

**Instytut Informatyki  
Kolegium Nauk Przyrodniczych  
Uniwersytet Rzeszowski**

**Dokumentacja techniczna projektu:**

***Interactive Restaurant***

**Wykonał:**

**Maksymilian Ziembicki**

**Rzeszów 2023**

Spis treści

[1. Temat projektu 3](#_Toc158586815)

[2. Cel projektu 3](#_Toc158586816)

[3. Aktualne funkcjonalności aplikacji 3](#_Toc158586817)

[4. Technologie 3](#_Toc158586818)

[5. Interesariusze aplikacji 3](#_Toc158586819)

[6. Harmonogram realizacji projektu 3](#_Toc158586820)

[6.1 planowanie 3](#_Toc158586821)

[6.2 Implementacja 3](#_Toc158586822)

[6.3 Testowanie 4](#_Toc158586823)

[6.4 Korekty i doskonalenie 4](#_Toc158586824)

[6.5 Wykres Gantta 5](#_Toc158586825)

[7. Projekt GUI 5](#_Toc158586826)

[8. Aktualna baza danych 8](#_Toc158586827)

[8.1 Cala baza danych 8](#_Toc158586828)

[8.2 Dania 9](#_Toc158586829)

[8.3 Składniki 9](#_Toc158586830)

[8.4 SkladnikSzablon 10](#_Toc158586831)

[8.5 Zamowienia 10](#_Toc158586832)

[8.6 Uzytkownicy 11](#_Toc158586833)

[8.7 Opis 11](#_Toc158586834)

[8.8 Zastosowanie poszczególnych tabel 12](#_Toc158586835)

[9. Diagramy UML 14](#_Toc158586836)

[9.1 Diagram klas 14](#_Toc158586837)

[9.2 Diagram przypadków użycia 15](#_Toc158586838)

[9.3 Definicja aktorów 15](#_Toc158586839)

[9.4 Definicje scenariuszy przypadków użycia 16](#_Toc158586840)

[10. Plan rozbudowy aplikacji 18](#_Toc158586841)

[11. Opis funkcjonalności, które będą wprowadzane w przyszłości 19](#_Toc158586842)

# Temat projektu

Tematem projektu jest aplikacja służąca do zarządzania zasobami restauracji oraz udostępniania szczegółowych informacji na temat restauracji.

# Cel projektu

Celem projektu jest usprawnienie działania restauracji, sprawniejsze zarządzanie oraz dostarczenie wielu szczegółowych danych na temat restauracji a w szczególności aspektu finansowego

# Aktualne funkcjonalności aplikacji

* Navbar, który sprawnie nawiguje nas po strukturze strony internetowej
* Panel magazynu, który przechowuje informacje o aktualnych produktach spożywczych umieszczonych w magazynie restauracji
* Panel składników, który służy nam do tworzenia nowych dań
* Filtracja składników po ich kategorii
* Panel zamówień, w którym możemy tworzyć zamówienia
* Obliczanie wartości zamówienia na podstawie dań w „koszyku”
* Możliwość zarządzania „koszykiem”
* Panel zamówień, który wyświetla szczegółowe dane na temat zamówień

# Technologie

JavaScript, Node.JS, HTML, CSS, Express.JS, MongoDB, Mongoose, EJS, Nodemon

# Interesariusze aplikacji

Personel restauracji.

# Harmonogram realizacji projektu

## 6.1 planowanie

1. **Rozmowa z opiekunem proseminarium**

* Ustalenie tematu oraz funkcjonalności projektu

1. **Analiza wymagań i celów**

* Zdefiniowanie funkcji i cech aplikacji
* Określenie docelowej grupy użytkowników

1. **Tworzenie specyfikacji**

* Dokładne opisanie funkcji i interfejsu użytkownika.
* Określenie technologii, które będą używane.

1. **Ustalenie harmonogramu**

* Określenie dat kluczowych etapów projektu.

