Project: Red Pony

Daniel Kim
Daniel Olivera
Jared Coleman
Mario Lopez
Viet Le

Problem

- Traditionally, access to items in a physical world is restricted by physical laws, such as
  - Location
  - Time
- Items in the digital realm are not restricted in this way
- This is often an advantage, but can sometimes be a burden
  - Items that belong to a location
  - Items that should not be shared
  - Items that should expire

Goal

- To create a framework for enforcing physical access restrictions for digital files
    - Tie digital files to the physical world
- To create an application that uses the framework and demonstrates its applicability
    - Location restrictions
    - Time restrictions

Proposed Solution

Astral

- Server-side framework for storing and providing access to files with location and time restrictions
- Tools
  - MongoDB
  - SSL
  - Node.js RESTful API
  - JSON Web Token authentication

Application - Application name goes here

- File Sharing
  - Share files to locations
  - Create rooms
    - File sharing
    - Discussion
- Tools
  - Ionic
  - AngularJS
  - HTML5

Astral

Overview

- Server
  - MongoDB
  - Node.js
- RESTful API
  - GET and POST files from/to server
  - Location and time restrictions verified server-side
  - JSON Web tokens for authentication
- Uses
  - Scalable
  - Useful for range of applications

MongoDB

File Schema:

```
{
  _id: <ObjectID>
  name: String
  location: String
  radius: Integer
  expiration: Integer
  data: <JSON Object>
}
```

User Schema:

```
{
  _id: <ObjectID>
  username: String
  password: String
  data: <JSON Object>
}
```

API

- Mongoose
  - Integrating MongoDB database with Node.js server
- Functions

New file:	POST {File}
Update file:	PUT {File}
Get nearby files:	GET {location, time}: [{File}]
Login:	GET {username, password}: {User}
New User:	POST {User}

Uses

- Scavenger hunt application
- Business foot-traffic
- Local social networks
- Virtual Classroom
- Community

Application name goes here

Overview

File sharing over the Physical world

- Access Restrictions
    - Location
    - Time
    - User ID
    - User demographic
- Virtual rooms
    - Reflect physical locations
    - File sharing
    - Discussion

