Project Title:

THE FOF-DESIGNER:
DIGITAL DESIGN SKILLS FOR FACTORIES OF THE FUTURE

Project Acronym:
DigiFoF

Grant Agreement number:
2018-2533 / 001-001
Project Nr. 601089-EPP-1-2018-1-RO-EPPKA2-KA

Subject:
D2.4 BROKERAGE SYSTEM[1]

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

This 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

[1] “Any communication or publication related to the action, made by the beneficiaries jointly or individually in any form and using any means, shall indicate that it reflects only the author's view and that the Agency and the Commission are not responsible for any use that may be made of the information it contains.”

Table of content

1 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

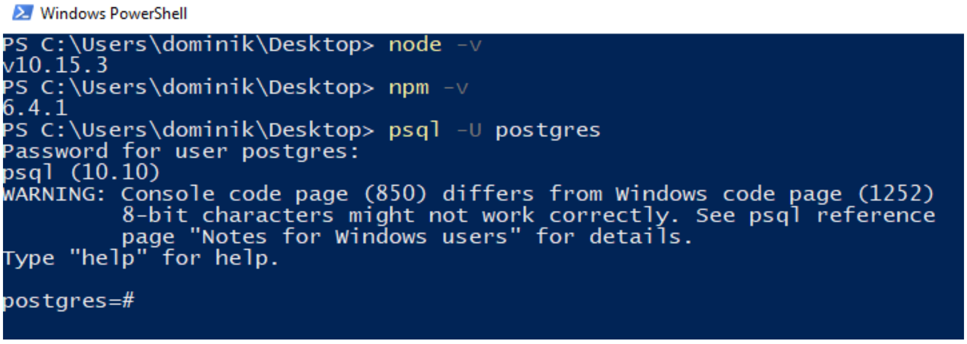
2 Brokerage system access data and installation guide

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[2]. Main component of web-platform is developed with Open Source[3] 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)



```
PS C:\Users\dominik\Desktop> node -v
v10.15.3
PS C:\Users\dominik\Desktop> npm -v
6.4.1
PS C:\Users\dominik\Desktop> psql -U postgres
Password for user postgres:
psql (10.10)
WARNING: Console code page (850) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

postgres=#
```

Manual consists of two parts: back-end and front-end.

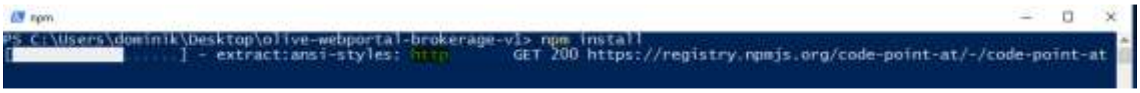
I. Back-end deployment

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:

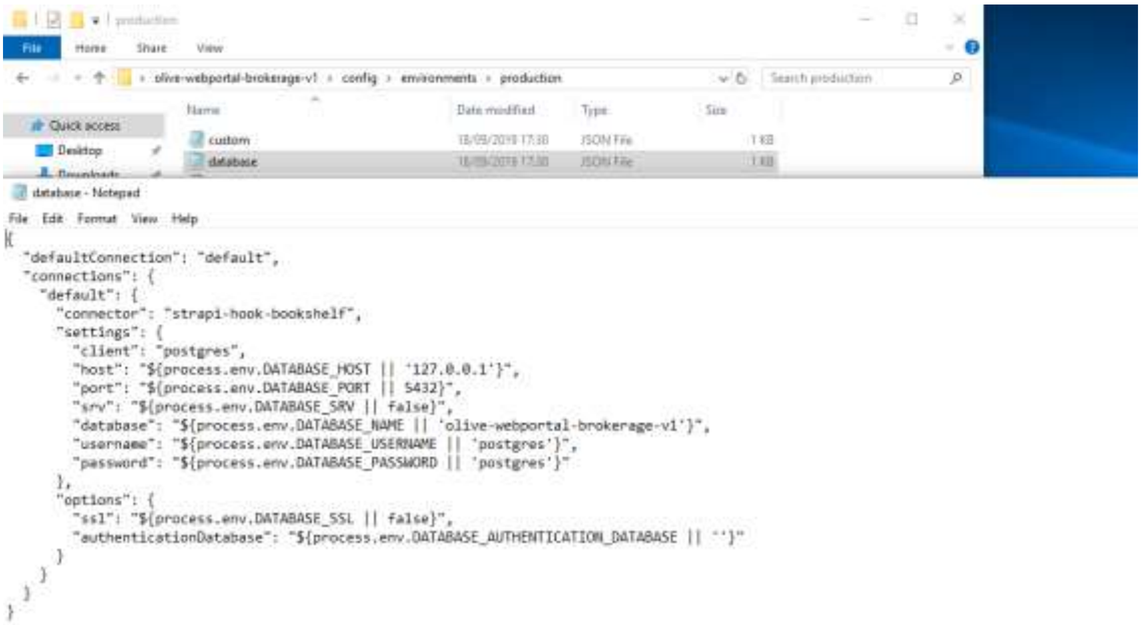
npm install



```
PS C:\Users\dominik\Desktop\olive-webportal-brokerage-v1> npm install
[              .....] - extract:ansi-styles: http      GET 200 https://registry.npmjs.org/code-point-at/-/code-point-at
```

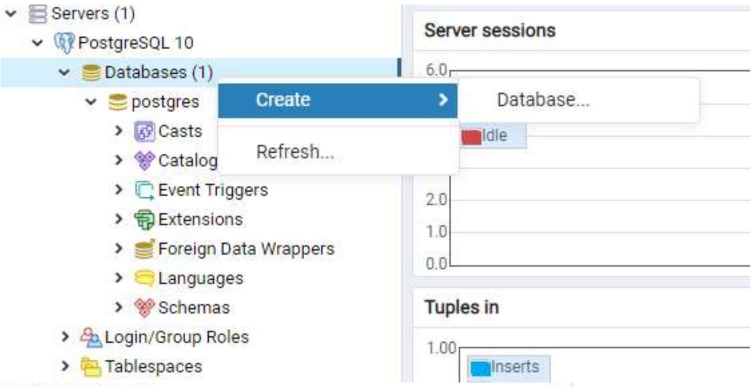
[2] https://www.adoxx.org/live/olive
[3] Copyright (c) 2015-2019 Strapi Solutions.
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation 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.

