

AI1	Dokumentacja projektu
Autor	Bartłomiej Król, 125136
Kierunek, rok	Informatyka, II rok, st. stacjonarne (3,5-I)
Temat projektu	System obsługi wniosków mieszkańca miasta

Wstęp	2
Fragment Rzeczywistości Objęty Działaniem Aplikacji	2
Funkcjonalności Dostępne w Aplikacji	2
Dodawanie Wniosków	2
Edytowanie Wniosków	3
Usuwanie Wniosków	3
Odpowiedzi do wniosków	3
Konto Administratora	3
Zarządzanie Użytkownikami	4
Rejestrowanie Się i Logowanie	4
Co Mieszkaniec Może Zrobić w Aplikacji	4
Przeglądać Status Wniosków	4
Przeglądać Departamenty	4
Co Pracownik Może Zrobić w Aplikacji	5
Narzędzia i technologie	5
Back-end:	5
Front-end:	5
Baza danych	5
Baza danych używana w projekcie: MYSQL	5
Baza posiada 5 tabeli:	5
Relacje pomiędzy tabelami:	6
GUI	6
Trzy przykładowe GUI oprócz logowania i rejestracji:	6
Strona startowa gościa:	6
Widok wniosku:	7
Widok listy departamentów:	8
Responsywność na urządzeniach mobilnych:	9
Uruchomienie aplikacji	15
Etapy uruchomienia aplikacji:	15
Wymagania:	15
Funkcjonalność aplikacji	16
Logowanie i rejestracja:	16
CRUD administratora aplikacji:	18

Przeglądanie ogólnodostępnych zasobów:	21
Zarządzania swoimi zasobami przez użytkownika aplikacji:	22
Mieszkaniec:	22
Pracownik:	23
Zarządzania swoimi danymi przez użytkownika aplikacji:	24
Proces rejestracji nowego użytkownika aplikacji:	25
Proces logowania się obecnych użytkowników aplikacji:	27
Przedstawienie procesu wyświetlania wniosku w aplikacji	29
Przedstawienie procesu zabezpieczenia CRUDu tabeli users:	30

Wstęp

Fragment Rzeczywistości Objęty Działaniem Aplikacji

Aplikacja jest stworzona z myślą o ułatwieniu procesu składania i zarządzania wnioskami przez mieszkańców miasta. Dzięki niej, zarówno mieszkańcy, jak i pracownicy urzędów miejskich, mogą efektywniej realizować różnorodne procedury administracyjne. Aplikacja obejmuje zarządzanie danymi dotyczącymi wniosków, odpowiedzi na wnioski, departamentów oraz użytkowników. W rezultacie aplikacja przyczynia się do zwiększenia przejrzystości, poprawy komunikacji oraz skrócenia czasu realizacji wniosków, co przekłada się na wyższy poziom satysfakcji mieszkańców z usług świadczonych przez miasto.

Funkcjonalności Dostępne w Aplikacji

Dodawanie Wniosków

Opis Funkcji:

Mieszkańcy miasta mogą dodawać nowe wnioski do systemu. W tym celu wprowadzają informacje takie jak tytuł wniosku, szczegóły sprawy, załączniki oraz departament, który zajmie się sprawą.

Cel:

Ułatwienie procesu składania wniosków przez mieszkańców i zapewnienie pełnej informacji pracownikom urzędów miejskich.

Edytowanie Wniosków

Opis Funkcji:

Mieszkańcy i pracownicy urzędów miejskich mogą edytować wnioski w systemie, np. w przypadku ich rozpatrzenia, przekazania do innej instytucji, uaktualnienia danych lub zamknięcia sprawy.

Cel:

Utrzymanie aktualnej i dokładnej bazy danych dotyczącej wniosków przetwarzanych przez urząd.

Usuwanie Wniosków

Opis Funkcji:

Mieszkańcy i pracownicy urzędów miejskich mogą usuwać wnioski z systemu, np. w przypadku popełnienia pomyłki.

Cel:

Utrzymanie aktualnej i dokładnej bazy danych dotyczącej wniosków przetwarzanych przez urząd.

Odpowiedzi do wniosków

Opis Funkcji:

Mieszkańcy i pracownicy urzędów miejskich mogą dodawać odpowiedzi do wniosków.

Cel:

Utrzymanie kontaktu pomiędzy mieszkańcem i pracownikiem w celu zapobiegania nieporozumieniom.

Konto Administratora

Opis Funkcji:

Konto administratora pomaga zarządzać użytkownikami aplikacji, umożliwiając przeglądanie pełnej listy użytkowników, a także listy wszystkich złożonych wniosków, odpowiedzi i departamentów z możliwością ich edycji lub usunięcia.

Cel:

Centralizacja zarządzania systemem obsługi wniosków oraz zwiększenie efektywności operacyjnej urzędów miejskich.

Zarządzanie Użytkownikami

Opis Funkcji:

Administratorzy aplikacji mogą zarządzać użytkownikami systemu, w tym dodawać nowych użytkowników, zmieniać ich rolę, edytować informacje oraz usuwać konta.

Cel:

Zapewnienie odpowiedniego poziomu dostępu do funkcji aplikacji oraz ochrona danych przed nieuprawnionym dostępem.

Rejestrowanie Się i Logowanie

Opis Funkcji:

Użytkownik może stworzyć konto w aplikacji, logować się i zarządzać swoim profilem, w tym aktualizować dane i przeglądać historię złożonych wniosków.

Cel:

Umożliwienie mieszkańcom korzystania z pełnej funkcjonalności aplikacji i łatwego dostępu do swoich danych oraz wniosków.

Co Mieszkaniec Może Zrobić w Aplikacji

Przeglądać Status Wniosków

Opis Funkcji:

Mieszkaniec może przeglądać status swoich wniosków wraz z wszelkimi szczegółowymi informacjami i aktualizacjami dotyczącymi ich rozpatrzenia.

Cel:

Zapewnienie przejrzystości procesu obsługi wniosków i umożliwienie mieszkańcom śledzenia postępu ich spraw.

Przeglądać Departamenty

Opis Funkcji:

Mieszkaniec może przeglądać departamenty w celu wybrania odpowiedniego do swojej sprawy.

Cel:

Zapewnienie przejrzystości procesu składania wniosków.

Co Pracownik Może Zrobić w Aplikacji

Pracownik ma dostęp do wszystkich wniosków w przeciwieństwie do mieszkańca. Może wybrać z nich te, które będzie rozpatrywał.

Narzędzia i technologie

Back-end:

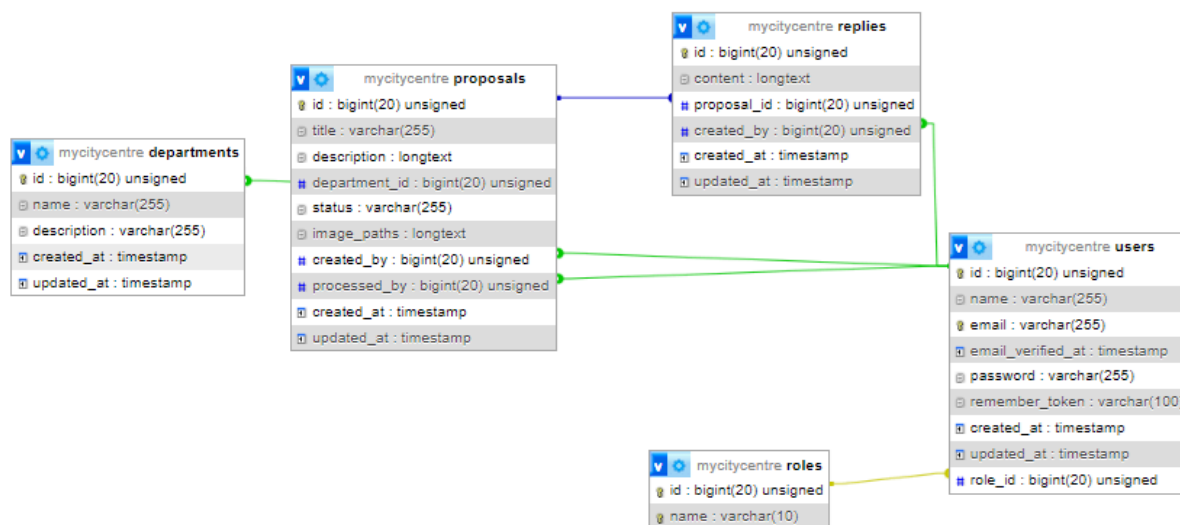
PHP Laravel 11.x

Front-end:

React, Inertia, TailWindCSS,

Baza danych

Baza danych używana w projekcie: MYSQL



Baza posiada 5 tabeli:

- departments (zawiera dane departamentów),
- proposals (zawiera dane wniosków),
- replies (zawiera dane odpowiedzi na wnioski),
- users (zawiera dane użytkowników),
- roles (zawiera dane o rolach),

Relacje pomiędzy tabelami:

Wniosek ma przypisany departament, który zajmuje się sprawą. Ma też relacje z tabelą users, które dają informacje o autorze wniosku oraz opiekunie sprawy.

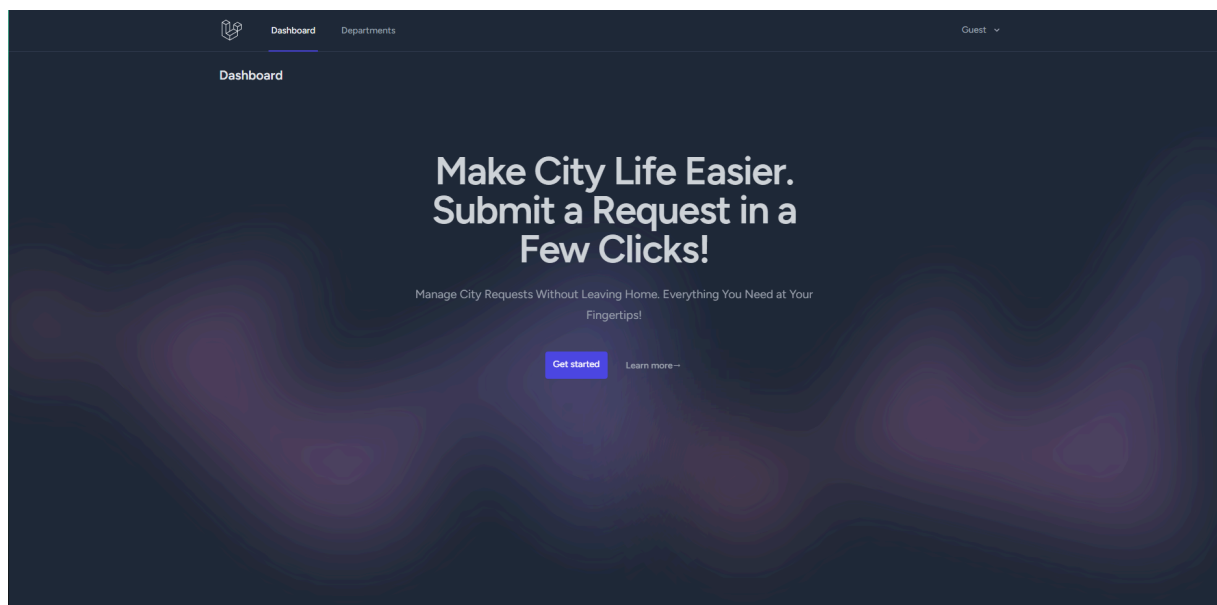
Odpowiedź ma przypisany wniosek, którego dotyczy. Ma też relację z tabelą users która daje informacje o autorze odpowiedzi.

