Good **Afternoon.**

Programming - REC 2018

Meet the Developers

Paul

Ian

Maks

Taylor

Software Engineering Students

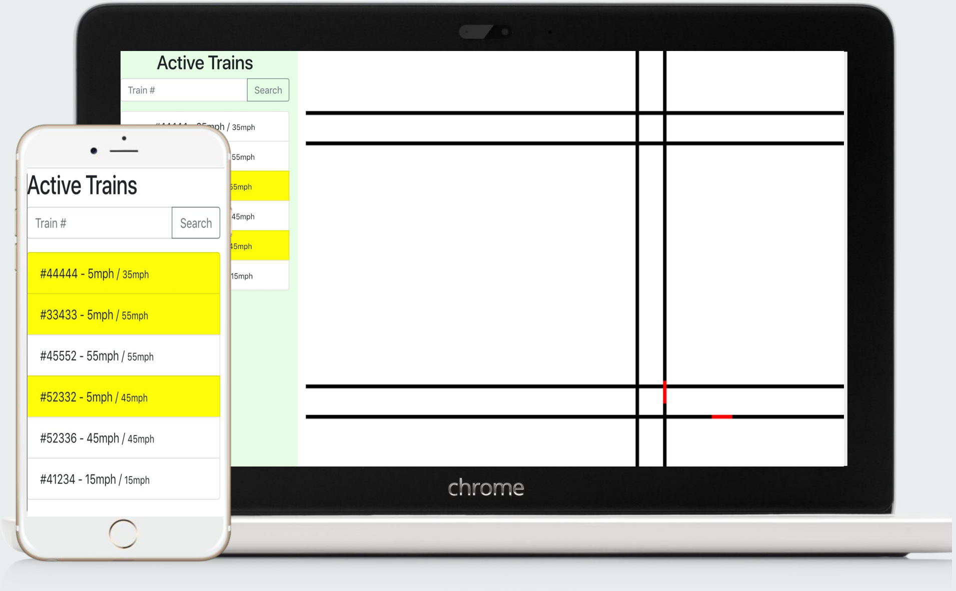
Our Solution

Problem Statement

“The city is holding a public opening for a program that can simulate a control panel for the current signalling system. This program must be able to control (and monitor) speeds of the trains, control (and monitor) the train timings and general logistics, update the user on malfunctions in the train and incorporate an automatic collision prevention system. Your company is requested to produce this program to be applied to the current subway system and also be able to fulfill needs of future upgrades (such as new transit lines and stations)”

TrainTrax

Trains on time



Active Trains

Train # | Search

- #44444 - 5mph / 35mph
- #33433 - 5mph / 55mph
- #45552 - 55mph / 55mph
- #52332 - 5mph / 45mph
- #52336 - 45mph / 45mph
- #41234 - 15mph / 15mph

TrainTrax

-Focus on the seven main functions (outlined in section 3.2)

-Flexible, responsive webapp

-Runs on all devices, all screen sizes

-Scalable, enterprise ready backend, with open API

-Industry standard tools allow easy implementation

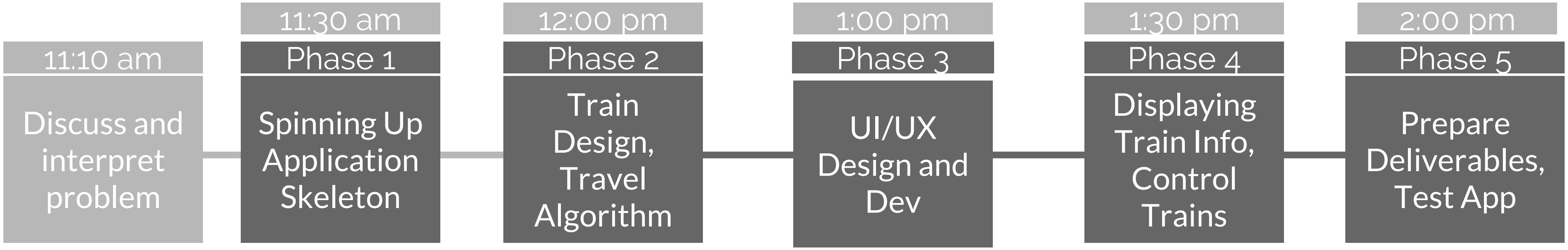
-RESTful API

Planning & Management

Planning & Management

- Agile development: short sprints, regular team meetings
- Used our whiteboard to design features before implementation
- Every developer took on a role to maximize work
- Pair programmed important features to reduce error
- Developed RESTful API web service
- Ensured scalable and enterprise ready practices were followed
- Social media login for validation/persistence
- Used GitHub for source control

