**Project Title:**

THE FOF-DESIGNER:

DIGITAL DESIGN SKILLS FOR FACTORIES OF THE FUTURE

**Project Acronym:**

**DigiFoF**

Project logo

**Grant Agreement number:**

**2018-2533 / 001-001**

**Project Nr. 601089-EPP-1-2018-1-RO-EPPKA2-KA**

**Subject:**

D2.4 BROKERAGE SYSTEM

**Dissemination Level:**

PUBLIC

**Lead Organization:**

BOC

**Project Coordinator:**

ULBS

**Contributors:**

All partners

**Reviewers:**

\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Revision | Preparation date | Period covered | Project start date | Project duration |
| V1 | May 2019 | Month 1-5 | 01/01/2019 | 36 Months |
| V2 | September 2019 | Month 1-9 | 01/01/2019 | 36 Months |
| flag_yellow_lowThis project has received funding from the European Union’s EACEA Erasmus+ Programme Key Action 2 - Knowledge Alliances under the Grant Agreement No 2018-2533 / 001-001 | | | | |

Table of content

[1 EXECUTIVE SUMMARY 3](#_Toc20157257)

[2 Brokerage system access data and installation guide 4](#_Toc20157258)

[I. Back-end deployment 4](#_Toc20157259)

[II. Frond-end deployment 10](#_Toc20157260)

[3 Conclusion 13](#_Toc20157261)

# EXECUTIVE SUMMARY

This deliverable accompanies the release of the version 2 of DigiFoF Brokerage system, which builds on DigiFoF web platform provided in Deliverable 2.3. The information provided is hence mainly technical in nature. The purpose is to provide the reader with the information on installation prerequisites, access data to Brokerage system files as well as installation guide.

The content on this deliverable can also be accessed at:

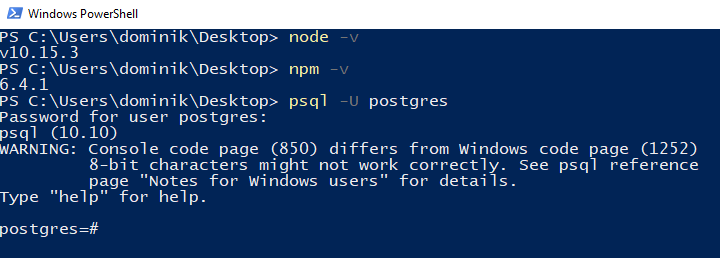
<http://boc-pl.com/digifof/D2.4/v2/brokeragemanual.html>

# Brokerage system access data and installation guide

This tutorial describes all necessary steps required to deploy the brokerage system as a part of your Laboratory web platform (provided in Deliverable 2.3). Because of technical reasons, this deliverable contains also content of web platform (Deliverable 2.3), i.e. after deploying content of this deliverable both web platform and brokerage system will be installed.

The brokerage system is using an open source micro-service architecture based on Olive made available by OMiLAB[[1]](#footnote-1). Main component of web-platform is developed with Open Source[[2]](#footnote-2) Strapi headless CMS.

Download brokerage system files from

<http://boc-pl.com/digifof/D2.4/v2/DigiFoF-webportal-brokerage-v2.zip>  
This manual is based on Windows 10 (also tested in Ubuntu 18)  
Prerequisites to download and install before deployment:  
-latest version of node (tested 10.15.3)  
-latest version of npm (tested 6.4.1)  
-latest version of postgresql (tested 10.10)  
  
Manual consists of two parts: back-end and front-end.

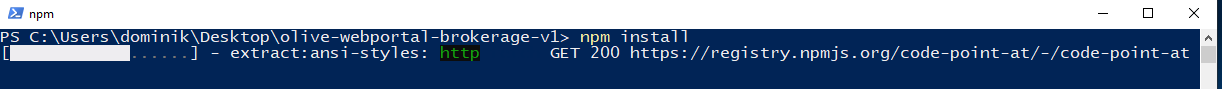
## I. Back-end deployment

**I.1 Extract provided project archive or clone project repository (if applicable).**

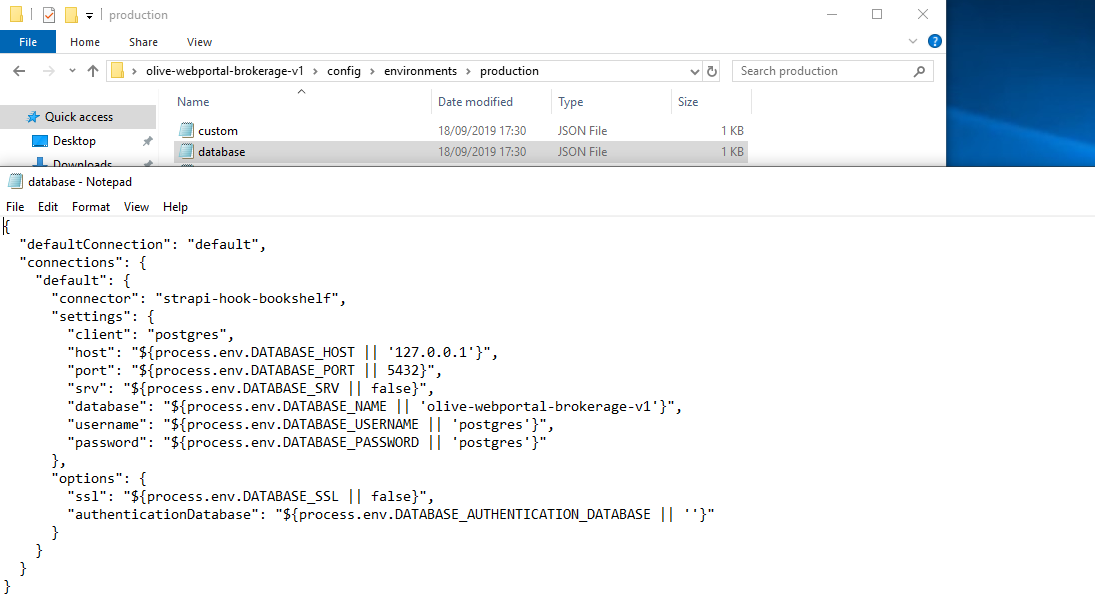
**I.2 Open command prompt/windows powershell/terminal inside project folder**