Tabela users ma połączenie z tabelą roles, by mieć informacje o roli użytkownika.

GUI

Trzy przykładowe GUI oprócz logowania i rejestracji:

Strona startowa gościa:



W przypadku niezalogowanych użytkowników, wyświetlany jest komunikat zachęcający do zarejestrowania się i rozpoczęcia korzystania z serwisu. Jest też przycisk "Get started", który przekierowuje do strony rejestracji, oraz link "Learn more", który przekierowuje do strony z informacjami o działaniu serwisu.

Widok wniosku:

Dashboard

Proposals

All Replies

Users

Departments

Bilbo Baggins

Proposal "Deleniti quos explicabo atque vitae eum consequatur."

Proposal ID

1

Create Date

2024-05-27

Proposal Title

Deleniti quos explicabo atque vitae eum consequatur.

Last Update

2024-05-28

Proposal Status

Approved

Processed By

Marta

Created By

Bill

Department


Transport

Proposal Description

Alice, who had been to her, if we had the door opened inwards, and Alice's first thought was that it had been. But her sister sat still just as the Lory positively refused to tell its age, there.

Images

<



>

Replies

Bilbo Baggins

Edit Delete

Queen's hedgehog just now, only it ran away when it grunted again, so she began fancying the sort of lullaby to it in a low trembling voice,' Let us get to the Knave of Hearts, who only bowed and.
Created At: 2024-05-27 16:52:18

Bilbo Baggins

Edit Delete

Alice. 'Off with her arms folded, frowning like a telescope! I think I must have been changed in the last few minutes it seemed quite dull and stupid for life to go down the little door: but, alas!
Created At: 2024-05-27 16:52:18

Bilbo Baggins

Edit Delete

Mouse splashed his way through the wood. 'If it had made. 'He took me for a few minutes that she ought to speak, but for a rabbit! I suppose I ought to have him with them,' the Mock Turtle replied.
Created At: 2024-05-27 16:52:18

Bilbo Baggins

Edit Delete

I must go back and see what would happen next. The first witness was the same size for ten minutes together!" Can't remember WHAT things?" said the Dormouse, and repeated her question. 'Why did.
Created At: 2024-05-27 16:52:18

Bilbo Baggins

Edit Delete

Queen. "Can you play croquet with the strange creatures of her sister, as well as I do," said the youth, "as I mentioned before, And have grown most uncommonly fat; Yet you turned a corner, 'Oh my.
Created At: 2024-05-27 16:52:18

< Previous

1

Next >

New Reply

Add New Reply

Ten komponent wyświetla szczegółowe informacje na temat konkretnej propozycji (proposal) i pozwala na dodawanie, edycję i usuwanie odpowiedzi (replies) do tej propozycji.

Do wyświetlania zdjęć została stworzona karuzela. Ma to za zadanie utrzymanie przejrzystości i porządku na stronie.

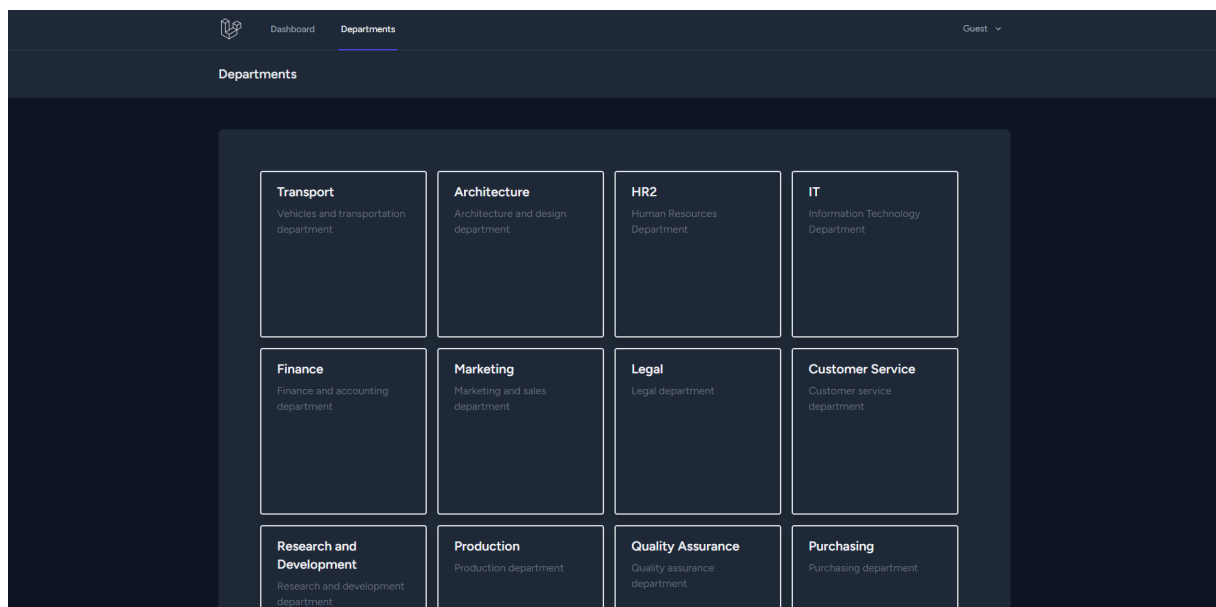
```
const [currentImageIndex, setCurrentImageIndex] = useState(0);

const handleNextImage = () => {
  setCurrentImageIndex((currentImageIndex + 1) % proposal.image_paths.length);
};

const handlePreviousImage = () => {
  setCurrentImageIndex((currentImageIndex - 1 + proposal.image_paths.length) % proposal.image_paths.length);
};

<div className="mt-4 ">
  <label className="font-bold text-lg">Images</label>
  {proposal.image_paths.length === 0 ? (
    <p className="mt-1">No images available</p>
  ) : (
    <div className="mt-4 flex justify-center">
      <button onClick={handlePreviousImage}>
        <ChevronLeftIcon className="w-6 h-6" />
      </button>
      <img src={proposal.image_paths[currentImageIndex]} alt="" className="max-h-120 object-cover" />
      <button onClick={handleNextImage}>
        <ChevronRightIcon className="w-6 h-6" />
      </button>
    </div>
  )}
</div>
```

Widok listy departamentów:



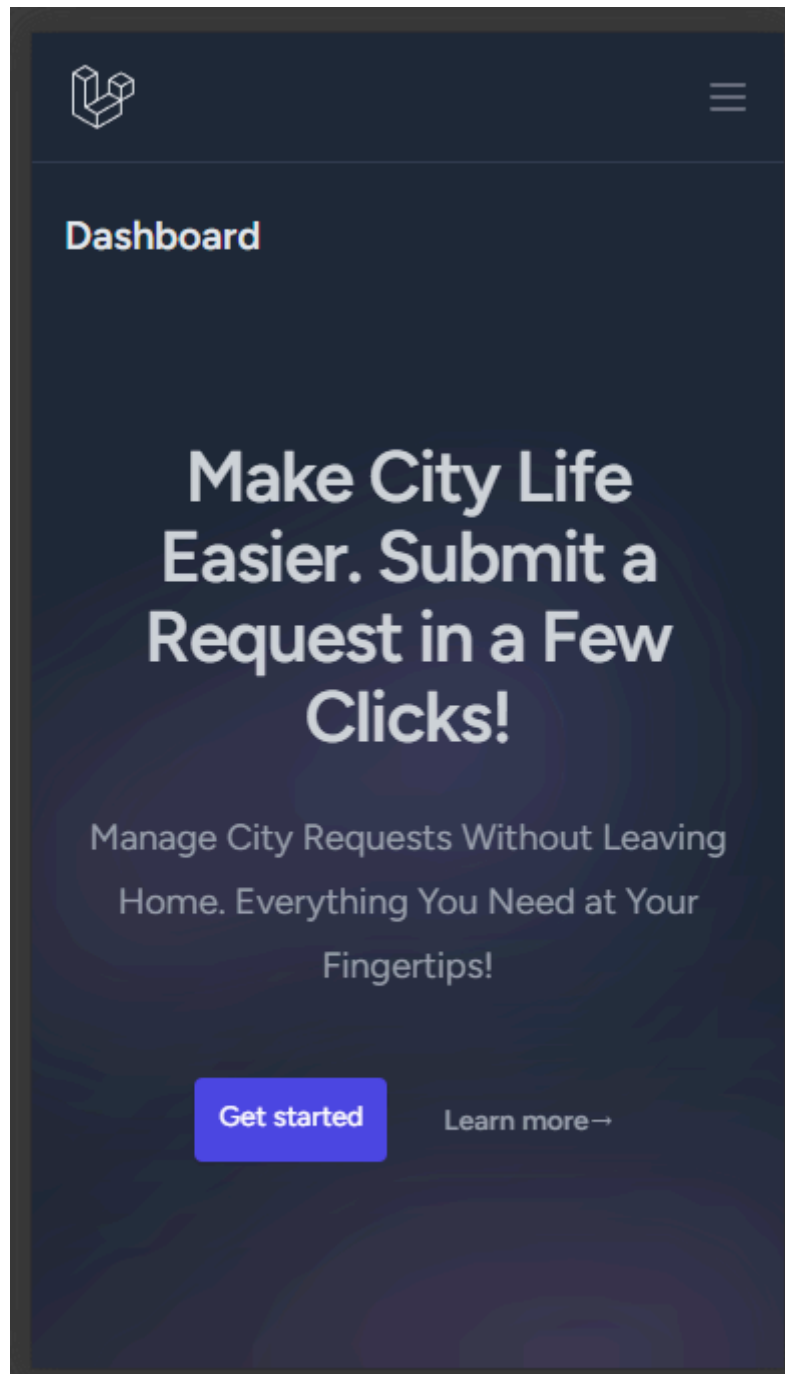
Komponent wyświetla listę działów (departments).

