



Střední průmyslová škola strojní  
a elektrotechnická a Vyšší odborná škola,  
Liberec 1, Masarykova 3

# VÝVOJ APLIKACE PRO CHYTRÉ HODINKY

Maturitní práce

Autor

**Vít Zeman**

Obor

**Informační technologie**

Vedoucí práce

**Ing. Marek Pospíchal**

Školní rok

**2022/2023**



SPŠSE A VOŠ  
LIBEREC

## Přihláška k maturitní práci

### Jméno a příjmení studenta

Zeman, Vít

### Název práce

Vývoj aplikace pro chytré hodinky

### Přidělené role

Vedoucí práce

### Třída

P4

### Školní rok

MP2022/23

### Oponent

Podpis

Pospíchal, Marek

Obecná ustanovení	Vypracování a odevzdání práce proběhne v souladu s platnými normami (vyhláška 177/2009 Sb.) a aktuálním dokumentem "Pokyny k vypracování prací" vydaným školou.
	Práce bude hodnocena z hlediska jejího praktického využití, zvládnutí dokumentace po věcné i formální stránce a obhajoby celé práce. Student byl seznámen s kritérii hodnocení maturitní práce.
	Práce bude odevzdána ve dvou stejnopisech vázaných pevnou nebo kroužkovou vazbou.
	Veškeré náklady na MP včetně vyhotovení obou tištěných kopií si student hradí sám.
Licenční ujednání	Ve smyslu § 60 (Školní dílo) autorského zákona č. 121/2000 Sb. poskytuji SPŠSE a VOŠ Liberec výhradní a neomezená práva k využití této mé maturitní práce.
	Bez svolení školy se zdržím jakéhokoliv komerčního využití mé práce.
	Pro výukové účely a prezentaci školy se vzdávám nároku na odměnu za užití díla.

### Finanční rozvaha - odhad celkových nákladů

V Kč	Náklady celkem	Hrazené školou
Výrobní	0	0
Na služby	0	0

Jedná se o MP, jejíž vypracování si škola vyžádala? ~~Ano~~ - Ne

### Podpis studenta (vyjadřuje souhlas s uvedenými údaji a ujednáními)

V Liberci 14.10.2022

### Konzultant

Práci podporuji

### Předmětová komise

Práci doporučuji

### Třídní učitel

Práci doporučuji

### Garant oboru

Práci doporučuji

### Ředitel školy

Práci doporučuji

Podpis

Podpis

Podpis

Podpis

Podpis

Podpis

## Zadání maturitní práce

**Název**

Vývoj aplikace pro chytré hodinky

**Předmět**

PRG, WEB

**Téma**

Práce se zabývá postupem vytvoření aplikace pro chytré hodinky Samsung Gear S3 Frontier s cílem získání dat ze senzorů.

**Použité prostředky**

Tizen Studio, Tizen Emulator, chytré hodinky (Gear S3), HTML, CSS, JS

**Cíle práce**

1	Vytvoření aplikace přistupující k datům ze senzorů chytrých hodinek
2	Realizace zpřístupnění dat ze senzorů vzdáleně
3	Nasazení a testování aplikace
4	Porovnání možností vývoje v C/C# a html/css/JS

**Osnova práce**

1	Nastudovat tvorbu aplikace pro OS Tizen
2	Nastudovat jak nahrát aplikaci do chytrých hodinek
3	Zjistit jak přistupovat k datům ze senzorů chytrých hodinek
4	Navrhnout způsob zpřístupnění dat ze senzorů vzdáleně

## Anotace

Práce se zabývá vytvořením aplikace pro chytré hodinky, která umožňuje přístup k senzorům ne jenom lokálně, ale i vzdáleně.

## Summary

The thesis deals with the creation of a smartwatch application that allows access to sensors not only locally, but also remotely.

## Čestné prohlášení

Prohlašuji, že jsem předkládanou maturitní práci vypracoval sám a uvedl jsem veškerou použitou literaturu a bibliografické citace.

V Liberci dne 15.03.2023

.....

Vít Zeman

# Obsah

Úvod.....	1
1    Vývojové prostředí Tizen studio.....	2
1.1    Emulátor .....	2
1.2    Operační systém Tizen.....	3
1.3    Správce Certifikátů .....	3
1.4    Správce zařízení .....	4
1.5    Správce balíčků.....	5
1.6    Tizen Advanced UI .....	5
2    Možnosti vývoje .....	7
2.1    Web Applications .....	7
2.1.1    Správce balíčků aplikací.....	7
2.1.2    Tizen Web API.....	8
2.2    Native Applications .....	8
2.3    Aplikace .NET .....	9
2.3.1    .NET Core.....	9
2.3.2    Xamarin.Forms .....	9
2.3.3    TizenFX API.....	9
3    Chytré hodinky .....	10
3.1    Chytré hodinky a zdraví.....	10
3.1.1    Fitness náramky.....	10
3.2    Samsung Gear S3 .....	11
3.2.1    Specifikace hodinek.....	11
3.2.2    Senzory hodinek .....	11
4    API .....	13
4.1    WEB API.....	13
4.2    REST API.....	13
4.2.1    CRUD.....	13
4.2.2    Stavové kódy.....	13

5	Příprava prostředí k vývoj.....	15
5.1	Instalace balíčků v Package Mangeru .....	15
5.2	Vytvoření profilů certifikátů .....	15
6	Vývoj Aplikace.....	16
6.1	Měření srdečního tepu.....	16
6.1.1	Human Activity Monitor .....	16
6.1.2	Průběh měření .....	17
6.1.3	Ukončení měření.....	19
6.2	Měření atmosférického tlaku.....	20
6.3	Měření úrovně světla.....	21
6.4	Data .....	21
6.5	API .....	23
6.5.1	Model.....	23
6.5.2	ApplicationDbContext.....	24
6.5.3	Vytvoření databáze.....	24
6.5.4	Kontrolér .....	25
6.6	Config .....	26
6.7	HTML dokument.....	26
7	Testování a nasazení aplikace.....	27
7.1	Měření tepu .....	27
7.2	Měření atmosférického tlaku.....	29
7.3	Měření úrovně světla.....	31
7.4	Data .....	32
7.5	Nasazení aplikace do hodinek.....	34
8	Problémy při vývoji.....	35
8.1	Emulátor .....	35
	Závěr .....	36
	Seznam zkratk a odborných výrazů.....	37
	Seznam obrázků.....	39
	Použitá literatura.....	41

A.	Seznam přiložených souborů .....	I
----	----------------------------------	---

## Úvod

Osobně mě motivoval k vytvoření této aplikace fakt, že ještě nikdo stejné téma maturitní práce na této škole nevypracoval. Také samotná myšlenka vytvoření aplikace pro chytré hodinky mě velmi zaujala a myslím si, že stojí za vyzkoušení a následnou realizaci. Aplikace by se dala využít také pro propojení hodinek s tréninkovým deníkem, ale to je pouze moje vize, u které si nejsem jistý, zda je realizovatelná.

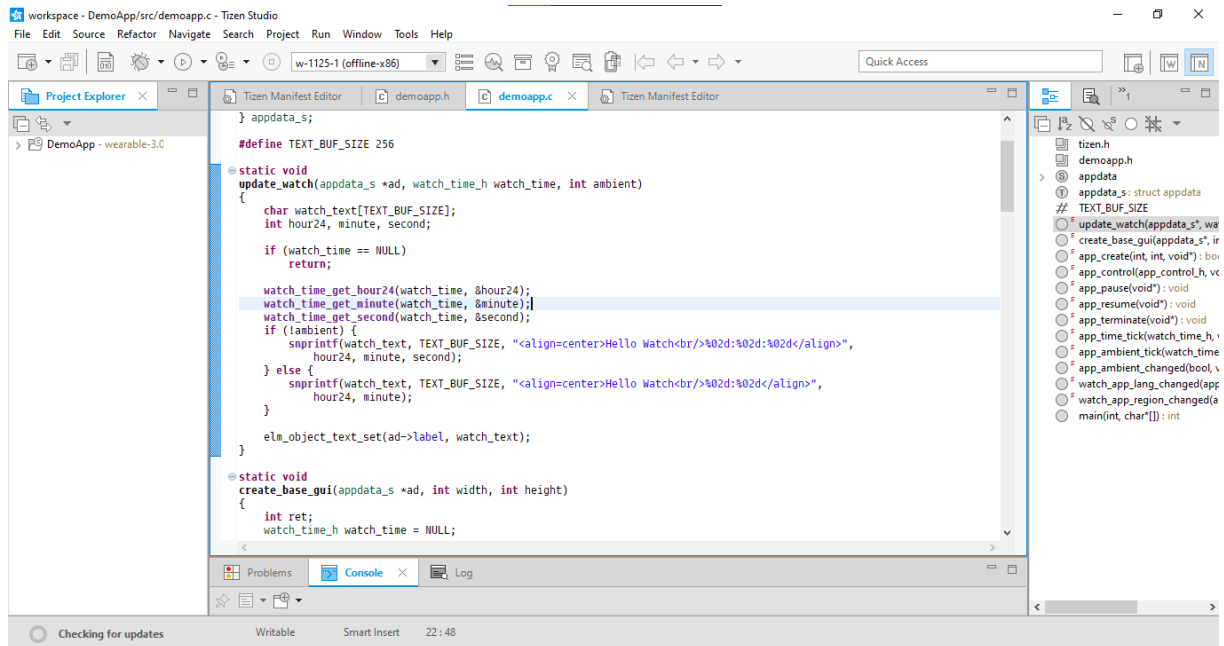
Tato práce je především zaměřena na výzkum způsobu a náročnosti vytvoření vlastní aplikace pro chytré hodinky, a to přesněji v aplikaci Tizen Studio. Hlavním cílem práce je vytvoření aplikace na chytré hodinky Samsung Galaxy Gear S3, která bude přistupovat k datům ze svých senzorů a následná realizace zpřístupnění dat ze senzorů vzdáleně.

