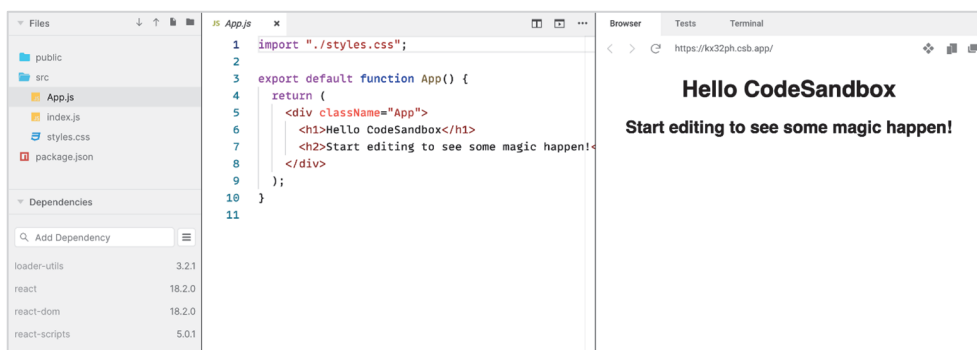


Kolorowe wersje rysunków w książce „React i TypeScript. Reaktywne tworzenie stron internetowych dla początkujących. Wydanie II”

Rozdział 1. Wprowadzenie do Reacta



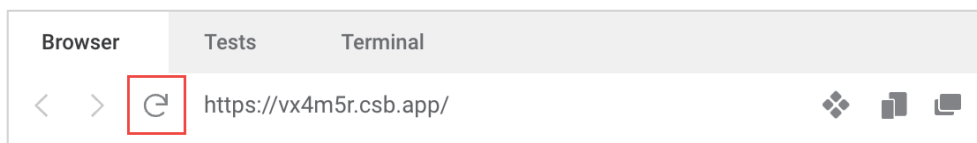
Rysunek 1.1. Projekt Reacta w CodeSandboxie

iSukces



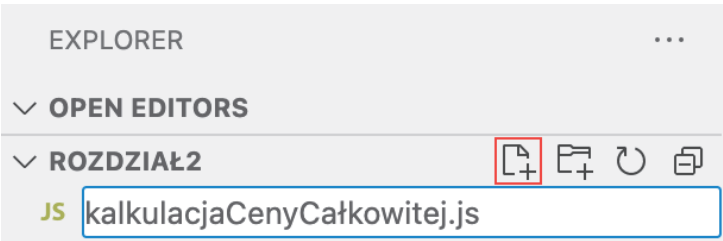
Wszystko jest naprawdę dobrze!

Rysunek 1.5. Przycisk close w komponencie Alert

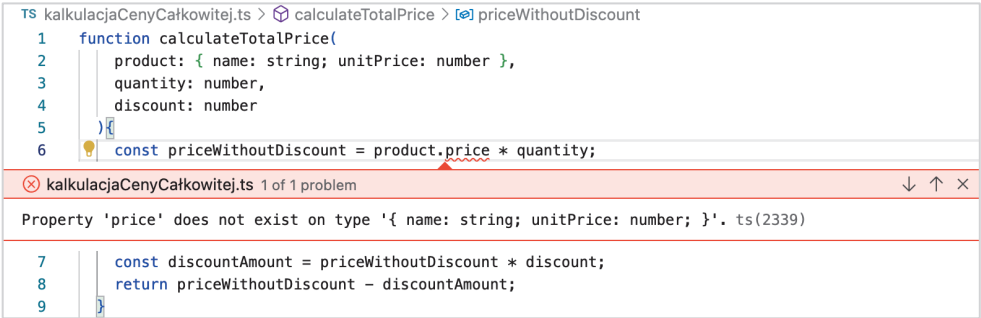


Rysunek 1.7. Opcja odświeżania w panelu Browser

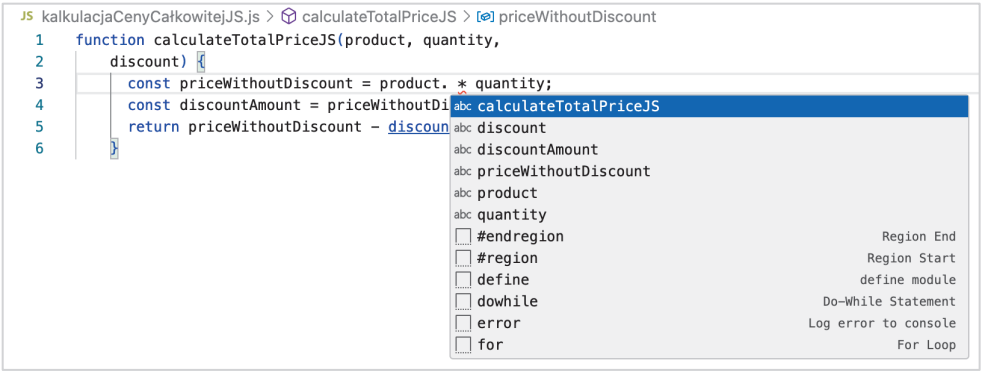
Rozdział 2. Wprowadzenie do TypeScriptu



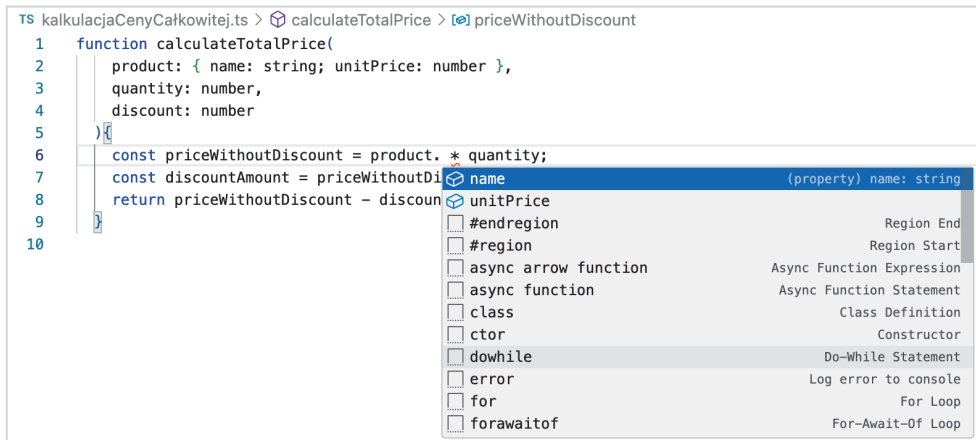
Rysunek 2.1. Tworzenie nowego pliku w Visual Studio Code



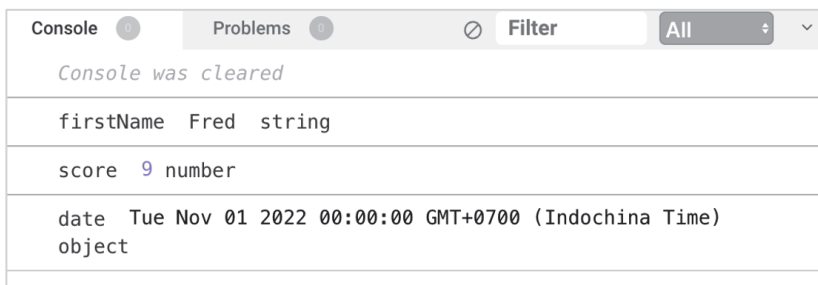
Rysunek 2.2. Podświetlony błąd typowania



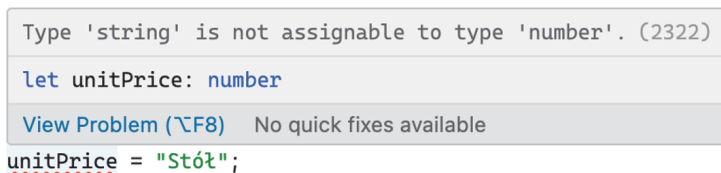
Rysunek 2.3. IntelliSense w pliku JavaScriptu



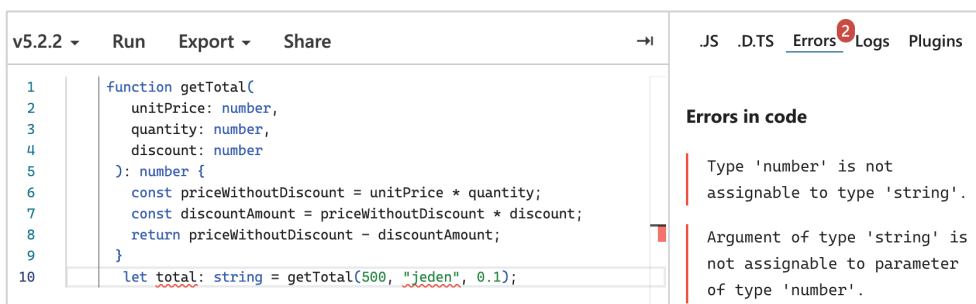
Rysunek 2.4. IntelliSense w pliku TypeScriptu



Rysunek 2.5. Przykłady typów w JavaScriptcie



Rysunek 2.7. Wykrycie błędu typowania



Rysunek 2.8. Dwa wykryte błędy typów

```
let flag: boolean
```

```
let flag = false;
```

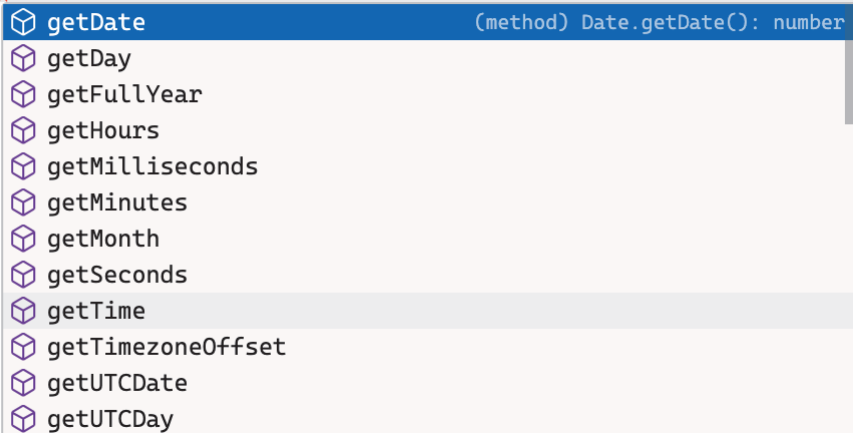
Rysunek 2.9. Najechnie kursorem na zmienną pokazuje jej typ

```
let today: Date
```

```
let today = new Date();
```

Rysunek 2.10. Dymek potwierdzający, że today ma typ Date

today.



Rysunek 2.11. IntelliSense świetnie radzi sobie z datami



Rysunek 2.12. Błąd typu przy próbie użycia nieistniejącej funkcji z datą

```
let flag: any
```

```
let flag;
```

Rysunek 2.13. Zmienna otrzymała typ any



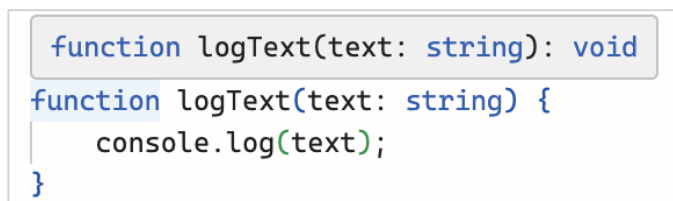
Rysunek 2.14. Atrybut firstName jest niezdefiniowany



Rysunek 2.15. Błąd typowania dla parametru data oznaczonego jako unknown



Rysunek 2.16. Rozszerzenie typu zmiennej data



Rysunek 2.17. Typ void, który zwróci funkcja



Rysunek 2.18. Problem związany z użyciem typu never

```
function foreverTask(taskName: string): void
function foreverTask(taskName: string) {
while (true) {
    console.log(`Doing ${taskName} over and over again
...`);
} }
}
```

Rysunek 2.19. Samodzielnie wydedukowany typ jako void

v5.2.2
Run
Export
Share
→

JS
.D.TS
Errors¹
Logs
Plugins

```

1  const numbers: Array<number> = [];
2  numbers.push(1);
3  numbers.push("dwa");

```

Errors in code

Argument of type 'string' is not assignable to parameter of type 'number'.