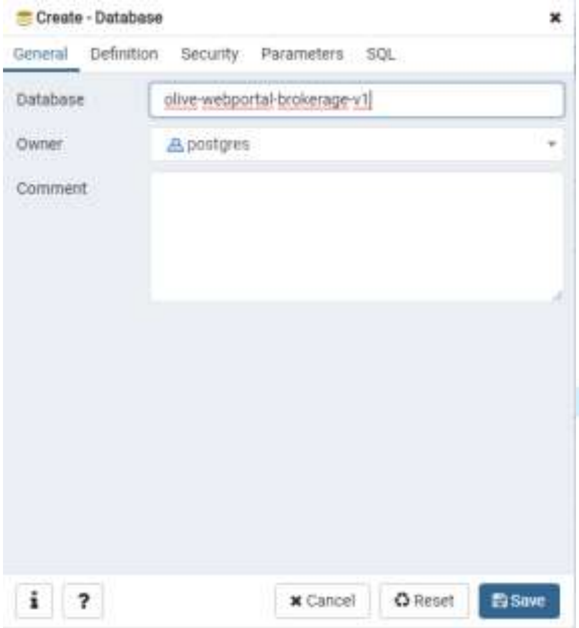
I.4 Adjust config for database connection (set user/pass/port/host as environmental variables) or type it in '\config\environments\production\database.json'.


production
File Home Share View
olive-webportal-brokerage-v1 › config › environments › production
Search production
Name Date modified Type Size
Quick access
Desktop
Downloads
custom 18/09/2019 17:30 JSON File 1 KB
database 18/09/2019 17:30 JSON File 1 KB
database - Notepad
File Edit Format View Help

```
{
  "defaultConnection": "default",
  "connections": {
    "default": {
      "connector": "strapi-hook-bookshelf",
      "settings": {
        "client": "postgres",
        "host": "${process.env.DATABASE_HOST || '127.0.0.1'}",
        "port": "${process.env.DATABASE_PORT || 5432}",
        "srv": "${process.env.DATABASE_SRV || false}",
        "database": "${process.env.DATABASE_NAME || 'olive-webportal-brokerage-v1'}",
        "username": "${process.env.DATABASE_USERNAME || 'postgres'}",
        "password": "${process.env.DATABASE_PASSWORD || 'postgres'}"
      },
      "options": {
        "ssl": "${process.env.DATABASE_SSL || false}",
        "authenticationDatabase": "${process.env.DATABASE_AUTHENTICATION_DATABASE || ''}"
      }
    }
  }
}
```

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).

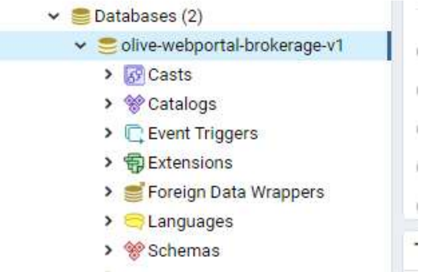




Database: olive-webportal-brokerage-v1

Owner: postgres

Comment:



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

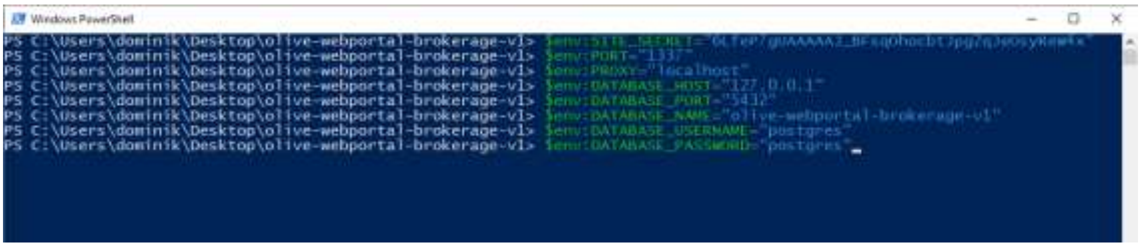
Environmental variables names:

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'.

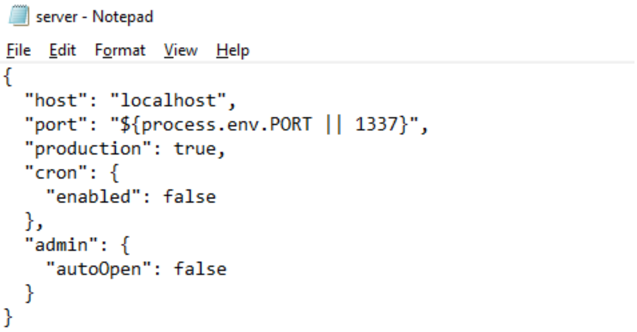


```
PS C:\Users\dominik\Desktop\olive-webportal-brokerage-v1> $env:SITE_SECRET="6LfeP7gUAAAAAJ_BFsqOhocbt7pg7qJeOsyKem4x"
PS C:\Users\dominik\Desktop\olive-webportal-brokerage-v1> $env:PORT="1337"
PS C:\Users\dominik\Desktop\olive-webportal-brokerage-v1> $env:PROXY="localhost"
PS C:\Users\dominik\Desktop\olive-webportal-brokerage-v1> $env:DATABASE_HOST="127.0.0.1"
PS C:\Users\dominik\Desktop\olive-webportal-brokerage-v1> $env:DATABASE_PORT="5432"
PS C:\Users\dominik\Desktop\olive-webportal-brokerage-v1> $env:DATABASE_NAME="olive-webportal-brokerage-v1"
PS C:\Users\dominik\Desktop\olive-webportal-brokerage-v1> $env:DATABASE_USERNAME="postgres"
PS C:\Users\dominik\Desktop\olive-webportal-brokerage-v1> $env:DATABASE_PASSWORD="postgres"_
```

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:

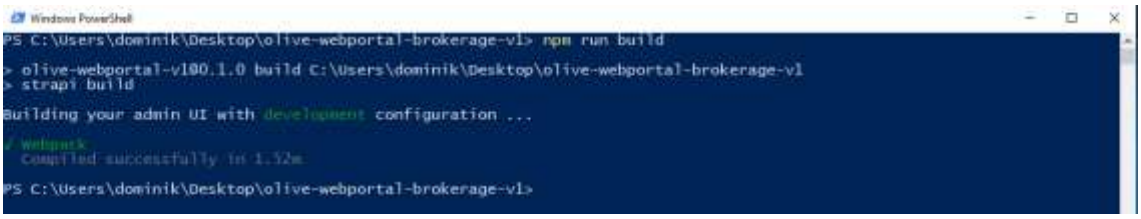
```
"proxy": {
"enabled": "${process.env.PROXY_ENABLE || true}",
"ssl": "${process.env.PROXY_SSL || true}",
"host": "${process.env.PROXY_HOST || 'digifof-brokerage-cms.herokuapp.com'}" ,
"port": "${process.env.PROXY_PORT || 443}"
},
```



```
{
  "host": "localhost",
  "port": "${process.env.PORT || 1337}",
  "production": true,
  "cron": {
    "enabled": false
  },
  "admin": {
    "autoOpen": false
  }
}
```

