



1

Socialize GPS Application

Team 02
Dylan Kehres & Michael Schott

Customer Requirements

Objective:

Create a software application that integrates GPS navigation with social media aspects that promotes usage through gamification.

Functions :

- Navigation
- Map Viewing
- Finding locations
- View Points of Interest (Gamification Aspect)
- Leave Feedback at Points of Interest
- Edit Account Settings
- Receive Help
- Show Location History of User
- Allow User to Save Places They Like Most
- View Leaderboards (Gamification Aspect)
- Add/Remove Friends
- Allow Friendly Competition
- Show User's Gamer Score (Gamification Aspect)
- Show User's Achievements (Gamification Aspect)
- View Friends' Scores/Achievements (Gamification Aspect)

Supporting Systems of the Software:

The software application should be supported on mobile operating systems, beginning with iOS and Android and, should funds permit, other mobile operating systems as well. There is no need for this application to be integrated with PC operating systems such as Windows. The application should have internet capabilities and must have access to the User's location in order to operate. It is to be written in JavaScript.

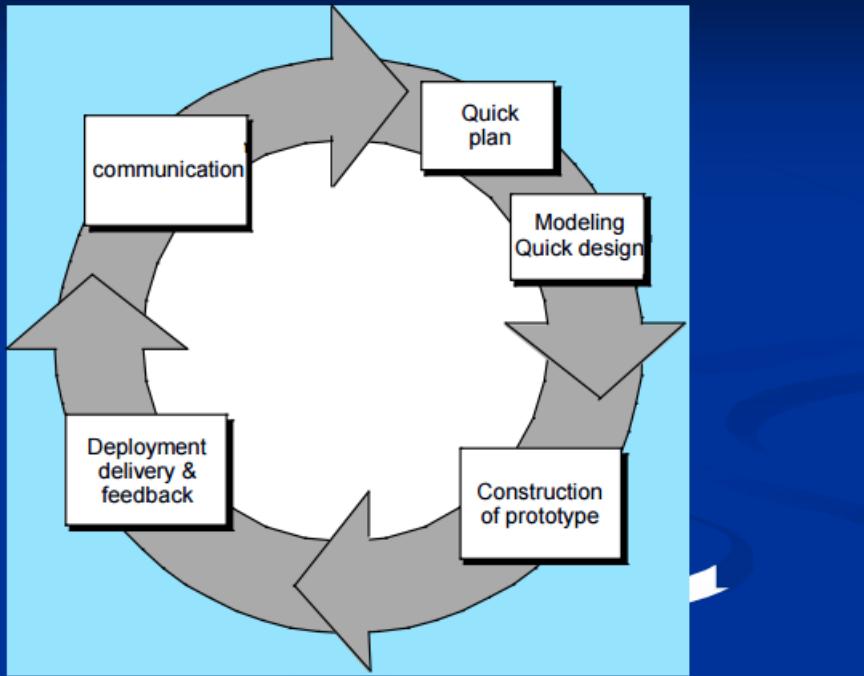
System Constraints:

1. The application will need internet connectivity to run smoothly. This will cause certain regions to be inaccessible to certain players depending on their mobile carrier due to the varying coverages of each phone company. For example, one Point of Interest may be accessible to Users using Verizon or Sprint as their mobile providers but inaccessible to AT&T users. This could create the potential for an unfair game.
2. The Mobile Device must has a GPS unit.
3. The Software System should take between 112-124 man months to complete.
4. The budget for the development of the system is \$1,000,000.

Software Engineering Process Model

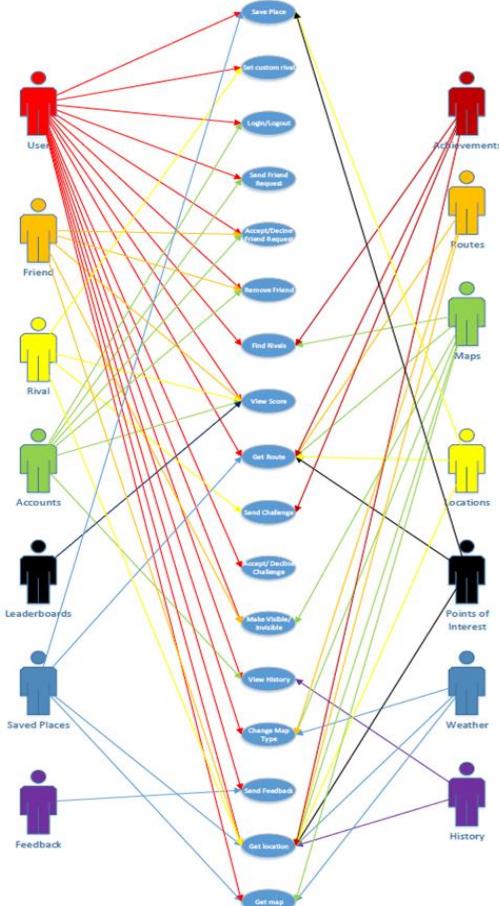


Evolutionary Process Flow: A Prototyping Model



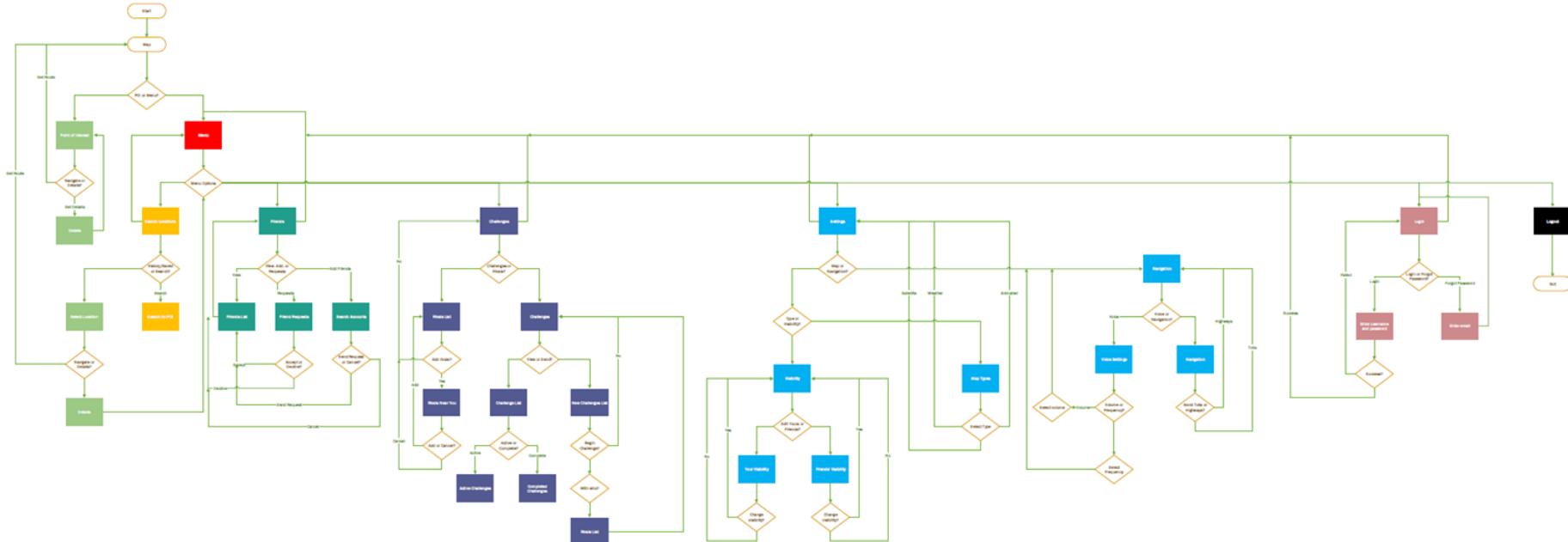
- Our software is blending social media and GPS navigation with a gaming aspect; therefore it is critical that the User is enjoying the product.
- Users will be able to provide us with important feedback on aspects of our application such as the GUI; which is critical as our application must be user friendly.
- Users will also be able to let us know what features need to be added in future updates of the application that will enhance the User's experience.