Rysunek 2.20. Dodanie tekstu do tablicy liczb wywołuje błąd typowania

```
const numbers: number[]
const numbers = [1, 2, 3];
```

Rysunek 2.21. Wnioskowanie typu array w TypeScriptie

v5.2.2
Run
Export
Share
→

JS
.D.TS
Errors²
Logs
Plugins

```

1  class Product {
2      name: string;
3      unitPrice: number;
4  }
5

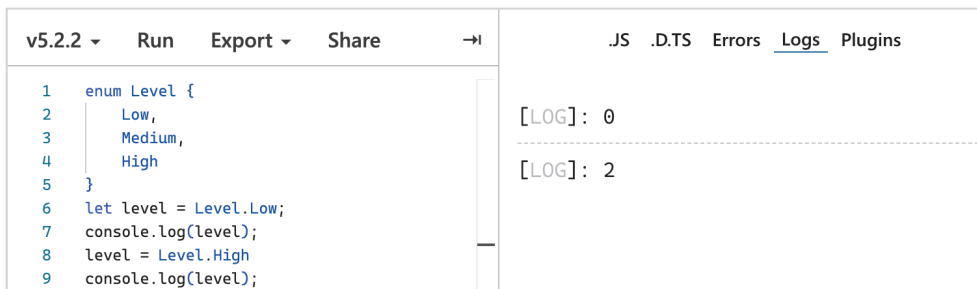
```

Errors in code

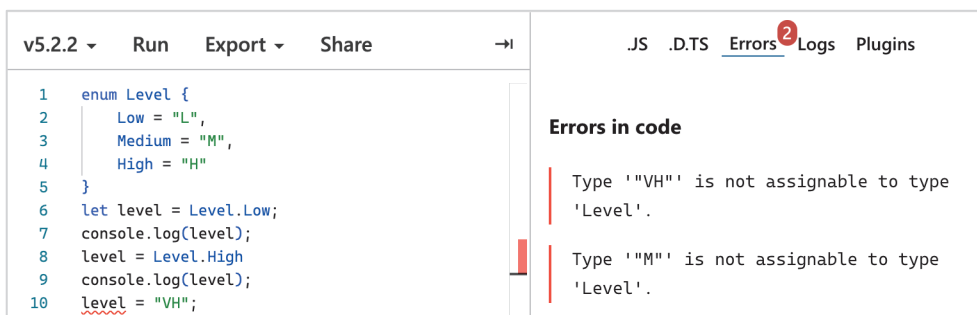
Property 'name' has no initializer and is not definitely assigned in the constructor.

Property 'unitPrice' has no initializer and is not definitely assigned in the constructor.

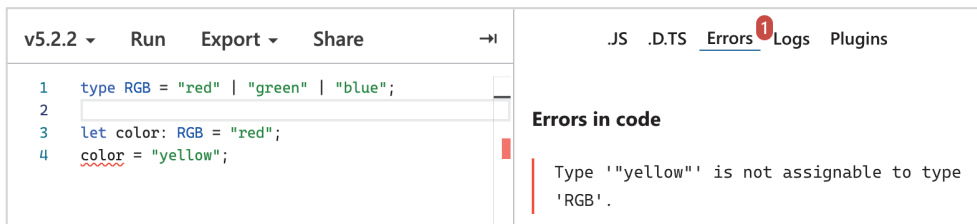
Rysunek 2.22. Problemy z typami w atrybutach klas



Rysunek. 2.23. Wartości typu wyliczeniowego



Rysunek 2.24. Dowód na to, że enumeracje oparte na ciągach znaków są bezpieczne pod względem typu



Rysunek 2.25. Błąd wynikający z niezgodności typu w unii

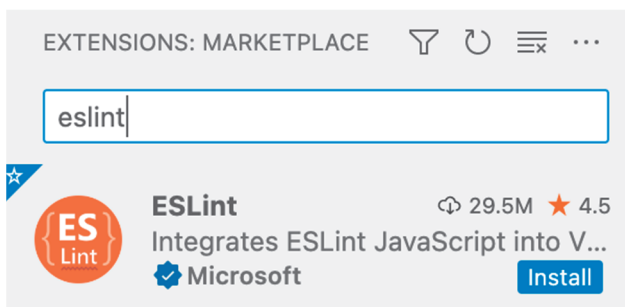
Rozdział 3. Konfiguracja Reacta i TypeScriptu



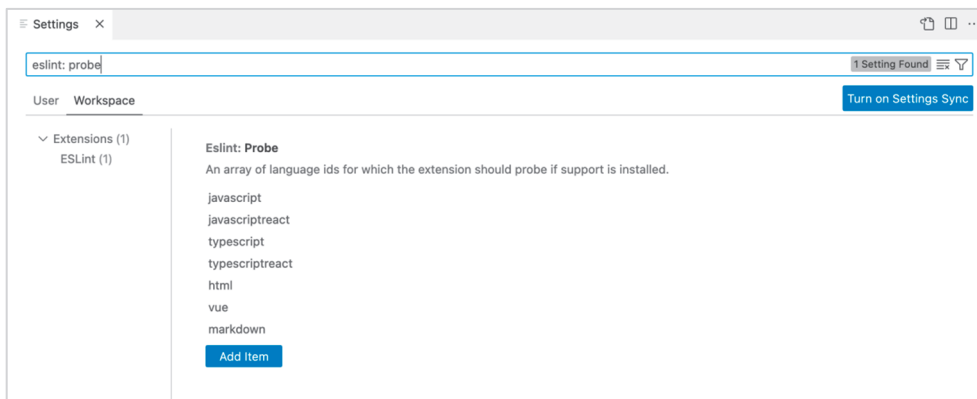
Rysunek 3.1. Aplikacja włączona w przeglądarce w trybie dla deweloperów



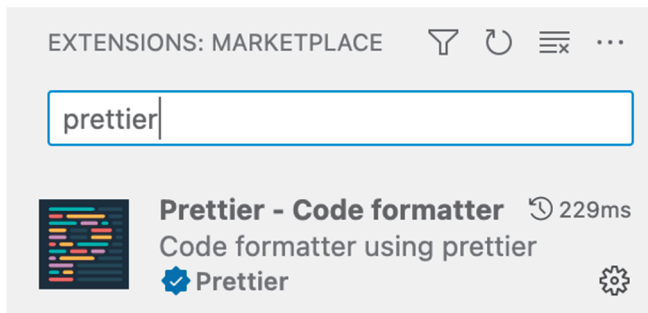
Rysunek 3.2. Automatyczne odświeżenie aplikacji



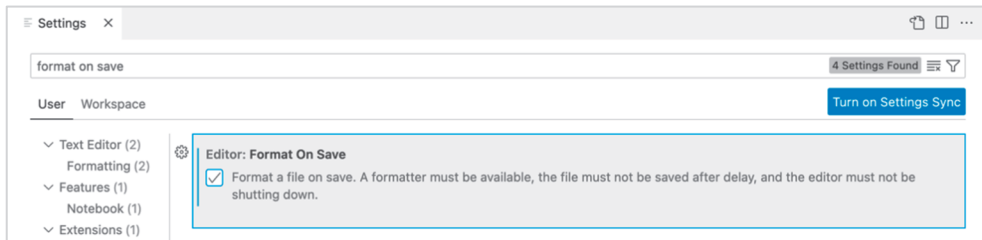
Rysunek 3.3. Rozszerzenie ESLint w Visual Studio Code



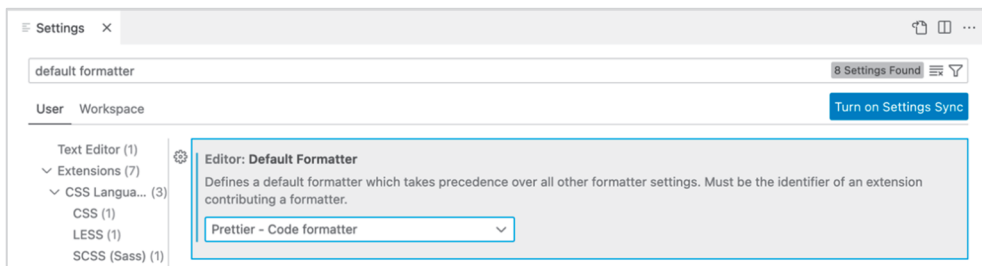
Rysunek 3.4. Konfiguracja ESLint Probe w Visual Studio Code



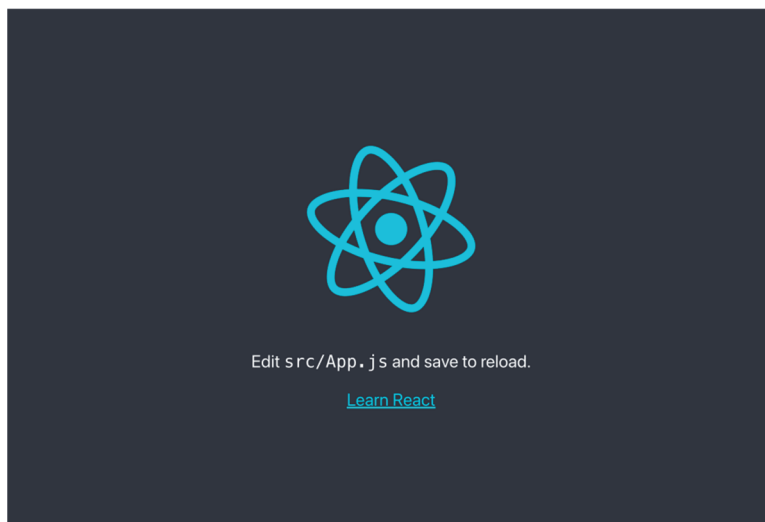
Rysunek 3.5. Rozszerzenie Prettier dla Visual Studio Code



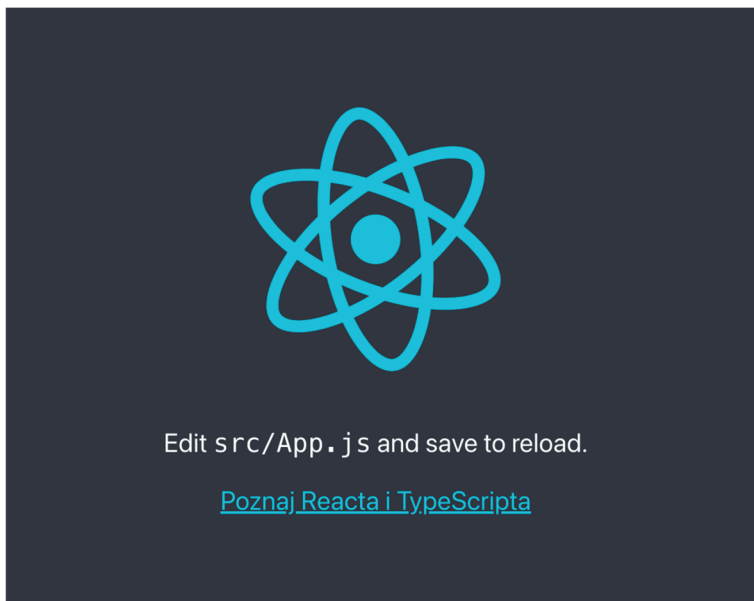
Rysunek 3.6. Ustawienie Format On Save w Visual Studio Code



