horizontal line

**Grupa 3**

Oleksii Nawrocki

Dominik Dziadosz

Baza danych restauracji

# Specyfikacja tematu

Tematem projektu jest system zarządzania restauracją. Przeznaczeniem tego systemu byłaby automatyzacja, zwiększenie jakości obsługi oraz umożliwienie lepszej kontroli nad działaniem w firmach gastronomicznych.

Założeniem tematu było utworzenie bazy danych która będzie przechowywała dane dotyczące dań, dodatków do dań, zamówień, pracowników w restauracji, oraz aplikacji internetowej dzięki której będzie możliwe zdalne a także wygodne zarządzanie funkcjonowaniem restauracji.

Zadaniem systemu była możliwość dodania zamówień złożonych przez klientów do kolejki aktualnie przygotowywanych z dokładnymi danymi wraz z zarządzaniem danym zamówieniem (zmiana statusu). System pozwalałby również na zapisywanie dań, dodatków do dań oraz składników, jakie restauracja aktualnie ma w swoim menu z możliwością edycji każdej informacji (nazwa, cena, dostępność etc). Dodatkowym elementem byłoby dodanie pracownika do listy zatrudnionych wraz z możliwością poprawienia jego danych a także rozwiązanie to pozwalałoby na zarządzanie stanem zatrudnienia danego pracownika.

Dzięki temu rozwiązaniu pracownik ma możliwość wygodnego kontrolowania każdego aspektu codziennej pracy w restauracji.

# 

# Aspekt projektowy bazy danych

W projekcie tworzymy bazę danych, która służyłaby dla systemu zarządzania restauracją. Przede wszystkim przechowywać ona musi dane odnośnie dań, składników, dodatków oraz zamówień. Dokładniej podaną bazę danych można podzielić na następujące obszary:

* **Dania** - w tym obszarze są tabele i relacje wszystkiego co dotyczy samych dań m. in. składniki, dodatki.
* **Zamówienia** - w tym obszarze są przechowywane dane odnośnie czasu, obsługi itd. zamówień.
* **Pracownicy** - w tym danym obszarze mielibyśmy wszystko dotyczące pracowników, ich adresy oraz stanowiska.
* **Dostawcy** - w tym obszarze łączą się dostawcy oraz składniki i dodatki.

Baza danych jest projektowana z myślą o intuicyjnym modelu użytkowania oraz technicznej niezawodności realizowanej rzeczywistości. Główną i najważniejszą ideą projektu jest stworzenie bazy danych w środowisku PostgreSQL z użyciem wszystkich dostępnych narzędzi i funkcjonalności np. schematy albo procedury.

# 

# Aspekt niezbędnych funkcjonalności

Funkcjonalność podobnej bazy danych jest największym wyzwaniem oraz zaletą systemu. Realizacja funkcjonalności będzie się odbywać w języku PG/PLSQL na wcześniej zaprojektowanej bazie. Między innymi ta funkcjonalność powinna zawierać narzędzia CRUD, czyli dla tworzenia, odczytu, aktualizacji oraz usunięcia poszczególnych rekordów.

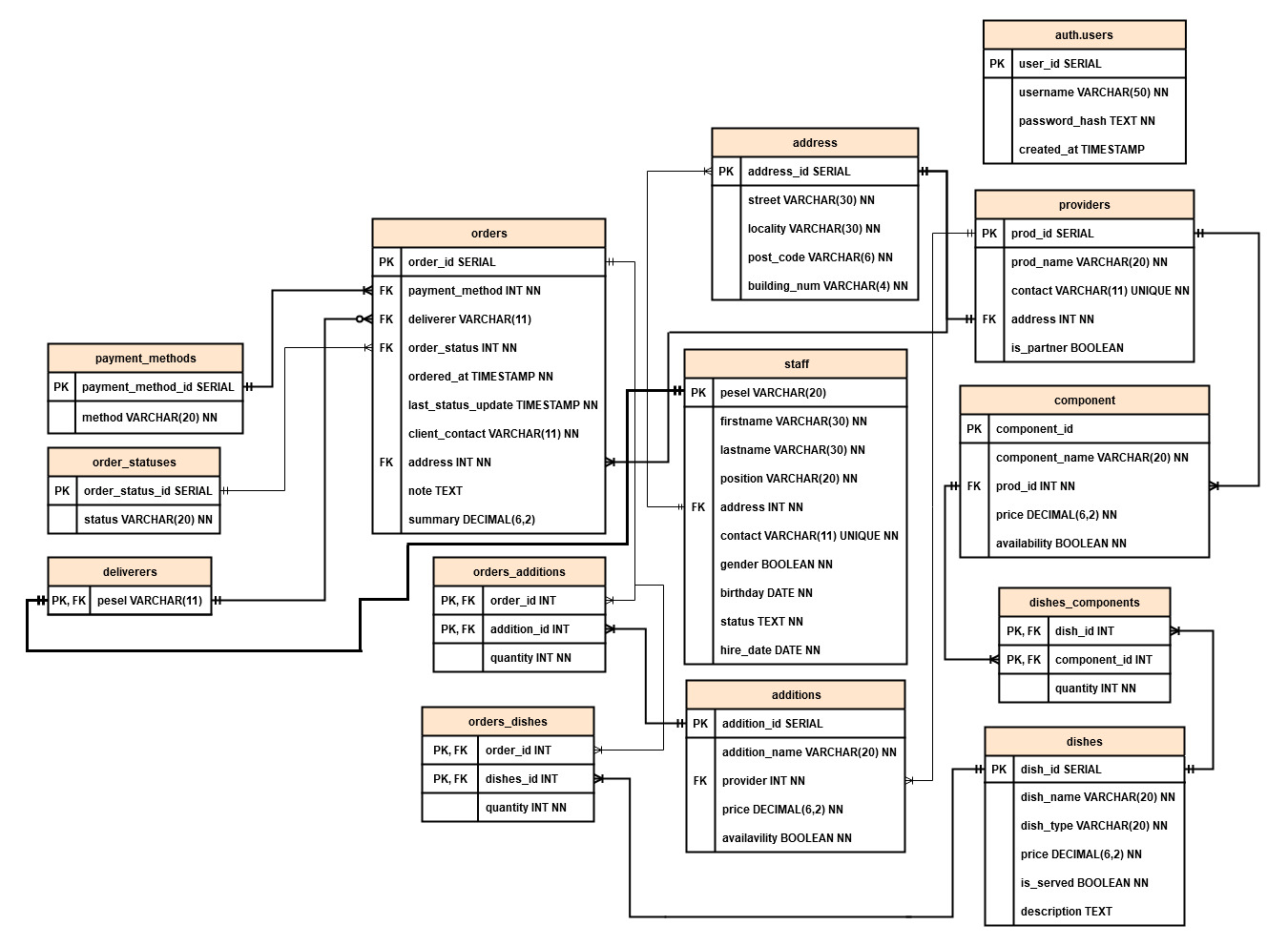
W pewnych przypadkach funkcję oraz procedury mające na celu spełnienie to założenia nabiorą wysoki poziom skomplikowania i niuansów. Z myślą o tym kolejnym niezbędnym założeniem będzie stworzenie maksymalnie uniwersalnego asortymentu mniejszych funkcji/procedur. W przypadku zniechęcenia podobną ideologią kod może się stać redundantnym i niepotrzebnie skomplikowanym.

Kolejnym ważnym aspektem realizacji funkcjonalności stanie się odpowiednie podzielenie projektu na zaprezentowane w PG/PLSQL schematy. Pozwoli to uchronić się przed potencjalnymi kolizjami oraz poprawi strukturę CRUD’u.

Z racji tego, że projekt będzie opierał się o contenery w maszynę wirtualną w systemie Docker, pozwoli to na łatwą operacje na bazie oraz zarządzania wszystkimi niezbędnymi systemami dla działania aplikacji. Konkretnie w przypadku funkcjonalności ustawione zostanie inicjalizowanie bazy danych od zera, według plików .sql oraz .csv. Dzięki temu praca nad bazą w stanie tworzenia będzie uproszczona, a przykładowe dane wraz z funkcjami itd. będą inicjalizowane od zera.

Ostatnim ważnym aspektem zostanie zapewnienie uniwersalności tworzonego systemu. Będzie on zapewniał możliwość rozbudowy oraz modyfikacji przestrzeni bez wielkich refactorów i zmian. W tym celu należy używać zrozumiałego standardu nazewnictwa oraz maksymalnie prosty i nieredundantny kod. Zapewni to możliwość dalszego rozwoju logiki biznesu.

# Powstała baza danych

**Diagram ERD powstałej bazy danych:**

Powstała baza wcieliła wszystkie założone aspekty projektowe, zarówno projektowe oraz funkcyjnie. Cała funkcjonalność od strony bazy została stworzona w języku PL/PGSQL.

### **Struktura tabel:**

* **addresses** (adresy)
* **providers** (dostawcy)
* **staff** (personel)
* **auth.users** (auth.użytkownicy)
* **components** (komponenty)
* **dishes** (dania)
* **dishes\_components** (dania\_komponenty)
* **additions** (dodatki)
* **payment\_methods** (metody\_płatności)
* **order\_statuses** (statusy\_zamówień)
* **deliverers** (dostawcy)
* **orders** (zamówienia)
* **orders\_dishes** (zamówienia\_dania)
* **orders\_additions** (zamówienia\_dodatki)

### **Opis tabel:**

1. **addresses (adresy) -** Przechowuje dane adresowe, takie jak ulica, lokalizacja, kod pocztowy oraz numer budynku.

* **address\_id** - Klucz główny, unikalny identyfikator adresu.
* **street** - Ulica (maksymalnie 30 znaków).
* **locality** - Lokalizacja (maksymalnie 30 znaków).
* **post\_code** - Kod pocztowy (maksymalnie 6 znaków).
* **building\_num** - Numer budynku (maksymalnie 4 znaki).

1. **providers (dostawcy) -** Zawiera dane o dostawcach: nazwa, kontakt, adres oraz status partnera.

* **prod\_id** - Klucz główny, unikalny identyfikator dostawcy.
* **prod\_name** - Nazwa dostawcy (maksymalnie 20 znaków).
* **contact** - Numer kontaktowy (unikalny, maksymalnie 11 znaków).
* **address** - Identyfikator adresu, klucz obcy do tabeli addresses.
* **is\_partner** - Czy dostawca jest partnerem (wartość domyślna: FALSE).

1. **staff (personel) -** Przechowuje informacje o pracownikach, w tym PESEL, imię, nazwisko, stanowisko, adres, kontakt, płeć, datę urodzenia, status oraz datę zatrudnienia.