Timeline



11:10 am
Discuss and interpret problem

11:30 am
Phase 1
Spinning Up Application Skeleton

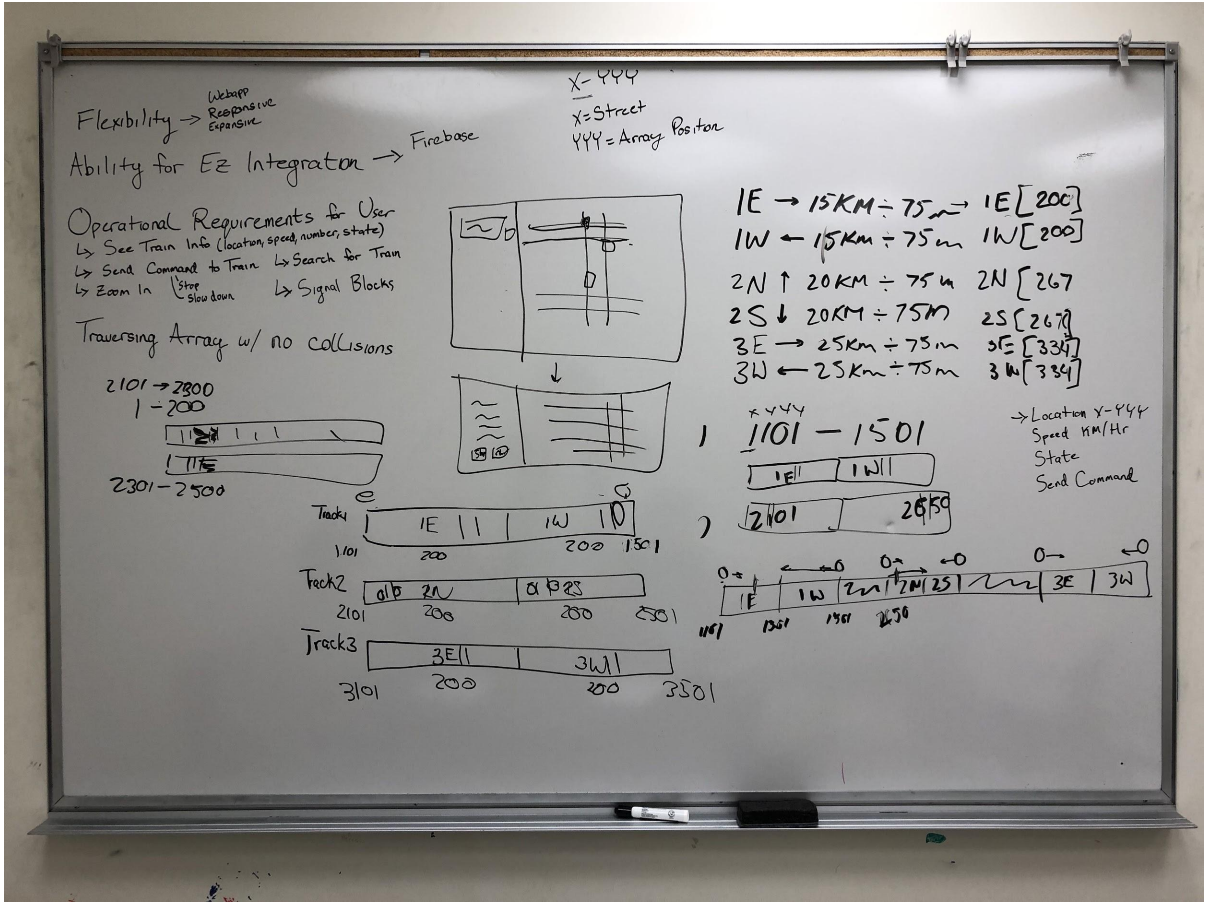
12:00 pm
Phase 2
Train Design, Travel Algorithm

