

# Team Noatpad

## Track N' Drive

[Team-Noatpad](#)

### Team Overview

Team Member Name	Github Usernames
Ian Torres (Team Manager)	itorres1994
Joseph Falco (Team Supervisor)	CaptainFalco
Abhinay Bushan	ab3721
Kaique Silva	Kaiquecs
Corey Schotte	casch0

### Project Overview

Our application's purpose is to keep track of the user's cars as well as repair and maintenance expenses with a unique and simple UI. The application is unique because it is able to combine multiple aspects of car maintenance from future repairs to statistics about the car's expenses. This tool when published would be offered for free.

### User Interface

#### User Profile

Track N' Drive

Vehicles ▾Technicians ▾Statistics ▾User ▾

Notifications

2010 Bugatti Chiron

Jacoby Lewis  
Change Filter  
Dec. 1, 2017  

Edit Done

Jacoby Lewis  
Change Oil  
Dec. 2, 2017  

Edit Done

Add

User

Change image

User Info

Name: Tim Richards

Edit Add

Costs and Repairs

2010 Bugatti Chiron

Change tires | \$100 | Jacoby Lewis | 191 North Pleasant Street, Amherst, Jiffy Lube | Dec. 12, 2017  

Edit Remove

Change tires | \$0 | Jacoby Lewis | 191 North Pleasant Street, Amherst, Jiffy Lube | Dec. 12, 2017  

Edit Remove

Add

The *user profile* view outlines for the user aspects. The page offers a listing of additional information about the user that he or she may find significant, listing future repairs that have been scheduled for a particular car, and repairs that have been done to a particular car. A user also has the ability to add, edit, or remove both future/past repairs.

## Car Profile

Rock 'N' Drive

Vehicles ▾Technicians ▾Statistics ▾User ▾

Notifications


2010 Bugatti Chiron

Jacoby Lewis  
Change Filter  
Dec. 1, 2017  
Edit Done

Jacoby Lewis  
Change Oil  
Dec. 2, 2017  
Edit Done

Add

2010 Bugatti Chiron

  
Change Image

Car Info

Car Operator: Tim Richards  
Year: 2010  
Make: Bugatti  
Model: Chiron  
Color: Black-blue  
Engine Type: 8.0 L (488 cu in) W16 with quad-turbocharger  
Oil Type: 10W-40  
Mileage: 10000  
VIN: 1D4GP21R34B557579  
Registration: 1VN131  
Edit

Costs and Repairs

2010 Bugatti Chiron

Change tires | \$100 | Jacoby Lewis | 191 North Pleasant Street, Amherst, Jiffy Lube | Dec. 12, 2017  
Edit Remove

Change tires | \$0 | Jacoby Lewis | 191 North Pleasant Street, Amherst, Jiffy Lube | Dec. 12, 2017  
Edit Remove

Add

*Car profile* gives the user the ability to view aspects about their car such as: the year, make, model, color, engine type, oil type, mileage, VIN, and registration number. A user also has the ability to add, edit, or remove both future/past repairs.

## Technician Profile

Rock 'N' Drive

Vehicles ▾Technicians ▾Statistics ▾User ▾

Notifications


2010 Bugatti Chiron

Jacoby Lewis  
Change Filter  
Dec. 1, 2017  
Edit Done

Jacoby Lewis  
Change Oil  
Dec. 2, 2017  
Edit Done

Add

Technician



Tech Info

Name: Jacoby Lewis  
Street: 191 North Pleasant Street  
City: Amherst  
Company: Jiffy Lube  
Edit