Osobní zkušenosti s vývojem takového typu aplikace nemám, ale s některými použitými technologiemi jsem se už v praxi setkal. Například s jazykem C, nebo HTML, CCS a JS jsem se již seznámil v hodinách mikroprocesorové techniky, nebo webových aplikací.



# 1 Vývojové prostředí Tizen studio

Jedná se o oficiální vývojové prostředí pro vývoj webových a nativních aplikací pro zařízení, které disponují operačním systémem Tizen. Jeho součástí je také emulátor, který velmi usnadní práci při testování aplikace.



Obrázek 1 Vývojové prostředí Tizen Studio

## 1.1 Emulátor

Emulátor v Tizen studiu nabízí možnost testování aplikace bez potřeby skutečného zařízení. Poskytuje také hardware podobný skutečnému zařízení, a to včetně ovládacích prvků. Je tedy velmi užitečný ke zkontrolování bezchybného běhu aplikace v reálném nasazení. Při emulaci nabízí také CPU, RAM a periferní zařízení.



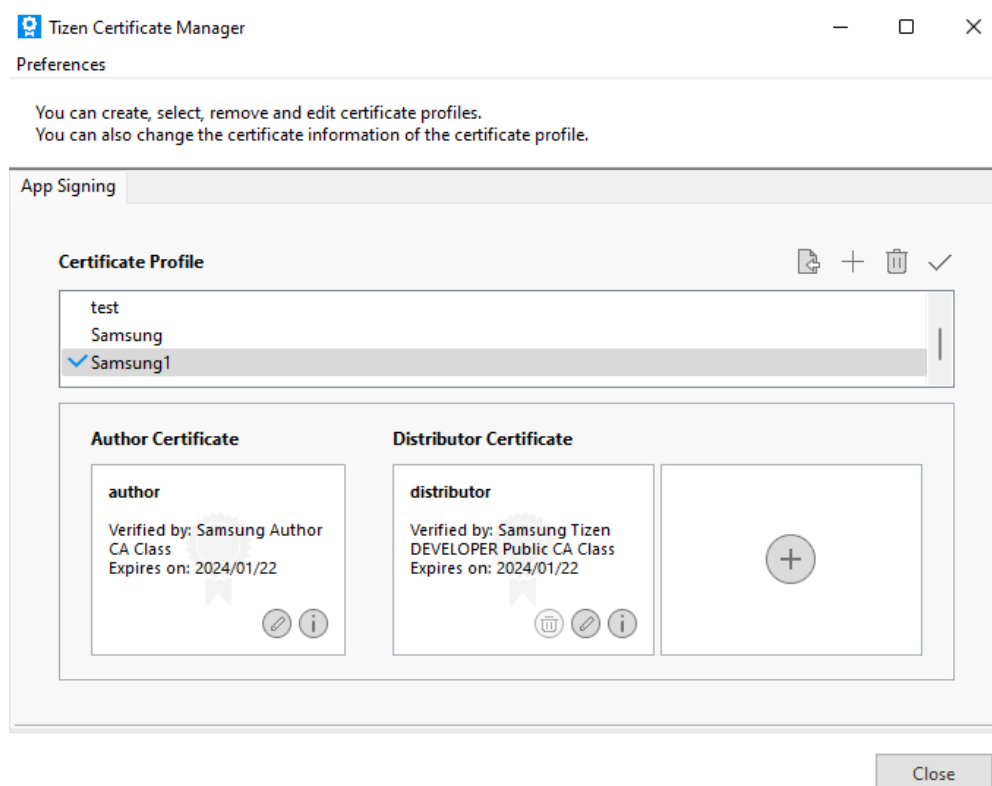
Obrázek 2 Emulátor

## 1.2 Operační systém Tizen

Tizen je open source operační systém založený na Linuxu. Byl vyvinut s myšlenkou propojení velké škály zařízení od chytrých televizorů, telefonů, hodinek a zařízení IoT. Tizen podporuje programovací jazyky jako jsou C, C++, JavaScript a C#. Nabízí podporu pro senzory, hardware a umožňuje tak vývojářům vytvářet aplikace s velmi pokročilými funkcemi. V současnosti není Tizen tak populární a rozšířený jako jiné operační systémy. Například Android nebo iOS. Samsung a ostatní výrobci často tento operační systém využívají pro chytré televize nebo hodinky.

## 1.3 Správce Certifikátů

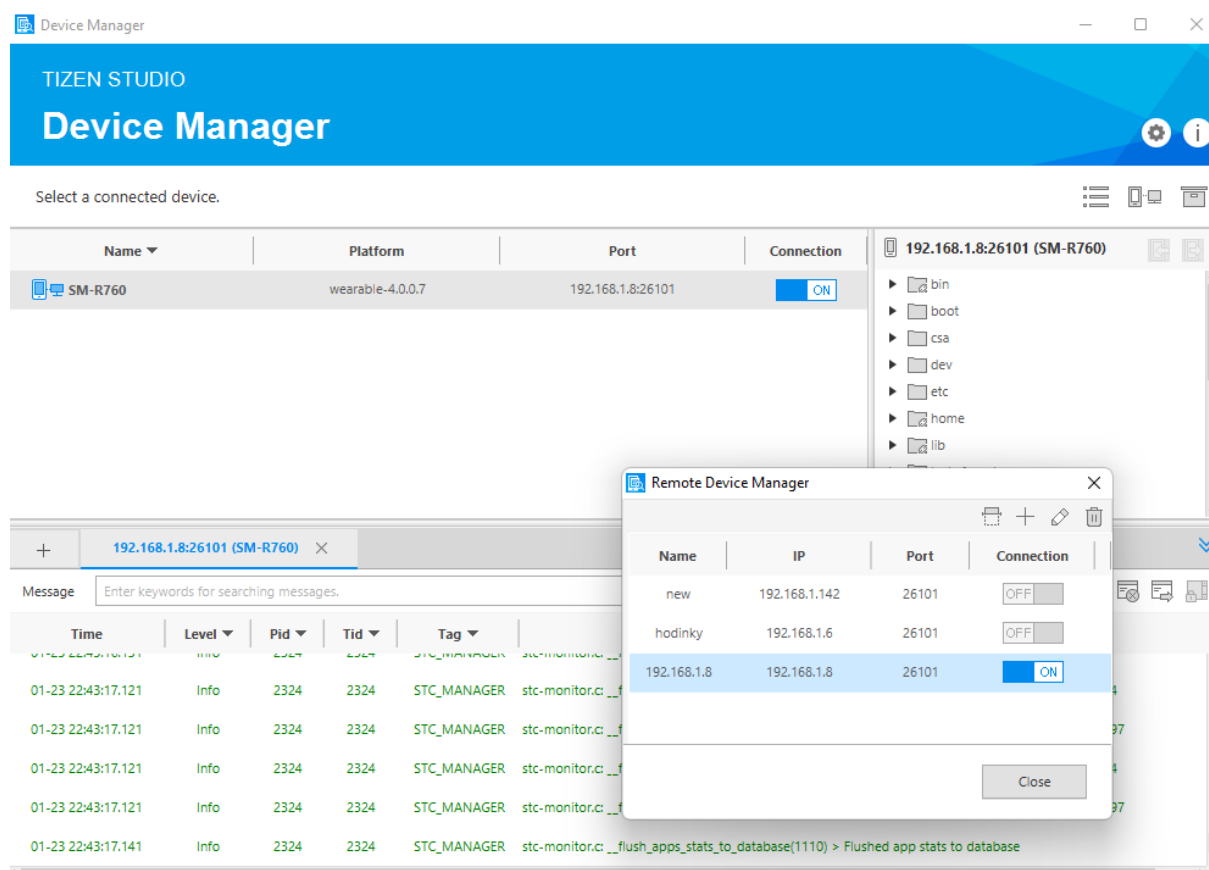
Správce certifikátů umožňuje vytvoření profilu, který je nutný před instalací aplikace do hodinek nahrát. Tento profil musí být podepsán a slouží k ověření zdroje a autora aplikace. Profil certifikátů se skládá z podpisových certifikátů.



Obrázek 3 Správce certifikátů

## 1.4 Správce zařízení

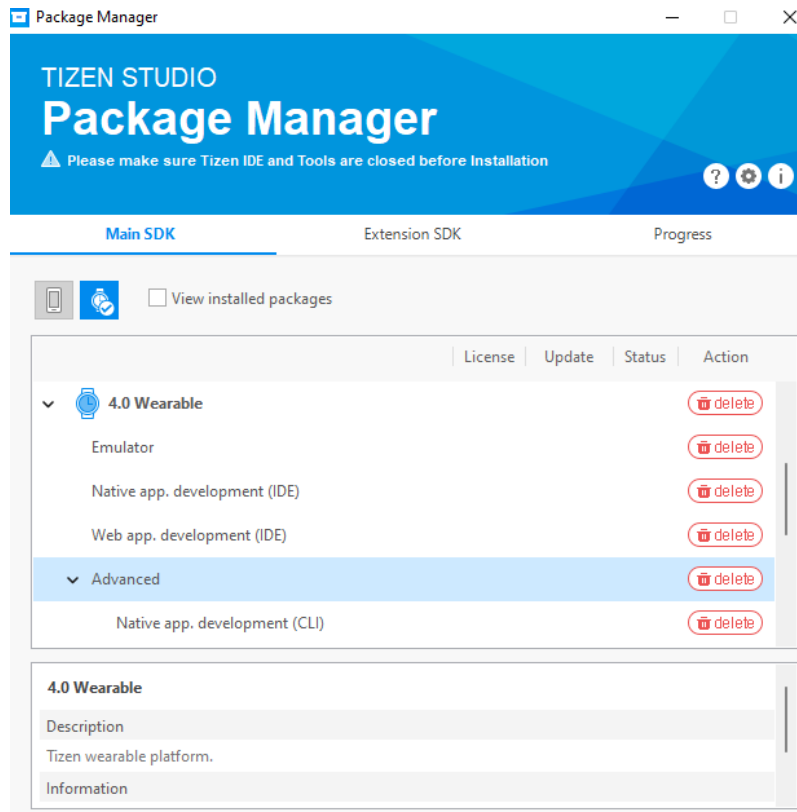
Správce zařízení je nástroj, který zobrazuje informace o připojených Tizen zařízeních, nebo emulátorech. Používá se v rámci ladění aplikace. S pomocí správce zařízení je možné spravovat připojená zařízení nebo emulátory. Díky tomuto je možné instalovat aplikace do zařízení, nebo stahovat obsah. Při použití vzdáleného správce zařízení se nabízí možnost připojit například chytré hodinky pomocí Wi-Fi.



