 Aplikacje sieciowe i webowe   
Projekt Zaliczeniowy  
„FileLink Send Service”

# 1. Założenia

Projekt zakłada implementację serwera File Link w języku PHP, wykorzystując framework Symfony. Główną funkcjonalnością aplikacji jest bezpieczne przesyłanie plików, przy użyciu filelinkSend (formalnie znanego jako Firefox Send). Program obsługuje zarówno wysyłanie, jak i odbieranie plików, a dodatkowo korzysta z WebSocketów do komunikacji w czasie rzeczywistym.

Serwer PHP oraz baza danych MySQL zostały uruchomione w środowisku Docker, co zapewnia izolację i łatwość wdrażania aplikacji.

Główne funkcjonalności systemu:

* Bezpieczne przesyłanie plików między użytkownikami.
* Obsługa plików poprzez API w Symfony.
* Użycie WebSocketów do zapewnienia komunikacji w czasie rzeczywistym.
* Mechanizmy zabezpieczeń plików przed nieautoryzowanym dostępem.

# 2. Dane pobierane przez program

Program pobiera następujące dane wejściowe:

* Pliki wysyłane przez użytkownika.
* Metadane pliku (nazwa, rozmiar, format).
* Czas ważności linku.
* Klucz szyfrujący (opcjonalnie, dla zaszyfrowanych przesyłek).
* Zapytania WebSocket od klientów odbierających pliki.

# 3. Dane otrzymywane z programu

Program pobiera następujące dane wejściowe:

• Link do pliku wysłanego przez użytkownika.

• Czas ważności linku.

• Klucz szyfrujący (opcjonalnie, dla zaszyfrowanych przesyłek).

• Zapytania WebSocket od klientów odbierających pliki.

# 4. Zastosowane algorytmy do działania programu

W projekcie wykorzystano następujące mechanizmy i algorytmy:

* Szyfrowanie plików: Pliki mogą być opcjonalnie szyfrowane przed wysłaniem.
* WebSocket Server: Do obsługi połączeń w czasie rzeczywistym.
* Obsługa sesji i tokenów: Mechanizm autoryzacji dostępu do plików.

Obraz zawierający tekst, zrzut ekranu, design, Czcionka

Opis wygenerowany automatycznie

Rysunek 1 Główny algorytm serwera

Obraz zawierający tekst, zrzut ekranu, diagram, Wielobarwność

Opis wygenerowany automatycznie

Rysunek 2 Algorytm endpointów

Obraz zawierający tekst, zrzut ekranu, diagram, Czcionka

Opis wygenerowany automatycznie

Rysunek 3 Algorytm działania serwera websocket

Struktura encji File

<?php  
  
namespace App\Entity;  
  
use Doctrine\ORM\Mapping as ORM;  
  
#[*ORM*\Entity]  
#[*ORM*\Table(name: 'files')]  
class File  
{  
// #[ORM\GeneratedValue(strategy: 'IDENTITY')]  
 #[*ORM*\Id]  
 #[*ORM*\Column(type: 'integer')]  
 private int $id;  
  
 #[*ORM*\Column(type: 'json')]  
 private array $metadata;  
  
 #[*ORM*\Column(type: 'blob', options: ["length" => 4294967295])]  
 private $data;  
  
 #[*ORM*\Column(type: 'text', length: 255)]  
 private $token;  
  
 public function getId(): ?int  
 {  
 return $this->id;  
 }  
  
 public function setId(int $id): self  
 {  
 $this->id = $id;  
  
 return $this;  
 }  
  
 public function getMetadata(): ?array  
 {  
 return $this->metadata;  
 }  
  
 public function setMetadata(array $metadata): self  
 {  
 $this->metadata = $metadata;  
  
 return $this;  
 }  
  
 public function getData(): ?string  
 {  
 return $this->data;  
 }  
  
 public function setData($data): self  
 {  
 $this->data = $data;  
  
 return $this;  
 }  
  
 public function getToken(): ?string  
 {  
 return $this->token;  
 }  
  
 public function setToken(string $token): self  
 {  
 $this->token = $token;  
  
 return $this;  
 }  
  
}

Struktura encji Password

<?php  
  
namespace App\Entity;  
  
use Doctrine\ORM\Mapping as ORM;  
  