1:00 pm
Phase 3
UI/UX Design and Dev

1:30 pm
Phase 4
Displaying Train Info, Control Trains

2:00 pm
Phase 5
Prepare Deliverables, Test App

The Whiteboard



Flexibility → Webapp
Responsive
Expansive

Ability for Ez Integraton → Firebase

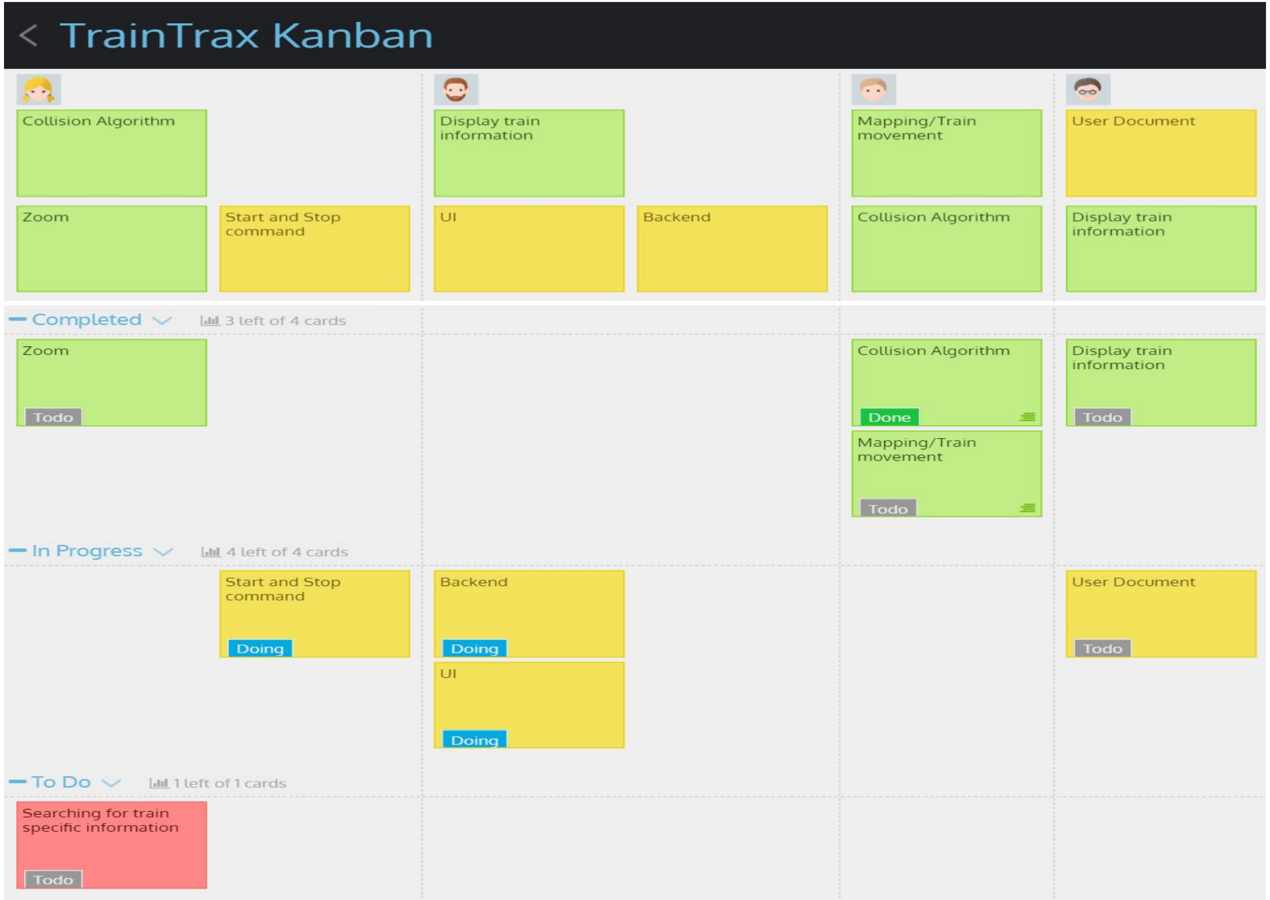
X-YYY
X=Street
YYY=Array Positon

Operational Requirements for User
↳ See Train Info (location, speed, number, state)
↳ Send Command to Train ↳ Search for Train
↳ Zoom In ↳ Stop
Slow down
↳ Signal Blocks