* **pesel** - Klucz główny, unikalny numer PESEL pracownika.
* **firstname** - Imię (maksymalnie 30 znaków).
* **lastname** - Nazwisko (maksymalnie 30 znaków).
* **position** - Stanowisko (maksymalnie 20 znaków).
* **address** - Identyfikator adresu, klucz obcy do tabeli addresses.
* **contact** - Numer kontaktowy (unikalny, maksymalnie 11 znaków).
* **gender** - Płeć (TRUE - mężczyzna, FALSE - kobieta).
* **birthday** - Data urodzenia.
* **status** - Status pracownika (wartości: 'HIRED', 'SUSPENDED', 'FIRED', domyślnie: 'HIRED').
* **hire\_date** - Data zatrudnienia (domyślnie: bieżąca data).

1. **auth.users (auth.użytkownicy) -** Dane użytkowników do logowania: nazwa użytkownika, zaszyfrowane hasło, data utworzenia konta.

* **user\_id** - Klucz główny, unikalny identyfikator użytkownika.
* **username** - Nazwa użytkownika (maksymalnie 50 znaków).
* **password\_hash** - Zaszyfrowane hasło.
* **created\_at** - Data utworzenia konta (domyślnie: bieżąca data).

1. **components (komponenty) -** Informacje o składnikach: nazwa, dostawca, cena oraz dostępność.

* **component\_id** - Klucz główny, unikalny identyfikator składnika.
* **component\_name** - Nazwa składnika (unikalna, maksymalnie 20 znaków).
* **prod\_id** - Identyfikator dostawcy, klucz obcy do tabeli providers.
* **price** - Cena składnika (liczba dziesiętna z dwoma miejscami po przecinku).
* **availability** - Dostępność składnika (TRUE - dostępny, FALSE - niedostępny).

1. **dishes (dania) -** Dane o daniach, w tym nazwa, typ, cena, czy jest serwowane oraz opis.

* **dish\_id** - Klucz główny, unikalny identyfikator dania.
* **dish\_name** - Nazwa dania (maksymalnie 20 znaków).
* **dish\_type** - Typ dania (maksymalnie 20 znaków).
* **price** - Cena dania (liczba dziesiętna z dwoma miejscami po przecinku).
* **is\_served** - Czy danie jest serwowane (domyślnie TRUE).
* **description** - Opis dania.

1. **dishes\_components (dania\_komponenty) -** Powiązanie między daniami a składnikami wraz z informacją o ilości składników.

* **dish\_id** - Identyfikator dania, klucz obcy do tabeli dishes.
* **component\_id** - Identyfikator składnika, klucz obcy do tabeli components.
* **quantity** - Ilość danego składnika w daniu (nieujemna).

1. **additions (dodatki) -** Przechowuje dane o dodatkach, takie jak nazwa, dostawca, cena i dostępność.

* **addition\_id** - Klucz główny, unikalny identyfikator dodatku.
* **addition\_name** - Nazwa dodatku (maksymalnie 20 znaków).
* **provider** - Identyfikator dostawcy, klucz obcy do tabeli providers.
* **price** - Cena dodatku (liczba dziesiętna z dwoma miejscami po przecinku).
* **availability** - Dostępność dodatku (TRUE - dostępny, FALSE - niedostępny).

1. **payment\_methods (metody\_płatności) -** Lista dostępnych metod płatności, np. karta kredytowa, gotówka, płatność online.

* **payment\_method\_id** - Klucz główny, unikalny identyfikator metody płatności.
* **method** - Nazwa metody płatności (maksymalnie 20 znaków).

1. **order\_statuses (statusy\_zamówień) -** Zawiera możliwe statusy zamówień, takie jak "W TRAKCIE REALIZACJI", "W DOSTAWIE", "ZREALIZOWANE", "ANULOWANE".

* **order\_status\_id** - Klucz główny, unikalny identyfikator statusu zamówienia.
* **status** - Nazwa statusu zamówienia (maksymalnie 20 znaków).

1. **deliverers (dostawcy) -** Przechowuje dane dostawców, którzy są jednocześnie pracownikami (powiązanie z tabelą staff).

* **pesel** - Klucz główny, unikalny numer PESEL pracownika (klucz obcy do tabeli staff).

1. **orders (zamówienia) -** Dane dotyczące zamówień, w tym metoda płatności, dostawca, status zamówienia, kontakt z klientem, adres, notatki i podsumowanie ceny.

* **order\_id** - Klucz główny, unikalny identyfikator zamówienia.
* **payment\_method** - Identyfikator metody płatności, klucz obcy do tabeli payment\_methods.
* **deliverer** - PESEL dostawcy, klucz obcy do tabeli deliverers.
* **order\_status** - Identyfikator statusu zamówienia, klucz obcy do tabeli order\_statuses.
* **ordered\_at** - Data i czas złożenia zamówienia (domyślnie bieżąca data).
* **last\_status\_update** - Data i czas ostatniej zmiany statusu zamówienia.
* **client\_contact** - Numer kontaktowy klienta (maksymalnie 11 znaków).
* **address** - Identyfikator adresu, klucz obcy do tabeli addresses.
* **note** - Notatka do zamówienia.
* **summary** - Podsumowanie ceny zamówienia (liczba dziesiętna z dwoma miejscami po przecinku).

1. **orders\_dishes (zamówienia\_dania) -** Powiązanie między zamówieniami a daniami wraz z ilością dania w zamówieniu.

* **dish\_id** - Identyfikator dania, klucz obcy do tabeli dishes.
* **order\_id** - Identyfikator zamówienia, klucz obcy do tabeli orders.
* **quantity** - Ilość dania w zamówieniu (nieujemna).

1. **orders\_additions (zamówienia\_dodatki) -** Powiązanie między zamówieniami a dodatkami oraz ich ilością w zamówieniu.

* **addition\_id** - Identyfikator dodatku, klucz obcy do tabeli additions.
* **order\_id** - Identyfikator zamówienia, klucz obcy do tabeli orders.
* **quantity** - Ilość dodatku w zamówieniu (nieujemna).

### **Relacje:**

Znaki odpowiadające typu relacji:

* < - jeden-do-wielu. Jeden rekord z lewej tabeli, może mieć wiele rekordów w prawej
* > - jeden-do-wielu. Jeden rekord z prawej tabeli, może mieć wiele rekordów w prawej
* = - jeden-do-jeden.

1. **providers.address = addresses.address\_id;**
2. **components.prod\_id > providers.prod\_id;**
3. **dishes.dish\_id < dishes\_components.dish\_id;**
4. **components.component\_id < dishes\_components.component\_id;**
5. **deliverers.pesel = staff.pesel;**
6. **orders.payment\_method > payment\_methods.payment\_method\_id;**
7. **orders.order\_status > order\_statuses.order\_status\_id;**
8. **orders.deliverer > deliverers.pesel;**
9. **orders.address > addresses.address\_id;**
10. **orders.order\_id < orders\_dishes.dish\_id;**
11. **orders.order\_id < orders\_additions.dish\_id;**
12. **dishes.dishes\_id < order\_dishes.dish\_id;**
13. **additions.addition\_id < order\_additions.addition\_id;**

### **Schematy:**

1. **tools** - ten schemat zawiera wszystkie niezbędne funkcję dla CRUD’u poza odczytem. Wszystkie procedury zawierają odpowiedni przedrostek na początku np. update albo create. Składają się z operacji logicznych, zza potrzeby ze złożonych typów danych oraz funkcjonalności schematu “utils”.
2. **utils** - składa się z kilku uniwersalnych i nie tylko, funkcji, które ułatwiają rozbudowę oraz skracają ilość zbędnego kodu. Używane są tylko i wyłącznie wewnątrz instancji schematu “tools” oraz “display”.
3. **auth** - wyspecjalizowany schemat, który został wdrożony do projektu w celu oddzielenia mechanizmu logowania od reszty.
4. **display** - schemat który zawiera funkcjonalność odczytu danych z bazy. Używa kursorów oraz kwerendy dla przekazania informacji do użytkownika. Używa niektórych funkcjonalności “utils”.

### **Funkcjonalność “utils”:**

1. **item\_exist -** zwraca “true” czy “false” w zależności czy podany argument jest obecny w bazie danych.

* p\_name VARCHAR - imię przedmiotu.
* p\_type VARCHAR - typ przedmiotu

1. **find\_item -** modyfikacja poprzedniej funkcji. Zwraca “id” danego przedmiotu gdy istnieje.

* p\_name VARCHAR - imię przedmiotu.
* p\_type VARCHAR - typ przedmiotu.

1. **find\_item\_alt -** alternatywna wersja poprzedniej funkcji. Zwraca “id” który nie jest typem “INT”.

* p\_name VARCHAR.

1. **assign\_deliverer\_to\_order -** dodaje do odpowiedniego zamówienia “order\_id” dostawcę.

* p\_order\_id INT - identyfikator zamówienia.

1. **new\_addres -** zwraca “id” adresu. Dodaje nowy gdy adres w argumentach nie był używany przed tym oraz zwraca instniejący.

* p\_address JSONB - jest to JSON z odpowiednio ustawionymi danymi. Przykład takiego JSON’a: [“ulica”,”miejscowosc”,”kod\_pocztowy”,”numer\_budynku”].

1. **new\_address\_alt -** wykonuję to samo co poprzednia, różnica tylko w logice działania.

* p\_address JSONB - jest to JSON z odpowiednio ustawionymi danymi. Przykład takiego JSON’a: [“ulica”,”miejscowosc”,”kod\_pocztowy”,”numer\_budynku”].

1. **order\_sum -** zlicza sumę zamówienia z wszystkich wybranych dań oraz dodatków.

* p\_order\_id INT - identyfikator zamówienia.

1. **parse\_address -** zwraca TEXT z sparsowanym adresem dla odczytu.

* p\_address\_id INT - identyfikator adresu.

1. **update\_last\_status\_update -** zwraca trigger który aktualizuję czas ostatniej zmiany statusu zamówienia.

