**Dokumentacja techniczna**

Panel zarządzania produktami dedykowany dla rozwiązań B2B

**Uczelnia:**

Politechnika Łódzka

**Aktualizacja:**

Standaryzacja dokumentacji [12.06.2020] autor: Mateusz Domalążek

**Temat:** Panel zarządzania produktami dedykowany dla rozwiązań B2B z wykorzystaniem Heroku – Cloud Application Platform

**Słowa kluczowe:** heroku, b2b, sklep internetowy, zarządzanie platformą zakupową,

Politechnika Łódzka ul. Żeromskiego 116, 90-924 Łódź, NIP: 727 002 18 95

Copyright © 2020 - Politechnika Łódzka

Daniel Duczymiński, Leon Wojtczak, Mateusz Domalążek, Łukasz Zalewski

Żadna część niniejszej publikacji nie może być powielana w żadnej formie, w elektronicznym systemie wyszukiwania lub w inny sposób, bez uprzedniej pisemna zgoda wydawcy

# Mechanizm kontroli wersji

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Numer porządkowy | Imię autora zmiany | Nazwisko autora zmiany | Data i godzina zmiany | Wersja po zmianie | Opis |
| 1 | Mateusz | Domalążek | 4.06.2020 12:40 | 1.1 | Utworzenie dokumentu |
| 2 | Mateusz | Domalążek | 5.06.2020 16:40 | 1.2 | Opis środowiska pracy programisty |
| 3 | Mateusz | Domalążek | 5.06.2020  18:50 | 1.3 | Utworzenie schematu ideowego aplikacji |
| 4 | Mateusz | Domalążek | 6.06.2020  17:10 | 1.5 | Dodanie diagramu klas |
| 5 | Mateusz | Domalążek | 6.06.2020 20:30 | 1.6 | dodanie opisu klas |
| 6 | Mateusz | Domalążek | 8.06.2020  9:35 | 1.7 | aktualizacja opisu klas |
| 7 | Mateusz | Domalążek | 12.06.2020  19:00 | 1.8 | dodanie metod i pól klas |
| 8 |  |  |  |  |  |
| 9 |  |  |  |  |  |
| 10 |  |  |  |  |  |
| 11 |  |  |  |  |  |
| 12 |  |  |  |  |  |
| 13 |  |  |  |  |  |
| 14 |  |  |  |  |  |
| 15 |  |  |  |  |  |
| 16 |  |  |  |  |  |
| 17 |  |  |  |  |  |

Spis treści

[**Mechanizm kontroli wersji**](#_mux7uy8q5356) **2**

[**Środowisko pracy**](#_w3agp880knv8) **5**

[Oprogramowanie aplikacji](#_otwpsc2o2ma) 5

[Środowisko programisty](#_1utafny1umsc) 5

[**Schemat ideowy aplikacji i hierarchiczny funkcji**](#_v1d7dk50eejz) **6**

[**Opis klas metod i funkcji aplikacji**](#_71drt5v6smvd) **7**

[public class TaxesInState](#_lbtdgz68teur) 7

[public class ProfitData](#_7cmxwht9hc5c) 8

[public class ProfitCalculator](#_4ut5bs5hlapu) 8

[public class Product](#_fjlx3p8jjmm1) 8

[public class Main](#_1i1ofaxpkwuu) 9

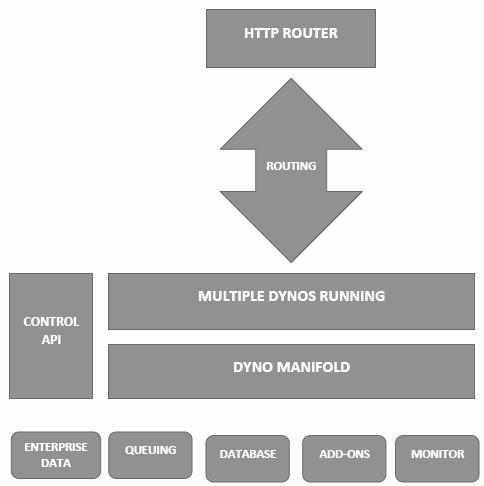
[public class DataBase](#_a2cixhae89fa) 9

[public class CsvReader](#_er48ndvhtchi) 9

# Środowisko pracy

## Oprogramowanie aplikacji

Heroku – platforma chmurowa stworzona w modelu Platform as a Service obsługująca kilka języków programowania. Aktualnie obsługuje języki Java, JavaScript (Node.js), Scala, Clojure, Python i PHP oraz nieudokumentowany Perl. Podstawowym systemem operacyjnym jest Debian lub bazujący na Debianie Ubuntu.



rys. 1 - architektura Heroku

## Środowisko programisty

IntelliJ IDEA – komercyjne zintegrowane środowisko programistyczne (IDE) dla Javy firmy JetBrains.

