

Color Coordinator

Benjamin Estremera, Paul Grubbs,
and Hampton Brewer

**Software Requirements Specification
Document**

Version: 1.02

Date:3/8/16

Change History:

Version #	Date of Revision	Description of Changes	Authors
1.00	2/11/16	Initial draft of Requirements Document	Benjamin Estremera, Paul Grubbs, Hampton Brewer
1.01	3/1/16	Fixed errors from class fault forms.	Benjamin Estremera, Paul Grubbs, Hampton Brewer
1.02	3/6/16	Adjusted feature descriptions	Benjamin Estremera

Table of Contents:

1.	Introduction	3
1.1.	Purpose	3
1.2.	Scope	3
1.3.	Document Overview	3
1.4.	Definitions	3
2.	General Description	4
2.1.	Product Perspective	4
2.2.	Product Functions	4
2.3.	User Characteristics	5
2.4.	Assumptions and Dependencies	5
3.	Requirements	5
3.1.	Functional Requirements	5
3.2.	Non-functional Requirements	6
4.	Diagrams and Analysis	6
4.1.	Use Case Diagram	6
4.2.	Class Diagram	7
4.3.	Activity Diagrams	8

1. Introduction

1.1 Purpose

This document describes the software requirements for an Android application named Color Coordinator. The goal of this specification is to help provide the designers, developers and maintainers with information about the project in the hope of increasing accuracy and efficiency toward the goal of creating the application.

1.2 Scope

The function of Color Coordinator is to help by assisting the user with their selection of clothes stored as an outfit. The application will require a picture to be taken of each piece of clothing and will then assess whether or not the outfit matches or not. It will either reject the outfit and ask for another, or it will accept it and inform the user.

1.3 Overview

The remainder of this document continues to explore the information related to Color Coordinator. The next section deals with important definitions. Then Chapter 2 goes into the description of Color Coordinator. Finally Chapter 3 will provide the specific functional requirements, external requirements, and the performance requirements of Color Coordinator.

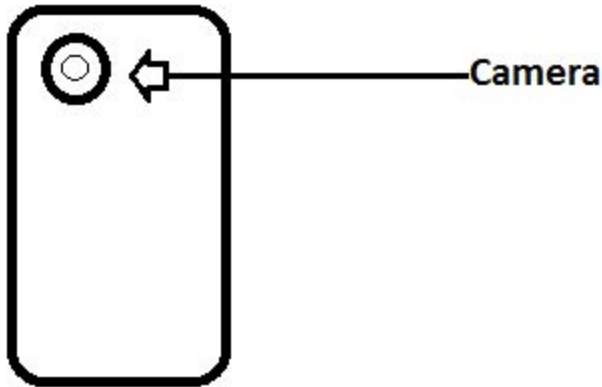
1.4 Definitions

- **Clothing Item**
 - This will be a user specific article of clothing.
- **Outfit**
 - This is the set of clothing items that have been selected by the user.
 - The outfit will be scanned to compare each set of clothing.
- **Stored Clothing**
 - Outfits already entered into the device will be saved.

- User can select them and then compare or take pictures for other outfits.

- **Camera**

- Required for app use.
- Camera refers to built in camera on android device.



- **Gui**

- Graphics used for the users interactions.
- It will allow the user to operate with the application.
- This is done by adding clothing or selecting outfit.

- **Match Threshold**

- How strict or not the user wants to be when choosing their outfit.

- **Match Rating**

- The value used to specify the relational value of clothing items from a 0-100 index.

- **User**

- Person utilizing the application.
- There are 3 different classes of users: users with no visual impairments, users with some visual impairment, and users with complete blindness.

2. General Description

2.1 Product Perspective

The Color Coordinator application is a self-contained application for the Android platform intended for version 4.0 (Ice Cream Sandwich) or later. The scope of this product only includes

client-side functionalities and is not tied to any other toolset. Additionally, all data will be stored locally.

