QUICKCARE

K34
Graś Rafał
Bąk Adrian
Guźniczak Julia

I – TEMATYKA PROJEKTU

Quick Care to aplikacja stworzona z myślą o przychodniach lekarskich. Zadaniem Quick Care jest usprawnienie oraz uporządkowanie przebiegu procesu rejestracji pacjenta na wizytę lekarską w przychodni XYZ.

Quick Care dla PACJENTA to możliwość szybkiej, internetowej rejestracji na wizytę lekarską w przychodni XYZ.

Quick Care dla PRACOWNIKA PRZYCHODNI to możliwość sprawnego przeprowadzenia procesu rejestracji pacjenta na wizytę lekarską w przychodni XYZ **(REJESTRATOR MEDYCZNY)**.

II - OPROGRAMOWANIE

Do stworzenia Quick care wykorzystano framework do aplikacji internetowych w języku PHP - Laravel.

III - DIAGRAM PRZYPADKÓW UŻYCIA

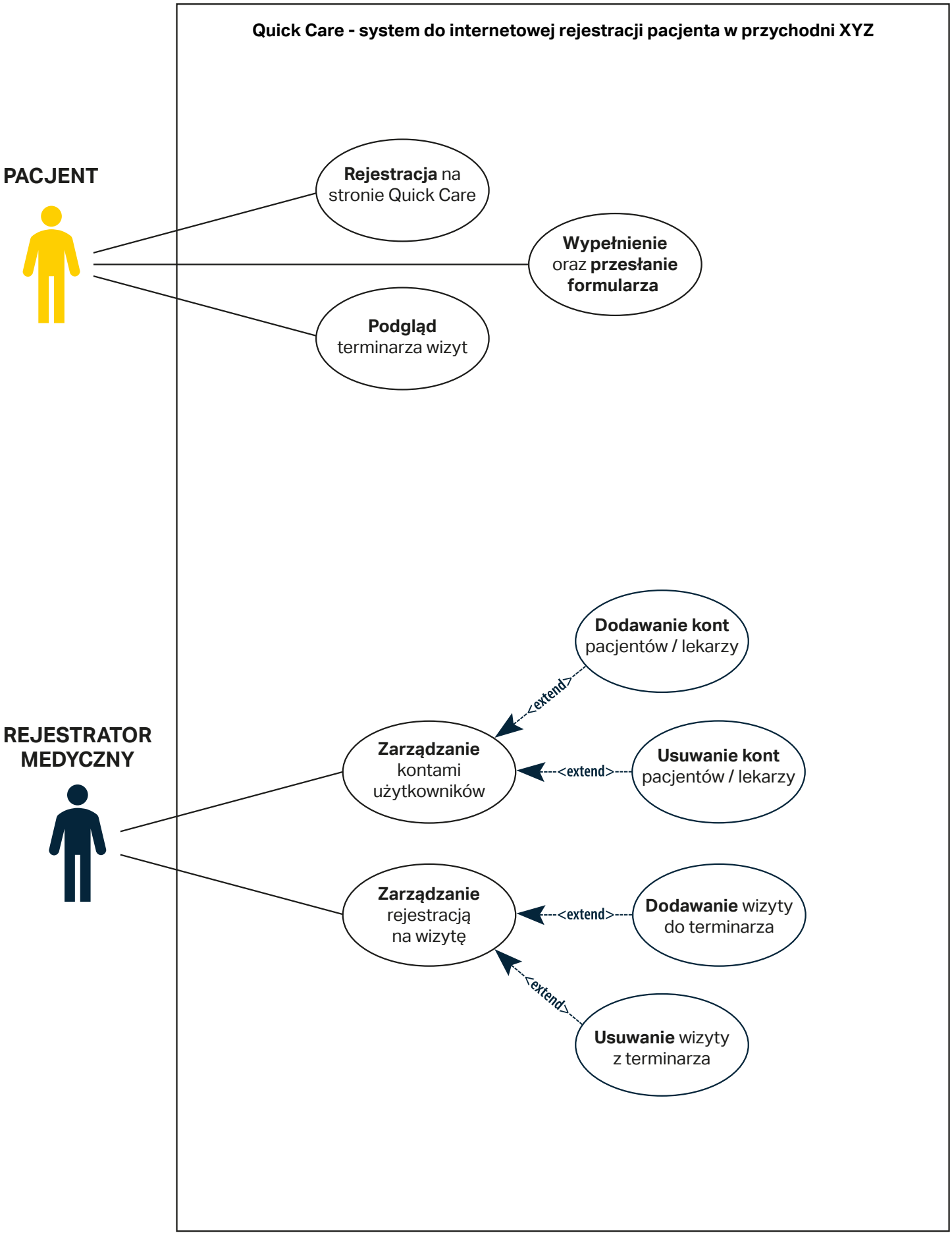
Wprowadzenie - AKTORZY I ICH ROLA / Pacjent oraz pracownik przychodni (Rejestrator Medyczny).

PACJENT

- Rejestracja na stronie Quick Care.
- Wypełnienie i przesłanie formularza w celu umówienia się na wizytę lekarską.
- Możliwość przeglądania terminarza i historii wizyt.

