Color Coordinator

Benjamin Estremera, Paul Grubbs,
and Hampton Brewer

Software Design Specification Document

Version: 1.02 **Date:4/5/16**

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
1.03	3/31/16	Design changes added	Hampton Brewer
1.04	3/31/16	Activity Diagrams edited	Hampton Brewer
1.05	4/4/16	Clarify Design	Hampton Brewer, Benjamin Estremera, Paul Grubbs
1.06	4/5/16	Sequence Diagram Updates	Hampton Brewer, Benjamin Estremera, Paul Grubbs

Table of Contents:

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.

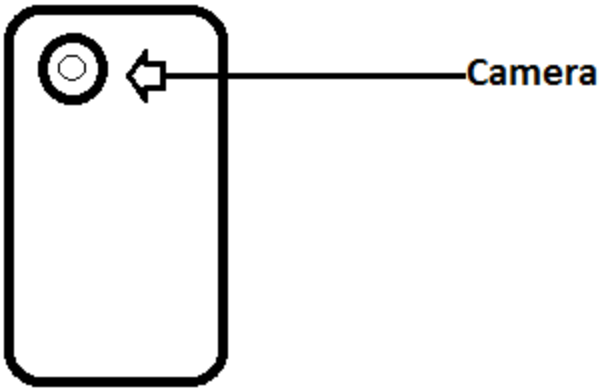
1.4 Definitions

- **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.

- **ListView**
    - The area that each clothing item will be stored in.
    - This will be viewable by the user and displays:
        - The image of the clothing item that was taken.
        - Colors of the clothing item.
        - Name of the clothing item.
        - A checkmark showing if the clothing item is selected or not.

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)
- Display each clothing item in an easy to understand listview.(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)
- Allow use to toggle sounds. (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. The user will also be able to set some preferences in the Options activity.

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 through a listview.

5. The listview must display all vital information to the user in a basic and well abstracted way.

6. The system should display a list of colors in the outfit along with the matching percentage once the user requests the matching information.

7. The system should recognize if multiple pieces of clothing contain different patterns and recommend against wearing both.

8. 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.
9. The user should have the ability to turn off accessibility prompts as a option.

10. The user should be able to turn off color recommendations as a option.

11. The user can turn off sound and set a different match threshold as a option.

12. Users will locally save the outfit and can access it from the main menu.

13. The outfit can be edited and modified later when accessing the application.

14. Users will be able to add items with an “add item” button on the outfit view screen.
    - Adding items will add all clothing items to a listview on the Outfit Screen.

15. Users will be able to remove items with a “remove item” button on the outfit view screen.
    - The remove item button will be linked to the listview.
    - Any item that is currently selected in the listview will be deleted.
    - This creates an easy way for the user to delete one or all listview items.

16. Tutorial should be provided on first use of app.

17. The Options and Outfit activity must each contain a back button that allows the user to access the Main activity.

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.

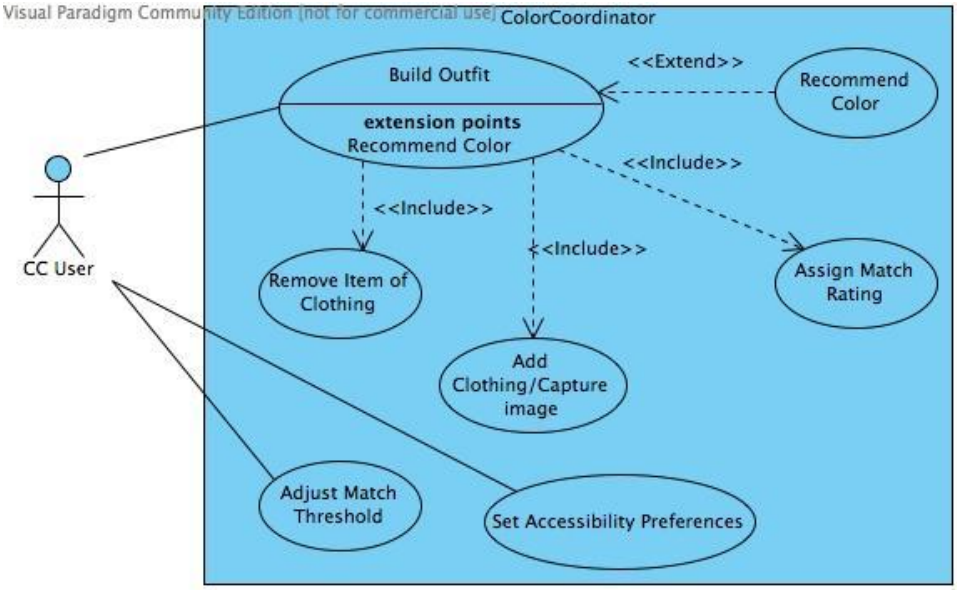
8. The system will not allow pictures to be taken when there is no storage space on the device.

9. Each activity will have the same background and style of buttons.

10. Each Image should be stored in a directory called “ColorCoordinator”.
    - Each image will be incrementally named colorImage_01….

