System wspomagający składanie i realizacje zamówień w restauracji

**Testy sprawdzające**

Spis treści

[Implementacja aplikacji 3](#_Toc137299252)

[Front end 3](#_Toc137299253)

[Back end 7](#_Toc137299254)

[Testy funkcjonalne 12](#_Toc137299255)

[Funkcjonalność klienta – przegląd menu i składanie zamówień 12](#_Toc137299256)

[Przegląd menu 12](#_Toc137299257)

[Szczegóły dania 13](#_Toc137299258)

[Dodanie do koszyka 13](#_Toc137299259)

[Koszyk 14](#_Toc137299260)

[Finalizacja zamówienia 16](#_Toc137299261)

[Wynik zamówienia w bazie danych 17](#_Toc137299262)

[Funkcjonalność pracownika – podgląd zamówień, przypisywanie dostawców i aktualizacja statusów 17](#_Toc137299263)

[Logowanie 17](#_Toc137299264)

[Panel pracownika 19](#_Toc137299265)

[Podgląd zamówienia 20](#_Toc137299266)

[Przypisanie dostawcy 20](#_Toc137299267)

[Zmiana statusu zamówienia 21](#_Toc137299268)

[Podsumowanie 21](#_Toc137299269)

# Implementacja aplikacji

## Front end

Interfejs graficzny aplikacji został stworzony przy użyciu HTML, CSS i JavaScript używając biblioteki React.js. Komunikacja z back endem aplikacji została zrealizowana za pomocą biblioteki jQuery, a dokładniej funkcji AJAX. Do front endu strony można uzyskać dostęp po uruchomieniu aplikacji w edytorze kodu poleceniem „npm start” na adresie przeglądarki <http://localhost:3000>. Routing podstron zrealizowano przy użyciu pakietu react-router-dom.

Przeważające kolory strony to #D3C4AE jako tło paska nawigacyjnego, #F2E1CC jako tło głównej części strony, #E5CDAE jako tło stopki oraz #D4C5AF jako tło wpisów dań w menu i w koszyku.

**Widoki strony:**

* Strona głównaObraz zawierający tekst, danie, Kuchnia, jedzenie

  Opis wygenerowany automatycznie
* Menu z daniami Obraz zawierający tekst, zrzut ekranu, oprogramowanie, komputer

  Opis wygenerowany automatycznie
* Szczegóły daniaObraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

  Opis wygenerowany automatycznie
* Strona „o nas” Obraz zawierający tekst, elektronika, zrzut ekranu, oprogramowanie

  Opis wygenerowany automatycznie
* Kontakt Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Ikona komputerowa

  Opis wygenerowany automatycznie
* Koszyk  
  Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Strona internetowa

  Opis wygenerowany automatycznie
* Finalizacja zamówienia  
  Obraz zawierający tekst, zrzut ekranu, oprogramowanie, wyświetlacz

  Opis wygenerowany automatycznie
* Logowanie dla pracownikówObraz zawierający tekst, zrzut ekranu, wyświetlacz, oprogramowanie

  Opis wygenerowany automatycznie
* Panel pracownikaObraz zawierający tekst, zrzut ekranu, Równolegle, design

  Opis wygenerowany automatycznie
* Podgląd zawartości zamówienia Obraz zawierający tekst, zrzut ekranu, diagram

  Opis wygenerowany automatycznie
* Przypisanie dostawcy do zamówieniaObraz zawierający tekst, zrzut ekranu, Czcionka

  Opis wygenerowany automatycznie
* Zmiana statusu zamówieniaObraz zawierający tekst, zrzut ekranu, oprogramowanie