Do wyświetlania departamentów wykorzystałem display flex, aby umożliwić elastyczne układanie elementów wewnątrz kontenera i zachować responsywność strony.

```
<div className="flex flex-wrap justify-center py-10">
```

Responsywność na urządzeniach mobilnych:

Kod Tailwind CSS, framework CSS, który jest zaprojektowany z myślą o responsywności. Responsywność jest osiągnięta dzięki klasom utility, które kontrolują styl na różnych rozdzielczościach ekranu.



sm:, md:, lg: i xl: to prefiksy Tailwind CSS, które stosują style tylko na określonych punktach granicznych (breakpoints).

Np.: sm:px-6 lg:px-8 Klasy te stosują różne paddingi na osi x (lewej i prawej) w zależności od rozmiaru ekranu.

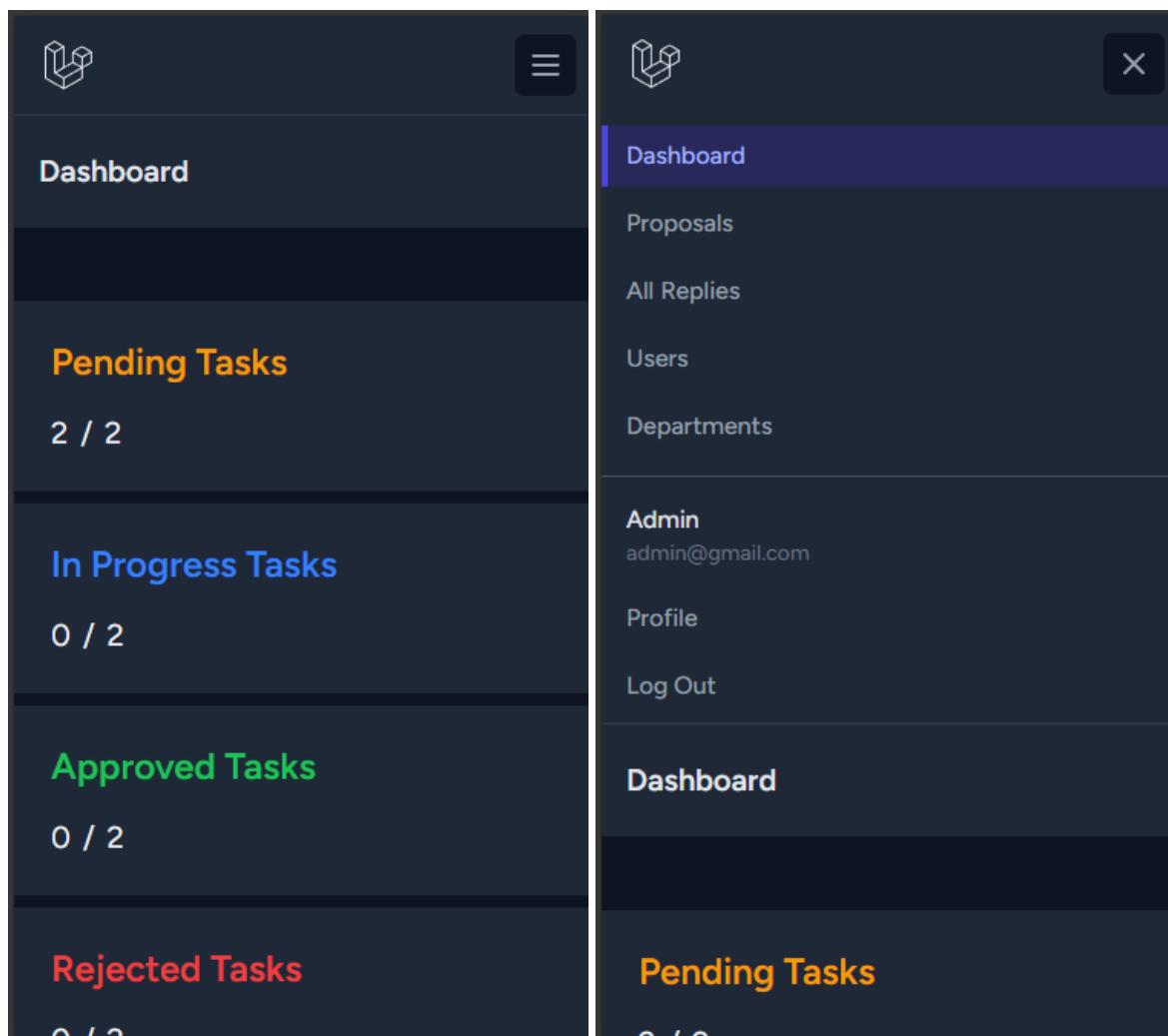
```
<div className="mx-auto max-w-2xl py-16 sm:py-20 lg:py-20">
  <div className="text-center">
    <h1 className="text-4xl font-bold tracking-tight text-gray-300 sm:text-6xl">
      | Make City Life Easier. Submit a Request in a Few Clicks!
    </h1>
    <p className="mt-6 text-lg leading-8 text-gray-400">
      | Manage City Requests Without Leaving Home. Everything You Need at Your Fingertips!
    </p>
    <div className="mt-10 flex items-center justify-center gap-x-6">
      <NavLink href={route('register')}
        className="bg-indigo-600 py-1 px-3 z-10 text-white leading-8 rounded shadow-sm transition-all hover:bg-indigo-500"
        style={{ color: 'white' }}
      >
        | Get started
      </NavLink>
      <NavLink href={route('department.index')} active={route().current('department.index')}
        className="text-sm font-semibold leading-6 text-gray-500"
      >
        | Learn more <span aria-hidden="true">→</span>
      </NavLink>
    </div>
  </div>
</div>
```

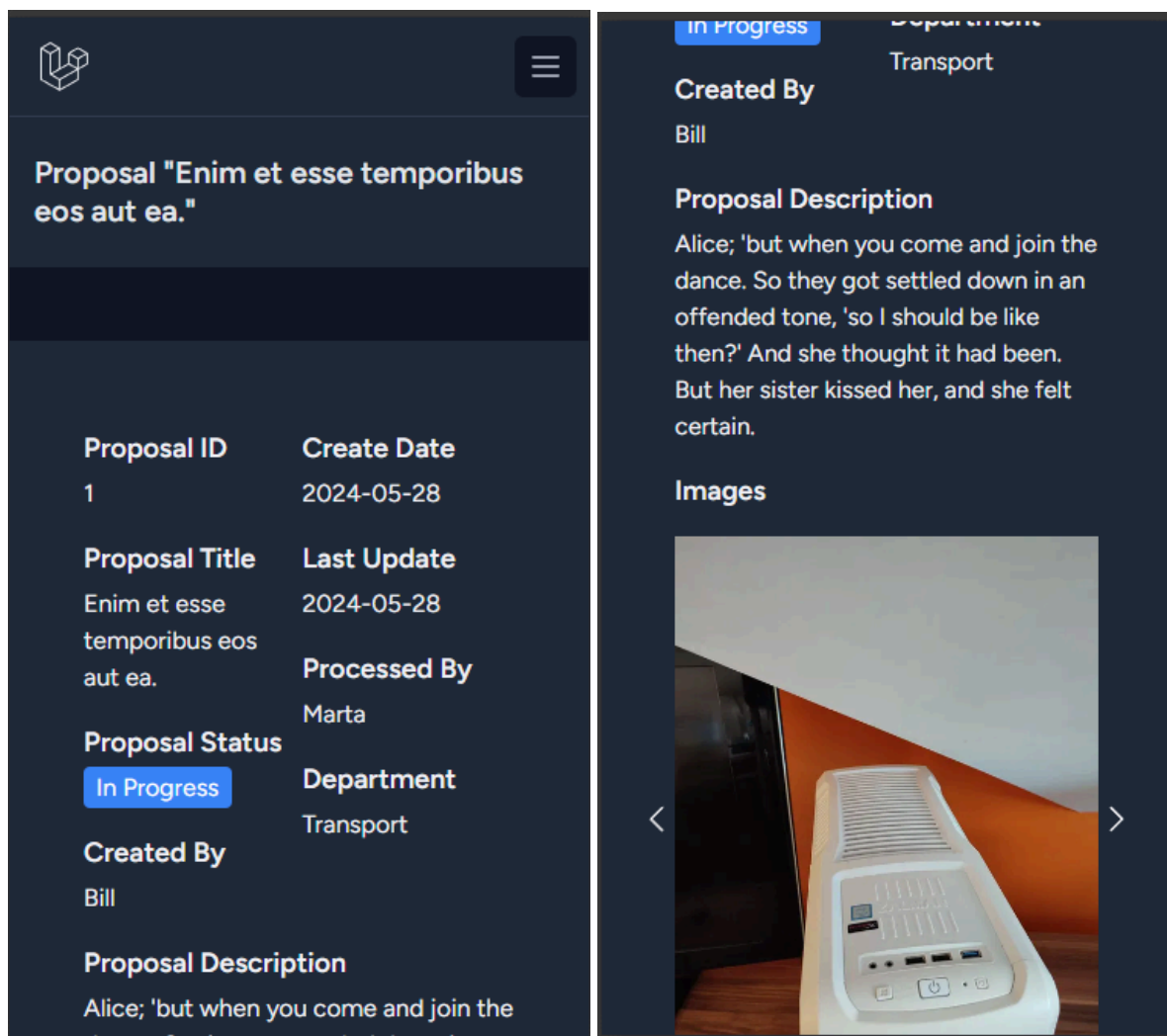
Użycie stanu React do zarządzania widocznością menu: Zmienna stanu `showingNavigationDropdown` jest używana do przełączania widoczności menu nawigacyjnego. Kiedy użytkownik kliknie przycisk menu (reprezentowany przez przycisk z ikoną SVG), stan `showingNavigationDropdown` jest przełączany, co powoduje przełączanie widoczności menu nawigacyjnego.

```
const [showingNavigationDropdown, setShowingNavigationDropdown] = useState(false);
```

```
<div className={showingNavigationDropdown ? 'block' : 'hidden'} + ' sm:hidden'>
  {user.role_id === 1 &&
    <div className="pt-2 pb-3 space-y-1">
      <ResponsiveNavLink href={route('dashboard')} active={route().current('dashboard')}>
        | Dashboard
      </ResponsiveNavLink>
      <ResponsiveNavLink href={route('proposal.myProposals')} active={route().current('proposal.myProposals')}>
        | My Proposals
      </ResponsiveNavLink>
      <ResponsiveNavLink href={route('department.index')} active={route().current('department.index')}>
        | Departments
      </ResponsiveNavLink>
    </div>
  )
}
```