**I.3 run:**

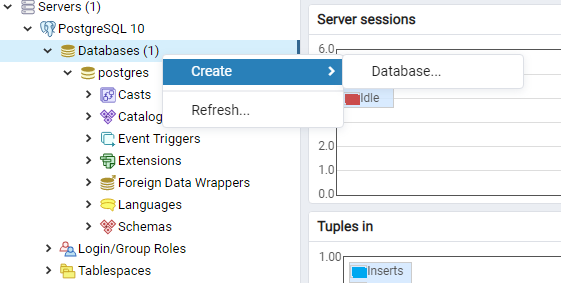
npm install

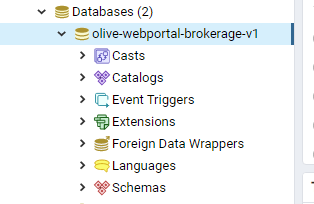
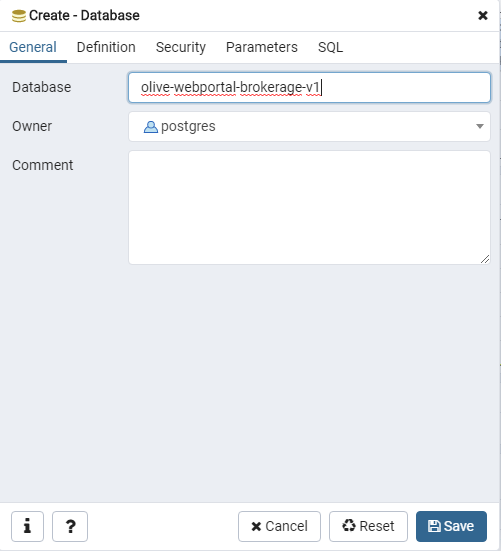


**I.4 Adjust config for database connection (set user/pass/port/host as environmental variables) or type it in '\config\environments\production\database.json'.**



Note, one of the configurations is the name of the database. It should be created beforehand (for example using pgadmin gui, that comes default with postgres installation).





Database user set in config has to have all the rights to the database.

Environmental variables names:

process.env.SITE\_SECRET (secret key needed to validate captcha for user registrations)

alternatively in ‘\middlewares\validateUser\index.js’

process.env.PORT (internal port of application, 1337 by default)

process.env.PROXY\_ENABLE (enable proxy support)

process.env.PROXY\_SSL (enable ssl support)

process.env.PROXY\_HOST (host, where your app is at)

process.env.PROXY\_PORT (port of hosted app)

alternatively in '\config\environments\production\server.json'.

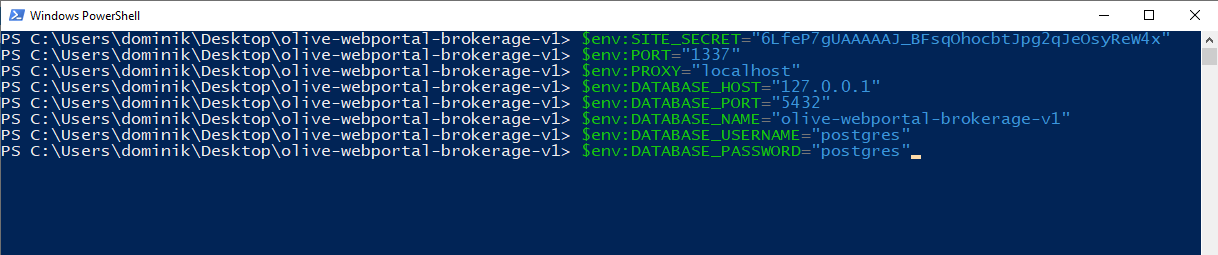
process.env.DATABASE\_HOST

process.env.DATABASE\_PORT

process.env.DATABASE\_NAME

process.env.DATABASE\_USERNAME

alternatively in '\config\environments\production\database.json'.



You need to provide proxy config in '\config\environments\production\server.json', so that account confirmation emails would point to proper url (replace „host” value with your own).

For testing and hosting on localhost, just remove config part below:

"proxy": {

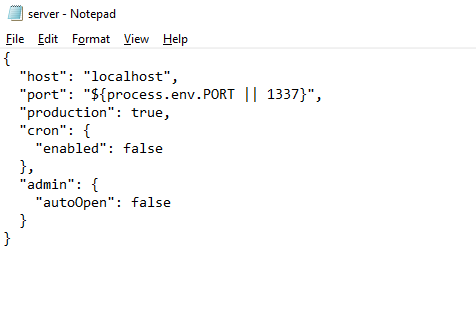
"enabled": "${process.env.PROXY\_ENABLE || true}",

"ssl": "${process.env.PROXY\_SSL || true}",

"host": "${process.env.PROXY\_HOST || 'digifof-brokerage-cms.herokuapp.com'}" ,

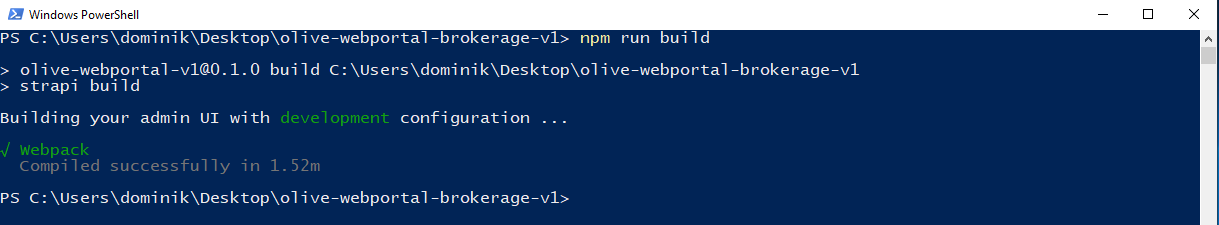
"port": "${process.env.PROXY\_PORT || 443}"

