Department of Computer Engineering

Senior Design Project

Photonom

Analysis Report

Group Members	Omer Faruk Babademez	21601216
	Idil Hanhan	21601289
	Beyza Kalkanli	21600944
	Berfin Kucuk	21502396
	Kerem Yilmaz	21601223
Supervisor	Asst. Prof. Hamdi Dibeklioğlu	
Jury Members	Prof. Ozcan Ozturk	
	Assoc. Prof. Selim Aksoy	

Analysis Report
Nov 11, 2019
This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfilment of the requirements of the Senior Design Project course CS491/2.

Contents

1 Introduction

Taking photos have turned into a daily practice, whether we are visiting a new country, celebrating a birthday or just having a normal day. In fact it has been estimated that more than 100 million photos and videos are uploaded to Instagram everyday [1]. Even though this is an impressive number, it does not even include the number of photos that people take for each upload. For some this number can be 20 and for other it might go up to 200 [2].

There are many reasons why one would have to take a lot of photos to get one shot they like. The light does not look right in one of them, someone walked behind them in another one, the building they want to be on the background did not fit the frame, they look too serious in that one and so on. Taking more photos can be a possible solution, but in most cases you might realise these problems when its too late. Going through your vacation photos to realise that picture you took in front of a historical church is missing the tip of the church is frustrating. Even if you do realise these problems when you have the chance to take more photos, some things cannot be changed. If it is raining and water droplets are in your lens, it is impossible to change the weather. And even if the problem is something you can change, why waste time taking tons of photos when it is possible for you to enhance or modify the ones you already have.

There are multiple tools that one can use to alter their photos. Advanced photoshop tools are often too complicated for occasional use. It is possible to find a tool that will remove a background item and then another one to alter your head position and/or facial expression and then find a filter in another app that reflects your style. In this case, it is not common to find all these features in one place and the users have to jump from one app to another to get what they want.

With Photonom, we are proposing a mobile application that will allow users to modify their photos interactively and in one place whilst conserving the true nature of the image itself. In the next section, Photonom will be described in more detail and constraints related to it will be explained. Then both the functional and nonfunctional requirements of Photonom will be explained. The references can be seen at the end of this report.

2 Proposed System

2.1 Overview

Photonom is a mobile application that provides its users different tools to edit their photos interactively. The tools of Photonom aim to enhance the user's photo whilst ensuring the true nature of the photo does not change. The users will be able to upload their photos to Photonom or take photos using the application. The tools provide the following functionalities; object removal, rain removal, theme transfer, image stitching, changing face expression, image quality evaluation and head pose modifier. Whilst the user is editing their photo, each change they make will be saved and the user will be able to revert their actions and see the entire history of their editing and go back to any state they want. After their editing is done they will be able to share their final product.

2.1.1 Photo Selection

In order to use the functionalities of Photonom, the user first has to select the photo(s) they want to edit or evaluate. There are multiple ways for a user to make this selection. The first one is the case where the user uploads a photo they already have in their device's gallery to Photonom. The second case is where the user uses Photonom to take photos. Additionally the user will also be able to continue from a photo they previously uploaded to Photonom. The choice for photo selection will be made by the user when they first open the application.

2.1.2 Editing Tools

Photonom provides many editing tools to it's users and in this section these tools will be introduced. They will then be explained in detail in section 2.2.

2.1.2.1 Remove Objects

User will use Remove Objects tool when they want to remove an object from their photos. After the user specifies the objects they want to remove and this operation is performed, the user will be able to modify the end result.

2.1.2.2 Rain Removal

The purpose of Rain Removal is to offer a tool that allows the users to remove the effect of rain from their photos such as the rain droplets on the camera lens.

2.1.2.3 Theme Transfer

The Theme Transfer will be used when the user has a sample theme that they want to apply to their photo. Once this tool is selected and the user provides a photo that represents the theme they want to adapt, the photo they selected for editing will be changed to fit this theme.

2.1.2.4 Image Stitching

User will use Image Stitching tool when their photo includes missing sections. The user will provide the photos to stitch and they will be placed by the system. The user will be able to determine the limits of the photo by cropping the stitched images. Then they will be given a final photo where the missing parts are completed.

2.1.2.5 Face Expression Modifier

The purpose of Face Expression Modifier is to allow the user to make slight changes to their facial expression according to their desire.The user will be able to choose the face they want to change and slightly change their facial expression by, for example, making them smile more.

2.1.2.6 Image Quality Evaluation

Using Photonom the user will also be able to evaluate the quality of a selection of photos. When the users select multiple photos, these photos will be evaluated and each will be given a score representing their quality. One of these photos can be chosen for editing.

2.1.2.7 Head Pose Modifier

This is a tool that will only be implemented if time permits. The purpose of Head Pose Modifier is to offer a tool that allows users to slightly change the position of their heads.

2.1.3 Undo and Redo

Whilst the user is editing their photo, it is possible that they will make a move that they regret and want to undo. Photonom will include an Undo option that can be used when the user wants to undo the most recent modification they made. Photonom will also include a Redo option which will allow the user to reapply the most recent modification they made.

2.1.4 History

In Photonom, the user will be able to see the entire history of their editing process. In here the user will see all previous states of the photo with the quality score of that state and will be able to go back and continue from any point in history.

2.1.5 Saving & Sharing the Edited Photo

The user will be able to save the state of their photo so that they can continue editing later on. When they finish their editing, they will be able to save their photo to their device's gallery and/or share their final product.

2.2 Functional Requirements

In this section the functional requirements of Photonom will be explained.

2.2.1 Uploading Pictures and Taking Photos

The first thing the user will do when they open the application is to decide whether they will upload the photo they want to edit or they will take it using the application. In order to use the other functionalities of Photonom, the user must successfully choose their photo(s). During the upload, they will also be able to adjust the size of their photo. If they choose to upload, then a view of their gallery will open and the user will select their photo. Or they could also use the camera function in Photonom to take a photo. If the user only provides a single photo then they will be directed to a page that shows the photo and includes all of the tools they can use for editing. In the case where the user provides multiple photos then they will be directed to a page for Quality Evaluation, explained in section 2.2.7.

2.2.2 Remove Objects

The users will be able to remove items, such as cars or people, from the background of their photo by using this tool. In order for this function to work effectively, the said item needs to be in front of objects, people, buildings or natural scene. When this tool is selected, the user will specify the object they want to remove by painting on it. Once the object is specified by the user, the object will be removed and then the version of the photo with that section removed will be shown to the user. The user will be able to patch the background.

2.2.3 Rain Removal

With this functionality, the users will be able to eliminate the effects of rain on their photos, whether that's by removing the rain droplets on the lens or increasing the clarity of the photo.

2.2.4 Theme Transfer

This functionality will allow users to transfer the theme of one photo to another. After the photo the user wants to edit is chosen and this tool is selected, the user will provide an additional photo which will represent the theme they want to transfer to their photo. Then their photo will be changed so that it reflects the chosen theme.

2.2.5 Image Stitching

The users will be able to stitch images together to complete the missing parts of their photo. After the user chooses the photo they want to edit and this tool is selected, the user will provide the image for the missing part. Then all of the images will be stitched together. The user will then be able to crop the photo to specify the limits of the final product.

2.2.6 Face Expression Modifier

This functionality will allow users to change their facial expression. When this tool is selected, the faces on the photo will be highlighted by the application. The user will have to choose a face and then will be able to change the expression of that face in order to make it more happy or sad etc. The user will make these changes with the help of a slider, so that the changes appear gradually during editing.