### **Funkcjonalność “tools”:**

1. **update\_order\_status** - Aktualizuje status zamówienia na kolejny w hierarchii. Jeśli nowy status to "IN DELIVERY", automatycznie przypisuje dostawcę.
   * **p\_order\_id INT -** identyfikato zamówienia.
2. **add\_new\_dish** - Dodaje nowe danie do bazy danych, łącznie z przypisaniem składników i dodatków. Jeśli danie już istnieje, procedura przerywa wykonanie.
   * **p\_order\_id INT** - Identyfikator zamówienia.
   * **p\_dish\_name VARCHAR** - Nazwa dania.
   * **p\_dish\_type VARCHAR** - Typ dania.
   * **p\_price DECIMAL(6,2)** - Cena dania.
   * **p\_description TEXT** - Opis dania.
   * **p\_components JSONB** (opcjonalne) - Lista składników w formacie JSON (zawiera "name" i "quantity").
   * **p\_additions JSONB** (opcjonalne) - Lista dodatków w formacie JSON (zawiera "id").
3. **create\_new\_order** - Tworzy nowe zamówienie w systemie, przypisując metody płatności, adres, dania oraz dodatki. Automatycznie oblicza i zapisuje podsumowanie zamówienia.
   * **p\_payment\_method\_name VARCHAR** - Nazwa metody płatności.
   * **p\_client\_contact VARCHAR** - Kontakt klienta.
   * **p\_note TEXT** - Notatka do zamówienia.
   * **p\_address JSONB** - Adres zamówienia w formacie JSON.
   * **p\_dishes JSONB** - Lista dań w formacie JSON (zawiera "name" i "quantity").
   * **p\_additions JSONB** - Lista dodatków w formacie JSON (zawiera "name" i "quantity").
4. **add\_provider** - Dodaje nowego dostawcę do bazy danych, przypisując adres.
   * **p\_prod\_name VARCHAR** - Nazwa dostawcy.
   * **p\_contact VARCHAR** - Kontakt dostawcy.
   * **p\_address JSONB** - Adres dostawcy w formacie JSON.
5. **add\_component** - Dodaje nowy składnik powiązany z dostawcą do bazy danych.
   * **p\_component\_name VARCHAR** - Nazwa składnika.
   * **p\_provider\_name VARCHAR** - Nazwa dostawcy.
   * **p\_price DECIMAL(6,2)** - Cena składnika.
   * **p\_availability BOOLEAN** - Dostępność składnika.
6. **add\_addition** - Dodaje nowy dodatek powiązany z dostawcą do bazy danych.
   * **p\_addition\_name VARCHAR** - Nazwa dodatku.
   * **p\_provider\_name VARCHAR** - Nazwa dostawcy.
   * **p\_price DECIMAL(6,2)** - Cena dodatku.
   * **p\_availability BOOLEAN** - Dostępność dodatku.
7. **add\_staff** - Dodaje nowego pracownika, przypisując dane osobowe i adres.
   * **p\_pesel VARCHAR** - PESEL pracownika.
   * **p\_firstname VARCHAR** - Imię pracownika.
   * **p\_lastname VARCHAR** - Nazwisko pracownika.
   * **p\_position VARCHAR** - Stanowisko pracownika.
   * **p\_address JSONB** - Adres w formacie JSON.
   * **p\_contact VARCHAR** - Kontakt pracownika.
   * **p\_gender BOOLEAN** - Płeć pracownika.
   * **p\_birthday DATE** - Data urodzenia pracownika.
8. **item\_soft\_toggle** - Włącza/wyłącza dostępność dania, składnika, dodatku lub partnerstwa dostawcy.
   * **p\_name VARCHAR** - Nazwa elementu.
   * **p\_type VARCHAR** - Typ elementu (np. "DISH", "COMPONENT").
9. **update\_dish** - Aktualizuje dane dania.
   * **p\_dish\_id INT** - Identyfikator dania.
   * **p\_new\_name VARCHAR** - Nowa nazwa dania.
   * **p\_type VARCHAR** - Typ dania.
   * **p\_price NUMERIC** - Nowa cena.
   * **p\_description TEXT** - Nowy opis.
   * **p\_is\_served BOOLEAN** - Czy danie jest podawane.
10. **update\_staff** - Aktualizuje dane pracownika, w tym adres.
    * **p\_pesel VARCHAR** - PESEL pracownika.
    * **p\_new\_firstname VARCHAR** - Nowe imię.
    * **p\_new\_lastname VARCHAR** - Nowe nazwisko.
    * **p\_position VARCHAR** - Nowe stanowisko.
    * **p\_contact VARCHAR** - Nowy kontakt.
    * **p\_gender BOOLEAN** - Nowa płeć.
    * **p\_birthday DATE** - Nowa data urodzenia.
    * **p\_status TEXT** - Nowy status.
    * **p\_address JSONB** - Nowy adres w formacie JSON.
11. **update\_addition** - Aktualizuje dane dodatku.
    * **p\_addition\_id INT** - Identyfikator dodatku.
    * **p\_name VARCHAR** - Nowa nazwa.
    * **p\_price NUMERIC** - Nowa cena.
    * **p\_provider\_name VARCHAR** - Nowy dostawca.
    * **p\_status BOOLEAN** - Nowa dostępność.
12. **update\_component** - Aktualizuje dane składnika.
    * **p\_component\_id INT** - Identyfikator składnika.
    * **p\_name VARCHAR** - Nowa nazwa.
    * **p\_price NUMERIC** - Nowa cena.
    * **p\_provider\_name VARCHAR** - Nowy dostawca.
    * **p\_status BOOLEAN** - Nowa dostępność.
13. **update\_provider** - Aktualizuje dane dostawcy, w tym adres.
    * **p\_provider\_id INT** - Identyfikator dostawcy.
    * **p\_name VARCHAR** - Nowa nazwa dostawcy.
    * **p\_contact VARCHAR** - Nowy kontakt.
    * **p\_address JSONB** - Nowy adres w formacie JSON.
    * **p\_status BOOLEAN** - Nowy status partnerstwa.
14. **cancel\_order -** Anuluje zamówienie o podanym identyfikatorze.
    * **p\_order\_id INT -** identyfikator zamówienia do anulowania.

### **Funkcjonalność “display”:**

1. **list\_all\_dishes** - Zwraca listę wszystkich dań w menu.
   * Zwraca: **dish\_id**, **dish\_name**, **dish\_type**, **price**, **is\_served**, **description**.
2. **list\_all\_additions** - Zwraca listę wszystkich dodatków wraz z informacjami o dostawcy.
   * Zwraca: **addition\_id**, **addition\_name**, **price**, **provider\_name**, **availability**.
3. **list\_all\_components** - Zwraca listę wszystkich składników oraz ich dostępność i cenę.
   * Zwraca: **component\_id**, **component\_name**, **price**, **provider\_name**, **availability**.
4. **list\_order\_dishes** - Zwraca listę dań w zamówieniu o określonym "order\_id".
   * **p\_order INT** - Identyfikator zamówienia.
   * Zwraca: **order\_id**, **dish\_name**, **quantity**.
5. **list\_order\_additions** - Zwraca listę dodatków w zamówieniu o określonym "order\_id".
   * **p\_order INT** - Identyfikator zamówienia.
   * Zwraca: **order\_id**, **addition\_name**, **quantity**.
6. **list\_providers** - Zwraca listę dostawców z danymi kontaktowymi i adresem.
   * Zwraca: **prod\_id**, **prod\_name**, **contact**, **parsed\_address**, **street**, **locality**, **post\_code**, **building\_num**, **is\_partner**.
7. **list\_all\_orders** - Zwraca szczegóły wszystkich zamówień w systemie.
   * Zwraca: **ord\_id**, **pay\_meth**, **summ**, **deliv**, **ordr\_stat**, **ord\_at**, **last\_update**, **client**, **address\_string**, **cust\_note**.
8. **list\_staff** - Zwraca listę pracowników wraz z ich danymi kontaktowymi i szczegółami adresu.
   * Zwraca: **staff\_id**, **fname**, **lname**, **fposition**, **fcontact**, **street**, **locality**, **post\_code**, **building\_num**, **faddress**, **fgender**, **fbirthday**, **fhire\_date**, **fstatus**.

### **Funkcjonalność “auth”:**

### **login** - Sprawdza poprawność danych logowania użytkownika.

### **username\_input VARCHAR** - Nazwa użytkownika.

### **password\_input VARCHAR** - Hasło użytkownika.

### Zwraca: **BOOLEAN** - "true", jeśli dane logowania są poprawne, w przeciwnym razie "false".

### 

# Koncepcja dostępu zdalnego

Po utworzeniu głównej części projektu tj. Bazy danych z kluczowymi procedurami i funkcjami, potrzebujemy wygodnego dostępu do zarządzania danymi i możliwościami przygotowanego systemu.

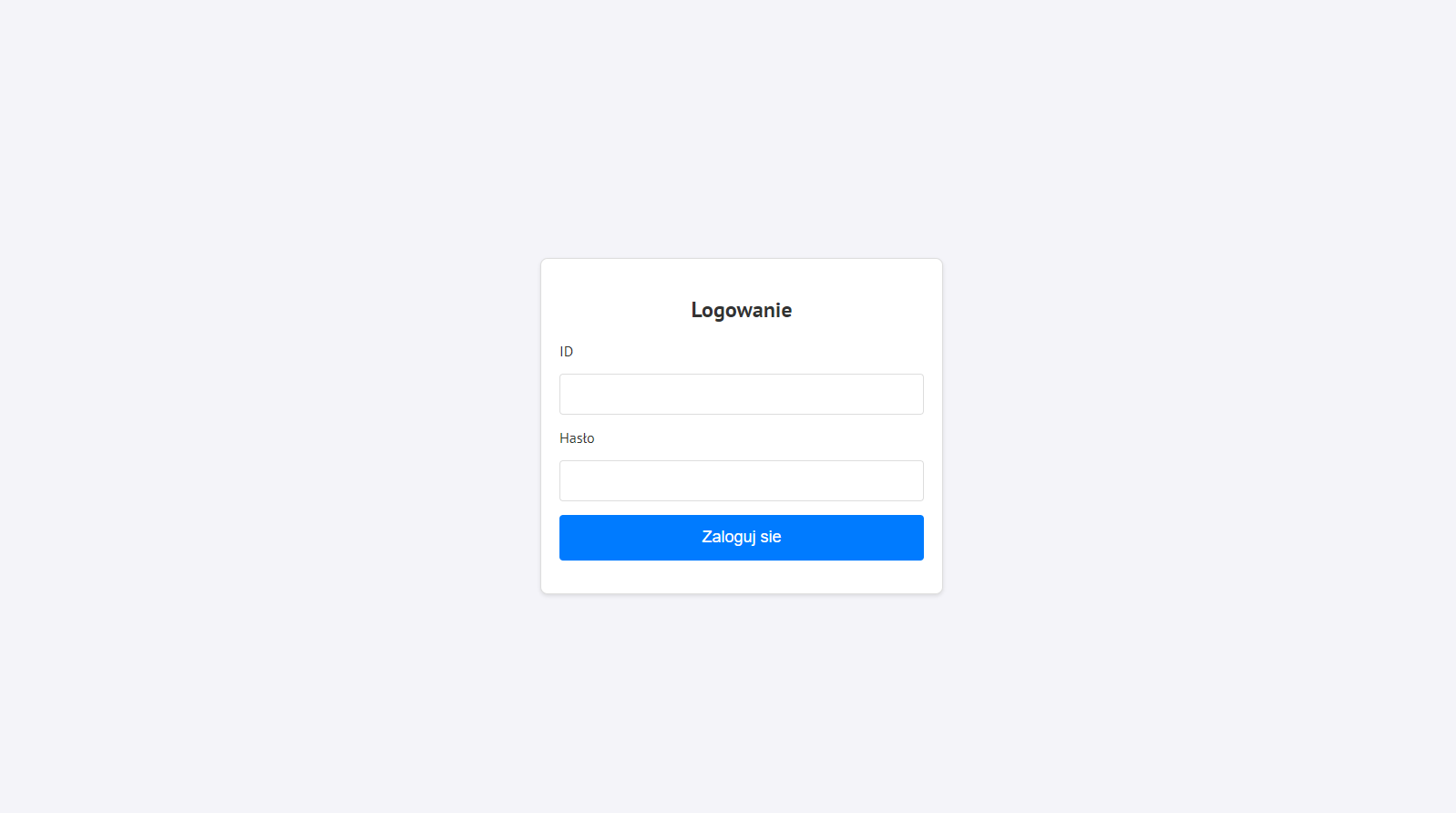
Koncepcją takiego dostępu zdalnego jest aplikacja internetowa. Rozwiązanie to pozwala na uruchomienie aplikacji na komputerze w restauracji oraz zalogowaniu się na konto przez osobę która zarządza restauracją. Pracownik taki będzie miał możliwość w formularzu na danej podstronie dodać zamówienie przyjęte od klienta i przekazanie go do realizacji kucharzom. Po przygotowaniu danego zamówienia pracownik kuchni może kliknąć na przycisk zmiany statusu zamówienia i w następnym kroku przekazać go do dostawcy który następnie dostarczy dane zamówienie.