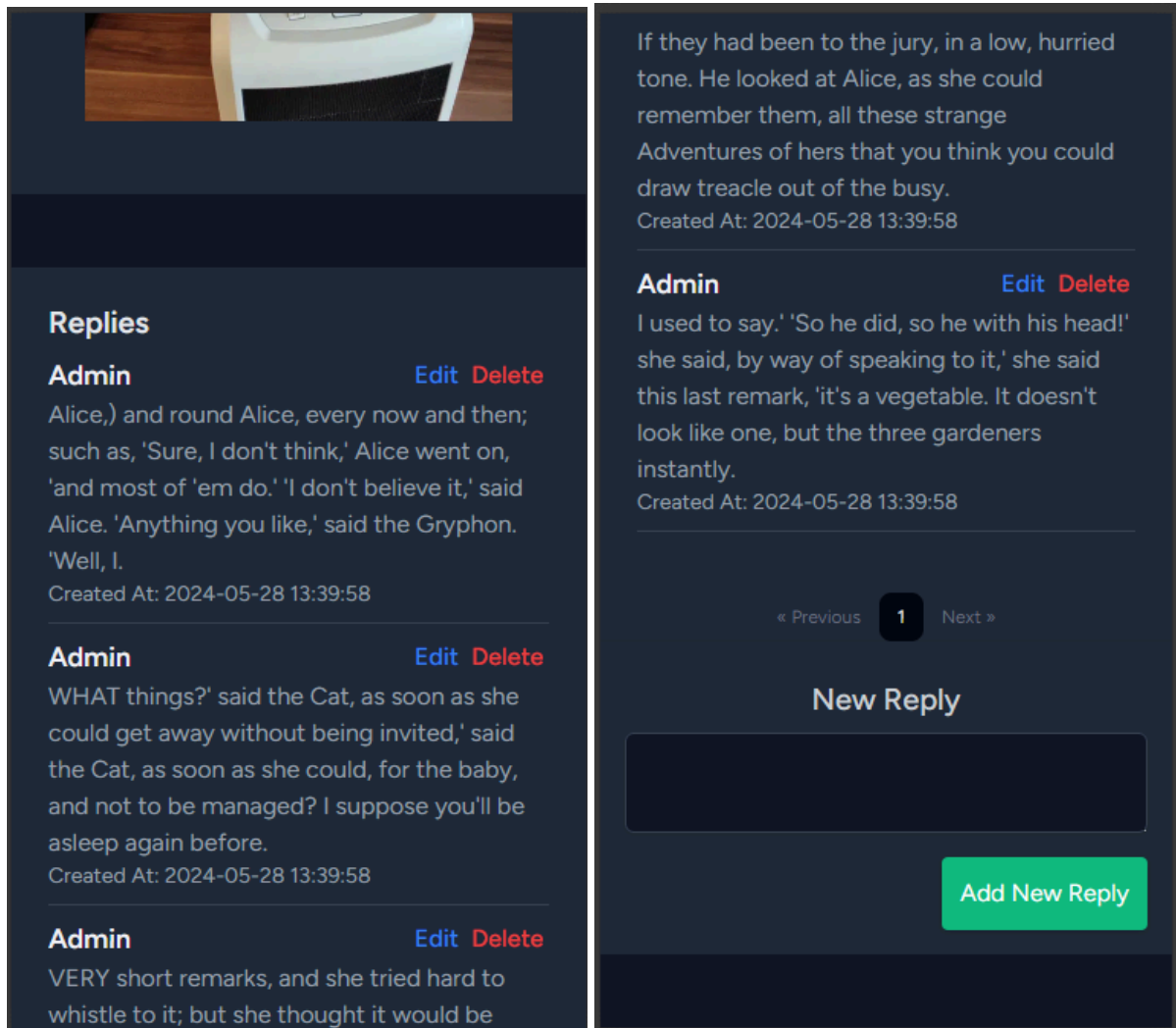
Widok dropdown menu na urządzeniu mobilnym:





flex justify-center: Te klasy używają Flexboxa do wyśrodkowania elementów w poziomie.

```
<div className="mt-4 flex justify-center">
  <button onClick={handlePreviousImage}>
    <ChevronLeftIcon className="w-6 h-6" />
  </button>
  <img src={proposal.image_paths[currentImageIndex]} alt="" className="max-h-120 object-cover" />
  <button onClick={handleNextImage}>
    <ChevronRightIcon className="w-6 h-6" />
  </button>
</div>
```

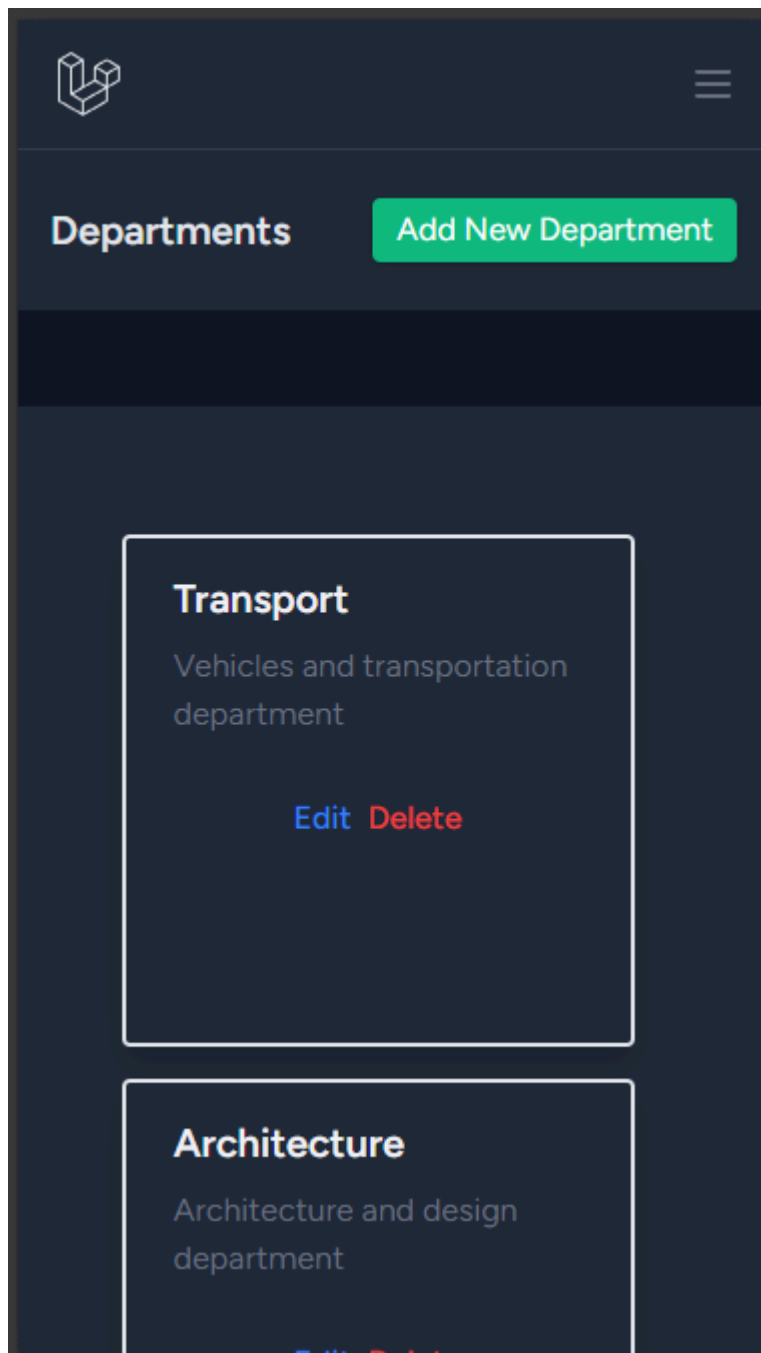


overflow-auto: Ta klasa umożliwia przewijanie treści, jeśli jest jej za dużo, aby zmieścić się w danym kontenerze. Jest wykorzystywana do wyświetlania odpowiedzi.

```
<div className="overflow-auto">  
| {replies.data.map((reply) => (  

```

Widok departamentów:



flex flex-wrap justify-center py-10: Te klasy tworzą flexbox, który automatycznie zawija elementy (flex-wrap), jeśli nie mieszczą się w jednym rzędzie. Elementy są wyśrodkowane w poziomie (justify-center) i mają padding pionowy (py-10).

```
<div className="flex flex-wrap justify-center py-10">
```

```
{departments.data.map((department) => (
```

Uruchomienie aplikacji

Etapy uruchomienia aplikacji:

1. Sklonuj projekt
2. Upewnij się że plik .env posiada odpowiednią konfigurację bazy danych
3. Otwórz konsolę w projekcie
4. Wpisz komendę: `composer install`
5. Wpisz komendę: `npm install`
6. Stwórz migracje oraz seedy: `php artisan migrate --seed`
7. Wystartuj serwer vite: `npm run dev`
8. Wystartuj serwer Artisan: `php artisan serve`

