Projekt aplikacji do organizowania pracy

Aleksander Kowalski, Grzegorz Bogusz, Kacper Frankowski

Legnica, May 2021

Spis treści

1 Wstęp

1.1 Założenia aplikacji

Trello Copy to aplikacja działająca w przeglądarce. Jej głównym celem jest pomoc w organizacji pracy za pomocą tworzenia tablic, zakładek oraz zadań. Użytkownik może dzielić tablice poprzez dodawanie do nich innych użytkowników. Aplikacja jest wysoce inspirowana znanym serwisem Trello.

1.2 Opis użytych technologii

Back-end aplikacji jest oparty o framework Laravel, który został wykorzystany do stworzenia API. Front-end jest oparty o bibliotekę React. Konsumuje on API, które jest z kolei hostowane na osobnym kontenerze z laravelem. Aplikacja jest skonteneryzowana, czyli każda instancja, np. back-end, jest osobnym kontenerem. Całość jest spięta w jedną sieć, ponieważ umożliwia to bezpieczną komunikacja pomiędzy kontenerami, aby to osiągnąć posłużyliśmy się narzędziem Docker. Aplikacja posiada także CI/CD, które jest oparte o github actions. Pomaga nam to zautomatyzować testowanie i deploy po commit'cie, na serwer docelowy. Back-end i zarówno Front-end posiadają testy funkcyjne, czyli testujemy wybrane funkcje serwisu i sprawdzamy po kodzie http czy odpowiedź jest poprawna i czy dana funkcjonalność działa.

2 Specyfikacja projektu

2.1 Zastosowanie aplikacji według roli

W aplikacji użyto trzech typów użytkowników:

- użytkownik niezalogowany, inaczej gość, osoba przed uwierzytelnieniem,
- użytkownik zalogowany, czyli osoba uwierzytelniona,
- administrator,

2.1.1 Użytkownik niezalogowany

Użytkownik niezalogowany widzi ekran logowania, blog i publiczne tablice. Logowanie odbywa się poprzez uprzednie zarejestrowanie na osobnej podstronie, potwierdzenie swojego maila, a następnie zalogowanie się. Jeżeli użytkownik wprowadzi złe dane to zostanie o tym powiadomiony. Jeżeli użytkownik wprowadzi odpowiednie dane to zaloguje się do aplikacji i będzie mógł z niej korzystać.

2.1.2 Użytkownik zalogowany

Użytkownik zalogowany może wykonywać takie czynności jak:

- tworzenie tablicy, zakładek i zadań,
- tworzenie zespołów i przypisywanie ich do tablic,
- korzystanie z cudzej tablicy jeżeli użytkownik został przypisany do teamu.

2.1.3 Administrator

Administrator może wykonywać takie czynności jak:

- tworzenie użytkowników, zarządzanie nimi oraz usuwanie ich,
- tworzenie pakietów, zarządzanie nimi oraz usuwanie ich,
- tworzenie tablic, zarządzanie nimi oraz usuwanie ich,
- tworzenie artykułów, zarządzanie nimi oraz usuwanie ich,
- tworzenie kategorii i typów artykułów, zarządzanie nimi oraz usuwanie ich,

3 Instrukcja uruchomienia

3.1 Lokalnie

Instrukcja uruchomienia aplikacji lokalnie znajduje się pod adresem:

https://github.com/PHReactive/trello$_{c}opy/tree/docker$

3.2 Zdalnie

Aplikacja jest dostępna w sieci pod linkiem:

http://azyl.tech:8185/login

W celu korzystania z aplikacji, należy się do niej zalogować.

W celu zalogowania się do panelu administracyjnego wystarczy wpisać te dane w panelu logowania:

Email: admin@admin.pl

Hasło: admin

4 Wnioski projektowe

4.1 Wstęp

Podczas wybierania technologii zdecydowaliśmy się na framework Laravel / PHP na back-endzie, a na front-endzie zdecydowaliśmy się na bibliotekę React / JS TS. Wybraliśmy Laravel z racji, iż jest to jeden z dużych i dobrych frameworków, który narzuca programiście swoją strukturę, nazewnictwo i wiele innych rzeczy, co czyni projekt bardzo czytelnym, niż w porównaniu do np. Yii. React wybraliśmy z uwagi na to, iż jest to jedna z wielu popularnych bibliotek do JS , co czyni ją łatwą do nauki. Dodatkowo, wsparcie sporej społeczności React'a pozwala na szybkie rozwiązywanie napotkanych problemów. Także, dzięki React'owi można było stworzyć aplikacje SPA bez większych nakładów pracy.