## 6.2 Implementacja

1. **Nauka języka**
   * Research języków programowania
   * Wybranie odpowiedniego języka
   * Realizacja kursów
   * Ćwiczenia we własnym zakresie
2. **Budowa podstawowej struktury aplikacji**

* Stworzenie struktury plików
* Instalacja odpowiednich pakietów
* Tworzenie szkieletu HTML
* Tworzenie struktury bazy danych
* Tworzenie arkuszy stylów
* Zastosowanie architektury MVC

1. **Navbar**
   * Utworzenie panelu nawigacyjnego
   * Dodanie funkcji responsywności
2. **Panel składników**
   * Utworzenie podstrony do zarządzania składnikami
   * Integracja z bazą danych
   * Implementacja funkcjonalności panelu składników
3. **Panel magazynu**
   * Utworzenie panelu magazynu
   * Połączenie panelu magazynu z modelem składników
   * Utworzenie podstron dokładnych statystyk
4. **Panel dań**
   * Utworzenie panelu tworzenia dań
   * Interakcja z bazą danych
   * Utworzenie funkcjonalności koszyka
5. **Panel zamówień**

* Utworzenie panelu zamówień

## 6.3 Testowanie

1. **Testy jednostkowe**
   * Testowanie poszczególnych komponentów aplikacji.
2. **Testy integracyjne**
   * Sprawdzenie współpracy między różnymi częściami aplikacji.

## 6.4 Korekty i doskonalenie

1. **Optymalizacja**
   * Poprawa wydajności i optymalizacja kodu.

## 6.5 Wykres Gantta

# Projekt GUI

* **Strona startowa**

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

* **Panel Magazynu**

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

* **Panel składników**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Strona internetowa

Opis wygenerowany automatycznie

* **Panel kreowania dania**

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

* **Panel zamówień**

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

# Aktualna baza danych

## 8.1 Cala baza danych

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

## 8.2 Dania

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie**

## 8.3 Składniki

**Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie**

## 8.4 SkladnikSzablon

**Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie**

## 8.5 Zamowienia

**Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie**

## 8.6 Uzytkownicy

**Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie**

## 8.7 Opis

MongoDB to popularna, nierelacyjna baza danych typu NoSQL, która jest oparta na modelu dokumentowym. Poniżej opisuję podstawowe koncepcje i sposób działania bazy MongoDB:

**Dokumenty:**

W MongoDB dane przechowywane są w formie dokumentów BSON (Binary JSON), które są zbliżone do formatu JSON.

Dokumenty składają się z pól i wartości, a każdy dokument jest unikalny w obrębie kolekcji (analogii do tabel w relacyjnych bazach danych).

**Kolekcje:**

Kolekcje w MongoDB są zbiorem dokumentów. Są to odpowiedniki tabel w relacyjnych bazach danych, ale w MongoDB nie mają określonej struktury.

Kolekcje umożliwiają grupowanie dokumentów na podstawie wspólnego kontekstu.

**Bazy danych:**

Baza danych w MongoDB to zbiór kolekcji. Każda baza danych jest niezależna i może zawierać wiele kolekcji o różnych strukturach.

**ID dokumentów:**

Każdy dokument w kolekcji ma unikalny identyfikator, który domyślnie jest polem o nazwie "\_id".

Można używać różnych typów danych jako identyfikator, ale najczęściej stosowanym jest ObjectI

## 8.8 Zastosowanie poszczególnych tabel

**Dishes:**

Dokument dishes zawiera następujące informacje:

* \_id: id dania
* Name: nazwa dania
* Price: cena dania
* ingredientTemplates: informacje o składnikach z których zrobione jest danie, ceny skladnikow oraz ich ilość

**Ingredient-templates:**

Dokument ingredient-template ma na celu przechowywanie danych, które następnie pomogą w organizacji magazynu oraz tworzenia nowych dań. Oto jej skład:

* \_id: id szablony składnika
* Name: nazwa szablonu składnika (nazwa składnika)
* expirationDate: data ważności
* category: kategoria

**Ingredients:**

Dokument zawiera informacje o składnikach, które są aktualnie przechowywane w magazynie

* \_id: id składniku
* Name: nazwa składniku
* Price: cena składniku
* Weight: waga składniku
* expirationDate: data ważności
* category: kategoria

**Orders:**

Dokument zawiera informacje o zamówieniach:

* \_id: id zamówienia
* Dishes: kolekcja, która zawiera szczegółowe dane o daniach które wchodzą w skład zamówienia
* Price: cena zamówienia