4. Diagrams and Analysis

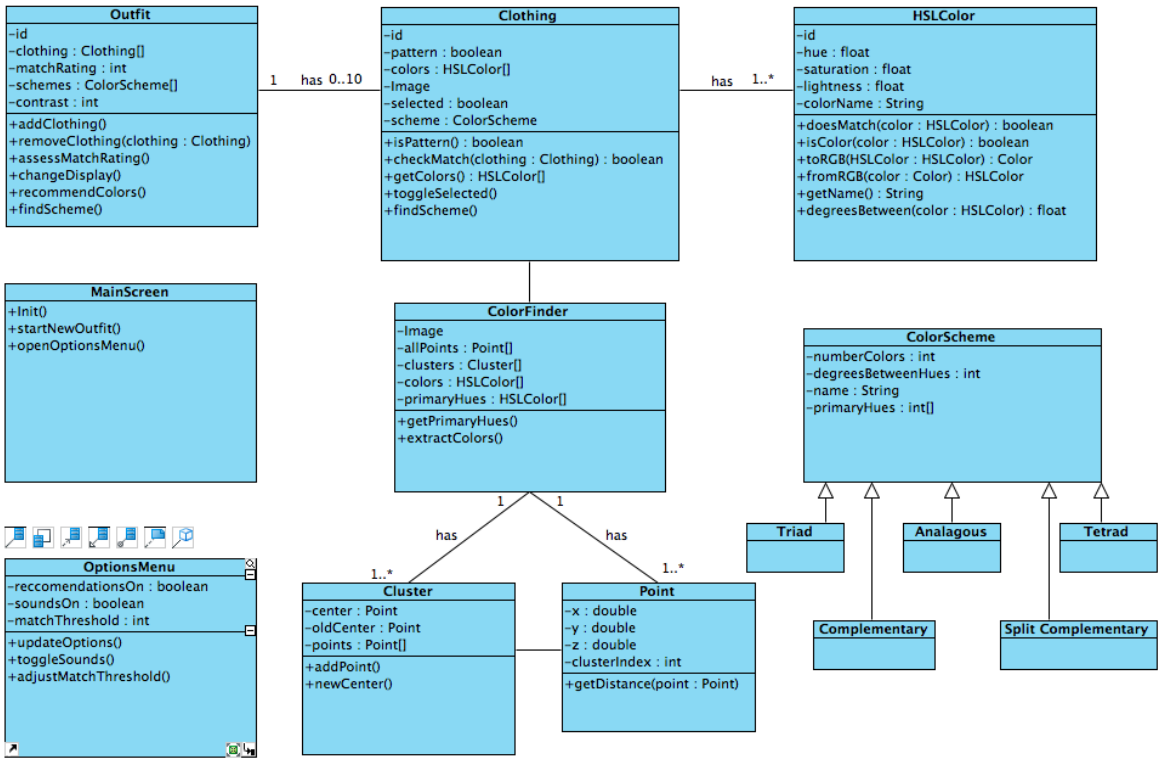
4.1 Use Case Diagram



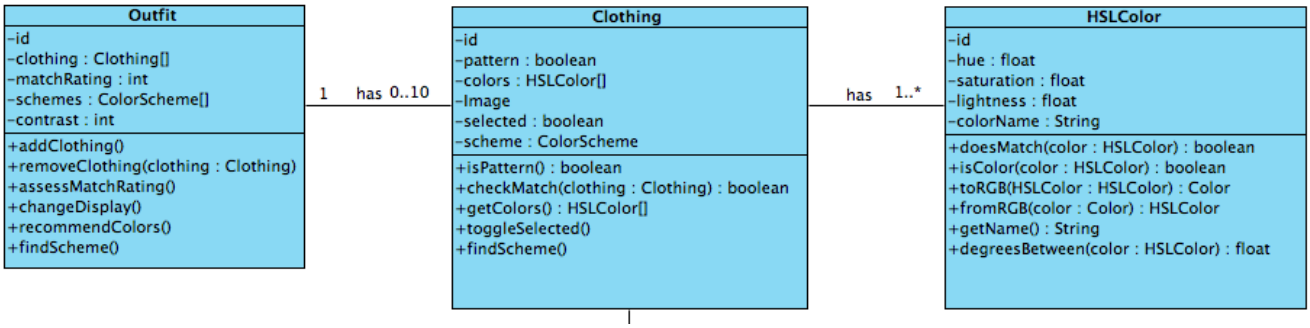
- The majority of use cases are found under the Build Outfit use case, as this is the core functionality of the app, where the user composes an outfit and is presented a match rating

- 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

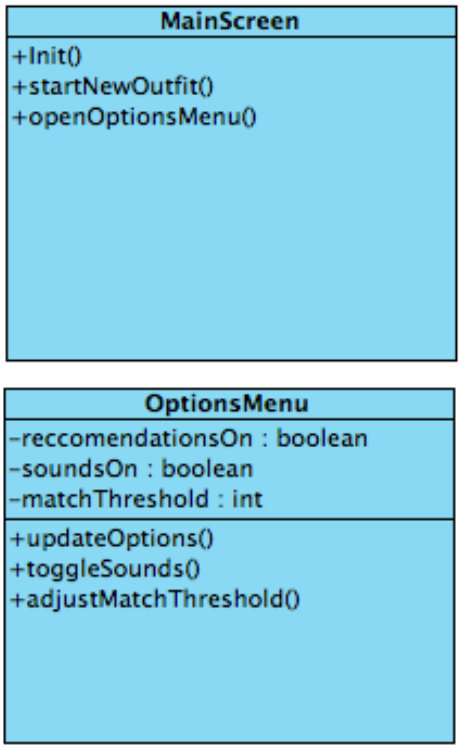


Above is the complete high level class diagram for the ColorCoordinator application. The classes will be broken down into groups so that they can be more easily seen and explained.



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

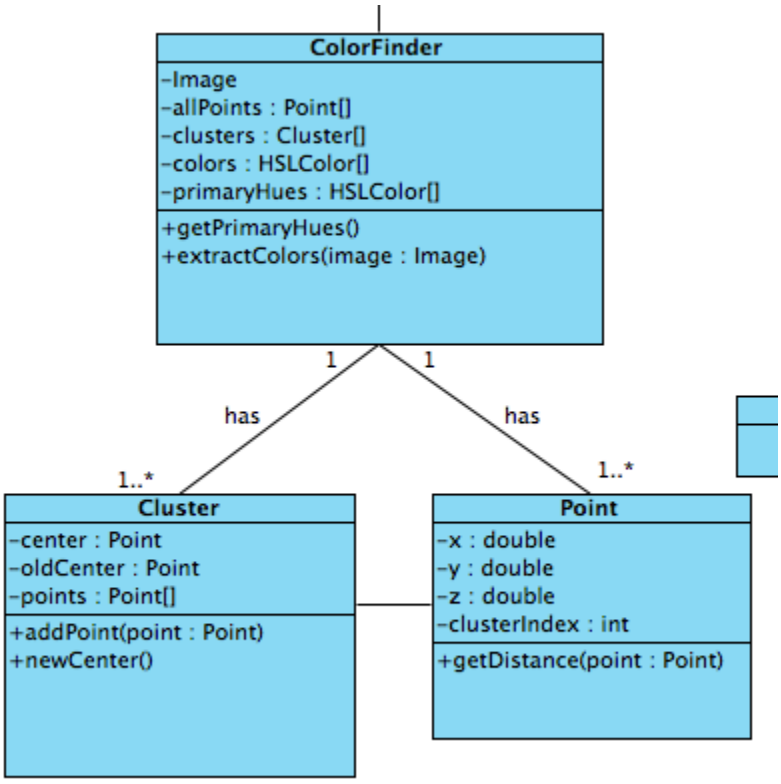
- All objects are to have a unique id.
- Colors are to be defined in the HSL color space, represented by the three attributes of hue, saturation, and lightness. The methods toRGB() and fromRGB() will convert between RGB color objects and HSLColor objects. The method doesMatch() will determine if one 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. This proximity can be calculated with degreesBetween(). Finally, getName() will return an approximate name of the color.
- 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. FindScheme will attempt to find the color scheme of an individual clothing object. Finally, toggleSelected() will inverse the selected boolean, which indicates if a clothing object should be included in the outfit match rating
- Outfit objects will be composed of Clothing objects. Clothing objects will be added or removed through their respective methods. The method findScheme() attempts to find the color scheme of the entire outfit. RecommendColor() will then return color recommendations based on this scheme. ChangeDisplay() changes the format that an outfit is displayed in. 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().
    - goBackToMain() is a method that will return the user to the Main menu, our current implementation of the Outfit class does not reflect this, but it should be there and will be added soon



Two additional classes are the MainScreen class and the OptionsMenu class.

- The MainScreen is the first activity in the app that will be accessed by the user.
    - From the MainScreen there will be two buttons added.
    - Outfit
        - This button uses onClick() to call the startNewOutfit() method and opens the outfit activity for the user.
    - Options
        - This button uses onClick() to call the openOptionsMenu() method and opens the options activity for the user.
- In the OptionsMenu the user will be able to access and edit options such as sound, match threshold and recommendations.
    - Recommendations
        - This will be a toggle button. The button will be on if the user wants recommendations displayed and off if the user does not want them displayed. It is ran from the updateOptions() method each time the button is toggled.
    - Sounds

        - This will be a toggle button. The button will be on if the user wants sounds on or if the user does not want sound. It is ran using the toggleSounds() method each time the button is toggled.
    - Match Threshold
        - The match threshold is a sliding bar that allows the user to choose between the values of 0 - 100. Each time the user changes the value the adjustMatchThreshold method will be called to change the threshold to the desired range.
- goBackToMain() is a method that will return the user to the Main menu, our current implementation of the Options class does not reflect this, but it should be there and will be added soon