Use Case Diagram



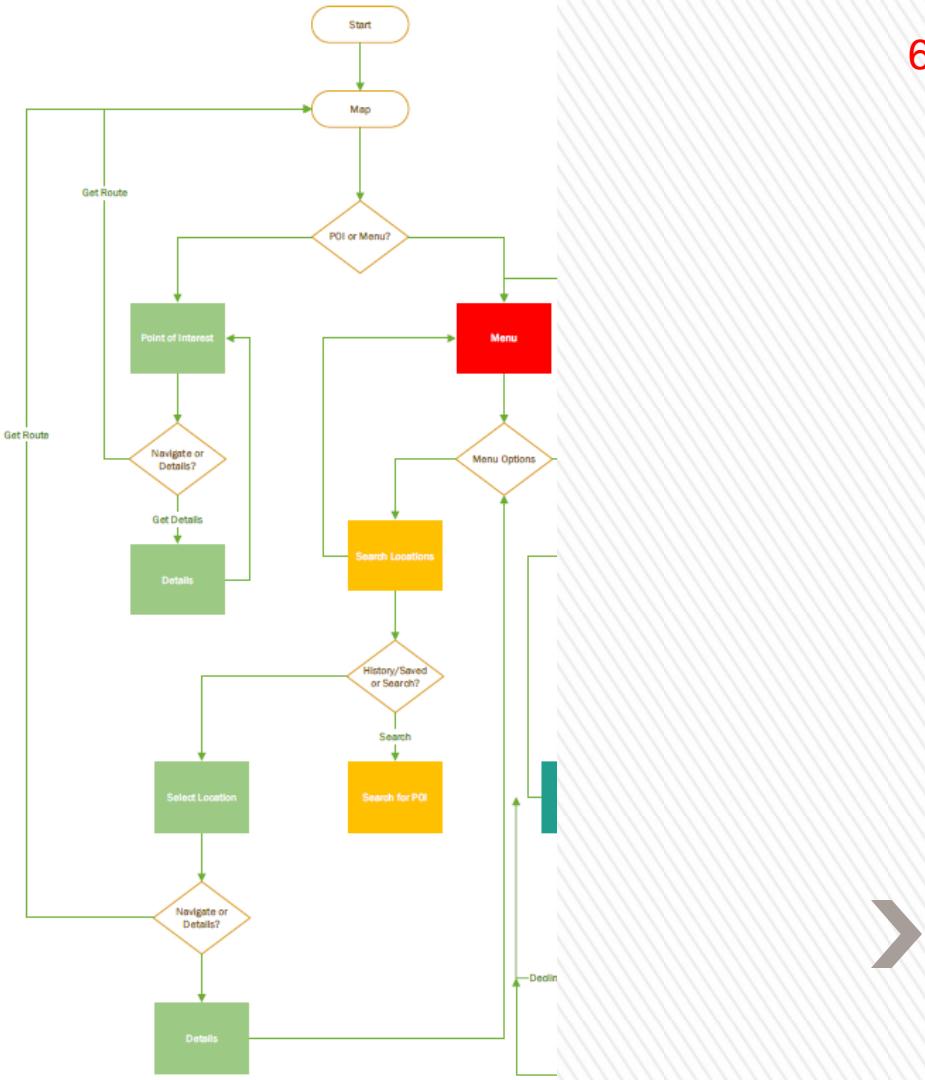
Activity Diagram

5



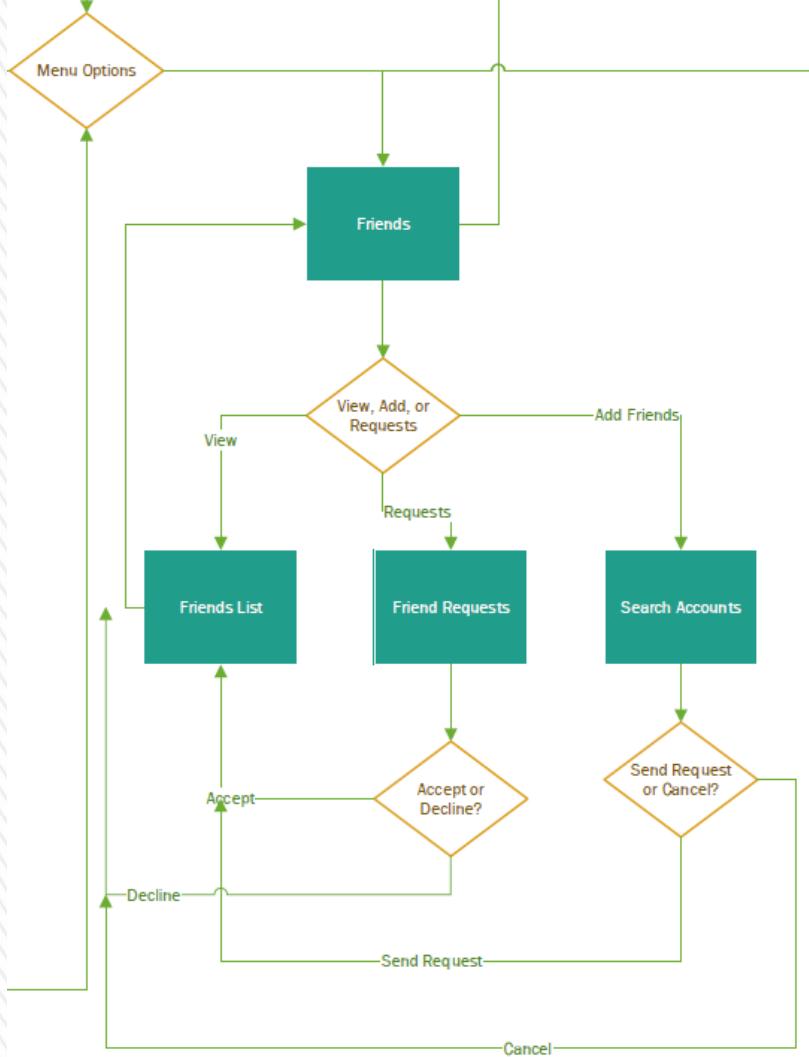
Activity Diagram

Part 1



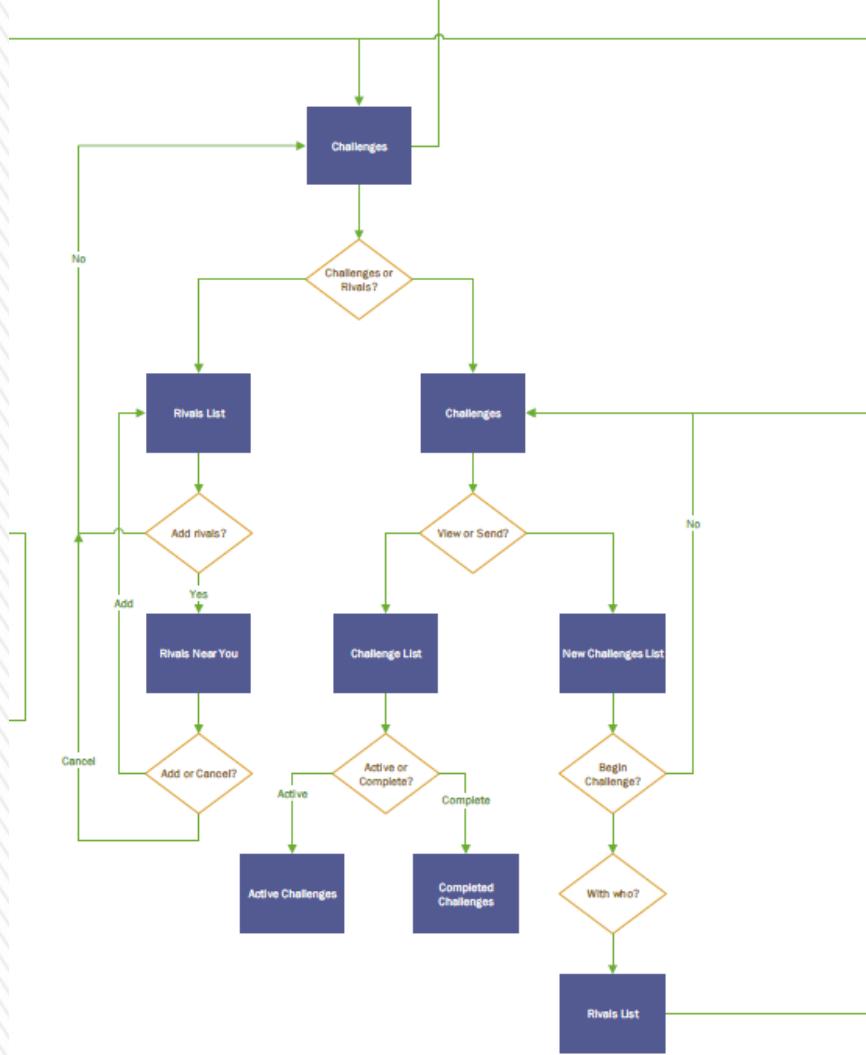
Activity Diagram

Part 2

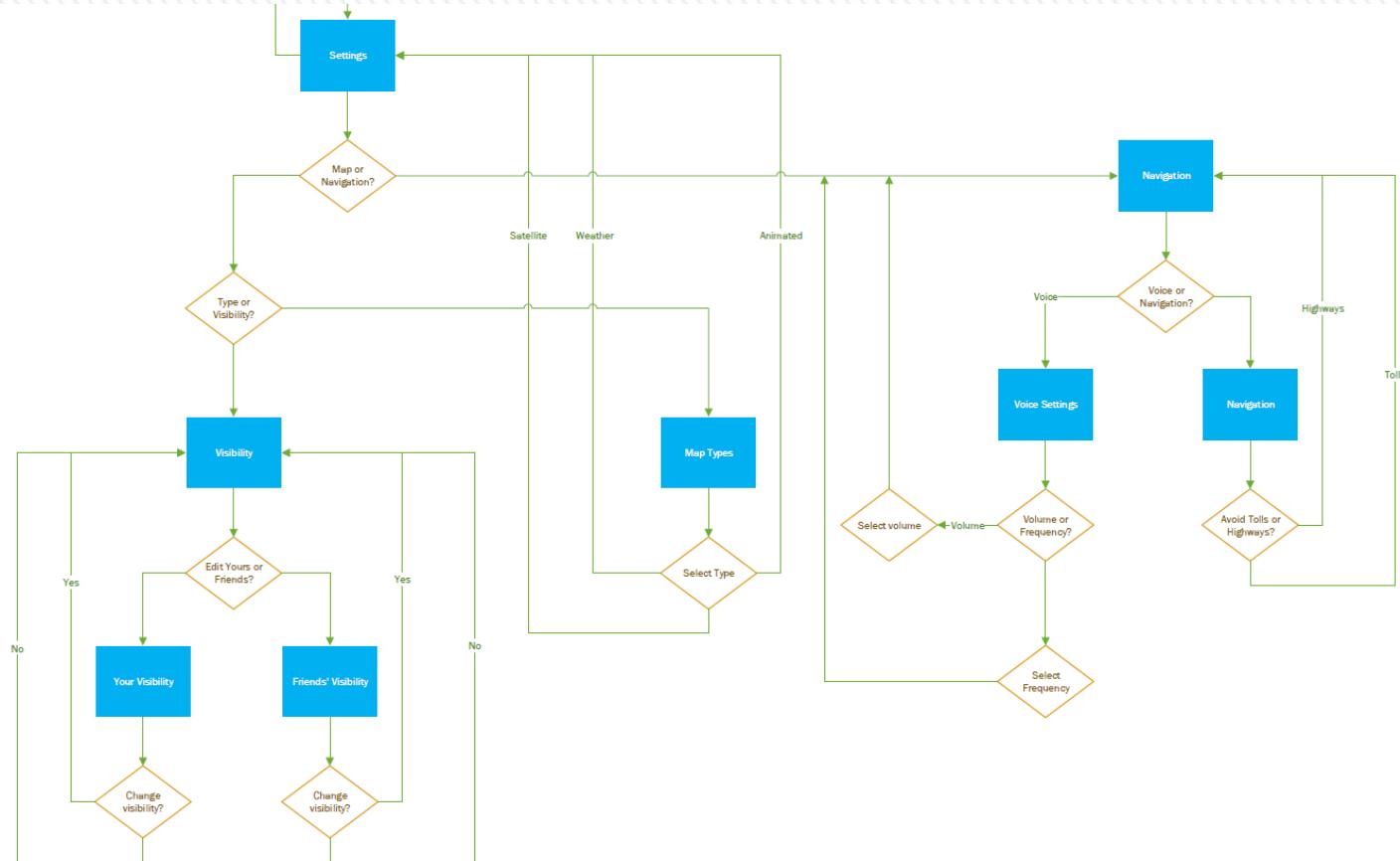


Activity Diagram

Part 3

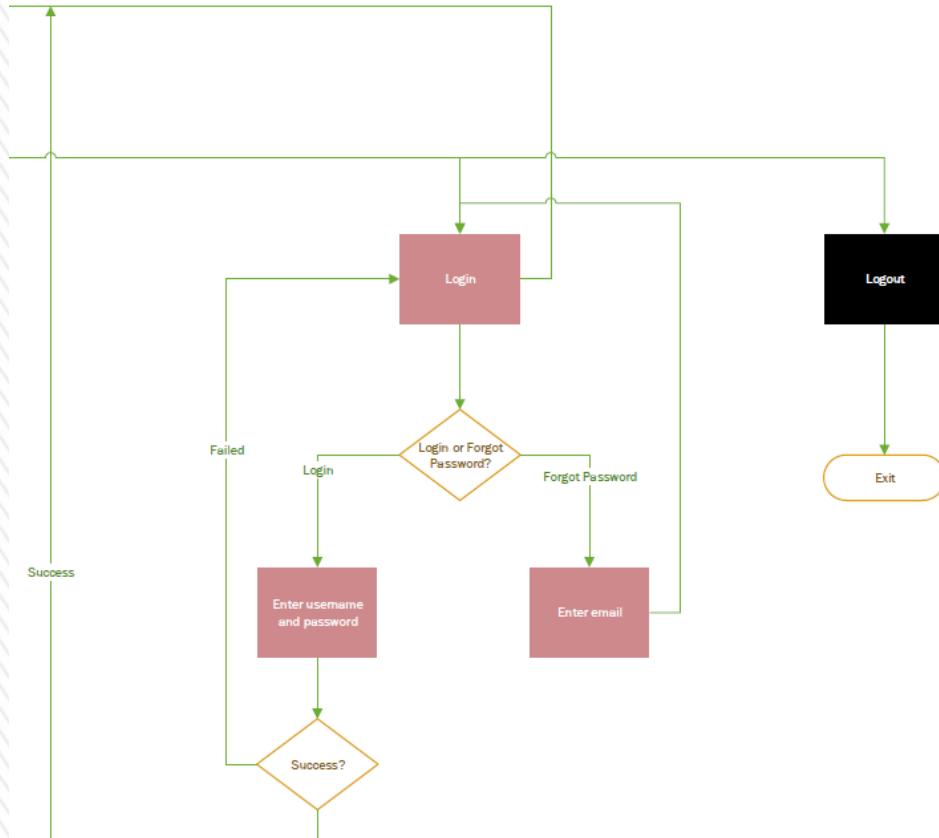


Activity Diagram Part 4



Activity Diagram

Part 5

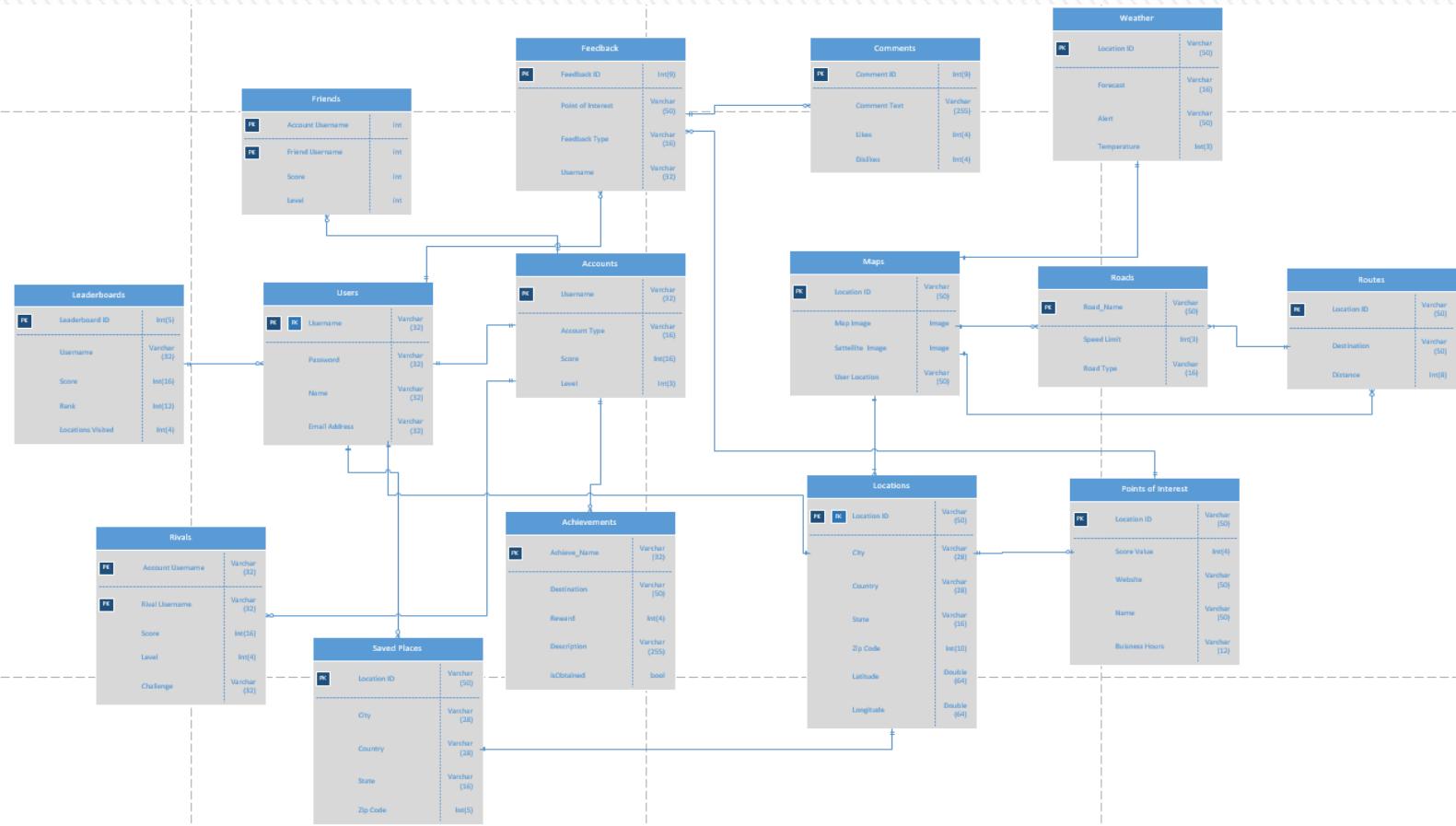


Class Object Diagram

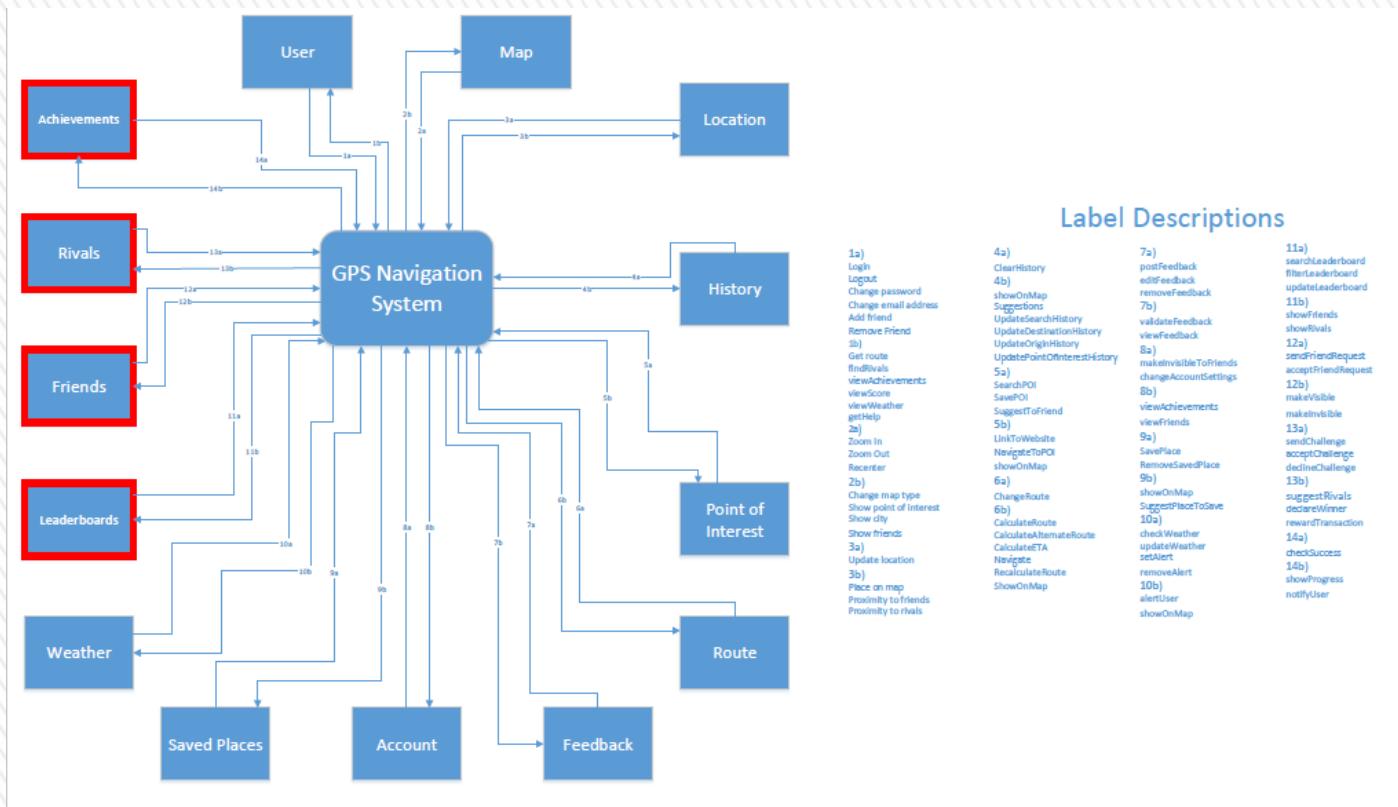
Users	Maps	Locations	Feedback	History	Points of Interest	Routes	
username password name email_address dateOfBirth dateJoined home Login Logout Change password Change email address Get route addFriend removeFriend findFriends viewAchievements viewScore viewWeather getHelp	MapImage SatelliteImage Roads PointOfInterest Borders Cities UserLocation FriendLocation Zoom In Zoom Out Change map type Show point of interest Show city Show friends Recenter	image (2GB) image (2GB) Location (2MB) Location (2MB) Location (2MB) Location (2MB) Location (2MB) Location (2MB)	Latitude Longitude Country State City Zip_Code Update location Place on map Proximity to friends Proximity to rivals	FeedbackType Username FeedbackTitle Comments Rating dateAdded postFeedback editFeedback viewFeedback removeFeedback validateFeedback	SearchHistory DestinationHistory OriginHistory PointOfInterestHistory numTimesV DatesVisited Suggestions UpdateSearchHistory UpdateDestinationHistory UpdateOriginHistory UpdatePointOfInterestHistory ClearHistory showOnMap	Name LocationType BusinessHours RelatedPOIs Website POI_Location ScoreValue SearchPOI NavigateToPOI SavePOI LinkToWebsite SuggestToFriend showOnMap	Roads Exits Turns Lanes Destination EstimatedTimeOfArrival RouteOptions CalculateRoute CalculateAlternateRoute CalculateETA ChangeRoute Navigate RecalculateRoute ShowOnMap
Achievements	Accounts	Saved Places	Weather	Leaderboards	Friends	Rivals	
name destination reward description dateAchieved progress checkSuccess showProgress notifyUser	AccountType VisibleToFriends FriendsList RivalsList Score Level Achievements dateJoined numTimesLoggedIn viewFriends makeVisibleToFriends viewAchievements changeAccountSettings	String (16) Boolean String[] (30000) String[] (30000) int (16) int (4) String[] (30000) Date int(10)	Name PlaceLocation SimilarPlaces Suggestions VisitFrequency VisitDates SavePlace RemoveSavedPlace SuggestPlaceToSave showOnMap	weatherLocation weatherType weatherDuration weatherSeverity alerts checkWeather updateWeather showOnMap setAlert removeAlert alertUser	username score level rank locationsVisited searchLeaderboard filterLeaderboard updateLeaderboard showFriends showRivals	username name visible friendRequest dateFriendship proximity DateOfBirth sendFriendRequest acceptFriendRequest makeVisible makeInvisible	username challenge reward difficulty proximity sendChallenge acceptChallenge declineChallenge declareWinner suggestRivals rewardTransaction