**Users:**

Zawiera informacje o użytkowniku

* Name: nazwa użytkownika
* Email: email użytkownika
* DishCart: koszyk, który pozwala na tworzenie zamówienia
* IngredientCart: koszyk, który pozwala na tworzenie dania

1. Architektura aplikacji

Aplikacja oparta jest na architekturze MVC, która jest popularnym wzorcem projektowym wykorzystywanym do strukturyzacji kodu źródłowego. MVC dzieli aplikację na trzy główne komponenty: Model, View i Controller, co ułatwia zarządzanie logiką biznesową, warstwą prezentacji i kontrolą przepływu danych.

**Model** odpowiada za zarządzanie danymi i logiką biznesową aplikacji. W moim przypadku:

Model danych: Przechowuje dane, zarządza dostępem do bazy danych (jeśli jest używana), i definiuje logikę biznesową. Wszelkie zmiany danych są odzwierciedlane w Modelu.

Interakcje z bazą danych: Odpowiada za pobieranie, aktualizację, dodawanie i usuwanie danych w bazie.

Definiuje strukturę komórki w dokumencie

W mojej aplikacji posiadam 5 modeli: dish, ingredient, ingredientTemplate, order oraz user które odpowiadają poszczególnym dokumentom w bazie danych.

W warstwie **View** znajdują się elementy związane z prezentacją danych użytkownikowi. W naszym przypadku:

Szablony HTML/CSS/JavaScript: Odpowiadają za prezentację danych użytkownikowi.

Wyświetlanie danych: View otrzymuje dane z Modelu i prezentuje je w zrozumiały sposób dla użytkownika.

W mojej aplikacji widoku podzielone sa na katalogi:

Admin: który zawiera widoku podstron do zarządzania zasobami

Includes: który zawiera fragmenty kodu powtarzające się na podstronach np. footer i navbar

Restaurant: który zawiera widoki dostępne dla gości aplikacji webowej

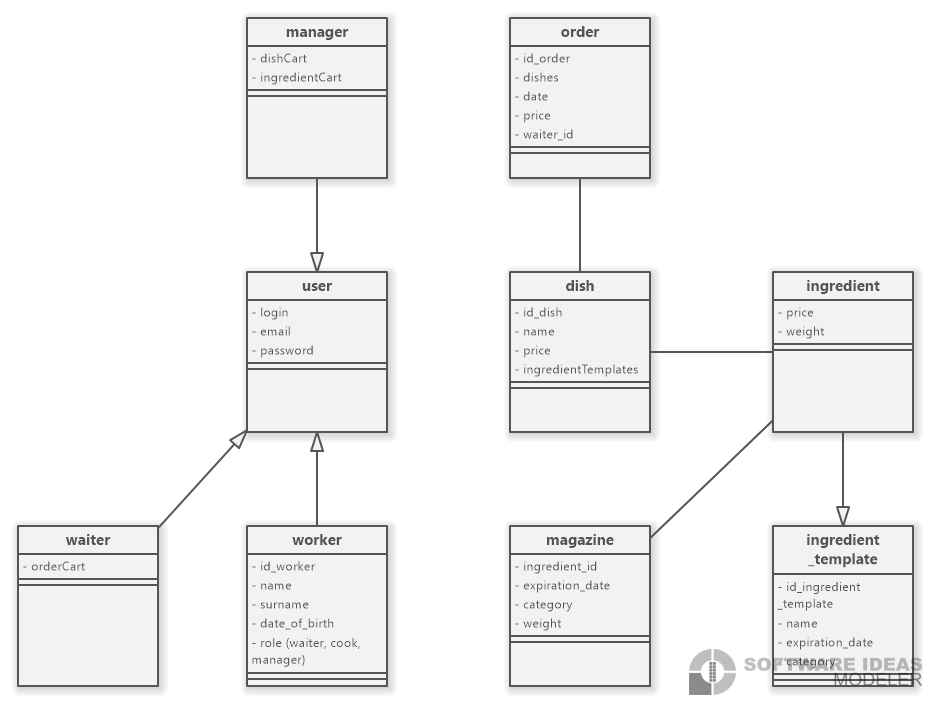
**Kontroler** zarządza przepływem danych pomiędzy Modelem a Widokiem. W naszym przypadku:

Obsługa żądań użytkownika: Kontroler reaguje na żądania użytkownika (np. HTTP Request w aplikacjach webowych).