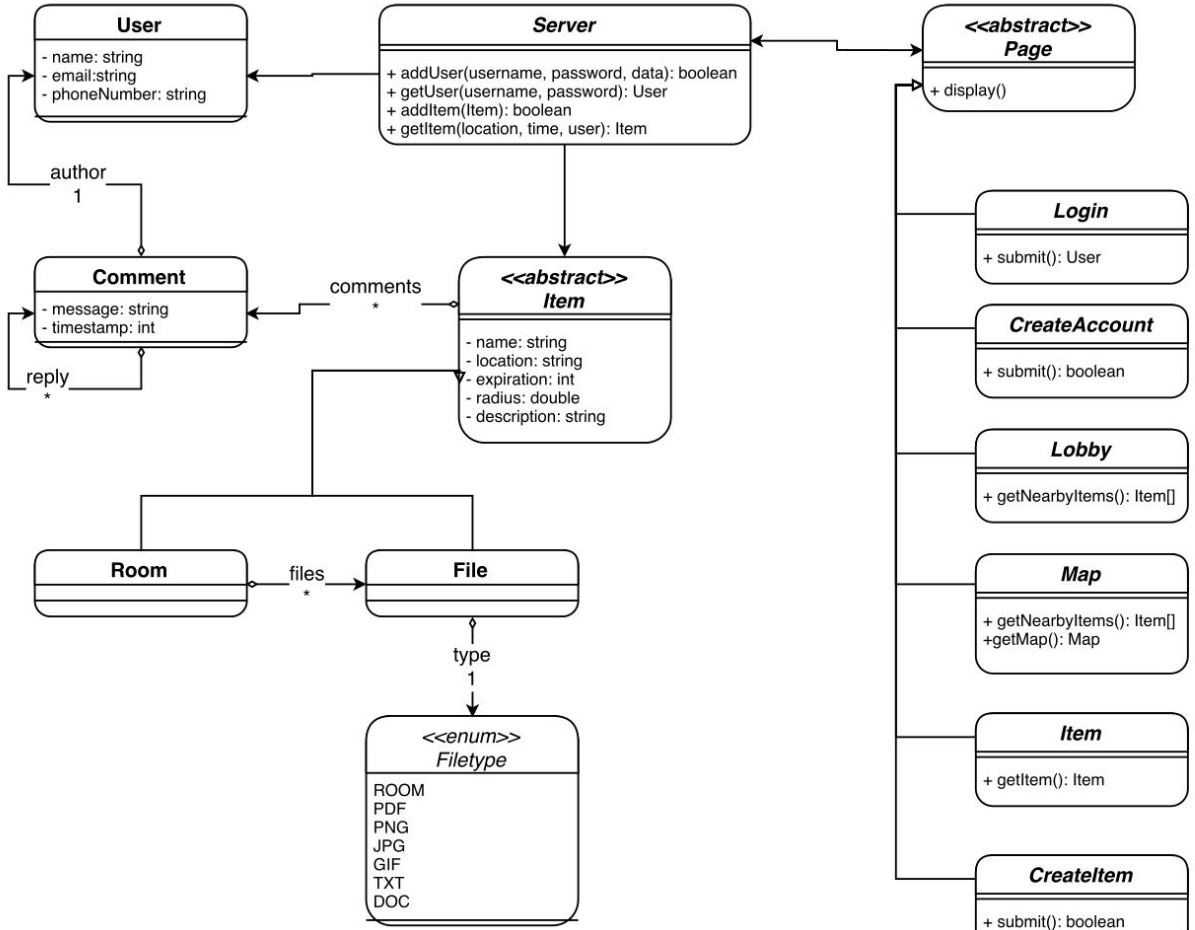
Sharing Files

- Users can drop files to their location
- Users can access files within a certain radius
- Users can see files that exist at a greater radius

Virtual Rooms

- Users can create virtual rooms at their location
- Virtual rooms only show up to invited members
- Virtual rooms are only accessible when user is at room location

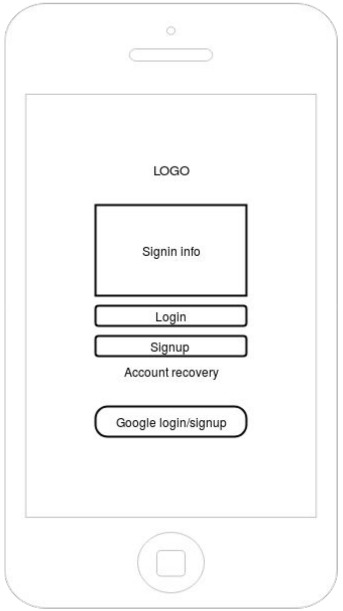
UML


User
- name: string
- email:string
- phoneNumber: string
Server
+ addUser(username, password, data): boolean
+ getUser(username, password): User
+ addItem(Item): boolean
+ getItem(location, time, user): Item
<<abstract>> Page
+ display()
author
1
Comment
- message: string
- timestamp: int
reply
*
comments
*
<<abstract>> Item
- name: string
- location: string
- expiration: int
- radius: double
- description: string
Room
files
*
File
type
1
<<enum>> Filetype
ROOM
PDF
PNG
JPG
GIF
TXT
DOC
Login
+ submit(): User
CreateAccount
+ submit(): boolean
Lobby
+ getNearbyItems(): Item[]
Map
+ getNearbyItems(): Item[]
+getMap(): Map
Item
+ getItem(): Item
CreateItem
+ submit(): boolean

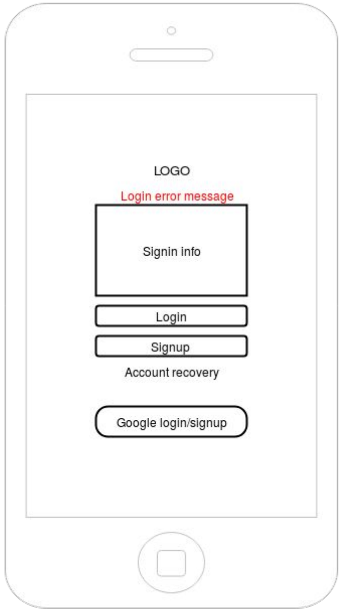
Ionic

- Framework for mobile application development with web technologies
  - Javascript
  - AngularJS
  - HTML5
- Free and open source
- Fully cross-platform
- Command Line Interface (CLI)
  - Developing
  - Testing on different platforms (IOS, Android, Windows, etc.)