  Opis wygenerowany automatycznie

## Back end

Back end strony został zrealizowany z wykorzystaniem pakietu XAMPP. Kod serwera powstał przy użyciu języka PHP, a bazę danych utworzono w systemie MySQL. Warstwa klienta, jak już wyżej wspomniano, komunikuje się z serwerem przy użyciu AJAX. Do serwera przekazywane są informacje, jakich danych zwrotnych potrzebuje klient, oraz opcjonalnie przekazywane są informacje potrzebne przy komunikacji z bazą danych, takie jak dane logowania użytkownika czy dane zamówień. W odpowiedzi są wysyłane klientowi dane zakodowane jako ciąg znaków JSON, które to po stronie front endu są odczytywane jako treść strony. Zdjęcia natomiast są odczytywane przez serwer PHP w oparciu o ścieżkę do zdjęcia zawartą w bazie danych, kodowane w formacie Base64 i przesyłane jako część odpowiedzi.

**Diagram relacji bazy danych** Obraz zawierający tekst, zrzut ekranu, Czcionka, Strona internetowa

Opis wygenerowany automatycznieTesty poza funkcjonalne

Testy poza funkcjonalne zostały przeprowadzone z użyciem narzędzia Google Lighthouse, a wyniki są podane poniżej.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

W wypadku testów wydajności uzyskała ona wynik 87/100, co jest dobrym, ale nie świetnym wynikiem. Jako potencjalne rozwiązania problemów wydajności podane zostały zmiana rozmiaru obrazów (0,36s oszczędności), wstępne wczytywanie elementu LCP, czyli obrazu największego wyrenderowania treści (0,29s), minifikacja plików JavaScript (0,20s) i ograniczenie nieużywanego JS (0,20s). Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

W testach ułatwień dostępu aplikacja uzyskała wynik 100/100, czyli doskonały. Jako zaliczone testy zostały wymienione m.in. podanie atrybutu „lang” znacznika <html> wraz z wartością, zawarcie znacznika <title>, atrybuty „alt” elementów graficznych, atrybuty „id” znaczników są unikalne oraz jest wystarczający współczynnik kontrastu między kolorami tła i pierwszego planu. Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

W testach pod kątem sprawdzonych metod strona ponownie uzyskała wynik 100/100. Jako zaliczone audyty zostały podane m.in. uwzględniona deklaracja <!DOCTYPE html>, brak użycia wycofanych interfejsów API, brak zarejestrowanych przez konsolę błędów przeglądarki i brak problemów w panelu „Issues” narzędzi deweloperskich Chrome oraz właściwa definicja zestawu znaków.  
Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

W testach pod kątem optymalizacji dla wyszukiwarek (SEO) ponownie został uzyskany wynik maksymalny 100/100. W tym wypadku zaliczone audyty to m.in. zawarty znacznik <meta name=”viewport”> z elementami „width” i „initial-scale”, zawarty element <title>, zawarty metaopis, niezablokowane indeksowanie strony, atrybuty „alt” elementów graficznych i brak wymagań wtyczek. Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Natomiast testy pod kątem progresywnej aplikacji internetowej nie zostały przeprowadzone z powodu nie spełniania wymagań instalowalności przez plik manifestu aplikacji lub skrypt service worker.

# Testy funkcjonalne

## Funkcjonalność klienta – przegląd menu i składanie zamówień

### Przegląd menu

Po otwarciu podstrony menu poprawnie pobierają się z bazy danych i serwera ze zdjęciami i wyświetlają się wszystkie dania – ich zdjęcie, nazwa, krótki opis i cena oraz przyciski przejścia do szczegółów dania i dodania go do koszyka.  
Obraz zawierający tekst, menu, posiłek, Fast food

Opis wygenerowany automatyczniePo zmianie kategorii na Dania główne, tak jak powinny, zostały tylko Kotlet schabowy i Rosół, a zniknęły Paszteciki, które są w bazie danych przystawką, oraz Sernik, czyli deser.  
Obraz zawierający tekst, zrzut ekranu, jedzenie

Opis wygenerowany automatycznie

### Szczegóły dania

Po kliknięciu w przycisk Szczegóły, poprawnie otwiera się karta dania, na której są podane zdjęcie, cena, nazwa dania, pełen opis i składniki.  
Obraz zawierający tekst, zrzut ekranu, jedzenie, menu

