

# POLITECHNIKA ŚLĄSKA W GLIWICACH

## WYDZIAŁ INŻYNIERII BIOMEDYCZNEJ



Programowanie Urządzeń Mobilnych

Projekt

Serwis internetowy do zamieszczania i obserwacji zapisów  
EKG z urządzenia mobilnego

Kamil Suchanek  
IiAM 2 sem. 6

Kierujący pracą  
Dr inż. Marcin Rudzki

Bytom – 15.09.2018

## Spis treści

1.	<i>Cel projektu</i>	3
2.	<i>Założenia</i>	3
3.	<i>Teoria</i>	3
3.1.	<i>Zarys elektrokardiografii</i>	3
3.2.	<i>Monitorowanie EKG metodą Holtera</i>	4
3.3.	<i>Heroku</i>	4
3.4.	<i>Ruby on Rails - RoR</i>	4
4.	<i>Realizacja projektu</i>	4
5.	<i>Specyfikacja wewnętrzna</i>	5
5.1.	<i>Struktura aplikacji</i>	5
5.2.	<i>Model User</i>	6
5.3.	<i>Widoki</i>	6
5.4.	<i>Kontroler – user_controller</i>	10
5.5.	<i>Błędy aplikacji, rozwiązania</i>	14
6.	<i>Specyfikacja zewnętrzna</i>	14
6.1.	<i>Start aplikacji – zakładanie nowego konta.</i>	14
6.2.	<i>Logowanie</i>	17
6.3.	<i>Lista użytkowników, profile, edycja i wiadomości</i>	18
7.	<i>Podsumowanie</i>	21
8.	<i>Źródła</i>	22

## **1. Cel projektu**

Stworzenie strony internetowej służącej to zamieszczania i wizualizacji elektrokardiogramu rejestrowanego przez urządzenie mobilne.

## **2. Założenia**

Strona ma być napisana za pomocą technologii Ruby on Rails. Serwis umożliwia utworzenie odrębnych kont dla użytkowników. Konto może mieć jedno z trzech rodzajów uprawnień: administrator, lekarz oraz pacjent. Uzyskany elektrokardiogram ma być użyteczny pod względem diagnostycznym. Ostatecznym odbiorcą jest każdy z trzech rodzajów użytkowników, gdzie każdy posiada swój profil, możliwość zamieszczenia swoich rekordów. Możliwa ma być interakcja pomiędzy użytkownikami za pośrednictwem strony z inicjatywy lekarza. Pacjent nie ma wglądu w rekordy innych użytkowników, w przeciwieństwie do lekarza, który ma owo również na profil innych lekarzy. Urządzenie powinno być ustawione tylko na jednego użytkownika.

## **3. Teoria**

### **3.1. Zarys elektrokardiografii**

Podstawowym urządzeniem służącym do rejestracji napięć elektrycznych tkanki żywej jest galwanometr. Obwód elektryczny składający się z dwóch zakończeń galwanometru (ujemnej elektrody i dodatniej) oraz dwóch punktów pola elektrycznego nazywa się odprowadzeniem. Rozróżnia się dwa rodzaje odprowadzeń: jednobiegunowe, gdzie jedną z elektrod umieszcza się w miejscu pomiaru potencjału a drugą w miejscu oddalonym od owego, oraz dwubiegunowe, gdzie obie elektrody umieszcza się w miejscach pomiaru potencjału. Pomiary uzyskane za pomocą odprowadzeń jednobiegunowych dostarczają bezwzględne wartości potencjałów. Dopiero zastosowanie odprowadzeń dwubiegunowych umożliwia ocenę zmiennego pola elektrycznego na powierzchni komórki.

Siła elektromotoryczna serca jest wartością wektorową. Rejestracji podlega wypadkowa wartość siły elektromotorycznej serca. Siła ta zmienia się w czasie w sposób cykliczny i odpowiada kolejnym ewolucjom cyklu sercowego. Wielkość mierzona na powierzchni ciała osiąga wartość do około 2 mV.

Krzywa elektrokardiograficzna jest graficznym przedstawieniem elektrycznej czynności serca. Składają się na nią wychylenia wzdłuż linii izoelektrycznej. Szczegóły morfologii tych wychyleń i odległości pomiędzy nimi przekładają się na wartość diagnostyczną. Również ważnym aspektem podlegającym ocenie typowego badania EKG jest częstość i cykliczność wychyleń zapisu.

### **3.2. Monitorowanie EKG metodą Holtera**

Potocznie holter – jest metodą rejestracji pracy serca przy pomocy urządzenia mobilnego. Aparat Holtera ma za zadanie rejestrować EKG w trybie ciągłym przez 24 godziny na dobę. Sygnał zapisywany jest w celu późniejszej analizy. Urządzenie to ze względu na zastosowanie musi być niewielkich rozmiarów z skutecznym i nie krępującym ruchów sposobem przytwierdzenia go do pacjenta.

### **3.3. Heroku**

Heroku jest platformą chmurową działającą jako usługa (ang. Platform as a Service – PaaS), czyli jest to usługa, która udostępnia wirtualne środowiska pracy przeznaczone głównie dla programistów. Jest to jedna z pierwszych tego rodzaju platform. Aktualnie obsługuje kilka języków programowania, a pierwszym z nich był Ruby. Heroku umożliwia wsparcie od utworzenia aplikacji po jej wdrożenie i konserwację. Ograniczając się do pewnych zasobów możliwe jest korzystanie bezpłatnie. Platforma ta udostępnia szczegółową dokumentację i wsparcie dla każdego z obsługiwanych języków. Na samą platformę składa się wiele różnych usług, między innymi związana z wdrażaniem, główna oraz Heroku Postgres związana z źródłami danych dla aplikacji.

### **3.4. Ruby on Rails - RoR**

Railsy to otwartoźródłowy framework służący do szybkiego prototypowania aplikacji webowych. Jego głównym twórcą jest David Heinemeier Jansson. RoR został napisany w języku Ruby z użyciem architektury MVC (ang. Model-View-Controller).

Główne założenia:

- Szybkość, łatwość i przyjemność pisania kodu,
- Reguła DRY (ang. Don't Repeat Yourself), polegająca na nie powtarzaniu tej samej pracy w kilku miejscach,
- Reguła Konwencja Ponad Konfigurację – polegająca na sprowadzeniu do minimum niezbędnej konfiguracji przez zastępowanie jej pewnymi domyślnymi wzorcami,
- Możliwość załączenia przeróżnych gotowych funkcjonalności jak logowanie czy przesyłanie i skalowanie obrazów.

