

UConn Course Searcher

John Silva and Jeremy Goode



Problem we're solving

UConn's course search is terrible on
Student Admin - slow and impossible to
filter by specific criteria

01

FRONT-END

React App

02

BACK-END

API using Node.js
and PostgreSQL





01

FRONT-END

React App

What is React?

Component-based JavaScript frontend library for building interfaces (specifically, single-page applications)

Why did we use React?

- It's simple
- Easy to manage state
- Re-use our components throughout the app



REACT APP

UConn Course Finder

Search Settings

Keywords (course #/title)

Campus Storrs

Semester Fall 2022

Subject Any

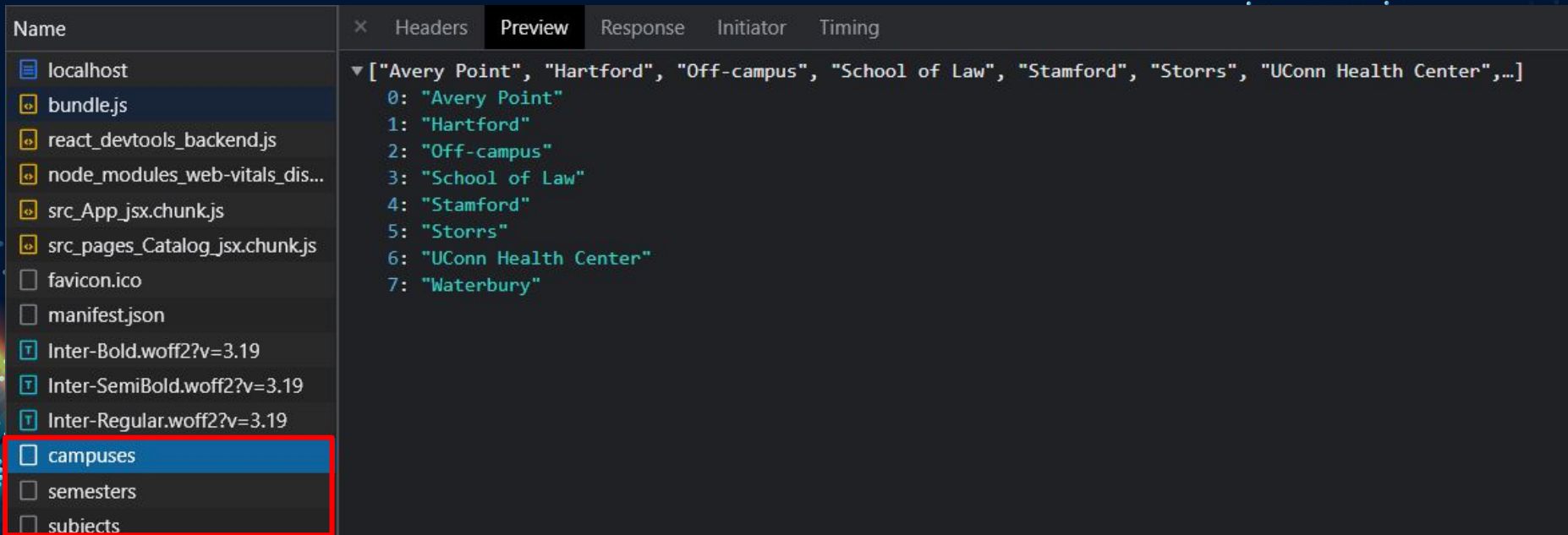
Course Criteria

- ☒ Quantitative (Q)
- ☒ Environmental (E)
- ☒ No Competencies
- ☒ CA1
- ☒ CA3
- ☒ Online
- ☒ Writing (W)
- ☒ International (INT)
- ☒ No Content Areas
- ☒ CA2
- ☒ CA4
- ☒ In Person

Reset Search

Our app calls backend API to get active campuses, semesters and subjects, then calls the course search to load results

Getting data for homepage



The screenshot displays a web browser's developer tools interface, specifically the 'Preview' tab of a REST client. The left sidebar lists various files, with 'campuses' highlighted. The main area shows a JSON array of campus names.

| Name | Headers | Preview | Response | Initiator | Timing |
|--------------------------------|---------|---------|----------|-----------|--------|
| localhost | | | | | |
| bundle.js | | | | | |
| react_devtools_backend.js | | | | | |
| node_modules_web-vitals_dis... | | | | | |
| src_App.jsx.chunk.js | | | | | |
| src_pages_Catalog.jsx.chunk.js | | | | | |
| favicon.ico | | | | | |
| manifest.json | | | | | |
| Inter-Bold.woff2?v=3.19 | | | | | |
| Inter-SemiBold.woff2?v=3.19 | | | | | |
| Inter-Regular.woff2?v=3.19 | | | | | |
| campuses | | | | | |
| semesters | | | | | |
| subjects | | | | | |

▼ ["Avery Point", "Hartford", "Off-campus", "School of Law", "Stamford", "Storrs", "UConn Health Center", ...]