REJESTRATOR MEDYCZNY

- Dodawanie oraz usuwanie pacjentów i lekarzy do bazy danych Quick Care.
- Zarządzanie procesem rejestracji pacjenta na wizytę u lekarza.


Quick Care - system do internetowej rejestracji pacjenta w przychodni XYZ
PACJENT
Rejestracja na stronie Quick Care
Wypełnienie oraz przesłanie formularza
Podgląd terminarza wizyt
REJESTRATOR MEDYCZNY
Dodawanie kont pacjentów / lekarzy
<extend>
Zarządzanie kontami użytkowników
<extend>
Usuwanie kont pacjentów / lekarzy
Zarządzanie rejestracją na wizytę
<extend>
Dodawanie wizyty do terminarza
<extend>
Usuwanie wizyty z terminarza

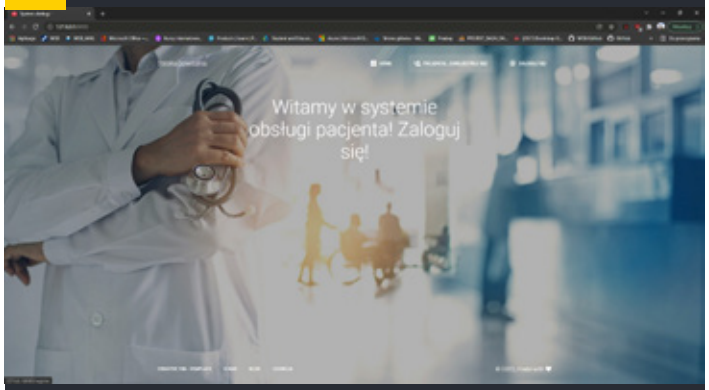
Opis techniczny / Jak działa Quick Care?

PACJENT

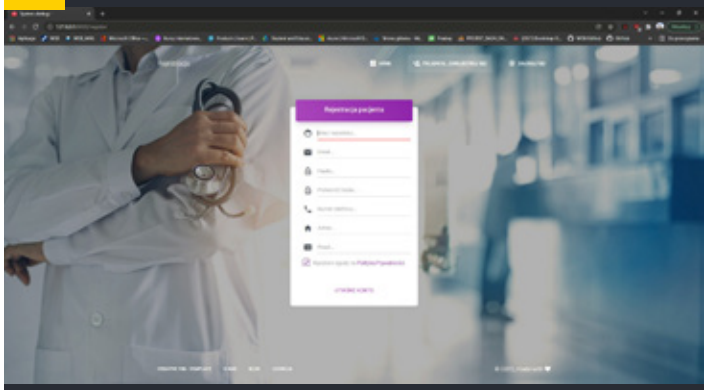
I - Rejestracja na stronie Quick Care

• Proces rejestracji na stronie Quick Care - PODGLĄD PACJENT:

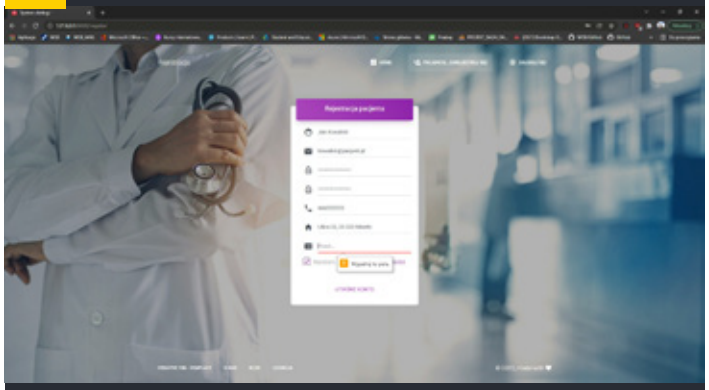
1



2



3



• Kod - formularz rejestracji pacjenta na stronir Quick Care:

```
protected function create(array $data)
{
    return User::create([
        'name' => $data['name'],
        'email' => $data['email'],
        'password' => Hash::make($data['password']),
        'phone' => $data['phone'],
        'address' => $data['address'],
        'pesel' => $data['pesel'],
        'status' => $data['status'],
        'type' => $data['type']
    ]);
}
```

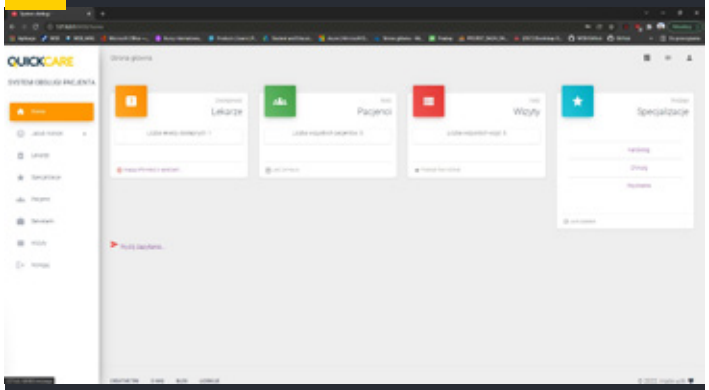
Opis techniczny / Jak działa Quick Care?