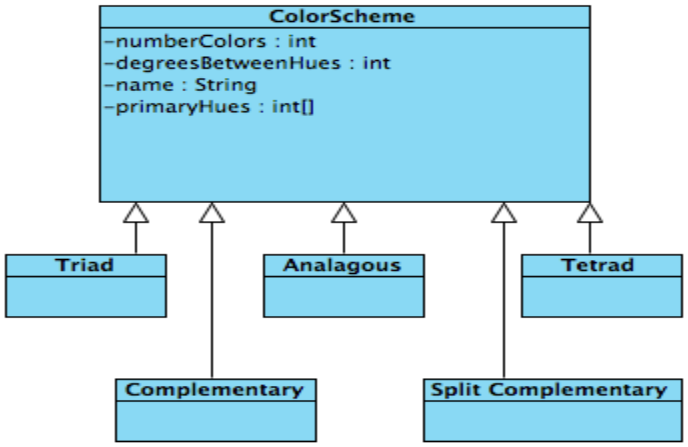


The following classes are related to the k-means process of extracting the primary colors from an image:

- Point is a basic implementation of a point in 3 dimensional space. In addition to the x, y, and z coordinates, there is also a clusterIndex attribute. This index relates to the cluster that the point is being associated with. Finally, the only method in the Point class is a getDistance() method that returns the euclidean distance between the current point and some other point.
- Cluster is an implementation of k-means clustering. Each cluster will be initialized with one point, and will have its center set to that point. Points will be added to the points

array through addPoint(). At some point during the k-means process, a new center will need to be calculated between all the points that have been added, which is accomplished through newCenter(). This method also sets the oldCenter to the current center before updating the center value.

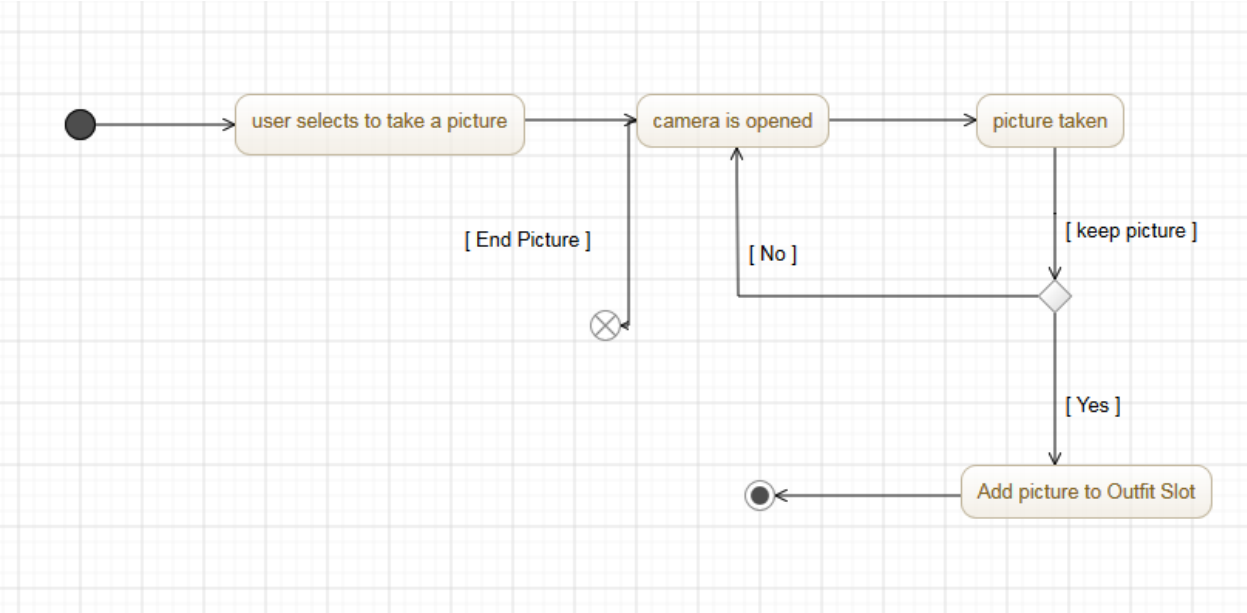
- ColorFinder handles the role of extracting the primary colors of an image. By calling extractColors() with a given Image file, it will create point objects for every rgb pixel and create k number of clusters from k random pixels. It then groups these



- The ColorScheme class will be the main class for getting and displaying the color ranges of the user taken images. It also plays a role in calculating the match rating, by determining if a color would fit the outfit's current scheme
    - It takes input as colors.
        - Then will display the RGB color codes from the image.
        - Also it will be able to display a named color for the user or named colors if more than one.
    - This class interacts with the Outfit class. Whenever addClothing() is called or removeClothing() is called, the colors of the current Clothing items will be processed and that Match Rating is then displayed to the user.
        - User will be updated if Match Rating falls below requested values.
        - Else Match Rating is only displayed.