2.2.7 Image Quality Evaluation

As mentioned before, if the user chooses to upload or take multiple photos then they will be directed to this functionality. The purpose of this functionality is to evaluate the photos according to both aesthetics and technical details in order to help the user in making a decision amongst these photos. The aesthetics will be assessed according to it's conformity of the photograph to photographic rules and practices such as golden ratio and the harmony of colours [3]. The technical quality will be evaluated according to resolution of the photo, contrast, and brightness. After the evaluation, the user will see a slideshow with slides with the score of each photo written below the slideshow.

2.2.8 Settings

Following functionalities will be accessed from the settings:

- Switch To Dark Mode: This will change the colouring of the application to dark mode.
- Change Language: Using this functionality the user will be able to change the language of the application.

2.2.9 Undo and Redo

Using Undo, the user will be able to undo the most recent modification they made. Using Redo, the user will be able to reapply the most recent modification they made.

2.2.10 History

Photonom will include a History for each photo which will show every previous state of that photo with the quality score of that version written next to it. Using this history map, the user will be able to go back and continue editing from any version.

2.2.11 Saving and Sharing the Edited Photo

Whilst the user is editing a photo the user will be able to save their current state and continue later on. When they ready to finish they will be able to save the photo to their device's gallery and/or share it on different platforms.

2.2.12 Head Pose Modifier

This is a functionality that will only be implemented if time permits. With this functionality the user will be able to slightly change their head position. This change will be limited to up to 10%.

2.3 Nonfunctional Requirements

In this section the nonfunctional requirements of Photonom will be explained.

2.3.1 Accessibility

Photonom should be accessible to everyone around the world. In order to ensure this, English will be the default language of the application and other languages will also be supported.

2.3.2 Availability

Photonom aims to provide many functionalities to its users and will be created to have a modular easily-pluggable design. We also aim to ensure that any update to Photonom do not affect the availability of the application. In order to provide high availability but not spend too much monetary resources we will have two servers and a simple orchestration tool supported cluster with 3 containers. Here is a simple flow for our release mechanism:

1. When there will be a new release for any of the modules, the load will be balanced between two containers instead of three.
2. The separated cluster will be deployed with the new changes and will be made sure its alive and well.
3. The load will be shifted to the newly deployed cluster.
4. After all previous jobs are completed in the previous two machines they will be deployed with the updated changes and they will be health checked.
5. The load will be separated between all clusters again.

2.3.3 Sufficient Network Bandwidth

Photonom will require around 10 Mbps (Megabits per second) download and upload speed. This is due to the fact that Photonom performs computationally heavy operations and the transfer of image will require such speed to the increase in the quality of images taken by mobile phones. As we progress into a new era of 5G, these bandwidth requirements are simply minimal.

2.3.4 Platforms and Portability

The system will persist on two different platforms: Server and the mobile applications. Since we are trying to produce a minimum viable product (MVP) at the end, we don't want to spend extra effort on our mobile application for different operating systems such as iOS, Android, or Windows

Phone. We'll use a cross-platform framework such as Flutter by Google so that the apps are easily ported and our application is available on most of the mobile phones.

2.3.5 Backup

The system will perform a set of operations on their photographs, if the user gives permission, we can also save the photo to their preferred cloud storage service such as Google Photos at different stages of the performed operations in case the user later wants to access one of the others.

2.3.6 License

Any dataset, product, library used in the product should be referenced properly. Moreover they need to be paid if necessary in the case of commercial usage.

2.3.7 Legal

The system will perform on the users' private photographs. Hence, we need to obtain the necessary permissions from the user as they start using the app. For general data protection regulation (GDPR) compliance, the system will show all the operations performed on the photographs and if the users want their data to be deleted from the system, it will also be available as an option [4]. For example their data could be their action logs in our database which we may use the track in order to improve our user experience.

2.3.8 Scalability

The system is aimed to provide its service globally. As the usage increase, the system will need to scale up to stay alive during high loads. To serve this purpose, we aim to design a dockerized server which will be easily deployed [5]. This way the system can scale itself up in case of high loads automatically by the use of cloud services such as AWS, GCP [6] [7].

2.3.9 Security

The personal data of the users will persist on the server during the operations. Hence, our server should be secure and should not permit any intruders for the sake of the privacy of the users' data. In such a case, all operations will be cancelled directly and the corresponding container will be quarantined.

2.3.10 Testing

The system is promised to be available at all times. To achieve this, there will be unit and integration tests that check the main functionality and the health of the system. There will also be a circle CI system that includes a coverage test and will not let the new changes to be applied to master.

2.3.11 Usability

The system will provide many functionalities and since the users will use it as a mobile app, the app should have a natural learning curve which may be supported with welcome tutorials if seen necessary.

2.3.12 Performance

Each action user wants to perform should not take longer than 3 seconds. Transition between screens and response to user functions should not take longer than a second.

2.3.13 Maintainability

The system is going to have a modular design, hence it should not be highly coupled and a change in one of the modules should not affect the others. Moreover, there will be a defined code standard along with additional automated checks regarding the new code changes. There will be a reviewer to each pull request to increase the maintainability of the code in the future changes.

2.4 Pseudo Requirements

In this section the pseudo requirements of Photonom will be specified.

- GitHub will be used for the maintenance of the code.
- Jira by Atlassan will be used to track the sprints of the project.
- Slack will be used to have more organised and grouped disccussion related to the project.
- TeamWeekGant will be used to create the gantt chart of the project.
- Open source libraries will be used during the implementation.
- Images that will be edited in the application will be obtained from either the gallery of the user's device or their camera.
- Server will be used for computations.
- Flutter will be used to ensure Photonom will be usable cross-platform.
- Python will be used for the implementation of neural networks.
- For the implementation of image inpainting four different datasets will be used for the training of the neural networks. The datasets are the following:

  - **CelebA [8]:** Faces
  - **Paris [9]:** Buildings
  - **ImageNet [10]:** Objects
  - **Places2 [11]:** Natural scenes

- For the modification of facial expressions feature, we will use EmotioNet dataset [12] during the trainining.

- For the assessment of image quality the neural networks will be trained with following two datasets:

  - **AVA [3]:** To be able to consider aesthetic properties of the photograph
  - **TID2013 [13]:** To be able to consider technical details such as contrast, brightness etc.

- In case additional computation power is needed Google Cloud Platform or Amazon Web Services' free credit may be used [7] [6].

- Users will be able to download and use the app without any fees or in app purchases.

- Publishing the app in Google Play Store may require additional costs.

- The pictures either uploaded to the application or taken by the application will not be shared with other applications or people without the user's consent.

- In case of a security breach the operations will be aborted and reported to the user.

- During the use of datasets and libraries their license will be taken into account and necessary producers will be followed to get the permission.

2.5 System models

2.5.1 Scenarios

2.5.1.1 Editing Vacation Photos

Scenario Name: Editing Vacation Photos

Participating Actor Instance: Irmak:User

Flow of Events:

1. Whilst Irmak is looking through her photos from her most recent vacation to Italy, she realises that the photo she took in front of Saint Mark's Basilica is too crowded and the dome of the church didn't completely fit the frame. In order to fix these she uploads this photo to Photonom.

2. First she chooses the "Remove Objects" tool in order to delete some of the crowd in the background. She selects the people she wants to remove and clicks apply.

3. Photonom removes the selected objects and shows Irmak alternatives of the end result. Irmak likes the 2nd choice because the blending of the background seems better so she chooses that one.

4. Then Irmak chooses another tool, Image Stitching, to add the missing dome to the picture. After selecting the tool, she highlights the missing part of her photo.

5. Photonom is able to find a picture from web for the dome of Saint Mark's Basilica dues to its fame and stitches the rest of the dome into Irmak's photo.

