**INSTRUKCJA LABORATORYJNA**

|  |  |
| --- | --- |
| Przedmiot | *Zaawansowane Techniki Programowania Aplikacji* |
| Laboratorium 3 | ***Aplikacja we frameworku*** |
| Nazwisko i imię | ***Kuzio Kornel*** |

**Wprowadzenie**

Dzisiejsze aplikacje należy wytworzyć i dostarczyć dosyć szybko. Istotny jest wybór

odpowiedniego frameworka aplikacji, dzieki któremu można skupić się na funkcjonalności aplkacji.

Taki framework posiada podstawowe moduły, np.: Routing, obsługa danych POST i GET, sesji itp.

**Zadanie**

1. Wybierz jeden z popularnych frameworków dla języka PHP i przygotuj mini-aplikację.

2. Wymagania wstępne dla web-aplikacji:

◦ przygotowanie routingu dla podstron (główna, kontakt, oferta, logowanie, profil),

◦ połączenie z bazą danych dla pobierania i wysyłania danych z/do BD,

◦ obsługa mechanizmu kont użytkowników.

3. Użycie menadżera pakietów Composer:

◦ zastosowanie bibliotek, np. PHPMailer w projekcie,

◦ opisanie jak działa, wraz z przykładami fragmentów kodu (czym jest autoload.php i co

się w nim znajduje),

◦ podanie przykładów innych menażerów pakietów.

4. Należy przygotować sposób wdrożenia aplikacji u klienta/użytkownika:

◦ kroki instalacji,

◦ obsługa ewentualnych bibliotek zależnych,

◦ sposób uruchamiania,

◦ pamiętając że klient może mieć dedykowany serwer (dostęp do terminala/composera)

lub hosting (bez dostępu do terminala/composera).

