Team Fire Nation - Nicole Cheng, Ziyan Lin, Yikai Wang*, Jessica Yang
*project manager

Overview: In this Flask app, we will provide users with real-time MTA updates: where trains are, how far they are from the user's location, and any major delays. This allows the grouchy New Yorker to be prepared for train troubles, should they occur. It will also provide an estimate of how much time one has to get to the station before the train arrives, aka whether or not one has enough time to make that much-needed stop at Starbucks.

Note: this app uses the *gtfs-realtime-bindings* module
To install:
pip install --upgrade gtfs-realtime-bindings

This module helps parse GTFS-realtime data feed, where each entity looks somewhat like these:

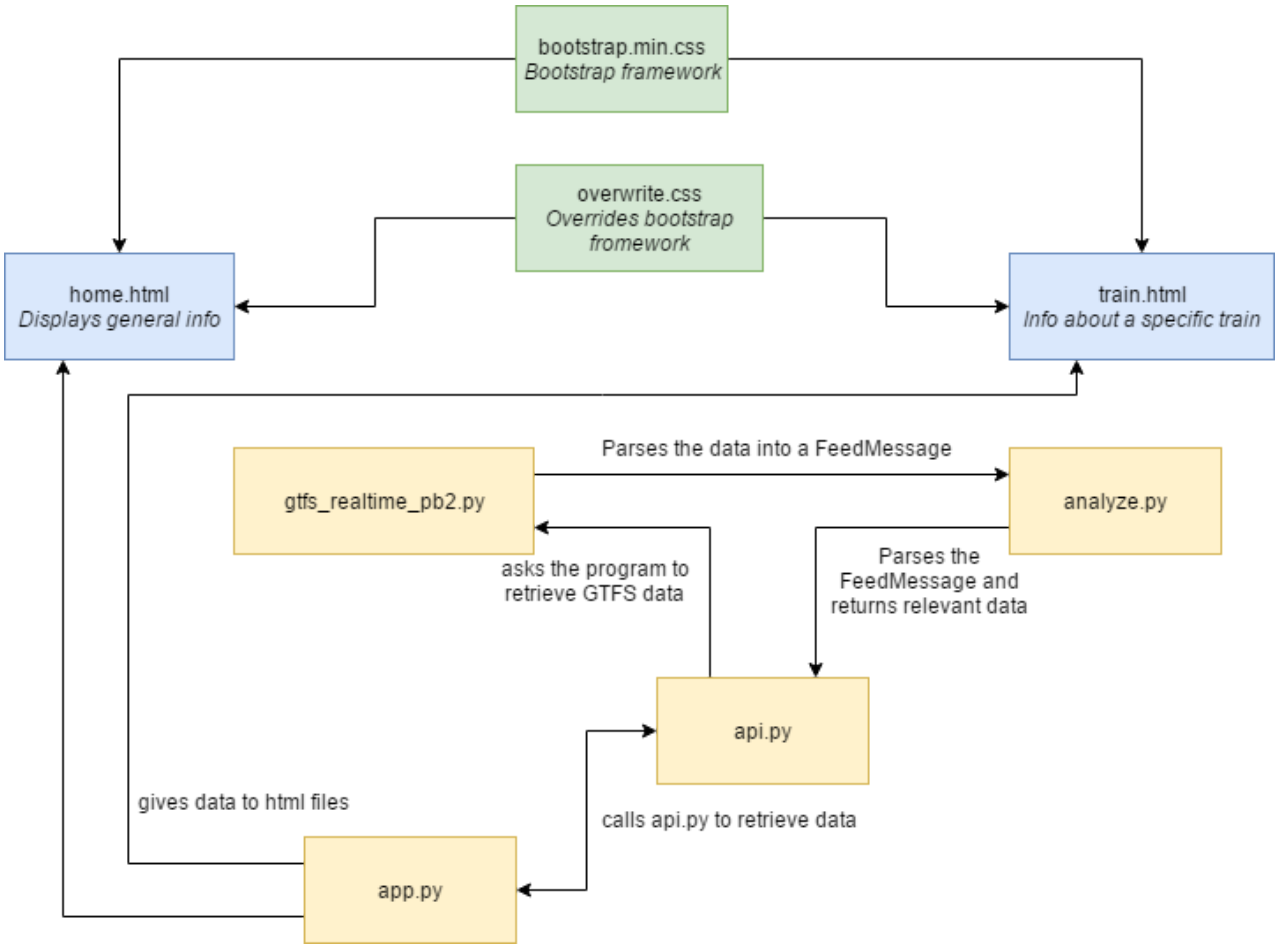
Example 1:

```
id: "000296"
trip_update {
  trip {
    trip_id: "131350_GS.S01R"
    start_date: "20170511"
    route_id: "GS"
  }
  stop_time_update {
    departure {
      time: 1494554010
    }
    stop_id: "902S"
  }
  stop_time_update {
    arrival {
      time: 1494554100
    }
    stop_id: "901S"
  }
}
```

