*Florida International University*

*School of Computing and Information Sciences*

Final Deliverable

TO DO LIST OPTIMIZER 1.0

**Team Members:** Salvador Ricardo, Daniel Gonzales, Manuel Garcia.

**Product Owner(s)**: Monique Ross

**Mentor(s)**: Monique Ross

**Instructor**: Masoud Sadjadi

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON INFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

## 

## Abstract

*To Do Optimizer is an application that automates the way task management and navigations are used together. By implementing the navigation into the Task management application, users will use an immersive application which will save them time and make their day more productive. Daily, while completing tasks at multiple locations, people have to hop from the task management app to the navigation app, and back again trying to find out which route is better, so they can complete their task. Entering in a whole guessing game to decide on the best route possible. To do List Optimizer completely takes the guessing out of the equation, and provides a simple route, from the starting point to the destination, it generates an always updating route which will save time both guessing and driving. Using the Google Maps API, we guaranteed a fast and productive way to complete the user’s tasks.*

## Table of Contents

[Abstract 3](#_Toc512547157)

[Table of Contents 4](#_Toc512547158)

[Introduction 9](#_Toc512547159)

[Current System 9](#_Toc512547160)

[Purpose of New System 10](#_Toc512547161)

[User Stories 11](#_Toc512547162)

[Implemented User Stories 11](#_Toc512547163)

[#666 - User Story Name: Setup Sign Up System [Front-End] 11](#_Toc512547164)

[#669 - User Story Name: Add Places 15](#_Toc512547165)

[#670 - User Story Name: Add tasks to places 19](#_Toc512547166)

[#671 - User Story Name: Show Itinerary and Total Time 23](#_Toc512547167)

[#673 - User Story Name: Setup Navigation Information 27](#_Toc512547168)

[#675 - User Story Name: Add Transition for Adding New Place While Navigating 31](#_Toc512547169)

[#676 - User Story Name: Update Route on Map View if Delay 35](#_Toc512547170)

[#679 - User Story Name: Track The Time Spent in Places 39](#_Toc512547171)

[#680 - User Story Name: Register User [Backend] 43](#_Toc512547172)

[#681 - User Story Name: Add Places to User [Backend] 47](#_Toc512547173)

[#683 - User Story Name: Add Task to Places [Backend] 51](#_Toc512547174)

[#684 - User Story Name: Delete Places When Tasks are Completed 55](#_Toc512547175)

[#685 - User Story Name: Delete Places Manually 59](#_Toc512547176)

[#687 - User Story Name: Update Places [Backend] 63](#_Toc512547177)

[#688 - User Story Name: Delete Task Manually [Backend] 67](#_Toc512547178)

[#690 - User Story Name: Create Scenes Basic Transitions 71](#_Toc512547179)

[#695 - User Story Name: Update User [Backend] 73](#_Toc512547180)

[#697 - User Story Name**: Delete User [Backend]** 77](#_Toc512547181)

[#703 - User Story Name: Center Map 81](#_Toc512547182)

[#704 - User Story Name: Overview Map 85](#_Toc512547183)

[#706 - User Story Name: Connect to google maps API to autocomplete place search 93](#_Toc512547184)

[#713 - User Story Name: Front End Restful API 101](#_Toc512547185)

[#714 - User Story Name: Move Backend to Production Server 105](#_Toc512547186)

[#715 - **User Story Name: Asynchronous Request of Possible routes** 107](#_Toc512547187)

[#716 - User Story Name: Improve Visuals in the Map View 109](#_Toc512547188)

[#717 - User Story Name: Delete task validation [Back-End] 111](#_Toc512547189)

[#718 - User Story Name: Add task validation [Back-End] 115](#_Toc512547190)

[#719 - User Story Name: Senior Project **Video** 119](#_Toc512547191)

[#721 - User Story Name: Senior Project Video 119](#_Toc512547192)

[#722 - User Story Name: Senior Project Poster 119](#_Toc512547193)

[#723 - User Story Name: Add place validation [Back-End] 120](#_Toc512547194)

[#724 - User Story Name: Delete place validation [Back-End] 124](#_Toc512547195)

[#725 - User Story Name: Senior Project Poster 128](#_Toc512547196)

[#726 - User Story Name: Senior Project Poster 128](#_Toc512547197)

[Non-Implemented User Stories 129](#_Toc512547198)

[#667 - Setup Login System [Frontend] 129](#_Toc512547199)

[#668 - Setup Reset Password [Frontend] 130](#_Toc512547200)

[#674 - View Places on the Map 131](#_Toc512547201)

[#677 - Add Favorite Places 132](#_Toc512547202)

[#678 - History of visited places 133](#_Toc512547203)

[Project Plan 134](#_Toc512547204)

[Hardware and Software Resources 135](#_Toc512547205)

[Sprint Plan 136](#_Toc512547206)

[Sprint 2 136](#_Toc512547207)

[Sprint 3 137](#_Toc512547208)

[Sprint 4 139](#_Toc512547209)

[Sprint 5 140](#_Toc512547210)

[Sprint 6 141](#_Toc512547211)

[System Design 142](#_Toc512547212)

[Architectural Patterns 142](#_Toc512547213)

[System and Subsystem Decomposition 142](#_Toc512547214)

[Deployment Diagram 143](#_Toc512547215)

[Design Pattern 143](#_Toc512547216)

[System Validation 144](#_Toc512547217)

[Glossary 150](#_Toc512547218)

[Appendix 151](#_Toc512547219)

[Appendix A – UML Diagrams 151](#_Toc512547220)

[#666 - User Story Name: Setup Sign Up System [Front-End] 151](#_Toc512547221)

[#669 - User Story Name: Add Places 154](#_Toc512547222)

[#670 - User Story Name: Add tasks to places 157](#_Toc512547223)

[#671 - User Story Name: Show Itinerary and Total Time 160](#_Toc512547224)

[#673 - User Story Name: Setup Navigation Information 163](#_Toc512547225)

[#675 - User Story Name: Add Transition for Adding New Place while Navigating 166](#_Toc512547226)

[#676 - User Story Name: Update Route on Map View if Delay 169](#_Toc512547227)

[#679 - User Story Name: Track The Time Spent in Places 172](#_Toc512547228)

[#680 - User Story Name: Register User [Backend] 175](#_Toc512547229)

[#681 - User Story Name: Add Places to User [Backend] 178](#_Toc512547230)

[#683 - User Story Name: Add Task to Places [Backend] 181](#_Toc512547231)

[#684 - User Story Name: Delete Places When Tasks are Completed 184](#_Toc512547232)

[#685 - User Story Name: Delete Places Manually 187](#_Toc512547233)

[#687 - User Story Name: Update Places [Backend] 190](#_Toc512547234)

[#688 - User Story Name: Delete Task Manually [Backend] 193](#_Toc512547235)

[#690 - User Story Name: Create Scenes Basic Transitions 196](#_Toc512547236)

[#695 - User Story Name: Update User [Backend] 197](#_Toc512547237)

[#697 - User Story Name: Delete User [Backend] 200](#_Toc512547238)

[#703 - User Story Name: Center Map 203](#_Toc512547239)

[#704 - User Story Name: Overview Map 206](#_Toc512547240)

[#705 - User Story Name: Display Markers of Places in the Map 209](#_Toc512547241)

[#706 - User Story Name: Connect to google maps API to autocomplete place search 212](#_Toc512547242)

[#707 - User Story Name: Create New Entry Scene 215](#_Toc512547243)

[#713 - User Story Name: Front End Restful API 218](#_Toc512547244)

[#714 - User Story Name: Move Backend to Production Server 221](#_Toc512547245)

[#715 - User Story Name: Asynchronous Request of Possible routes 222](#_Toc512547246)

[#716 - User Story Name: Improve Visuals in the Map View 223](#_Toc512547247)

[#717 - User Story Name: Delete task validation [Back-End] 224](#_Toc512547248)

[#718 - User Story Name: Add task validation [Back-End] 227](#_Toc512547249)

[#723 - User Story Name: Add place validation [Back-End] 230](#_Toc512547250)

[#724 - User Story Name: Delete place validation [Back-End] 233](#_Toc512547251)

[Appendix B – User Interface Design 236](#_Toc512547252)

[Sign In/Sign Up Screen 236](#_Toc512547253)

[Itinerary List Screen (Empty) 237](#_Toc512547254)

[Itinerary List Screen (filled) 238](#_Toc512547255)

[Itinerary List Screen (places expanded) 239](#_Toc512547256)

[Map overview Screen (Navigation NOT started) 240](#_Toc512547257)

[Map center Screen (Navigation Started) 241](#_Toc512547258)

[Task List Screen (When Arrived to place) 242](#_Toc512547259)

[Appendix C – Sprint Review Reports 243](#_Toc512547260)

[Sprint 2 243](#_Toc512547261)

[Sprint 3 244](#_Toc512547262)

[Sprint 4 245](#_Toc512547263)

[Sprint 5 246](#_Toc512547264)

[Sprint 6 247](#_Toc512547265)

[Appendix D – Sprint Retrospective Reports 248](#_Toc512547266)

[Sprint 2 248](#_Toc512547267)

[Sprint 3 249](#_Toc512547268)

[Sprint 4 250](#_Toc512547269)

[Sprint 5 251](#_Toc512547270)

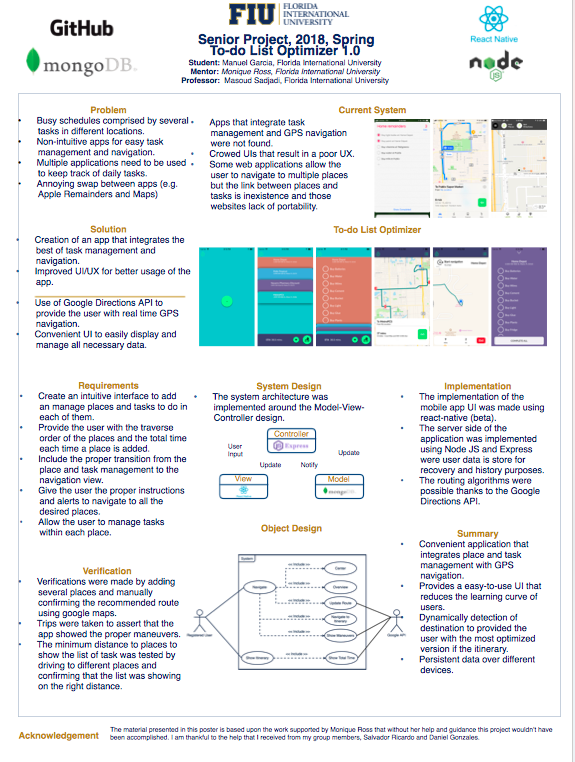
[Sprint 6 252](#_Toc512547271)

[Appendix F - User Manuals, Installation/Maintenance Document, Shortcomings/Wish list Document and other documents 253](#_Toc512547272)

[Videos 253](#_Toc512547273)

[Posters 254](#_Toc512547274)

[Salvador Ricardo 254](#_Toc512547275)

[Manuel Garcia 255](#_Toc512547276)

[Daniel Gonzalez 256](#_Toc512547277)

# Introduction

To Do-list Optimizer attempts to be a solution for the busy day schedule. It is meant to be a mobile application to make it accessible at any time to the user. The main focus of the Todo-list Optimizer is to combine the best of routing and task management algorithms in an easy-to-use and improved user interface (UI) that enhances the user experience (UX). Todo-list optimizer provides a seamless integration between navigation and task management, improving efficiency using intelligent routing.

# Current System

The current system consists of applications such as Google Maps, Waze, Apple Reminders, TodoList, among others. These systems work well as independent modules but it does not exist an application that combines those features (task management and efficient routing). Some applications(e.g. MapQuest, Apple Remainders) have attempted to combine these obtaining non-intuitive user interfaces or excessive steps to accomplish a goal. Overall, there are no application in the market that combines the task management with routing algorithm in an efficient and pleasant way or within the same application.

# Purpose of New System

To Do-List Optimizer!{Current.System}

To Do-List optimizer attempts to improve the daily schedule by providing a better integration between task management and routing algorithms. The application enables the user to create tasks (e.g. milk, vegetables, toothpaste, etc.) associated with places (e.g. Walgreens, Home Depot, Walmart, Home) to find the optimal route to accomplish them. It serves not only as a navigation system, but also, once the user is in the desired place, he or she can follow up with the task related to it. The application allows the user to save his or her favorite places so the next time it will be easy to access and them. Another feature is that the application learns user patterns (e.g. time that the user spends in an specific place or type of places) allowing it to predict more accurately the length of the solving all the tasks.

# User Stories

The following section provides the detailed user stories that were implemented in this iteration of the To Do-List Optimizer project. These user stories served as the basis for the implementation of the project’s features. This section also shows the user stories that are to be considered for future development.

## Implemented User Stories

### #666 - User Story Name: Setup Sign Up System [Front-End Back-End]

* Description: As a user, I would like to be able to register with the system so I can only I have access to my tasks and itineraries.

Acceptance Criteria

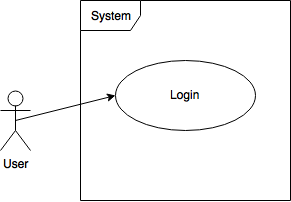
* Verify the user enters a valid username(email) and password in order to register.
* Verify if the user’s username and password are not valid, User must be notified and reenter a valid username and password in order to register.
* Verify if the username is in use, then verify the User
* Once registered, the User will be able to navigate through all of the application views.

Use Case

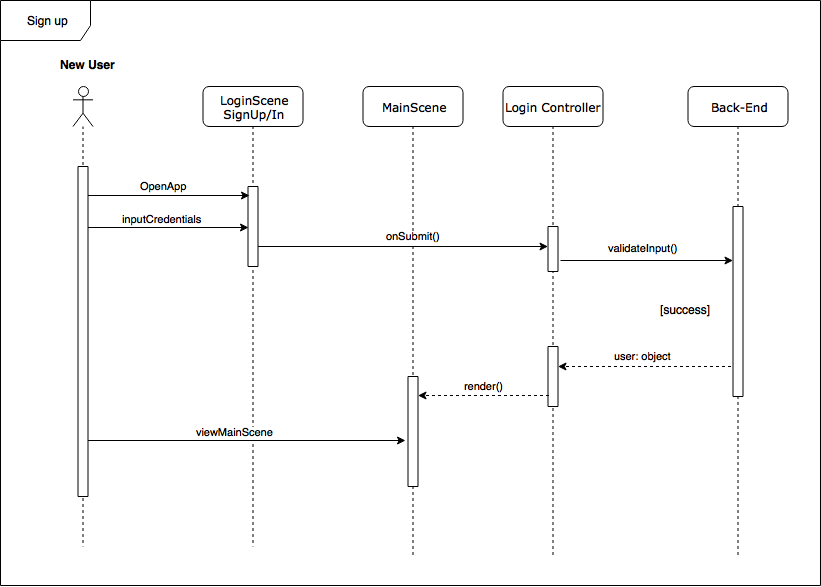
* Name: Setup Sign Up System
* Actor: User
* Preconditions:
  + User is required to have email account.
* Description:

1. Use case starts when users open app for the first time and the login scene is rendered.
2. The system validates inputted credentials.
3. The system signs up the user to the To-do List app and redirects the user to the main scene.
4. Alternate Flow of Events
5. Validation of user credentials fail.
6. Prompt the user to try other credentials.

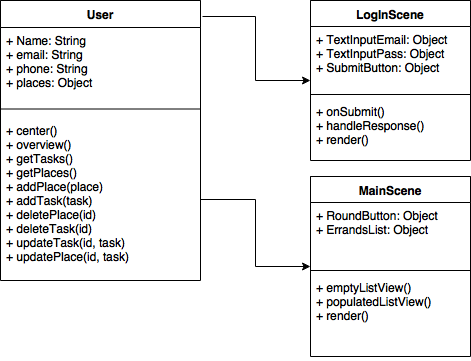
Use Case Diagram

[****](https://www.draw.io/#G16iTk2pezNMW1Mqd04RT8Vdwa1eyhSy10)

Sequence Diagram

[](https://www.draw.io/#G1ErlZFRjS5QwWbFQ7KTCciWGIzhaD0LyI)

Class Diagram

[****](https://www.draw.io/#G1ZjaYELaq4wQogBqoPuZAwQd8VjSe8a9N)

### 

### #669 - User Story Name: Add Places

* Description: As a registered user, I would like to be able to add a place so that I have an itinerary.

Acceptance Criteria

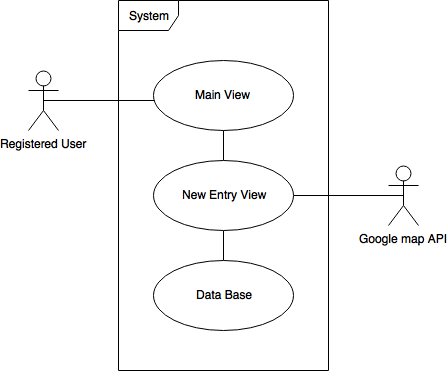
* Verify the User is able to search for a valid entity (e.g. CVS, Walmart).
* Verify the User is able to search for a valid address
* Verify the User is able to pick from the pool of favorite places.
* Verify the User selected one of the previous choices in order to add a new place to the itinerary.

Use Case

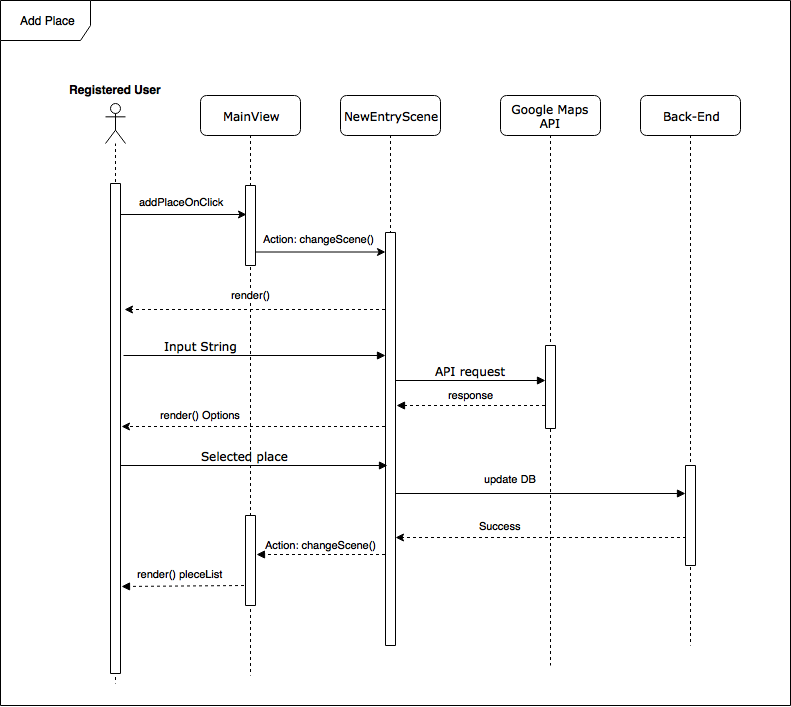
* Name: Add Places
* Actor: Registered User
* Preconditions:
  + User has to be authenticated
  + Visible scene: MainScene
* Description:

1. Add place button pressed.
2. Search place
3. Select place
4. Display all list of places

Use Case Diagram

[](https://www.draw.io/#G12j4998rb_dwNG2ocbi8ZiOYRygm_-twx)

Sequence Diagram

[****](https://www.draw.io/#G1_g7H1PZMPFAcw85oWUATyUPeQK_Y29Fq)

Class Diagram

[****](https://www.draw.io/#G1DT4i0XJVQscmImq_Ek3wh5WMqvw3Ybx2)

### 

### #670 - User Story Name: Add tasks to places

* Description: As a registered user, I would like to be able to add tasks to be done in a specific place of the itinerary so that I can have a list of task per place.

Acceptance Criteria

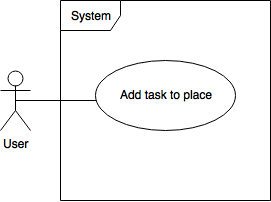
* Verify the User has a place in the itinerary in order to add a task.
* Verify the User added descriptive text as a task.

Use Case

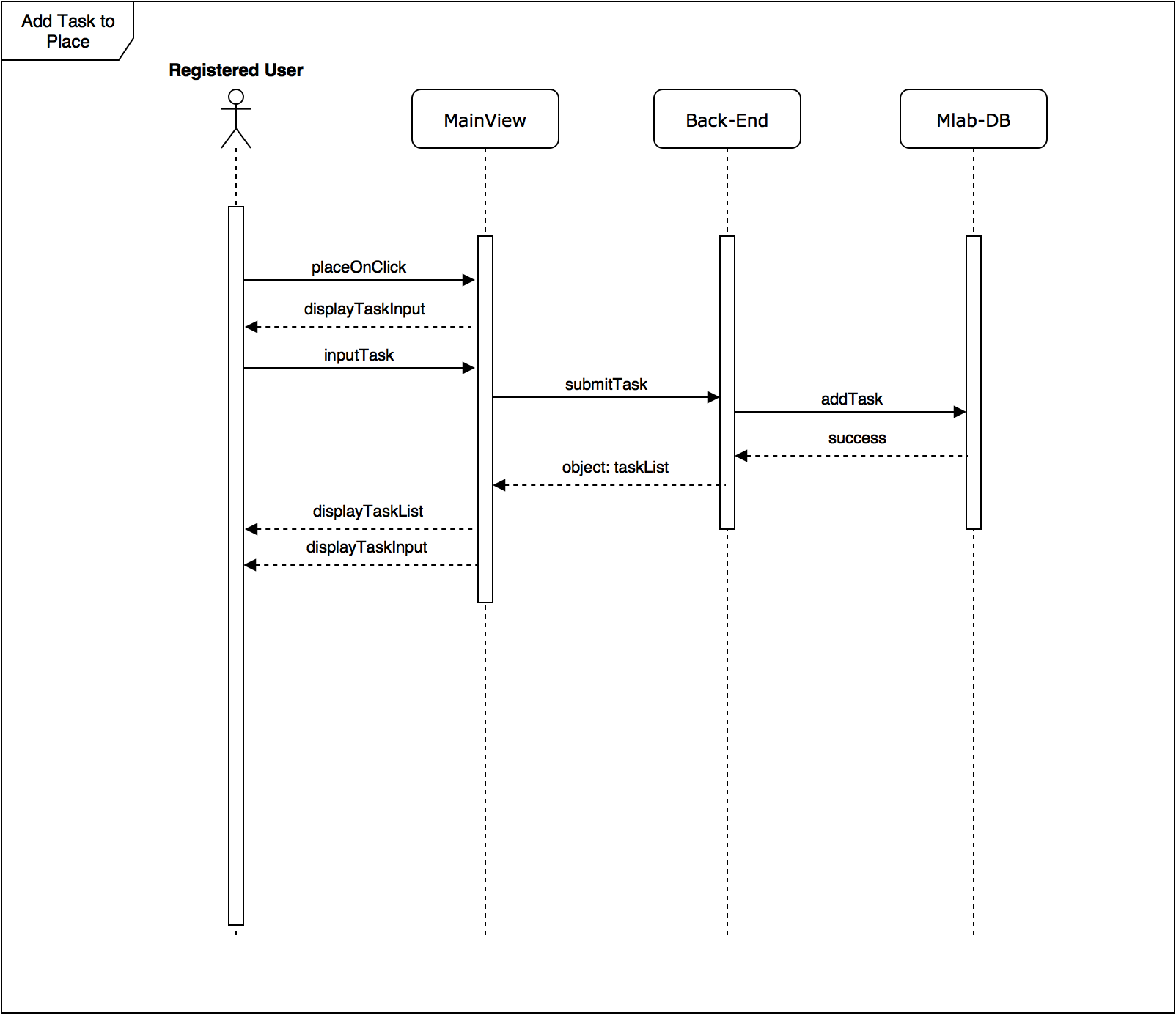
* Name: Add Task To Places
* Actor: Registered User
* Preconditions:
  + At least one place has to exist
* Description:

1. Click add task or new place has been just created.
2. Input name of task.
3. Save task.

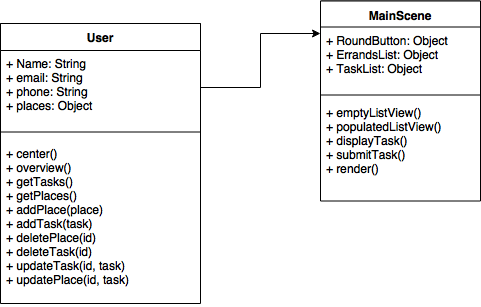
Use Case Diagram



Sequence Diagram

[****](https://www.draw.io/?scale=2#G13gbxcYIfxItb1yysYhxXYyUeIT7zApUG)

Class Diagram

[****](https://www.draw.io/#G1QosaGMEeGlS2AqMy5-AuBaDMfBXW1Kny)

### #671 - User Story Name: Show Itinerary and Total Time

* Description: As a registered user, I would like to be able to see an overview of all the places in the itinerary and how long it would take to go through all of them.

Acceptance Criteria

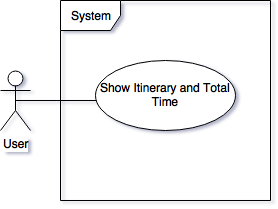
* Verify the User has at least one place in the itinerary.
* Verify the User is able to see an overview of the itinerary and the optimized order of the places.

Use Case

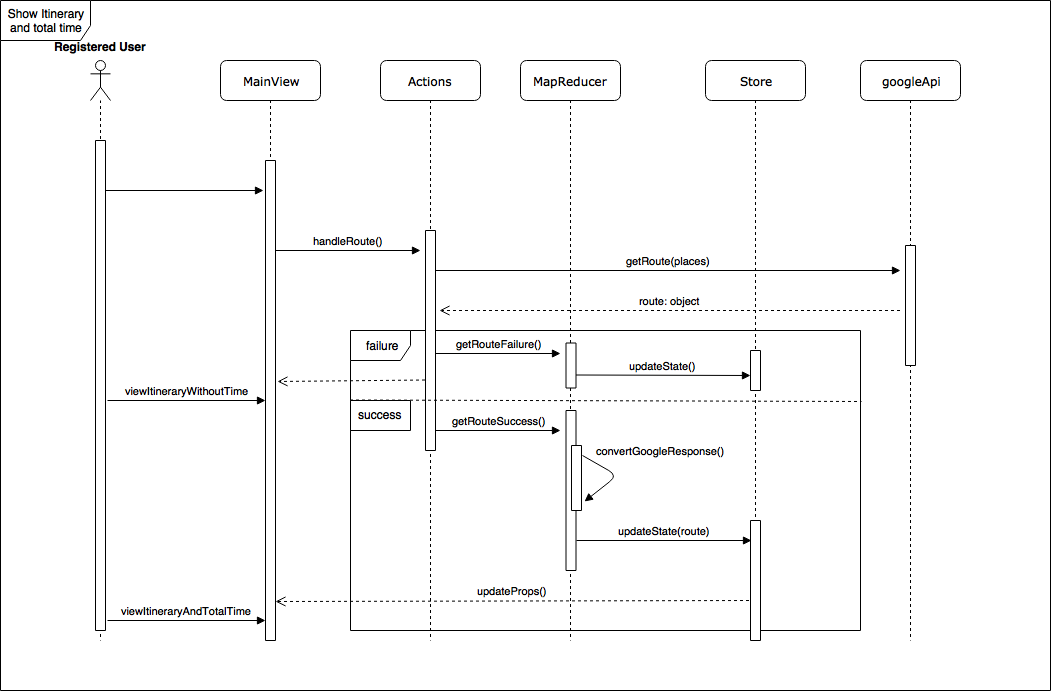
* Name: Show Itinerary and Total Time
* Actor: Registered User
* Preconditions:
  + At least one place added to the itinerary.
* Description:

1. The User clicks the add button.
2. The User adds a place by name or address.
3. The User may or may not add some tasks to do in the place.
4. The User confirms the place and tasks.
5. The User is taken to the main screen where all the added places and the total time are shown.

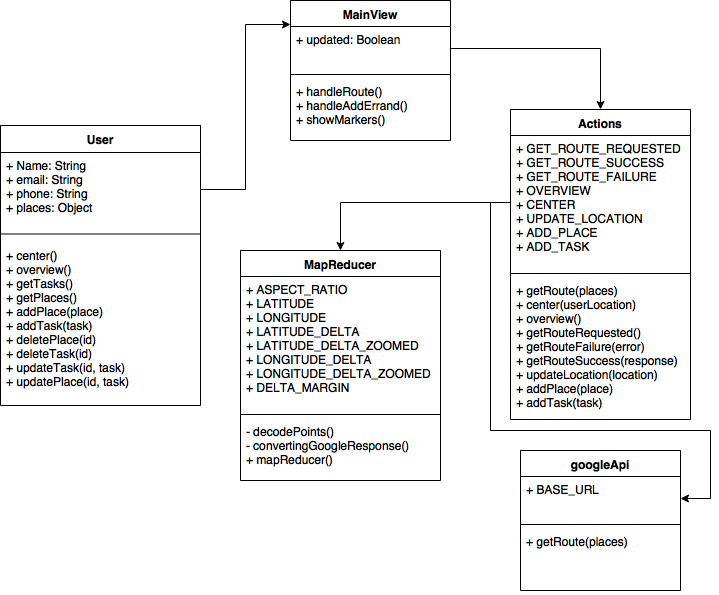
Use Case Diagram

[](https://www.draw.io/#G17qR4hswbXMCXslEBmGztrD8hbSnLaO4t)

Sequence Diagram

[****](https://www.draw.io/#G1s6EtzdjKOrn8EnegClYDHqMrVZ0JAynY)

Class Diagram

[****](https://www.draw.io/#G1Aajq_7ANgno1dYggZi2ONa1_OWk7rZX3)

### #673 - User Story Name: Setup Navigation Information

* Description: As a registered user, I would like to be able to have a navigation mode, so I follow step by step instructions to go through my itinerary.

Acceptance Criteria

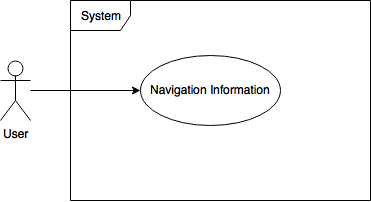
* Verify the User has at least one place in the itinerary.
* Verify the User is able to see the estimated time to the place he/she is heading to.
* Verify the User should be able to see the maneuver that he/she has to perform.

Use Case

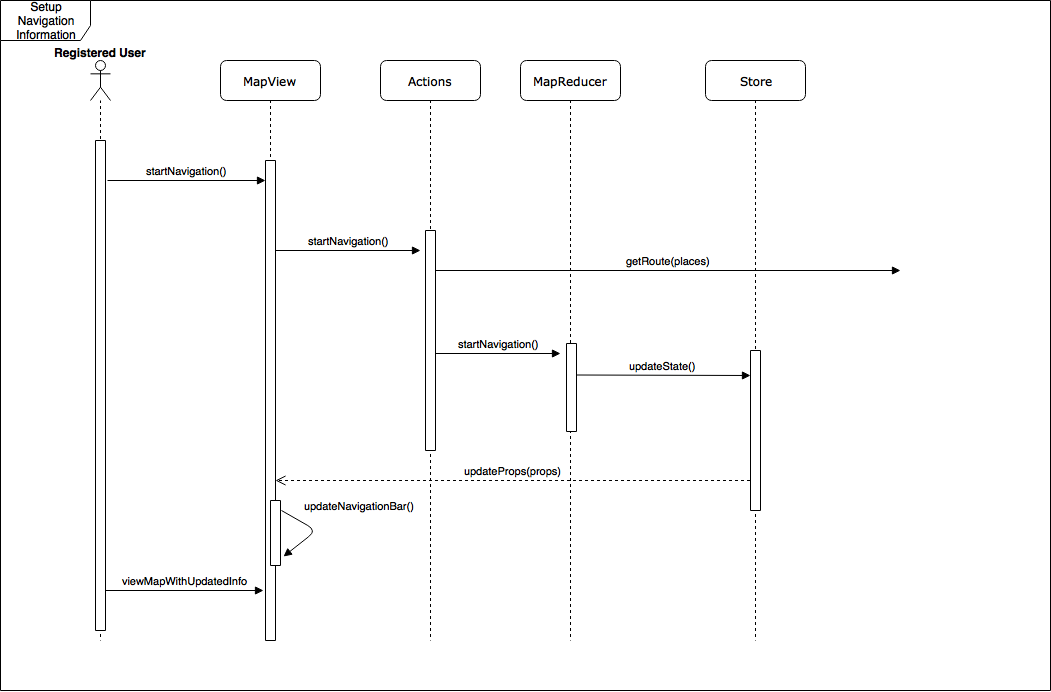
* Name: Navigation Information
* Actor: Registered User
* Preconditions:
  + At least one places added to the itinerary and the navigation button pressed
* Description:

1. The User adds at least a place to the itinerary.
2. The User presses the start navigation button
3. The User is presented with the next maneuver.
4. The User is presented with the place he/she is heading to.
5. The User is presented with the estimated time to the targeted place.

