Suiter

Software Design Document

Names: Eli Haimov, Yosi Golubchik

Date: 16.03.20

1. Introduction

1.1 Purpose

This software design document describes the architecture and system design of the Suiter application.

In contrast to the Software Requirements Specification (that is written for the client and user), most of this Software Design Description is written for knowledgeable software professionals and designers. Thus the Client will not be within the intended audience for this document, which is:

- Team
- Supervisor
- Auditors and Reviewers

1.2 Scope

This Software Design Description (SDD) describes the detailed structure of the components of the Suiter application and the precise implementation details required to satisfy the requirements as specified in the Software Requirements Specification (SRS). It is assumed that the reader has read the SRS, since this document also defines the implementation details of the desired behaviour given the requirements within it.

1.3 Overview

The Software Design Document is divided into 11 sections with various subsections. The sections of the Software Design Document are:
1 Introduction
2 System Overview
3 System Architecture
4 Data Design
5 Component Design
6 Human Interface Design
7 Requirements Matrix
8 Appendices

1.4 Reference Material

This section is optional.

1.5 Definitions and Acronyms

This section is optional.

2. System Overview

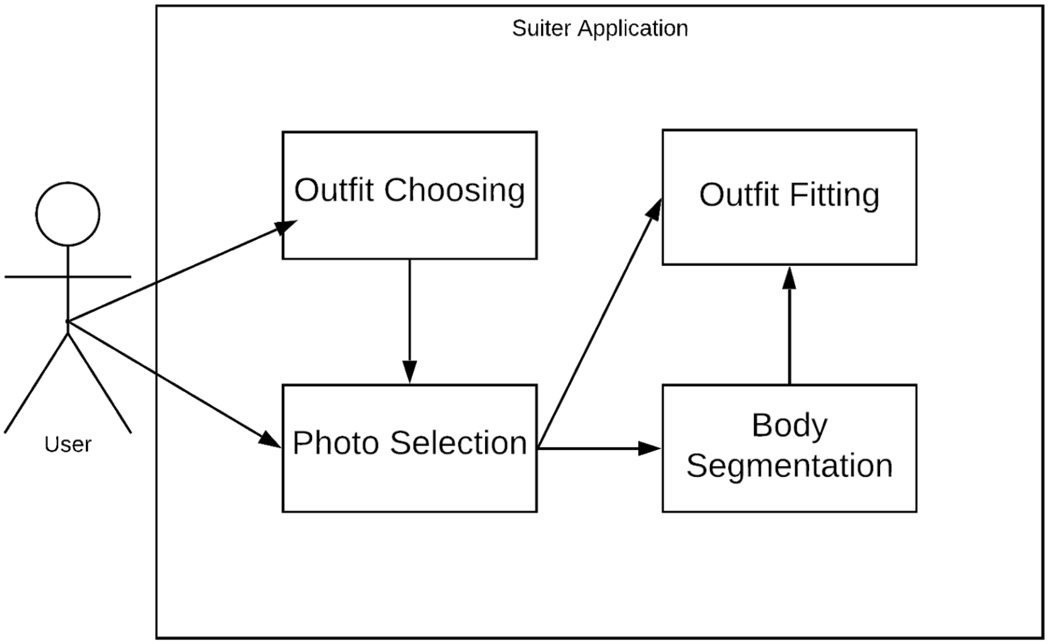
The Suiter application will allow the user to choose a suit for himself and see how it will look on him. The user will provide a photograph of a front view of himself and choose colors for the suit, and the application will place the suit on top of him accordingly.