1. Repozytorium z web-aplikacją do zadania wraz ze screenem potwierdzającym działanie.

**Wymagania:**

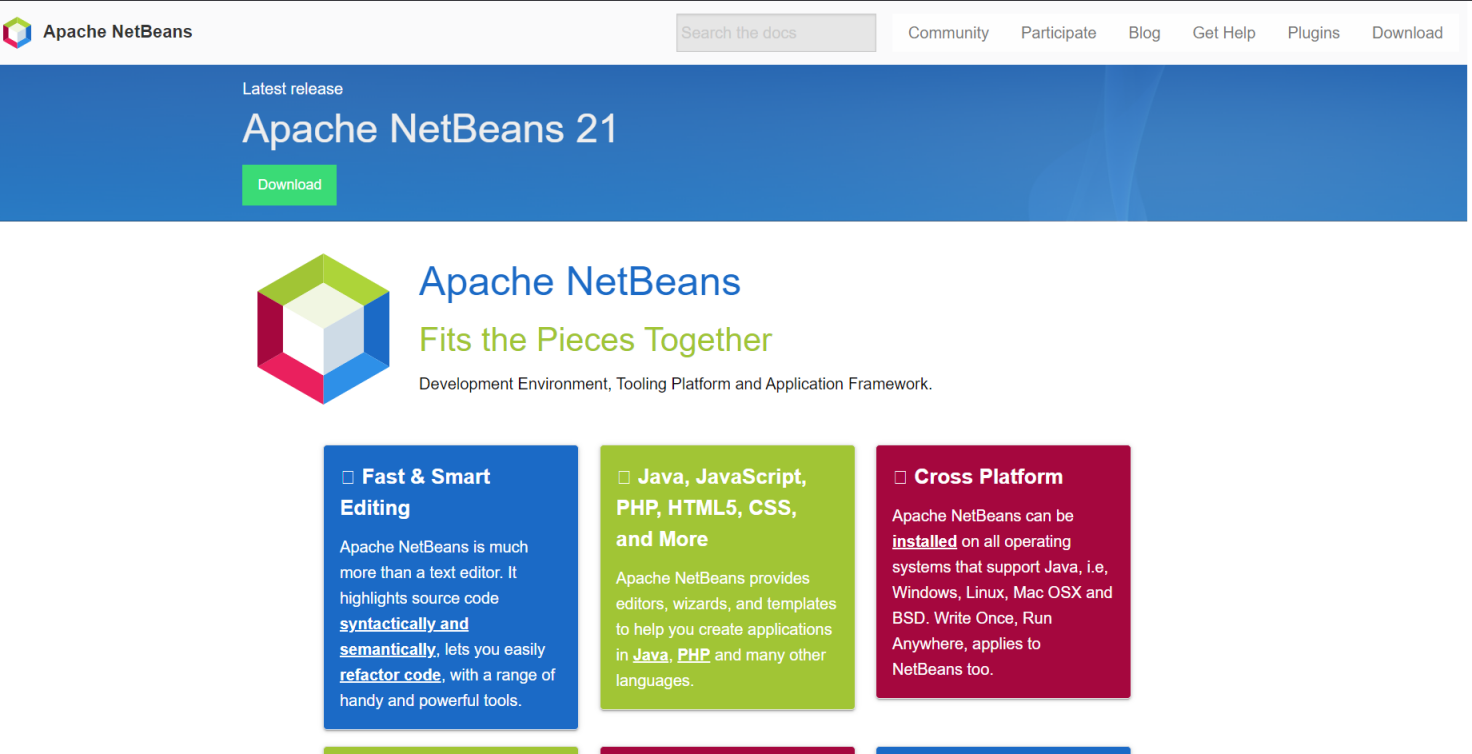
a) język PHP

b) założenie Repozytorium z dostarczeniem odpowiednich instrukcji i opisów

c) przygotowanie w Repozytorium stron Wiki z opisem zadania Composer

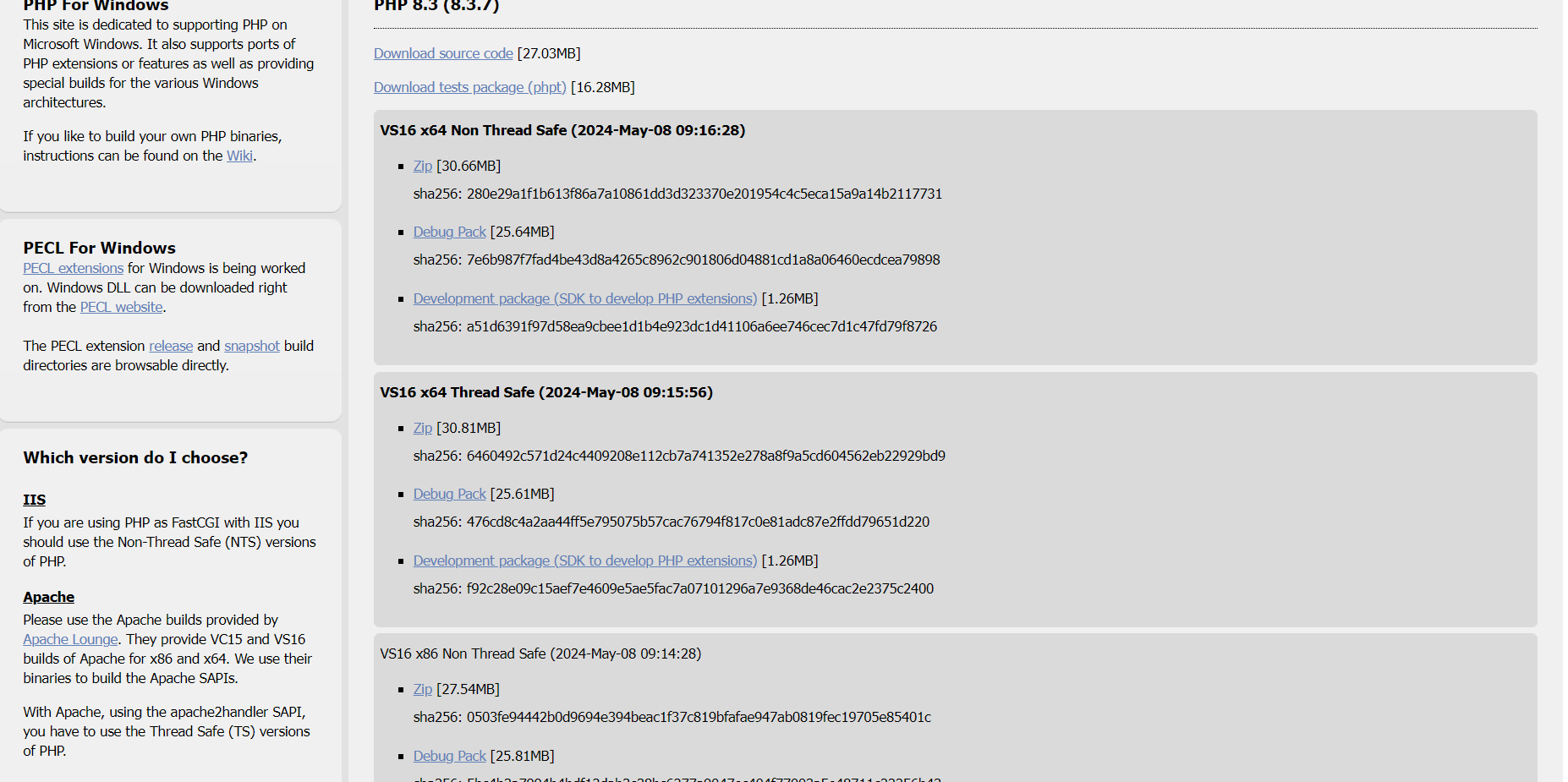
**Krok 1: Instalacja NetBeans i PHP**

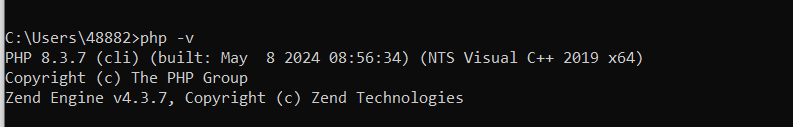
Pobierz i zainstaluj Apache NetBeans IDE: Możesz pobrać NetBeans z oficjalnej strony NetBeans.



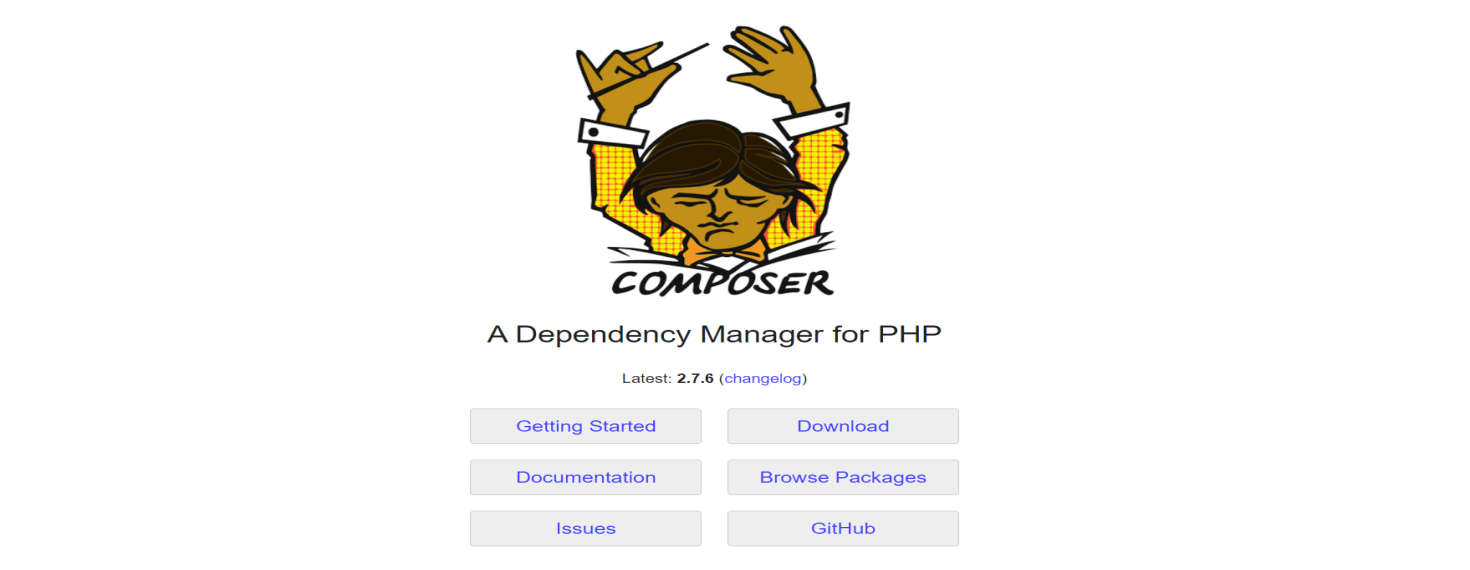
Zainstaluj PHP: Upewnij się, że masz zainstalowane PHP. Możesz to sprawdzić, wpisując php -v w terminalu.

Pobieramy php ze strony a nastepnie sprawdzamy jaką ma wersję.





Zainstaluj Composer: Pobierz Composer z Composer.



Następnie należy pobrać cygwin i pobrać nastepujące pakiety: curl, git, php, composer, openssh, wget



Ostanią rzeczą jaką należy zrobić, trzeba w pliku php.ini zmienić ustawienia extension paru opcjom ,żeby nie było komplikacji w programie. Trzeba zmienić w ; usunąć w nazwach:

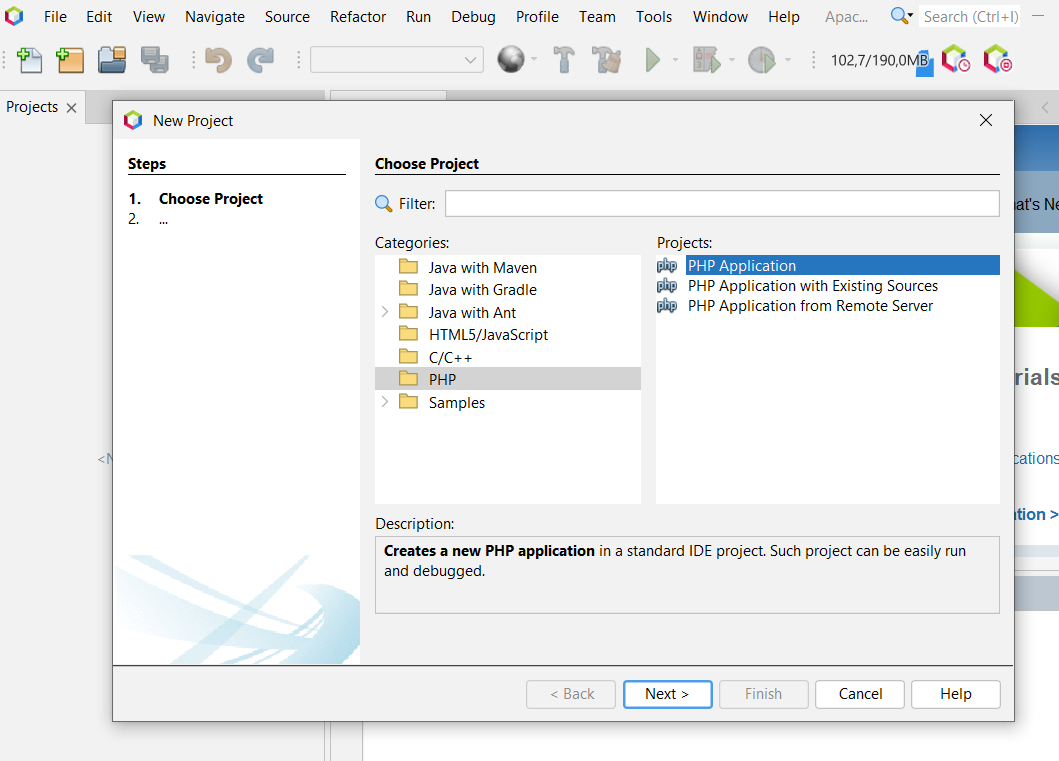
Extension=zip, fileinfo, pdo\_mysql, pdo\_sqlite, pdo\_pgsql