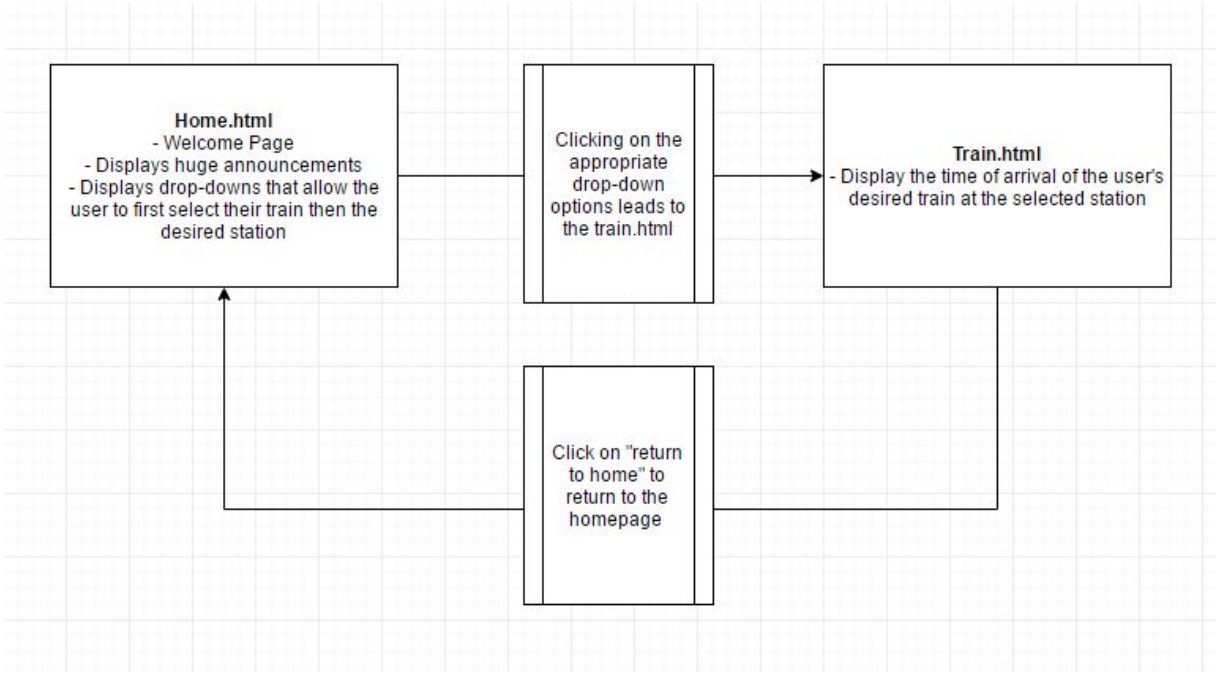
Example 2:

```
id: "000297"
alert {
  informed_entity {
    trip {
      trip_id: "128000_2..S01R"
      route_id: "2"
    }
  }
  informed_entity {
    trip {
      trip_id: "101350_4..N64R"
      route_id: "4"
    }
  }
  informed_entity {
    trip {
      trip_id: "058750_6..N01X004"
      route_id: "6"
    }
  }
  header_text {
    translation {
      text: "Train delayed"
    }
  }
}
```

Component Map:



Site Map:



Component Description:

- home.html : the home page, welcome page, huge announcements (smaller ones will be on individual train pages), user can click on a tab to redirect to specific train (train.html)
- train.html : uses jinja to match the user's tab request (aka their train)
- app.py : does the appy / Flasky stuff
- utils/api.py : takes user input from app.py and calls the API. Calls gtfs_realtime_pb2.py to parse the data (in GTFS-realtime) returned by the API, then feeds that into analyze, and finally returns the "user-friendly" information to app.py to be rendered to user
- utils/gtfs_realtime_pb2.py : Google's python file that parses data into a FeedMessage
- utils/analyze.py : iterates over the FeedMessage stuff and gives it to api.py
- The rest would just be bootstrap and overwrite.css (which overwrites some bootstrap formats).

File distribution:

```
/firenation
     app.py
     /utils
          gtfs_realtime_pb2.py
          analyze.py
          api.py
     /templates
          home.html
          train.html
     /static
          Bootstrap stuff
          overwrite.css
```

Possible Additions:

- Pinpoint user's location to help determine which station to look at using a GPS API
- Provide alternate routes if there are delays

Styling guide:

- We don't use camel-case.

Task Delegation:

- Nicole: MTA API stuff
- Yikai: Front-end bootstrap / javascript / jinja
- Ziyan: Front-end bootstrap / javascript / jinja
- Jessica: Flask

Timeline:

Tentative Date	Task
05/16 (Tuesday)	home.html with links to train.html pages
05/16 (Tuesday)	Working flask app
05/16 (Tuesday)	Confirmation that the MTA API works for the project
05/19 (Friday)	FeedMessage parser working
05/23 (Tuesday)	Crudely functional website
05/26 (Friday)	Bootstrap layout done
05/29 (Monday)	Override CSS done
06/02 (Friday)	Working javascript
06/07 (Wednesday)	Testing/debugging
06/12 (Monday)	Entirely functional website