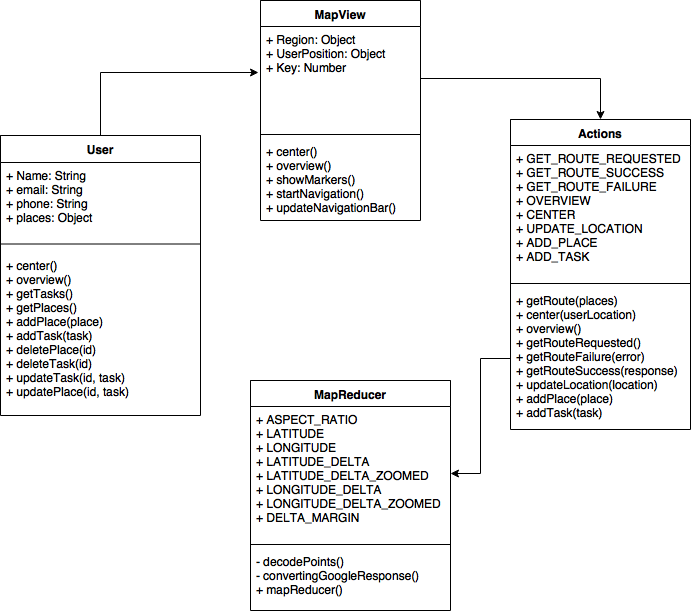
Use Case Diagram

[](https://www.draw.io/#G1p_GskKTEQlvJw-Dvo5OufQ3KeuHI9GQ8)

Sequence Diagram

[****](https://www.draw.io/#G17n5fidxMU3THd0hoQx3lql4gD4Hx44Ym)

Class Diagram

[****](https://www.draw.io/#G1BScoTOqfW9GYSTzvp3pLsO180H67KuQQ)

### #675 - User Story Name: Add Transition for Adding New Place While Navigating

* Description: As a registered user, I would like to be able to add a place to modify my itinerary once the navigation has started.

Acceptance Criteria

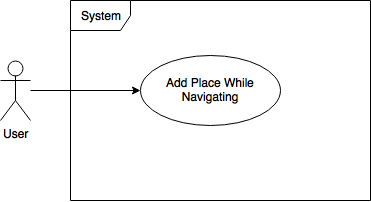
* Verify the User is able to have a gesture to add new place into the itinerary from the map view.
* Verify The new place is inserted in the itinerary.

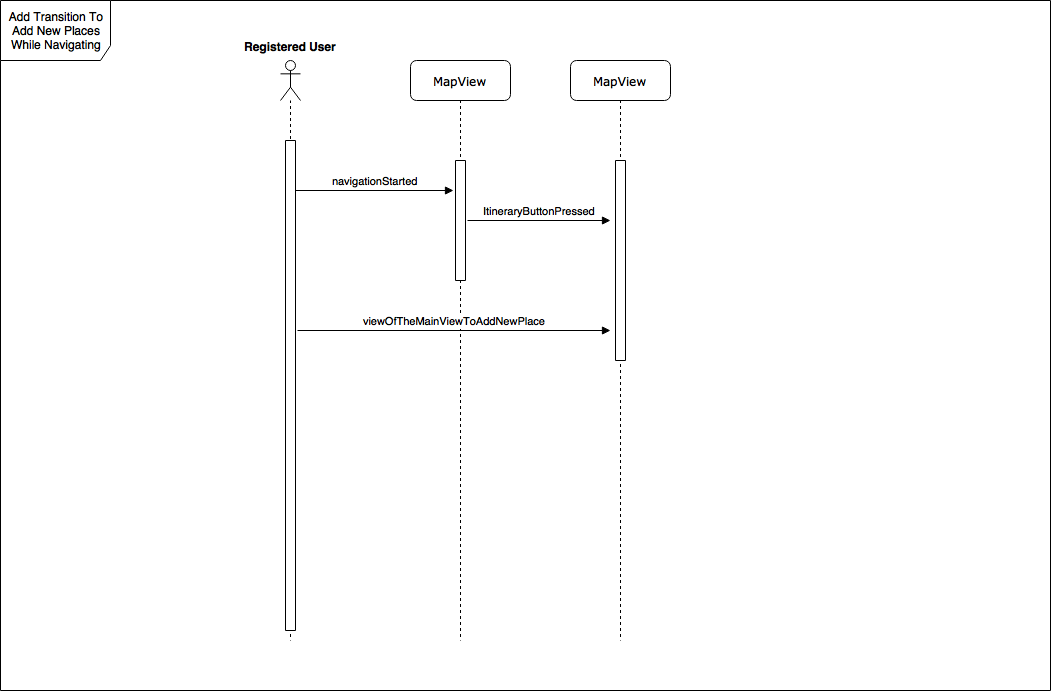
Use Case

* Name: Add Place While Navigating
* Actor: Registered User
* Preconditions:
  + At least one place added to the itinerary.
  + Navigation has started.
  + New place added.
* Description:

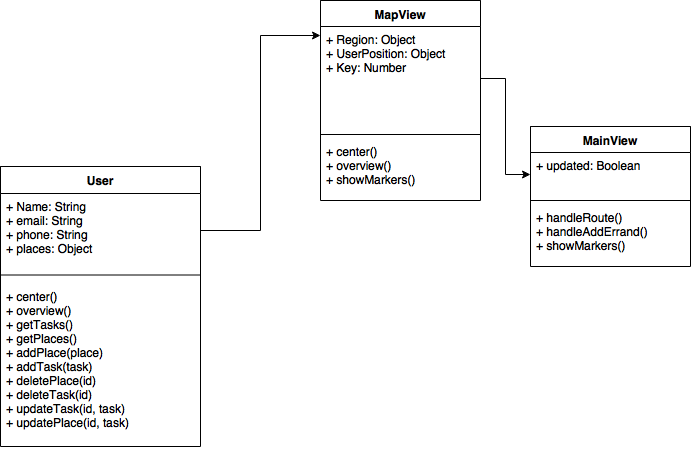
1. Navigation button pressed.
2. Gesture that takes the user to the Itinerary View.
3. Add a new place.
4. Optimize route with the new place added.

Use Case Diagram

[](https://www.draw.io/#G123fpUjI1G01psZkEHPdf-RLd0teSEwsz)Sequence Diagram



Class Diagram



### #676 - User Story Name: Update Route on Map View if Delay

* Description: As a registered user, I would like to be able to follow an optimized path at all times.

Acceptance Criteria

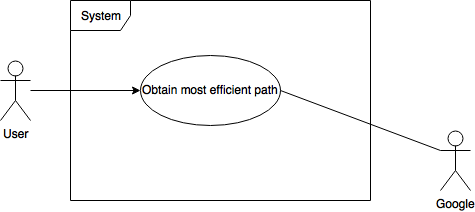
* Verify the User has at least one place in the itinerary.
* Verify the User is notified when changes to the ETA occur.
* Verify if/when changes occur, routes will be updated with the new optimized path.
* Verify when the optimized path is ready, User will be redirected to follow the new route.

Use Case

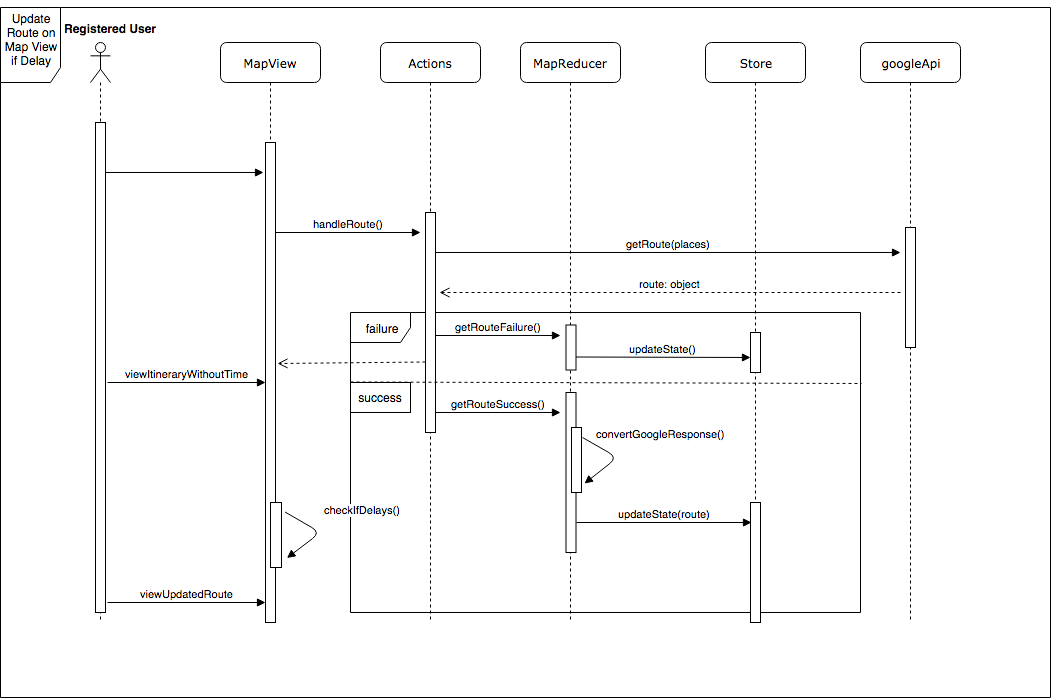
* Name: Obtain most efficient path
* Actor: Registered User
* Preconditions:
  + At least one place added to the itinerary.
  + Navigation Started
* Description:

1. Added places to the itinerary.
2. Start Navigation
3. Check if changes in the route to update the itinerary

Use Case Diagram

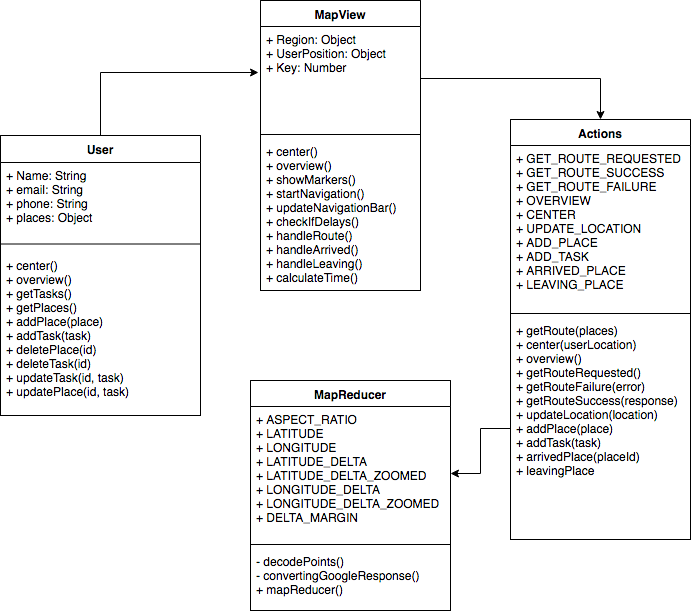
[](https://www.draw.io/#G1wncR35QbcMqEprmrvRVEKKHwpkF76KDN)

Sequence Diagram

[****](https://www.draw.io/#G1ZbcsLVCyxfFfKHNIOikVuoi_7QJlQ51L)

### 

Class Diagram

[****](https://www.draw.io/#G1_6-4d4y3yn_UQN3qO1FOxYN88TKZTd0R)

### #679 - User Story Name: Track The Time Spent in Places

* Description: As a registered user, I would like to be able to see the time I spent in places I have visited before.

Acceptance Criteria

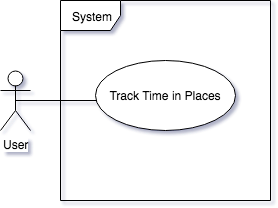
* Verify the User must have visited at least one place to have a record of time.
* Verify every time a user visits a place in the itinerary, the time spent will be recorded.
* Verify the time of a visit will be in the History View included in a visit record for the User to review the time spent in a specific visit.
* Verify the time of a visit will be used for future itinerary estimates.

Use Case

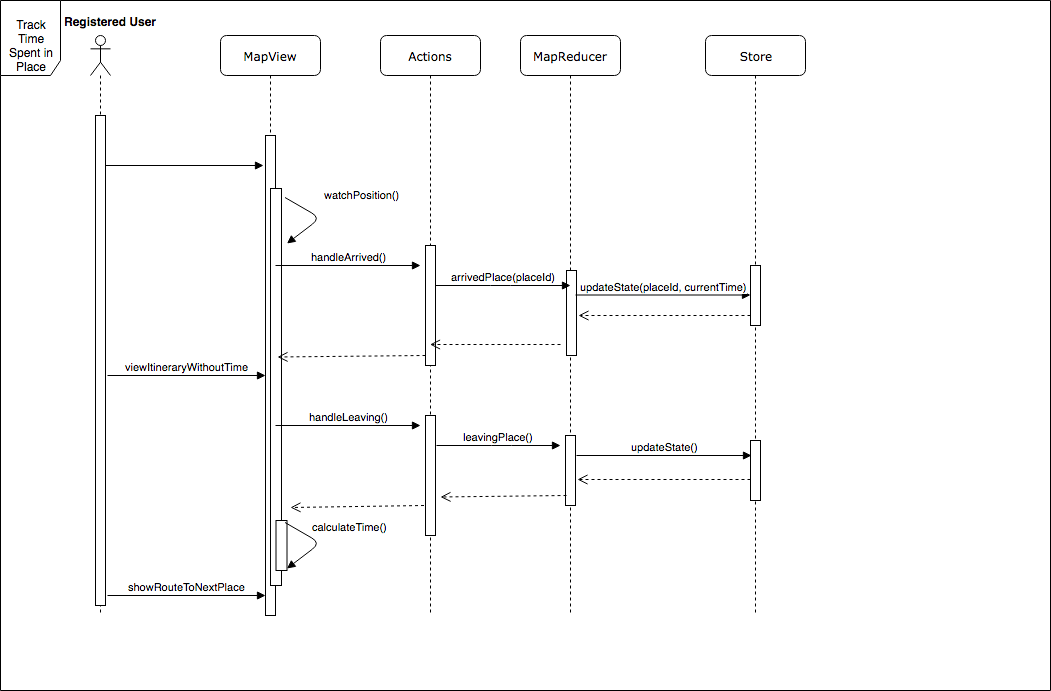
* Name: Track Time in Places
* Actor: Registered User
* Preconditions:
  + At least one place added to the itinerary.
  + Navigation Started
* Description:

1. Added places to the itinerary.
2. Navigation started
3. Be close enough to the most immediate destination

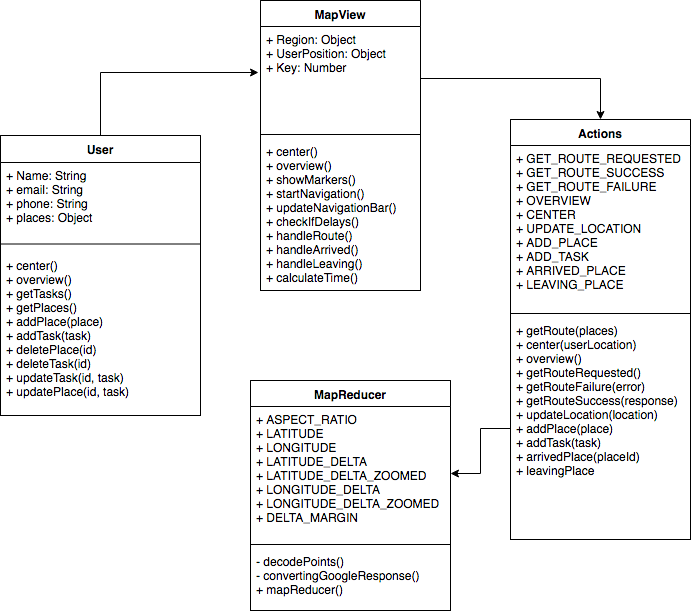
Use Case Diagram

[](https://www.draw.io/" \l "G1yTBTzPlOuzIYrloQzKp1gV-w9srTpFwJ)

Sequence Diagram

[](https://www.draw.io/" \l "G1x8hkh72pkYcRTmwYSFV5zwPiJwGeT1ZG)

Class Diagram

[****](https://www.draw.io/#G1_6-4d4y3yn_UQN3qO1FOxYN88TKZTd0R)

### 

### #680 - User Story Name: Register User [Backend]

* Description: As a developer, I would like to be able to add users to the database, so that users can be registered.

Acceptance Criteria

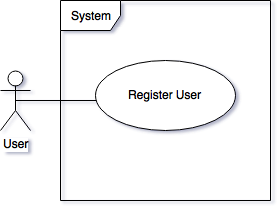
* Verify the User’s information is correctly filled.
* Verify the User’s email is unique.
* Verify the User was added to the Database

Use Case

* Name: Register User [Backend]
* Actor: User
* Preconditions:
  + User is not Registered
* Description:

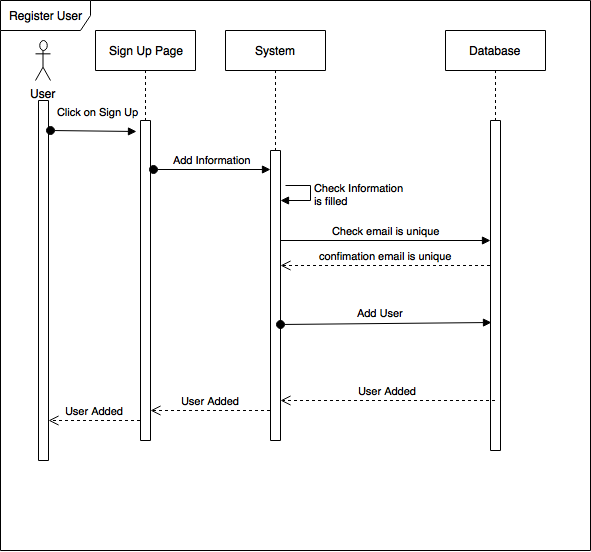
1. The System will verify the user data is filled.
2. The system will check the User’s email is unique.
3. The system will add the user to the database.

Use Case Diagram

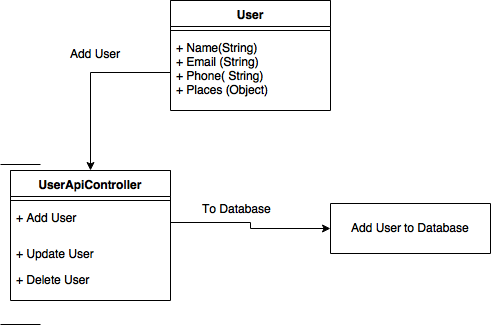
[](https://www.draw.io/#G1w_NJwyEDbY2vOOVr5aLVedeh8lFzmU_q)

Registering user: To-do List Optimizer allows users to register using the mobile app.

Sequence Diagram

[](https://www.draw.io/#G1LsF-a433NMOuDXI_s5d9K_w_9_QqXWFn)

Class Diagram

[](https://www.draw.io/#G1bwhPAyMTBE2li3uBzBoWDNqbsp8MSQ6J)

## 

### #681 - User Story Name: Add Places to User [Backend]

* Description: As a developer, I would like to be able to add Places to a User’s Itinerary and reflect this change in the database, so that I can add Places to a certain User.

Acceptance Criteria

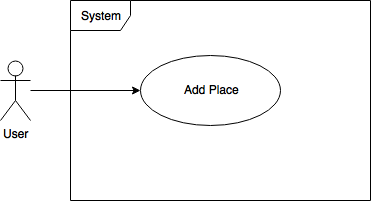
* Verify the Place is added to the correct User.
* Verify the Place is added to the User’s Itinerary.
* Verify the Place’s address is correct.

Use Case

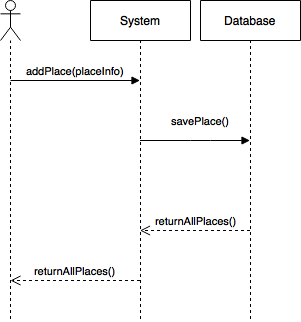
* Name: Add Place to User
* Actor: Registered User
* Preconditions:
  + User must be registered.
* Description:

1. User will add a place by entering the name and choosing the address.
2. Place will be added to the database.

Use Case Diagram

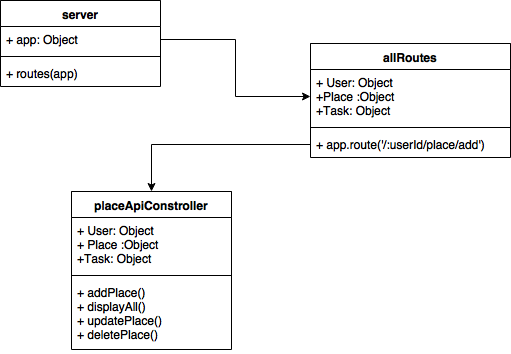
[](https://www.draw.io/#G1NFKmu8fMA0PMM_H0QNHjAAQWucBiHQgn)

Sequence Diagram

[](https://www.draw.io/#G1iqN0Gh_igm4UhFdtG7hVpz8FapZ_Jp2Y)

### 

Class Diagram

[](https://www.draw.io/#G1ejV4hoNXkfQ2_JDeldmhMLhWbD3ZKZTS)

### #683 - User Story Name: Add Task to Places [Backend]

* Description: As a developer, I would like to be able to add one or more tasks to a Place, so that I can assign certain task to a certain Place.

Acceptance Criteria

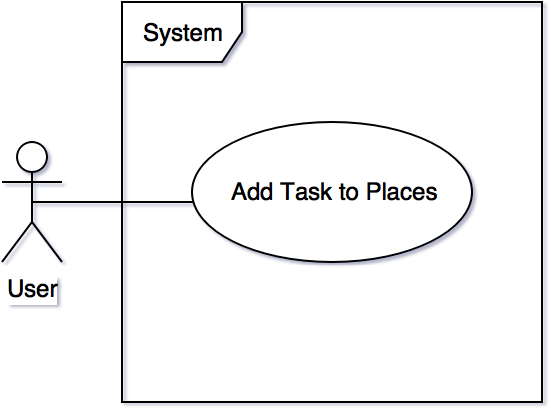
* Verify the task is added to the correct Place
* Verify the Task are added to the database.

Use Case

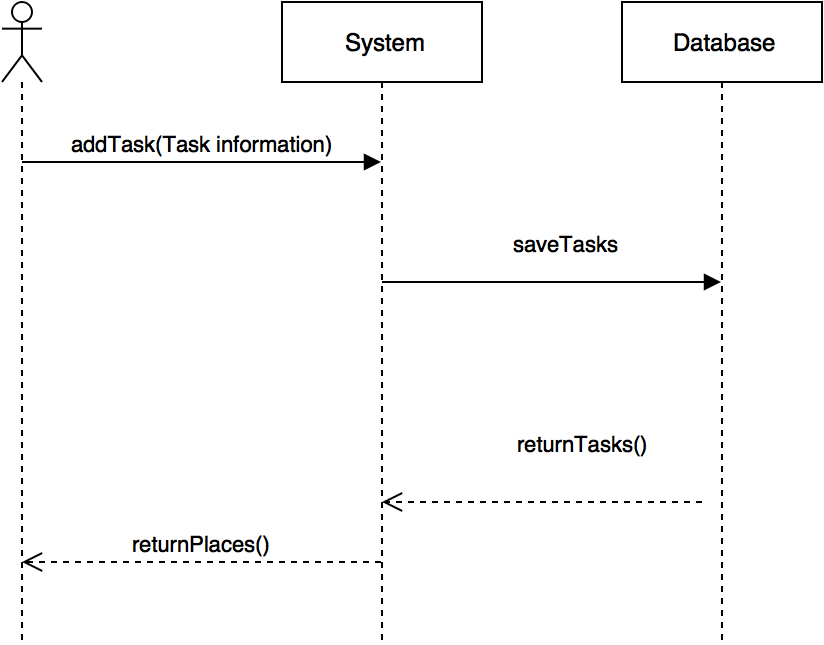
* Name: Add Task to Places
* Actor: Registered User
* Preconditions:
  + User must have a Place to which to add a Task
* Description:

1. User adds a Task to a given Place
2. Task is added to the specific Place and stored in database.

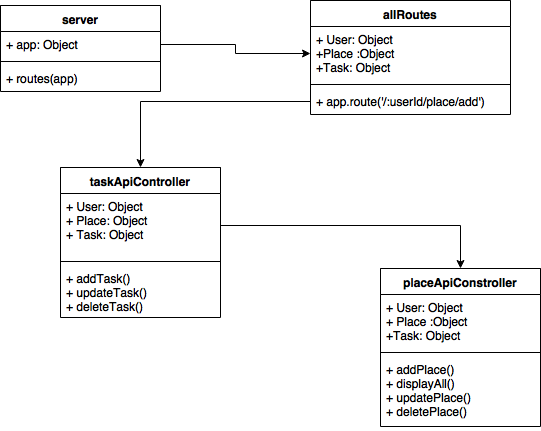
Use Case Diagram

[](https://www.draw.io/?scale=2" \l "G1G3a9oc4pP5SpJ6JoCLgqPYOm8boCGyMA)

Sequence Diagram

[](https://www.draw.io/?scale=2#G1p6S2ElNBuvQEi8cPub9oGQvSy4SJjmBi)

Class Diagram

[](https://www.draw.io/#G1WDnxkGl2TANiK9EvLlLagbVEoGlMd1eO)

### #684 - User Story Name: Delete Places When Tasks are Completed

* Description: As a developer, I would like to delete a Place corresponding to a User, so that once the User completed all the tasks in the Place, the Place is deleted.

Acceptance Criteria

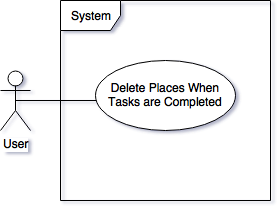
* Verify all the Tasks in a Place are completed.
* Verify the Place is deleted from the User’s Account
* Verify the Place is deleted from the Database.

Use Case

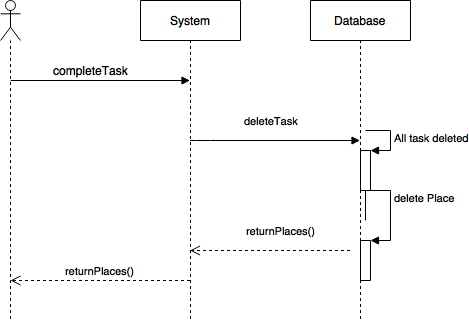
* Name: Delete Places When Tasks are Completed
* Actor: Registered Users
* Preconditions:
  + User must have places added
* Description:

1. User Completes all the tasks in the Place.
2. The Place is deleted from the database.

Use Case Diagram

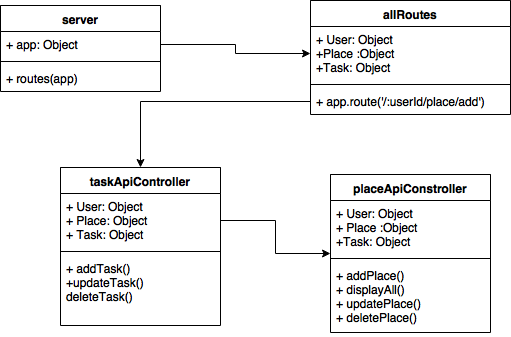
[](https://www.draw.io/" \l "G1zCKeRx-cxwymuS23il0WtrZ5XBa0tleb)

Sequence Diagram

[](https://www.draw.io/" \l "G1hrwiSWAVEPMlfVRauajDOxY7HDMz_ikC)

### 

Class Diagram

[](https://www.draw.io/#G1_gOyi8VzGAVwJhRb3pFZHCrUmZvsrPLb)

### 

### #685 - User Story Name: Delete Places Manually

* Description: As a developer, I would like to delete a Place manually, so that I can delete a Place without completing all the tasks

Acceptance Criteria

* Verify all the tasks corresponding to the Place are deleted from the database.
* Verify the Place is deleted from the database.

Use Case

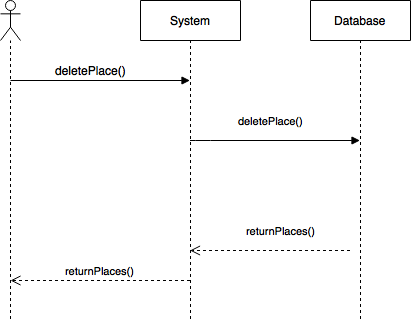
* Name: Delete Places Manually
* Actor: Registered User
* Preconditions:
  + User must have a Place added to their list.
* Description:

1. User deletes the Place from their list
2. The Place is deleted from the database, and all the Tasks associated with the place are also deleted.

Use Case Diagram

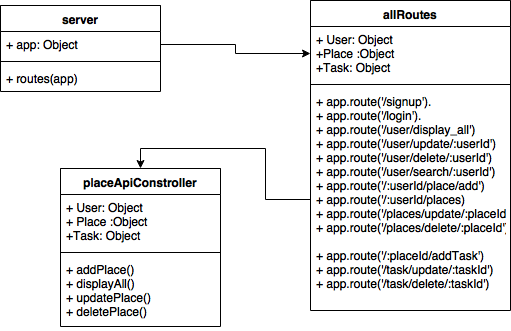
[](https://www.draw.io/" \l "G1SlxZaPdPgI8mKiyohJL1to9jwSUF4nSK)

Sequence Diagram

[](https://www.draw.io/" \l "G1mYKurHxw3lYVTWafbUEOfJy2A9ZCseg7)

### 

Class Diagram

[](https://www.draw.io/" \l "G1ebEVd37Gg0tyFgtmNjOuE-yrf5ZxBdz0)

### #687 - User Story Name: Update Places [Backend]

* Description: As a developer, I would like to be able to update a Place (change address or Name) corresponding to a User, so that then I can update a Place without deleting the Place.

Acceptance Criteria

* Verify the Place information is updated in the Database.

Use Case

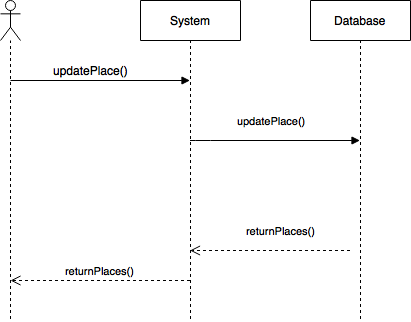
* Name: Update Places
* Actor: Registered Users
* Preconditions:
  + User must have a Place added
* Description:

1. User updates information of a Place
2. The Place is updated in the Database.

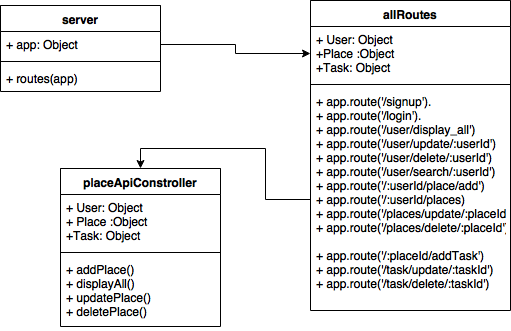
Use Case Diagram

[](https://www.draw.io/" \l "G1rE3DLxBmOgkD27pqFVu6gmvKVFllWpSk)

Sequence Diagram

[](https://www.draw.io/#G1IY-Glfu1wPByB0MyE7Krtqcanu96UDnj)

Class Diagram

[](https://www.draw.io/#G1peyUS0gBKvBHv38wO0x2STXmItTwrCjd)

### #688 - User Story Name: Delete Task Manually [Backend]

* Description: As a developer, I would like to be able to delete tasks from a Place, so that the tasks are deleted before completing them.