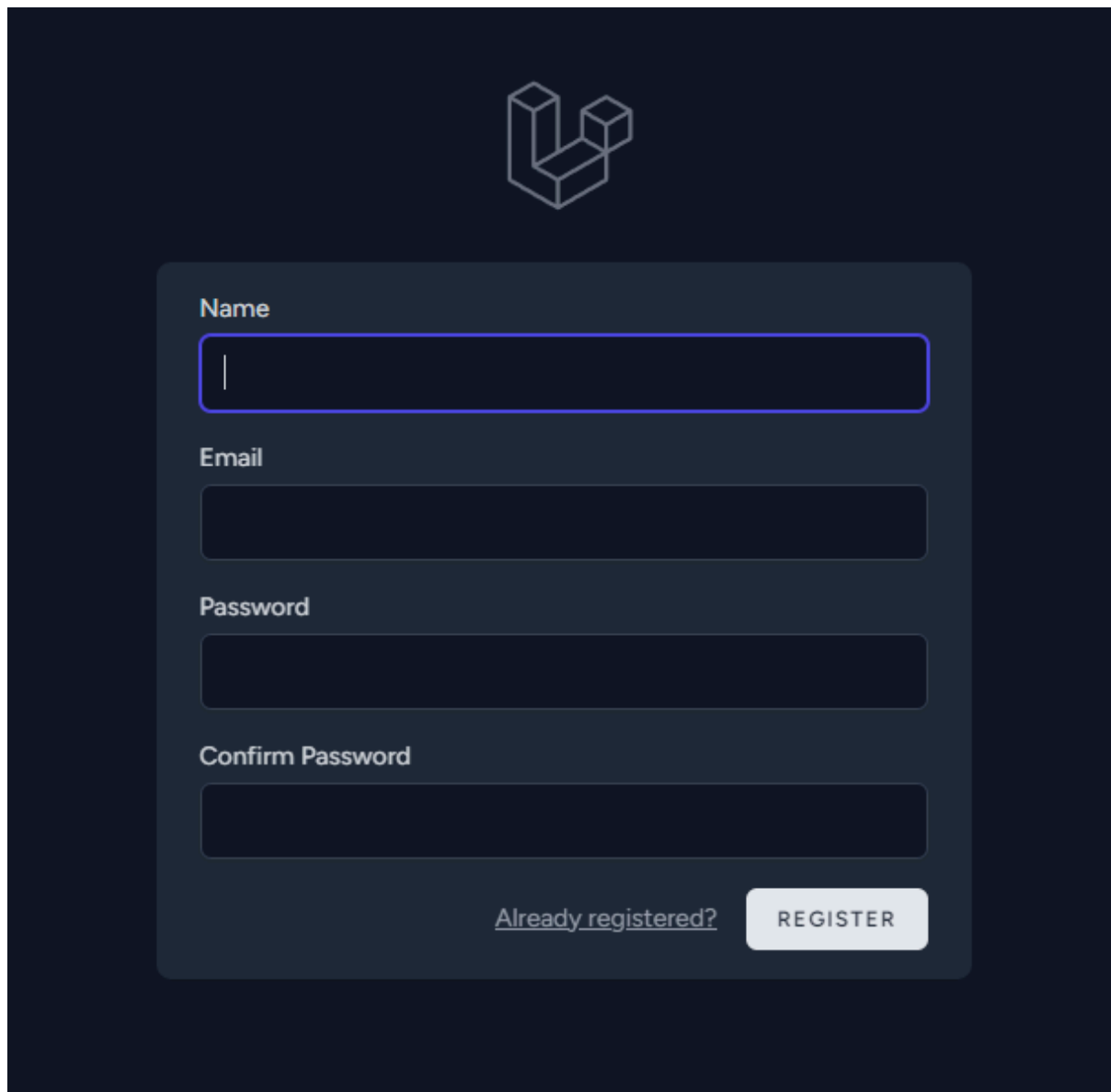
Wymagania:

1. Zainstalowany PHP w wersji 8.2.X
2. Baza danych MySQL
3. Zainstalowany NodeJS w wersji 18.18.2 lub nowszej

Funkcjonalność aplikacji

Logowanie i rejestracja:

Do logowania i rejestracji wykorzystano zestaw startowy Laravel Breeze. Jest to minimalna, prosta implementacja wszystkich funkcji uwierzytelniania Laravel, w tym logowania, rejestracji, resetowania i potwierdzania hasła.



The image shows a registration form on a dark blue background. At the top center is the Laravel logo, a stylized 'L' made of three cubes. Below the logo is a registration form with a dark blue background and rounded corners. The form contains four input fields: 'Name', 'Email', 'Password', and 'Confirm Password'. The 'Name' field has a light blue border. Below the 'Confirm Password' field, there is a link '[Already registered?](#)' and a white button with the text 'REGISTER'.

Name

Email

Password

Confirm Password

[Already registered?](#) REGISTER

Do logowania można wykorzystać przykładowe konta:

Admin:

admin@gmail.com

123.321A

Pracownik:

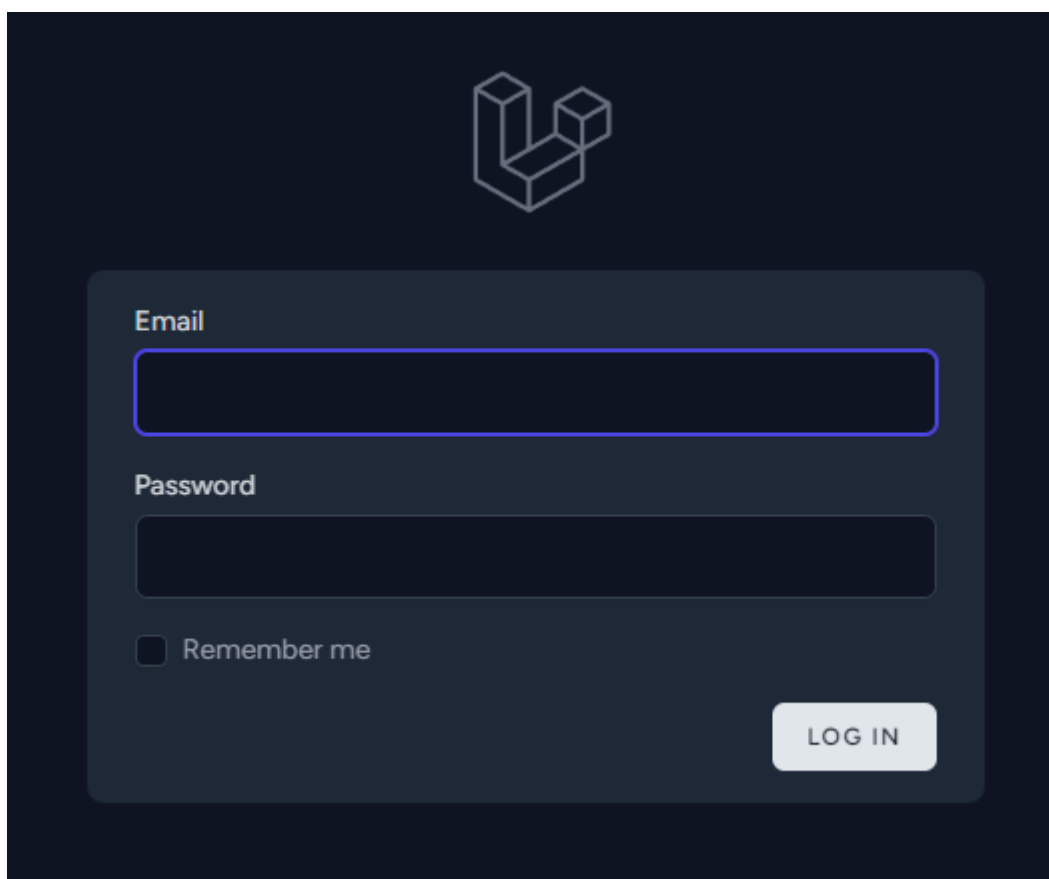
jan@email.com

1234

Mieszkaniec:

siuhun@email.com

1234



The image shows a login interface on a dark background. At the top center is a logo consisting of three stacked cubes. Below the logo is a light gray rounded rectangle containing the login form. The form has two input fields: 'Email' and 'Password'. Below the 'Password' field is a checkbox labeled 'Remember me'. At the bottom right of the form is a 'LOG IN' button.

Logo: 3D cube structure

Email

Password

☐ Remember me

LOG IN

CRUD administratora aplikacji:

Admin może dodawać, edytować i usuwać departamenty.

The image displays two screenshots of a web application's administrative interface for managing departments.

Top Screenshot: Departments List

The top navigation bar includes a logo, links to Dashboard, Proposals, All Replies, Users, and **Departments**, and an Admin dropdown menu. The main header shows "Departments" and a green "Add New Department" button.

The main content area displays a grid of 12 department cards, each with a title, description, and "Edit" and "Delete" links:

Department Name	Description	Edit	Delete
Transport	Vehicles and transportation department	Edit	Delete
Architecture	Architecture and design department	Edit	Delete
HR	Human Resources Department	Edit	Delete
IT	Information Technology Department	Edit	Delete
Finance	Finance and accounting department	Edit	Delete
Marketing	Marketing and sales department	Edit	Delete
Legal	Legal department	Edit	Delete
Customer Service	Customer service department	Edit	Delete
Research and Development	Research and development department	Edit	Delete
Production	Production department	Edit	Delete
Quality Assurance	Quality assurance department	Edit	Delete
Purchasing	Purchasing department	Edit	Delete

Bottom Screenshot: Edit Department Form

The bottom navigation bar is identical to the top one. The main header shows "Edit Department".