},

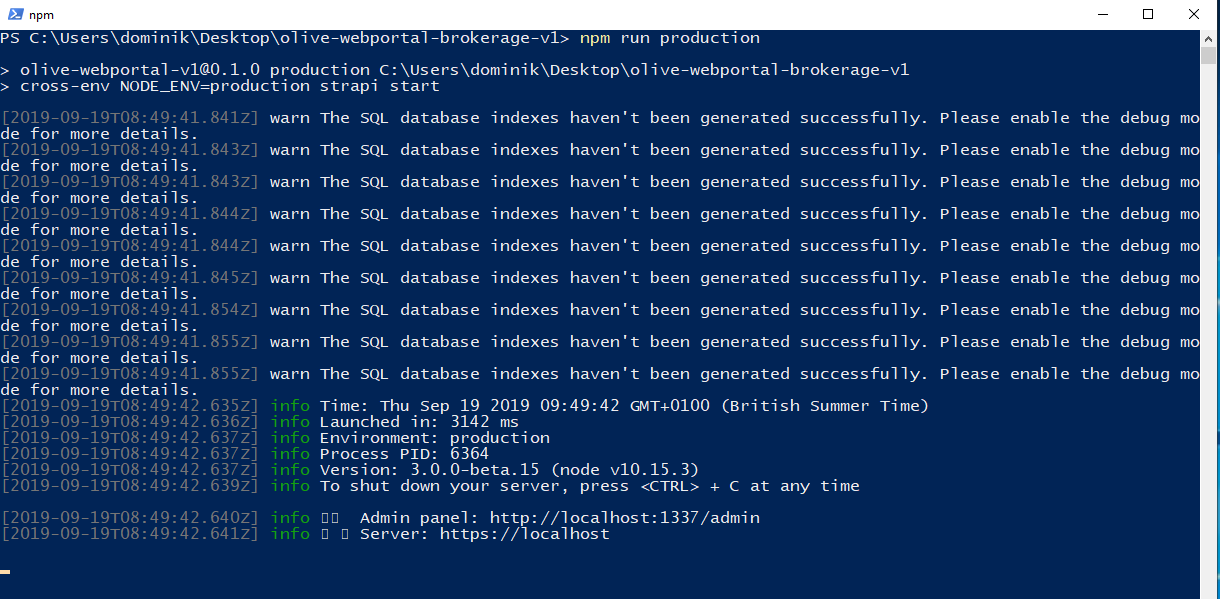


**I.5 run:**

npm run build



npm run production



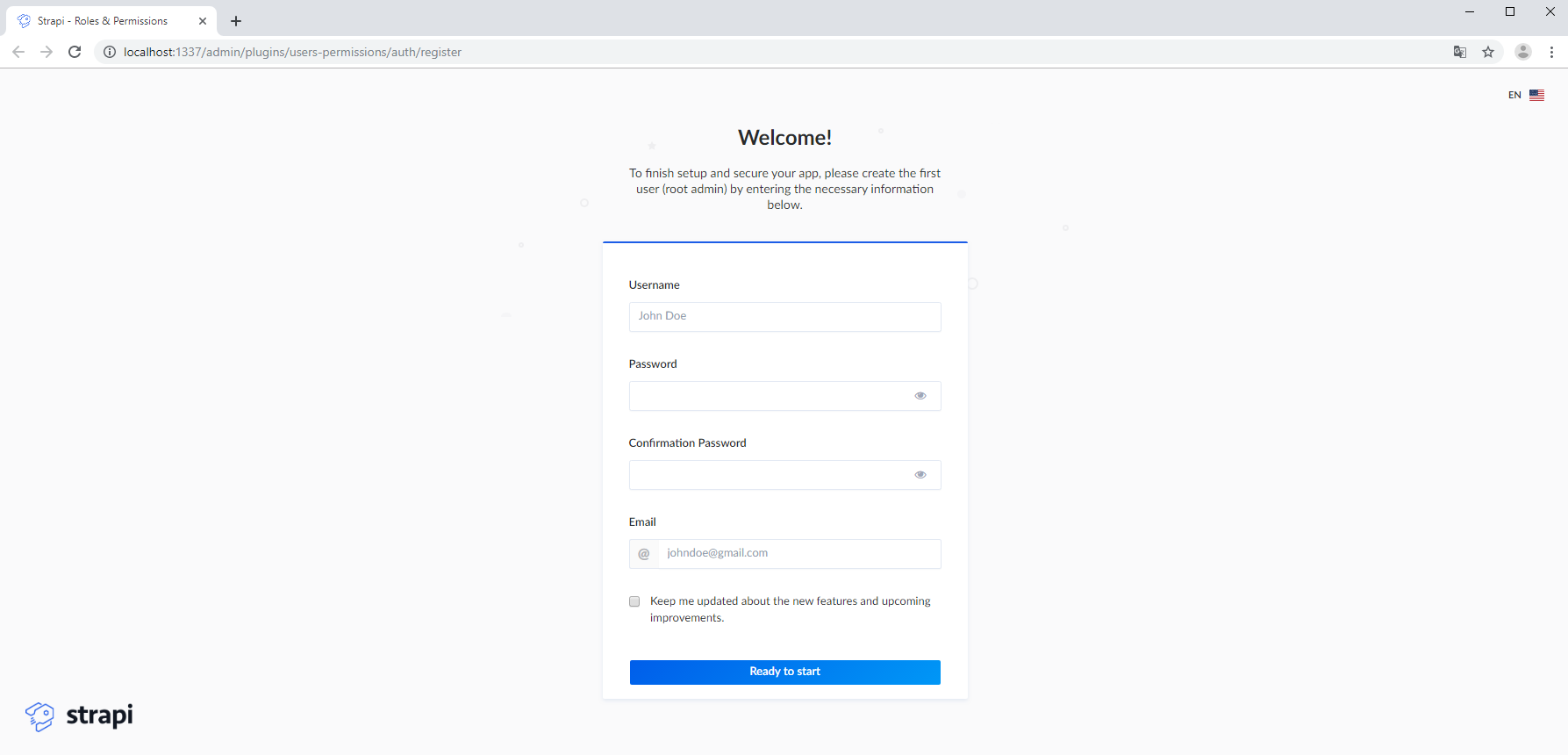
By default, your application will run on localhost, port 1337 and your admin dashboard would be available at:

<http://localhost:1337/admin>

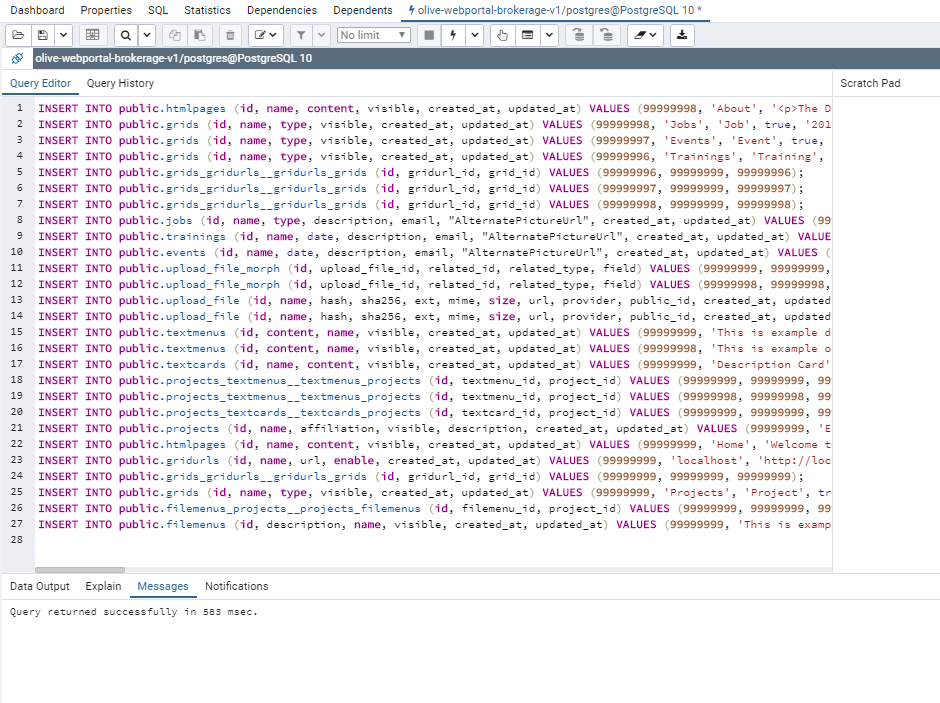
Above is configurable in '\config\environments\production\server.json' - also environmental variable is an option.

To be able to receive proper account confirmation emails, please also fill in your proxy settings if host is different from localhost.

**I.6 Now, after accessing admin panel, you can create your administrator account.**



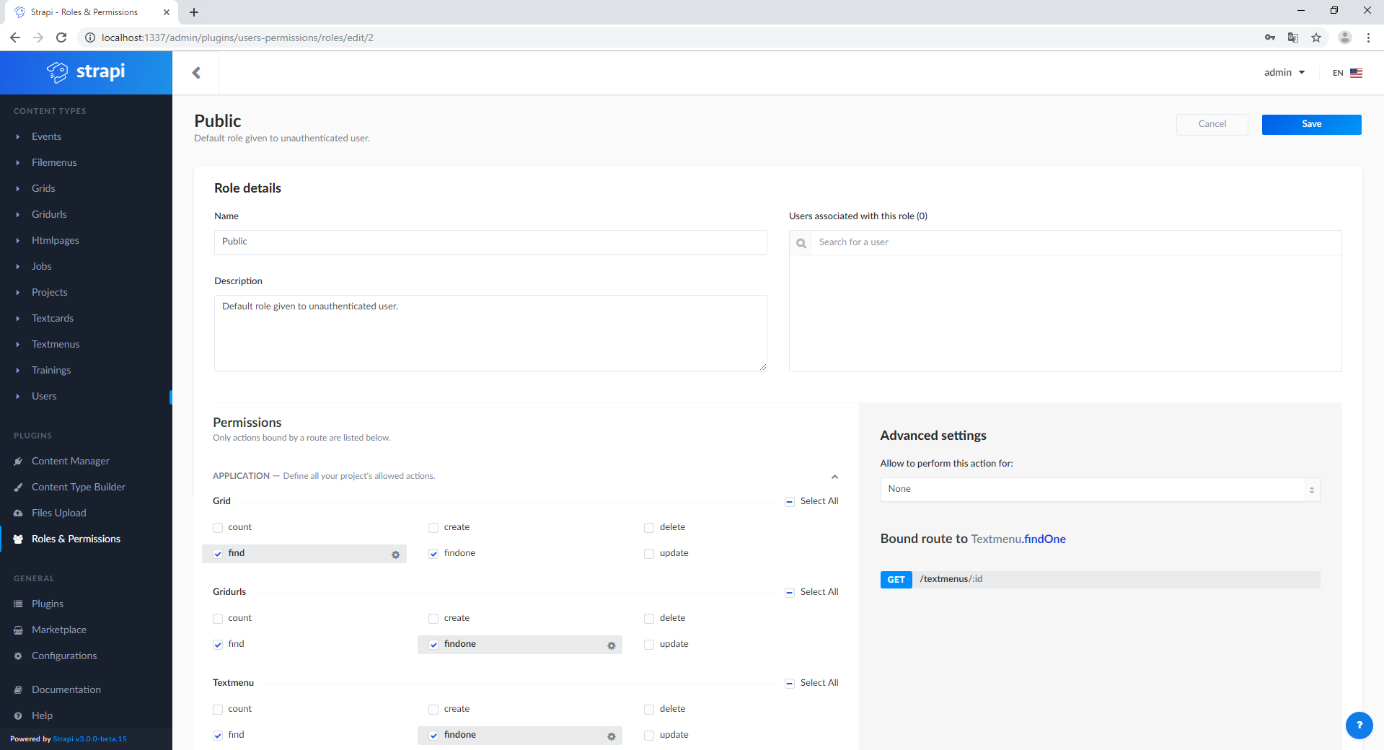
If you would like to use migration package, to see example content, now is the time, when database is empty - just run provided sql commands (migrationbrokerage.sql) on the database you created.



Create your admin account and log in.