Acceptance Criteria

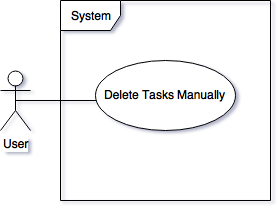
* Verify the Task is deleted from the Place

Use Case

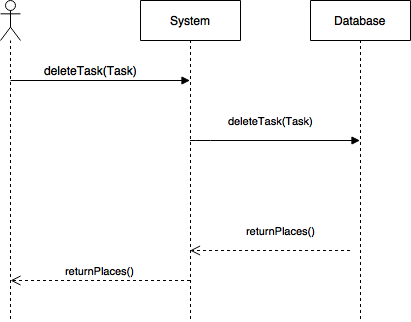
* Name: Delete Task Manually
* Actor: Registered User
* Preconditions:
  + User must have a Task added to a Place
* Description:

1. User selects Task it wishes to delete
2. Task is deleted from the Place without been completed.

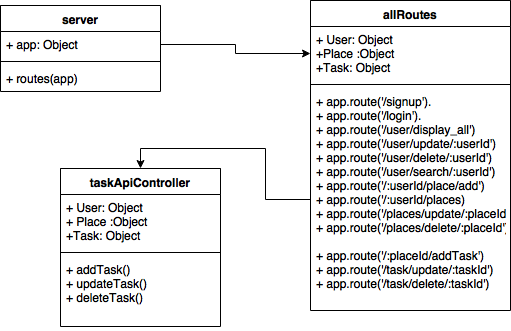
Use Case Diagram

[](https://www.draw.io/" \l "G1lVPPXlgZy5Vz7LBHhxfTjv54IJGBoJBc)

Sequence Diagram

[](https://www.draw.io/#G1C1EgRsvb0GTiMQjn9sMAkVTz1UEjOdVE)

Class Diagram

[](https://www.draw.io/#G1hKnRoDpa1TJer3jAeGDBE0qNjgAYL6fd)

### #690 - User Story Name: Create Scenes Basic Transitions

* Description: As a Developer I want to be able to transition between the scenes so that we have the layout of the whole project.

Acceptance Criteria

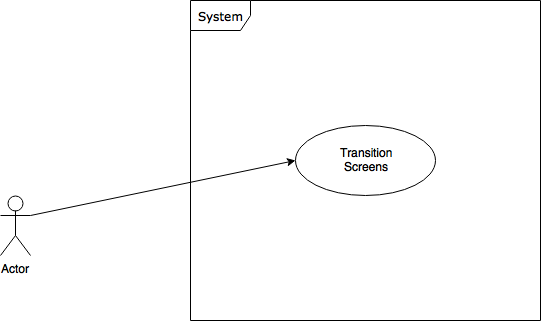
* Verify that all the scenes transition as desired in the design.

Use Case

* Name: Create Scene Basic Transition
* Actor: Registered User
* Preconditions:
  + React-native-route-flux installed and added to the project
* Description:

1. The system transitions properly between screens.
2. The proper order of scenes happens when the indicated buttons are pressed.

Use Case Diagram



## 

### #695 - User Story Name: Update User [Backend]

* Description: As a developer, I would like to update User’s information so that the user could change their name, email, and phone number.

Acceptance Criteria

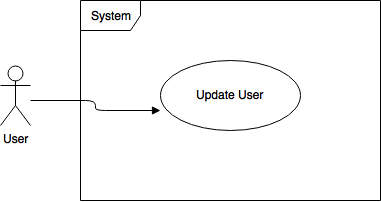
* Verify the User’s information is correctly filled.
* Verify the User’s information has been updated in the database.

Use Case

* Name: Update User
* Actor: Registered User
* Preconditions:
  + User must be Registered.
* Description:

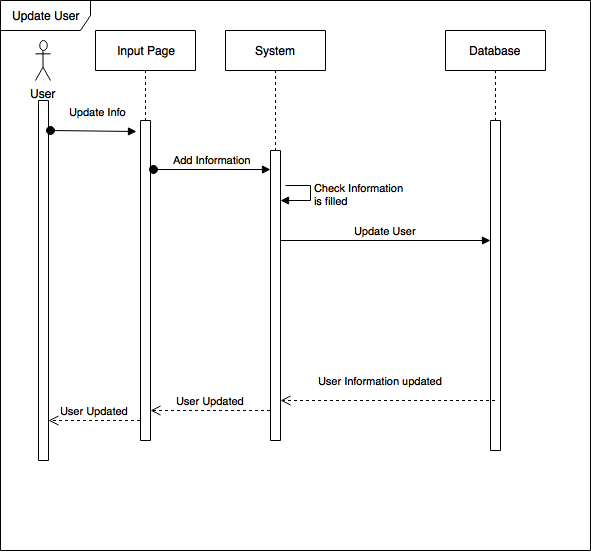
1. User changes the personal information.
2. User’s information is updated in the database.

Use Case Diagram

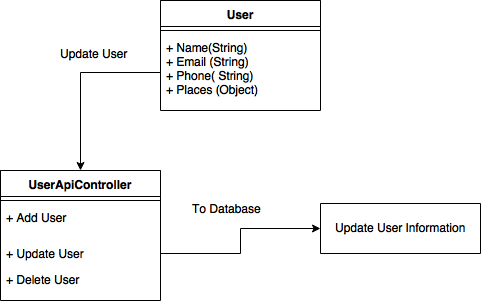
[](https://www.draw.io/#G18ydgocI36M8FX5HJn6IjJmnjy2mXApll)

Updating user: To-do List Optimizer allows registered users to update their information using the mobile app.

Sequence Diagram

[](https://www.draw.io/" \l "G1ZkP08lvlTVhrs8j3IS78N6oMwyVbDITr)

Class Diagram

[](https://www.draw.io/#G1eEEqdfKGsGYiMXu3LAWmzI63N1vbub8y)

### #697 - User Story Name**: Delete User [Backend]**

* Description: As a developer, I would like to be able to delete a User’s account, so that the User’s don’t appear in the database anymore.

Acceptance Criteria

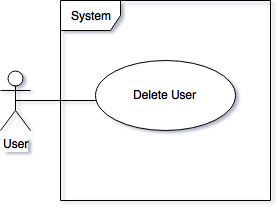
* Verify the User is no longer in the Database
* Verify all Places designated to the User are also deleted.

Use Case

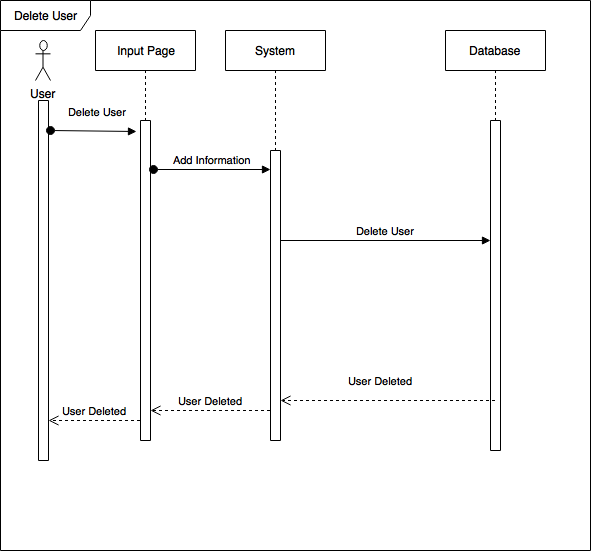
* Name: Delete User
* Actor: Registered User
* Preconditions:
  + User Must be Registered and Logged in.
* Description:

1. User would choose to delete the account.
2. The account would be deleted from the database.

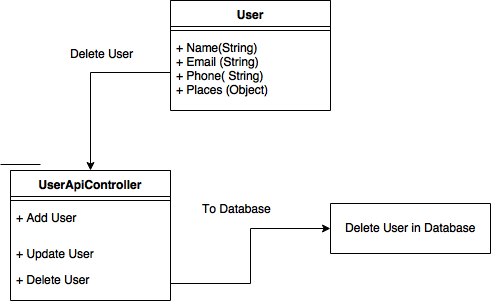
Use Case Diagram

[](https://www.draw.io/#G1Ex0yw77MI-4eeomUQHYK18_c9tyS3E2k)

Sequence Diagram

[](https://www.draw.io/" \l "G1tXdeFH7-fLFdFl9WiTd4w0BU4kEe19MO)

Class Diagram

[](https://www.draw.io/#G1JiT5-fXYdhPbMG2Fe79JQQ-3-FqBXdAc)

### #703 - User Story Name: Center Map

* Description: As a User I would like to have a closer look to my position so that I could see myself in the middle of the map and locate myself better.

Acceptance Criteria

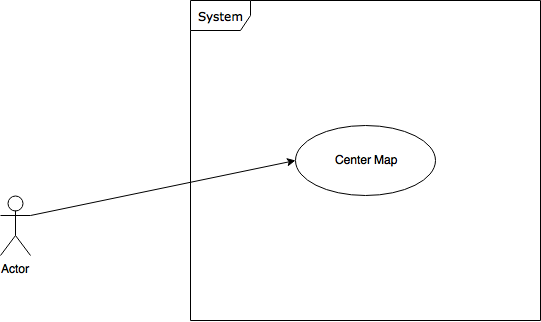
* Verify that the User is not center in order to be able to center.
* Verify that the Map changes to reflect the user position in the middle.

Use Case

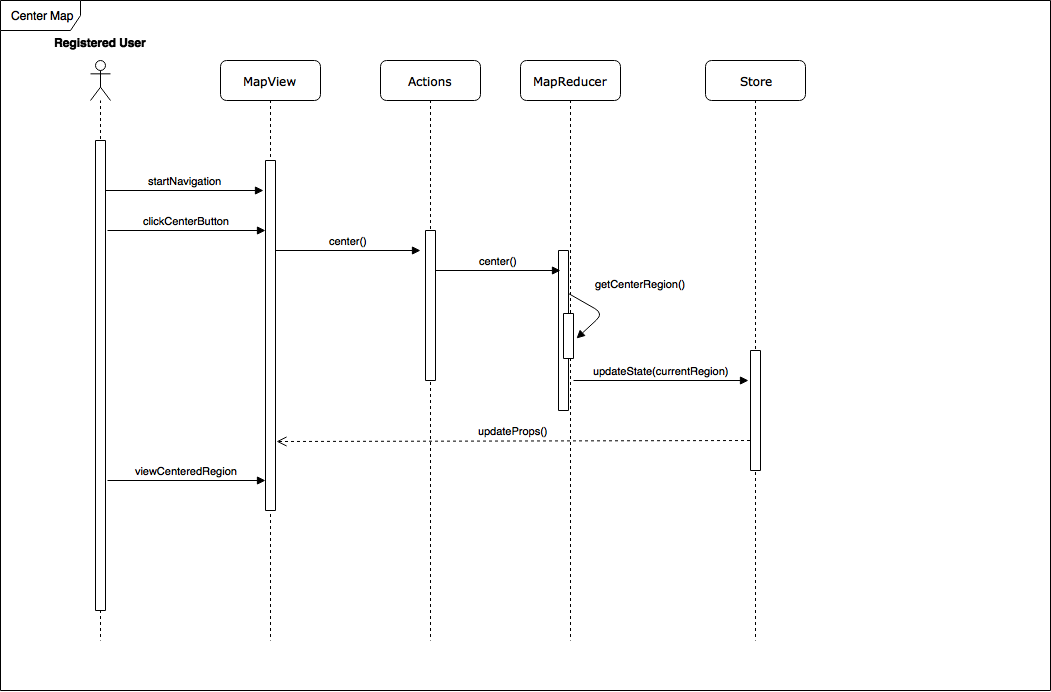
* Name: Center Map
* Actor: Registered User
* Preconditions:
  + The user should have added places to the itinerary.
  + The user should have started navigation.
* Description:

1. Use case starts when the user clicks the **Center** button.
2. The system calculates the region based on the User’s current position.
3. The System changes the map current region to the previously calculated.

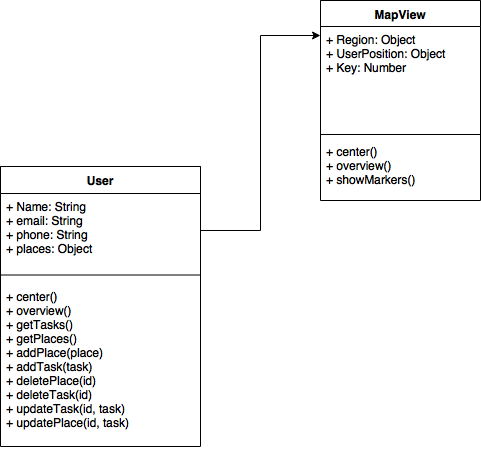
Use Case Diagram



Sequence Diagram



Class Diagram

****

### #704 - User Story Name: Overview Map

* Description: As a Developer I want to be able to transition between the scenes so that we have the layout of the whole project.

Acceptance Criteria

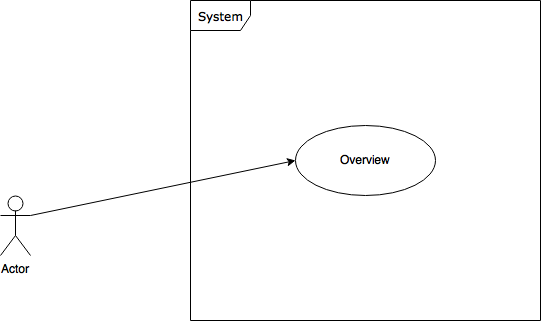
* Verify that all the scenes transition as desired in the design.

Use Case

* Name: Create Scene Basic Transition
* Actor: Registered User
* Preconditions:
  + React-native-route-flux installed and added to the project
* Description:

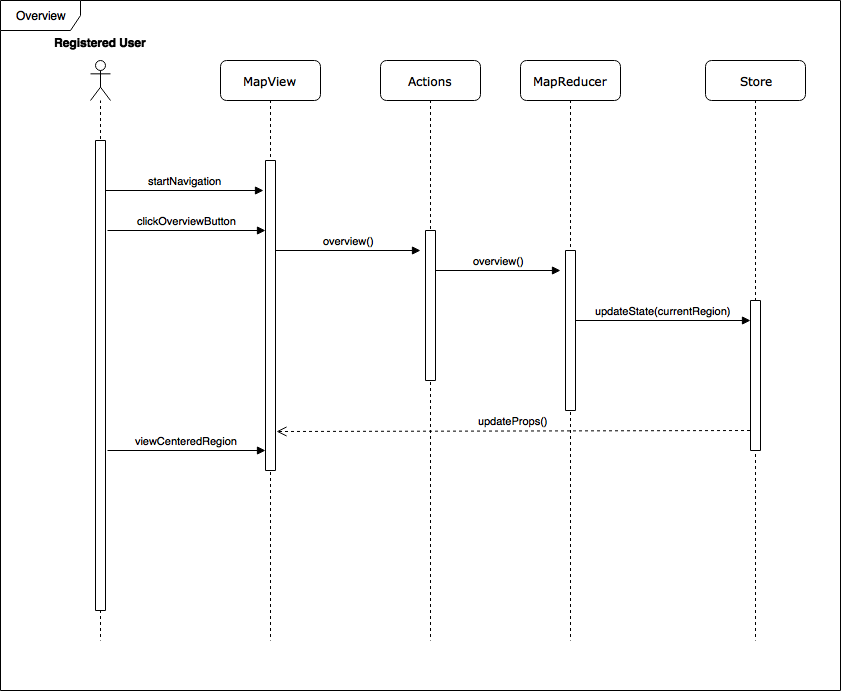
1. The system transitions properly between screens.
2. The proper order of scenes happens when the indicated buttons are pressed.

Use Case Diagram

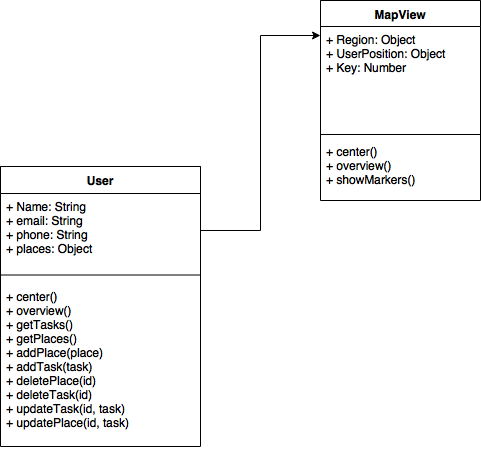


## 

Sequence Diagram



Class Diagram

#705 - User Story Name: Display Markers of Places in the Map

* Description: As a user I would like to see the places clearly defined the map so that I can locate them faster.

Acceptance Criteria

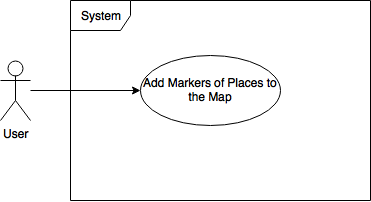
* Verify that all the places listed in the trajectory are display in the map
* Verify that the places are visible at all times.
* Verify that the markers identify the places by position in the route.

Use Case

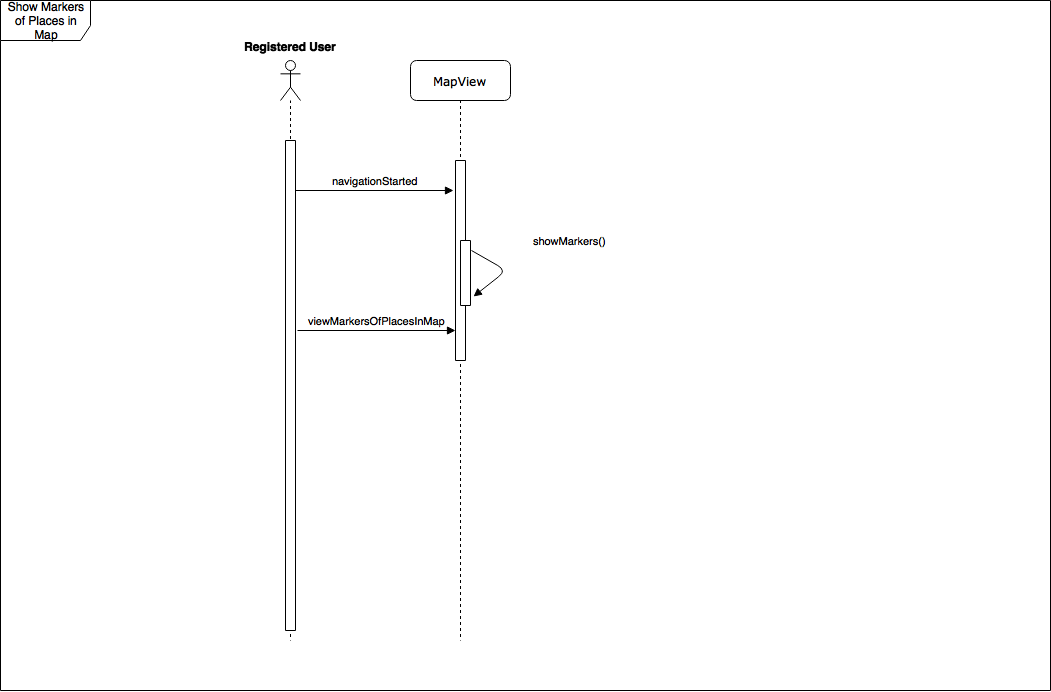
* Name: Show Markers in the Map
* Actor: Registered User
* Preconditions:
  + At least one place added to the itinerary.
  + Navigation has started.
* Description:

1. Added places to the itinerary.
2. Navigation button pressed.
3. View of the map shown with the markers in it.

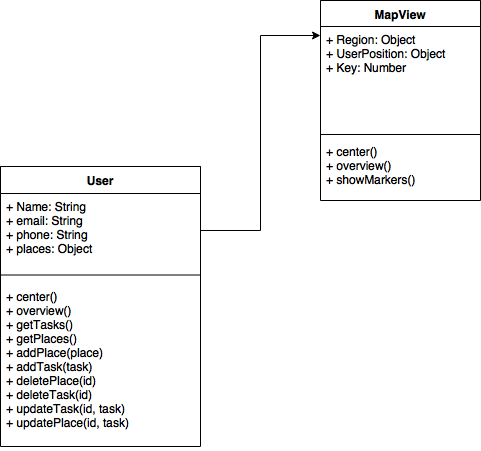
Use Case Diagram

[](https://www.draw.io/" \l "G1NFKmu8fMA0PMM_H0QNHjAAQWucBiHQgn)

Sequence Diagram

[****](https://www.draw.io/#G1XlZsZUG7eJqnew0_dAdiMyHCbYDnHG6L)

Class Diagram

[****](https://www.draw.io/#G1TBbFMoZiZsPOP7HIC6MVc4QJOYcVNEtI)

### #706 - User Story Name: Connect to google maps API to autocomplete place search

* Description: As a developer I want to be able to request nearby places from google maps API

Acceptance Criteria

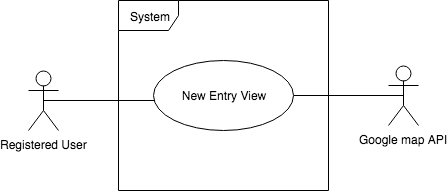
* Verify that response from API is in a Json format so it can be parsed easily.
* Verify that response is valid, so a place could be selected.

Use Case

* Name: Connect to google maps API
* Actor: Registered developer
* Preconditions:
  + Collected input from user and API KEY for google maps
* Description

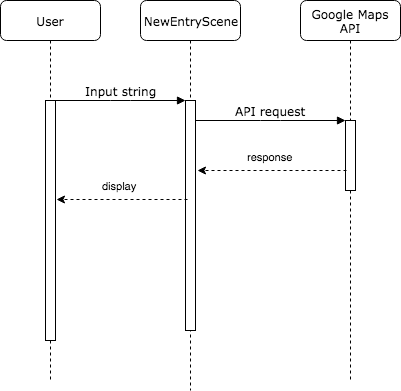
1. Create request URL + INPUT + API KEY
2. Parse response to use as Json object
3. Create list of places to be displayed for autocomplete

Use Case Diagram

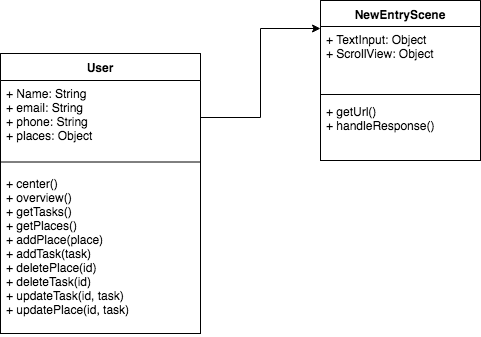
****

## 

Sequence Diagram



Class Diagram

#707 - User Story Name: Create New Entry Scene

* Description: As a Developer I want to be able to collect the input of a new place from the user so that I can update the DB with new place.

Acceptance Criteria

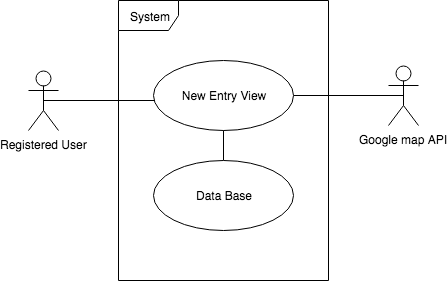
* Verify that there is a scene is created so that input components can be placed.
* Verify that TextInput is created so information can be collected

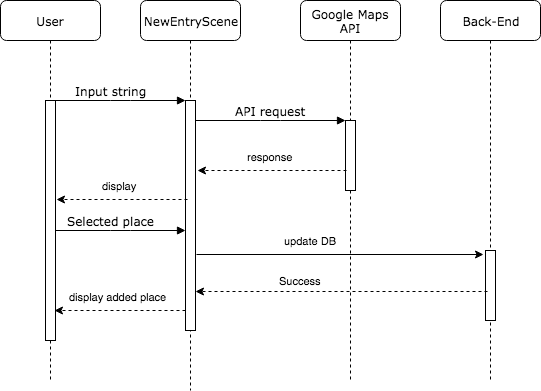
Use Case

* Name: Create New Entry Scene
* Actor: Registered User
* Preconditions:
  + User is registered
  + Visible scene: New Entry Scene
* Description

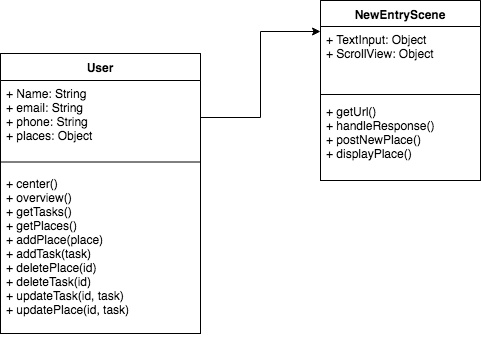
1. Create request URL + INPUT + API KEY
2. Parse response to use as Json object
3. Create list of places to be displayed for autocomplete

Use Case Diagram

****Sequence Diagram



Class Diagram



### #713 - User Story Name: Front End Restful API

* Description: As a developer, I would like to access the Back-end API so that I can access the database through the front-end application.

Acceptance Criteria

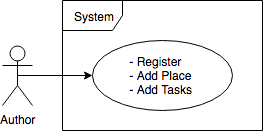
* Verify the Functions push to database
* Verify the Database returns right information.

Use Case

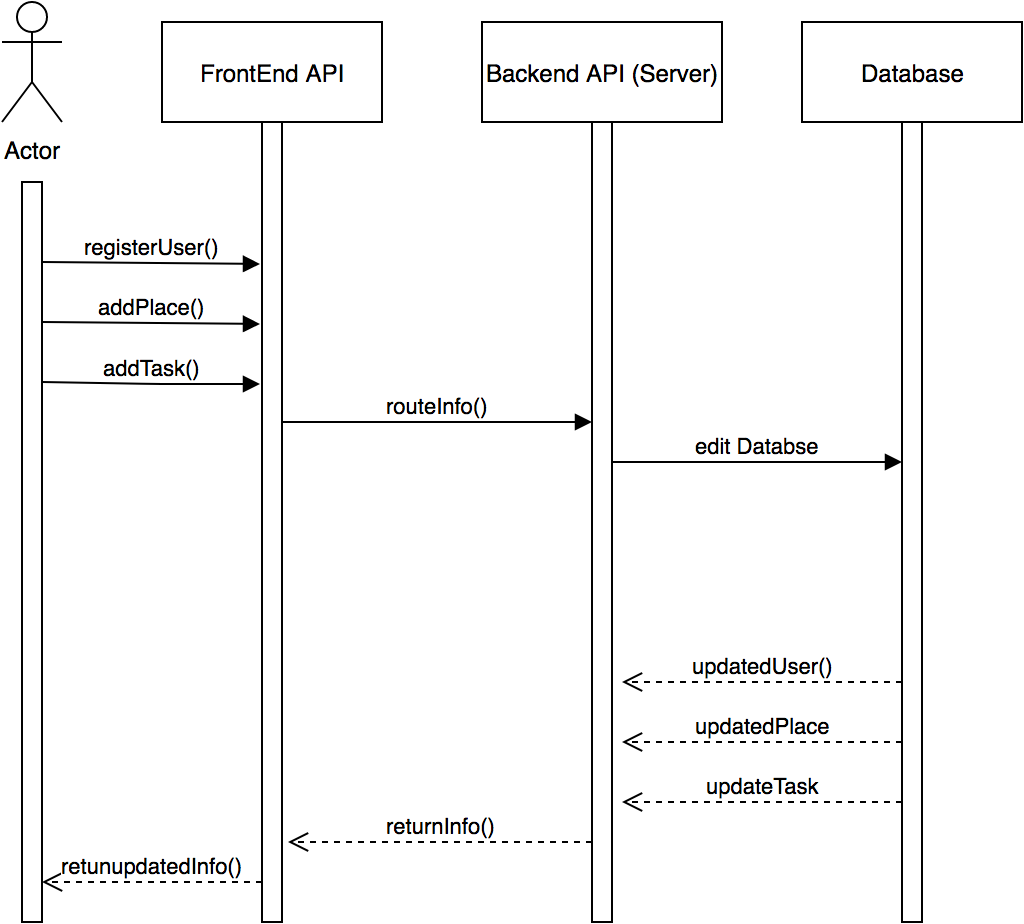
* Name: Front End Restful API
* Actor: registered user, guest
* Preconditions:
  + None
* Description:

1. Users register, or adds a place, or a task, the front end API will connect to the backend API which then connects to the database, and then return a detailed query.

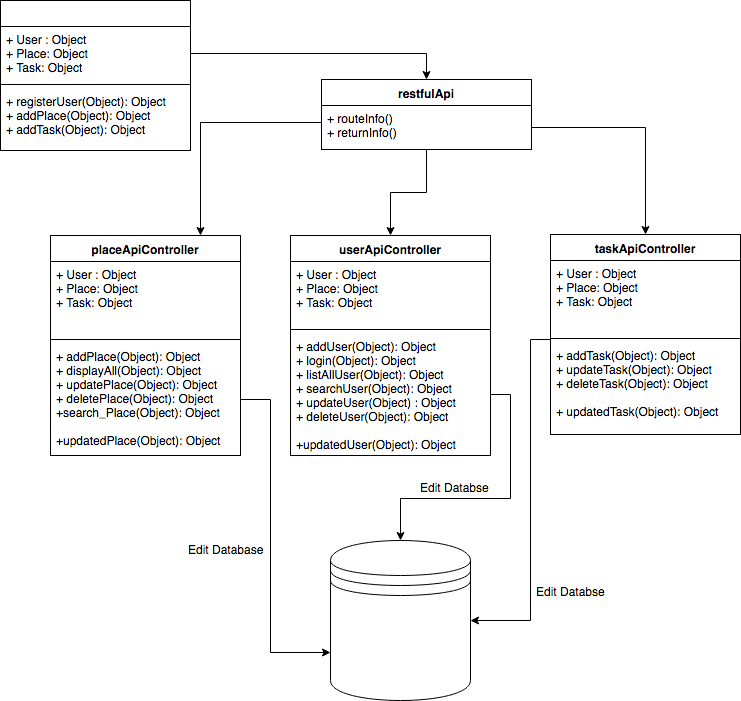
Use Case Diagram

[](https://www.draw.io/" \l "G1-qwB8YheWty-jBjPC0FkuwOGO7J7GcsO)

Sequence Diagram

[](https://www.draw.io/?scale=2" \l "G1xlDZX5vVxW-GRKFcr_jVZBzdIRcB2wr3)

Class Diagram

[](https://www.draw.io/" \l "G1Ys_xcZGLi5C-85NN7nfBpVXL_LIx-mxt)

### #714 - User Story Name: Move Backend to Production Server

* Description: As a developer, I would like to switch servers so that U can user the production server instead of the testing server.

Acceptance Criteria

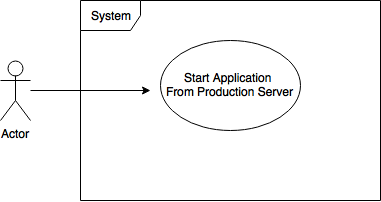
* Verify the application is running on the production server.
* Verify all the functions and endpoints work on the production server
* Verify all sensitive data is not in the code to avoid security risk.

Use Case

* Name: Move Backend to Production Server
* Actor: Developer
* Preconditions:
  + None
* Description:

1. The Application will run in the new server instead of the production testing server.

Use Case Diagram

[](https://www.draw.io/#G1wEl8Qy63QZL-aySNCNAFQb-dyNnIw7F0)

### #715 - **User Story Name: Asynchronous Request of Possible routes**

* Description: As a User I would like to obtain the most optimal way to traverse all the places I have to visit such that I will visit all the places in the less time possible.

Acceptance Criteria

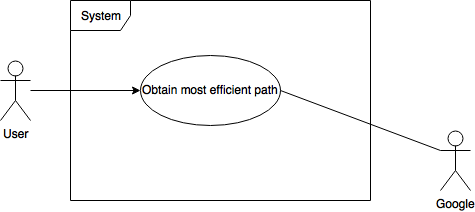
* Verify that the final destination is independent of the number of places.
* Verify that each place is tested for the final place candidate in order to find the best route out of all possibilities.