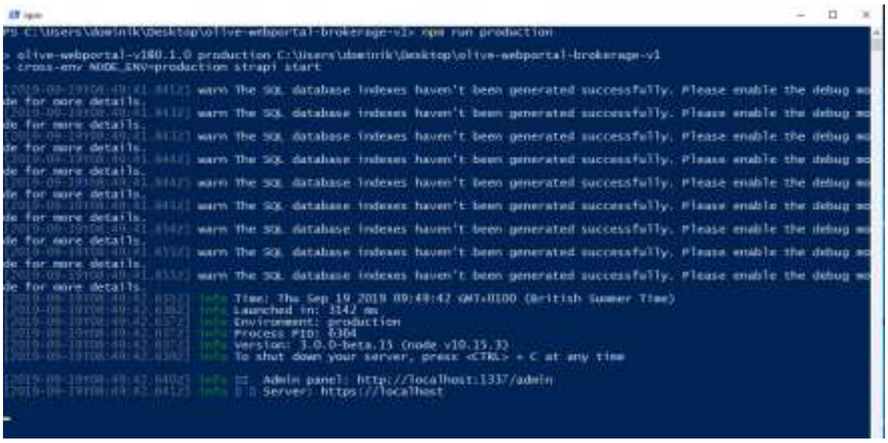
I.5 run:

npm run build



```
PS C:\Users\dominik\Desktop\olive-webportal-brokerage-v1> npm run build

> olive-webportal-v180.1.0 build C:\Users\dominik\Desktop\olive-webportal-brokerage-v1
> strapi build

Building your admin UI with development configuration ...

✓ webpack
  Compiled successfully in 1.52m

PS C:\Users\dominik\Desktop\olive-webportal-brokerage-v1>
```

npm run production



```
PS C:\Users\dominik\Desktop\olive-webportal-brokerage-v1> npm run production

> olive-webportal-v180.1.0 production C:\Users\dominik\Desktop\olive-webportal-brokerage-v1
> cross-env NODE_ENV=production strapi start

[2019-09-19T08:49:41.841Z] warn The SQL database indexes haven't been generated successfully. Please enable the debug mo
de for more details.
[2019-09-19T08:49:41.841Z] warn The SQL database indexes haven't been generated successfully. Please enable the debug mo
de for more details.
[2019-09-19T08:49:41.841Z] warn The SQL database indexes haven't been generated successfully. Please enable the debug mo
de for more details.
[2019-09-19T08:49:41.844Z] warn The SQL database indexes haven't been generated successfully. Please enable the debug mo
de for more details.
[2019-09-19T08:49:41.844Z] warn The SQL database indexes haven't been generated successfully. Please enable the debug mo
de for more details.
[2019-09-19T08:49:41.841Z] warn The SQL database indexes haven't been generated successfully. Please enable the debug mo
de for more details.
[2019-09-19T08:49:41.854Z] warn The SQL database indexes haven't been generated successfully. Please enable the debug mo
de for more details.
[2019-09-19T08:49:41.851Z] warn The SQL database indexes haven't been generated successfully. Please enable the debug mo
de for more details.
[2019-09-19T08:49:41.855Z] warn The SQL database indexes haven't been generated successfully. Please enable the debug mo
de for more details.
[2019-09-19T08:49:42.635Z] info Time: Thu Sep 19 2019 09:49:42 GMT+0100 (British Summer Time)
[2019-09-19T08:49:42.636Z] info Launched in: 3142 ms
[2019-09-19T08:49:42.637Z] info Environment: production
[2019-09-19T08:49:42.637Z] info Process PID: 6364
[2019-09-19T08:49:42.637Z] info Version: 3.0.0-beta.15 (node v10.15.3)
[2019-09-19T08:49:42.638Z] info To shut down your server, press <CTRL> + C at any time

[2019-09-19T08:49:42.640Z] info ☰  Admin panel: http://localhost:1337/admin
[2019-09-19T08:49:42.641Z] info ⚡  Server: https://localhost
```

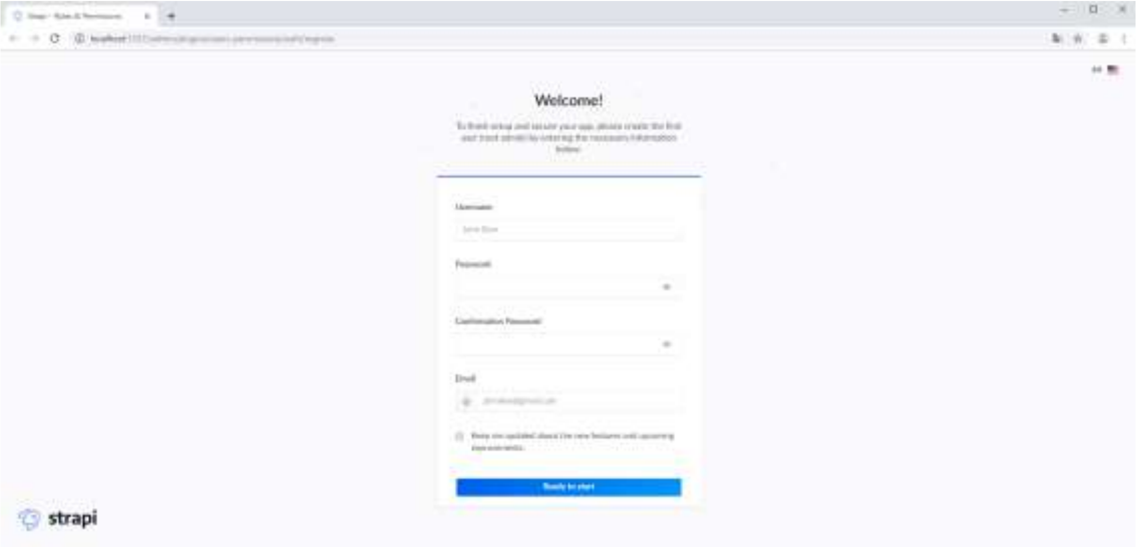
By default, your application will run on localhost, port 1337 and your admin dashboard would be available at:

http://localhost:1337/admin

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

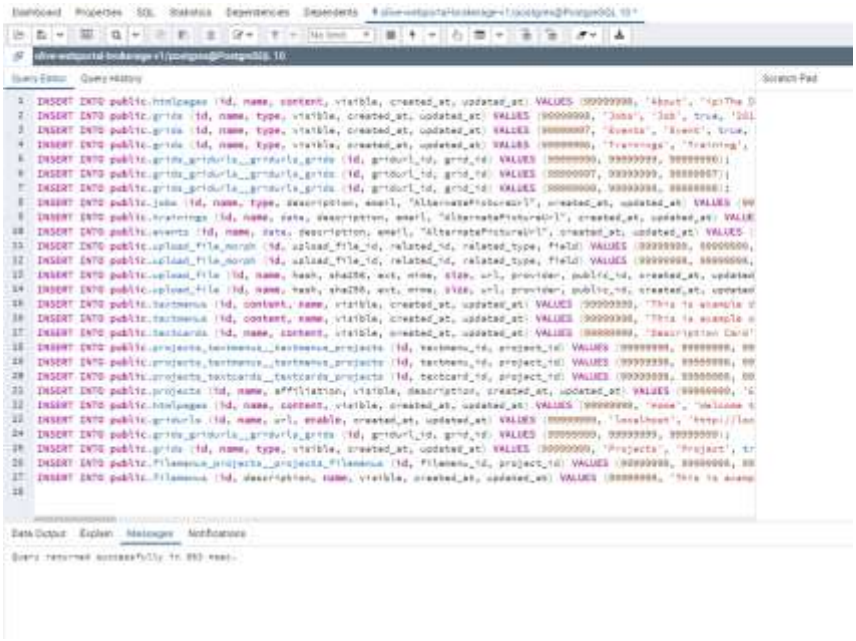
To be able to receive proper account confirmation emails, please also fill in your proxy settings if host is different from localhost.

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



Welcome!
To finish setup and secure your app, please create the first user (root admin) by entering the necessary information below.
Username
John Doe
Password
Confirmation Password
Email
johndoe@gmail.com
Keep me updated about the new features and upcoming improvements.
Ready to start
strapi

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.



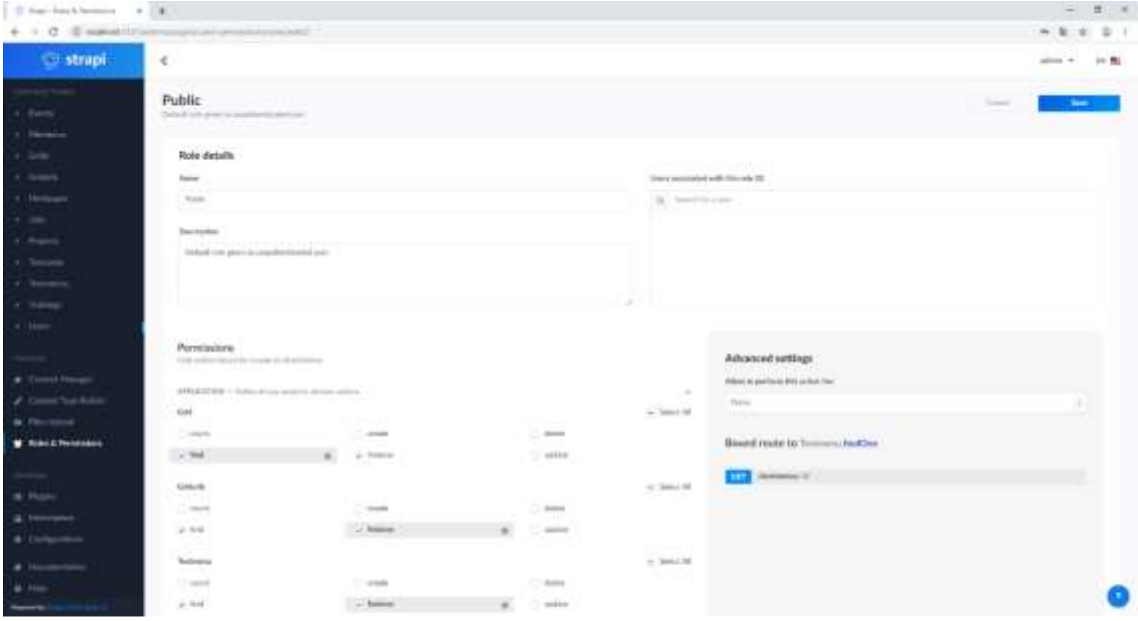
```
INSERT INTO public.htmlpages (id, name, content, visible, created_at, updated_at) VALUES (99999998, 'About', '<p>The D
INSERT INTO public.grids (id, name, type, visible, created_at, updated_at) VALUES (99999998, 'Jobs', 'Job', true, '201
INSERT INTO public.grids (id, name, type, visible, created_at, updated_at) VALUES (99999997, 'Events', 'Event', true,
INSERT INTO public.grids (id, name, type, visible, created_at, updated_at) VALUES (99999996, 'Trainings', 'Training',
INSERT INTO public.grids_gridurls__gridurls_grids (id, gridurl_id, grid_id) VALUES (99999998, 99999999, 99999998);
INSERT INTO public.grids_gridurls__gridurls_grids (id, gridurl_id, grid_id) VALUES (99999997, 99999999, 99999997);
INSERT INTO public.grids_gridurls__gridurls_grids (id, gridurl_id, grid_id) VALUES (99999996, 99999999, 99999996);
INSERT INTO public.jobs (id, name, type, description, email, "AlternatePictureUrl", created_at, updated_at) VALUES (99
INSERT INTO public.trainings (id, name, date, description, email, "AlternatePictureUrl", created_at, updated_at) VALUE
INSERT INTO public.events (id, name, date, description, email, "AlternatePictureUrl", created_at, updated_at) VALUES (
INSERT INTO public.upload_file_morph (id, upload_file_id, related_id, related_type, field) VALUES (99999999, 99999999,
INSERT INTO public.upload_file_morph (id, upload_file_id, related_id, related_type, field) VALUES (99999998, 99999998,
INSERT INTO public.upload_file (id, name, hash, sha256, ext, mime, size, url, provider, public_id, created_at, updated
INSERT INTO public.upload_file (id, name, hash, sha256, ext, mime, size, url, provider, public_id, created_at, updated
INSERT INTO public.textmenus (id, content, name, visible, created_at, updated_at) VALUES (99999999, 'This is example t
INSERT INTO public.textmenus (id, content, name, visible, created_at, updated_at) VALUES (99999998, 'This is example o
INSERT INTO public.textcards (id, name, content, visible, created_at, updated_at) VALUES (99999999, 'Description Card'
INSERT INTO public.projects_textmenus__textmenus_projects (id, textmenu_id, project_id) VALUES (99999999, 99999999, 99
INSERT INTO public.projects_textmenus__textmenus_projects (id, textmenu_id, project_id) VALUES (99999998, 99999998, 99
INSERT INTO public.projects_textcards__textcards_projects (id, textcard_id, project_id) VALUES (99999999, 99999999, 99
INSERT INTO public.projects (id, name, affiliation, visible, description, created_at, updated_at) VALUES (99999999, 'G
INSERT INTO public.htmlpages (id, name, content, visible, created_at, updated_at) VALUES (99999999, 'Home', '<Welcome t
INSERT INTO public.gridurls (id, name, url, enable, created_at, updated_at) VALUES (99999999, 'localhost', 'http://loc
INSERT INTO public.grids_gridurls__gridurls_grids (id, gridurl_id, grid_id) VALUES (99999999, 99999999, 99999999);
INSERT INTO public.grids (id, name, type, visible, created_at, updated_at) VALUES (99999999, 'Projects', 'Project', tr
INSERT INTO public.filemenus_projects__projects_filemenus (id, filemenu_id, project_id) VALUES (99999999, 99999999, 99
INSERT INTO public.filemenus (id, description, name, visible, created_at, updated_at) VALUES (99999999, 'This is examp
```

```
Query returned successfully in 893 msec.
```