6. Irmak is satisfied with the resulting photo and accepts the changes.

7. Before leaving Photonom Irmak looks at the history of her photo to see how the quality of her photo increased with her changes. She sees that the final photo has the highest photo and leaves the History to look at her photo again.

8. Irmak wants to share this photo in Instagram so she goes to the share feature of Photonom and selects Instagram.

2.5.1.2 Deciding What to Post

Scenario Name: Deciding What to Post

Participating Actor Instance: Zeki:User

Flow of Events:

1. Zeki is looking through his photo gallery to find what he should post next on his social media. He finds a couple of options but cannot decide on a single one. In order to evaluate the quality of each photo he opens Photonom.

2. Zeki goes to upload the photos and selects all of his options

3. Photonom evaluates each of the photos Zeki uploaded, and once the evaluation is completed shows each photo to Berk with their respective quality score.

4. Zeki selects the photo with the highest score.

5. Photonom asks Zeki whether he wants to edit the photo or not.

6. Zeki is satisfied with the photo so he saves the photo to his gallery and exits Photonom.

2.5.2 Use Case Model

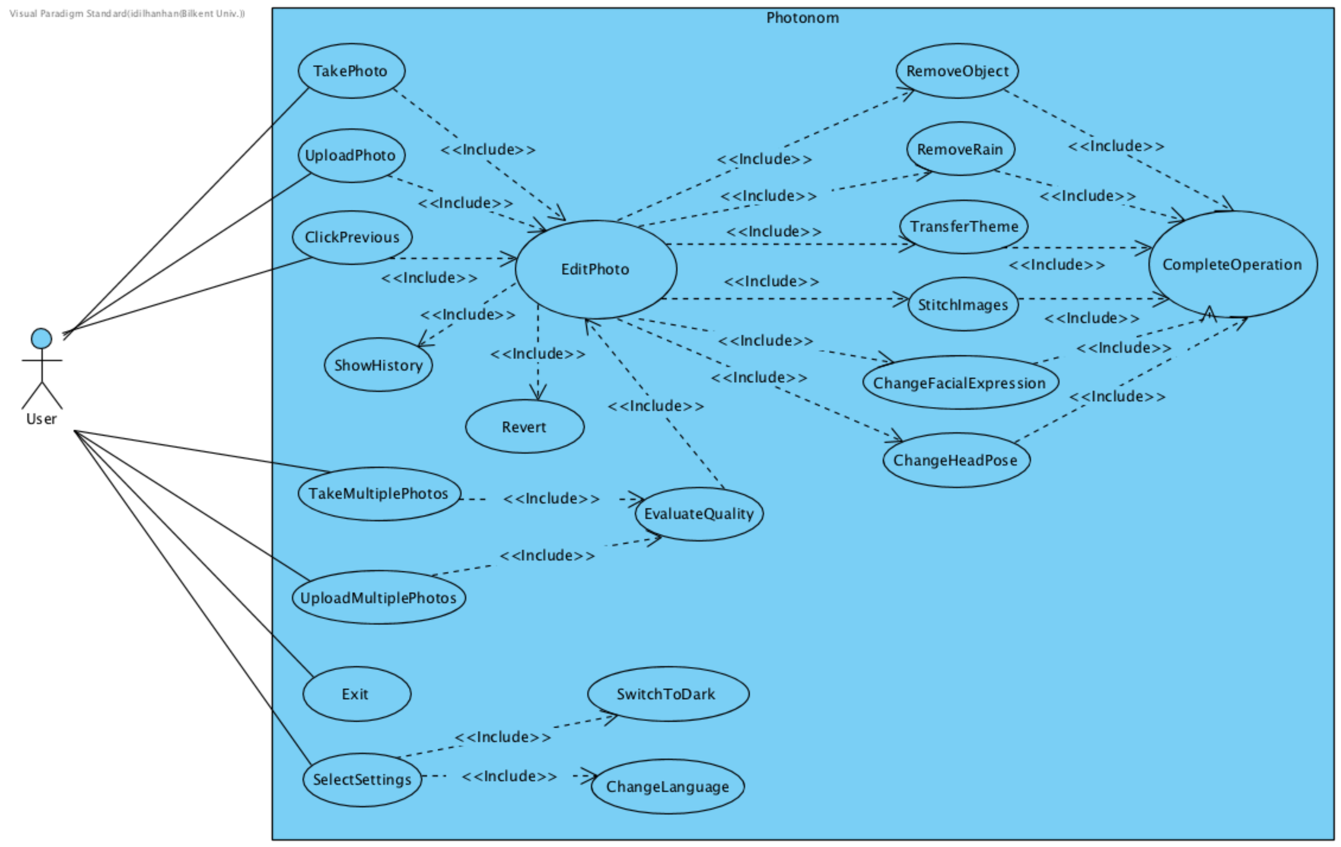
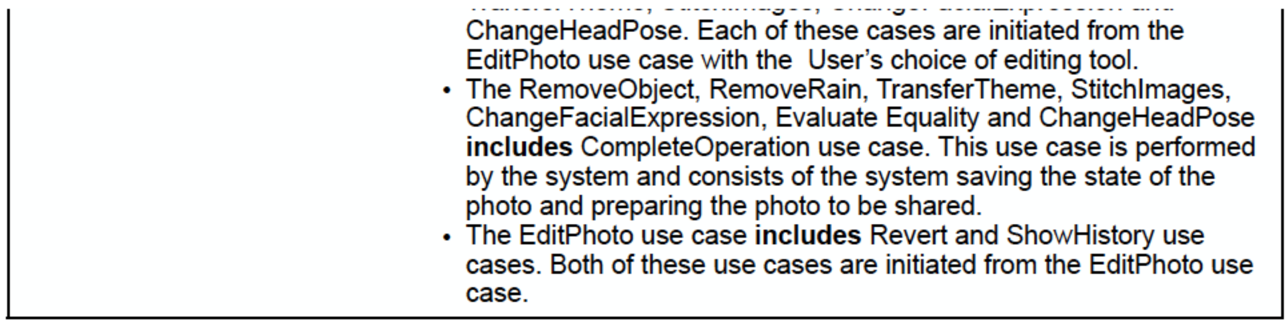


Figure 1: Use case diagram

2.5.2.1 EditPhoto Textual Representation

Use Case Name:	EditPhoto
Participating Actor:	User
Entry Condition:	• User has selected the photo they want to edit.
Exit Condition:	• User has edited the photo.
Flow of Events:	
1.Basic Flow:	
1.1. User activates the “Edit Photo” function by taking a photo or uploading a photo to edit.	
	1.2. The application displays the chosen photo and the tools that can be used to edit the photo.
1.3. If User selects the remove object tool.	
	1.3.1. The application activates the remove object tool.
1.3.2. User selects the object to remove.	
	1.3.3. The application removes the object, shows the resulting photo to User and asks for approval.
1.3.4. User approves the changes.	
	1.3.5. The application saves the current state.
1.4. If User selects the rain removal tool.	
	1.4.1. The application removes the rain, shows the resulting photo to User and asks for approval.
1.4.2. User approves the changes.	
	1.4.3. The application saves the current state.
1.5. If User selects the theme transfer tool.	
	1.5.1. The application prompts User to provide a photo to be used as theme.
1.5.2. User supplies a theme photo.	
	1.5.3. The application transfer the theme, shows the resulting photo to User and asks for approval.
1.5.4. User approves the changes.	
	1.5.5. The application saves the current state.
1.6. If User selects image sticking tool.	
	1.6.1. The application prompts User to provide other photos to stitch.
1.6.2. User supplies additional photo.	
	1.6.3. The application stitches the image together, shows the resulting photo to User and asks for approval.