PACJENT

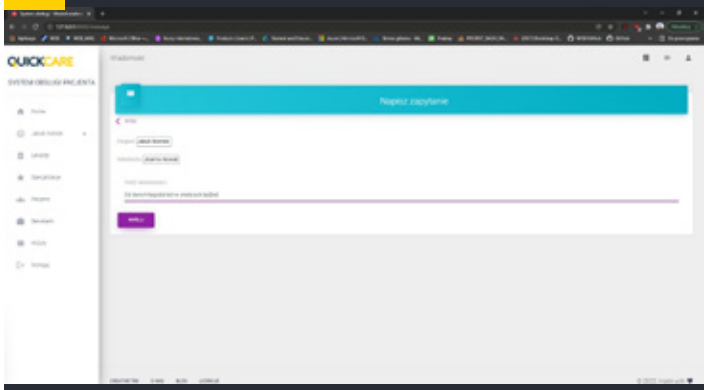
II - Wypełnienie oraz przesłanie formularza w celu umówienia się na wizytę lekarską

• Proces wypełniania oraz wysyłania formularza w celu umówienia się na wizytę lekarską - PODGLĄD PACJENT:

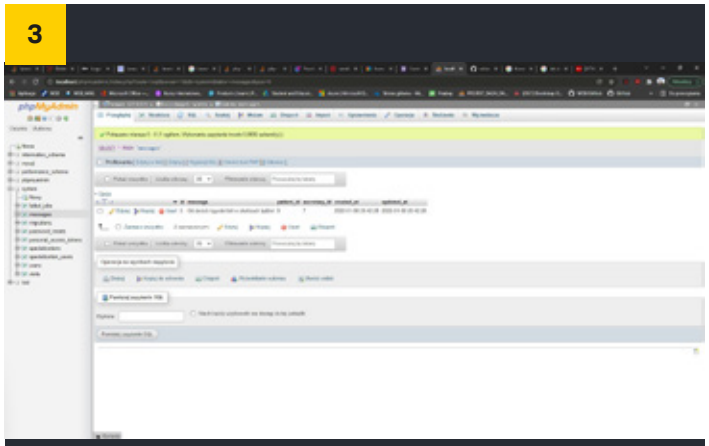
1



2



3



• Kod:

```
Route::group(['middleware' => 'auth'], function () {
  Route::get('/message/','App\Http\Controllers\MessageController@create');
  Route::post('/message/','App\Http\Controllers\MessageController@store');
});
```

```
public function create(UserRepository $userRepo){

  $patients = $userRepo->getAllPatients();
  $secretary = $userRepo->getAllSecretary();

  return view('message.create',["patients"=>$patients,"secretary"=>$secretary,"title"=>"Moduł wiadomości"]);
}

public function store(Request $request){

  $message = new Message;
  $message->patient_id = $request->input('patient');
  $message->secretary_id = $request->input('secretary');
  $message->message = $request->input('message');
  $message->save();

  $patient = User::find($message->patient_id);
  $secretary = User::find($message->secretary_id);

  Mail::send('emails.message',['message' => $message], function ($m) use ($message,$secretary){
    $m->to($secretary->email,$secretary->name)->subject('Nowe zapytanie');
  });

  return redirect()->action('App\Http\Controllers\HomeController@index');
 }
}
```

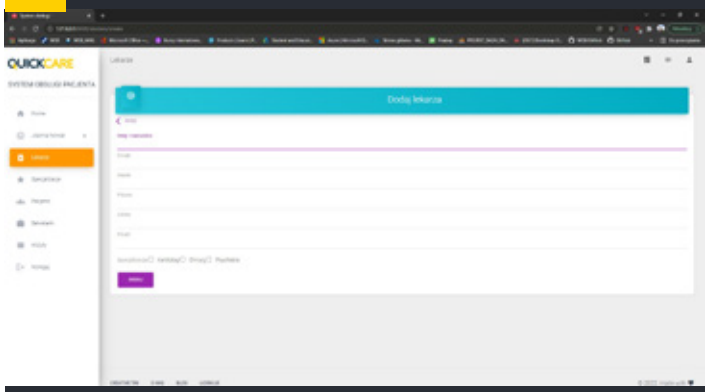
Opis techniczny / Jak działa Quick Care?

REJESTRATOR MEDYCZNY

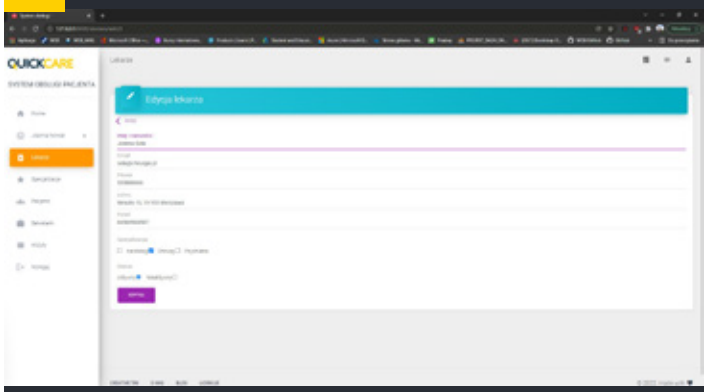
I - Dodawanie oraz usuwanie lekarzy

• Proces dodawania oraz usuwania kont lekarzy - PODGLĄD REJESTRATOR MEDYCZNY:

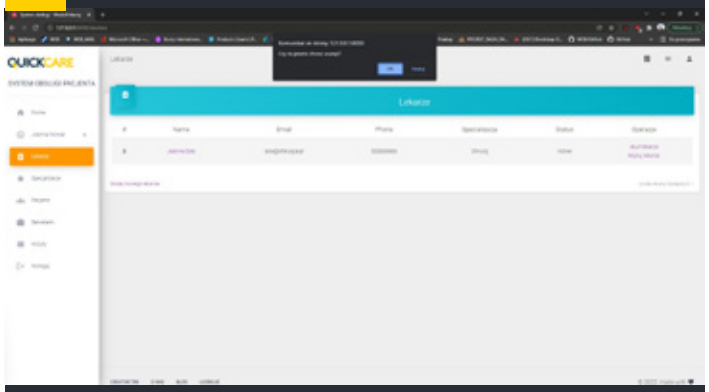
1



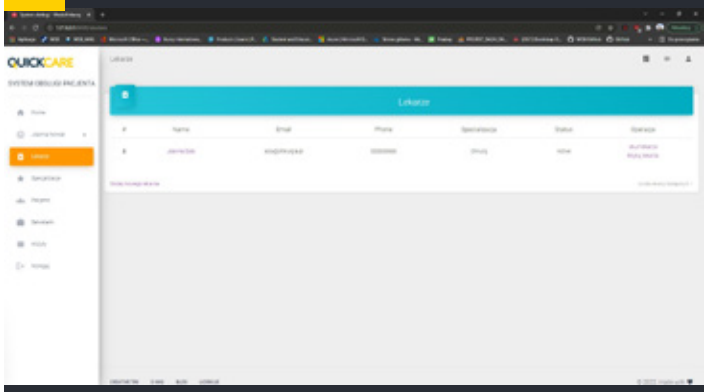
2



3



4



• Kod - dodawanie / edytowanie / usuwanie konta lekarza:

```
public function store(Request $request){

  if(Auth::user()->type != 'secretary' && Auth::user()->type != 'admin')
  {
    return redirect()->route('login');
  }

  $request->validate([
    'name' => 'required|max:255',
    'email' => 'required|email|unique:users,email',
    'password' => 'required|min:5',
    'phone' => 'required',
    'address' => 'required',
    'pesel' => 'required|min:11|max:11'
  ]);

  $doctor = new User;
  $doctor->name = $request->input('name');
  $doctor->email = $request->input('email');
  $doctor->password = bcrypt($request->input('password'));
  $doctor->phone = $request->input('phone');
  $doctor->address = $request->input('address');
  $doctor->pesel = $request->input('pesel');
  $doctor->states = $request->input('status');
  $doctor->type = 'doctor';
  $doctor->save();
  $doctor->specializations()->sync($request->input('specializations'));

  return redirect()->action('App\Http\Controllers\DoctorController@index');
}
```

```
public function create(){

  if(Auth::user()->type != 'secretary' && Auth::user()->type != 'admin')
  {
    return redirect()->route('login');
  }

  $specializations = Specialization::All();

  return view('doctors.create', ["specializations" => $specializations]);
}
```

```
Route::get('/doctors/create','App\Http\Controllers\DoctorController@create');
```

Opis techniczny / Jak działa Quick Care?

• Kod - dodawanie / edytowanie / usuwanie konta lekarza:

```php
public function editStore(Request $request){

  if(Auth::user()->type != 'secretary' && Auth::user()->type != 'admin')
  {
    return redirect()->route('login');
  }

  $doctor = User::find($request->input('id'));
  $doctor->name = $request->input('name');
  $doctor->email = $request->input('email');
  $doctor->phone = $request->input('phone');
  $doctor->address = $request->input('address');
  $doctor->pesel = $request->input('pesel');
  $doctor->status = $request->input('status');
  $doctor->save();
  $doctor->specializations()->sync($request->input('specializations'));

  return redirect()->action('App\Http\Controllers\DoctorController@index');
}
```

```php
Route::post('/doctors/edit/','App\Http\Controllers\DoctorController@editStore');
```

```php
public function delete(UserRepository $userRepo,$id){

  if(Auth::user()->type != 'secretary' && Auth::user()->type != 'admin')
  {
    return redirect()->route('login');
  }

  $doctor = $userRepo->delete($id);
  return redirect('doctors');
}
```

```php
Route::get('/doctors/delete/{id}', 'App\Http\Controllers\DoctorController@delete');
```

```php
public function show(UserRepository $userRepo,$id){

  $doctor = $userRepo->find($id);

  return view('doctors.show',["doctor"=>$doctor,"title"=>"Moduł lekarzy"]);
}
```

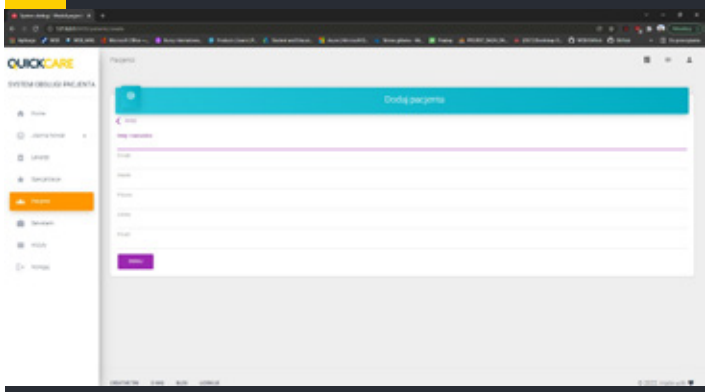
Opis techniczny / Jak działa Quick Care?