Opis wygenerowany automatycznie

### Dodanie do koszyka

Po wciśnięciu na karcie dania „Dodaj do koszyka” pojawia się komunikat o dodaniu dania do koszyka. Obraz zawierający tekst, jedzenie, zrzut ekranu, Fast food

Opis wygenerowany automatycznie  
Podobny rezultat osiągamy przy dodaniu do koszyka z poziomu menu. Obraz zawierający tekst, zrzut ekranu, jedzenie

Opis wygenerowany automatycznie

### Koszyk

W koszyku zgodnie z moim dodawaniem do niego wcześniej wyświetlają się nam wszystkie dania które dodaliśmy, cena za jedną sztukę dania razy ilość sztuk w koszyku oraz poniżej całkowita cena zamówienia i możliwość przejścia do finalizacji zamówienia. Obraz zawierający tekst, jedzenie, zrzut ekranu, Fast food

Opis wygenerowany automatycznie

Po zmianie ilości pasztecików na 1 i zatwierdzeniu zmiany przyciskiem Zmień ilość, strona się odświeżyła i podała nową cenę pasztecików i nową całkowitą cenę zamówienia. Obraz zawierający tekst, zrzut ekranu, jedzenie, Fast food

Opis wygenerowany automatycznie   
Gdy kliknąłem przy Serniku „Usuń z koszyka”, rezultat był również oczekiwany – sernik zniknął z koszyka, a strona się odświeżyła bez niego. Obraz zawierający tekst, Fast food, zrzut ekranu, jedzenie

Opis wygenerowany automatycznie

Czas na sprawdzenie co się stanie, gdyby podać pożądaną ilość dania ujemną (np. -1 kotlet schabowy). Obraz zawierający tekst, Fast food, zrzut ekranu, jedzenie

Opis wygenerowany automatycznie

Po zatwierdzeniu zmiany ilości jak można się spodziewać zniknął z koszyka. Niestety znalazłem pierwszy problem funkcjonalny strony – po kliknięciu raz na dodanie do koszyka kotleta, nie pojawił się on ponownie. Dopiero drugi raz, gdy jego ilość w koszyku stała się wartością dodatnią, pojawił się. Ma to związek z mechanizmem zapamiętywania koszyka – zapamiętywany jest on jako ciasteczko z tyloma wierszami, ile jest dań, a przy każdym daniu jest jego ilość w koszyku. Zapomniałem zaimplementować blokady, że w razie wykrycia wartości ujemnej jest ona resetowana na 0. W każdym razie zmieńmy ponownie kotlet schabowy na -1 i przejdźmy do finalizacji zamówienia, zweryfikujemy reakcję serwera na taką informację.

### Finalizacja zamówienia

Tutaj wita nas formularz, w którym podajemy dane kontaktowe. Niestety pojawił się kolejny problem – w razie braku podanych danych kontaktowych zamówienie i tak zostaje złożone. Spowodowane jest to najpewniej tym, że choć pola formularza są ustalone jako required, to formularza nie wysyłam przyciskiem submit, a zdarzeniem OnClick zwykłego przycisku, przez co nie są weryfikowane wartości pól. W każdym razie po złożeniu zamówienia pokazuje się alert potwierdzający ten fakt. Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

Ponadto przekierowanie na stronę główną po złożeniu zamówienia nie działa, pojawia się błąd React.

### Wynik zamówienia w bazie danych

Obraz zawierający tekst, zrzut ekranu, linia, numer

Opis wygenerowany automatycznie

Jak widać powyżej, zamówienie zostało dodane, niestety bez danych kontaktowych (przez nie podanie ich w formularzu) oraz z ujemną ceną (przez -1 kotlet schabowy w koszyku).

W tabeli przypisań dan do zamówień natomiast jest poprawnie, tylko jeden zestaw pasztecików jest przypisany do naszego zamówienia.