Aplikacja ta zostałaby napisana w HTML, CSS, JavaScript oraz PHP. HTML służyłby do nadania szkieletu całej aplikacji. CSS pozwalałby na nałożenie szaty graficznej na aplikację aby była przejrzysta, intuicyjna i łatwa w obsłudze. JavaScript posłużyłby do tworzenia skryptów pozwalających na dynamiczne przetwarzanie danych lub które ułatwiłyby wykonanie niektórych operacji. PHP służyłby do połączenia części która służy do interakcji z użytkownikiem z bazą danych która dostarcza potrzebne funkcje i procedury do wykonywania operacji na danych w bazie i przesyłaniu ich do widoku.

Aplikacja byłaby zabezpieczona logowaniem się, dzięki temu możemy zapobiec nieautoryzowanemu dostępowi do danych wrażliwych a każde dane wprowadzane przez użytkownika walidowane by uniemożliwić wykonaniu błędnej lub niebezpiecznej operacji.

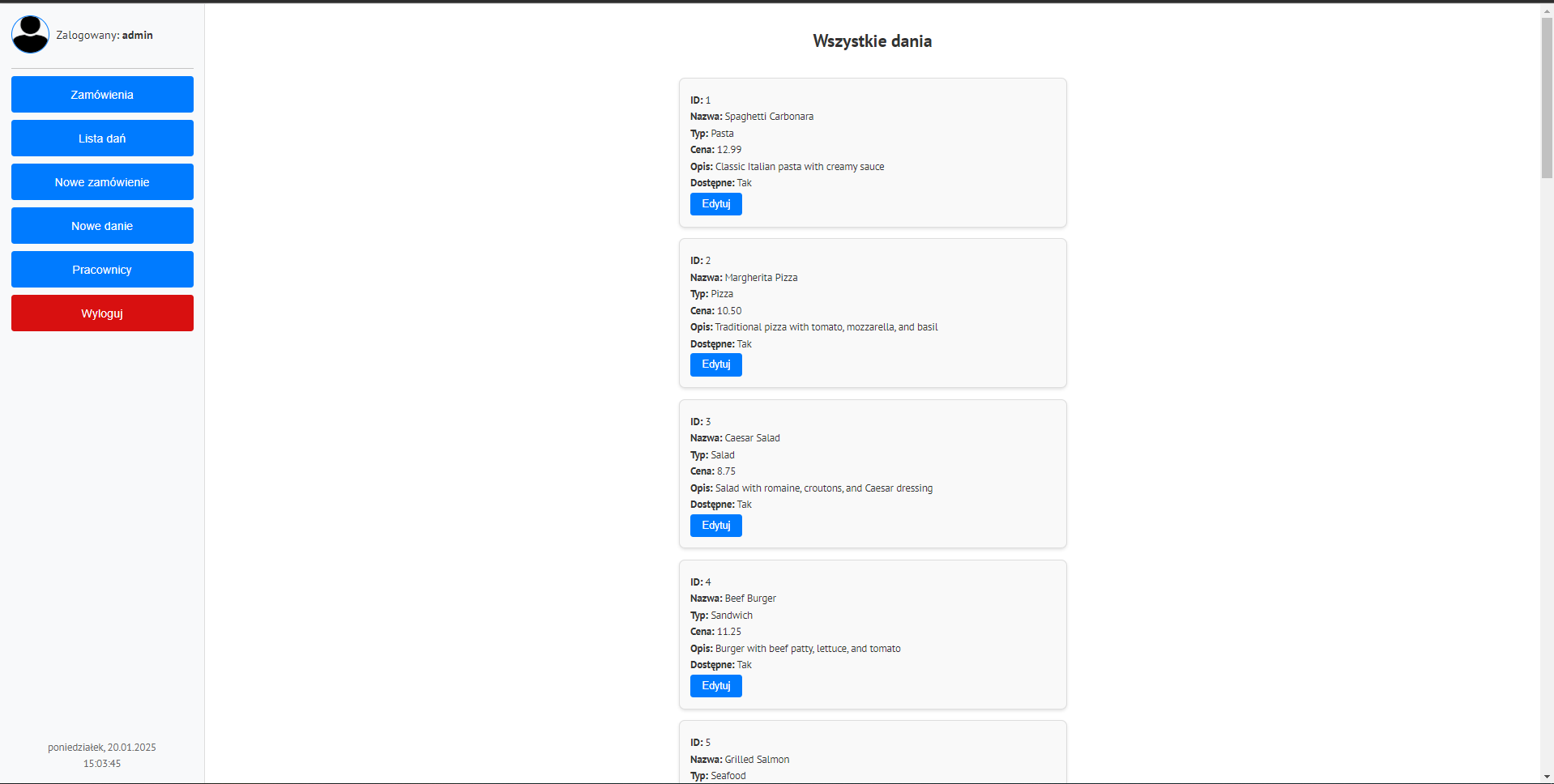
# Realizacja dostępu zdalnego

Po uruchomieniu strony wyświetla się formularz logowania do aplikacji

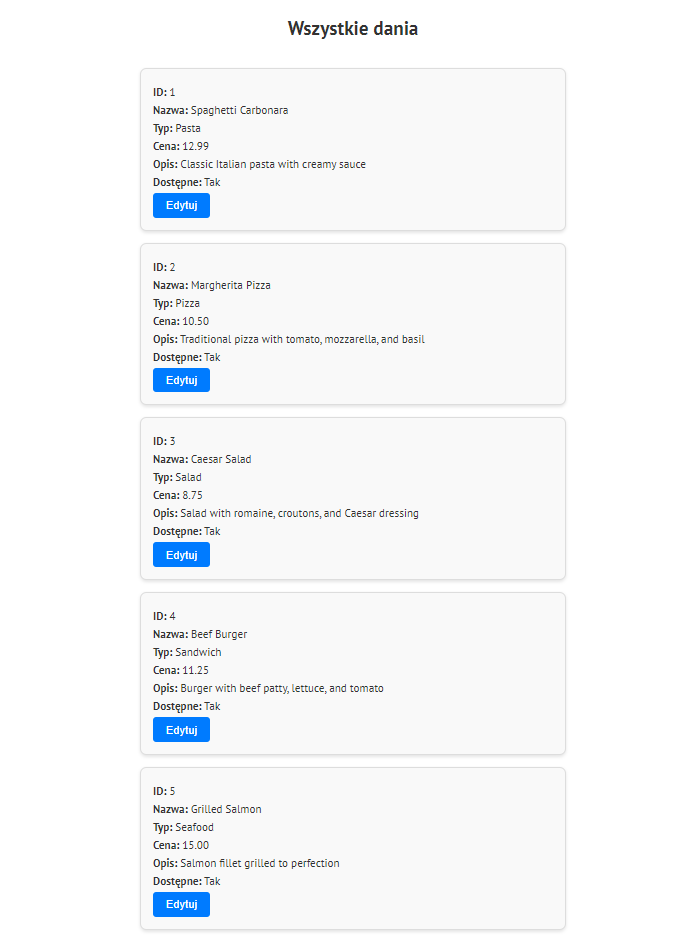


Po podaniu ID oraz hasła, wykorzystywana jest tutaj procedura *login* z pakietu *auth* (auth.login) która służy do zalogowania się do systemu.

Po zalogowaniu się domyślnym widokiem jest lista wszystkich dań.