REJESTRATOR MEDYCZNY

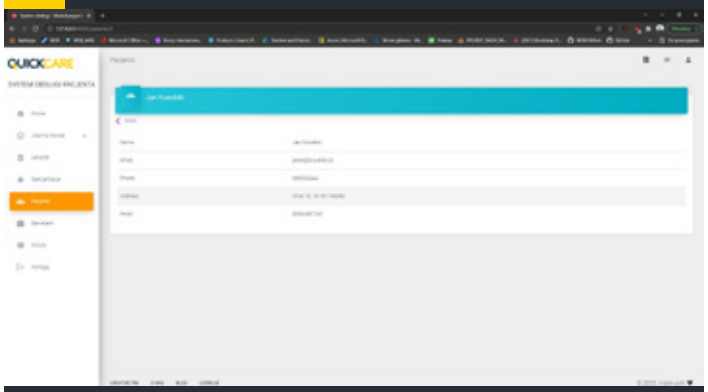
II - Dodawanie oraz usuwanie pacjentów

• Proces dodawania oraz usuwania kont pacjentów - PODGLĄD REJESTRATOR MEDYCZNY:

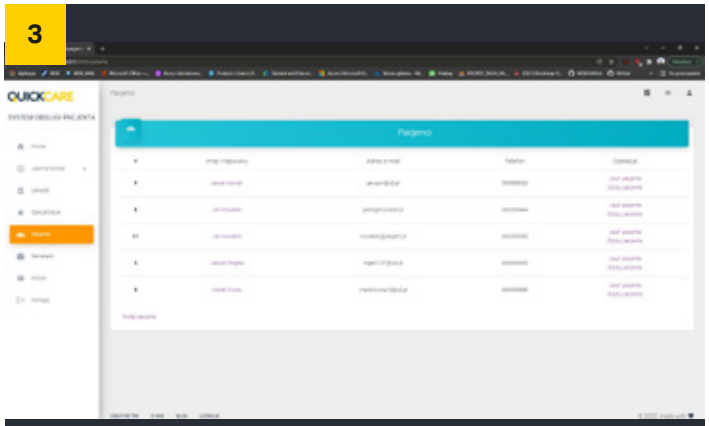
1



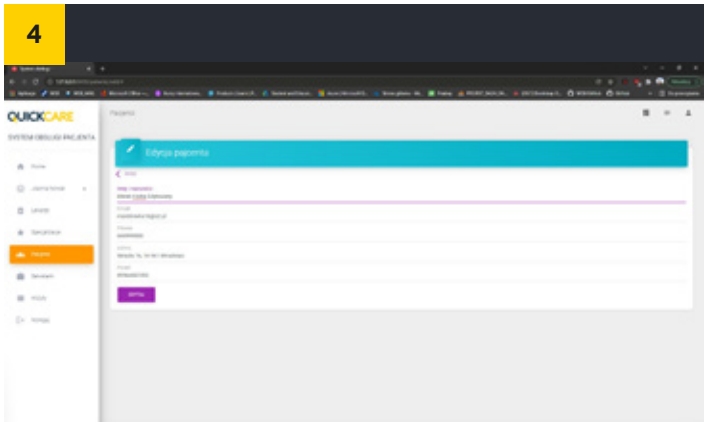
2



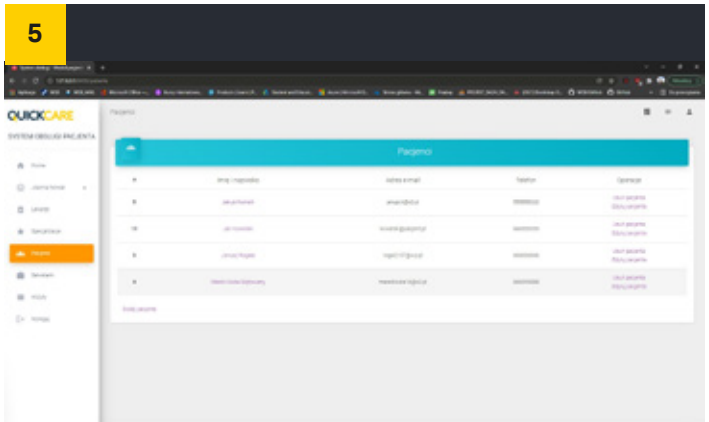
3



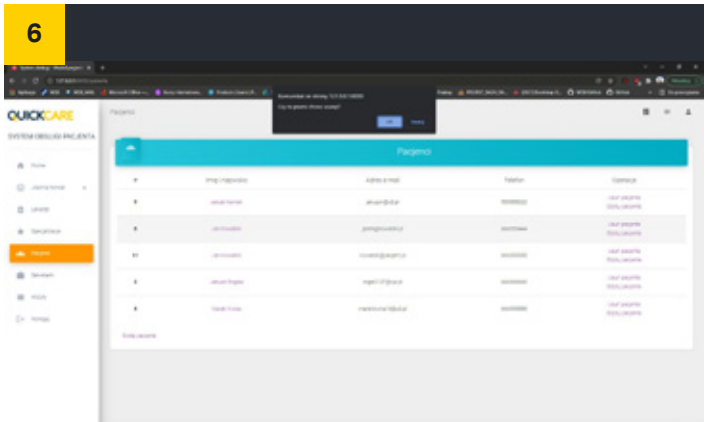
4



5



6

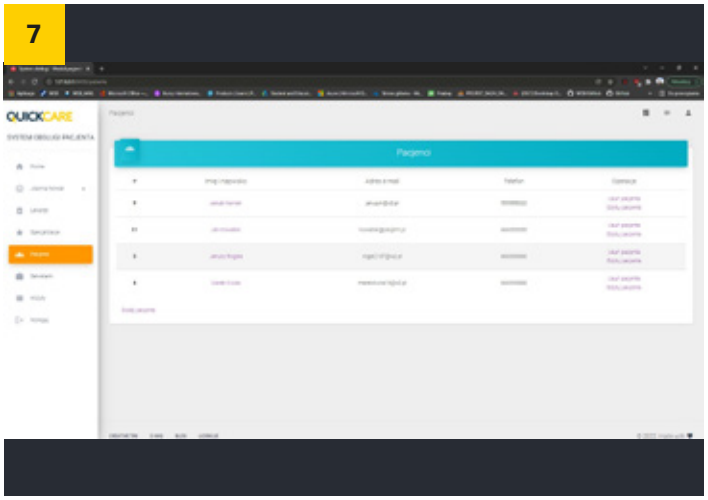


Opis techniczny / Jak działa Quick Care?

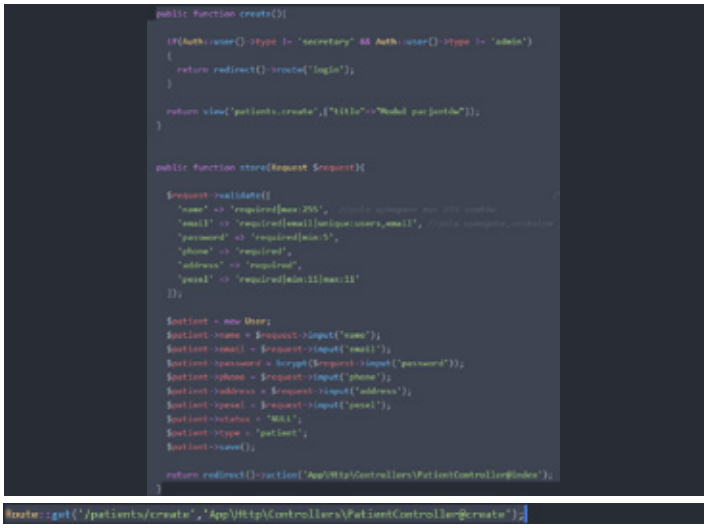
REJESTRATOR MEDYCZNY

III - Dodawanie oraz usuwanie pacjentów

• Proces dodawania oraz usuwania kont pacjentów - PODGLĄD REJESTRATOR MEDYCZNY:



• Kod:



```
public function create(){

  if(Auth::user()->type != 'secretary' && Auth::user()->type != 'admin')
  {
    return redirect()->route('login');
  }

  return view('patients.create',["title"=>"Moduł pacjentów"]);
}


public function store(Request $request){

  $request->validate([
    'name' => 'required|max:255',
    'email' => 'required|email|unique:users,email',
    'password' => 'required|min:5',
    'phone' => 'required',
    'address' => 'required',
    'pesel' => 'required|min:11|max:11'
  ]);

  $patient = new User;
  $patient->name = $request->input('name');
  $patient->email = $request->input('email');
  $patient->password = bcrypt($request->input('password'));
  $patient->phone = $request->input('phone');
  $patient->address = $request->input('address');
  $patient->pesel = $request->input('pesel');
