Programowanie urządzeń mobilnych

laboratorium

Dokumentacja Projektowa

Jakub Engielski
Jan Kwiatkowski
Szymon Bacański
Michał Gałęzyka

Prowadzący mgr inż. Marcin Tracz

Spis treści:

1. Temat oraz opis projektu

1.1 Temat projektu

Plan zajęć bazujący na uczelnianym harmonogramie zajęć.

1.2 Opis projektu

Celem naszego projektu jest:
Stworzenie aplikacji na środowisko android z wykorzystaniem zewnętrznego API wykorzystujące technologie scrapowania na stronie harmonogramu uczelni Collegium Witelona.

2. Grupa Projektowa:

- Jakub Engielski - api, dokumentacja
- Jan Kwiatkowski - api, api-dev-ops
- Szymon Bacański - frontend, frontend-dev-ops
- Michał Gałęzyka - mobile, mobile-dev-ops

3. Opis funkcjonalny systemu:

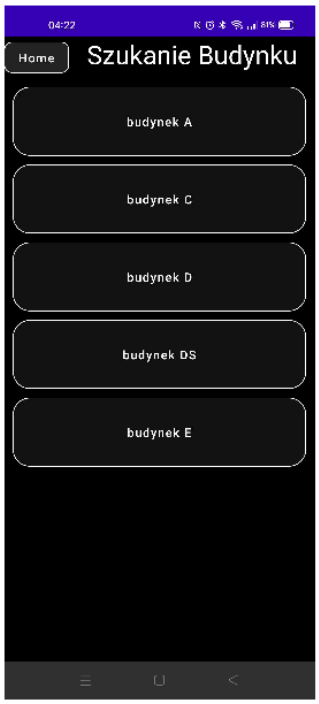
Część api, odpowiedzialna jest za pobranie danych z strony harmonogramu uczelni, dane te są odpowiednio sformatowane i umieszczone w bazie danych. Ze względu na sposób działania wystawiania api w symfony, baza danych jest wymagana. Dane te potem wystawiane są w odpowiednich endpointach w wcześniej ustalonym formacie. Oczywiście występują pewne wyjątki, które są konieczne ze względu na naturę tego jak przechowywane są dane w bazie danych oraz na to jak działa swagger.
Api zgodnie z ustalonym harmonogramem, np. raz na dzień pobiera dane z strony uczelni, wykonuje odpowiednie funkcje oraz algorytmy a następnie wystawia endpointy przy pomocy rest api.
Część mobilna jak i część web jest odpowiedzialna za pobranie odpowiednich danych poprzez odpytywanie endpointów. Następnie te dane muszą zostać przetworzone tak aby język danej części był w stanie je rozczytać. Następnie dane te są wyświetlane zgodnie z wybranym przez użytkownika widoku.
Dzięki obecności bazy danych, użytkownik nie musi odczuwać każdego działania scrapera, gdyż dane, które mają być udostępnione zawsze są pod ręką.

4. Technologie oraz narzędzia:

- **Api**
    - Symfony 6.1 - jest to framework wspierający język PHP, dzięki któremy jesteśmy w stanie postawić ten projekt bez potrzeby pisania wielu rzeczy od nowa. Wybraliśmy Symfony gdyż jesteśmy z nim najbardziej zapoznani.
    - Docker - umożliwia nam łatwe pakowanie, dostarczanie oraz uruchamianie aplikacji w formie lekkiego, przenośnego i samowystarczalnego kontenera. Co pozwala nam to uruchomić szybko i praktycznie wszędzie.
    - Api Platform - jest paczką, która pozwala na wygodne i szybkie wystawienie rest api. Bez tego prace zostały by wydłużone bez żadnego dobrego powodu.
    - Goutte - jest paczką pozwalającą nam na postawienie samego scrapera. Jest to niezbędna część tego projektu, bez niej postawienie scrapera w php byłoby o wiele bardziej obciążające jeśli chodzi o moc obliczeniową (Symfony Panther) albo wymagałoby olbrzymiej pracy na około aby obejść prostotę systemu (Guzzle)
    - Doctrine - jest paczką pozwalającą nam w wygodny sposób pracować nad entity. Dzięki czemu jesteśmy w stanie wprowadzić poprawne dane w formie api.
    - Postgres - baza danych, którą można opisać jako zaawansowany system do zarządzania relacyjnymi bazami danych, do tego jest to projekt open source.
- **Mobile**
    - Kotlin - jest językiem przystosowanym pod pisanie aplikacji mobilnych na androida. Jest używany przez większość profesjonalistów zajmujących się tym tematem więc jest to dość oczywisty wybór jeśli bierzemy pod uwagę aplikację, działającą na tym systemie. Poza tym na zajęciach na uczelni mieliśmy się okazję z nim zapoznać więc była to doskonała okazja aby poszerzyć wiedzę na jego temat.
    - Retrofit - jest paczką dla Kotlin’a, która ułatwia pracę z api. Pozwala ona na uproszczenie kodu z jednoczesnym

