Dokument specifikace požadavků

# Specifikace požadavků

Vytvořit aplikaci, která bude zobrazovat informace o počasí. Zobrazení počasí se očekává aktuální a předpověď. Pro přihlášené uživatele (předplatné) je přidána možnost uložit si místa jako oblíbená a zobrazovat historická data o počasí. Zároveň má uživatel možnost zobrazit si data pře REST endpoint. Počasí bude možné získat pro celou EU. Aplikace musí být multiplatformní, prozatím bude postačovat počítač a telefon.

# Specifikace aplikace

Aplikaci vytvoříme jako webovou aplikace, tím docílíme požadavku, aby aplikace byla spustitelná na více typech zařízení.

## Specifikace aplikace

### Jazyk aplikace

Aplikaci bude vytvořena jako 2 aplikace: REST API a React aplikace. Rest Api bude vytvořeno pomocí jazyka c# ve frameworku ASP.NET Core Web Api. Tato část projektu bude zpracovávat všechna dat, takže frontend v Reactu bude pouze komunikovat s naším REST Api. Frontend vyvinutý v React za pomoci Node.js bude pouze poskytovat vizualizaci dat z REST Api.

REST Api se bude zabývat získáváním dat o počasí pomocí použitých knihoven. Zárověň bude ukládat data o uživatelích, a to tak, že každého uživatele uloží ve formátu json do json dokumentu. U každého uživatele, se uložené lokace uloží v poli lokací do stejného souboru. Bylo by možné ukládat data do databáze, ale zvýšili by se náklady na provoz aplikace, protože databáze nejsou nejlevnější záležitostí.

### Použité knihovny

REST API vytvořené v ASP.NET Core WEB API bude získávat data o předpovědi a aktuálním počasí od [OpenWeatherMap](https://openweathermap.org/). Data historická získá od [WeatherApi](https://www.weatherapi.com/). Jelikož se budou ukládat místa pro placené uživatele a zároveň uživatelé budou zadávat pouze název města a stát, ve kterém se město nachází, je potřeba získávat informace o daném místě. K tomu nám pomůže [OpenCage Geocoder](https://opencagedata.com/).

Vizuální zobrazení dat bude provedeno v React aplikaci za pomoci Node.js 20 LTS. Knihovnu, která bude implementovat UI zvolíme Rsuite se kterým máme zkušenosti z minulých let. Na zobrazení počasí formou obrázku využijeme ikony z balíčku react-icons. Dotazování na server pro data o počasí, uživatelích a lokacích bude zprostředkovávat balíček axios.

Aby bylo splněno, že se dá na počasí podívat na více zařízeních, použijeme Gid od Rsuite, který umožní zobrazovat aplikaci čitelně nejen na počítači a telefonu, ale i na dalších typech zařízení.

### Publikování aplikace

Pro publikování aplikace se nabízí mnoho variant. Jelikož máme aplikaci napsantou v .NET a React, bude přínosné obě tyto části poblikovat na Microsoft Azure. Pokud by bylo potřeba přesunout frontend na jiný server, mohlo by být využito také GitHub Pages.

## Definice rizik

Rizika aplikace mohou nastat, pokud vývojáři zvolných knihoven pro implementaci razantně zasáhnou do knihoven. Pokud některé API poskytující informace o počasí změní formát výstupu (jiný formát json) nebo přestanou poskytovat námi očekávaná data, může dojít ke kritickým stavům aplikace. Zároveň je kritické místo použití verze jazyků. U .NET se jedná o nejnovější verzi (.NET 8) a u React se jedná také o nejnovější verzi Node.js (verze 20). Oba tyto jazyky jsou v LTS (Long Term Support), což by mělo znamenat, že by tato situace (že se vyskytne chyba s podporou) neměla v dohledné době nastat. Jelikož se tyto jazyky stále vyvíjejí, je možné, že může nastat chyba s kompatibilitou v prohlížečích. Při publikování aplikace na server může nastat chyba, která bude složitější na opravu, tím by se mohlo nasazování zpozdit.

# Předpokládaná délka vývoje aplikace

|  |  |
| --- | --- |
| Vykonávaná akce | Čas  (člověk\*den – den brán jako 24 hodin) |
| Návrh vývoje (vývojové prostředí, návrh databáze/ukládání dat, získávání dat, prezentace dat) | 3 |
| Vývoj API | 2 |
| Testování Api | 4,5 |
| Vývoj frontendu | 3 |
| Publikování na externí server | 2-3 |
| Celkem | 15,5-16,5 |