Obrázek 4 Správce zařízení

## 1.5 Správce balíčků

Správce balíčků se používá k instalaci požadovaných softwarových součástí, které jsou potřeba k vývoji aplikace a k následnému získání podrobných informací o nainstalovaných balíčcích.



Obrázek 5 Správce balíčků

## 1.6 Tizen Advanced UI

Tizen Advanced UI Framework umožňuje vytvářet a spravovat různé druhy komponent uživatelského rozhraní. TAU je standardní knihovna rozhraní pro platformu Tizen. Klíčovou vlastností je zjednodušení kódování a rychlost vytváření aplikací.

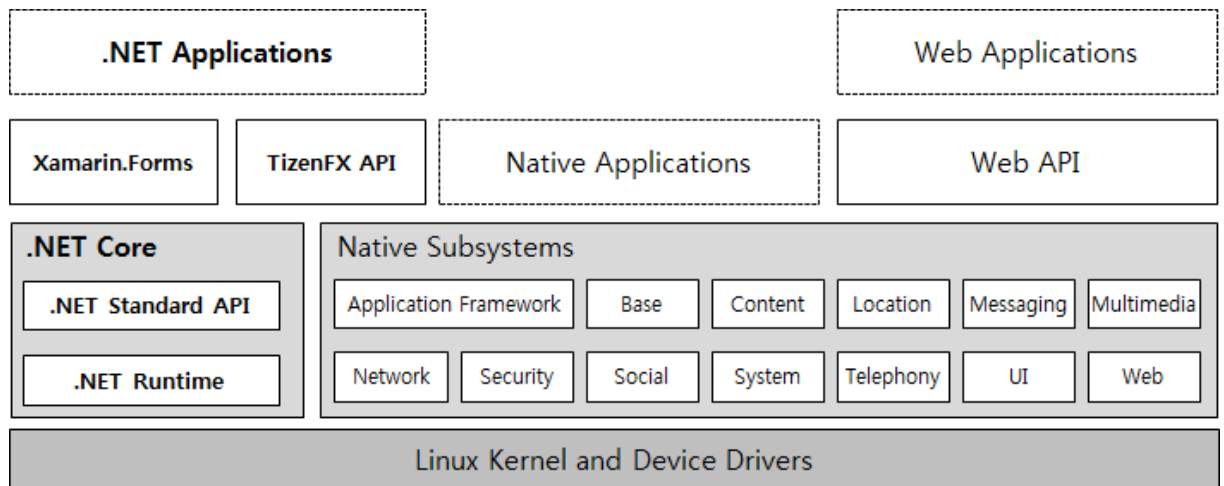
Používání TAU přináší následující výhody:

- TAU je samostatná knihovna.
- Lze používat s jQuery, jelikož zpřístupňuje speciální API.
- Veškerý kód je vyladěn pro maximální výkon.
- TAU je optimalizováno pro nositelná, mobilní a televizní zařízení.

- TAU lze přizpůsobit a snadno rozšířit (vytvořit nové komponenty uživatelského rozhraní)

## 2 Možnosti vývoje

Při vývoji je možný výběr ze tří typů aplikací, přičemž každý z nich nabízí trochu odlišný způsob vývoje. Také je možné vytvářet hybridní aplikace. V tomto případě se zabalí dohromady webová a nativní aplikace za účelem vytvoření výkonnějšího programu. Zvolení vhodného způsobu vývoje závisí hlavně na tom, jaké funkce budou potřeba a také na preferencích programátora.



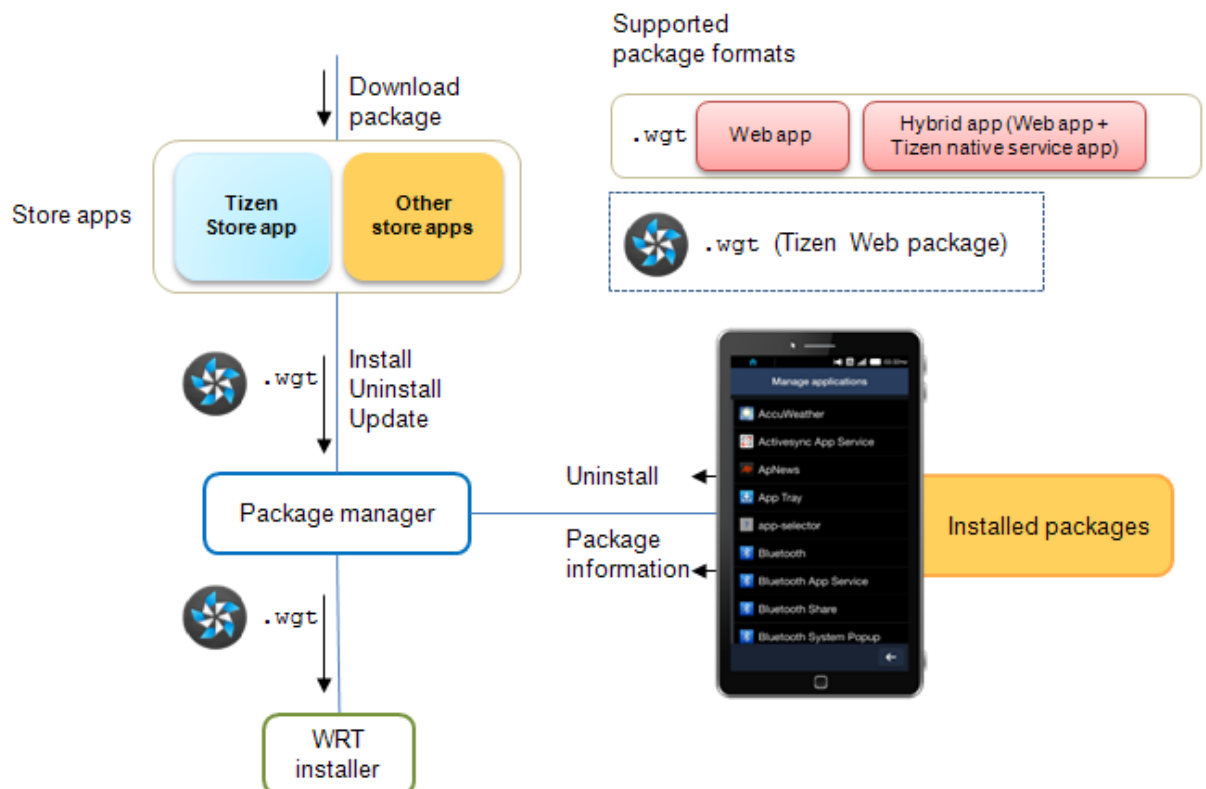
Obrázek 6 Architektura Tizen [6]

### 2.1 Web Applications

Používá se pro vytváření webových aplikací pro mobilní, nositelná a televizní zařízení. Spojuje HTML, JavaScript a CSS do balíčků, které je možné následně nainstalovat do zařízení. V mém případě se bude jednat o chytré hodinky. Vytvořený balíček obsahuje všechny soubory potřebné pro samostatné fungování programu. Po nainstalování nepotřebuje už žádné externí zdroje nebo internetové připojení. Za instalaci, odstranění a aktualizaci je zodpovědný Správce balíčků aplikací.

#### 2.1.1 Správce balíčků aplikací

Správce balíčků aplikací řídí instalaci, odstranění a aktualizaci balíčků. Využívá se také pro získávání informací o balíčcích, které jsou nainstalované v zařízení.



Obrázek 7 Správce balíčků aplikací [7]

### 2.1.2 Tizen Web API

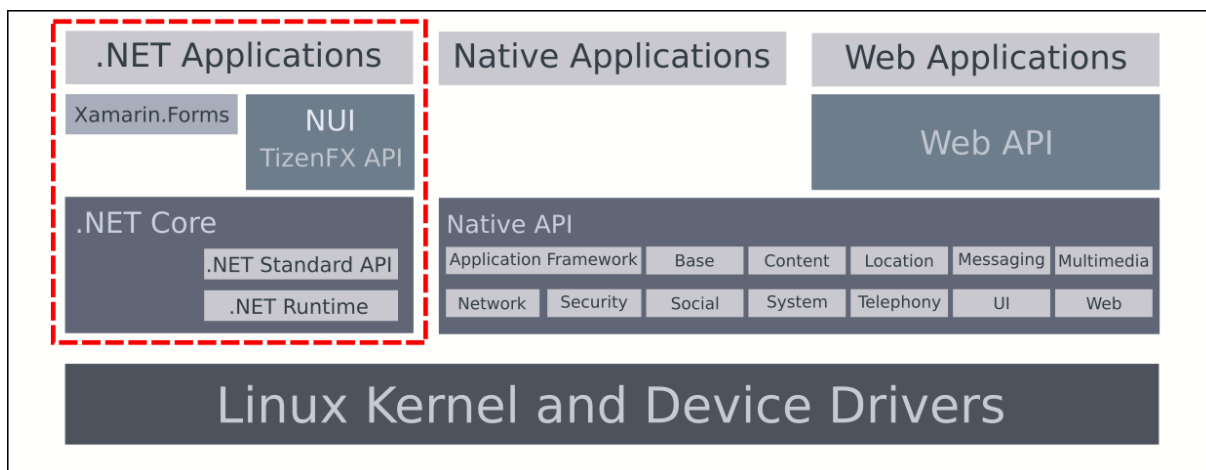
Tizen Web API je nástroj, pomocí kterého je možné při tvorbě webové aplikace přistupovat k funkcím na zařízení. Díky tomu je možné stále pracovat například se senzory zařízení a zobrazovat jejich hodnoty v reálném čase.