Use Case

* Name: Obtain most efficient path
* Actor: Registered User
* Preconditions:
* At least one place added to the itinerary.
* Description:

1. Added places to the itinerary.
2. After each place is added the request for the most efficient way is performed

Use Case Diagram

[](https://www.draw.io/#G1wncR35QbcMqEprmrvRVEKKHwpkF76KDN)

### #716 - User Story Name: Improve Visuals in the Map View

* Description: As a User I would like to have several metrics of my trip in the Map view such that I can be more informed.

Acceptance Criteria

* Verify that the general route of the trip is shown in the Map view
* Verify that the next place information metrics are shown in the Map view
* Verify that the next maneuver is listed
* Verify that the distance to the next maneuver is shown
* Verify that controls for overview and center are properly accessible during navigation
* Verify that the maneuver text is shown when each maneuver appears.

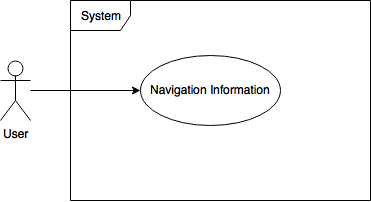
Use Case

* Name: Obtain most efficient path.
* Actor: Registered User.
* Preconditions:
  + At least one place added to the itinerary.
  + Navigation started.
* Description:

1. Added places to the itinerary.
2. Hit the Map Button

### 

Use Case Diagram

[](https://www.draw.io/" \l "G1p_GskKTEQlvJw-Dvo5OufQ3KeuHI9GQ8)

### #717 - User Story Name: Delete task validation [Back-End]

* Description: As a developer, I would like to be able to validate when a task is deleted from the DB, so that I can ensure that a user can only delete a task that belong the same user that created the task.

Acceptance Criteria

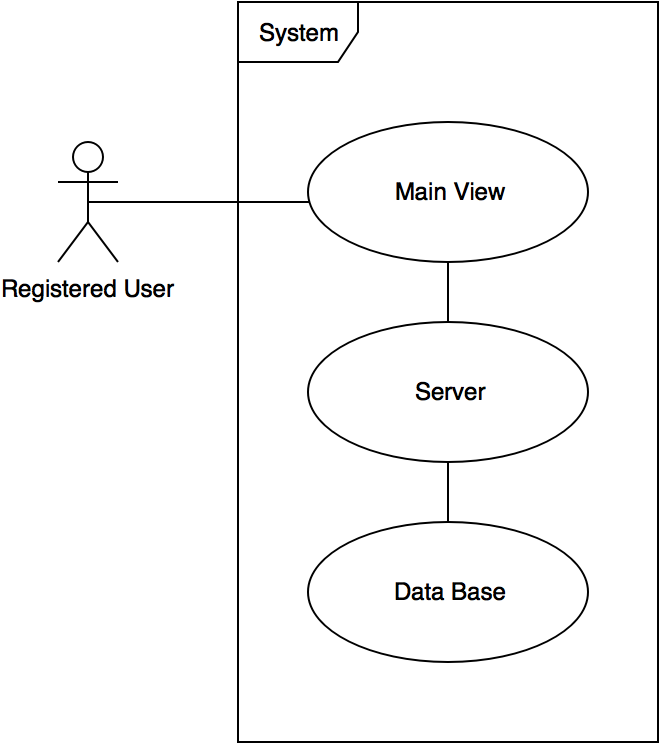
* Verify the user deleting a task is registered.
* Verify the user has an opened session.
* Verify the task belongs to the user deleting it

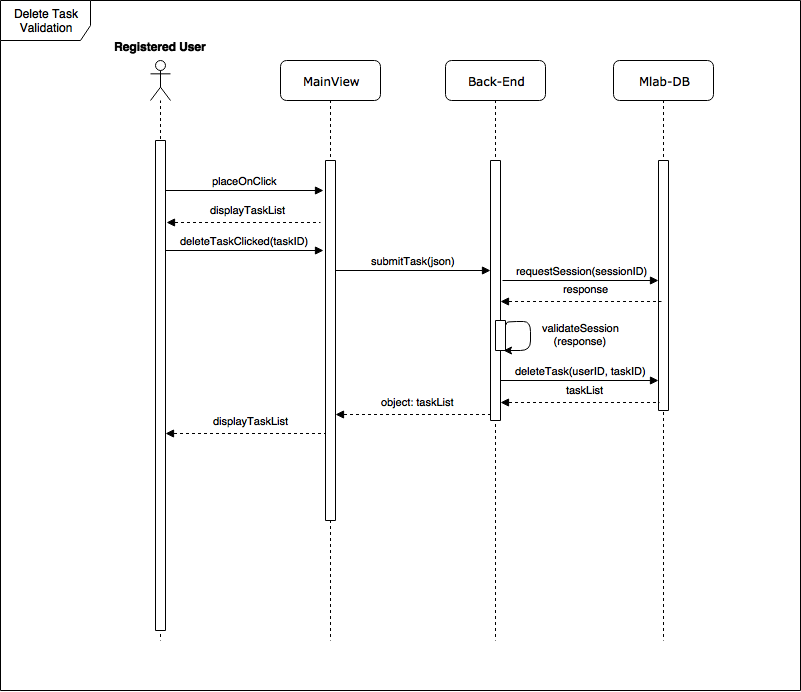
Use Case

* Name: Delete task validation
* Actor: Registered User
* Preconditions:
  + A task has to exist
* Description:

1. Request task to be deleted.
2. Verify user session
3. Delete task from DB
4. Response to client with updated task list

Use Case Diagram

[****](https://www.draw.io/?scale=2#G16kmwZ7tN0hOcAXyromkoszv2mR6PfxuL)Sequence Diagram

[****](https://www.draw.io/#G1Rfk6K8gB7LpKD3VadzmoVpVKDORc9UiL)

Class Diagram

[****](https://www.draw.io/#G1DT4i0XJVQscmImq_Ek3wh5WMqvw3Ybx2)

### #718 - User Story Name: Add task validation [Back-End]

* Description: As a developer, I would like to be able to validate when a task is Added to the DB, so that I can ensure that a user can only add a task that belong the same user.

Acceptance Criteria

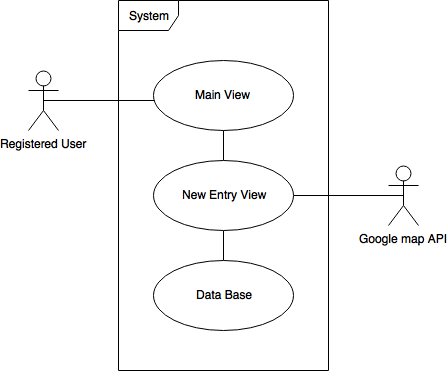
* Verify the user adding a task is registered.
* Verify the user has an opened session.
* Verify the task being added will be added under the same user adding it.

Use Case

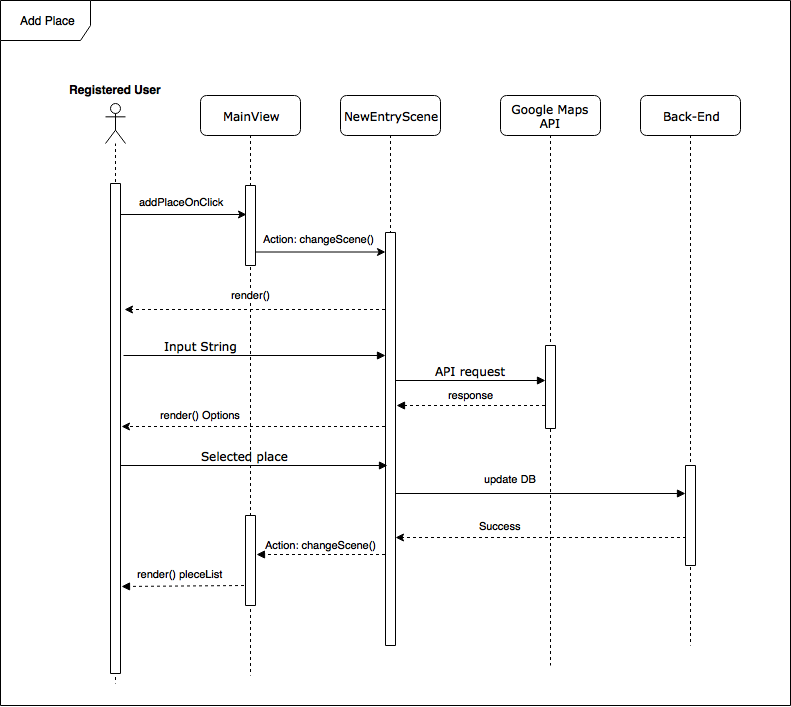
* Name: Add task validation
* Actor: Registered User
* Preconditions:
  + A task has been entered
* Description:

1. Add place button pressed.
2. Search place
3. Select place
4. Display all list of places

Use Case Diagram

[](https://www.draw.io/#G12j4998rb_dwNG2ocbi8ZiOYRygm_-twx)

Sequence Diagram

[****](https://www.draw.io/#G1_g7H1PZMPFAcw85oWUATyUPeQK_Y29Fq)

Class Diagram

[****](https://www.draw.io/#G1DT4i0XJVQscmImq_Ek3wh5WMqvw3Ybx2)

### #719 - User Story Name: Senior Project **Video**

* Description: As a Developer I need to create a video so that I can showcase my best user stories such that I explain the project.

Acceptance Criteria

* Verify that the video shows the flow of the user story
* Verify that the video describes how it is implemented
* Verify that the video show how the story behaves.

### #721 - User Story Name: Senior Project Video

* Description: As a Developer I need to create a video so that I can showcase my best user stories such that I explain the project.

Acceptance Criteria

* Verify that the video shows the flow of the user story
* Verify that the video describes how it is implemented
* Verify that the video show how the story behaves.

### #722 - User Story Name: Senior Project Poster

* Description: As a developer, I would like to have a Senior Project Poster so I can portray what the project is about in Senior Showcase.

Acceptance Criteria

* Verify the poster explains the full project
* Verify the poster explains my contribution to the project.

### 

### #723 - User Story Name: Add place validation [Back-End]

* Description: As a developer, I would like to be able to validate when a place is added to the DB, so that I can ensure that a user can only add a place that belong to the same user.

Acceptance Criteria

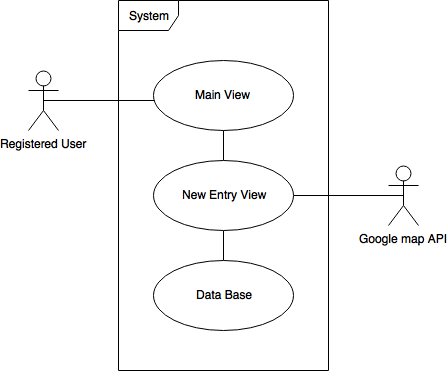
* Verify the user adding a place is registered.
* Verify the user has an opened session.
* Verify the place being added will be added under the same user adding it.

Use Case

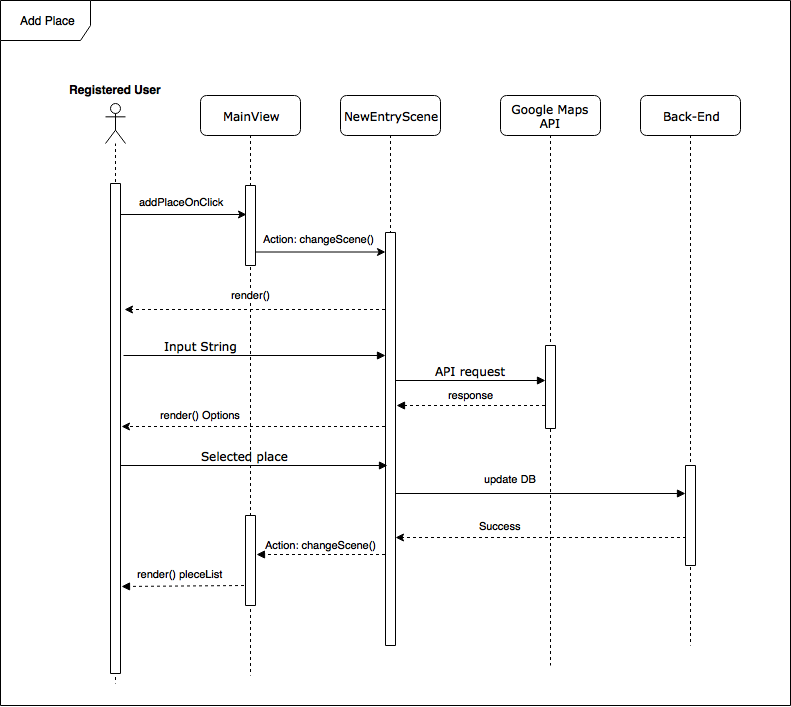
* Name: Add place validation
* Actor: Registered User
* Preconditions:
  + A place has been searched for=
* Description:

1. Add place button pressed.
2. Search place
3. Select place
4. Display all list of places

Use Case Diagram

[](https://www.draw.io/#G12j4998rb_dwNG2ocbi8ZiOYRygm_-twx)

Sequence Diagram

[****](https://www.draw.io/#G1_g7H1PZMPFAcw85oWUATyUPeQK_Y29Fq)

Class Diagram

**[](https://www.draw.io/" \l "G1DT4i0XJVQscmImq_Ek3wh5WMqvw3Ybx2)**

### #724 - User Story Name: Delete place validation [Back-End]

* Description: As a developer, I would like to be able to validate when a place is deleted from the DB, so that I can ensure that a user can only delete a place that belong the same user that created the place.

Acceptance Criteria

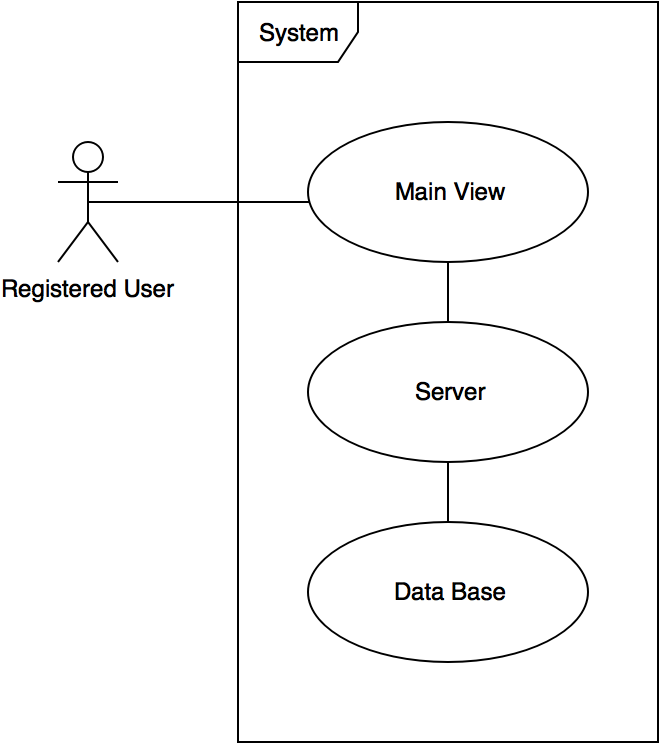
* Verify the user deleting a place is registered.
* Verify the user has an opened session.
* Verify the place belongs to the user deleting it

Use Case

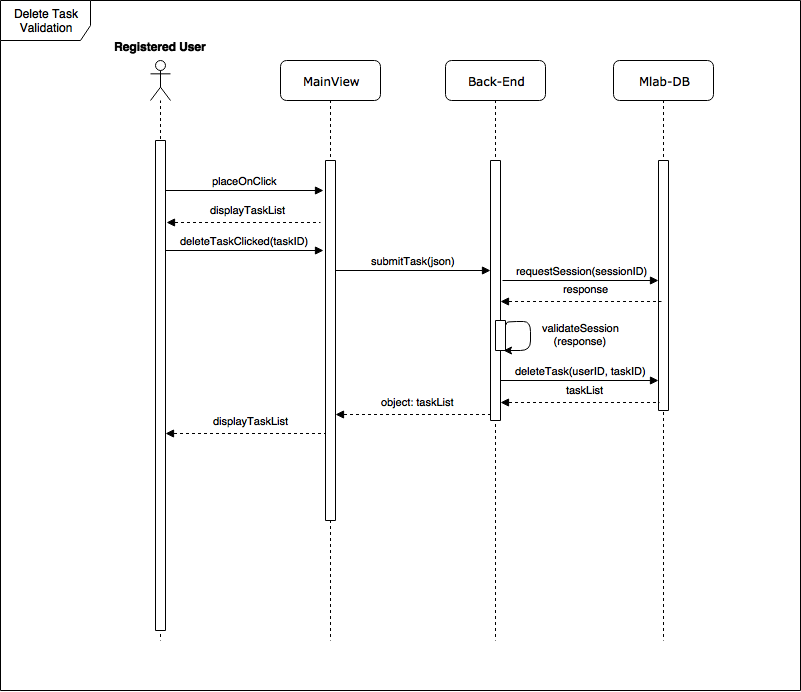
* Name: Delete task validation
* Actor: Registered User
* Preconditions:
  + A task has to exist
* Description:

1. Request place to be deleted.
2. Verify user session
3. Delete place from DB
4. Response to client with updated task list

Use Case Diagram

**[](https://www.draw.io/?scale=2" \l "G16kmwZ7tN0hOcAXyromkoszv2mR6PfxuL)**

Sequence Diagram

[****](https://www.draw.io/#G1Rfk6K8gB7LpKD3VadzmoVpVKDORc9UiL)

Class Diagram

**[](https://www.draw.io/" \l "G1DT4i0XJVQscmImq_Ek3wh5WMqvw3Ybx2)**

### #725 - User Story Name: Senior Project Poster

* Description: As a developer, I would like to have a Senior Project Poster so I can portray what the project is about in Senior Showcase.

Acceptance Criteria

* Verify the poster explains the full project
* Verify the poster explains my contribution to the project.

### #726 - User Story Name: Senior Project Poster

* Description: As a developer, I would like to have a Senior Project Poster so I can portray what the project is about in Senior Showcase.

Acceptance Criteria

* Verify the poster explains the full project
* Verify the poster explains my contribution to the project.

## 

## Non-Implemented User Stories

Due to time constraints, and technical difficulties, we couldn’t complete some User Stories in time. In this section we explain the User Story and their acceptance criteria

### #667 - Setup Login System [Frontend]

* Description: As a registered user, I would like to be able to authenticate to the system so I can access to my tasks and itineraries.

Acceptance Criteria:

* Verify if user enters correct username and password, the user can authenticate successfully to the system
* Verify if user’s username or password does not match with the records in the system, User must be notified that credentials were not correct, but the notification will not specify if the username was valid or not.
* Verify the usernames and passwords must be encrypted when being transferred over the network, and they most never be sent as plain text.
* Users can be authenticated through a third party (Facebook, Google+)
  + If the email collected from a third-party authentication matches an existing account, the account will be linked to the third-party authentication method.
  + If there is no match with an existing account, the third-party credentials will be used to sign-up/in.
* Once signed in, the User should be able to navigate through all of the application views with its tasks and itineraries

### #668 - Setup Reset Password [Frontend]

* Description: As a registered user, I would like to be able to reset my password in case I forget my current password, or if someone has attempt several times to access my account.

Acceptance Criteria:

* Verify the User entered a registered email in order to receive an email with the recovery code and instructions.
* Verify the User is able to enter a new password and confirm it after the proper code has been inserted.
* Verify the new password is different from the old password.

## 

### #674 - View Places on the Map

* Description: As a registered user, I would like to be able to see all of the places of my itinerary on the map, and an excerpt of the tasks.

Acceptance Criteria:

* Verify the User has at least one place in the itinerary.
* Verify the User must be in the map view in order to tap on places and see an excerpt of tasks.
* Verify that after a place is taped, moving the map or tapping outside of the pin will hide the excerpt of the place.

### 

### #677 - Add Favorite Places

* Description: As a registered user, I would like to be able to save places that I visit often as a favorite list.

Acceptance Criteria:

* Verify the User has at least one place in the itinerary.
* Verify the User is in the Place View to selected as a frequently used place.
* Verify if a User visits a place often (5) it will also be added to the frequently used place

### 

### #678 - History of visited places

* Description: As a registered user, I would like to be able to see the places I have visited before.

Acceptance Criteria:

* Verify the User has visited at least one place to have a history of places.
* Verify that every time a user visits a place in the itinerary, the visit will be recorded.
* Verify there is a History View for the user to review the places and tasks accomplished.

### 

# Project Plan

This section describes the planning that went into the realization of this project. This project incorporated the agile development techniques and as such required the sprints to be planned. These sprint plans are detailed in the section. This section also describes the components, both software and hardware, chosen for this project.

In order to plan out a successful execution of To-Do list optimizer, an agile and effective methodology was used to keep all developers, project managers, and product owners on track. Various brainstorming sessions took place detailing issues ranging from user interface design to whether features should be prioritized sooner rather than later. The team also aimed to have the design of each feature completed by our designer Eugenia Imazio before the frontend team began working on them to minimize the cost of development.

Iterations on the product were divided into sprints. Each sprint lasted 2 weeks, with a sprint checkpoint meeting halfway. Their goal was to plan out what each developer had to work on for the next two weeks. Once a feature was built out, demos took place in weekly sprint meetings. Each team member would present their work from the past week and discuss with the rest of the team. This served as a method for all team members to be aware of what their colleagues were working on. Additionally, each day the team had scrum meetings which outlined what each member was working on, what they accomplished since the day before, and what hurdles they were currently facing.

# Hardware and Software Resources

The following is a list of all hardware and software resources that were used in this project:

1. Hardware
   1. Local Server (MacBook Pro)
      1. 16 GB RAM
      2. 256 GB SSD Disk
      3. 1000 GB Transfer
   2. Client
      1. iOS - 11.2.0 or higher
      2. Android - lollipop or higher
2. Software
   1. View
      1. npm - 4.6.1,
      2. react - 16.2.0,
      3. react-native - 0.52.0,
      4. react-native-elements - 0.19.0,
      5. react-native-fontawesome - 5.7.0,
      6. react-native-maps - 0.19.0,
      7. react-native-router-flux - 4.0.0-beta.28,
      8. react-native-vector-icons - 4.5.0,
      9. react-redux - 5.0.6,
      10. redux - 3.6.0,
      11. redux-logger - 3.0.6,
      12. redux-thunk - 2.2.0,
      13. whatwg-fetch - 2.0.3
      14. yarn - 1.3.2
   2. Controller
      1. bcryptjs - 2.4.3,
      2. body-parser - 1.18.2,
      3. cookie-parser - 1.4.3,
      4. debug - 2.6.9,
      5. express - 4.15.5,
      6. jade - 1.11.0,
      7. jsonwebtoken - 8.2.1,
      8. mongoose - 4.13.9,
      9. morgan - 1.9.0,
      10. serve-favicon - 2.4.5
   3. Model
      1. mongodb - 3.4.10

## 

# Sprint Plan

## Sprint 2

Sprint Planning Meeting

Release #: 1

Sprint #: 3

Date: 1/29/2018

Attendees: Salvador Ricardo, Manuel Garcia, Daniel Gonzales, Monique Ross

Start Time: 2:30 pm

End Time: 3:00 pm

After discussion, the velocity of the team was estimated to be 30 hours

* Manuel Garcia
* Daniel Gonzales
* Salvador Ricardo

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* #680 Register User [Backend]
* #690 Scene Basic Transitions
* #695 Update User [ Backend]
* #697 Delete User [Backed]
* #703 Center Map
* #704 Overview Map
* #706 Connect to google maps API to autocomplete place search [front-end]
* #707 Create New Entry Scene [front-end]

The team members indicated their willingness to work on the following user stories.

* Salvador Ricardo
* #680 Register User [Backend]
* #695 Update User [Backend]
* #697 Delete User [Backed]
* Manuel Garcia
* #690 Scene Basic Transitions
* #703 Center Map
* #704 Overview Map
* Daniel Gonzalez
* #706 Connect to google maps API to autocomplete place search [front-end]
* #707 Create New Entry Scene [front-end]

## Sprint 3

Sprint Planning Meeting

Release #: 1

Sprint #: 3

Date: 2/12/2018

Attendees: Salvador Ricardo, Manuel Garcia, Daniel Gonzales, Monique Ross

Start Time: 2:30 pm

End Time: 2:00 pm

After discussion, the velocity of the team was estimated to be 38 hours

* Manuel Garcia
* Daniel Gonzales
* Salvador Ricardo

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* #666 Setup Sign Up
* #669 Add Places
* #670 Add tasks to places
* #671 Show Itinerary and Total Time
* #673 Setup Navigation Information
* #681 Add Places to User [Backend]
* #683 Add Task to Places [Backend]
* #685 Delete Places Manually [Backend]
* #687 Update Places [Backend]
* #688 Delete Task [Backend]
* #684 Delete Places [Backend]
* #705 Display Markers of Places in the Map

The team members indicated their willingness to work on the following user stories.

* Salvador Ricardo
* #681 Add Places to User [Backend]
* #683 Add Task to Places [Backend]
* #685 Delete Places Manually [Backend]
* #687 Update Places [Backend]
* #688 Delete Task [Backend]
* #684 Delete Places [Backend]
* Manuel Garcia
* #673 Setup Navigation
* #705 Display Markers
* #671 Show Itinerary and Total Time
* Daniel Gonzalez
* #666 Setup Sign Up
* #669 Add Places
* #670 Add tasks to places

## Sprint 4

Sprint Planning Meeting

Release #: 1

Sprint #: 4

Date: 2/26/2018

Attendees:

Start Time: 5:00 pm

End Time: 5:30 pm

After discussion, the velocity of the team was estimated to be 36 hours

* Manuel Garcia
* Daniel Gonzales
* Salvador Ricardo

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* #675 Add New Place while Navigating
* #676 Update Route on Map View if Delays
* #679 Track the time spent in places
* #713 Front End Restful API
* #714 Move Backend to Production Server
* #715 Asynchronous request of possible routes

The team members indicated their willingness to work on the following user stories.

* Salvador Ricardo
* #713 Front End Restful API
* #714 Move Backend to Production Server
* Manuel Garcia
* #675 Add New Place while Navigating
* #676 Update Route on Map View if Delays
* #679 Track the time spent in places
* #715 Asynchronous request of possible routes
* Daniel Gonzalez
* #717 Delete task validation [Back-End]
* #718 Add task validation [Back-End]

## Sprint 5

Sprint Planning Meeting

Release #: 1

Sprint #: 5

Date: 3/19/2018

Attendees: Salvador Ricardo, Manuel Garcia, Daniel Gonzales, Monique Ross

Start Time: 6:00 pm

End Time: 6:45 pm

After discussion, the velocity of the team was estimated to be 52 hours

* Manuel Garcia
* Daniel Gonzales
* Salvador Ricardo

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* # 713 Front-end restful API.
* # 679 Track time spent in places.
* # 676 Update route on map view if delays.
* # 716 Improve visuals in the map view.

The team members indicated their willingness to work on the following user stories.

* Salvador Ricardo
* # 713 Frontend restful API.
* Manuel Garcia
* # 679 Track time spent in places.
* # 676 Update route on map view if delays.
* # 716 Improve visuals in the map view.
* Daniel Gonzalez
* #723 Delete place validation [Back-End]
* #724 Add place validation [Back-End]

## Sprint 6

Sprint Planning Meeting

Release #:1

Sprint #: 6

Date: 4/2/2018

Attendees: Salvador Ricardo, Manuel Garcia, Daniel Gonzales, Monique Ross

Start Time: 6:00 pm

End Time: 6:45 pm

After discussion, the velocity of the team was estimated to be 52 hours

* Manuel Garcia
* Daniel Gonzales
* Salvador Ricardo

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their numbers.

* #719 Senior Project Video
* #725 Senior Project Poster
* #721 Senior Project Video
* #722 Senior Project Poster
* #726 Senior Project Poster
* #727 Senior Project Video

The team members indicated their willingness to work on the following user stories.

* Manuel Garcia
* #719 Senior Project Video
* #726 Senior Project Poster
* Daniel Gonzales
* #725 Senior Project Poster
* #727 Senior Project Video
* Salvador Ricardo
* #721 Senior Project Video
* #722 Senior Project Poster

# System Design

This section contains information on the design decisions that went into this project. The architecture patterns are outlined and explained. The entire system is shown in a package diagram and the subsystems are explained. Finally, the design patterns used in the project are discussed.

# Architectural Patterns

For the architecture of the application we decided to use the Model View Controller architecture. This an architectural pattern commonly used for developing using interfaces that divides an application into three interconnected parts. This is done to separate internal representations of information from the ways information is presented to and accepted from the user. The Model View Controller design pattern decouples these major components allowing for efficient code reuse and parallel development.

For our project we used the Model View Controller because it was the design pattern that better suited the fast development of the application. By Using the agile methodology, we were able to produce result much faster by dividing the work between the frontend, backend, and database. This way we can use a Restful API to send information to the users.

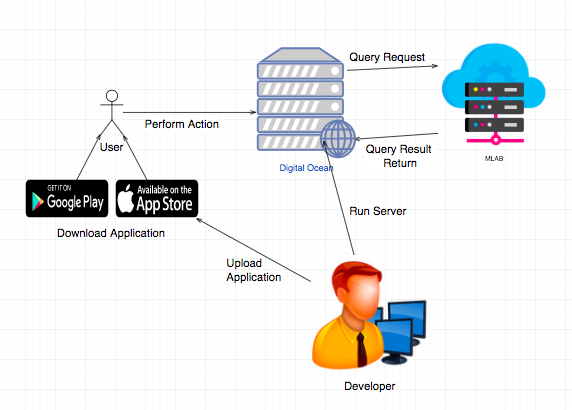
The backend server implements the REST architectural pattern for serving the API endpoints in the form of HTTP methods to the client-side application. On top of the Restful backend server for the client to interact with our frontend.

# System and Subsystem Decomposition

The system can be decomposed into two different aspects. The frontend and the backend. The frontend is divided in two subsystem, Views, and Controllers. Views are used to display the information to the user, and the controllers connect to the backend server where the API is called. The Backend system is divided into two different subsystems, the Server and the Database API. The server is where the application backend is running, this server is always running to keep the application live. This server is hosted in Digital Ocean and will be accessed via SSH. The other subsystem is the Database API where the frontend connects to modify and receive information from the database. The database will be stored in MLAB which offers free storage. The API is build using mongoose, which is a library for MongoDB.

# Deployment Diagram

Deployment of the application is done through two different vias, the front end is distributed through the apple store and the google play store. The application will be submitted to both companies where users can later go and download the application. This application will then connect to an external server which will be hosted in Digital Ocean which in part connects to MLAB where the information is stored.



# Design Pattern

On the frontend, functions of the mobile system have been built using React Native. Each page has its own views written in React with Redux. The Views each use JavaScript and express to connect to the backend server in which the API request are made. The server is written using Node Js with Mongoose in order to connect to the database in MLAB. The Server also has multiple schemas written in MongoDB where the database is defined.

# System Validation

System Validation was done using Postman for the backend User stories.