Optional ERD



Context Dataflow Diagram

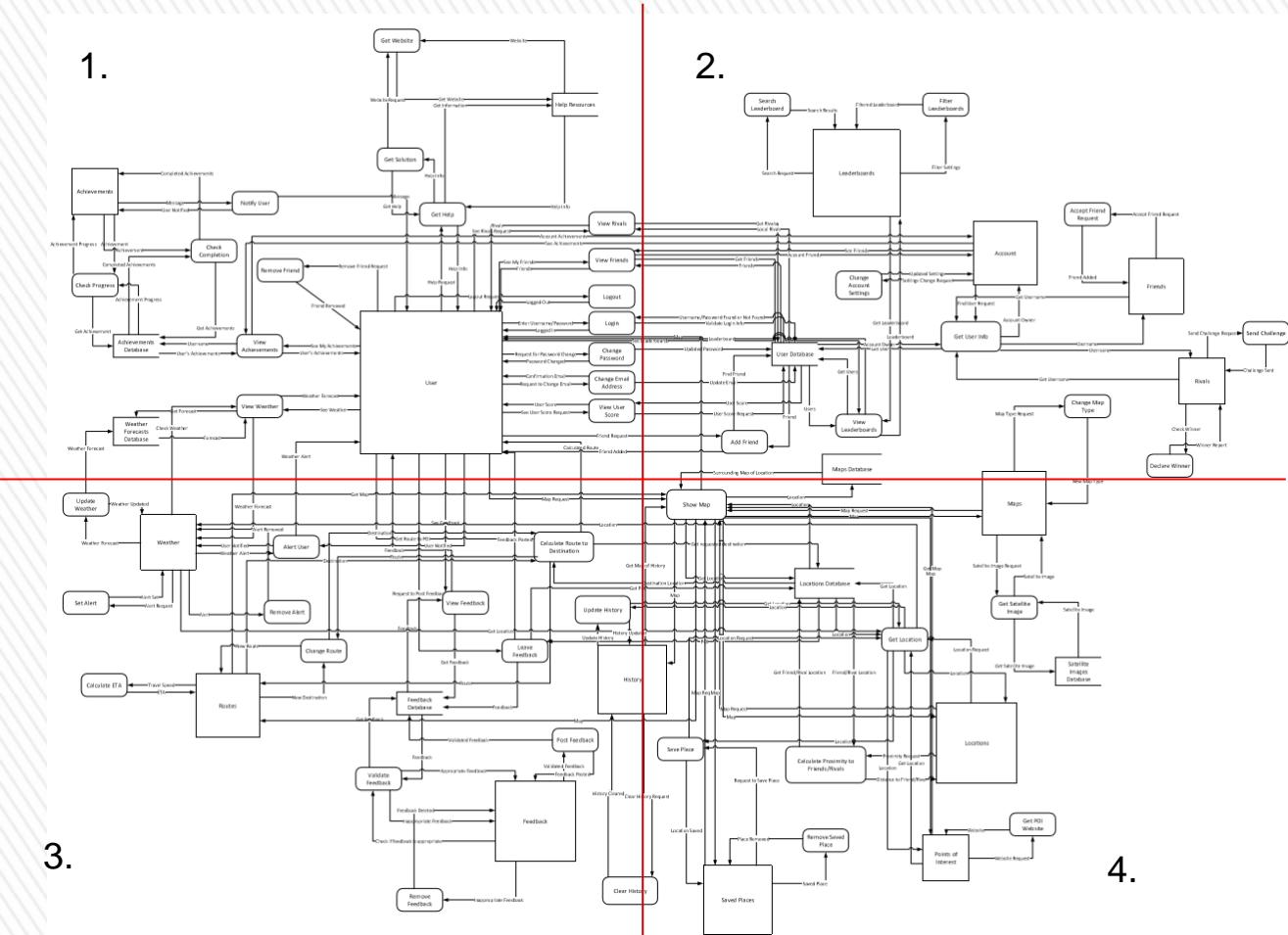


Context DFD Label Descriptions

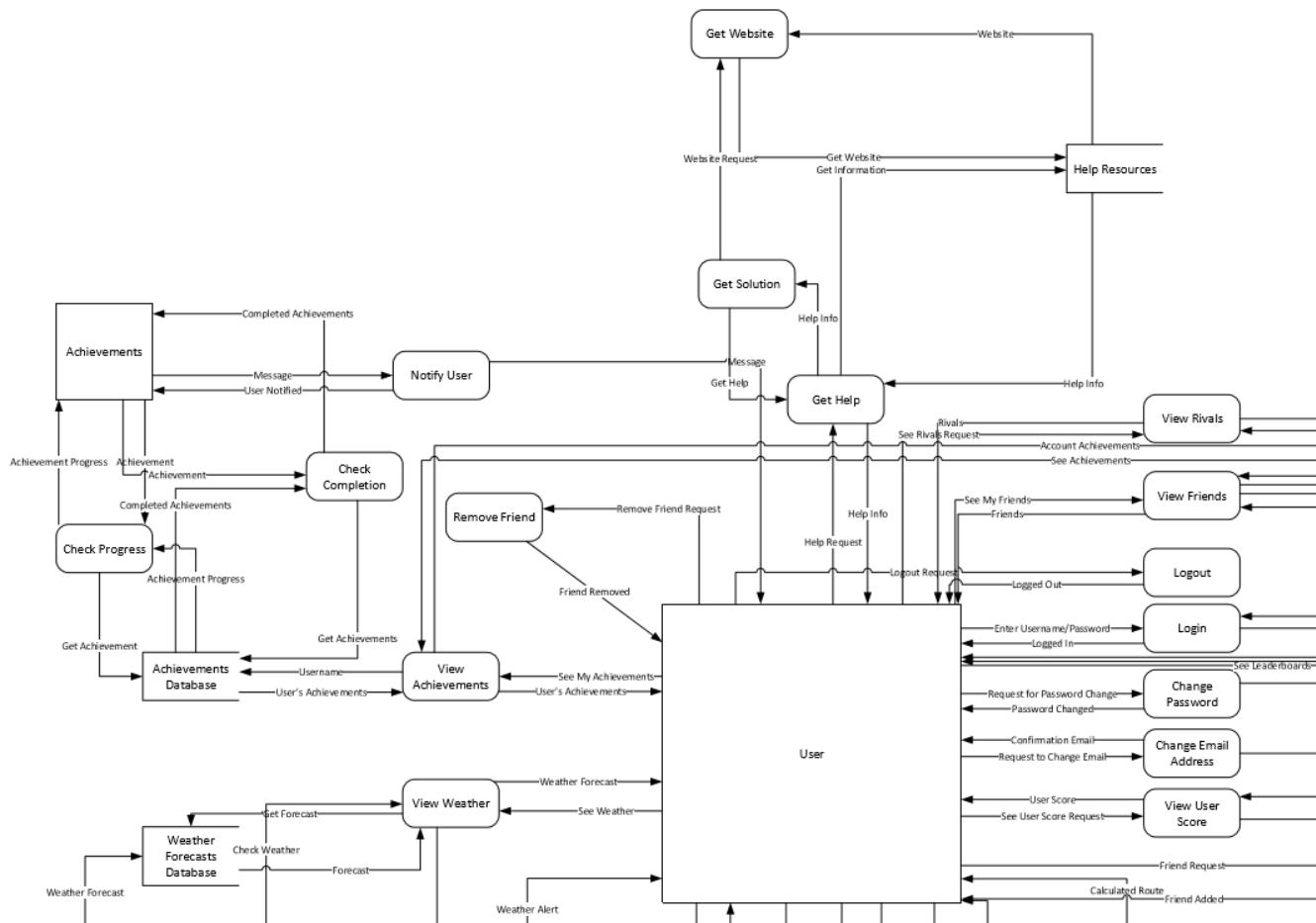
1a) Login Logout Change password Change email address Add friend Remove Friend 1b) Get route findRivals viewAchievements viewScore viewWeather getHelp 2a) Zoom In Zoom Out Recenter 2b) Change map type Show point of interest Show city Show friends 3a) Update location 3b) Place on map Proximity to friends Proximity to rivals	4a) ClearHistory 4b) showOnMap Suggestions UpdateSearchHistory UpdateDestinationHistory UpdateOriginHistory UpdatePointOfInterestHistory 5a) SearchPOI SavePOI SuggestToFriend 5b) LinkToWebsite NavigateToPOI showOnMap 6a) ChangeRoute 6b) CalculateRoute CalculateAlternateRoute CalculateETA Navigate RecalculateRoute ShowOnMap	7a) postFeedback editFeedback removeFeedback 7b) validateFeedback viewFeedback 8a) makeInvisibleToFriends changeAccountSettings 8b) viewAchievements viewFriends 9a) SavePlace RemoveSavedPlace 9b) showOnMap SuggestPlaceToSave 10a) checkWeather updateWeather setAlert 10b) removeAlert alertUser showOnMap	11a) searchLeaderboard filterLeaderboard updateLeaderboard 11b) showFriends showRivals 12a) sendFriendRequest acceptFriendRequest 12b) makeVisible makeInvisible 13a) sendChallenge acceptChallenge declineChallenge 13b) suggestRivals declareWinner rewardTransaction 14a) checkSuccess 14b) showProgress notifyUser
--	---	--	---