**I.7 Set user permissions to view content, that you would like to display (this could be done in Plugins->Roles&Permissions->Public)**

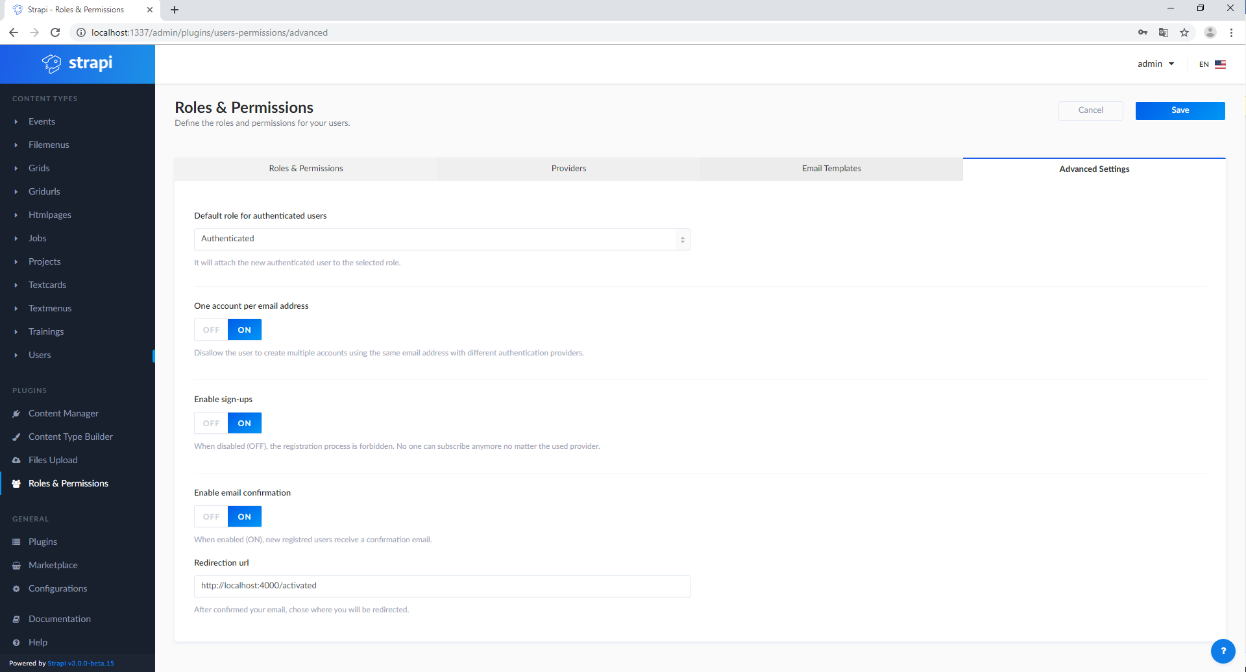
As a default, you should set permission to 'find' and 'findone' for all public roles for all predefined content types (remember about saving – button in top right corner).



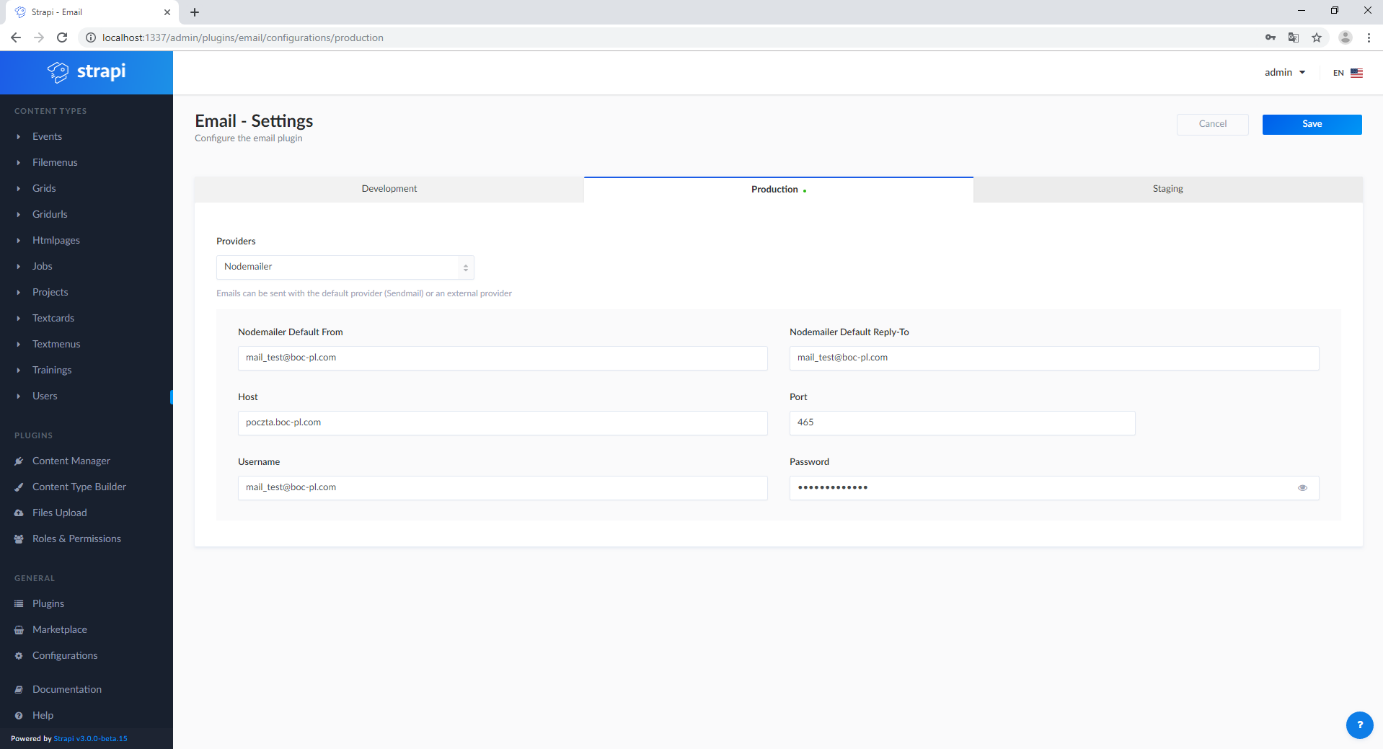
**I.8 Turn on new user account confirmation in Plugins->Roles&Permissions->Advanced Settings**

Also provide redirection url to your front-end application, so users could be taken there after account confirmation (eg. http://localhost:4000/activated).

Remember, that after registering and signing in, users will get default role 'Authenticated'. Don't forget to give them proper permissions, same as you did with 'Public' role or they will be not able to see any content after signing in!