Costs and Repairs

2010 Bugatti Chiron

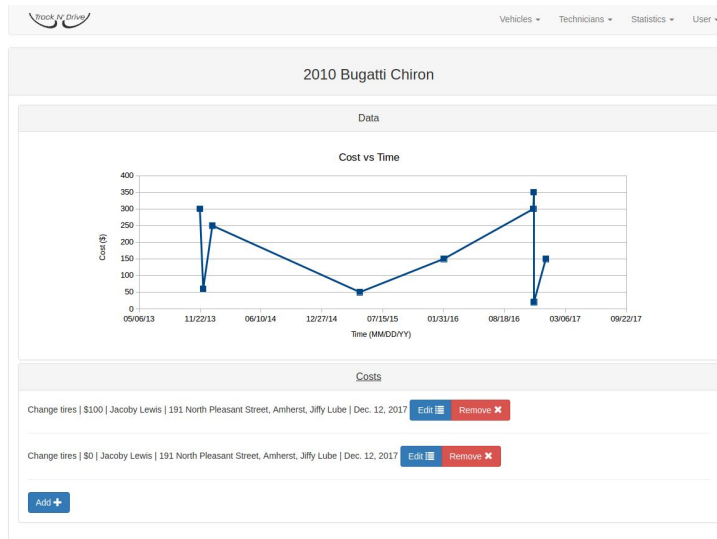
Change tires | \$100 | Jacoby Lewis | 191 North Pleasant Street, Amherst, Jiffy Lube | Dec. 12, 2017  
Edit Remove

Change tires | \$0 | Jacoby Lewis | 191 North Pleasant Street, Amherst, Jiffy Lube | Dec. 12, 2017  
Edit Remove

Add

*Technician profile* gives the user the ability to view the location of their technicians shop via a Google Map embedding as well as repairs that have been completed by this particular mechanic for a specific car. A user also has the ability to add, edit, or remove both future/past repairs.

## Statistics



*Statistics* gives the user the ability to view costs that have been incurred by a specific car as well as a graph of these listed costs vs the time it was repaired. Unfortunately, we have not gotten Google Graphs to work correctly so currently this page is not fully functional. However, the user still has the ability to add, edit, or remove repairs for the car.

## Settings

The screenshot shows the 'Settings' page. It has a navigation bar with 'Vehicles', 'Technicians', 'Statistics', and 'User'. The main content area is divided into two sections: 'Notification Settings' and 'Contact Settings'.

**Notification Settings**

Future Repair Notifications

2017-12-01: Change Oil	<a href="#">Edit</a>
2017-12-02: Change Filter	<a href="#">Edit</a>