Traversing Array w/ no collisions

1E → 15KM ÷ 75m → 1E[200]
1W ← 15Km ÷ 75m 1W[200]
2N ↑ 20KM ÷ 75m 2N[267
2S ↓ 20KM ÷ 75M 2S[267]
3E → 25Km ÷ 75m 3E[334]
3W ← 25Km ÷ 75m 3W[334]

→ Location X-YYY
Speed KM/Hr
State
Send Command



TrainTrax Kanban

	Collision Algorithm Zoom Start and Stop command	Display train information UI Backend	Mapping/Train movement Collision Algorithm	User Document Display train information
Completed — 3 left of 4 cards	Zoom (Todo)		Collision Algorithm (Done) Mapping/Train movement (Todo)	Display train information (Todo)
In Progress — 4 left of 4 cards	Start and Stop command (Doing)	Backend (Doing) UI (Doing)		User Document (Todo)
To Do — 1 left of 1 cards	Searching for train specific information (Todo)			

The KanBan

The Tech

Algorithm Analysis

- “Traverse the array without collisions”
- Breakup the tracks into positional arrays
- Trains continuously run and update location in real time
- Collision zones (safe zones) at track intersections
- Only utilize signal blocks close to intersections

Tech Stack

Enterprise Standard Frameworks

-ReactJS

-WebPack

-NodeJS

-Bootstrap4

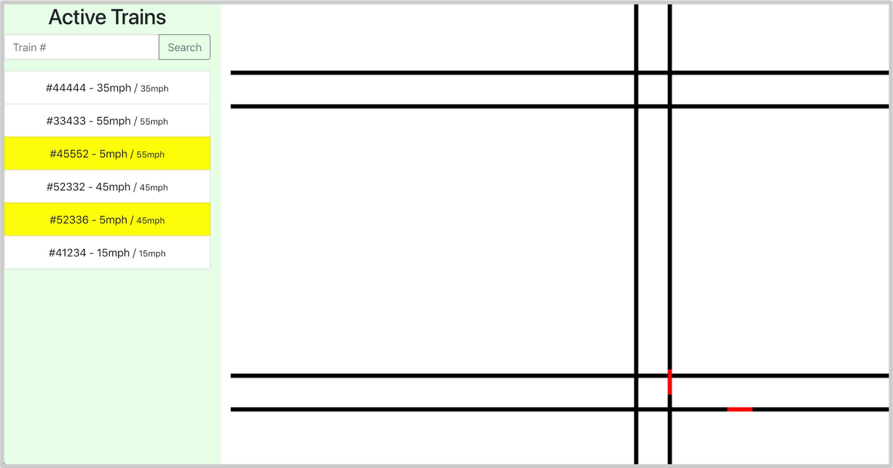
-PhaserJS

-Lots of Love

Features

3.2.1 - Display location of trains at all times

- TrainTrax has a map that occupies 2/3rds of the screen space
- On this map, you can see the trains traverse the tracks in real time



