze zdroje zpracování a odeslání do části na straně klienta.

Další prostředek, který se dá využít v projektu, je **Redis** [5]. Jedná se o open source prostředek, který dokáže ukládat strukturovaná data do vyrovnávací paměti, čímž zaručí vysokou rychlost jak při čtení, tak zápisu dat. S pamětí se pak pracuje jako s databází či cache a umožňuje ukládat řetězce, hashovací tabulky, pole, množiny, množiny s řazením a mnoho dalších.

Zobrazitelné technické údaje

Hlavním prvkem celé aplikace bude zobrazit co nejpřesnější simulaci pohledu z pilotní kabinu. K tomuto kroku se váže výběr technických veličin, které se budou zobrazovat.



Obrázek 3.1: Pohyb kolem osy x - klonění, y - klopení, z- bočení.

Každé letadlo se může pohybovat kolem tří os X, Y a Z, které určují základní orientační osy v třírozměrném prostoru (Obrázek 3.1).

Při **otáčení kolem kolmé osy Z** (osa od středu Země procházející těžištěm letadla) letadlo zatáčí. Tento pohyb je vyvolán pilotem při zásahu do směrového kormidla, které je pohyblivou součástí svislé ocasní plochy. Pilot směrové kormidlo ovládá pedály.

Klasická zatáčka musí být kombinací použití směrového kormidla a také pohybu křidélek. Křidélka jsou umístěna na koncích křídel. Pilot je ovládá řídící pákou v bočním směru a ovlivňuje tak **otáčení letadla kolem osy X** (osa procházející trupem letadla v přímém směru).

Naklopení letadla podél příčné osy Y (osa vedoucí křídlem) tedy způsobuje naklopení nahoru a dolů. Tento pohyb je způsoben ovládáním výškovky, což je pohyblivá část umístěná na vodorovné ocasní ploše. Pilot ji ovládá kniplem ve podélném směru tak, že pohyb od sebe způsobuje naklopení dolů a pohyb k sobě má za následek klopení nahoru.

Mezi zobrazenými veličinami nesmí chybět aktuální rychlost, kterou právě letadlo letí. Rychlost letadla může být dvojího typu. Je buď určena rychlostí letadla v porovnání se zemským povrchem, nebo v porovnání s aktuálním letovým prostředím. V tomto údaji jsou zohledněny i povětrnostní podmínky ovzduší, takže při silném protivětru je relativní rychlost (AIS - indikovaná vzdušná rychlost) vyšší než rychlost vůči zemskému povrchu (GS) a naopak.

Jednou z důležitým hodnot je aktuální výška, ve které se letadlo momentálně nachází. Tento údaj může být uveden vůči zemskému povrchu či nulové nadmořské výšce. Tento údaj je spolu s aktuální rychlostí jedním z nejkritičtějších a je tedy nutno ho umístit na viditelné místo.

Pokud chceme zaručit vypovídací hodnotu celého zobrazení, je nutno zajistit zobrazení aktuálních GPS souřadnic, včetně zobrazení záchytných bodů.

Kompas je také nedílnou součástí vybavení kokpitu letadla a není možné hodnotu aktuálního směru letu přeskočit a nezobrazovat.

4 Návrh řešení

V této kapitole je popsán postup návrhu řešení práce a postupný výběr vhodných prostředků k jeho uskutečnění. V návrhu práce je potřeba zohlednit všechny zjištěné aspekty projektu a navrhnout co nejlepší řešení, které bude splňovat cíle, které od aplikace očekáváme a které budou splnitelné v rozumném časovém horizontu. Na těchto základech je vybudována aplikace zobrazující výhled z kokpitu dopravního letadla včetně zobrazení mapy, veličin a umělého horizontu.

Časový plán

Jedním ze základních faktorů projektu je časový rámec a plán, v jakém se jednotlivé části budou implementovat. První fází průzkumu je třeba strávit dostatek času pro zajištění a získání kvalitních a vypovídajících údajů. Další navazující fáze musí časově korespondovat s celkovým časem určeným na realizaci. Tento přístup zajistí dokončení projektu v určený čas.

Časový harmonogram řešení projektu:

- studium dosavadních řešení,
- studium dostupných technologií k realizaci,
- grafický návrh aplikace,
- návrh datové struktury a komunikace celého projektu,
- implementace části na straně klienta se simulovanými vstupy,
- testování a opravy části na straně klienta,
- implementace napojení na existující letecký systém,
- testování a opravy části na straně serveru,
- napojení částí a finální testování,
- zveřejnění finálního řešení.

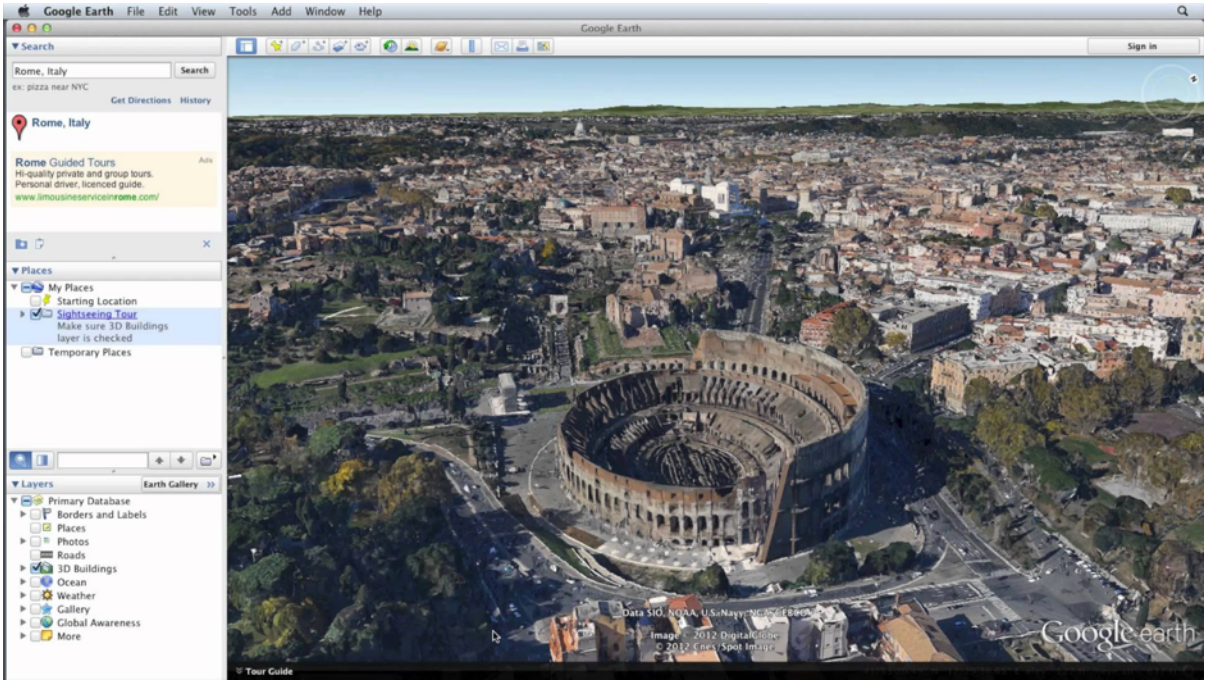
Návrh prvků a zobrazených veličin

Hlavními prvky aplikace bude jak výhled z pilotního prostoru do krajiny, tak i modul palubní desky, který bude obsahovat umělý horizont, hodnoty aktuální rychlosti a výšky a mapu zobrazující polohu na mapě. Cílem rozložení těchto prvků bude snaha zajistit udržení jednoho smysluplného zobrazení jak na větších zařízeních s patnáctipalcovými displeji, tak i na mobilních zařízeních s úhlopříčkou čtyř palců.

Cílem realizace bude vytvořit aplikaci s čistým grafickým rozhraním bez rušivých elementů a tím vytvořit uživatelsky příjemné prostředí pouze s prvky, které mají upoutat pozornost. Takto vytvořená aplikace má ve výsledku daleko větší šanci na úspěch a popularitu než aplikace s množstvím pohyblivých a rušivých prvků, které odvádějí pozornost od těch prvků, které jsou součástí hlavního návrhu dané aplikace.

Knihovny pro zobrazení výhledu z kokpitu a mapy

V práci půjde hlavně o co nejkvalitnější zobrazení výhledu z kabiny a celkově polohu na mapě - to znamená získat co nejlepší představu o možnostech, jak tuto virtuální realitu zobrazit.

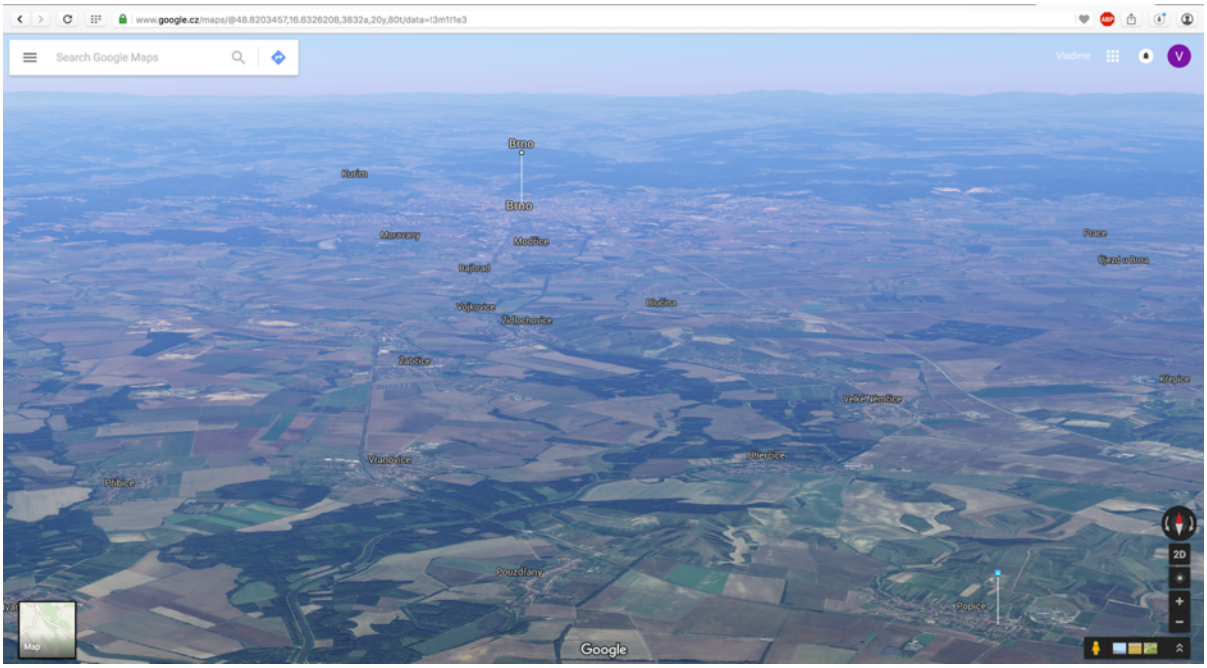


Obrázek 4.1: 3D zobrazení map v GoogleEarth.

