Project Phase II Proposal

WalkAbout by Ryszard Kilarski (U81-39-8560)

Contents

[1. Response to Phase 1 Comments 2](#_Toc349665718)

[2. Final Requirements 6](#_Toc349665719)

[Definite Requirements 6](#_Toc349665720)

[Nice To Do Requirements 7](#_Toc349665721)

[Screen Shots 7](#_Toc349665722)

[Waypoint List 7](#_Toc349665723)

[Add/Edit Form 8](#_Toc349665724)

[Waypoint Map Form 9](#_Toc349665725)

[3. Feature Cross-Reference 9](#_Toc349665726)

[4. Design 10](#_Toc349665727)

[Screen Transition Diagram 10](#_Toc349665728)

[Object Relationship Diagram 10](#_Toc349665729)

[Waypoint 11](#_Toc349665730)

[Image 11](#_Toc349665731)

[AppSettings 11](#_Toc349665732)

[Database 11](#_Toc349665733)

[User Interface Object Relationship Diagram (High-Level) 12](#_Toc349665734)

[User Interface Object Relationship Diagram (Detail) 12](#_Toc349665735)

[5. Partial Implementation of Risks 13](#_Toc349665736)

[Risks Associated With Requirements 13](#_Toc349665737)

[Riskiest Elements 13](#_Toc349665738)

[Risk 1: Time and Knowledge 13](#_Toc349665739)

[Risk 2: Google Maps Integration 13](#_Toc349665740)

[Risk 3: Photo App Integration 13](#_Toc349665741)

[Risk 4: Facebook Integration 14](#_Toc349665742)

[6. Schedule 14](#_Toc349665743)

# Response to Phase 1 Comments

* 1. **Summary Description**

When going on vacation, I like to keep track of where I’ve been, what photos I’ve taken at particular waypoints, and even create a map of where I’ve walked around.

Introducing: **WalkAbout**, the Android app that allows you walkabout and then talkabout your trip.

In this application, you will be able to create a “waypoint” (geotagged to your location), and then take (or add from the camera roll) as many photos as you like into it. You can add your own description to each waypoint, move photos between waypoints, and even “check in” to a particular area using Facebook. Additionally, you will be able to export your photos to your Camera Roll and also select a beginning and ending waypoint to produce a Google map of your path.

* 1. **I/O Examples**

|  |  |
| --- | --- |
| **Waypoint List**  The main user interface is a list of all the waypoints and the photographs within them. The user can scroll through all the waypoints and see all their photos.  Tapping on a particular photo displays the photo in full-screen mode. Swiping left or right in that mode scrolls you through all the photos in that waypoint.  The top of the waypoint displays the waypoint details: the description, date/time, and possibly location information.  Buttons display at the top of each waypoint to add additional images either from the camera roll or via the camera.  A button displays to take you to the viewpoint detail.  The Tools button accesses a settings page and other tools, such as “export all photos to camera roll”. |  |

|  |  |
| --- | --- |
| **Waypoint Add/Edit/Delete**  The Waypoint Add/Edit/Delete screen allows you to change the waypoint description, date/time, and even re-geotag/Facebook Check-in.  When adding a new waypoint, the date/time and geolocation will be automatically filled in for you.  (Not shown) This screen also contains the “Move Images To Another Waypoint” button. |  |

|  |  |
| --- | --- |
| **Waypoint Photo Move**  The Waypoint Photo Move screen allows you to select one or more photos in one waypoint, and move them to another. |  |

* 1. **Requirements**
     1. **Definite**
        1. Create a geotagged waypoint
        2. Add or take photos into the waypoint
        3. Ability to scroll and view all waypoints
     2. **Nice-to-do**
        1. Produce a Google Map from waypoints
        2. Facebook “check-in” integration
        3. Export waypoint to Facebook album (new or existing)
     3. **In the middle**
        1. Exporting waypoint photos to camera roll
        2. Edit existing waypoint (description, “check-in”, re-geotag…) or delete it
        3. Ability to move photos between waypoints
  2. **Android Features**

**Database Access**

*Requirements:* 1.3.1.1 and 1.3.1.2

The ability to create and save a waypoint and its photographs is central to the app.

**Camera**

*Requirements:* 1.3.1.2

The ability to access the camera to take photos is important when creating a waypoint. Alternatively, you should have the ability to access existing photos from the camera roll.