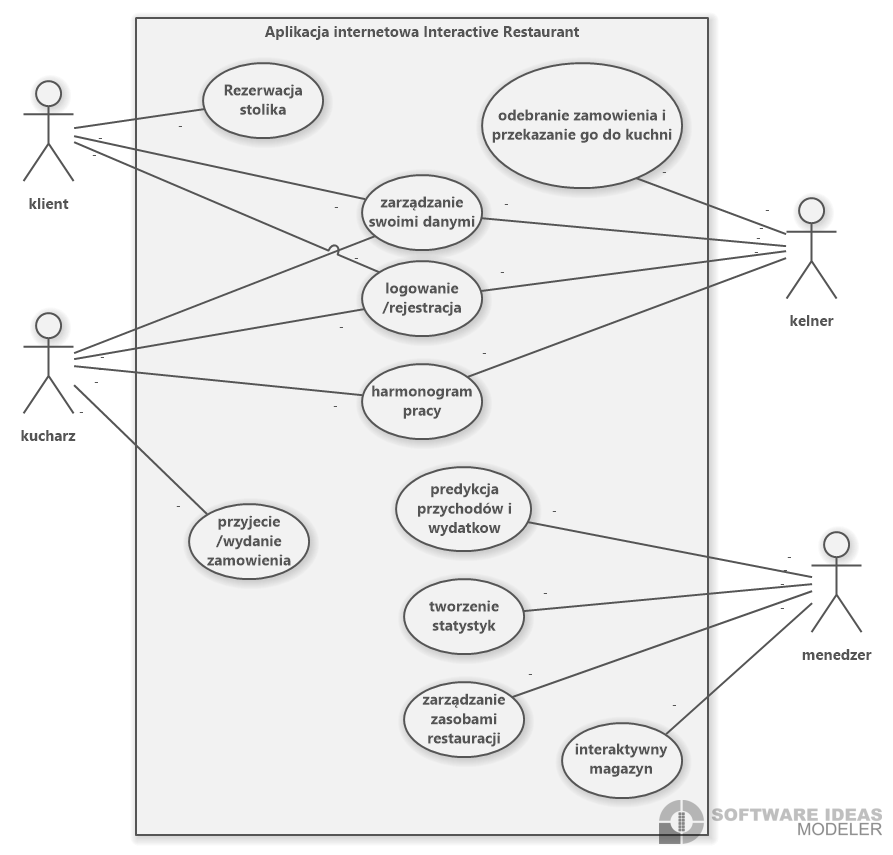
Przetwarzanie danych: Kontroler przetwarza dane wejściowe od użytkownika, komunikuje się z Modelem w celu uzyskania/zmiany danych i aktualizuje Widok.

# Diagramy UML

## 9.1 Diagram klas



## 9.2 Diagram przypadków użycia



## 9.3 Definicja aktorów

**AKTOR:** Klient

**OPIS:** Klient może utworzyć konto, zarządzać swoimi danymi oraz zarezerwować stolik

**PRZYPADKI UŻYCIA:**

• PU Logowanie/rejestracja

• PU Zarządzanie swoimi danymi

• PU Rezerwacja stolika

**AKTOR:** Kucharz

**OPIS:** Klient może utworzyć konto, zarządzać swoimi danymi, uzupełniać harmonogram prac, przyjmować oraz wydawać zamówienia

**PRZYPADKI UŻYCIA:**

• PU Logowanie/rejestracja

• PU Zarządzanie swoimi danymi

• PU Harmonogram pracy

• PU Przyjmowanie zamówienia

• PU Wydanie zamówienia

**AKTOR:** Kelner

**OPIS:** Klient może utworzyć konto, zarządzać swoimi danymi, uzupełniać harmonogram pracy oraz odbierać zamówienia od klientów i przekazywać je kuchni

**PRZYPADKI UŻYCIA:**

• PU Logowanie/rejestracja

• PU Zarządzanie swoimi danymi

• PU Harmonogram pracy

• PU Odebranie zamówienia i przekazanie go do kuchni

**AKTOR:** Menedżer

**OPIS:** Menedżer może za pomocą poszczególnych paneli przedyktować bilans finansowy restauracji, posiadać dostęp do obszernych statystyk, monitorować zasoby restauracji oraz zarządzać nimi.