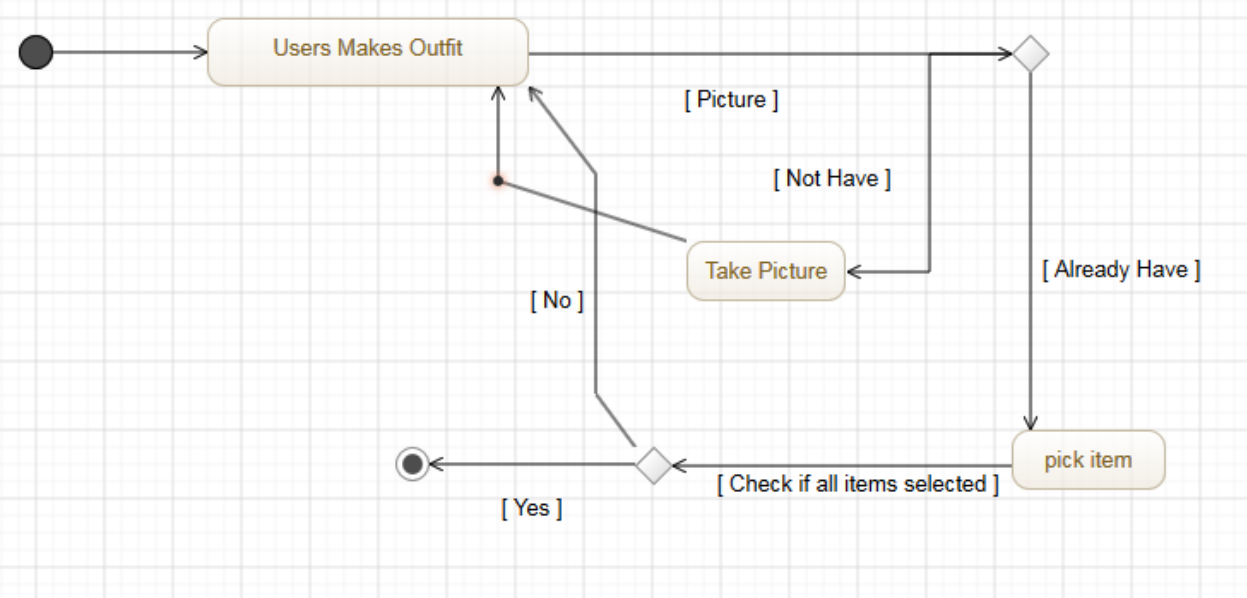
4.3 Activity Diagrams

1. Take Picture



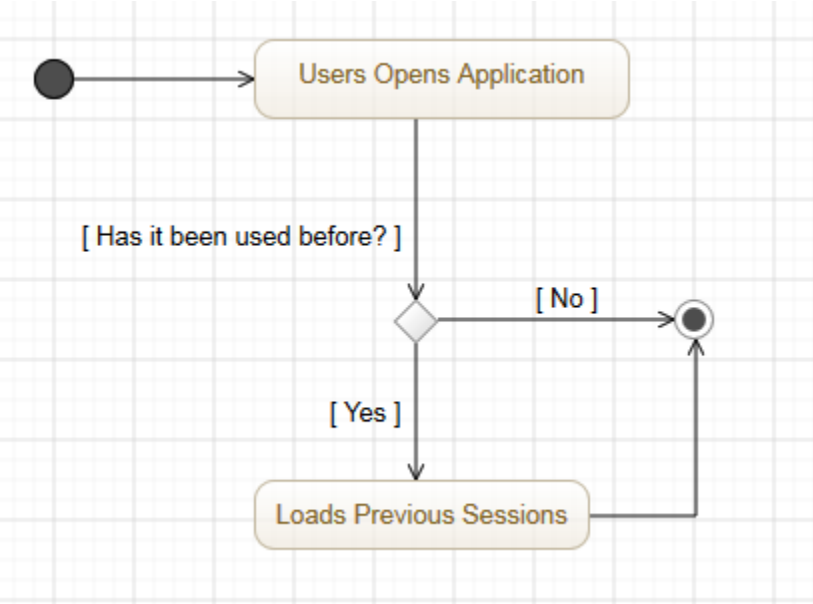
- The take picture activity follows the actions of the user taking a picture to add to the outfit. The user will select to add a new clothing item to the outfit. When this happens the camera app will be opened. Once user takes picture they can choose to keep it or not. If they do not keep the picture then they will be directed back to the camera. Else if the user does keep the picture then the picture is added to a new outfit slot.

2. Outfit Selection



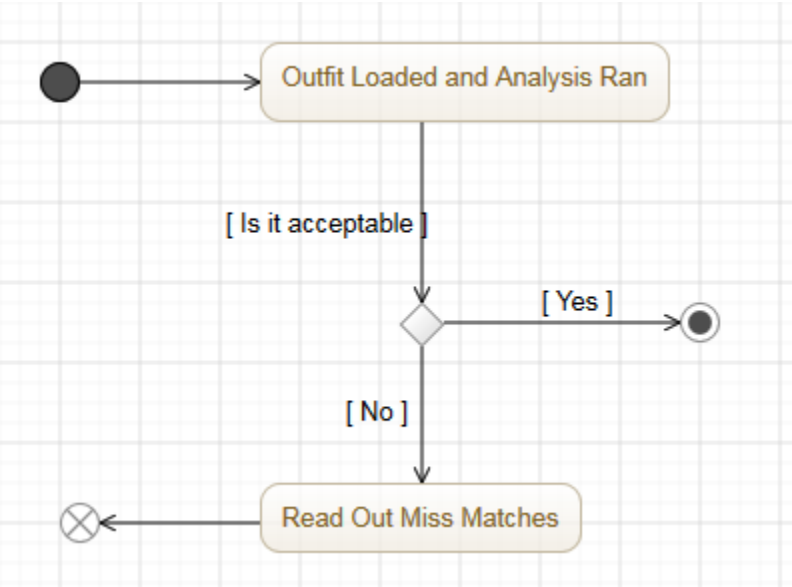
- The Outfit Selection activity deals with the process of the user interacting with the outfit setup process. On the main menu page user can select to open new outfit or load a saved one. When the saved outfit button is pressed then a list of previous outfits is displayed. The user then selects the one they would like and it is opened.

3. Open Application



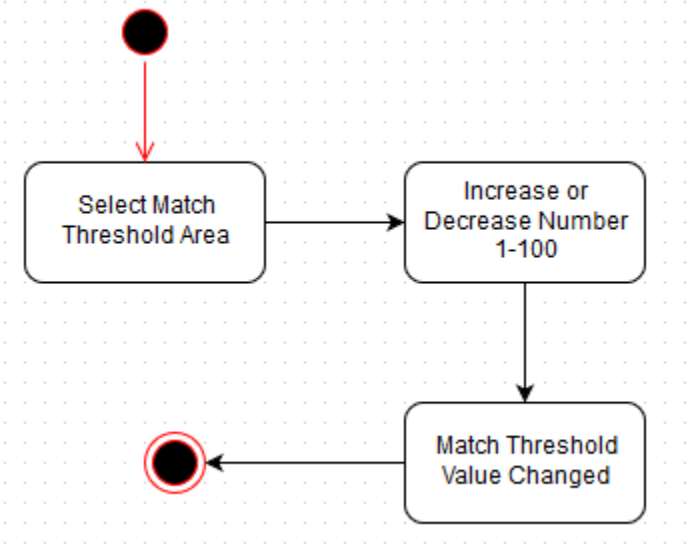
- Open Application is the handling of the initialization of Color Coordinator. When the app is opened it loades previous sessions. This is then displayed to the user.

4. Compute Outfit Results



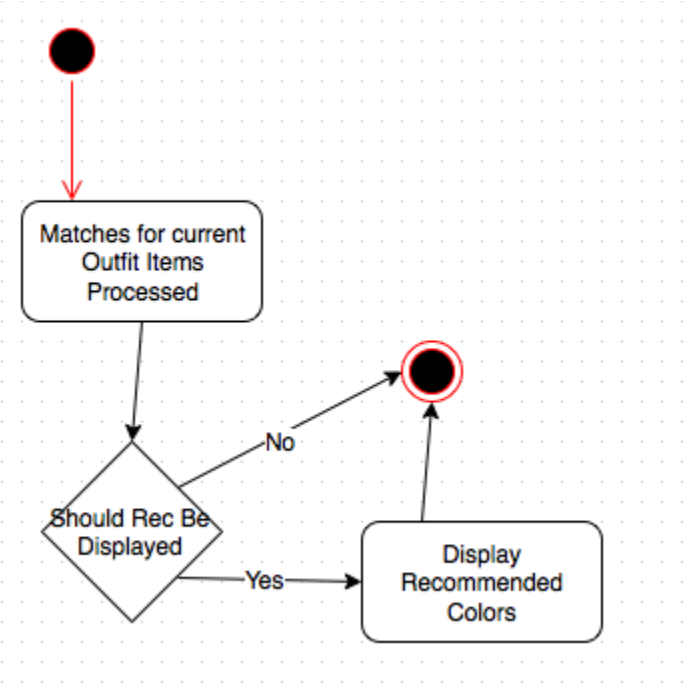
- This activity displays the testing of the Outfit to determine if it matches or not. While an outfit is loaded each newly added clothing item to the outfit then has it's new match rating computed. This is done by using the added color codes in each clothing item to compare and assess the ratings.

5. Adjust Match Threshold



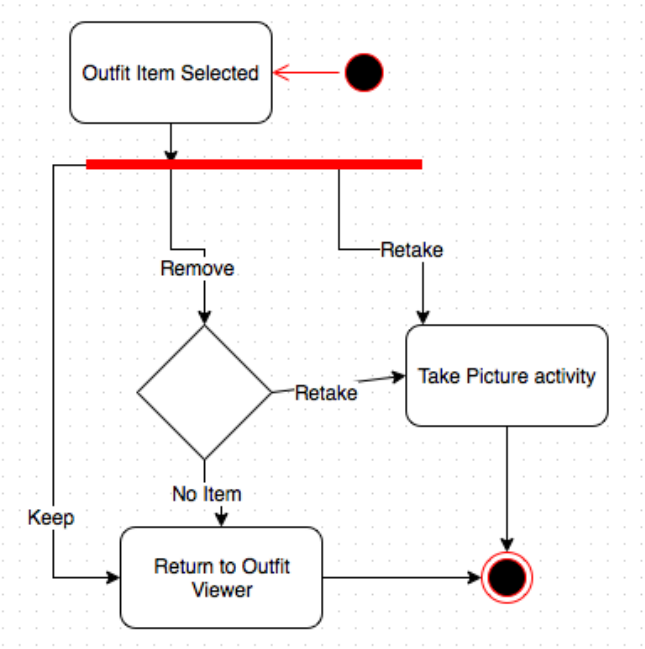
- The activity shows how the user can change the match percent of the overall outfit. The user will enter the options page from the main menu. Then the user selects the match rating bar and can slide it between 1-100. This is then saved and the new match threshold will be changed depending on this value.