**GPS/Geotagging**

*Requirements:* 1.3.1.1 and 1.3.2.1 and 1.3.2.2 and 1.3.3.2

Access to the location services will allow attaching the GPS coordinates to each waypoint and to produce the Google map that outlines your path.

**Connectivity**

*Requirements:* 1.3.2.1 and 1.3.2.2 and 1.3.2.3 and 1.3.3.2  
Android connectivity (3G/4G/WiFi) will allow the app to access Facebook for check-in and uploading purposes, and allow the creation of the Google Map that will be the result of the path you took.

**Accelerometer/Orientation**

*Requirements:* 1.3.3.1 and 1.3.3.2 and 1.3.3.3

Changing the orientation on the device will take you to editing modes that allow you to edit your waystation information, move photos from one waystation to another.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **D** | **C-** | **C+** | **B-** | **B+** | **A** |  |
| **1. Clarity** | Disorganized or hard-to-understand | | Satisfactory but some parts of the submission are disorganized or hard to understand | Generally organized and clear | Very clear, organized and persuasive presentation of ideas and designs | Exceptionally clear, organized and persuasive presentation of ideas and designs | A |
| **2. Technical Soundness** | Little understanding of, or insight into material technically | | Some understanding of material technically | Overall understanding of much material technically | Very good overall understanding of technical material, with some real depth | Excellent, deep understanding of technical material and its inter-relationships | B+ |
| **3. Thoroughness & Coverage** | Hardly covers any of the major relevant issues | | Covers some of the major relevant issues | Reasonable coverage of the major relevant areas | Thorough coverage of almost all of the major relevant issues | Exceptionally thorough coverage of all major relevant issues | A |
| **4. Relevance** | Mostly unfocused | Focus is off topic or on insubstantial or secondary issues | Only some of the content is meaningful and on topic | Most or all of the content is reasonably meaningful and on-topic | All of the content is reasonably meaningful and on-topic | All of the content is entirely relevant and meaningful | A |

# 

# Final Requirements

## Definite Requirements

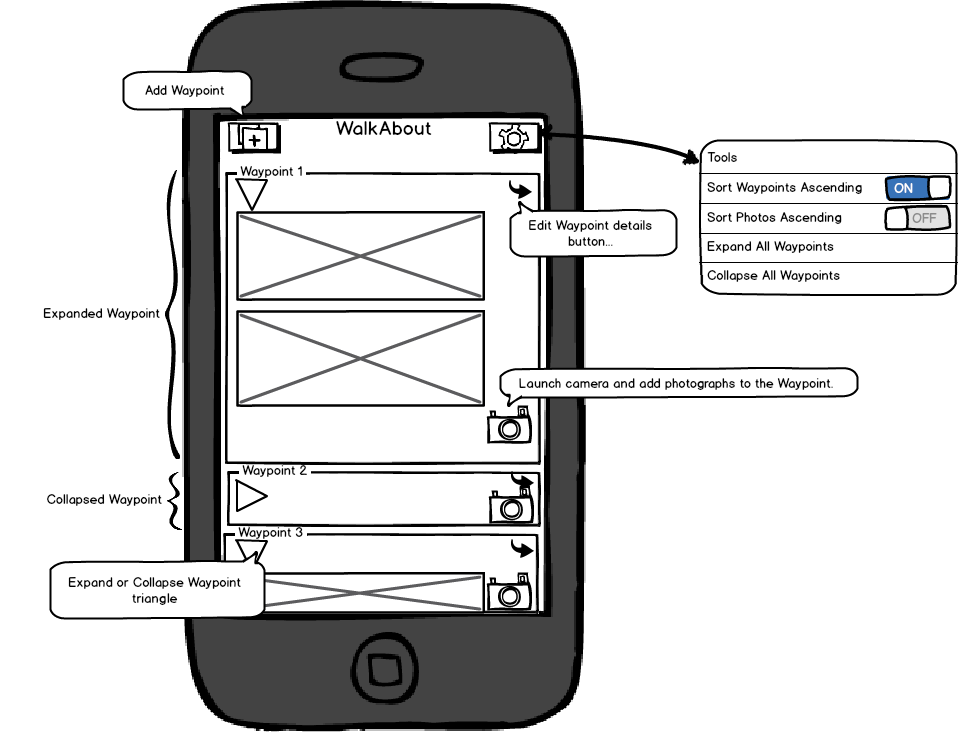
1. The App shall allow you to add a new Waypoint
   1. The Waypoint will be geotagged
   2. The Waypoint will have time, date, and a name.
2. The App shall allow you to edit an existing Waypoint
   1. Rename a Waypoint
   2. Re-geotag a Waypoint
   3. Change date/time
3. The App shall allow you to delete an existing Waypoint.
4. The App shall allow you to scroll and view Waypoints.
5. The App shall allow you to take photos with the device camera and add them to the Waypoint.
6. The App shall allow you to tap to view a photo fullscreen.
7. The App shall allow you to delete photos from a Waypoint.
8. The App shall allow you to produce a map from all the photos in a Waypoint.

