Noppa.us

Final report

Rami Alatalo <rami.alatalo@aalto.fi>
Kristoffer Snabb <kristoffer.snabb@aalto.fi>
Henrik Horstia <henrik.horstia@aalto.fi>

Introduction

The first plan was to implement a service that could recommend courses to a student based on the student's previous courses and other students' evaluations of the courses. This plan did not work as we did not have any way of retrieving students' own courses data.

Due to this problem we decided to change the topic in a direction where we can complete a Noppa service without relying on outside support for data retrieval. Instead of directly recommending courses to students the service will work as a portal to evaluate and browse popular courses. An effective way of browsing through relevant and popular courses is a substitute for recommending the courses directly to the student.

The final application did just this. It became a mobile friendly portal of browsing, evaluating and commenting courses for others to see which course is good and which is not. In addition to this we developed a functional, easy-to-deploy REST interface to Noppa courses, that also works as the basis for our application.

Installation instructions

The application is built using Django with the only additional requirement being Beautifulsoup4. Beautifulsoup4 and Django can be installed in the following way (setuptools is required):

```
> easy_install django
and
> easy_install beautifulsoup4
```

The source code can be found in github and can be cloned using git.

```
> git clone git@github.com:hhorstia/NoppaCRA.git
```

To run the code you need to create the database for evaluations first.

```
> cd NoppaCRA/src/django/
> python manage.py syncdb
```

And finally you can run the development server:

> python manage.py runserver

Now the application should be running on your localhost port 8000.

Description of the application

The service provides a possibility for the student to grade courses by giving a certain course 1 to 10 points. It is also possible to add an optional comment about the course. Other users can see the comments and evaluations made. By collecting evaluations from students about courses the application can provide an effective way to browse and sort popular courses.

The service also provides an easy to use interface to search for and see details of courses with a mobile device. The user can customize the how the app displays courses by changing the sorting criteria used and he/she can hide unwanted courses from the list.

The user interface and usage guide is described in more detail in appendix 1.

Application architecture

The server side of the application is kept as small as possible. It includes a REST interface, a cache and a way to scrape Noppa on the fly as a request is made.

The REST API

The Noppa data API we developed gives a JSON REST interface to the Noppa website. The website will be scraped for the requested information upon request to the API. The API will cache the scraped information for one week to improve the performance.

The API works in the following way:
/noppa/<school_code>/<department_code>/<course_code>

school -- the school where the course information can be found (optional)
department -- the department where the course can be found (optional)
course -- the course code of the course to retrieve. (optional)

Only one of the parameters needs to be given.

Examples of GET requests:
/noppa/ → will return a list of faculties
/noppa/sci/ → will return a list of departments in faculty sci
/noppa/sci/t3030 → will return a list of courses in department t3030
/noppa/sci/t3030/t-111.5360 → will return information about the course t-111.5360

To evaluate a course the service will send a POST request with a payload including the ‘grade’ to the course URL:
/noppa/sci/t3030/t-111.5360 → saves the evaluation points for the course

Additionally to this information retrieval directly from Noppa the API will provide an URL where

the user can query the neighbours of a course according to links and text information scraped from the course page in Noppa.

/node/<course_code> → returns information about neighbours for this course + the faculty and department the course belongs to

The neighbour information can be used to find relevant or closely related courses.

The database schema

The database schema is simple and only consists of one table. The evaluation table which is described in the Django ORM in the following way:

```
class Evaluation(models.Model):

    faculty = models.CharField(max_length = 100)
    department = models.CharField(max_length = 100)
    course = models.CharField(max_length = 15)
    grade = models.PositiveSmallIntegerField(
        choices = zip( range(1,10), range(1,10) )
    )
    comment = models.TextField()
    user = models.IntegerField()
```

The table is used to store the evaluation data of the courses. It includes fields for the grade given, the comment the users has made and user id of the user evaluating. It also has fields to distinguish the course, it's department and faculty.

References

- Django https://www.djangoproject.com/
- Scrapy http://scrapy.org/
- Beautiful Soup 4 http://www.crummy.com/software/BeautifulSoup/
- jQuery Mobile http://jquerymobile.com/
- Sencha Touch 2 http://www.sencha.com/products/touch
- Code Conventions for the JavaScript http://javascript.crockford.com/code.html
- Style Guide for Python Code http://www.python.org/dev/peps/pep-0008/