2.2 Product Functions

The major functions of this application are as follows:

- Have a user-friendly GUI in addition to being accessible to users with TalkBack (Required)
- Allow the user to take pictures of clothing items (Required)
- Create a new outfit, starting with a single piece of clothing (Required)
- Add other articles to that outfit and assign a match rating from 0-100 (Required)
- Allow the user to adjust the match threshold to be more or less strict (Required)
- Recommend colors that could be added to complete an outfit for users without visual impairments (Required)
- Outfits must be able to be saved locally to allow the user to utilize them while shopping. (Possible feature)
- Allow the user to include pictures taken elsewhere in the outfit (Possible feature)
- Recognize whether an article has a pattern or does not (pattern type doesn't matter) (Future work)

2.3 User Characteristics

The Color Coordinator app is intended for use by 3 different classes of users: users with no visual impairments, users with some visual impairment, and users with complete blindness. The application should be usable by the first two classes of user without any training. For blind users, some instruction will be required to assist them in taking pictures of the clothing, given through either TalkBack or an audio recording.

2.4 Assumptions and Dependencies

One assumption about the product is that it will be used on a mobile device with enough storage space to store the pictures taken by the application. If the user does not have enough space to take any pictures, the product will not be able to function properly.

Another assumption is that the camera on the user's device adequately captures the correct colors. If there is a problem with the camera and it is not capturing the true color of an article of clothing, the product cannot be expected to give an appropriate match rating. The system will assume that outfit pictures are taken in similar conditions for comparison.

3. Requirements

3.1 Functional Requirements

1. The user should be notified by an audial cue and an accompanying visual cue when the system has finished computing a match rating of the outfit.
2. The user should be reminded to put clothing and only clothing in the target box when taking pictures so that background objects are not included in the analysis.
3. The matching percentage of an outfit should always be displayed on the outfit selection screen.
4. The pictures taken by the users should be displayed on the outfit selection screen.
5. The system should display a list of colors in the outfit along with the matching percentage once the user requests the matching information.
6. The system should recognize if multiple pieces of clothing contain different patterns and recommend against wearing both.
7. The system should display an alternate color suggestion for the user if an item of the outfit is not matchable with the other items and prompt the user to see if they want to remove that item.
8. The user should have the ability to turn off accessibility prompts as a setting.
9. The user should be able to turn off color recommendations as a setting.
10. Users can locally save each outfit with a user-given name and can access them from the main menu.
11. Each outfit can be edited and modified later.
12. Users will be able to add items with an “add item” button on the outfit view screen.
13. Users will be able to remove items with a “remove item” button on the outfit view screen.
14. Tutorial should be provided on first use of app.

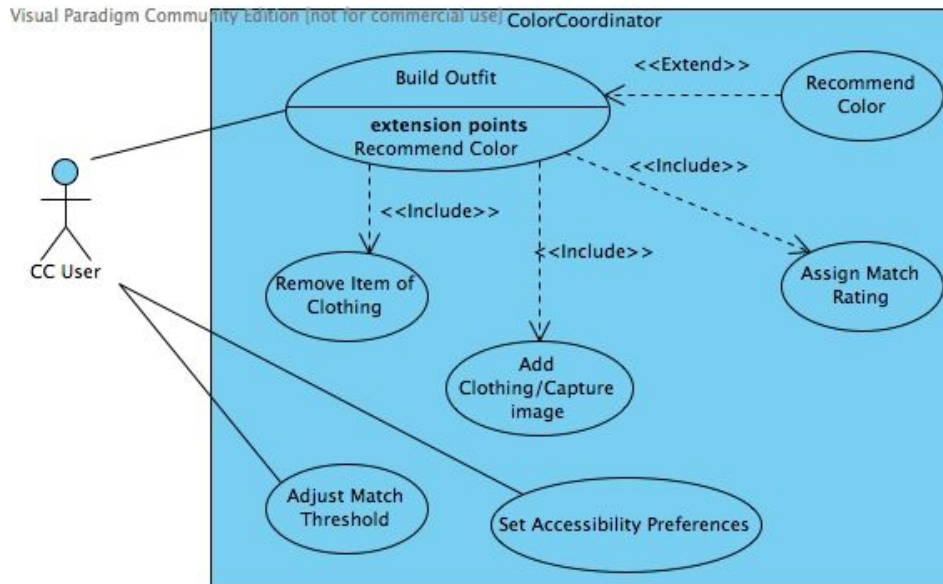
3.2 Non-Functional Requirements

1. The system must have a working camera in order to be able to use this application.
2. The system must have at least Android API 14 (Ice Cream Sandwich) in order to use the application.
3. Prompts instructing or alerting the user and any non-textual controls should be able to be read by the system’s TalkBack functionality.
4. The current outfit state should still be available if the app has become out of focus or view as long as it is still running in memory.
5. The colors used in the user interface should be readable even when high-contrast mode is enabled in the system settings.