**Krok 2: Tworzenie projektu Laravel w NetBeans**

W pierwszej kolejności trzeba utworzyć nowy projekt PHP w NetBeans:

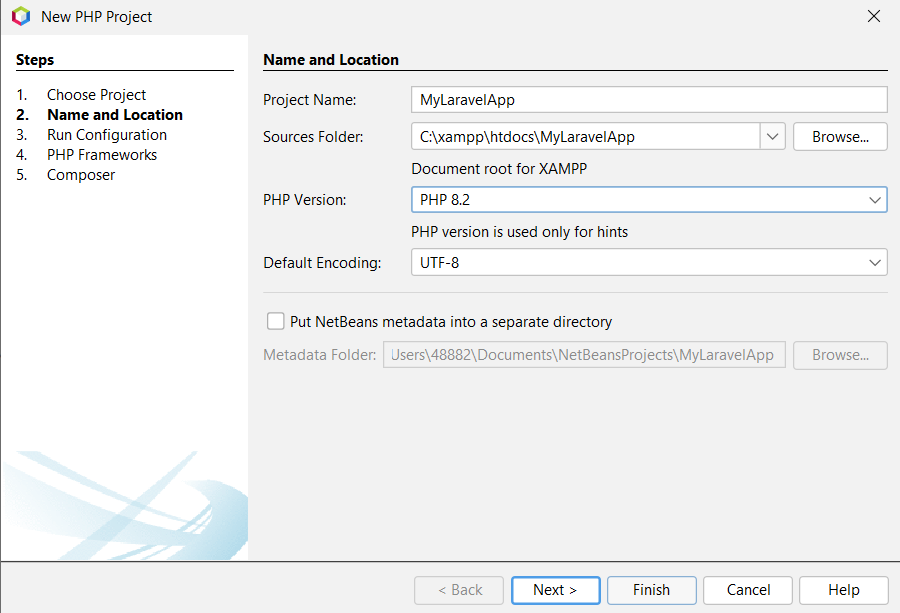
Aby to zrobić należy otworzyć NetBeans, wybierz File -> New Project.

Wybrać PHP -> PHP Application i kliknij Next.



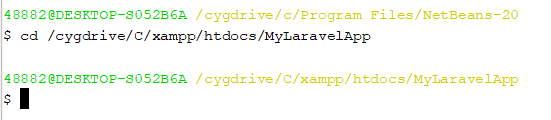
Wprowadź nazwę projektu (np. MyLaravelApp) i ścieżkę, gdzie projekt będzie przechowywany.

Kliknij Finish.



Następnie trzeba zainstalować Laravel w projekcie, aby to zrobić należy usunąc cała zawartość tego folderu, i w terminalu przekazac położenie naszej aplikacji. U mnie bedzie to: C:\xampp\htdocs\MyLaravelApp

Przechodzimy do terminala i wpisujemy taką komende:

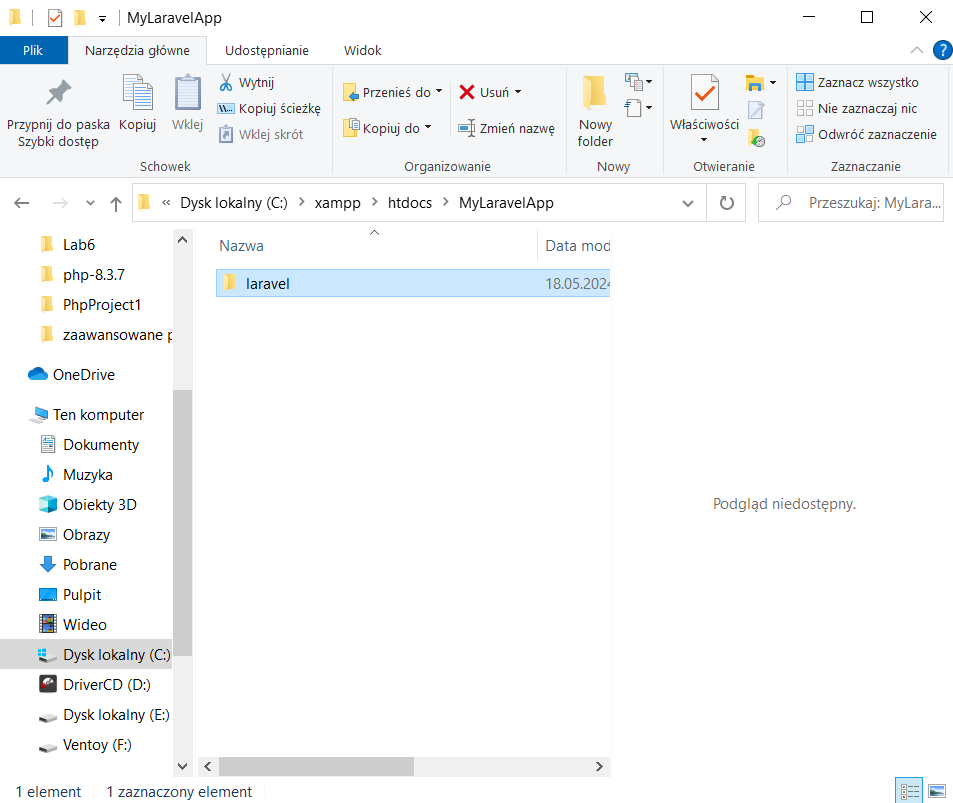


Powinno nam zmienić lokalizacje projektu, nastepnie trzeba użyć composer i zainstalować laravel

Za pomocą komendy:

composer create-project --prefer-dist laravel/laravel .