## Nice To Do Requirements

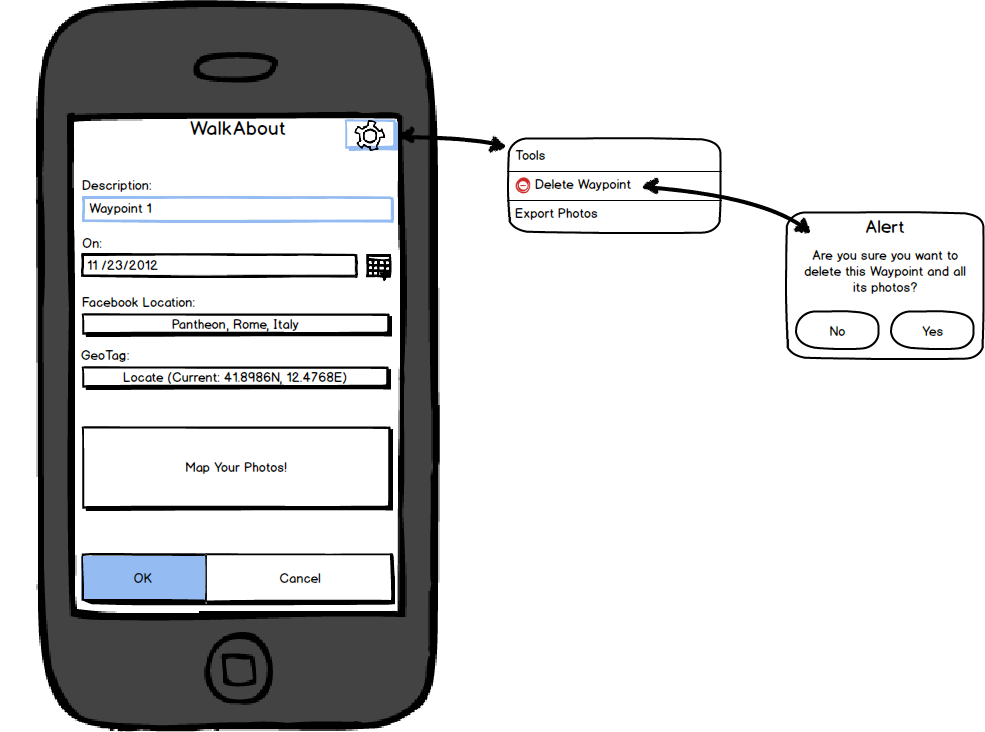
1. The App shall allow you to “check in” a Waypoint using Facebook.
2. The App shall allow you to export all photos in a Waypoint to a Facebook album.
3. The App shall allow you to export all photos in a Waypoint to the device album.
4. The App shall allow you to move photos between Waypoints.
5. The App shall provide the ability to change the Waypoint Order By setting.
6. The App shall provide the ability to change the Photo Order By setting.
7. The App shall allow you to expand/collapse individual Waypoints in the list, only displaying the name/datetime.
8. The App shall allow you to change the order of photos in the Waypoint.

## Screen Shots

### Waypoint List



### Add/Edit Form



### Waypoint Map Form



# Feature Cross-Reference

Provide a table with your requirement numbers D1, D2, ..., N1, N2, ... at the top, Android features at the left, and "x" in the table's body where applicable.