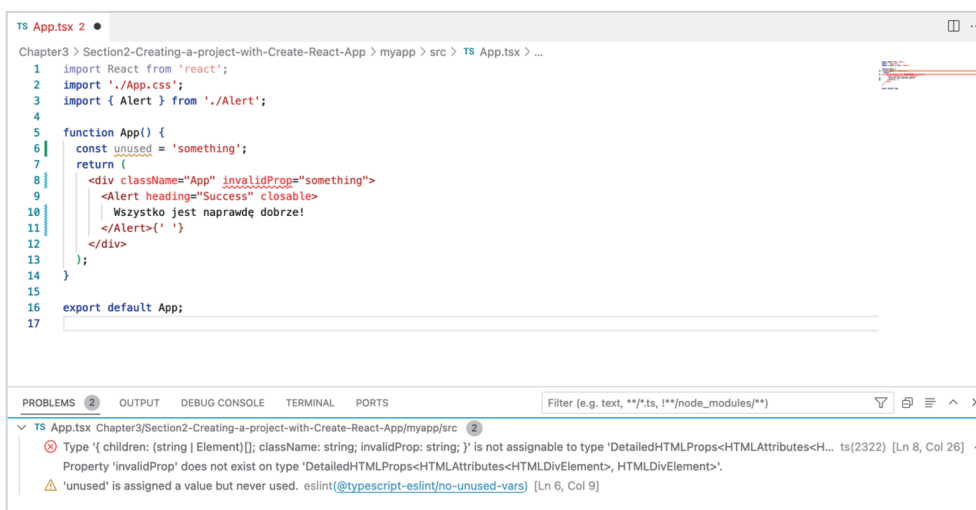
Rysunek 3.7. Wybór Prettiera jako domyślnego formatera w Visual Studio Code



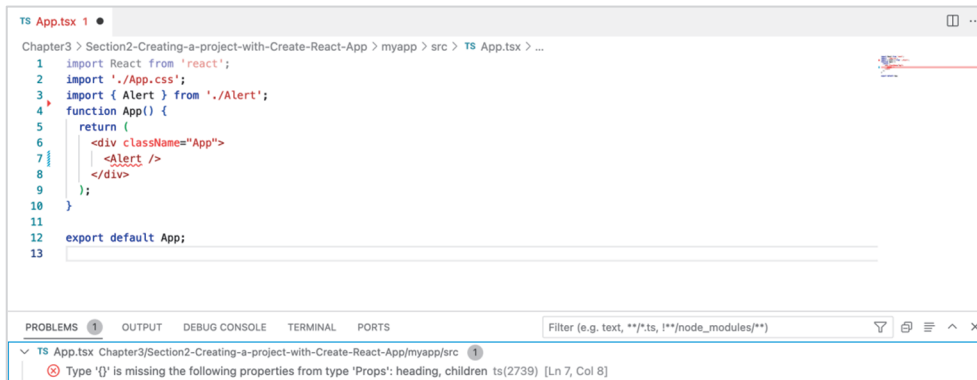
Rysunek 3.8. Działająca aplikacja Reacta w trybie dla programistów



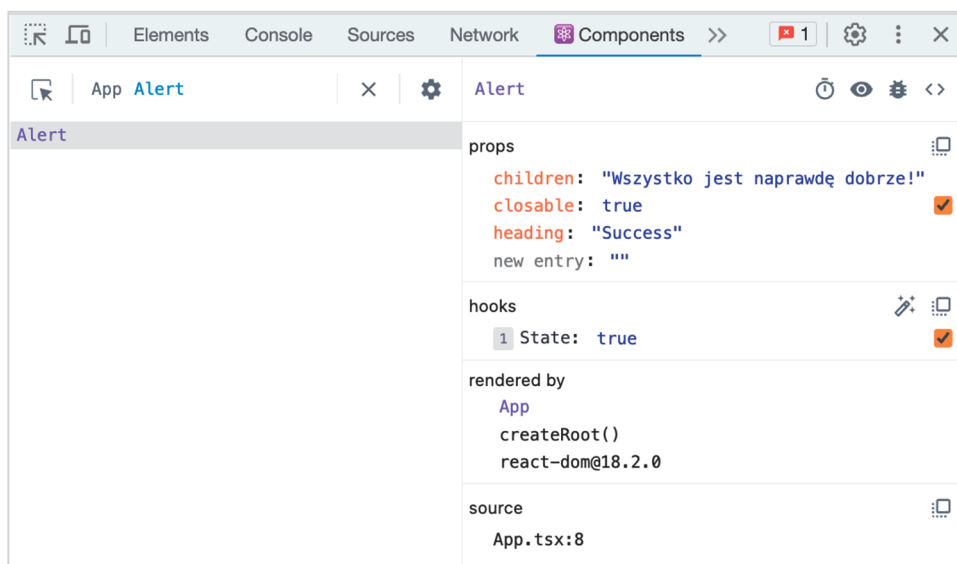
Rysunek 3.9. Odświeżony wygląd aplikacji Reacta



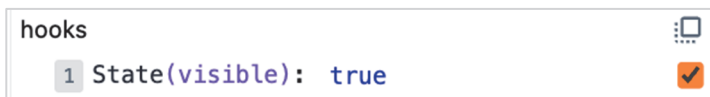
Rysunek 3.10. Nieprawidłowości wykryte przez Visual Studio Code



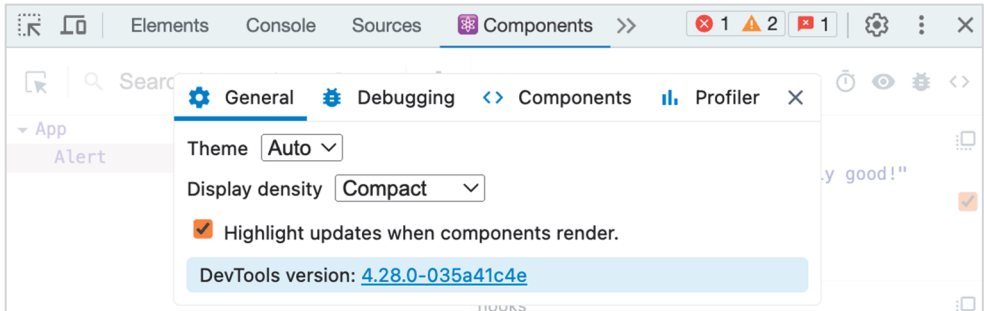
Rysunek 3.11. Błąd w komponencie Alert



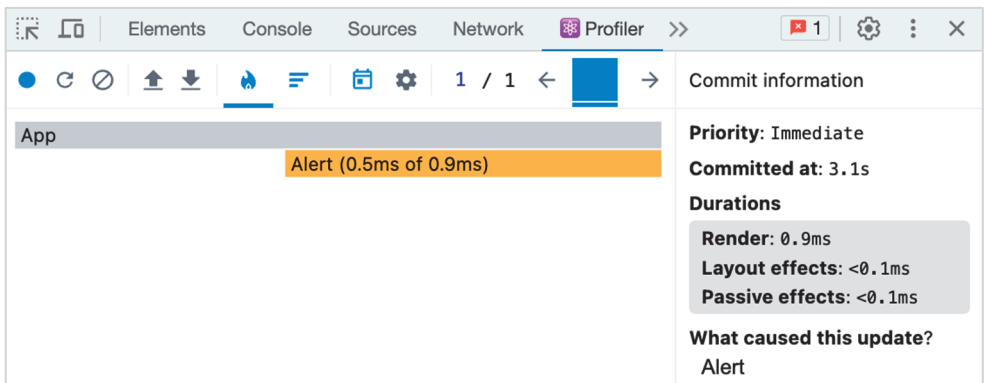
Rysunek 3.12. Panel Components w narzędziu React DevTools



Rysunek 3.13. Dokładna nazwa stanu po naciśnięciu różdżki

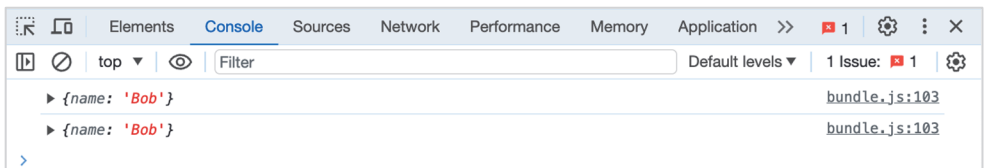


Rysunek 3.14. Zaznaczenie opcji, która zaznacza zmienione fragmenty po ponownym załadowaniu



Rysunek 3.16. Narzędzia React DevTools

Rozdział 4. Korzystanie z hooków Reacta



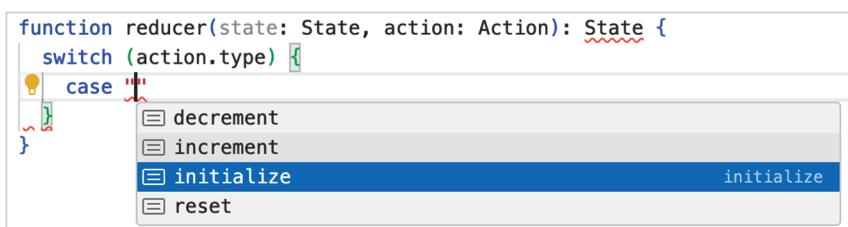
Rysunek 4.1. Rezultat użycia efektu

⊗ ▶ Warning: useEffect must not return anything besides a function, which is used for clean-up. [react-dom.development.js:86](#)

It looks like you wrote `useEffect(async () => ...)` or returned a Promise. Instead, write the async function inside your effect and call it immediately:

```
useEffect(() => {
  async function fetchData() {
    // You can await here
    const response = await MyAPI.getData(someId);
    // ...
  }
  fetchData();
}, [someId]); // Or [] if effect doesn't need props or state
```

