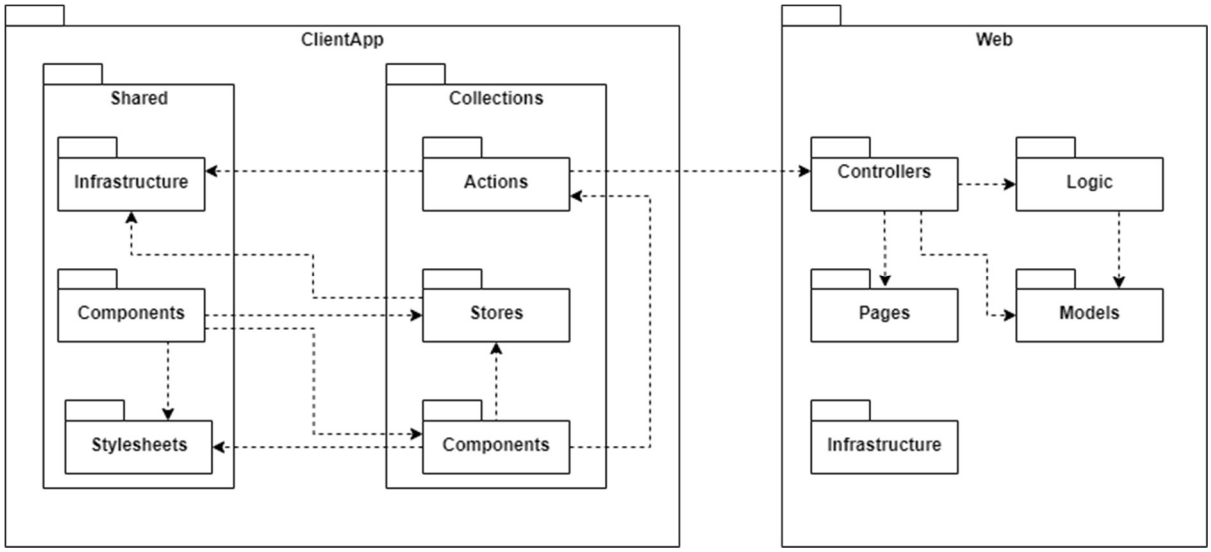
2019/20 WIZ, Informatyka **Zaawansowane technologie webowe - laboratorium**		
Rasz Arkadiusz 242493 Szałajko Karol 242557	Zajęcia nr 3: **Architektura aplikacji:** **Strona internetowa do grupowania i współdzielenia linków**	
Termin zajęć: Poniedziałek 15:15-16:45	Data oddania dokumentu: 16.03.2020r.	Ocena:

Spis treści

1. Architektura aplikacji

Diagram Pakietów



Rysunek 1 - diagram pakietów

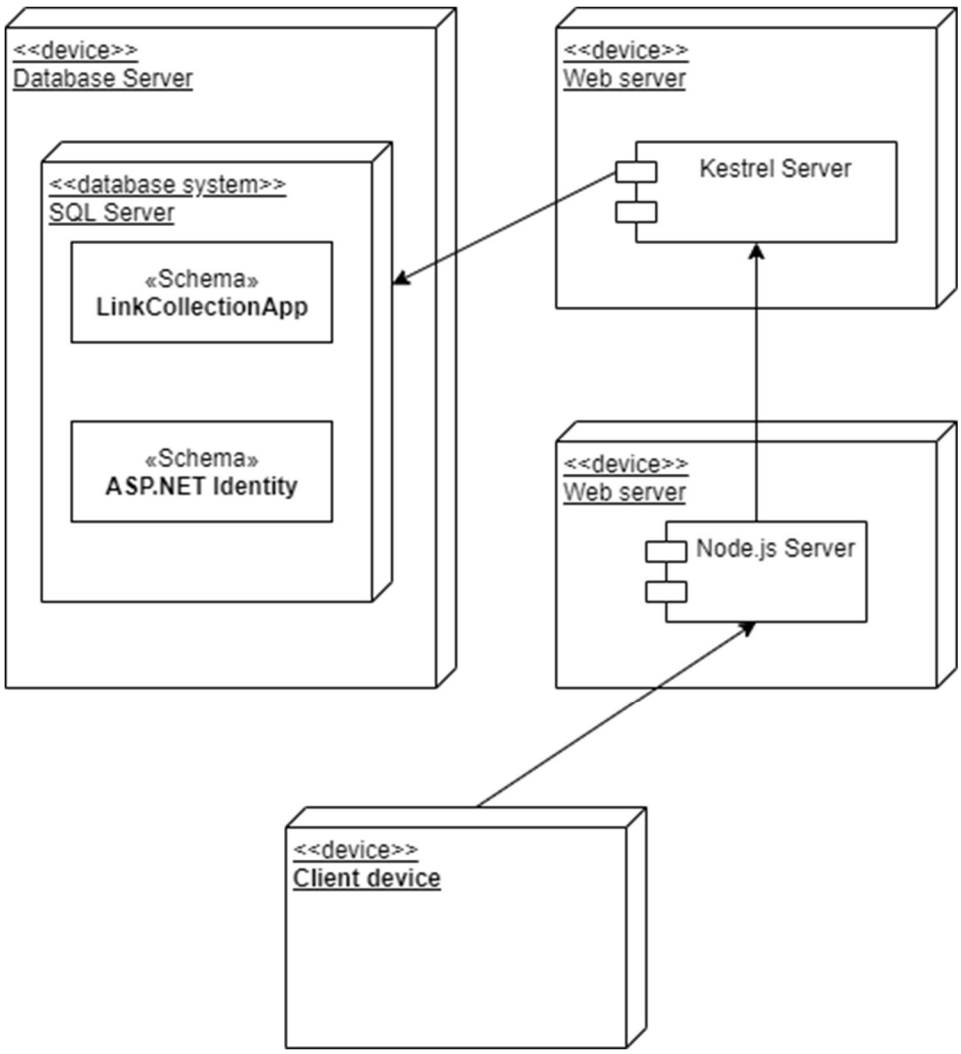
Aplikacja webowa do grupowania i udostępniania linków napisana będzie z użyciem technologii ASP.NET oraz React. Pierwsza z nich posłuży do stworzenia komponentu serwerowego wystawiającego odpowiednie metody w kontrolerach dla strony internetowej. Posłuży do komunikowania się z bazą danych przy pomocy mapowania z użyciem technologii Entity Framework oraz będzie odpowiadała za całą logikę biznesową odbywającą się w aplikacji.

