**WindStats App**

**Michał Stopyra (292794)**

**Specyfikacja projektu**

Ogólny opis aplikacji:

Jest to aplikacja kierowana do wind/kite surferów, którzy chcąc pojechać na

Zagraniczny (i nie tylko) wyjazd aby pływać i szukają najlepszych warunków do uprawiania tych

sportów w danym miesiącu. Na świecie jest bardzo dużo spotów znanych z tego, że

wieje tam mocny wiatr lub są duże fale, jednak żaden z nich nie oferuje takich

warunków przez cały rok. Co oznacza, że odpowiedź na pytanie gdzie pojechać w

danym miesiącu nie jest taka prosta. Ta aplikacja na podstawie preferencji

użytkownika oraz danych z poprzednich lat ma za zadanie wskazać kilka najlepszych

miejsc do podróży w celu pływania na desce.

**Funkcjonalności**:

1. Użytkownik wybiera swoje preferencje:

* Miesiąc, w którym planuje podróż
* Siła wiatru (węzły)
* Rodzaj spotu – fale, płaska woda, “chop”

Następnie na podstawie podpiętego weather api aplikacja policzy statystyki z

poprzednich sezonów i na ich podstawie poda listę spotów najbardziej

odpowiednich do preferencji użytkownika

2. Wybranie konkretnego spotu i wyświetlenie statystyk z podziałem na kolejne

miesiące.

3. Wybranie kraju z klikalnej mapy w celu zobaczenia listy spotów w jego obrębie i porównaniu

Statystyk.

**Architektura Systemu:**

Frontend - Vaadin Framework (dodatkowo Polymer i odrobina JS oraz html i css przy tworzeniu klikalnej mapy) ->

Backend – Spring Boot, Java-> weather api – Dark Sky Api

Wszystkie dane są zapisywane do lokalnej bazy danych (JPA – H2) i aplikacja w trakcie działania pracuje tylko na tej bazie danych. Oznacza to, że api z pogodą zostało wywołane tylko na etapie budowania bazy danych.

**Instrukcja Instalacji:**

Aplikację można używać na dwa sposoby:

1. W folderze z plikiem pom.xml należy otworzyć terminal i wpisać komendę

"mvn spring-boot:run", a następnie otworzyć w przeglądarce internetowej

adres <http://localhost:8080>

1. Można także zainstalować aplikację na swoim komputerze, jako, że jest to Progressive Web App i używać niezależnie od okna przeglądarki internetowej. Po wykonaniu kroków opisanych w punkcie 1. wyświetli się okienko któro zapyta czy chcemy zainstalować aplikację na komputerze - należy wybrać tę opcję, a następnie znaleźć ikonkę z aplikacją w menu z aplikacjami na komputerze. Po wybraniu tej ikony aplikacja uruchomii się w swoim własnym oknie.

**Testy:**

1. WeatherTest, ToolsTest – proste unitesty sprawdzające działanie kilku metod.
2. SpotServiceTest – sprawdzanie funkcji działających na bazie danych.
3. Testy w paczce views - testują kilka funkcjonalności konkretnych widoków.

**Opis działania aplikacji:**

Przy pierwszym uruchomieniu aplikacji została zbudowana baza danych z krajami, spotami oraz statystykami pogodowymi z całego roku 2019. Przy kolejnych uruchomieniach aplikacja korzysta z gotowej bazy.

Użytkownik uruchamia aplikację:

1. Pojawia się widok MainView + MainLayout (pokazuje się w każdym widoku aplikacji). Tam możemy wybrać parametry.
2. Zostaje uruchomiony widok ListView z listą spotów -> po kliknięciu spotu pojawia się okno SpotInfoTextView, gdzie mamy dodatkowe informacje na temat spotu.
3. Możemy przejść do okna MapView gdzie z mapy europy możemy wybrać kraj dla którego wyświetli się CountryView gdzie również po kliknięciu uruchomi się SpotInfoView - będą na nim dodatkowo pokazane statystyki dla każdego miesiąca.

Opis backendowych klas

1. Klasy w paczce entity tworzą obiekty na których pracujemy w aplikacji - są one wrzucane do bazy danych.
2. Interfejsy w paczce repository łączą te obiekty z bazą danych. Repozytoria rozszerzają klasę JPARepository.
3. Klasy w paczce service zawierają funkcje przydatne do obróbki danych – czyli encji z bazy danych. Używają konkretnych repozytoriów w celu wykonania danego działania.
4. Klasy w paczce weatherapi:
   1. Tools – przydatne funkcje do obróbki danych np. Liczbowych. Są używane na obiektach różnych typów.
   2. ApiKey – klasa która zawiera kod do api – (Powinna być jakoś zaszyfrowana).
   3. UserPrefenrences – klasa zawierająca statyczne pola, które są dostępne w całej aplikacji. Opisują one wybory parametrów dokonanych przez użytkownika.
   4. Weather – klasa używana w celu dostępu do api z pogodą.
5. Application – Klasa uruchamiająca aplikację
6. Klasy Test

Dodatkowo w folderze webapp znajdują się wszystkie zdjęcia, ikony itd. Oraz strona wyświetlana gdy aplikacja jest offline.

W folderze frontend znajduje się plik map-view.js który tworzy widok mapy w MapView.

**Komentarze:**

1. Niestety Darksky weather api nie spełniło moich oczekiwań odnośnie prawidłowości statystyk wiatrowych. Są one mocno przekłamane. Jestem w stanie to stwierdzić, ponieważ mniej więcej orientuję się jak wygląda pogoda na opisanych przeze mnie spotach surfingowych w danych miesiącach. Uznałem zatem, że bez sensu będzie budowanie bazy danych na podstawie kilku ostatnich sezonów i zostałem tylko przy danych z 2019 roku. Zdecydowałem się na ograniczenie danych głównie ze względu na to, że musiałbym zapłacić parę dolarów za wołanie api dziesiątki tysięcy razy w celu zbudowania bazy, a prawdopodobnie nie sprawiłoby to, że wyniki byłyby bliższe rzeczywistości. Mimo wszystko wyniki te nie wpływają na działanie aplikacji – wszystko działa tak jak powinno, jednak po prostu nie należy ufać statystykom, które są wyświetlane.
2. Zdaję sobie sprawę, że niektóre użyte przeze mnie rozwiązania prawdopodobnie nie są efektywne i istnieją lepsze sposoby na osiągnięcie tego samego, jednakże uważam, że napisanie tej aplikacji wyposażyło mnie w dużą ilość wiedzy w dziedzinie aplikacji webowych.
3. W widoku mapy po kliknięciu na któryś kraj pojawia się Error dotyczący funkcji, której zupełnie nie używam, nie wpływa on na działanie aplikacji. Próbowałem schować go używając htmla ora js’a jednak framework vaadin bardzo utrudnia korzystanie z tych języków i niestety mi się to nie udało.
4. Ciekawi mnie na ile sensownym rozwiązaniem jest używanie Frameworków typu Vaadin, które w niekonwencjonalny sposób odzielają front od backendu w aplikacji. Z tego co udało mi się do tej pory dowiedzieć to jednak zwykle stosuje się standardowy podział aplikacji na frontend i backend - oczywiście są to trudniejsze rozwiązania do ogarnięcia z zerowym poziomem wiedzy na starcie.
5. Bazę danych można podejrzeć po wpisaniu adresu <http://localhost:8080/console> a następnie zaakceptowaniu domyślnego loginu i hasła.