**Performance Media Test**

**Dokumentacja techniczna**

|  |  |
| --- | --- |
| Autor:  Wersja: | Paweł Liwocha PAWELDESIGN  1.0 |
| Data utworzenia: | 17.05.2020 |

# **Historia zmian**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Autor** | **Podsumowanie zmian** | **Wersja** |
| 17.05.2020 | Paweł Liwocha | Wersja początkowa | 1.0 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Spis Treści**

[Historia zmian 2](#_Toc40628391)

[1. Wstęp i opis 4](#_Toc40628392)

[1.1. Opis 4](#_Toc40628393)

[1.2. Zespół 4](#_Toc40628394)

[1.3. Uogólniony przebieg prac 4](#_Toc40628395)

[2. Technologie i wzorce projektowe 4](#_Toc40628396)

[2.1. PHP, MySQL 4](#_Toc40628397)

[2.2. Frameworki 4](#_Toc40628398)

[3. Funkcjonalności systemu 4](#_Toc40628399)

[4. Struktura systemu 5](#_Toc40628400)

[4.1. Elementy systemu 5](#_Toc40628401)

[5. Rozszerzony opis plików, klas i funkcji systemu 6](#_Toc40628402)

[5.1. Controller - indexController 6](#_Toc40628403)

[5.2. Service - AddGoogleSheets 6](#_Toc40628404)

[5.3. Command - dataFileCommand 6](#_Toc40628405)

[5.4. Entity - PerformanceData 7](#_Toc40628406)

[5.5. Repository - PerformanceDataRepository 7](#_Toc40628407)

[6. Deploy systemu 7](#_Toc40628408)

[7. Stawianie projektu 8](#_Toc40628409)

[7.1. Stawianie projektu na Windows 9](#_Toc40628410)

[8. Podstawowe użycie systemu 9](#_Toc40628411)

# Wstęp i opis

## Opis

Projekt Systemu przetworzenia danych z pliku wejściowego JSON z zapisem danych do lokalnej bazy danych oraz dodaniem ich do dokumentu GoogleSheets.

## Zespół

Zespół składa się z: 1 kierownika projektu i 1 programistę.

|  |  |  |
| --- | --- | --- |
| **Imię i nazwisko** | **Funkcja** | **Zakres obowiązków** |
| Paweł Liwocha | Programista | Część Back-end |

## Uogólniony przebieg prac

|  |  |
| --- | --- |
| **Data** | **Zadanie** |
| 17.05.2020 | Stworzenie pełnej wersji aplikacji |
|  |  |
|  |  |

# Technologie i wzorce projektowe

## PHP, MySQL

PHP interpretowany skryptowy język programowania zaprojektowany do generowania stron internetowych i budowania aplikacji webowych w czasie rzeczywistym. MySQL to wolnodostępny, otwartoźródłowy system zarządzania relacyjnymi bazami danych.

Wymienione technologie zostały użyte do implementacji systemu Form API ze względu na ich popularność, elastyczność oraz bardzo dobrą ich znajomość przez programistów zespołu.

## Frameworki

Frameworkiem back-end'owym użytym do stworzenia systemu został Symfony w wersji 5. Jest to framework oparty na strukturze Model-View-Controller (MVC).

* Model – moduły zarządzające danymi aplikacji,
* Widok – wizualna reprezentacja aplikacji,
* Kontroler – moduły pełniące funkcję pośrednika między modelami i widokami.

# Funkcjonalności systemu

System posiada kilka funkcjonalności z odbiorem danych oraz zapisem do bazy danych:

* Odbieranie danych z pliku o nazwie podanej przy wywołaniu;
* Przetworzenie danych i zapis danych do bazy danych MySQL w odpowiednie kolumny,
* Zapis danych do arkusza Google Sheets.

