



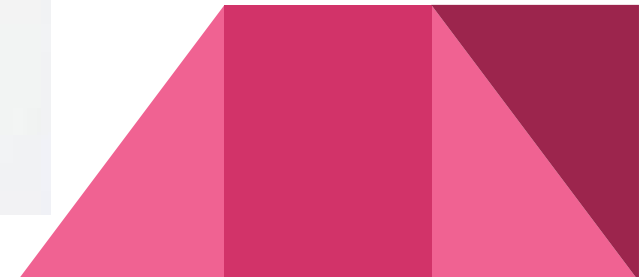
!Java presents

Schedule Master v1.0

Sam Burdick
Bridger Dunn
Hunter Kurtis
Jahrme Risner

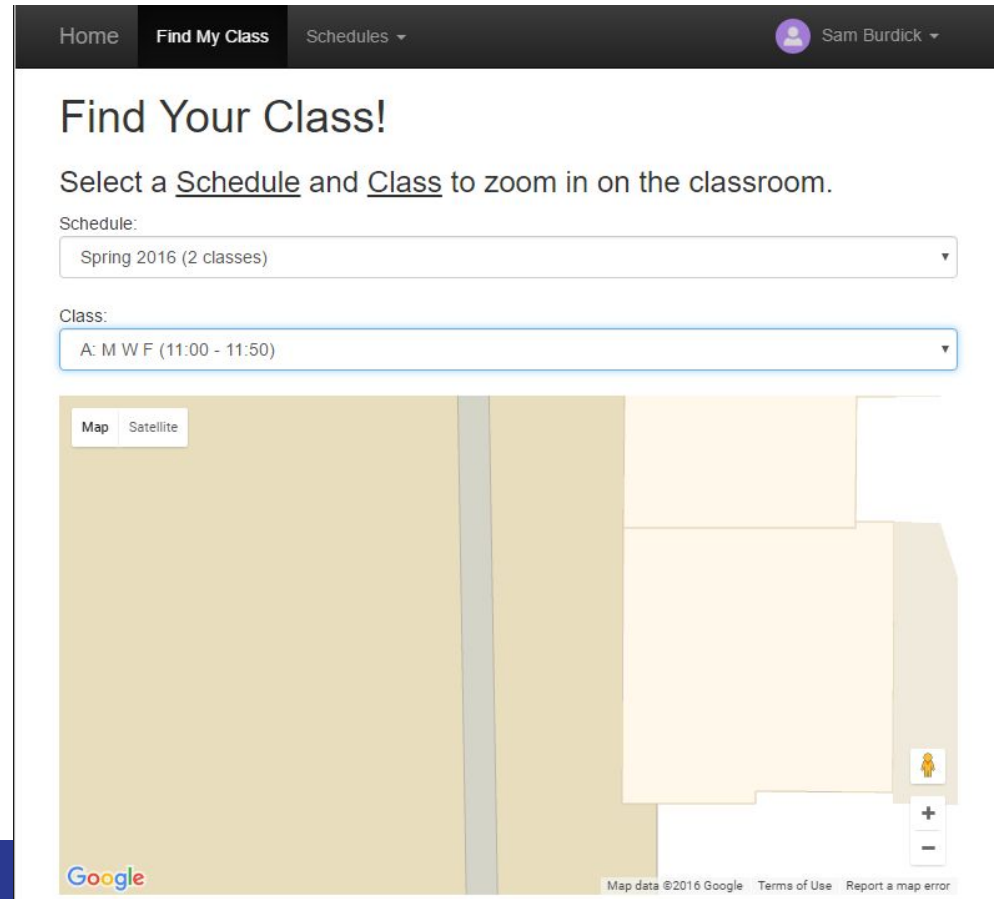
The Problem

- Students mindlessly wandering Thompson hallways
- Lack of knowledge of classroom locations
- Disorganized first year students



What is Schedule Master?

- It is an SAAS that allows students to create schedule, add classes
- Then, students can view their classes on an interactive map.



