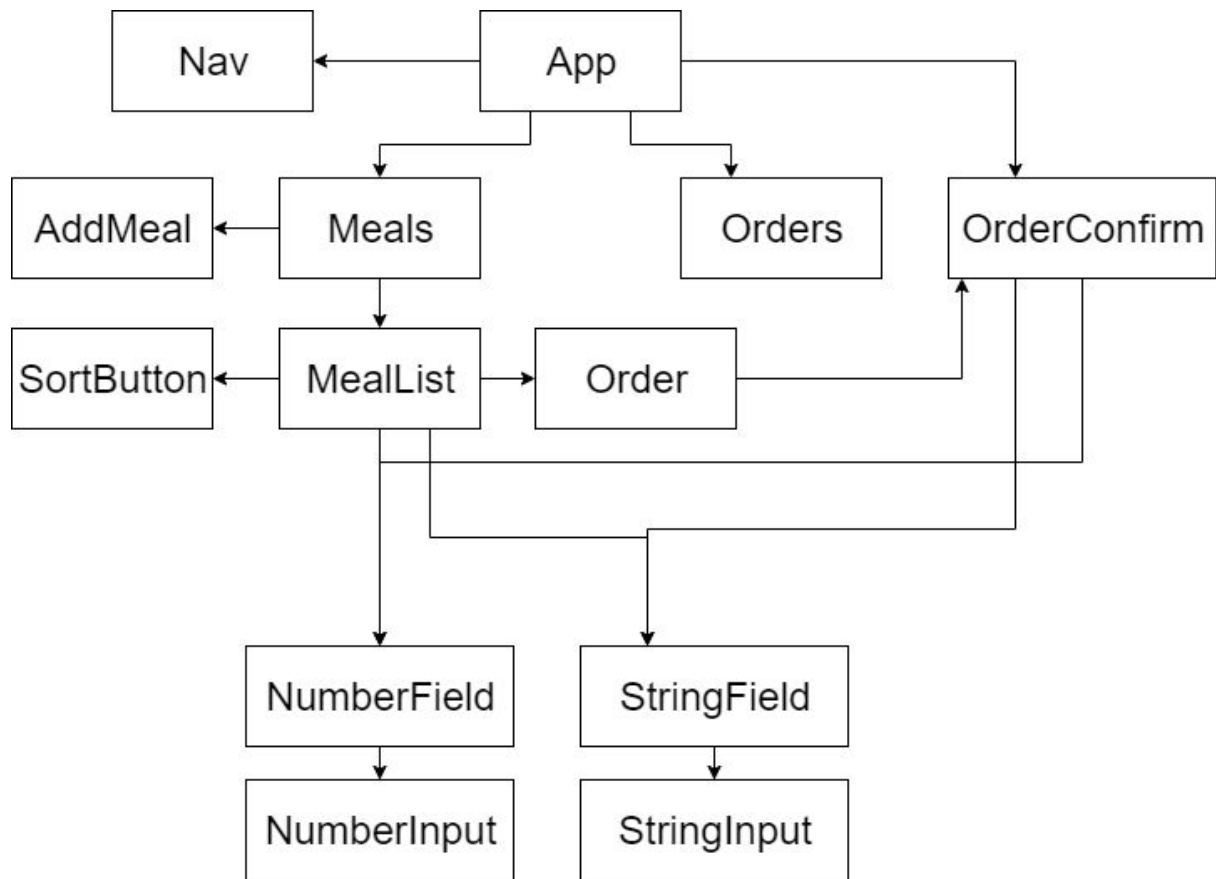


**Aplikacje Internetowe Oparte o
Komponenty
Projekt Zaliczeniowy
“Restauracja”**

1. Wizualizacja architektury komponentów



- App
 - komponent funkcyjny
 - nie otrzymuje wartości wejściowych, zarządza mechanizmem routingu
- Nav
 - komponent funkcyjny
 - nie otrzymuje wartości wejściowych, zarządza paskiem nawigacji
- AddMeal
 - komponent klasowy
 - odpowiada za dodawanie pozycji do menu
 - otrzymuje funkcję dodawania posiłku z komponentu Meals
- Meals
 - komponent klasowy
 - odpowiada za zarządzanie listą posiłków
 - jeżeli ktoś wraca z komponentu OrderConfirm, otrzymuje on listę posiłków w zamówieniu
- MealList
 - komponent klasowy
 - odpowiada za wyświetlanie listy posiłków