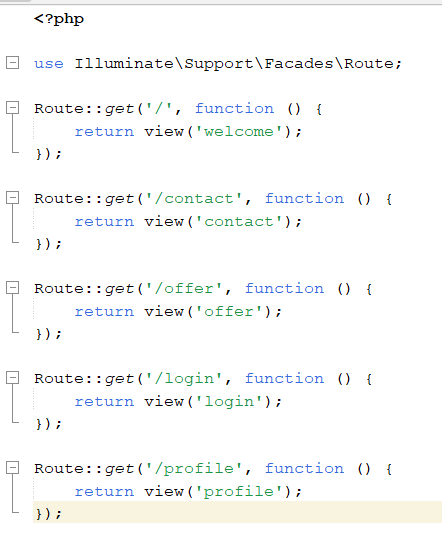
Jeśli wszystko sie udało powinien stworzyć się nam folder w docelowym pliku naszej aplikacji



**Krok 3: Konfiguracja routingu**

Musimy teraz z edytować plik routes/web.php, aby dodać wymagane trasy:

Tak powinien mniej więcej wyglądać:



**Krok 4: Połączenie z bazą danych**

Teraz trzeba skonfigurować połączenie z bazą danych w pliku .env. Laravel domyślnie korzysta z MySQL.

Przykładowa konfiguracja:

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

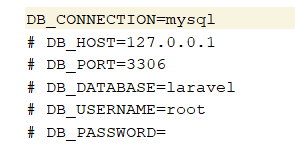
DB\_PORT=3306

DB\_DATABASE=your\_database

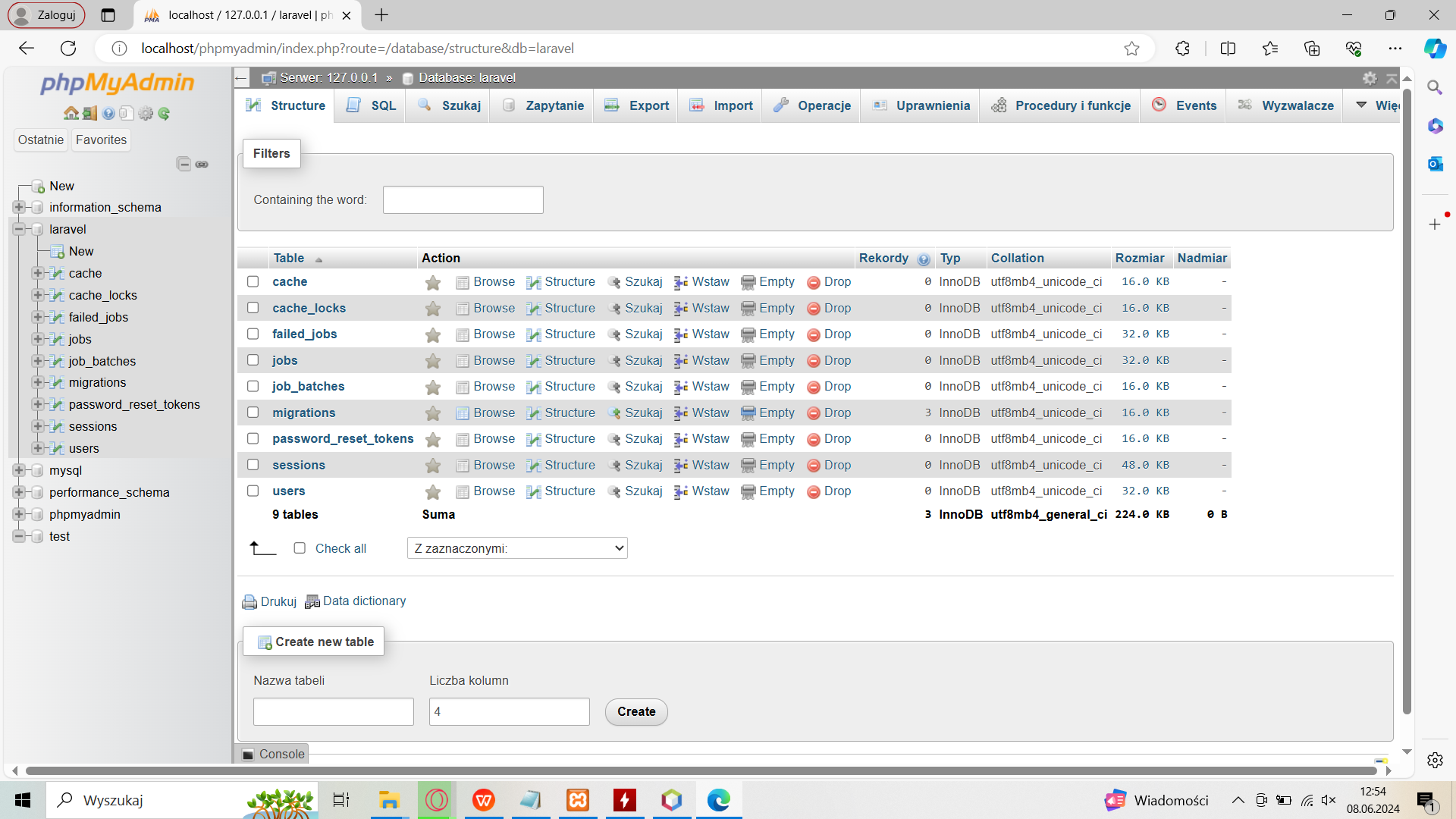
DB\_USERNAME=your\_username

DB\_PASSWORD=your\_password

Dużo nie trzeba było zmieniać tylko połaczenie bazy dany z sqlite na mysql



W phpMyAdmin pokazała nam sie baza danych z aplikacji laravel



**Krok 5: Obsługa użytkowników**

Laravel oferuje gotowy system autoryzacji. Wykorzystaj komendę Artisan, aby wygenerować odpowiednie struktury.

Trzeba otworzyć terminal w NetBeans i uruchomić nastepującą komende, ale zanim wcześniej trzeba zmienic katalog w którym znajduje sie artisan u mnie np w laravel sie znajdował:

php artisan ui:auth

48882@DESKTOP-S052B6A /cygdrive/C/xampp/htdocs/MyLaravelApp/laravel

$ php artisan ui:auth

The [auth/login.blade.php] view already exists. Do you want to replace it? (yes/no)

[no]

❯ yes

The [auth/passwords/confirm.blade.php] view already exists. Do you want to replace

it? (yes/no) [no]

❯ yes

The [auth/passwords/email.blade.php] view already exists. Do you want to replace it

? (yes/no) [no]

❯ yes

The [auth/passwords/reset.blade.php] view already exists. Do you want to replace it

? (yes/no) [no]

❯ yes

The [auth/register.blade.php] view already exists. Do you want to replace it? (yes/

no) [no]

❯ yes

The [auth/verify.blade.php] view already exists. Do you want to replace it? (yes/no

) [no]

❯ yes

The [home.blade.php] view already exists. Do you want to replace it? (yes/no) [no]

❯ yes

The [layouts/app.blade.php] view already exists. Do you want to replace it? (yes/no

) [no]

❯ yes

The [HomeController.php] file already exists. Do you want to replace it? (yes/no) [

yes]

❯ yes

The [Controller.php] file already exists. Do you want to replace it? (yes/no) [yes]

❯ yes

INFO Authentication scaffolding generated successfully.

Następnie robimy migracje:

$ php artisan migrate

WARN The database 'laravel' does not exist on the 'mysql' connection.

Would you like to create it? (yes/no) [yes]

❯ yes

INFO Preparing database.

Creating migration table ........................................... 26.24ms DONE

INFO Running migrations.

0001\_01\_01\_000000\_create\_users\_table .............................. 105.77ms DONE

0001\_01\_01\_000001\_create\_cache\_table ............................... 21.02ms DONE

0001\_01\_01\_000002\_create\_jobs\_table ................................ 63.87ms DONE

48882@DESKTOP-S052B6A /cygdrive/C/xampp/htdocs/MyLaravelApp/laravel

**Krok 6: Użycie PHPMailer**

Teraz użyjemy PHPMailer, w pierwszej kolejności należy pobrać go za pomocą Composera.