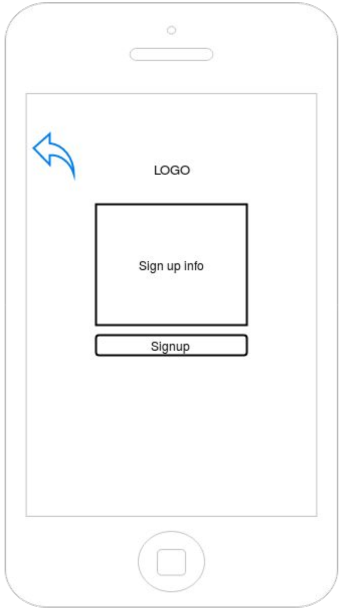
UI



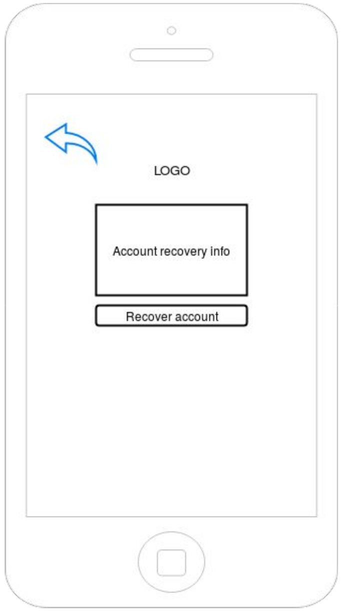
Main login



Login error

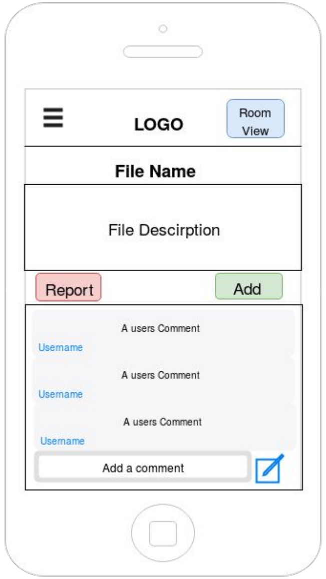


Sign up

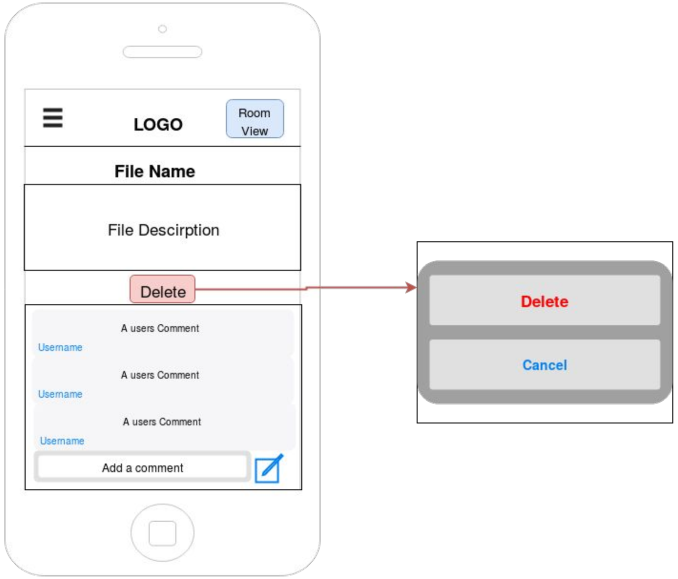


Account recovery

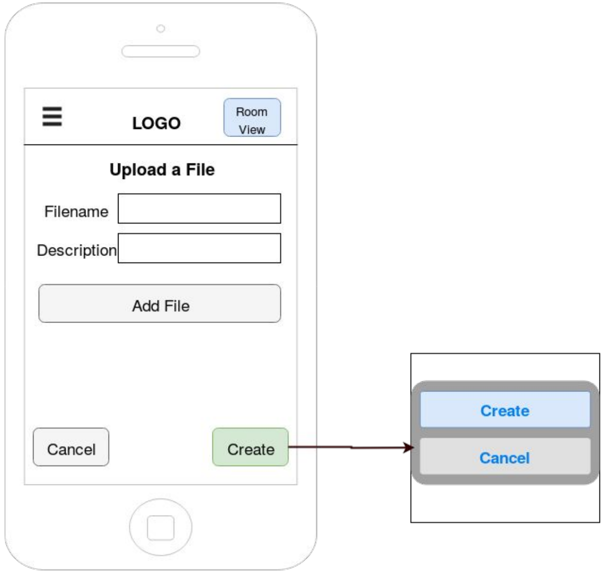
UI



User Room View