**I.9 Adjust your email server settings if you would like to recover passwords and confirm user accounts: plugins->email(cogwheel icon)->production.**



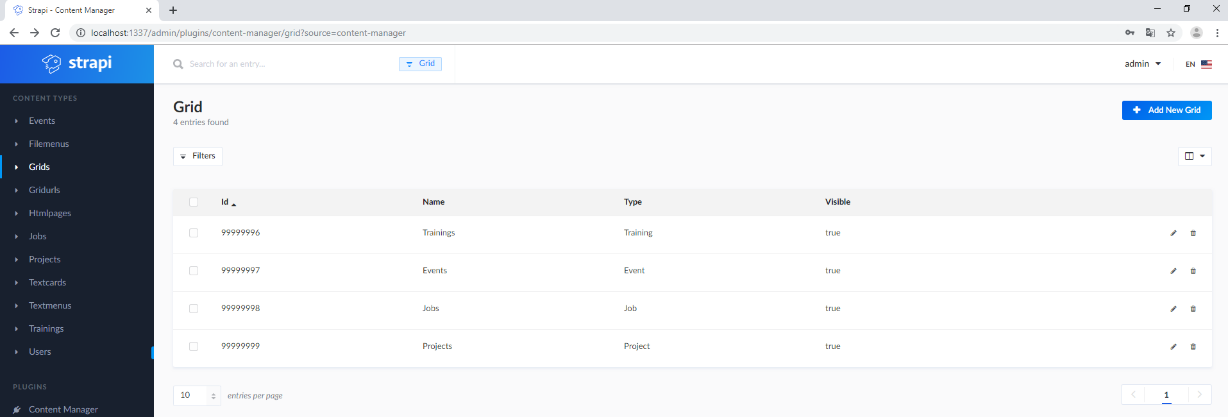
**I.10 Content creation**

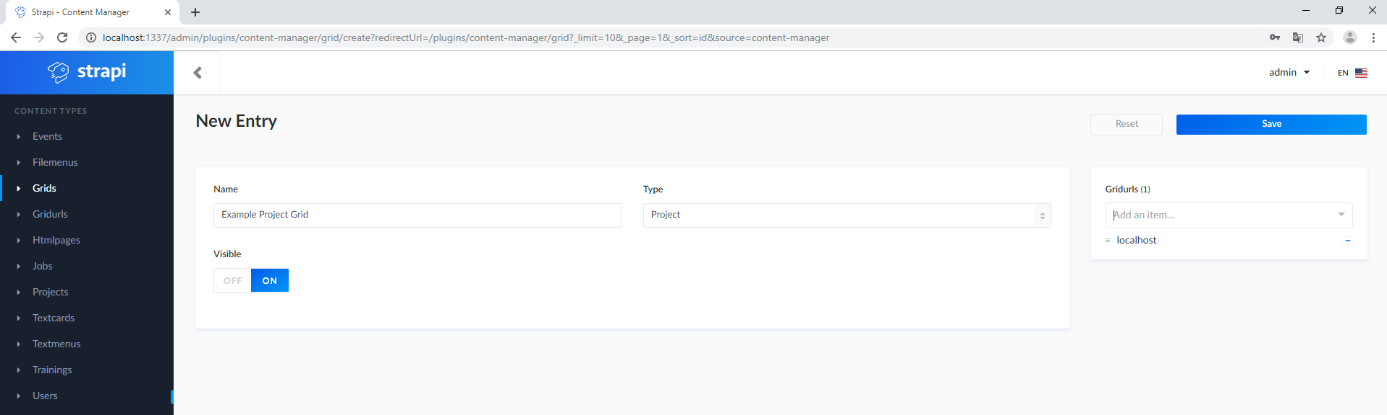
New content could be created, by selecting names of content types and clicking 'Add New [contenttype]' button.

After filling in required fields and clicking 'Save', new content item will be created in database.

Some fields are relations between content types. One of more important relation examples is one between 'Grid' and 'Gridurl' - to be able to use 'Grid' properly, you must create one or more 'Gridurl' items and link it with desired 'Grid' item, so the 'Grid' component in front-end application would know, where to look for data.

Please use unique names for htmlpages and grids - otherwise you will run into unexpected behaviour in frontend.





These are the most basic configuration options for back-end part of the portal.

Since web platform is created using Strapi headless cms, for more advanced configuration options, you could look at source documentation: <https://strapi.io/documentation/3.0.0-beta.x/>

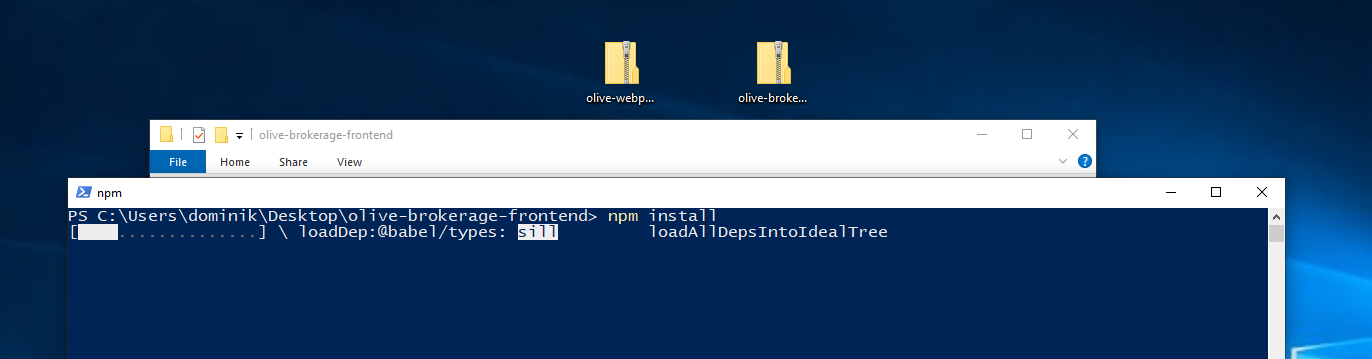
## II. Frond-end deployment

**II.1 Extract provided project archive or clone project repository (if applicable).**

**II.2 Open command prompt/windows powershell/terminal inside project folder**

**II.3 run:**

npm install

****

**II.4 Adjust config in src/config.js file:**

Most importantly backendurl and frontendurl - to enable communication between elements of portal.

You can also adjust styling for select components such as header or footer.

Sometimes you are not limited to styling options provided- it might be possible to add new ones.