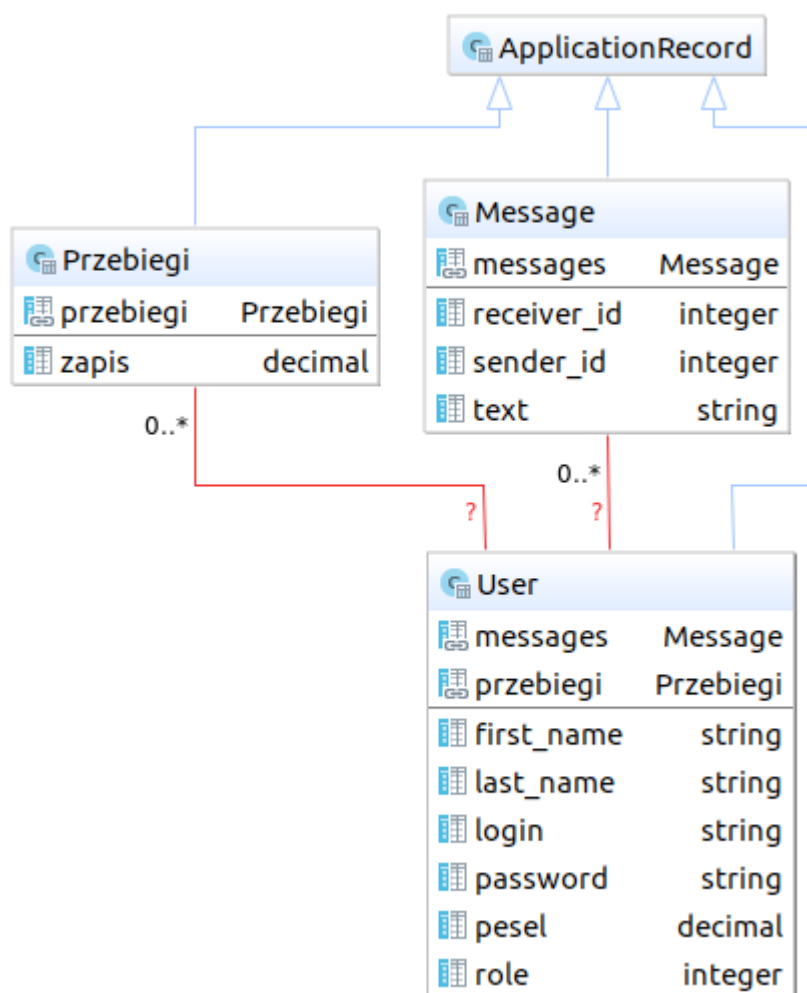
## **4. Realizacja projektu**

Projekt został napisany przy pomocy środowiska RubyMine. Funkcjonalności aplikacji z wyjątkiem rysowania wykresów zostały napisane ręcznie bez użycia generatorów zawartych w odpowiednich gemach jak Devise. Do wizualizacji EKG wykorzystano klejnot chartkick opierający się na Google Charts. Rekordy zamieszczone na stronie mają długość po minutę każdy i wyświetlane są w ilości sześć na raz po 10 sekund. Funkcjonalność przekazywania wiadomości i edycji użytkowników nie działa prawidłowo, ewentualnie przebywanie w danym fragmencie aplikacji nie powoduje jednak błędów.

## 5. Specyfikacja wewnętrzna

### 5.1. Struktura aplikacji

Aplikacja składa się z trzech modeli (Rys. 1.). Działanie jej opiera się o model User. Message oraz Przebiegi służą definicji bazy danych aplikacji oraz ustaleniu relacji wewnątrz kodu owej.



Rys. 1. Diagram modeli – środowisko RubyMine.

## 5.2. Model User

```
1 class User < ApplicationRecord
2   has_many :przebiegi
3   has_many :messages
4   enum role: [:patient, :doctor, :admin]
5   after_initialize :set_default_role, :if => :new_record?
6
7   def set_default_role
8     self.role ||= :patient
9   end
10
11   def hash_password
12     require 'digest/sha2'
13     self.password = Digest::SHA2.new << password
14   end
15   validates :first_name, :presence => true
16   validates :last_name, :presence => true
17   validates :pesel, :presence => true
18   validates_uniqueness_of :pesel
19   validates :login, :presence => true, :length => { :minimum => 4, :maximum => 15 }
20   validates_uniqueness_of :login
21
22   #validates :email, :format => { :with => /\A[a-zA-Z0-9.\-_]+\@[a-zA-Z0-9.\-_]+\.[a-z]{2,4}\Z/ }
23   #validates_uniqueness_of :email
24
25   validates :password, :length => { :minimum=> 4 }
26   after_validation :hash_password
27
28 end
```

Rys. 2. Kod modelu User.

W modelu ustalona została relacja pomiędzy obiektami user a przebiegami oraz wiadomościami – każdy z użytkowników może posiadać wiele wiadomości albo przebiegów. Same relacje nie są definiowane dla bazy danych co można by uzyskać ręcznie bądź za pomocą odpowiednich migracji. Jednak relacje na poziomie kodu zupełnie wystarczą do operowania nimi poprawnie w obrębie aplikacji. Dalej zostały ustalone role użytkowników odpowiadające liczbom całkowitym 0, 1, 2 w bazie danych, a z poziomu kodu już jako konkretne nazwy. Poniżej powstał zapis zapewniający, by każdy nowoutworzony użytkownik posiadał rolę pacjenta. W modelu również zostaje rozwiązany problem walidacji i ewentualnego szyfrowania danych przesyłanych do bazy z poziomu aplikacji (Rys. 2.).

## 5.3. Widoki

Widoki zostały podporządkowane akcjom kontrolera *user\_controller*. Odpowiadają one za wygląd konkretnych stron aplikacji i dostarczają obiektów potrzebnych do interakcji z kontrolerem.