6. The application should be able to compute a matching rating for at most 10 items.
7. The matching rating should be updated within a second of adding a clothing item.
8. The system will not allow pictures to be taken when there is no storage space on the device.

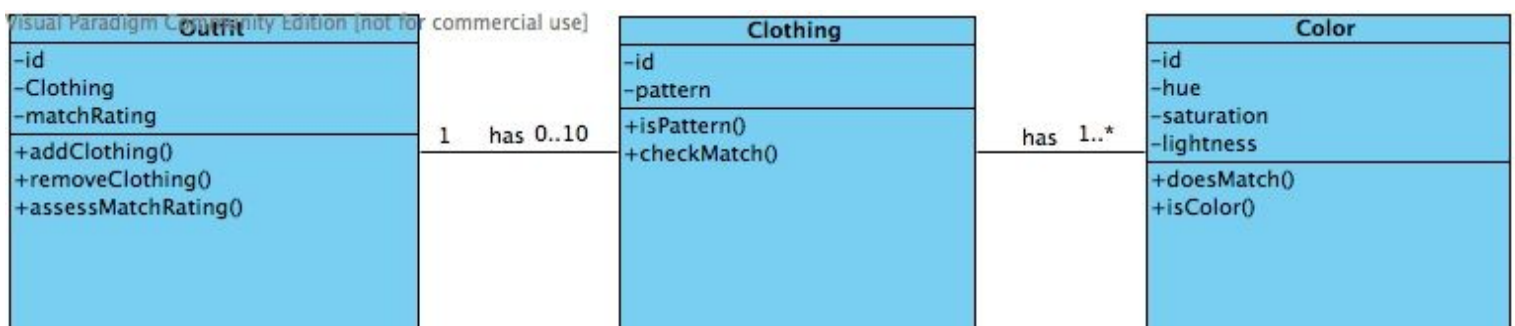
4. Diagrams and Analysis

4.1 Use Case Diagram



- The majority of use cases are found under the Build Outfit use case, as this is the core functionality of the app, where the user composes an outfit and is presented a match rating
- The user will be able to interact with the app to both add and remove articles of clothing. When adding, they will be instructed to capture an image of the article of clothing.
- The system should be able to both assign a match rating and recommend a color based on the clothing that the user adds to the outfit
- Outside of the build outfit case, users should be able to both adjust their match threshold to be more or less strict and adjust their accessibility settings including toggling the color recommendation feature

4.2 Class Diagram

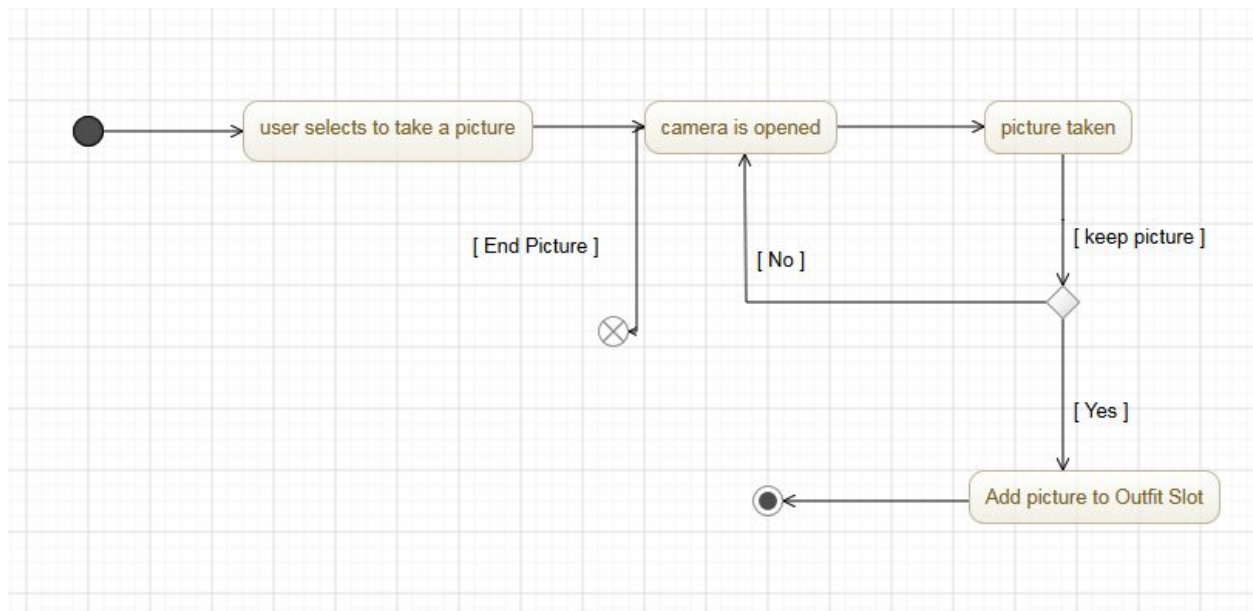


Above are three of the primary classes needed for this system, along with some basic methods and attributes required. Their descriptions are as follows:

- All objects are to have a unique id.
- Colors are to be defined in the HSL color space, represented by the three attributes of hue, saturation, and lightness. The method `doesMatch()` will determine if one color object matches another. In addition, the method `isColor()` will be used to discern if one color object is practically identical to another. A color will be declared equivalent to another if they lie within a relatively close proximity on a color wheel.
- Clothing objects will be defined primarily by the colors that compose them, shown by the “has” relationship above. A boolean value named `pattern` will be set to true when a clothing object is composed of multiple different colors, and `isPattern()` will return this value. Additionally, the `checkMatch()` method will compare the array of colors in one clothing object to another to return if they match or not.
- Outfit objects will be composed of Clothing objects, one of which being the baseClothing object that the outfit is built around. Clothing objects will be added or removed through their respective methods. A `matchRating` attribute will maintain the current match rating from 0-100 for the current outfit. This value will be reevaluated and updated with every call to `assessMatchRating()`.

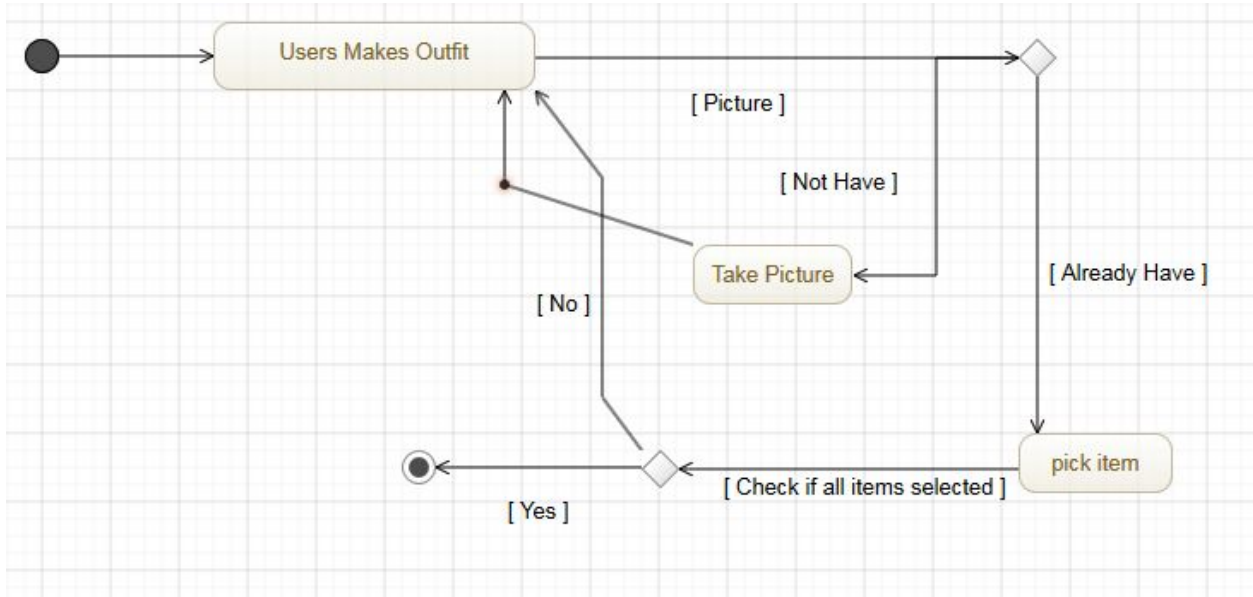
4.3 Activity Diagrams

1. Take Picture



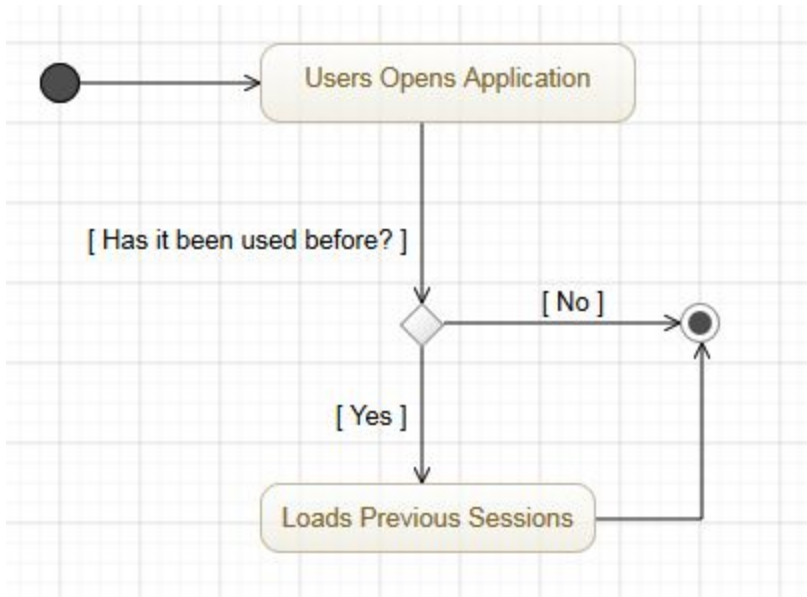
- The take picture activity follows the actions of the user taking a picture to add to the outfit.

2. Outfit Selection



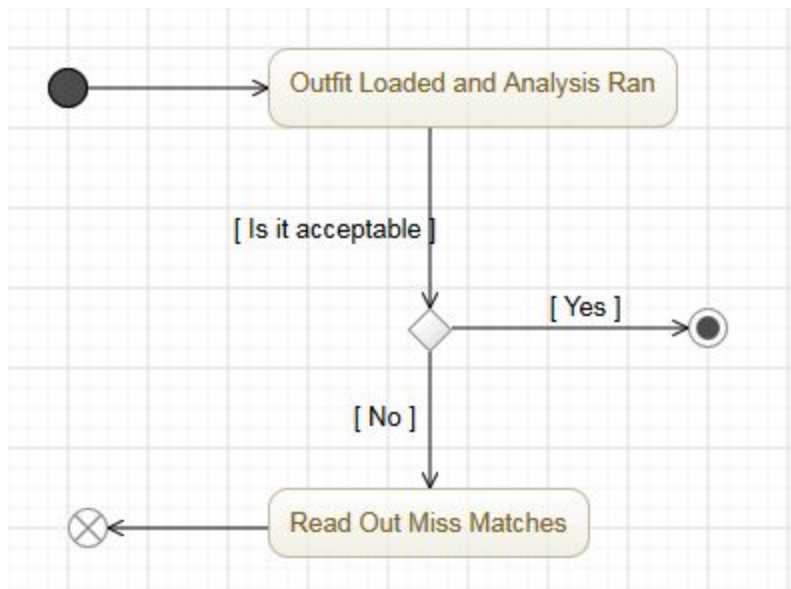
- The Outfit Selection activity deals with the process of the user interacting with the outfit setup process.

3. Open Application



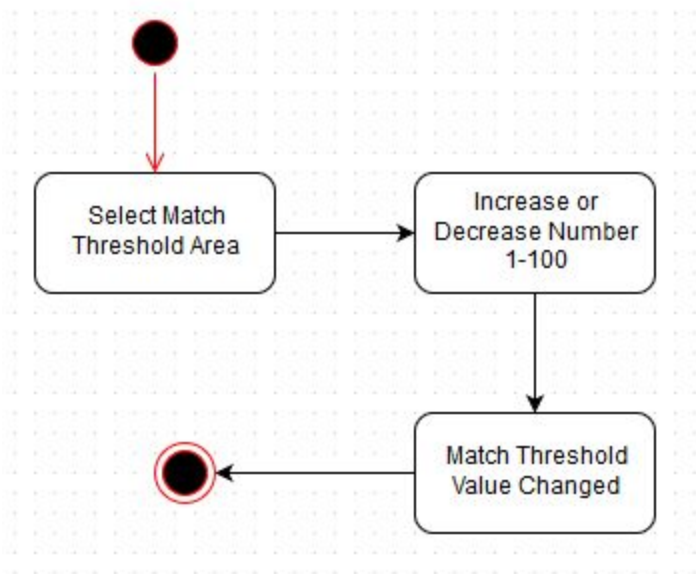
- Open Application is the handling of the initialization of Color Coordinator.