Rozšířená služba pro 3D zobrazení země je **GoogleEarth** (Obrázek 4.1), která získala oblibu v celém světě. Přehled výhod a nevýhod je zobrazen v Tabulce 4.1.

Tabulka 4.1: GoogleEarth

Výhody:	**Nevýhody:**
• rozšířenost, • podpora více platforem.	• ukončená podpora API, • nutnost instalace GoogleEarth modulu do prohlížeče.

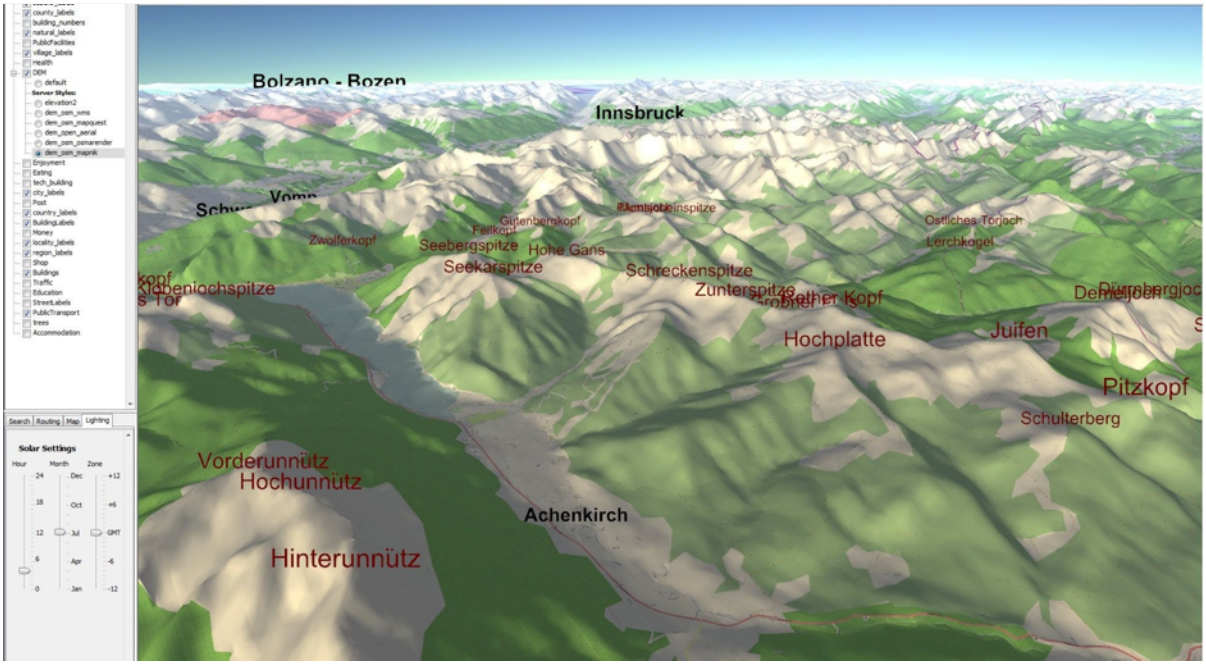


Obrázek 4.2: 3D zobrazení map v GoogleMaps.

Mezi nejznámější službu s mapami patří **GoogleMaps** (Obrázek 4.2). Poskytuje uživateli nespočet typů zobrazení map od klasické cestovní až po 3D zobrazení budov ve městech a zobrazení fotografií pořízených pod 45° úhlem. Přehled výhod a nevýhod je zobrazen v Tabulce 4.2.

Tabulka 4.2: GoogleMaps

Výhody:	**Nevýhody:**
• rozšířenost, • jednoduché API, • podpora více platforem.	• 3D verze není podporována přes API, • omezená použitelnost kvůli limitacím licencí.



Obrázek 4.3: 3D využití map od OpenStreetMap.

Open source službou s mapami včetně podpory 3D zobrazení je **OpenStreetMap-3D** (Obrázek 4.3), avšak v omezené míře a kvalitě. Přehled výhod a nevýhod je zobrazen v Tabulce 4.3.

Tabulka 4.3: OpenStreetMap-3D

Výhody:	**Nevýhody:**
• open source projekt, • podpora více platforem.	• omezená funkčnost v 3D zobrazení.

Obrázek 4.4: Využití knihovny CesiumJS.

Knihovna **CesiumJS** (Obrázek 4.4) pro zobrazení 3D map, postavená na programovacím jazyku JavaScript a WebGL s velkým výběrem jak mapových, tak terénních podkladů včetně jednoduché implementace a interakce s webovým prostředím. Přehled výhod a nevýhod je zobrazen v Tabulce 4.4.

Tabulka 4.4: CesiumJS

Výhody:	**Nevýhody:**
• open source projekt, • podpora všech aktuálních platforem, • kvalita podkladů a map, • neomezené možnosti nastavení a modifikací, • srozumitelné API, • nativní podpora widgetu, • podpora zobrazení dynamické vizualizace, • snadná migrace z GoogleMaps.	• vysoké nároky na hardware.

Po důkladném průzkumu všech dostupných prostředků pro implementaci a začlenění grafického zobrazení terénu do práce jsem zvolil knihovnu **CesiumJS** [6]. Tato knihovna splňuje kritické požadavky, jež jsou od knihovny v aplikaci požadovány. Hlavní přednosti, kvůli kterým jsem knihovnu zvolil, jsou možnosti specifikace nastavení a kompatibility na většině platforem.

Původní návrh počítal s využitím prostředků GoogleEarth, avšak od této možnosti bylo nutno ustoupit z důvodu ukončení API přístupu a ukončení podpory ve více platformách.

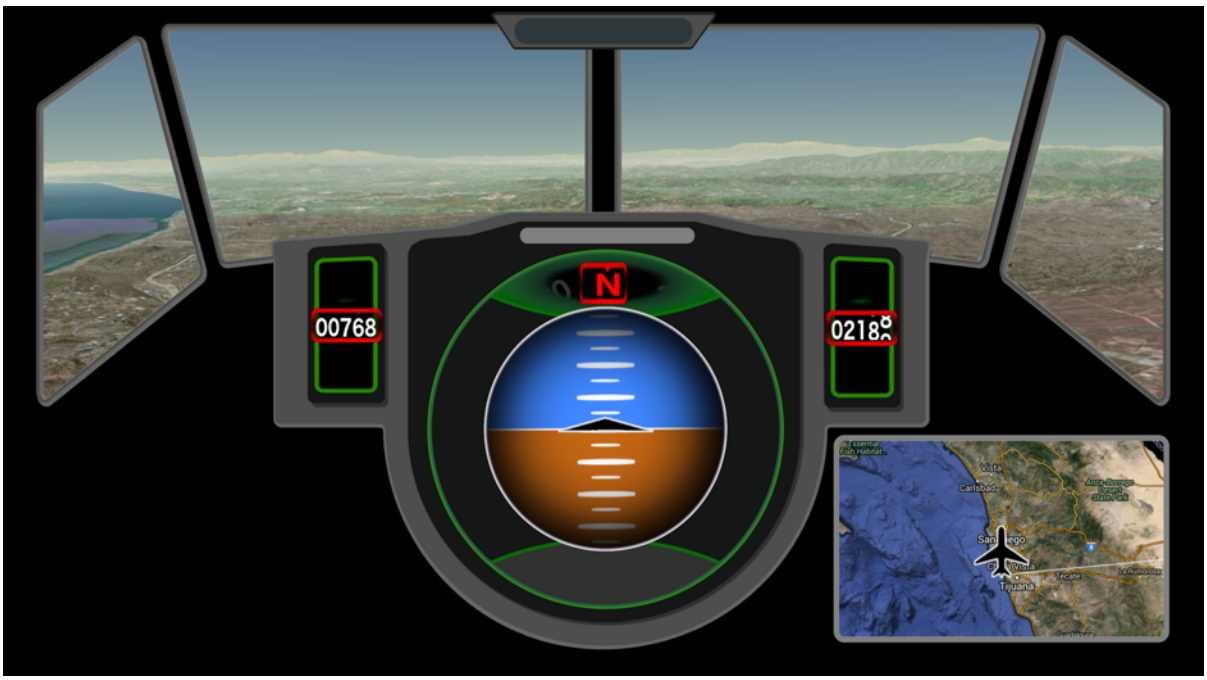
V průběhu práce nakonec byla použita i služba **GoogleMaps** [7], která zastřešuje zobrazení aktuální polohy letadla na mapě. Tato služba se vyznačuje jednoduchou obsluhou a proto její začlenění do aplikace nic nebránilo. Na rozdíl od CesiumJS má minimální nároky na hardware, tudíž kombinace těchto dvou prostředků zaručuje bezproblémový běh, kdy jsou zásadně čerpány hardwarové zdroje jenom v nevyhnutelných případech.