- 0: "Avery Point"
- 1: "Hartford"
- 2: "Off-campus"
- 3: "School of Law"
- 4: "Stamford"
- 5: "Storrs"
- 6: "UConn Health Center"
- 7: "Waterbury"

Getting search results

▼ General

Request URL: `http://localhost:8000/courses?campus=Storrs&semester=Fall+2022&subjects=CSE&quantative=true&environmental=true&writing=true&international=true&hasCompetency=false&online=true&inPerson=true&ca1=true&ca2=true&ca3=true&ca4=true&hasContentArea=false`

Request Method: GET

Status Code: ● 200 OK

Remote Address: 127.0.0.1:8000

Referrer Policy: strict-origin-when-cross-origin

× Headers Payload Preview Response Initiator Timing

```
▼ [{"subject": "CSE", "subjectLong": "School of Engineering", "catalogNumber": "1010",...},...]  
▼ 0: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "1010",...}  
  cal: false  
  ca2: false  
  ca3: false  
  ca4: false  
  catalogNumber: "1010"  
  environmental: false  
  hasCompetency: false  
  hasContentArea: false  
  international: false  
  maxCredits: 3  
  minCredits: 3  
  quantative: false  
  subject: "CSE"  
  subjectLong: "School of Engineering"  
  title: "Intro Computing for Engineers"  
  writing: false  
▶ 1: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "1729",...}  
▶ 2: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "2050",...}  
▶ 3: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "2102",...}  
▶ 4: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "2301",...}  
▶ 5: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "2500",...}  
▶ 6: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "3100",...}  
▶ 7: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "3140",...}  
▶ 8: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "3150", title: "C++ Essentials",...}  
▶ 9: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "3300",...}  
▶ 10: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "3400",...}  
▶ 11: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "3500",...}  
▶ 12: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "3666",...}  
▶ 13: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "3800", title: "Bioinformatics",...}  
▶ 14: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "3802",...}  
▶ 15: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "4095",...}  
▶ 16: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "4099",...}  
▶ 17: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "4300",...}  
▶ 18: {subject: "CSE", subjectLong: "School of Engineering", catalogNumber: "4302",...}
```




02

BACK-END

Node.js process running a
webserver with Express using
PostgreSQL for data storage

Data Gathering (professors)

RateMyProfessor API

We found out the underlying RateMyProfessor API by looking at other packages source code, and using the same methods.

After finding all of UConn's professors, we store their name, average rating, and total number of ratings to display.

Data Gathering (courses)

Class Schedule Snapshots

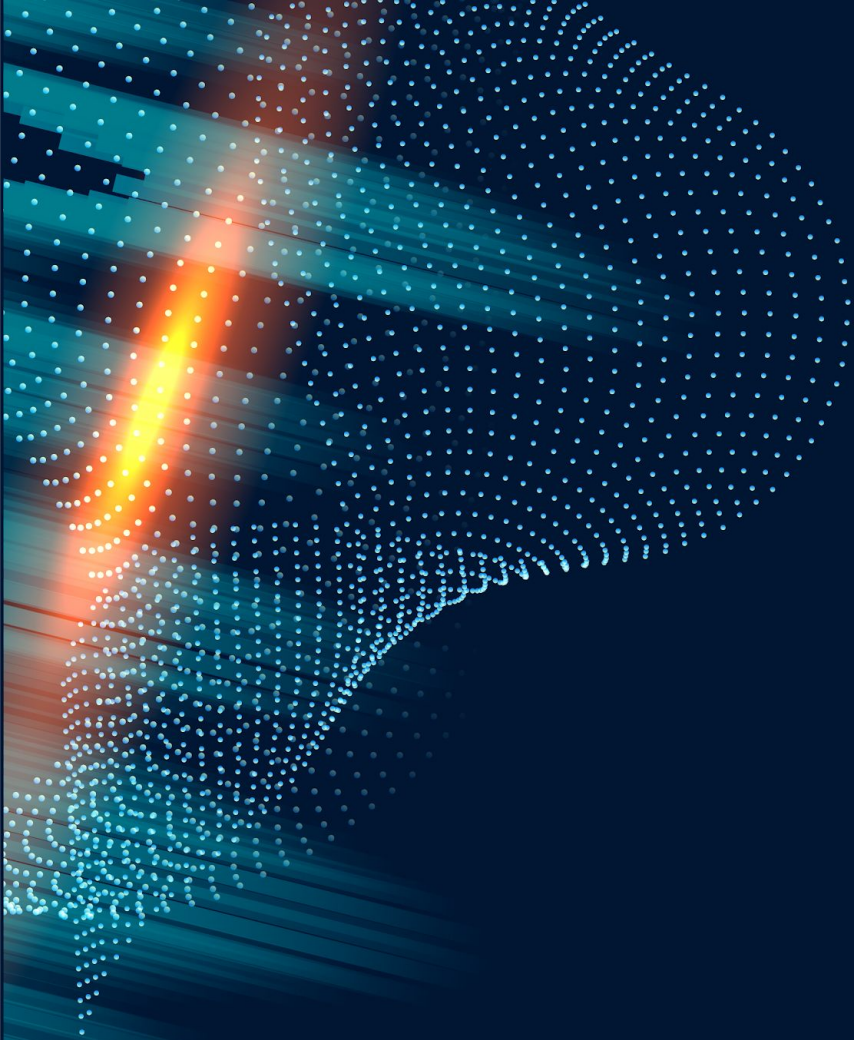
Every night, schedules of all classes available are refreshed and available through an Excel spreadsheet. We gather this information and parse it.

<https://scheduling.uconn.edu/class-schedule-snapshots/>

Data Storage

After parsing the data we then store all the data in our PostgreSQL instance for quick access anytime there is a request from a client.

We also link them to our professor ratings so we don't have to do it multiple times in the future.



Demo!