Projekt R.E.W.A.K - Bieddit

JAKUB ENGIELSKI, SZYMON MICHNO, JAN KWIATKOWSKI

Maj 2021

1 Opis funkcjonalny systemu

Celem Projektu R.E.W.A.K było stworzenie systemu bazowanego na reddit.com oraz wykop.pl nazwany przez nas jako Bieddit, co jest połączeniem słowa biedny i reddit. Po wejściu na stronę naszego systemu użytkownik przywitany będzie informacją o tym, że aby uzyskać dostęp do dalszych funkcji Bieddita musi założyć konto bądź zalogować się na już istniejące. Użytkownicy, już zalogowani, zostaną automatycznie przeniesieni na stronę dla zalogowanych użytkowników po wykryciu, że istnieje ich sesja, korzystając z middleware User na podstawie Auth. Dalej użytkownicy otrzymają możliwość wybrania na który board, bądź po polsku Tablicę się udać. Jest ona generowana automatycznie korzystając z zapytania do bazy danych. W danej tablicy użytkownicy są w stanie przeglądać istniejące już posty, dodać własny post, bądź sprawdzić czy dany post ma komentarze bądź dodać własny komentarz. Jest też specjalna wersja administracyjna. Użytkownik z odpowiednimi uprawnieniami może zamienić /home na /admin i uzyskać dostęp do funkcji moderacyjnych. Przy pomocy Autha sprawdzane są uprawnienia i jeśli zostanie wykryte, iż użytkownik nie posiada odpowiednich uprawnień zostaje odesłany na stronę główną z informacją, iż ma niewystarczające uprawnienia. Administracja ma możliwość usuwania oraz edytowania postów oraz usuwania komentarzy. Dodatkowo pobieramy i wystawiamy API. Pod końcówką /api/bieddit_info wystawiamy nasze api, w którym pokazujemy liczbę użytkowników oraz informacje o tablicach stworzonych w Bieddicie. Natomiast pod końcówką /home/cryptoprice pobieramy średnią wymianę BTC na dolary.

2 Wyszczególnione zagadnienia z punktu 1.

- HTML5 – Skoro używamy strony internetowej jako view naszego projektu jest to dość oczywiste, że korzystamy z HTML5 do budowy całego view. Kod HTML znajduje się w blade'ach, które są wykorzystywane do wyprowadzania danych.
- CSS3 – W większości wykorzystujemy bazowe class-y dostarczone razem z bootstrapem. Poza tym w paru miejscach używamy wewnętrznego CSS do poprawienia rozmiarów konkretnych elementów.

- Formularze są wykorzystywane w wielu miejscach. Dokładniej przy systemie logowania/rejestracji oraz dodawania/modyfikacji postów oraz komentarzy.
- Baza danych – połączenie z bazą danych wykorzystywane jest do przechowywania danych użytkownika oraz wszelkich informacji o postach oraz komentarzach.
- Router – korzystamy z routingu aby zwiększyć bezpieczeństwo naszego serwisu. Dzięki temu rzeczywiste pliki nie są narażone i wyświetlane są tylko z góry określone adresy. Dzięki laravelowi znajduję się to w web.php.
- Uwierzytelnianie – Już parę razy wspomniane ale korzystamy z modelu View, Model, Controller z odpowiednio ustawionym Validatorem aby umożliwić naszym użytkownikom zakładanie kont oraz logowanie. Jest to niezbędne, gdyż korzystamy także z autoryzacji więc bez tego nie ma dostępu do żadnych funkcji naszego serwisu.
- MVC – jest wykorzystywany głównie przy rejestracji i logowaniu. Blade jest View – widokiem, Controller – używamy controllerów w których określone są sposoby łączenia z bazą danych, zapytania oraz jakie informacje tam przesyłać i z tamtego miejsca pobierać, np. RegisterController.php bądź LoginController.php, Model – w tym wypadku będzie to User.php, w którym określone są pola, które są wypełniane, te które są ukryte oraz te które są z góry castowane do konkretnego natywnego typu danych
- CRUD – w tym wypadku omówię to na przykładzie postów. Każdy z użytkowników może dodawać posty oraz komentarze, jednak tylko administracja jest w stanie je edytować i usuwać. Więc spełnia to wymagania CRUD.
- ORM – są to rzeczy w modelu User.php, które określają w jaki sposób określane są dane pola.
- Wystawienie API – jak wspomniane w opisie wystawiamy dość proste api w route /api/bieddit_info gdzie wyświetlamy liczbę użytkowników oraz dane na temat naszych tablic.
- Konsumowanie API – konsumujemy api w route /home/cryptoprice z https://github.com/binance/binance-spot-api-docs gdzie wyświetlamy obecną średnią kwotę wymiany BTC na USDT.
- Mail – maile z systemu wysyłane są w wypadku gdy użytkownik zapomni hasła i chce skorzystać z funkcji Przypomnienia Hasła. Daje to link, który pozwala na jego zresetowanie.
- Lokalizacja – Obecnie udostępniamy wsparcie dla użytkowników Polsko i Anglo-języcznych. Więc zależnie od wykrytego języka systemowego wyświetlona zostanie odpowiednia wersja strony.

- RWD – umożliwiamy tą funkcję korzystając z Bootstrap'a dzięki czemu nasza strona skaluje się poprawnie na wszystkich urządzeniach.
- Logger – korzystamy z wbudowanego loggera w Laravelu i zapisujemy w pliku logów id użytkownika i kiedy wchodzi na naszą stronę, funkcja do tego znajduje się w middleware User.php.
- System Zarządzania Zależnościami – wykorzystujemy wbudowany w Laravel system korzystający z pliku composer.json.
- SEO – zoptymalizowaliśmy to pod kontem dodania odpowiednich meta tagów.