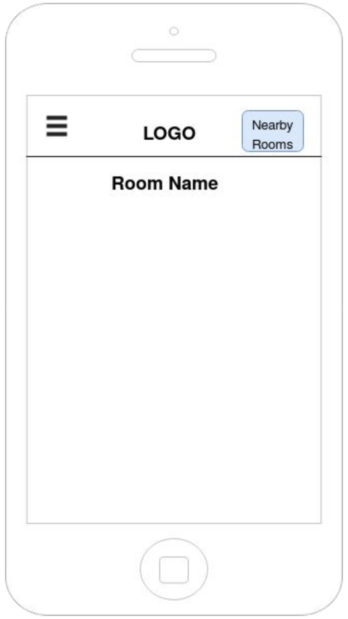


Creator Room View

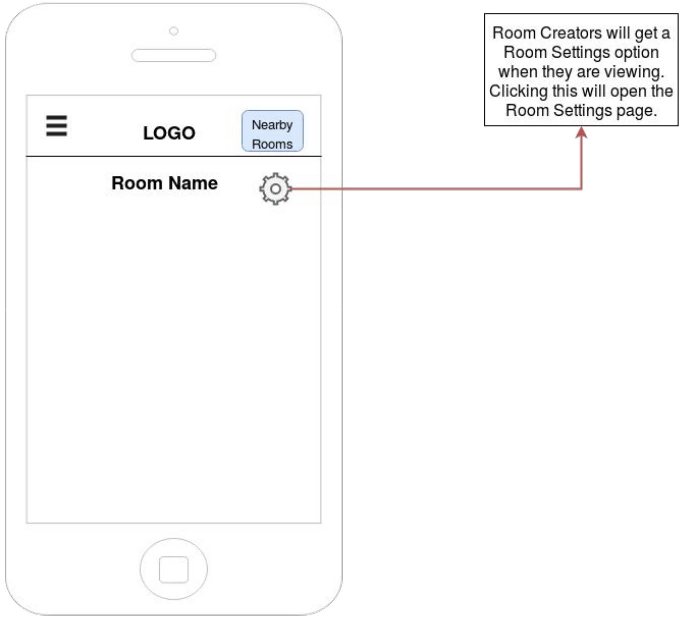


Create Room View

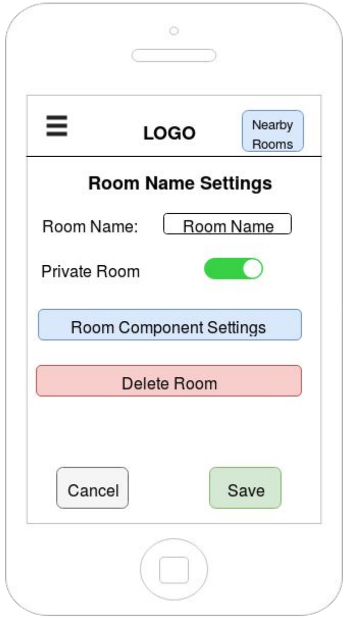
UI



User Room View

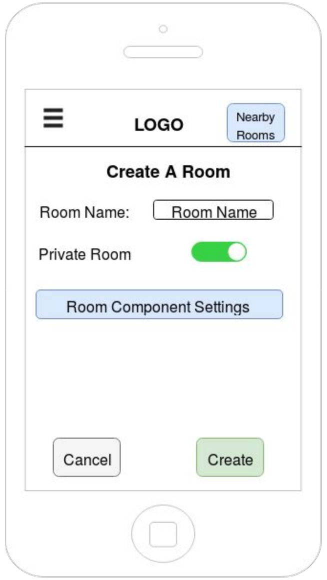

LOGO
Nearby Rooms
Room Name
Room Creators will get a Room Settings option when they are