Active Trains

#44444 - 35mph / 35mph

#33433 - 55mph / 55mph

#45552 - 5mph / 55mph

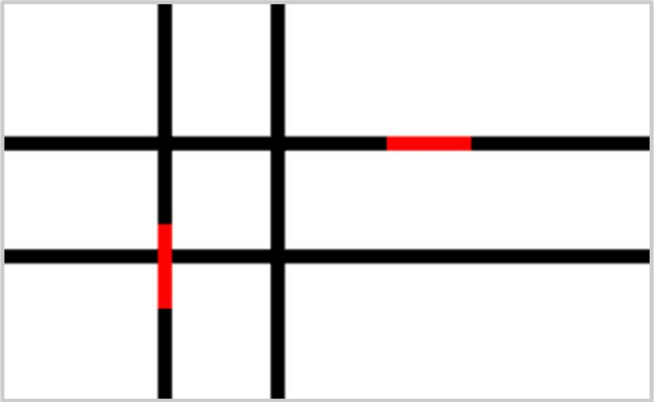
#52332 - 45mph / 45mph

#52336 - 5mph / 45mph

#41234 - 15mph / 15mph

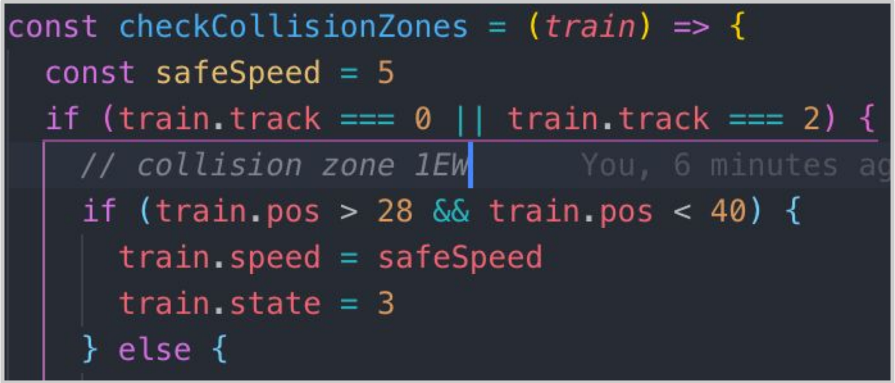
3.2.2 - Zoom in on train tracks

- Using a scroll wheel or touch pad, you are able to to zoom in on the map
- Once zoomed, click and drag to ‘pan’ around the map
- Intuitive controls standard to most webapps



3.2.3 - Utilize signal block concept

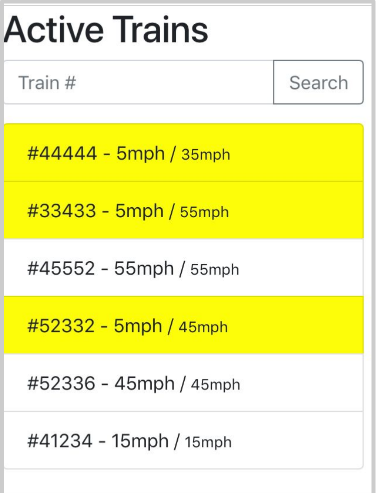
- Each train will slow down down when approaching intersections
- If two trains get close, they will slow down
- Using our algorithm, we are able to have 6 trains navigate the map and not collide



```
const checkCollisionZones = (train) => {
  const safeSpeed = 5
  if (train.track === 0 || train.track === 2) {
    // collision zone 1EW        You, 6 minutes ag
    if (train.pos > 28 && train.pos < 40) {
      train.speed = safeSpeed
      train.state = 3
    } else {
```

3.2.4 - Search for trains and display information

- Using the navbar on the left hand side, you can search for trains
- Once you search using the trains ID, information will be displayed



#44444 - 5mph / 35mph

#33433 - 5mph / 55mph

#45552 - 55mph / 55mph

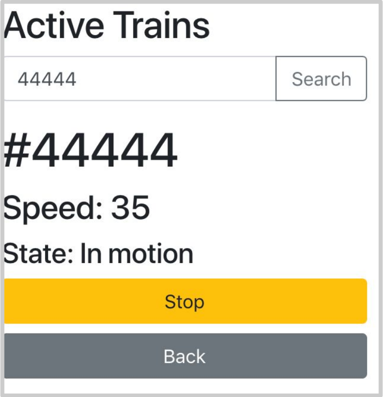
#52332 - 5mph / 45mph

#52336 - 45mph / 45mph

#41234 - 15mph / 15mph

3.2.5 - Able to send commands to trains

- Once you have searched for a train, you have the option to control the train using buttons
- Includes “Stop” and “Slow Down” as per functional requirements



Active Trains

44444

#44444

Speed: 35

State: In motion

3.2.6/7 - Architecture & System Variables

- Using industry standard tech, our architecture is stable and easily readable
- As we have developed a webapp, it is usable on all platforms

Demo

Trivia

Lines of Code: Lots

Merge Conflicts: 0

Chicken Skewers Consumed: 16

Tzatziki Consumed: ~ 1 Cup

Stress Levels: High

Questions?