Level-0 Data Flow Diagram

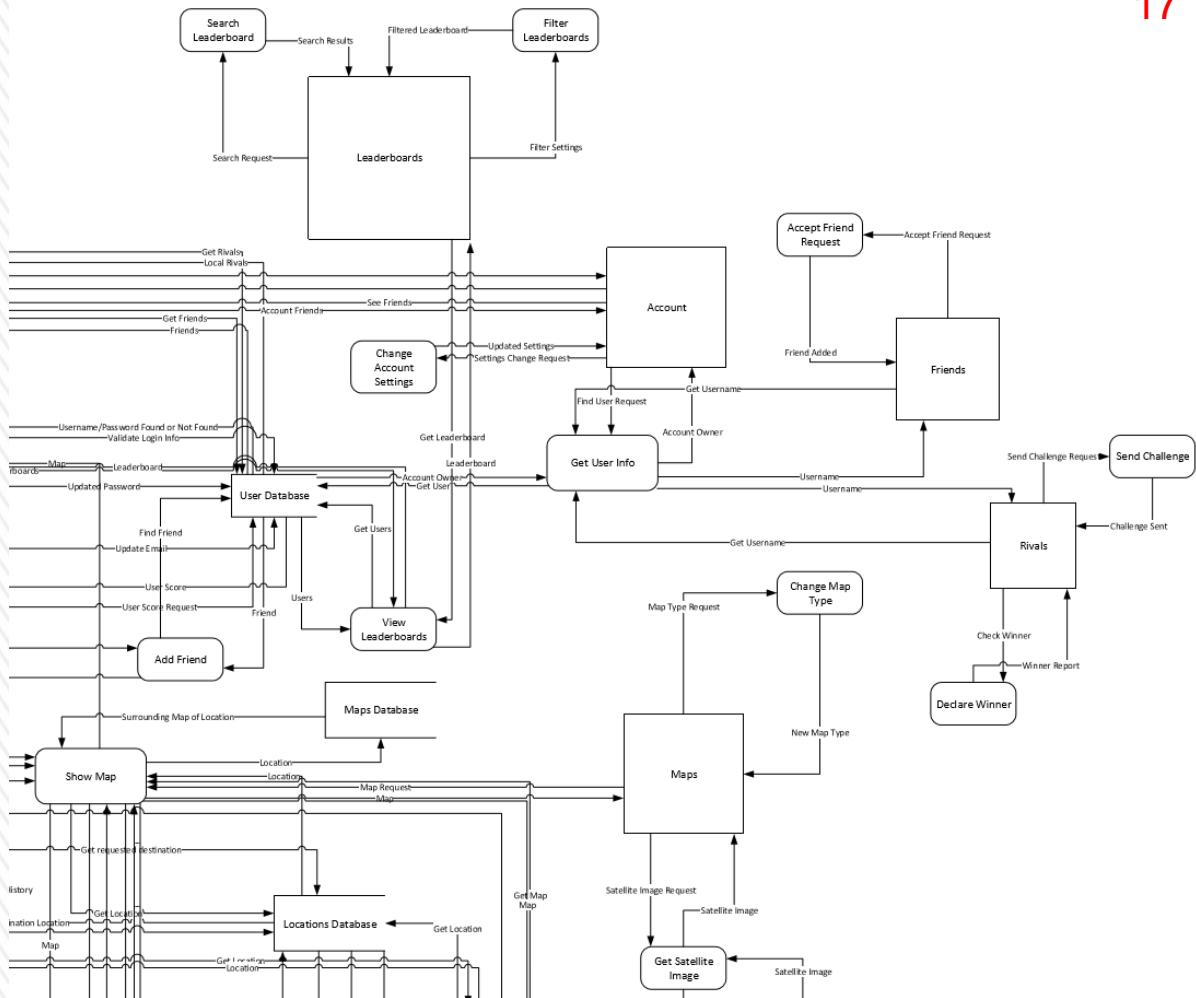


Level-0 DFD Section 1

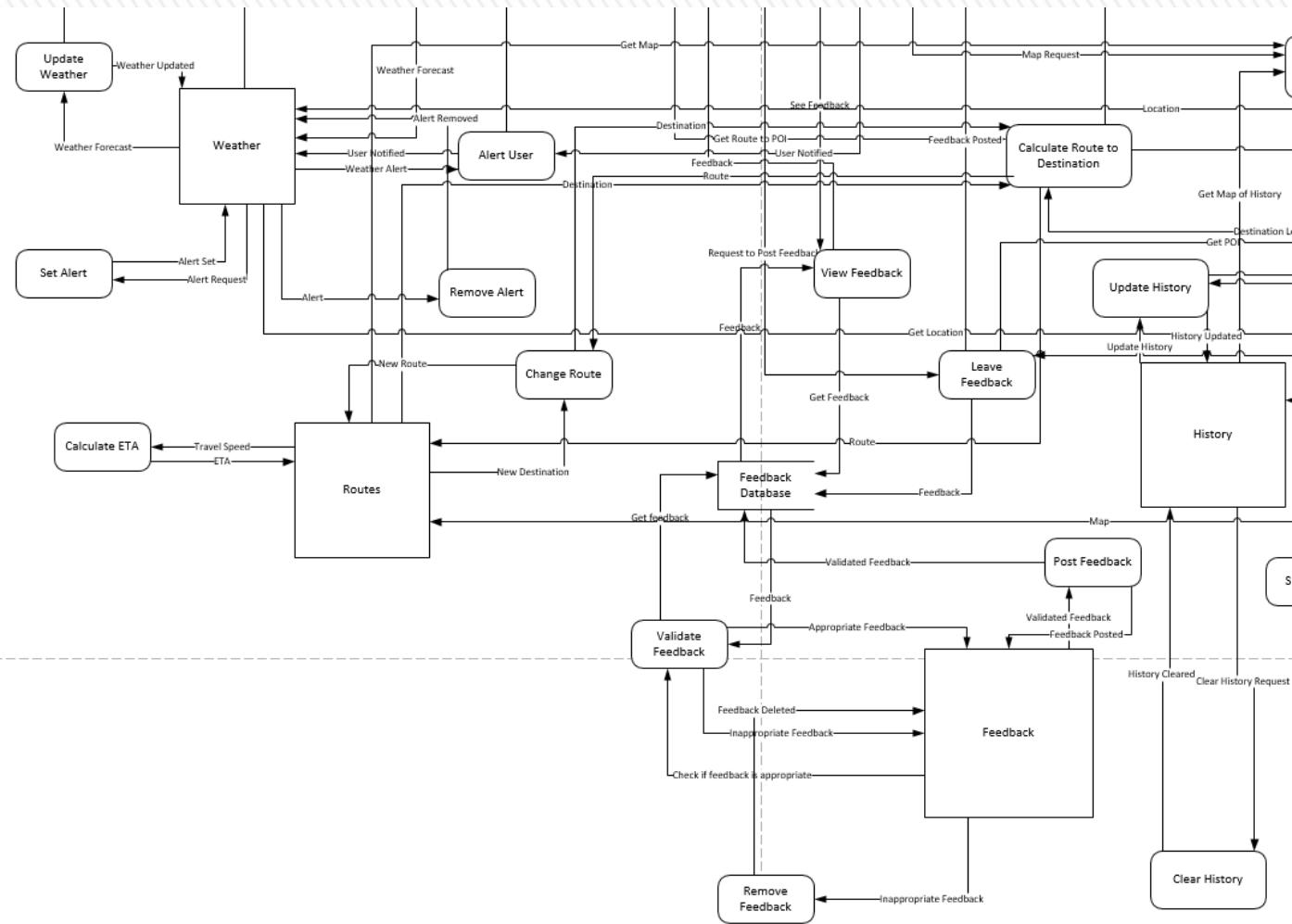


Level-0 DFD

Section 2

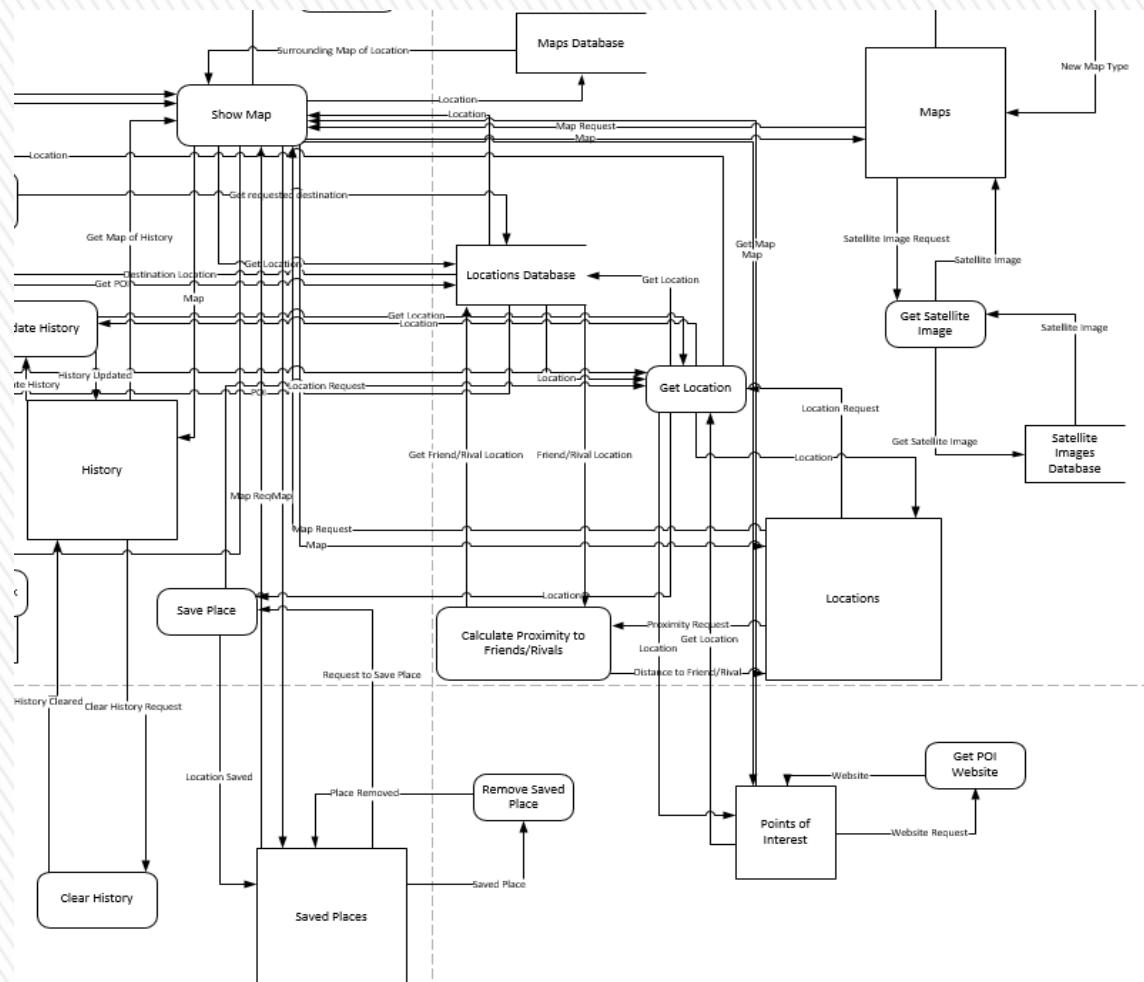


Level-0 DFD Section 3



Level-0 DFD

Section 4



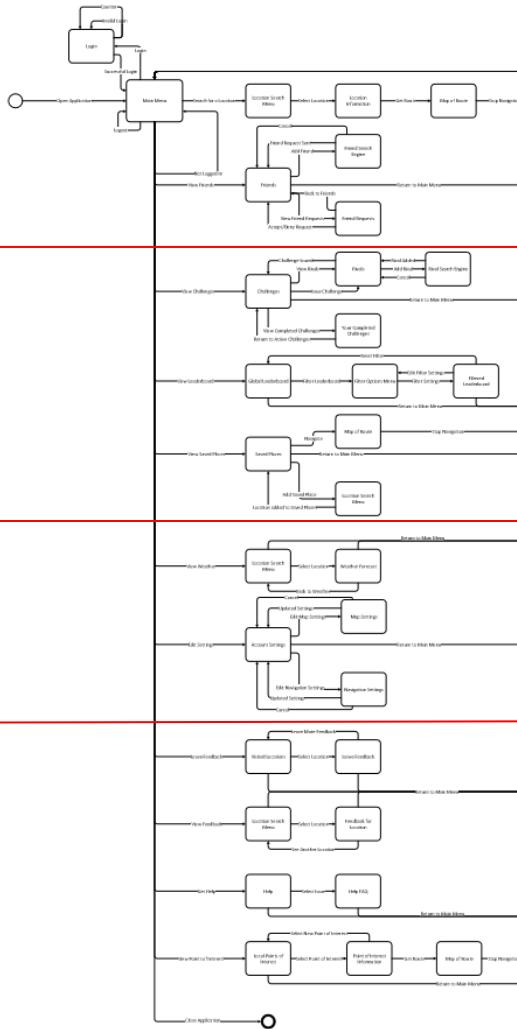
State Transition Diagram

1.

2.

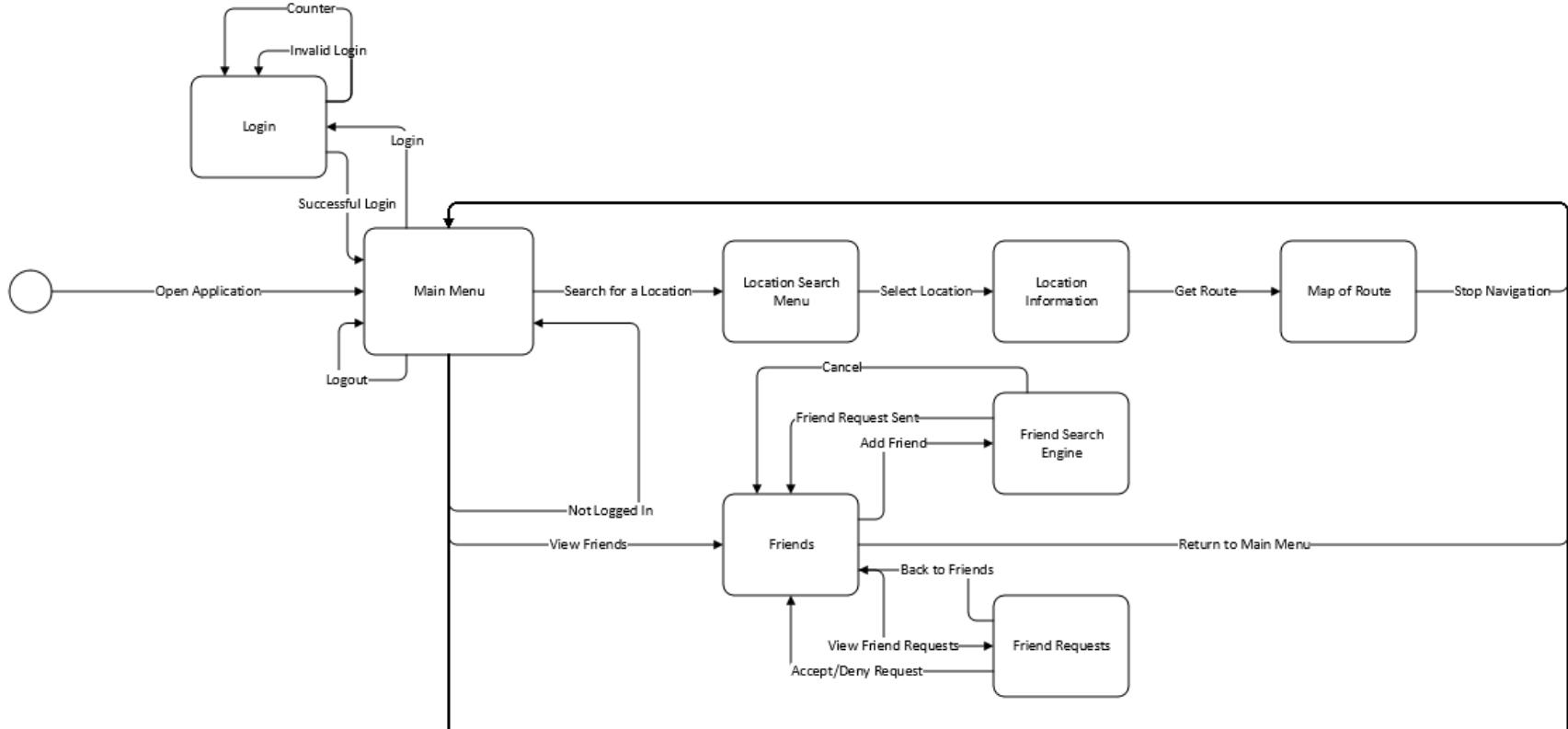
3.

4.

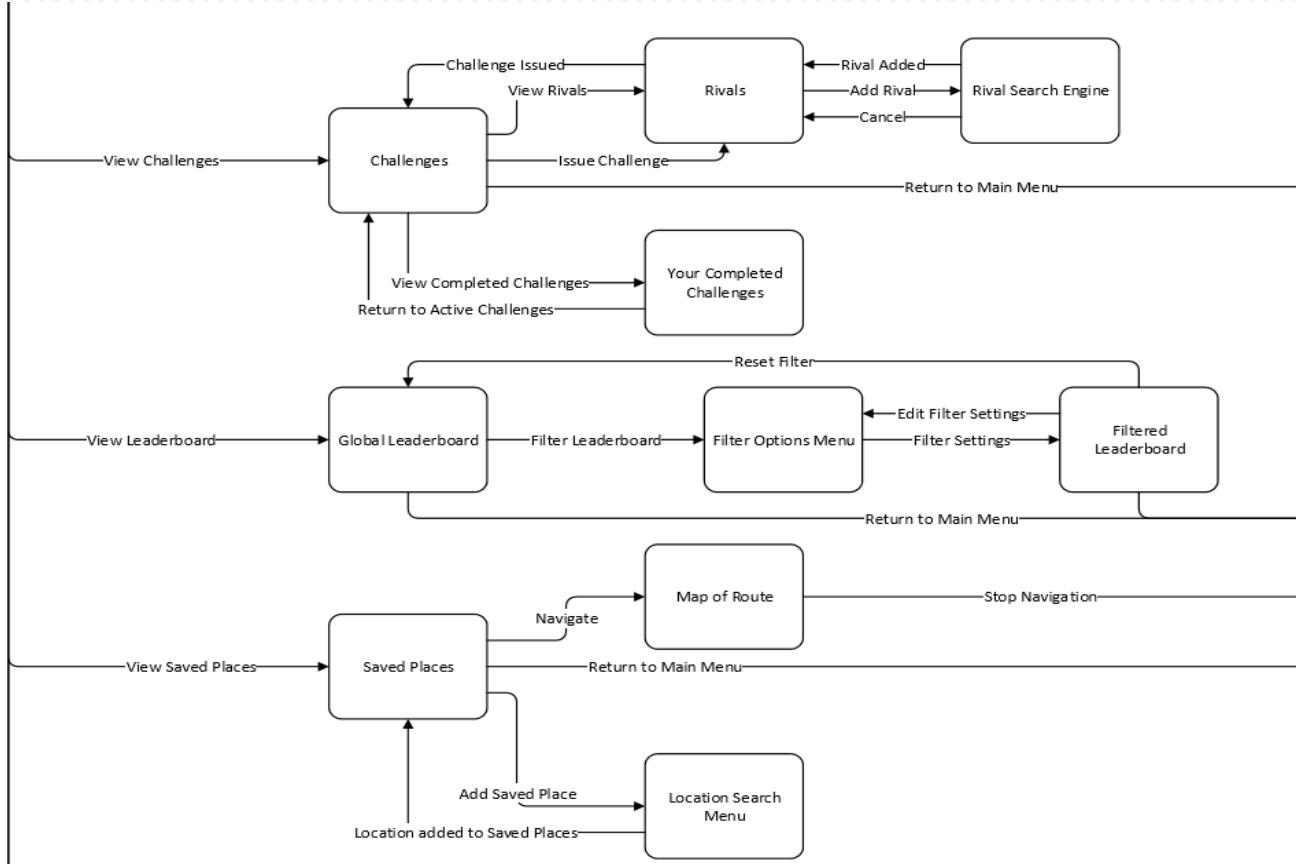


We decided to allow users to access the navigation system without having to log in. In doing so it allows the user to not have their location displayed to their friends and/or rivals that are also using the app; without the user having to change their account settings each time they would like to use the GPS feature in an incognito mode.