Ovládací prvky aplikace

Aplikace má být určena k zobrazení aktuálních veličin a není tedy nutno řešit výraznou aktivní interakci s uživatelem. Navzdory tomuto faktu však zobrazení bude obsahovat aktivní prvek na aktivaci zobrazení mapy. V základním zobrazení bude aplikace zobrazovat aktuální výhled z kokpitu, umělý horizont, kompas a dané hlavní veličiny.

Grafický návrh aplikace

Hlavním cílem aplikace je upoutání uživatele. Jak již bylo zmíněno, kompletní vzhled tady hraje nejvýznamnější roli.



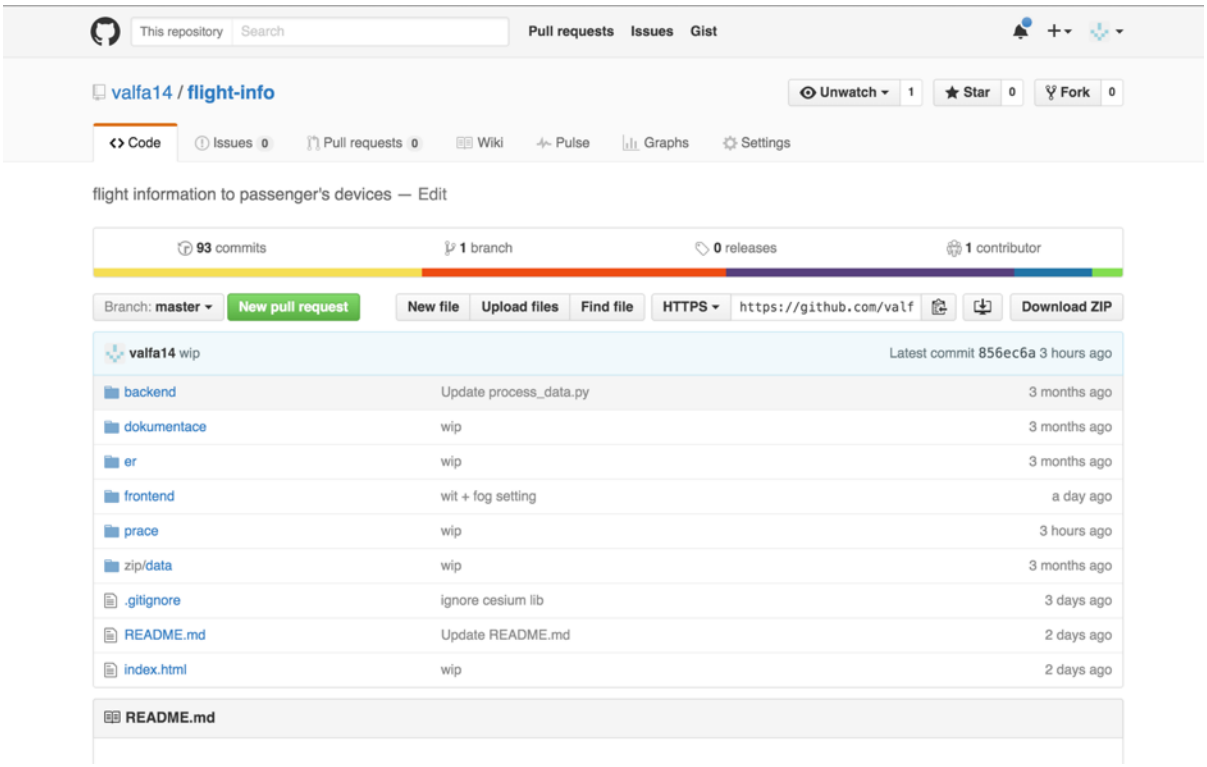
Obrázek 4.5: Celkový náhled na grafický návrh aplikace.

Na obrázku 4.5 je vyobrazen grafický námět aplikace. Jedná se o návrh bez jakékoliv funkčnosti či modularity. Slouží k získání reálného pohledu na realizovaný projekt. Již náhled dokáže odhalit spoustu chyb a záludností, které bude nutno v průběhu implementace řešit. Samotná realizace řešení by měla vést z dosažení vytvoření aplikace shodného vzhledu s veškerou funkčností jednotlivých prvků.

Problémy zjištěné během implementace, které působí odchylky od původního návrhu, by měly vést k vytvoření nového modifikovaného návrhu, aby se předešlo budování slepé cesty vývoje a tím i ohrožení celého projektu.

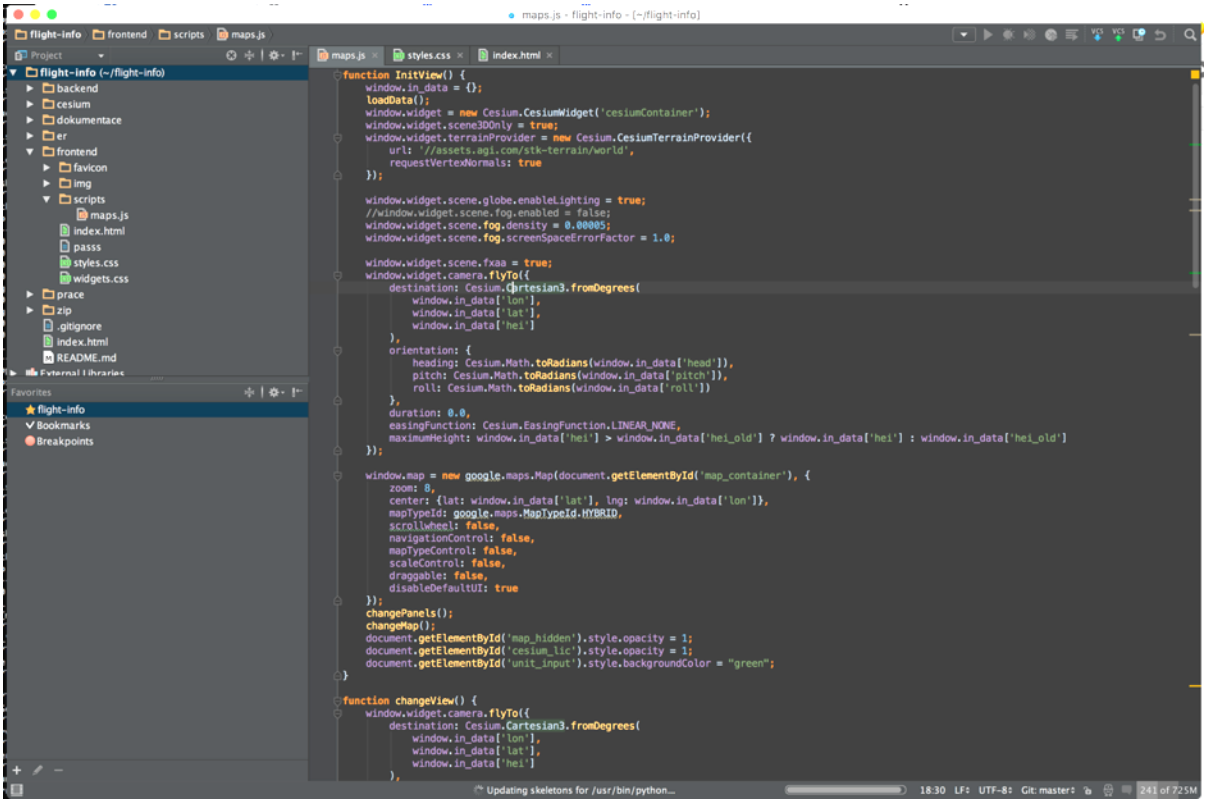
Použité vývojové prostředky

Pro vývoj aplikace byly využity vývojové prostředky usnadňující práci s projektem a zajišťující hladký průběh vývoje.



Obrázek 4.6: Celkový náhled na projekt ve webovém prostředí Github.com.

Pro udržení přehlednosti projektu je využit systém správy verzí Git. Jedná o službu **Github.com** (Obrázek 4.6), jejíž hlavní předností je jednoduchá obsluha, příjemné uživatelské prostředí a rozšířenost. Díky využití tohoto nástroje je správa celého projektu jednodušší a je minimální riziko ztráty nějaké části vyvíjené aplikace. Všechny provedené změny jsou ukládány a zaručují případné zpětné dohledání předchozích verzí v průběhu vývoje. Také umožňují jednoduchou distribuci řešení na více výpočetních strojů bez nutnosti manuálního kopírovaní zdrojového kódu.