Unit Tests:

Test case ID: user\_signup\_1 - #666

* Description/Summary of Test: User will input their information, and an account will be created with their information.
* Pre-condition: N/A
* Expected Result: An account with the user information will be created in the database.
* Actual Result: An account was created in the database.
* Status (Fail/Pass): Pass

Test case ID: add\_Places - #669

* Description/Summary of Test: User will be able to add Places to their account.
* Pre-condition: User must be registered.
* Expected Result: A place will be added to the User List
* Actual Result: A place was added to the User’s List
* Status (Fail/Pass): Pass

Test case ID: add\_Task\_to\_Places - #670

* Description/Summary of Test: Once the place is created the user should input a new task an it should be displayed on the screen.
* Pre-condition: Place added the view.
* Expected Result: A new task is displayed on the screen.
* Actual Result: Task is displayed on the screen.
* Status (Fail/Pass): Pass

Test case ID: show\_itineraries - #671

* Description/Summary of Test: User will be able to see their itineraries and the total time it takes.
* Pre-condition: User must have places to drive.
* Expected Result: User will see on the screen the time it takes to complete the itinerary
* Actual Result: A User can see the time it takes to complete the itinerary.
* Status (Fail/Pass): Pass

Test case ID: set\_up\_navigation - #673

* Description/Summary of Test: Developer will be able to set up navigation
* Pre-condition: N/A
* Expected Result: Navigation will show on the screen
* Actual Result: Navigation shows up on the screen.
* Status (Fail/Pass): Pass

Test case ID: add\_Transition\_For\_Adding\_New\_Place\_While\_Navigating - #675

* Description/Summary of Test: User should be able to switch back to the itinerary when navigating on in the map view in general.
* Pre-condition: User is in the map view.
* Expected Result: User will be able to navigate back to the itinerary screen when the button is pressed
* Actual Result: The user is able to successfully navigate back to the itinerary
* Status (Fail/Pass): Pass

Test case ID: setup\_sign\_up\_system\_front\_end - #666

* Description/Summary of Test: User should be able to register in the system
* Pre-condition: User must have not used the app before.
* Expected Result: User will have access to the new account created
* Actual Result: User has access to add places and tasks
* Status (Fail/Pass): Pass

Test case ID: track\_time\_in\_places - #679

* Description/Summary of Test: Developer will be able to track time the user spends in places.
* Pre-condition: N/A
* Expected Result: Time spent in different places is stored.
* Actual Result: The time the User spent in a certain place was saved into the database.
* Status (Fail/Pass): Pass

Test case ID: register\_User - #680

* Description/Summary of Test: User should be able to create a new account to be able to use the application
* Pre-condition: User is not registered.
* Expected Result: A new account of the user should be created in the DB
* Actual Result: A new account of the user is added to the DB as soon as the user enters the information
* Status (Fail/Pass): Pass

Test case ID: Add\_places\_to\_User - #681

* Description/Summary of Test: Developer will be able to add places to an user.
* Pre-condition: N/A
* Expected Result: A place should be added to the User’s account by the developer.
* Actual Result: The place was added to the user’s database account by the developer.
* Status (Fail/Pass): Pass

Test case ID: add\_Tasks\_To\_Places [Backend] - #683

* Description/Summary of Test: User should be able to add a new task to the DB associated with the place.
* Pre-condition: Place added to the DB.
* Expected Result: The new task is added to the DB under the correspondent place.
* Actual Result: New addition is reflected in the DB and it is linked with the existent place id.
* Status (Fail/Pass): Pass

Test case ID: delete\_places\_with\_tasks - #684

* Description/Summary of Test: Place will be deleted from the user’s list when all tasks are completed.
* Pre-condition: User must have tasks in a place.
* Expected Result: The place where all the Tasks are completed should be deleted from the User’s List of places.
* Actual Result: The Place was deleted from the User’s Account.
* Status (Fail/Pass): Pass

Test case ID: delete\_Place\_Manually - #685

* Description/Summary of Test: User should be able to remove an added place at any moment and see the changes in the screen.
* Pre-condition: Place added to the DB.
* Expected Result: The place is removed from the screen and from the DB.
* Actual Result: The place is not displayed anymore on the DB nor the screen.
* Status (Fail/Pass): Pass

Test case ID: update\_Places- #687

* Description/Summary of Test: The User should be able to update already existing place without finishing the tasks.
* Pre-condition: User must have places.
* Expected Result: The place should be deleted from the User’s List.
* Actual Result: The place was deleted from the user’s account.
* Status (Fail/Pass): Pass

Test case ID: delete\_Task\_Manually\_Back\_End #688

* Description/Summary of Test: The Developer should be able to delete a task.
* Pre-condition: none
* Expected Result: Task should be deleted
* Actual Result: Task was deleted from database
* Status (Fail/Pass): Pass

Test case ID: create\_Scene\_Basic\_Transitions - #690

* Description/Summary of Test: User should be able to transition between the screens based on intuitive gestures.
* Pre-condition: None.
* Expected Result: User should be able to smoothly transition between the screens (e.g from the itinerary to the map view).
* Actual Result: Smooth transitions are observed
* Status (Fail/Pass): Pass

Test case ID: update\_User - #695

* Description/Summary of Test: The user will edit and update their information, it could be their email, name, or phone number.
* Pre-condition: User must be registered.
* Expected Result: The information of the user must get updated in the database.
* Actual Result: The new user’s information was updated in the database.
* Status (Fail/Pass): Pass

Test case ID: delete\_User\_Backend - #697

* Description/Summary of Test: User should be able to delete his or her account at any time.
* Pre-condition: User registered in the system.
* Expected Result: User should click the delete button and the account should no longer exist in the DB and the User must be logged out.
* Actual Result: User gets logged out and the account is remove from the DB.
* Status (Fail/Pass): Pass

Test case ID: center\_Map - #703

* Description/Summary of Test: The user will be able to center the map when in the navigation screen, showing the User in the center of the Map.
* Pre-condition: User must be in the navigation screen.
* Expected Result: Map is centered around the User.
* Actual Result: The map was centered around the user, showing the user in the map.
* Status (Fail/Pass): Pass

Test case ID: overview\_Map - #704

* Description/Summary of Test: User should be able to see an overview of all the places she or he has to visit.
* Pre-condition: At least a place should be added to the DB.
* Expected Result: When the overview button is pressed the whole route should be displayed on the map entirely.
* Actual Result: The whole route is displayed on the map.
* Status (Fail/Pass): Pass

Test case ID: display\_Markers\_of\_Places - #705

* Description/Summary of Test: The application will be able to display the markers for all the locations in the applications.
* Pre-condition: User must be in the navigation screen, and have Places added to their list.
* Expected Result: All places will be marked in the Map.
* Actual Result: All the places were marked and identified in the map.
* Status (Fail/Pass): Pass

Test case ID: autocomplete\_Googlemap\_API - #706

* Description/Summary of Test: User should be able to select place from a list of places
* Pre-condition: User registered in the system.
* Expected Result: User should click the selected place from an autocomplete list
* Actual Result: User selected place from a list presented.
* Status (Fail/Pass): Pass

Test case ID: create\_New\_Entry\_Scene - #707

* Description/Summary of Test: As a Developer I would code the necessary components that are going to be on the entry scene, so the user can interact with them.
* Pre-condition: Transition to the new entry scene created.
* Expected Result: Components properly added according to the design and interaction with the DB properly implemented.
* Actual Result: The screen looks like the design and all the components work as expected with the DB.
* Status (Fail/Pass): Pass

Test case ID: front\_End\_Restful\_API- #713

* Description/Summary of Test: The developer will create an API which when the front-end calls, will modify the database, and return information.
* Pre-condition: Developer must comply with the API endpoints rules.
* Expected Result: The information in the database will change, and the Server will send back information about what happened.
* Actual Result: The Database sent back information about what happens in the database and modified the database.
* Status (Fail/Pass): Pass

Test case ID: move\_Backend\_To\_Production\_Server - #714

* Description/Summary of Test: As a developer, I would test that the production server where the application will run and be permanently deployed works.
* Pre-condition: N/A
* Expected Result: Application will reboot seamlessly in the new server, and work like it was in the testing server.
* Actual Result: After the reboot, the application ran perfectly in the new server. Meaning the application was ready for deployment.
* Status (Fail/Pass): Pass

Test case ID: asynchronous\_Request\_Of\_Possible\_Routes - #715

* Description/Summary of Test: As a Developer I would like to implement multiple calls to the Google API with all the possible routes and return the most efficient one.
* Pre-condition: At least two places added to the DB.
* Expected Result: The route with the least time should be returned.
* Actual Result: The returned route was correlated manually using google maps website and it was the one with the least amount of time.
* Status (Fail/Pass): Pass

Test case ID: improve\_Visuals\_In\_Map\_View - #716

* Description/Summary of Test: As a Developer I would code the necessary components that to improve user experience
* Pre-condition: Map View is created
* Expected Result: Components properly added according to the design.
* Actual Result: The screen looks like the design and all the components work as expected.
* Status (Fail/Pass): Pass

Test case ID: delete\_Task\_Validation - #717

* Description/Summary of Test: The User should be validated when they try to delete a task in their application.
* Pre-condition: The user should be logged in.
* Expected Result: The user will be validated when deleting a task, and if the user is logged in the tasks will be deleted.
* Actual Result: The user was successfully validated, and the tasks was deleted.
* Status (Fail/Pass): Pass

Test case ID: add\_Task\_Validation - #718

* Description/Summary of Test: As a Developer everytime the user adds a task needs to be validate that it is a right user and it is logged in.
* Pre-condition: User must be logged in.
* Expected Result: Only logged in users can add tasks and only the tasks that belong to them.
* Actual Result: Proper tasks are added to its specific users.
* Status (Fail/Pass): Pass

Test case ID: add\_Place\_Validation - #723

* Description/Summary of Test: The System will validate the user is who they say they are before adding a place to their list.
* Pre-condition: User must be logged in.
* Expected Result: The User will be validated, and a place will be added to their list.
* Actual Result: The user was successfully validated by the system, and the place added to their list.
* Status (Fail/Pass): Pass

Test case ID: delete\_Place\_Validation - #724

* Description/Summary of Test: As a Developer everytime the User deletes a place needs to be validate that it is a right user and it is logged in.
* Pre-condition: User must be logged in.
* Expected Result: Only logged in users can delete places and only the places that belong to them.
* Actual Result: Proper tasplacesks are deleted to its specific users.
* Status (Fail/Pass): Pass

## 

# Glossary

* MVC: Model, View, Controller architecture design paradigm.
* React native: A popular JavaScript framework to develop native mobile applications developed by Facebook and it is currently in its beta version. Used to develop the view part of the applications and keep the status of the running app.
* Express server: A NodeJS module necessary to run the restful API for the To-do List application.
* Mlab: Online hosting service. Used to host the mongo DB database to store user data and sessions.

## 

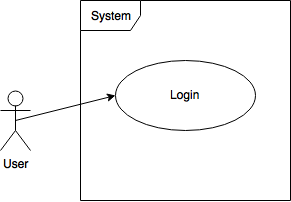
# Appendix

## Appendix A – UML Diagrams

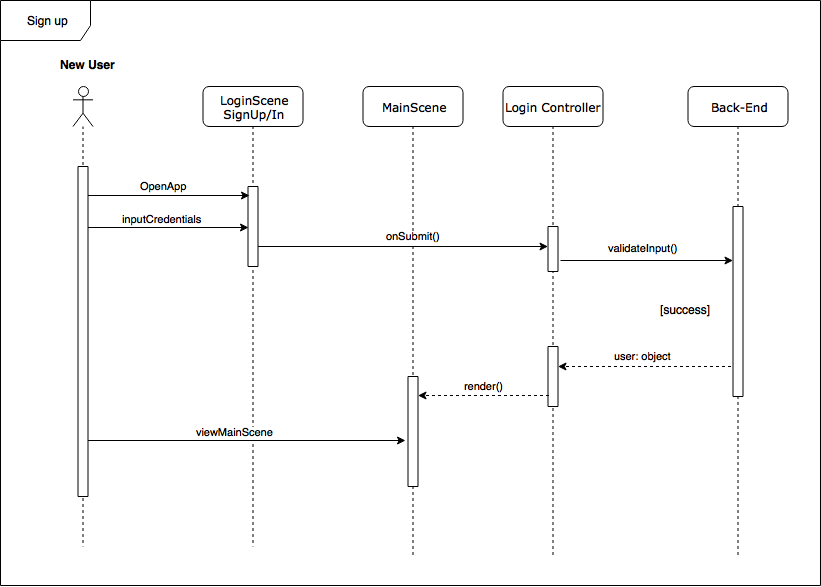
### 

### #666 - User Story Name: Setup Sign Up System [Front-End Back-End]

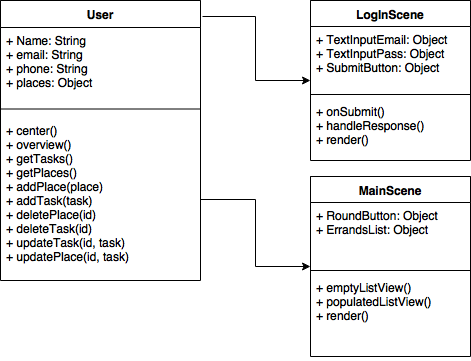
Use Case Diagram

[****](https://www.draw.io/#G16iTk2pezNMW1Mqd04RT8Vdwa1eyhSy10)

Sequence Diagram

[](https://www.draw.io/#G1ErlZFRjS5QwWbFQ7KTCciWGIzhaD0LyI)

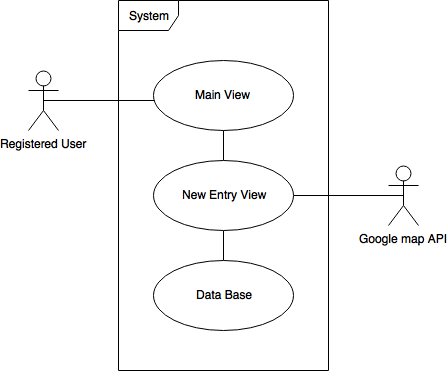
Class Diagram

[****](https://www.draw.io/#G1ZjaYELaq4wQogBqoPuZAwQd8VjSe8a9N)

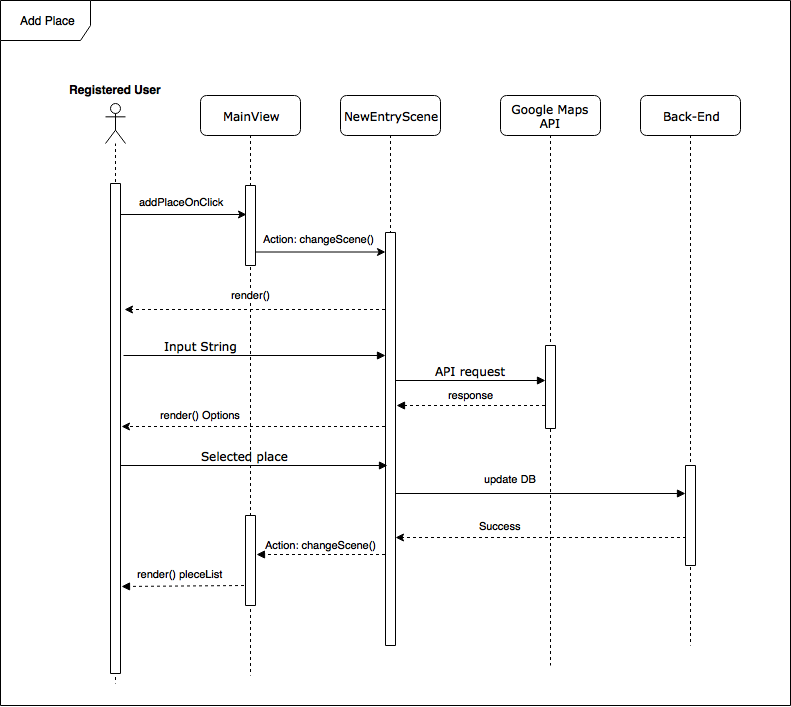
### 

### #669 - User Story Name: Add Places

Use Case Diagram

[](https://www.draw.io/#G12j4998rb_dwNG2ocbi8ZiOYRygm_-twx)

Sequence Diagram

[****](https://www.draw.io/#G1_g7H1PZMPFAcw85oWUATyUPeQK_Y29Fq)

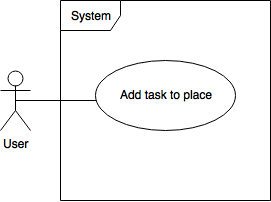
Class Diagram

[****](https://www.draw.io/#G1DT4i0XJVQscmImq_Ek3wh5WMqvw3Ybx2)

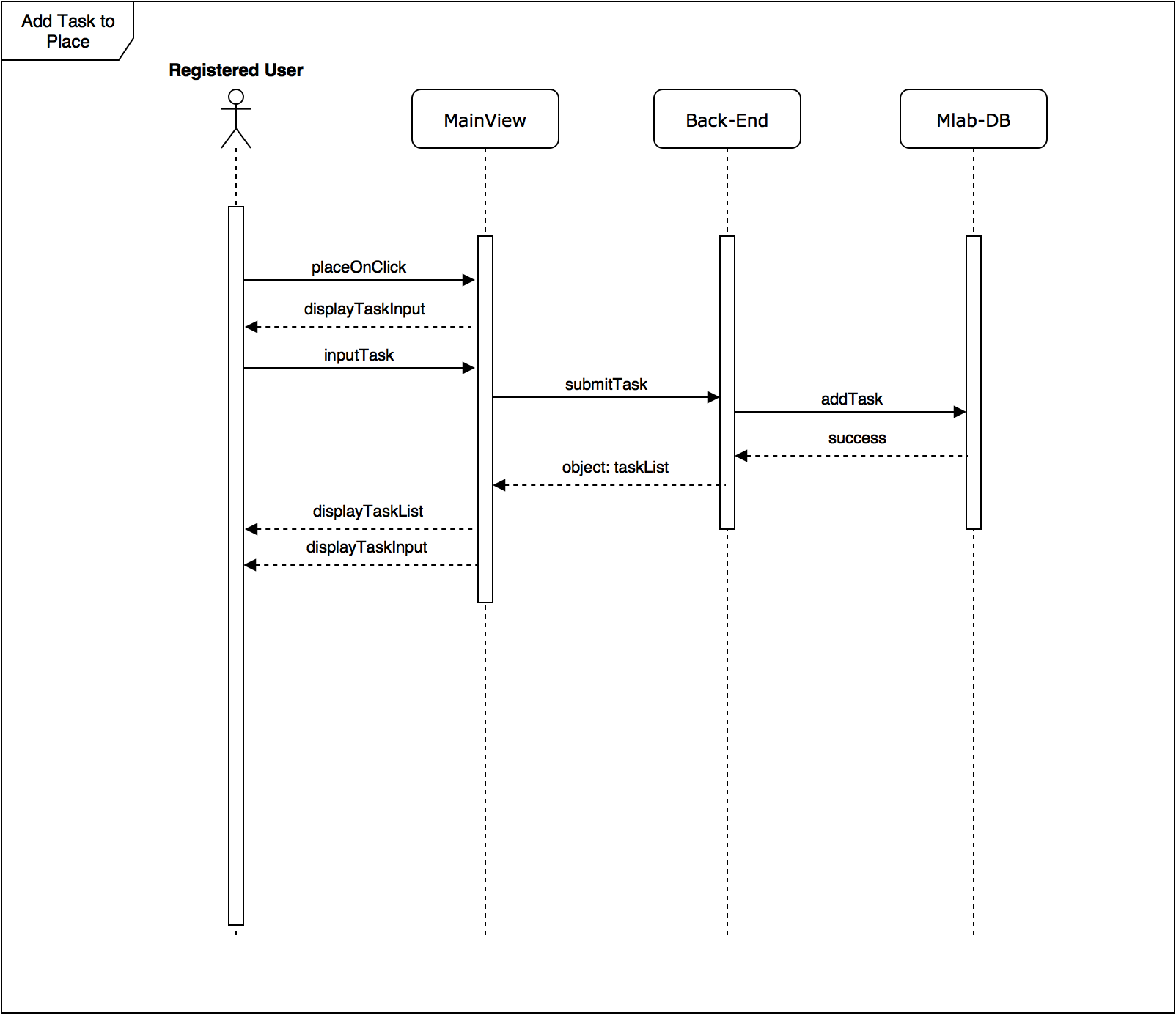
### 

### #670 - User Story Name: Add tasks to places

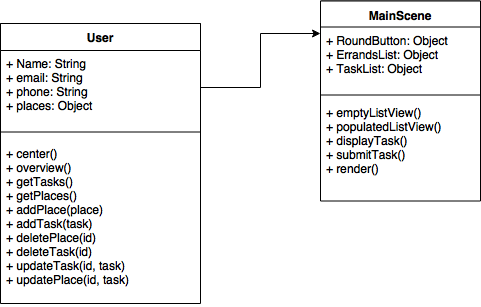
Use Case Diagram



Sequence Diagram

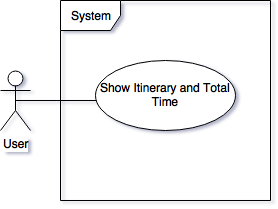
[****](https://www.draw.io/?scale=2#G13gbxcYIfxItb1yysYhxXYyUeIT7zApUG)

Class Diagram

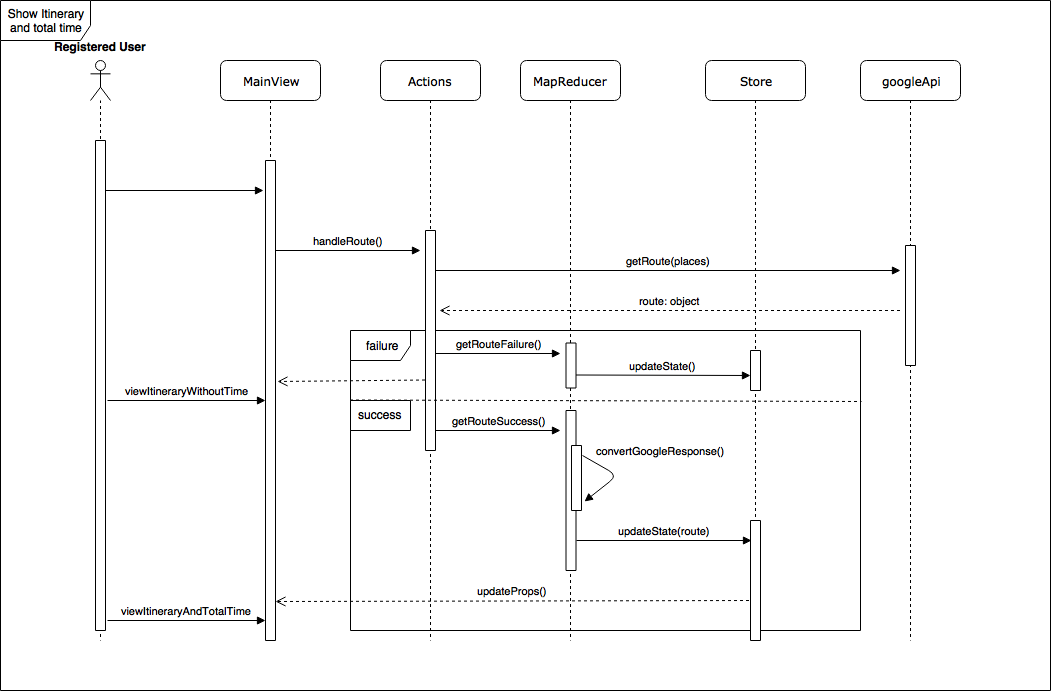
[****](https://www.draw.io/#G1QosaGMEeGlS2AqMy5-AuBaDMfBXW1Kny)

### #671 - User Story Name: Show Itinerary and Total Time

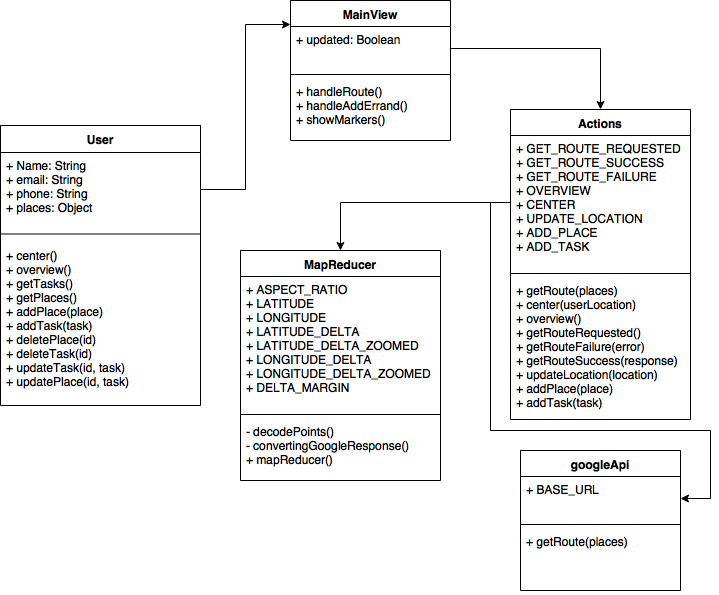
Use Case Diagram

[](https://www.draw.io/#G17qR4hswbXMCXslEBmGztrD8hbSnLaO4t)

Sequence Diagram

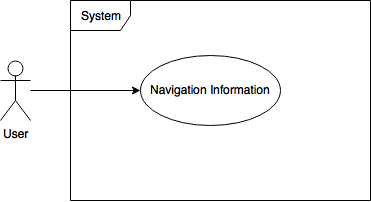
[****](https://www.draw.io/#G1s6EtzdjKOrn8EnegClYDHqMrVZ0JAynY)

Class Diagram

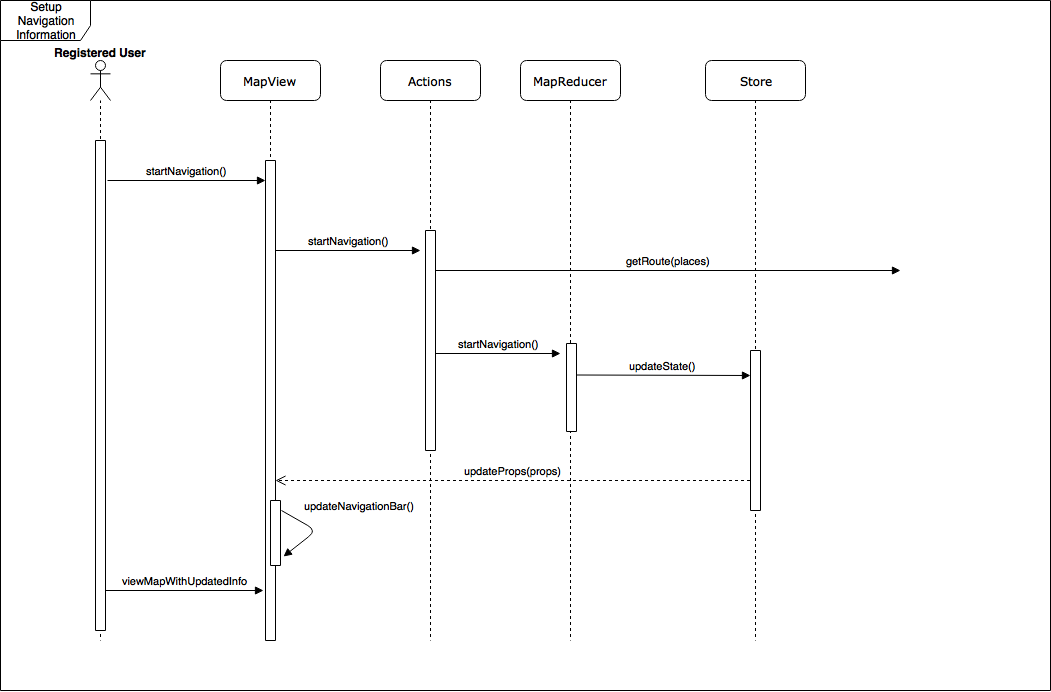
[****](https://www.draw.io/#G1Aajq_7ANgno1dYggZi2ONa1_OWk7rZX3)

### #673 - User Story Name: Setup Navigation Information

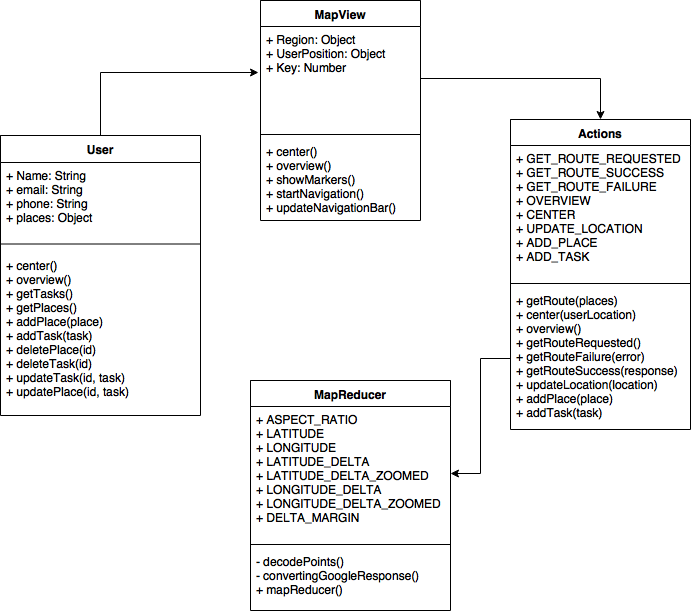
Use Case Diagram

[](https://www.draw.io/#G1p_GskKTEQlvJw-Dvo5OufQ3KeuHI9GQ8)

Sequence Diagram

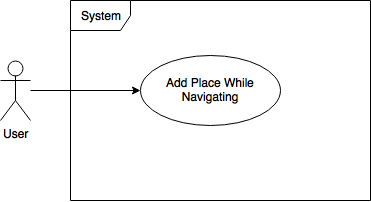
[****](https://www.draw.io/#G17n5fidxMU3THd0hoQx3lql4gD4Hx44Ym)

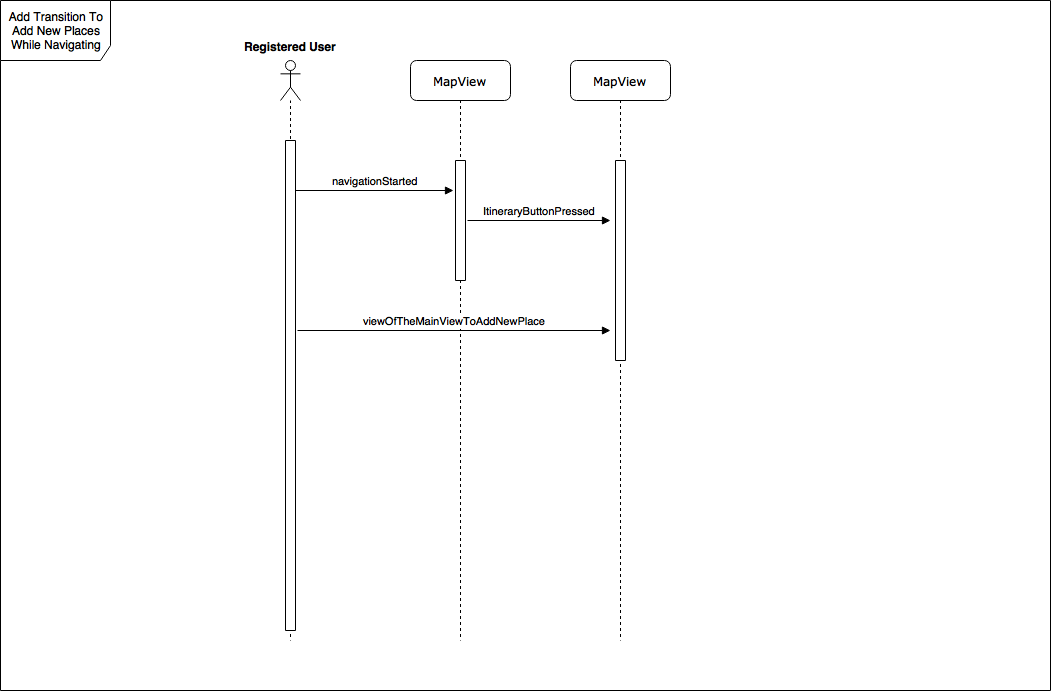
Class Diagram

[****](https://www.draw.io/#G1BScoTOqfW9GYSTzvp3pLsO180H67KuQQ)

