

# Sprawozdanie

## Aplikacja CURRENCY\_CONV

### 1. Wybór języka programowania i opis działania programu.

#### Wybór języka programowania:

Do stworzenia aplikacji wybrałem język JavaScript, z wykorzystaniem biblioteki React. React jest popularną biblioteką służącą do budowy interfejsów użytkownika w aplikacjach internetowych. React pozwala na łatwe zarządzanie stanem aplikacji oraz umożliwia tworzenie dynamicznych komponentów.

#### Opis działania programu:

Program to kalkulator walutowy, który pozwala użytkownikowi na przeliczenie kwoty w PLN na wybraną walutę obcą. Program jest aplikacją internetową, która działa w trybie okienkowym (GUI), a nie w trybie konsolowym. Użytkownik podaje kwotę w PLN, wybiera walutę docelową, a program oblicza równowartość tej kwoty w wybranej walucie. Wynik jest wyświetlany na ekranie.

Dane wejściowe:

Kwota w PLN (wartość liczby, która jest wprowadzana przez użytkownika w polu tekstowym).

Waluta, na którą użytkownik chce przeliczyć (wybór z listy rozwijanej, np. USD, EUR, GBP).

Dane wyjściowe:

Wynik konwersji, czyli kwota w PLN przeliczona na wybraną walutę, wyświetlana na ekranie użytkownika w formacie: "Kwota PLN jest równa X w wybranej walucie".

#### Tryb interfejsu użytkownika:

Program działa w trybie okienkowym (GUI). Użytkownik wchodzi w interakcję z aplikacją za pomocą formularzy, przycisków i rozwijanych list. Wszelkie obliczenia są wykonywane po kliknięciu przycisku "Convert", a wynik jest natychmiast wyświetlany w interfejsie.

## **Użyte elementy obiektowości, algorytmy i biblioteki:**

### **Elementy obiektowości:**

Program wykorzystuje React (biblioteka JavaScript), która jest oparta na komponentach. Każdy komponent (np. formularz wejściowy, przycisk, wynik) może być traktowany jako obiekt, który przechowuje stan i reaguje na zdarzenia użytkownika.

Algorytmy: Program opiera się na prostym algorytmie konwersji walut. Po wprowadzeniu kwoty i wybraniu waluty, program sprawdza odpowiedni kurs wymiany z obiektu ExchangeRates i przelicza kwotę w PLN na wybraną walutę.

### **Biblioteki:**

Wykorzystano React do budowy aplikacji oraz useState z Reacta do zarządzania stanem aplikacji (kwoty wejściowej, waluty, wyniku).

Obsługa błędów: Program posiada prostą obsługę błędów. Jeżeli użytkownik nie wprowadzi liczby lub wpisze nieprawidłową wartość, wyświetli się komunikat o błędzie. Przykładem jest walidacja, która sprawdza, czy wprowadzone dane są liczbą oraz czy użytkownik wprowadził wartość przed kliknięciem przycisku. Użyto w tym celu instrukcji warunkowych i funkcji isNaN().

W przypadku błędów użytkownika, takich jak wprowadzenie tekstu zamiast liczby, wyświetlany jest komunikat z prośbą o poprawne wprowadzenie danych.

## 2. Kod źródłowy programu oraz zrzuty ekranu

Kod źródłowy:

App.js

```
1 // Import React, { useState } from 'react';
2 // Import './App.css';
3
4 // Exchange rates - you can modify or fetch them dynamically from an API
5 const ExchangeRates = {
6   USD: 1.0,
7   EUR: 0.92,
8   GBP: 0.78,
9   JPY: 109.00,
10  CAD: 1.35,
11 };
12
13 function App() {
14   const [amount, setAmount] = useState(''); // Amount in PLN
15   const [currency, setCurrency] = useState('USD'); // Default currency
16   const [result, setResult] = useState(null); // Conversion result
17
18   // Function to handle currency conversion
19   const handleConvert = () => {
20     if (!amount || isNaN(amount)) {
21       alert('Please enter a valid amount');
22       return;
23     }
24     const rate = ExchangeRates[currency];
25     const convertedAmount = (amount / rate).toFixed(2);
26     setResult(`${amount} PLN is equal to ${convertedAmount} ${currency}`);
27   };
28
29   return (
30     <div className="app-container">
31       <div>
32         <div>Currency Converter</div>
33         <div>
34           <div>
35             Amount in PLN:
36             <input
37               type="number"
38               value={amount}
39               onChange={(e) => setAmount(e.target.value)}
40               placeholder="Enter amount in PLN"
41             />
42           </div>
43           <div>
44             Select currency:
45             <div>
46               <div>
47                 value={currency}
48                 onChange={(e) => setCurrency(e.target.value)}
49               </div>
50               <div>
51                 {Object.keys(ExchangeRates).map((cur) => (
52                   <div key={cur} value={cur}>
53                     {cur}
54                   </div>
55                 ))}
56               </div>
57             </div>
58           </div>
59           <div>
60             <button onClick={handleConvert}>Convert</button>
61           </div>
62           <div>
63             <div>
64               <div>Result</div>
65               <div>Result: {result}</div>
66             </div>
67           </div>
68         </div>
69       </div>
70     </div>
71   );
72 }
73
74 export default App;
```

