

MEAN Stack - obsługa poczty elektronicznej, biblioteka Nodemailer

Tworzenie serwisów Web 2.0

dr inż. Robert Perliński
rperlinski@icis.pcz.pl

Politechnika Częstochowska
Instytut Informatyki Teoretycznej i Stosowanej

22 marca 2019

Plan prezentacji

- 1 Obsługa poczty elektronicznej
 - Nodemailer
 - gmail
 - t.pl
 - szablon, rejestracja i aktywacja konta
- 2 Źródła

Biblioteki obsługujące pocztę elektroniczną w Node.js

Biblioteki obsługujące pocztę elektroniczną w Node.js:

- **Nodemailer** - <https://nodemailer.com> - najlżejsza, najbardziej rozbudowana, jedna z nielicznych, która ma własną dokumentację, inną niż na npmjs.com czy github.com,
- **express-mailer** - <https://www.npmjs.com/package/express-mailer>
- **express-mail** - <https://www.npmjs.com/package/express-mail>
- **mail** - <https://www.npmjs.com/package/mail>
- **mail2** - <https://www.npmjs.com/package/mail2>

Przestarzałe, dołączone do biblioteki Nodemailer:

- **mailcomposer** - <https://nodemailer.com/extras/mailcomposer/>, nie pozwala na wysyłanie maili tylko na ich komponowanie i wysyłanie do otwartego już kanału SMTP czy do pliku
- **mailparser** - <https://nodemailer.com/extras/mailparser/>, do parsowania maili, nawet bardzo dużych i w dużych ilościach...



Nodemailer

Easy as cake e-mail sending from your Node.js applications

<https://nodemailer.com>

O bibliotece Nodemailer:

- proste wysyłanie maili (ang. easy as cake - proste jak ...),
- projekt rozpoczął się przed rokiem 2010 kiedy nie było jakiegś sensownej biblioteki to wysyłania poczty,
- dzisiaj Nodemailer jest niemal domyślnie wybierany do obsługi poczty elektronicznej w Node.js,
- obecna wersja ma numer 5.1.1 i jest na licencji MIT,
- istnieje też płatna licencja komercyjna.

Możliwości biblioteki Nodemailer:

- pojedynczy moduł bez żadnych dodatkowych zależności - kod łatwy w zarządzaniu,
- duży nacisk położono na bezpieczeństwo, brak takich usterek jak RCE (Remote Code Execution),
- wsparcie dla Unicode pozwala na użycie dowolnych znaków, również emoji,
- wsparcie dla Windows, wystarczy zainstalować jak dowolny inny moduł,
- treść wiadomości może być tekstowa albo w HTML,
- pozwala na dodawanie załączników,
- możliwość umieszczania obrazów z załącznika w treści HTML wiadomości,

Możliwości biblioteki Nodemailer:

- bezpiecznie przesyłanie wiadomości dzięki TSL/STARTTLS,
- różne metody przesyłania danych poza domyślnym protokołem SMTP,
- możliwość podpisywania wiadomości kluczem DKIM (DomainKeys Identified Mail),
- system wtyczek do modyfikowania wiadomości,
- dodatkowy system autoryzacji OAuth2 korzystający z tokenów,
- obsługa Proxy dla połączeń SMTP,
- obsługa kodu ES6 - brak niezamierzonych wycieków pamięci dzięki obsłudze `var`.

Instalacja i wymagania:

- instalacja: `npm install nodemailer --save`,
- Nodemailer wymaga tylko Node.js w wersji 6+, żadnych innych specyficznych wymagań.

- 1 Tworzymy warstwę transportową używając SMTP (domyślnie) albo jakąś inną.
- 2 Ustawiamy wszystkie dane odnośnie wiadomości (kto, do kogo, treść, załączniki,...).
- 3 Dostarczamy wiadomość używając metody `sendMail()` utworzonej wcześniej warstwy transportowej.