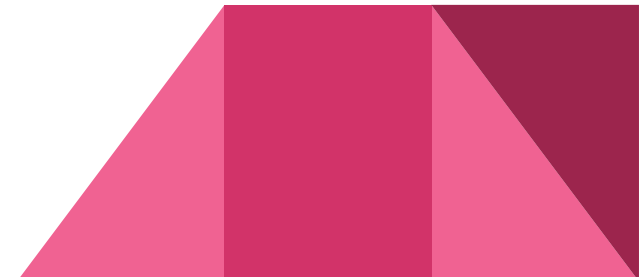
Development Stories

As a user, I want to...

1. Create Schedule, Add Class
2. View Class on Map
3. Share Schedule
4. View Schedule on Calendar

Developer story:

- All of the term and class data is stored in a Mongo database.
- This data is subject to change, so I want a flexible, modular database structure.



App Demo

Volunteer?

Url: <http://not-java.herokuapp.com/>

Password is:

Password123!

Sign up to create your schedules!

First Name

Guest

Last Name

User

Email

g@u.com

Username

guest

Password

••••••••••

Password Requirements

100%

Sign up

or Sign in

Create New Schedule, Add Class

New schedule

Term

Spring 2016 ▼

Create

Add class to Spring 2016 itinerary

Department

Computer Science ▼

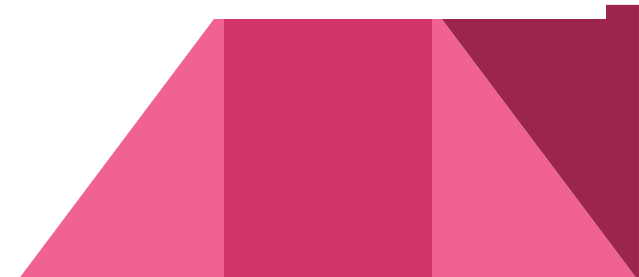
Course

161 - Intro to Computer Science ▼

Section


A - M W F, 9:00-9:50 ▼

Add Class



Map

In order to find their classroom, the user can select their schedule, and upon choosing a class, the map zooms in on the location associated with the class.

[Home](#) [Find My Class](#) [Schedules](#)  Sam Burdick

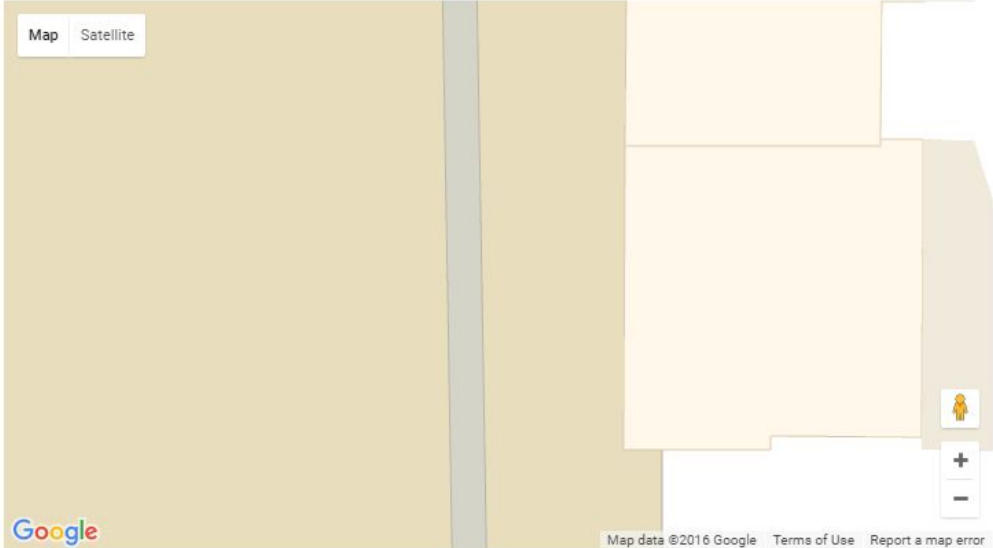
Find Your Class!




Select a Schedule and Class to zoom in on the classroom.