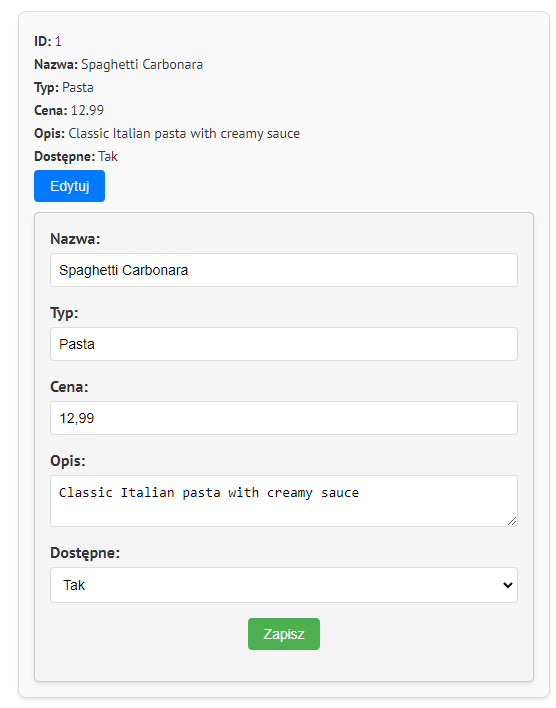


W panelu z lewej strony mamy nazwę aktualnie zalogowanego użytkownika. Poniżej mamy całe menu które umożliwia nam nawigację po całej aplikacji. Na dole lewego panelu mamy aktualną datę i godzinę

Na tej stronie mamy wypisane wszystkie dania które zostały zapisane w bazie danych

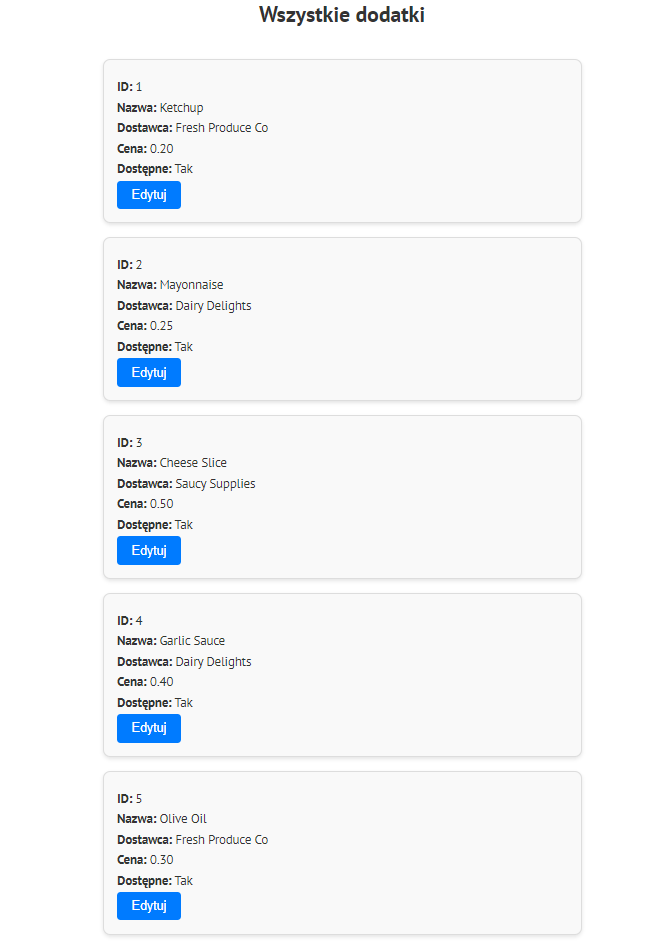
Do wypisania ich używamy funkcji *list\_all\_dishes* z pakietu *display* (display.list\_all\_dishes). Funkcja ta wypisuje ID dania, nazwę dania, typ dania, cenę, opis oraz dostępność.

Każde danie można również edytować przyciskiem z prawej strony “Edytuj”. Po kliknięciu pojawia się nam formularz który to umożliwia



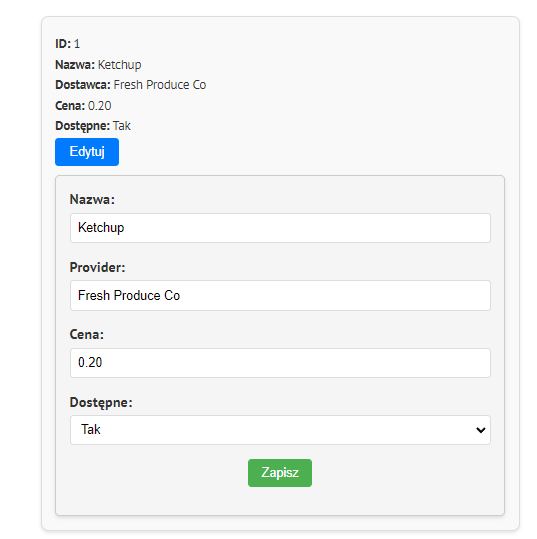
Pola w formularzu są automatycznie uzupełniane aktualnymi danymi a następnie po kliknięciu przycisku “Zapisz”, wykorzystywana jest procedura *update\_dish* z pakietu *tools* (tools.update\_dish), aktualizuje ona pola w tabeli dania o dane podane w formularzu.

Adekwatnie działa wypisywanie dodatków do dań



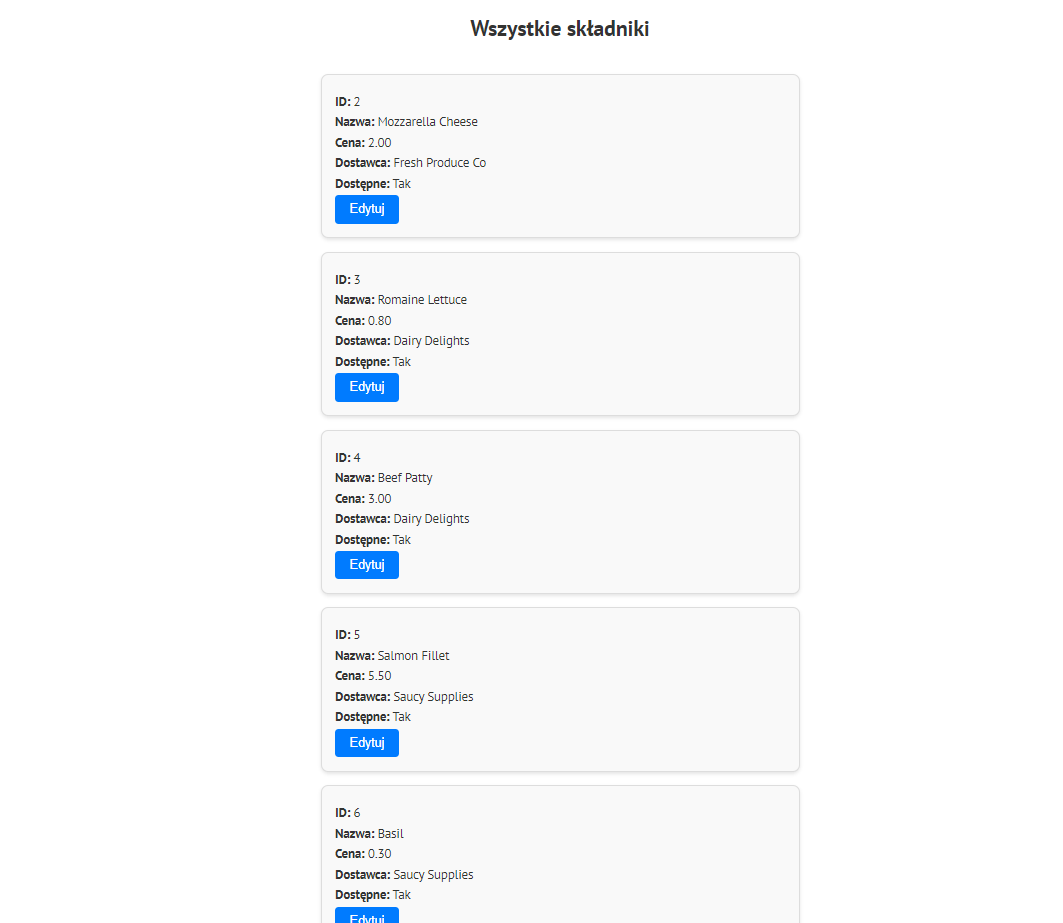
Tutaj wykorzystywana jest funkcja *list\_all\_additions* z pakietu *display* (display.list\_all\_additions).

Edycja również działa tak samo



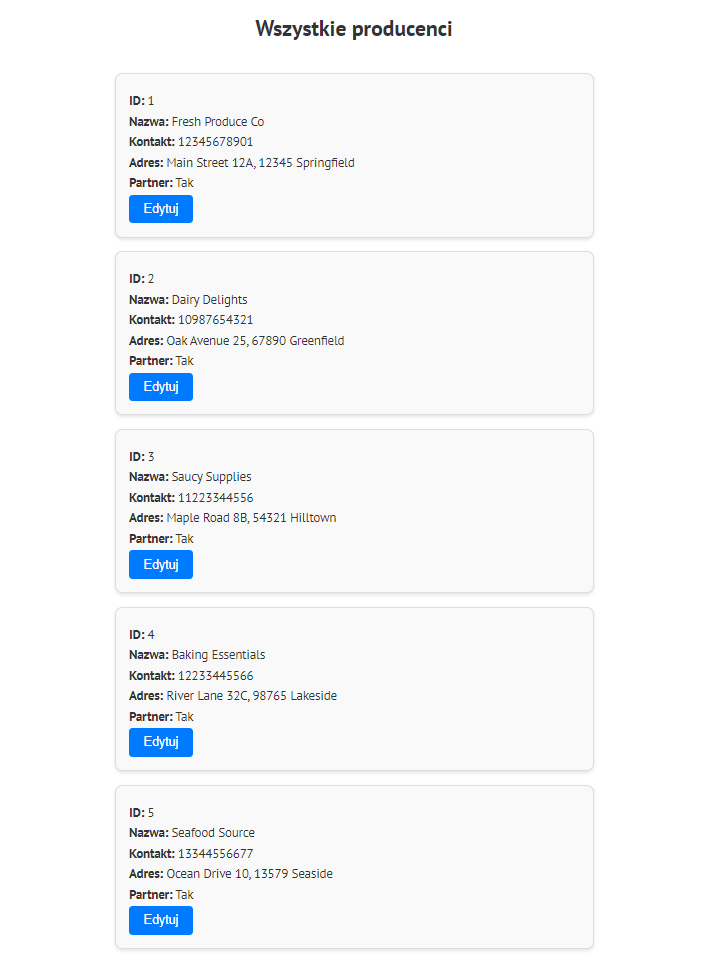
Do edycji dodatku wykorzystujemy procedurę *update\_addition* z pakietu *tools* (tools.update\_addition)

Kolejne elementy wyświetlane w tym widoku to składniki dań. Działanie jest takie samo jak do elementów wyżej



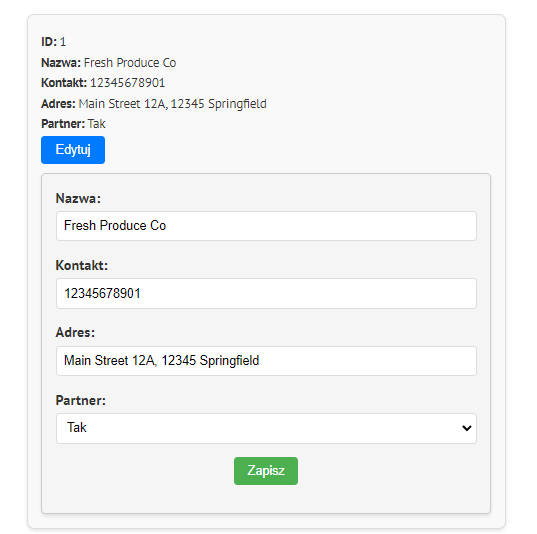
Funkcja która jest tu wykorzystana to *list\_all\_components* z pakietu *display* (display.list\_all\_components)

Ostatnim elementem który jest wyświetlany to lista dostawców. Tu również działanie jest identyczne.