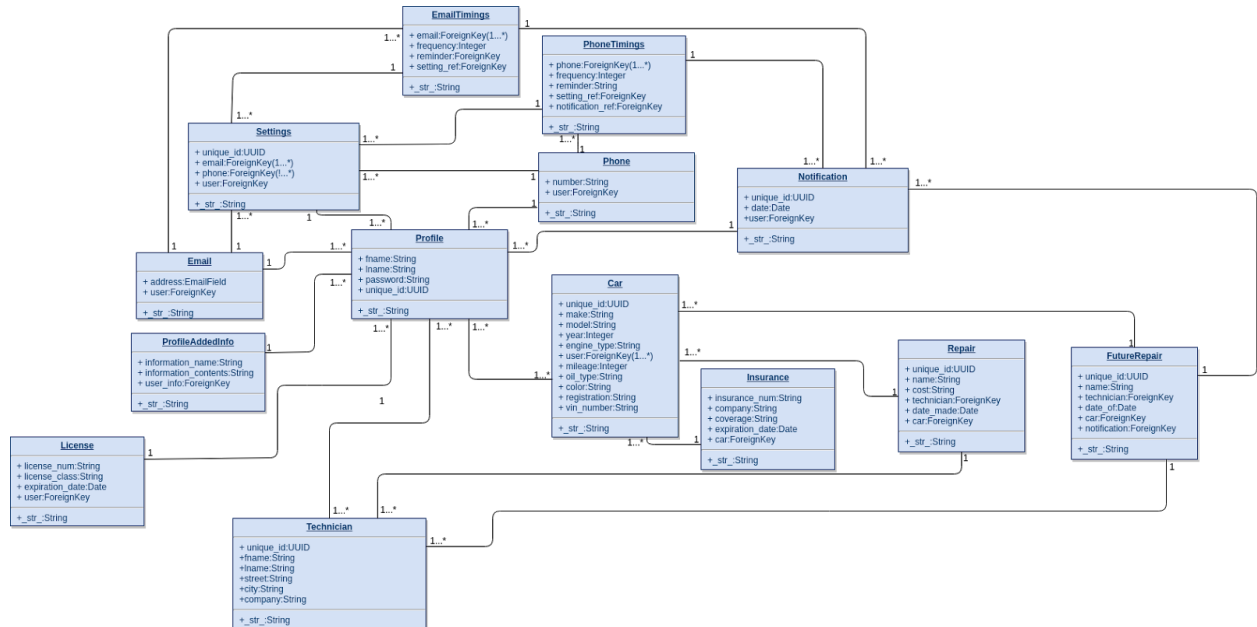
**Contact Settings**

Mobile Phone: 0000000000 [Edit](#)

Email: something@example.com [Edit](#)

*Settings* gives the user the ability to change when they will be notified about a future repair as well as editing their mobile phone number and email that will be used to notify the user of their future repair. Currently this view is not functional due to unintentional data model obfuscation.

## Data Model



### Profile

*Profile* is the central data model that shares connections with all aspects of the web application. These connections include: Email, Settings, Phone, Notification, Car, Insurance, Repair, Future Repair, Technician, License, and Profile Added Info. Besides these connections it holds standard user information: first name, last name, and a unique id.

### Email

*Email* holds relevant information such as an email address as well as sharing connections with Email Timings, Settings, and Profile.

### Settings

*Settings* has connections to Email, Email Timings, Phone, Phone Timings, and Profile. An identifying piece of data is the unique id.

### Email Timings

*Email Timings* contains the frequency in which a user will be notified, as well as sharing connections with Settings, Email, and Notifications.

### Notification

*Notification* holds the date that a user will be notified of a Future Repair, which is one of the connections it has, as well as connections with Email Timings, Phone Timings, and

Profile. However, this part of the data model has too many connections that are needless unfortunately, which causes the settings page to be non-functioning

### **Phone**

*Phone* stores the user's phone number as well as a connection with Phone Timings, Settings, and Notification.

### **Phone Timings**

*Phone Timings* contains the frequency in which a user will be notified, as well as sharing connections with Settings, Phone, and Notifications.

### **Car**

*Car* contains information about the make, model, year, engine type, oil type, mileage, color, registration, and VIN as well as a unique id. The model also shares connection with the Profile, Insurance, Future Repair, and Repair.

### **Insurance**

*Insurance* stores information such as: insurance number, insurance company, coverage, and expiration date. The model shares a connection with Car

### **Repair**

*Repair* stores information such as: a unique id, name of repair, cost, and date made, as well sharing a connection with Car and Technician

### **Future Repair**

*Future Repair* stores information such as: name of the future repair and date of repair, as well sharing a connection with Car, Technician, and Notification.

### **Technician**

*Technician* contains the first name and last name of the technician, street, city, and business name, as well as sharing a connection with Profile, Repair, and Future Repair.

### **License**

*License* contains the license number, license class, and expiration date, as well as a connection with Profile.

### **Profile Added Info**

*Profile Added Info* simply stores additional information the user may want (information name and information contents) and shares a connection with Profile.