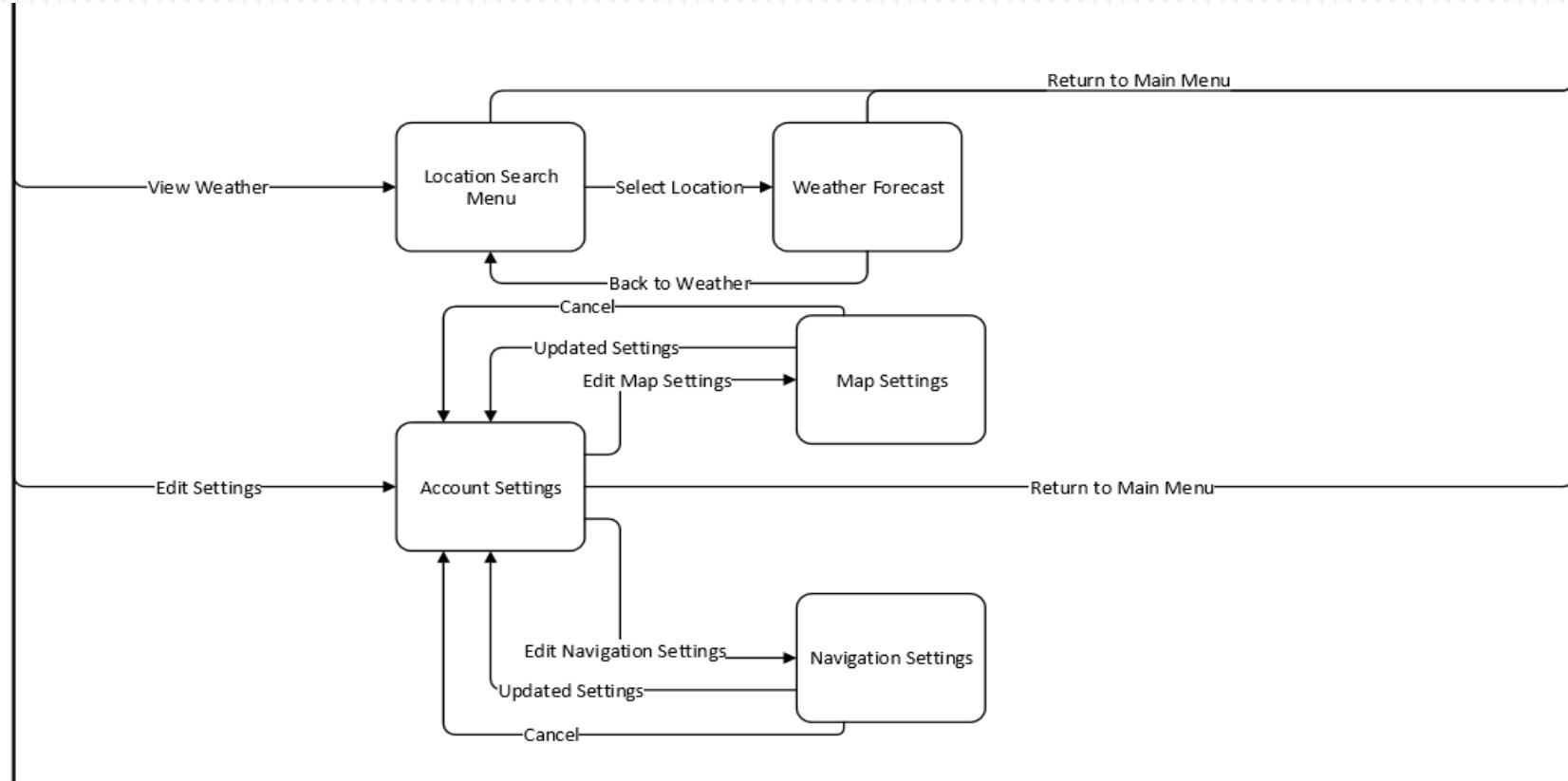
STD (Section 1)



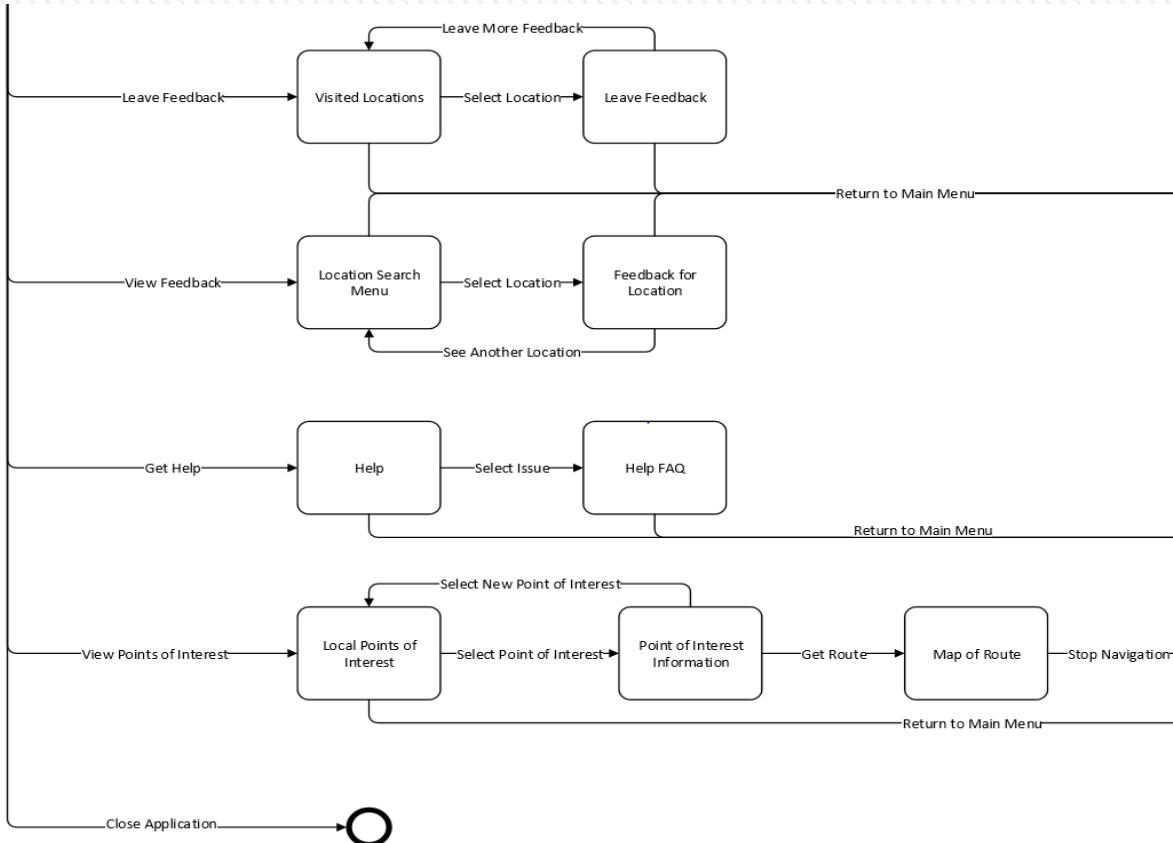
STD (Section 2)



STD (Section 3)



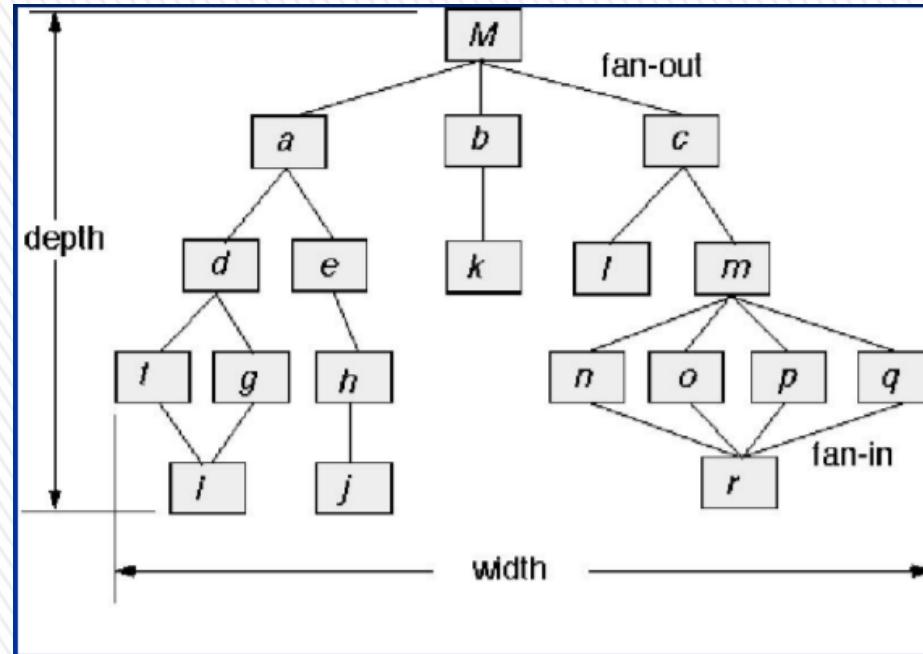
STD (Section 4)



Software Architectural Model

Call and Return Model

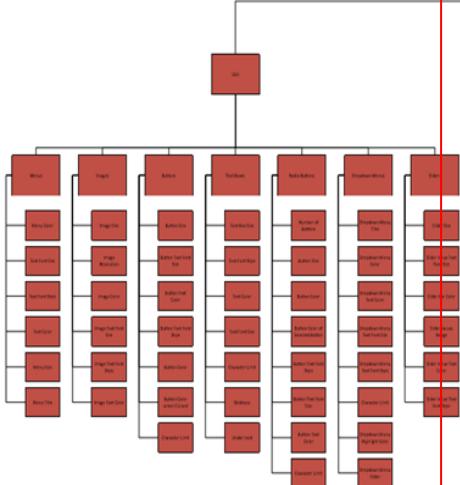
- User inputs address to obtain route which is then passed into a subfunction.
- User searches for locations or friends in a data base, and that data is then handled by sub functions.



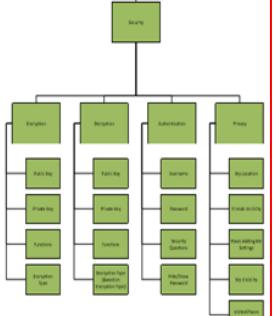
Hierarchical Model (System Architecture)

26

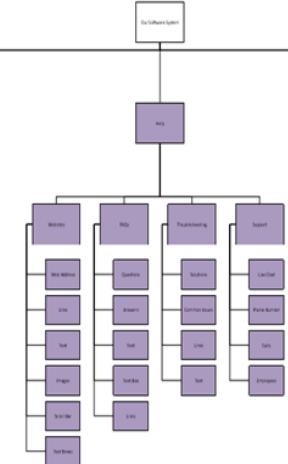
1.



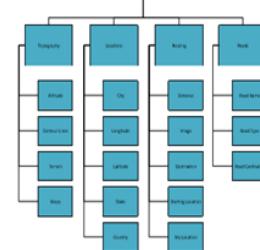
2.



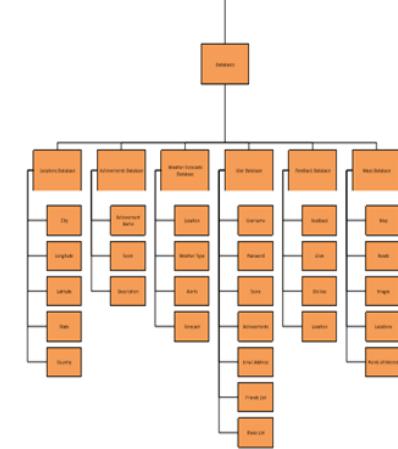
3.



4.

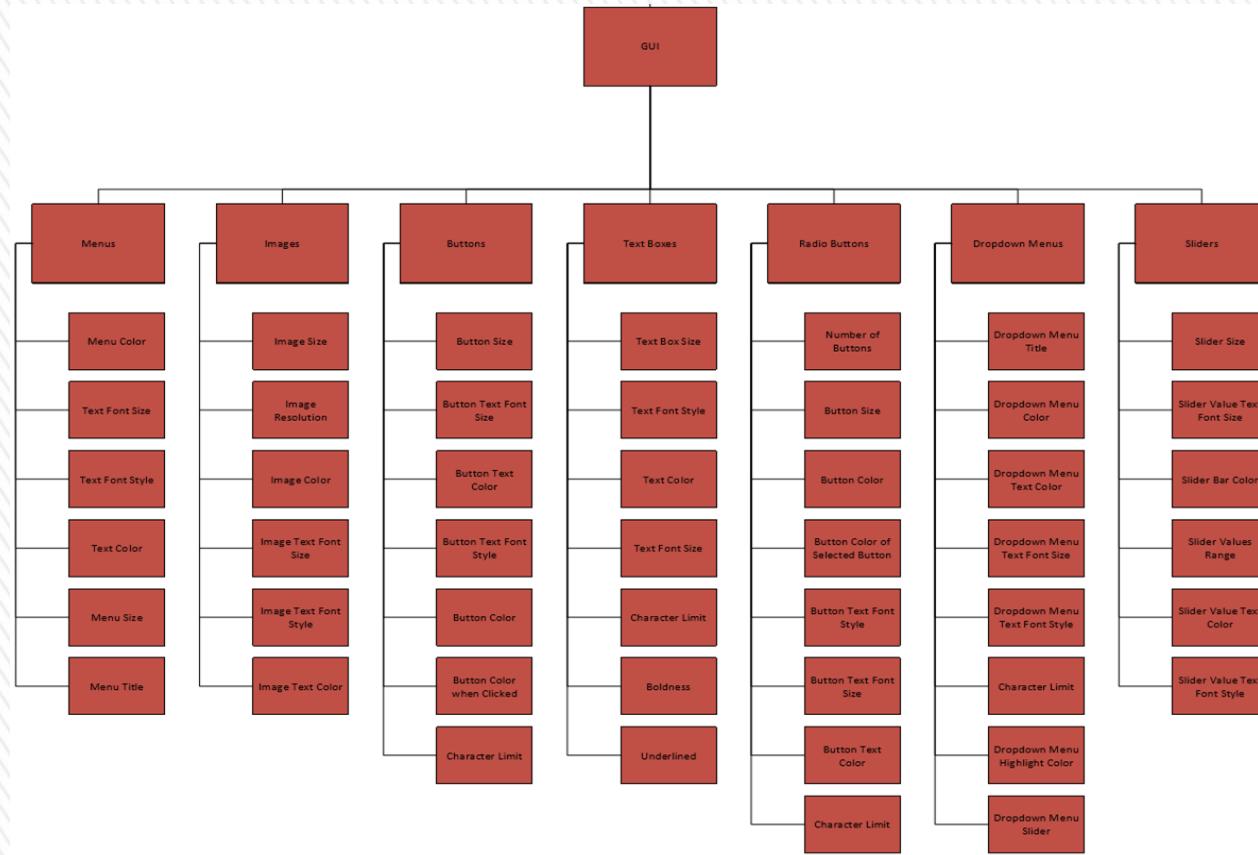


5

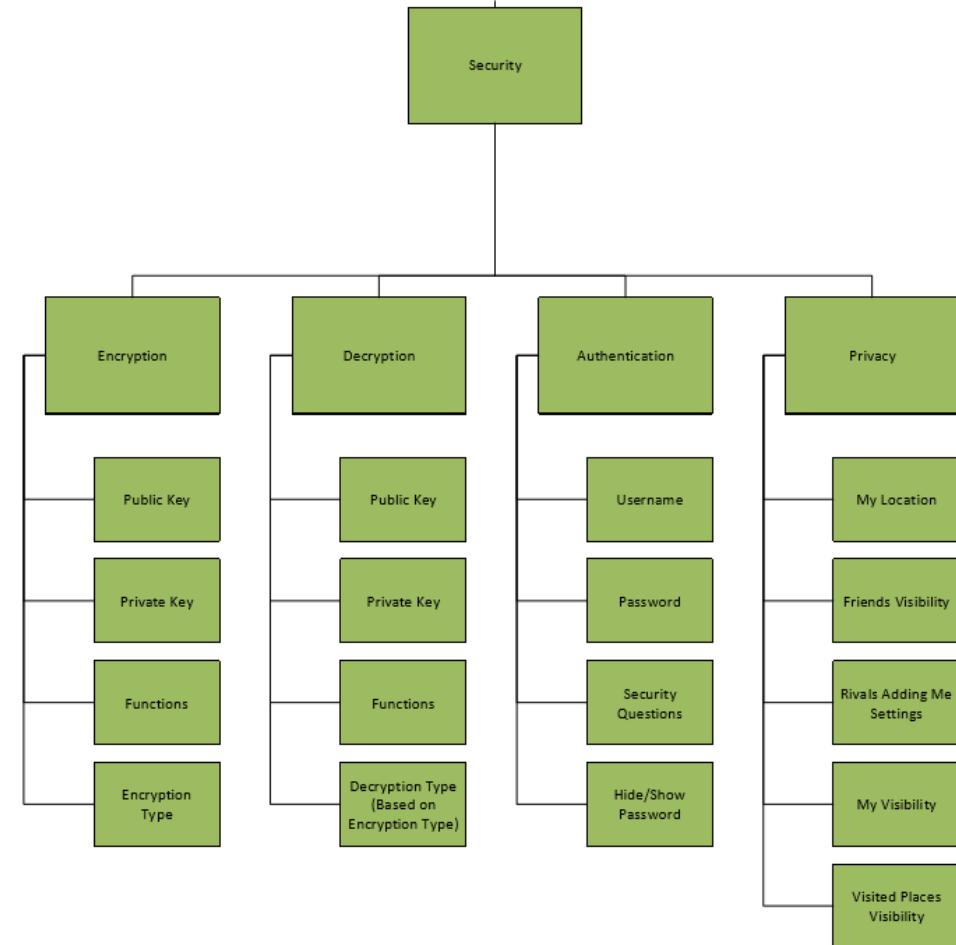


Contains the software system's subsystems, the subsystems' components, and then finally details for each component of the subsystems.