Użycie Nodemailer w skrócie - TL;DR

- 1 Tworzymy warstwę transportową używając SMTP (domyślnie) albo jakąś inną.

```
utils/mailerTransporter.js
```

```
const nodemailer = require('nodemailer');

// tworzy obiekt warstwy transportowej, domyślnie używa protokołu SMTP
let gmailTransporter = nodemailer.createTransport({
  service: 'gmail',
  auth: {
    user: 'rperlinski.pcz@gmail.com',
    pass: 'SuperMegaTajneHaslo'
  }
});

module.exports = { gmailTransporter: gmailTransporter }
```


Użycie Nodemailer w skrócie - TL;DR

- 2 Ustawiamy wszystkie dane odnośnie wiadomości (kto, do kogo, treść, załączniki,...).

index.js

```
router.get('/send-email', function(req, res, next) {
  // wysyła wiadomości, dane mogą być w unicode
  let mailOptions = {
    // adres nadawcy
    from: '"Robert Perliński" <rperlinski.pcz@gmail.com>',
    // lista odbiorców
    to: 'pierwszy.adres@gmail.com, rperlinski@icis.pcz.pl',
    // temat wiadomości
    subject: 'Testowa wiadomość',
    // treść wiadomości tekstowej
    text: 'Treść wiadomości jako czysty tekst',
    // treść wiadomości w html
    html: '<b>Treść wiadomości w HTML</b>
    <p>Jakieś znaki w unicode, grecki alfabet:  $\alpha$ ,  $\beta$ ,  $\gamma$ , ...</p>'
  };
  ...
});
```

Użycie Nodemailer w skrócie - TL;DR

- 3 Dostarczamy wiadomość używając metody `sendMail()` utworzonej wcześniej warstwy transportowej.

index.js

```
router.get('/send-email', function(req, res, next) {  
  ...  
  
  // wysła maila dla ustawionej warstwy transportowej dla danych opcji  
  transporter.sendMail(mailOptions, (error, info) => {  
    if (error) {  
      return console.log(error);  
    }  
    console.log('Wiadomość %s wysłana: %s',  
      info.messageId, info.response);  
    res.send('Wiadomość ' + info.messageId +  
      ' wysłana: ' + info.response);  
  });  
});
```

Wiadomość <51c06368-d687-ef1a-5785-c581e9db0a36@gmail.com>
wysłana: 250 2.0.0 OK 1490142828 a16sm39369551fk.24 - gsmtpt

Odebrany mail

Odebrany mail w formie HTML:

From Robert Perliński ☆

Subject **Testowa wiadomość**

To Jan Kowalski ☆, Me <rperlinski@icis.pcz.pl> ☆

Treść wiadomości jak HTML

Jakieś znaki w unicode, grecki alfabet: α, β, γ, ...

Odebrany mail w formie tekstowej

Subject: Testowa wiadomość
From: Robert Perliński <rperlinski.pcz@gmail.com>
Date: Wed, March 22, 2017
To: [REDACTED] (less)
rperlinski@icis.pcz.pl
Priority: Normal
Options: [View Full Header](#) | [View Printable Version](#) | [Download this as a file](#)

Treść wiadomości jako czysty tekst

Używanie gmail'a

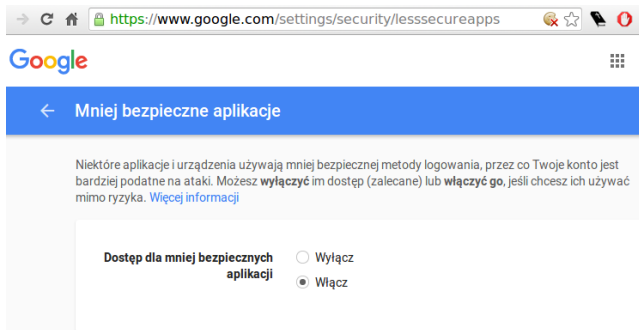
Gmail'a może działać od razu albo może wymagać dodatkowych ustawień:

- Gmail oczekuje prawdziwego użytkownika, nie robota; wykorzystuje różne heurystyki aby zapobiec logowaniu przez roboty,
- Gmail ma opcję `less secure`;
<https://support.google.com/accounts/answer/6010255?hl=pl>, która pozwala na dostęp każdemu kto tylko zna hasło,
- może być potrzeba również wypełnienie kodu weryfikacyjnego (Captcha) zanim będziemy mogli wysyłać maile:
<https://accounts.google.com/b/0/displayunlockcaptcha>
- Gmail zastępuje też zawsze nadawcę wiadomości danymi uwierzytelnionego użytkownika,
- w celu uniknięcia problemów z logowaniem należy użyć autoryzacji przez token (OAuth2) albo innego dostawcy.

Używanie gmail'a - wyłączenie less secure

Opcję less secure można ustawić tutaj:

<https://www.google.com/settings/security/lesssecureapps>



The screenshot shows a web browser window with the address bar containing the URL <https://www.google.com/settings/security/lesssecureapps>. The page title is "Mniej bezpieczne aplikacje" (Less secure apps). The main content area contains a warning: "Niektóre aplikacje i urządzenia używają mniej bezpiecznej metody logowania, przez co Twoje konto jest bardziej podatne na ataki. Możesz **wyłączyć** im dostęp (zalecane) lub **włączyć go**, jeśli chcesz ich używać mimo ryzyka. [Więcej informacji](#)". Below this, there is a toggle switch for "Dostęp dla mniej bezpiecznych aplikacji" (Access for less secure apps), which is currently set to "Włącz" (On).




Używanie gmail'a - wyłączenie less secure

Opcja less secure w koncie google:


W informacji o naszym koncie wybieramy Logowanie się i zabezpieczenia:

Kontroluj i chroń swoje konto – wszystko w jednym miejscu

Moje konto zapewnia szybki dostęp do ustawień i narzędzi, które pozwolą Ci chronić Twoje dane i prywatność oraz określić, w jaki sposób Twoje informacje wykorzystane, by lepiej dopasować usługi Google do Twoich potrzeb.

 Logowanie się i zabezpieczenia >	 Dane osobowe i prywatność >	 Ustawienia konta
Zarządzaj hasłami i ustawieniami dostępu do konta Google. Logowanie się w Google Aktywność urządzeń i powiadomienia Połączone aplikacje i witryny	Zarządzaj swoimi ustawieniami widoczności i danymi, których używamy, by dostosować nasze usługi do Twoich potrzeb. Twoje informacje osobowe Zarządzaj aktywnością w Google Ustawienia reklam	Skonfiguruj język, ułatwienia dostępu i ustawienia, które ułatwią Ci korzystanie z Google. Język i narzędzia wprowadzania tekstu Ułatwienia dostępu Twoje miejsce na Dysku Google

Opcja na samym dole:

Zezwalaj na mniej bezpieczne aplikacje: **WŁĄCZONE** 

Niektóre aplikacje i urządzenia korzystają z mniej bezpiecznych technologii logowania, co może obniżać bezpieczeństwo Twojego konta. Możesz wyłączyć dostęp do takich aplikacji (i to zalecamy) lub nadal ich używać mimo zagrożeń.



Darmowa poczta elektroniczna email

Poczta email t.pl oferuje fajne, zgrabne adresy typu ty@t.pl. Nasza poczta elektroniczna posiada dużą pojemność i pozwala przysyłać duże załączniki. Dodatkowo darmowa poczta t.pl posiada ochronę antyspamową, dostęp przez www i program pocztowy.