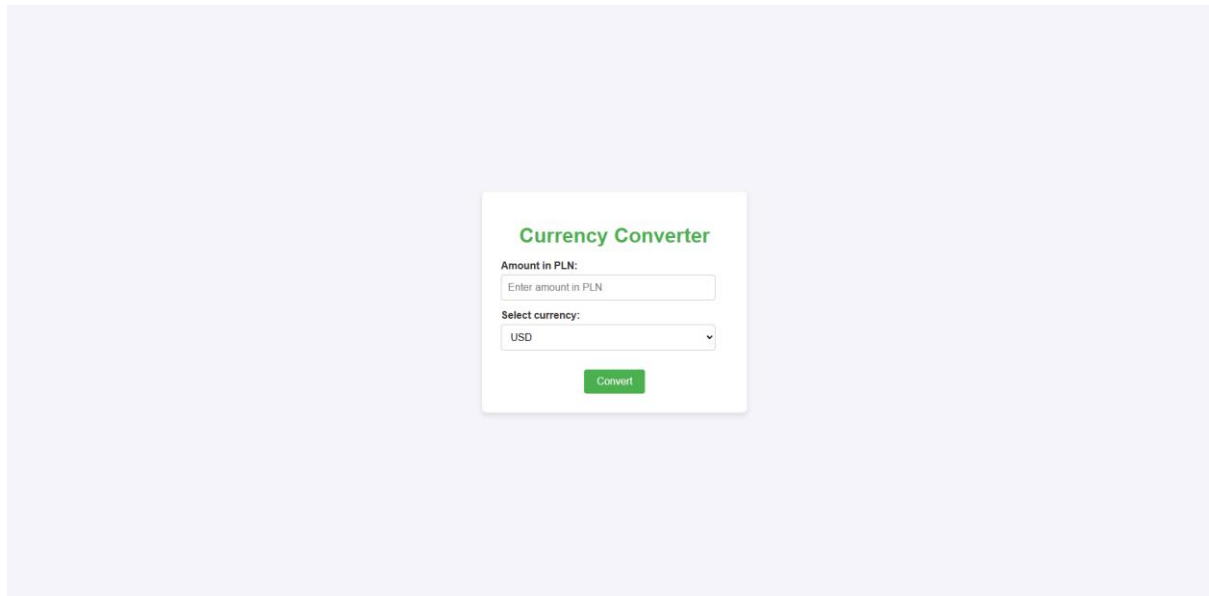
App.css

```
1 body {
2   font-family: Arial, sans-serif;
3   padding: 20px;
4   background-color: #f0f0f0;
5 }
6
7 .app-container {
8   background-color: white;
9   padding: 20px;
10  border-radius: 10px;
11  box-shadow: 0 10px 20px #ccc;
12 }
13
14 .currency-selector {
15   margin-bottom: 10px;
16 }
17
18 .currency-selector select {
19   padding: 5px 10px;
20   border: 1px solid #ccc;
21   border-radius: 5px;
22 }
23
24 .amount-input {
25   margin-bottom: 10px;
26 }
27
28 .amount-input input {
29   padding: 5px 10px;
30   border: 1px solid #ccc;
31   border-radius: 5px;
32 }
33
34 .convert-button {
35   padding: 10px 20px;
36   background-color: #007bff;
37   color: white;
38   border: none;
39   border-radius: 5px;
40   cursor: pointer;
41 }
42
43 .result-container {
44   margin-top: 10px;
45 }
46
47 .result-container div {
48   padding: 10px;
49   background-color: #f8d7da;
50   border: 1px solid #f5c6cb;
51   border-radius: 5px;
52 }
53
54 .result-container div p {
55   margin: 0;
56 }
57
58 .result-container div p span {
59   font-weight: bold;
60 }
```

