# locoMotive Design Document

## I. Identification

**Project/Team Name:** locoMotive
**Project Leader:** Brian On (bon@princeton.edu)
**Team Members:**
- Adam Berman (agberman@princeton.edu)
- Trey Todnem (ttodnem@princeton.edu)

## II. Overview

We will create a hybrid mobile application that streamlines the train-riding experience for Princeton students. Most students frequently use the trains to travel to and from select destinations, such as the university itself and the nearby airports. Thus, we hope to create a mobile application that allows our fellow students to quickly and easily access train schedules for these locations. Moreover, the application will offer easy access to the maps for these select routes and GPS visualization for a given route so users will be able to track their progress. We hope to use this GPS tracking to offer another helpful feature: alerts, in the form of an alarm or push notification, that will notify users when they are nearing their destination so they don't miss their stop.

## III. Requirements and Target Audience

For Princeton students, accessing the NJ Transit and SEPTA train schedules online is a tedious affair. Most students only travel to and from a handful of destinations, like Princeton, Princeton Junction, Newark Liberty International Airport, Penn Station New York, Trenton, and Philadelphia 30th Street Station. However, both the web browser and mobile applications offer a long list of places through which Princeton student users must nevertheless sift. Furthermore, the mobile application lags -- possibly due to bloat from features that most students probably don't use (e.g. the bus feature). Both the web and mobile applications also lack certain features that would greatly improve one's train riding experience, such as facile access to maps of relevant routes and mobile alerts to let a user know when nearing a destination.

However, for Princeton students, accessing the NJ Transit and SEPTA train schedules is also a necessary affair. Consequently, we hope to create a mobile application that will serve as a substitute for the NJ Transit/SEPTA applications and greatly simplify the rail system experience for them. While useful for everyone, we believe it will be especially helpful for uninitiated freshmen or prospective students. As stated in the overview, our application will only deliver train schedules and maps for select destinations frequently accessed by students, which will greatly reduce the complexity of the UI compared to the official NJ Transit/SEPTA ones. Furthermore, we hope to use GPS tracking to allow users to visually keep track of their progress along a

given route, as well as notify them when they near their destination. This will prevent users from missing their stops, e.g. if they fall asleep.

## IV. Functionality

### Scenario 1: Herbie Needs a Train

Herbie is a typical Princeton student: moderately lazy with a tendency to leave things until the last minute. It's well past midnight and he's quite exhausted: he just finished packing. However, he has a flight scheduled at 11:00 AM departing from the Newark Airport tomorrow morning. Consequently, because he intends to take the train from Princeton to the airport, he will have to wake up pretty early in the morning. All he wants to do is go to sleep, but he just remembered he hasn't checked the train schedules on the NJ Transit so he knows when he should leave campus in the morning. Fortunately, he downloaded locoMotive and manages to find the appropriate train with just a few taps, rather than having to waste several minutes dealing with the bloated NJ Transit application.

### Scenario 2: Herbie Fully Loaded

Herbie makes it to the connecting train after the dinky, which heads directly to the Newark Airport. However, he stayed up until 3 AM last night packing and had to wake up at 7 AM to get to the Princeton station on time. With only four hours of sleep under his belt, Herbie feels drowsiness weighing heavily on his eyelids. It's quite tempting to drift off for Herbie: the train ride is about an hour long. Normally, one would have to endure the sleepiness while riding a train; if one fell asleep, one might miss one's stop. However, thanks to locoMotive, Herbie can rest easy. He simply opens the app and enables destination notification for his route. Now, he will receive an alarm when he approaches his stop. He falls asleep with a smile on his face (and wakes up at the appropriate time).

### Scenario 3: Herbie the Newbie

Herbie is a freshman heading home for spring break. He's currently riding the Dinky to Princeton Junction, where he will board a train bound for Trenton. His ultimate destination is the Philadelphia International Airport. But he's a bit nervous; he has only ridden the regional rail system a couple times and he's still not sure at which stops he should get off. The route to the Philadelphia airport is particularly intricate, as it involves several connections (Princeton → Princeton Junction → Trenton → Philadelphia 30th Street → Philadelphia airport). But Herbie, for the most part, is not worried: last night, he downloaded locoMotive on his phone, which overlays his progress along a given route in real-time, allowing him to keep track of his location and determine how many stops are remaining for a given connection. With his application, he has the knowledge of a seasoned senior traveler and even helps out some fellow passengers who are unsure of *their* stops.