1.6.4. User approves the changes.	
	1.6.5. The application saves the current state.
1.7. If User selects change facial expression tool.	
	1.7.1. The application highlights the faces in the photo and activates the change facial expression tool.
1.7.2. User selects the face and does the changes.	
	1.7.3. The application changes the facial expression, shows the resulting photo to User and asks for approval.
1.7.4. User approves the changes.	
	1.7.5. The application saves the current state.
1.8. If User selects change head pose tool.	
	1.8.1. The application highlights the faces in the photo and activates the change head pose tool.
1.8.2. User selects the face and does the changes.	
	1.8.3. The application changes the head position, shows the resulting photo to User and asks for approval.
1.8.4. User approves the changes.	
	1.8.5. The application saves the current state.
1.9. User downloads the final product or choses another tool to use.	
2.Alternative Flow:	
2.1. User clicks on revert.	
	2.1.1. The application reverts the last change made to the photo by getting previous version of the photo.
2.2. User clicks show history.	
	2.2.1. The application displays a map that has a node representing each past version of the photo with the image quality of that photo written next to the respective node.
2.2.2. The use clicks on one of the past nodes.	
	2.2.3. The application gets the version of the photo represented by the clicked node and brings that version.
Special/Quality Requirements:	• TakePhoto, UploadPhoto and ContinuePrevious use cases **include** the EditPhoto use case. These use cases are all related to choosing a photo to edit. After a single photo is taken or chosen each of them initiate the EditPhoto use case. • The EditPhoto use case **includes** the RemoveObject, RemoveRain, TransferTheme, StitchImages, ChangeFacialExpression and

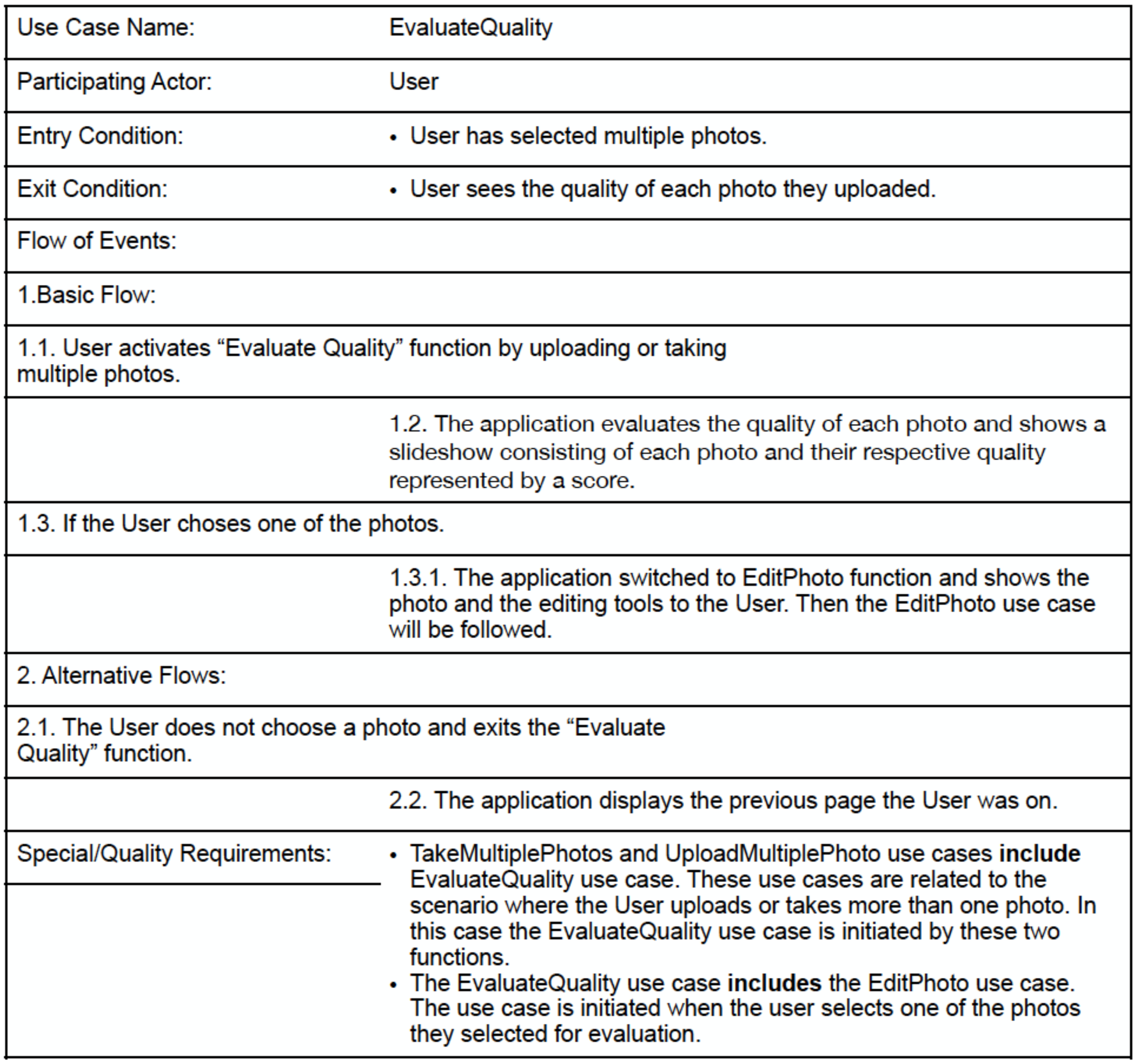


ChangeHeadPose. Each of these cases are initiated from the EditPhoto use case with the User's choice of editing tool.

- The RemoveObject, RemoveRain, TransferTheme, StitchImages, ChangeFacialExpression, Evaluate Equality and ChangeHeadPose **includes** CompleteOperation use case. This use case is performed by the system and consists of the system saving the state of the photo and preparing the photo to be shared.
- The EditPhoto use case **includes** Revert and ShowHistory use cases. Both of these use cases are initiated from the EditPhoto use case.