4.2 Baza danych

W trakcie projektowania aplikacji zdecydowaliśmy się na MySQL, ponieważ jest to najbardziej popularny system do zarządzania relacyjną bazą danych. Jego zalety przewyższają wady, a na potrzeby tego projektu w pełni to zdaje egzamin.

4.3 Konteneryzacja aplikacji

Do konteneryzacji aplikacji wykorzystaliśmy Dockera, z uwagi na jego popularność wśród systemów do konteneryzacji aplikacji. W jego sieci są obrazy do każdego możliwego języka, co czyni te narzędzie "easy to learn hard to master". W wielkim skrócie, aby już zacząć swoją przygodę z Dockerem, wystarczy napisać prosty plik "docker-compose", który pobierze i skonfiguruje dany obraz. Jeżeli chcemy tworzyć własne obrazy to tutaj przyda się znajomość powłoki Bash, ponieważ Docker jest oparty na jądrze linuxowym. Docker jest dostępny na każdym systemie, co czyni go uniwersalnym i pozwala pracować wielu osobom przy projekcie bez względu na używany przez nich system.

4.4 Back-end

W back-endzie aplikacji wykorzystaliśmy Laravel, ponieważ już wcześniej mieliśmy z nim do czynienia. Jest to jeden z popularnych i dobrych frameworków na rynku. Posiada bardzo rozbudowaną społeczność, co czyni go łatwym do nauki i nie tylko. Posiada także bardzo dużą bibliotekę dedykowanych funkcjonalności, co przyśpiesza znacznie prace nad projektem.

4.5 Front-end

Po stronie front-endu wykorzystaliśmy jeden z najpopularniejszych frameworków na rynku, czyli React. Technologia ta była wspierana Reduxem, który był odpowiedzialny za zarządzanie stanem aplikacji, oraz Tailwincdcss który znacznie wspomógł proces działania z wyglądem. Dodatkowo wykorzystane zostały mniejsze biblioteki do konkretnych celów, np. tworzenie modali, fetchowanie api. Wybory te wynikły z powodu doświadczenia w pracy z tymi technologiami, czy to w poprzednich projektach lub też zawodowo.

4.6 Procesy CI-CD

W projekcie wykorzystaliśmy procesy CI/CD w celu zautomatyzowania pewnych powtarzalnych czynności takich jak deploy aplikacji na serwer docelowy i testowanie aplikacji po każdym commit'cie. Dzięki zaoszczędzonemu czasu mogliśmy przyjrzeć się bardziej tym procesom i to co udostępnia github actions, na których oparliśmy ten proces.

4.7 Autoryzacja

Autoryzacje oparliśmy o token JWT, ponieważ posiadamy dwie niezależne od siebie instancje - back-end i front-end. Dzięki temu standardowi umożliwiamy bezpieczną autoryzacje użytkownika i łatwe zarządzanie uprawnieniami szczególnie, iż Laravel posiada dedykowany pakiet JWT, więc nie trzeba wymyślać koła na nowo co przyspiesza prace nad projektem.

4.8 Podsumowanie

Była to kolejna realizacja aplikacji webowej oraz projekt zespołowy. Dzięki temu projektowi utrwaliliśmy sobie rzeczy poznane semestr wcześniej, czyli framework Laravel, React, Docker i wiele innych. Aplikacja jest wykonana lepiej, niż aplikacja prezentowana semestr wcześniej, ponieważ wyciągnęliśmy wnioski z własnych błędów, a w wolnej chwili przećwiczyliśmy sobie braki, które można było zauważyć w poprzednim projekcie. Wykorzystaliśmy także nowe technologie, które używa się na co dzień w firmach np. CI / CD. Nauczyliśmy się również pisać o wiele lepsze obrazy w Dockerze i spinać to w sensowną całość, natomiast zdajemy sobie sprawę z tego, że czynności takie jak skonfigurowanie projektu lokalnie można było skrócić do jednej komendy wykorzystując funkcje Composera. Dzięki temu projektowi podszkoliliśmy się w pisaniu API, z którego później korzysta frontendowiec. To wymusiło na nas dobrą komunikacje, dokumentowanie pewnych procesów i zaplanowanie wszystkiego od góry. Jesteśmy świadomi, że architektura projektu pozostawia wiele do życzenia, lecz jesteśmy pewni, że następne projekty będą tylko lepsze. Projekty zespołowe są jedną z lepszych części studiów, ponieważ oprócz szlifowania umiejętności twardych, szlifujemy również umiejętności miękkie, co w przyszłości zaowocuje w pracy.