- edit – odpowiada za stronę edytowania danych użytkowników.

```

1 <h1>Edycja</h1>
2
3 <%= if @save %>
4   <%= if @save.errors %>
5     <ul>
6       <%= @save.errors.full_messages.each do |err| %>
7         <li>
8           <%= err %></li>
9         <%= end %>
10      <%= end %>
11    <%= end %>
12  <%= end %>
13
14  <%= form_for "edit", :url => { :action => "edit" }, :remote => true do |f| %>
15    <%= f.label "first_name", "Imię" %><br>
16    <%= f.text_field "first_name", :value => @current_user.first_name %><br><br>
17    <%= f.label "last_name", "Nazwisko" %><br>
18    <%= f.text_field "last_name", :value => @current_user.last_name %><br><br>
19    <%= f.label "pesel", "Pesel" %><br>
20    <%= f.text_field "pesel", :value => @current_user.pesel.to_s %><br><br>
21    <%= f.label "role", "Rola" %><br>
22    <%= f.select "role", ["patient", "doctor", "admin"], :selected => @current_user.role, :style => "height:30px; width:100px" %><br>
23    <br><br><br><br>
24    <%= f.submit "Zapisz zmiany" %>
25    <p><a href="/user/index">Powrót</a></p>
26  <%= end %>
27
28
29
30
31

```

Rys. 3. Widok edit.

- index – ten widok odpowiada za stronę z listą użytkowników.

```

1 <h1>Lista użytkowników</h1>
2
3 <table class="tabela">
4   <thead>
5
6     <tr>
7       <td class = "czerwony">Id</td>
8       <td>Imię</td>
9       <td>Nazwisko</td>
10      <td>Login</td>
11      <td>Pesel</td>
12      <td>Rola</td>
13      <td>Edycja</td>
14      <td>Profil</td>
15    </tr>
16  </thead>
17
18  <tbody>
19    <%= @users.each do |user| %>
20      <tr>
21        <td class = "czerwony"><%=h user.id %></td>
22        <td><%=h user.first_name %></td>
23        <td><%=h user.last_name %></td>
24        <td><%=h user.login %></td>
25        <td><%=h user.pesel %></td>
26        <td><%=h user.role %></td>
27        <td><%=h link_to 'Edytuj', user_edit_path(:ciaramba => user.id) %></td>
28        <td><%=h link_to 'Profil', user_profile_path(:ciaramba => user.id) %></td>
29      </tr>
30    <%= end %>
31  </tbody>
32 </table>
33
34 <p><a href="/user/start">Powrót do strony głównej.</a></p>

```

Rys. 4. Widok index.

- login – Widok odpowiadający za układ strony logowania użytkowników.

```
login.html.erb x
1 <h3>Logowanie</h3>
2 <form method="POST" action="/user/login">
3   <label for="user">User:</label><input type="text" name="login[login]" id="user"><br>
4   <label for="password">Password:</label><input type="password" name="login[password]" id="password"><br>
5   <input type="submit" value = "Zaloguj!">
6   <input name="authenticity_token" type="hidden" value="<%= form_authenticity_token %>">
7   <p><a href="/user/new">Zarejestruj się!</a></p>
8 </form>
```

Rys. 5. Widok login.

- logout – bez zastosowania.
- message – widok odpowiadający za pisanie i wysyłanie wiadomości do innych użytkowników.

```
message.html.erb x
1 <h1>Wyślij wiadomość do <%= @ciaramba.first_name %> <%= @ciaramba.last_name %></h1>
2
3 <%= link_to 'Powrót', :back%>
4
5 <% if @saveMessage %>
6   <% if @saveMessage.errors %>
7     <ul>
8       <% @saveMessage.errors.full_messages.each do |err| %>
9         <li>
10          <%= err %></li>
11        <% end %>
12      </ul>
13    <% end %>
14  <% end %>
15
16 <%= form_for "Message", :url => { :action => "message" }, :remote => true do |f| %>
17
18   <%= f.label "Treść wiadomości:" %><br>
19   <%= f.text_field "text" %><br>
20
21   <%= f.submit "Wyślij!" %>
22
23
24 <% end %>
```

Rys. 6. Widok message.



- new – ten widok odpowiada za stronę rejestracji użytkowników.

```

1  <h1>Rejestracja</h1>
2
3  <% if @save %>
4    <% if @save.errors %>
5      <ul>
6        <% @save.errors.full_messages.each do |err| %>
7          <li>
8            <%= err %></li>
9          <% end %>
10       </ul>
11     <% end %>
12  <% end %>
13
14  <%= form_for "user", :url => { :action => "new" } do |f| %>
15
16    <%= f.label "login" %><br>
17    <%= f.text_field "login" %><br>
18
19    <%= f.label "password" %><br>
20    <%= f.password_field "password" %><br>
21
22    <%= f.label "first_name" %><br>
23    <%= f.text_field "first_name" %><br>
24
25    <%= f.label "last_name" %><br>
26    <%= f.text_field "last_name" %><br>
27
28    <%= f.label "pesel" %><br>
29    <%= f.text_field "pesel" %><br>
30
31    <%= f.submit %>
32    <p><a href="/user/login">Masz już konto? Zaloguj!</a></p>
33
34  <% end %>

```

Rys. 7. Widok new.

- start – jest to strona startowa aplikacji.

```

1  <h1>Zalogowany jako <%= @first_name %> <%= @last_name %> </h1>
2  <p></p>
3  <p><a href="/user/logout">Wyloguj</a></p>
4  <p><a href="/user/index">Lista Użytkowników</a></p>
5  <p>Wiadomości: </p>
6  <% if @Messages %>
7
8    <table class="tabela">
9      <thead>
10
11        <tr>
12          <td class = "czerwony">Nadawca</td>
13          <td>Treść</td>
14          <td>Data</td>
15        </tr>
16      </thead>
17
18      <tbody>
19        <tr>
20          <td class = "czerwony"><%=h @sender.first_name%> <%=h @sender.last_name %></td>
21          <td><%=h @Messages.text %></td>
22          <td><%=h @Messages.created_at %></td>
23        </tr>
24      </tbody>
25    </table>
26
27  <% end %>

```

Rys. 8. Widok start.