Użyjemy komendy do tego : composer require phpmailer/phpmailer

Konfiguracja PHPMailer poniższy kod do kontrolera, aby wysyłał maile:

use PHPMailer\PHPMailer\PHPMailer;

use PHPMailer\PHPMailer\Exception;

function sendEmail() {

$mail = new PHPMailer(true);

try {

$mail->isSMTP();

$mail->Host = 'smtp.example.com';

$mail->SMTPAuth = true;

$mail->Username = 'user@example.com';

$mail->Password = 'secret';

$mail->SMTPSecure = PHPMailer::ENCRYPTION\_STARTTLS;

$mail->Port = 587;

$mail->setFrom('from@example.com', 'Mailer');

$mail->addAddress('joe@example.net', 'Joe User');

$mail->isHTML(true);

$mail->Subject = 'Here is the subject';

$mail->Body = 'This is the HTML message body <b>in bold!</b>';

$mail->AltBody = 'This is the body in plain text for non-HTML mail clients';

$mail->send();

echo 'Message has been sent';

} catch (Exception $e) {

echo "Message could not be sent. Mailer Error: {$mail->ErrorInfo}";

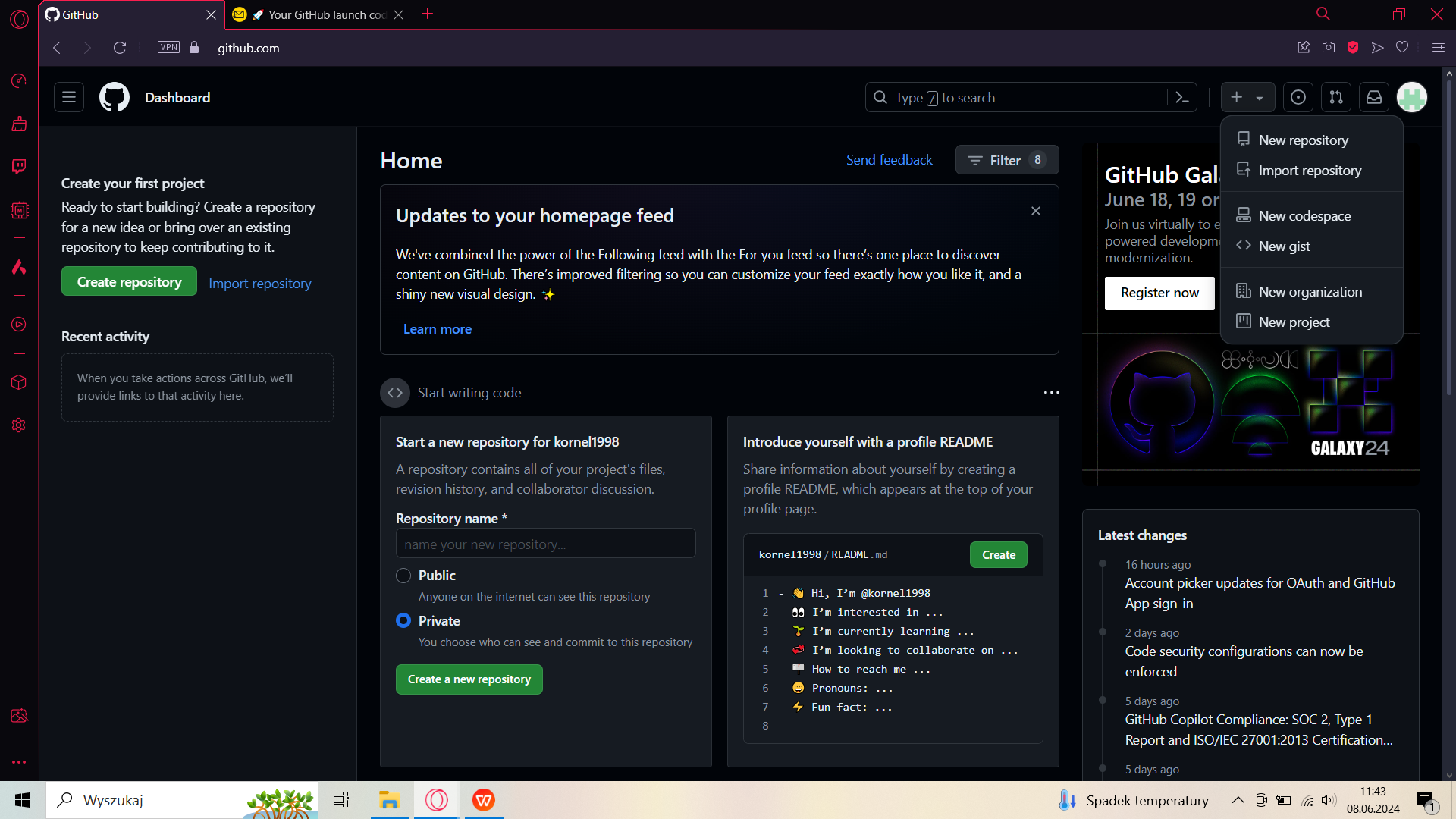
}

}

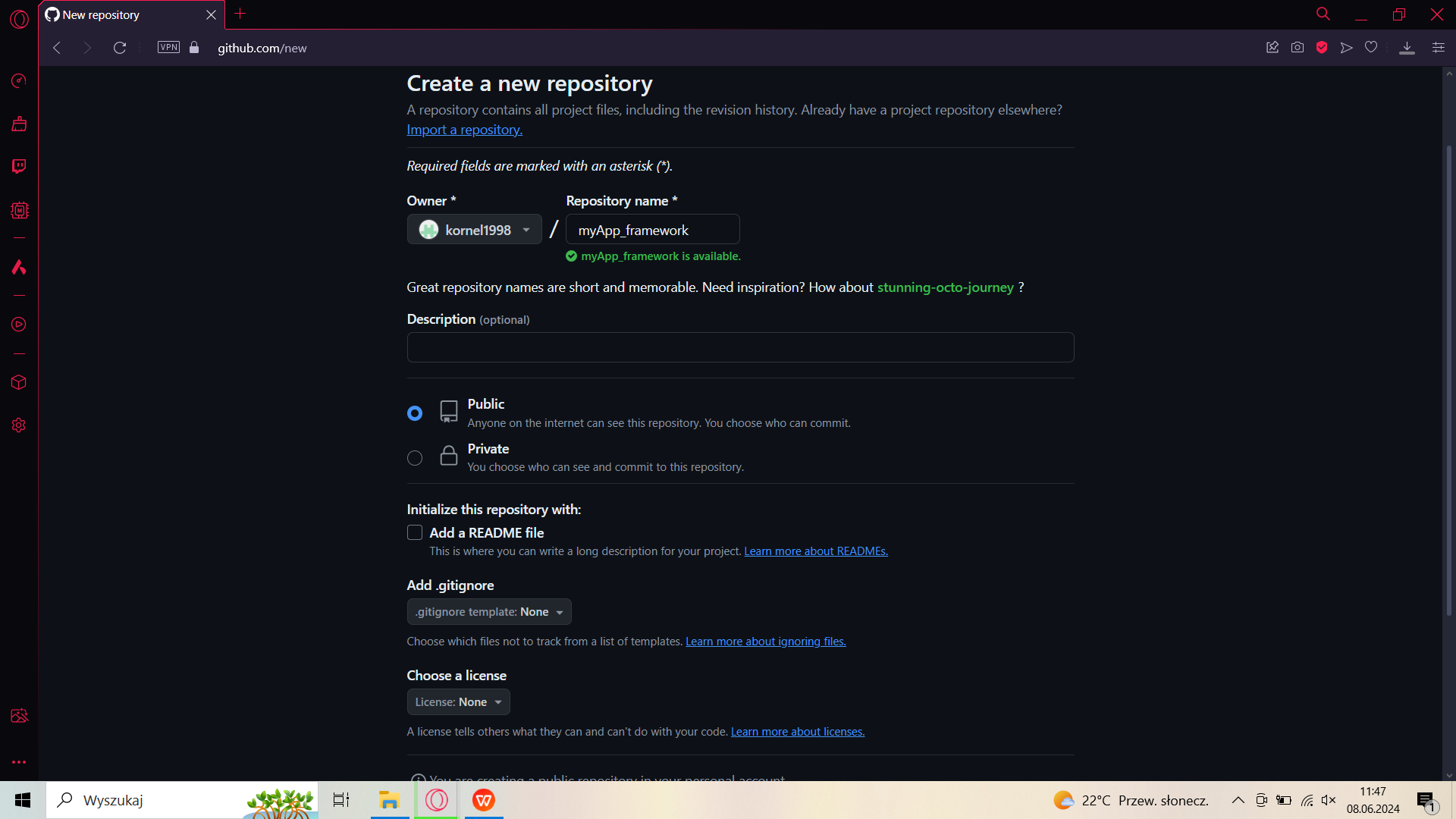
Plik autoload.php generowany przez Composer automatycznie załaduje klasy zainstalowanych bibliotek.