# Struktura systemu

## Elementy systemu

System oparty jest na 1 kontrolerze:

* [indexController](#_Controller_-_indexController) – obsługa części wyświetlania systemu (index),

Oraz serwis:

* [AddGoogleSheets](#_Service_-_AddGoogleSheets) – Odebranie danych, przygotowanie oraz zapis do Google Sheets;

Przygotowane są również funkcje konsolowe (command):

* [dataFileCommand](#_Command_-_dataFileCommand) – Klasa przetwarzania danych z pliku i zapisu do lokalnej bazy danych oraz Google Sheets

Encje danych – tablice bazy danych (enity):

* [PerformanceData](#_Entity_-_PerformanceData) – Klasa przetrzymująca kolumny i ich typy z bazy danych oraz funkcje odbierające/przypisujące dane;

Repozytoria danych – funkcje z zapytaniami na bazie danych (repository):

* [PerformanceDataRepository](#_Repository_-_PerformanceDataReposit) – Klasa przetrzymująca funkcje z zapytaniami do bazy danych dla odpowiedniej tablicy – Performacne\_data

# Rozszerzony opis plików, klas i funkcji systemu

## Controller - indexController

Klasa odpowiedzialna za widok systemu pod jego adresem. Nie przyjmuje żadnych frontów

public function index ()

Funkcja działa pod adresem /.  
Funkcja zwraca tekst *Index of Performance Media Test* jako stronę główną Serwisu;

## Service - AddGoogleSheets

Klasa zapisująca dane do Google Sheets. Tworzy zmienne logger dostępne dla funkcji wewnątrz

private function saveDateToGoogleSheets($inputData)

Funkcja tworzy zmienną $arrayData z nagłówkami arkusza  
Tworzy nową klasę Google\_Client i przypisuje jej nazwę, Scopes oraz autoryzację.

Odebrane dane są w pętli przerabiane i dodawane do zmiennej $arrayData aby były bez kluczy (same wartości);

Przypisywane do zmiennych dane dotyczące arkusza Google Sheet;

Tworzona jest nowa klasa Google Sheets oraz wysyłane są dane do zapisu przez Google Sheet;

Funkcja zwraca dane z update bądź 0;

## Command - dataFileCommand

Klasa przetwarza dane z pliku oraz zapisuje je do lokalnej bazy danych i Google Sheets. Tworzy zmienne logger, container oraz addGoogleSheets;

protected function configure()

Funkcja przypisuje klasie nazwę: *perfo:data* oraz opis funkcji;

public function execute(InputInterface $input, OutputInterface $output, $ignoreInterval = false)

Funkcja przetwarza dane z pliku oraz zapisuje je do lokalnej bazy danych i Google Sheets.

Przypisuje do zmiennej nazwę pliku podanego przy wywołaniu;

Sprawdza czy została nazwa podane. Jeśli jest pobiera dane pliku;  
Jeśli dane istnieją przypisuje do zmiennej aktualny czas operacji;

Dekoduje zmienne JSON, oraz pętlą przypisuje dane z pliku do klasy PerformanceData;

Dokonuje zapisu wszystkich danych do bazy danych oraz wywołuje funkcję saveDataToGoogleSheets wysyłając dane z pliku;

Przypisuje do zmiennej obecny czas operacji;

Wypisuje na ekranie dane o operacji;

## Entity - PerformanceData

Klasa przygotowująca tablice z kolumnami do bazy danych MySQL dla Doctrine; Tablica zwiera zmienne prywatne jako kolumny bazy danych.

Przy każdej zmiennej jest dodany typ danych, długość oraz inne opcje dodatkowe.

Klasa przetrzymuje też funkcje set oraz get dla danych dla każdej kolumny w tablicy danych.

## Repository - PerformanceDataRepository

Klasa przetrzymujące funkcje które zwracają dane z bazy danych;

Klasa przyjmuje podstawowe zapytania Symfony jak *find*, *findOneBy*, *findAll*, *findBy* ;

# Deploy systemu

Do systemu jest zaciągnięty dodatek deployer, za którego pomocą będą tworzone deploye na serwer.

Tworzenie pliku z kluczem ssh np.: <https://help.github.com/en/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent>

Komendy do deploya:

spradzamy, czy agent działa:

$ eval $(ssh-agent -s)

> Agent pid 59566

dodajemy klucz do agenta:

ssh-add ~/.ssh/id\_rsa

dep deploy test <- test nazwa hosta

# Stawianie projektu

Wymagania:

* PHP >7.2.5
* MySQL
* Composer

## Stawianie projektu na Windows

* Kopia z repozytorium
* Zamienić w pliku `.env--sample` konfigurację mysql i swiftmailer zmienić nazwę pliku na `.env` lub `.env.local`
* `composer install`
* ‘php bin/console doctrine:migrations:migrate’

# Podstawowe użycie systemu

Wywołanie w konsoli funkcji **perfo:data nazwa\_pliku**