6. Recommend Color For Next Item



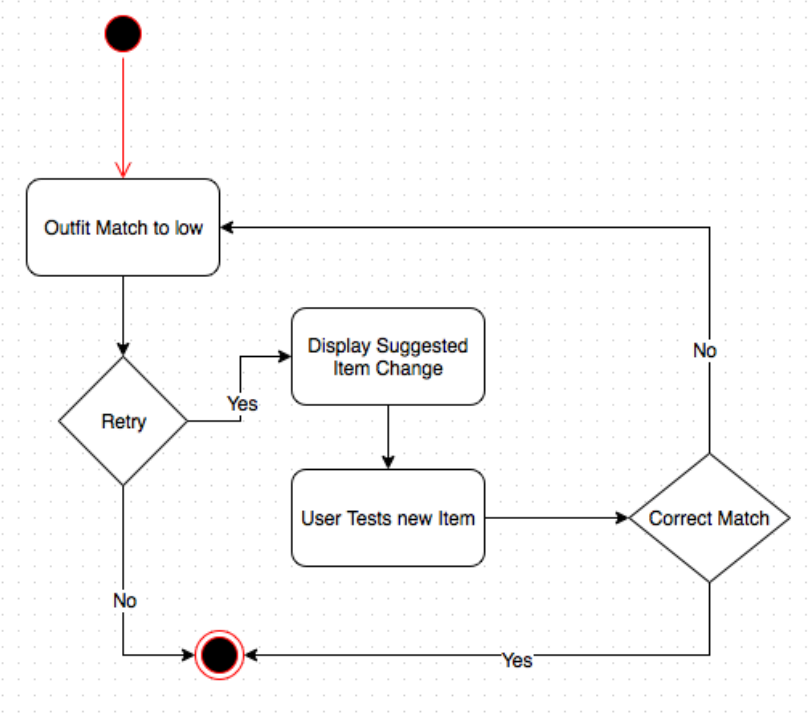
- Activity for recommending a color to the user as a suggestion for what would match well with other Outfit items. Each new clothing item added to outfit will make the recommended color need to be updated. The user will be able to see this recommended color on the outfit screen and it is changed when a new outfit is added.

7. Remove Outfit Item



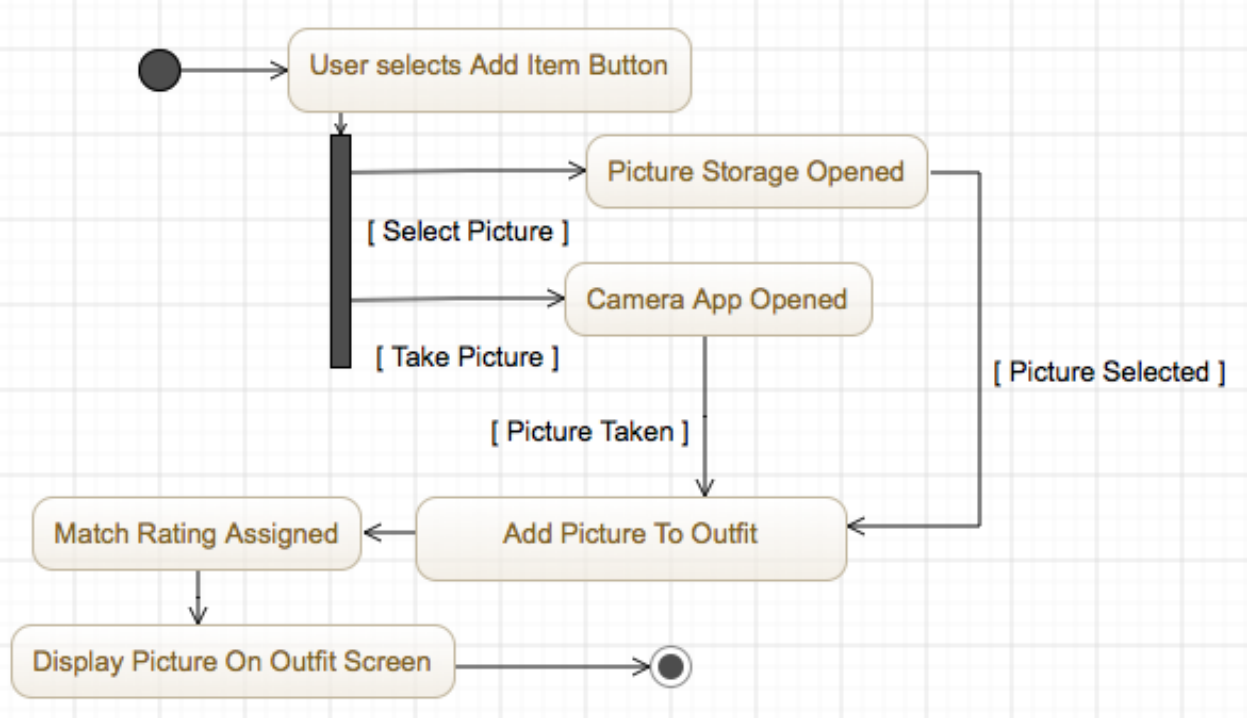
- Activity to remove an item from the Outfit. There will be a list of clothings on the outfit page. The user will be able to hit the X button to the right of each piece of clothing. When this X is pressed by the user they will be asked ‘are you sure you want to remove item?’ If yes then the clothing item will be deleted and the new match rating changed. Otherwise if no there will be no change and no deletion.

8. Alternate Item Suggestion



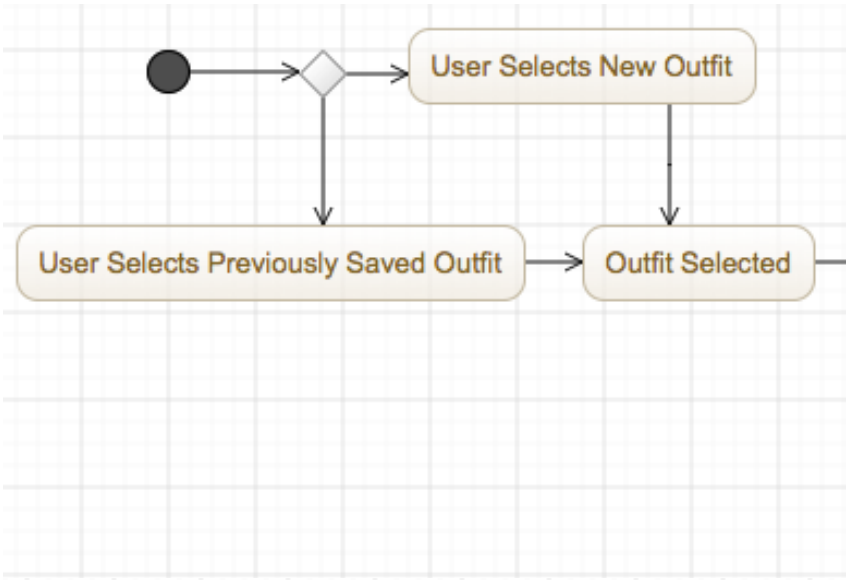
- If the color did not match enough the user is given the chance to change out the item in the Outfit. The user selects yes change item and then the camera app will open again and request photo from user. Once photo is taken the app processes the photo and then agrees with the chosen color and the match rating threshold is high enough or it is not and the cycle continues or until the user denies the request for a different color.