Figure 2: EditPhoto Use Case Textual Representation

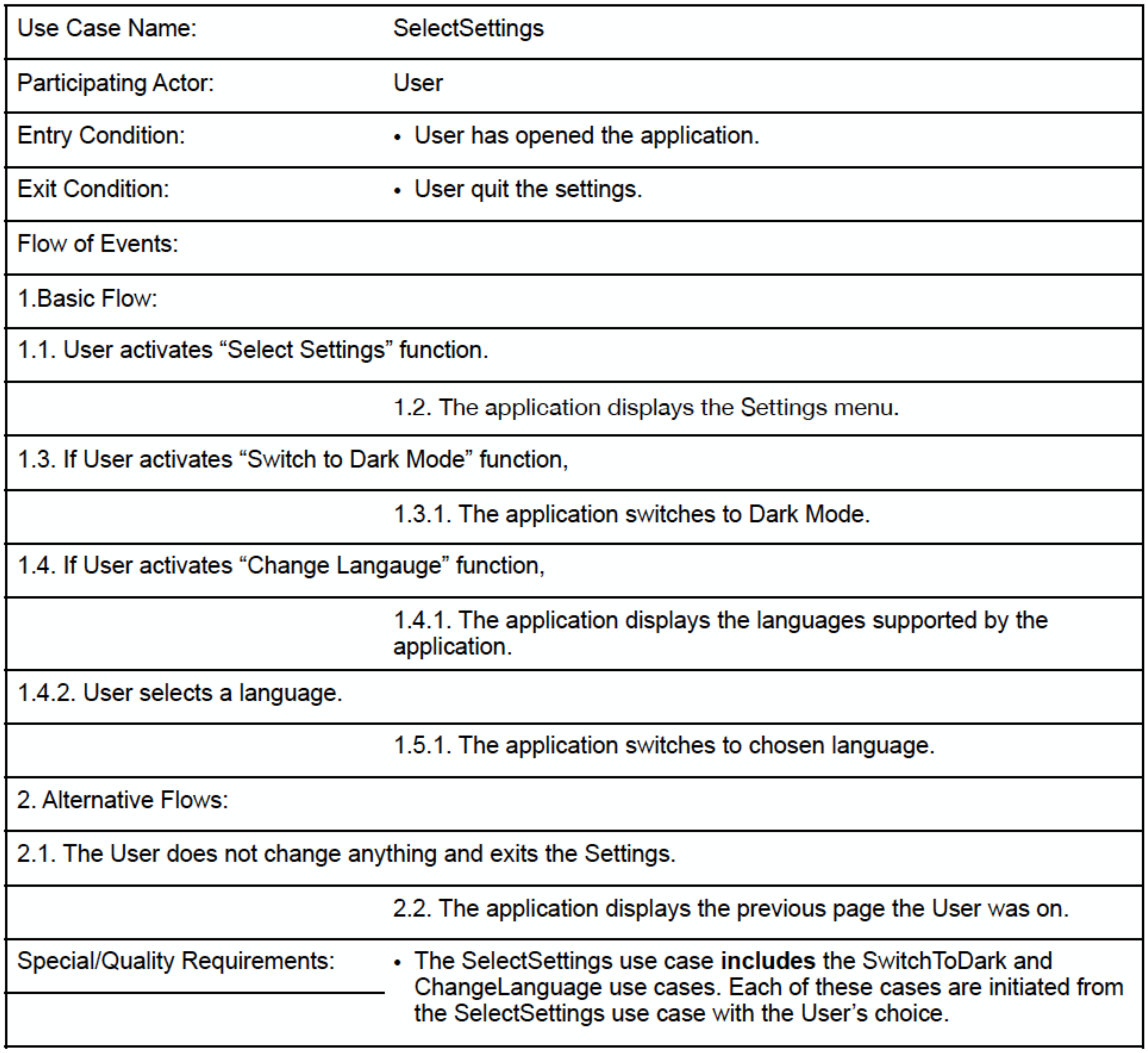
2.5.2.2 EvaluateQuality Textual Representation



Use Case Name:	EvaluateQuality
Participating Actor:	User
Entry Condition:	• User has selected multiple photos.
Exit Condition:	• User sees the quality of each photo they uploaded.
Flow of Events:	
1.Basic Flow:	
1.1. User activates “Evaluate Quality” function by uploading or taking multiple photos.	
	1.2. The application evaluates the quality of each photo and shows a slideshow consisting of each photo and their respective quality represented by a score.
1.3. If the User choses one of the photos.	
	1.3.1. The application switched to EditPhoto function and shows the photo and the editing tools to the User. Then the EditPhoto use case will be followed.
2. Alternative Flows:	
2.1. The User does not choose a photo and exits the “Evaluate Quality” function.	
	2.2. The application displays the previous page the User was on.
Special/Quality Requirements:	• TakeMultiplePhotos and UploadMultiplePhoto use cases **include** EvaluateQuality use case. These use cases are related to the scenario where the User uploads or takes more than one photo. In this case the EvaluateQuality use case is initiated by these two functions. • The EvaluateQuality use case **includes** the EditPhoto use case. The use case is initiated when the user selects one of the photos they selected for evaluation.

Figure 3: EvaluateQuality Use Case Textual Representation

2.5.2.3 SelectSettings Textual Representation



Use Case Name:	SelectSettings
Participating Actor:	User
Entry Condition:	• User has opened the application.
Exit Condition:	• User quit the settings.
Flow of Events:	
1.Basic Flow:	
1.1. User activates “Select Settings” function.	
	1.2. The application displays the Settings menu.
1.3. If User activates “Switch to Dark Mode” function,	
	1.3.1. The application switches to Dark Mode.
1.4. If User activates “Change Langauge” function,	
	1.4.1. The application displays the languages supported by the application.
1.4.2. User selects a language.	
	1.5.1. The application switches to chosen language.
2. Alternative Flows:	
2.1. The User does not change anything and exits the Settings.	
	2.2. The application displays the previous page the User was on.
Special/Quality Requirements:	• The SelectSettings use case **includes** the SwitchToDark and ChangeLanguage use cases. Each of these cases are initiated from the SelectSettings use case with the User’s choice.

Figure 4: SelectSettings Use Case Textual Representation

2.5.3 Object and Class Model

The class diagram draft for our project can be seen below in Figure 5.