viewing. Clicking this will open the Room Settings page.

Creator Room View



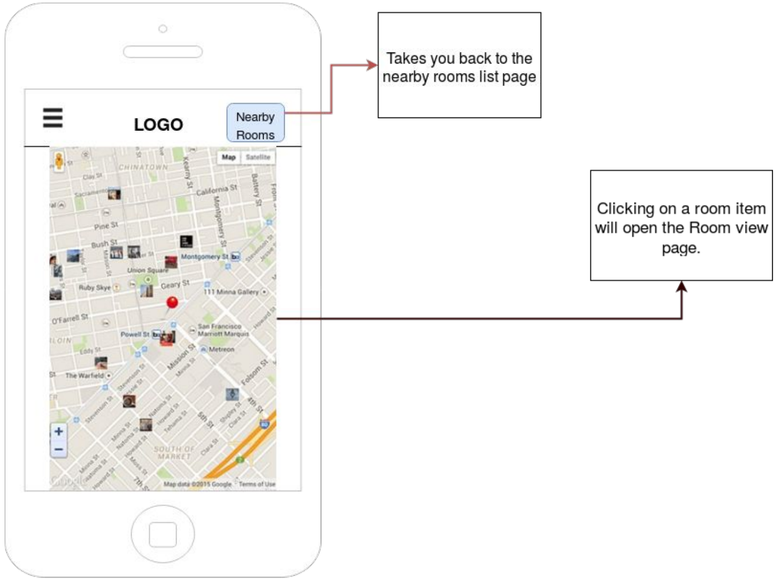
Room Settings View

UI



Create Room

UI

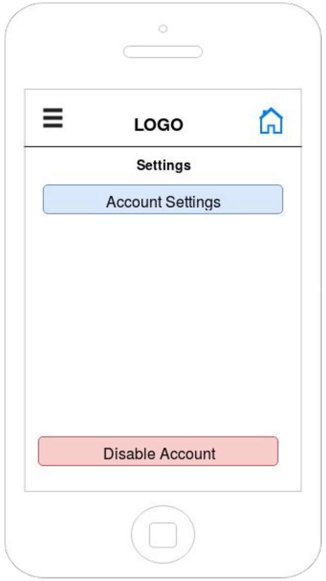


Takes you back to the nearby rooms list page

Clicking on a room item will open the Room view page.

