


DEFINICJA ENCJI NA POTRZEBY ORM

W ramach aplikacji korzystać docelowo będziemy z bazy RethinkDB i przygotowanej dla niej na potrzeby **Node.js** biblioteki OR/M Thinkagain.

Pierwszym krokiem będzie instalacja Thinkagain co robimy standardowo z użyciem **npm**:

 Cmder

```
C:\Users\toor\workspace\ccgallery
npm install -g thinkagain
+ thinkagain@0.1.6
added 7 packages in 6.9s

C:\Users\toor\workspace\ccgallery
|
```

Może się okazać przy uruchomieniu projektu, że biblioteka jest niewidoczna z jego poziomu, co objawia się następującym komunikatem:






```
C:\Users\toor\workspace\ccgallery
swagger project start ccgallery
Starting: C:\Users\toor\workspace\ccgallery\ccgallery\app.js...
project started here: http://localhost:10010/
project will restart on changes.
to restart at any time, enter `rs`
Error initializing middleware
Error: Cannot find module 'thinkagain'
    at Function.Module._resolveFilename (module.js:536:15)
    at Function.Module._load (module.js:466:25)
    at Module.require (module.js:579:17)
    at require (internal/module.js:11:18)
    at Object.<anonymous> (C:\Users\toor\workspace\ccgallery\ccgallery\api\model\model.js:1:82)
    at Module._compile (module.js:635:30)
    at Object.Module._extensions..js (module.js:646:10)
    at Module.load (module.js:554:32)
    at tryModuleLoad (module.js:497:12)
    at Function.Module._load (module.js:489:3)
```

W celu eliminacji tego problemu wywołujemy polecenie *npm link thinkagain*:

```
C:\Users\toor\workspace\ccgallery
npm link thinkagain
C:\Users\toor\workspace\ccgallery\node_modules\thinkagain -> C:\Users\toor\AppData\Roaming\npm\node_modules\thinkagain
```

Kolejnym krokiem będzie zdefiniowanie modelu danych na potrzeby naszego projektu. Skorzystamy w tym celu z otwartego standardu **json-schema**: <http://json-schema.org/>

Nasze definicje zostaną umieszczone w pliku *model.js* w katalogu *model* (konieczne jest jego utworzenie jako podkatalogu dla *api*):

 controllers	09.11.2017 11:38	Folder plików
 helpers	08.11.2017 18:37	Folder plików
 mocks	08.11.2017 18:37	Folder plików
 model	10.11.2017 11:15	Folder plików
 swagger	08.11.2017 18:37	Folder plików

Plik modelu zawiera definicję wszystkich klas i związków pomiędzy nimi. Przykładowo dla klasy **Image** może on wyglądać następująco:

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

// Image

var Image = thinkagain.createModel('Image', {
  type: 'object',
  properties: {
    id: { type: 'string' },
    title: { type: 'string' },
    description: { type: 'string' },
    date: { type: 'string', format: 'date-time' },
    path: { type: 'string' },
    size: { type: 'integer' }
  },
  required: [ 'title', 'path' ]
});

exports.Image = Image;
```

Dodajemy następnie do naszego kontrolera stosowną deklarację w celu dołączenia modelu:

```
var model = require('../model/model.js');
```

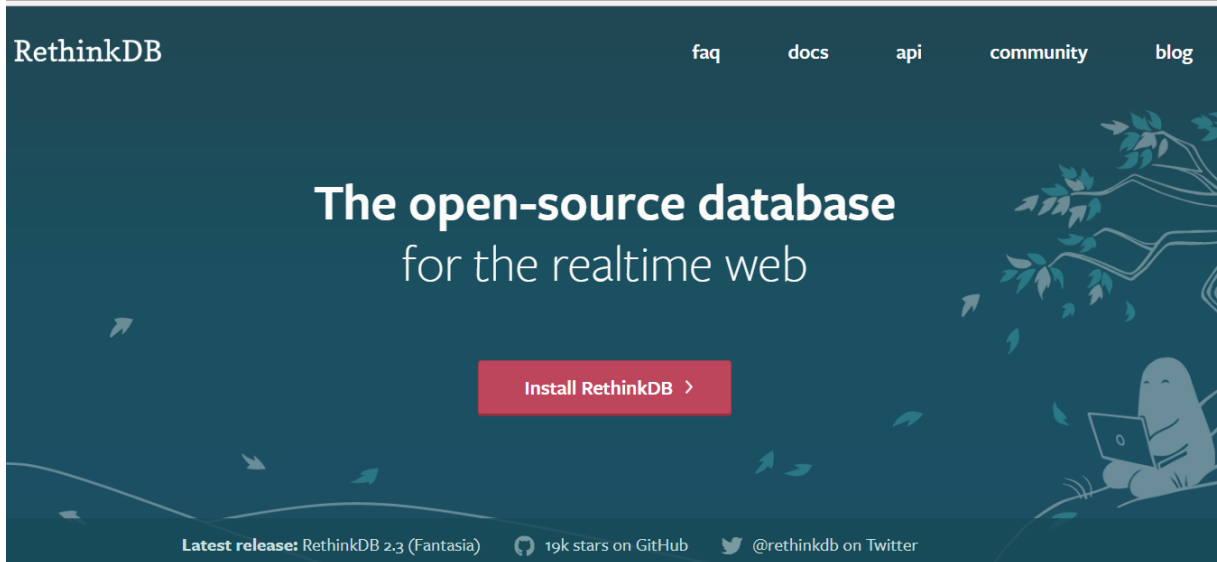
Utworzenie testowego obiektu wygląda następująco:

```
var testData;
testData = new model.Image({
  id: "0123456789abcd",
  title: "Testowy obrazek",
  description: "Opis do obrazka",
  date: "2017-11-09T10:20:00.214Z",
  path: "/library/images/",
  size: 1024
});
```

INSTALACJA RETHINKDB

Pobieramy ze strony <https://rethinkdb.com> plik odpowiedni dla naszego systemu i rozpakowujemy w wybranej lokacji.

<https://rethinkdb.com>

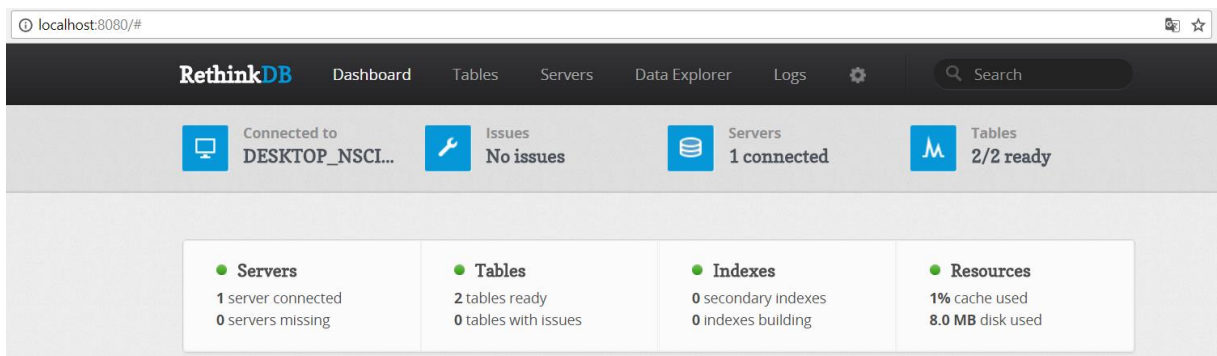


Następnie uruchamiamy demona serwera:

```
C:\Users\toor\workspace\rethinkdb-2.3.6\rethinkdb-2.3.6
ls
rethinkdb.exe* rethinkdb_data/

C:\Users\toor\workspace\rethinkdb-2.3.6\rethinkdb-2.3.6
rethinkdb
Running rethinkdb 2.3.6-windows (MSC 190024215)...
Running on 6.2.9200 (Windows 8, Server 2012)
Loading data from directory C:\Users\toor\workspace\rethinkdb-2.3.6\rethinkdb-2.3.6\rethinkdb_data
Listening for intracluster connections on port 29015
Listening for client driver connections on port 28015
Listening for administrative HTTP connections on port 8080
Listening on cluster address: 127.0.0.1
Listening on driver address: 127.0.0.1
Listening on http address: 127.0.0.1
To fully expose RethinkDB on the network, bind to all addresses by running rethinkdb with the `--bind all` command line option.
Server ready, "DESKTOP_NSCIKDB_zhn" c9baba68-45cf-44a1-902a-ba0313e9681a
```

Jeżeli wszystko udało się to powinno być możliwe otworenie strony zarządzania serwerem pod adresem: <http://localhost:8080>:



ZADANIE

Zdefiniuj klasę, która będzie wykorzystywana na potrzeby galerii **Gallery** grupujących obrazy. Połączenie pomiędzy klasami modeli realizuje się za pomocą metody *belongsTo*:

```
Image.belongsTo(Gallery, 'gallery', 'idGallery', 'id');
```

O co należy uzupełnić klasę **Image**, aby takie połączenie było możliwe.