Pakiet Infrastructure będzie zawierał wszelkie pliki dotyczące procesu implementacji oraz wdrażania, w tym pliki konfiguracyjne Azure Pipelines, git.

Część interfejsowa napisana będzie w React z użyciem najpopularniejszego wzorca architektonicznego tej technologii – Flux. W skrócie:

- **Actions** – Metody służące do przekazywania danych i informacji o zmianie stanów do dispatchera
- **Dispatcher** – Przyjmuje akcje i rozprowadza dane do zarejestrowanych callbacków w postaci storów
- **Stores** – Przechowują stan i logikę aplikacji, otrzymują informacje za pomocą callbacków z dispatcherów
- **Components** – Komponenty biblioteki React, które pobierają stan ze storów i przekazują je do elementów widocznych na ekranie.

Diagram rozmieszczenia



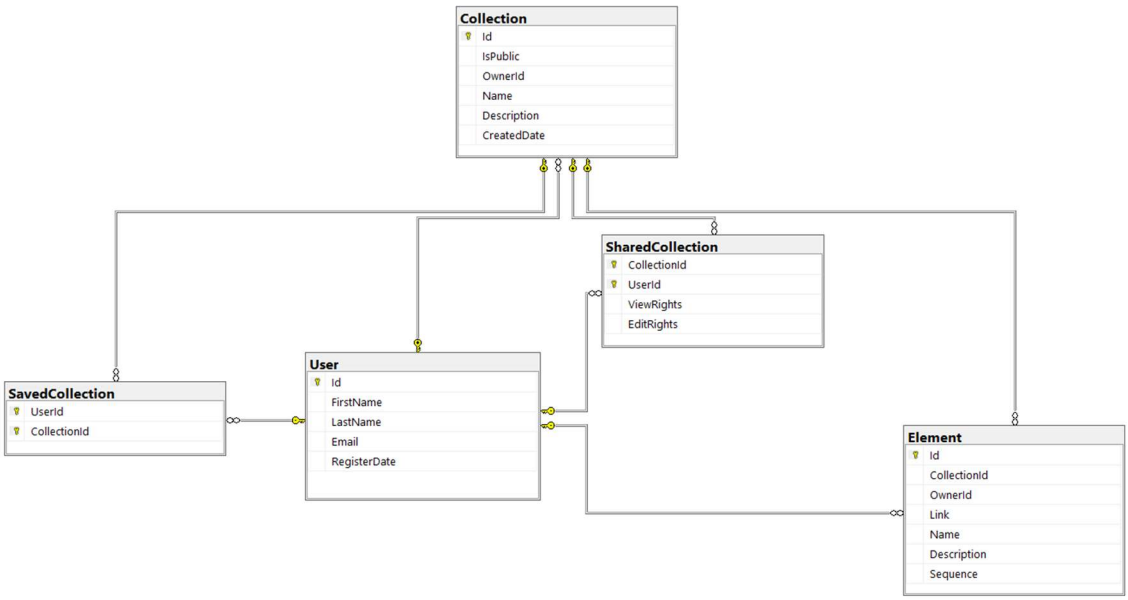
Rysunek 2 - diagram rozmieszczenia

Architektura podzielona jest na 3 moduły:

- Serwer bazy danych – Przechowuje dane o kolekcjach oraz autoryzacji użytkowników
- Kestrel Serwer – Uruchomiony serwer ASP.NET Core, opisany wyżej
- Node.js Server – Odpowiedzialny za dostarczanie interfejsu użytkownikowi oraz reagowanie na jego akcje. Wywołuje logikę biznesową na serwerze Kestrel

Wszystkie te komponenty mogą działać na niezależnych maszynach w zależności od potrzeby. W ramach zajęć, wszystkie te moduły będą działać na jednej maszynie (na środowisku implementacji oraz wdrażania). Nie jest wykluczona opcja wprowadzenia wsparcia dla kontenerów za pomocą technologii Docker.