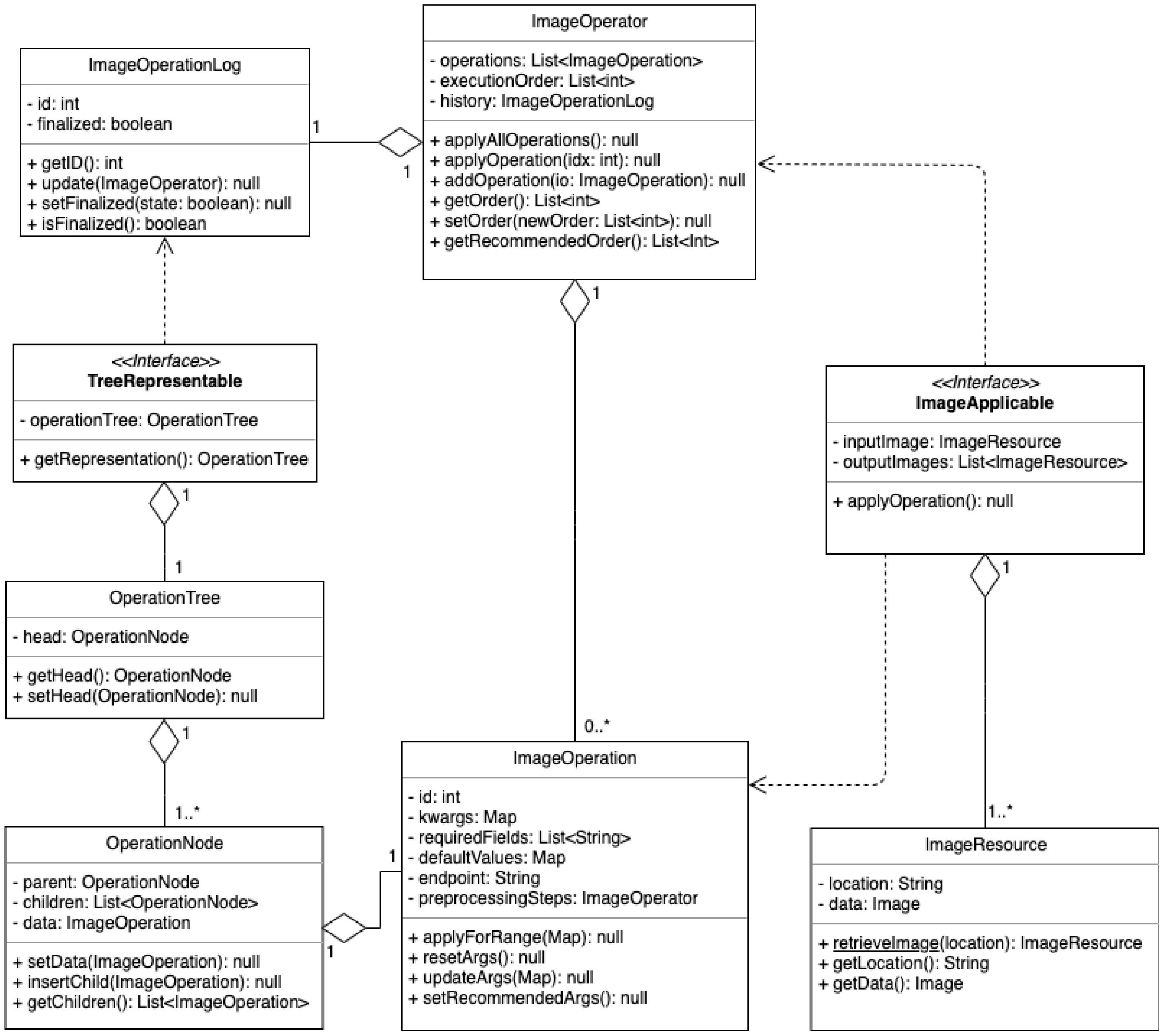

ImageOperationLog
- id: int
- finalized: boolean
+ getID(): int
+ update(ImageOperator): null
+ setFinalized(state: boolean): null
+ isFinalized(): boolean
ImageOperator
- operations: List<ImageOperation>
- executionOrder: List<int>
- history: ImageOperationLog
+ applyAllOperations(): null
+ applyOperation(idx: int): null
+ addOperation(io: ImageOperation): null
+ getOrder(): List<int>
+ setOrder(newOrder: List<int>): null
+ getRecommendedOrder(): List<Int>
<<Interface>>
TreeRepresentable
- operationTree: OperationTree
+ getRepresentation(): OperationTree
<<Interface>>
ImageApplicable
- inputImage: ImageResource
- outputImages: List<ImageResource>
+ applyOperation(): null
OperationTree
- head: OperationNode
+ getHead(): OperationNode
+ setHead(OperationNode): null
OperationNode
- parent: OperationNode
- children: List<OperationNode>
- data: ImageOperation
+ setData(ImageOperation): null
+ insertChild(ImageOperation): null
+ getChildren(): List<ImageOperation>
ImageOperation
- id: int
- kwargs: Map
- requiredFields: List<String>
- defaultValues: Map
- endpoint: String
- preprocessingSteps: ImageOperator
+ applyForRange(Map): null
+ resetArgs(): null
+ updateArgs(Map): null
+ setRecommendedArgs(): null
ImageResource
- location: String
- data: Image
+ retrieveImage(location): ImageResource
+ getLocation(): String
+ getData(): Image
1
1
1
1
1
1
1
1..*
1
0..*
1
1
1
1..*

Figure 5: Class diagram for Photonom

This diagram above has 8 classes in total. Now, we will define these classes and briefly explain their responsibilities and content.

2.5.3.1 ImageResource

This class is a wrapper resource around our images. It has two different attributes:

- **location:** This attribute is of type String and stores the location of the image in the user's mobile phone. It is immutable after initialization. If the image is an original image taken from the user's gallery, the location of the image will be the original location. If the user has applied some operations on that image, it will be saved into the application storage in the user's phone until the user completes the operation sequence or starts performing on another image.

- **data:** This attribute will be of type Image and it will contain the image itself which the operations will be applied on. This attribute will be immutable after initialization. If any operation is applied, the new Image will persist in a new ImageResource object.

The methods of this class will be trivial getters and setters. This class will be used as a manager between the image itself and its operators.

- **retrieveImage(location):** This method is a static method and it is used to create the ImageResource instance. It accepts one argument for location and the argument type should be String. If the given location represents a valid image, this function returns the corresponding ImageResource object after setting the location and data attributes. If there is a problem regarding the type of the image or the given location, it will handle these either by applying some operations to validate the Image or the input or by raising an error which will be propagated to the user interface.

- **getLocation():** This method is a basic getter method which returns the location attribute of the instance it is called on. The return type is String.

- **getData():** This method is a basic getter method which returns the data attribute of the instance it is called on. The return type is Image.

2.5.3.2 ImageApplicable

This is an interface. If a class implements this interface it means that the class implementing can be applied to an image. It has the following attributes.

- **inputImage:** This attribute is an ImageResource attribute. It is the input to the ImageOperation, it does not necessarily is the original image. It can also be an product of another ImageOperation.

- **outputImages:** This attribute is a list of ImageResources. It contains the product or products if the operation is applied on a range of arguments differently.

The method of the interface are written to apply the operation.

- **applyOperation:** The classes that implement this interface should also implement this method which applies the operation to the input image and saves the output into the outputImages.

2.5.3.3 ImageOperation

This class implements ImageApplicable interface. It represents a single operation that can be applied on the cloud. If applyOperation method is called upon an instance of ImageOperation, the input image and the kwargs are sent to the endpoint with a POST request so that the server receives the request applies the operation and then returning a response. The required fields should be included in the kwargs, otherwise the request will not be sent. The class has the following fields.

- **id:** This id, which is of type int, represents the type of the image operation. In other words, its the 'type' but id is fancier.

- **kwargs:** This attribute is of type Map. It contains key and values corresponding to the modifiable parameters of the operation. It is send inside the POST request which is explained above.

- **requiredFields:** This attribute is of type List and it contains String values. If any of the keys in this list is missing inside the kwargs attribute, and there is no default value set for the required field, there will be an exception raised and a message propagated to the user interface.

- **defaultValues:** This attribute is also of type Map and if there are missing fields in the kwargs Map, the missing values are filled with the default ones.

- **endpoint:** This is the URL to the one of the server's endpoints that will be being listened to. We will send a POST request to this endpoint with the required parameters if the conditions are satisfied and the applyOperation method is called.

- **preprocessingSteps:** This attribute is stored as a list of ImageOperations. Some operations' models may require some preprocessing by default or if its the case that we have less computing power than the proposed minimal. These operations that should be applied before applying the main operation are stored here.

It has the following methods.

- **applyForRange(kwargs):** This method takes and argument called kwargs and which is of type Map. The Map contains key and value pairs which in this case pairs are lists. The user's actions may be propagated to the surver as multiple ranges so this method applies the all combinations of operation attributes and appends them to outputImages attribute of the ImageApplicable interface.

- **resetArgs():** Resets kwargs with the default values.
- **updateArgs(kwargs):** This method takes input a Map object and updates the kwargs attribute.
- **setRecommendedArgs():** This method sets the kwargs with our recommended parameters for the methods.

2.5.3.4 ImageOperator

This class is the main operational unit in our system. It also implements ImageApplicable interface. It has the following attributes.

- **operations:** This attribute is a list of ImageOperation instances. It contains the operations that the operator is responsible for applying.
- **executionOrder:** This attribute is a list of integers. It has the same length as the operations attribute above. If operator is executed, operator will apply the operations following the index order of this attribute.
- **history:** This attribute is of type ImageOperationLog. It will be used to get the tree representation of user's actions.

This class contains the following methods.

- **applyAllOperations():** This method applies all operations that are in operations attribute in the order of executionOrder attribute.
- **applyOperation(idx):** This method accepts an argument idx of type int. This argument should be between 0 and the length of operations. It applies the operation at the given index.

- **addOperation(imageOperation):** This method accepts an argument imageOperation of type ImageOperation. It appends the operation to the operations attribute and appends its index to the executionOrder attribute's end.

- **getOrder():** Returns the executionOrder attribute.

- **setOrder(order):** This method accepts an argument of type list of integers. It validates the order argument and if it's valid it sets the executionOrder.

- **getRecommendedOrder()** This method returns the recommended order by looking at the operations attribute.

2.5.3.5 TreeRepresentable

This is an interface. If a class implements this interface it means that the class can be represented using a tree view. The inspiration for this interface comes from git's branching logic.

- **operationTree:** This attribute is of type OperationTree.

Following are the methods of this interface.