**PRZYPADKI UŻYCIA:**

• PU Predykcja przychodów i wydatków

• PU Tworzenie statystyk

• PU Zarządzanie zasobami restauracji

• PU Interaktywny magazyn

## 9.4 Definicje scenariuszy przypadków użycia

**PU LOGOWANIE**

**CEL:** Zalogowanie do systemu

**PRZEBIEG:**

Logowanie do systemu jest dostępne poprzez zakładkę logowanie. Użytkownik musi podać

email oraz hasło, które trzeba jeszcze dodatkowo potwierdzić. Odpowiednie alerty

„wyskakują”, w zależności od stanu powodzenia rejestracji/logowania. Użytkownik ma 5 prób

logowania, jeśli przekroczy ta ilość to zostanie przeniesiony na stronę główna. Gdy użytkownik

pomyślnie się zaloguje/zarejestruje zostanie przeniesiony na stronę ze swoimi danymi

osobowymi. Gdy administrator się zaloguje zostanie on przeniesiony do panelu administratora

**PU REZERWACJA STOLIKA**

**CEL:** Zarezerwowanie stolika

**PRZEBIEG:**

Klient za pomocą specjalnego panelu może zobaczyć rozstaw stolików w restauracji, ich dostępność oraz dokonać rezerwacji stolika na określoną datę i na określony czas.

**PU ZARZĄDZANIE SWOIMI DANYMI**

**CEL:** Manipulacja swoimi danymi

**PRZEBIEG:**

Użytkownik, który posiada konto ma dostęp do panelu zarządzającego jego danymi takimi jak np. hasło.

**PU HARMONOGRAM PRACY**

**CEL:** utworzenie harmonogramu w celu monitorowania pracowników.

**PRZEBIEG:**

Harmonogram pracy dostarcza informacje kiedy jaki pracownik pracował i ile pracował. Pracownicy wpisują w harmonogram ile pracowali oraz kiedy. Ponadto mają możliwość podglądu swoich godzin pracy oraz manipulowanie nimi.

**PU PRZYJĘCIE/WYDANIE ZAMÓWIENIA**

**CEL:** odebranie zamówienia przez kucharzy i rozpoczęcie jego realizacji

**PRZEBIEG:**

Kelner odbiera zamówienie od stolika i przekazuje go kuchni. Kucharze przygotowują dane, następnie ustawiają status jako „gotowe do odbioru”. Gdy kelner odbierze wszystkie dania wchodzące w skład zamówienia to zamówienie zaksięguje się w systemie.

**PU ODEBRANIE ZAMÓWIENIA I PRZEKAZANIE GO DO KUCHNI**

**CEL:** odebranie zamówienia od kelnera i przekazanie go do kuchni

**PRZEBIEG:**

Kelner pobiera od stolika zamówienie i następnie podaje je do kuchni.

**PU PREDYKCJA PRZYCHODÓW I WYDATKÓW**

**CEL:** Przewidywanie budżetu restauracji na podstawie przychodów i wydatków.

**PRZEBIEG:**

Menedżer ma dostęp do panelu, który na podstawie algorytmu analizującego historie dochodów restauracji dostarcza przewidywania budżetu.

**PU DOSTARCZANIE STATYSTYK**

**CEL:** Dostarczenie menedżerowi szczegółowych statystyk

**PRZEBIEG:**

Menedżer ma dostęp do panelu, który zbiera i przedstawia szczegółowe statystyki dotyczące działania restauracji. Ma to na celu pomóc menedżerowi w określeniu strategii restauracji

**PU ZARZĄDZANIE ZASOBAMI RESTAURACJI**

**CEL:** Zarządzanie zasobami restauracji

**PRZEBIEG:**

Menedżer ma możliwość sprawnego zarządzania zasobami restauracji takimi jak: dania, składniki, zamówienia oraz dane pracowników.