Schedule:

Class:

Map Satellite



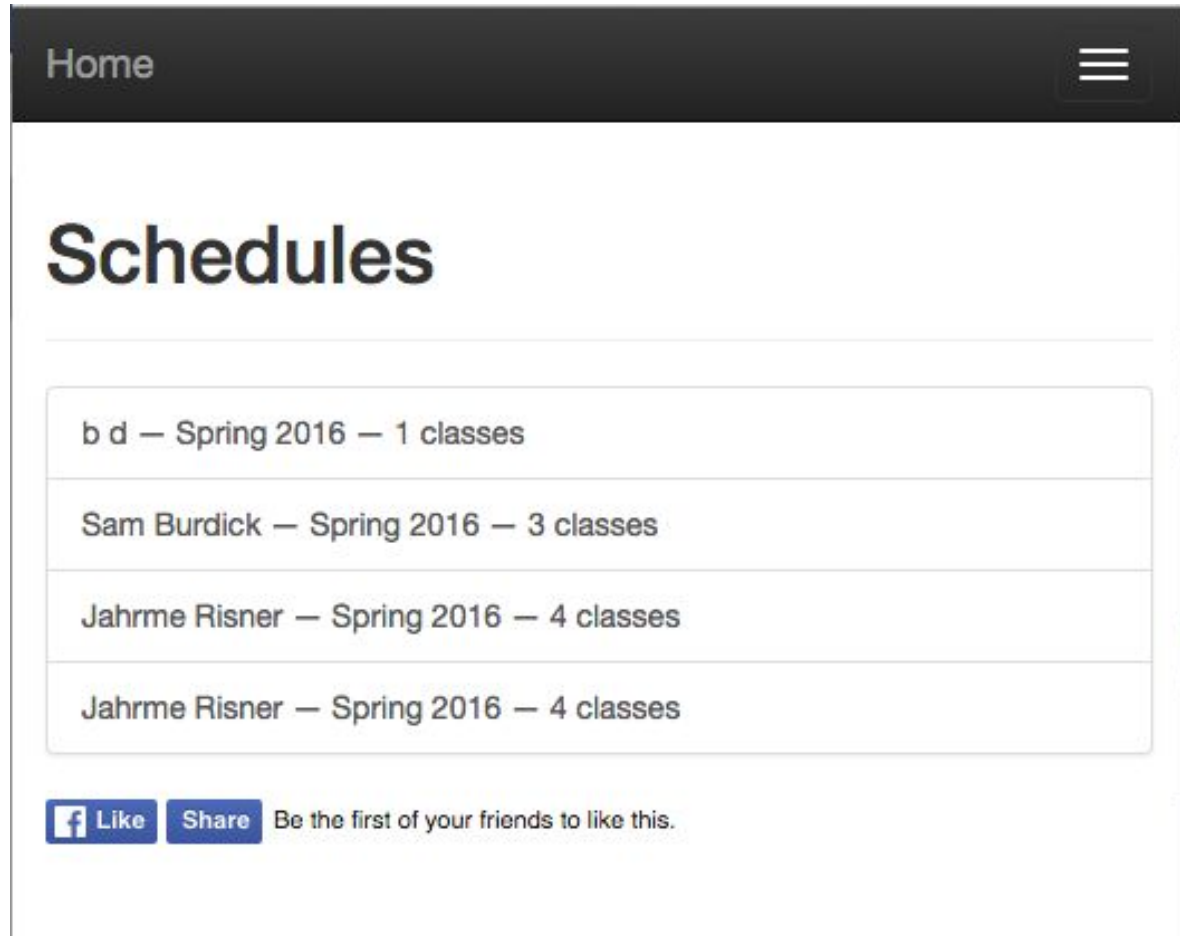




 Map data ©2016 Google [Terms of Use](#) [Report a map error](#)

Share Button


Allows users to share their schedule on Facebook

Sometimes appears on the bottom of the View Schedule page



The screenshot shows a web application interface. At the top is a dark header with the word 'Home' on the left and a hamburger menu icon on the right. Below the header, the main content area has the title 'Schedules' in a large, bold font. Underneath the title is a list of four schedule items, each in a light gray box with a thin border. The items are: 'b d — Spring 2016 — 1 classes', 'Sam Burdick — Spring 2016 — 3 classes', 'Jahrme Risner — Spring 2016 — 4 classes', and 'Jahrme Risner — Spring 2016 — 4 classes'. At the bottom of the page, there is a Facebook sharing section. It includes a Facebook 'f' logo, the text 'Like', a blue 'Share' button, and the text 'Be the first of your friends to like this.' The bottom right corner of the image features a decorative graphic of overlapping pink and red triangles.