- **getRepresentation():** This method will return a representation of the operationTree attribute in a way that can be understood and transformed by the user interface.

2.5.3.6 ImageOperationLog

This class is a basic history and version logger. It implements TreeRepresentable interface. The attributes are as follows.

- **id:** This attribute is of type int and is a unique representation of users set of operations.

- **finalized:** This attribute is of type boolean and if the user finished applying operations on an image it will be set as true.

The following are the methods of this class.

- **getID():** Returns the id attribute.
- **update(imageOperation):** Adds the imageOperation of type ImageOperation to the log.
- **setFinalized(state):** This method accepts an attribute of type boolean and sets the finalized attribute to that value.
- **isFinalized():** Returns finalized attribute.

2.5.3.7 OperationTree

This class is basically a tree implementation which is used for versioning. It represents the timeline and the branching of the image.

- **head:** The root node of the tree. It is of type OperationNode.

The following are the methods of this class.

- **getHead():** Returns the head.
- **setHead(node):** This method accepts an argument of type OperationNode and it sets the head accordingly.

2.5.3.8 OperationNode

This class is basically a tree node class. Each node represents a version of the image.

- **parent:** This attribute is of type OperationNode, it is the parent of the current node or null if the node is the root node.

- **children:** This attribute is a list of OperationNode, it is the children of the current node or null if the node is a leaf node.

- **data:** This attribute is of type ImageOperation and stores the version of the image.

Methods

- **setData(imageOperation):** This method takes an argument of type ImageOperation and it sets the data attribute.

- **insertChild(imageOperation):** This method takes an argument of type ImageOperation and creates child nodes with the corresponding data attributes.

- **getChildren():** This method returns children attribute.

2.5.4 Dynamic Models

2.5.4.1 Sequence Diagram

The sequence diagrams of Photonom are given below. For the sake of simplicity, there are multiple diagrams which are separated with respect to the functionalities.

2.5.4.1.1 Choose Image

In order to start editing a photo, the user has to first choose the photo that is going to be worked on. When the user clicks the plus sign on the main screen, s/he will be given two choices to choose the photo. The user can either choose the photo/s from the gallery or take photo/s using the camera. Depending on the number of photos chosen the user will be directed to different screens and the number of the photos will be checked using checkSinglePhoto() function. If the user selects one photo from the gallery or takes only one photo, Tool Selection Screen will be opened. In the multiple photos case, Quality Evaluation Screen which demonstrates the quality scores of each photo will be opened. Since the application will be an interactive photo editing tool, the app is going to only demonstrate the scores of the photos, not choose the photo that will be used. After deciding which photo will be used, the user will be directed to the Tool Selection Screen as in the case of single photo. For the sake of simplicity, the sequence diagram of different tools are separate from this diagram. The way of going to different tools' screens and coming back to the Tool Selection Screen will be explained further in the related sections. After each operation with a tool, the user will come back to the Tool Selection Screen for further operations. In the Tool Selection Screen in addition to choosing the tools, the user can also display the history of the operations and different versions of the photo as a result of these different operations. When the user clicks the history button, s/he will be led to the History Screen. The user can go back to the Tool Selection Screen by either choosing a version to work on or clicking the back button. If the user decides to finalize the editing of the photo, by clicking the finalize editing button s/he can be directed to the Editing Completed Screen. In this screen the user can share or save the final version of the photo or go back to the main screen by choosing to work with a new photo. In addition to starting the editing by choosing a new photo with the use of a gallery or camera, the user can also continue to the last edit s/he was working on. In the main screen there will be a button to enable the user work on the last photo. After being led to the Tool Selection Screen, the user can follow the same way that s/he can follow with the editing of the new photo case. A loop covers all these operations since starting the editing of a photo and finalizing it is a repetitive process. The interactions of the user with different screens and relationships between different objects while choosing a photo can be seen in Fig. 6.

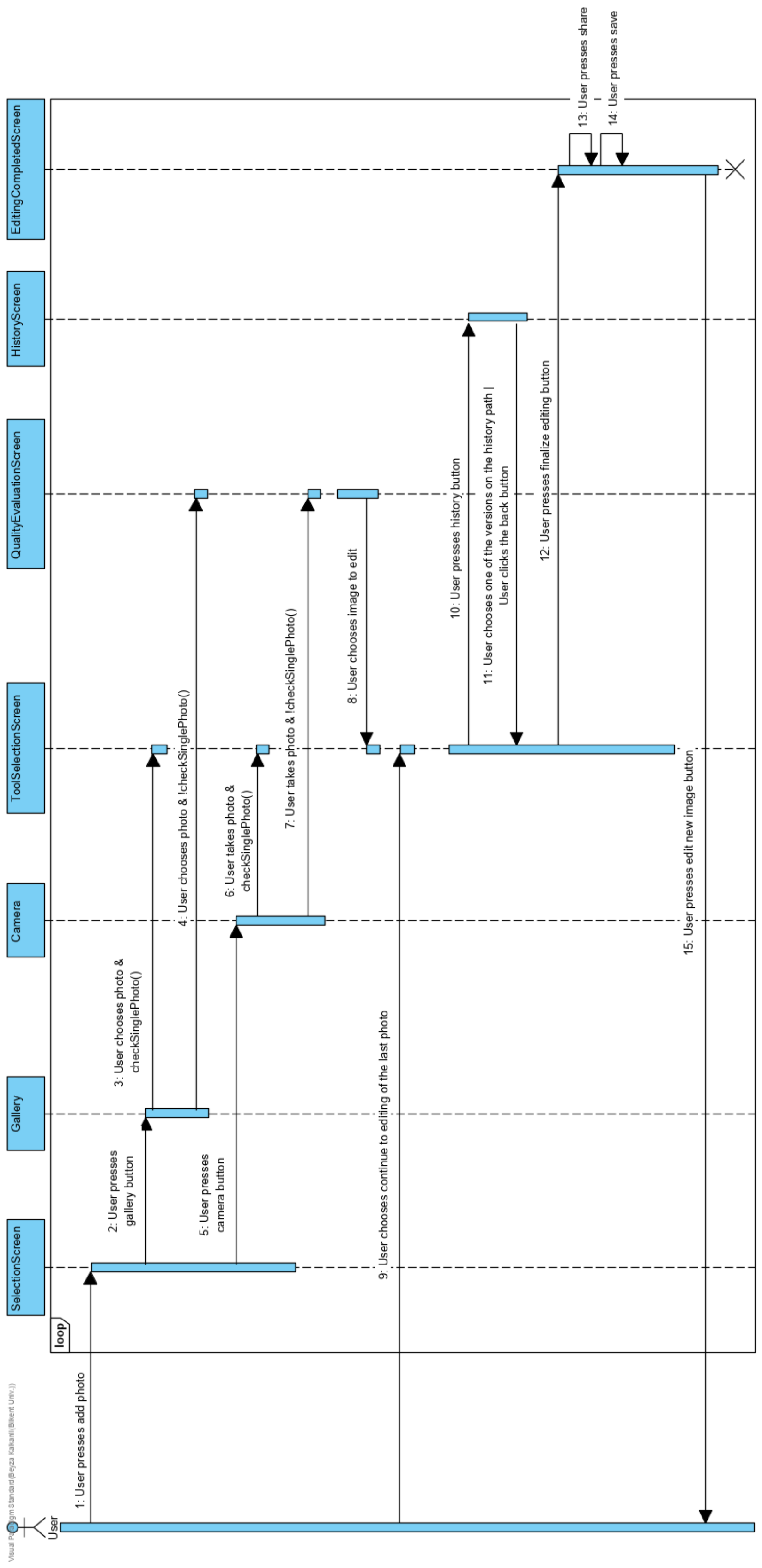
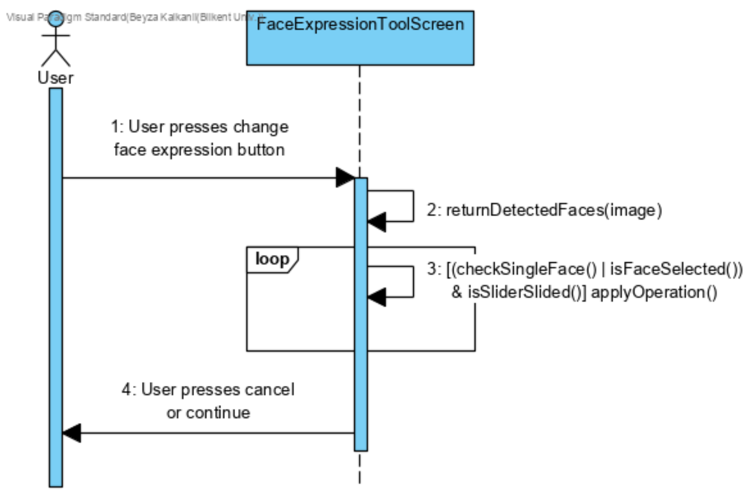