Rysunek 4.2. Błąd związany z asynchronicznością efektu



Rysunek 4.5. Podpowieź IntelliSense w funkcji redukującej

Bob, 0

Dodaj

Odejmij

Zresetuj

Rysunek 4.6. Zaznaczenie przycisku Dodaj

Bob, 3

49995000

Dodaj

Odejmij

Zresetuj

Elements

Console

Sources

Network

Performance

Memory

Application

>>

1 Issue

1

×

top

Filter

Default levels

1 Issue

1

×

Wywołanie kosztownej funkcji

bundle.js:104

Wywołanie kosztownej funkcji

installHook.js:1

Wywołanie kosztownej funkcji

bundle.js:104

Wywołanie kosztownej funkcji

react_devtools_backend_compact.js:13029

Wywołanie kosztownej funkcji

bundle.js:104

Wywołanie kosztownej funkcji

react_devtools_backend_compact.js:13029

Wywołanie kosztownej funkcji

bundle.js:104

Wywołanie kosztownej funkcji

react_devtools_backend_compact.js:13029

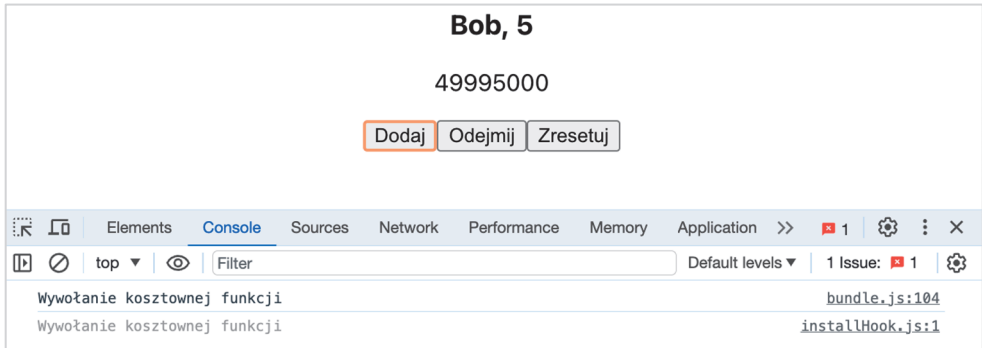
Wywołanie kosztownej funkcji

bundle.js:104

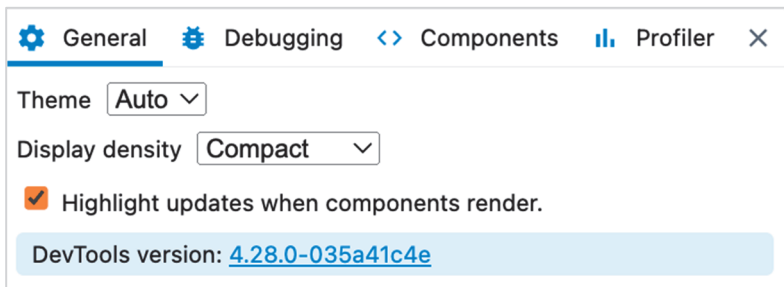
Wywołanie kosztownej funkcji

react_devtools_backend_compact.js:13029

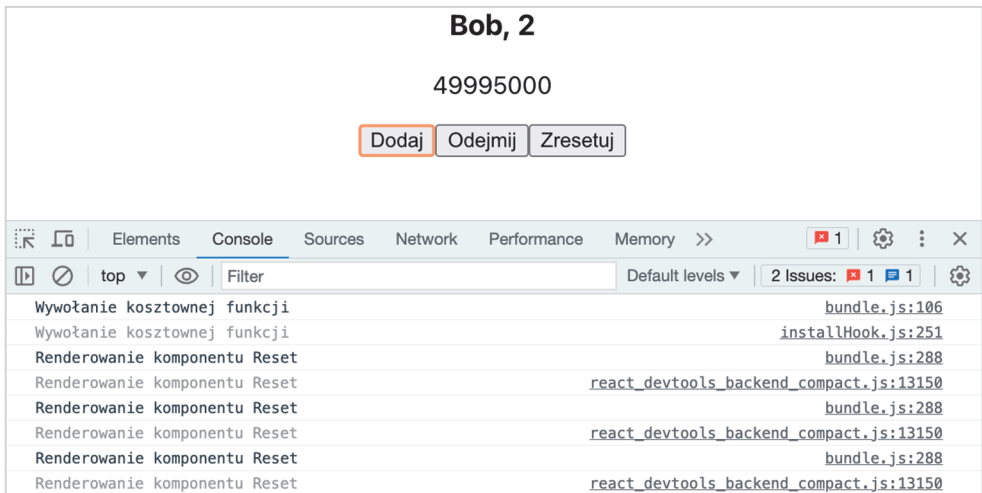
Rysunek 4.7. Wielokrotne wywołanie funkcji



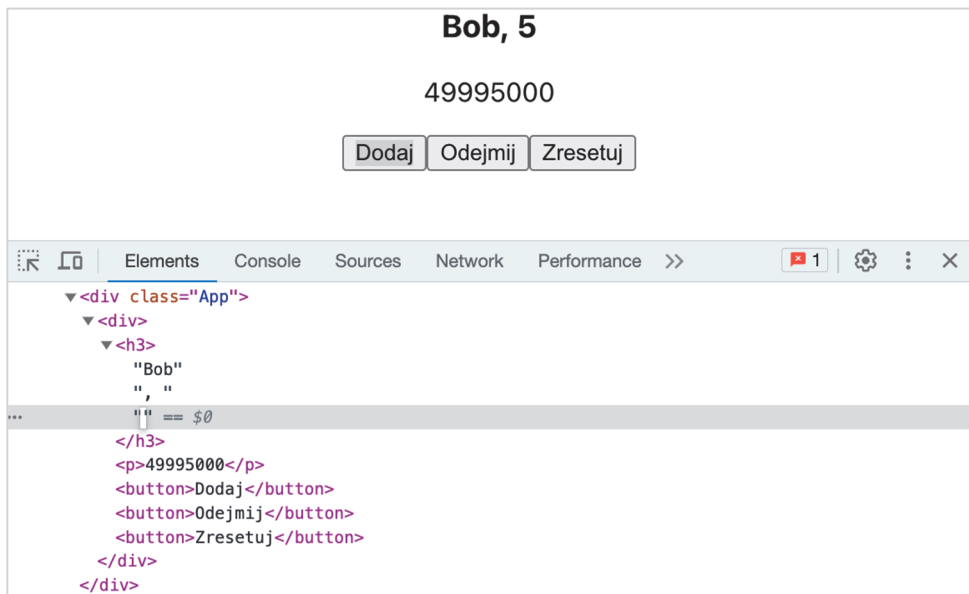
Rysunek 4.8. Zwrócenie zapamiętanych wartości funkcji



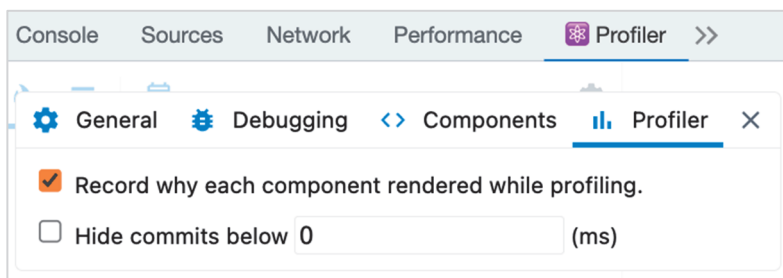
Rysunek 4.9. Opcja podświetlenia przy ponownym renderowaniu



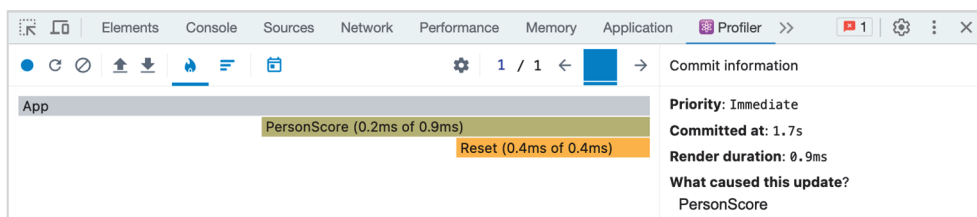
Rysunek 4.10. Zbędne ponowne renderowanie komponentu Reset



Rysunek 4.11. Zaktualizowany element h3 po renderowaniu



Rysunek 4.12. Upewnij się, że zaznaczona jest opcja Record why each component rendered while profiling



Rysunek 4.13. Szczegóły na temat ponownego renderowania komponentu Reset

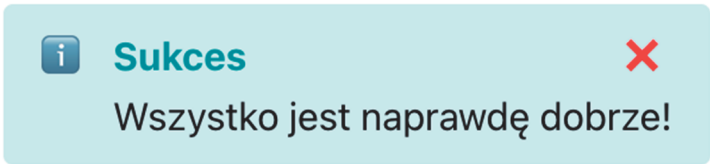
Rozdział 5. Stylizacja interfejsów w Reaccie



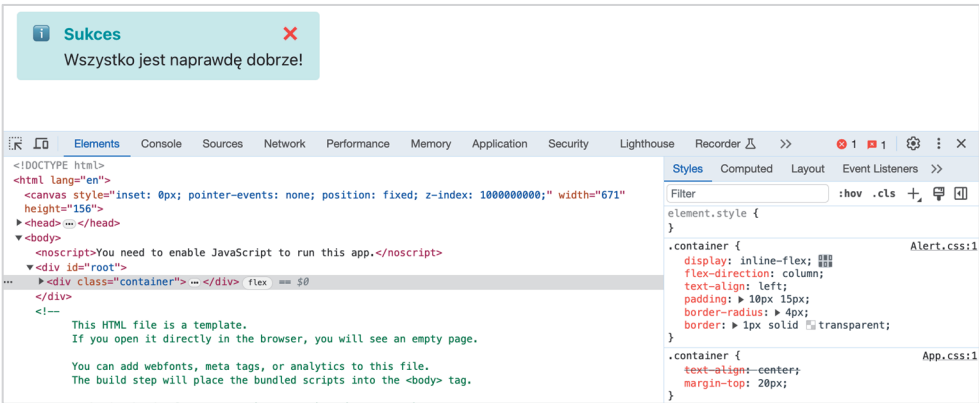
Rysunek 5.1. Element link w pliku index.html



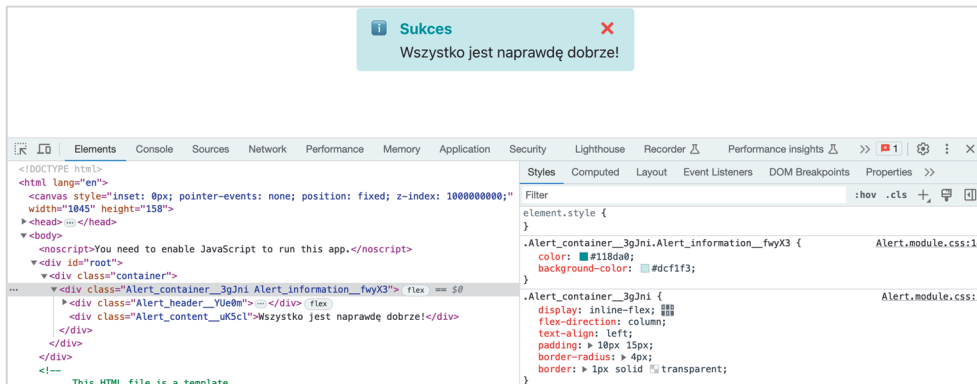
Rysunek 5.2. Dołączony plik CSS, wraz ze zbędną klasą App-header



Rysunek 5.3. Komponent Alert ostylowany przy użyciu standardowego CSS



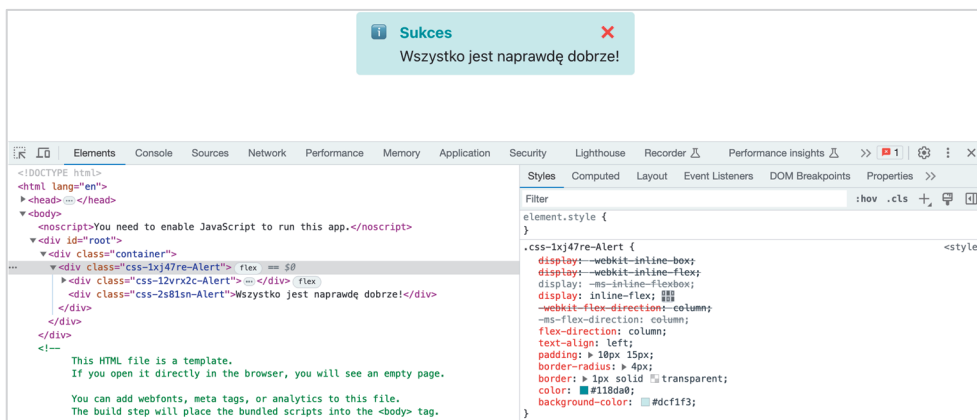
Rysunek 5.4. Kolizja klas CSS



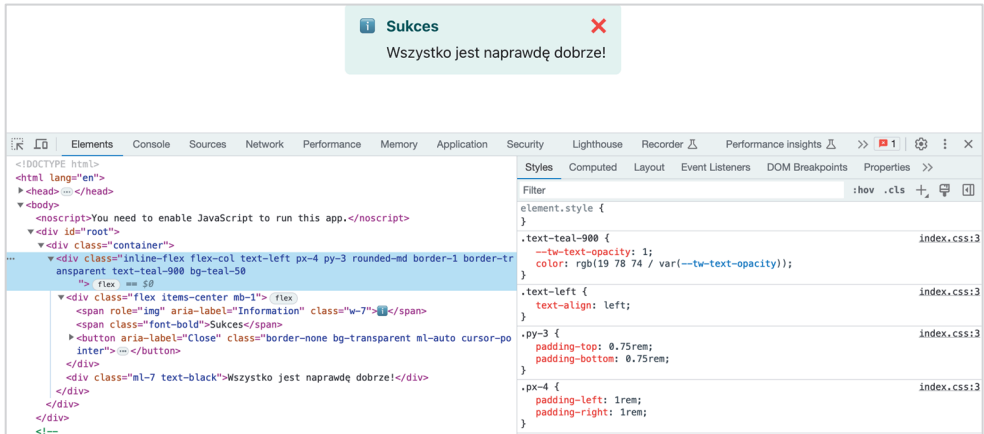
Rysunek 5.5. Nazwy klas w zakresie modułu CSS

```
build > static > css > # main.6adac877.css > ...
.Alert_content__uK5cl{color:#000;margin-left:30px}.Alert_redundant__78xaR{color:red}
```