- otrzymuje funkcje dodawania, wybierania i usuwania, a także listę posiłków, listę posiłków do wyświetlenia oraz listę posiłków w zamówieniu (wszystko od komponentu Meals)
- Order
 - komponent funkcyjny
 - odpowiada za zamówienie
 - otrzymuje listę posiłków w zamówieniu, funkcję usuwania i dodawania posiłku do zamówienia oraz funkcję dodawania zamówienia (od komponentu MealList)
- Orders
 - komponent klasowy
 - nie otrzymuje wartości wejściowych
 - odpowiada za zarządzanie złożonymi zamówieniami
- OrderConfirm
 - komponent klasowy
 - otrzymuje listę posiłków w zamówieniu, całkowitą wartość zamówienia oraz funkcję finalizującą zamówienie (od komponentu Order)
- NumberField
 - komponent funkcyjny (reżywalny)
 - otrzymuje etykietę label oraz wszystkie parametry do utworzenia inputu
 - zarządza wyświetlaniem pola tekstowego
- NumberInput
 - komponent funkcyjny (reżywalny)
 - otrzymuje parametry do utworzenia inputu: className, placeholder, defaultValue oraz funkcję onChange
- StringField
 - komponent funkcyjny (reżywalny)
 - otrzymuje etykietę label oraz wszystkie parametry do utworzenia inputu
 - zarządza wyświetlaniem pola tekstowego
- StringInput
 - komponent funkcyjny (reżywalny)
 - otrzymuje parametry do utworzenia inputu: className, placeholder, defaultValue oraz funkcję onChange
- SortButton
 - komponent funkcyjny (reżywalny)
 - otrzymuje funkcję sortującą oraz napis, który będzie wyświetlany na przycisku
 - odpowiada za wyświetlenie przycisków sortowania

2. Szczegółowe odniesienia do elementów punktowanych

- Struktura danych:

Dla posiłków:

```
let meals = [{
  id: 1,
  name: "Pepperoni",
  price: 22.99,
  ingredients: ["sos pomidorowy", "podwójne pepperoni", "podwójny ser mozzarella"],
  image: "https://www.dominospizza.pl/upload/images/3/thumb/220-151-t-pep_800x600.jpg"
},
{
  id: 2,
  name: "Hawajska",
  price: 21.99,
  ingredients: ["sos pomidorowy", "ser mozzarella", "szynka", "ananas"],
  image: "https://www.dominospizza.pl/upload/images/3/thumb/220-151-t-hawa-800x600.jpg"
},
]
```

Dla zamówień:

```
class OrderConfirm extends Component {
  constructor(props) {
    super(props)
    this.state = {
      meals: null,
      name: null,
      surname: null,
      city: null,
      street: null,
      building_number: null,
      house_number: null,
      phone_number: null
    }
  }
}
```

- Dodane funkcjonalności:
 - dodawanie posiłku do menu
 - usuwanie posiłku z menu
 - edycja posiłku w menu
 - tworzenia zamówienia
 - usuwanie zamówienia
 - wyszukiwanie pozycji w menu
 - sortowanie pozycji w menu

- Własna walidacja:

Własna walidacja została zastosowana, żeby sprawdzić czy adres do zdjęcia został podany poprawnie.

```
function url(props, propName, componentName) {
  componentName = componentName || 'UNKNOWN';
  if (props[propName]) {
    let value = props[propName];
    if (typeof value === 'string') {
      if (value.includes('http')) {
        return null;
      }
      else return new Error(propName+' in '+componentName+' is not URL.');
```

- Weryfikacja danych przekazanych do komponentów:

Sprawdzone jest, czy lista posiłków została przekazana poprawnie.

```
MealsList.propTypes = {
  meals: PropTypes.arrayOf(PropTypes.shape({
    id: PropTypes.number,
    name: PropTypes.string,
    price: PropTypes.number,
    ingredients: PropTypes.arrayOf(PropTypes.string),
    image: url,
    selectedMeal: PropTypes.func,
    updateMeal: PropTypes.func,
    deleteMeal: PropTypes.func
  })))
}
```

- Komponenty:
 - aplikacja składa się z 12 komponentów (w tym 8 funkcyjnych)
- Dwukierunkowa komunikacja pomiędzy komponentami:
 - pomiędzy Meals oraz MealsList

- Komponenty reużywalne:
 - StringField oraz NumberField:

```
<form onSubmit={this.onOrderAdd.bind(this)}>

  <div className="form-group col-9 mx-auto">
    <StringField label="Name: " defaultValue={this.state.name} className={"form-control"} placeholder={"Name"} onChange={this.handleChan
  </div>
  <div className="form-group col-9 mx-auto">
    <StringField label="Surname: " defaultValue={this.state.surname} className={"form-control"} placeholder={"Surname"} onChange={this.h
  </div>
  <div className="form-group col-9 mx-auto">
    <StringField label="City: " defaultValue={this.state.city} className={"form-control"} placeholder={"City"} onChange={this.handleChan
  </div>
  <div className="form-group col-9 mx-auto">
    <StringField label="Street: " defaultValue={this.state.street} className={"form-control"} placeholder={"Street"} onChange={this.hand
  </div>
  <div className="form-group col-9 mx-auto">
    <StringField label="Building number: " defaultValue={this.state.building_number} className={"form-control"} placeholder={"Building n
  </div>
  <div className="form-group col-9 mx-auto">
    <NumberField label="House number: " defaultValue={this.state.house_number} className={"form-control"} placeholder={"House number"} o
  </div>
  <div className="form-group col-9 mx-auto">
    <NumberField label="Phone number: " defaultValue={this.state.phone_number} className={"form-control"} placeholder={"Phone number"} o
  </div>

  <button type="submit" href="/" className="btn btn-secondary p-1 mt-2">SUBMIT ORDER</button>
```

- SortButton:

```
<div className="m-2">
  <SortButton sortfunction={this.sort.bind(this, 0)} text={"NAME ^"}></SortButton>
  <SortButton sortfunction={this.sort.bind(this, 1)} text={"NAME v"}></SortButton>
  <SortButton sortfunction={this.sort.bind(this, 2)} text={"PRICE ^"}></SortButton>
  <SortButton sortfunction={this.sort.bind(this, 3)} text={"PRICE v"}></SortButton>
</div>
```

- Walidacja danych formularza:

```
if (this.state.name === null || this.state.name === "") alert("Please insert correct name.");
else if (this.state.surname === null || this.state.surname === "") alert("Please insert correct surname.");
else if (this.state.city === null || this.state.city === "") alert("Please insert correct city.");
else if (this.state.street === null || this.state.street === "") alert("Please insert correct street.");
else if (this.state.building_number === null || this.state.building_number === "") alert("Please insert correct building number.");
else if (this.state.house_number === null || this.state.house_number === 0) alert("Please insert correct house number.");
else if (this.state.phone_number === null || this.state.phone_number === 0) alert("Please insert correct phone number.");
else if (this.state.ingredients.length === 0) alert("Please insert correct ingredients (ingredient1,ingredient2,...).");
else if (this.state.image === null || this.state.image === "") {
  alert("Please insert correct image address.");
}
```

```
if (this.state.name === null || this.state.name === "") {
  alert("Please insert correct name.");
} else if (this.state.price === null || this.state.price === ""){
  alert("Please insert correct price.");
} else if (this.state.ingredients.length === 0) {
  alert("Please insert correct ingredients (ingredient1,ingredient2,...).");
} else if (this.state.image === null || this.state.image === "") {
  alert("Please insert correct image address.");
}
```