**Krok 7: Repozytorium**

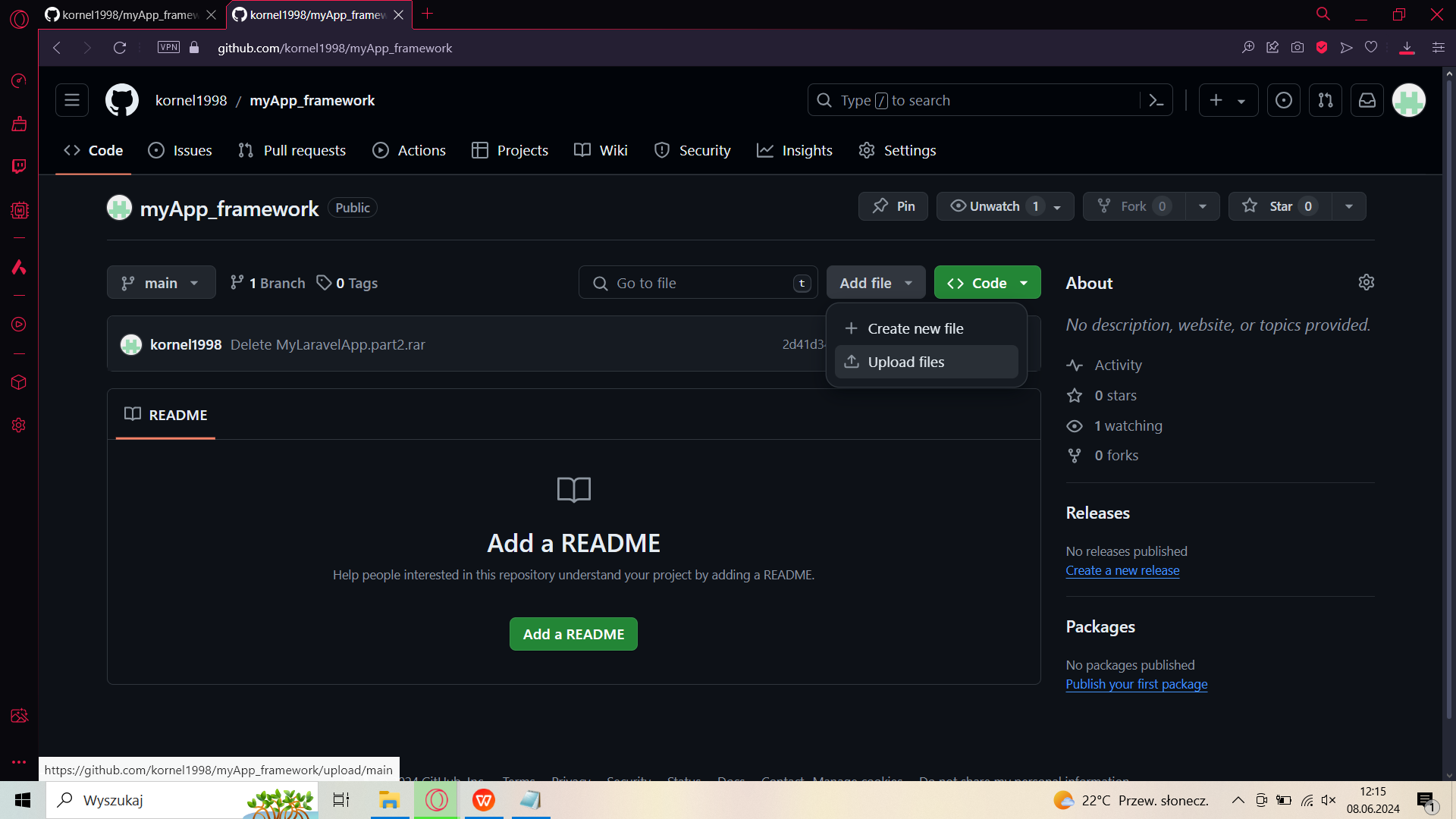
W pierwszej kolejności należy się zalogować na strone github i dodać nowe repozytorium