  $patient->status = "NULL";
  $patient->type = 'patient';
  $patient->save();

  return redirect()->action('App\Http\Controllers\PatientController@index');
}
```

```
public function show(UserRepository $userRepo,$id){

  if(Auth::user()->type != 'secretary' && Auth::user()->type != 'admin')
  {
    return redirect()->route('login');
  }


  $patient = $userRepo->find($id);

  return view('patients.show',["patient"=>$patient,
                               "title"=>"Moduł pacjentów"]);
}
```

```
Route::get('/patients/create','App\Http\Controllers\PatientController@create');
```

```
public function getAllPatients(){
  return $this->model->where('type','patient')->orderBy('name','asc')->get();
}
```

```
Route::get('/patients/edit/{id}','App\Http\Controllers\PatientController@edit');
Route::post('/patients/edit/','App\Http\Controllers\PatientController@editStore');
```

```
public function editStore(Request $request){

  if(Auth::user()->type != 'secretary' && Auth::user()->type != 'admin')
  {
    return redirect()->route('login');
  }

  $patient = User::find($request->input('id'));
  $patient->name = $request->input('name');
  $patient->email = $request->input('email');
  $patient->phone = $request->input('phone');
  $patient->address = $request->input('address');
  $patient->pesel = $request->input('pesel');
  $patient->status = 'NULL';
  $patient->save();

  return redirect()->action('App\Http\Controllers\PatientController@index');
}
```

Opis techniczny / Jak działa Quick Care?

