semesterly Documentation

Release 0.6

norofe

Aug 05, 2017

Contents

Contents:

CHAPTER 1

Semesterly Data Pipeline

CHAPTER 2

Locking Implementation

Data Representation

Locking courses involves mapping from a course (specifically, a course id), to a map from a section type (e.g. tutorial, lecture, etc. Any group of sections which will reference the same locked section) to the locked section in question (specifically, that section's code).

For example, say we have a course calculus with id 100, and lectures 'L1', 'L2', and tutorials 'T1' and 'T2'. Then one way to specify locked courses would be: {100: {'L': 'L2', 'T': 'T1'}}, which says that course 100 has L2 locked for section type L and T1 locked for section type T.

Implementation

This is implemented as the "course_to_section" variable in update_timetable.js. The issue is that when an action occurs on a front end which involves locking, the front end has no way of knowing what the section type of the locked item is - this is stored on the back end. Therefore the front end must send the section id to the backend, separately from the courses_to_sections object, (currently in the POST this is under updated_courses, which has a list of courses and their locked sections, if any). The back end then looks up the section type, updates the backend representation of the locked section (LOCKED_SECTIONS), and then resends an updated courses_to_sections object to the front end.

CHAPTER 3

Backend Endpoints

Create timetables

description This endpoint generates timetables

endpoint `/get_timetables/`

view `get_timetables`

request format example

```
{
    "courseSections": {
        "6076": {                                          # course id
            "P": "P0201"                                   # maps section type␣
↪code to locked section
        },
        "6090": {}                                         # empty object if no␣
↪locked sections
    },
    "customSlots": [],
    "numOptionCourses": 0,
    "optionCourses": [],
    "preferences": {
        "sort_metrics": [
            {
                "metric": "days with class",
                "order": "least",
                "selected": false
            },
            {
                "metric": "number of conflicts",
                "order": "least",
                "selected": false
            },
            {
```

```
                "metric": "time on campus",
                "order": "least",
                "selected": false
            },
            {
                "metric": "course rating stars",
                "order": "most",
                "selected": false
            }
        ],
        "try_with_conflicts": false
    },
    "school": "uoft",
    "semester": {
        "name": "Fall",
        "year": "2016"
    },
    "sid": "k#abILevGEL7vbhIYJwL&7!wl4tBVD",
    "updated_courses": [
        {
            "course_id": 6090,
            "section_codes": [                              # list of locked␣
↪section codes
                "P0201"
            ]
        }
    ]
}
```

response format

```
{
    "new_c_to_s": {
        "6076": {
            "P": "P0201"
        },
        "6090": {
            "P": "P0201"
        }
    },
    "timetables": [
        {
            "avg_rating": 0,
            "courses": [
                {
                    "code": "EMU150H1",
                    "department": "MUS",
                    "enrolled_sections": [
                        "P0201"
                    ],
                    "id": 6076,
                    "name": "Instrumental-Violin & Viola",
                    "num_credits": 0.17,
                    "slots": [
                        {
                            "_semester": " ",
                            "course": 6076,
                            "day": "W",
```

```
                            "enrolment": -1,
                            "id": 26674,
                            "instructors": "Rapoport",
                            "location": "120",
                            "meeting_section": "P0201",
                            "section": 15726,
                            "section_type": "P",
                            "semester": 1,
                            "size": -1,
                            "textbooks": [],
                            "time_end": "11:00",
                            "time_start": "10:00",
                            "waitlist": -1,
                            "waitlist_size": -1
                        }
                    ],
                    "textbooks": {
                        "P0201": []
                    }
                },
                ...
            ],
            "days_with_class": 1,
            "has_conflict": false,
            "num_conflicts": 0,
            "num_friends": 0,
            "time_on_campus": 24.8
        }
    ]
}
```


Create Timetable Share Link

description This endpoint creates a new share link and stores it in the db

endpoint `/share/link`

view `create_share_link`

request format example

response format

Load Share Timetable Link

description This endpoint looks up share link in the db

endpoint `/share/[code]`

view `share_timetable`

request format example

response format

Basic Course Search

description This endpoint is used to search based on string query (main in search bar)

endpoint `/search/[school]/[semester name]/[year]/[query]/`

view `course_search`

request format example

response format

Advanced Course Search

description This endpoint is used to do search from advanced search modal

endpoint `/advanced_search/`

view `advanced_course_search`

request format example

response format

CHAPTER 4

Contribution Pipeline

Pre Commit Hooks

lint staged

Travis CI

Code Review

Readability, Zappr

Deployment

CHAPTER 5

Indices and tables

- genindex
- modindex
- search