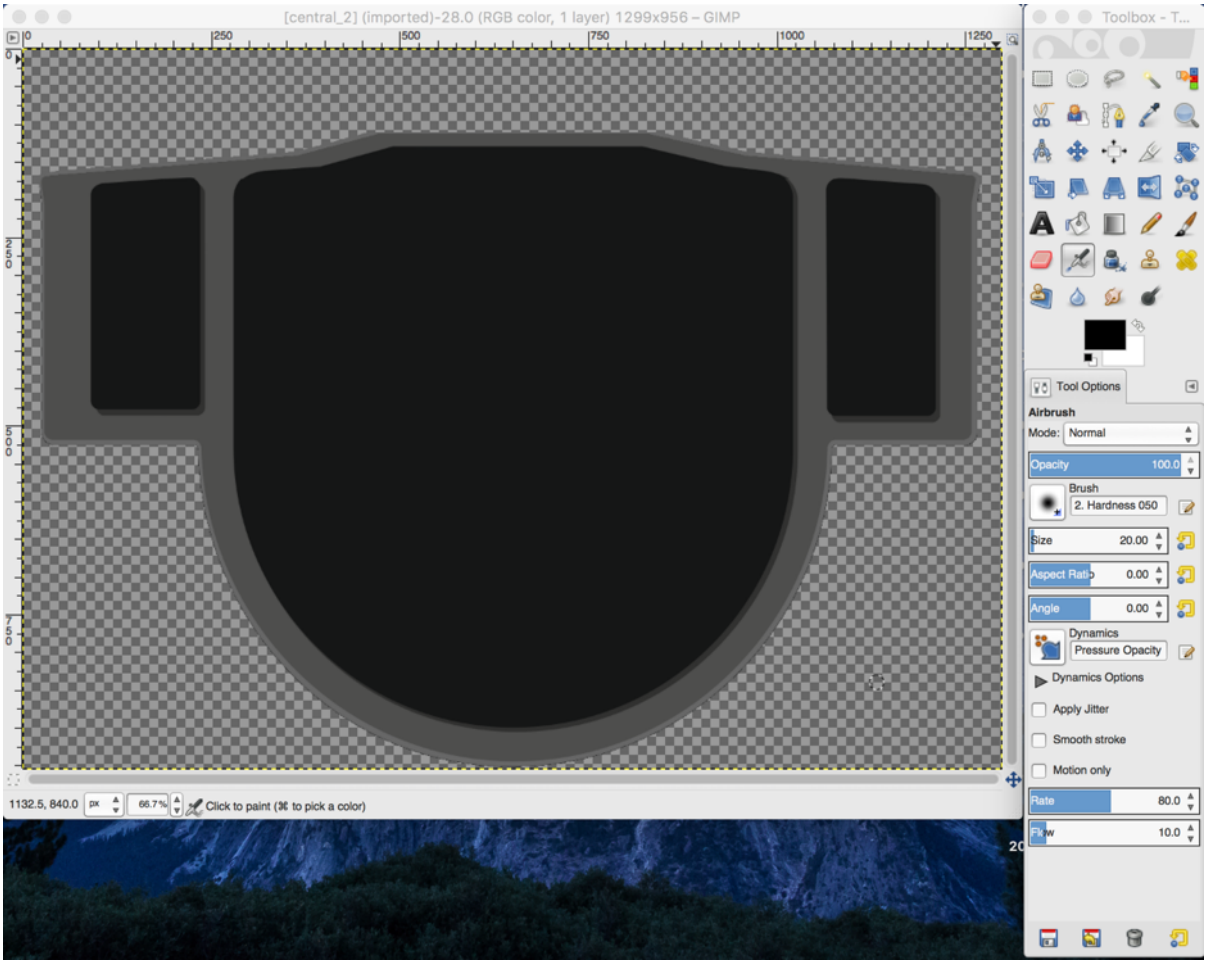


Obrázek 4.7: Náhled na projekt v prostředí PyCharm.

Celý projekt je psán v textovém editoru **PyCharm** (Obrázek 4.7). Jedná se o produkt z rodiny editorů, vyvíjený společností JetBrains. PyCharm jako takový je určen k vývoji pro programovací jazyk Python, ale plně podporuje zobrazení syntaxe a závislostí i ostatních jazyků jako je použitý JavaScript, HTML a CSS. Díky tomuto prostředku jsou minimalizovány chyby způsobené překlepy v průběhu vývoje. Jedná se zejména o kontrolu deklarace funkcí a proměnných v různých souborech, které mají vzájemnou závislost. Další velkou výhodou je možnost kompletního nastavení celého editoru od barevné varianty pro zamezení únavy očí až po nastavení hraničních hodnot pro kontrolu korektního stylování a zajištění přehlednosti kódu.

Webový prohlížeč **Opera Browser**, odvozený od jádra prohlížeče Chrome, poskytuje funkci pro editaci aktuálně spuštěné webové stránky. Tímto nástrojem lze upravovat implementované části takřka v reálném čase a tím i ladit umístění, překryt a vlastnosti daných prvků na stránce bez nutnosti přímé editace zdrojových kódů. Po úspěšném usazení prvku tak stačí upravené hodnoty jednou uložit do původního souboru a vše je již otestováno.

Přístup k terminálu přes program **iTerm** dovoluje jednoduchou a účelnou práci s příkazovou řádkou a vývojovým prostředím. Díky tomuto přístupu je spuštění jednoduchého lokálního serveru otázkou sekundy a dovoluje nám vyvíjet webové aplikace v lokálním prostředí bez jakýchkoli omezení. V kombinaci se základní instalací pythonu tak máme vše potřebné ke kompletnímu vývoji.

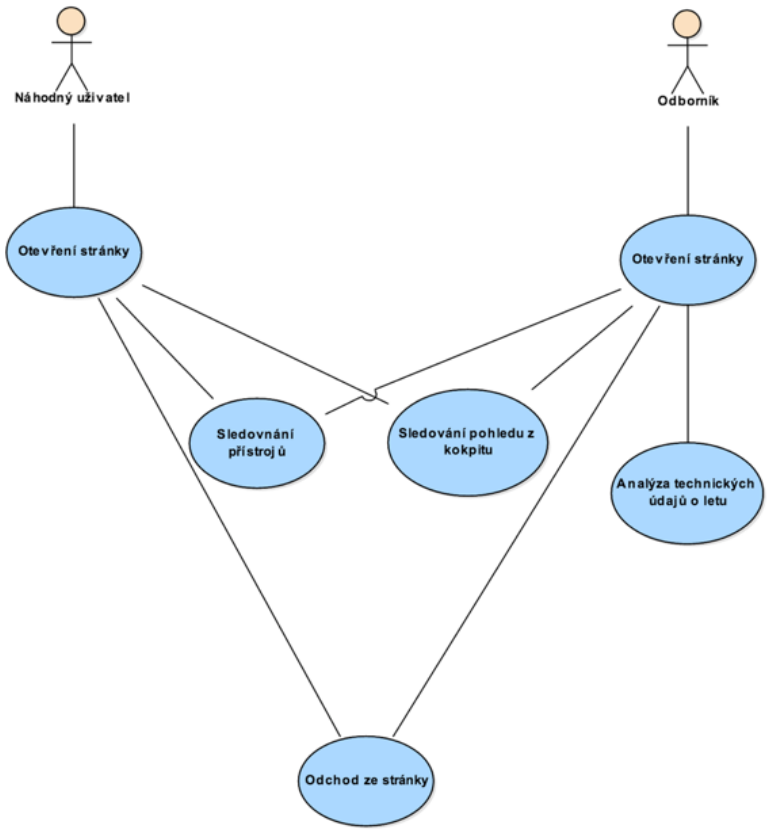


Obrázek 4.8: Otevřené okno grafického editoru Gimp s průhledností vrstev.

Volně dostupný grafický editor **Gimp** (Obrázek 4.8) je jednou z možných alternativ profesionálních nástrojů pro úpravu obrázků a grafiky. V tomto prostředí lze vytvářet a upravovat rastrové i vektorové objekty, včetně nativní podpory průhlednosti jednotlivých vrstev a jednoduché editace vybraných výseků. Jedná se o rozšířený editor s nezměrným počtem rozšířených funkcí. Editor dovoluje export grafických objektů a snímků do mnoha formátů včetně hodně využívaného formátu PNG.

Diagram případu užití

Na obrázku 4.9 je vidět základní diagram případu užití v jazyku UML, který vychází ze základních specifikací potřeb dvou rozdílných uživatelů. V diagramu figurují dva aktéři „Odborník“ a „Náhodný uživatel“.



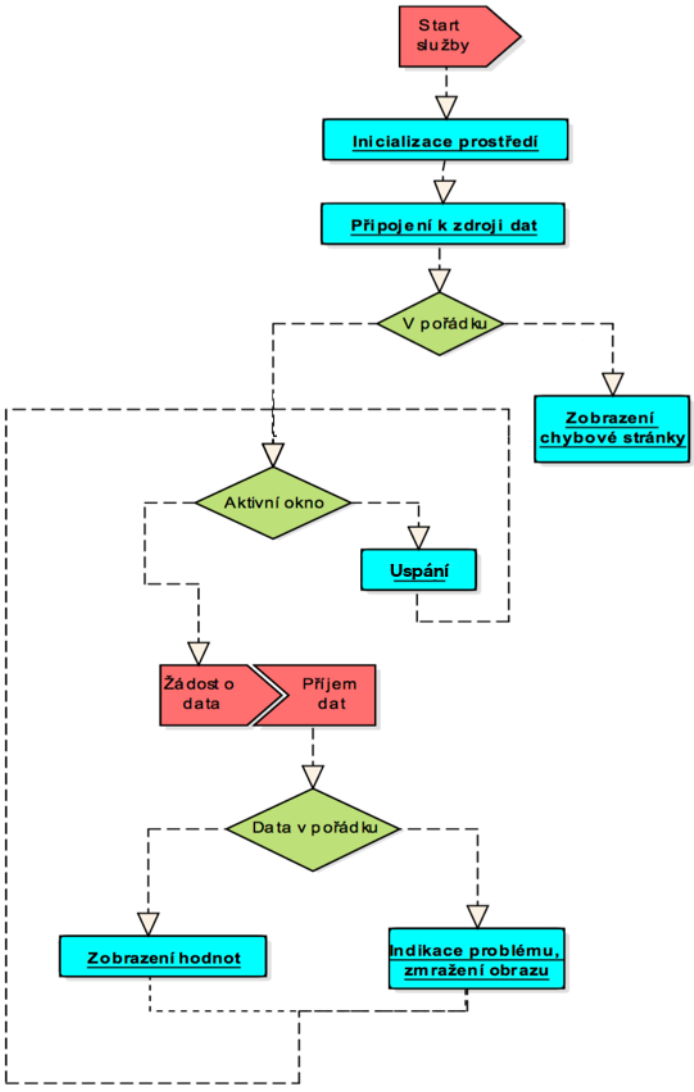
Obrázek 4.9: Diagram případu užití

Graf na Obrázku 4.9 zobrazuje rozdílné využití jednotlivými subjekty, kteří budou se systémem pracovat. Jak lze z diagramu vyčíst, „Odborník“ rozšiřuje pohled „Náhodného uživatele“ o adekvátní analýzu zobrazených informací, díky svým nabytým odborným znalostem, kdežto „Náhodný uživatel“ bude pouze sledovat zajímavost zobrazení bez hlubší analýzy zobrazovaných veličin. Z grafu je jasně čitelný fakt, že interakce aplikace s uživatelem bude ve své podstatě nulová, jelikož se jedná o zobrazovač daných informací.