### #675 - User Story Name: Add Transition for Adding New Place while Navigating

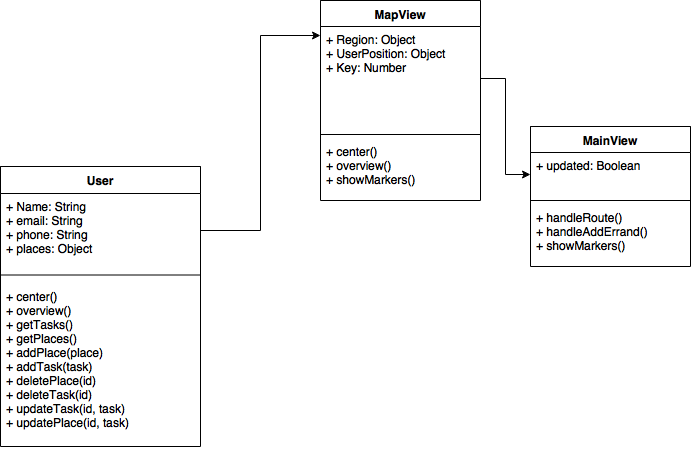
Use Case Diagram

[](https://www.draw.io/#G123fpUjI1G01psZkEHPdf-RLd0teSEwsz)Sequence Diagram

**[](https://www.draw.io/" \l "G1JlPrMZsqyHkTmmRC-4tbYbr7Hcx8V-Rf)**

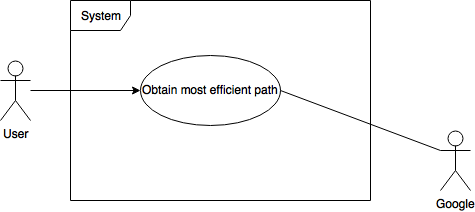
### 

Class Diagram

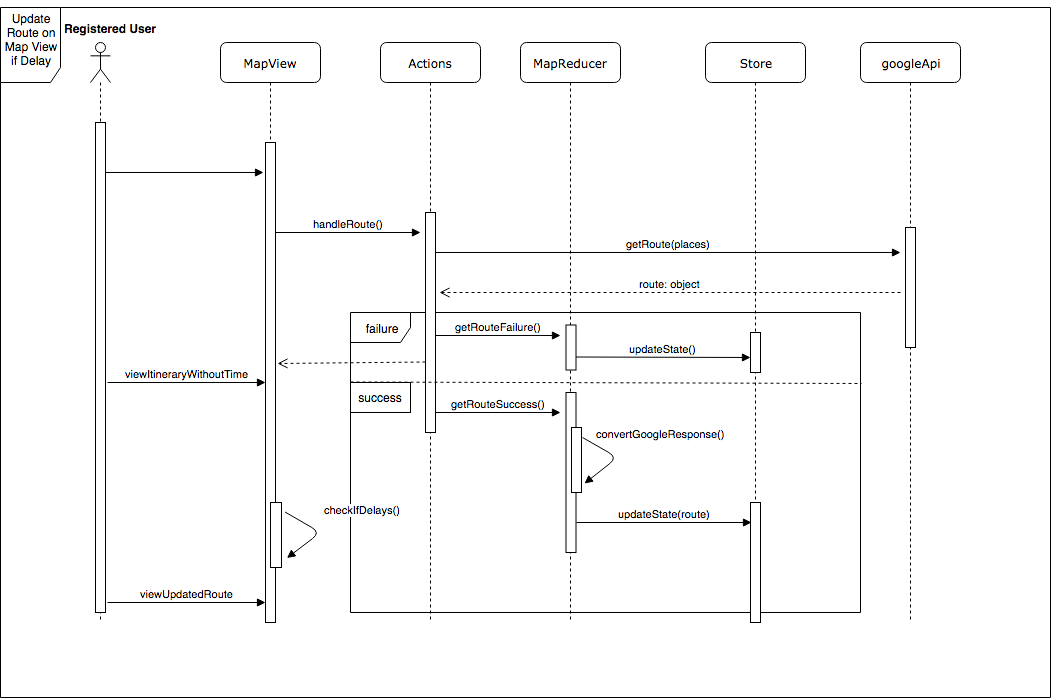
**[](https://www.draw.io/" \l "G1DYBMdiqB_FWGEOlYtws7lenICoNeRnhn)**

### #676 - User Story Name: Update Route on Map View if Delay

Use Case Diagram

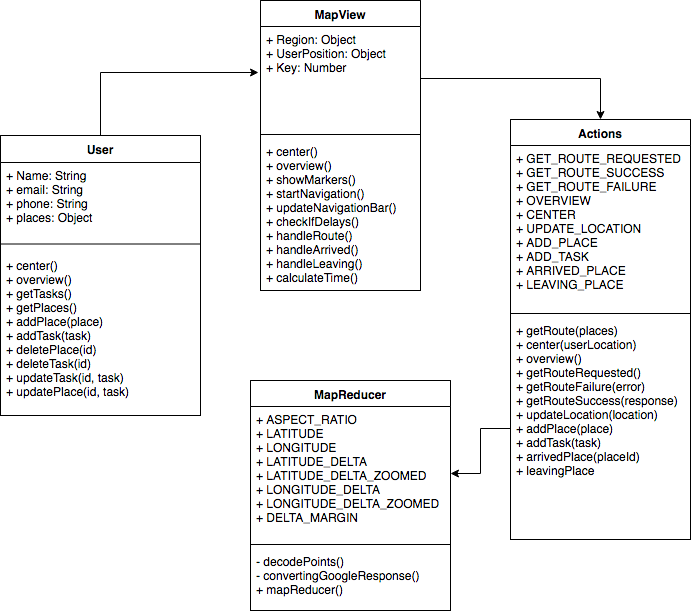
[](https://www.draw.io/" \l "G1wncR35QbcMqEprmrvRVEKKHwpkF76KDN)

Sequence Diagram

**[](https://www.draw.io/" \l "G1ZbcsLVCyxfFfKHNIOikVuoi_7QJlQ51L)**

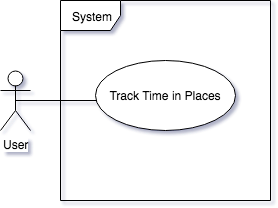
### 

Class Diagram

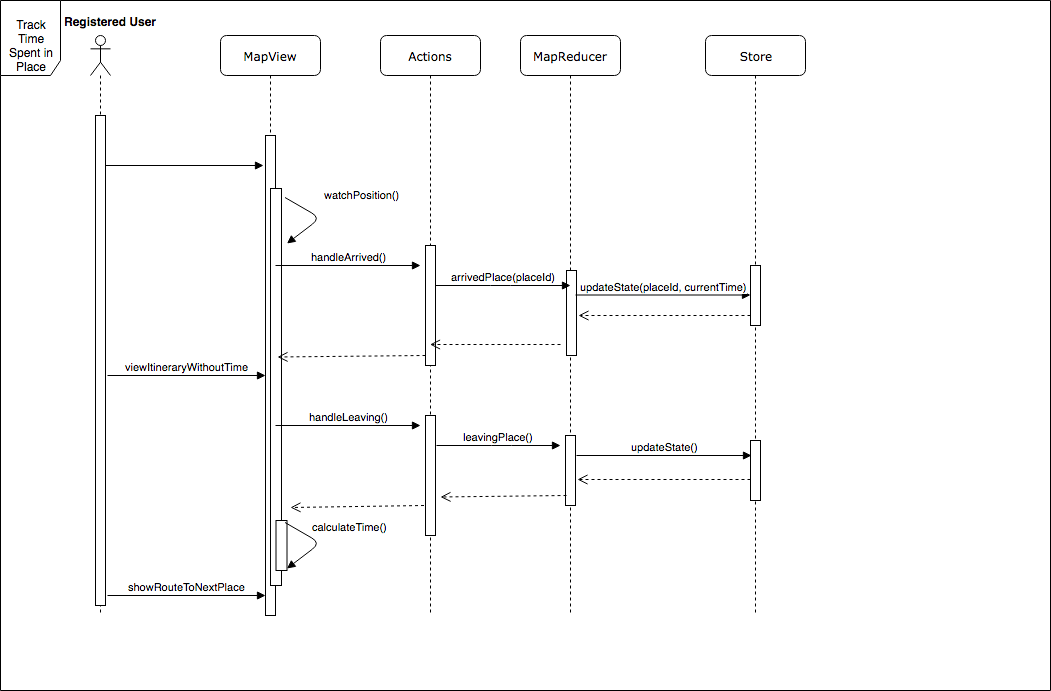
**[](https://www.draw.io/" \l "G1_6-4d4y3yn_UQN3qO1FOxYN88TKZTd0R)**

### #679 - User Story Name: Track The Time Spent in Places

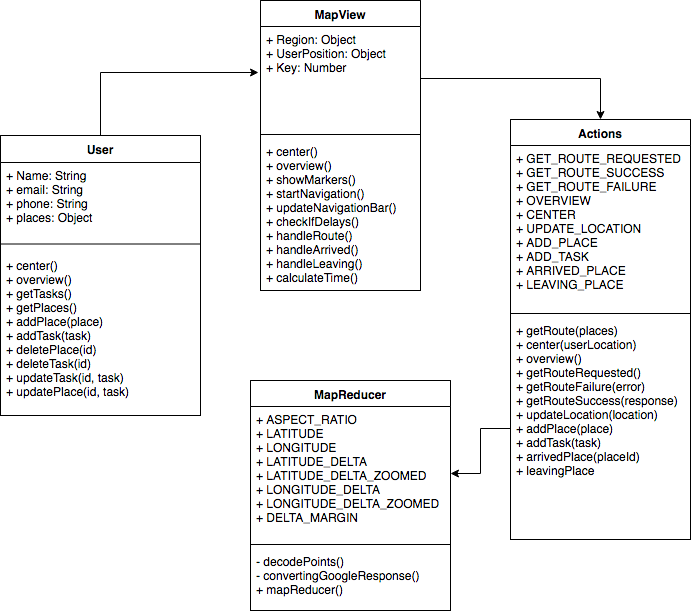
Use Case Diagram

[](https://www.draw.io/" \l "G1yTBTzPlOuzIYrloQzKp1gV-w9srTpFwJ)

Sequence Diagram

**[](https://www.draw.io/" \l "G1x8hkh72pkYcRTmwYSFV5zwPiJwGeT1ZG)**

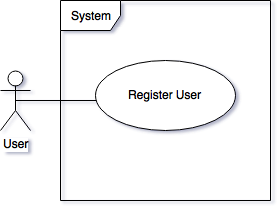
Class Diagram

[****](https://www.draw.io/#G1_6-4d4y3yn_UQN3qO1FOxYN88TKZTd0R)

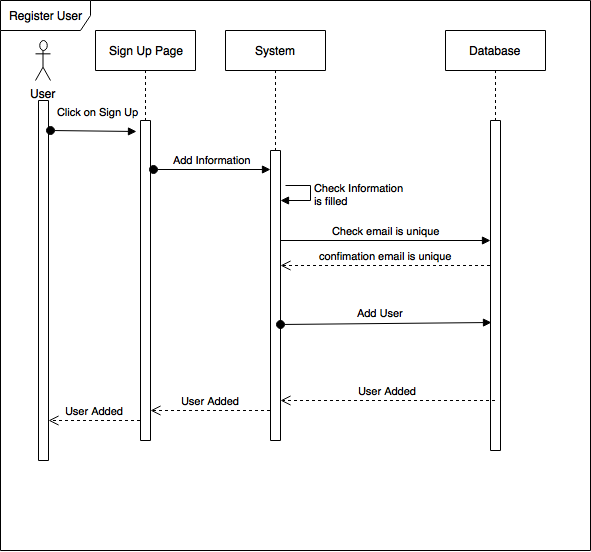
### 

### #680 - User Story Name: Register User [Backend]

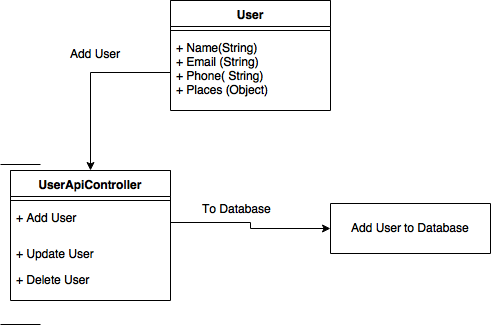
Use Case Diagram

[](https://www.draw.io/#G1w_NJwyEDbY2vOOVr5aLVedeh8lFzmU_q)

Sequence Diagram

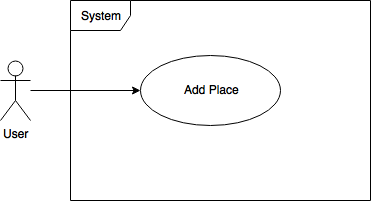
[](https://www.draw.io/#G1LsF-a433NMOuDXI_s5d9K_w_9_QqXWFn)

Class Diagram

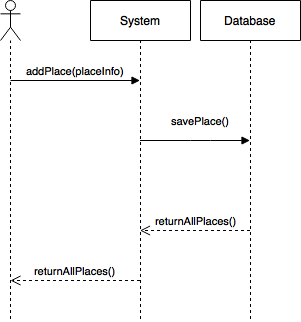
[](https://www.draw.io/#G1bwhPAyMTBE2li3uBzBoWDNqbsp8MSQ6J)

### #681 - User Story Name: Add Places to User [Backend]

Use Case Diagram

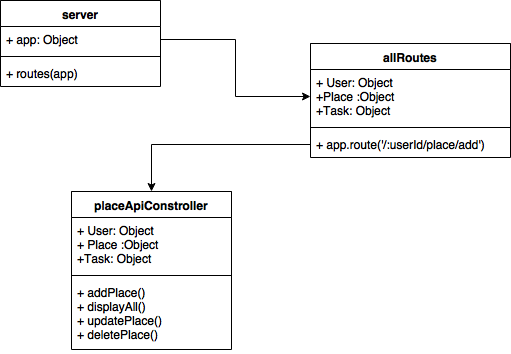
[](https://www.draw.io/#G1NFKmu8fMA0PMM_H0QNHjAAQWucBiHQgn)

Sequence Diagram

[](https://www.draw.io/#G1iqN0Gh_igm4UhFdtG7hVpz8FapZ_Jp2Y)

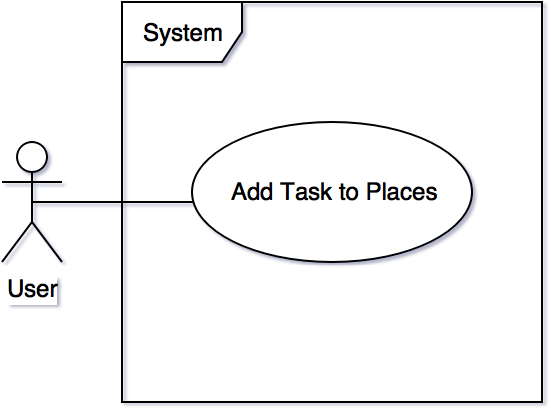
### 

Class Diagram

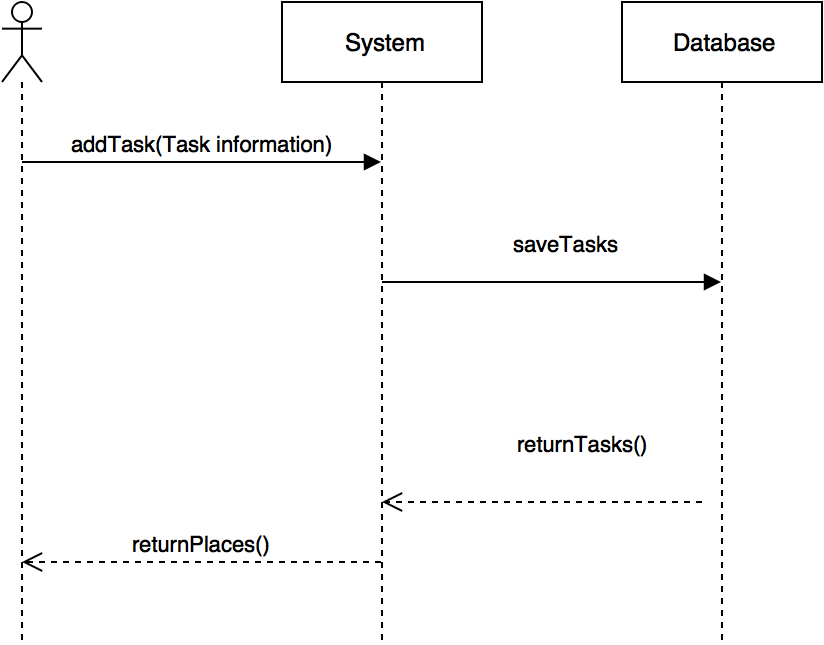
[](https://www.draw.io/#G1ejV4hoNXkfQ2_JDeldmhMLhWbD3ZKZTS)

### #683 - User Story Name: Add Task to Places [Backend]

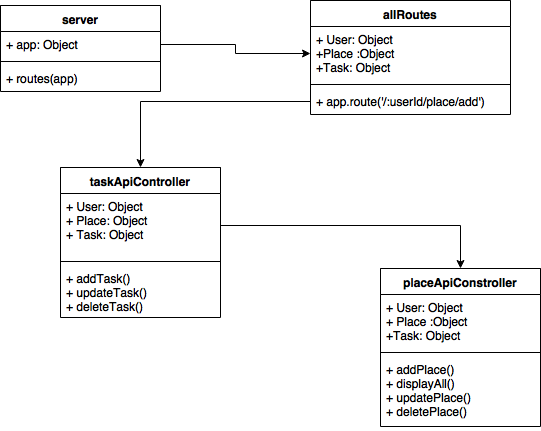
Use Case Diagram

[](https://www.draw.io/?scale=2" \l "G1G3a9oc4pP5SpJ6JoCLgqPYOm8boCGyMA)

Sequence Diagram

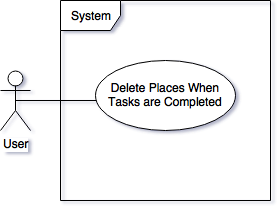
[](https://www.draw.io/?scale=2#G1p6S2ElNBuvQEi8cPub9oGQvSy4SJjmBi)

Class Diagram

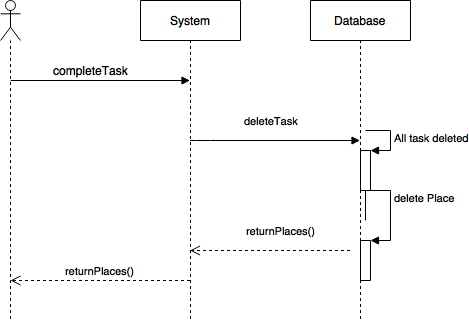
[](https://www.draw.io/#G1WDnxkGl2TANiK9EvLlLagbVEoGlMd1eO)

### #684 - User Story Name: Delete Places When Tasks are Completed

Use Case Diagram

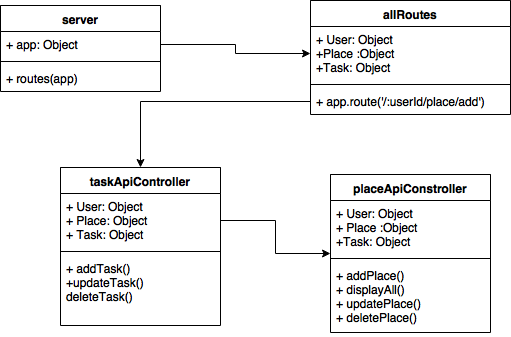
[](https://www.draw.io/" \l "G1zCKeRx-cxwymuS23il0WtrZ5XBa0tleb)

Sequence Diagram

[](https://www.draw.io/" \l "G1hrwiSWAVEPMlfVRauajDOxY7HDMz_ikC)

### 

Class Diagram

[](https://www.draw.io/#G1_gOyi8VzGAVwJhRb3pFZHCrUmZvsrPLb)

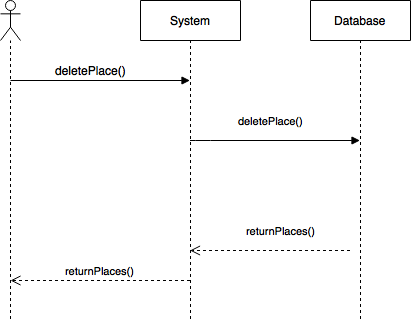
### 

### #685 - User Story Name: Delete Places Manually

Use Case Diagram

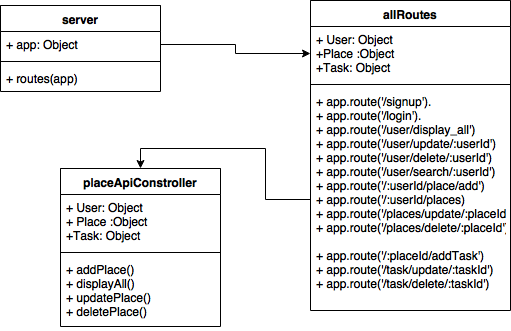
[](https://www.draw.io/" \l "G1SlxZaPdPgI8mKiyohJL1to9jwSUF4nSK)

Sequence Diagram

[](https://www.draw.io/" \l "G1mYKurHxw3lYVTWafbUEOfJy2A9ZCseg7)

### 

Class Diagram

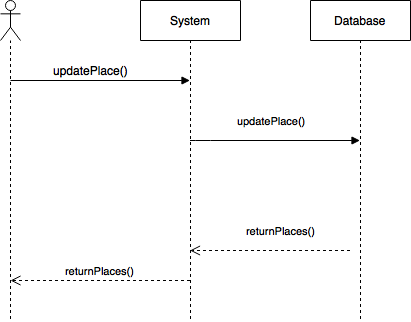
[](https://www.draw.io/" \l "G1ebEVd37Gg0tyFgtmNjOuE-yrf5ZxBdz0)

### #687 - User Story Name: Update Places [Backend]

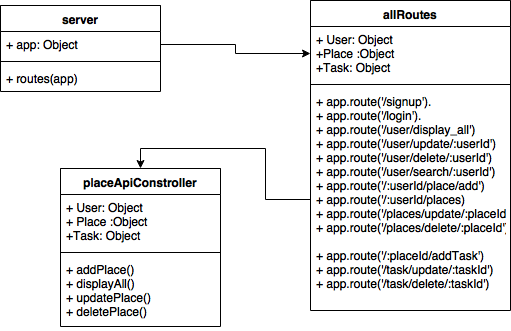
Use Case Diagram

[](https://www.draw.io/" \l "G1rE3DLxBmOgkD27pqFVu6gmvKVFllWpSk)

Sequence Diagram

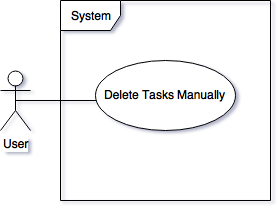
[](https://www.draw.io/#G1IY-Glfu1wPByB0MyE7Krtqcanu96UDnj)

Class Diagram

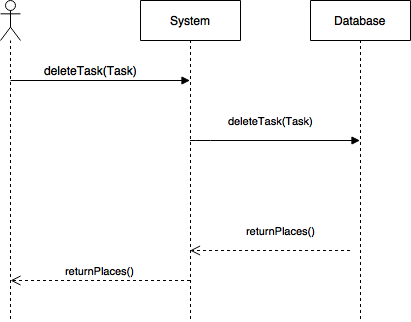
[](https://www.draw.io/#G1peyUS0gBKvBHv38wO0x2STXmItTwrCjd)

### #688 - User Story Name: Delete Task Manually [Backend]

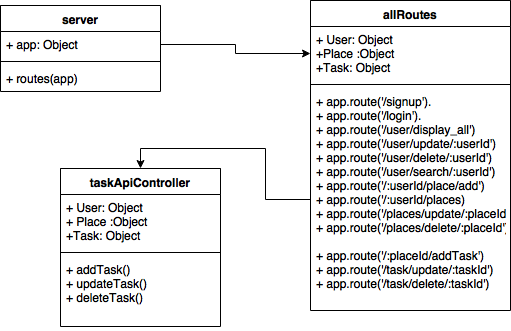
Use Case Diagram

[](https://www.draw.io/" \l "G1lVPPXlgZy5Vz7LBHhxfTjv54IJGBoJBc)

Sequence Diagram

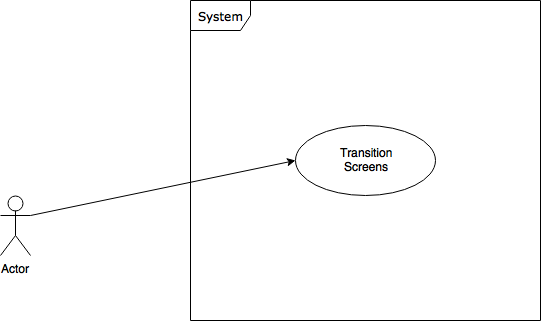
[](https://www.draw.io/#G1C1EgRsvb0GTiMQjn9sMAkVTz1UEjOdVE)

Class Diagram

[](https://www.draw.io/#G1hKnRoDpa1TJer3jAeGDBE0qNjgAYL6fd)

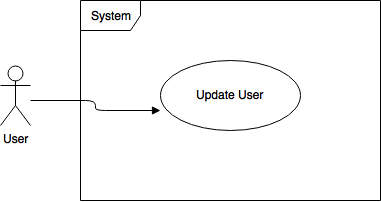
### #690 - User Story Name: Create Scenes Basic Transitions

Use Case Diagram



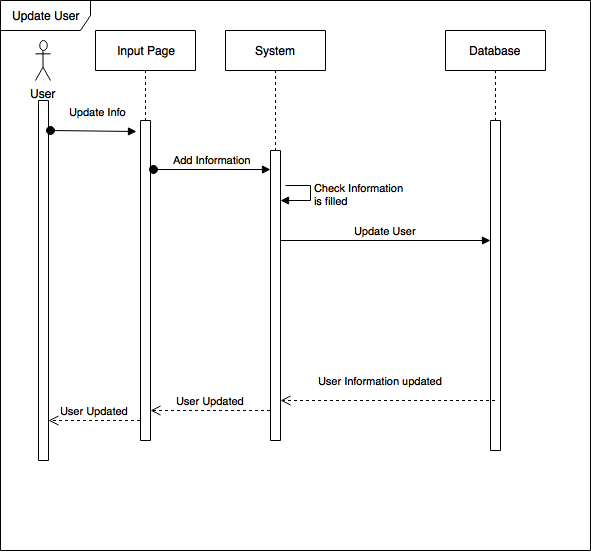
### #695 - User Story Name: Update User [Backend]

Use Case Diagram

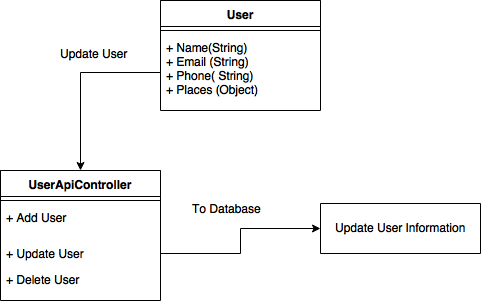
[](https://www.draw.io/#G18ydgocI36M8FX5HJn6IjJmnjy2mXApll)

Updating user: To-do List Optimizer allows registered users to update their information using the mobile app.

Sequence Diagram

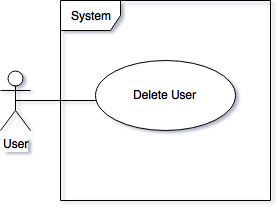
[](https://www.draw.io/" \l "G1ZkP08lvlTVhrs8j3IS78N6oMwyVbDITr)

Class Diagram

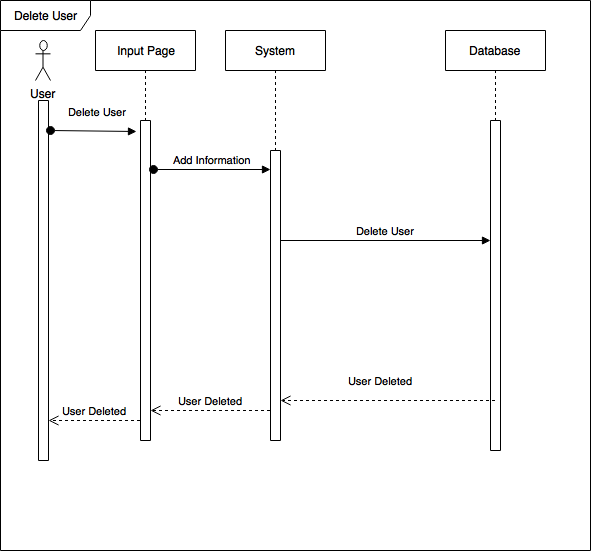
[](https://www.draw.io/" \l "G1eEEqdfKGsGYiMXu3LAWmzI63N1vbub8y)

### #697 - User Story Name: Delete User [Backend]

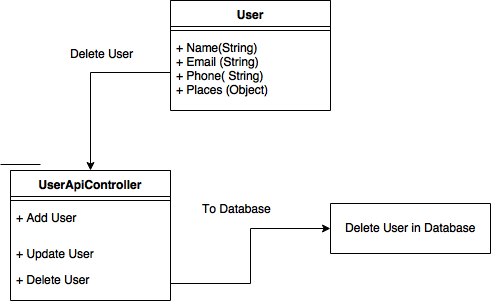
Use Case Diagram

[](https://www.draw.io/#G1Ex0yw77MI-4eeomUQHYK18_c9tyS3E2k)

Sequence Diagram

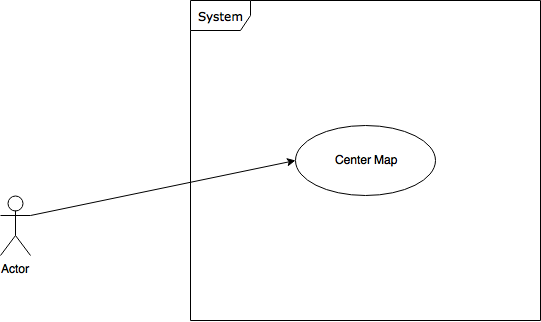
[](https://www.draw.io/" \l "G1tXdeFH7-fLFdFl9WiTd4w0BU4kEe19MO)

Class Diagram

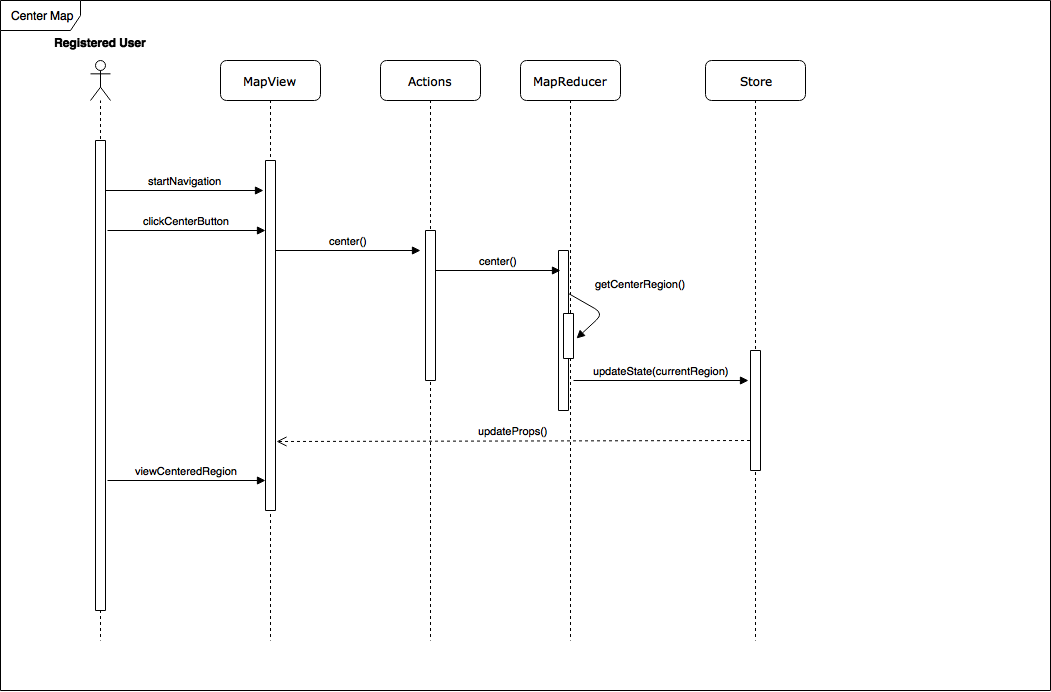
[](https://www.draw.io/#G1JiT5-fXYdhPbMG2Fe79JQQ-3-FqBXdAc)