[zaloguj się](#)[założ konto](#)

- Fajny, zgrabny adres typu ty@t.pl
- Duża pojemność
- Obsługa dużych załączników
- Ochrona antyspamowa
- Dostęp przez www i program pocztowy

Przykład dla poczty t.pl - warstwa transportowa

Przygotowanie warstwy transportowej:

utils/mailerTransporter.js

```
...
// Serwer poczty przychodzącej: POP3, t.pl STARTTLS
// Serwer poczty wychodzącej: SMTP, t.pl, bez szyfrowania
let selfSignedConfig = {
  host: 't.pl',
  port: 465,
  secure: true, // używa TLS
  auth: {
    user: 'web20@t.pl', pass: '*****'
  },
  tls: {
    // nie przerywa przy błędnym certyfikacie
    rejectUnauthorized: false
  }
};
let tTransporter = nodemailer.createTransport(selfSignedConfig);

module.exports = { gmailTransporter: gmailTransporter,
                  tTransporter: tTransporter };
```


Przykład dla poczty t.pl - wysyłanie

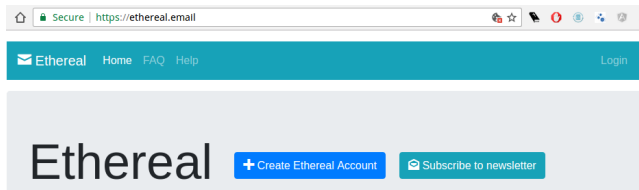
Wysyłanie wiadomości:

routes/index.js

```
router.get('/send-email3', function(req, res, next) {
  let mailOptions = {
    from: '"Student Web 2.0" <web20@t.pl>', // adres nadawcy
    to: 'asdf@asdf.com, jkowalski@icis.pcz.pl', // lista odbiorców
    subject: 'Testowa wiadomość', // temat wiadomości
    text: 'Treść wiadomości jako czysty tekst', // treść wiadomości tekstowej
    // treść w HTML

    html: '<b>Treść wiadomości jak HTML</b>
    <p>Jakieś znaki w unicode, grecki alfabet:  $\alpha$ ,  $\beta$ ,  $\gamma$ , ...</p>';
  };
  // wysła maila dla ustawionej warstwy transportowej dla danych opcji
  tTransporter.sendMail(mailOptions, (error, info) => {
    if (error) {
      return console.log(error);
    }
    console.log('Wiadomość %s wysłana: %s', info.messageId, info.response);
    res.send('Wiadomość ' + info.messageId + ' wysłana: ' + info.response);
  });
});
```

Usługa Eternal.email



Eternal.email:

- sztuczna usługa SMTP, zwykle używana przez użytkowników Nodemailera
- pozwala za darmo "wysłać" fikcyjne maile i zobaczyć jak się prezentują
- pozwala to na zweryfikowanie treści i układu maila czy całej ich grupy
- wysłane/odebrane maile można przejrzeć dowolnym klientem email, również za pomocą przeglądarki

Usługa Eternal.email I

Eternal.email, konfiguracja warstwy transportowej:

utils/mailerTransporter.js

```
...  
  
let eTransporter = nodemailer.createTransport({  
  host: 'smtp.ethereal.email',  
  port: 587,  
  auth: {  
    user: 'kpa6xtekmqbbohkw@ethereal.email',  
    pass: 'FRuUkYStPeTgme6Xt9'  
  }  
});  
  
module.exports = { gmailTransporter: gmailTransporter,  
  tTransporter: tTransporter,  
  eTransporter: eTransporter,  
  getTestMessageUrl: nodemailer.getTestMessageUrl };
```