zwiększeniem wydajności. Zwalnia też to programistę z konieczności pamiętania o wszystkich szczegółach, co pozwala ograniczyć występujące błędy.
    - Jetpack Compose - jest paczką dla Kotlin'a upraszczająca tworzenie na nim interfejsu użytkownika. Skraca ona czas wymagany na utworzenie interfejsu jednocześnie pozwalając na uproszczenie kodu i zwiększenie wydajności
    - Gson - biblioteka Kotlin do konwersji obiektów Kotlin'owych na odpowiednik JSON, a także ciąg JSON na równoważny obiekt Kotlin'owy.
    - kizitonwose/Calendar - paczka pozwalająca na tworzenie gotowych widoków kalendarza przy użyciu elementów Jetpack Compose
- **Web**
    - Node.js - jest środowiskiem pozwalającym na łatwiejsze tworzenie aplikacji webowych korzystających z javascript. Wspiera to najpopularniejsze frameworki jak React, Vue czy Angular. Wspiera ono wiele platform, w tym te najbardziej popularne jak windows czy linux.
    - React - jest biblioteką języka javascript, wykorzystywana do tworzenia frontendowych aplikacji. Składa się ona z małych, oddzielonych od siebie elementów. Pozwala ona na utworzenie dynamicznych aplikacji web, które skalują się w zależności od urządzenia.
    - FullCalendar - jest biblioteką javascript, która bez problemu działa z aplikacjami opartymi na najbardziej popularnych bibliotekach jak Ract, Vue czy Angular. Wspomaga ona tworzenie aplikacji, które wykorzystują wyświetlanie danych w stylu kalendarzowym.

5. Zrzuty Ekranu z Interfejsu Użytkownika:

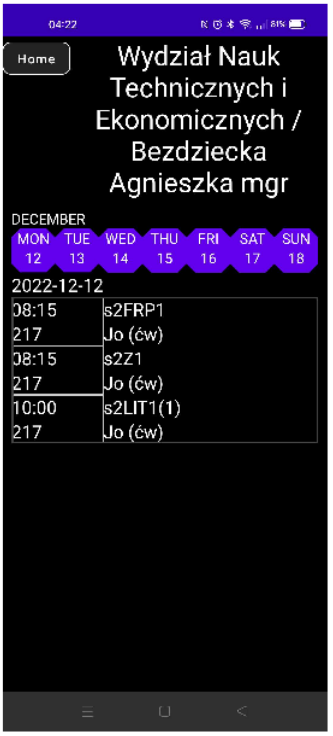

Home
Szukanie Budynku
budynek A
budynek C
budynek D
budynek DS
budynek E


Home
budynek C
021 (Kuchnia)
10
11
110
111
113
122


Home
budynek C / 11
kalendarz
MON 9
TUE 10
WED 11
THU 12
FRI 13
SAT 14
SUN 15
2023-01-09
13:30 Rewak Krzysztof mgr inż. s2INF1(1) / Pipo (lab)
15:15 Nadybski Piotr dr inż. s2INF1(1) / Sk (lab)
17:00 Nadybski Piotr dr inż. s2INF1(2) / Sk (lab)
2023-01-10
17:00 Żmuda Marcin mgr inż. s2INF2(1) / Sk (lab)
2023-01-11
17:00 Sadowski Krystian mgr inż. s3GK1 / Msi (lab)
18:45 Sadowski Krystian mgr inż. s3PAM1 / Msi (lab)
2023-01-12
08:15 Hojniak Roman mgr inż. s1INF1(1) / Pgk (lab)
10:00 Hojniak Roman mgr inż. s1INF2(1) / Pgk (lab)