Windows 10 – system operacyjny wyprodukowany przez amerykańską międzynarodową firmę technologiczną Microsoft i wydany jako część rodziny systemów operacyjnych Windows NT.

Google Chrome (Wersja 83.0.4103.97) – przeglądarka internetowa rozwijana przez Google. Jej kod został napisany w oparciu o rozwiązania open source częściowo oparte na innych aplikacjach, z wyjątkiem wersji na iOS

# Schemat ideowy aplikacji i hierarchiczny funkcji

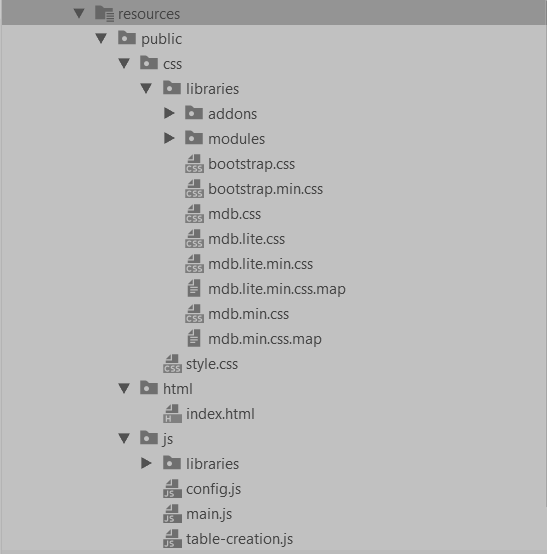
Aplikacja składa się z 8 klas: TaxesInState, ProfitData, Product, Main, ProfitCalculator, CsvReader, CalculatorResult oraz Database.



rys. 2 - diagram klas

Platforma wykorzystuje bibliotekę Bootstrap v4.5.0 – zestaw przydatnych narzędzi ułatwiających tworzenie interfejsu graficznego stron oraz aplikacji internetowych. Bazuje głównie na gotowych rozwiązaniach HTML oraz CSS.

Generowanie dynamicznych elementów witryny odbywa się za pośrednictwem JavaScript – skryptowy język programowania, stworzony przez firmę Netscape, najczęściej stosowany na platformach webowych. W celu usprawnienia działania użyto jQuery – lekka biblioteka programistyczna dla języka JavaScript.



rys. 3 - zasoby platformy

# Opis klas metod i funkcji aplikacji

## public class TaxesInState

Klasa przechowująca informacje o stawkach podatkowych w danym stanie.

private String name;

private double baseTax;

private double maxLocalSurtax;

private String groceries;

private String preparedFood;

private String prescriptionDrug;

private String nonPrescriptionDrug;

private String clothing;

private String intangibles;

private double logisticCost;

public double getLogisticCost()

public String getName()

public double getBaseTaxValue()

public String getTaxForCategory(String category)

## public class ProfitData

Klasa magazynująca dane o zysku w oparciu o stan

private String nameOfState;

private double profit;

private double priceWithoutTaxes;

private double logisticCost;

public String getNameOfState()

public void setNameOfState(String nameOfState)

public double getPriceWithoutTaxes()

public double getProfit()

public void setProfit()

public String toString()

## public class ProfitCalculator

Klasa generująca informacje o cenach produktu w oparciu o stawki podatkowe według określonej kategorii.

private ArrayList<TaxesInState> taxesInStates;

public String CalculateForAllStates(String id, String category, String basePrice, String finalPrice)

## public class Product

Klasa przechowująca informacje o numerze identyfikacyjnym produktu, nazwie, kategorii oraz cenie.

private int id;

private String name;

private String category;

private double price;

public int getId()

public void setId(int id)

public String getName()

public void setName(String name)

public String getCategory()

public void setCategory(String category)

public double getPrice()

public void setPrice(double price)

## public class Main

Główna klasa aplikacji generująca kontent oraz konfigurująca połączenie z platformą Heroku.

public static void main(String[] args)

private static String renderContent(String htmlFile)

private static String getProductsJsonString()

static int getHerokuAssignedPort()

## public class DataBase

Klasa odpowiedzialna za połączenie z bazą danych oraz obsługująca zapytania SQL.

public static Connection getConnection()

public static ArrayList<Product> getProductById(int id)

public static ArrayList<Product> getProductsFromDB()

public static void putProductIntoDB(Product pr)

public void deleteProductById(int id)

## public class CsvReader

Klasa przetwarzająca dane z pliku .csv oraz wczytująca produkty z zewnętrznego pliku do tablicy produktów.

private ArrayList<Product> products;

static ArrayList<Product> readProductsFromFile(String path)