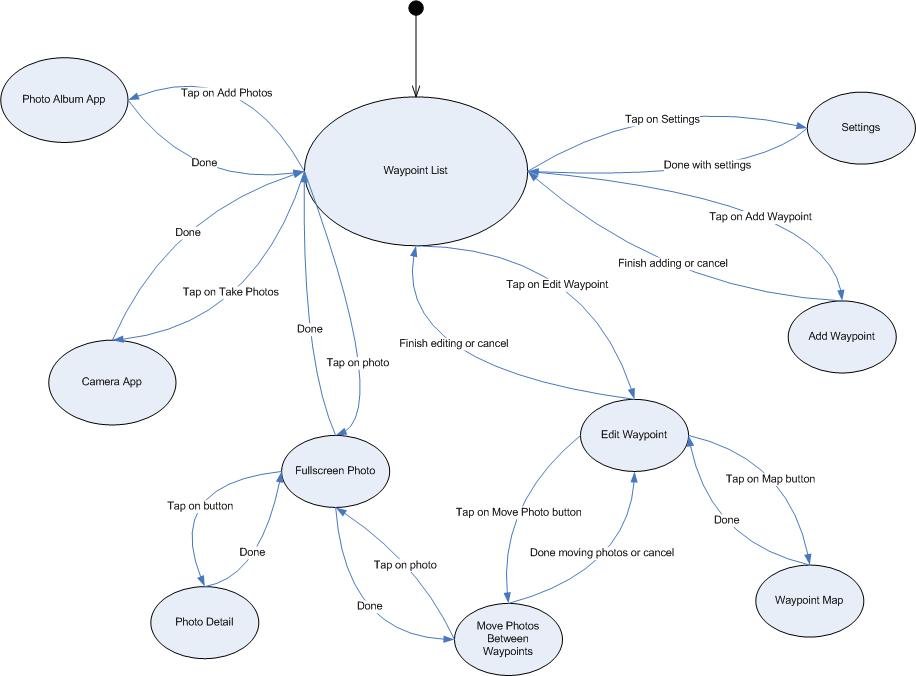
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Android Feature |  | Requirements | | | | | | | | | | | | | | | |
|  | D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 | N1 | N2 | N3 | N4 | N5 | N6 | N7 | N8 |
| Database Access | X | X | X |  | X |  | X | X | X | X | X | X |  |  |  | X |
| Camera |  |  |  |  | X | X |  |  |  |  |  |  |  |  |  |  |
| Local filesystem |  |  |  |  |  |  |  |  |  |  | X |  |  |  |  |  |
| GPS/Geotag | X | X |  |  |  |  |  | X |  |  |  |  |  |  |  |  |
| Data Connectivity |  |  |  |  |  |  |  | X | X |  |  |  |  |  |  |  |
| Accelerometer / Orientation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Touch/UI | X | X |  | X |  | X | X |  |  |  |  | X | X | X | X | X |
| Google Maps Integration |  |  |  |  |  |  |  | X |  |  |  |  |  |  |  |  |
| Facebook Integration |  |  |  |  |  |  |  |  | X | X |  |  |  |  |  |  |