### #703 - User Story Name: Center Map

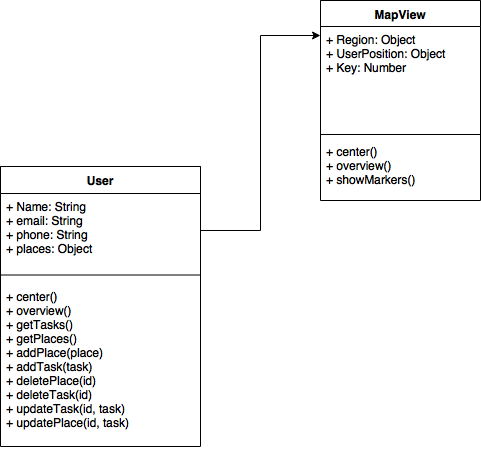
Use Case Diagram



Sequence Diagram



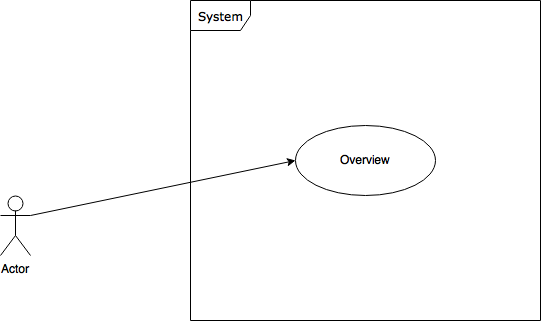
Class Diagram

****

## 

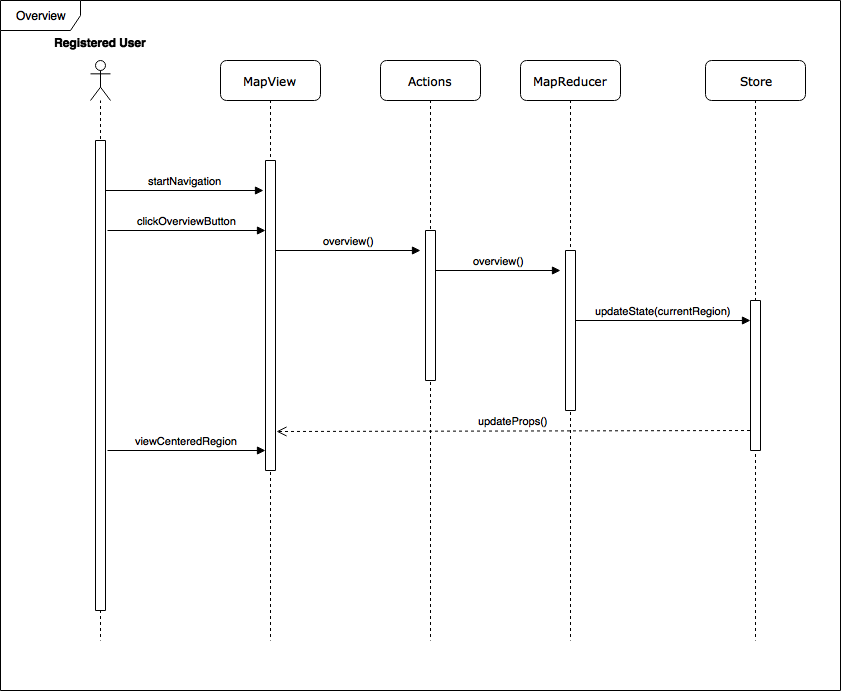
### #704 - User Story Name: Overview Map

Use Case Diagram

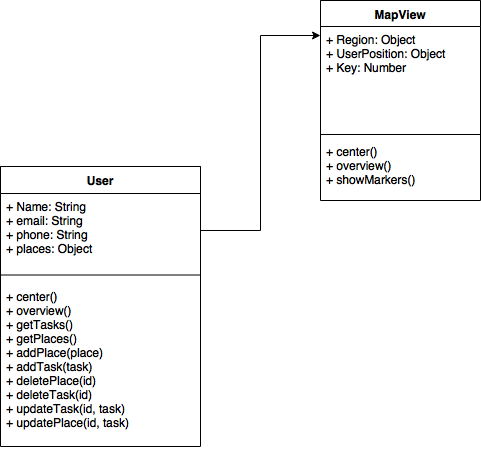


## 

Sequence Diagram

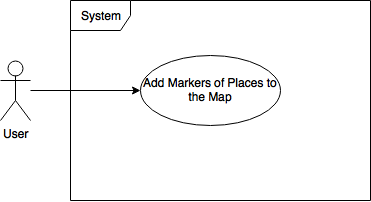


Class Diagram

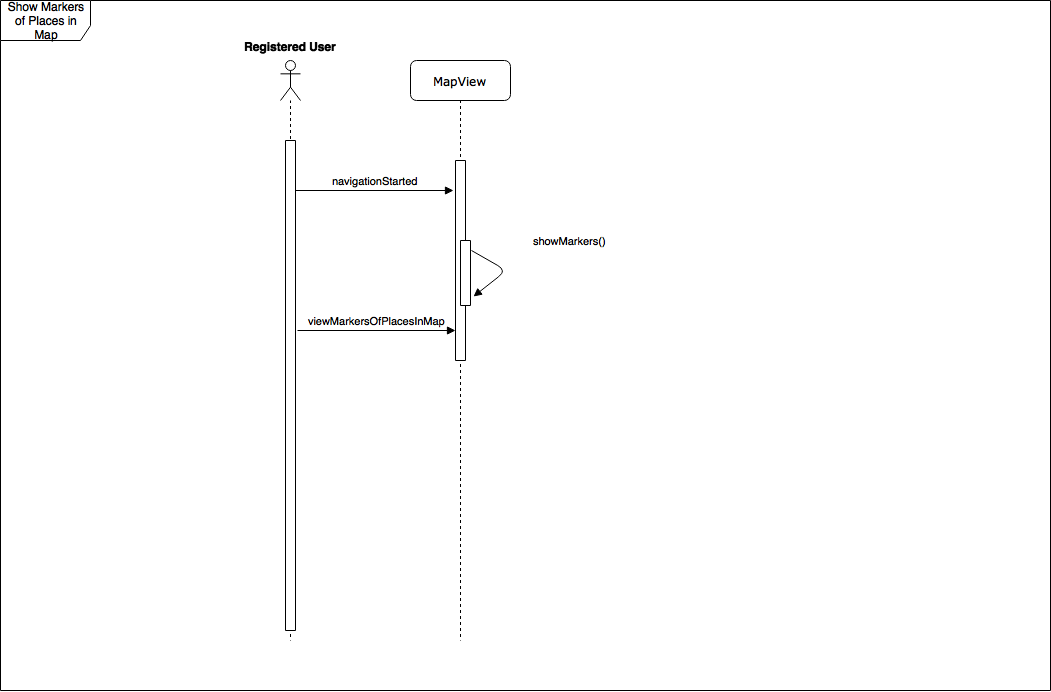


### #705 - User Story Name: Display Markers of Places in the Map

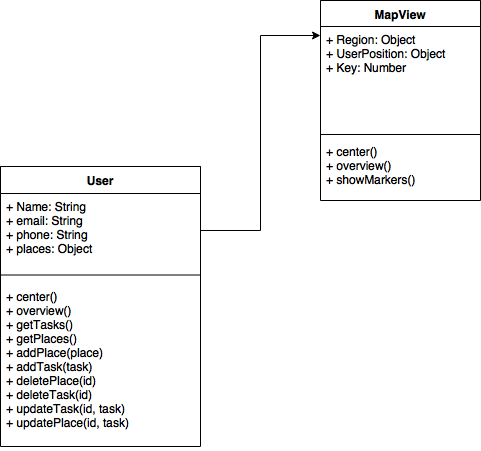
Use Case Diagram

[](https://www.draw.io/" \l "G1NFKmu8fMA0PMM_H0QNHjAAQWucBiHQgn)

Sequence Diagram

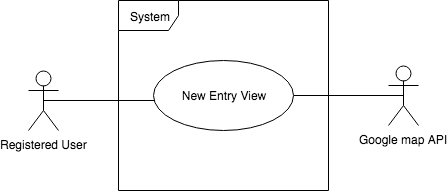
[****](https://www.draw.io/#G1XlZsZUG7eJqnew0_dAdiMyHCbYDnHG6L)

Class Diagram

[****](https://www.draw.io/#G1TBbFMoZiZsPOP7HIC6MVc4QJOYcVNEtI)

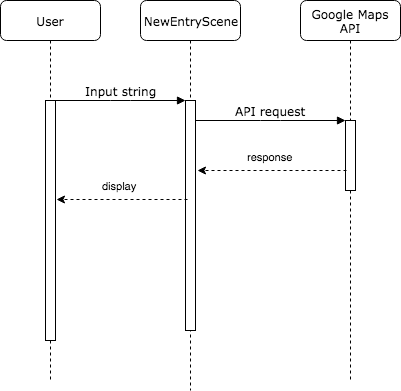
### #706 - User Story Name: Connect to google maps API to autocomplete place search

Use Case Diagram

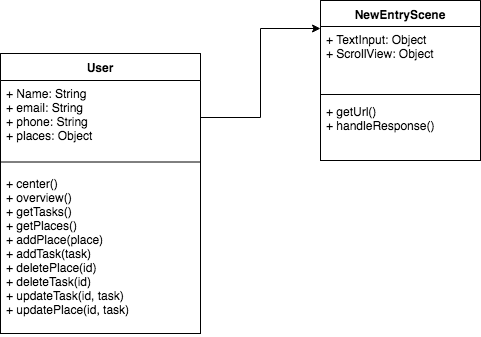
****

## 

Sequence Diagram

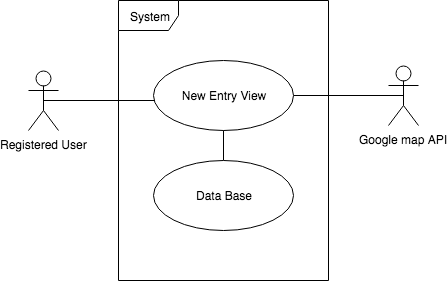


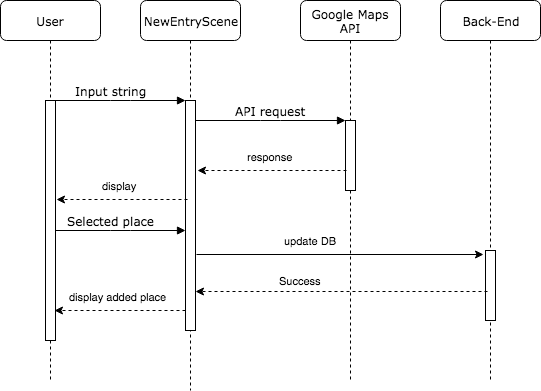
Class Diagram



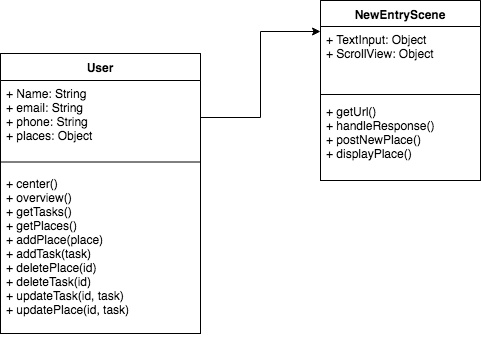
### #707 - User Story Name: Create New Entry Scene

Use Case Diagram

****Sequence Diagram

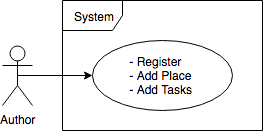


Class Diagram

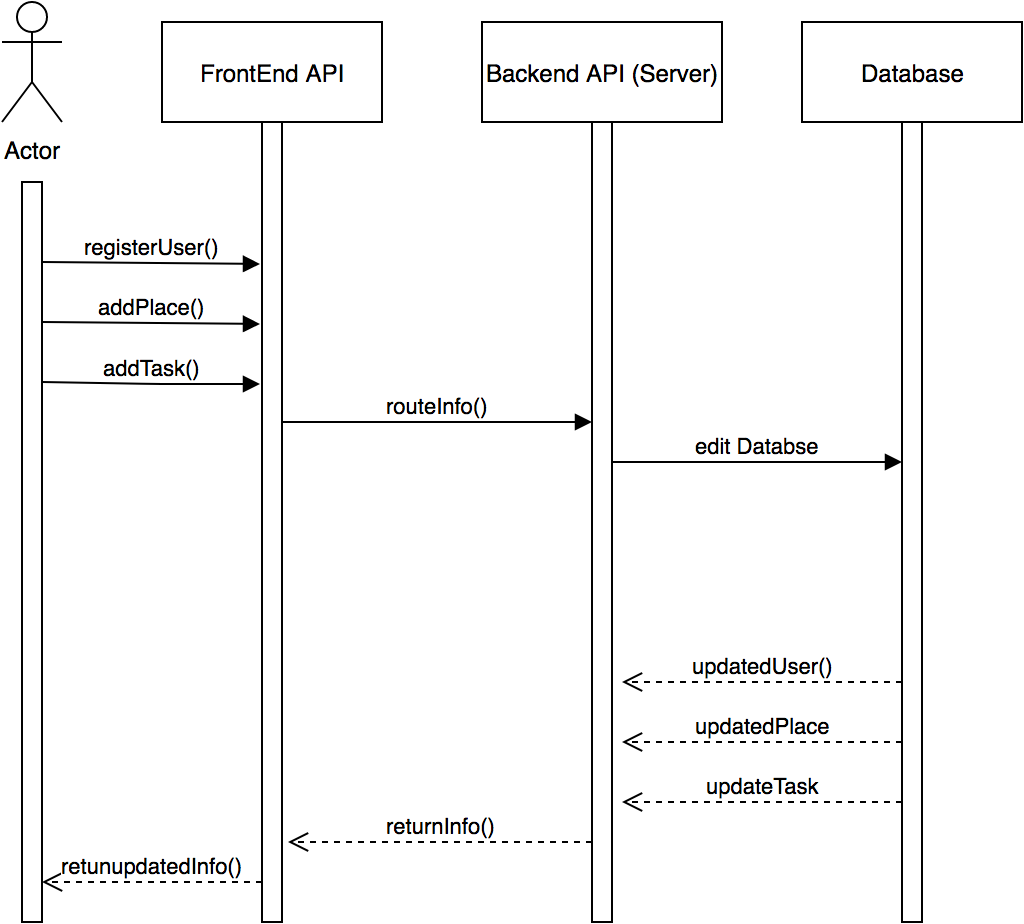


### #713 - User Story Name: Front End Restful API

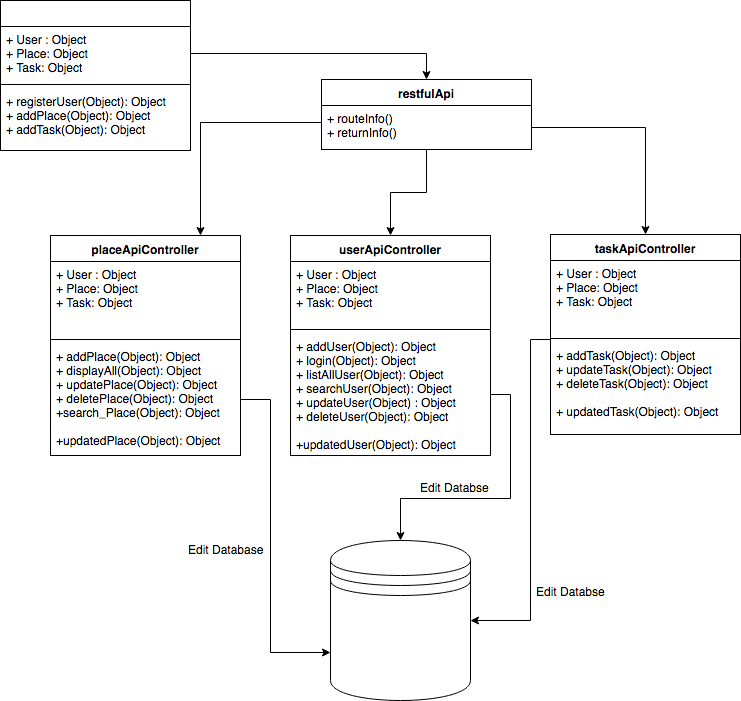
Use Case Diagram

[](https://www.draw.io/" \l "G1-qwB8YheWty-jBjPC0FkuwOGO7J7GcsO)

Sequence Diagram

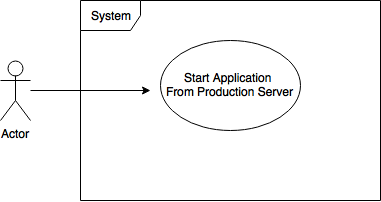
[](https://www.draw.io/?scale=2" \l "G1xlDZX5vVxW-GRKFcr_jVZBzdIRcB2wr3)

Class Diagram

[](https://www.draw.io/" \l "G1Ys_xcZGLi5C-85NN7nfBpVXL_LIx-mxt)

### #714 - User Story Name: Move Backend to Production Server

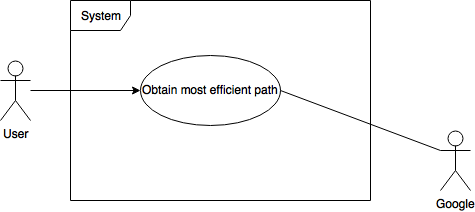
Use Case Diagram

[](https://www.draw.io/#G1wEl8Qy63QZL-aySNCNAFQb-dyNnIw7F0)

## 

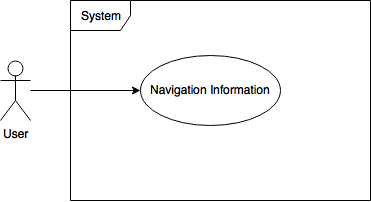
### #715 - User Story Name: Asynchronous Request of Possible routes

Use Case Diagram

[](https://www.draw.io/" \l "G1wncR35QbcMqEprmrvRVEKKHwpkF76KDN)

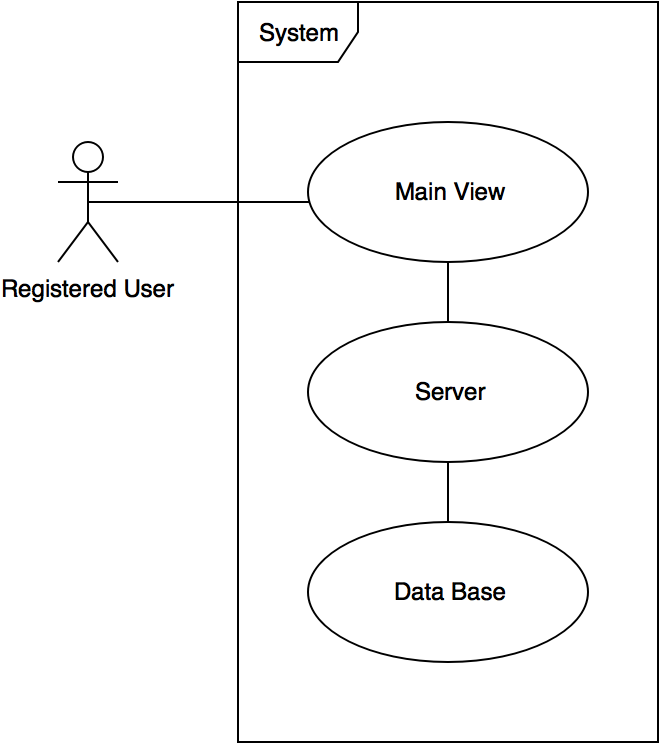
### #716 - User Story Name: Improve Visuals in the Map View

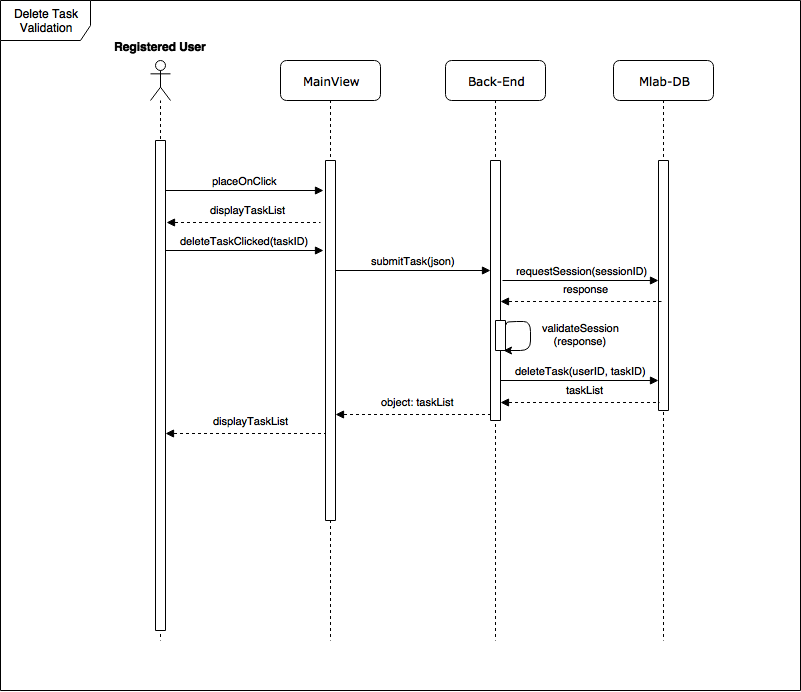
Use Case Diagram

[](https://www.draw.io/" \l "G1p_GskKTEQlvJw-Dvo5OufQ3KeuHI9GQ8)

### #717 - User Story Name: Delete task validation [Back-End]

Use Case Diagram

[****](https://www.draw.io/?scale=2#G16kmwZ7tN0hOcAXyromkoszv2mR6PfxuL)Sequence Diagram

[****](https://www.draw.io/#G1Rfk6K8gB7LpKD3VadzmoVpVKDORc9UiL)

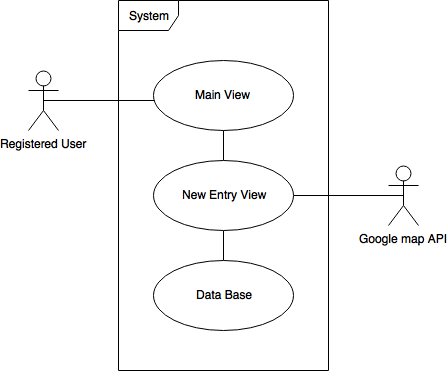
Class Diagram

[****](https://www.draw.io/#G1DT4i0XJVQscmImq_Ek3wh5WMqvw3Ybx2)

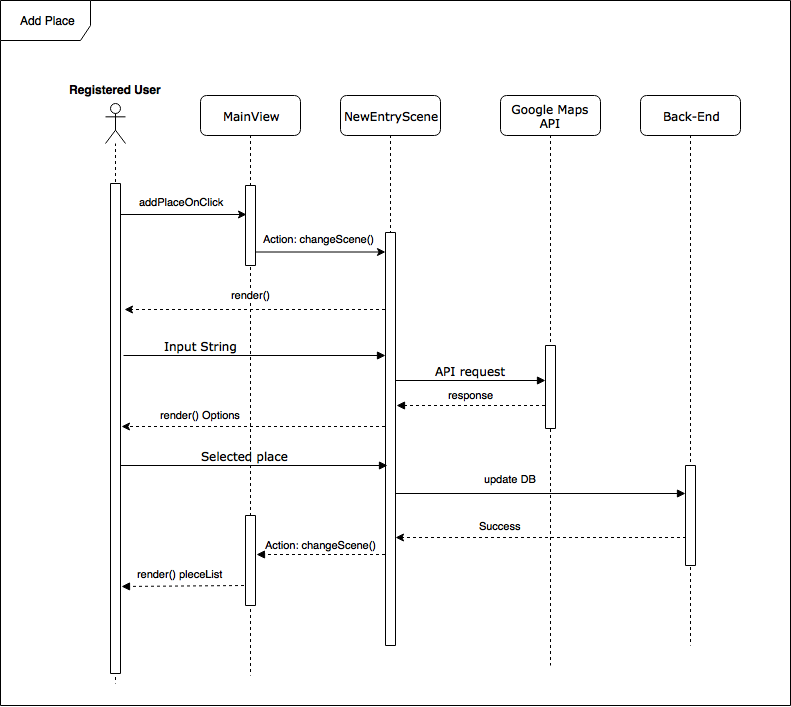
## 

### #718 - User Story Name: Add task validation [Back-End]

Use Case Diagram

[](https://www.draw.io/#G12j4998rb_dwNG2ocbi8ZiOYRygm_-twx)

Sequence Diagram

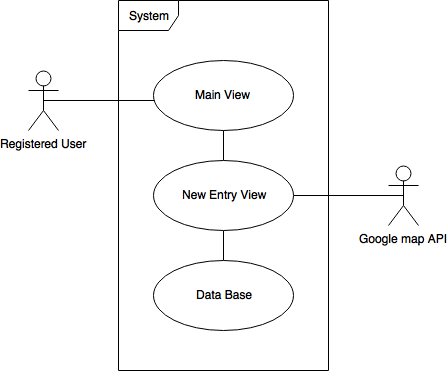
[****](https://www.draw.io/#G1_g7H1PZMPFAcw85oWUATyUPeQK_Y29Fq)

Class Diagram

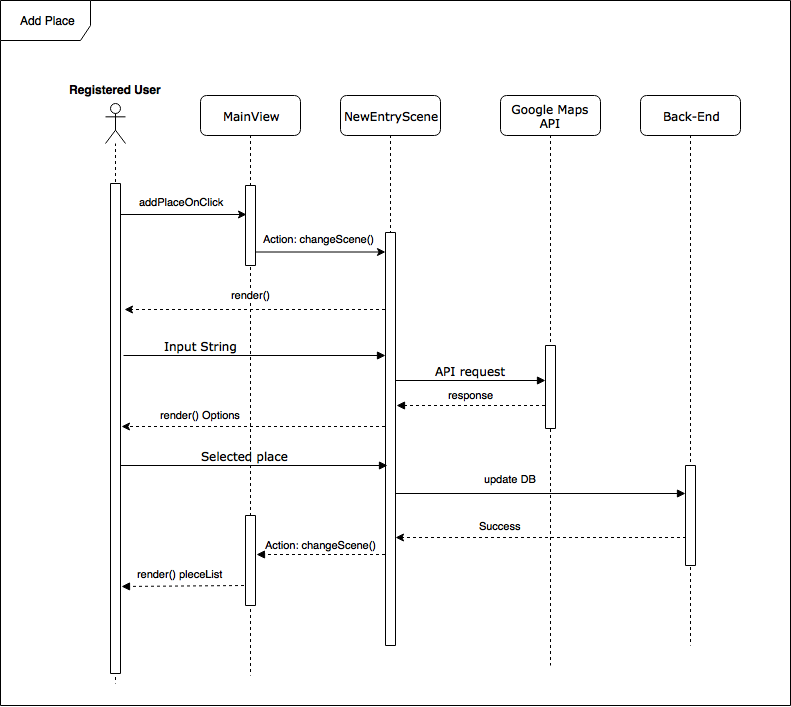
[****](https://www.draw.io/#G1DT4i0XJVQscmImq_Ek3wh5WMqvw3Ybx2)

### #723 - User Story Name: Add place validation [Back-End]

Use Case Diagram

[](https://www.draw.io/#G12j4998rb_dwNG2ocbi8ZiOYRygm_-twx)

Sequence Diagram

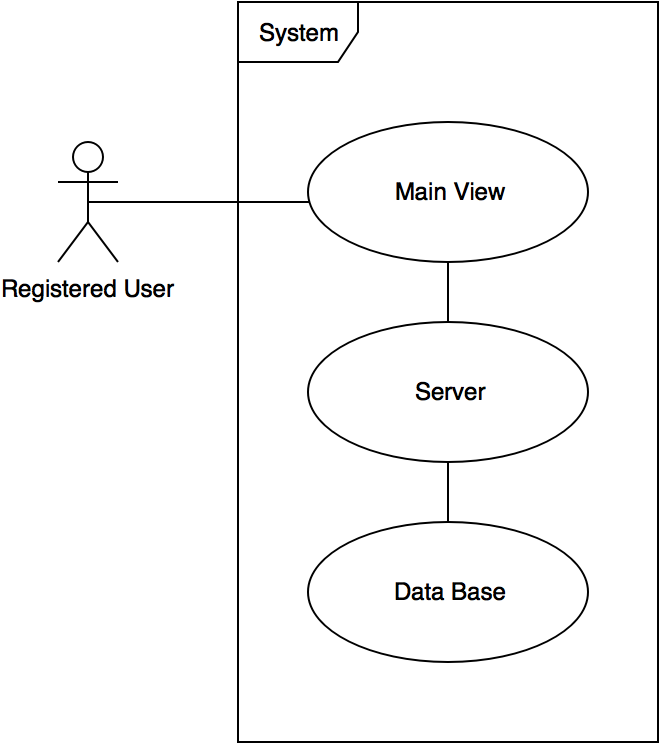
[****](https://www.draw.io/#G1_g7H1PZMPFAcw85oWUATyUPeQK_Y29Fq)

Class Diagram

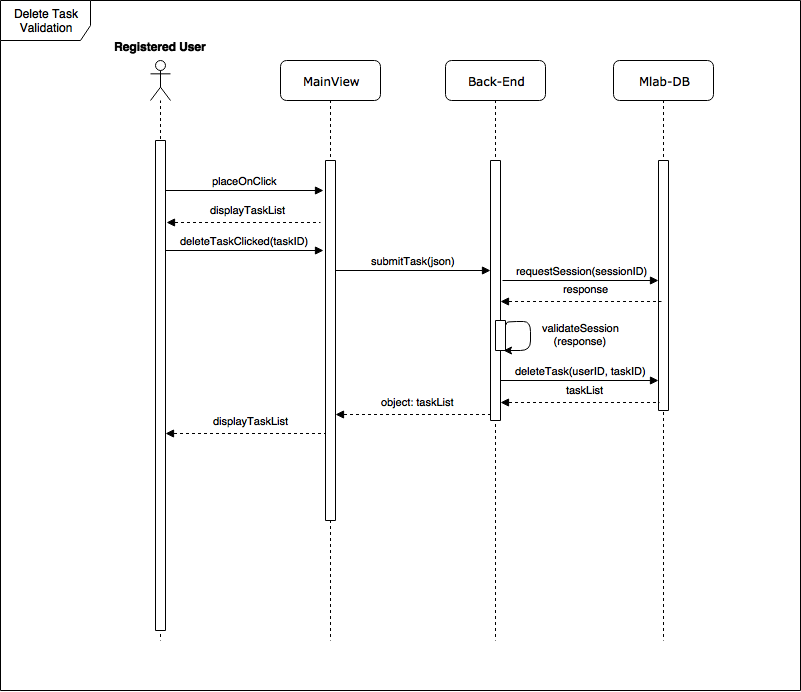
**[](https://www.draw.io/" \l "G1DT4i0XJVQscmImq_Ek3wh5WMqvw3Ybx2)**

### #724 - User Story Name: Delete place validation [Back-End]

Use Case Diagram

**[](https://www.draw.io/?scale=2" \l "G16kmwZ7tN0hOcAXyromkoszv2mR6PfxuL)**

Sequence Diagram

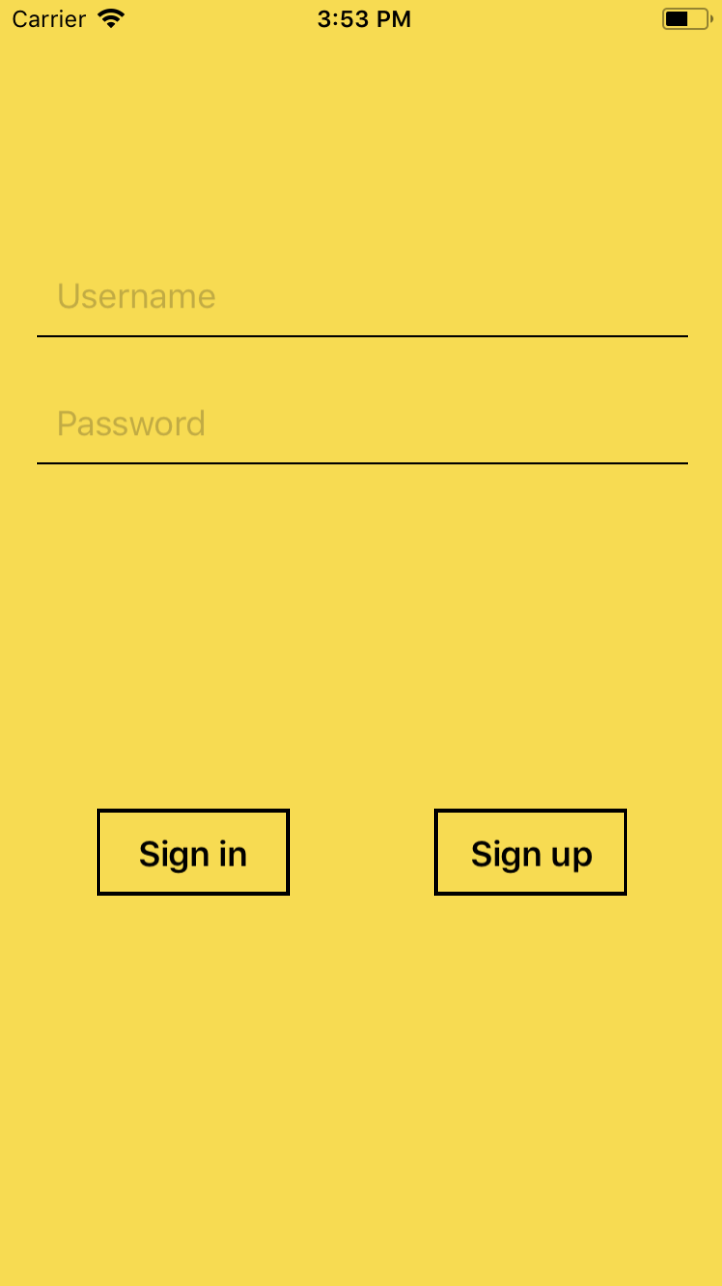
[****](https://www.draw.io/#G1Rfk6K8gB7LpKD3VadzmoVpVKDORc9UiL)