Entitní vztahové diagramy

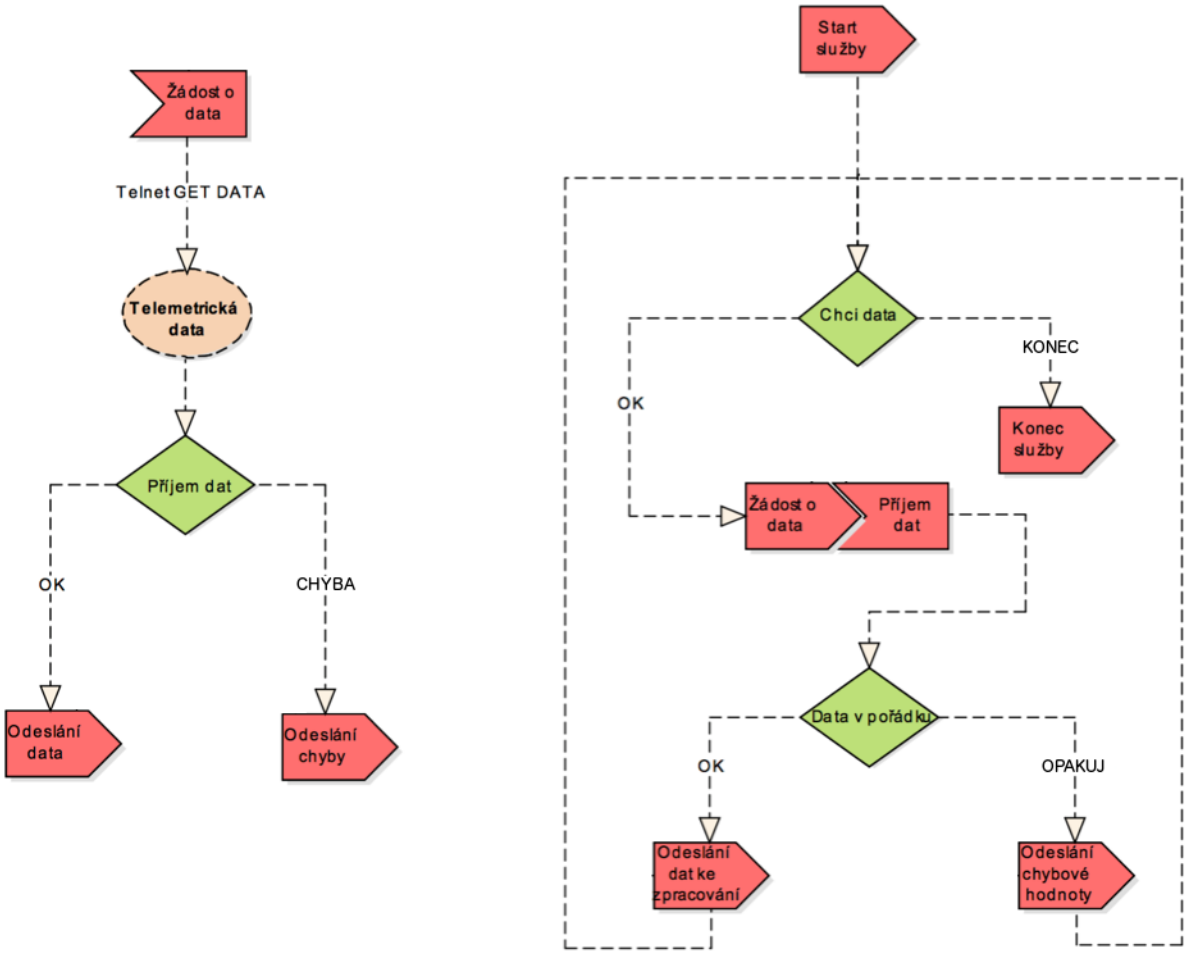
K úspěšné realizaci celého projektu je potřeba vytvořit kompletní mapu celého projektu jak datové, tak zobrazovací části. V projektu je nutno jasně definovat prostředky, díky kterým budou jednotlivé části navzájem komunikovat. Díky takto definovaným podmínkám lze minimalizovat situace, kdy by se mohla část či celý systém dostat do nedefinovaného stavu, který nebude možno testovat a zaručit validitu jeho výstupu. Při návrhu je nutno brát ohledy na jednotlivé komponenty projektu a možnosti jejich komunikace. Pokud se jedná o nutnost přenosu kritických dat, které mají za následek úpravu stavu celého systému, je nutno zaručit stabilní a spolehlivý přenos těchto informací s největší prioritou. Na druhou stranu data, zasílaná pouze v případě stavu, kdy se aplikace vyvíjí, nemusí mít připravenou strukturu komunikace pro stabilní použití v běžném provozu.

Při přihlédnutí k těmto základním požadavkům vznikne struktura, která by měla být odolná vůči zatížení s podporou rozšiřitelnosti dat přenášených během vnitřní komunikace jednotlivých komponentů.



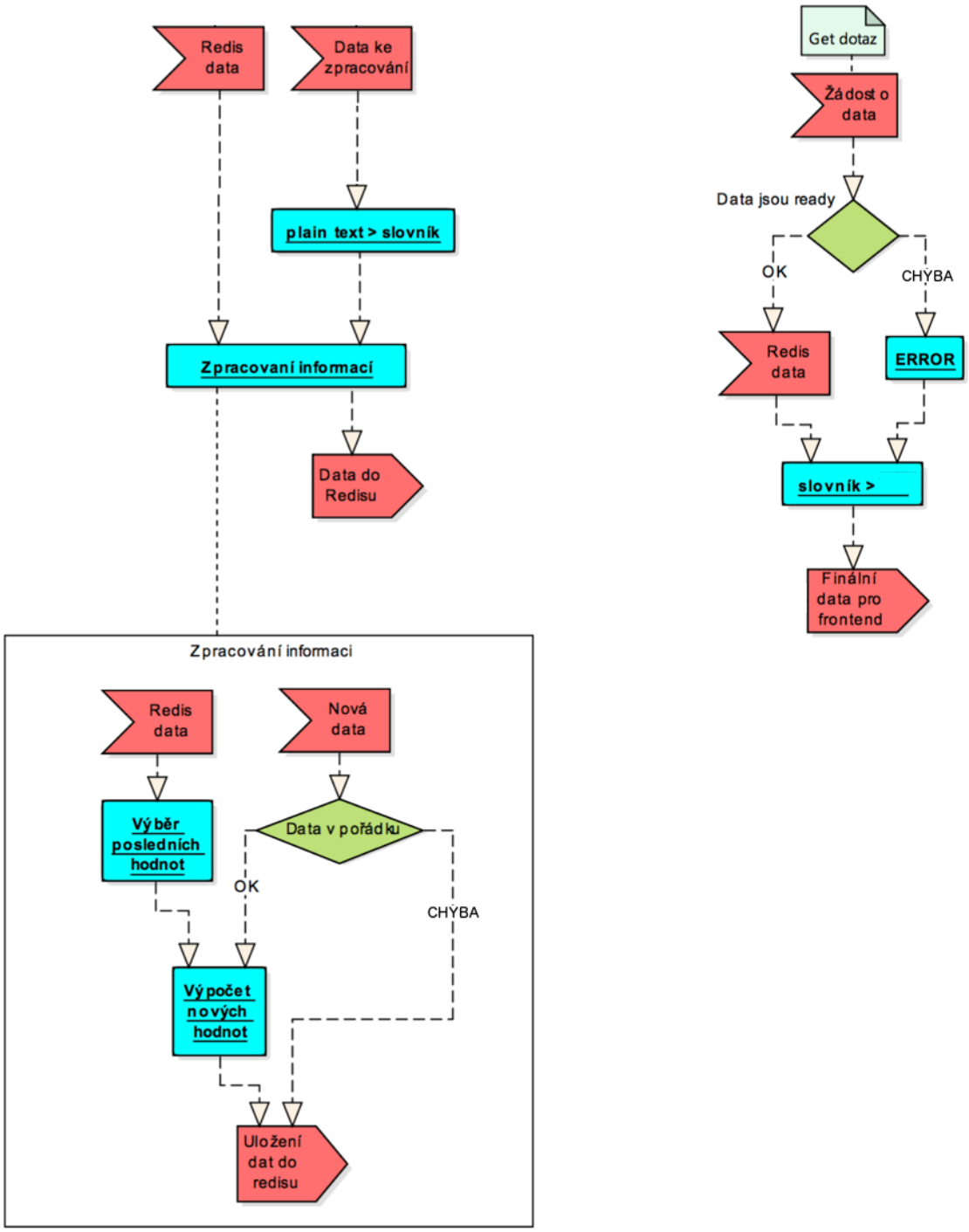
Obrázek 4.10: Generický ER diagram projektu.

Na obrázku 4.10 je ER diagram popisující jednotlivé stavy a sekce celého projektu. Jedná se o postupnou inicializaci prostředí a načtení jednotlivých grafických komponentů. Po úspěšném načtení je provedena kontrola dostupnosti dat, určených k zobrazení. Při běhové chybě během inicializace je zobrazena chybová hláška a tím je uživatel informován o daném problému. Při aktivním zobrazení je nutno zaručit systematické získávání dat. O tato data je periodicky žádán server zpracovávající zdrojová data od leteckého systému.