2. Wstępny schemat bazy danych



Rysunek 3 - Schemat bazy danych

Baza danych składa się z 5 tabel:

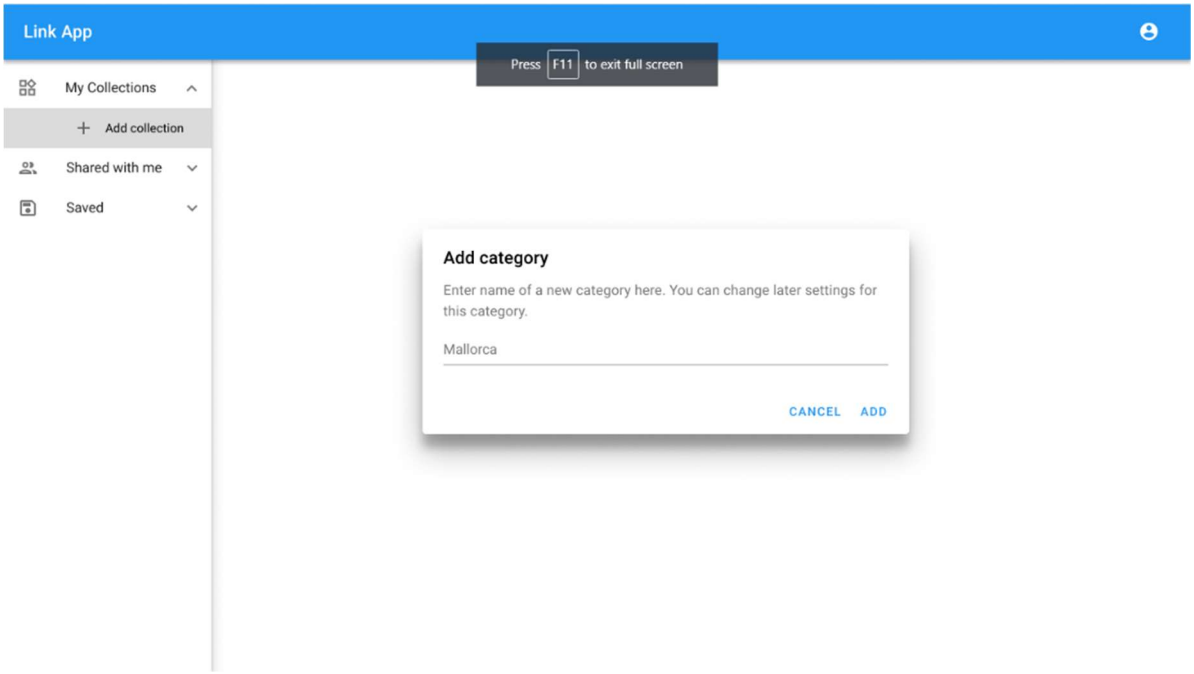
- Collection – zawiera dane dotyczące pojedynczej kolekcji (id, czy publiczna, właściciel, nazwa, opis, data utworzenia)
- Element – zawiera dane dotyczące elementu kolekcji (id, kolekcja, właściciel, link (url), nazwa, opis, numer w kolekcji)
- User – zawiera dane dotyczące zarejestrowanego użytkownika (id, imie, nazwisko, email, data rejestracji)
- SavedCollection – zawiera dane dotyczące publicznej kolekcji zapisanej przez użytkownika (użytkownik, kolekcja)
- SharedCollection – zawiera dane dotyczące uprawnień użytkowników do udostępnionej kolekcji (kolekcja, użytkownik, uprawnienia widoku, uprawnienia edycji)

Schemat ASP.NET Identity potrzebny do autoryzacji został pominięty z powodu automatycznego generowania go za pomocą narzędzi ASP.NET Core.