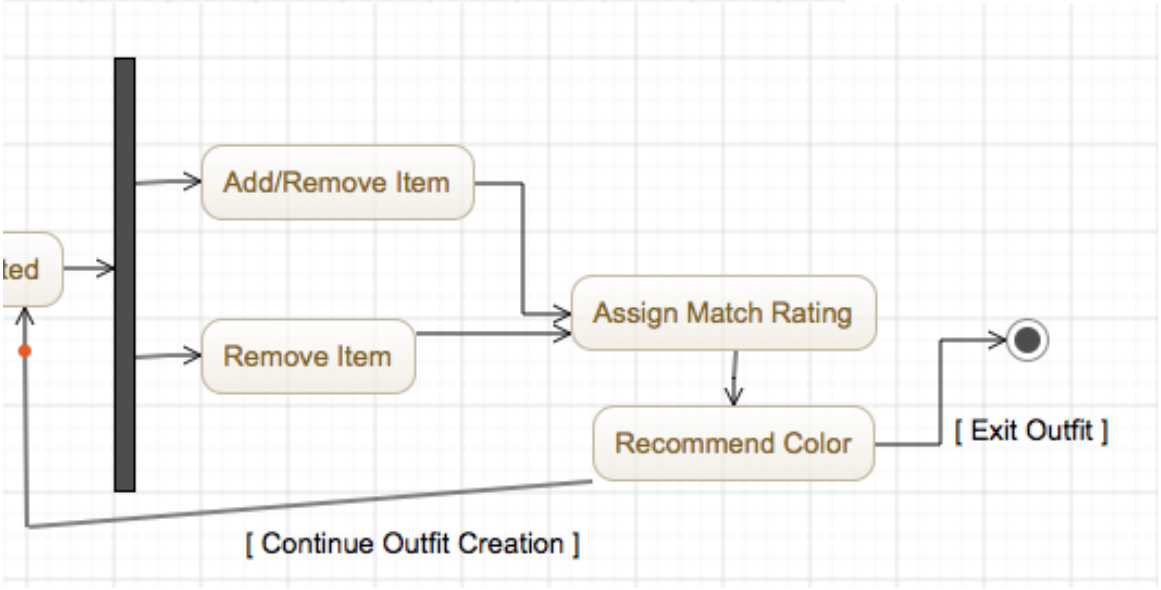
9. Add Outfit Item



- To add an outfit the user will use the addOutfit button on the outfit menu. This will be indicated by a + symbol. The camera will then open and require user to take a photo of the clothing item. Once taken the compute photo is ran and stored with an rgb code with it. Using this the device then processes the color and displays it to the user and processes the new match rating for the outfit.

10.Build Outfit

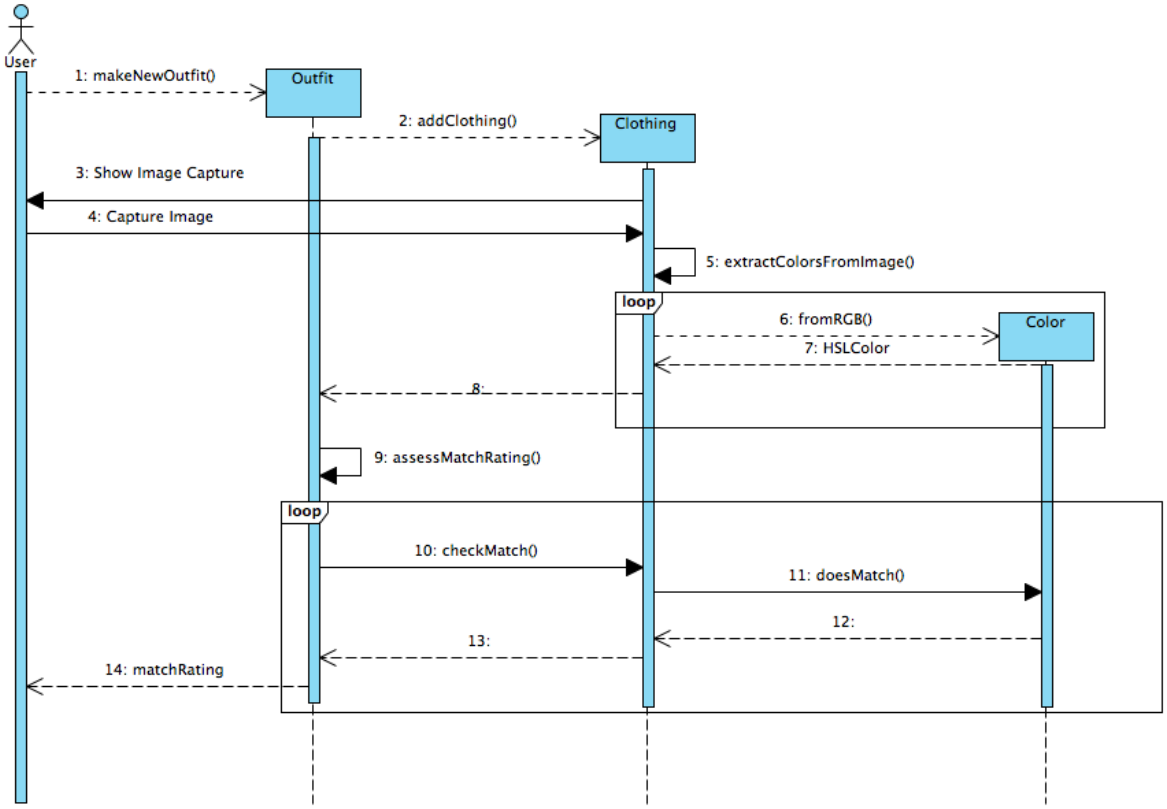




- The Above two diagrams are together and should be considered as such.
- Overall diagram that is used for building the outfit activity. To build an outfit user goes to the outfit menu page. In this page the user can select to add or remove items. Once added or removed a new match rating is processed and found. Then if the user has recommended colors on, the user is alerted of recommended colors.