Wykorzystywana funkcja to *list\_providers* z pakietu *display*

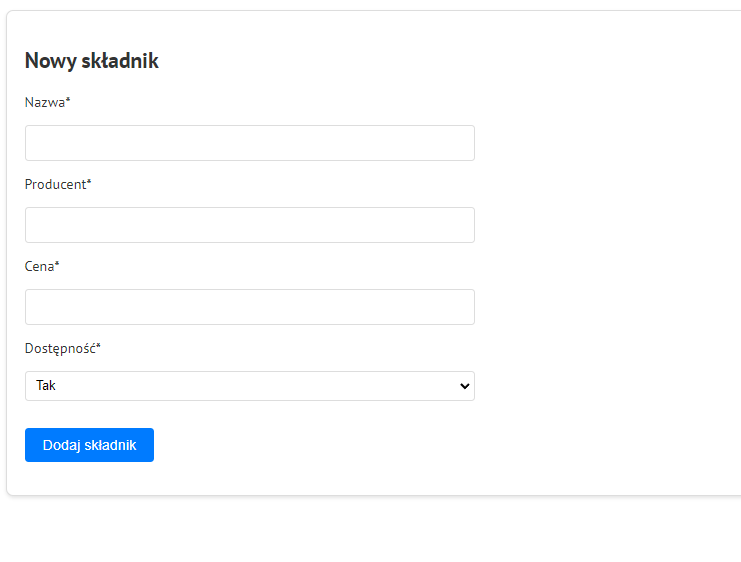
Edycja również działa tak jak w elementach wyżej.



Używana jest tu procedura *update\_provider* z pakietu *tools* (tools.update\_provider)

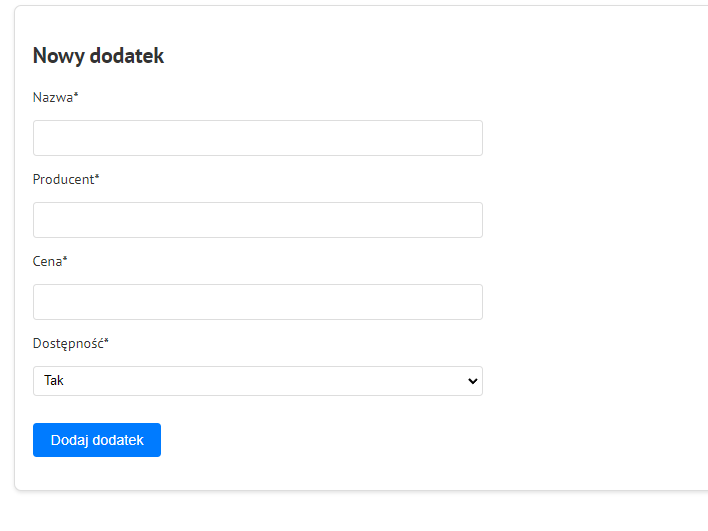
System umożliwia także dodawanie nowego składniku, dodatku i producent pod przyciskiem

Klikając przycisk “Nowy składnik” możemy dodać składnik do bazy.



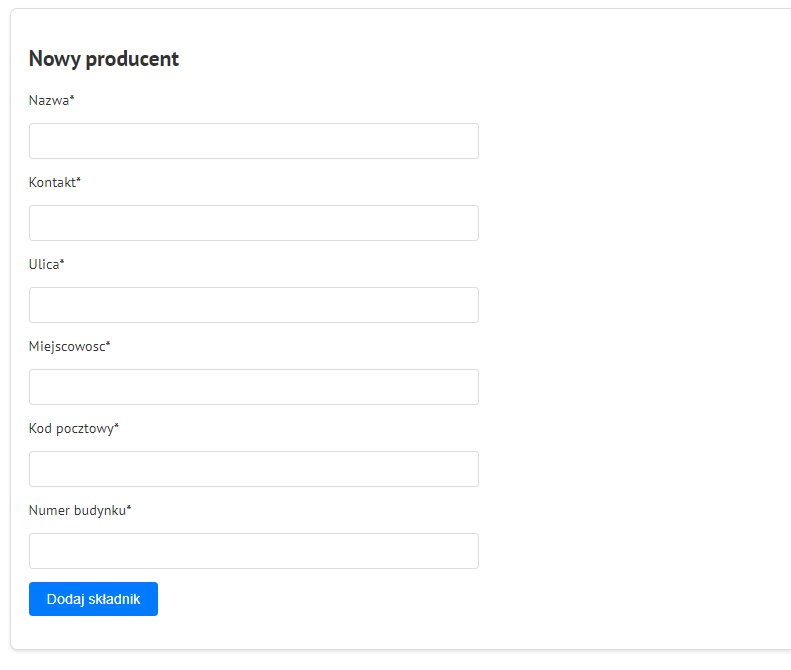
Do dodania składnika do bazy wykorzystujemy procedurę add\_component z pakietu tools.

Klikając przycisk “Nowy dodatek” możemy dodać dodatek do dania do bazy.



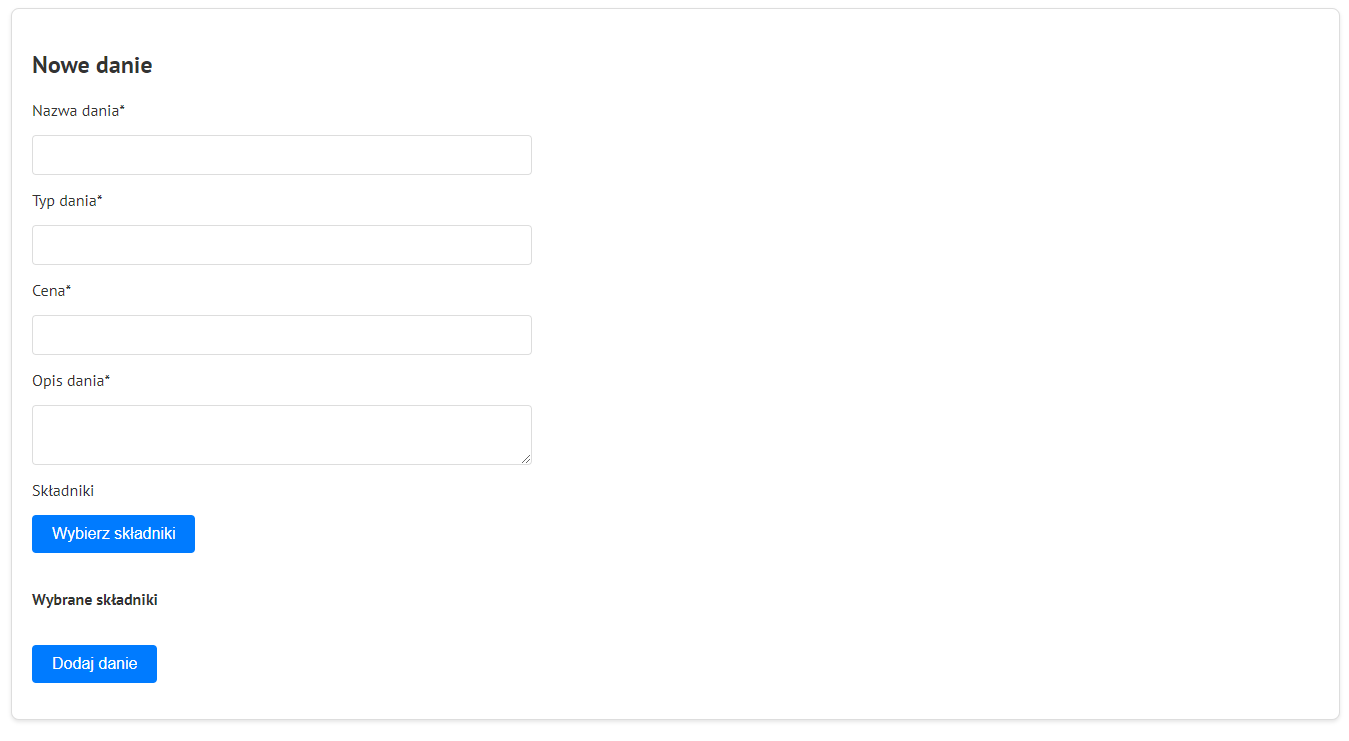
Wykorzystujemy do tego celu procedurę add\_addition z pakietu tools

Klikając przycisk “Nowy producent” możemy dodać producenta do bazy.



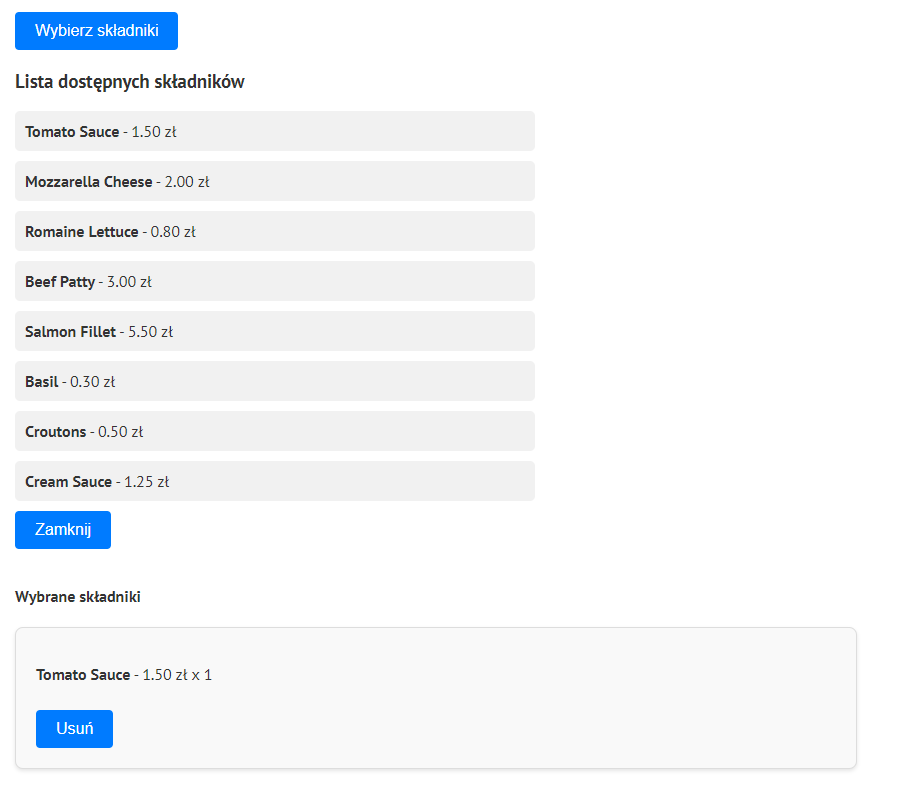
Wykorzystujemy tutaj procedurę *add\_provider* z pakietu *tools*