**PU INTERAKTYWNY MAGAZYN**

**CEL:** Dostarczanie informacji o stanie magazynu.

**PRZEBIEG:**

System udostępnia menedżerowi szczegóły dotyczące stanu magazynu produktów, gdzie widzi dokładne dane o ilości zasobów.

# 10. Plan rozbudowy aplikacji

**Zamówienia i sprzedaż:**

Dzienny/tygodniowy/miesięczny obrót.

Najpopularniejsze dania

Średnia wartość zamówienia.

Trendy sprzedaży w czasie.

Predykcje sprzedażowe

Modyfikowanie menu

**Magazyn i zaopatrzenie:**

Poziom zapasów.

Najczęściej zamawiane produkty.

Historia dostaw i koszty zaopatrzenia.

Obliczanie strat związanych z marnowaniem jedzenia

**Personel:**

Wydajność pracowników.

Grafik pracy.

**Finanse:**

Rentowność danego dania lub kategorii produktów.

Analiza marży.

Płatności cykliczne

Obliczanie opłacalności biznesu

Predykcje finansowe

**Zarządzanie rezerwacjami:**

Ilość rezerwacji online i offline.

Czas oczekiwania na stolik.

Historia rezerwacji klientów.

**Połączenie pracy menadżera, kucharza oraz kelnera za pomocą interfejsu, który przetwarza zamówienia**

# 11. Opis funkcjonalności, które będą wprowadzane w przyszłości

**obliczanie ceny dania:**

składnik w magazynie: 5kg pomidorów, cena 200zl

składnik w jednym daniu: pomidor 100g

cena składnika w daniu: 5000g = 200zl, 100g = 4zl

cena dania += cena pomidorów

**miesięczne cele sprzedaży:**

albo importuje się z symulacji albo użytkownik je ustala z góry

cel: sprzedanie 200 w miesiąc

tablica z liczbami dni pracy (będzie można je edytować) w panelu admina

załóżmy z w maju jest 20 dni pracy: 200/20 = 10

sprzedane danie=200-1, pod koniec dnia pracowania restauracji 200-sprzedane,np 200-4=196

następnego dnia: 196/19 = 10.3 = 11, zawsze zaokrąglam w górę

panel celów sprzedaży na każdy dzień miesiąca

**panel dania:**

żądanie ze strony: id\_dania:

wyszukiwanie dania na podstawie id

dołączanie innych informacji na temat dania za pomocą populate

marża = cena dania – suma Składników

odpowiedz serwera: marża oraz informacje na temat dania

**obliczanie przychodu:**

żądanie: obszar Czasowy (np. rok, kwartał itp. wybieram po prostu jaki przedział czasowy)

żądanie: danie

wyszukiwanie zamówień danego dania i w wybranym okresie czasowym.

suma = 0

wykonanie pętli bo wszystkich zamówieniach

suma += marża (cena dania - cena składników)

suma = tyle ile restauracja zarobi na tym daniu w określonym przedziale czasu

odpowiedz serwera: suma[z przedziałami czasowymi]

**tworzenie wykresu:**

będzie można oglądać ile się chce ale dla uproszczenia podam przykład na rok

żądanie: okres czasowy, danie

okres Czasu = [12], 12->length

wyszukiwanie zamówień danego dania i w danym okresie

następuje pętla po zamówieniach

suma += marża (cena dania - cena składników)

jeżeli data zamówienia = 01\* (styczeń)

okresCzasu[1] = suma

odpowiedz serwera: okresCzasu[]

**zmiana ilości porcji oraz symulacja:**

działanie:

wybranie dania , zmiana porcji składników czymś w rodzaju suwaka

np. każdego składnika jest o 10% mniej przez co cena produkcji spada o X%

pobranie aktualnego miesiąca np. luty i z requesta jaki czas ma pokazywać symulacja, np. 3 miesiące

pseudokod:

żądanie: id\_dania, okres Czasu, procent Zmniejszenia Porcji

wyszukiwanie dania po id\_dania

pobranie informacji o składnikach dania za pomocą populate()

następuje pętla po składnikach

cena składniku -= 10%

zapisanie informacji o zmienionym daniu do szablonu dania, nie następuje zmiana danych prawdziwego wpisu dania w bazie

następnie:

pobranie danych zmienionego dania

pobieram dane ile razy to danie zostało sprzedawane w 3 następnych miesiącach czyli marcu kwietniu i maju