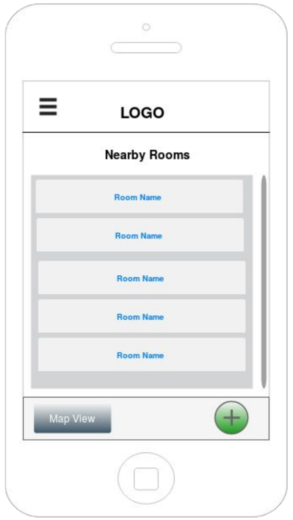
Map View

UI

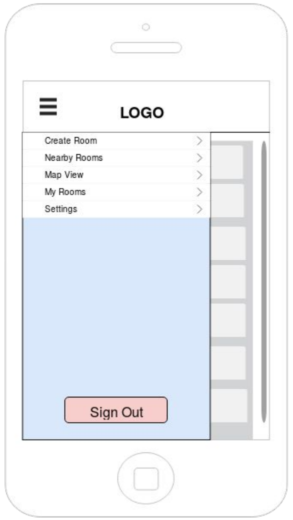


Settings

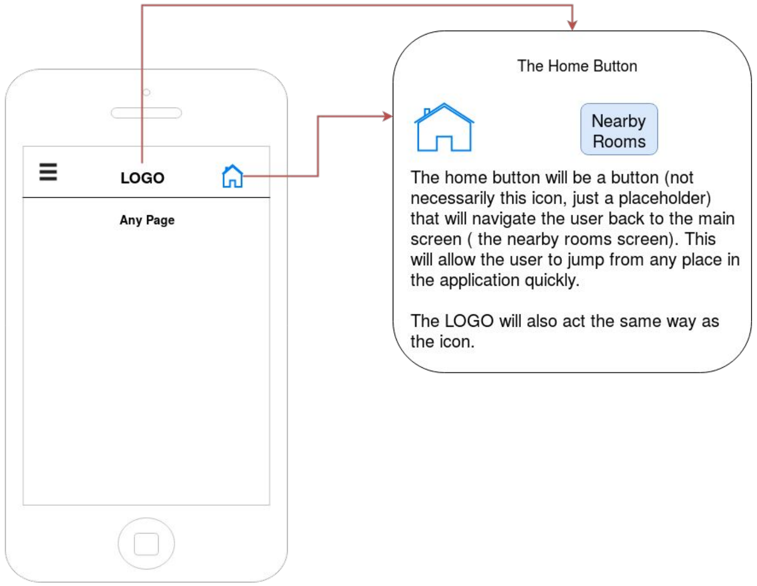
UI



Main



Side Menu



The home button will be a button (not necessarily this icon, just a placeholder) that will navigate the user back to the main screen (the nearby rooms screen). This will allow the user to jump from any place in the application quickly.

The LOGO will also act the same way as the icon.