4. Compute Outfit Results



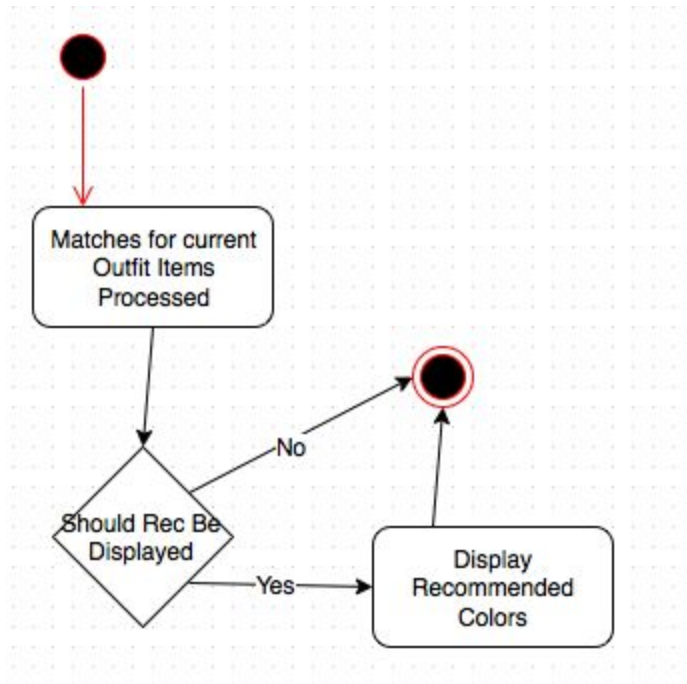
- This activity displays the testing of the Outfit to determine if it matches or not.

5. Adjust Match Threshold



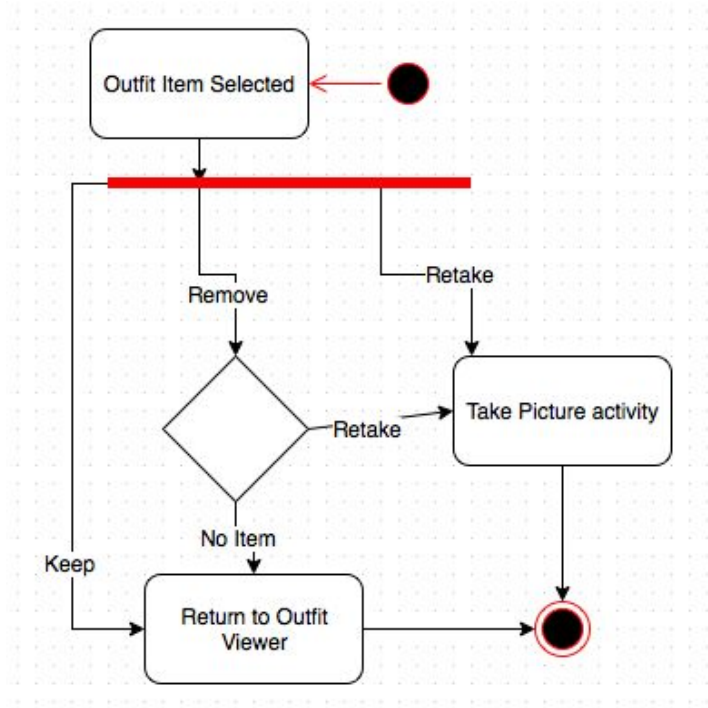
- The activity shows how the user can change the match percent of the overall outfit.