## 2.2 Native Applications

Při vývoji nativní aplikace se používá programovací jazyk C. Díky přímému přístupu k funkcím zařízení je možné vytvářet programy s pokročilejšími mechanismy. Nativní aplikace podporují grafické uživatelské rozhraní, ale také servisní aplikace, které nemají grafické rozhraní a běží takzvaně na pozadí systému. Aplikace s uživatelským rozhraním a servisní aplikace se dají spojit do jednoho balíčku a lze vytvořit kombinovaný balíček, který bude obsahovat jednu aplikaci uživatelského rozhraní a více servisních aplikací.

## 2.3 Aplikace .NET

Tento způsob je poměrně nový a umožňuje vývoj pomocí programovacího jazyku C#. Přináší jednoduchý způsob vývoje pro zařízení s operačním systémem Tizen, běžící na více jak 50 milionech zařízení. NUI poskytuje možnost jednoduše vytvářet uživatelské prostředí a TizenFX API poskytuje přístup k funkcím zařízení.



Obrázek 8 Architektura Tizen .NET [10]

### 2.3.1 .NET Core

Tuto univerzální open source vývojovou platformu spravuje společnost Microsoft. Jedná se o multiplatformní nástroj pro vytváření programů v jazyku C#, F# nebo Visual Basic. Umožňuje vývoj aplikací, které budou podporovány na více operačních systémech, jako jsou Windows, macOS a Linux.

### 2.3.2 Xamarin.Forms

Xamarin Forms je knihovna umožňující vytváření multiplatformních aplikací s použitím jazyka C#. Je podporován napříč spoustou operačních systémů jako jsou iOS, Android, Windows a Tizen. Umožňuje tak vývoj aplikací na různé platformy s použitím jednotného kódu C# a XAML.

### 2.3.3 TizenFX API

TizenFX API umožňuje přistupovat k funkcím zařízení podobně jako v nativní aplikaci, ale bez nutné znalosti jazyka C. Mezi hlavní funkce, které jsou přístupné prostřednictvím TizenFX API jsou senzory, bezdrátová komunikace nebo lokalizace.



## 3 Chytré hodinky

Co jsou vlastně chytré hodinky? Jedná se o zařízení v podobě náramkových hodinek, které ale nabízí o poznání více funkcí než klasické hodinky, které ukazují jenom aktuální čas nebo datum. Zjednodušeně řečeno se jedná o chytrý telefon v podobě hodinek. V dnešní době již tyto zařízení nabízejí opravdu velkou škálu funkcí. Některé modely dokonce nabízejí vestavěný fotoaparát, nebo slot pro SIM kartu. V tomto případě se dá z hodinek i volat. Většinou se ale dají pomocí technologie Bluetooth připojit k chytrému telefonu. V tomto případě jsou hodinky schopné nás upozorňovat na příchozí hovory nebo zprávy. Připojení k wifi je také u spousty modelů samozřejmostí.

### 3.1 Chytré hodinky a zdraví

Chytré hodinky jsou velmi populárním nástrojem pro měření a sledování parametrů, jako jsou tepová frekvence, spánek, kroky a aktivita. Většina těchto funkcí je integrována přímo do zařízení a umožňuje tak podrobně sledovat zdravotní údaje.

Nejběžnější funkcí je sledování kolik kroků uživatel ujde za den. Hodinky po celý den automaticky sledují počet nachozených kroků a umožňuje lépe plánovat cvičení a sledovat dosažených cílů.

Chytré hodinky dokážou také měřit srdeční tep a umožňuje tak uživateli sledovat tyto hodnoty během cvičení nebo chůze. Tato funkce může být užitečná pro lidi, kteří mají problémy se srdcem.

Pro mě nejzajímavější funkce je sledování spánku. Hodinky sledují během celého spánku v jaké fázi spánku se zrovna uživatel nachází, nebo jak dlouho trvalo uživateli usnout. Tato data se následně analyzují a poskytují uživateli zpětnou vazbu ohledně kvality a délky spánku.

Důležité je zmínit, že chytré hodinky nejsou náhradou za lékařské vyšetření a neměli by být používán jako jediný zdroj pro diagnostiku a léčbu.

#### 3.1.1 Fitness náramky

Fitness náramky jsou chytrá zařízení, která podobně jako chytré hodinky měří a sledují zdravotní parametry. Jedním z hlavních rozdílů je design a funkce. Na fitness náramcích jsou většinou funkce zaměřené hlavně na měření zdravotních parametrů a

chybí zde například funkce jako telefonní hovory, textové zprávy, kalendář a další. Tyto náramky jsou často i vzhledem na první pohled rozeznatelné. Na rozdíl od chytrých hodinek jsou menší a jednodušší. Výdrž baterie je tady na tom o něco lépe než u chytrých hodinek, jelikož mají omezené funkce a o poznání menší display. Pořizovací cena se zde pohybuje na nižších částkách, než u chytrých hodinek.

## 3.2 Samsung Gear S3

Chytré hodinky Samsung Gear S3 je model hodinek, na které budu aplikaci vyvíjet. Jedná se o velice robustní a výkonné hodinky, které jsou díky svým funkcím jako stvořené pro aktivní životní styl. Poskytují otočnou lunetu a dvě hardwarová tlačítka pro snadné ovládání a nastavení hodinek. Díky vestavěné GPS jsou vhodné jak pro turistiku, nebo na kolo. Dokáže poskytnout ty nejlepší turistické trasy přímo z vašeho zapěstí. Zařízení také sbírá veškeré údaje o aktivitách a pomocí upozornění se vás snaží udržet po celý den v pohybu. Disponují certifikací IP68, což znamená že dokážou odolat prachu, nečistotě a vodě až do hloubky 1,5 metru po dobu 30 minut.

### 3.2.1 Specifikace hodinek

Hodinky jsou vybaveny bohatou výbavou, která poskytuje vysoký výkon pro potřebné funkce. Přesněji se jedná o:

- Velkost (hlavní displej) => 32.9 mm
- Váha => 63 g
- Kapacita baterie => 380 mAh
- Bluetooth => Bluetooth v4.2
- Operační systém => Tizen
- Interní paměť => 4 GB
- Velikost RAM => 0.75 GB
- Displej => Super AMOLED (360 x 360)
- Procesor => dvoujádrový (1 GHz)

### 3.2.2 Senzory hodinek

Senzory, které jsou vestavěné v hodinách, dokážou poskytovat přesné informace například o srdečním tepu, akceleraci, nebo úrovni světla. Jelikož senzory jsou hlavní věc,

kterou se budu při vývoji aplikace zabývat, dovolím si zde vypsát veškeré senzory, které tyto hodinky nabízejí:

- Akcelerometr
- Barometr
- Gyroskop
- Snímač srdečního tepu
- Světelný senzor



*Obrázek 9 Chytré hodinky*

## 4 API

API je rozhraní, obsahující sadu funkcí, které umožňují programátorům přistupovat k specifickým funkcím a datům aplikace, nebo jiných služeb

### 4.1 WEB API

Webové API je API, ke kterému je možné přistupovat přes WEB, například pomocí protokolu HTTP. Rozhraní webového API lze vytvořit pomocí nejrůznějších technologií, jako je například .NET.

### 4.2 REST API

REST API je architektura, která umožňuje přístup k datům a následné provádění CRUD operací. Lze ho velice snadno použít s protokolem http, který je velice rozšířený. Poskytuje určité standardy, takže použití cizího API není problém.

#### 4.2.1 CRUD

CRUD je označení pro celkově čtyři základní operace, vytvořit, číst, aktualizovat a smazat. V plnohodnotné aplikaci se CRUD skládá ze tří částí: API, databáze a uživatelské rozhraní. Každé písmeno odpovídá konkrétnímu HTTP dotazu:

- Create (vytvoření) – POST
- Read (získání dat) – GET
- Update (úpravy) – PUT nebo PATCH
- Delete (smazání) – DELETE

#### 4.2.2 Stavové kódy

V http nalezneme stavové kódy, v REST API jsou nejčastěji používány:

200: OK – požadavek proběhl v pořádku

201: Created – při POST, pokud byl vytvořen nový obsah.

204: No Content – požadavek na server proběhl v pořádku, ale server nic nevrátí

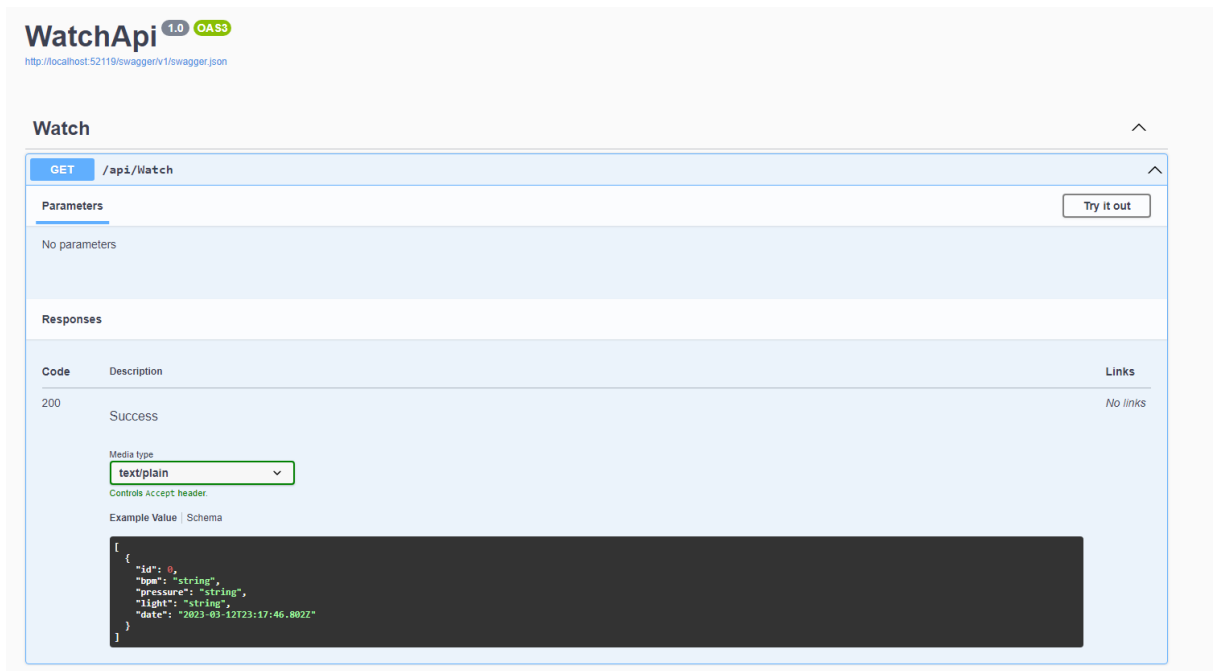
304: No Modified – pokud nebyl od posledního požadavku změněn obsah