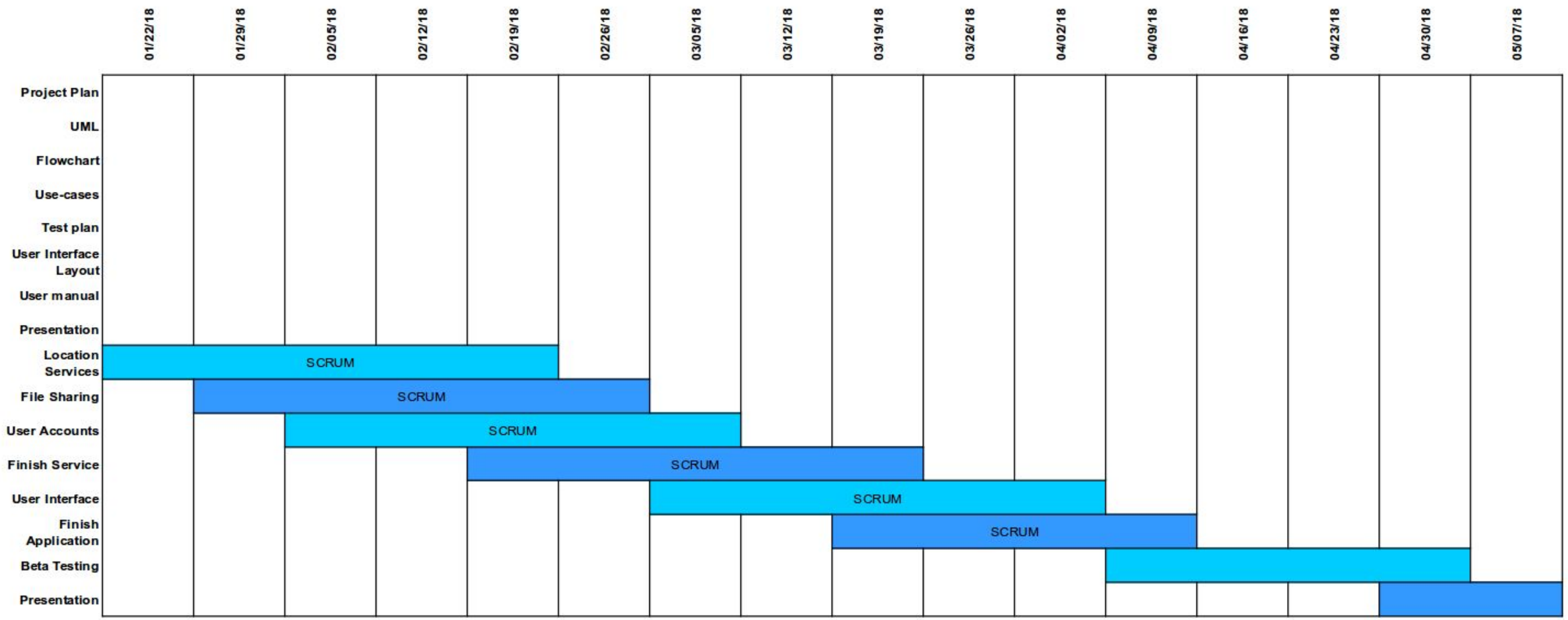
Home Button

Security

- Secure Socket Layer (SSL)
  - Establish an encrypted layer between server and client
- Authentication
  - JSON Web Tokens for user authentication
- Passwords
  - User passwords are hashed with salt **then** stored
  - Logins will compare hashes
- Two-factor authentication
  - Users will create an account with a phone number and a password
  - Text message verification code for authentication

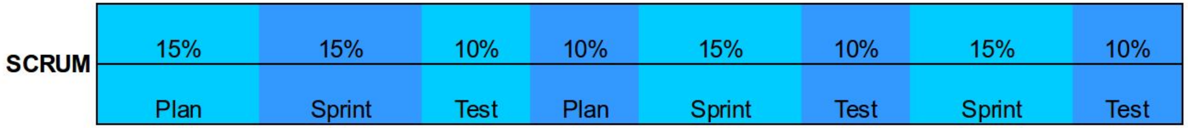
Plan

Timeline



	01/22/18	01/29/18	02/05/18	02/12/18	02/19/18	02/26/18	03/05/18	03/12/18	03/19/18	03/26/18	04/02/18	04/09/18	04/16/18	04/23/18	04/30/18	05/07/18
Project Plan																
UML																
Flowchart																
Use-cases																
Test plan																
User Interface Layout																
User manual																
Presentation																
Location Services	🟦	🟦	🟦 SCRUM	🟦	🟦											
File Sharing		🟦	🟦	🟦 SCRUM	🟦	🟦										
User Accounts			🟦	🟦	🟦 SCRUM	🟦	🟦									
Finish Service					🟦	🟦	🟦 SCRUM	🟦	🟦							
User Interface							🟦	🟦	🟦 SCRUM	🟦	🟦					
Finish Application									🟦	🟦 SCRUM	🟦	🟦				
Beta Testing												🟦	🟦	🟦	🟦	
Presentation															🟦	🟦

SCRUM



SCRUM	15%	15%	10%	10%	15%	10%	15%	10%
	Plan	Sprint	Test	Plan	Sprint	Test	Sprint	Test

- Plan
  - Goals for each sprint
  - Account for sprint duration
- Sprint
  - Develop to meet sprint goals
  - Working version should be complete by end of sprint
- Test
  - Test working version
  - Find and document bugs for next sprint

Testing

- Use / Develop tools for testing when necessary
  - Ionic testing
  - Postman
- Test for all applicable test cases
- Document results in a detailed test report
  - ID, date, time, tester
  - Test cases
  - Results / Comments
- Evaluate need for new test cases

References

RedPony github: https://github.com/Doliveraa/RedPony
Ionic documentation: http://ionicframework.com/docs/
MongoDB documentation: https://docs.mongodb.com/
Node.js documentation: https://nodejs.org/en/docs/
SCRUM: https://www.scrum.org/
Draw.io (used for all diagrams): https://www.draw.io/
Postman: https://www.getpostman.com/docs/