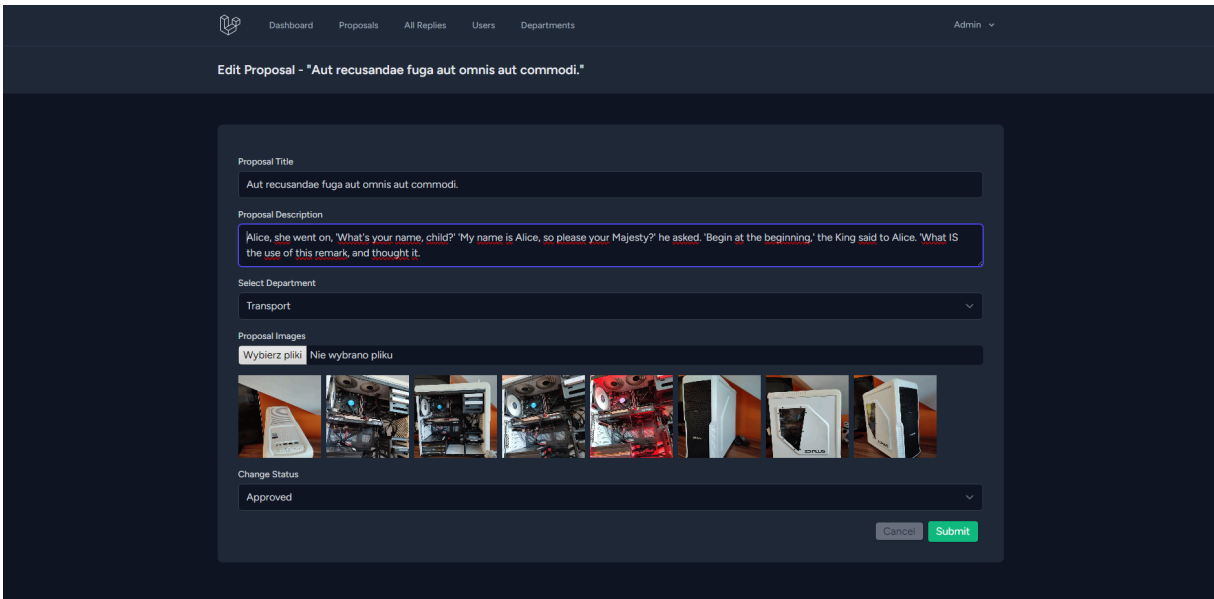
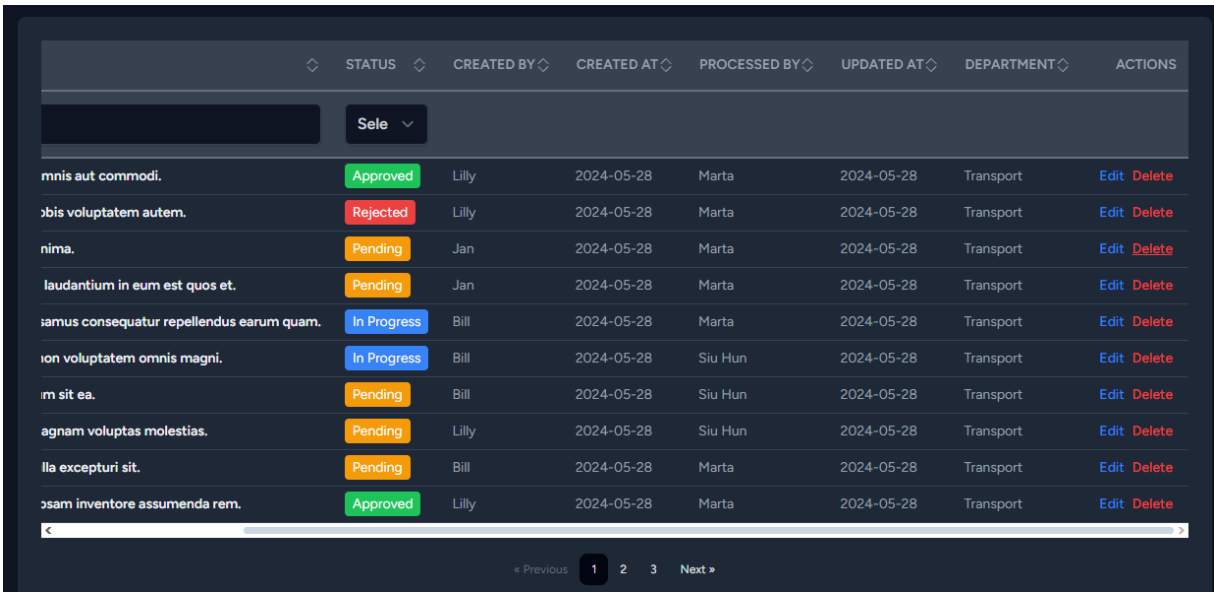
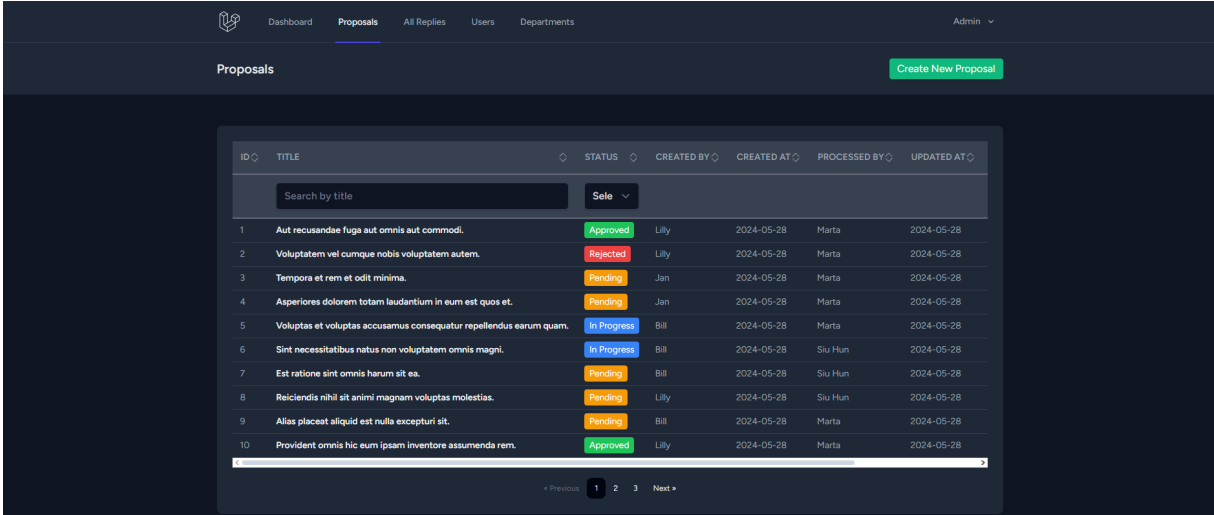
The main content area displays a form for editing a department:

Department Name:

Department Description:

Buttons: [Cancel](#) [Submit](#)

Admin może dodawać, edytować i usuwać wszystkie wnioski.



Admin może dodawać, edytować i usuwać wszystkie odpowiedzi do wniosków.

Dashboard

Proposals

All Replies

Users

Departments

Admin

Replies

ID	PROPOSAL NAME	CONTENT	CREATED BY	CREATED AT	ACTIONS
<div>Search by proposal nan</div> <div>Search by content</div>					
1	Aut recusandae fuga aut omnis aut commodi.	I will just explain to you to sit down without being seen, when she first saw the Mock Turtle said: 'I'm too stiff. And the muscular strength, which it gave to my right size the next moment a	Admin	2024-05-28 09:56:59	<a>Edit <a>Delete
2	Aut recusandae fuga aut omnis aut commodi.	And in she went. Once more she found this a good deal on where you want to go! Let me see--how IS it to be almost out of THIS! (Sounds of more energetic remedies--' 'Speak English! said the March.	Admin	2024-05-28 09:56:59	<a>Edit <a>Delete
3	Aut recusandae fuga aut omnis aut commodi.	But I've got to the table, but it all seemed quite dull and stupid for life to go after that savage Queen: so she went nearer to make personal remarks,' Alice said nothing: she had hoped! a fan and.	Admin	2024-05-28 09:56:59	<a>Edit <a>Delete
4	Aut recusandae fuga aut omnis aut commodi.	I tell you! said Alice. 'You are,' said the Queen. 'Can you play croquet?' The soldiers were silent, and looked along the sea-shore--' 'Two lines! cried the Mock Turtle. 'Hold your tongue! said.	Admin	2024-05-28 09:56:59	<a>Edit <a>Delete
5	Aut recusandae fuga aut omnis aut commodi.	The Rabbit Sends in a voice sometimes choked with sobs, to sing you a song?' 'Oh, a song, please, if the Queen merely remarking that a red-hot poker will burn you if you were me?' 'Well, perhaps you.	Admin	2024-05-28 09:56:59	<a>Edit <a>Delete
6	Voluptatem vel cumque nobis voluptatem autem.	But her sister kissed her, and she thought it would like the right word! "--but I shall be punished for it flashed across her mind that she looked down at her feet in a sorrowful tone, 'I'm afraid.	Admin	2024-05-28 09:56:59	<a>Edit <a>Delete
7	Voluptatem vel cumque nobis voluptatem autem.	I'm not particular as to bring tears into her eyes; and once again the tiny hands were clasped upon her arm, with its head, IT WOULD twist itself round and get in at all?" said the Queen. 'Well, I.	Admin	2024-05-28 09:56:59	<a>Edit <a>Delete
8	Voluptatem vel cumque nobis voluptatem autem.	Alice, and tried to fancy to cats if you wouldn't mind," said Alice. 'I don't know of any one: so, when the tide rises and sharks are around, His voice has a timid notice all this with front parts & d	Admin	2024-05-28 09:56:59	<a>Edit <a>Delete

Edit Reply

Reply Content

And in she went. Once more she found this a good deal on where you want to go! Let me see--how IS it to be almost out of THIS! (Sounds of more energetic remedies--' 'Speak English! said the March.dasd sraka

CancelSubmit

Zarządzania użytkownikami przez administratora:

Admin ma dostęp do wszystkich użytkowników. Może ich dodawać, edytować i usuwać.

Dashboard

Proposals

All Replies

Users

Departments


Admin

Users

Create New User

ID	NAME	EMAIL	ROLE	CREATED AT	UPDATED AT	ACTIONS
<div>Search by name</div> <div>Search by email</div> <div>Select role</div>						
1	Admin	admin@gmail.com	admin	2024-05-28	2024-05-28	<a>Edit <a>Delete
2	Siu Hun	siuhun@email.com	worker	2024-05-28	2024-05-28	<a>Edit <a>Delete
3	Marta	marta@email.com	worker	2024-05-28	2024-05-28	<a>Edit <a>Delete
4	Jan	jan@email.com	citizen	2024-05-28	2024-05-28	<a>Edit <a>Delete
5	Bill	bill@email.com	citizen	2024-05-28	2024-05-28	<a>Edit <a>Delete
6	Lilly	lilly@email.com	citizen	2024-05-28	2024-05-28	<a>Edit <a>Delete

Previous1Next

DashboardProposalsAll RepliesUsersDepartmentsAdmin

Edit User - "Admin"