• Kod:

```php
public function edit(UserRepository $userRepo,$id){

  if(Auth::user()->type != 'secretary' && Auth::user()->type != 'admin')
  {
    return redirect()->route('login');
  }

  $patient = $userRepo->find($id);

  return view('patients.edit', ["patient" => $patient,
                                "footerYear" =>date("Y")]);
}
```

```php
public function delete(UserRepository $userRepo,$id){

  if(Auth::user()->type != 'secretary' && Auth::user()->type != 'admin')
  {
    return redirect()->route('login');
  }

  $patient = $userRepo->delete($id);
  return redirect('patients');
}
```

```php
Route::get('/patients/delete/{id}', 'App\Http\Controllers\PatientController@delete');
```

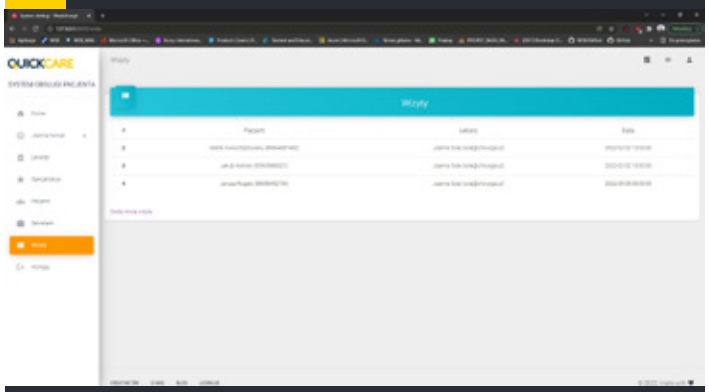
Opis techniczny / Jak działa Quick Care?

REJESTRATOR MEDYCZNY

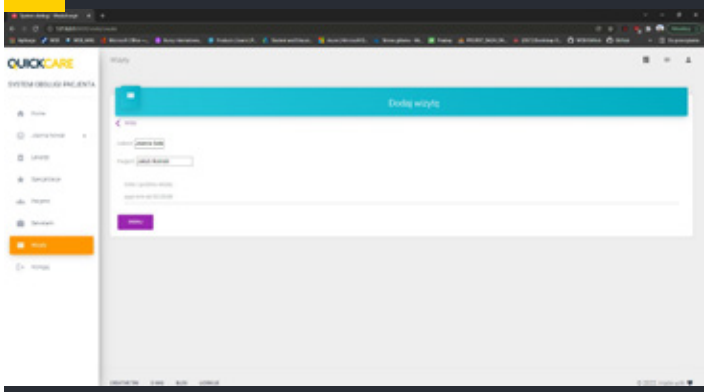
IV - Zarządzanie procesem rejestracji pacjenta na wizytę u lekarza

• Proces zarządzania rejestracją pacjenta na wizytę lekarską - PODGLĄD REJESTRATOR MEDYCZNY:

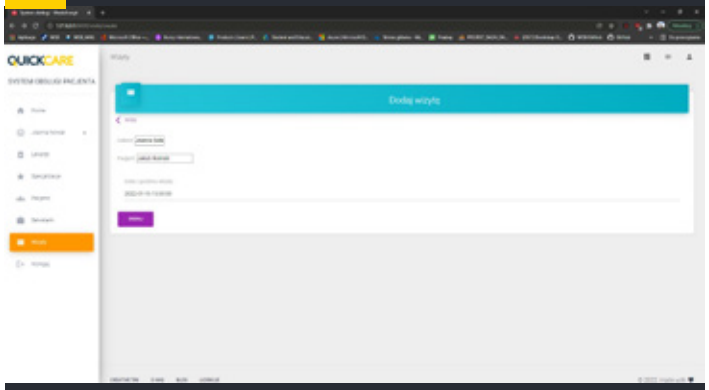
1



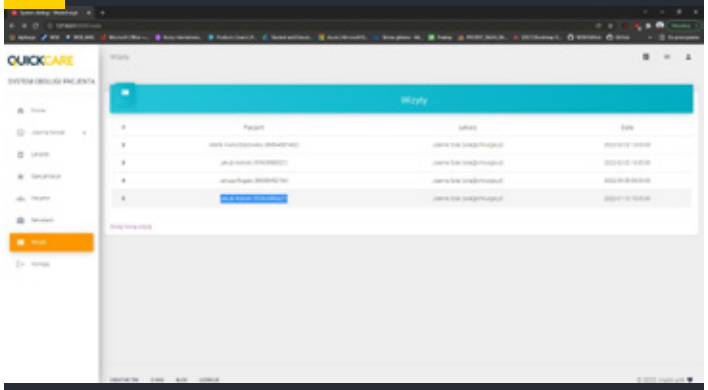
2



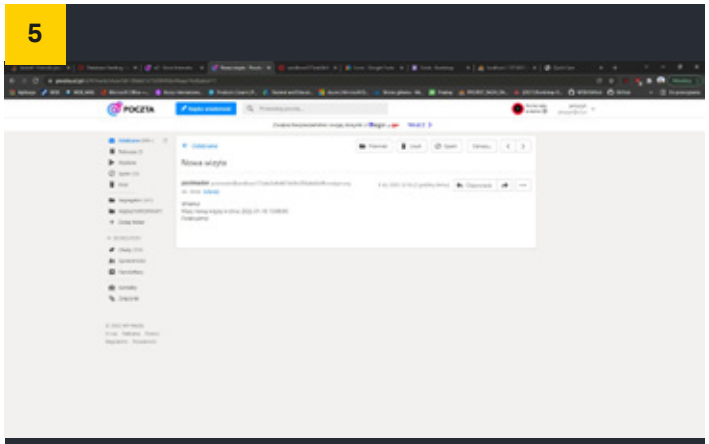
3



4



5



Opis techniczny / Jak działa Quick Care?

• Kod:

```
public function create(UserRepository $userRepo){

  if(Auth::user()->type != 'secretary' && Auth::user()->type != 'admin')
  {
    return redirect()->route('login');
  }

  $doctors = $userRepo->getAllDoctors();
  $patients = $userRepo->getAllPatients();

  return view('visits.create',["patients"=>$patients, "doctors" =>$doctors ,"footerYear"=>date("Y"),"title"=>"Moduł wizyt"]);
}
```

```
public function store(Request $request){

  if(Auth::user()->type != 'secretary' && Auth::user()->type != 'admin')
  {
    return redirect()->route('login');
  }
  $request->validate([
    'date' => 'required'
  ]);

  $visit = new Visit;
  $visit->doctor_id = $request->input('doctor');
  $visit->patient_id = $request->input('patient');
  $visit->date = $request->input('date');
  $visit->save();

  $patient = User::find($visit->patient_id);

  Mail::send('emails.visit',['visit' => $visit], function ($m) use ($visit,$patient){
    $m->to($patient->email,$patient->name)->subject('Nowa wizyta');
  });

  return redirect()->action('App\Http\Controllers\VisitController@index');
}
```



```
Route::get('/visits/create','App\Http\Controllers\VisitController@create');
```

```
  $visits = $visitRepo ->getAll();

  return view('visits.list',["visits"=>$visits,
                             "footerYear"=>date("Y"),
                             "title"=>"Moduł wizyt"]);
}
```

I V – OPIS TECHNICZNY

Opis techniczny / Problemy i zagrożenia

Potencjalne problemy i zagrożenia:
1. Ograniczenia czasowe na realizację projektu.
2. Potencjalny problem z obsługą formularza wiadomości wysyłanego przez pacjenta - docelowo wiadomość ma trafić na adres e-mail rejestratora medycznego (sekretarki).
3. Brak oddzielnych stron głównych (dashboardów) dla poszczególnych uzytkowników - problem częściowo rozwiązany poprzez funkcję sprawdzającą typ użytkownika - blade dla każdego uzytkownika jest ten sam.

V – SPIS TABEL BAZ DANYCH

Tabele baz danych:
1. **messages** - odpowiada za przechowywanie danych z formularza wysyłanego prze pacjenta
2. **specialization** - odpowiada za przechowywanie specjalizacji lekarzy
3. **specialization_users** - przechowuje id lekarza oraz specjalizacji, relacja pomiedzy tabela users a specialization
4. **users** - przechowuje dane użytkowników systemu
5. **visits** - przechowuje dane na temat wizyt (id lekarza, id pacjenta, data wizyty)

VI – ZAŁOŻENIA PROJEKTU I ICH REALIZACJA

ZAŁOŻENIA	REALIZACJA	KOMENTARZ
PACJENT - rejestracja na stronie Quick Care	Tak	Udało się zrealizować założenie.
PACJENT - wypełnienie / przesłanie formularza w celu umówienia się na wizytę lekarską	Tak	Udało się zrealizować założenie.
REJESTRATOR MEDYCZNY - dodawanie oraz usuwanie pacjentów i lekarzy	Tak	Udało się zrealizować założenie.
REJESTRATOR MEDYCZNY - zarządzanie procesem rejestracji pacjentów na wizytę	Tak	Udało się zrealizować założenie.
REJESTRATOR MEDYCZNY / LEKARZ - podgląd terminarza wizyt	Nie	W założeniu Pacjent oraz Lekarz mieli mieć możliwość podlądu terminarza wizyt w formie kalendarza. Założenia nie udało się zrealizować.

VII – DANE KONFIGURACYJNE APLIKACJI

Seeder -> domyslny uzytkownik Admin
E-mail: admin@admin.pl
Hasło: admin@admin.pl

Nazwa bazy danych : system

Skonfigurowany mailer
mailgun