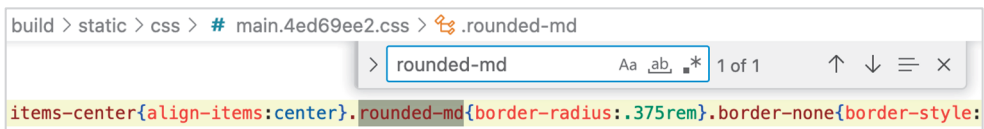
Rysunek 5.6. Niepotrzebna klasa CSS w zestawie CSS



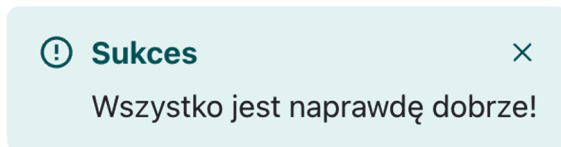
Rysunek 5.7. Nazwy klas Emotiona z podziałem na zakresy



Rysunek 5.8. Alert z zastosowanymi stylami z Tailwinda



Rysunek 5.9. Klasy CSS Tailwinda w złączonym pliku CSS



Rysunek 5.10. Alert z ikoną SVG

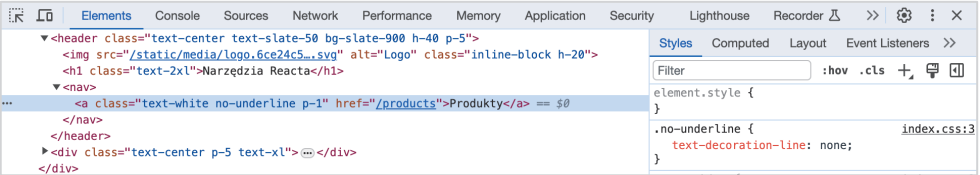
Rozdział 6. Routing przy użyciu biblioteki React Router



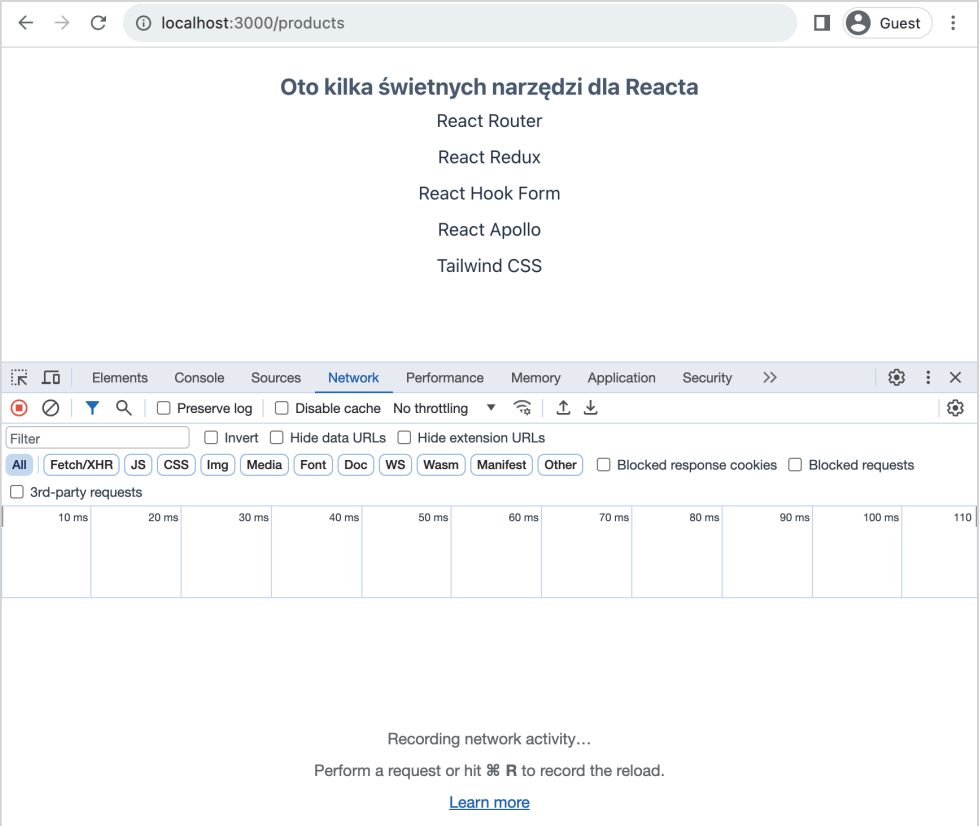
Rysunek 6.1. Standardowa strona biblioteki React Router pokazująca błąd



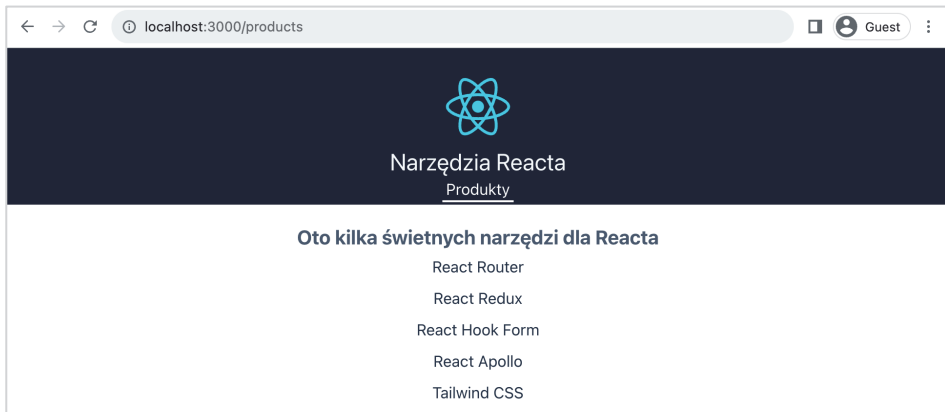
Rysunek 6.3. Główny nagłówek aplikacji



Rysunek 6.4. Analiza komponentu Header



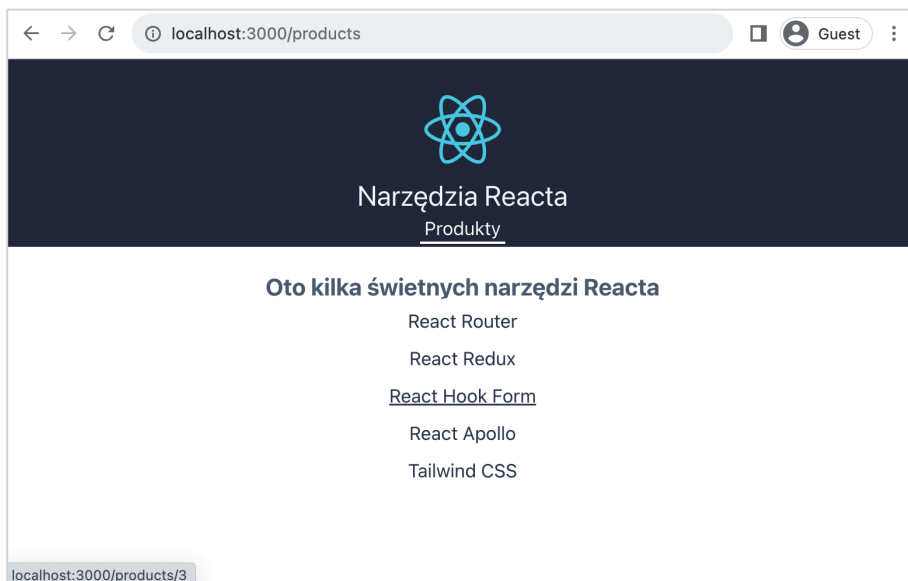
Rysunek 6.5. Nawigacja wewnętrzna aplikacji



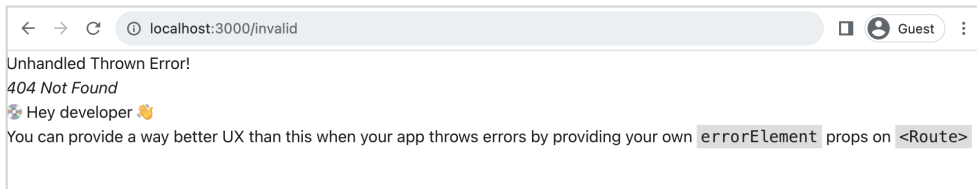
Rysunek 6.7. Główny nagłówek na stronie z listą produktów



Rysunek 6.8. Strona prezentująca produkt



Rysunek 6.9. Linkowane produkty na liście



Rysunek 6.10. Domyślna strona błędu w bibliotece React Router



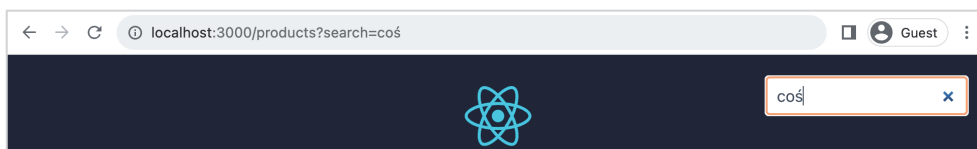
Rysunek 6.11. Nasza strona błędu



Rysunek 6.12. Strona błędu z informacją o wystąpieniu błędu



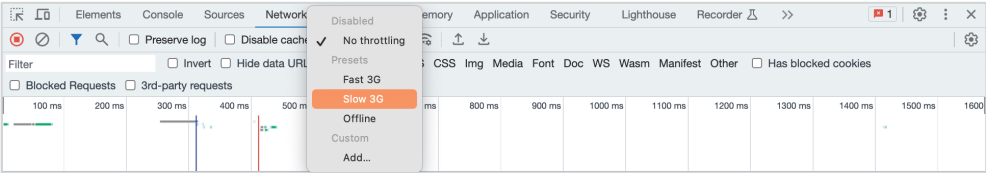
Rysunek 6.13. Strona powitalna



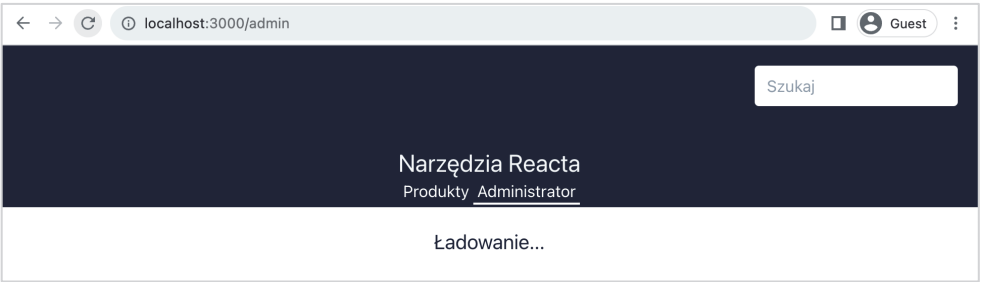
Rysunek 6.14. Dodanie parametru wyszukiwania do URL