Kolejną funkcjonalnością systemu jest dodawanie nowego dania. Formularz dostępny jest pod przyciskiem na bocznym panelu “Nowe danie”



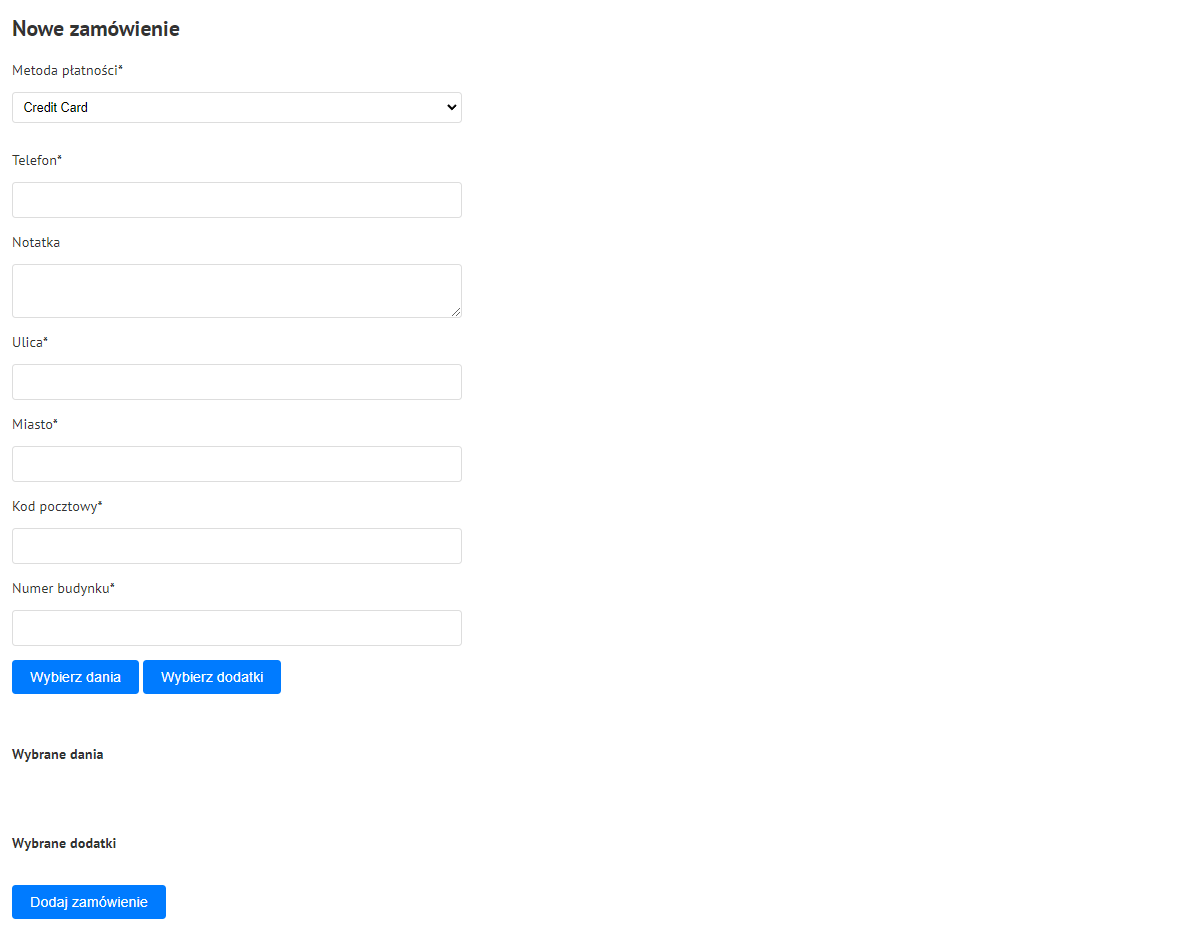
By dodać nowe danie musimy podać nazwę dania, typ dania, cenę, opis, oraz wybrać danie z listy dostępnych. Lista dostępnych dań jest generowana i wyświetlana dynamicznie z pomocą AJAX oraz funkcji do wyświetlenia listy dań. Wykorzystywana funkcja list\_all\_dishes.

Po kliknięciu danego dania, dodawane jest one do wybranych dań, kolejne kliknięcie sprawia że zwiększa się ilość dań w zamówieniu.



Po kliknięciu przycisku “Dodaj danie”, dane są przesyłane do procedury *add\_new\_dish* z pakietu *tools*.

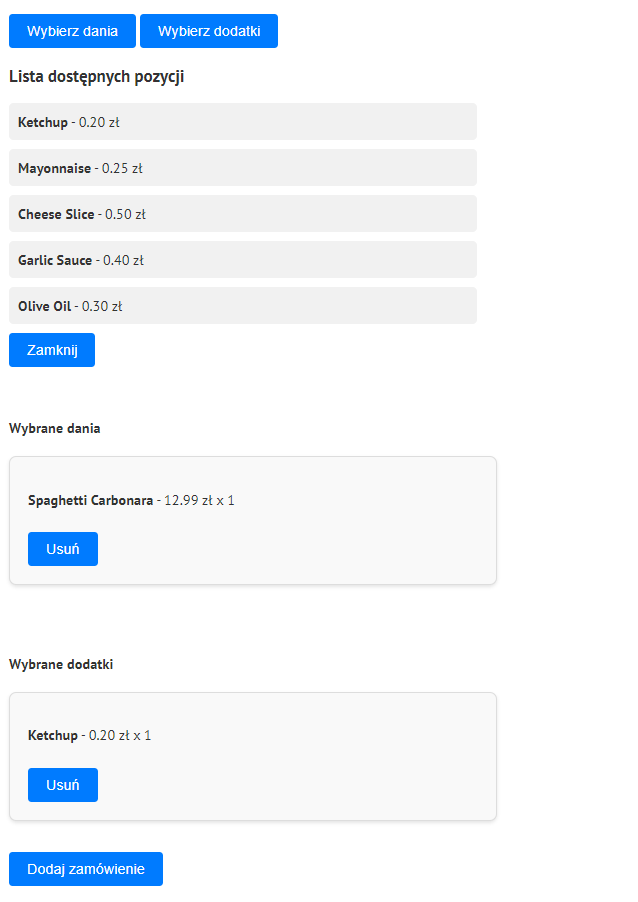
Podobnie działa funkcjonalność ze składaniem nowego zamówienia. Po kliknięciu przycisku w menu “Nowe zamówienie”, wyświetli nam się widok z formularzem do złożenia zamówienia.



Do złożenia nowego zamówienia wybieramy typ płatności, telefon klienta, opcjonalna notatka do zamówienia, adres w postaci ulica, miasto, kod pocztowy i numer budynku.

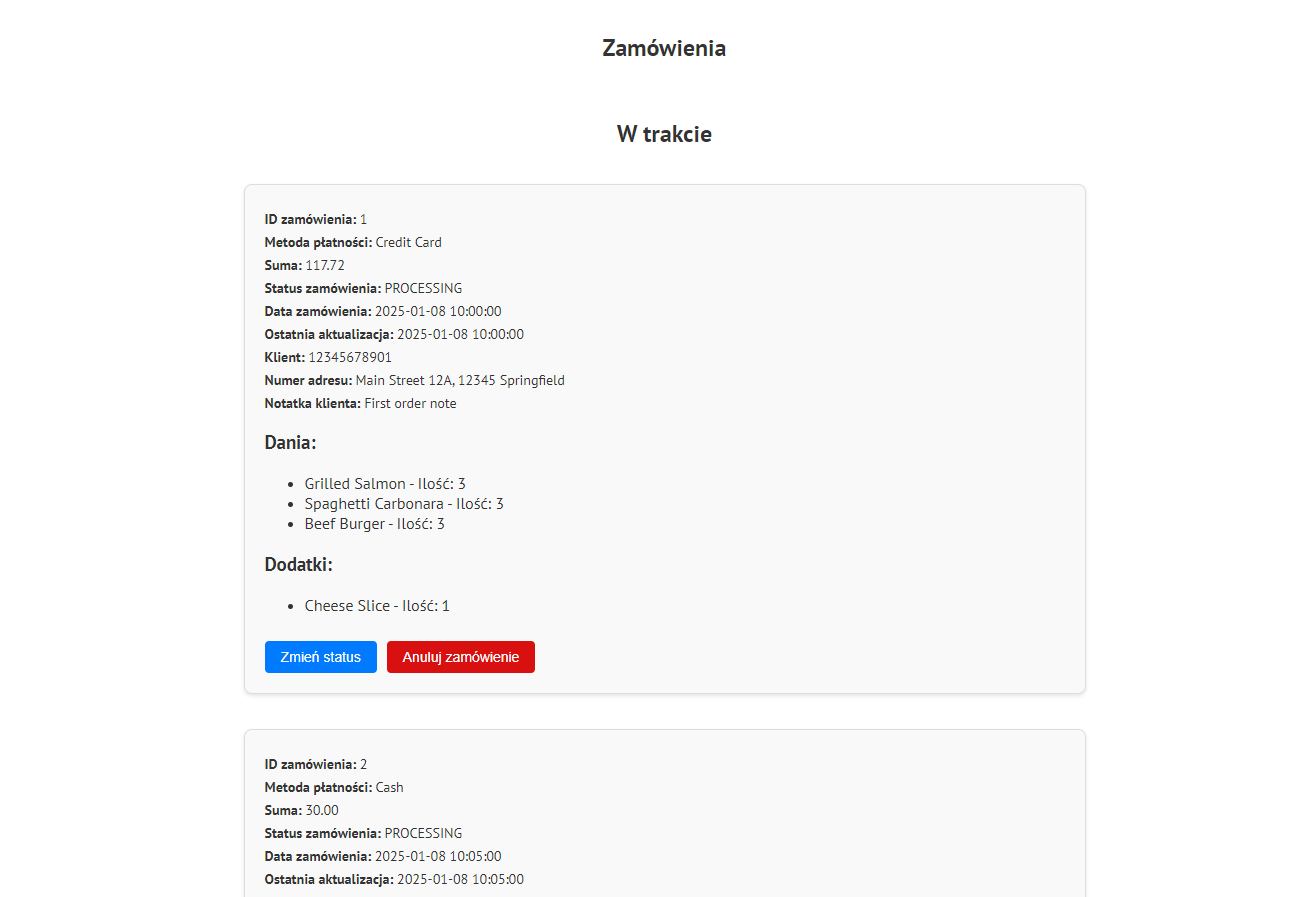
Wybór dania i dodatku opiera się na tym samym rozwiązaniu jak dodawanie nowego dania i jego komponentów.