- profile – jest to profil użytkownika, umieszczona jest na nim tabela z rekordami EKG oraz wykresy pojawiające się ponad nią w przypadku wybrania określonego rekordu z tabeli.

```

profile.html.erb
1  <%= javascript_include_tag "//www.google.com/jsapi", "chartkick" %>
2  <h1>Profil użytkownika <%=user.first_name%> <%=user.last_name%></h1>
3  <a><%=signal%></a>
4  <p><a href="/user/index">Powrót</a></p>
5  <p><a href="/user/start">Powrót do strony głównej</a></p>
6  <p><a><%= link_to 'Napisz wiadomość', user_message_path(:ciaramba => params[:ciaramba]) %>
7  </a></p>
8  <%=if @signal%>
9
10     <%= @data_line_chart = [] %>
11     <%= mytime = 0 %>
12     <%= myS = Time.at(mytime).utc.strftime("%S:%L") %>
13     <%= rec = @signal1 %>
14     <%= rec.each do |data| %>
15         <%= table = [] %>
16         <%= table << myS %>
17         <%= table << data - 500 %>
18         <%= mytime += 5 %>
19         <%= myS = Time.at(mytime.to_f/1000).utc.strftime("%S:%L") %>
20         <%= @data_line_chart << table %>
21     <%= end %>
22     <%= size = @data_line_chart.size %>
23     <%= s = size/6 %>
24
25     <p><%= line_chart @data_line_chart[0..s], points: false, xtitle: "czas [s:ms]", ytitle: "amplituda [mV]" %> </p>
26     <p><%= line_chart @data_line_chart[s+1..2*s], points: false, xtitle: "czas [s:ms]", ytitle: "amplituda [mV]" %> </p>
27     <p><%= line_chart @data_line_chart[2*s+1..3*s], points: false, xtitle: "czas [s:ms]", ytitle: "amplituda [mV]" %> </p>
28     <p><%= line_chart @data_line_chart[3*s+1..4*s], points: false, xtitle: "czas [ms]", ytitle: "amplituda [mV]" %> </p>
29     <p><%= line_chart @data_line_chart[4*s+1..5*s], points: false, xtitle: "czas [ms]", ytitle: "amplituda [mV]" %> </p>
30     <p><%= line_chart @data_line_chart[5*s+1..size-1], points: false, xtitle: "czas [ms]", ytitle: "amplituda [mV]" %> </p>
31
32 <%=end%>
33
34 <table class="tabela">
35   <thead>
36
37     <tr>
38       <td class = "czerwony"><%=h sig.id %></td>
39       <td>Data załadowania zapisu</td>
40     </tr>
41   </thead>
42
43   <tbody>
44     <%= @SIG.each do |sig| %>
45       <tr>
46         <td class = "czerwony"><%=h sig.id %></td>
47         <td><%=h sig.created_at %></td>
48         <td><%=h link_to "Wyświetl", {:REC=> sig.id, :ciaramba => params[:ciaramba]} %></td>
49       </tr>
50     <%= end %>
51   </tbody>
52 </table>
53

```

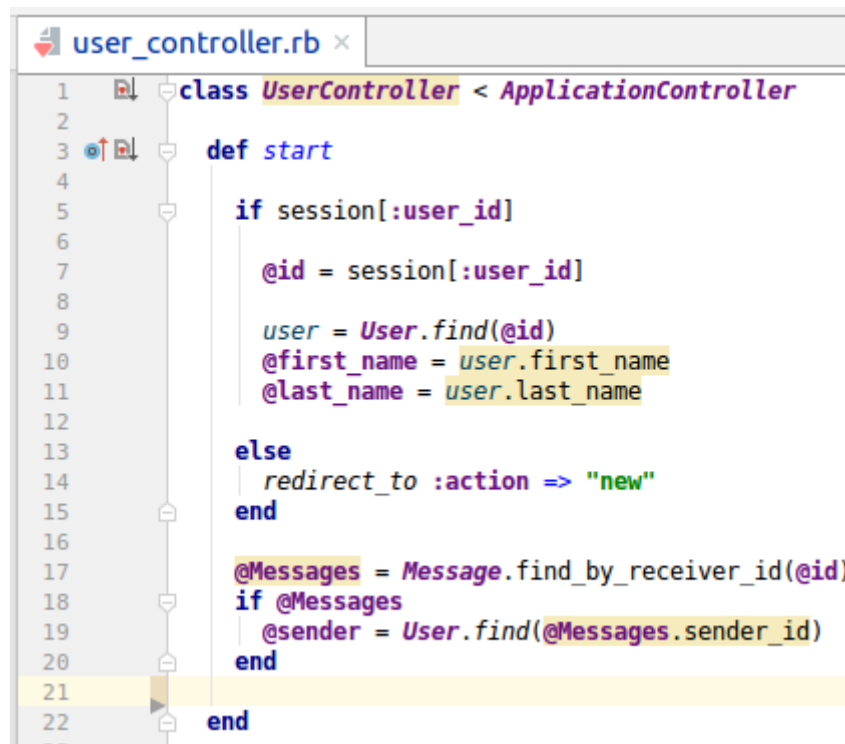
Rys. 9. Widok profile.

- remove – bez zastosowania.

## 5.4. Kontroler – user\_controller

Ze względu na to, że działanie aplikacji opiera się o model User, definicja kontrolerów dla pozostałych modeli była zbędna. Kontroler ten zarządza pracą widoków o analogicznej nazwie akcji zawartej w nim. Owe akcje prezentują się następująco:

- start – kontroler sprawdza czy użytkownik jest zalogowany na podstawie danych z sesji. Jeśli owy nie jest użytkownikiem tylko gościem, zostaje on przekierowany do strony rejestracji (widok new).



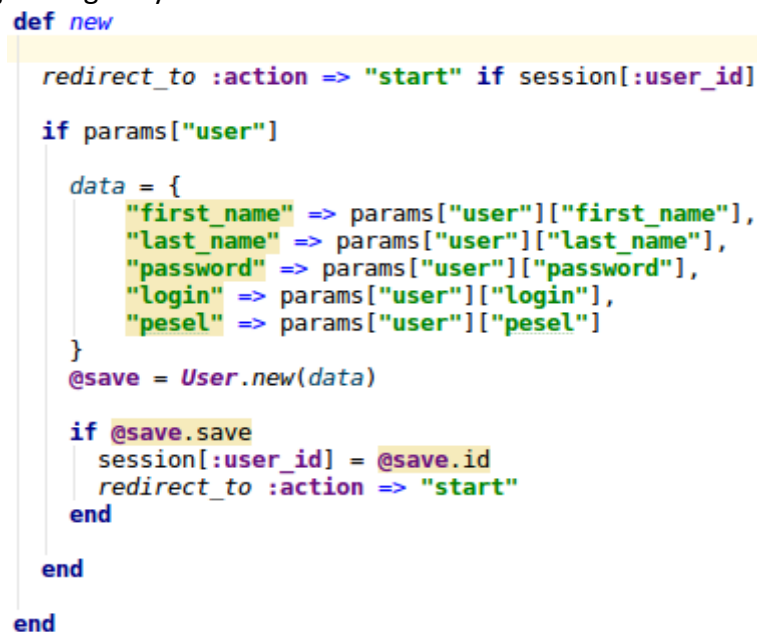
```

1 class UserController < ApplicationController
2
3   def start
4
5     if session[:user_id]
6
7       @id = session[:user_id]
8
9       user = User.find(@id)
10      @first_name = user.first_name
11      @last_name = user.last_name
12
13     else
14       redirect_to :action => "new"
15     end
16
17     @Messages = Message.find_by_receiver_id(@id)
18     if @Messages
19       @sender = User.find(@Messages.sender_id)
20     end
21   end
22 end

```

Rys. 10. Akcja start kontrolera user\_controller.