## URL Routes/Mappings

```
url(r'^$', views.index, name='index'),
url(r'^edit-user$', views.edit_user, name='edit user'),
url(r'^car-prof-(?P<unique_id>[-\w]+)/$', views.car_prof, name='car'),
url(r'^tech-prof-(?P<unique_id>[-\w]+)/$', views.tech_prof, name='tech'),
url(r'^stats-(?P<unique_id>[-\w]+)/$', views.stats, name='stat'),
url(r'^setting$', views.setting, name='settings'),
url(r'^add-tech$', views.add_technician1, name='add technician'),
url(r'^tech-prof-(?P<unique_id>[-\w]+)/add-tech$', views.add_technician2, name='add technician'),
url(r'^add-car$', views.add_car, name='add car'),
url(r'^car-prof-(?P<unique_id>[-\w]+)/add-car-info$', views.add_car_info, name='add car info'),
url(r'^car-prof-(?P<unique_id>[-\w]+)/add-car$', views.edit_car, name='add car'),
url(r'^car-prof-(?P<unique_id>[-\w]+)/add-future-repair$', views.add_future_repair, name='add future repairs'),
url(r'^[-\w]+/edit-future-repair-(?P<unique_id>[-\w]+)/$', views.edit_future_repair, name='add future repairs'),
url(r'^edit-future-repair-(?P<unique_id>[-\w]+)/$', views.edit_future_repair, name='add future repairs'),
url(r'^remove-repair-(?P<unique_id>[-\w]+)/$', views.remove_repair, name='index'),
url(r'^[-\w]+/remove-repair-(?P<unique_id>[-\w]+)/$', views.remove_repair, name='index'),
url(r'^add-repair-(?P<unique_id>[-\w]+)/$', views.add_repair, name='index'),
url(r'^[-\w]+/add-repair-(?P<unique_id>[-\w]+)/$', views.add_repair, name='index'),
url(r'^edit-repair-(?P<unique_id>[-\w]+)/(?P<car_id>[-\w]+)/$', views.edit_repair, name='edit repairs'),
url(r'^[-\w]+/edit-repair-(?P<unique_id>[-\w]+)/(?P<car_id>[-\w]+)/$', views.edit_repair, name='edit repairs'),
url(r'^done-future-repair-(?P<unique_id>[-\w]+)/(?P<car_id>[-\w]+)/$', views.done_future_repair,
    name='done future repairs'),
url(r'^[-\w]+/done-future-repair-(?P<unique_id>[-\w]+)/(?P<car_id>[-\w]+)/$', views.done_future_repair,
    name='done future repairs'),
url(r'^add-user-info$', views.add_user_info, name='add_user_info'),
```

1. Renders the user profile view
2. Renders a form for editing the user profile
3. Renders a car profile view
4. Renders a technician profile view
5. Renders a statistics view
6. Renders a settings view
7. Renders a form for adding technicians
8. Renders a form for editing technician info
9. Renders a form for adding cars
10. Renders a form for adding car info (not in use)
11. Renders a form for editing a car
12. Renders a form for adding a future repair
13. Renders a form for editing a future repair
14. Renders a form for editing a future repair
15. Removes a repair
16. Removes a repair
17. Renders a form for adding a repair
18. Renders a form for adding a repair
19. Renders a form for editing a repair
20. Renders a form for editing a repair
21. Adds a repair via a future repair and deletes the future repair
22. Adds a repair via a future repair and deletes the future repair
23. Renders a form for adding additional user data

## Authentication/Authorization

Our authentication and authorization is handled via Django's standards. We restrict the user from accessing the entire web app unless they are logged in via view Http redirects to the

login page if the user is currently anonymous. The user has the permissions that are provided by the URL mappings outlined above.

## **Team Choice**

Our team choice components include a Google Maps Embedding in the Technician profile view as well as a Google Graph in the statistics view. However, we were unable to get Google Graphs to work with our web app.

## **Conclusion**

We have learned the importance of version control systems on the software workflow process in order to keep all team members up to date on new versions of the web app as well as measuring individual progress. Django's efficiencies such as url mapping, data modeling, and template rendering interfaces are understable as well as robust, which gave the team an edge during software development. Some of our team members had not taken a database course, so some of our team members were not able to contribute as much as they had wanted to during the data modeling phase. A hurdle the team was not able to resolve, unfortunately, was Google Graphs.