Schedule
b d — Spring 2016 — 1 classes
Sam Burdick — Spring 2016 — 3 classes
Jahrme Risner — Spring 2016 — 4 classes
Jahrme Risner — Spring 2016 — 4 classes

 Like [Share](#) Be the first of your friends to like this.

MongoDB: ObjectIds

In the database, documents reference other documents by their ObjectId.

Observe that departments are referenced in this way: (as viewed in RoboMongo)

Key	Value	Type
▼ (1) ObjectId("5724e418e1822056a9...")	{ 3 fields }	Object
_id	ObjectId("5724e418e1822056a943a0db")	ObjectId
termName	Spring 2016	String
▼ departments	[1 element]	Array
[0]	ObjectId("5724e4b0e1822056a943a0dc")	ObjectId

In this example, 'ObjectId("5724e4b0e1822056a943a0dc")' references the Computer Science department.

The upshot: if one attribute of a document is changed, then other documents referencing it can link to the new data. That way we don't have to 'hard-code' every attribute in the documents, just each attribute's ID.

The Database

Hierarchy: Term > Department > Course > Section > Location, Instructor

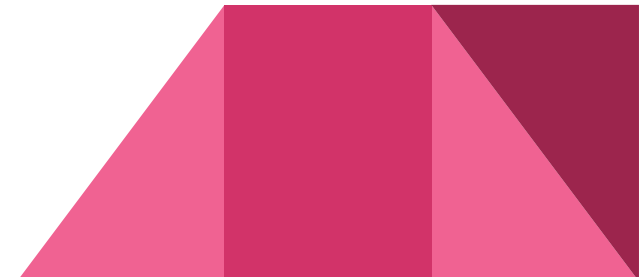
Term requires: departments

Department requires: courses

Courses require: sections

Section requires: locations, instructors

Location/Instructor: stored separately



Computer Science Department & Courses Table

Key	Value	Type
▼ (1) ObjectId("5724e4b0e1822056a9...")	{ 4 fields }	Object
<input type="checkbox"/> _id	ObjectId("5724e4b0e1822056a943a0dc")	ObjectId
<input type="checkbox"/> name	Computer Science	String
▼ <input type="checkbox"/> courses	[13 elements]	Array
<input type="checkbox"/> [0]	ObjectId("5724d166e1822056a943a0bd")	ObjectId
<input type="checkbox"/> [1]	ObjectId("5724d347e1822056a943a0bf")	ObjectId
<input type="checkbox"/> [2]	ObjectId("5724d44fe1822056a943a0c3")	ObjectId
<input type="checkbox"/> [3]	ObjectId("5724d553e1822056a943a0c6")	ObjectId
<input type="checkbox"/> [4]	ObjectId("5724d604e1822056a943a0c8")	ObjectId
<input type="checkbox"/> [5]	ObjectId("5724d64ce1822056a943a0ca")	ObjectId
<input type="checkbox"/> [6]	ObjectId("5724d6f9e1822056a943a0cd")	ObjectId
<input type="checkbox"/> [7]	ObjectId("5724d767e1822056a943a0cf")	ObjectId
<input type="checkbox"/> [8]	ObjectId("5724d7e8e1822056a943a0d2")	ObjectId
<input type="checkbox"/> [9]	ObjectId("5724d867e1822056a943a0d4")	ObjectId
<input type="checkbox"/> [10]	ObjectId("5724da4ce1822056a943a0d6")	ObjectId
<input type="checkbox"/> [11]	ObjectId("5724dae9e1822056a943a0d8")	ObjectId
<input type="checkbox"/> [12]	ObjectId("5724db36e1822056a943a0da")	ObjectId
<input type="checkbox"/> term	ObjectId("5724e418e1822056a943a0db")	ObjectId