#[*ORM*\Entity]  
#[*ORM*\Table(name: 'passwords')]  
class Password  
{  
 #[*ORM*\Id]  
 #[*ORM*\GeneratedValue(strategy: 'IDENTITY')]  
 #[*ORM*\Column(type: 'integer')]  
 private int $id;  
  
 #[*ORM*\Column(type: 'text', length: 255)]  
 private $password;  
  
 #[*ORM*\Column(type: 'integer')]  
 private $file\_id;  
  
 public function getId(): ?int  
 {  
 return $this->id;  
 }  
  
  
 public function getPassword(): ?string  
 {  
 return $this->password;  
 }  
  
 public function setPassword($password): self  
 {  
 $this->password = $password;  
  
 return $this;  
 }  
  
 public function getFileId(): ?int  
 {  
 return $this->file\_id;  
 }  
  
 public function setFileId($file\_id): self  
 {  
 $this->file\_id = $file\_id;  
  
 return $this;  
 }  
  
}

# 5. Testy poprawności działania programu

Testy przeprowadzono w następujących obszarach:

* Poprawność przesyłania plików – Sprawdzono różne formaty i rozmiary plików.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Rysunek 4 Testy - Wysyłanie pliku

Obraz zawierający tekst, oprogramowanie, Ikona komputerowa, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Rysunek 5 Testy - Plik przesłany do serwera oraz serwer zwrócił dane do autoryzacji

* Testy obciążeniowe – Wysyłanie wielu plików jednocześnie.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Rysunek 6 Testy - Przesyłanie 2 plików jednocześnie

Obraz zawierający tekst, oprogramowanie, Ikona komputerowa, Strona internetowa

Opis wygenerowany automatycznie

Rysunek 7 Testy - Oba pliki zostały poprawnie przesłane

# 6. Wnioski

Program spełnia założenia projektowe i pozwala na bezpieczne przesyłanie plików między użytkownikami. Dzięki WebSocketom zapewniono szybki dostęp do informacji o plikach, a mechanizmy szyfrowania gwarantują ich bezpieczeństwo.

Możliwe przyszłe usprawnienia obejmują:

* Dodanie interfejsu użytkownika do lepszej obsługi aplikacji.
* Rozbudowanie autoryzacji dostępu do plików.

# 7. Źródła

* <https://symfony.com/doc/current/index.html>
* <http://socketo.me/docs/>
* <https://docs.docker.com>
* <https://www.jetbrains.com/help/phpstorm/getting-started.html>
* <https://github.com/tdulcet/Thunderbird-Send?tab=readme-ov-file>
* <https://www.microsoft.com/pl-pl/microsoft-365/visio/flowchart-software>

# 8. Spis ilustracji

[Rysunek 1 Główny algorytm serwera 3](#_Toc189261214)

[Rysunek 2 Algorytm endpointów 4](#_Toc189261215)

[Rysunek 3 Algorytm działania serwera websocket 5](#_Toc189261216)

[Rysunek 4 Testy - Wysyłanie pliku 9](#_Toc189261217)

[Rysunek 5 Testy - Plik przesłany do serwera oraz serwer zwrócił dane do autoryzacji 10](#_Toc189261218)

[Rysunek 6 Testy - Przesyłanie 2 plików jednocześnie 11](#_Toc189261219)

[Rysunek 7 Testy - Oba pliki zostały poprawnie przesłane 11](#_Toc189261220)

# 9. Spis treści

Spis treści

[1. Założenia 1](#_Toc189261235)

[2. Dane pobierane przez program 1](#_Toc189261236)

[3. Dane otrzymywane z programu 2](#_Toc189261237)

[4. Zastosowane algorytmy do działania programu 2](#_Toc189261238)

[5. Testy poprawności działania programu 9](#_Toc189261239)

[6. Wnioski 12](#_Toc189261240)

[7. Źródła 13](#_Toc189261241)

[8. Spis ilustracji 14](#_Toc189261242)

[9. Spis treści 14](#_Toc189261243)