- new – ta akcja nie pozwala na przebywanie w widoku new jeśli jest się zalogowanym. Odpowiada za przechwycenie danych z widoku i utworzenie nowego rekordu w bazie danych, po czym w przypadku powodzenia operacji przekierowanie do strony startowej nowego użytkownika.



```

def new
  redirect_to :action => "start" if session[:user_id]

  if params["user"]
    data = {
      "first_name" => params["user"]["first_name"],
      "last_name" => params["user"]["last_name"],
      "password" => params["user"]["password"],
      "login" => params["user"]["login"],
      "pesel" => params["user"]["pesel"]
    }
    @save = User.new(data)

    if @save.save
      session[:user_id] = @save.id
      redirect_to :action => "start"
    end
  end
end

```

Rys. 11. Akcja new kontrolera user\_controller.

- remove – bez zastosowania.
- login – tutaj kontroler ponownie przekierowuje zalogowanego użytkownika do strony startowej. Akcja ta odpowiada również za sprawdzenie danych w czasie logowania i odszyfrowanie hasła oraz rozpoczęcie sesji użytkownika zalogowanego.

```
def login
  redirect_to :action => "start" if session[:user_id]
  if params["login"]
    if user = User.find_by_login(params["login"]["login"])
      password = Digest::SHA2.new << params["login"]["password"]
      if password == user.password
        session[:user_id] = user.id
        redirect_to :action => "start"
      end
    end
  end
end
```

Rys. 12. Akcja login kontrolera user\_controller.

- logout – bez zastosowania.
- profile – Akcja zapewniająca dane użytkownika na profilu wraz z jego przebiegami. Parametr przekazywany w aplikacji :ciaramba stanowi numer ID w bazie danych użytkownika, którego podglądamy. Parametr :REC natomiast jest numerem ID jednogodzinowego przebiegu EKG.

```
def profile
  @user = User.find(params[:ciaramba])
  @SIG = Przebiegi.where(user_id: @user.id).all
  if params[:REC]
    @signal = Przebiegi.find(params[:REC])
    @signal1 = @signal.zapis
  end
end
```

Rys. 13. Akcja profile kontrolera user\_controller.

- index – w tym miejscu po raz kolejny sprawdzana jest rola użytkownika, ponieważ tylko lekarze i administrator mają dostęp do listy użytkowników.

```
def index
  id = session[:user_id]
  user = User.find(id)
  if user.patient?
    redirect_to :action => "start", :alert => "Brak uprawnień!"
  end
  @users = User.all
end
```

Rys. 14. Akcja index kontrolera user\_controller.

- edit – ta akcja odpowiada za dopuszczenie jedynie administratora z poziomu listy użytkowników, do której dostęp mają również lekarze. Element przekazywania danych i uaktualnienia jest analogiczny jak w przypadku akcji new.

```
def edit

  redirect_to :action => "index" if !params[:ciaramba]

  id = session[:user_id]
  user = User.find(id)
  if !user.admin?
    redirect_to :action => "index", :alert => "Brak uprawnień!"
  end

  if params[:ciaramba]
    @current_user = User.find(params[:ciaramba])
    @ciaramba = :ciaramba
  end

  if params["edit"]
    data = {
      "first_name" => params["edit"]["first_name"],
      "last_name" => params["edit"]["last_name"],
      "pesel" => params["edit"]["pesel"],
      "role" => params["edit"]["role"]
    }

    @user = User.find(params[:ciaramba])

    if @user.update_attributes(data)
      redirect_to :action => "index"
    else
      render 'edit'
    end
  end
end

end
```

Rys. 15. Akcja edit kontrolera user\_controller.

- message – akcja kierująca zapisem wiadomości do bazy danych.

```
def message

  redirect_to :action => 'profile' if !params[:ciaramba]

  @ciaramba = User.find(params[:ciaramba])

  if params["Message"]
    data = {
      "text" => params["Message"]["text"],
      "sender_id" => session[:user_id],
      "receiver_id" => params[:ciaramba]
    }

    @saveMessage = Message.new(data)

    if @saveMessage
      redirect_to user_profile_path(:ciaramba => params[:ciaramba])
    end
  end

end
```

