CSyllabus app = Django REST + Angular 4

Made by team from POLIMI(Italy) and FER(Croatia University)

Motivation

CSyllabus is imagined as a web platform which should ease up process of finding and comparing courses on domestic and foreign faculties. It will enable users to discover and compare courses on interactive way through web application. This “one click” app will save time and provide very useful information to interested parties.

Instalation guide for the backend:

Install PostgreSQL 9.6x (https://www.postgresql.org/download/). Install with pgadmin. Create new server hostname -> “localhost”. Write down password and username for root (usually username = postgres). After installation create a database in pgadmin to be used with the csyllabus and write down the name used. If you create a new username and password for the database write it down too. Install python 2.7 (https://www.python.org/downloads/). Check python version in command line with:

```
python -V
```

Install pip for python 2.7 (it allready comes shipped with python 2.7.9+) . Check pip version with

```
pip -V
```

Install django with

`pip install django` Check django version with 'python -c "import django; print(django.get_version())"'. Position yourself in csyllabus root folder. `pip install -r backend/requirements/devl.pip` In file backend/settings/devl.py field change DATABASES according to database name, username and password you wrote down in first.

To finish load the migrations and fixtures into the database:

```
python manage.py migrate
python manage.py loaddata
backend/apps/csyllabusapi/fixtures/epfl_fixtures_json.json
python manage.py loaddata backend/apps/csyllabusapi/fixtures/fer_fixtures_json.json
python manage.py loaddata
backend/apps/csyllabusapi/fixtures/laquila_fixtures_json.json
python manage.py loaddata
backend/apps/csyllabusapi/fixtures/mockup_fixtures_json.json
python manage.py loaddata
backend/apps/csyllabusapi/fixtures/polimi_fixtures_json.json
python manage.py loaddata
backend/apps/csyllabusapi/fixtures/stanford_fixtures_json.json
python manage.py loaddata
backend/apps/csyllabusapi/fixtures/texas_fixtures_json.json
python manage.py loaddata
backend/apps/csyllabusapi/fixtures/ucla_fixtures_json.json
python manage.py createsuperuser
```

And run django server:

```
python manage.py runserver
```

Instalation guide for the frontend:

Install nodeJS (https://nodejs.org/en/download/). Position youself in the frontend/csyllabus folder. Run `npm install` Serve angular app with `ng serve`

Coding the backend

Backend is made using Django REST Framework, main custom functionalities are located in /backend/csyllabusapi folder which represents a custom djangoo app. It contains custom models, views, managements scripts, helper scripts and database migrations and fixtures. Basic knowledge of Django Framework is required to make changes in the backend.

Recommended IDE is PyCharm, but if you are using other IDE make sure it is connected to statics code analyzer which checks adherence to PEP 8 standard.
(https://www.python.org/dev/peps/pep-0008/)

In writing the API make sure to adhere to these standards:

https://google.github.io/styleguide/jsoncstyleguide.xml https://cloud.google.com/apis/design/

Coding the frontend:

Frontend is made using Angular4 Framework with Angular Material Components and Angular CLI, main custom functionalities are located in /frontend/csyllabus and

/frontend/admin folders which represent two different custom angular apps. One is for a public website and the other is for the admin website. It contains custom modules, custom components, templates and styles. Basic knowledge of Angular Framework is required to make changes in the frontend.

To adhere to code conventiones we must code using tslint to make sure we convey to these style guidelines (https://angular.io/guide/styleguide).

Connect your IDE with tslinf file: frontend/csyllabus/tslint.json

Usually IDE-s do this autoamticaly but if they for some reason didn't or if you are unsure if they did:

Instrunctions for WebStorm: https://www.jetbrains.com/help/webstorm/tslint.html Instrunction fro PyCharm: https://www.jetbrains.com/help/pycharm/tslint.html Intrunctions for VisualStudio Code: https://www.youtube.com/watch?v=-lgBFAtKJ1k

Before pushing to your branch I recommend running `ng lint` and `ng test`. If you coded listening to linter errors and warnings `ng lint` should say all is fine, if you didnt it will tell you what to correct.

Running `ng test` will check for unit tests, now these are great and building components and services with angular cli automatically creates some unit tests which give fair code coverage.