Do wyświetlania listy dostępnych dań i dodatków służy odpowiednio funkcja *display.list\_all\_dishes* oraz *display.list\_all\_additions*.



Pola z formularza oraz wybrane dania przesyłane są do procedury *create\_new\_order* z pakietu *tools* i zamówienie jest dodawane do listy zamówień.

Następną funkcją aplikacji jest zarządzanie złożonymi zamówieniami. Po kliknięciu na bocznym panelu przycisk “Zamówienia”, wyświetlone zostaną aktualnie złożone zamówienia wraz ze statusem.



Wykorzystywana funkcja do wyświetlania zamówień to *list\_all\_orders* z pakietu *display*. Do wpisania dań w zamówieniu służy funkcja *list\_order\_dishes* z pakietu *display,* a do wypisania dodatków *list\_order\_additions* również pakietu *display*.

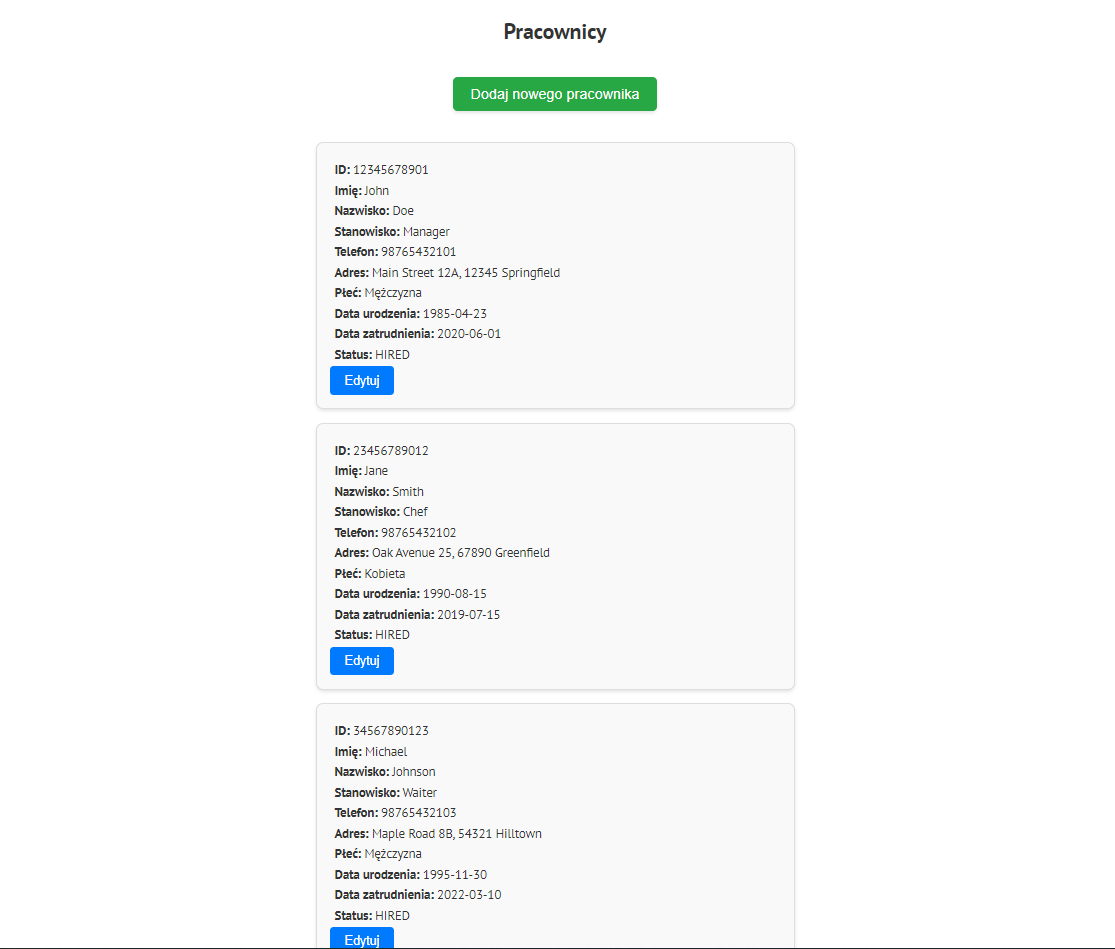
Przy każdym zamówieniu mamy dwa przyciski “Zmień status” oraz “Anuluj zamówienie”. Po kliknięciu przycisku “Zmień status”, status zmienia się automatycznie o jeden wyżej (w przygotowaniu, w doręczeniu, dostarczone). Wykorzystywana jest tutaj procedura

*update\_order\_status* z pakietu *tools*.

Po kliknięciu drugiego przycisku “Anuluj zamówienie” zamówienie zostaje anulowane, tutaj wykorzystujemy procedurę *cancel\_order* z pakietu *tools*

Wszystkie zamówienia również sortowane są po statusie zamówienia i odpowiednio przenoszone do swojej kategorii po jego zmianie.

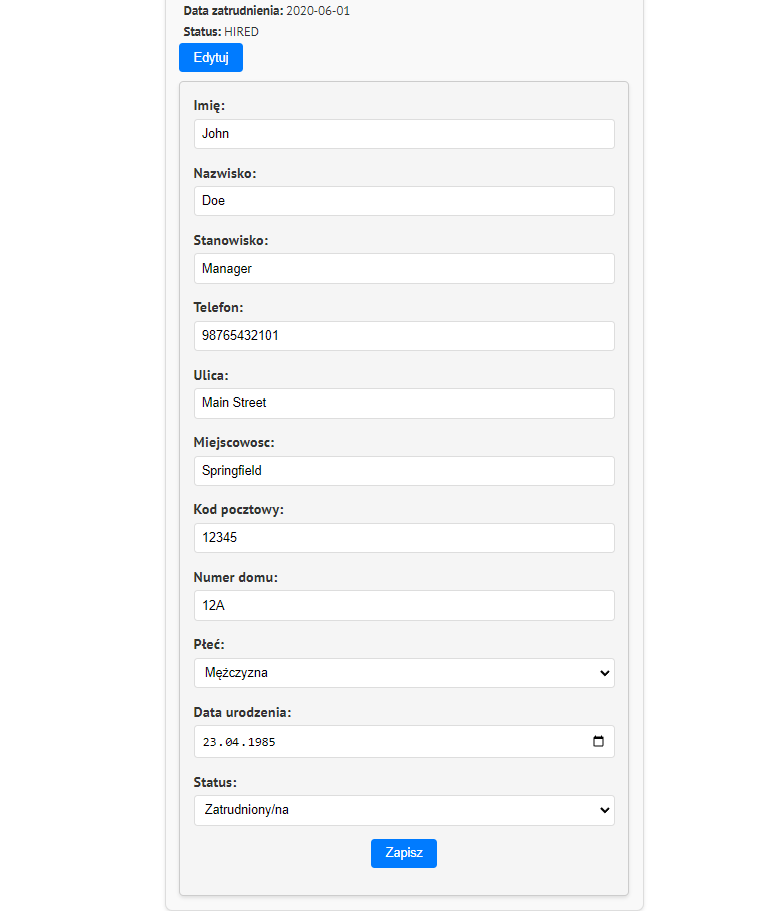
Dodatkową funkcją jest zarządzanie pracownikami. Po kliknięciu przycisku na menu “Pracownicy”, wyświetla nam się lista pracowników



Służy do tego funkcja *list\_staff* z pakietu *display* (display.list\_staff).

W tej sekcji możliwe jest również edytowanie danego pracownika oraz dodanie nowego pracownika.

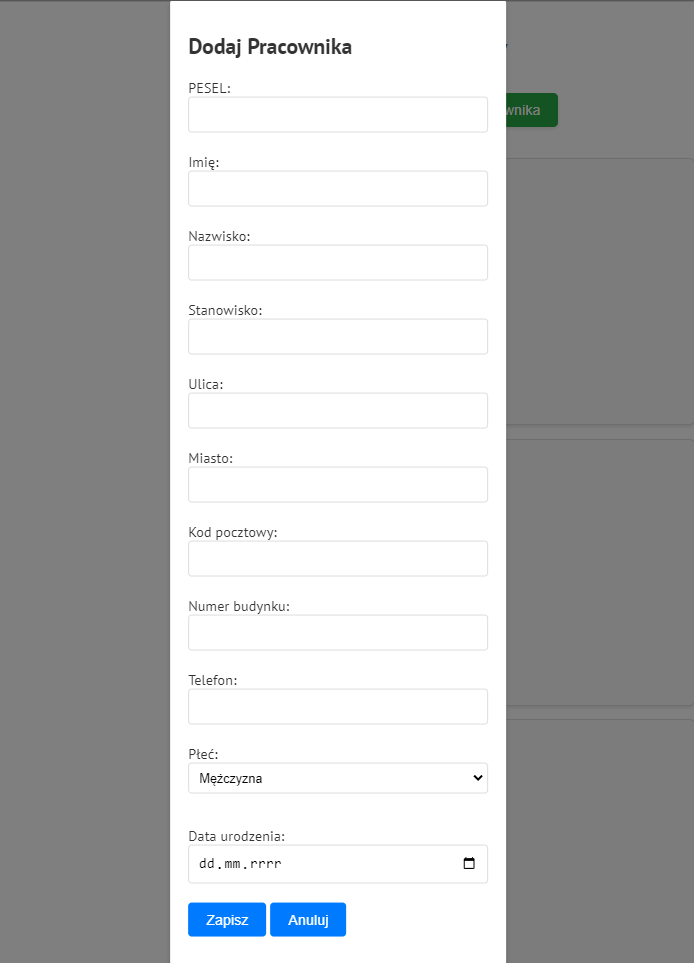
Po kliknięciu przycisku “Edytuj” pojawia nam się formularz do edycji danych



Po kliknięciu przycisku “Zapisz” przesyłamy dane i wykorzystujemy procedure *update\_staff* z pakietu *tools* do zaktualizowania danych.

Ostatni przycisk w menu “Wyloguj” służy do zamknięcia sesji i wylogowania użytkownika z aplikacji.

Dodanie nowego pracownika realizowane jest po kliknięciu przycisku “Dodaj nowego pracownika”. Pojawia się wtedy pole z formularzem do wpisania danych pracownika



Podajemy tutaj wszystkie potrzebne dane a do zapisania ich w bazie wykorzystujemy procedure *add\_staff* z pakietu *tools.*