# 

# Design

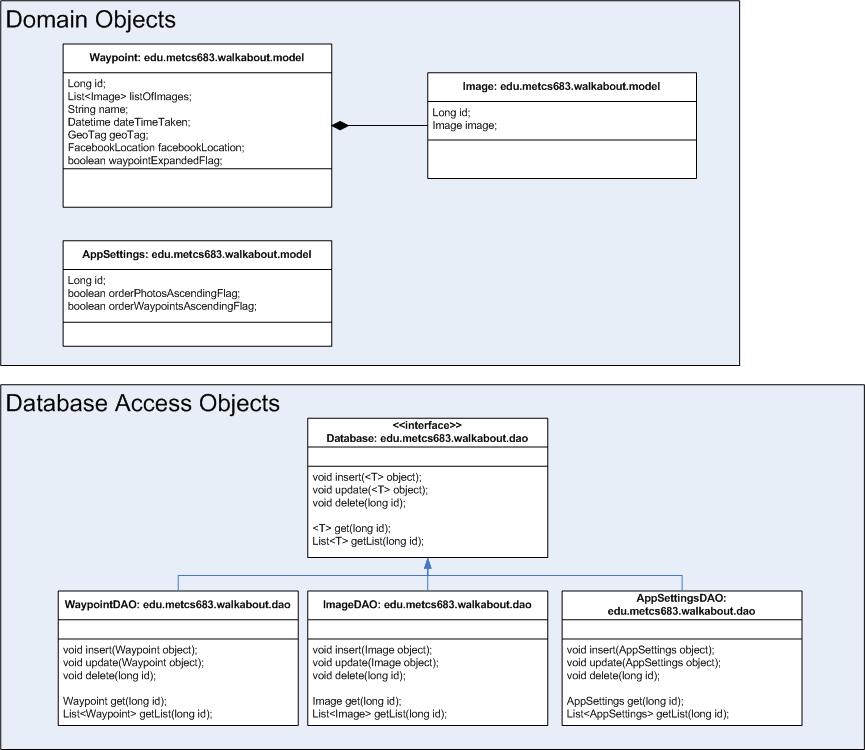
## Screen Transition Diagram

This diagram shows the flow of screens from the central Waypoint List.



## Object Relationship Diagram

This diagram shows the major domain and database objects. (All properties can be assumed to have their appropriate getter and setter methods.)



### Waypoint

The Waypoint is the main domain object, containing a list of images as well as “miscellaneous” data about the Waypoint. This includes

* The Waypoint’s unique ID in the database,
* The Waypoint’s name,
* The date/time it was created,
* Its geotag,
* Its Facebook location,
* Whether it is visually expanded or contracted by the user.

### Image

The image object contains the photograph itself and a unique ID.

### AppSettings

The AppSettings object contains the system settings for the application.

### Database

The Database interface contains the methods needed by all the DAO objects to load, save, and get a list of those objects.

#### WaypointDAO

This object contains the methods to load, save, and get a list of Waypoints.

#### ImageDAO

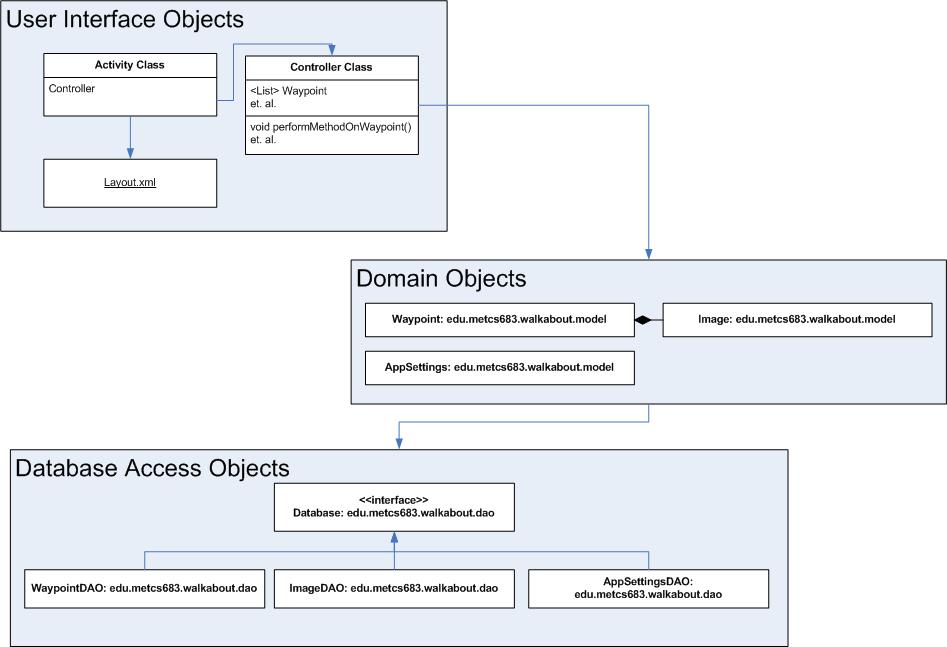
This object contains the methods to load, save, and get a list of images.

#### AppSettingsDAO

This object contains the methods to load, save, and get the Application settings object.

## User Interface Object Relationship Diagram (High-Level)

This diagram shows the high-level concept of how UI objects interface with the domain and DAO objects.