4.4 Sequence Diagrams

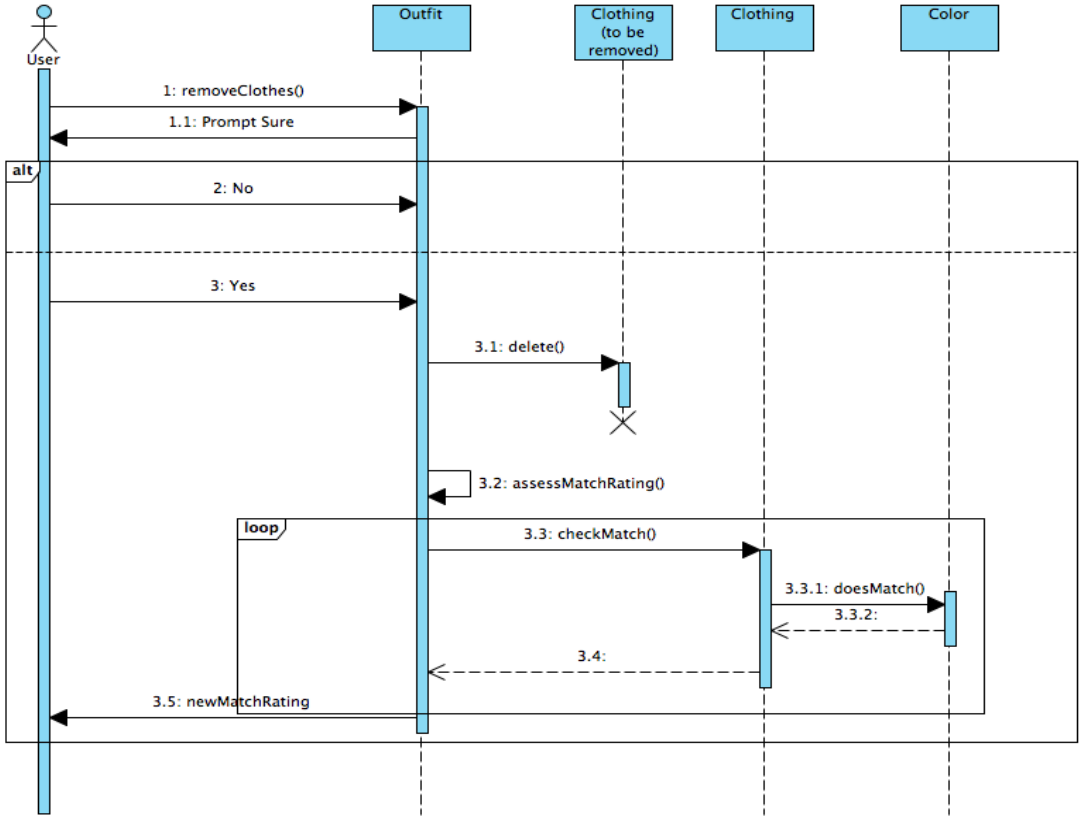
1. Make Outfit Sequence



- When making a new outfit, the user will select a button called “Outfit” from the Main Activity, which will call the makeNewOutfit() method.
    - The makeNewOutfit() method calls a new Outfit Activity.
        - The Outfit Activity is where everything dealing with the Outfit is computed.
        - It contains an add clothing button, remove clothing button, recommended colors, match rating, and a listview of each added clothing item.
    - When the user presses the add clothing button the addClothing() method is called.

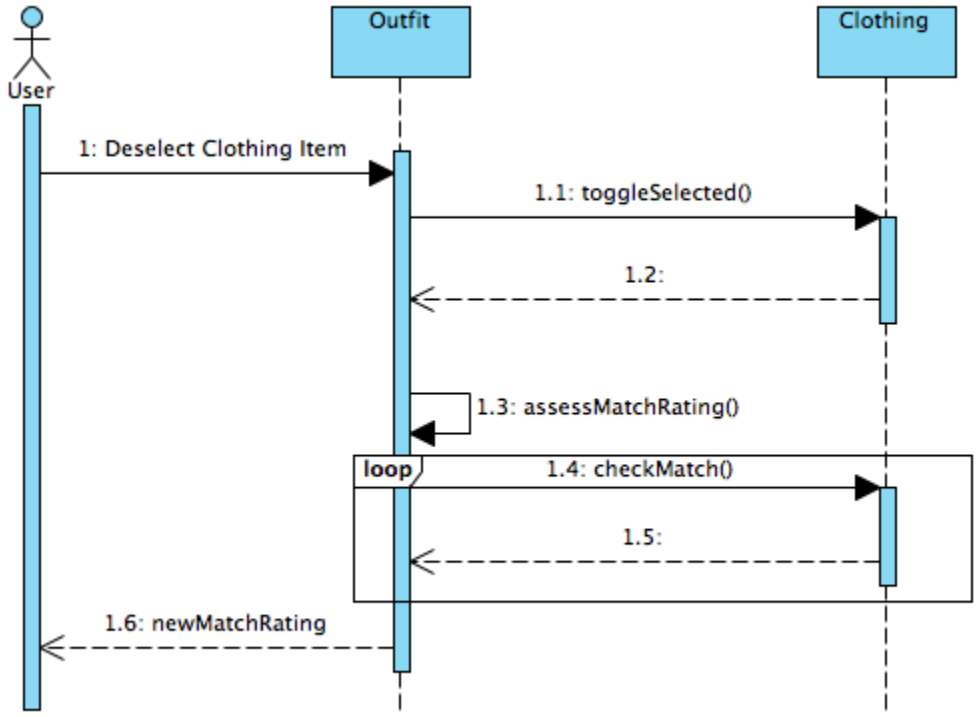
- The addClothing() method initializes an intent for an image capture screen that will be shown to the user in the form of the default Android camera app.
- When the image has been captured, its colors will be extracted in a loop and converted to the HSL color scheme.
    - The extraction is done through the color class which will be called from addClothing() by the extractColorsfromImage() method.
    - Once extracted the fromRGB() method gets the RGB values of the images pixels and computes color codes from them.
    - Also then computing simple color names that relate to the RBG color values.
- After the item of clothing is added, the match rating will be assessed again through assessMatchRating() by checking how well the colors of the clothing match with checkMatch().
- After these methods are ran the addClothing() method will add the image and the relating values into the listview on the Outfit Activity page.
- The Match Rating match rating is then also displayed to the user along with the newly added clothing item.

2. Remove Clothes Sequence



- To remove clothing the user will be able to select one or more Outfit clothing items.
    - Each clothing item is placed into a list view.
    - Each item in the listview is an image of the taken clothing and a quick description of the image. It also contains an area to select each item of the listview.
- Once the clothing item or items has been selected the user will be able to delete all selected items.
    - The delete will be ran by a delete() method within the Outfit Activity Class.
- After the clothing items have been deleted the assessMatchRating() method is ran to find the new Match Rating for the current Outfit after the deletion of the clothing item or items.
    - This will be called directly after the delete() method is called.
    - The Match Rating call loops through checkMatch() for each clothing item and then returns the new match rating as the result.