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)

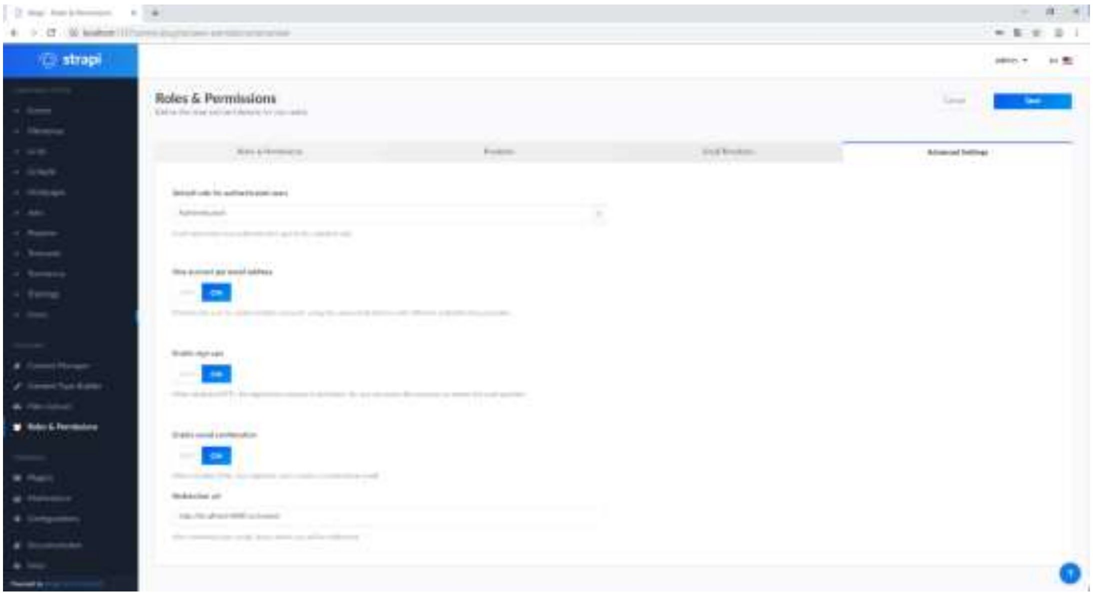
As a default, you should set permission to 'find' and 'findone' for all public roles for all predefined content types (remember about saving – button in top right corner).



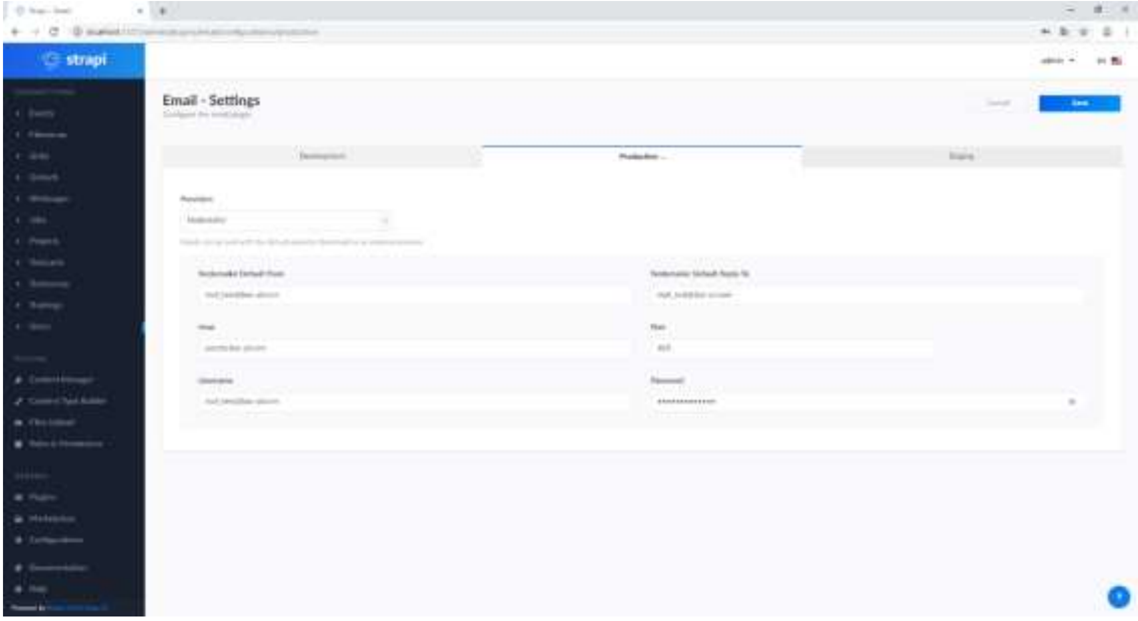
I.8 Turn on new user account confirmation in Plugins->Roles&Permissions->Advanced Settings

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.