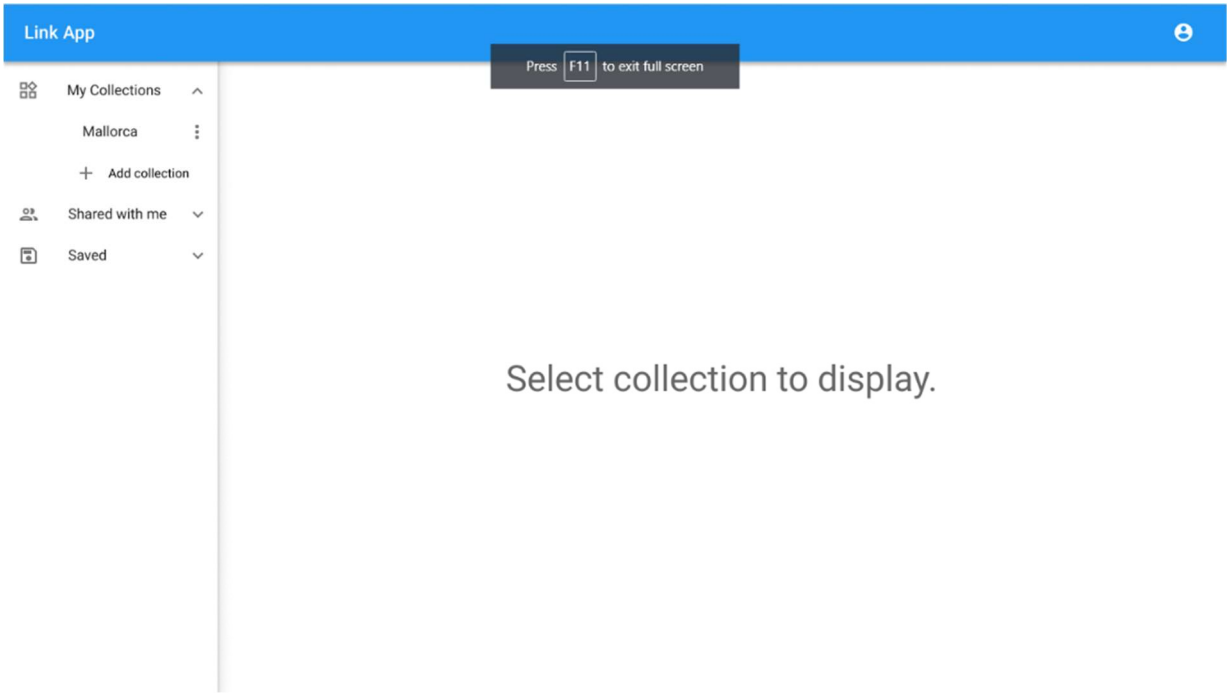
3. Mockupy aplikacji

Mockupy aplikacji w formie interaktywnej dostępne są pod adresem:

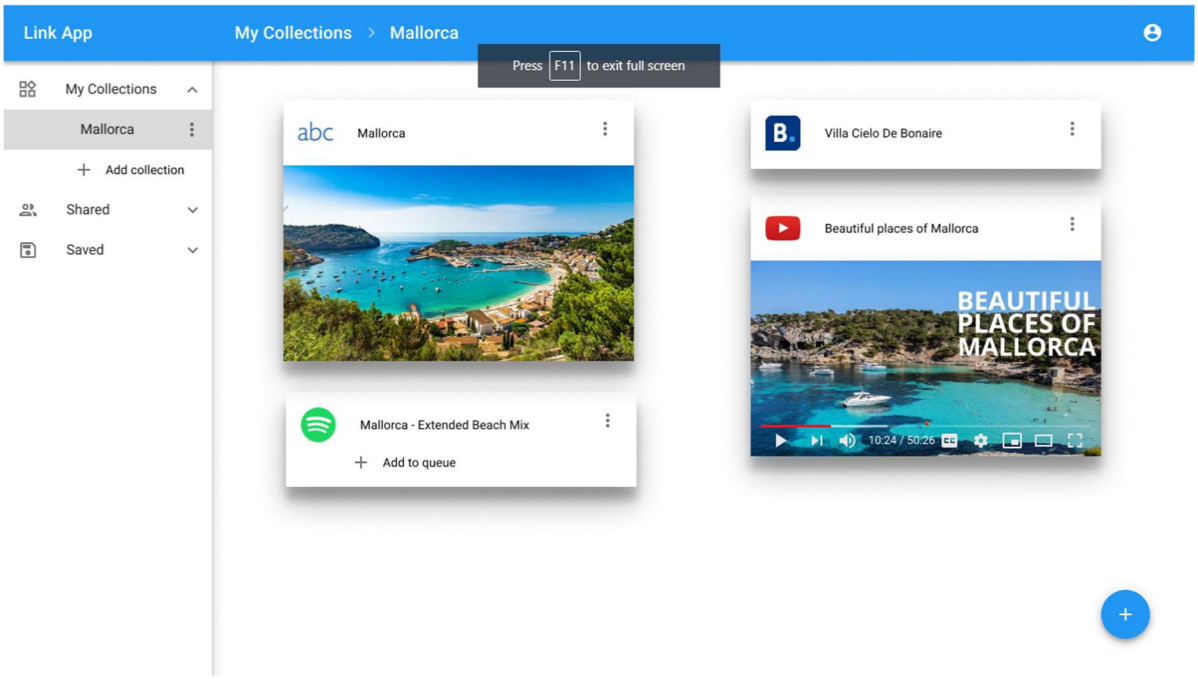
https://www.figma.com/proto/nQz8yuAn9bGBCHzuRM8q51/App_mocup?node-id=0%3A1&scaling=min-zoom