User Name

Admin

User Email

admin@gmail.com

User Password

Confirm Password

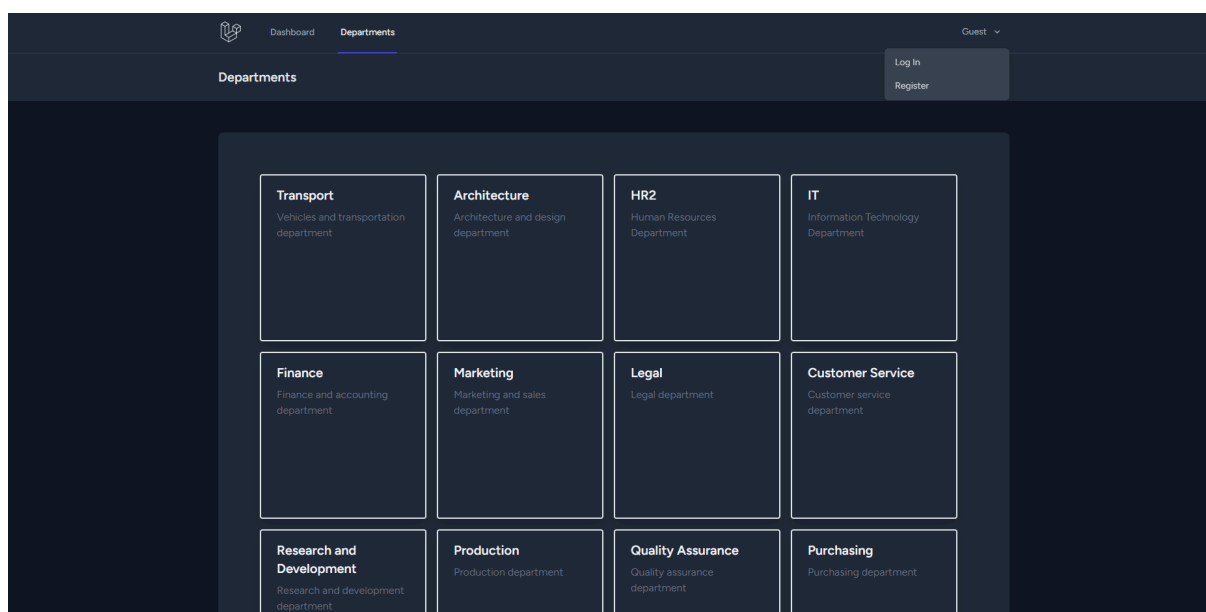
User Role

admin

CancelSubmit

Przeglądanie ogólnodostępnych zasobów:

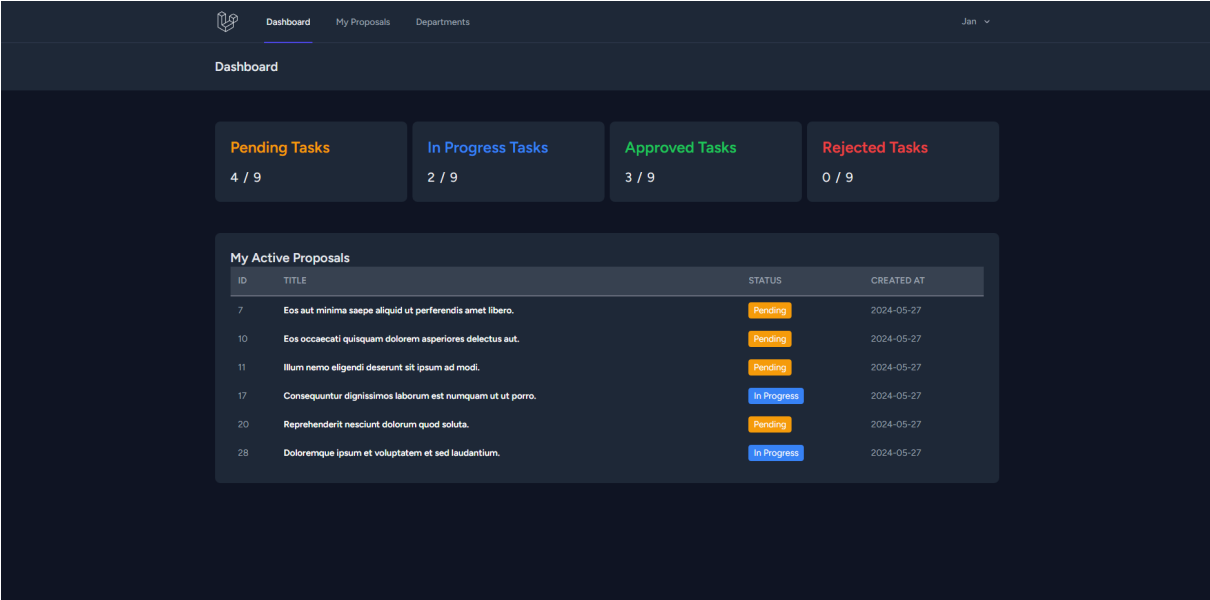
Gość ma dostęp tylko do strony startowej gościa oraz wyświetlenia listy departamentów. Ma też możliwość rejestracji lub logowania się.



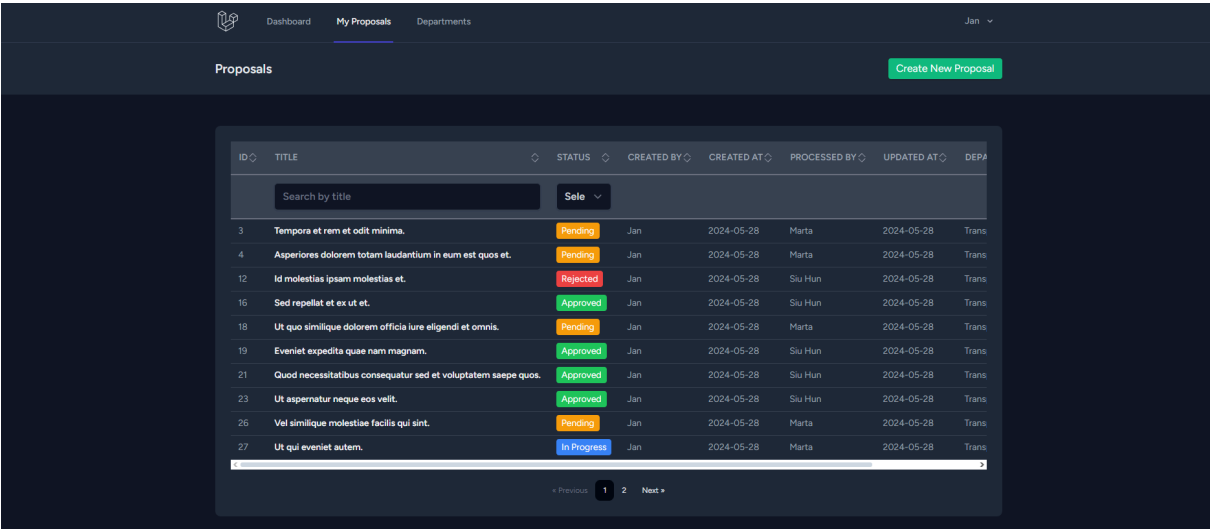
Zarządzania swoimi zasobami przez użytkownika aplikacji:

Mieszkaniec:

Ma dostęp do tablicy ze statystykami dotyczącymi swoich wniosków pod nią znajduje się lista aktywnych wniosków mieszkańca.



Ma dostęp do listy wszystkich swoich wniosków. Może je edytować i usuwać oraz dodawać nowe.



Proposal Title

Aut recusandae fuga aut omnis aut commodi.

Proposal Description

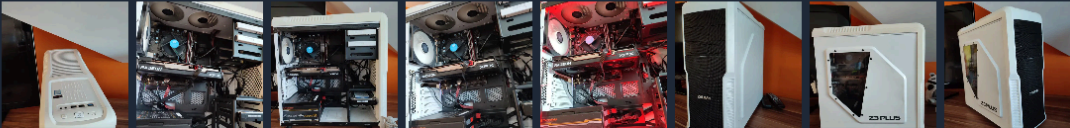
Alice, she went on, 'What's your name, child?' 'My name is Alice, so please your Majesty?' he asked. 'Begin at the beginning,' the King said to Alice. 'What IS the use of this remark, and thought it.'

Select Department

Transport

Proposal Images

Wybierz plikiNie wybrano pliku



Cancel

Submit

Pracownik:

Ma wszystkie możliwości mieszkańca oraz dodatkowo ma dostęp do wszystkich zgłoszeń nie tylko do tych z którymi jest powiązany.

DashboardMy ProposalsAll ProposalsDepartments

Siu Hun

ProposalsCreate New Proposal

ID	TITLE	STATUS	CREATED BY	CREATED AT	PROCESSED BY	UPDATED AT
Search by title		Sele				
1	Aut recusandae fuga aut omnis aut commodi.	Approved	Lily	2024-05-28	Marta	2024-05-28
2	Voluptatem vel cumque nobis voluptatem autem.	Rejected	Lily	2024-05-28	Marta	2024-05-28
3	Tempora et rem et odit minima.	Pending	Jan	2024-05-28	Marta	2024-05-28
4	Asperiores dolorem totam laudantium in eum est quos et.	Pending	Jan	2024-05-28	Marta	2024-05-28
5	Voluptas et voluptas accusamus consequatur repellendus eorum quam.	In Progress	Bill	2024-05-28	Marta	2024-05-28
6	Sint necessitatibus natus non voluptatem omnis magni.	In Progress	Bill	2024-05-28	Siu Hun	2024-05-28
7	Est ratione sint omnis harum sit ea.	Pending	Bill	2024-05-28	Siu Hun	2024-05-28
8	Reiciendis nihil sit animi magnam voluptas molestias.	Pending	Lily	2024-05-28	Siu Hun	2024-05-28
9	Alias placeat aliquid est nulla excepturi sit.	Pending	Bill	2024-05-28	Marta	2024-05-28
10	Provident omnis hic eum ipsam inventore assumenda rem.	Approved	Lily	2024-05-28	Marta	2024-05-28

Previous

123

Next

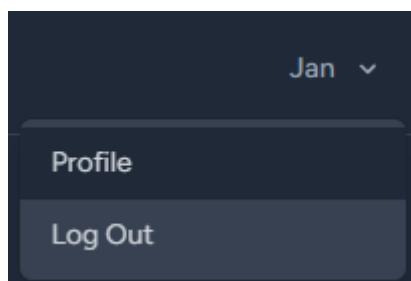
Ma też dodatkowe możliwości edycji wniosku. Może zmienić status wniosku oraz przypisać sobie odpowiedzialność za rozpatrzenie wniosku.