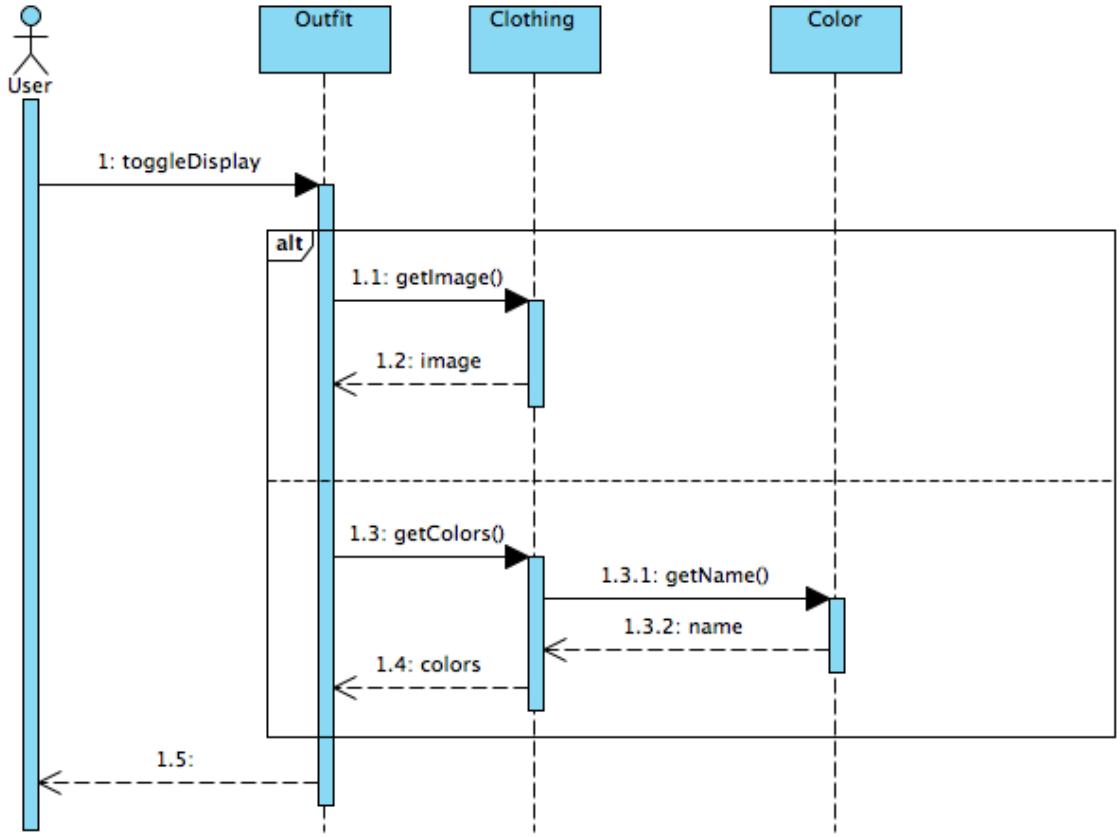
3. Deselect Clothes Sequence



- Selecting Clothing items will be important for the user. It will create a way for the user to quickly view what the Match Rating will be of certain items in the outfit without having to delete and retake more pictures.
    - This will allow the user to be more efficient in their hunt for a good match rating.
- Since each clothing item will be stored in a listview that contains the image,colors of the image, and an area that shows the user has the clothing item selected or not.
    - Once an item is added to the Outfit it will be selected.
        - This means the clothing item is apart of the Outfits Match Rating.
    - If the user deselects the item.
        - The toggleSelect() method is called and it will deselect the item and take away the checkmark in the listview.
        - Then assessMatchRating() is also ran taking the current item out of the process. By doing this the checkMatch() will not use the deselected item within the computation of the Match Rating.
        - The new Match Rating is returned and displayed to the user and the current item stays deselected until the user decides to select it again.
    - If the user deselects the clothing item and it has already be deselected then the clothing item will do the opposite.
        - toggleSelected() will then make the clothing item selected within the viewlist.

- This will cause the clothing item to then be computed again during the assessMatchRating() and the checkMatch() methods.
- Again the Match Rating will be displayed to the user.

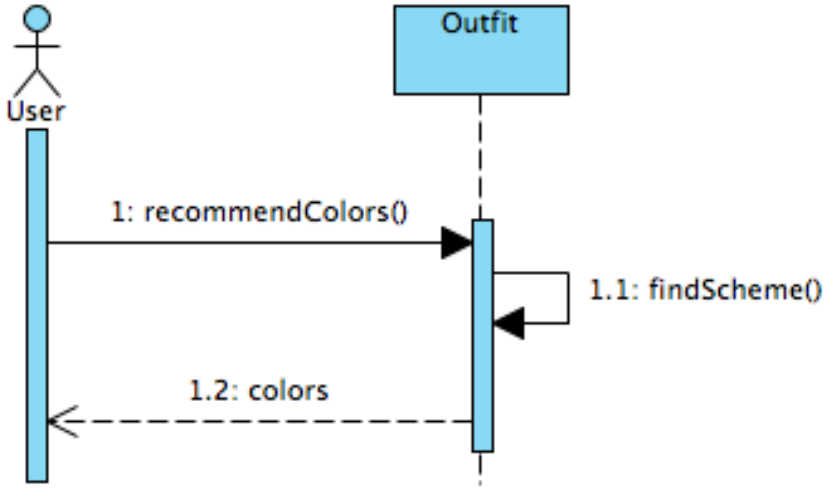
4. Toggle Display Sequence



- Toggle display sequence is used for when the user wants to not display or display the clothing image in the list view.
    - The image will be displayed by default. If the user does not want them displayed(downsides are that with it displayed, it could take up to much visual space), then the user will hit the toggle display button.
        - When toggle display is used the images are found from getImage().
            - Then the colors of each image are found using the getColors() method.
            - The names of each image are also found through the method getName().

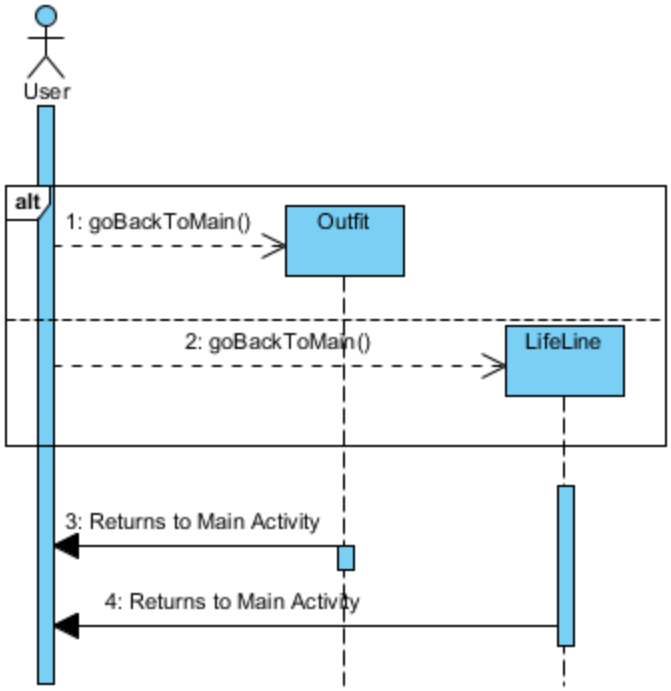
        - Once each Method has ran the values will be returned and only the names and colors of each clothing item will be displayed in the listview.
    - Else If the images are already displayed the toggle display will allow the user to view the images of each color item in the view display.
        - If the user has previously toggled the display then the user might not be able to see the images. To display them again the user must again hit the toggle display images.
            - When toggle display is used the images are found from getImage().
                - Then the colors of each image are found using the getColors() method.
                - After the getColors() call the getName() method is called to retrieve the names of each clothing item.
            - Once each Method has been ran the values will be returned and the listview will be updated.
            - The listview will then display the image, name, and colors of the clothing item for the user to view.

5. Recommend Colors Sequence



- The user will be able have a set of recommended colors displayed to them.
    - In the options menu the recommended colors option is a boolean and can be toggled off or on to the user's discretion.
    - The initial value of recommendationsOn is true then the recommended colors will be on.
    - If recommendationsOn is true then the user will see a list of recommended colors.
        - The recommendedColors() method is ran each time a new clothing item is removed or added.
            - recommendedColors() runs another method findScheme() that determines the overall color aspects of the current outfit and returns the color values as basic color names.
            - The intent is to make it easy for the user to understand and read. IE using colors such as green, dark-red, or light-blue.
        - Once returned the user will be able to see a list of the recommended colors in the bottom of the Outfit Activity.

6. Return to Main Activity



- The User will always be able to go back to the Main activity while they are in the Options or the Outfit activity page.
    - When the user hits the Back button:
        - Through an onClick() call the method goBackToMain() is called.
        - All current page information will be saved.
        - An intent to send the app to the Main activity page will be ran.
        - Once redirected the user can go back to Options or to Create an Outfit
- This method is currently not in the Class Diagram and it should be. As of now it is an added activity and will be added to reflect the correct Class Diagram later.