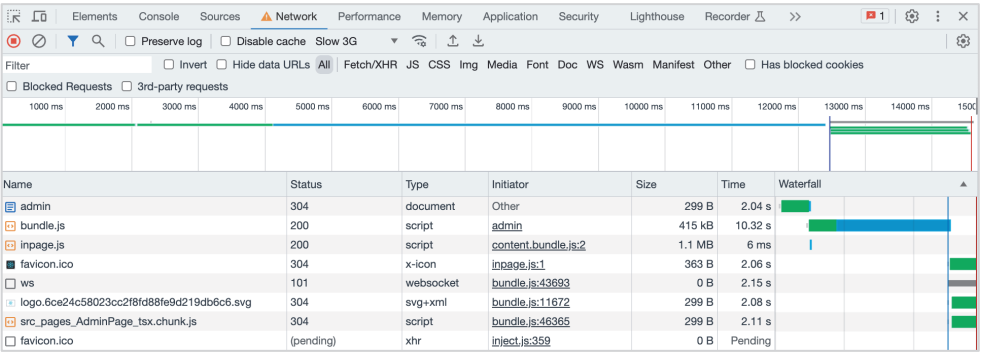
Rysunek 6.15. Lista produktów z opcją filtrowania



Rysunek 6.16. Ustawienie wolniejszego połączenia



Rysunek 6.17. Wskaźnik ładowania



Rysunek 6.18. Proces ładowania strony administratora

Rozdział 7. Praca z formularzami

Skontaktuj się z nami

Jeśli podasz szczegóły, odezwiemy się do Ciebie tak szybko, jak to możliwe.

Imię

Adres e-mail

Powód kontaktu

Dodatkowe informacje

Wyślij

Rysunek 7.2. Podświetlenie konturu i ponowne renderowanie po każdym naciśnięciu klawisza

Skontaktuj się z nami

Jeśli podasz szczegóły, odezwiemy się do Ciebie tak szybko, jak to możliwe.

Imię

Adres e-mail

Powód kontaktu

Wsparcie

Dodatkowe informacje

Przydatne informacje

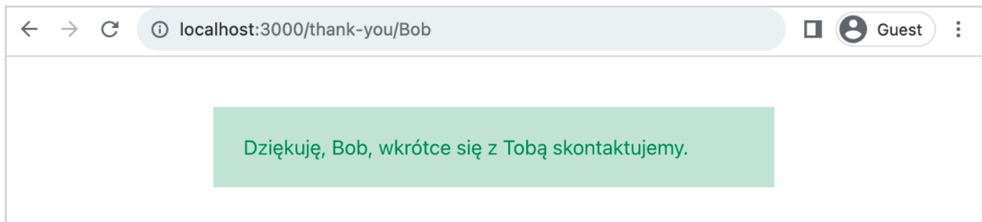
Wyślij

Elements Console Sources Network Performance Memory Application Security >>

top Filter Default levels 2 Issues 2

Obiekt: ▶ {name: 'Bob', email: 'bob@somewhere.com', reason: 'Support', notes: 'Przydatne informacje'} [ContactPage.tsx:22](#)

Rysunek 7.3. Kompletny formularz z danymi przesłanymi do konsoli



Rysunek 7.4. Strona z podziękowaniami

Skontaktuj się z nami

Jeśli podasz szczegóły, odezwiemy się do Ciebie tak szybko, jak to możliwe.

Imię

Adres e-mail

! Please fill in this field.

Powód kontaktu

Dodatkowe informacje

Rysunek 7.5. Komunikat o błędzie walidacji formularza HTML w polu na imię

Skontaktuj się z nami

Jeśli podasz szczegóły, odezwiemy się do Ciebie tak szybko, jak to możliwe.

Imię

Bob

Adres e-mail

somewhere.com

Powód



Please include an '@' in the email address.
'somewhere.com' is missing an '@'.



Dodatkowe informacje

Wyślij

Rysunek 7.6. Komunikat dotyczący walidacji formularza HTML w polu na e-mail

Skontaktuj się z nami

Jeśli podasz szczegóły, odezwiemy się do Ciebie tak szybko, jak to możliwe.

Imię

Musisz podać swoje imię

Adres e-mail

Musisz podać swój adres e-mail

Powód kontaktu

Musisz podać powód kontaktu

Dodatkowe informacje

Submit

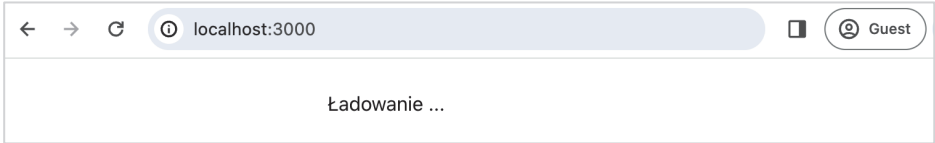
Rysunek 7.7. Zaznaczone ponowne renderowanie i błędy walidacji przy próbie wysłania formularza

Rozdział 8. Zarządzanie stanem w aplikacji

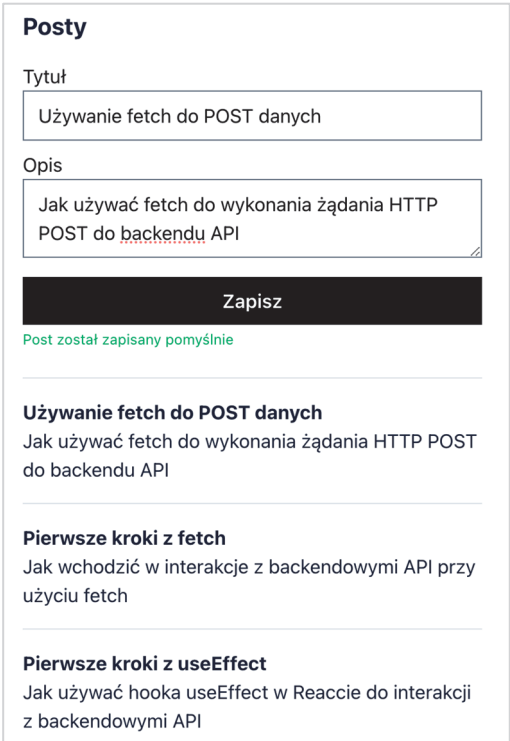


Rysunek 8.2. Wygląd aplikacji przed zalogowaniem

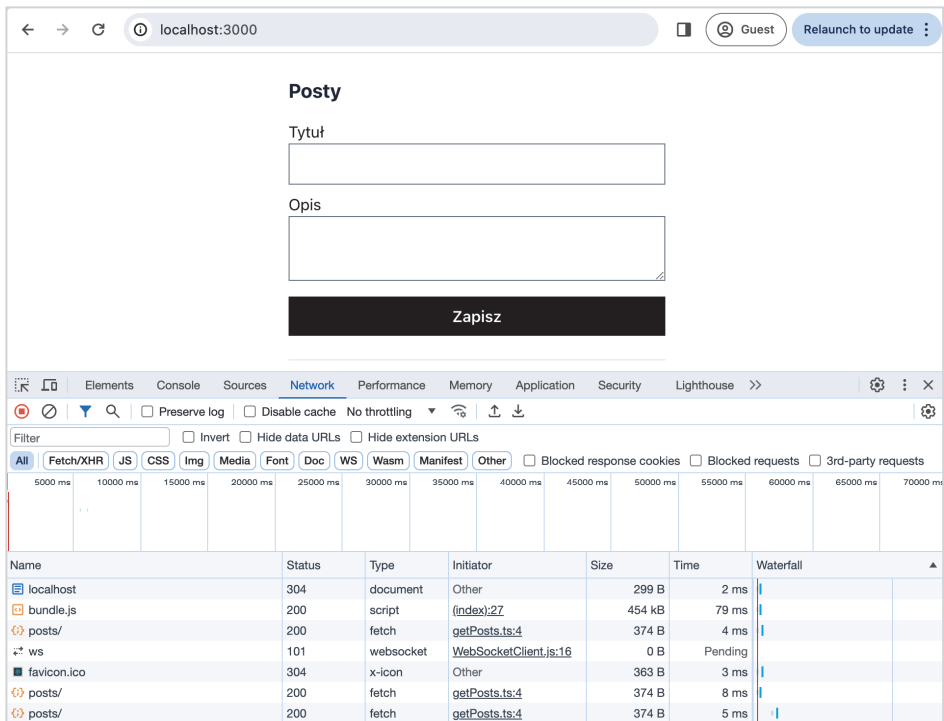
Rozdział 9. Praca z interfejsami RESTful API



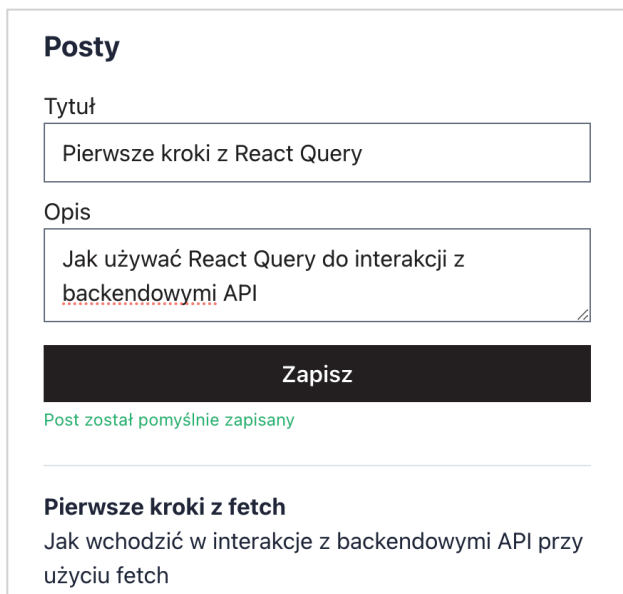
Rysunek 9.3. Wskaźnik ładowania



Rysunek 9.6. Nowy post dodany do listy

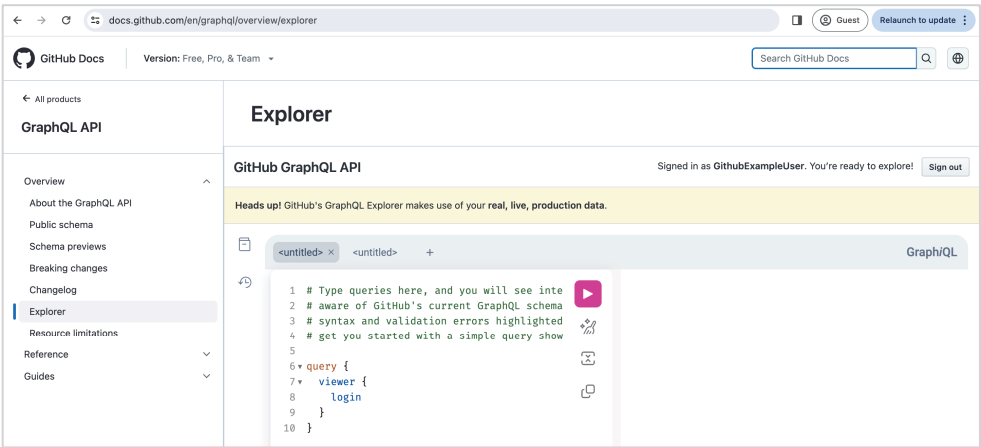


Rysunek 9.9. Dwa zapytania sieciowe dotyczące postów

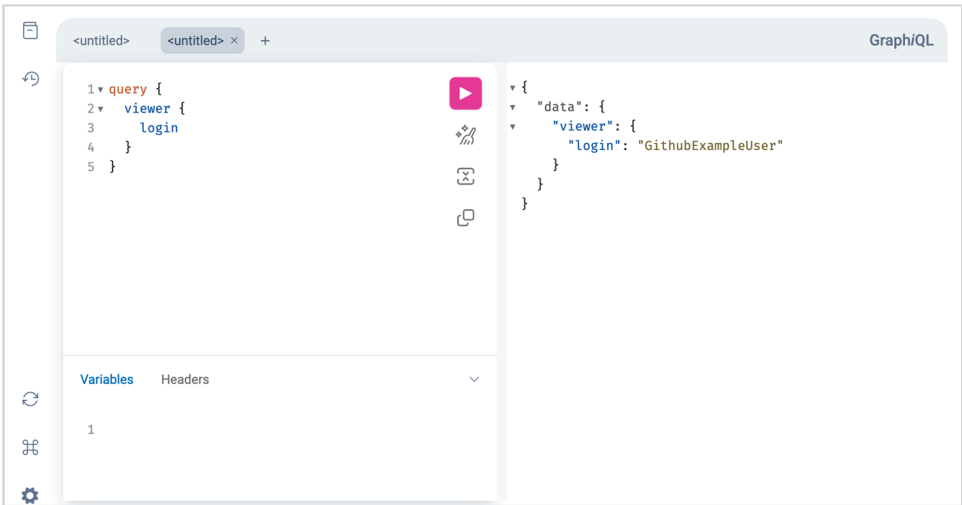


Rysunek 9.10. Nowo dodany post na liście postów

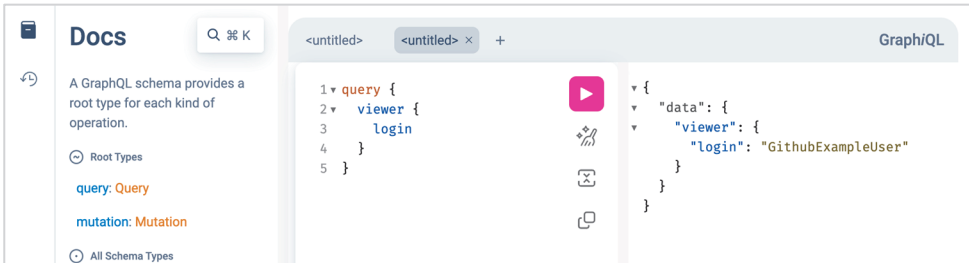
Rozdział 10. Praca z API GraphQL



Rysunek 10.1. Eksplorator API GraphQL GitHuba



Rysunek 10.2. GitHub GraphQL



Rysunek 10.3. Panel Documentation Explorer

`topic(name: String!): Topic`

Look up a topic by name.

`user(login: String!): User`

Lookup a user by login.

`viewer: User!`

The currently authenticated user.

Rysunek 10.4. Obiekt viewer w eksploratorze dokumentacji

Fields

`anyPinnableItems(type: PinnableItemType): Boolean!`

Determine if this repository owner has any items that can be pinned to their profile.

`avatarUrl(size: Int): URI!`

A URL pointing to the user's public avatar.

`bio: String`

The user's public profile bio.

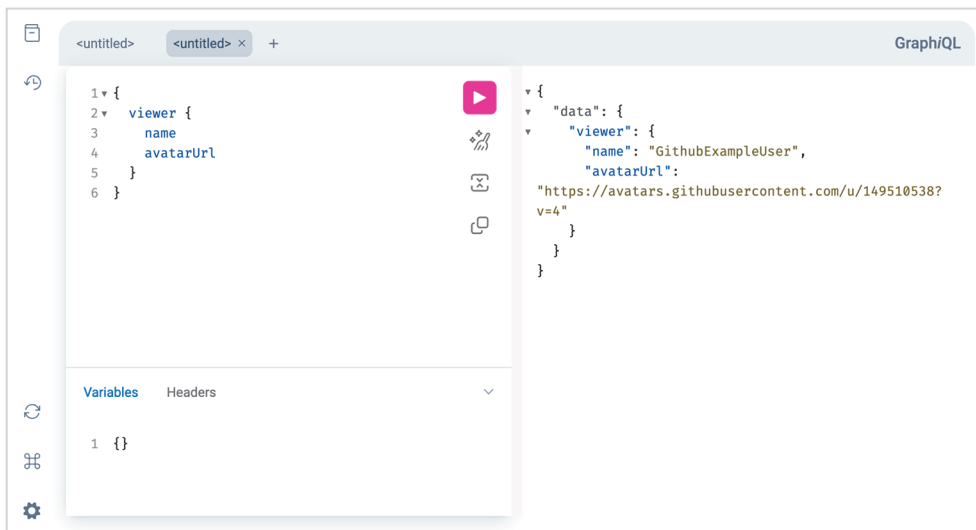
`bioHTML: HTML!`

The user's public profile bio as HTML.

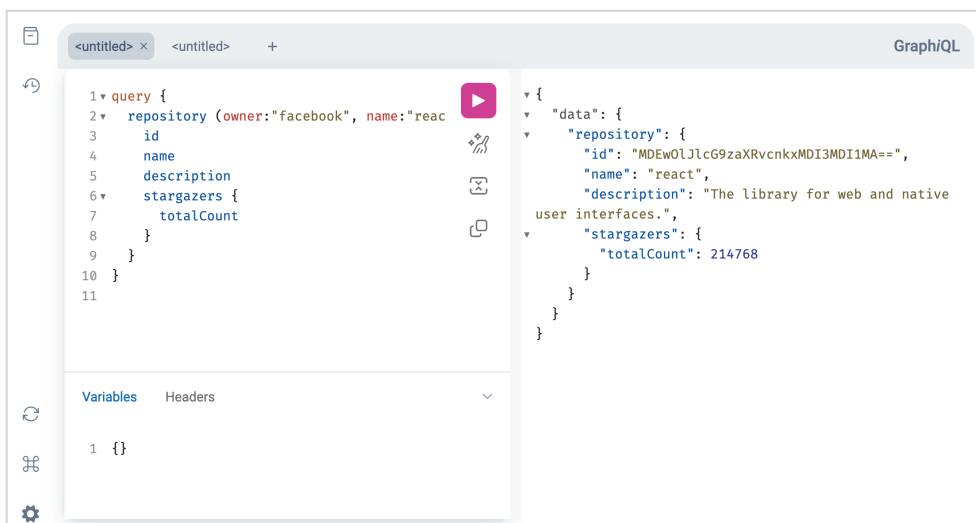
`canReceiveOrganizationEmailsWhenNotificationString!): Boolean!`

Could this user receive email notifications, if the organization had notification restrictions enabled?

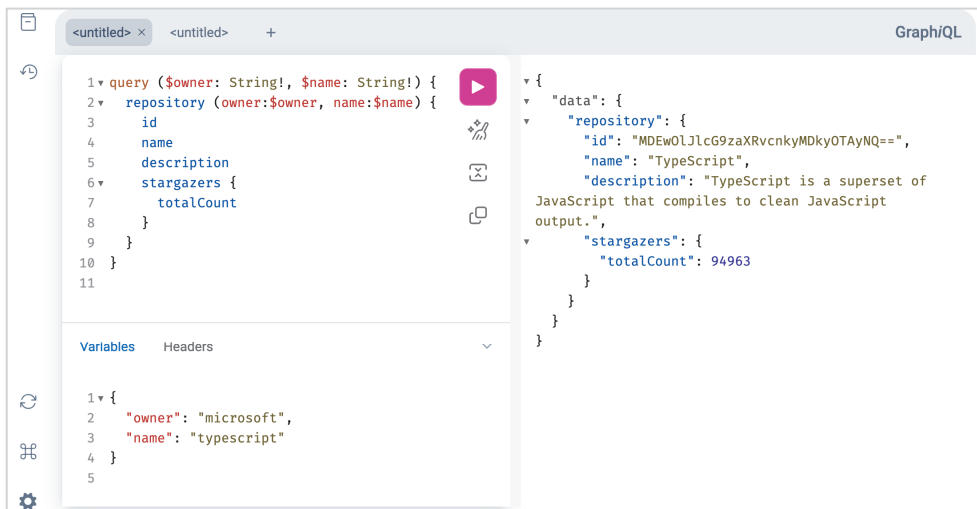
Rysunek 10.5. Lista pól dla typu User



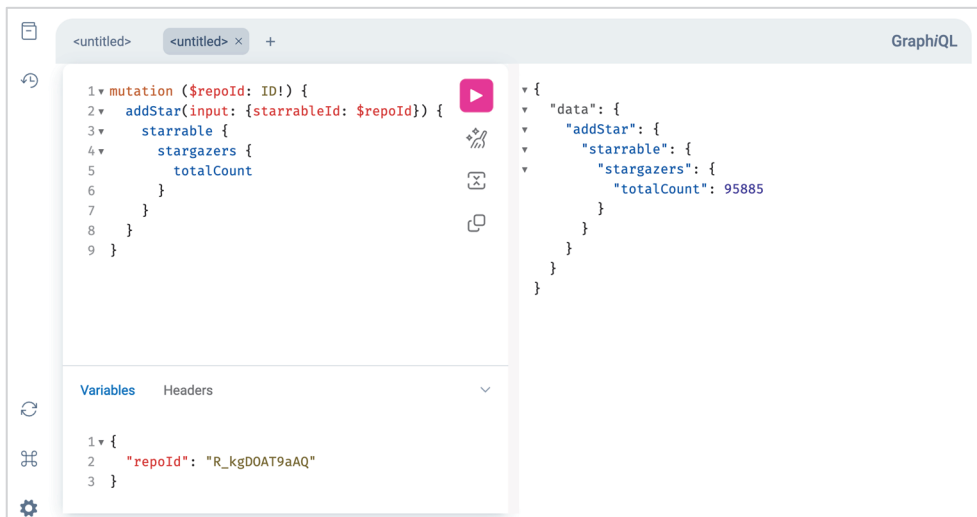
Rysunek 10.6. Zaktualizowany wynik zapytania z dodanym polem avatarUrl



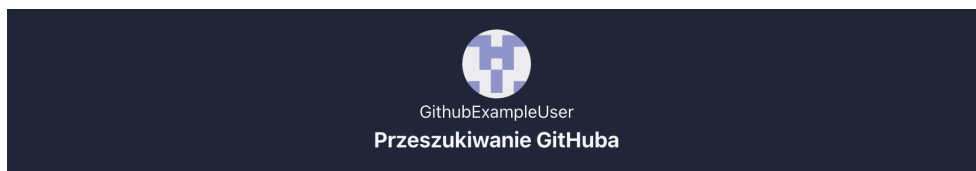
Rysunek 10.7. Zapytanie o konkretne repozytorium



Rysunek 10.8. Zapytanie dotyczące repozytorium TypeScriptu z określonymi parametrami



Rysunek 10.9. Mutacja do oznaczania repozytorium gwiazdką



Rysunek 10.10. Nagłówek z awatarem i imieniem



GithubExampleUser


Przeszukiwanie GitHuba

Organizacja

Repozytorium

Szukaj

Rysunek 10.12. Formularz do wyszukiwania repozytoriów



GithubExampleUser

Przeszukiwanie GitHuba

Organizacja

Repozytorium

Szukaj

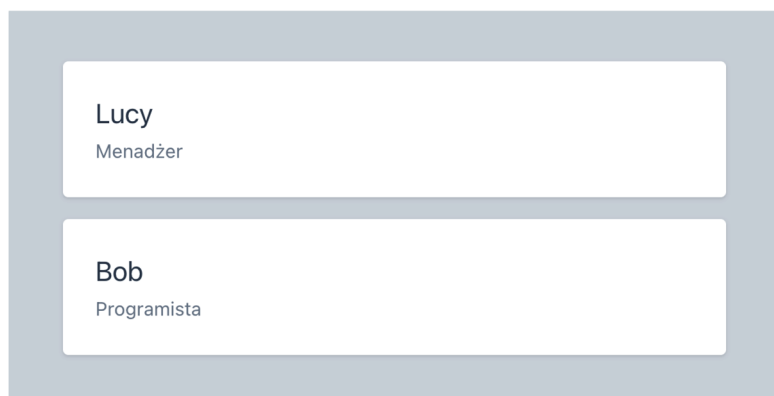
query 36867 Gwiazdki

📦 Powerful asynchronous state management, server-state utilities and data fetching for the web. TS/JS, React Query, Solid Query, Svelte Query and Vue Query.

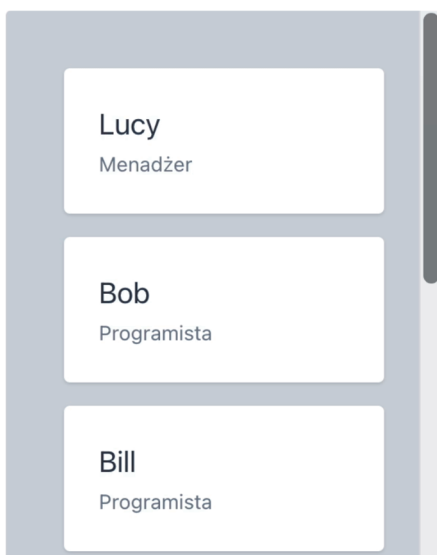
Dodaj gwiazdkę

Rysunek 10.13. Znalezione repozytorium z przyciskiem Dodaj gwiazdkę

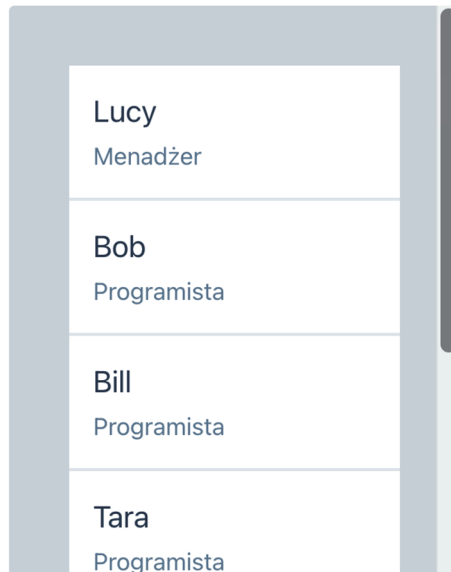
Rozdział 11. Komponenty wielokrotnego użytku



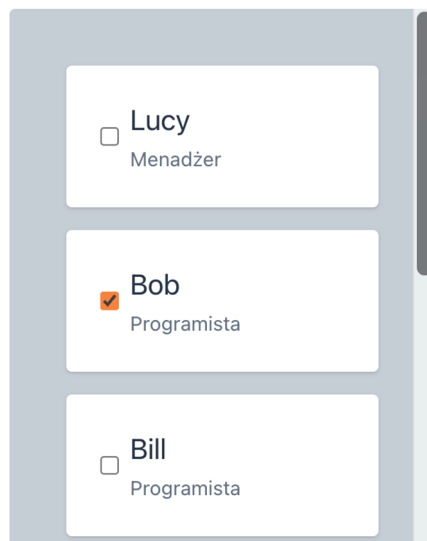
Rysunek 11.1. Prosta wersja komponentu listy



Rysunek 11.2. Dostosowana lista w komponencie



Rysunek 11.3. Zmodyfikowane elementy listy



Rysunek 11.4. Pola wyboru przy każdym elemencie

Rozdział 12. Testy jednostkowe z użyciem frameworka Jest i biblioteki React Testing Library

```
PASS src/Checklist/assertValueCanBeRendered.test.ts
PASS src/Checklist/isChecked.test.ts

Test Suites: 2 passed, 2 total
Tests:       5 passed, 5 total
Snapshots:   0 total
Time:        0.866 s, estimated 1 s
Ran all test suites.

Watch Usage
  > Press f to run only failed tests.
  > Press o to only run tests related to changed files.
  > Press q to quit watch mode.
  > Press p to filter by a filename regex pattern.
  > Press t to filter by a test name regex pattern.
  > Press Enter to trigger a test run.
```

Rysunek 12.1. Pierwsze uruchomienie testów

```
PASS src/Checklist/isChecked.test.ts
FAIL src/Checklist/assertValueCanBeRendered.test.ts
  ● should raise exception when not a string or number

    expect(received).toThrow(expected)

    Expected substring: "value is not a string or a numberX"
    Received message:   "value is not a string or a number"

       5 |   ): asserts value is IdValue {
       6 |     if (typeof value !== "string" && typeof value !== "number") {
    >    7 |       throw new Error("value is not a string or a number");
         |             ^
       8 |     }
       9 |   }
      10 |
```

Rysunek 12.2. Nieudany test

```
PASS src/Checklist/isChecked.test.ts
  ✓ should return true when in checkedIds (1 ms)
  ✓ should return false when not in checkedIds

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   0 total
Time:        0.274 s, estimated 1 s
Ran all test suites matching /isChecked/i.

Watch Usage: Press w to show more.
```

Rysunek 12.4. Jest uruchamia plik testowy w poszukiwaniu pasującego wzorca

PASS src/Checklist/isChecked.test.ts

Test Suites: 1 skipped, 1 passed, 1 of 2 total

Tests: 4 skipped, 1 passed, 5 total

Snapshots: 0 total

Time: 0.577 s, estimated 1 s

Ran all test suites with tests matching "should return false when not in checkedIds".

Watch Usage: Press w to show more.█

Rysunek 12.5. Jest uruchamianie test o nazwie pasującej do wzorca

PASS src/Checklist/Checklist.test.tsx

✓ should render correct list items when data specified (15 ms)

✓ should render correct list items when renderItem specified (2 ms)

Test Suites: 1 passed, 1 total

Tests: 2 passed, 2 total

Snapshots: 0 total

Time: 0.333 s, estimated 1 s

Ran all test suites matching /Checklist\.test\.tsx/i.

Watch Usage: Press w to show more.█

Rysunek 12.6. Testy komponentów zaliczone

PASS src/Checklist/Checklist.test.tsx

✓ should render correct list items when data specified (12 ms)

✓ should render correct list items when renderItem specified (2 ms)

✓ should render correct checked items when specified (2 ms)

Test Suites: 1 passed, 1 total

Tests: 3 passed, 3 total

Snapshots: 0 total

Time: 0.432 s, estimated 1 s

Ran all test suites matching /Checklist\.test\.tsx/i.

Watch Usage: Press w to show more.█

Rysunek 12.7. Wszystkie trzy testy komponentów zaliczone

PASS src/Checklist/Checklist.test.tsx

✓ should render correct list items when data specified (13 ms)

✓ should render correct list items when renderItem specified (1 ms)

✓ should render correct checked items when specified (3 ms)

✓ should check and uncheck items when clicked (38 ms)

✓ should call onCheckedIdsChange when clicked (11 ms)

Test Suites: 1 passed, 1 total

Tests: 5 passed, 5 total

Snapshots: 0 total

Time: 0.498 s, estimated 1 s

Ran all test suites matching /Checklist\.test\.tsx/i.

Watch Usage: Press w to show more.█

Rysunek 12.8. Pięć zaliczonych testów komponentów

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	57.44	62.5	60	57.77	
src	0	0	0	0	
App.tsx	0	0	0	0	5-14
index.tsx	0	100	100	0	7-19
reportWebVitals.ts	0	0	0	0	3-10
src/Checklist	93.1	93.75	90	92.85	
Checklist.tsx	100	100	100	100	
...ValueCanBeRendered.ts	100	100	100	100	
getNewCheckedIds.ts	50	50	50	50	9-10
index.ts	0	0	0	0	
isChecked.ts	100	100	100	100	
types.ts	0	0	0	0	
useChecked.ts	100	100	100	100	

Rysunek 12.9. Raport pokrycia kodu w terminalu

All files

57.44% Statements 27/4762.5% Branches 15/2460% Functions 9/1557.77% Lines 26/45

Press *n* or *j* to go to the next uncovered block, *b*, *p* or *k* for the previous block.

Filter:

File	Statements	Branches	Functions	Lines
src	0%	0/18	0%	0/5
src/Checklist	93.1%	27/29	90%	92.85%

Rysunek 12.10. Raport pokrycia kodu testami w HTML

All files src/Checklist

93.1% Statements 27/2993.75% Branches 15/1690% Functions 9/1092.85% Lines 26/28

Press *n* or *j* to go to the next uncovered block, *b*, *p* or *k* for the previous block.

Filter:

File	Statements	Branches	Functions	Lines
Checklist.tsx	100%	10/10	100%	10/10
assertValueCanBeRendered.ts	100%	2/2	100%	2/2
getNewCheckedIds.ts	50%	2/4	50%	2/4
index.ts	0%	0/0	0%	0/0
isChecked.ts	100%	1/1	100%	1/1
types.ts	0%	0/0	0%	0/0
useChecked.ts	100%	12/12	100%	11/11

Rysunek 12.11. Raport pokrycia kodu testami dla plików komponentu listy kontrolnej

All files / src/Checklist getNewCheckedIds.ts

50% Statements 2/4 50% Branches 1/2 50% Functions 1/2 50% Lines 2/4

Press *n* or *j* to go to the next uncovered block, *b*, *p* or *k* for the previous block.

```
1 import { isChecked } from "../isChecked";
2 import { IdValue } from "../types";
3
4 export function getNewCheckedIds(
5   currentCheckedIds: IdValue[],
6   checkedId: IdValue
7 ) {
8   2x if (isChecked(currentCheckedIds, checkedId)) {
9     return currentCheckedIds.filter(
10      (itemCheckedId) => itemCheckedId !== checkedId
11    );
12   } else {
13     2x return currentCheckedIds.concat(checkedId);
14   }
15 }
16
```

Rysunek 12.12. Raport pokrycia dla pliku getNewCheckedIds.ts

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	61.7	66.66	66.66	62.22	
src	0	0	0	0	
App.tsx	0	0	0	0	5-14
index.tsx	0	100	100	0	7-19
reportWebVitals.ts	0	0	0	0	3-10
src/Checklist	100	100	100	100	
Checklist.tsx	100	100	100	100	
...ValueCanBeRendered.ts	100	100	100	100	
getNewCheckedIds.ts	100	100	100	100	
index.ts	0	0	0	0	
isChecked.ts	100	100	100	100	
types.ts	0	0	0	0	
useChecked.ts	100	100	100	100	

Rysunek 12.13. 100% pokrycia komponentu listy kontrolnej

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	65.9	66.66	66.66	66.66	
src	0	0	0	0	
App.tsx	0	0	0	0	5-14
reportWebVitals.ts	0	0	0	0	3-10
src/Checklist	100	100	100	100	
Checklist.tsx	100	100	100	100	
...ValueCanBeRendered.ts	100	100	100	100	
getNewCheckedIds.ts	100	100	100	100	
isChecked.ts	100	100	100	100	
useChecked.ts	100	100	100	100	

Rysunek 12.14. Pliki types.ts i index.ts usunięte z raportu pokrycia