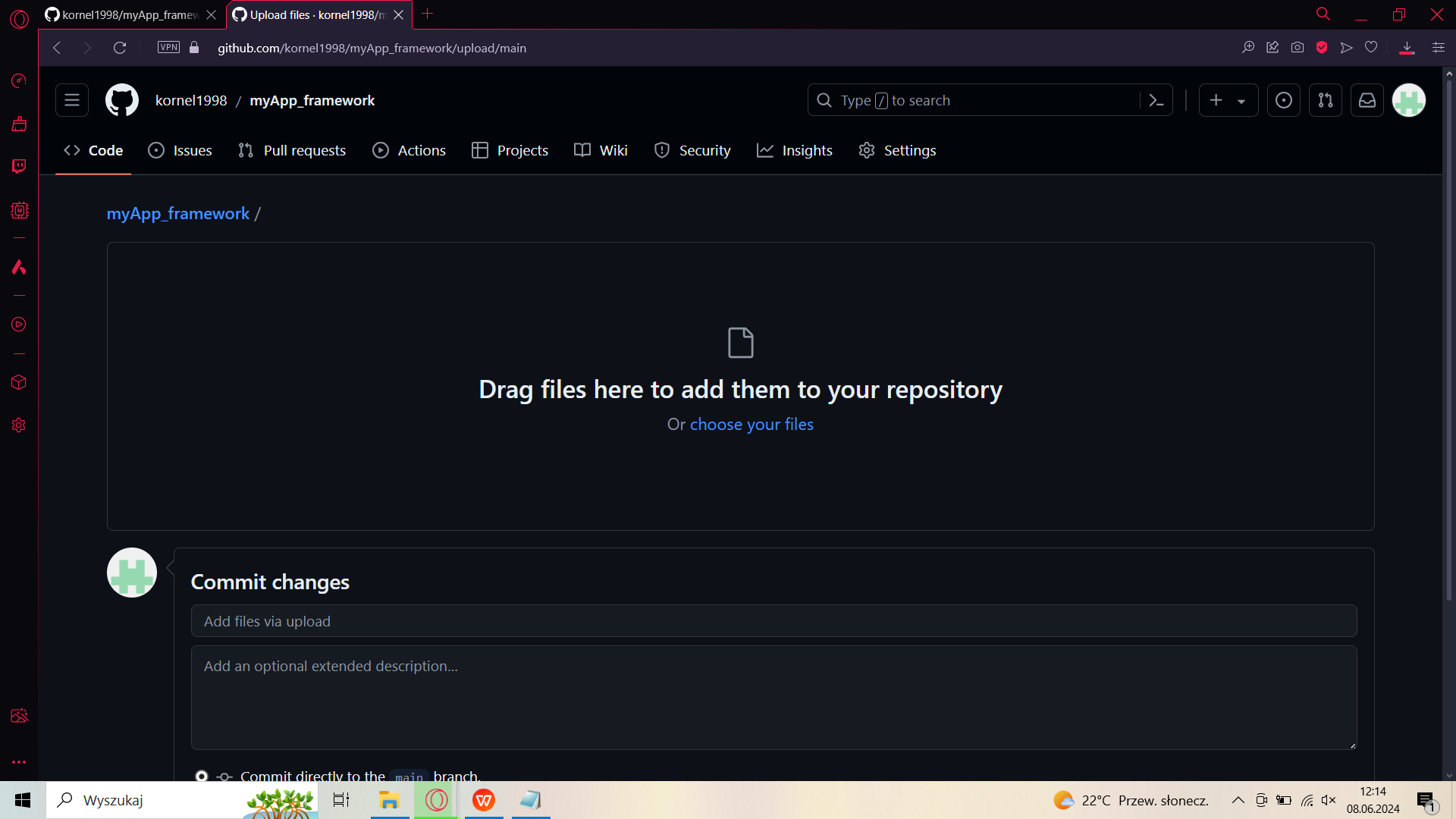
Kolejnym krokiem bedzię nazwanie naszego repozytorium w moim przypadku myApp\_framework



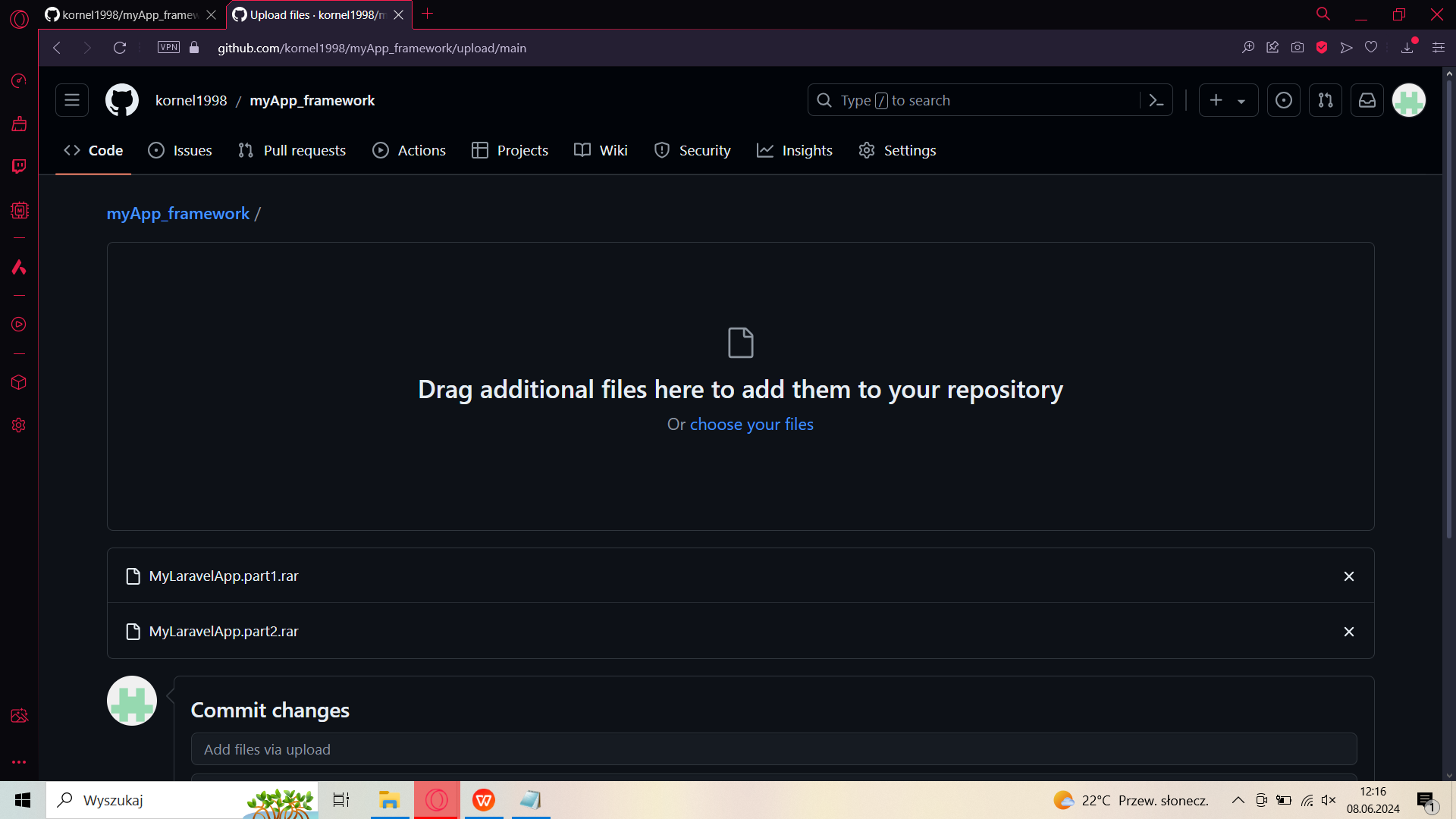
Kolejnym krokiem w tym będzie dodanie naszej aplikacji do naszego repozytorium