Usługa Eternal.email II

Eternal.email, wysyłanie maila:

```
routes/index.js
```

```
var eTransporter = require('../utils/mailerTransporter').eTransporter;
var getTestMessageUrl = require('../utils/mailerTransporter').getTestMessageUrl;
...

router.get('/send-email4', function(req, res, next) {
  let mailOptions = {
    from: '"Jack W." <jackw@mydomain.com>', // adres nadawcy
    to: 'asdf@asdf.com, jkowalski@icis.pcz.pl', // lista odbiorców
    subject: 'Testowa wiadomość - ethernet.email', // temat wiadomości
    text: 'Treść wiadomości jako czysty tekst', // wiadomość tekstowa
    html: '<b>Treść wiadomości jak HTML</b> // wiadomość HTML
    <p>Jakieś znaki w unicode, grecki alfabet:  $\alpha$ ,  $\beta$ ,  $\gamma$ , ...</p>'
  };
  eTransporter.sendMail(mailOptions, (error, info) => {
    if (error) { return console.log(error); }
    var wiad = 'Wiadomość $info.messageId wysłana: $info.response.<br> Dostępna
      pod <a href="$getTestMessageUrl(info)">$getTestMessageUrl(info)</a>';
    res.send(wiad);
  });
});
```

Usługa Eternal.email III

Wynik w przeglądarce



Przegląd maili online:

← → ↻ 🏠 🔒 Secure | <https://ethereal.email/message/WrnClogVtwoumt.oWrnIkLeHA0PjtOIGAAAABi3e3uioP53npJwl.6LzE1c>

Ethereal Home FAQ Help Messages


Headers **Envelope** Source

Subject: Testowa wiadomość dla eternal.email
From: Jack W. <jackw@mydomain.com>
To: <rperlinski.pcz@gmail.com>, <rperlinski@icis.pcz.pl>
Time: Today at 6:29
Message-ID: <79d29c51-2549-504b-31af-14ea2346e392@mydomain.com>

HTML Plaintext

Treść wiadomości jak HTML


Jakieś znaki w unicode, grecki alfabet: α, β, γ, ...



Usługa Eternal.email IV

Przegląd maili online po zalogowaniu:

Secure | <https://etereal.email/messages> ☆ 🖨️ 🔍 🌐 🔄 🗑️

 **Ethereal** [Home](#) [FAQ](#) [Help](#) [Messages](#) [Logout](#)

Messages for kpa6xtekmqbbohkw@etereal.email Page 1 [← Newer](#) [Older →](#)

To: < rperlinski.pcz@gmail.com >, < rperlinski@icis.pcz.pl >	Testowa wiadomość dla etereal.email	Today at 6:29
To: < rperlinski.pcz@gmail.com >, < rperlinski@icis.pcz.pl >	Testowa wiadomość dla etereal.email	Today at 6:28
To: < rperlinski.pcz@gmail.com >, < rperlinski@icis.pcz.pl >	Testowa wiadomość dla etereal.email	Today at 6:23
To: < rperlinski.pcz@gmail.com >, < rperlinski@icis.pcz.pl >	Testowa wiadomość dla etereal.email	Today at 6:23
To: < rperlinski.pcz@gmail.com >, < rperlinski@icis.pcz.pl >	Testowa wiadomość dla etereal.email	Today at 6:19
To: < rperlinski.pcz@gmail.com >, < rperlinski@icis.pcz.pl >	Testowa wiadomość dla etereal.email	Today at 6:08

© 2017 Ethereal Email info@etereal.email | [service status](#)

Tworzenie konta i jego aktywacja

Wymagania w stosunku do aplikacji:

- formularz rejestracji zawierający przynajmniej **adres email i hasło**
- wysyłanie maila z linkiem aktywacyjnym do użytkownika po jego rejestracji
- link aktywacyjny aktywuje konto użytkownika
- użytkownik może się logować, korzystać z dodatkowych uprawnień

utils/db.pug

```
...
// schemat dokumentu opisującego użytkowników w kolekcji users
var userSchema = new mongoose.Schema({
  username: { type: String, required: true, unique: true},
  password: { type: String, required: true},
  email: { type: String, required: true, unique: true},
  active: {type: Boolean, default: false}
});
```

Dwóch domyślnie utworzonych użytkowników:

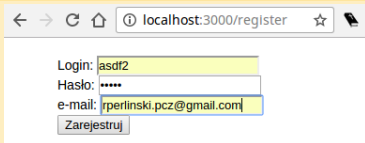
← → ↻ 🏠 ⓘ localhost:3000/users

```
[{"active":true,"_id":"5ab9e4268154e25e57a068c1","username":"admin","pa
{"active":false,"_id":"5ab9e4268154e25e57a068c2","username":"asdf","pas
```

Formularz rejestracji

views/register.pug

```
block content
  form(action="/rejestracja", method="post")
    div
      label Login:
      input(type="text" name="username")
    div
      label Hasło:
      input(type="password" name="password")
    div
      label e-mail:
      input(type="email" name="email")
    div
      input(type="submit" value="Zarejestruj")
```



← → ↻ 🏠 ⓘ localhost:3000/register ☆ 🖨

Login: asdf2

Hasło:

e-mail: rperlinski.pcz@gmail.com

Zarejestruj

views/register.pug

```
router.get('/register', function(req, res) {
  res.render('register');
});
```


Rejestracja użytkownika - obsługa formularza

routes/index.js

```
router.post('/rejestracja', function(req, res) {
  let mailOptions = {
    from: "MyApp Team" <web20@t.pl>', // adres nadawcy
    to: req.body.email,                // lista odbiorców
  };

  // tworzenie użytkownika i wysyłanie maila...
  var newUser = new User({"username":req.body.username, "password":sha1(req.body.password),
    "email": req.body.email, "active": false});
  newUser.save(function(err,data) {
    if (err) return console.error(err);
    var host = "http://localhost:3000";
    // var host2 = "http://myapp.com";
    var aL = host + '/activate/' + data._id;

    res.render('mailBody', {activationLink: aL}, function(err, body){
      mailOptions.html = body;
      mailOptions.subject = 'Aktywacja konta';
      console.log(body);
      tTransporter.sendMail(mailOptions, (error, info) => {
        if (error) { return console.log(error); }
        res.send('
          <h3>Utworzony użytkownik:</h3> ${data}<br>
          Wiadomość o ID ${info.messageId} wysłana: ${info.response}.
        ');
      });
    });
  });
});
```

Treść maila generowana z szablonu

- Tworzymy szablon zawierający oczekiwaną treść maila, np. `mailBody.pug`.
- Renderujemy szablon i na jego zawartości wywołujemy funkcję, w której wyślemy maila.
- Wysyłamy maila zastępując treść wiadomości danymi zrenderowanymi z szablonu.

```
views/mailBody.pug
```

```
h3 Informacje
```

```
p
```

```
| Dziękujemy za utworzenie konta na naszej stronie.
```

```
| Poniżej znajdziesz link aktywacyjny do twojego konta.
```

```
h4 Aktywacja konta
```

```
p Link aktywacyjny:
```

```
a(href=activationLink)= activationLink
```

```
p Your App Team &copy; 2018
```

Mail aktywacyjny - wysyłanie maila z szablonu

routes/index.js

```
router.get('/send-email2', function(req, res, next) {
  var aL = 'http://myapp.com/activate/36899d041ce8968435625b52b0ea827b5f08d645';

  res.render('mailBody', {activationLink: aL}, function(err, body){
    mailOptions.html = body;
    mailOptions.subject = 'Account activation';
    console.log(body);

    transporter.sendMail(mailOptions, (error, info) => {
      if (error) {
        return console.log(error);
      }
      console.log('Wiadomość %s wysłana: %s', info.messageId, info.response);
      res.send('Wiadomość ' + info.messageId + ' wysłana: ' + info.response);
    });
  });
});
```

Wiadomość <126d04e2-6356-c356-71e2-19c0c7607983@gmail.com>
wysłana: 250 2.0.0 OK 1490148792 193sm38992861jj.4 - gsmtip

Mail aktywacyjny - wynik

Treść maila:

<h3>Informacje</h3>

<p>Dziękujemy za utworzenie konta na naszej stronie.