The general rules on how the objects relate are as follows:

* Each Activity references a corresponding xml file containing its layout.
* Each Activity has methods (and private classes) to enable user interface functionality.
* Each Activity also has a corresponding controller.
* The controller maintains the appropriate domain objects and exposes public methods that are used by the Activity to maintain the domain objects.
* The controllers do not perform any UI work, nor do they know anything about the database/DAO. They simply maintain the domain objects and provide methods for the UI to do its business.
* The domain objects privately have access to the database via the DAO objects. The rest of the application (the UI and controller) do not know about the database.

## User Interface Object Relationship Diagram (Detail)

This diagram shows the major UI objects and how they interface with the domain and DAO objects. (All properties can be assumed to have their appropriate getter and setter methods.)

# Partial Implementation of Risks

## Risks Associated With Requirements

Definite Requirements Risks (with higher risks first)

* D8: Integration with Google Maps
* D5: Integration with camera

Nice To Do Requirements Risks (with higher risks first)

* N1: Facebook “check-in” integration
* N2: Facebook album integration

## Riskiest Elements

### Risk 1: Time and Knowledge

#### Description

The highest risk in this project is in getting everything done and at the same time learning all I need to in order to do so. So there will be times where I’m reading chapters from the book (probably not in the syllabus sequence) and immediately implementing them in the code, in order to implement the requirements in the proper order.

#### Risk Retirement Plan

The best I can do is in the next section: create a schedule by week on what is needed to be done. Following the schedule, the time to implement the requirements can be managed so that the highest priority items can be delivered in a working product, even if many of the “nice to haves” fall off the list.

### Risk 2: Google Maps Integration

#### Description

I was unsure of the Maps integration and if I could get the software to do what I wanted it to do. That is, put a marker down on a map for each photo in a Waypoint. In this way, I can create a map for a Waypoint and show the user all the places they took a photo.

#### Risk Retirement Plan

This document describes how to create multiple markers on a map. I can even put images in the map itself!

<https://developers.google.com/maps/documentation/android/marker>

And here is a full tutorial on the MapView object. Except for the geo location used, it will be almost exactly what I can use for this project.

<https://developers.google.com/maps/documentation/android/v1/hello-mapview>

### Risk 3: Photo App Integration

#### Description

The App requires calling the photo application and allowing the user to take one or more photos which will then be imported into the Waypoint.

#### Risk Retirement Plan

I reviewed Chapter 10.4.1 of Android in Action and found almost exactly the code I need in order to get images into the application.

### Risk 4: Facebook Integration

#### Description

Similarly, I was unsure how to integrate the Facebook “check in” functionality into the app, in order to associate a Waypoint with a Facebook Check-in location.

#### Risk Retirement Plan

4b. Select the three riskiest elements. Using partial implementations to show evidence that you implemented enough to settle these risks. Ideally, these should cover all risks in your "definite" category. The "partial implementations" can be prototypes or even experiment code.

# Schedule

|  |  |  |  |
| --- | --- | --- | --- |
| Week | Phase | General Description | Milestone at end of week |
| Feb 25 | Design | Phase II Proposal completed. | Design is done.  **Delivery of Phase II Documents on Friday, March 1** |
| March 4 | Coding | Domain Objects and DAO infrastructure   * Domain objects |  |
| March 11 | Coding | Domain Objects and DAO infrastructure   * DAO objects and integration with domain objects | Infrastructure ready to fulfill all definite requirements. |
| March 18 | Coding | User interface and navigation.   * Forms and Intents | All Intents created, major navigation works. |
| March 25 | Coding | User interface and must-have requirements.   * Camera and database integration * Add functionality | Ability to take photos, add to Waypoint and save to database. |
| April 1 | Coding | User interface and must-have requirements.   * GPS and Google integration * Edit functionality | Ability to geotag and edit Waypoint details.  **Definite requirements met.** |
| April 8 | Coding | User interface and nice-to-have requirements.   * Export functionality * Move functionality * Facebook integration | Ability to export and move photos.  Ability to integrate with Facebook. |
| April 15 | Coding | User interface and nice-to -have requirements.   * Settings and sorting functionality | **Nice-to-do requirements met or retired.** |
| April 22 | QA | Bug fixing (and any spillover from nice-to-have requirements that were almost finished) |  |
| April 29 | Panic | Additional bug fixing | ZERO bugs! |
| May 6 | Spit & Polish | Final rush! | **Delivery on Thursday, May 9** |