Home
Szukanie Pracownika
Wydział Nauk o Zdrowiu i Kulturze Fizycznej
Wydział Nauk Technicznych i Ekonomicznych
Wydział Nauk Społecznych i Humanistycznych
Wychowanie fizyczne
Erasmus


Home
Wydział Nauk Technicznych i Ekonomicznych
Andrzejczyk Paweł mgr inż.
Bednarz Nadine mgr
Bernatek-Zaguła Izabela dr
Bezdziecka Agnieszka mgr
Bonarski Piotr mgr inż.
Wibracje przy połączeniach i powiadomieniach
Burduk Anna dr hab. inż.


Home
Wydział Nauk Technicznych i Ekonomicznych / Bezdziecka Agnieszka mgr
DECEMBER
MON 12
TUE 13
WED 14
THU 15
FRI 16
SAT 17
SUN 18
2022-12-12
08:15 s2FRP1 217 Jo (ćw)
08:15 s2Z1 217 Jo (ćw)
10:00 s2LIT1(1) 217 Jo (ćw)

6. Instrukcja Uruchomienia Aplikacji:

6.1 Postawienie Lokalne API

Wymagane oprogramowanie:

- PHPStorm lub dowolny inny IDE
- Docker Desktop
- Terminal Windows (nie jest potrzebny osobno jeśli jest wbudowany w IDE jak w PHPStorm)
- Github Desktop – aby móc wprowadzać zmiany, bądź pobierać aktualizacje jeśli są potrzebne. (nie jest wymagany jeśli jest wbudowany w IDE, bądź jeśli ktoś posiada zainstalowany pakiet GIT do użytku poprzez terminal)

Jeżeli wszystko zainstalowałeś, przejdź do instrukcji poniżej:

1. Pobranie projektu z repozytorium oraz przejście do jego folderu:

git clone https://github.com/sbacanski0730/RIP-Rewak-and-PUM-API.git
cd RIP-Rewak-and-PUM-API

2. Konfiguracja pliku .env

cp .env .env.local

3. Uruchomienie kontenerów dockerowych:

docker-compose up -d –build

4. Pobranie zależności Composer:

docker-compose exec php composer install

5. Pobranie zależności npm:

docker-compose exec php npm install

6. Budowanie Assetów:

docker-compose exec php npm run dev

Migracje:

1. Utworzenie migracji:

php bin/console doctrine:migrations:diff
php bin/console doctrine:migrations:migrate

Bez migracji:

1. Utworzenie schematu bazy danych:

php bin/console doctrine:schema:update –force

2. Załadowanie zdefiniowanych wczesniej danych do bazy danych:

php bin/console doctrine:fixtures:load
Usługi:
• *api https://localhost:443/api*
• *postgres localhost:5432*
Przydatne Komendy:
1. Docker - uruchomienie kontenerów:
docker-compose up -d
2. Docker - zatrzymanie kontenerów:
docker-compose stop
3. Docker - zatrzymanie i usunięcie kontenerów
docker-compose down

6.2 Postawienie Lokalne Web:

Wymagane oprogramowanie:

- Visual Studio Code lub dowolny inny IDE
- npm
- Terminal Windows (nie jest potrzebny osobno jeśli jest wbudowany w IDE)
- Github Desktop – aby móc wprowadzać zmiany, bądź pobierać aktualizacje jeśli są potrzebne. (nie jest wymagany jeśli jest wbudowany w IDE, bądź jeśli ktoś posiada zainstalowany pakiet GIT do użytku poprzez terminal)

Jeżeli wszystko zainstalowałeś, przejdź do instrukcji poniżej:

Jak postawić środowisko testowe ?
Jeżeli wszystko zainstalowałeś, przejdź do instrukcji poniżej:
1. Pobranie projektu z repozytorium oraz przejście do jego folderu:
git clone
https://github.com/sbacanski0730/RIP-Rewak-and-PUM-Web.git
cd RIP-Rewak-and-PUM-Web
2. Uruchomienie środowiska deweloperskiego
npm start

6.3 Postawienie Lokalne Mobile:

Wymagane oprogramowanie:

- Android Studio
- Github Desktop – aby móc wprowadzać zmiany, bądź pobierać aktualizacje jeśli są potrzebne. (nie jest wymagany jeśli jest wbudowany w IDE, bądź jeśli ktoś posiada zainstalowany pakiet GIT do użytku poprzez terminal)

Jeżeli wszystko zainstalowałeś, przejdź do instrukcji poniżej:

Jak postawić środowisko testowe ?
Wystarczy pobrać zawartość repozytorium oraz otworzyć jako projekt w Android Studio

7. Wnioski projektowe:

7.1 Api

Pisanie pierwszego scrapera z api może być wyzwaniem. Tak było też w tym wypadku. Zwłaszcza, że z błędnych założeń chcieliśmy na początku napisać całość bez używania bazy danych. Spowodowało to wiele opóźnień i problemów. To oraz inne problemy związane z sprzętem komputerowym jednego członka zespołu opóźniły rozpoczęcie pisania tej części projektu. Patrząc na to z obecnej perspektywy zamiast próbować tworzyć nowe rozwiązania samemu zamiast skorzystać z gotowej dokumentacji nie było najlepszym wyborem. Podeszliśmy do tego myśląc, że pozwoli to na optymalizację projektu. Pomimo tego projekt wymagał abyśmy szukali wiedzy poza standardową dokumentacją. Gdyż nie wszystko to co było potrzebne w naszym wypadku było w niej dostępne. Więc poza pisaniem kodu oraz korzystaniem z dokumentacji musieliśmy dokonać własnego zapoznania się z tematem. Wiele w scraperze zależy od kodu na stronie, z której informacje są pobierane. Jak mogliśmy się przekonać z planem naszej uczelni, nie wszystkie strony posiadają klasy oraz kod zgodne z dobrymi praktykami. Więc wydobywanie informacji z planu wymagało trochę twórczego myślenia związanego z zapytaniami regex oraz odpowiednimi algorytmami.

7.2 Web

Tworzenie projektu opierającego się na api z podziałem zadań na dwa podzespoły jest wymagające. Do pewnego stopnia można obsłużyć dane korzystając z wygenerowanych przykładowych plików json jednak mogą wystąpić różnice między danymi, które końcowo są wyprowadzane z api, a tymi, które znajdowały się w wygenerowanych przykładowych json’ach. Różnice zdań oraz wymagania technologiczne mogą doprowadzić do małych zderzeń między podzespołami, jednak kluczem do napisania działającego projektu jest poradzenie sobie z tym pomimo istniejących problemów. Wiele stresu mogą też powodować zbliżające się terminy, które nieubłaganie wydają się być coraz bliżej. Jednak dobra współpraca i poradzenie sobie z różnicami zdań pozwala wyjść ponad takie problemy i dokończyć projekt.

7.3 Mobile

Tworzenie projektu używając języka, z którym wcześniej nie miało się zbyt wiele do czynienia potrafi być wyzwaniem. Zwłaszcza jeśli pod uwagę wejdą opóźnienia z innych części projektu ze względu na rzeczy, których nie da się przewidzieć. Dobrą pomocą w pisaniu projektu podczas uczenia się języka jest obszerna dokumentacja oraz różnego rodzaju poradniki. Dzięki tego rodzaju pomocy pisanie projektu nie jest aż tak trudne jak mogłoby się z pozoru wydawać. Przy tym poznanie nowego języka jest dobrym doświadczeniem na przyszłość oraz może pomóc na rynku pracy

8. Repozytoria:

1) https://github.com/sbacanski0730/RIP-Rewak-and-PUM-Mobile - Właściwa Aplikacja Mobilna
2) https://github.com/sbacanski0730/RIP-Rewak-and-PUM-API - “zewnętrzne api”
3) https://github.com/sbacanski0730/RIP-Rewak-and-PUM-Web - strona internetowa