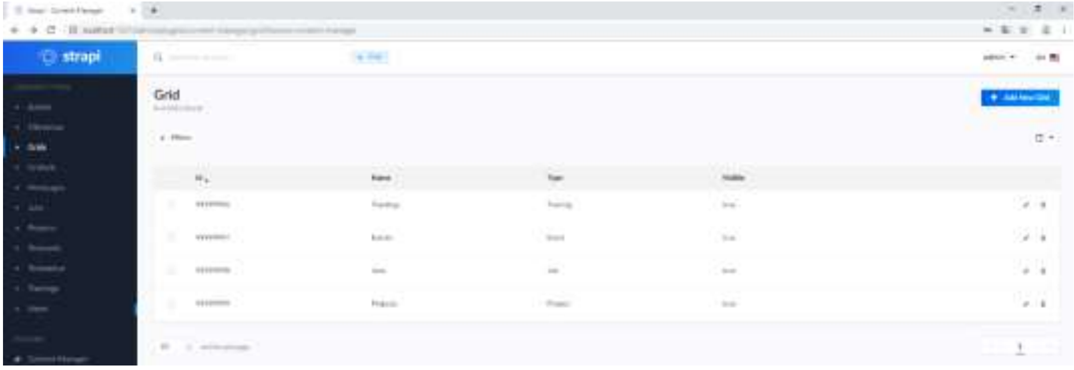
I.10 Content creation

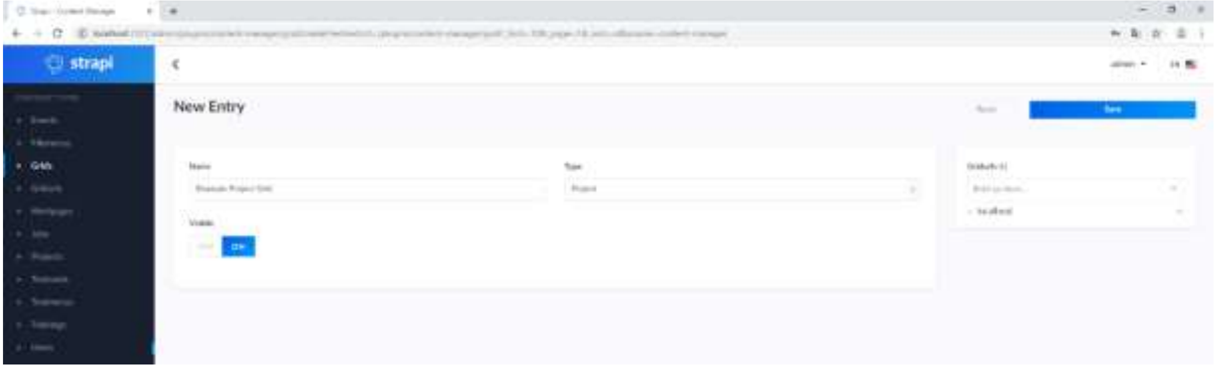
New content could be created, by selecting names of content types and clicking 'Add New [contenttype]' button.

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/

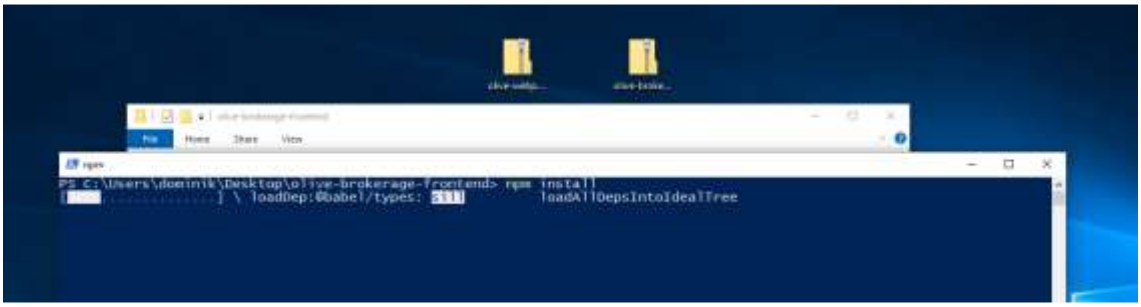
II. Frond-end deployment

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



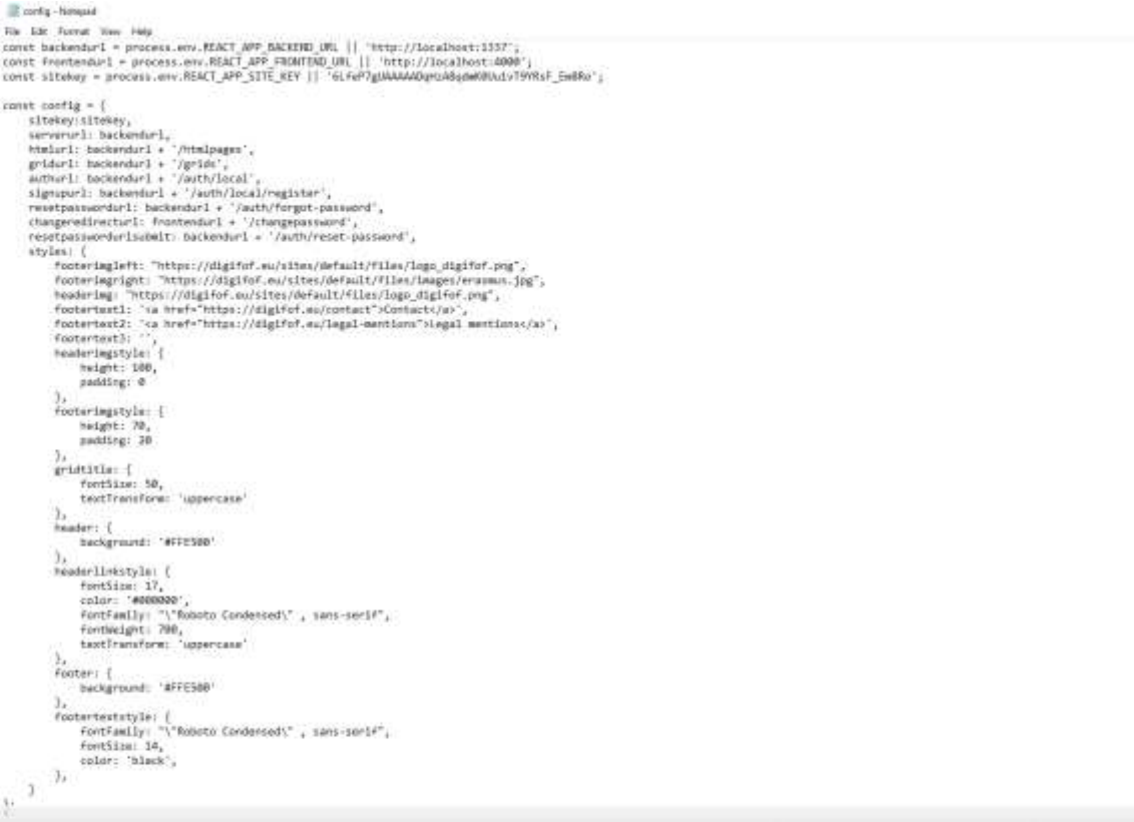
```
PS C:\Users\dominik\Desktop\olive-brokerage-frontend> npm install
[         ..............] \ loadDep:@babel/types: sill         loadAllDepsIntoIdealTree
```

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.