Obrázek 4.11: ER diagramy sekce získávající technická data z leteckého systému.

Na diagramech, zobrazených na obrázku 4.11, je vyobrazen návrh sekce, která má na starosti komunikace mezi aplikačním systémem a jednotkou letadla, která poskytuje technické údaje o aktuálním stavu letadla. Tyto informace, získané v textové podobě, je nutno zpracovat pro další komunikaci v rámci interní komunikace aplikace. Bez této korektury by nebylo možné zaručit konzistenci datové struktury. Rozšíření aplikace pro více různých vstupních systémů by tak bylo velice náročné.



Obrázek 4.12: ER diagramy zobrazující sekci pro práci se zpracovanými daty.

Při zpracovávání velkého množství dat je nutné zaručit jejich dostupnost i při drobném výpadku jejich získávání. Jedním řešením může být řešení zobrazené na digramech na obrázku 4.12. Klíčem k tomuto řešení je využití postupného ukládání zpracovaných informací a jejich dočasné uchování v Redis databázi. Toto řešení je implementačně jednoduché a zaručuje rychlost a spolehlivost i při vysokém zatížení systému.

5 Realizace aplikace

Po finálním návrhu a vytvoření struktury, na které bude celá aplikace postavena, je na řadě samotná implementace daného řešení, které bylo zvoleno v návrhové části projektu.

Adresářová struktura projektu

Cílem navržené struktury projektu bylo zaručit přehlednost daného řešení a tudíž i hierarchickou strukturu jednotlivých komponent. V kořenové složce se nachází tři složky: **frontend, backend a cesium**, určující základní stavební celky.

Adresář pojmenovaný **frontend** zaštiťuje část projektu obsahující zdrojové kódy a elementy části aplikace na straně klienta. Tyto soubory jsou dále řazeny do příslušných složek.

Složka **backend** obsahuje kompletní strukturu pro část běžící na straně serveru. V této složce jsou všechny python skripty nutné pro korektní běh aplikace.

Adresář **cesium** obsahuje volně dostupnou knihovnu CesiumJS (aktuálně ve verzi 1.20), nutnou pro korektní zobrazení trojrozměrného horizontu v aplikaci.

Implementace rozhraní na straně klienta

V této části práce je popsán vývoj a problémy, se kterými jsem se setkal během implementace, týkající se realizace aplikace na straně klienta.

V kořenovém adresáři pro tuto část **frontend** jsou umístěny složky:

- img – obsahující všechny statické grafické prvky aplikace (soubory formátu PNG),
- scripts – obsahující Javascript zdrojové soubory, starající se o dynamiku aplikace,
- favicon – obsahující všechny informace o ikoně

a soubory:

- index.html – HTML struktura projektu,
- styles.css – CSS struktura projektu,
- widgets.css – CSS struktura CesiumJS widgetu.

Základní stavební prvky rozhraní

Definice celkového zobrazení a rozložení stránky je v souboru index.html, kde je definována celková HTML struktura aplikace. V hlavičce tohoto souboru je nalinkována šablona kaskádových stylů ze souboru styles.css, kde je určena konkrétní podoba jednotlivých elementů zobrazené stránky a javascript soubor maps.js, obsahující definice funkcí definující chování elementů.

Celá stránka je ohraničena tenkým rámem k zaručení čitelnosti všech elementů nedotýkajících se krajů a celkové grafické estetičnosti práce. Tělo stránky tvořeno `div` elementem `main`, který je rozdělen do sekcí díky elementům identifikovaných pomocí unikátních názvů `unit_panel`, `over_panel`, `terrain_panel`, `top_panel`, `horizont_panel`, `center_panel` a `bottom_panel`.

Element „unit_panel“

Element nacházející se nad výhledem z kabiny (Obrázek 5.1). V tomto elementu je umístěn element `unit_input` na ovládání zobrazení licenčních údajů pomocí vyvolání funkce `hideLic()`, ovládá viditelnost licenčních značek mapových podkladů, díky editaci stylových atributů elementu `cesium_lic` a `map_container`. Díky kliku je také měněna barva samotného elementu, jakožto indikace aktuálního stavu.

Obrázek 5.1: Element unit_panel ovládající zobrazení licenčních značek.

Element „over_panel“

Jedná se o jeden ze dvou hlavních elementů aplikace (Obrázek 5.2), ukrývající kompletní implementaci zobrazení centrálního panelu.

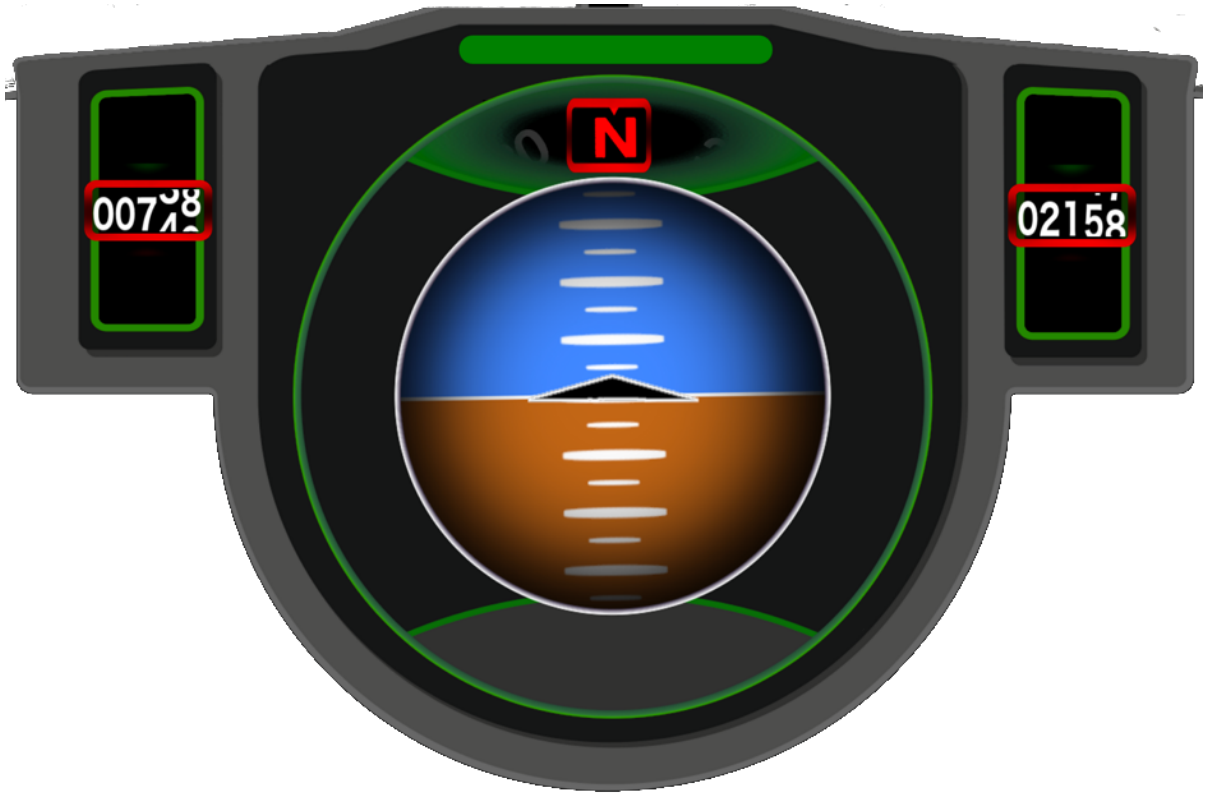
V samotném elementu jsou další tři elementy `left_center`, `central_center` a `right_center`, dělící zobrazovací prostor na menší, lépe zpracovatelné jednotky.

Levá i pravá část vycházejí ze stejného formátování, v kterém se nacházejí elementy, zobrazující aktuální letovou rychlost v levém panelu a aktuální výšku v pravém panelu. V pozadí těchto elementů jsou zobrazeny aktuálně probíhající změny těchto hodnot pomocí zelených a červených hodnot vyjadřující nárůst či pokles hodnot.

V centrálním panelu `central_center` je umístěn prvek `signal_input` volající funkci `hideMap()`, která ovlivňuje zobrazení mapy, zobrazující aktuální polohu. Toto zobrazení je ve výchozím nastavení vypnuto.

Dále panel obsahuje element `direction_view`, která zobrazuje směr letu pomocí zobrazení aktuální hodnoty na kompasu, na kterou daný let míří.

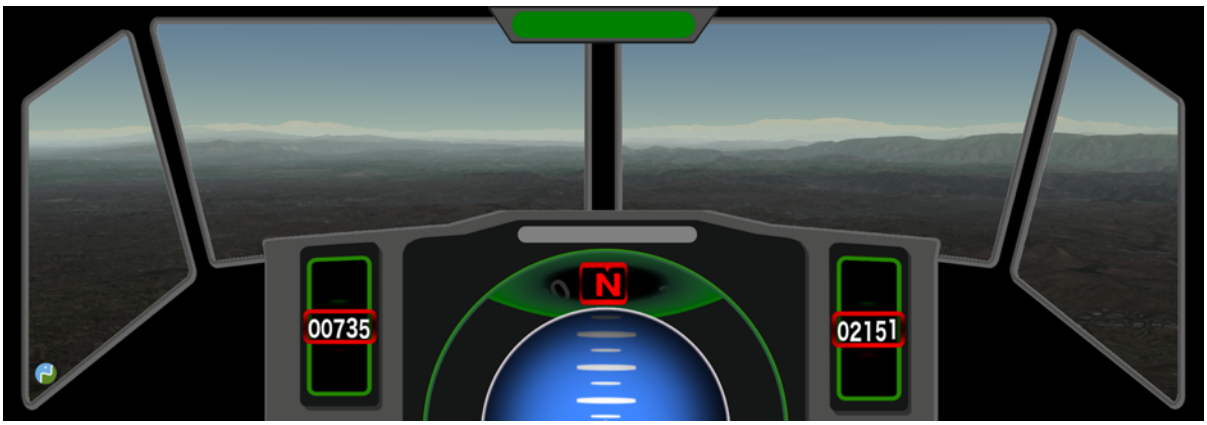
Hlavním elementem celého panelu je umělý horizont `globe_sector`. Tento element se skládá z několika filtrujících elementů a několika elementů, zobrazujících jednotlivé části jako jsou horizontální linky, značka letadla a pozadí určující klopení letadla.