Poniżej znajdziesz link aktywacyjny do twojego konta.

</p>

<h4>Aktywacja konta</h4>

<p>Link aktywacyjny:

http://localhost:3000/activate/5ab9e4328154e25e57a068c3

</p>

<p>Your App Team © 2018</p>

Aktywacja konta Odebrane x



MyApp Team <web20@t.pl>

do mnie ▾

Informacje

Dziękujemy za utworzenie konta na naszej stronie. Poniżej znajdziesz link aktywacyjny do twojego konta.

Aktywacja konta

Link aktywacyjny:

<http://localhost:3000/activate/5ab9e4328154e25e57a068c3>

Your App Team © 2018

Mail:

Rejestracja użytkownika - informacja w przeglądarce

Wynik rejestracji użytkownika w systemie:



Utworzony użytkownik:

```
{ active: false, _id: 5ab9e4328154e25e57a068c3, username: 'asdf2', password:
'c9c35480ccbf411c0ae8d26e3023986b92f87f12', email: 'rperlinski.pcz@gmail.com', __v: 0 }
Wiadomość o ID <461432ec-d29f-b3b8-c69a-79b8d3612ecb@t.pl> wysłana: 250 2.0.0 Ok:
queued as 9D839801A493A.
```

Aktywacja konta użytkownika

Kod aktywujący konto:

routes/index.js

```
// aktywowanie użytkownika o podanym w parametrze ID
router.get('/activate/:id', function(req, res, next) {
  User.findByIdAndUpdate( req.params.id, { $set:{active:true} },
                          {new: false}, function(err, data) {
    res.send("Aktywacja konta użytkownika: " + req.params.id + " ### " + data);
  });
});
```

localhost:3000/activate/5ab9e4328154e25e57a068c3

Aktuwacja konta użytkownika: 5ab9e4328154e25e57a068c3 ### { active: false, _id: 5ab9e4328154e25e57a068c3, username: 'asdf2', password: 'c9c35480ccbf411c0ae8d26e3023986b92f87f12', email: 'rperlinski.pcz@gmail.com', __v: 0 }

Dane użytkowników po aktywacji konta:

localhost:3000/users

```
[{"active":true,"_id":"5ab9e4268154e25e57a068c1","username":"admin","p
{"active":false,"_id":"5ab9e4268154e25e57a068c2","username":"asdf","pa
{"active":true,"_id":"5ab9e4328154e25e57a068c3","username":"asdf2","pa
```

Wymagane zmiany w uwierzytelnieniu

Dodatkowa metoda sprawdzająca uaktywnienie konta:

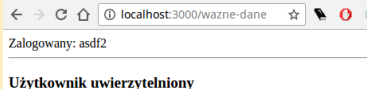
utils/db.js

```
...
userSchema.methods.accountActivate = function() {
  return this.active;
};
```

Mała zmiana w lokalnej strategii uwierzytelnienia:

app.js

```
passport.use(new LocalStrategy(
  function(username, password, done) {
    User.findOne({ username: username }, function (err, user) {
      if (err) { return done(err); }
      if (!user) {
        return done(null, false, { message: 'Incorrect username.' });
      }
      if (!user.validPassword(password)) {
        return done(null, false, { message: 'Incorrect password.' });
      }
      if (!user.accountActivate()) {
        return done(null, false, { message: 'Account not activated.' });
      }
      return done(null, user);
    });
  }
));
```



- <http://mongoosejs.com/>
- <https://nodemailer.com>
- <https://www.google.com/gmail/>
- <http://t.pl/>
- <https://blog.ragingflame.co.za/2012/6/28/simple-form-handling-with-express-and-nodemailer>
- <http://www.ryanray.me/sending-emails-with-jade-node-js-and-nodemailer>