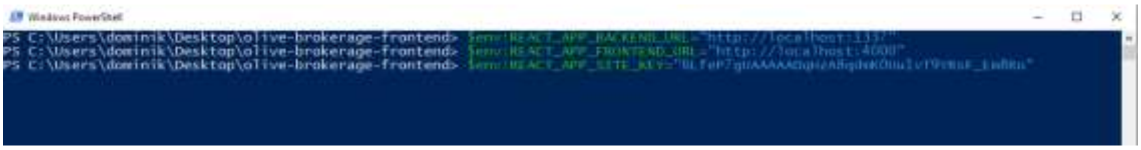
```
const backendurl = process.env.REACT_APP_BACKEND_URL || 'http://localhost:1337';
const frontendurl = process.env.REACT_APP_FRONTEND_URL || 'http://localhost:4000';
const sitekey = process.env.REACT_APP_SITE_KEY || '6LfeP7gUAAAAADgHzA8qdeK0UudvT9YRsF_Ew8Ro';

const config = {
    sitekey:sitekey,
    serverurl: backendurl,
    htmlurl: backendurl + '/htmlpages',
    gridurl: backendurl + '/grids',
    authurl: backendurl + '/auth/local',
    signupurl: backendurl + '/auth/local/register',
    resetpasswordurl: backendurl + '/auth/forgot-password',
    changeredirecturl: frontendurl + '/changepassword',
    resetpasswordurlsubmit: backendurl + '/auth/reset-password',
    styles: {
        footerimgleft: "https://digifof.eu/sites/default/files/logo_digifof.png",
        footerimgright: "https://digifof.eu/sites/default/files/images/erasmus.jpg",
        headerimg: "https://digifof.eu/sites/default/files/logo_digifof.png",
        footertext1: `<a href="https://digifof.eu/contact">Contact</a>`,
        footertext2: `<a href="https://digifof.eu/legal-mentions">Legal mentions</a>`,
        footertext3: ``,
        headerimgstyle: {
            height: 100,
            padding: 0
        },
        footerimgstyle: {
            height: 70,
            padding: 20
        },
        gridtitle: {
            fontSize: 50,
            textTransform: 'uppercase'
        },
        header: {
            background: '#FFE500'
        },
        headerlinkstyle: {
            fontSize: 17,
            color: '#000000',
            fontFamily: "\"Roboto Condensed\" , sans-serif",
            fontWeight: 700,
            textTransform: 'uppercase'
        },
        footer: {
            background: '#FFE500'
        },
        footertextstyle: {
            fontFamily: "\"Roboto Condensed\" , sans-serif",
            fontSize: 14,
            color: 'black',
        },
    }
};
```

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



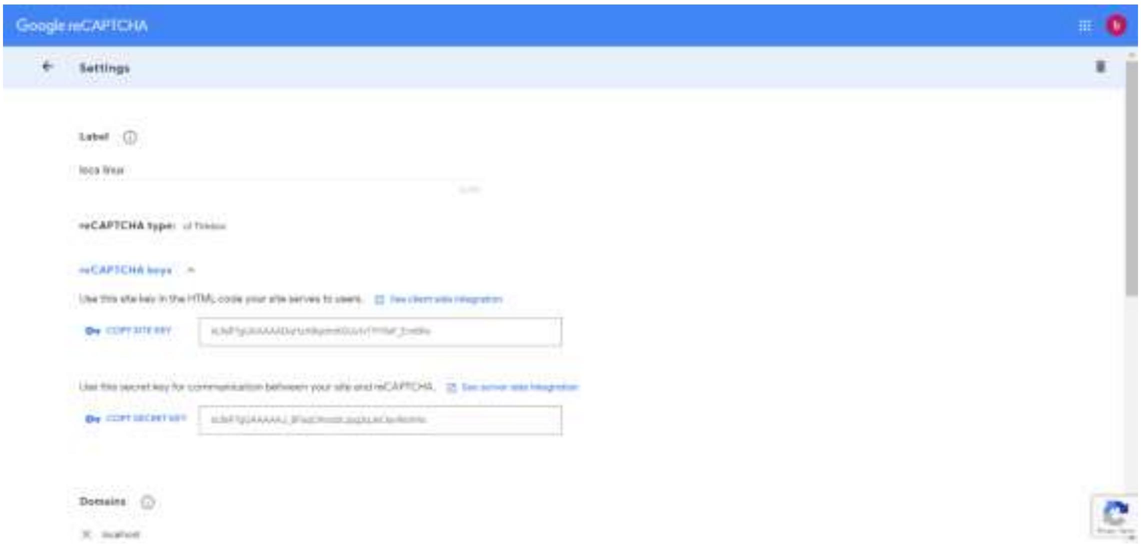
```
PS C:\Users\dominik\Desktop\olive-brokerage-frontend> $env:REACT_APP_BACKEND_URL="http://localhost:1337"
PS C:\Users\dominik\Desktop\olive-brokerage-frontend> $env:REACT_APP_FRONTEND_URL="http://localhost:4000"
PS C:\Users\dominik\Desktop\olive-brokerage-frontend> $env:REACT_APP_SITE_KEY="6LfeP7gUAAAAADgHzA8qdeK0UuIvT9YRsF_Ew8Ro"
```

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 !)

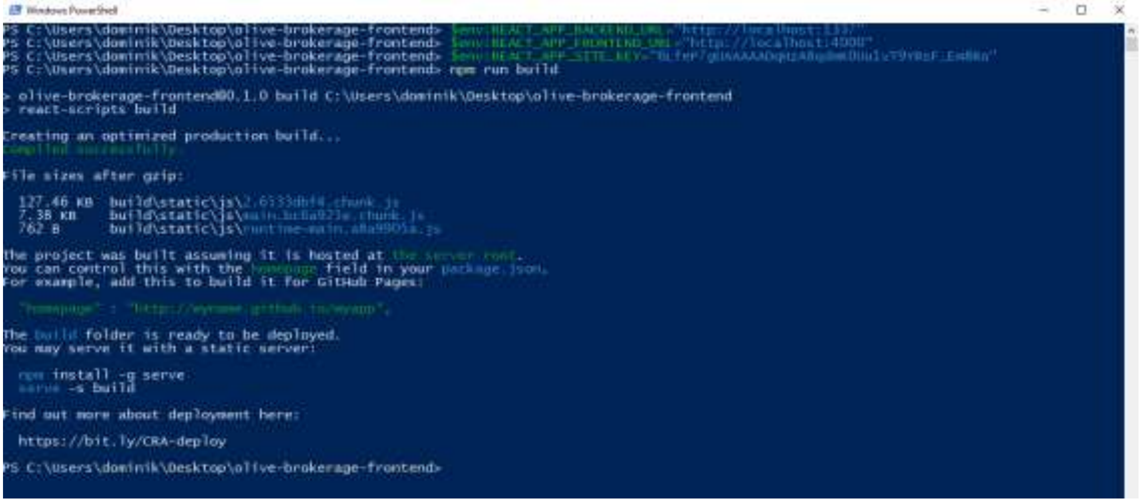


Use this site key in the HTML code your site serves to users. See client side integration