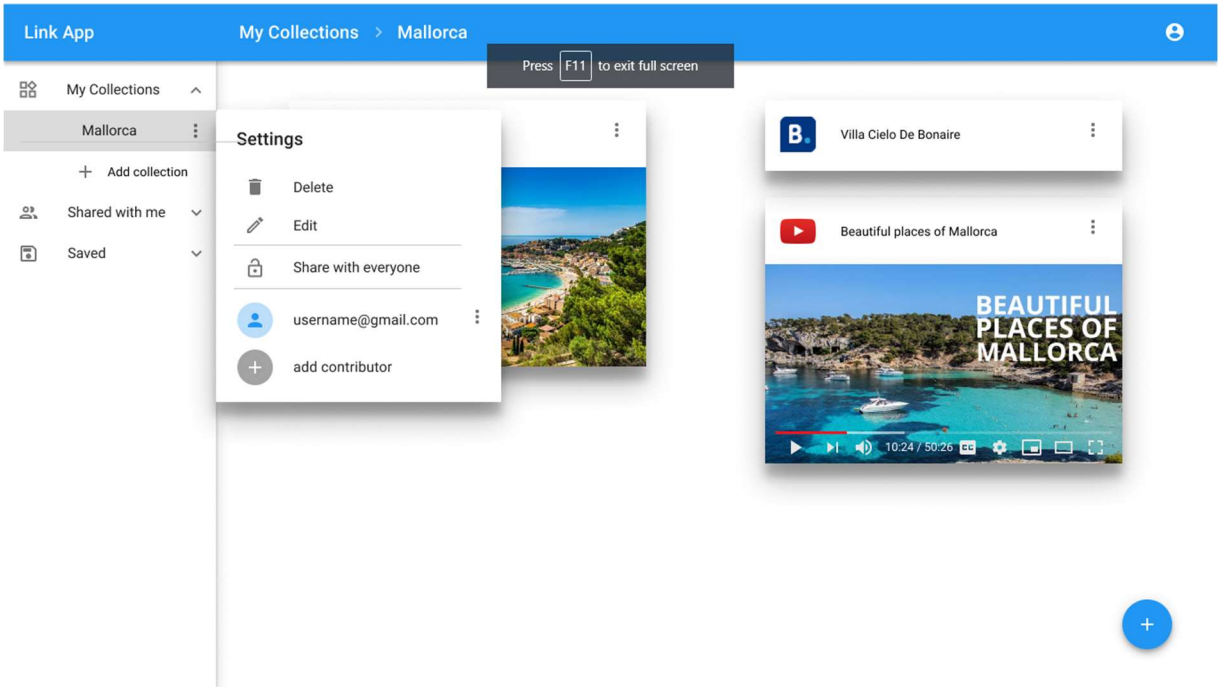
Rysunek 4 - Opcja dodawania nowej kolekcji



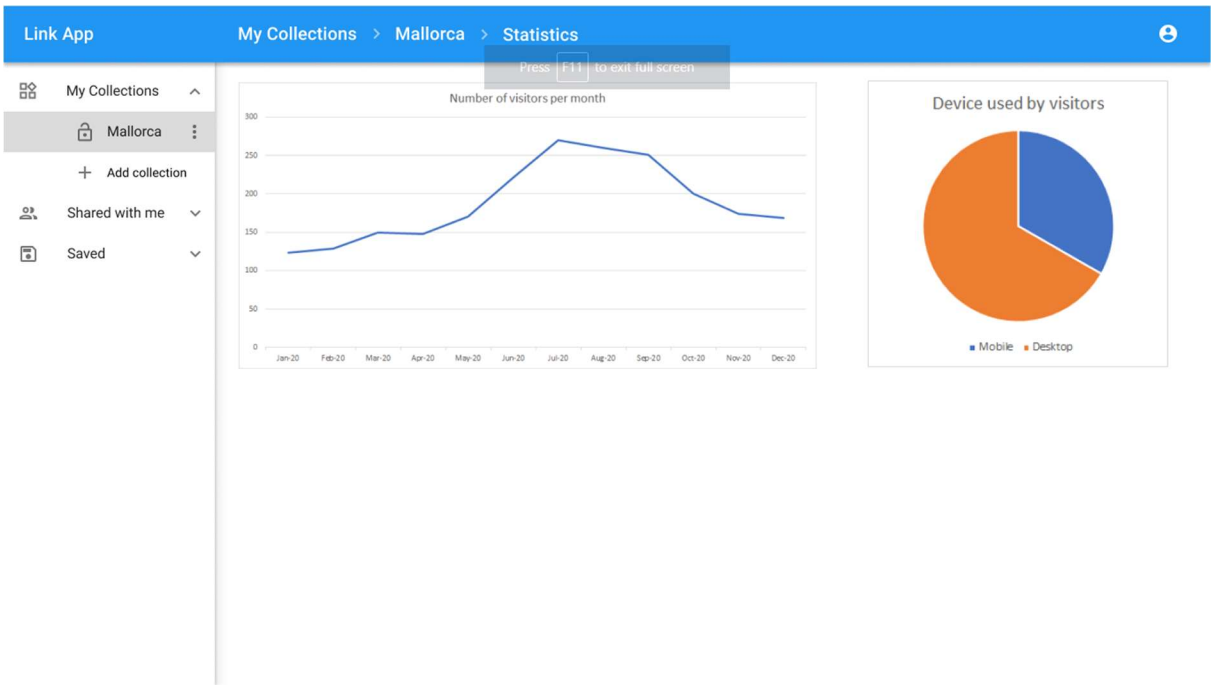
Rysunek 5 - Ekran startowy z rozwiniętym panelem “Moich kolekcji”