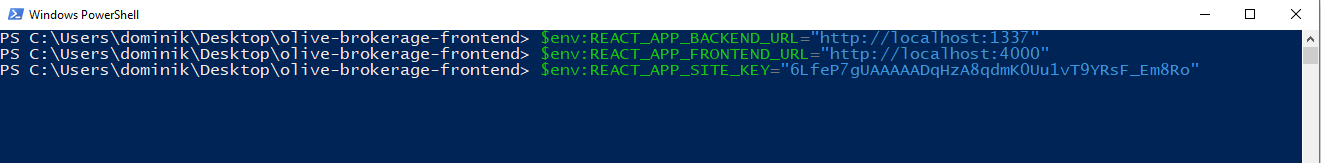


There is also possibility to adjust most important options using process env variables:

process.env.REACT\_APP\_BACKEND\_URL

process.env.REACT\_APP\_FRONTEND\_URL

process.env.REACT\_APP\_SITE\_KEY

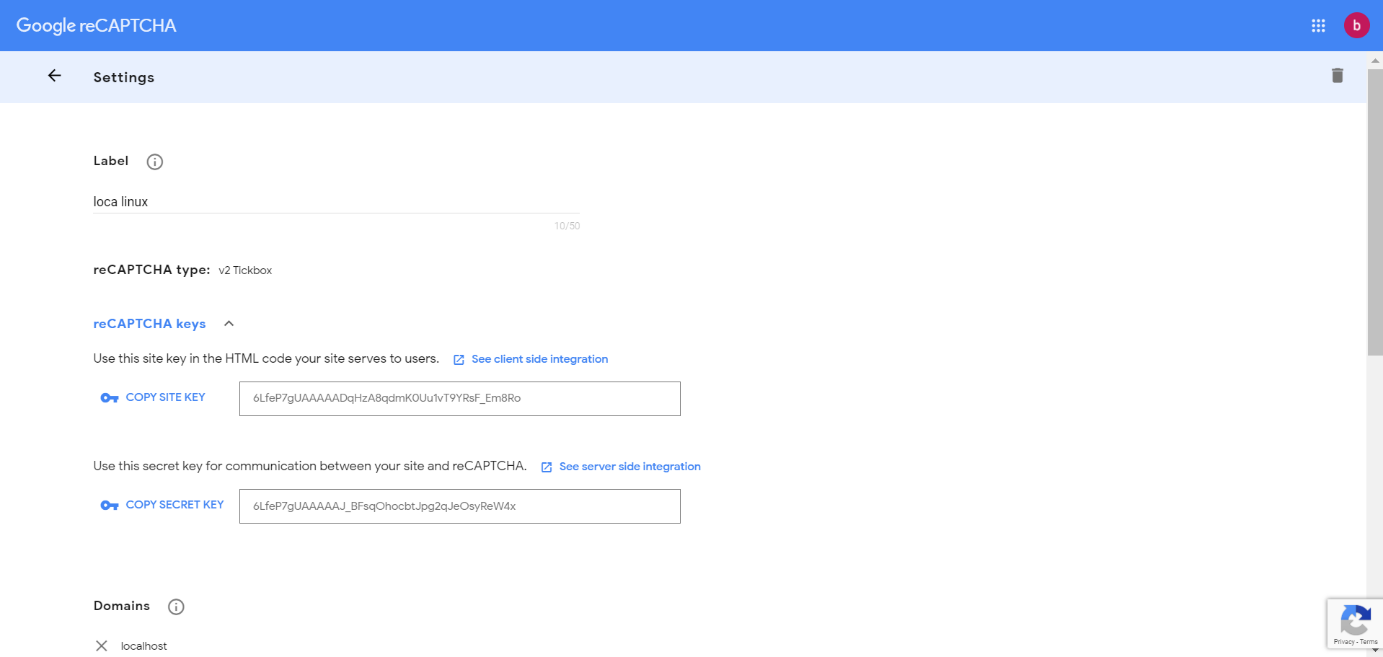


**II.5 To enable new user registration, you also need to generate CAPTCHA site key, from https://www.google.com/recaptcha/admin**

After adding your domain, update sitekey in src/config.js

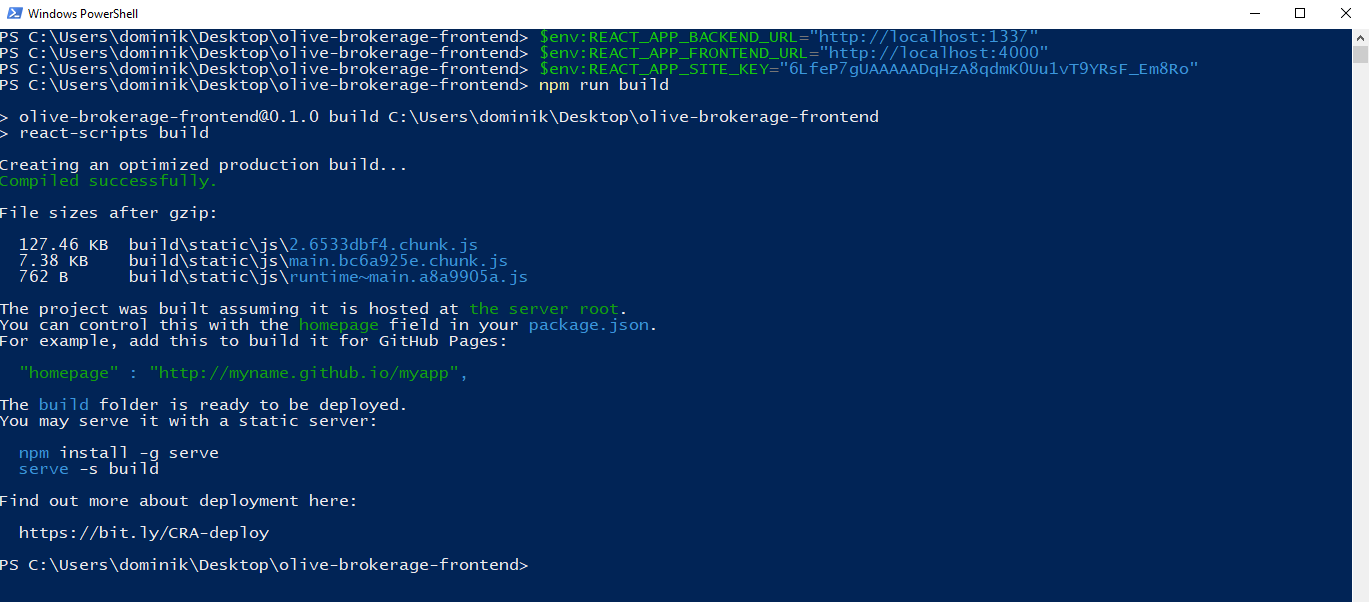
You must also add your secret to ‘middlewares/validateUser/index.js’ in your backend api or using env variable:

process.env.SITE\_SECRET (in your backend api process !)