3. System Architecture

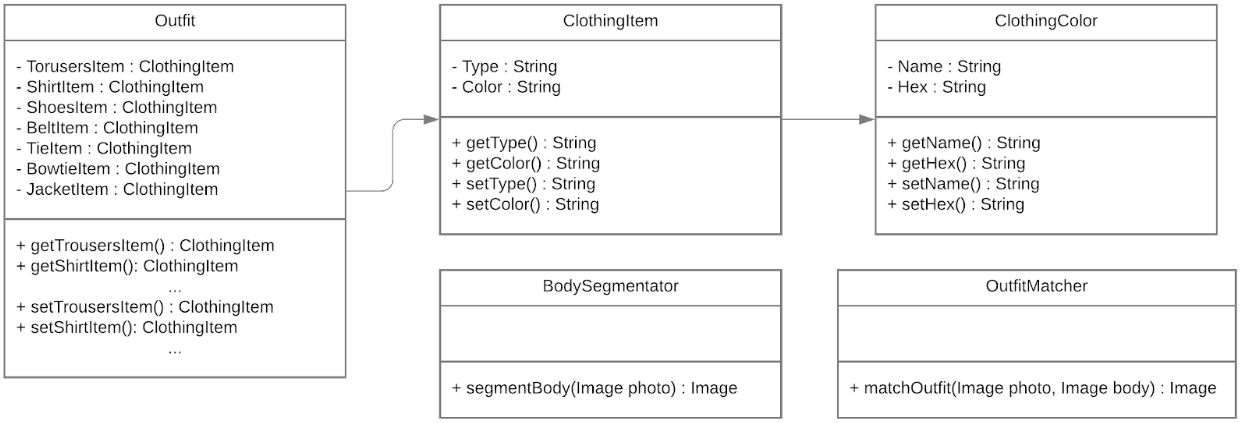
3.1 Architectural Design

Suit Creation: user chooses desired clothing items (shirt, trousers, shoes, belt, tie/bow tie and jacket) and their colors.
Photo Choosing: The user chooses a photo from the gallery or takes a new photo with the camera, which will be used for matching the outfit to.
Body Segmenter: Uses deep learning algorithms to find where the body is in the photo, to be used in the matching algorithm.
Suit Matching: Uses machine learning algorithms to match the suit photos onto the body in the photograph provided by the user.