Obraz zawierający tekst, Czcionka, linia, zrzut ekranu

Opis wygenerowany automatycznie

## Funkcjonalność pracownika – podgląd zamówień, przypisywanie dostawców i aktualizacja statusów

### Logowanie

Przy logowaniu serwer sprawdza dane logowania i porównuje je z bazą danych – czy jest taki użytkownik, a jeśli jest to czy jego hasło zgadza się z podanym czy nie. Rezultat jest taki, że nawet gdy nie podałem wszystkich danych, to żądania AJAX są przekazywane do serwera, co może być wprawdzie małym, lecz niepotrzebnym obciążeniem. Ponadto z powodu małego błędu w kodzie weryfikacji odpowiedzi serwera wszystkie żądania, nawet niewłaściwe, skutkowały zalogowaniem do systemu.

W przypadku podania nieistniejącego pracownika bądź złego hasła dla istniejącego pojawia się odpowiedni komunikat.Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Natomiast gdy podajemy właściwe dane, zostajemy prawidłowo zalogowani do systemu z komunikatem, jaki użytkownik się właśnie loguje. Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Następnie zostajemy przekierowani do panelu głównego pracownika.

### Panel pracownika Obraz zawierający tekst, zrzut ekranu, Równolegle, numer Opis wygenerowany automatycznie

Domyślnie pokazane są tylko zamówienia w trakcie realizacji, lecz można kliknąć przycisk „Wszystkie zamówienia”, by przełączyć się na wszystkie. Wtedy pokazują się te dwa poniżej.Obraz zawierający tekst, zrzut ekranu, linia, Równolegle

Opis wygenerowany automatycznie

Kliknięcie na „Niezrealizowane zamówienia” prawidłowo przywraca nas do startowego widoku.

### Podgląd zamówienia

Kliknięcie w „Zawartość zamówienia” pokazuje nam dokładnie to, co jest napisane na przycisku.  
Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

### Przypisanie dostawcy

Po kliknięciu w „Przypisz dostawcę” pokazuje nam się lista dostawców w systemie. Wyszarzeni są dostawcy niedostępni, i można wybrać tylko dostępnych. Przypisanie dostawcy do zamówieni wiąże się z zablokowaniem go przed przypisywaniem do kolejnych zamówień, dopóki nie będzie znowu dostępny). Przypiszę teraz dostawcę do tego zamówienia, a potem zobaczę, czy można go przypisać chwilę później do innego.   
Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Jak widać, teraz już nie mogę przypisać Dariusza Ostawczaka do drugiego zamówienia, bo już jest zajęty z jednym.

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Teraz spróbuję nie przypisać żadnego dostawcy i zatwierdzić wybór.  
Obraz zawierający tekst, elektronika, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

Jak widać, zapomniałem uwzględnić tego przypadku – pojawia się błąd.

### Zmiana statusu zamówienia

Po kliknięciu w „Zmień status zamówienia” pojawia się nam okno ustalenia statusu zamówienia. Tutaj uwzględniłem zabezpieczenie w postaci wyszarzonych poprzednich statusów, żeby nie można było się do nich cofnąć. Ponadto obecny status jest zaznaczony domyślnie, więc nie można wybrać statusu żadnego.  
Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Poza tym, gdy wybierzemy status „Dostarczono” lub „Anulowano” już nie możemy edytować zamówienia, gdyż jest ono zakończone.

# Podsumowanie

Jak widać powyżej, system nie był wolny od różnego rodzaju błędów. Jednakże jest on na tyle prosty, że błędy dało się poprawić, a i narzędzie Lighthouse nie miało za dużo zastrzeżeń. Błędy zostały już załatane i owe poprawki zostały wprowadzone do repozytorium projektu na GitHub ([https:// https://github.com/Mordoriusz/czesciowa-implementacja](https://github.com/Mordoriusz/czesciowa-implementacja))