Computer Science Course table

db.getCollection('courses').find({})		
local localhost:27017 mean-dev		
db.getCollection('courses').find({})		
courses 0.001 sec.		
Key	Value	Type
▶ (1) ObjectId("5724d166e1822056a9...")	{ 4 fields }	Object
▶ (2) ObjectId("5724d347e1822056a9...")	{ 4 fields }	Object
▶ (3) ObjectId("5724d44fe1822056a9...")	{ 4 fields }	Object
▼ (4) ObjectId("5724d553e1822056a9...")	{ 4 fields }	Object
_id	ObjectId("5724d553e1822056a943a0c6")	ObjectId
name	Software Engineering	String
courseNumber	240	Double
sections	[1 element]	Array
▶ (5) ObjectId("5724d604e1822056a9...")	{ 4 fields }	Object
▶ (6) ObjectId("5724d64ce1822056a9...")	{ 4 fields }	Object
▶ (7) ObjectId("5724d6f9e1822056a9...")	{ 4 fields }	Object
▶ (8) ObjectId("5724d767e1822056a9...")	{ 4 fields }	Object
▶ (9) ObjectId("5724d7e8e1822056a9...")	{ 4 fields }	Object
▶ (10) ObjectId("5724d867e1822056a9...")	{ 4 fields }	Object
▶ (11) ObjectId("5724da4ce1822056a9...")	{ 4 fields }	Object
▶ (12) ObjectId("5724dae9e1822056a9...")	{ 4 fields }	Object
▶ (13) ObjectId("5724db36e1822056a9...")	{ 4 fields }	Object

Software Engineering Section table

Key	Value	Type
▶ (1) Objectld("5723cab2e1822056a9...)	{ 5 fields }	Object
▶ (2) Objectld("5724d077e1822056a9...)	{ 5 fields }	Object
▶ (3) Objectld("5724d09ae1822056a9...)	{ 5 fields }	Object
▶ (4) Objectld("5724d31ae1822056a9...)	{ 5 fields }	Object
▶ (5) Objectld("5724d3a8e1822056a9...)	{ 5 fields }	Object
▶ (6) Objectld("5724d3d4e1822056a9...)	{ 5 fields }	Object
▶ (7) Objectld("5724d400e1822056a9...)	{ 5 fields }	Object
▼ (8) Objectld("5724d518e1822056a9...)	{ 5 fields }	Object
_id	Objectld("5724d518e1822056a943a0c5")	Objectld
name	A	String
instructor	Objectld("57238ea4e1822056a943a0b3")	Objectld
location	Objectld("5724d499e1822056a943a0c4")	Objectld
▶ times	{ 2 fields }	Object
▶ (9) Objectld("5724d5dde1822056a9...)	{ 5 fields }	Object
▶ (10) Objectld("5724d638e1822056a...)	{ 5 fields }	Object
▶ (11) Objectld("5724d6b5e1822056a...)	{ 5 fields }	Object
▶ (12) Objectld("5724d734e1822056a...)	{ 5 fields }	Object
▶ (13) Objectld("5724d7bbe1822056a...)	{ 5 fields }	Object
▶ (14) Objectld("5724d83ce1822056a...)	{ 5 fields }	Object
▶ (15) Objectld("5724d8a1e1822056a...)	{ 5 fields }	Object
▶ (16) Objectld("5724dab3e1822056a...)	{ 5 fields }	Object
▶ (17) Objectld("5724db17e1822056a...)	{ 5 fields }	Object

Location table

Key	Value	Type
▶ ⓘ (1) ObjectId("57239012e1822056a9...)	{ 3 fields }	Object
▶ ⓘ (2) ObjectId("57239035e1822056a9...)	{ 3 fields }	Object
▶ ⓘ (3) ObjectId("57239065e1822056a9...)	{ 3 fields }	Object
▼ ⓘ (4) ObjectId("5724d499e1822056a9...)	{ 3 fields }	Object
_id	ObjectId("5724d499e1822056a943a0c4")	ObjectId
name	Thompson 399	String
▼ ⓘ coords	[2 elements]	Array
[0]	47.263439	Double
[1]	-122.482846	Double
▶ ⓘ (5) ObjectId("5724d69ce1822056a9...)	{ 3 fields }	Object
▶ ⓘ (6) ObjectId("5724d795e1822056a9...)	{ 3 fields }	Object