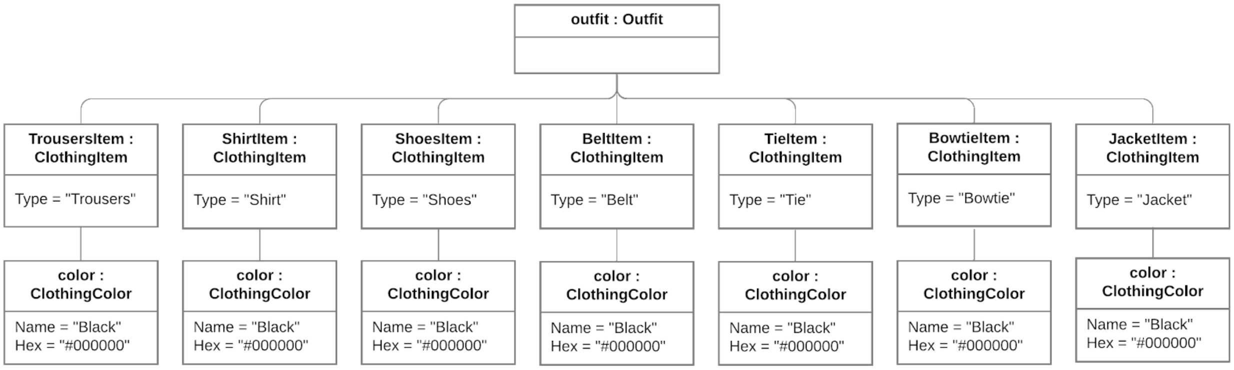
3.2 Decomposition Description

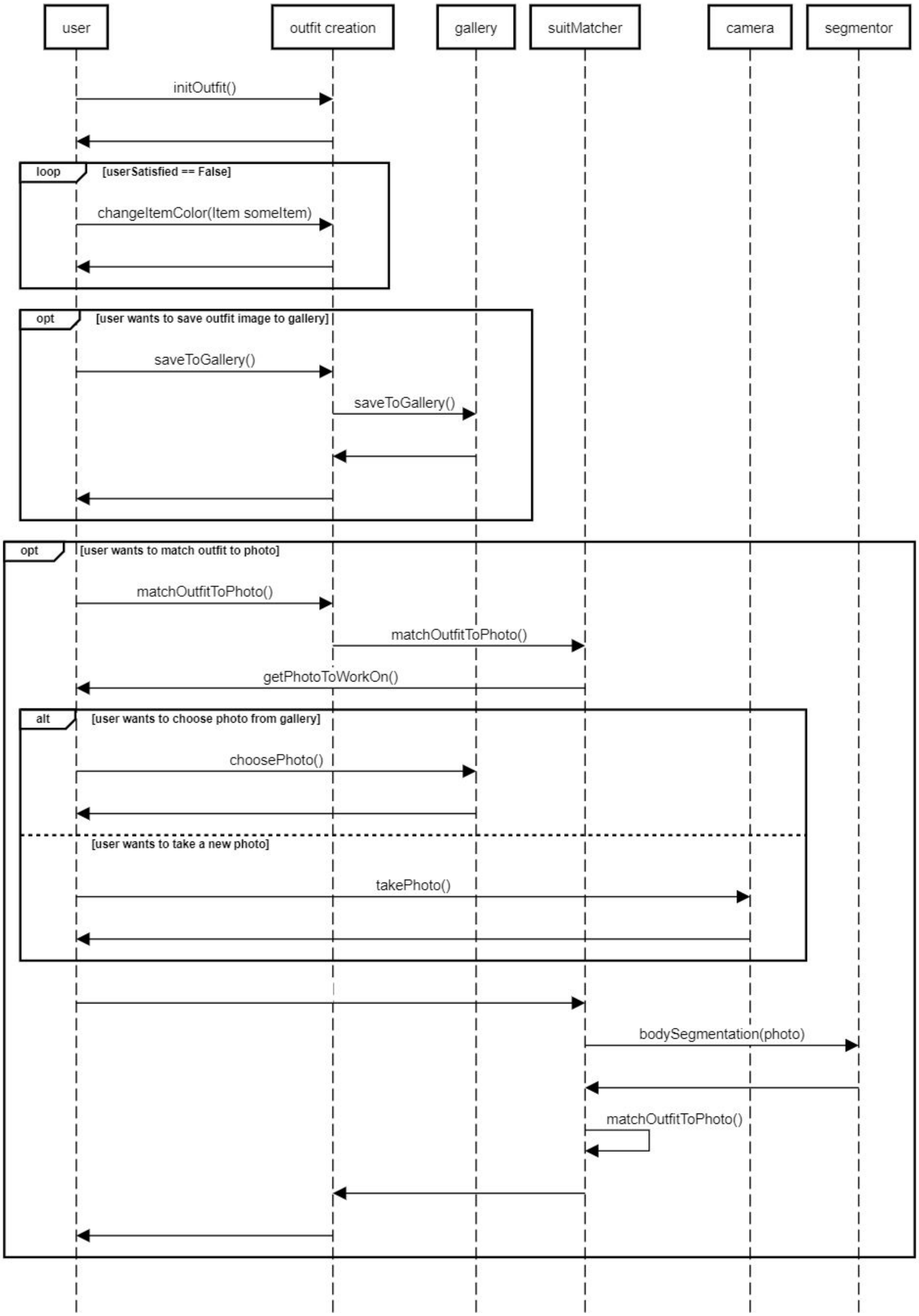


Class Diagram


Outfit
- TorusersItem : ClothingItem
- ShirtItem : ClothingItem
- ShoesItem : ClothingItem
- BeltItem : ClothingItem
- TieItem : ClothingItem
- BowtieItem : ClothingItem
- JacketItem : ClothingItem
+ getTrousersItem() : ClothingItem
+ getShirtItem(): ClothingItem
...
+ setTrousersItem() : ClothingItem
+ setShirtItem(): ClothingItem
...
ClothingItem
- Type : String
- Color : String
+ getType() : String
+ getColor() : String
+ setType() : String
+ setColor() : String
ClothingColor
- Name : String
- Hex : String
+ getName() : String
+ getHex() : String
+ setName() : String
+ setHex() : String
BodySegmentator
+ segmentBody(Image photo) : Image
OutfitMatcher
+ matchOutfit(Image photo, Image body) : Image

Object Diagram




user
outfit creation
gallery
suitMatcher
camera
segmentor
initOutfit()
loop
[userSatisfied == False]
changeItemColor(Item someItem)
opt
[user wants to save outfit image to gallery]
saveToGallery()
saveToGallery()
opt
[user wants to match outfit to photo]
matchOutfitToPhoto()
matchOutfitToPhoto()
getPhotoToWorkOn()
alt
[user wants to choose photo from gallery]
choosePhoto()
[user wants to take a new photo]
takePhoto()
bodySegmentation(photo)
matchOutfitToPhoto()

3.3 Design Rationale

We tried to select the architecture which follows the rules of object oriented programming as best as possible, and divided the software into seperate parts in the most logical way, such that each part has one responsibility.