400: Bad Request – požadavek na server je nějakým způsobem nečitelný

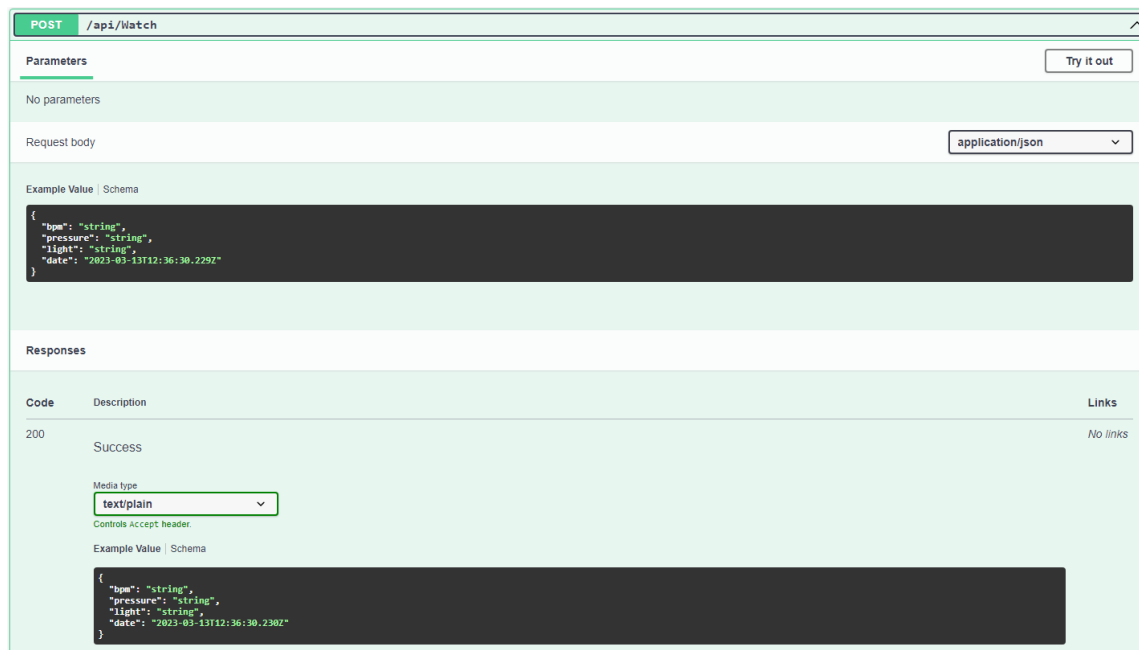
401: Unauthorized – klient není ověřen

403: Forbidden – Klient nemá přístup k danému obsahu

404: Not Found – Zdroj není nalezen



Obrázek 10 WatchAPI GET



Obrázek 11 WatchApi POST

## 5 Příprava prostředí k vývoj

Jelikož jsem ještě aplikaci pro operační systém Tizen nevyvíjel, bude potřeba připravit vývojové prostředí a nainstalovat veškeré nezbytné balíčky a rozšíření, které budu při vývoji používat. Instalaci samotného Tizen studia pro vývoj tady nebudu rozebírat, jelikož se jedná jenom o stažení instalačního souboru a instalaci. Zajímavější však už je instalace balíčků v Package Mangeru.

### 5.1 Instalace balíčků v Package Mangeru

První, na co je potřeba se zaměřit je instalace 4.0 Wearable modulu pro vývoj webové aplikace. Dalším velice důležitým balíčkem je Samsung Certificate Extension a Samsung Wearable Extension. Tyto rozšíření umožňují vytváření certifikátů pro spuštění aplikace na emulátoru, nebo nahrání aplikace přímo do chytrých hodinek.

### 5.2 Vytvoření profilů certifikátů

Ještě před spuštěním emulátoru, nebo nasazením aplikace do hodinek, musí být vytvořeny certifikáty v Certificate Manageru. Tizen certificate slouží pro spouštění aplikace přes emulátor a Samsung certificate slouží pro nasazení aplikace přímo do chytrých hodinek a vytváří se až po připojení hodinek, abychom mohli udělit povolení na instalaci aplikace přímo specifickému zařízení.

## 6 Vývoj Aplikace

V této kapitole se budu zabývat podrobným popisem vývoje aplikace.

### 6.1 Měření srdečního tepu

Prvním krokem při vývoji aplikace bylo vytvořit funkci, která umožní uživateli změřit aktuální srdeční tep a uloží průměrnou hodnotu do lokálního úložiště hodinek. V této fázi zatím nebudu data zpřístupňovat vzdáleně.

#### 6.1.1 Human Activity Monitor

Pomocí Human Activity se spustí senzor pro sledování srdečního tepu a následně je možné tyto data získávat.

```
function startMonitoring()
{
  try {
    isMonitoring = true; // nastavíme proměnnou pro
sledování na true
    hrpData.innerHTML = "Příprava měření";
    stopBtn.innerHTML = "Zrušit"
    // Spuštění sledování senzoru a zobrazení informace
    tizen.humanactivitymonitor.start("HRM", onchangedHRM,
onerror, {batchInterval: 100});
    // Získání dat ze senzoru a zobrazení informace

    tizen.humanactivitymonitor.getHumanActivityData("HRM",
onsuccessHRM, onerror);
  } catch (error) {
    hrpData.innerHTML = "Chyba: " + error.message;
    isMonitoring = false;
  }
}
```

Po spuštění senzoru může nastat několik stavů v závislosti na tom, zda se vyskytl nějaký error, nebo přístup k senzoru proběhl úspěšně a dále se sleduje každá změna naměřené hodnoty.

```
// Funkce pro zpracování úspěšného získání dat srdečního tepu
function onsuccessHRM(hrmInfo) {
    var hrm = hrmInfo.heartRate;
    mesureHRM(hrm);
    //hrmData.innerHTML = "Srdeční tep: " + hrm;
}
//Funkce pro zpracování změny dat srdečního tepu
function onchangedHRM(hrmInfo) {
    var hrm = hrmInfo.heartRate;
    mesureHRM(hrm);
    //hrmData.innerHTML = "Srdeční tep: " + hrm;
}
// Funkce pro zpracování chyby při získávání dat srdečního tepu
function onerror(error) {
    hrmData.innerHTML = "Chyba: " + error.message;
    isMonitoring = false;
}
```

### 6.1.2 Průběh měření

Po úspěšném přistoupení a zahájení aktivity senzoru srdečního tepu se spustí proces ukládání hodnot a následného výpočtu průměrného srdečního tepu po ukončení měření. Jako první se zavolá metoda pro spuštění animace textu, která se bude opakovat po dobu měření.

```
function mesureHRM(hrm) {
    mesureAnimation();
}
```

Dále kontroluji jestli není hodnota ze senzoru menší jak 1, což znamená, že senzor v tu chvíli neměří žádná data. Pokud senzor není schopný měření z jakéhokoliv důvodu, začne



se počítat počet neúspěšných pokusů. Po určitém počtu těchto neúspěšných pokusů se senzor zastaví a zobrazí se zpráva o neúspěšném měření.

```
//postupně budu testovat zda senzor stále nezaznamenává žádné hodnoty
    zeroCount++;
    if (zeroCount > 250) //pokud po určeném počtu pokusů stále nic nezaznamenává ukončím měření a zobrazím chybovou hlášku
    {
        errorStat = true;
        stopMonitoring();
        zeroCount = 0;
        isMonitoring = false;
    }
```

Pokud měření proběhlo v pořádku, vypočítá se průměrná hodnota ze všech naměřených, následně se uloží do lokálního úložiště a zobrazí se uživateli zpráva o úspěšném měření.

```
zeroCount = 0; //nastavím proměnnou na výchozí hodnotu
    //hrmData.innerHTML = "Měření: " + hrm; //zobrazování průběžné hodnoty při probíhající měření
    hrmSum = hrmSum + hrm;
    hrmCount++;
    if (hrmCount === maxCount) {
        isMonitoring = false;
        hrmAvg = hrmSum / hrmCount; //vypočtení průměru z naměřených hodnot
        stopMonitoring(); //zastavení měření
        //hrmData.innerHTML = "Srdeční tep: " + hrmAvg.toFixed(0); //vypsání hodnoty uživateli
        lastBPM.innerHTML = hrmAvg.toFixed(0) + " BPM";
        //vypsání hodnoty uživateli
        localStorage.setItem("avgHRM", hrmAvg.toFixed(0).toString()); //uložení hodnoty do lokálního úložiště pod klíčem "avgHRM"
    }
```

### 6.1.3 Ukončení měření

Po zavolání funkce na ukončení měření se zastaví senzor a veškeré proměnné se změní na původní hodnotu.

```
tizen.humanactivitymonitor.stop("HRM");//Zastavení sledování
senzoru
//Nastavení proměnných na výchozí hodnoty
hrmSum = 0;
hrmCount = 0;
zeroCount = 0;
```

Dále se vyhodnotí co bylo příčinou ukončení měření a zobrazí se zpráva uživateli.

```
if (errorStat === false && isMonitoring === false) {
    StopMessage("Měření proběhlo úspěšně");
} else if (errorStat === false && isMonitoring === true)
{
    StopMessage("Měření zastaveno");
} else if (errorStat === true) {
    StopMessage("Při měření došlo k chybě, opakujte
akci!!");
}
```

Během měření bylo také potřeba zajisti, aby uživatel nemohl spustit senzor několikrát.

```
// Přidání posluchačů na tlačítka pro spuštění monitorování
srdečního tepu
startBtn.addEventListener("click", function() {
    if (!isMonitoring) { // pokud měření již běží, nevytvoříme
další instanci
        startMonitoring();
    }
});
```

Ukončení měření může mimo tlačítko „Zrušit“ ukončit také stisknutí hardwarového tlačítka, nebo zhasnutí displeje a přesunutím aplikace do pozadí.