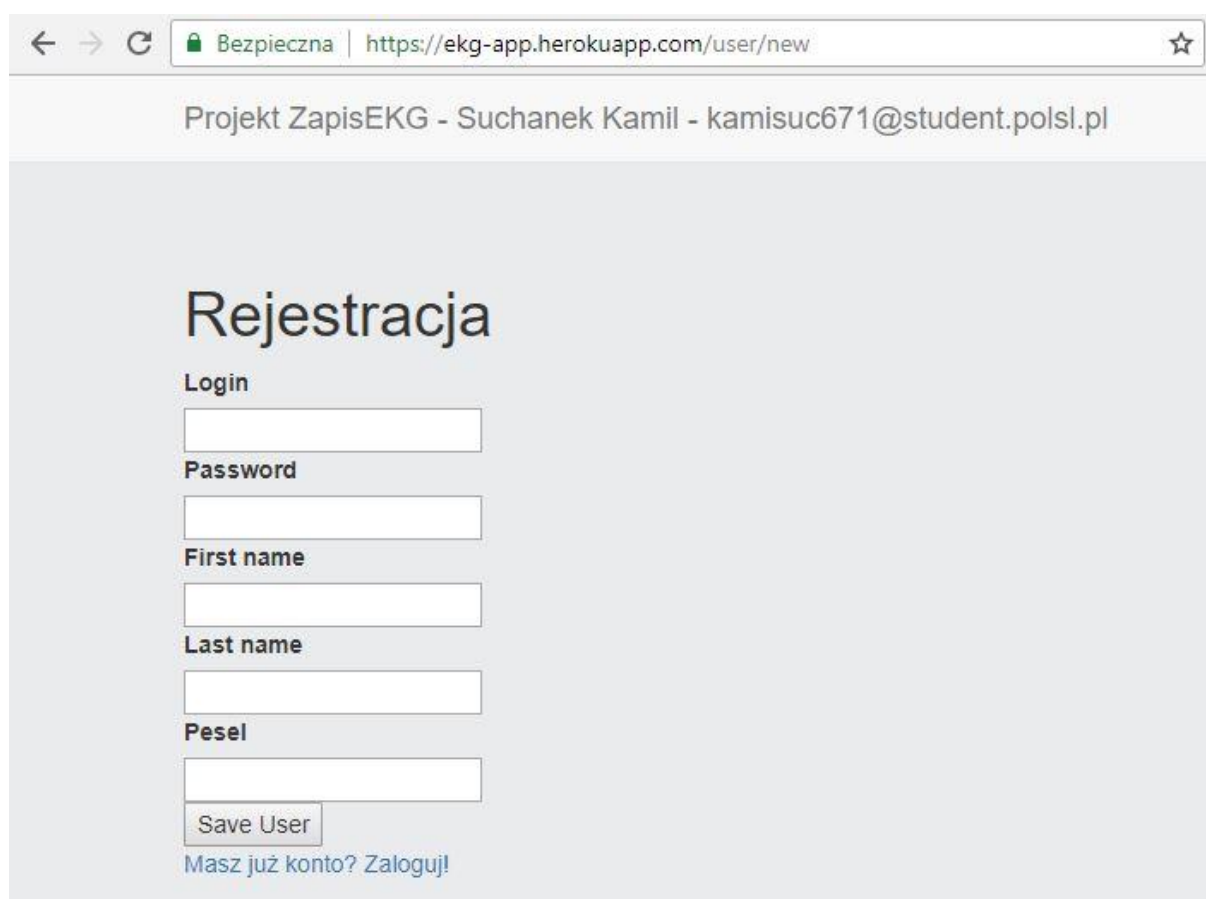
Rys. 16. Akcja message kontrolera user\_controller.

## 5.5. Błędy aplikacji, rozwiązania

Błędy ograniczają się do funkcji edytowania i przesyłania wiadomości pomiędzy użytkownikami. Wyeliminowanie ich polegało na uniemożliwieniu odświeżania strony po zatwierdzeniu zmian w formularzach na poszczególnych stronach, przez co strona od strony wizualnej nie reaguje na nieudane próby przesłania wiadomości i uaktualnienia danych użytkowników i dzięki temu możliwy jest bezawaryjny powrót do poprzednich widoków serwisu. W czasie realizacji o wiele słuszniejszym rozwiązaniem prezentacji wykresów było AmCharts oferujące skalowane wykresy z przydatnymi suwakami, jednak z powodu braku zgrania aplikacji z tym rozwiązaniem trzeba było podzielić rekord na kilka wykresów. Ewentualne dłuższe funkcjonowanie serwisu przyniesie potrzebę zredukowania tabeli rekordów na profilach użytkowników by wyświetlała rekordy z zadanego przedziału czasowego.

## 6. Specyfikacja zewnętrzna

### 6.1. Start aplikacji – zakładanie nowego konta.



← → ↻ Bezpieczna | https://ekg-app.herokuapp.com/user/new ☆

Projekt ZapisEKG - Suchanek Kamil - kamisuc671@student.polsl.pl

## Rejestracja

Login

Password

First name

Last name

Pesel

Save User

[Masz już konto? Zaloguj!](#)

Rys. 17. Etap rejestracji użytkownika aplikacji.

Zakładając konto należy podać unikalny login, hasło zawierające co najmniej 4 znaki, dowolne imię i nazwisko oraz PESEL będący liczbą całkowitą.

The screenshot shows a web browser window with the address bar displaying "Bezpieczna | https://ekg-app.herokuapp.com/user/new". The page title is "Projekt ZapisEKG - Suchanek Kamil - kamisucf". The main heading is "Rejestracja". Below the heading, there is a list of error messages:

- First name can't be blank
- Last name can't be blank
- Pesel can't be blank
- Login can't be blank
- Login is too short (minimum is 4 characters)
- Password is too short (minimum is 4 characters)

The form fields are labeled and empty:

**Login**

**Password**

**First name**

**Last name**

**Pesel**

At the bottom of the form, there is a "Save User" button and a link "Masz już konto? Zaloguj!".

Rys. 18. Reakcja formularza na błędne dane.

Wprowadzenie błędnych danych i zatwierdzenie ich spowoduje pojawienie się komunikatu o tym jakie błędy popełniliśmy (Rys. 18.). Wprowadzenie przykładowych, poprawnych danych (Rys. 19.). Skutek ich zatwierdzenia przyciskiem *Save User* (Rys. 20.) – nowy użytkownik został przekierowany na stronę startową.

## Rejestracja

- First name can't be blank
- Last name can't be blank
- Pesel can't be blank
- Login can't be blank
- Login is too short (minimum is 4 characters)
- Password is too short (minimum is 4 characters)

### Login

### Password

### First name

### Last name

### Pesel

[Masz już konto? Zaloguj!](#)

Rys. 19. Wprowadzenie przykładowych, poprawnych danych do formularza.





Rys. 20. Automatyczne zalogowanie po utworzeniu konta i przekierowanie na stronę startową.

Przestrzeń pod napisem *Wiadomości:* nie zawiera tabelki ze względu na brak wiadomości do wyświetlenia. Przykład wiadomości do wyświetlenia dla innego konta (Rys. 21.).

Wiadomości:

Nadawca	Treść	Data
Kamil Suchanek	hej	2018-09-15 18:44:09 UTC

Rys. 21. Przykładowa wiadomość.

## 6.2. Logowanie

Po wylogowaniu trafiaamy na stronę logowania (Rys. 22.).

Bezpieczna | https://ekg-app.herokuapp.com/user/login

Projekt ZapisEKG - Suchanek Kamil - kamisuc671@student.polsl.pl

## Logowanie

User:

Password:

Zaloguj!