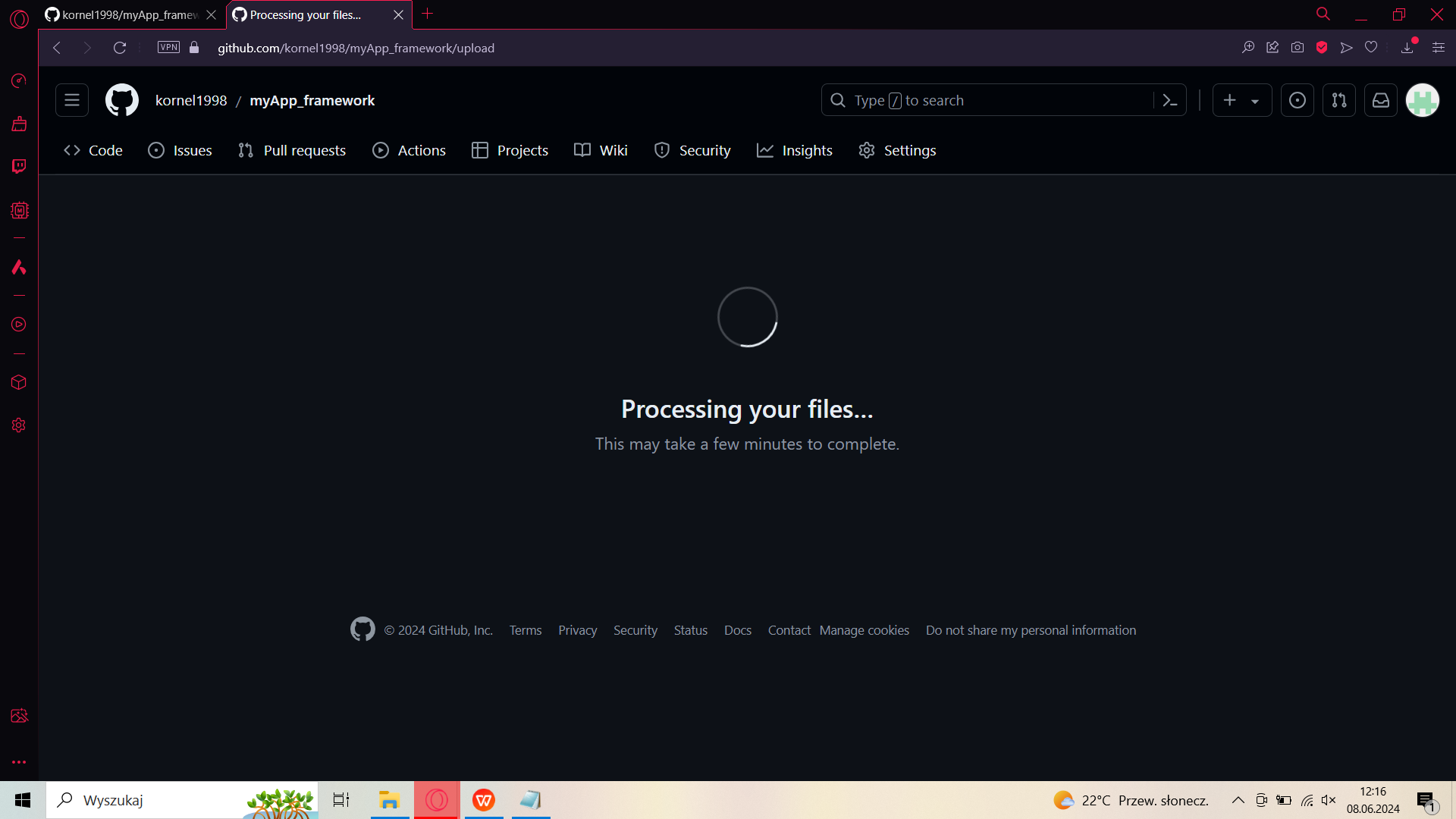
Następnie należy przesłać nasz projekt, który zrobilismy



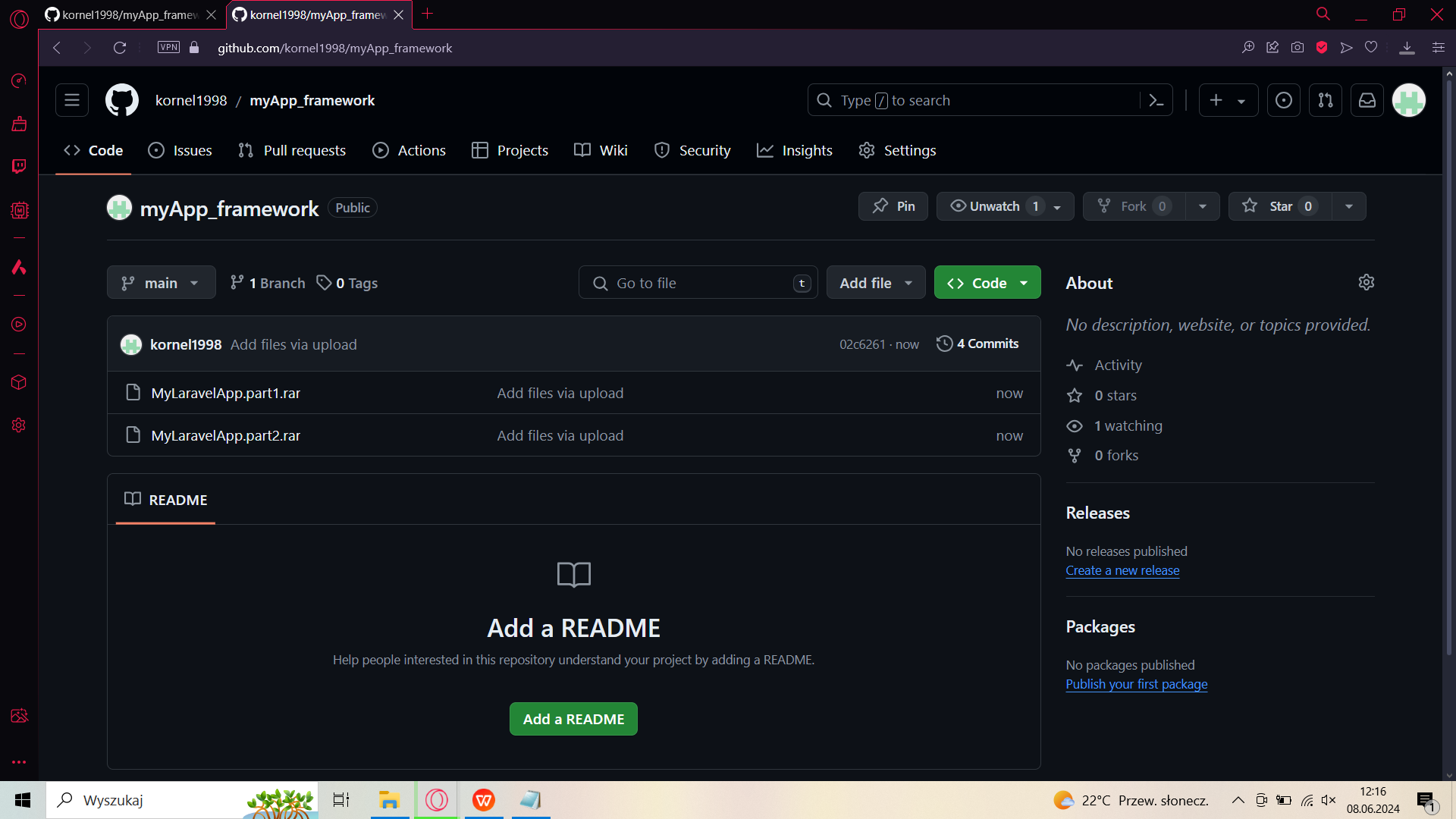
W moim przypadku musialem rozbić archiwum na 2 cześci ponieważ, limit wynosił do stu plików przy dodawaniu lub 25mb jeden plik. Mój program posiadał wage cos w okolicach 38mb.



Jeżeli spełniliśmy wszystkie aspekty pliki powinny sie dodać do naszego repozytorium



Nasza aplikacja została finalnie dodana do repozytorium



**Podsumowanie**

Ostatecznie, nasza mini-aplikacja w Laravel została zbudowana w sposób umożliwiający skuteczne zarządzanie trasami, komunikację z bazą danych, obsługę kont użytkowników oraz integrację z PHPMailer do obsługi komunikacji e-mailowej. Ta aplikacja może być łatwo dostosowywana i rozbudowywana zgodnie z potrzebami biznesowymi i wymaganiami klienta.

**Link do repozytorium:**

https://github.com/kornel1998/myApp\_framework