## 6.2 Měření atmosférického tlaku

Tato část kódu implementuje funkci, která spustí senzor atmosférického tlaku a následně zobrazí naměřená data uživateli a uloží hodnoty do lokálního úložiště chytrých hodinek.

```
Var pressureSensor =  
tizen.sensorservice.getDefaultSensor("PRESSURE"); //uložení  
senzoru do proměnné pro pozdější manipulaci  
pressureSensor.start(onsuccessCB); //spuštění senzoru
```

Práce s tímto senzorem je o něco jednodušší, jelikož není potřeba žádný výpočet a pokud měření proběhne úspěšně, hodnota se zobrazí instantně.

```
function onsuccessCB() { //tato funkce se zavolá, pokud start  
senzoru proběhl úspěšně  
    pressureSensor.getPressureSensorData(onGetSuccessCB);  
    //zavolání funkce pro uložení a zobrazení naměřených hodnot  
    pressureSensor.stop(); //zastavení senzor  
}
```

Následně se naměřená hodnota uloží do proměnných pro zobrazování naměřených dat a uloží se do lokálního úložiště.

```
function onGetSuccessCB(sensorData) {  
    var bar = sensorData.pressure.toFixed(0); //odstranění  
plovoucích řádových čárek z naměřené hodnoty  
    barData.innerHTML = "Atmosférický tlak: " + bar + barUnit;  
    //zobrazení naměřených dat, včetně jednotky  
    lastBar.innerHTML = bar + barUnit; //vypsání naměřených  
dat do elementu, který zobrazuje hodnoty z posledního měření  
    localStorage.setItem("bar", bar); //uložení hodnoty do  
lokálního úložiště pod klíčem "bar"  
}
```

Ukončení měření je zde řešeno podobně jako u senzoru srdečního tepu. Je zde taky funkce, která zavírá popup okno po stisknutí příslušného tlačítka.

## 6.3 Měření úrovně světla

Při měření úrovně světla se implementují podobné funkce jako u předchozího senzoru atmosférického tlaku. Hlavní rozdíl je v pojmenování proměnných, nahrávání výsledků a zpráv uživateli do jiných HTML elementů a samozřejmě použití příslušného senzoru.

```
var lightSensor =
tizen.sensorservice.getDefaultSensor('LIGHT');
function onSuccessLightCB() {
    lightSensor.getLightSensorData(onGetSuccessLightCB);
    lightSensor.stop();
}
function onGetSuccessLightCB(sensorData) {
    var light = sensorData.lightLevel.toFixed(0);
    lightData.innerHTML = "Úroveň světla: " + light + " Lux";
    lastLight.innerHTML = light + " Lux"; //vypsání hodnoty
    uživateli
    localStorage.setItem("light", light); //uložení hodnoty
    do lokálního úložiště pod klíčem "bar"
}
```

## 6.4 Data

Tato část kódu zpracovává data ze senzorů, která se uložila do lokálního úložiště.

```
openStats.addEventListener("click", function() {
    avgHRM = localStorage.getItem("avgHRM"); //zobrazím data s
    klíčem avgHRM z lokálního úložiště
    bar = localStorage.getItem("bar"); //zobrazím data s klíčem
    bar z lokálního úložiště
    light = localStorage.getItem("light"); //zobrazím data s
    klíčem light z lokálního úložiště
```

Pomocí funkce `localStorage.getItem(„klíč“)` a klíče, pod kterým byla hodnota uložena, získá data z lokálního úložiště.

Tyto data se nejprve vypíší uživateli.

```
if(avgHRM !== null) //pokud není NaN (ještě nebyla nahrána  
žádná hodnota pod tímto klíčem  
{  
    BPMVal.innerHTML = avgHRM + " BPM"; //zobrazím získaná  
data uživateli  
}  
else{ //jinak nahraj do proměnné string "none" (nutné pro  
defaultní hodnotu při POST)  
    avgHRM = "none";  
}
```

A následně jsou pomocí Fetch API odeslána do vzdáleného úložiště, kde se uloží do SQL databáze.

```
fetch("http://192.168.1.140:52119/api/Watch", { //navázání  
spojení s end pointem  
    method: "POST", //nastavení metody na POST  
    headers: {  
        "Content-Type": "application/json" //formát dat  
určený v hlavičce  
    },  
    body: JSON.stringify({ //určení dat, které se budou  
posílat v těle požadavku  
        "bpm": String(avgHRM),  
        "pressure": String(bar),  
        "light": String(light),  
        "date": formattedDate  
    })  
})
```

Po navázání spojení a pokusu o odeslání dat se uživateli zobrazí stavový kód a zpráva o úspěšném, nebo neúspěšném odeslání dat na určitou adresu a end point.

```

.then(function(response) { //co se má stát, pokud požadavek
na REST API proběhne v pořádku
    console.log(response);
    messageData.innerHTML = response.status + ": " +
response.statusText; //zobrazení HTTP status kódu a HTTP
status zprávy
})
.catch(function(error) { //proběhne, pokud nastane
problém při komunikaci s REST API (API je vypnuté, hodinky
nejdou připojeny k síti)
    messageData.innerHTML = "Chyba při spojení s
API"; //zobrazení chybové hlášky
});

```

## 6.5 API

Abych mohl zpřístupnit data vzdáleně, bylo zapotřebí vytvořit REST API a end point, na který data posílám.

### 6.5.1 Model

Pro vytvoření databáze je potřeba nejdříve vytvořit model.

```

public class WatchData
{
    [Key]
    public int Id { get; set; }
    public string BPM { get; set; }
    public string Pressure { get; set; }
    public string Light { get; set; }
    public DateTime Date { get; set; }
}

```

Jedná se o třídu v jazyku C#, která popisuje data z chytrých hodinek. První položka v této třídě je atribut [Key], který označuje „Id“ jako primární klíč databázové tabulky. Další tři vlastnosti třídy popisují data, která jsou na chytrých hodinkách sledována. Každá z těchto položek má datový typ „string“, ale jelikož jsou uloženy v databázi, mohou být později

převezeny na číselné hodnoty. Poslední položka má datový typ „DateTime“ a je určena k uložení přesného data a času vytvoření záznamu do databáze.

### 6.5.2 ApplicationDbContext

Tato část kódu představuje třídu „ApplicationDbContext“, která je odvozena od třídy „DbContext“. Slouží jako kontext k vytvoření a spravování připojení k databázi a definuje tabulku „WatchData“ v databázi.

```
public class ApplicationDbContext : DbContext
{
    public ApplicationDbContext(DbContextOptions
options) : base(options)
    {
    }

    public DbSet<WatchData> WatchData { get; set; }

    protected override void OnModelCreating(ModelBuilder
modelBuilder)
    {
        base.OnModelCreating(modelBuilder);
    }
}
```

Definice vlastnosti „WatchData“ typu DbSet umožňuje přístup k datům, která jsou uložena v databázi.

### 6.5.3 Vytvoření databáze

V ASP.NET Core se používají Entity Framework Core a migrace pro vytváření a aktualizaci struktury databáze.

Příkazem „add-migration“ se vytvoří nová migrace. Migrace obsahuje kód, který provede změny v databázi.

Příkaz „update-database“ se používá k aktualizaci databáze, nebo vytvoření nové databáze, pokud ještě neexistuje. Spouští migrace, které ještě nebyly provedeny a aplikuje kód, který byl definován v kódu migrace.

#### 6.5.4 Kontrolér

Kontrolér je třída, která obsahuje jednotlivé akce (metody), které zpracovávají http požadavky. V tomto konkrétním případě pracuje s ApplicationDbContext pro práci s databází.

```
private readonly ApplicationDbContext _context;

public WatchController(ApplicationDbContext context)
{
    _context = context;
}
```

Dále obsahuje metody „GetWatch()“, která slouží k získání všech záznamů v tabulce „WatchData“, „PostWatch()“, která umožňuje vytvořit nový záznam v tabulce „WatchData“ a metodu „DeleteAll()“, která slouží k odstranění všech záznamů z tabulky „WatchData“. Veškeré změny se uloží do databáze pomocí „DbContext“.

```
//příklad metody GetWatch()
// GET: api/Watch
[HttpGet]
public async Task<ActionResult<IEnumerable<WatchData>>>
GetWatch()
{
    return await _context.WatchData.ToListAsync();
}
```

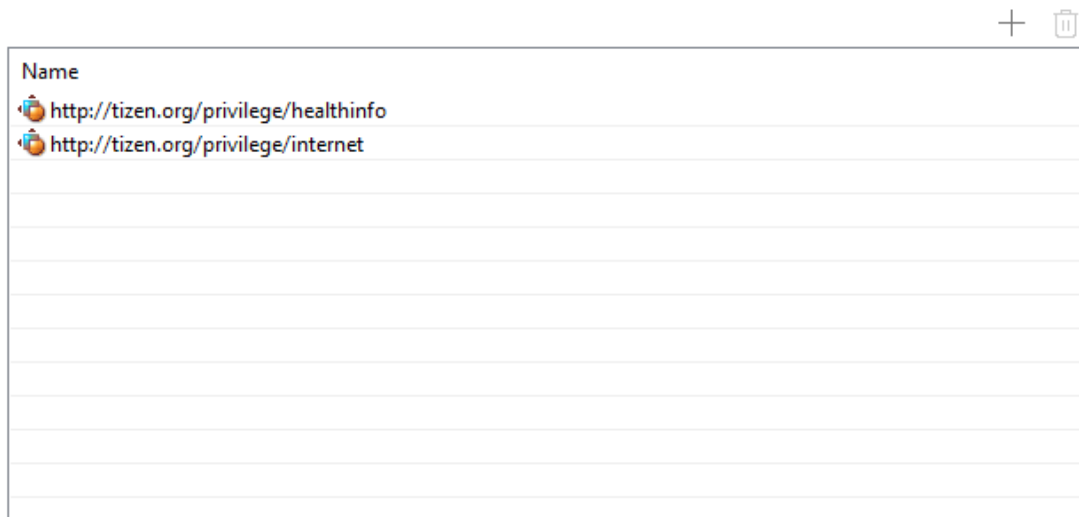




## 6.6 Config

V souboru config.xml bylo potřeba přidat privilegia pro přístup k internetu a také pro přístup k senzoru srdečního tepu.