6. Recommend Color For Next Item



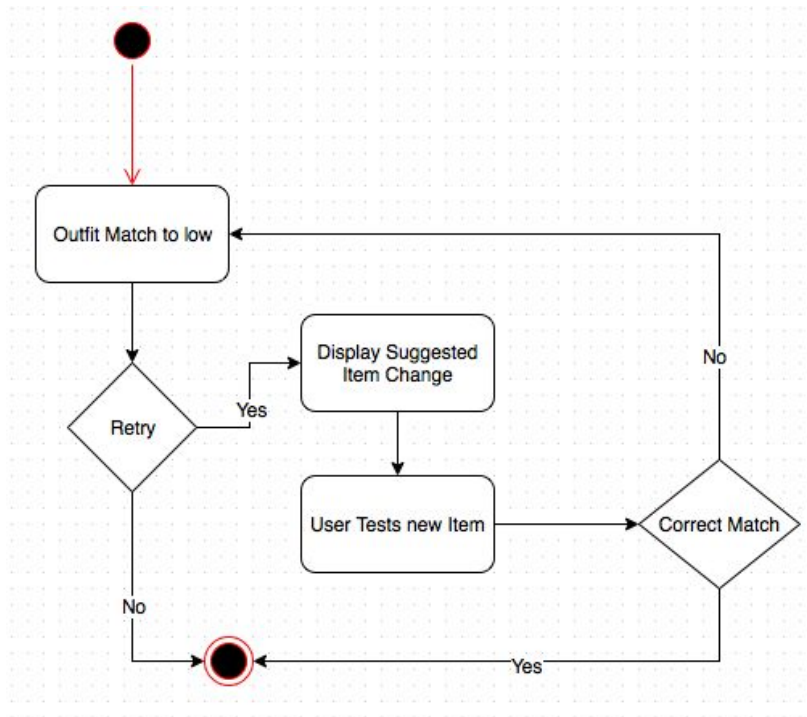
- Activity for recommending a color to the user as a suggestion for what would match well with other Outfit items.

7. Remove Outfit Item



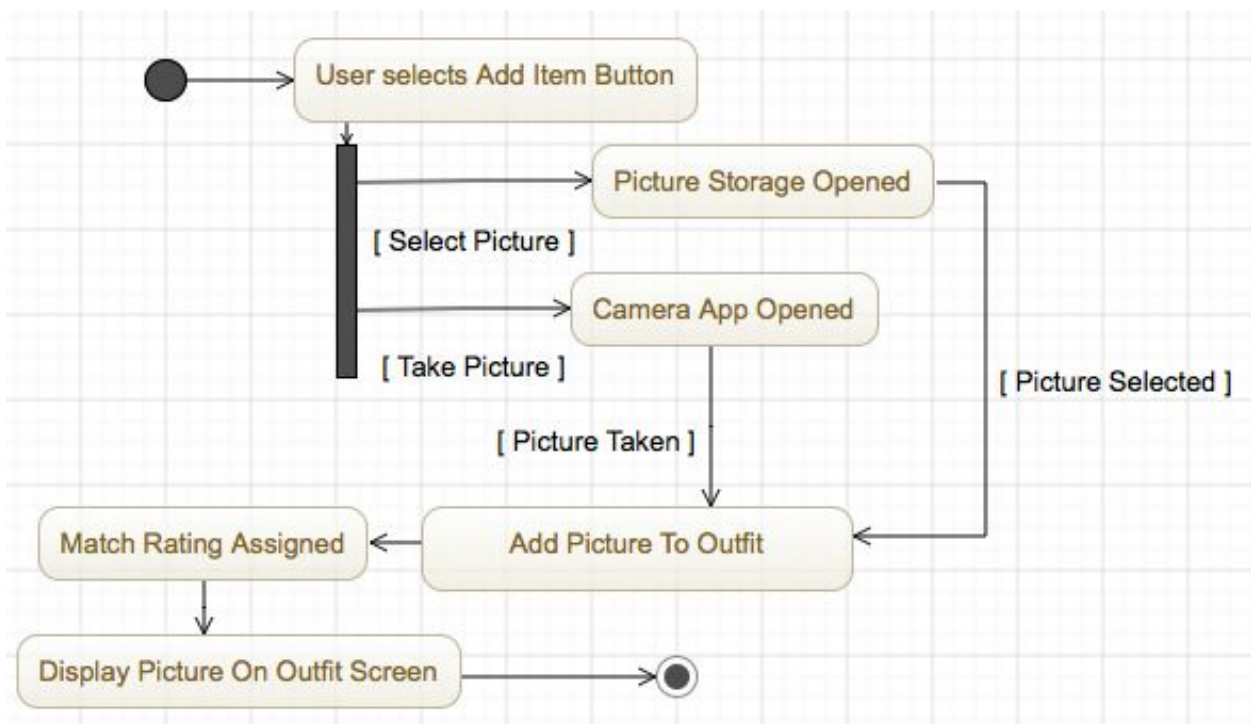
- Activity to remove an item from the Outfit.

8. Alternate Item Suggestion



- If the color did not match enough the user is given the chance to change out the item in the Outfit.

9. Add Outfit Item



10. Build Outfit