Class Diagram

**[](https://www.draw.io/" \l "G1DT4i0XJVQscmImq_Ek3wh5WMqvw3Ybx2)**

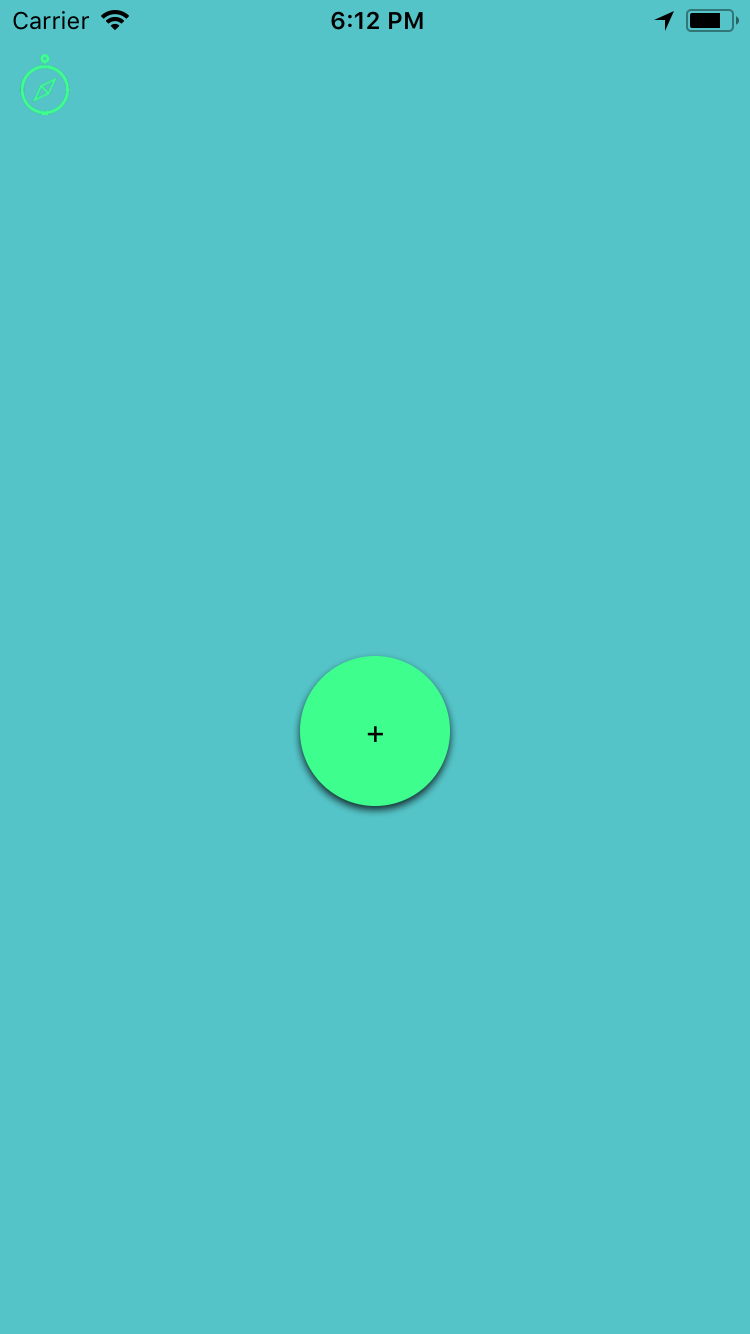
## Appendix B – User Interface Design

### Sign In/Sign Up Screen

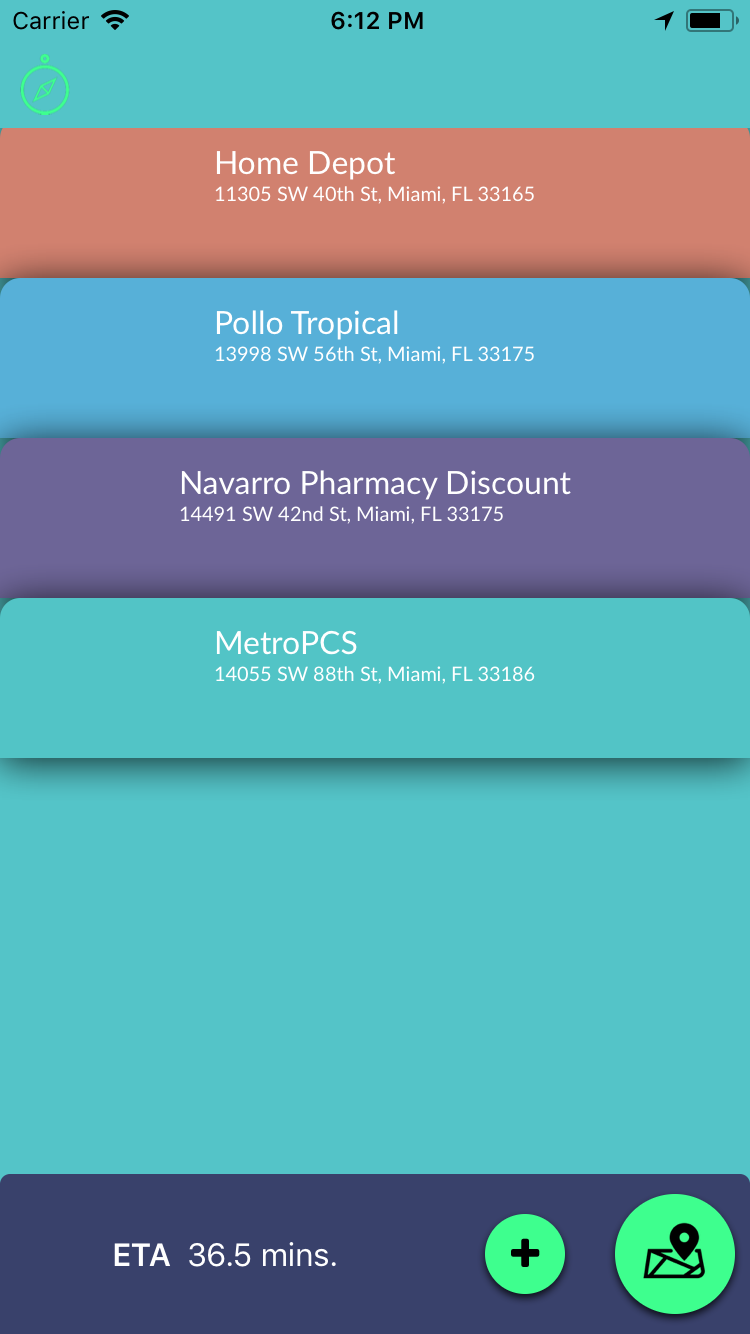


### 

### Itinerary List Screen (Empty)

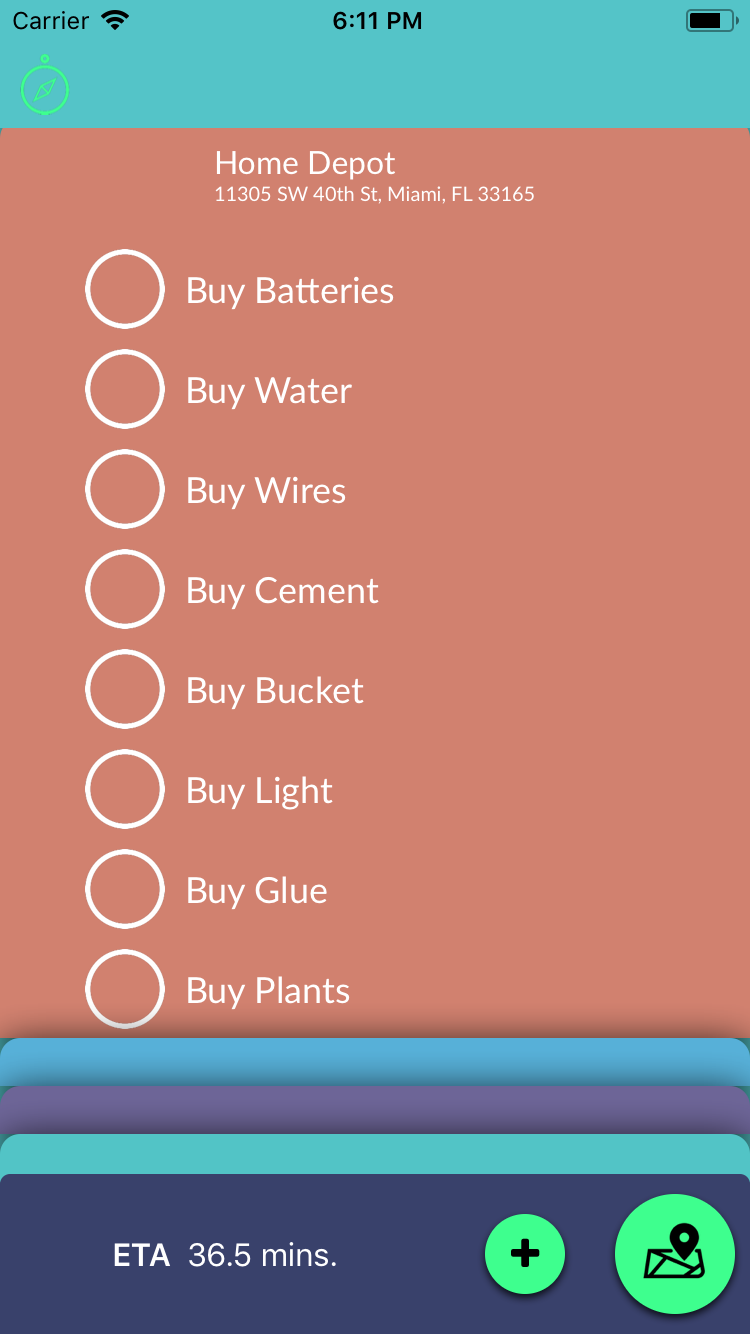


### Itinerary List Screen (filled)



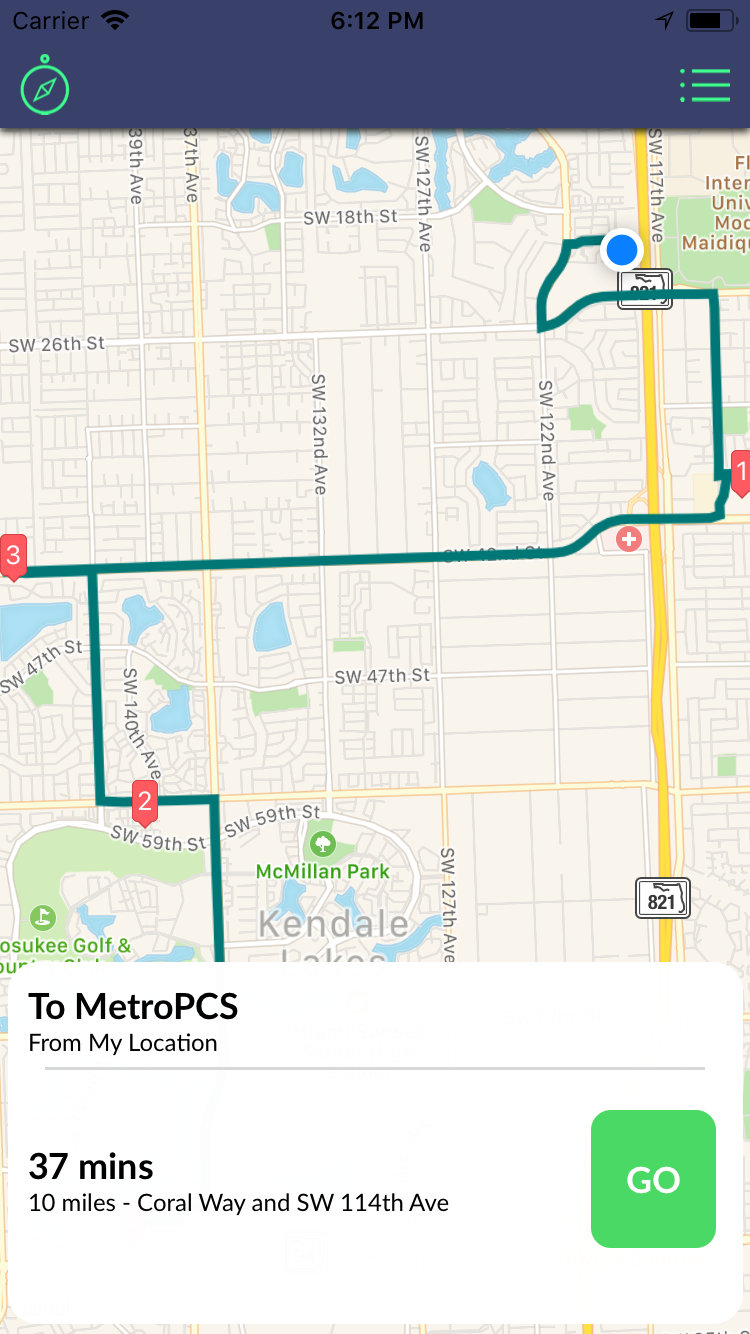
### 

### Itinerary List Screen (places expanded)



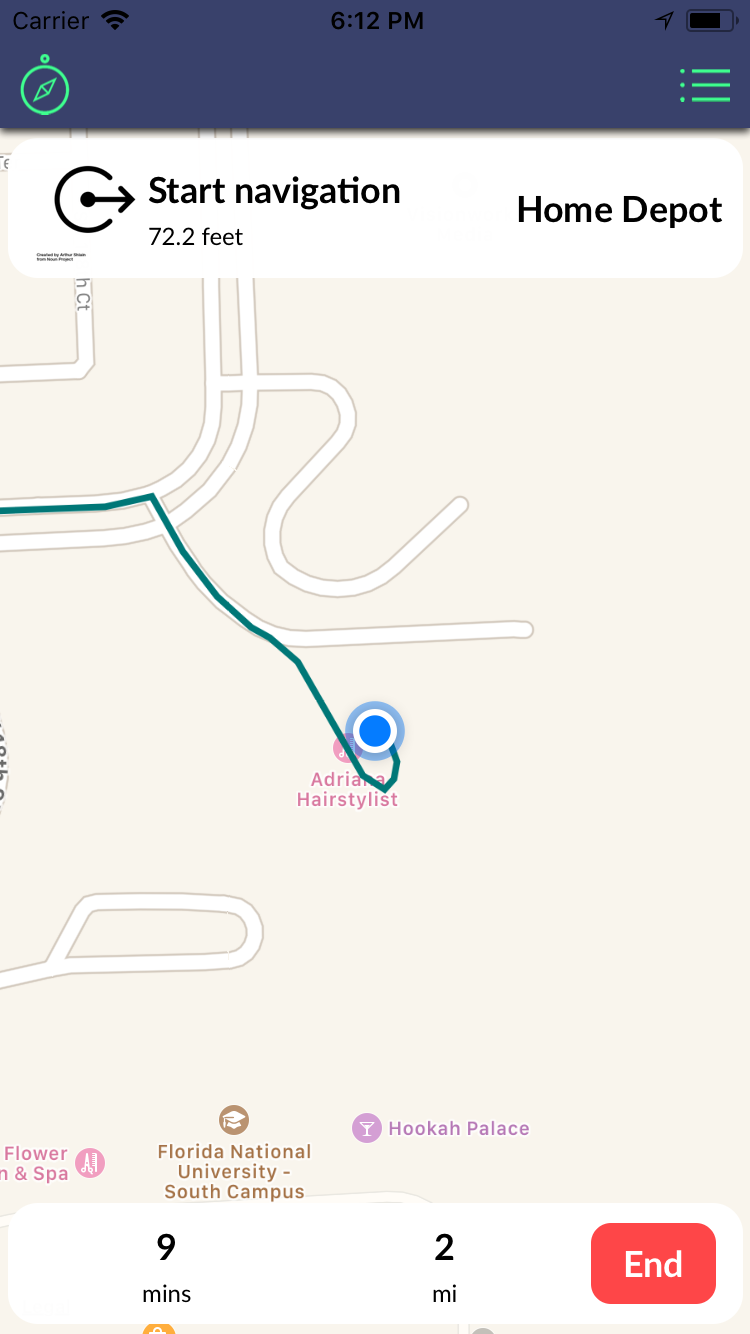
### 

### Map overview Screen (Navigation NOT started)



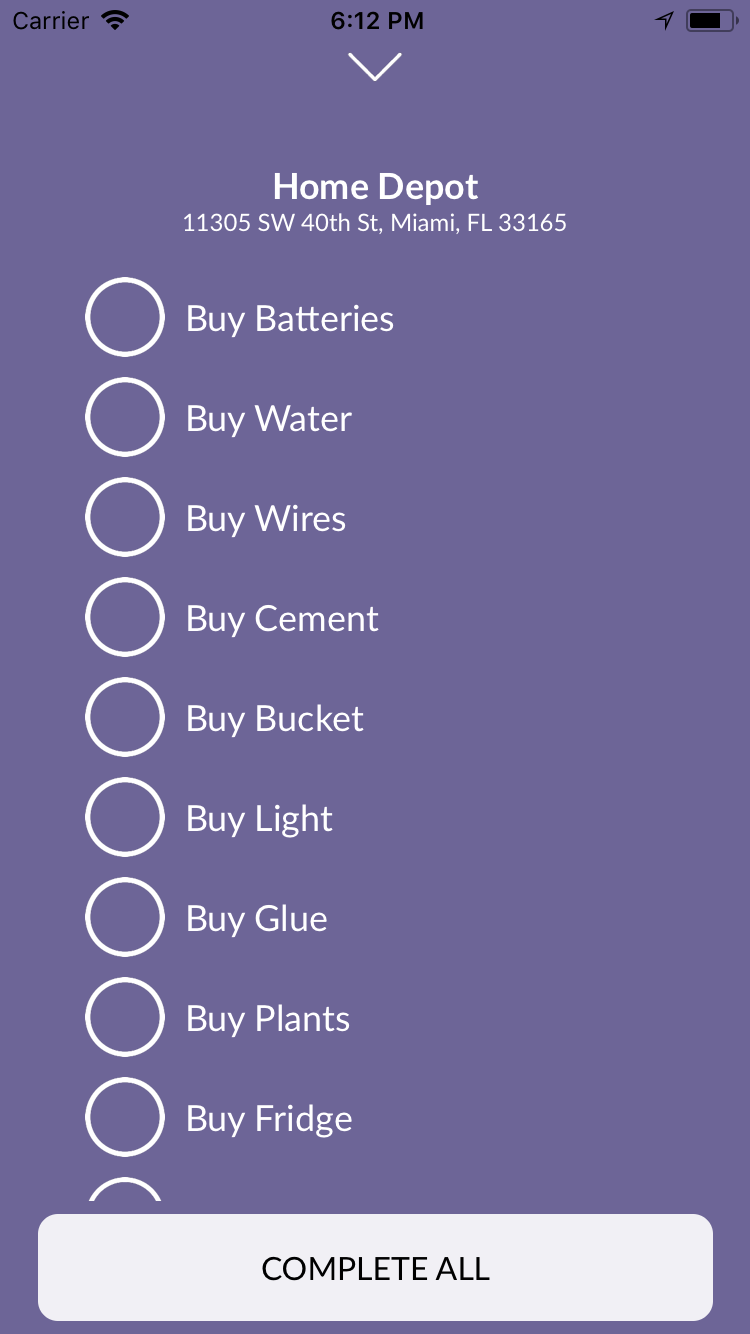
### 

### Map center Screen (Navigation Started)



### 

### Task List Screen (When Arrived to place)



## 

## Appendix C – Sprint Review Reports

### Sprint 2

Attendees:

* Salvador Ricardo, Manuel Garcia, Daniel Gonzales, Monique Ross

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

* #680 Register User [Backend]
* #690 Scene Basic Transitions
* #695 Update User [ Backend]
* #697 Delete User [Backed]
* #703 Center Map
* #704 Overview Map
* #706 Connect to google maps API to autocomplete place search [front-end]
* #707 Create New Entry Scene [front-end]

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

* No User Stories were rejected

### Sprint 3

Attendees:

* Salvador Ricardo, Manuel Garcia, Daniel Gonzales, Monique Ross

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

* #666 Setup Sign Up
* #669 Add Places
* #670 Add tasks to places
* #671 Show Itinerary and Total Time
* #673 Setup Navigation Information
* #681 Add Places to User [Backend]
* #683 Add Task to Places [Backend]
* #685 Delete Places Manually [Backend]
* #687 Update Places [Backend]
* #688 Delete Task [Backend]
* #684 Delete Places [Backend]

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

* No User Stories were rejected

### Sprint 4

Attendees:

* Salvador Ricardo, Manuel Garcia, Daniel Gonzales, Monique Ross

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

* #675 Add New Place while Navigating
* #714 Move Backend to Production Server
* #715 Asynchronous request of possible routes

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

* #713 Front End RestFul Api
* #679 Track the time spent in places
* #676 Update Route on Map View if Delays

### Sprint 5

Attendees:

* Salvador Ricardo, Manuel Garcia, Daniel Gonzales, Monique Ross

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

* # 713 Front-end restful API.
* # 679 Track time spent in places.
* # 676 Update route on map view if delays.
* # 716 Improve visuals in the map view.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

* None

### Sprint 6

Attendees:

* Salvador Ricardo, Manuel Garcia, Daniel Gonzales, Monique Ross

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

* # 721 Senior Project Video - Salvador
* # 722 Senior Project Poster - Salvador
* # 725 Senior Project Poster - Daniel
* # 718 Senior Project Video – Manuel

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

* None

## Appendix D – Sprint Retrospective Reports

### Sprint 2

Attendees:

* Salvador Ricardo, Manuel Garcia, Daniel Gonzales, Monique Ross

What went wrong?

* Did we do a good job estimating our team's velocity?
  + We did a good job, it could have been better, but it was pretty good. A lot of learning required to start working.
* Did we do a good job estimating the points (time required) for each user story?
  + We underestimated the time required for each thing. Sometimes it took more than the expected amount.
* Did each team member work as scheduled?
  + Yes.

What went right?

* Coupling as a team was something that worked really well during this sprint.

How to address the issues in the next sprint?

* How to improve the process?
  + Study more about the user stories technologies needed to estimate better velocity.
* How to improve the product?
  + We are planning on finding state-of-the-art products using the same technology and look at how they structure their products to improve ours.

### Sprint 3

Attendees:

* Salvador Ricardo, Manuel Garcia, Daniel Gonzales, Monique Ross

What went wrong?

* Did we do a good job estimating our team's velocity?
  + We did a good a good job managing our team velocity although we underestimated the time some tasks would take.
* Did we do a good job estimating the points (time required) for each user story?
  + Some of the stories took more time than the estimated time
* Did each team member work as scheduled?
  + All the team member worked as scheduled, and although we took a little more time, every team member finished their User Stories.

What went right?

* We finished all the User Stories assigned to the Sprint 3 and didn’t moved any back to the backlog.

How to address the issues in the next sprint?

* How to improve the process?
  + Now that we have a better understanding of the framework we can estimate better the amount of time that the tasks would take and be more thorough.
* How to improve the product?
  + Since we are using a framework that is not native to the language of the platforms, we can learn some swift and help improve the code by adding native code.

### Sprint 4

Attendees:

* Salvador Ricardo, Manuel Garcia, Daniel Gonzales, Monique Ross

What went wrong?

* Did we do a good job estimating our team's velocity?
  + We did do a good job estimating the team’s velocity, but some User Stories where longer than 1 sprint’s worth.
* Did we do a good job estimating the points (time required) for each user story?
  + We did a good job estimating the points for each user story
* Did each team member work as scheduled?
  + Yes, all the team members worked as scheduled.

What went right?

* We made progress in the app, both in the frontend and backend. We are certainly on time to finish the application.

How to address the issues in the next sprint?

* How to improve the process?
  + Try to better estimate the points for next sprint.
* How to improve the product?
  + To improve the product, we will subject the application to extensive testing.

### Sprint 5

Attendees:

* Salvador Ricardo, Manuel Garcia, Daniel Gonzales, Monique Ross

What went wrong?

* Did we do a good job estimating our team's velocity?
  + At this point of the semester the team is adjusting better to the phase and it is calculating better the time that each story would take since the knowledge of the framework has increased.
* Did we do a good job estimating the points (time required) for each user story?
  + The team is doing a better job estimating the points, but we still have those stories that we under rate.
* Did each team member work as scheduled?
  + Every team member worked according to the schedule and was on time for each of the scheduled meetings.

What went right?

* The synergy among the member of the team.

How to address the issues in the next sprint?

* How to improve the process?
  + A little bit of communication would improve the process since this sprint we had some conflicts on some commits.
* How to improve the product?
  + At this point the best way to improve the product would be to release a beta so people can try it out and give some feedback

### Sprint 6

Attendees:

* Salvador Ricardo, Manuel Garcia, Daniel Gonzales, Monique Ross

What went wrong?

* Did we do a good job estimating our team's velocity?
* We did a good job estimating our team velocity. Being the end of the semester, we successfully did all the tasks assigned to the sprint.
* Did we do a good job estimating the points (time required) for each user story?
  + We think we did a good job estimating the points, but then again we were a couple of hours off by underestimating the time, but much better than at the beginning.
* Did each team member work as scheduled?
  + Yes, every team member worked according to the schedule and was on time for each of the scheduled meetings.

What went right?

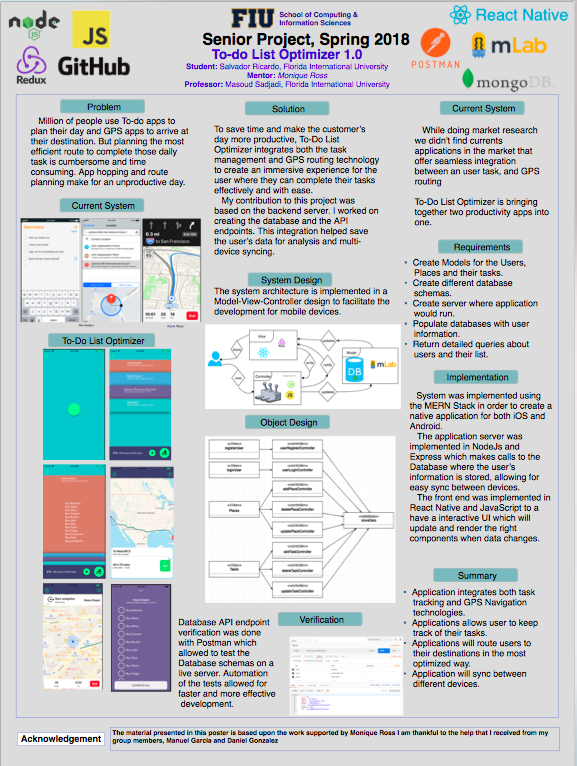
* We finished all the tasks in the sprint.

## Appendix F - User Manuals, Installation/Maintenance Document, Shortcomings/Wish list Document and other documents

### Videos

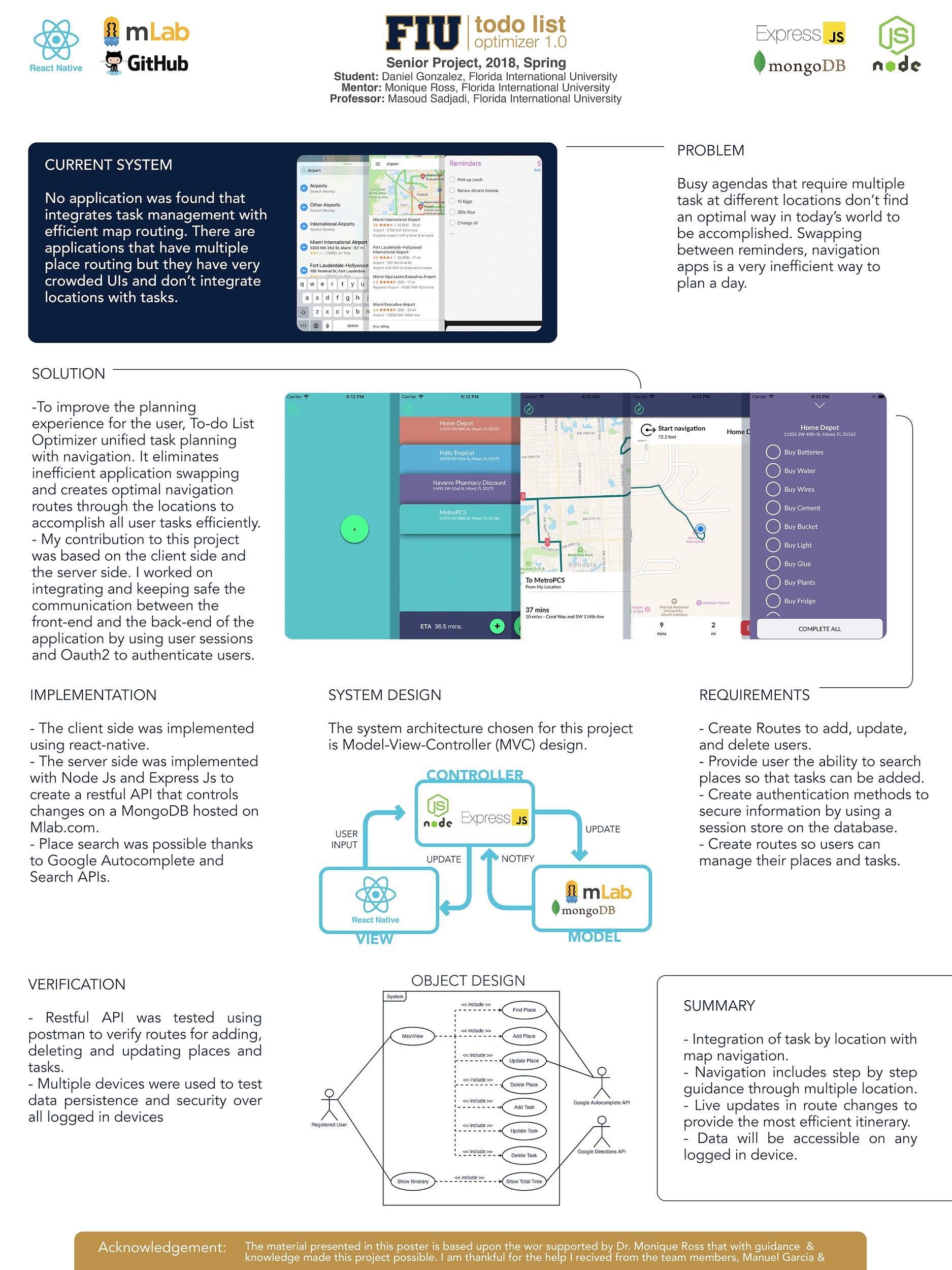
* Intro Video

# Posters

****

## Salvador Ricardo

## ../../../../../Desktop/Screen%20Shot%202018-04-25%20at%204.13.13%2Manuel Garcia



## Daniel Gonzalez