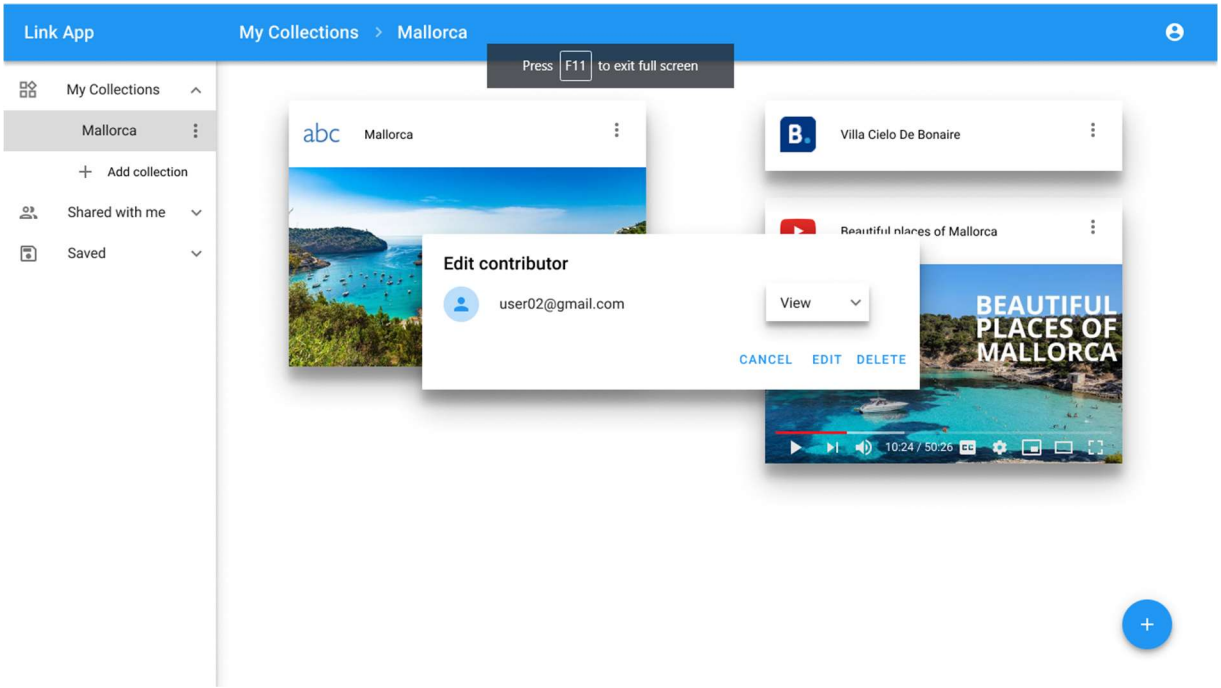
Rysunek 6 - Ekran kolekcji z wyświetlonymi elementami