Instructor Table

Key	Value	Type
▶ ⓘ (1) ObjectId("57238de4e1822056a9...")	{ 2 fields }	Object
▼ ⓘ (2) ObjectId("57238ea4e1822056a9...")	{ 2 fields }	Object
_id	ObjectId("57238ea4e1822056a943a0b3")	ObjectId
name	Anthony Mullen	String
▶ ⓘ (3) ObjectId("57238eb4e1822056a9...")	{ 2 fields }	Object
▶ ⓘ (4) ObjectId("57238edbe1822056a9...")	{ 2 fields }	Object
▶ ⓘ (5) ObjectId("57238eebe1822056a9...")	{ 2 fields }	Object

Calendar

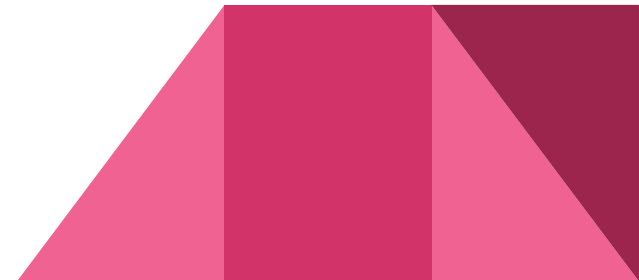
A calendar like this one would be placed next to the map.

The user should be able to select one of his/her classes (in a box), making the location of that class appear on the map.

<< previous week							
Week of 8/29/2016 - 9/4/2016							
next week >>							
Show Week of 08/29/2016  Start Time 8:00AM End Time 6:00PM  refresh calendar							
Time	Monday Aug 29	Tuesday Aug 30	Wednesday Aug 31	Thursday Sep 1	Friday Sep 2	Saturday Sep 3	Sunday Sep 4
8:00AM							
9:00AM	MATH 433 - A Lecture 9:00AM - 9:50AM Thompson Hall 374	MATH 433 - A Lecture 8:30AM - 9:20AM Thompson Hall 374		MATH 433 - A Lecture 8:30AM - 9:20AM Thompson Hall 374	MATH 433 - A Lecture 9:00AM - 9:50AM Thompson Hall 374		
10:00AM							
11:00AM							
12:00PM							
1:00PM	MATH 420 - A Lecture 1:00PM - 1:50PM Thompson Hall 374	MATH 420 - A Lecture 1:00PM - 1:50PM Thompson Hall 374		MATH 420 - A Lecture 1:00PM - 1:50PM Thompson Hall 374	MATH 420 - A Lecture 1:00PM - 1:50PM Thompson Hall 374		
2:00PM	CSCI 315 - A Lecture 2:00PM - 2:50PM Thompson Hall 399		CSCI 315 - A Lecture 2:00PM - 2:50PM Thompson Hall 399		CSCI 315 - A Lecture 2:00PM - 2:50PM Thompson Hall 399		
3:00PM							

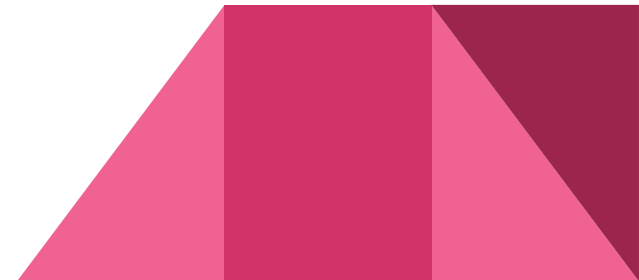
Difficulties

1. Consistency of Facebook share button.
 - Page refresh issues - it does not always appear
2. Ultimately, we could not find an appropriate Angular calendar module or implement our own.
3. Creating the database structure.
 - Routing the data from backend to frontend and displaying it properly



What worked well: Specialization, Agile

- Development of multiple individual modules that could be independently tested
- Frequently tested finished code, schemas
- Calendar: early feature, but scrapped towards the end
- Putting it all together



Questions?