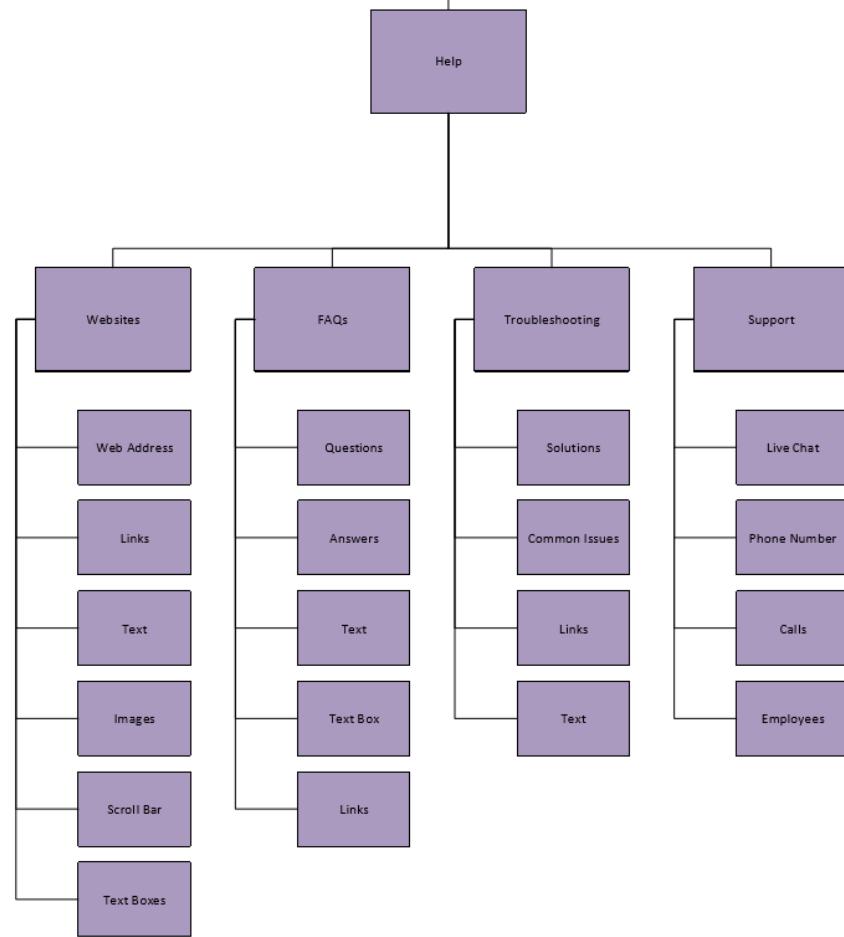
GUI Subsystem (1)



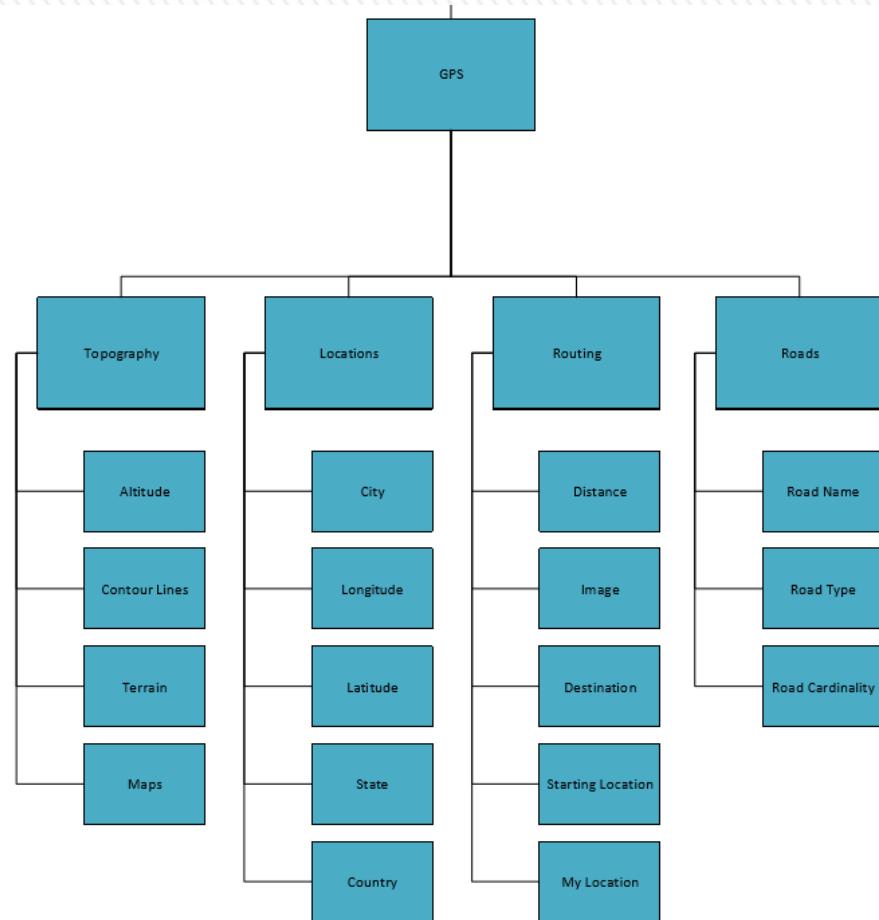
Security Subsystem (2)



Help Subsystem (3)

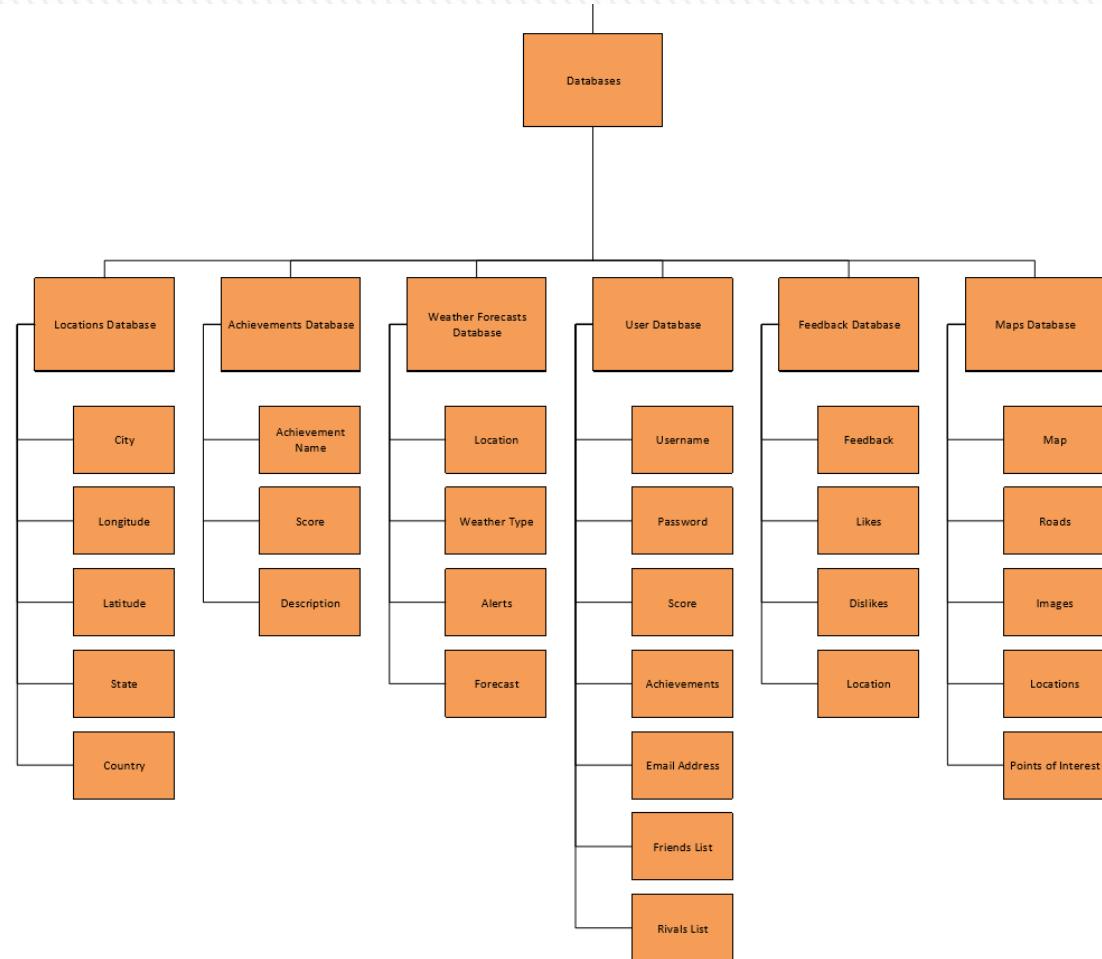


GPS Subsystem (4)

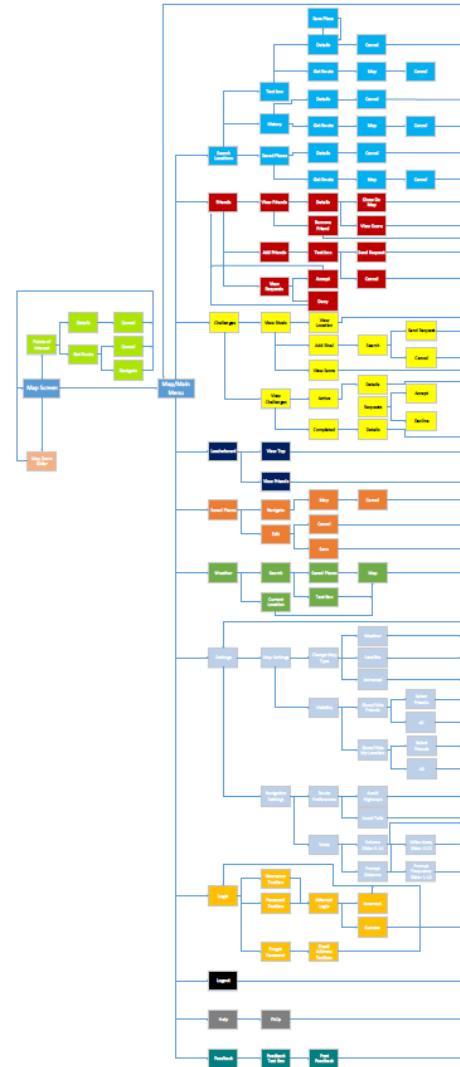


Databases

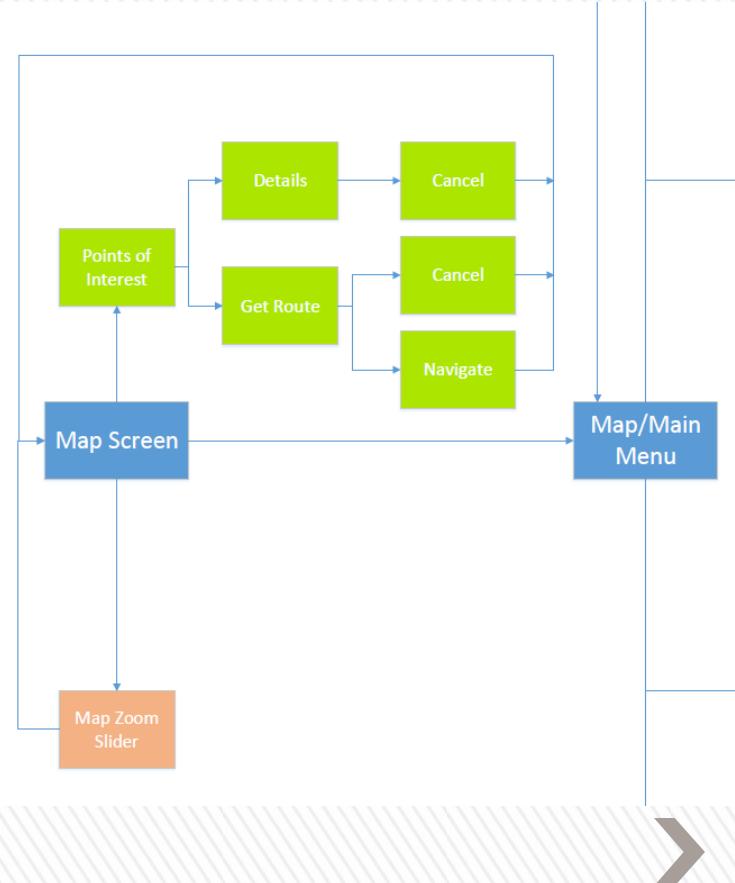
Subsystem (5)