3 Streszczenie opisu technologicznego

Korzystamy z framework'u Laravel wspierającego język PHP, który udostępnia nam krytycznie niezbędne funkcje wymagane do poprawnego działania naszego projektu, dzięki czemu byliśmy w stanie tak sprawnie go stworzyć. Technologii do bazy danych mysql dzięki, której jesteśmy w stanie wykorzystać w sprawny i poprawny sposób implementację Bazy Danych do naszego serwisu. Bootstrap aby zapewnić skalowalność strony oraz łatwość przypisywania klas Bootstrapowych aby szybko i sprawnie utworzyć layout strony. TinyMCE 5 aby umożliwić formatowanie tekstu, dzięki czemu mamy gotowe zaimportowane rozwiązanie którego nie musimy przygotowywać sami. Docker Desktop, który pozwala nam tworzyć lokalne środowisko testowe dzięki czemu byliśmy w stanie sprawdzać nasz progres bez wymagania rzeczywiście wykupionego serwera. I bez ciągłego przesyłania na niego plików. IDE PHPStorm, które wyświetlało nam wewnętrzne błędy oraz udostępnia wygodny wewnętrzny terminal od razu umiejscowiony w dobrym folderze. Github, na którym utworzyliśmy nasze repozytorium co pozwalało nam na stworzenie historii starych wersji oraz pracę wielu osób na wielu komputerach bez obawy o różnice w kodzie.

4 Instrukcję lokalnego i zdalnego uruchomienia systemu

4.1 Postawienie systemu lokalnie

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.

Instrukcja skopiowana z naszego repo: https://github.com/Deorganization-Team-B/Projekt-R.E.W.A.K/tree/docker

Jak postawić środowisko testowe na dockerze?
Instrukcja dla Windowsa
Instrukcja dla Linuxa

Jeżeli wszystko zainstalowałeś, przejdź do instrukcji poniżej:

1. Sklonuj repozytorium (Clone a repository):

   $ git clone https://github.com/Deorganization-Team-B/Projekt-R.E.W.A.K

2. Następnie przechodzisz do folderu sklonowanego repozytorium:

   $ cd store

3. I budujesz obraz naszego środowiska testowego:

   $ docker-compose up -d

4. Potem instalujesz zależności w projekcie za pomocą Composera:

   $ docker exec app composer install

5. Oraz generujesz w Artisanie key dla Laravela:

   $ docker-compose exec app php artisan key:generate

   Środowisko testowe będzie dostępne pod tym adresem, na localhoscie: `localhost:8080`

   Usługi:

- www port:8080
- SFTP port:2249
- phpmyadmin port:8081

Jeżeli przy budowaniu wystąpią przykładowe błędy, skorzystajcie z instrukcji poniżej:

Błąd timeout:

$ export COMPOSE_HTTP_TIMEOUT=120
$ export DOCKER_CLIENT_TIMEOUT=120

Nie można połączyć się z kontenerem laravela poprzez ssh:

1. Szukamy kontener z laravelem i kopiujemy jego id

   $ docker ps

2. Wchodzimy do shella kontenera

   $ docker exec -it ¡container id¿ /bin/bash

3. Restartujemy usługę ssh

   $ service ssh restart bash

4.2 Postawienie systemu zdalnie

W zależności od miejca tworzenia naszej instancji aplikacji internetowej (bezpośredni hosting stron www, serwer www czy w przygotowanym przez nas kontenerze na serwerze dedykowanym) sposób instalacji będzie się różnił ale w każdym przypadku będziemy musieli posłuzyc się narzędziami do transferu plików (FTP) i przesyłania komend do serwera (SSL) w celu wykonania zadań na serwerze.

W przypadku instalacji na maszynie dedykowanej możemy zainstalowac serwer Apache lub uprzednio przygotować kontener w np. w Docerze. Po instalacji serwera należy dobrać bibloteki jeżeli nie zostały uprzednio pobrane przez instalator, po czym za pomocą FTP przesłać pliki strony internetowej na serwer (w przypadku hostingu od razu przechodzimy do tego kroku czyli wysyłamy pliki na serwer). Zmieniamy konfiguracje wskazując na odpowiednią bazę danych np. PhpMyAdmin-MySQL, jeżeli nie mamy zainstalowanego narzędzia do obsługi baz danych, uprzednio je instalujemy.

5 Wnioski projektowe

Na pewno nie odkładać robienia takiego projektu na tak późny okres zwłaszcza w języku i frameworku, w którym nie wszyscy byli tak dobrze obeznani. Na pewno też mieliśmy za dużo ambicji w porównaniu do rzeczywistych naszych możliwości, zwłaszcza w takim okresie czasu, który można zobaczyć po tym kiedy były robione commity, pozostałości tego można jeszcze znaleźć w kodzie oraz bazie danych. Gdzie nie jest w ogóle wprowadzony system karmy, nie mamy komentarzy do komentarzy (zagnieżdżonych komentarzy). Ale na pewno wbudowane i gotowe funkcje w Laravelu pozwoliły nam ukończyć ten projekt tak szybko. Gdybyśmy mieli pisać ten projekt od 0 bez korzystania z frameworku, w tym momencie nie bylibyśmy nawet w 10