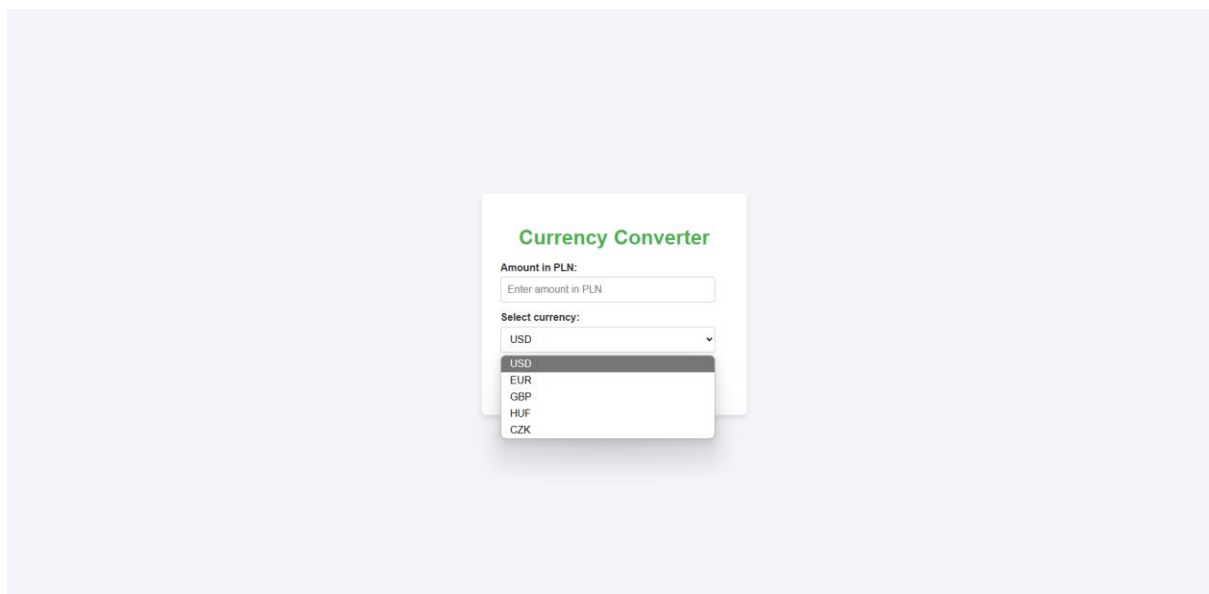
## Zrzuty ekranu:

### Widok aplikacji:

Pokazuje formularz z polem do wprowadzenia kwoty, rozwijaną listą walut i przyciskiem konwersji.

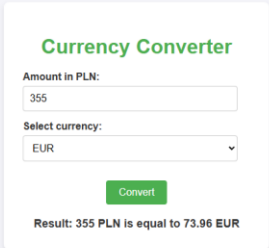


### Wybór walut:



Wynik po konwersji:

Po kliknięciu przycisku "Convert" pojawia się wynik przeliczenia.



### 3. Ocena programu

Ocena działania programu: Program działa zgodnie z założeniami. Użytkownik może wprowadzić kwotę w PLN, wybrać walutę i uzyskać wynik przeliczenia. Obsługuje błędy takie jak niepoprawny format danych wejściowych (np. litery zamiast liczb), wyświetlając komunikat ostrzegawczy.

Obsługa błędów:

#### Błędy miękkie (soft errors):

Aplikacja obsługuje błędy związane z wprowadzeniem danych, np. jeżeli użytkownik nie poda liczby w polu kwoty lub poda wartość nie będącą liczbą (tekst), aplikacja wyświetli komunikat o błędzie, prosząc o poprawne wprowadzenie danych. Komunikaty błędów są prostymi alertami JavaScript.

#### Błędy twarde (hard errors):

W przypadku błędów, które wystąpiłyby w wyniku problemów w kodzie (np. zmiana w strukturze ExchangeRates), aplikacja nie będzie w stanie kontynuować działania.

Takie błędy powinny zostać złapane i obsłużone przez odpowiednie mechanizmy, np. użycie try-catch, ale obecnie aplikacja nie ma zaawansowanej obsługi błędów.

Przykład obsługi błędu:

Wprowadzenie tekstu zamiast liczby w polu kwoty skutkuje wyświetleniem komunikatu: "Please enter a valid amount!".

Program również poprawnie obsługuje wybór nieistniejącej waluty w przypadku zmiany kursów wymiany.

Zrzuty ekranu:

Wszystkie komunikaty o błędach i sukcesie są wyświetlane w oknie alertu lub w elemencie HTML.

### **Podsumowanie:**

Aplikacja jest prosta, ale funkcjonalna. Wykorzystuje podstawy React, takie jak komponenty, zarządzanie stanem oraz prostą logikę konwersji walut. Obsługuje błędy w sposób podstawowy i jest gotowa do dalszego rozwinięcia, np. o dynamiczne pobieranie kursów wymiany z API.