## V. Design

locoMotive is going to be a hybrid mobile application. The design of the application can be encapsulated by the classic 3-tier system: user interface, process, and data management.

**Process Backend and Data management:**

Our application will primarily deal with train schedules for data for the routes our application serves. We will scrape this data from the NJ Transit website using BeautifulSoup and store it on the cloud using Amazon Web Services' DynamoDB. The backend of our application will use Django REST Framework to query this data from the database and store it in models. The Django REST Framework allows us to easily use these models to create REST endpoints for frontend consumption through "serialization", where the models are converted into JSON and made available through views. The models also is where the business logic will happen: the first time we populate the database, it will be manually/separate from the application using our scraping functions; however after that, the application itself will have the ability to "refresh" the data, i.e. scrape the site for up-to-date data from time to time (e.g. every 3 days or something). Users will also have the ability to "refresh" train schedules whenever they want (through PUT/GET requests from the views), i.e. re-scrape the data to ensure the schedule is accurate. The scraping, whether automatic or initiated by the user, and updating of the database will occur in the models.

**User interface (frontend)**

The frontend of our application will be designed using Ionic, which is an HTML mobile app development framework. We will create our UI using a mixture of Ionic's available mobile components/animations, AngularJS (Ionic makes extensive use of AngularJS in its framework), and our own Javascript, CSS, and HTML. We will also be using AngularJS to use the REST endpoints (the views) made available by our backend to manipulate the models (get, put, or update data). Ionic uses Cordova under the hood, which will allow our application to access native device features like push notifications and GPS. Finally, Cordova will also wrap our web application in a native skin for us to it can be deployed to Android and iPhone devices like a native application. We will be using the Google Maps API (Javascript) to implement route visualization and GPS tracking on the frontend.

## VI. Timeline

| | |
|---|---|
| Successfully scrape desired information from NJTransit website | 20 March |

| | |
|---|---|
| Create basic interface to interact with data in a web application | 26 March |
| Create minimum viable backend to store collected data and communicate with front end in set intervals of time | 26 March |
| Present brief “pitch” to class | Exact date unknown at this time |
| Port to native Android application (we plan on using a hybrid platform so this should not involve rewriting the program) | 1 April |
| Implement GPS tracking of passenger | 10 April |
| Implement push notifications in alignment with GPS tracking | 10 April |
| Improve user interface for a clean, minimalistic style (this may have been accomplished before), tune up backend to scrape data more frequently within thirty or so minutes of a user’s train departure to ensure up-to-date information | 15 April |
| Use listservs and word of mouth to acquire a large user testing base to uncover overlooked bugs and further unclutter UI | 16 April |
| Prepare a sleek presentation about application | 16 April onward |
| Final presentation | Exact date unknown at this time |

## VII. Risks and Outcomes

### GPS Tracking

We will need to learn how to implement GPS tracking – we also do not know how reliable GPS will be on the trains, especially in tunnels. We do not want problems to arise when GPS tracking fails.

### Map Building

Ideally, we want to display a map from which the user can choose their destination . However, we want the map to be minimalistic, so this may require us to make our own maps. We would rather not have to do this, but it just depends what is available as we implement.

**Interface Questions**

The focal point of this app is sleekness and ease of use. While implementing our design, we will have to make decisions regarding simplicity versus functionality. We may be tempted to add lots of alternate functionality and user options, but that eventually will start to detract from the style. We will have to decide how much functionality is just the right amount.

**Data Scraping Reliability**

If the website from which we get our data goes down, we may not have up-to-date information. Also, considering how much we do not like the websites as they are, there is probably a good chance that they will be renovated. We'll have to redo our scraping system if this happens.