4. Data Design

4.1 Data Description

The objects are stored in two tables, “Items” and “Colors”.

This database will be stored on the users device and updated when downloading updates for the app. We won’t be using a server database because the database changes very rarely (only when we add new items or colors).

4.2 Data Dictionary

Items:

ItemID	Type	ColorID
“a54ggdD3”	“Shirt”	“6643gGGd”

Colors:

ColorID	Name	Hex
“6643gGGd”	“Royal Black”	“#000001”

5. Component Design

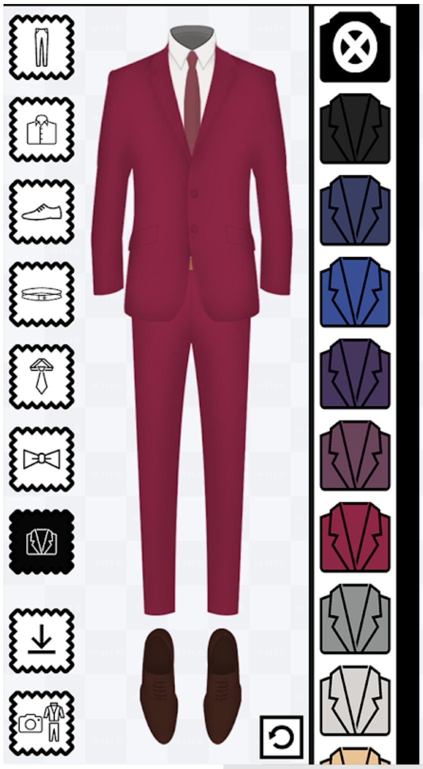
```
initOutfit():
    outfit = new Outfit()
end-initOutfit

changeItemColor(Item someItem, Color color)
    someItem.setColor(color)
end-changeItemColor

saveToGallery(Outfit outfit)
    Image outfitImage = imageFromOutfit(Outfit)
    Device.Gallery.Save(outfitImage)
end-saveToGallery

matchOutfitToPhoto(Outfit outfit)
    Image photo = getPhotoToWorkOn()
    Image bodySegment = Segmentor.bodySegmentation(photo)
    matchOutfit(outfit, photo, body)
end-matchOutfitToPhoto

getPhotoToWorkOn()
    Image photo = askUserForPhoto()
end-getPhotoToWorkOn

bodySegmentation(Image Photo)
    /// Deep learning algorithm using CNN
end-bodySegmentation

matchOutfit(Outfit outfit, Image photo, Image body)
    /// Computer vision algorithms (warping, points of
    /// interest…)
end-matchOutfit
```