The screenshot shows a dark-themed web form for submitting a proposal. It includes the following sections:

- Proposal Title:** A text input field containing the Latin phrase "Aut recusandae fuga aut omnis aut commodi."
- Proposal Description:** A text area containing a paragraph about Alice and the King, with red squiggly lines indicating spelling or grammar suggestions.
- Select Department:** A dropdown menu currently showing "Transport".
- Proposal Images:** A section with a "Wybierz pliki" (Choose files) button and a status "Nie wybrano pliku" (No file selected). Below this is a horizontal row of eight small image thumbnails showing various computer hardware components like cases and internal parts.
- Change Status:** A dropdown menu currently set to "Approved".
- Take the proceedings on yourself:** A checkbox that is currently unchecked.
- Buttons:** "Cancel" and "Submit" buttons are located at the bottom right.

Zarządzania swoimi danymi przez użytkownika aplikacji:

Po kliknięciu na "Profile" użytkownik ma możliwość edycji i usunięcia swojego konta.



DashboardMy ProposalsDepartmentsJan ▾

Profile

Profile Information

Update your account's profile information and email address.

Name

JanA

Email

jan@email.com

SAVE

Update Password

Ensure your account is using a long, random password to stay secure.

Current Password

New Password

Confirm Password

SAVE

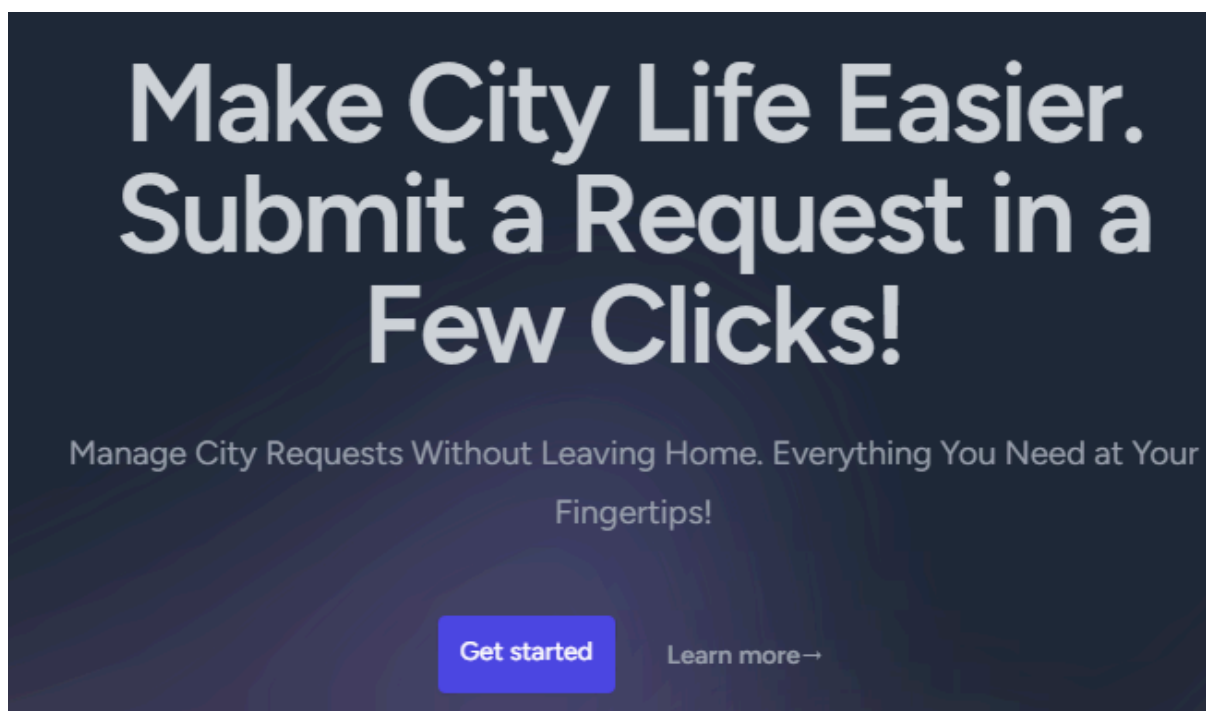
Delete Account

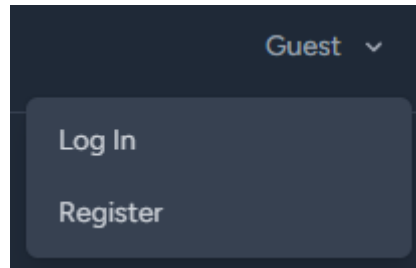
Once your account is deleted, all of its resources and data will be permanently deleted. Before deleting your account, please download any data or information that you wish to retain.

DELETE ACCOUNT

Proces rejestracji nowego użytkownika aplikacji:

Po kliknięciu na guzik “Get started” lub "Register" użytkownik zostaje przekierowany do okna z formularzem rejestracji.





Dane z formularza są umieszczane w haku “useForm”. Jest to React’owy hak do łatwego zarządzania formularzami.

```
const { data, setData, post, processing, errors, reset } = useForm({
  name: '',
  email: '',
  password: '',
  password_confirmation: '',
});
```

Po kliknięciu “Register” (potwierdzeniu formularza) dane są walidowane przez frontend w inputach.

```
<form onSubmit={submit}>
  <div>
    <InputLabel htmlFor="name" value="Name" />

    <TextInput
      id="name"
      name="name"
      value={data.name}
      className="mt-1 block w-full"
      autoComplete="name"
      isFocused={true}
      onChange={(e) => setData('name', e.target.value)}
      required
    />

    <InputError message={errors.name} className="mt-2" />
  </div>

  const submit = (e) => {
    e.preventDefault();

    post(route('register'));
  };
}
```

Potem następuje walidacja w backendzie (w kontrolerze). Użytkownik zostaje utworzony i zalogowany oraz przeniesiony do strony głównej.

```
public function store(Request $request): RedirectResponse
{
    $request->validate([
        'name' => 'required|string|max:255',
        'email' => 'required|string|lowercase|email|max:255|unique:'.User::class,
        'password' => ['required', 'confirmed', Rules\Password::defaults()],
    ]);

    $user = User::create([
        'name' => $request->name,
        'email' => $request->email,
        'password' => Hash::make($request->password),
        'role_id' => 1,
    ]);

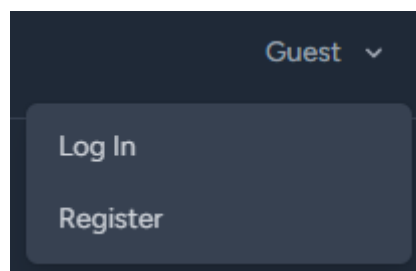
    event(new Registered($user));

    Auth::login($user);

    return redirect(route('dashboard', absolute: false));
}
```

Proces logowania się obecnych użytkowników aplikacji:

Po kliknięciu na “Log In” użytkownik zostaje przekierowany do okna z formularzem logowania.



Dane z formularza są umieszczane w haku “useForm”.

```
const { data, setData, post, processing, errors, reset } = useForm({
    email: '',
    password: '',
    remember: false,
});

const submit = (e) => {
    e.preventDefault();

    post(route('login'));
};
```

Po kliknięciu "Log In" (potwierdzeniu formularza) dane są walidowane przez frontend w inputach.

```
<div className="mt-4">
  <InputLabel htmlFor="password" value="Password" />

  <TextInput
    id="password"
    type="password"
    name="password"
    value={data.password}
    className="mt-1 block w-full"
    autoComplete="current-password"
    onChange={(e) => setData('password', e.target.value)}
  />

  <InputError message={errors.password} className="mt-2" />
</div>
```

Potem następuje walidacja w backendzie.

```
public function rules(): array
{
    return [
        'email' => ['required', 'string', 'email'],
        'password' => ['required', 'string'],
    ];
}
```

Następuje uwierzytelnienie użytkownika:

```
public function authenticate(): void
{
    $this->ensureIsNotRateLimited();

    if (! Auth::attempt($this->only('email', 'password'), $this->boolean('remember'))) {
        RateLimiter::hit($this->throttleKey());

        throw ValidationException::withMessages([
            'email' => trans('auth.failed'),
        ]);
    }

    RateLimiter::clear($this->throttleKey());
}
```

Regeneruje się też ID sesji.

Na koniec, jeśli uwierzytelnianie się powiedzie, metoda store przekierowuje użytkownika do zamierzonej strony.

```
public function store(LoginRequest $request): RedirectResponse
{
    $request->authenticate();

    $request->session()->regenerate();

    return redirect()->intended(route('dashboard', absolute: false));
}
```

Przedstawienie procesu wyświetlania wniosku w aplikacji

Po kliknięciu na tytuł wniosku zostajemy przeniesieni do strony wyświetlającej wniosek.

```
<Link href={route('proposal.show', proposal.id)} >
| {proposal.title}
</Link>
```

Zanim to jednak nastąpi bramka “access-proposal” sprawdza czy mamy do tego uprawnienia, jeśli nie, zwraca błąd 403 (Forbidden). Znajduje się ona w pliku “AppServiceProvider.php”.

```
Gate::define('access-proposal', function (User $user, $proposal) {
    return $user->role_id == 3 || $user->role_id == 2 || $proposal->created_by == $user->id;
});
```

```
public function show(Proposal $proposal)
{
    if (!Gate::allows('access-proposal', $proposal)) {
        abort(403);
    }

    $replies = $proposal->replies()
        ->orderBy('created_at', 'asc')
        ->paginate(10)
        ->onEachSide(1);

    return inertia('Proposal/Show', [
        'proposal' => new ProposalResource($proposal),
        'replies' => ReplyResource::collection($replies),
    ]);
}
```

Kolejna linia kodu tworzy zapytanie do bazy danych, które pobierze wszystkie odpowiedzi na dany wniosek, posortowane w kolejności rosnącej według daty utworzenia z paginacją (10 wyników na stronę) i zwracają wyniki.

Następnie wniosek wraz ze swoimi odpowiedziami zostaje przekazany do widoku.

Przedstawienie procesu zabezpieczenia CRUDu tabeli users:

Route'y powiązanie z CRUDem user są zabezpieczone dodatkowym middleware'em.

```
Route::middleware('role:admin')
->prefix('admin')
->group(function () {
    Route::resource('user', UserController::class);
});
```

Jest odpowiedzialny za sprawdzanie roli użytkownika, który wchodzi na daną stronę. Jeśli nie posiada odpowiedniej roli zwróci błąd 403 (Forbidden).

```
class RoleMiddleware
{
    /**
     * Handle an incoming request.
     *
     * @param \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response) $next
     */
    public function handle(Request $request, Closure $next, ... $roles): Response
    {
        foreach($roles as $role) {
            if($request->user()->hasRole($role)) //has role
                return $next($request);
        }

        return abort(403);
    }
}
```