**II.6 run:**

npm run build



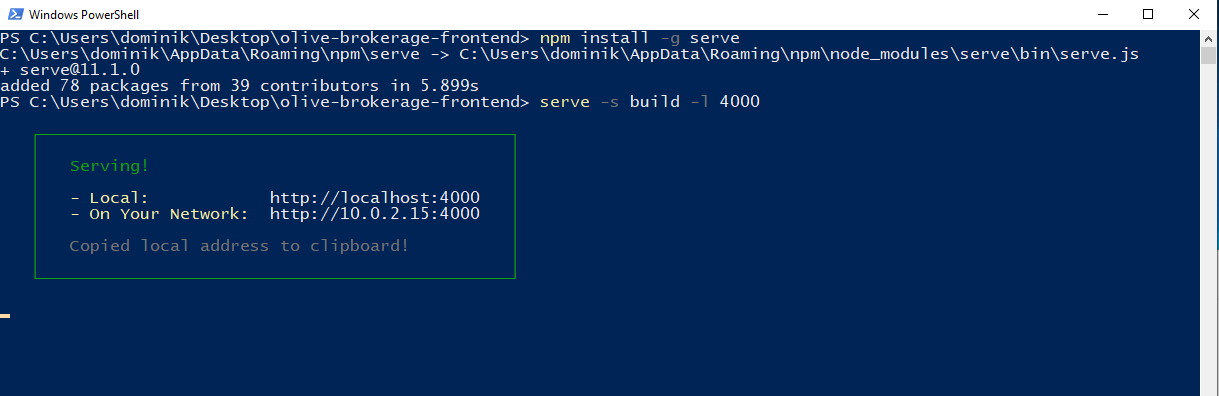
**II.7 Host the built app (build folder)**

Example hosting using serve:

npm install -g serve

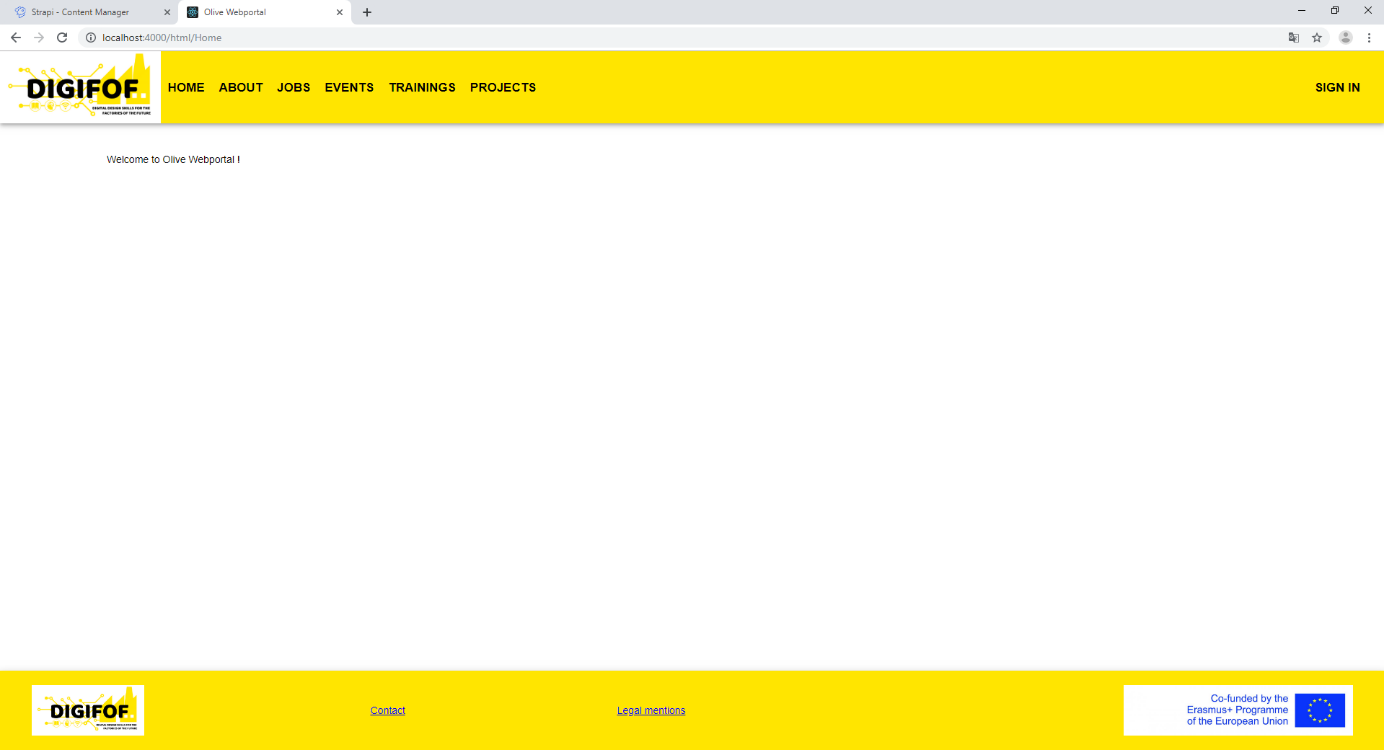
serve -s build -l 4000

In above example, port where your app will be available is 4000 - you can adjust it to your needs



**II.8 Having your back-end api deployed and front-end served, you should see your app server at the configured port, with example data**

Please note, that if you host both apps on localhost, and you would like to test features like user registration, you would need to allow CORS in your browser (e.g. with Chrome plugin).



**Further Questions**

in case of problems and for more information contact us at [faq@adoxx.org](mailto:faq@adoxx.org)

# Conclusion

The release of Brokerage system installation files and guide will allow project partners to create their own installations of the system.

It will allow project partners to publish available positions, internships, trainee positions etc. in relation to DigiFoF. Academia will be able to announce open position for student, teacher and staff mobility. There is also an event and training publishing service.

The Brokerage System will be maintained and updated during whole project duration.

1. https://www.adoxx.org/live/olive [↑](#footnote-ref-1)
2. Copyright (c) 2015-2019 Strapi Solutions.

   Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

   The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. [↑](#footnote-ref-2)