Obrázek 5.2: Centrální panel celého zobrazení v unikátním elementu „over_panel“.

Elementy „terrain_input“ a „terrain_panel“

Hlavní zobrazovací element (Obrázek 5.3) projektu slouží k simulovanému zobrazení okolní krajiny pomocí knihovny CesiumJS. Toto zobrazení je definováno pomocí elementu `terrain_input` a krycího elementu `terrain_panel`, který slouží jako maska zobrazených informací uživateli a určuje tvar výhledu z letadla.



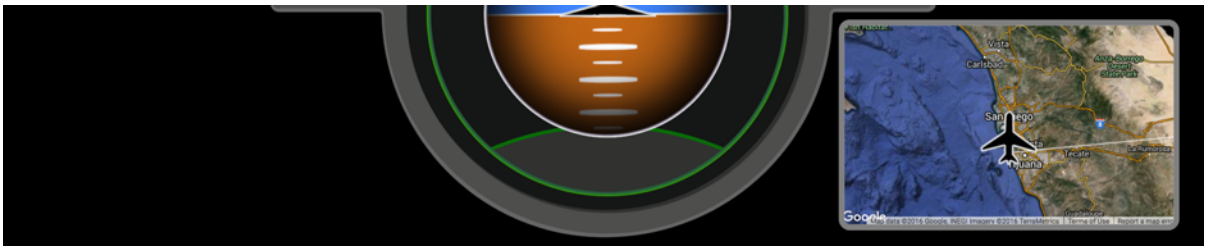
Obrázek 5.3: Výhled z kabiny letadla díky elementům „terrain_input“ a „terrain_panel“.

Element „top_panel“

Element vyhraňující nevyužitou část v horní části obrazovky, která je nutná k správnému umístění ostatních navazujících elementů. Jedná se pouze tedy o výplňovou část zobrazení.

Element „bottom_panel“

Element spodní části obrazovky (Obrázek 5.4) pro zaručení korektního zobrazení na všech zařízeních s umístěnou mapou, pokud si uživatel zvolí její zobrazení.



Obrázek 5.4: Spodní část zobrazení ukrývající mapu zapouzdřená v elementu „bottom_panel“.

Vybrané funkce ovládající zobrazení

V této kapitole jsou popsány některé funkce, které řídí chování jednotlivých elementů zobrazení a bez kterých by nebylo možno celý projekt dostat do funkční rozpohybované podoby. Funkce jsou definovány v javascriptovém souboru maps.js. Proměnné, které musí mít globální dostupnost a dopad, jsou definovány v datovém prostoru `window`, který zaručuje dostupnost těchto hodnot v rámci celého zobrazeného okna. Pro základní funkčnost je volána funkce `reload()` v neustálých intervalech.

Funkce „InitView()“

Jak již název napovídá, jedná se o funkci, která inicializuje aplikační prostředí pro hladký běh celé aplikace a definuje všechny potřebné informace.

V prvním kroku je zavolána funkce `loadData()`, která načte vstupní hodnoty všech nutných veličin pro zobrazení.

Jako druhý krok je vytvořen nový objekt `Cesium.CesiumWidget` deklarovaný pro element stránky s unikátním názvem `cesiumContainer` s přiřazením do globální proměnné `window.widget`. Tomuto objektu je nutno nastavit pomocí atributu `scene3DOnly` zobrazování pouze trojrozměrných objektů a také do atributu `terrainProvider` přiřadit nový objekt `CesiumTerrainProvider` čerpající hodnoty z webové zdroje.

Pro zaručení čitelného a vypovídajícího zobrazení bylo nutno experimentovat s nastavením efektu mlhy, který je definován v scéně námi vytvořeného Cesium widgetu. Při nesprávném nastavení těchto atributů je buď celý horizont zakryt mlhou, nebo je potřebný výpočetní výkon tak vysoký, že není možné zaručit bezproblémový chod na mobilních zařízeních. Přidání mlhy do Cesium knihovny je novinkou posledního půl roku, pro zajištění snížení hardwarové náročnosti aplikací používajících tuto knihovnu v řádu několika desítek procent. Scéna musí mít nastaven atribut `fxaa` pro zaručení vyhlazení zobrazení horizontu. Bez tohoto nastavení je zobrazení nevzhledné a kazí celkový dojem aplikace.

Po tomto základním nastavení máme připraveno korektně nastavené prostředí widgetu CesiumJS. Poté už stačí pouze zavolat funkci knihovny CesiumJS `camera.flyTo()` s výchozími hodnotami zobrazení. Tyto informace jsou převáděny z klasickým informací o zeměpisné šířce a

délce, náklonu, klopení a směru letu ve stupních. Informace o aktuální výšce je v metrických jednotkách.

Po přidání klasické mapy do projektu je nutno při inicializaci vytvořit nový objekt `google.maps.Map` a nastavit zobrazení na aktuální pozici na mapě. Je nutno také zamezit možnosti interagovat s mapou pomocí tahu myší a zamezit zobrazení ovládacích prvků předdefinovaného uživatelského zobrazení.

Na konci úspěšné inicializace jsou zavolány funkce pro aktualizaci dat na panelech pomocí funkce `changePanels()` a mapy pomocí `changeMap()`.

Funkce „changeView()“

Jedná se o funkci, která bere hodnoty z globální proměnné typu slovník `in_data`, ve které jsou uloženy aktuální hodnoty k zobrazení. Funkce využívá funkci knihovny CesiumJS `camera.flyTo()` s aktuálními hodnotami. Při volání této funkce je nutno nastavit délku animace, jež je definována v slovníku pod záznamem „`duration`“ v sekundách, maximální výšku animace, jež je rovna hodnotě maximální hodnotě výšky během letu z výchozího do cílového bodu. Ve výchozím nastavení animace `flyTo()` je použita kvadratická funkce a tudíž by let byl cukaný a nerealistický. Pro zajištění simulace opravdového letu je použit lineární mód animace, tudíž je rychlost pohybu kamery po dobu animace konstantní.

Funkce „changeMap()“

Funkce má za úkol ovlivňovat zobrazení na klasické mapě aktuálními hodnotami. Samotná pozice mapy je editována voláním funkcí `panTo()`, která umisťuje střed mapy na pozici zeměpisné šířky a délky přijatých v parametrech. Dále je uprostřed mapy zobrazena ikona letadla, zobrazující aktuální polohu a směr letu. Pomocí funkce `document.getElementById()` je vybrán element ikony. Tomuto prvku je upraveno stylování v podobě otočení kolem jeho osy (směr letu) a doba této animace.

Funkce „changePanels()“

Základní funkcí této funkce je rozpohybovat dané elementy stránky na základě aktuálních hodnot veličin. Všechny editace jsou prováděny editací stylových atributů.

Pro náklon umělého horizontu a otáčení kompasu je použit atribut `transform` a jeho základní funkce `rotate` s přijímanou hodnotou v stupních. Všechny hodnoty jsou reprezentovány v jednom textovém řetězci (funkce, hodnota, jednotka).

Další využitou funkcí, umožňující pohyb elementů, je úprava `backgroundPosition` atributu. Díky této funkci je možno pohybovat s pozadím daných elementů a tudíž i simulací posunů. Tato funkce je využita v případě pohybu umělým horizontem a pohybu indikátorů pozitivních a negativních změn rychlosti a výšky. Jako poslední krok funkce volá funkce `changeHeight()` a `changeSpeed()` pro aktualizaci zobrazení těchto hodnot.

Funkce „hideMap()“ a „hideLic()“

Tyto funkce ovlivňují celkové zobrazení daných elementů, zobrazujících mapu a licenční známky produktů použitých při implementaci. Funkce jsou volány při kliknutí na dané elementy a upravují stylové atributy daných elementů.

Funkce „changeSpeed()“ a changeHeight()“

Obě funkce jsou funkčně i implementačně identické s ovlivňováním jiných elementů na základě odlišných vstupů. V prvním kroku je uložena aktuální hodnota veličiny do lokální proměnné `value` a je vypočítán rozdíl `diff` od minulé hodnoty. Z tohoto kroku je určena proměnná ab, která indikuje, zdali se jedná o kladnou či zápornou změnu.

Do proměnné x jsou načteny všechny elementy (jednotlivá desetinná místa) zobrazující dohromady aktuální hodnotu. Pomocí cyklu, ovlivňujícím tyto elementy, jsou postupně kráceny jednotlivé bloky a posunovány dané řády na aktuální hodnotu. Díky této implementaci je zaručeno několikanásobné otočení nízkých řádů při velkých změnách.

Funkce „loadData()“

Funkce zpracovávající přijaté informace a jejich uložení do globálního slovníku `in_data`. V aktuálním řešení jsou v této funkci uloženy aktuální hodnoty výšky a rychlosti a poté načteny nové hodnoty z odpovědi serveru.

Implementace aplikace na straně serveru

Pro běh aplikace musí být funkční i část na straně serveru. Aplikaci je nutno zaručit dostatek prostředků pro hladký běh. Současná aplikace běží na VPS serveru s adresou 104.236.248.102. Jedná se o stroj pronajímaný od společnosti DigitalOcean řady NYC3 s 1GB RAM, 20GB SSD diskem a 4 jádrovým procesorem. Na jednotce běží operační Unix systém Ubuntu 14.04 x64.