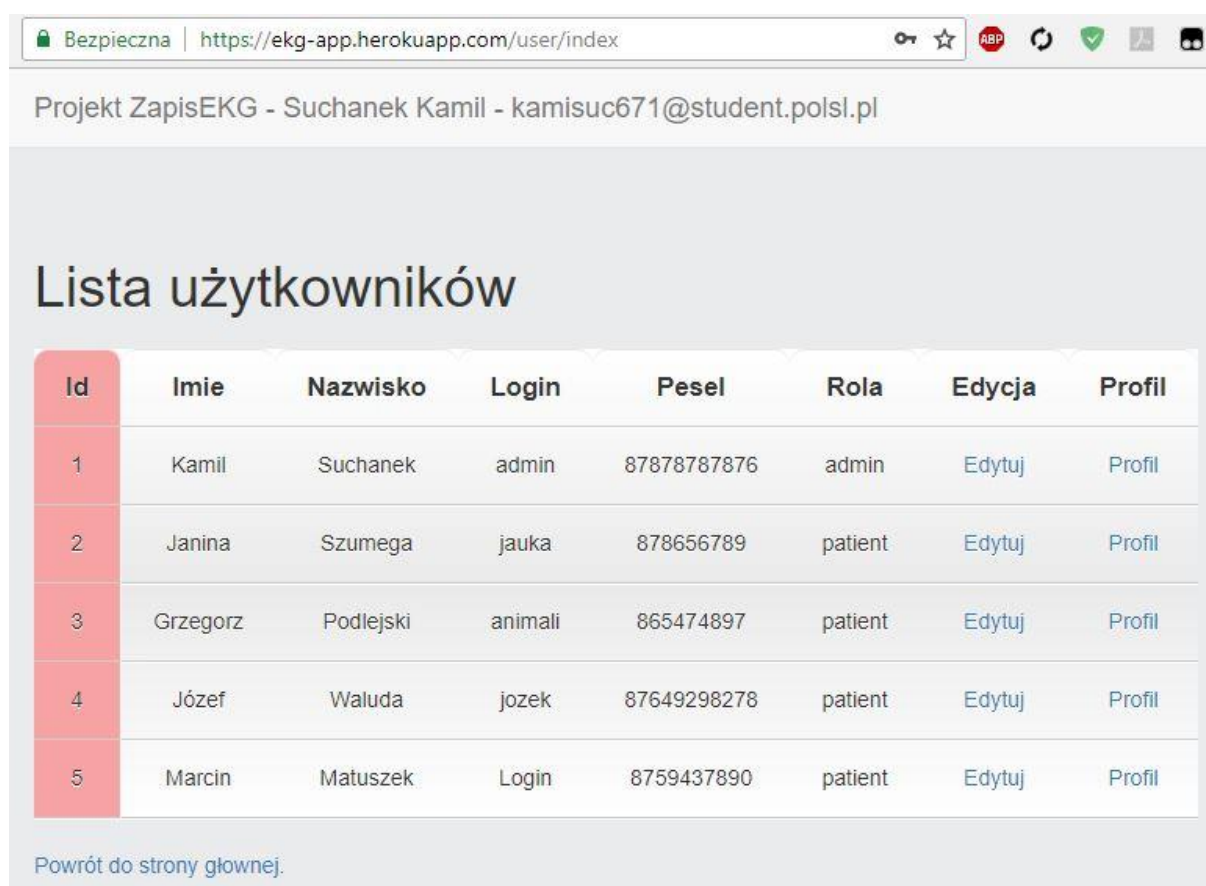
[Zarejestruj się!](#)

Rys. 22. Strona logowania.

Jeśli nie posiadamy konta możemy dostać się do strony rejestracji przy pomocy przycisku *Zarejestruj się!*, a jeśli trafimy na stronę rejestracji a posiadamy konto to należy nacisnąć przycisk *Masz już konto? Zaloguj!* (Rys. 17-19.). Błędne wypełnienie formularza logowania zakończy się odświeżeniem strony logowania i wyczyszczeniem wypełnionego formularza. Po udanym procesie logowania zostaniemy przekierowani na stronę startową (Rys. 20.).

### 6.3. Lista użytkowników, profile, edycja i wiadomości

Konto *Kamil Suchanek* posiada rolę administratora, więc na dostęp do strony, do której przekierowuje link *Lista użytkowników*, podobnie jak użytkownicy z rolą lekarza (Rys. 20.). Po kliknięciu w owy link następuje przekierowanie do strony z listą użytkowników (Rys. 23.).



Id	Imie	Nazwisko	Login	Pesel	Rola	Edycja	Profil
1	Kamil	Suchanek	admin	87878787876	admin	<a href="#">Edytuj</a>	<a href="#">Profil</a>
2	Janina	Szumega	jauka	878656789	patient	<a href="#">Edytuj</a>	<a href="#">Profil</a>
3	Grzegorz	Podlejski	animali	865474897	patient	<a href="#">Edytuj</a>	<a href="#">Profil</a>
4	Józef	Waluda	jozek	87649298278	patient	<a href="#">Edytuj</a>	<a href="#">Profil</a>
5	Marcin	Matuszek	Login	8759437890	patient	<a href="#">Edytuj</a>	<a href="#">Profil</a>

[Powrót do strony głównej.](#)

Rys. 23. Strona z listą użytkowników.

Z tej strony można dostać się do strony edycji i do poszczególnych profili użytkowników. Przykładowy profil wygląda jak na rysunku nr. 24. Żeby wyświetlić przebieg z bazy danych należy kliknąć *Wyświetl* w wybranym wierszu tabeli (Rys. 24.). Wynikiem tej akcji będzie wyświetlenie sześciu wykresów pokazujących po dziesięć sekund zapisu, w celu lepszej widoczności należy rozszerzyć stronę na cały ekran (Rys. 25.). Pod wyświetlonymi sześcioma wykresami w dalszym ciągu znajduje się tabelka, z której można zmienić wyświetlany rekord klikając ponownie *Wyświetl*.