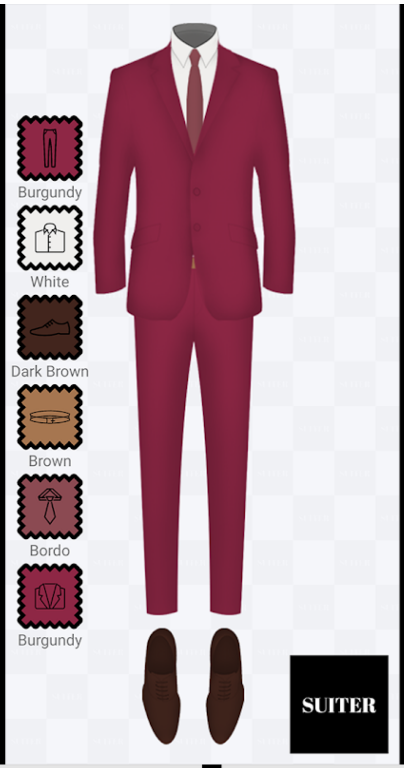
6. Human Interface Design

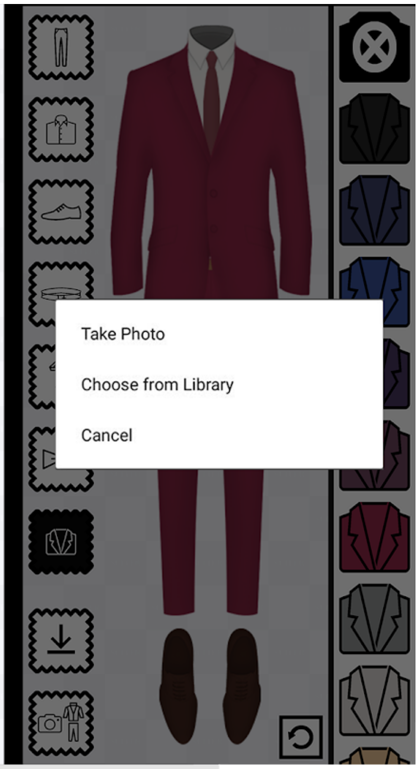
6.1 Overview of User Interface

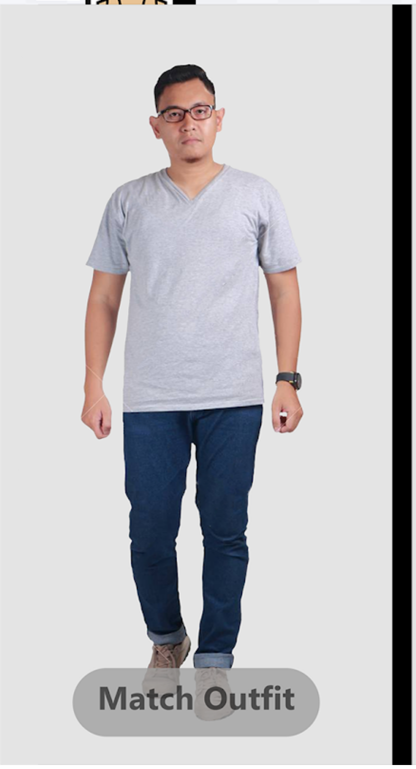
When the user opens the application he will start on the outfit choosing activity, which will allow him to choose clothing items and their colors. There will be a button to save the outfit as an image to the gallery, and a button to match the outfit to a photo. If the user chooses to match the outfit to the photo, the app will ask him to either choose a photo from the gallery or take a new photo. Afterwards the matching process will start, and when it finishes the user will get the result and could save it if he wants, or go back to choosing a different outfit.

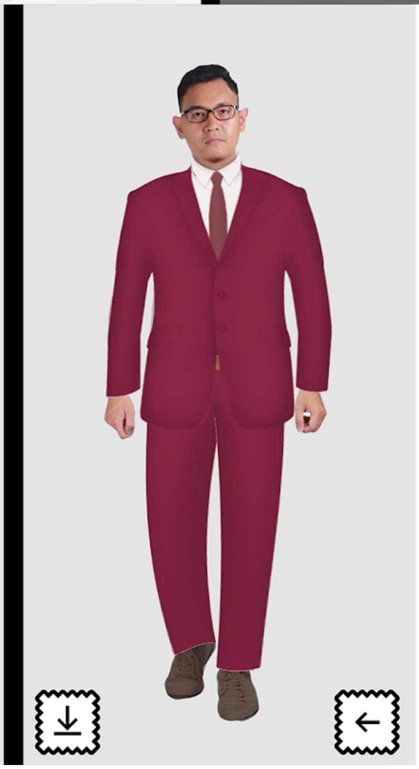
6.2 Screen Images











6.3 Screen Objects and Actions

In screen 1 - First 7 buttons from top left - Choosing clothing item category
Next button saves the outfit as an image (screen 2 is an example)
Next button starts the outfit matching process - asks you for photo (screen 3)
The right column of buttons allow you to choose different colors for current category.
The small rotation button allows you to see the outfit from the back.
In screen 3 you can either choose a photo from gallery or take a new photo.
In screen 4 there is a single button which starts the matching process on the provided photo.
In screen 5 there is a save button which saves the result as an iamge, and a back button that leads you back to outfit creation.

7. Requirements Matrix

Req ID	Description	Component Satisfying It
1	The system shall allow the user to create an outfit by choosing a color for each of the 7 clothing items (Trousers, Shirt, Shoes, Belt, Tie, Bow Tie, Jacket).	Outfit Choosing
2	The system shall allow the user to choose a tie or a bow tie or neither, but not both at the same time.	Outfit Choosing
3	The system should allow the user to save the result as a picture to the device's storage.	Outfit Choosing
4	The user should be able to either choose a photo from the gallery or take a new photo to be used in the outfit fitting process.	Photo Selection
5	The system will use semantic segmentation to find the pixels corresponding to the body in the photograph given by the user.	Body Segmentation
6	The system will use computer vision algorithms to warp the picture of the outfit onto the photograph given by the user, with help from the result given by the body segmentation process.	Outfit Fitting