**Funkcja porównywania lat**

i zamieniam danie z zapisanym wcześniej szablonem dania

następnie oblicza potencjalna zarobioną kwotę za 3 następne miesiące na podstawie nowych cen

np. w 3 miesiące było 1000 zamówień no to 1000\*nowa Cena Dania

**Porównywanie lat**

styczeń 100

luty 120

marzec 90

kwiecień 105

120/100 \* 100 = 120 - wzrosło o 20%

90/120 \* 100 = 75 - zmalało 0 25%

105/90 \* 100 = 116,6 - wzrosło o 16,6%

wejście [100, 120, 90, 105] -> liczba zamówień dania na miesiąc

wyniki [0, 120, 75, 116,6] -> tabela zmian procentowych

zamówienia[24], ponieważ 2 lata = 24 miesiące

żądanie: przedział czasowy

wyszukiwanie zamówień z poprzednich 2 lat

następuje pętla która będzie się wykonywać tyle ile wynosi length tablicy zamówienia (24)

zamówienia[i] = liczba Zamówień Z Poszczególnego Miesiąca

różnice[24] -> tabela która będzie przechowywała różnice pomiędzy miesiącami

następuje pętla która będzie się wykonywać tyle ile wynosi length tablicy zamówienia (24)

różnice[i] = zamówienia[i+1]/zamówienia[i]\*100

poprzednieLata[12]

następuje petla po zamowieniach

poprzednieLata[i] = (zamowienie[i]+zamowienie[i+12])/2

pobranie miesięcy z aktualnego roku:

let tenRok[12]

wyszukuje zamówienia z aktualnego roku

następuje pętla po tablicy tenRok

tenRok[i] = zamówienia[i].liczba Zamówień Z Poszczególnych Miesięcy

let roznica

nastepuje petla po tablicy tenRok

roznica = tenRok[i] - poprzednieLata[i]

faktyczna Różnica = różnice/ilość miesięcy jakie upłynęły w tym roku

i teraz do różnic w dwóch ostatnich latach następuje dodanie faktycznej różnicy i na tym polega symulacje aktualnego roku

następuje pętla po tablicy porzednieLata

poprzednieLata[i] += faktycznaRoznica

**Przykład:**

Różnice miesięcy z dwóch poprzednich lat

[0, 110, 90, 103,4]

[0, 120, 80, 116,6] +

---------------------

[0. 230, 170, 220]

Dzielenie przez 2

[0, 115, 85, 110] -> zbliżone różnice miedzy miesiącami

miesiące z aktualnego roku, obliczanie różnic

[0, 120, 80, 120]

aktualny rok jest odpowiednio lepszy o 5%, -5%, 10%

sumowanie do siebie różnic = 10%

dzielenie na 3 miesiące w celu uzyskania ile średnio wyniki różnią się od poprzednich lat, 10/3=3.3%

patrząc na to ze w aktualnym roku wyniki są lepsze o 2.5% to uwzględniam to w predykcjach

wiec zamiast odpowiednio 115, 85 i 110 będzie 118,3, 88,3 i 113,3 - to pozwala uzyskać dokładniejszy wynik

**Zmiana ceny dania**

**Cel:**

Obniżenie ceny dania o 1zł i uzyskanie obrazu jaki wpływ ma zmiana ceny na popularność dania z uwzględnieniem, że w przeszłości taka procedura miała miejsce natomiast odbyła się ona w lutym a przyjmuje ze aktualnie jest lipiec

Pobranie informacji o lutym oraz następnych trzech miesiącach

luty 100

marzec 110

kwiecień 115

maj 125

obliczanie miesięcznych przyrostów

110/100 \* 100 = 110

115/110 \* 100 = 104.54

125/115 \* 100 = 108.63

wyniki[0, 110, 104.54, 108.63]

pobranie wyników różnicy miesięcy w skali 2 poprzednich lat i ich średnia np. [0, 103, 101, 104]

wiec wychodzi na to ze zamiast wzrostów na poziomie 10%, 4.5%, 8.6% są zmiany w wysokości 7%, 1.5% oraz 4.63% i są to wyniki bardziej wiarygodne