### Required Privileges

'tizen:privilege' element indicates what sensitive API or API groups this web application declares that it may attempt to access.



Name
 http://tizen.org/privilege/healthinfo
 http://tizen.org/privilege/internet

## 6.7 HTML dokument

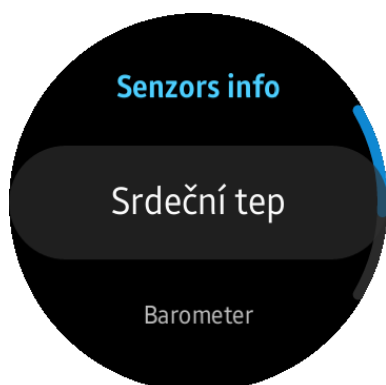
Pro tuto aplikaci jsem vytvořil html dokument, který obsahuje několik částí. Každý blok reprezentuje jednu stránku aplikace a pro přepínání stránek používám knihovnu TAU. Tuto knihovnu také používám pro veškeré stylování v aplikaci a také pro zobrazování a zavírání popup oken, která slouží k zobrazování zpráv, nebo aktuálního stavu měření.

## 7 Testování a nasazení aplikace

V této kapitole se budu zabývat testování funkčnosti aplikace na chytrých hodinkách. Testování bude probíhat na emulátoru, jelikož zde mohu nasimulovat veškeré situace, které mohou nastat.

### 7.1 Měření tepu

Otestování úspěšného měření srdečního tepu a následné zobrazení zprávy.



Obrázek 12 Hlavní menu 1

Z hlavního menu se pomocí tlačítka dostanu do menu pro měření srdečního tepu.



Obrázek 13 HRM menu

Po stisknutí tlačítka „Začít měřit“ se zobrazí pop-up okno s aktuálním stavem měření.



Obrázek 14 HRM aktuální stav

Po úspěšném měření se zobrazí zpráva uživateli.



Obrázek 15 Úspěšné měření BPM

Zároveň se v menu pro měření srdečního tepu zobrazí poslední naměřená hodnota.



Obrázek 16 HRM menu hodnota

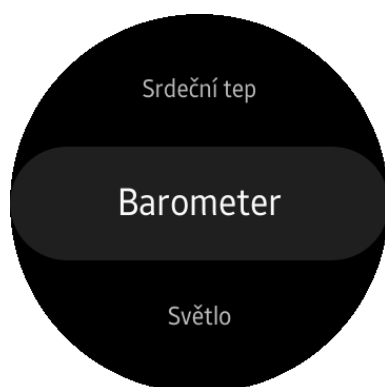
Pokud při měření dojde z jakéhokoliv důvodu k chybě, zobrazí se uživateli chybová zpráva.



Obrázek 17 HRM chybová zpráva

## 7.2 Měření atmosférického tlaku

Otestování měření atmosférického tlaku a následného zobrazení zprávy uživateli.



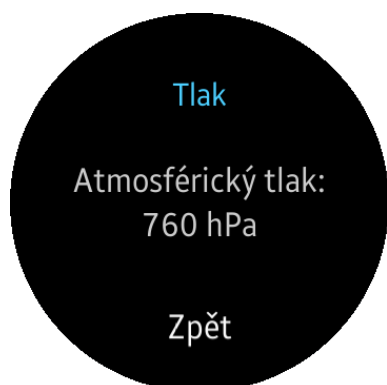
Obrázek 18 Hlavní menu 2

Z hlavního menu se pomocí tlačítka dostanu do menu pro měření atmosférického tlaku.



Obrázek 19 Menu pro měření atmosférického tlaku

Po stisknutí tlačítka začít měřit se změří atmosférický tlak a zobrazí se naměřená hodnota.



Obrázek 20 Úspěšné změření atmosférického tlaku

Zároveň se taky tato hodnota zobrazí v menu měření atmosférického tlaku.

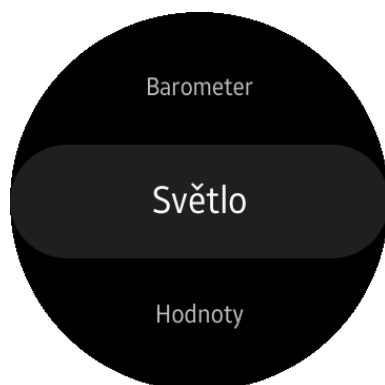


Obrázek 21 Menu pro měření atmosférického tlaku – hodnota

U tohoto senzoru se mi nepodařilo nasimulovat chybný stav na fyzickém zařízení, ani na emulátoru.

## 7.3 Měření úrovně světla

Otestování měření úrovně světla a následného zobrazení zprávy uživateli. Toto měření funguje na stejném principu jako měření u senzoru atmosférického tlaku.



Obrázek 22 Hlavní menu 3

Z hlavního menu se pomocí tlačítka dostanu do menu pro měření úrovně světla.



Obrázek 23 Menu pro měření úrovně světla

Po stisknutí tlačítka začít měřit se změří úroveň světla a zobrazí se naměřená hodnota.



Obrázek 24 Úspěšné změření úrovně světla

Zároveň se taky tato hodnota zobrazí v menu měření úrovně světla.



Obrázek 25 Menu pro měření úrovně světla – hodnota

## 7.4 Data

Otestování zobrazení naměřených dat z lokálního úložiště hodinek a následné odeslání do vzdáleného úložiště.



Obrázek 26 Hlavní menu 4

Z hlavního menu se pomocí tlačítka dostanu do menu pro odesílání dat.



Obrázek 27 Menu pro odesílání dat 1



Obrázek 28 Menu pro odeslání dat 2

Po stisknutí tlačítka „Odeslat hodnoty“ se spustí proces odesílání dat do vzdáleného úložiště.



Obrázek 29 Odesílání dat

Při úspěšném odeslání dat se zobrazí zpráva uživateli.



Obrázek 30 Úspěšné odeslání dat

Pokud se při odesílání vyskytne chyba, například kvůli nedostupnosti API, nebo z důvodu nepřipojení hodinek k internetu, zobrazí se chybová zpráva.





Obrázek 31 Neúspěšné odeslání dat

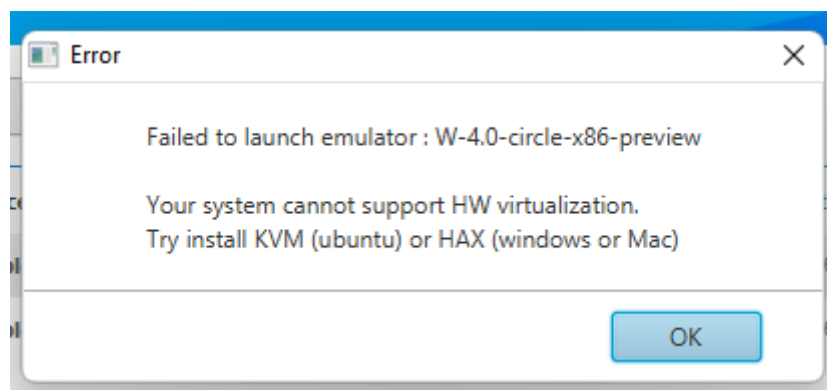
## 7.5 Nasazení aplikace do hodinek

Nasazení aplikace do hodinek jsem zprostředkoval pomocí Tizen Studia. Hodinky jsem pomocí Wi-Fi připojil k místní síti a také k Tizen Studio pomocí Device Manageru. Poté jsem jenom projekt spustil a automaticky se aplikace nainstalovala na chytré hodinky.

## 8 Problémy při vývoji

### 8.1 Emulátor

Při vývoji a testování aplikace jsem objevil problém se spuštěním emulátoru na počítači, který disponuje procesorem AMD. Když jsem se pokusil emulátor spustit na notebooku s procesorem Intel, bez problému se spustil. Problém spočíval v tom, že pro spuštění emulátoru je potřeba HAXM, který ale nefunguje na AMD procesorech, jelikož by vytvořen přímo firmou Intel. Jedním z řešení je nainstalovat operační systém Linux (Ubuntu), nebo využít pro testování aplikace místo emulátoru přímo chytré hodinky, do kterých jde aplikace nahrát pomocí místní sítě. Já jsem přistoupil na druhé řešení, tedy používat chytré hodinky při vývoji s AMD procesorem a emulátor při vývoji s Intel procesorem.



Obrázek 32 Emulátor error

## Závěr

Výstupem toho projektu je funkční aplikace na chytré hodinky Samsung Gear S3, která umožňuje uživateli ovládat senzory zařízení a ukládat jejich data do lokálního úložiště chytrých hodinek. Tyto data je následně možné odeslat pomocí API do vzdáleného úložiště, kde se uloží do SQL databáze.

Aplikace je nyní ve finální podobě, ale mohla by být do budoucna rozšířena o integrované API v hodinkách, aby bylo možné měření provádět průběžně a data mohla být odeslána na vyžádání uživatele například z osobního PC. Také by byla možná implementace animací a celkové zlepšení uživatelského prostředí aplikace.