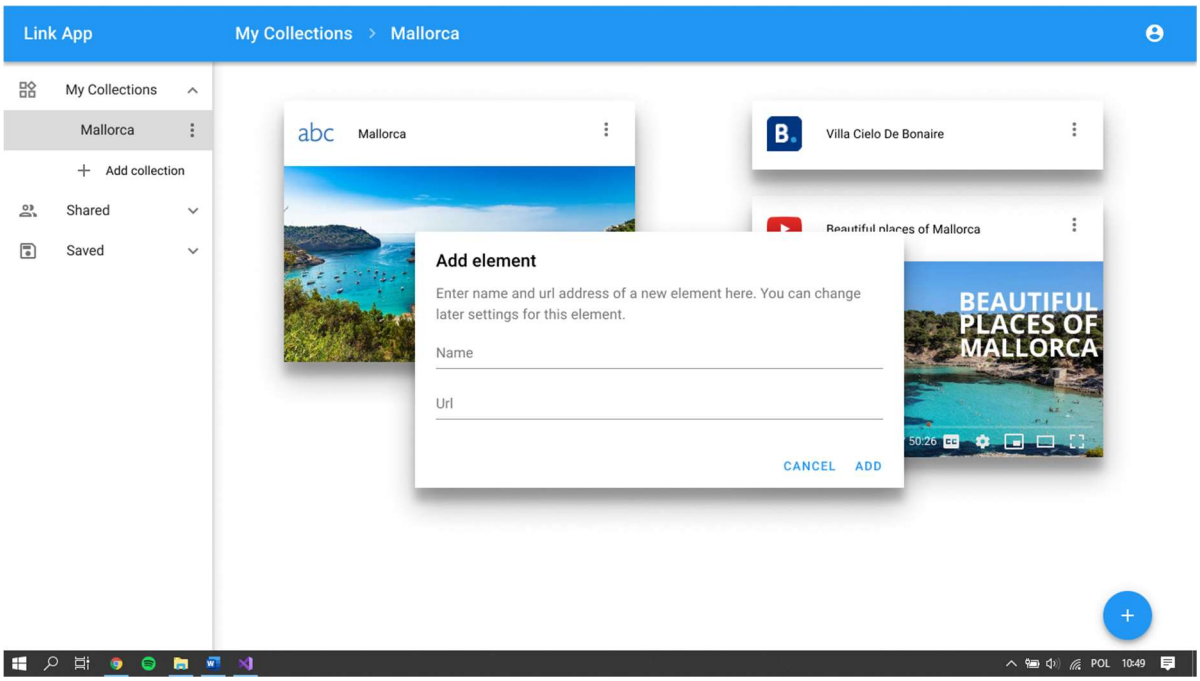
Rysunek 7 - Panel opcji kolekcji



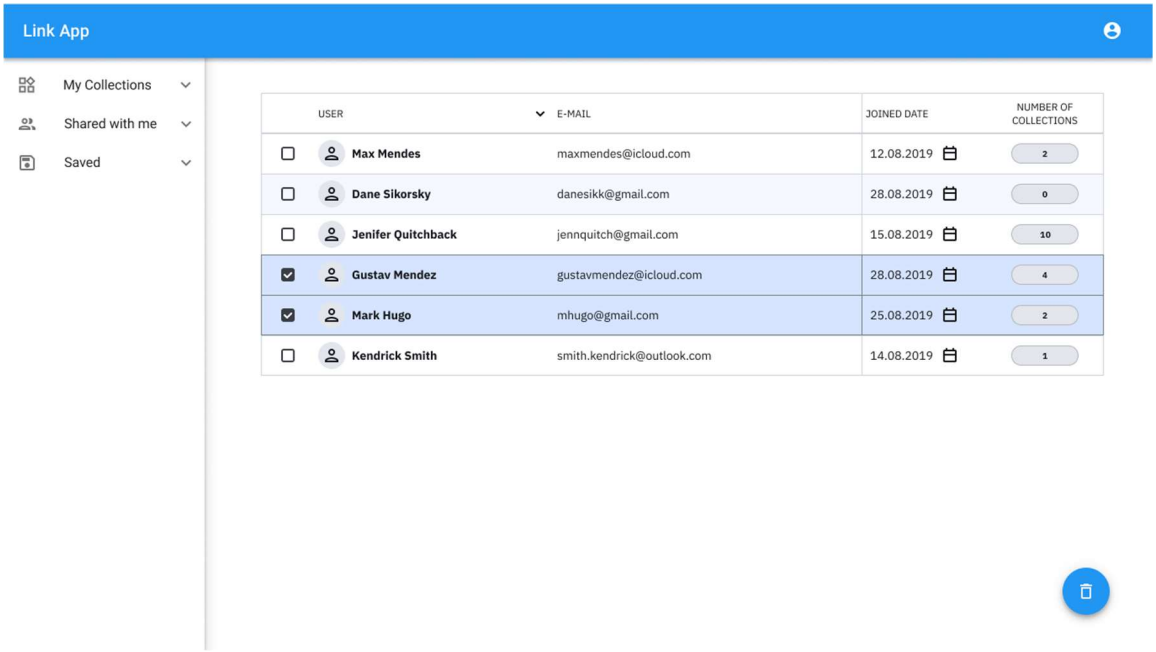
Rysunek 8 - Ekran statystyk publicznej kolekcji



Rysunek 9 - Panel edycji uprawnień współpracowników



Rysunek 10 - Panel dodawania nowego elementu kolekcji



Rysunek 11 - Ekran widoku administratora

4. Harmonogram prac implementacyjnych

Proponowany harmonogram pracy. Zadania wypisane do każdej funkcjonalności mogą być doprecyzowane podczas dalszej pracy.

Data zajęć	Funkcjonalność	Zadania
23.03.20	Autoryzacja użytkownika	Utworzenie nowej instancji SQL Server
		Implementacja autoryzacji za pomocą APS.NET Identity
		Wyświetlanie zalogowanego użytkownika po stronie interfejsu
30.03.20	Wyświetlanie prywatnych kolekcji	Utworzenie bazy danych kolekcji
		Logika zaczytywania kolekcji i elementów danego użytkownika
		Wyświetlanie listy kolekcji
		Wyświetlanie elementów kolekcji
06.04.20	Tworzenie nowej kolekcji, dodawanie elementów	Logika tworzenia kolekcji
		Logika dodawania elementów do kolekcji
		Widok tworzenia nowej kolekcji
		Widok dodawania nowego elementu
20.04.20	Udostępnianie kolekcji, przeglądanie udostępnionych kolekcji	Logika udostępniania kolekcji
		Logika pobierania udostępnionych kolekcji
		Widok udostępniania kolekcji
		Widok listy udostępnionych kolekcji
27.04.20	Edycja elementów kolekcji	Logika edycji kolekcji
		Uprawnienia użytkowników do wyświetlania / edycji
		Widok edycji elementu
04.05.20	Kolekcje publiczne, dopracowanie interfejsu	Logika publikowania kolekcji
		Wyłączenie autoryzacji dla kolekcji publicznych
		Grupowanie/filtrowanie elementów kolekcji
11.05.20	Statystyki kolekcji publicznych	Przechowywanie danych o wejściach w odnośnik
		Panel z wykresami statystyk
18.05.20	Panel administracyjny	Uprawnienia administratora
		Logika zarządzania użytkownikami
		Tabela zarządzania użytkownikami
25.05.20	Integracja z usługami Youtube oraz Spotify	Wyświetlanie odtwarzacza wideo dla odnośników YouTube
		Autoryzacja w usłudze Spotify
		Dodawanie utworów do kolejki użytkownika za pomocą Spotify API
01.06.20	Dopracowanie interfejsu użytkownika	Wykończenie stylów
		Wykończenie elementów graficznych
		Inne usprawnienia
08.06.20	Prezentacja wyników	Przygotowanie prezentacji multimedialnej oraz demonstracji aplikacji
15.06.20	Ostateczna dokumentacja	Przygotowanie dokumentacji

Równolegle z nowymi funkcjonalnościami będą tworzone odpowiednie testy oraz trwały prace nad usprawnianiem procesu CI/CD.