Bezpieczna | https://ekg-app.herokuapp.com/user/profile?ciaramba=1

Projekt ZapisEKG - Suchanek Kamil - kamisuc671@student.polsl.pl

## Profil użytkownika Kamil Suchanek

[Powrót](#)

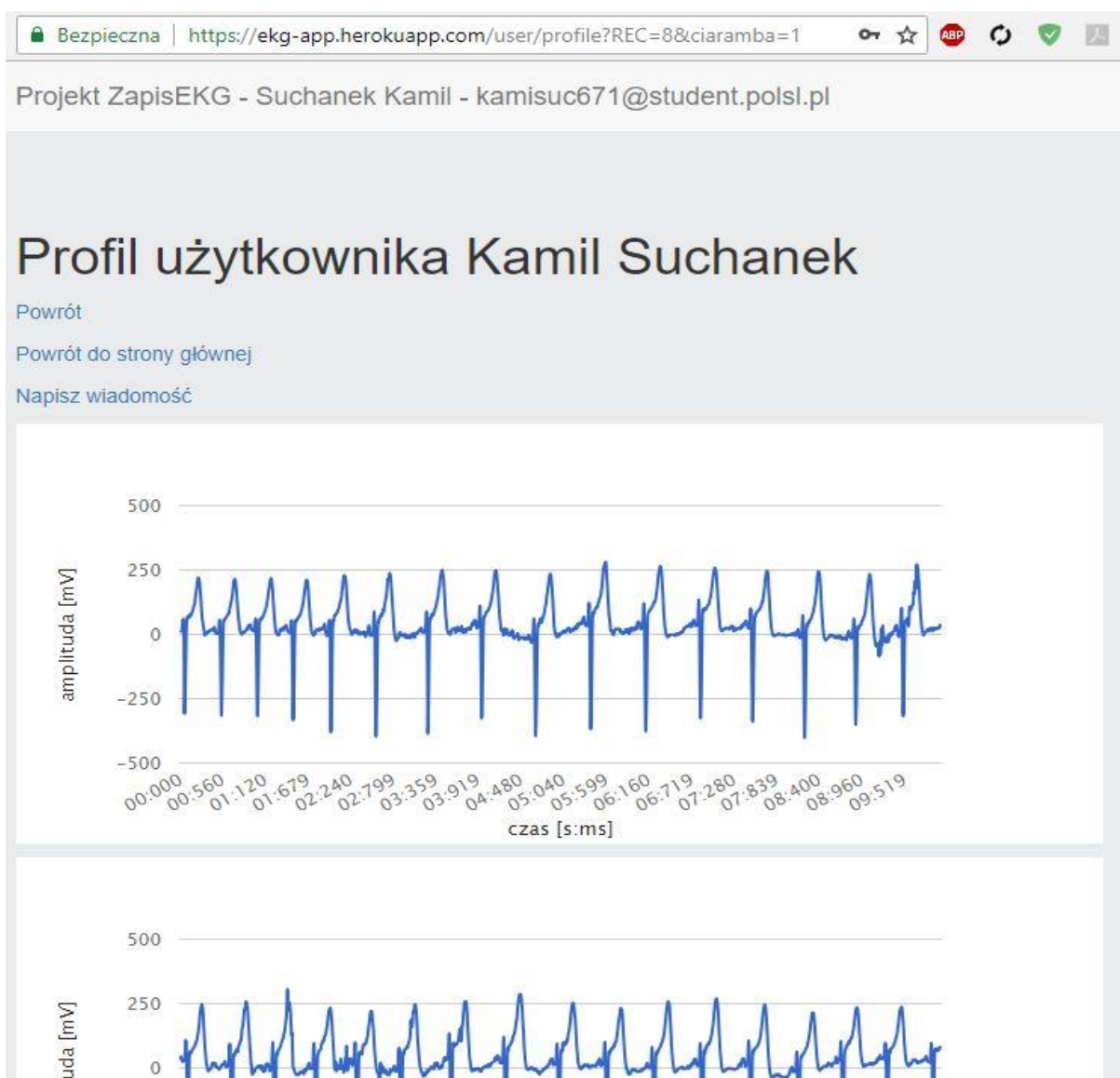
[Powrót do strony głównej](#)

[Napisz wiadomość](#)

Id	Data załadowania zapisu	
8	2018-09-14 14:59:32 UTC	<a href="#">Wyświetl</a>
9	2018-09-14 14:59:35 UTC	<a href="#">Wyświetl</a>
10	2018-09-14 14:59:41 UTC	<a href="#">Wyświetl</a>
11	2018-09-14 14:59:45 UTC	<a href="#">Wyświetl</a>
12	2018-09-14 14:59:49 UTC	<a href="#">Wyświetl</a>
13	2018-09-14 14:59:53 UTC	<a href="#">Wyświetl</a>

Rys. 24. Przykładowy profil użytkownika posiadającego zapisy EKG.

Znajdując się na danym profilu można napisać wiadomość do danego użytkownika klikając link *Napisz wiadomość*, po tym zostaniemy przekierowani do strony pisania wiadomości (Rys. 26.).



Rys. 25. Wykresy na profilu użytkownika.

Bezpieczna | <https://ekg-app.herokuapp.com/user/message?ciaramba=1>

Projekt ZapisEKG - Suchanek Kamil - [kamisuc671@student.polsl.pl](mailto:kamisuc671@student.polsl.pl)

## Wyślij wiadomość do Kamil Suchanek

[Powrót](#)

**Treść wiadomości:**

Rys. 26. Strona pisania wiadomości.

Jeśli wybierzemy opcję Edytuj (Rys. 23.) zostaniemy przekierowani do strony edycji (Rys. 27.).

Bezpieczna | <https://ekg-app.herokuapp.com/user/edit?ciaramba=1>

Projekt ZapisEKG - Suchanek Kamil - [kamisuc671@student.polsl.pl](mailto:kamisuc671@student.polsl.pl)

## Edycja

**Imię**

**Nazwisko**

**Pesel**

**Rola**

[Powrót](#)

Rys. 27. Strona edycji użytkownika.

## 7. Podsumowanie

Utworzona strona internetowa spełnia podstawową funkcjonalność polegającą na wizualizacji przebiegów EKG. Założenie dotyczące komunikowania się lekarza z pacjentem za pośrednictwem serwisu nie zostało spełnione.

## 8. Źródła

1. [www.canva.com](http://www.canva.com).
2. [https://pl.wikipedia.org/wiki/Ruby\\_on\\_Rails](https://pl.wikipedia.org/wiki/Ruby_on_Rails)
3. [https://pl.wikipedia.org/wiki/Monitorowanie\\_EKG\\_metod%C4%85\\_Holtera](https://pl.wikipedia.org/wiki/Monitorowanie_EKG_metod%C4%85_Holtera)
4. <https://en.wikipedia.org/wiki/Heroku>
5. S. Bober, B. Dąbrowska, A. Dąbrowski – „Elektrokardiografia Praktyczna” – PZWL 1971.
6. <https://www.jetbrains.com/ruby/>
7. <https://www.heroku.com/>
8. <https://www.postgresql.org/docs/>
9. <https://ekg-app.herokuapp.com/>
10. <https://github.com/KamilSuchanek95/ZapisEKG>