Use this secret key for communication between your site and reCAPTCHA. See server side integration

II.6 run:

npm run build



```
PS C:\Users\dominik\Desktop\olive-brokerage-frontend> $env:REACT_APP_BACKEND_URL="http://localhost:1337"
PS C:\Users\dominik\Desktop\olive-brokerage-frontend> $env:REACT_APP_FRONTEND_URL="http://localhost:4000"
PS C:\Users\dominik\Desktop\olive-brokerage-frontend> $env:REACT_APP_SITE_KEY="6Lfer7gUAAAAADqpzA8g[illegible]_Ex8Rn"
PS C:\Users\dominik\Desktop\olive-brokerage-frontend> npm run build

> olive-brokerage-frontend@0.1.0 build C:\Users\dominik\Desktop\olive-brokerage-frontend
> react-scripts build

Creating an optimized production build...
Compiled successfully.

File sizes after gzip:

  127.46 KB  build\static\js\2.6133dbf4.chunk.js
  7.38 KB    build\static\js\main.bc8a923e.chunk.js
  762 B      build\static\js\runtime-main.a8a9905a.js

The project was built assuming it is hosted at the server root.
You can control this with the homepage field in your package.json.
For example, add this to build it for GitHub Pages:

  "homepage" : "http://myname.github.io/myapp",

The build folder is ready to be deployed.
You may serve it with a static server:

  npm install -g serve
  serve -s build

Find out more about deployment here:

  https://bit.ly/CRA-deploy

PS C:\Users\dominik\Desktop\olive-brokerage-frontend>
```

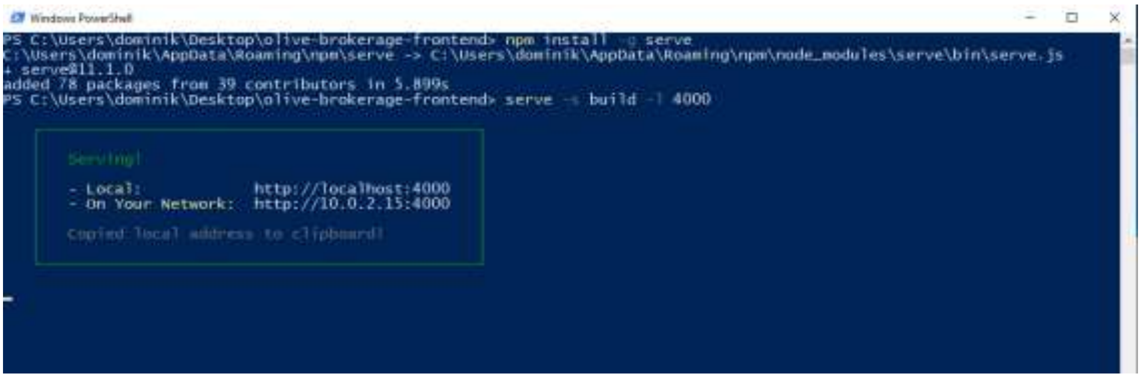
II.7 Host the built app (build folder)

Example hosting using serve:

npm install -g serve

serve -s build -l 4000

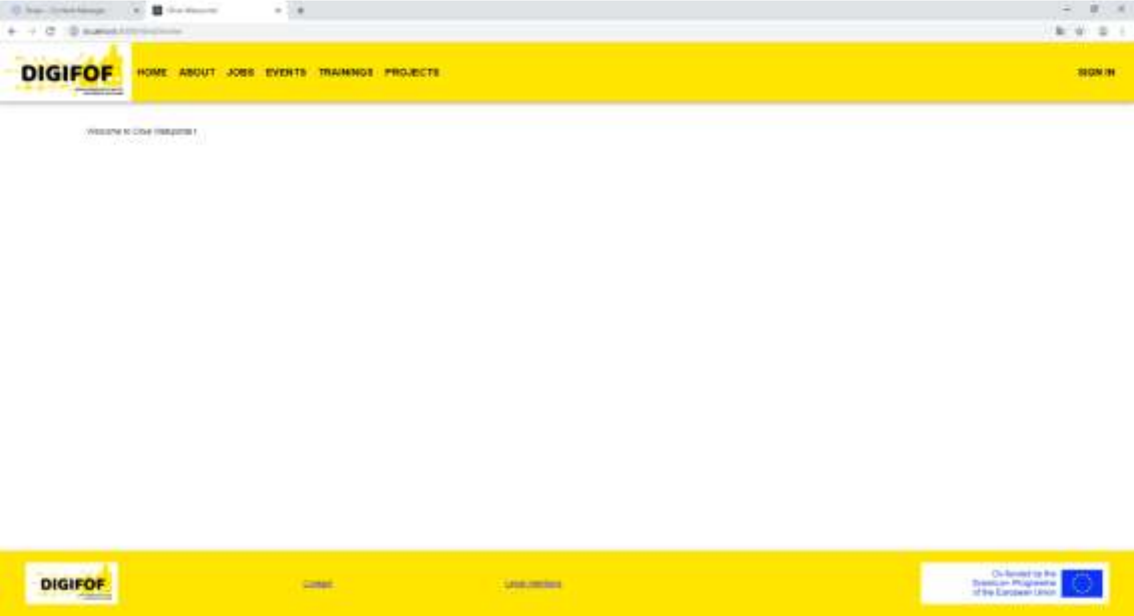
In above example, port where your app will be available is 4000 - you can adjust it to your needs



```
PS C:\Users\dominik\Desktop\olive-brokerage-frontend> npm install -g serve
C:\Users\dominik\AppData\Roaming\npm\serve -> C:\Users\dominik\AppData\Roaming\npm\node_modules\serve\bin\serve.js
+ serve@11.1.0
added 78 packages from 39 contributors in 5.899s
PS C:\Users\dominik\Desktop\olive-brokerage-frontend> serve -s build -l 4000

   Serving!

   - Local:            http://localhost:4000
   - On Your Network:  http://10.0.2.15:4000

   Copied local address to clipboard!
```

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

3 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.