Závěrem bych chtěl říct, že díky této práci jsem si rozšířil svoje povědomí o vývoji aplikací na různých platformách a poprvé jsem si vyzkoušel jaké to je vést dlouhodobý projekt. Celkově jsem na tomto projektu strávil přibližně 100 hodin a musel jsem si pečlivě rozvrhnout svůj čas, který jsem tomuto projektu věnoval.

## Seznam zkratk a odborných výrazů

### **HTML**

HyperText Markup Language – značkovací jazyk používaný pro tvorbu webových stránek.

### **CSS**

Cascading Style Sheets – Kaskádové styly pro popis způsobu zobrazení elementů na stránkách.

### **JS**

JavaScript – skriptovací jazyk

### **API**

Application Programming Interface – Sběrka procedur, funkcí, tříd a protokolů, které může programátor využívat.

### **IoT**

Síť fyzických zařízení, které jsou vybaveny elektronikou, softwarem, senzory a síťovou konektivitou, která umožňuje vzájemné propojení těchto zařízení a následnou výměnu dat.

### **XAML**

Extensible Application Markup Language - (obdoba HTML) využívaný k popisu grafického rozhraní v aplikacích společnosti Microsoft nové generace.

### **Wi-Fi**

Wireless Fidelity – Hlavním znakem Wi-Fi připojení je bezdrátová funkce, která přenáší signál.

### **mAh**

Kapacita baterie se uvádí v jednotce mAh. Standardní hodnoty jsou 1 000–10 000 mAh. Čím vyšší kapacita, tím delší výdrž baterie.

### **TAU**

Tizen Advanced UI

## **HAXM**

Multiplatformní hardwarově podporovaný virtualizační engine (hypervizor), který se široce používá jako akcelerátor pro emulátor Androidu a QEMU. Vždy podporoval běh v systémech Windows a macOS a byl portován i na další hostitelské operační systémy.

## **HTTP**

Hypertext Transfer Protocol je internetový protokol určený pro komunikaci s WWW servery. Slouží pro přenos hypertextových dokumentů ve formátu HTML, XML, i jiných typů souborů.

## **.NET**

Soubor technologií v softwarových produktech, které tvoří celou platformu, která je dostupná nejen pro web.

## **Fetch API**

Fetch API poskytuje rozhraní Javascriptu pro přístup a manipulaci s částmi protokolu, jako jsou požadavky a odpovědi.

## **SQL**

Zkratka pro standardizovaný strukturovaný dotazovací jazyk, který je používán pro práci s daty v relačních databázích.

## Seznam obrázků

Obrázek 1 Vývojové prostředí Tizen Studio.....	2
Obrázek 2 Emulátor .....	3
Obrázek 3 Správce certifikátů .....	3
Obrázek 4 Správce zařízení .....	4
Obrázek 5 Správce balíčků.....	5
Obrázek 6 Architektura Tizen [6].....	7
Obrázek 7 Správce balíčků aplikací [7].....	8
Obrázek 8 Architektura Tizen .NET [10].....	9
Obrázek 9 Chytré hodinky .....	12
Obrázek 10 WatchAPI GET .....	14
Obrázek 11 WatchApi POST.....	14
Obrázek 12 Hlavní menu 1 .....	27
Obrázek 13 HRM menu.....	27
Obrázek 14 HRM aktuální stav .....	28
Obrázek 15 Úspěšné měření BPM.....	28
Obrázek 16 HRM menu hodnota.....	28
Obrázek 17 HRM chybová zpráva .....	29
Obrázek 18 Hlavní menu 2 .....	29
Obrázek 19 Menu pro měření atmosférického tlaku .....	29
Obrázek 20 Úspěšné změření atmosférického tlaku.....	30
Obrázek 21 Menu pro měření atmosférického tlaku – hodnota.....	30
Obrázek 22 Hlavní menu 3 .....	31
Obrázek 23 Menu pro měření úrovně světla .....	31
Obrázek 24 Úspěšné změření úrovně světla.....	31
Obrázek 25 Menu pro měření úrovně světla – hodnota.....	32
Obrázek 26 Hlavní menu 4 .....	32
Obrázek 27 Menu pro odesílání dat 1 .....	32
Obrázek 28 Menu pro odesílání dat 2 .....	33

Obrázek 29 Odesílání dat .....	33
Obrázek 30 Úspěšné odeslání dat .....	33
Obrázek 31 Neúspěšné odeslání dat .....	34
Obrázek 32 Emulátor error .....	35

## Použitá literatura

- [1] Samsung Developers. *Samsung Developers* [online]. Copyright © 2023 SAMSUNG, a [cit. 23.01.2023]. Dostupné z: <https://developer.samsung.com/smarttv/develop/tools/tizen-studio.html>
- [2] Running Applications on the Emulator | Tizen Developers. *Tizen Developers | An open source, standards-based software platform for multiple device categories*. [online]. Copyright © 2012 Tizen Project, a [cit. 23.01.2023]. Dostupné z: <https://developer.tizen.org/development/tizen-studio/web-tools/running-and-testing-your-app/emulator>
- [3] Introduction to Tizen | Tizen Docs. *Tizen Docs* [online]. Copyright © 2023 Tizen Project, a Linux Foundation Project, a [cit. 23.01.2023]. Dostupné z: <https://docs.tizen.org/platform/what-is-tizen/overview/>
- [4] Certificate Manager | Tizen Docs. *Tizen Docs* [online]. Copyright © 2023 Tizen Project, a Linux Foundation Project, a [cit. 25.01.2023]. Dostupné z: <https://docs.tizen.org/application/vstools/tools/certificate-manager/>
- [5] Device Manager | Tizen Docs. *Tizen Docs* [online]. Copyright © 2023 Tizen Project, a Linux Foundation Project, a [cit. 25.01.2023]. Dostupné z: <https://docs.tizen.org/application/vstools/tools/device-manager/>
- [6] Tizen Frameworks | Tizen Docs. *Tizen Docs* [online]. Copyright © 2023 Tizen Project, a Linux Foundation Project, a [cit. 25.01.2023]. Dostupné z: <https://docs.tizen.org/application/>
- [7] Tizen Web Application | Tizen Docs. *Tizen Docs* [online]. Copyright © 2023 Tizen Project, a Linux Foundation Project, a [cit. 27.01.2023]. Dostupné z: <https://docs.tizen.org/application/web/>
- [8] Tizen Web Application | Tizen Docs. *Tizen Docs* [online]. Copyright © 2023 Tizen Project, a Linux Foundation Project, a [cit. 27.01.2023]. Dostupné z: <https://docs.tizen.org/application/web/#package>



- [9] Tizen Native Application | Tizen Docs. *Tizen Docs* [online]. Copyright © 2023 Tizen Project, a Linux Foundation Project, a [cit. 27.01.2023]. Dostupné z: <https://docs.tizen.org/application/native/>
- [10] .NET Application | Tizen Docs. *Tizen Docs* [online]. Copyright © 2023 Tizen Project, a Linux Foundation Project, a [cit. 12.02.2023]. Dostupné z: <https://docs.tizen.org/application/dotnet/>
- [11] *Wikipedie: Otevřená encyklopedie: Chytré hodinky* [online]. c2022 [citováno 12. 02. 2023]. Dostupný z WWW: <[https://cs.wikipedia.org/w/index.php?title=Chytr%C3%A9\\_hodinky&oldid=20908524](https://cs.wikipedia.org/w/index.php?title=Chytr%C3%A9_hodinky&oldid=20908524)>
- [12] Package Manager | Tizen Docs. *Tizen Docs* [online]. Copyright © 2023 Tizen Project, a Linux Foundation Project, a [cit. 12.02.2023]. Dostupné z: <https://docs.tizen.org/application/native/guides/app-management/package-manager/>
- [13] Gear S3 Frontier | SM-R760NDAAXEZ | Specifikace a popis | Samsung Česká republika. [online]. Copyright © 1995 [cit. 16.02.2023]. Dostupné z: <https://www.samsung.com/cz/wearables/gear-s3-frontier/SM-R760NDAAXEZ/>
- [14] Tizen Advanced UI | Tizen Docs. *Tizen Docs* [online]. Copyright © 2023 Tizen Project, a Linux Foundation Project, a [cit. 16.02.2023]. Dostupné z: <https://docs.tizen.org/application/web/guides/tau/tau/>
- [15] GitHub - intel/haxm: Intel® Hardware Accelerated Execution Manager (Intel® HAXM). *GitHub: Let's build from here · GitHub* [online]. Copyright © 2023 GitHub, Inc. [cit. 28.02.2023]. Dostupné z: <https://github.com/intel/haxm>
- [16] Learn ASP.NET Web API using Step-by-Step Tutorials. *TutorialsTeacher – Learn Technologies* [online]. Copyright © 2023 TutorialsTeacher.com, a [cit. 28.02.2023]. Dostupné z: <https://www.tutorialsteacher.com/webapi>
- [17] Stopařův průvodce REST API. itnetwork.cz - Učíme národ IT [online]. Copyright © 2023 itnetwork.cz. Veškerý obsah webu [cit. 10.03.2023]. Dostupné z:

<https://www.itnetwork.cz/programovani/nezarazene/stoparuv-pruvodce-rest-api/>

[18] CRUD Operations – What is CRUD?. [online]. [cit. 10.03.2023]. Dostupné z: <https://www.freecodecamp.org/news/crud-operations-explained/>

## A. Seznam příložených souborů

Na přiloženém datovém nosiči se nacházejí následující soubory a složky:

- **MP2023-Zeman-Vít-P4-Vývoj\_aplikace\_pro\_chytré\_hodinky.docx** – editovatelná verze dokumentace maturitní práce
- **MP2023-Zeman-Vít-P4-Vývoj\_aplikace\_pro\_chytré\_hodinky.pdf** – tisknutelná verze dokumentace maturitní práce
- **Aplikace** – zdrojové kódy
- **API** – Zdrojové kódy

Odkaz na API: <https://watchapi20230314070640.azurewebsites.net/api/Watch>