Implementace získávání dat

V složce backend se nachází tři soubory flight_info.py, proces_data.py a telnet_connection.py. Aktuálně je využíván pouze soubor flight_info.py. V tomto souboru se definována kompletní funkčnost pro získávání dat. API je vybudováno jako Flask aplikace, běžící na portu 51000. Při prvním spuštění jsou pevně nastaveny všechny vstupní hodnoty.

Obnova dat

Ve funkci `update_values()` jsou dané hodnoty upraveny požadovaným způsobem pomocí funkce `random()`. Tímto způsobem je zaručena základní simulace letu. Toto funkce je rekurzivně volána s časovačem nastaveným na 1 sekundu. Toto řešení zaručuje nerušenou obnovu dat bez zásahu poslouchajících zařízení. Každých 10 minut je generování trasy vynulováno na původní hodnoty pomocí funkce `reset()`.

Flask aplikace má poté definovanou routu `getdata` pro dotaz klienta na aktuální hodnoty v slovníku dat a routu `resetme` pro okamžitý reset na výchozí hodnoty.

Instalace a spuštění aplikace

Projekt je možno nainstalovat pomocí vytvoření klonu repositáře za pomocí příkazu `git clone https://github.com/valfa14/flight-info.git`. Poté je nutno zkopírovat obsah zip souboru CesiumJS staženého ze http://cesiumjs.org/downloads.html do složky cesium v kořenovém adresáři projektu.

V kořenovém adresáři projektu pak pouze stačí spustit skript bez přerušení pomocí příkazů `nohup python backend/flight_info.py` a spustit webový server pomocí `nohup python -m SimpleHTTPServer 8000`. Pro tyto účely jsem vytvořil nového uživatele `www_user`, který má práva pouze pro čtení obsahu daného projektu a nehrozí žádné bezpečnostní riziko. A vše je hotovo a mělo by běžet na adrese `http://host:8000`.

V případě chtěného resetu aktuálního zobrazení letu do výchozí pozice stačí zaslat dotaz na adresu `http://host:51000/resetme`, který vrátí odpověď o úspěšnosti resetu

6 Testování aplikace

V této kapitole je vysvětleno, jak probíhaly přípravy, samotné testování a výsledky. Hlavním cílem projektu je zaručit dostupnost zobrazení co největšímu počtu pasažérů. Kvůli tomuto aspektu bylo nutno sehnat pokud možno co nejvíc zařízení, která by měla být schopna tato data zobrazit. Nakonec bylo zvoleno k testování několik zařízení různých typů.

Obrázek 6.1: Běžící aplikace na notebooku Apple MacBook Pro 15 Retina 2015.

Počátky vývoje aplikace probíhaly na zařízení Asus Zenbook X303 (Windows 8.1). Tento počítač neměl s během aplikace problém, ale zklamalo hardwarové zpracování a notebook musel do opravy. Na finální testování již byl zpět a potvrdil běh aplikace i v rozlišení QHD.

Největší část implementace byla prováděna na stroji Apple MacBook Pro 13 Retina 2015 (OS X) bez sebemenších problémů. Jako konečné testovací zařízení byl použit notebook Apple MacBook Pro 15 Retina 2015 s externí grafikou (Obrázek 6.1). Tento stroj potvrdil funkčnost celé aplikace na všech různých rozlišeních včetně běhu na 4K monitoru v plném rozlišení bez znatelných problémů.

Obrázek 6.2: Běžící aplikace na mobilním telefonu Sony Xperia Z s operačním systémem Android.

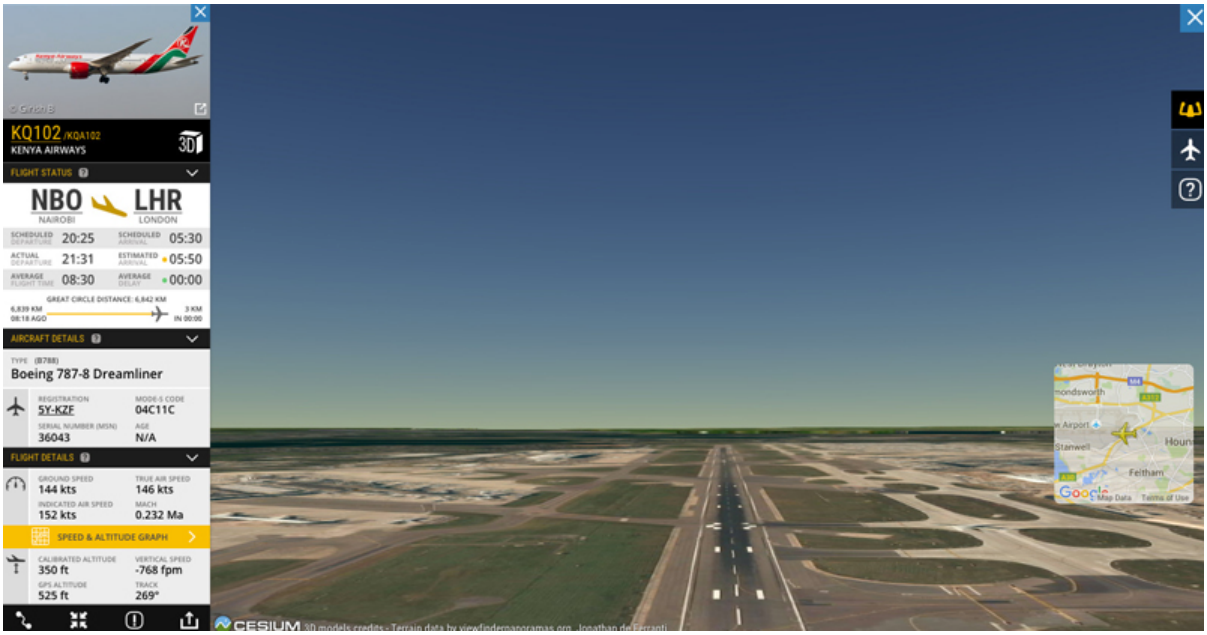
Obrázek 6.3: Běžící aplikace na mobilním telefonu Apple iPhone 5s.

Testování na mobilních zařízeních probíhalo v poslední fázi implementace po spuštění na serveru s úspěšným výsledkem. Aplikace byla testována jak na operačním systému Android (Obrázek 6.2), tak i na operačním systému iOS 9 na telefonu značky Apple (Obrázek 6.3).

Veškeré testování ukázalo výhody použití designu s relativním formátováním a vyplatilo se i využití základních transformačních a editačních funkcí, které jsou podporovány ve všech zařízeních a platformách.

7 Budoucí trendy zobrazení letových informací

V posledním měsíci mé práce se objevila na internetu informace o spolupráci serveru flightradar.com a knihovny CesiumJS. K této spolupráci došlo a výsledkem je 3D zobrazení aktuálních informací o letu. Lze tedy v blízké době očekávat velký rozmach tohoto druhu zobrazení, které má potenciál ve velké oblibě.



Obrázek 7.1: Zobrazení výhledu z pohledu pilota v budoucí verzi flightradar24.com.

Obrázek 7.2: Pohled na letící letadlo z externího pohledu.

8 Závěr

Projekt byl zaměřen na zpřístupnění aktuálních letových informací cestujícím na palubě letadla bez nutnosti vysokých investic do nákupu integrovaných zobrazovacích zařízení letadla. Díky využití aktuálně dostupných technologií byla vytvořena webová aplikace, která je schopna tyto informace zobrazit všem cestujícím na jejich elektronických zařízeních bez nutnosti dalších investic. Uživateli jsou zobrazeny informace o aktuálním pohledu z kabiny letadla pomocí virtuálního pohledu a realistické simulace okolního prostředí včetně horizontu, stínu a reliéfů. Hodnoty o náklonu a naklopení je také možno vyčíst z umělého horizontu, který je hlavním prvkem centrální části zobrazení včetně aktuálních hodnot výšky, rychlosti a směru letu.

Pokud má uživatel zájem, může si kliknutím na interaktivní tlačítko aktivovat zobrazení klasické mapy v pravém horním rohu, která uživateli poskytne informaci o aktuální poloze a směru letu na mapě, která zobrazuje letecké záběry na území pod letadlem včetně popisů jednotlivých měst a zvýraznění dopravních komunikací.

Aplikace dokáže nízkonákladovým společnostem nabídnout alternativní službu k řešením, která používají velké korporátní společnosti s mnohonásobně nižšími provozními náklady. Další velkou výhodou je centrální běh serverové části, která nabízí jednoduchou možnost oprav či vylepšení služby bez dalších nákladů.

Literatura

[1] SHARKIE, Craig a Andrew FISHER. *Responzivní webdesign: okamžitě*. Brno: Computer Press, 2015. ISBN 978-80-251-4384-1.

[2] CASTRO, Elizabeth a Bruce HYSLOP. *HTML5 a CSS3: názorný průvodce tvorbou WWW stránek*. Brno: Computer Press, 2012. ISBN 978-80-251-3733-8.

[3] LAZARIS, Louis. *CSS okamžitě*. Brno: Computer Press, 2014. ISBN 978-80-251-4176-2.

[4] UTZ, Mark. *Programming Python*. 4th ed. Beijing: O'Reilly Media, 2011. programming/Python. ISBN 978-0-596-15810-1.

[5] *Redis: the Definitive Guide Data Modeling, Caching, and Messaging*. Oreilly & Associates Inc, 2011. ISBN 1449396097.

[6] Cesium documentation. *CesiumJS.* [online]. 1.5.2016 [cit. 2016-05-01]. Dostupné z: *https://cesiumjs.org/Cesium/Build/Documentation/index.html*

[7] Google Maps documentation. *Google Maps.* [online]. 1.5.2016 [cit. 2016-05-01]. Dostupné z: *https://developers.google.com/maps/documentation/javascript/*

Seznam obrázků

Seznam tabulek

Seznam příloh

Příloha 1. CD