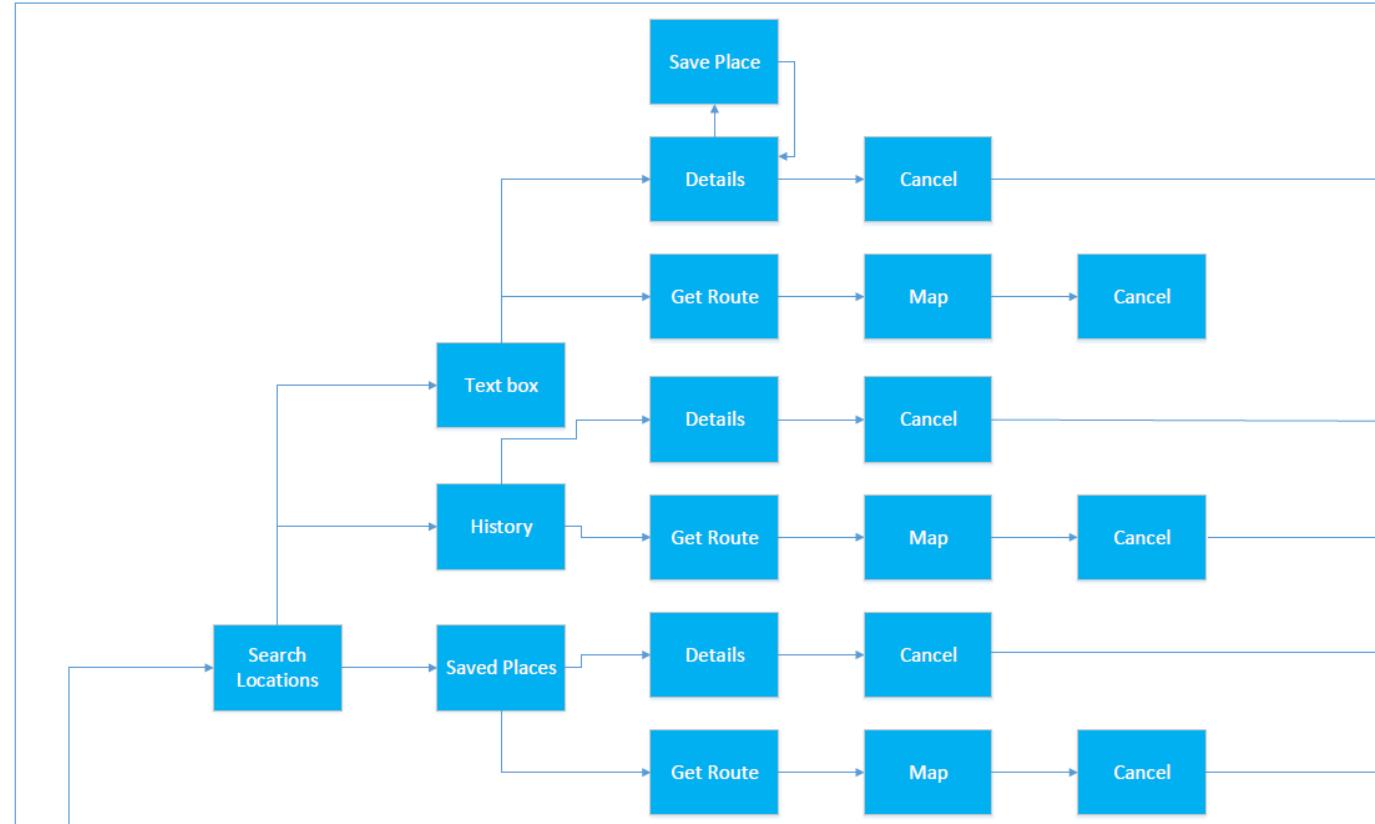
GUI Functional View



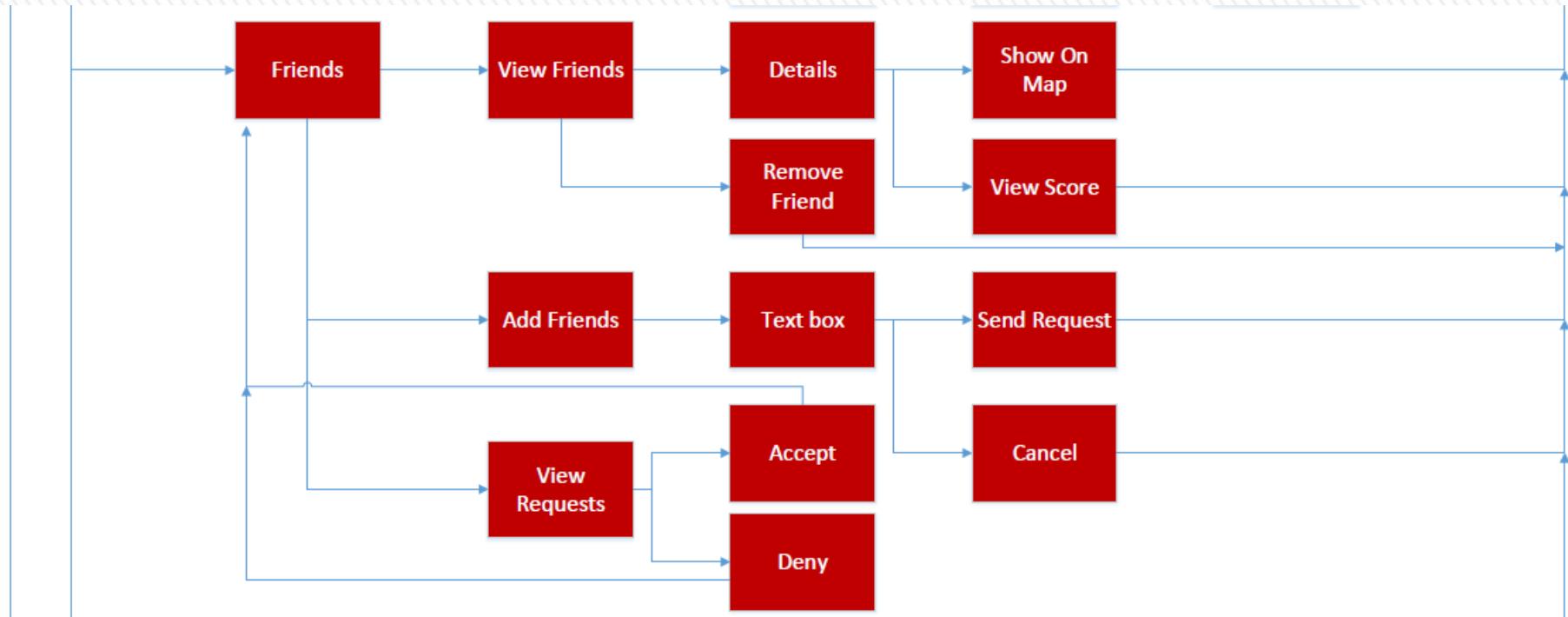
GUI Functional View Part 1



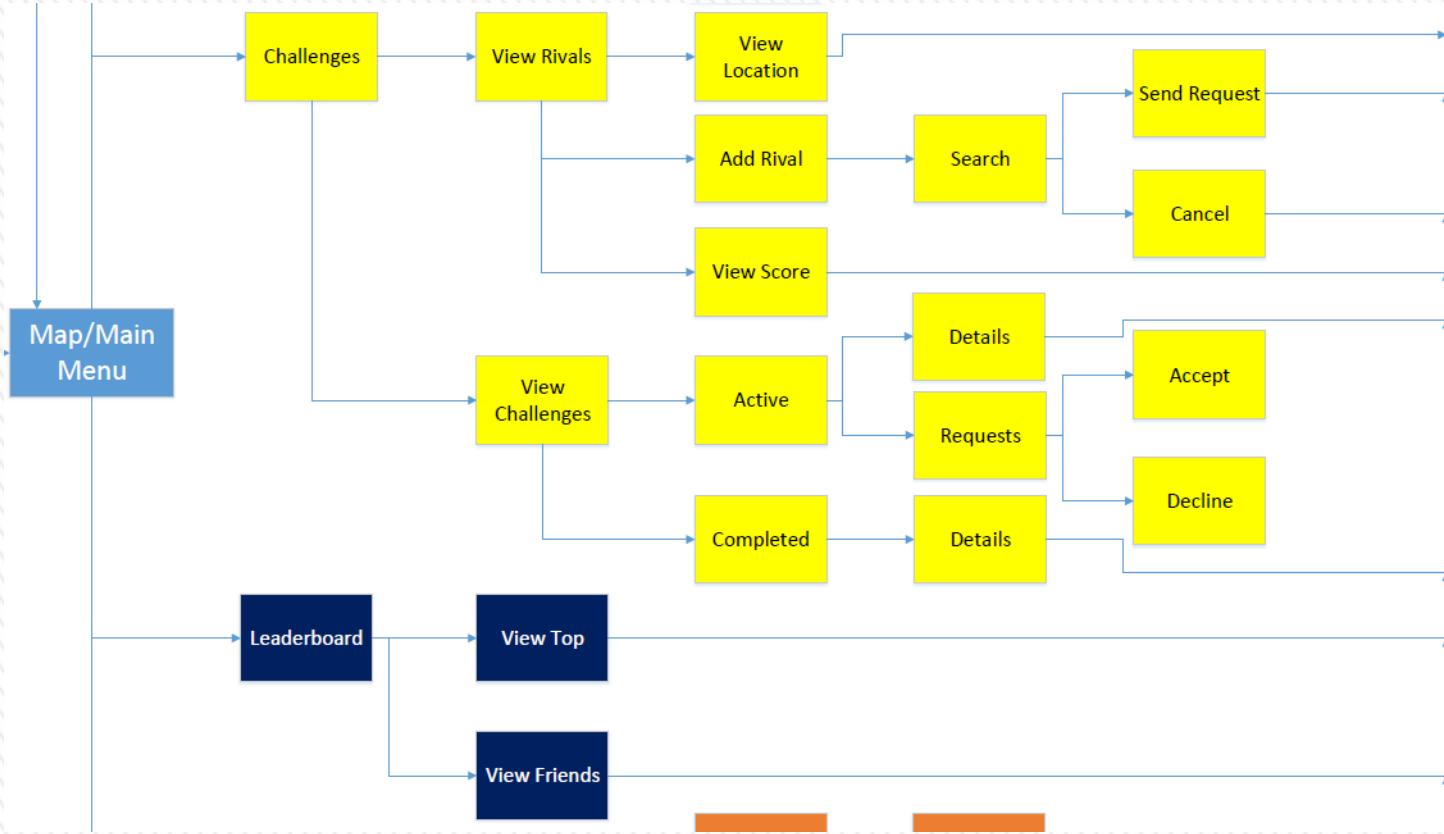
GUI Functional View Part 2



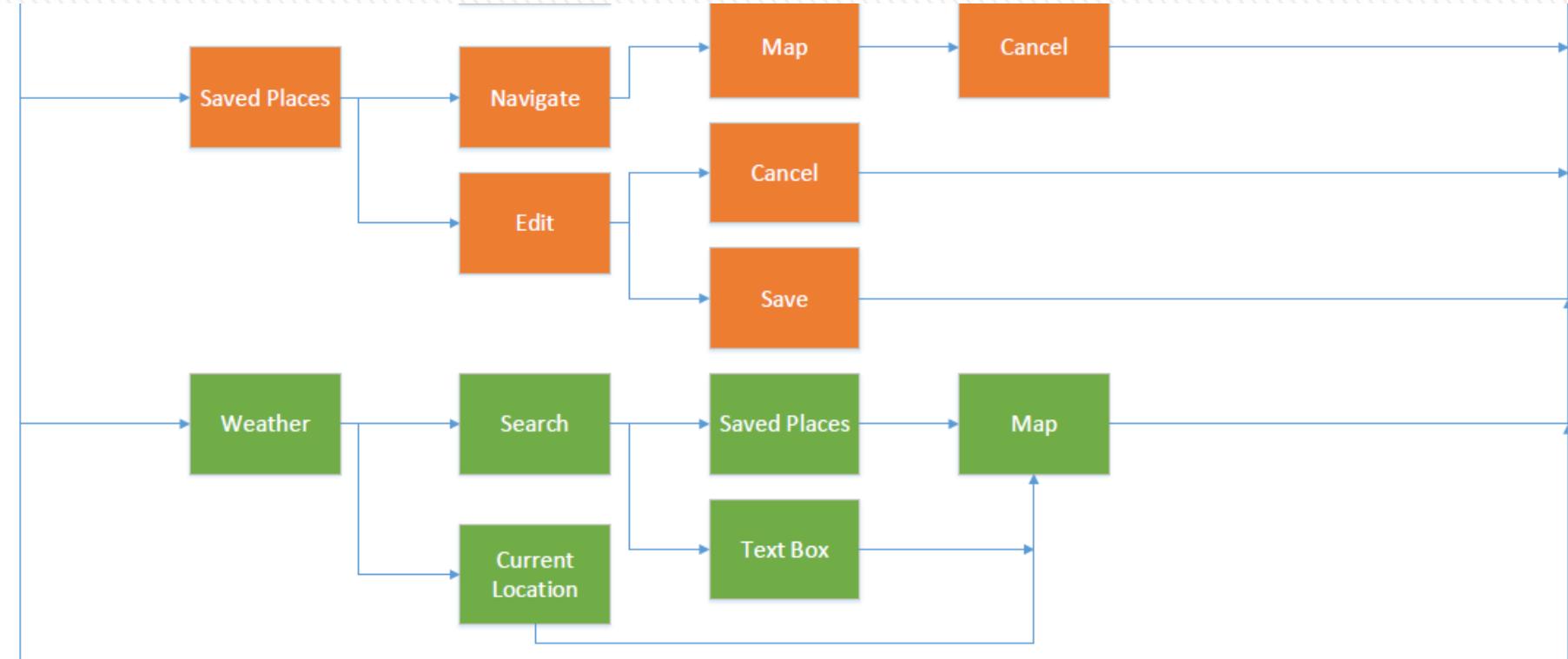
GUI Functional View Part 3



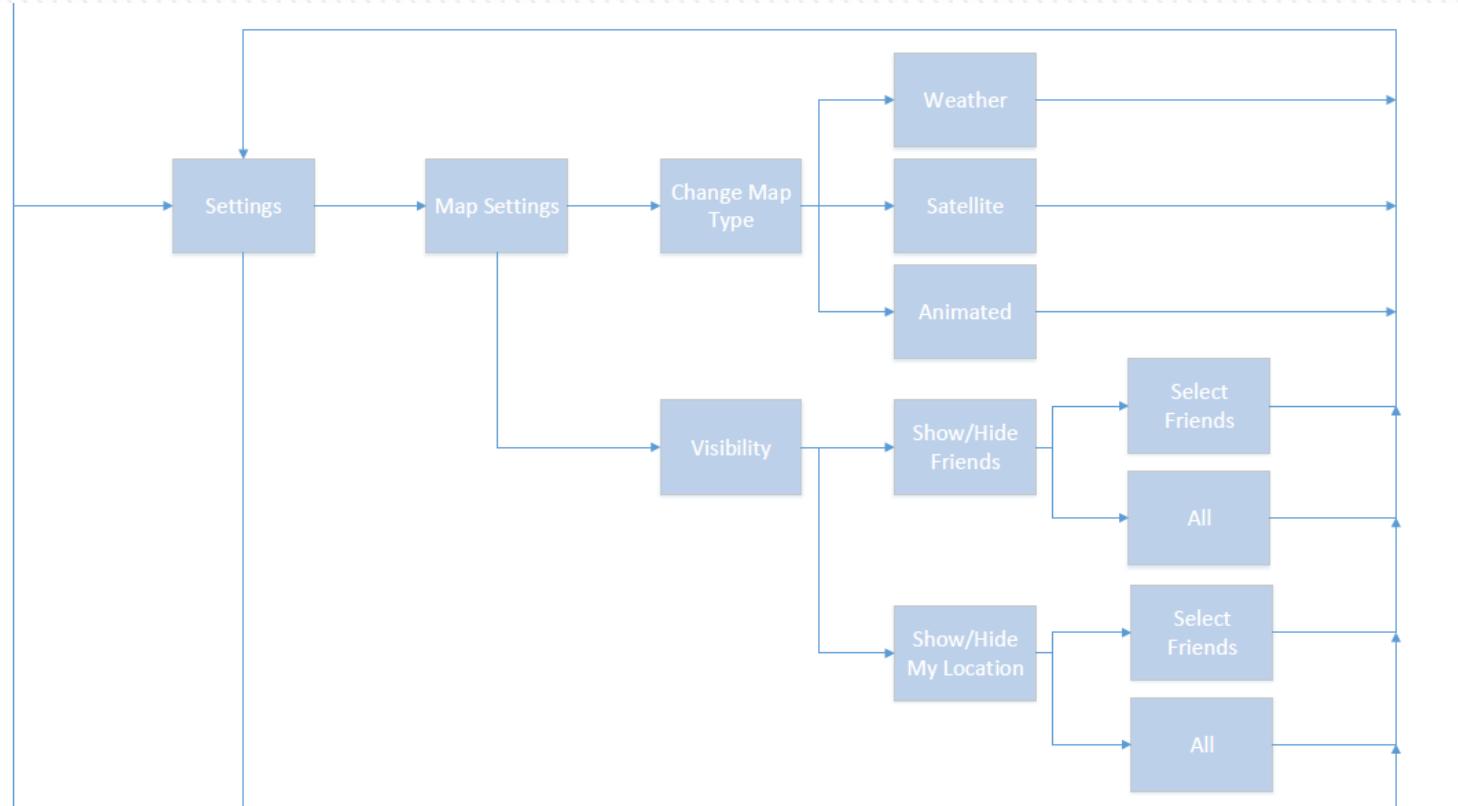
GUI Functional View Part 4



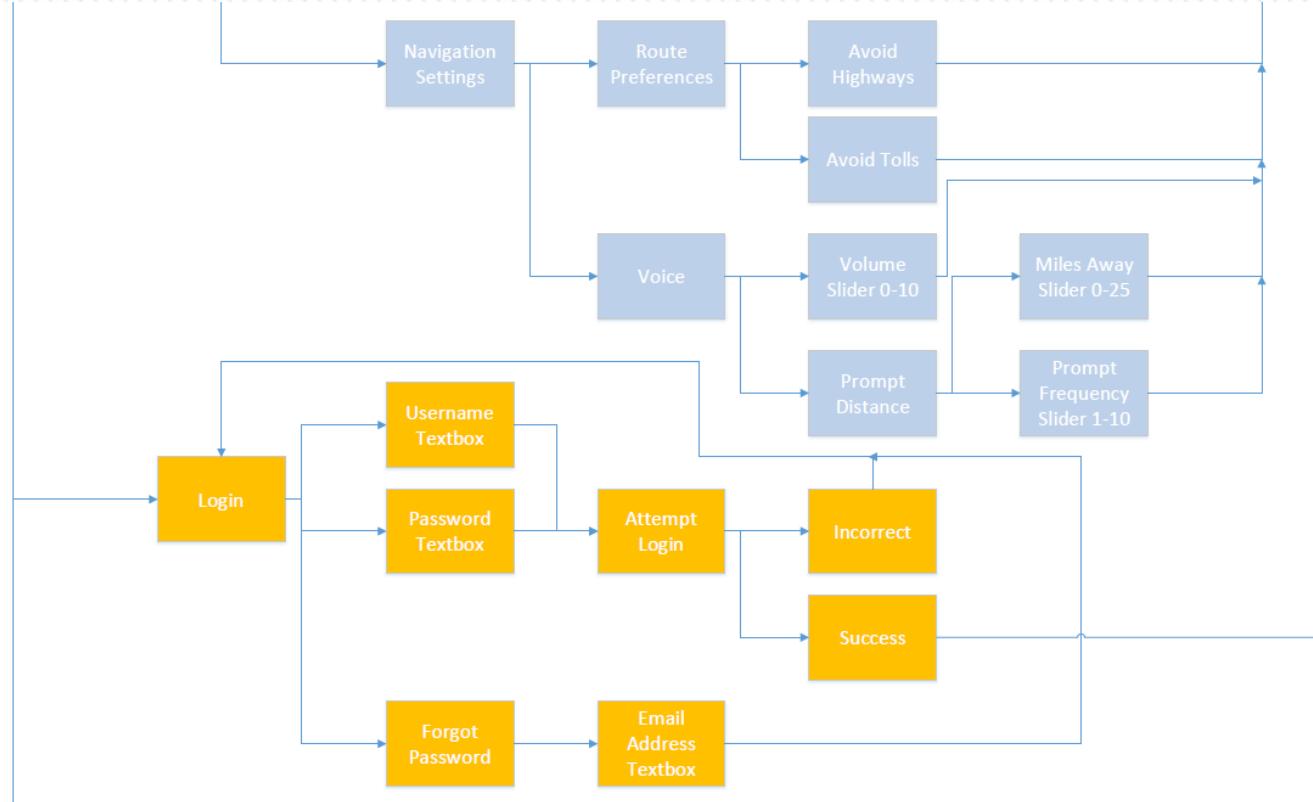
GUI Functional View Part 5



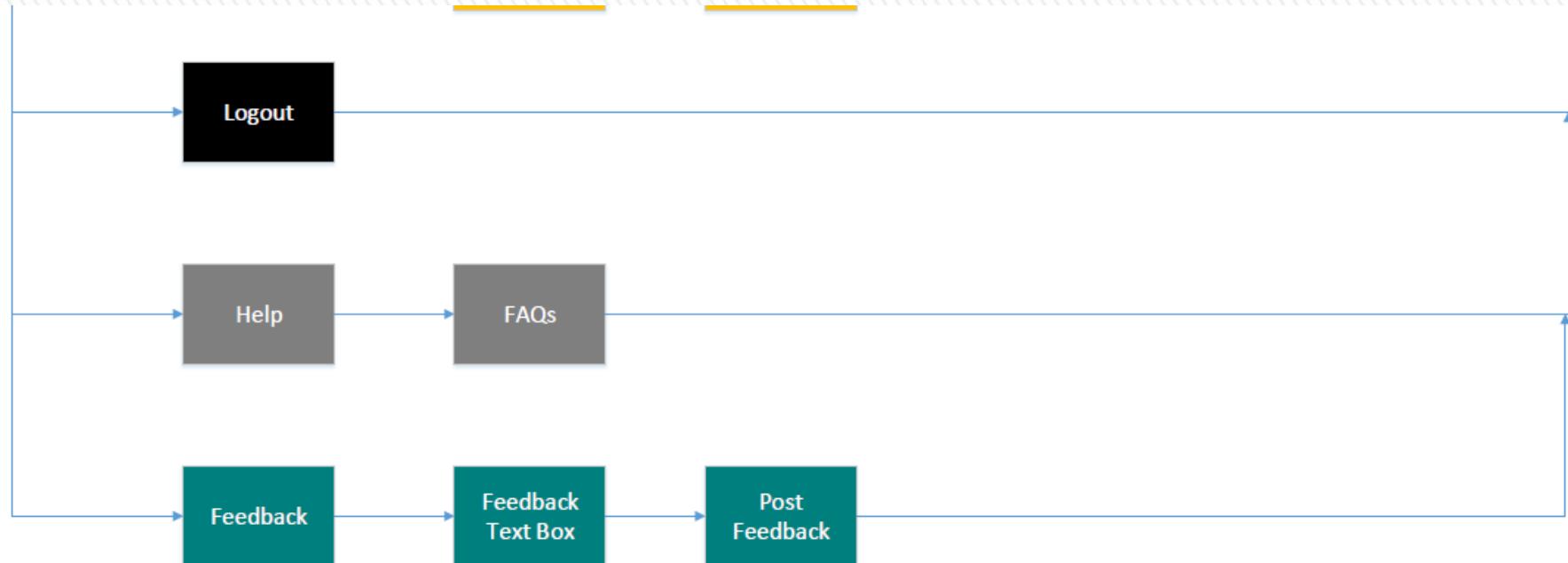
GUI Functional View Part 6



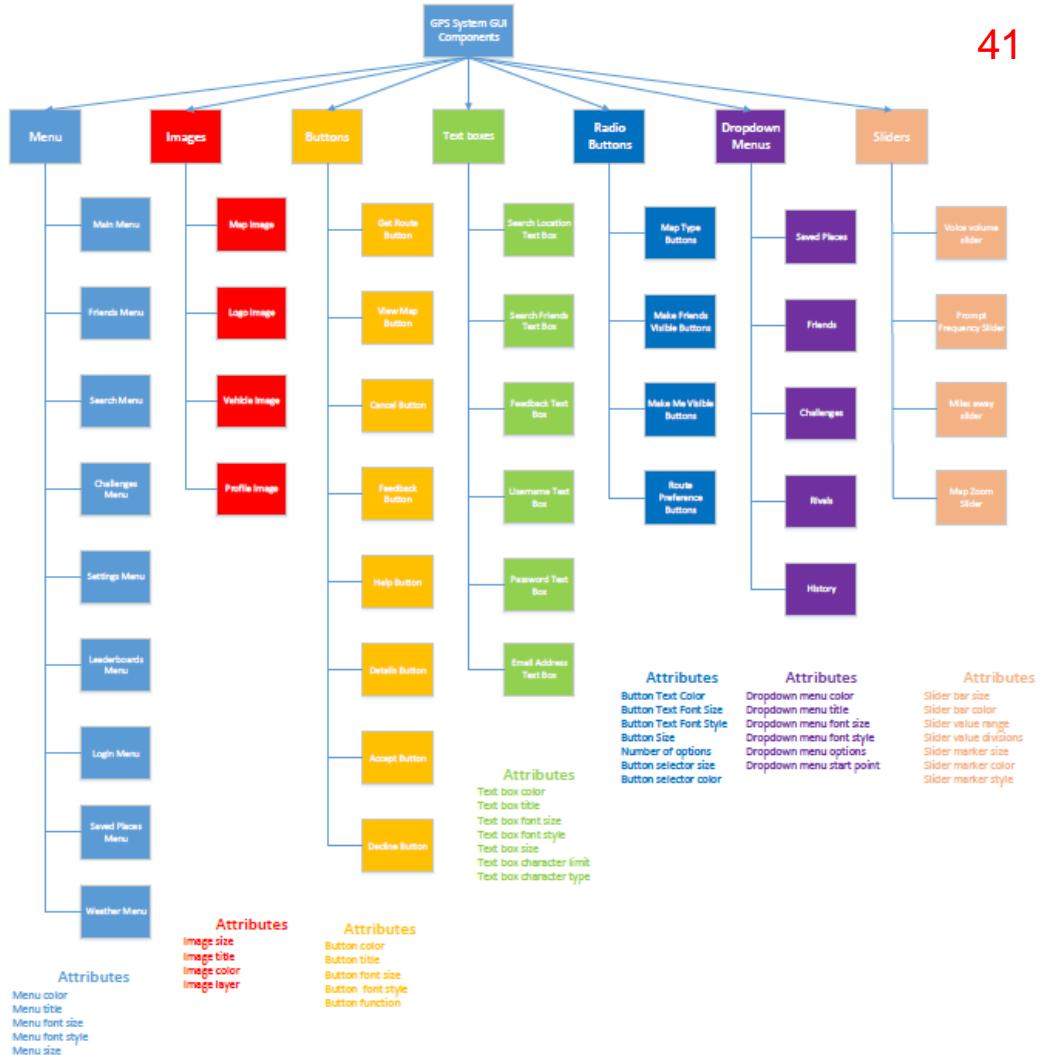
GUI Functional View Part 7



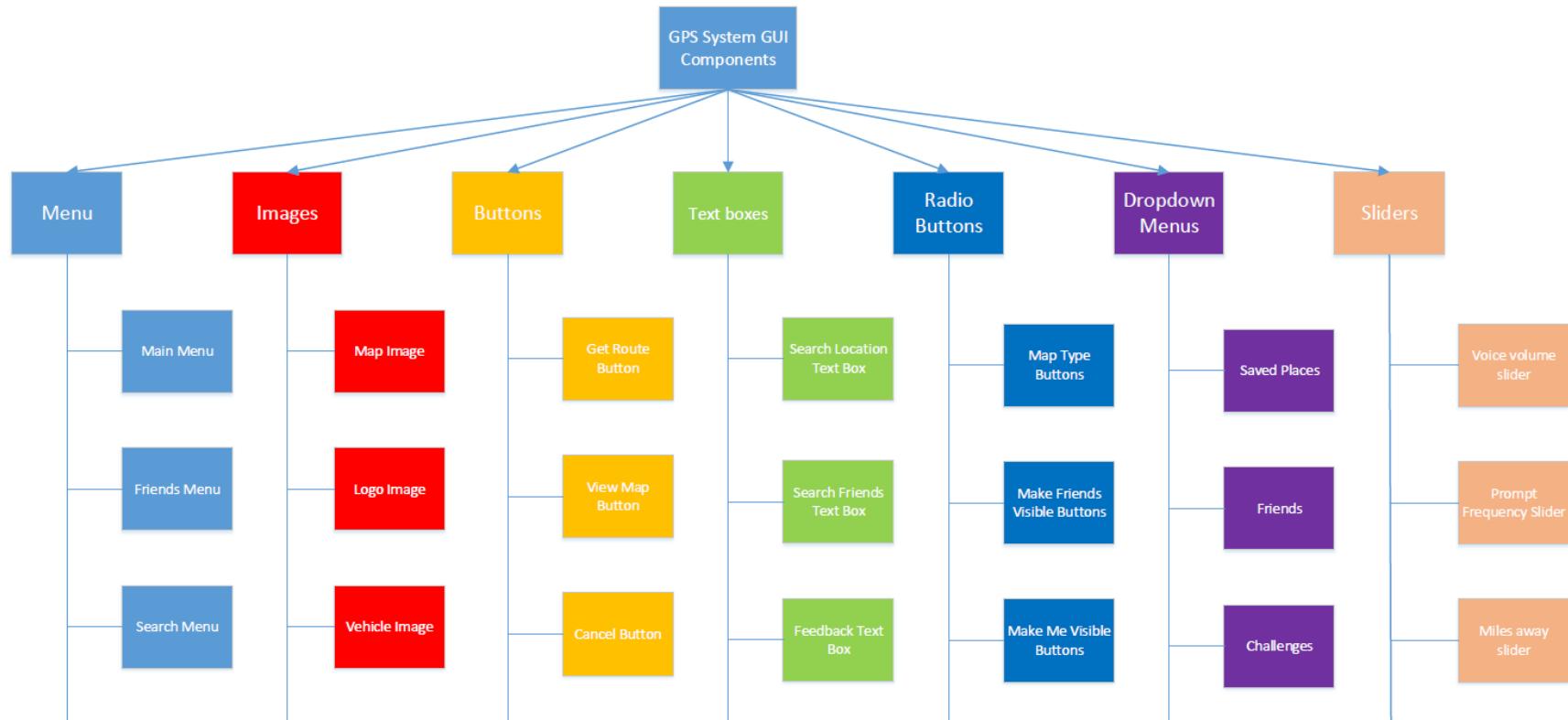
GUI Functional View Part 8



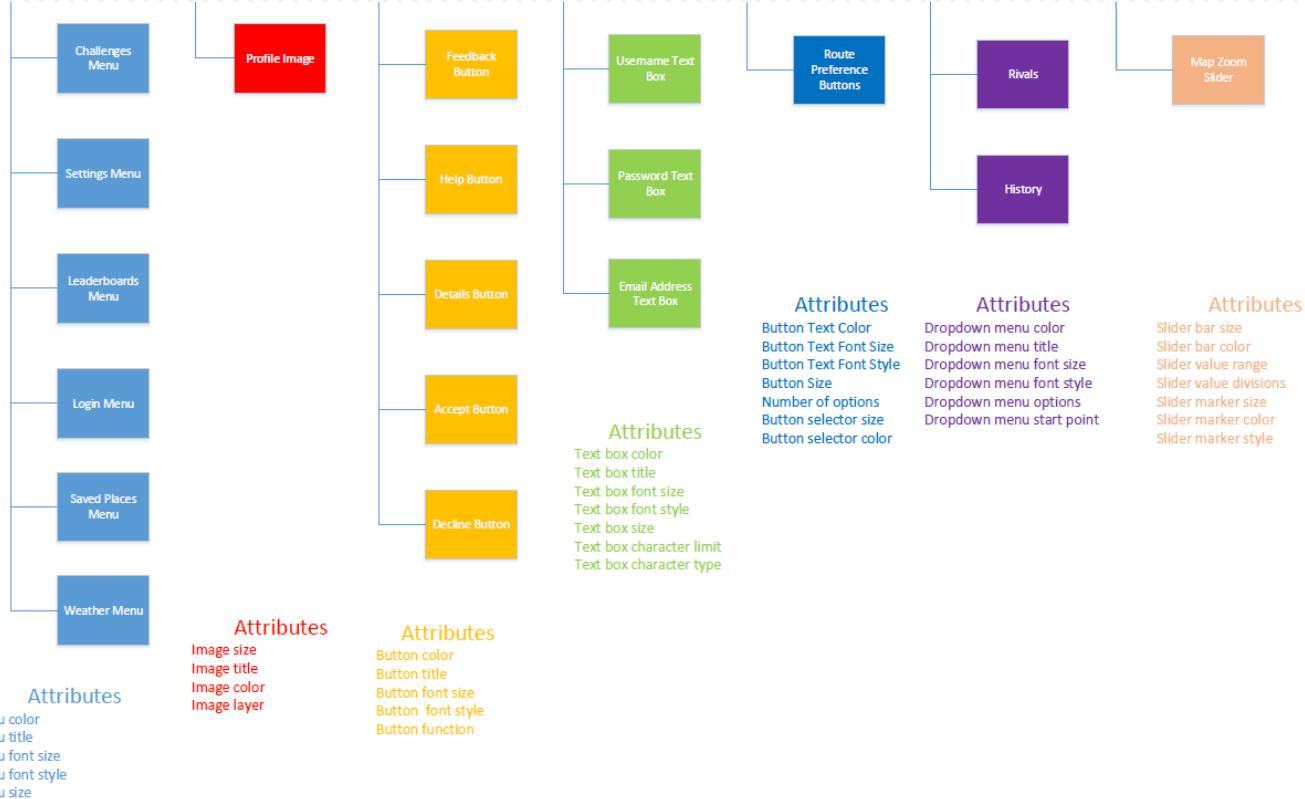
GUI Component View



GUI Component View Part 1



GUI Component View Part 2



FP-Based Estimation

Information Domain Value	Optimal	Most Likely	Pessimistic Estimation	Estimated Count	Weight	FP-Count
Number of inputs	45	60	75	60	4	240
Number of outputs	35	50	80	50	8	400
Number of inquiries	10	15	25	15	3	45
Number of files	8	12	18	12	6	72
Number of external interfaces	4	6	8	6	8	48
Total						805



FP- Based Estimation

Number of Function Points: 805

Average productivity: 6.5 FP/month

Burdened Labor Rate: \$8000/month

The project will take 124 man months and cost \$992,000 to produce.



LOC-Based Estimation

Function System	Estimated LOC
User Interface	5,000
Route Calculation	35,000
Location Calculation	20,000
Database Management	10,000
Gamification	10,000
Total Estimate	70,000

Lines of code: 70,000

Average productivity: 620 LOC/month

Burdened Labor Rate: \$8000/month

The project will take 113 man months and cost \$904,000 to produce.



Thank You!

Are there any questions?

Team 02

Dylan Kehres & Michael Schott