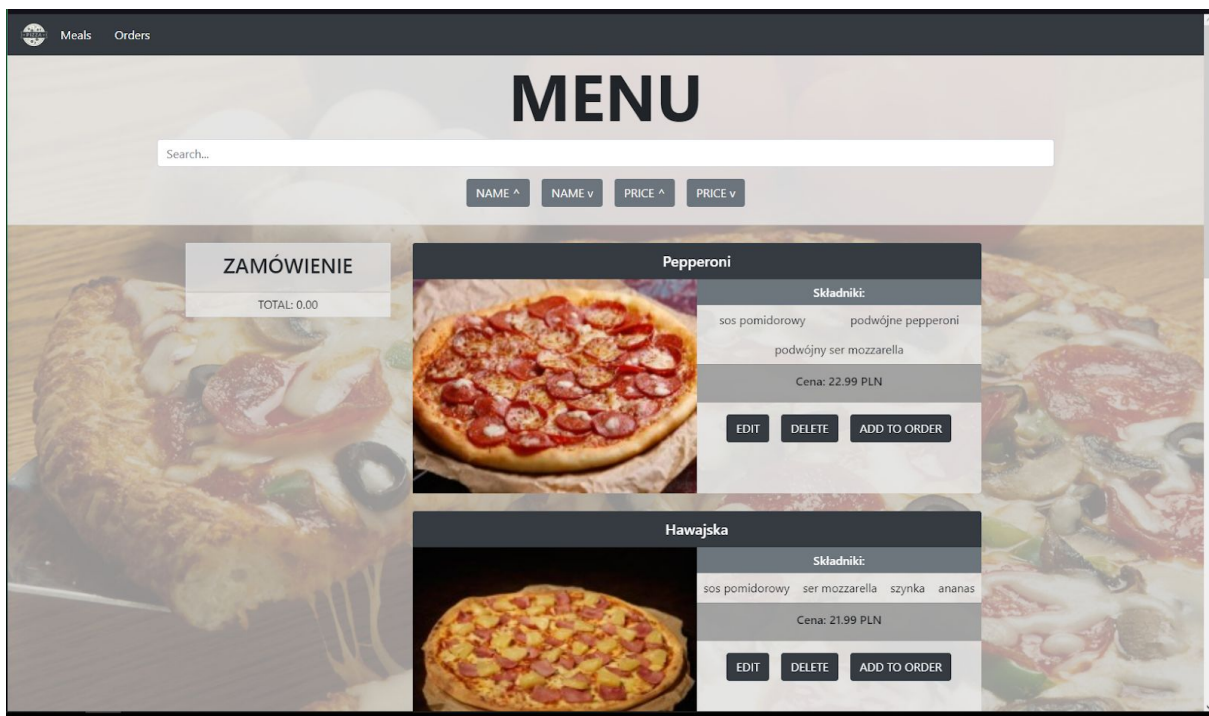
- Obsługa żądań http:
 - w liście posiłków używane są żądania get, put, post oraz delete
 - w zamówieniach używane są żądania get, post oraz delete

- Routing:

```
<Router>
  <div className="App">
    <Nav />
    <Switch>
      <Route exact path="/" component={Meals} />
      <Route path="/finishorder" component={OrderConfirm} />
      <Route path="/orders" component={Orders} />
    </Switch>
  </div>
</Router>
```

- Brak błędów w konsoli:
 - na etapie testowania nie wykryto żadnych błędów w konsoli

3. Instrukcja użytkownika



Jest to główny widok aplikacji. Znajduje się tu lista posiłków. Możliwe jest dodawanie produktów do zamówienia, edycja posiłku, usunięcie posiłku.

Po dodaniu kilku pozycji do zamówienia, jego widok prezentuje się następująco:



ZAMÓWIENIE

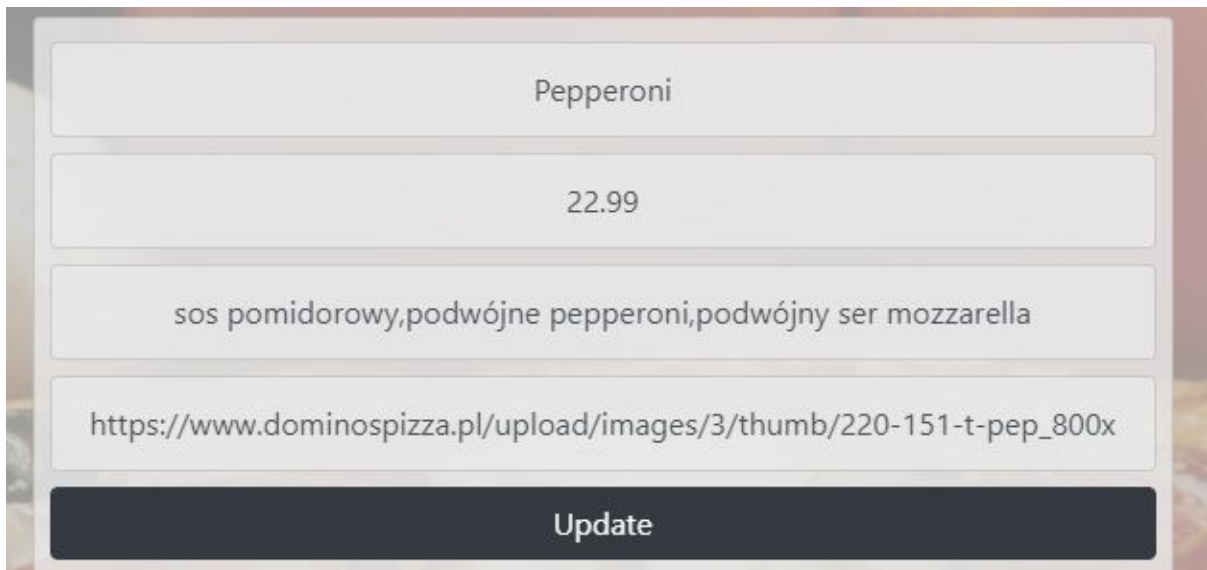
- 4x Pepperoni 91.96 + -
- 3x Hawajska 65.97 + -

TOTAL: 157.93

FINISH

Używając przycisków + oraz - możemy modyfikować liczbę produktów w zamówieniu.

Edycja posiłku na liście wygląda następująco:



Pepperoni

22.99

sos pomidorowy, podwójne pepperoni, podwójny ser mozzarella

https://www.dominospizza.pl/upload/images/3/thumb/220-151-t-pep_800x

Update

Na dole strony znajduje się przycisk Add do dodawania nowego posiłku. Po kliknięciu widoczny jest następujący formularz:

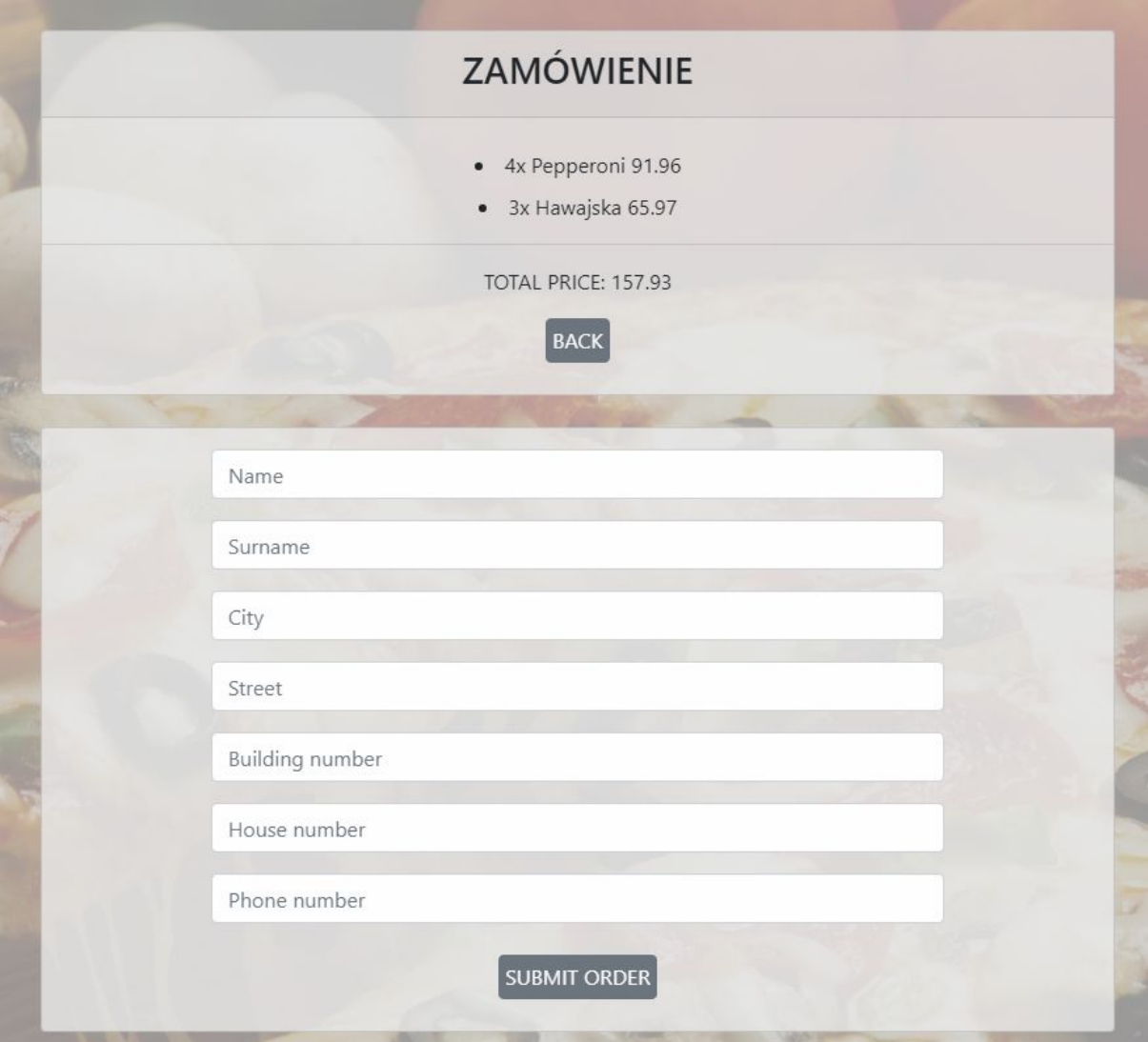
Nowa pizza
<input type="text"/>
<input type="text"/>
<input type="text"/>
<input type="text"/>
<input type="button" value="Add"/>

Dodatkowo mamy opcje wyszukiwania oraz sortowania:

<input type="text"/>
<input type="button" value="NAME ^"/> <input type="button" value="NAME v"/> <input type="button" value="PRICE ^"/> <input type="button" value="PRICE v"/>

Wpisując kolejne litery/słowa w wyszukiwarkę, lista posiłków zostanie przefiltrowana. Przyciski pod wyszukiwarką służą do sortowania listy ze względu na nazwę oraz na cenę.

Po kliknięciu przycisku Finish w zamówieniu zostajemy przeniesieni do okna finalizowania zamówienia:



The image shows a web interface for finalizing a pizza order. The background is a blurred image of pizzas. The interface is divided into two main sections. The top section, titled 'ZAMÓWIENIE', lists the order items: '4x Pepperoni 91.96' and '3x Hawajska 65.97', with a 'TOTAL PRICE: 157.93'. Below this is a 'BACK' button. The bottom section contains a form with input fields for 'Name', 'Surname', 'City', 'Street', 'Building number', 'House number', and 'Phone number'. At the bottom of this section is a 'SUBMIT ORDER' button.

ZAMÓWIENIE	
•	4x Pepperoni 91.96
•	3x Hawajska 65.97
TOTAL PRICE: 157.93	
BACK	

Name
Surname
City
Street
Building number
House number
Phone number
SUBMIT ORDER

Po wypełnieniu danych i kliknięciu Submit Order zamówienie zostaje złożone, a my zostajemy przeniesieni na stronę główną.

Dodatkowo jest możliwe zobaczenia wszystkich zamówień oraz oznaczenie ich jako zrealizowanych:

