Przydatne rozszerzenia do Visual Studio Code

Volar

Vue Language Features (Volar)

Johnson Chu | 304,586 installs | ★★★★★ (36) | Free

Language support for Vue 3

Installation

Launch VS Code Quick Open (`Ctrl+P`), paste the following command, and press enter.

```
ext install johnsoncodehk.volar
```

Copy | More Info

Instalacja i uruchomienie aplikacji

Pierwsze uruchomienie aplikacji

Instalujemy vue-cli

```
1   npm install -g @vue/cli
```

Tworzymy defaulotwy projekt

```
1   vue create front
```

```
Vue CLI v5.0.1
? Please pick a preset: Manually select features
? Check the features needed for your project: Babel, TS, Linter
? Choose a version of Vue.js that you want to start the project with 3.x
? Use class-style component syntax? No
? Use Babel alongside TypeScript (required for modern mode, auto-detected polyfills, transpiling JSX)? Yes
? Pick a linter / formatter config: Basic
? Pick additional lint features: Lint on save
? Where do you prefer placing config for Babel, ESLint, etc.? In dedicated config files
? Save this as a preset for future projects? No
```

Pierwsze uruchomienie aplikacji - 2

Jeśli mamy problem z uprawnieniami (windows)

```
vue.cmd create front
```

Przechodzimy do katalogu

```
cd front
```

Pierwsze uruchomienie aplikacji - 3

Edytujemy vue.config.js

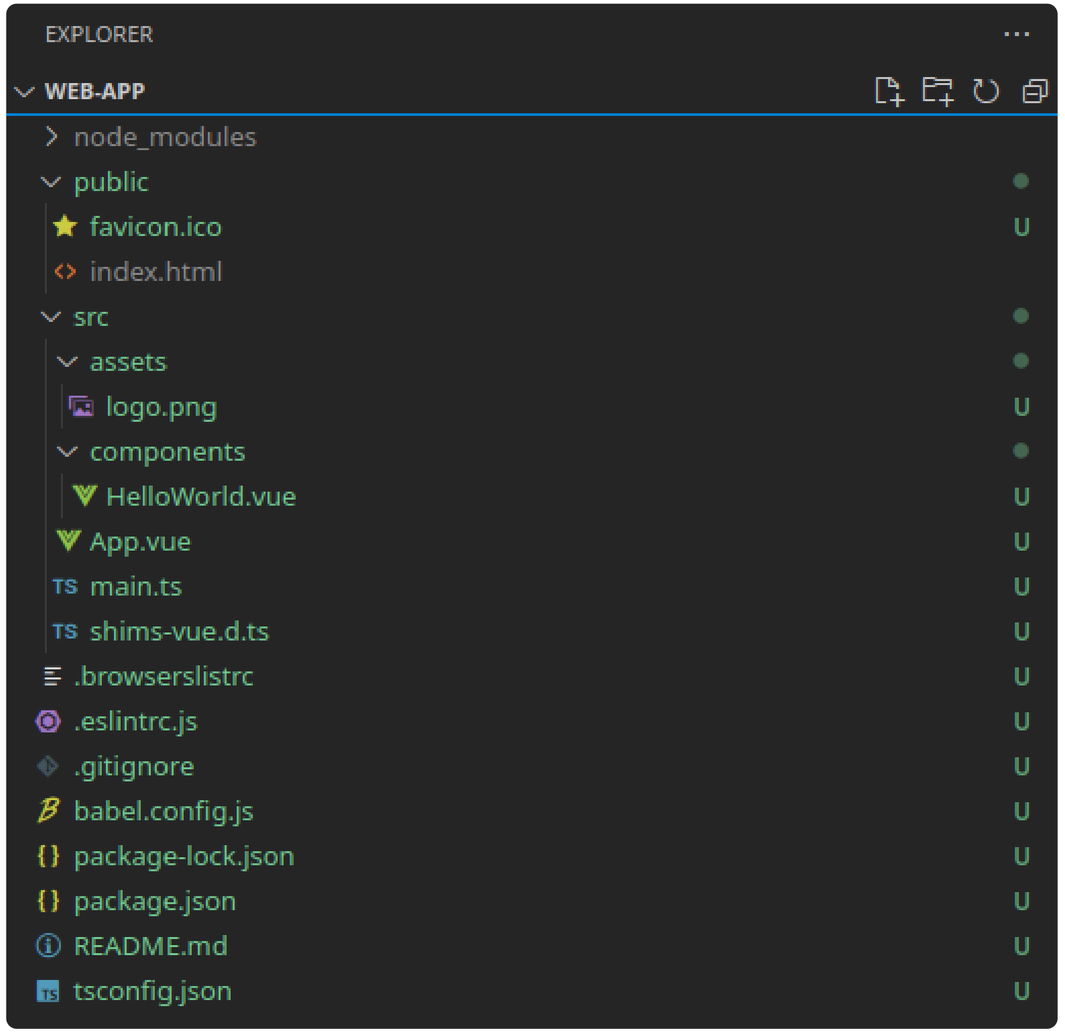
```
const { defineConfig } = require('@vue/cli-service');

module.exports = defineConfig({
  transpileDependencies: true,
  devServer: {
    historyApiFallback: true,
    allowedHosts: "all",
  },
});
```

Uruchamiamy aplikacje w trybie developerskim

```
npm run serve
```

Struktura katalogów



Data binding & event-emiting

App.vue HelloWorld.vue × + Import Map

```
1 v <template>
2     <HelloWorld msg="Welcome to Your Vue.js +
  TypeScript App" />
3   </template>
4
5 v <script lang="ts">
6   import { defineComponent } from 'vue'
7   import HelloWorld from './HelloWorld.vue'
8
9 v export default defineComponent({
10    name: 'App',
11 v  components: {
12      HelloWorld,
13    },
14  })
15  </script>
16
17 v <style>
18 v #app {
```

PREVIEW JS CSS SSR

Welcome to Your Vue.j TypeScript App

For a guide and recipes on how to configure / custc project,
check out the vue-cli documentation.

Installed CLI Plugins

babel typescript eslint

Essential Links

Dodajemy routing do aplikacji

```
vue add router
```

Sprawdźmy jak wyglądają nasze dwa widoki

HomeView.vue

```vue
<template>
    <div class="home">
      <img alt="Vue logo" src="../assets/logo.png" />
      <HelloWorld msg="Welcome to Your Vue.js App" />
    </div>
</template>

<script lang="ts">
  // @ is an alias to /src
  import HelloWorld from '@/components/HelloWorld.vue'

  export default {
    name: 'Home',
    components: {
      HelloWorld,
    },
  }
</script>
```

AboutView.vue

```vue
<template>
  <div class="about">
    <h1>This is an about page</h1>
  </div>
</template>
```

```ts
import { createRouter, createWebHashHistory, RouteRecordRaw } from 'vue-router'
import Home from '../views/Home.vue'

const routes: Array<RouteRecordRaw> = [
  {
    path: '/',
    name: 'Home',
    component: Home,
  },
  {
    path: '/about',
    name: 'About',
    // route level code-splitting
    // this generates a separate chunk (about.[hash].js) for this route
    // which is lazy-loaded when the route is visited.
    component: () =>
      import(/* webpackChunkName: "about" */ '../views/About.vue'),
  },
]

const router = createRouter({
  history: createWebHashHistory(),
  routes,
})

export default router
```

Tworzymy widok startView

```vue
<template>
  <p>start</p>
</template>
<script lang="ts">
import { defineComponent } from 'vue'

export default defineComponent({
  data: () => ({
    number: '',
  }),
  methods: {
    async call() {
      await fetch('http://3000-<adres naszego workspace>/call', {
        method: 'POST',
        headers: { 'Content-type': 'application/json; charset=UTF-8' },
        body: JSON.stringify({ number: this.number }),
      })
    },
  },
})
</script>
```

Dodajemy nową ścieżkę do routingu

src/router/index.ts

```
1   import { createRouter, createWebHashHistory, RouteRecordRaw } from 'vue-router'
2   import Start from '../views/StartView.vue'
3
4   const routes: Array<RouteRecordRaw> = [
5     {
6       path: '/',
7       name: 'start',
8       component: Start,
9     },
10  ]
11
12  const router = createRouter({
13    history: createWebHashHistory(),
14    routes,
15  })
16
17  export default router
```

Edytujemy App.vue

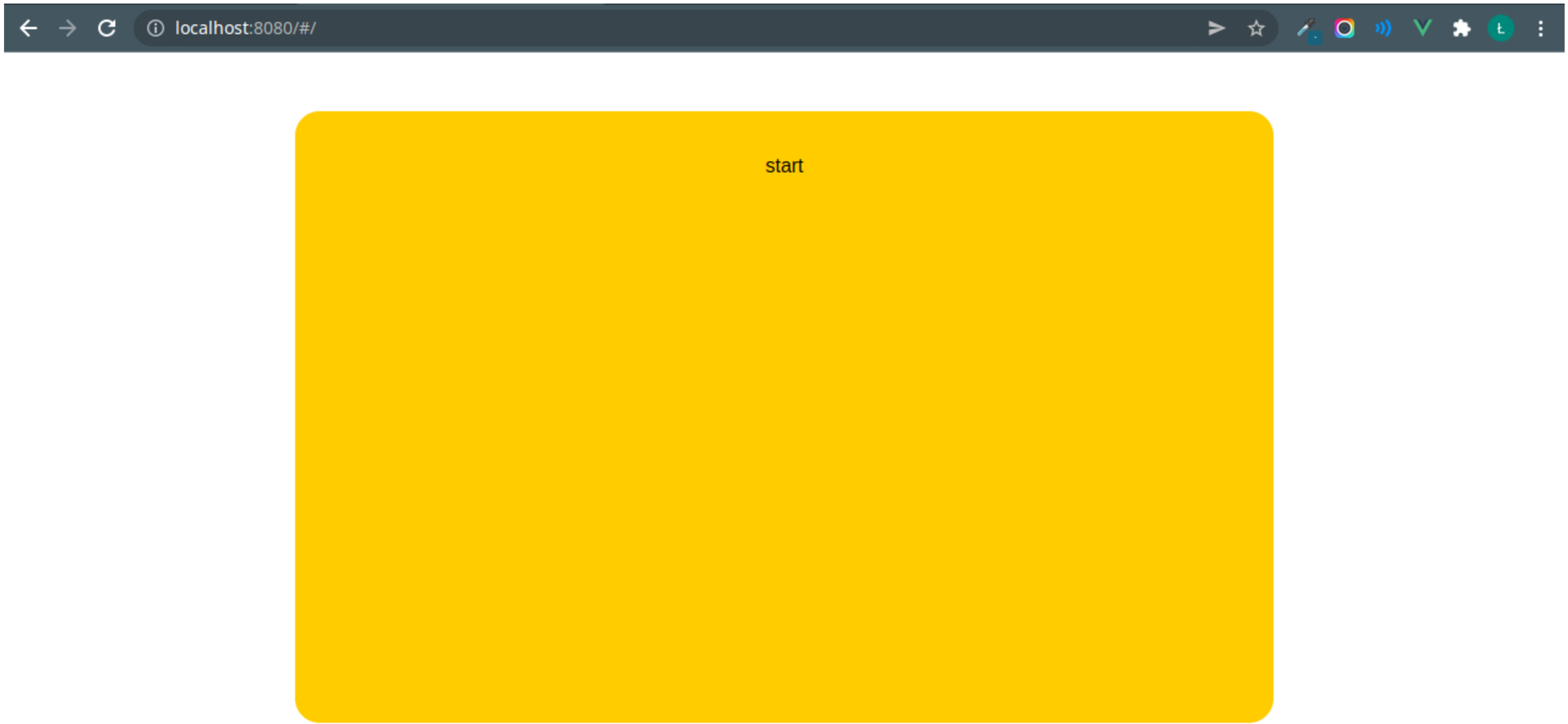
src/views/App.vue

```
<template>
  <div class="window">
    <router-view />
  </div>
</template>

<style>
/* Tu wklej style
Dostępne w plikach na teams lub pod przyciskiem poniżej
*/
</style>
```

Klik

Widok start



Dodajemy template do StartView.vue

src/views/StartView.vue

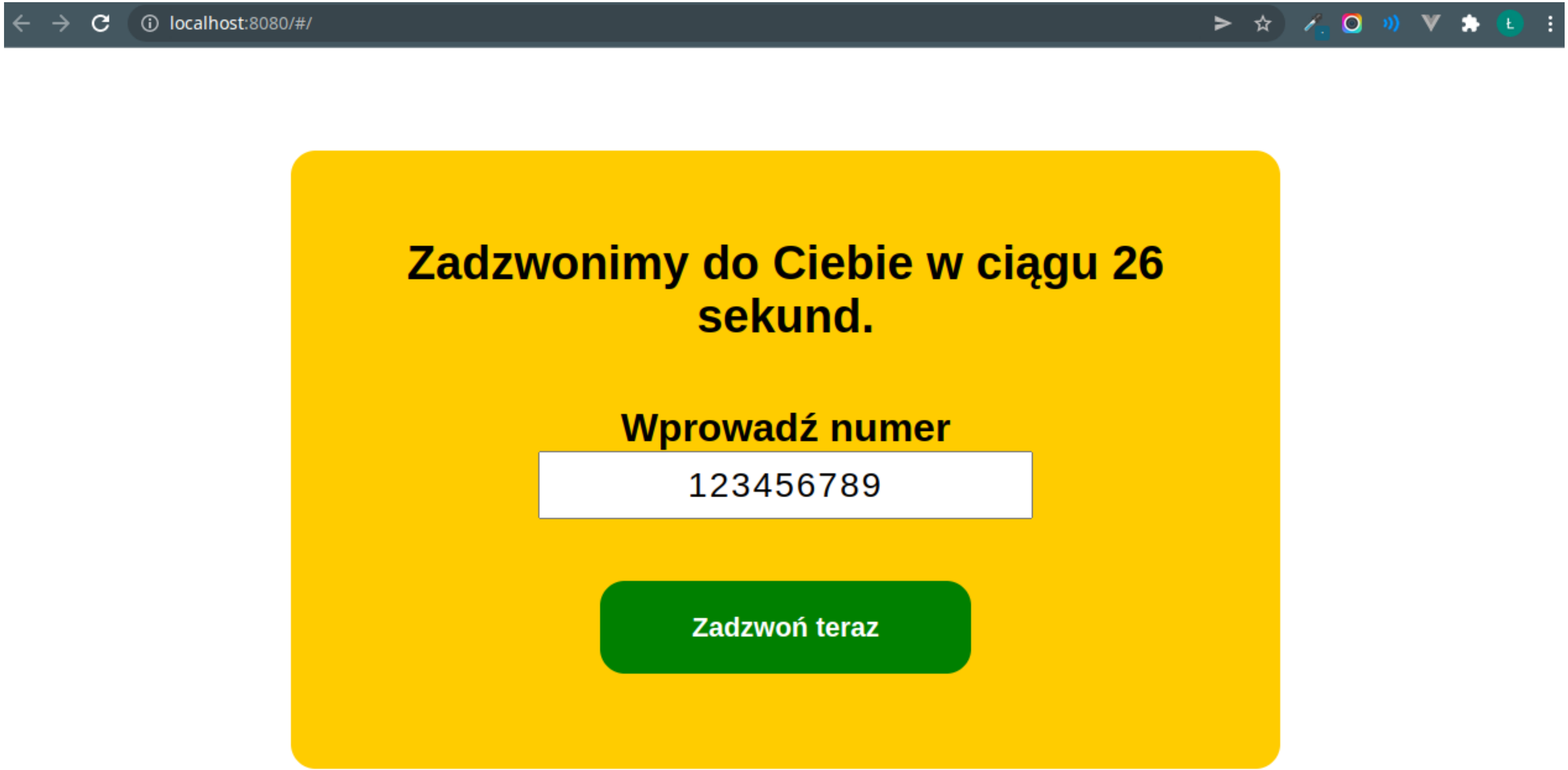
```
<template>
  <div>
    <div class="header clearfix">Zadzwonimy do Ciebie w ciągu 26 sekund.</div>
    <label class="form-label clearfix" for="form-number">
      Wprowadź numer
    </label>
    <input v-model="number" class="form-number clearfix" id="form-number" />
    <div class="call-button" @click="call()">Zadzwoń teraz</div>
  </div>
</template>
```

Ustawiamy numer na backendzie

```
app.post('/call/', async (req, res) => {
   const body = req.body;
   const number1 = body.number;
   const number2 = '<twój numer>';
   bridge = await Dialer.call(number1, number2);
   res.json({ success: true });
  })
```

Sprawdźmy działanie!

Podaj numer i wykonaj połączenie



Znane problemy

- Nie uruchomiony backend - node app.js w katalogu z backendem
- Nie podane dane autoryzacyjne w app.js na backendzie
- Serwer odpalony na innym porcie
- W logach backendu mam Ringing-Connected-Answered a telefon nie dzwoni - backend działa w trybie mock
- Błąd że cannot find id - numeru nie ma w naszej bazie - zgłoś problem prowadzącemu
- Miałeś inny problem? Daj znać i dopiszemy go do listy!

Tworzymy nowy widok RingingView

src/views/RingingView.vue

```
1  <template>
2    <div class="box">
3      <div class="text">Zaraz nastąpi połączenie z konsultantem.</div>
4    </div>
5  </template>
```

Dodajemy nową ścieżkę do routingu

src/router/index.ts

```
1   import { createRouter, createWebHashHistory, RouteRecordRaw } from 'vue-router'
2   import Start from '../views/StartView.vue'
3   import Ringing from '../views/RingingView.vue'
4
5   const routes: Array<RouteRecordRaw> = [
6     {
7       path: '/',
8       name: 'start',
9       component: Start,
10    },
11    {
12      path: '/',
13      name: 'ringing',
14      component: Ringing,
15    },
16  ]
17
18  const router = createRouter({...})
19
20  export default router
```

Rozszerzamy widok Start

```vue
<script lang="ts">
import { defineComponent } from 'vue'

export default defineComponent({
  data: () => ({
    number: '',
  }),
  methods: {
    async call() {
      const responseStream = await fetch('https://localhost:3000/call', {
        method: 'POST',
        headers: {
          'Content-type': 'application/json; charset=UTF-8',
        },
        body: JSON.stringify({ number: this.number }),
      })
      const response = await responseStream.json()
      this.$router.push({ name: 'ringing', params: { callsId: response.id } })
    },
  },
})
</script>
```

Co dalej?

- Pobieranie statusu oraz zmiana widoków (pooling/socket.io)
- Obsługa błędów - niepowodzenie w trakcie nawiązywania połączenia

Sprawdź też

- https://education.github.com/pack
- https://frontendmasters.com/learn/vue/
- https://www.vuemastery.com/courses
- https://router.vuejs.org/