5. Środowisko implementacyjne

Do implementacji posłuży środowisko zintegrowane Visual Studio oraz w pewnych przypadkach Visual Studio Code dla części interfejsowej aplikacji.

Do kontroli wersji oraz synchronizacji pracy użyty jest git, z repozytorium hostowanym na platformie GitHub:

https://github.com/arkadR/Link-Collection-App.

Do usług Continuous Integration oraz Delivery posłuży platforma Azure Pipelines.
https://dev.azure.com/242493/Link-Collection-App.

Do zarządzania zadaniami oraz śledzenia postępu prac wykorzystywana jest platforma Trello:
https://trello.com/b/SNJG2yln/link-collection-app

6. Proces implementacyjny

W ramach laboratorium, wprowadzona zostanie dodatkowa konfiguracja do wykorzystywanych usług w celu poprawienia wspólnej znajomości kodu, poprawienia jego jakości oraz szybszego wyłapywania błędów.

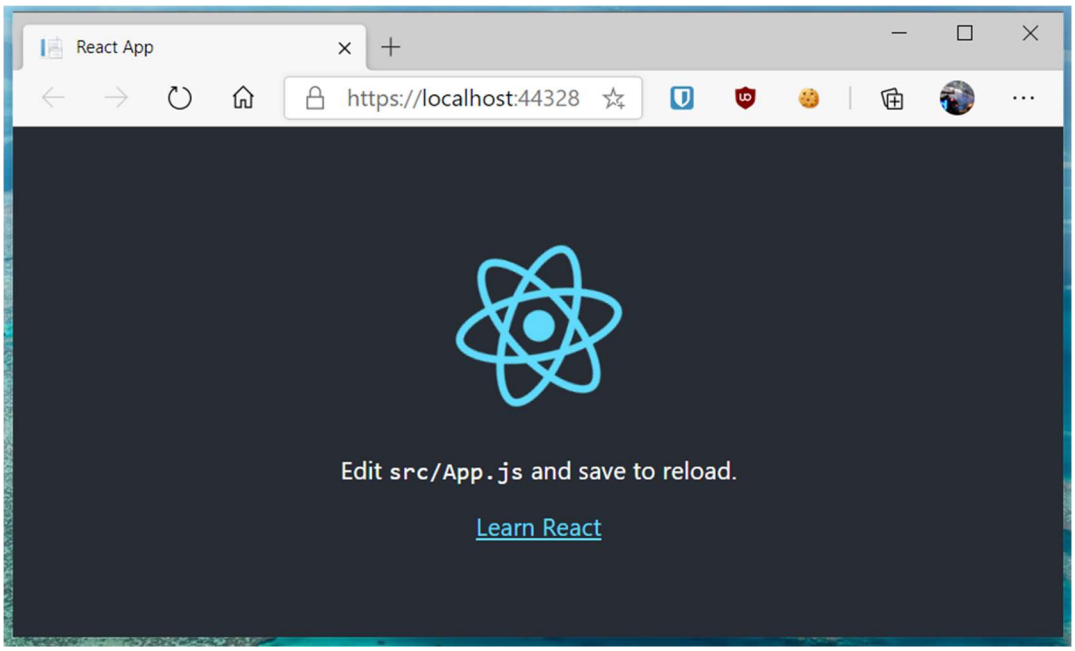
Zmiany w kodzie zachodziły będą na dodatkowych gałęziach, innych niż *master* (głównie będzie to utworzony już branch *dev*). Każda aktualizacja do *master* wymaga utworzenia nowego pull requesta. Pull request może być zaakceptowany po zatwierdzeniu zmian w kodzie przez drugą osobę oraz zaliczenia wszystkich testów na platformie Azure Pipelines.

Budowa oraz uruchomienie testów na platformie Azure Pipelines dzieje się po każdym commicie do dowolnej gałęzi repozytorium oraz utworzenia nowego pull requesta na gałęź master. Zbudowana aplikacja będzie wdrażana na środowisko Azure po każdym zaakceptowanym pull request na gałąź master.

7. Postęp pracy od ostatnich zajęć

Udało nam się utworzyć nowy projekt ASP.NET Core z częścią interfejsową z użyciem React. Dodaliśmy wsparcie dla języka TypeScript dla zalet statycznego typowania. Zainstalowaliśmy dodatkowe biblioteki z repozytoriów NuGet oraz npm, które z pewnością zostaną przez nas używane podczas pracy.

Aplikacja buduje się na naszych środowiskach, na razie jest to domyślna aplikacja z szablonu create-react-app:



Skonfigurowaliśmy platformą GitHub jak opisano wyżej.

Azure Pipelines buduje projekt oraz uruchamia testy po każdym nowym commicie. Wdrażanie zbudowanej aplikacji nie jest jeszcze skonfigurowane.

Na platformie Trello utworzyliśmy wstępne zadania oraz skonfigurowaliśmy ich znaczniki.