User
SelectionScreen
Gallery
Camera
ToolSelectionScreen
QualityEvaluationScreen
HistoryScreen
EditingCompletedScreen
loop
1: User presses add photo
2: User presses gallery button
3: User chooses photo & checkSinglePhoto()
4: User chooses photo & !checkSinglePhoto()
5: User presses camera button
6: User takes photo & checkSinglePhoto()
7: User takes photo & !checkSinglePhoto()
8: User chooses image to edit
9: User chooses continue to editing of the last photo
10: User presses history button
11: User chooses one of the versions on the history path | User clicks the back button
12: User presses finalize editing button
13: User presses share
14: User presses save
15: User presses edit new image button

Figure 6: Sequence diagram for Choosing Image

2.5.4.1.2 Face Expression Modification

When the user clicks the change face expression button in the Tool Selection Screen, s/he will be directed to Face Expression Tool Screen. In this screen the faces on the chosen image will be demonstrated with bounding boxes around them. In order to show the faces in the image, as soon as the user chooses to use this operation returnDetectedFaces(image) will return the faces in the given image. If there are multiple faces in the image, the user will be able to choose the face that s/he is going to work on from the faces given as options. In case that when there is only one face in the image, automatically this face will be chosen by the application. If the user chooses the face to work on and changes the value of the slider on this screen which corresponds to the intensity level of the expression, then this face expression will be reflected to the chosen face. Since these operations can be applied multiple times to one face or to the different faces, these operations are shown within a loop. If the user presses the cancel button, the user will be directed to the Tool Selection Screen without applying the changes to the image. If the user presses continue, the resulting image will be the base of the new operations in the Tool Selection Screen. Since the effect of cancel and continue buttons are the same for all tools, they are not going to be explained again in the other tools' sequence diagrams. In Fig. 7, the interactions between the user and the different objects while changing the face expressions can be observed.



Visual Paradigm Standard(Beyza Kalkanli(Bilkent Uni
User
FaceExpressionToolScreen
1: User presses change face expression button
2: returnDetectedFaces(image)
loop
3: [(checkSingleFace() | isFaceSelected()) & isSliderSlided()] applyOperation()
4: User presses cancel or continue

Figure 7: Sequence diagram for Face Expression Modification

2.5.4.1.3 Object Removal

The user can reach the Object Removal Tool Screen by pressing the remove objects operation button in the Tool Selection Screen. The user will paint the object that s/he want to remove from the image. After the user decides the object s/he wants to remove, s/he will press the fill button. This will trigger the generateNewFill() function which will return one alternative covering for the background after removal. If the user presses the previous button and there is a previous fill, then the previous version will be shown using the loadPreviousFill() function. If the user used the previous button and came back to the older filling versions, s/he can go to the newer filling versions by pressing the next button. In this case whether is there a next filling option or not will be checked and loadNextFill() function will be called to load the next version. After the user presses either previous, next or fill buttons the related functions mentioned above will be called. In order to apply the changes to the photo applyOperation() function will be called. These operations will be a loop since they are repetitive operations. Also, there will be a slider in this screen to let the user to choose different brush sizes. In the case of a change of the brush size, changeBrushSize(newSize) function will be called to change the current brush size into the new one. There will be another loop which also covers this case, since this can be also repeated multiple times. There is a need of using two nested loops because changing the brush size is not a mandatory operation for the user. As in the case of the previous tool, the user have two options to go back to the Tool Selection Screen, either by pressing cancel or continue. The interactions explained above while removing the objects from a photo can be seen in Fig. 8.

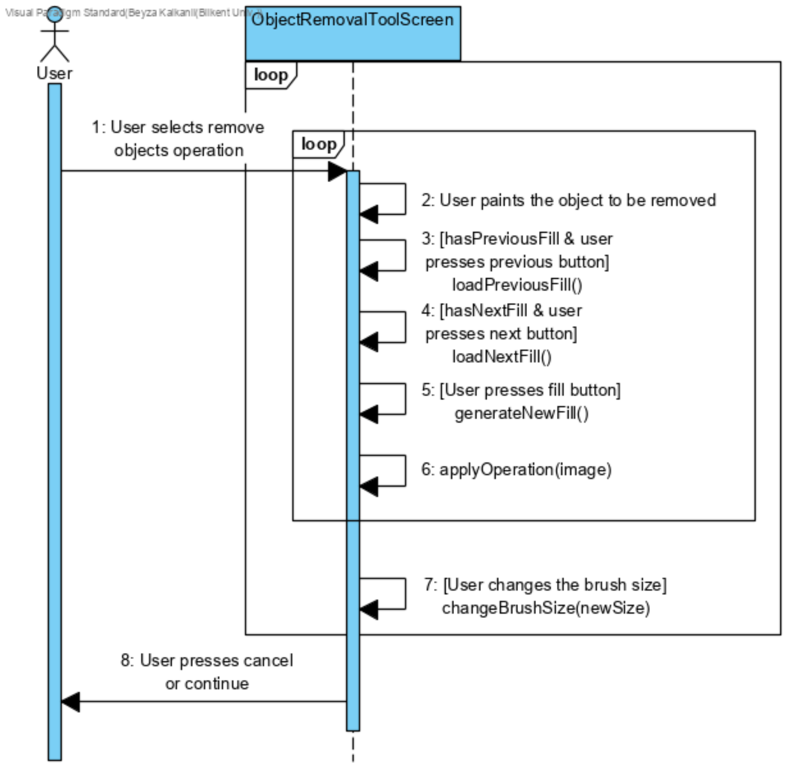

Visual Paradigm Standard(Beyza Kalkanli(Bilkent Univ
User
ObjectRemovalToolScreen
loop
loop
1: User selects remove objects operation
2: User paints the object to be removed
3: [hasPreviousFill & user presses previous button] loadPreviousFill()
4: [hasNextFill & user presses next button] loadNextFill()
5: [User presses fill button] generateNewFill()
6: applyOperation(image)
7: [User changes the brush size] changeBrushSize(newSize)
8: User presses cancel or continue

Figure 8: Sequence diagram for Object Removal

2.5.4.1.4 Image Stitching

The user will be directed to the Image Stitching Tool Screen when s/he clicks the stitch images button in the Tool Selection Screen. Since this operation requires more than one photo, there will be an interaction between this screen and the Gallery. If the user presses add photo button, s/he will see the Gallery to choose the image to be stitched the current version of the photo. photoNotAlreadySelected() function will be used to check whether the chosen is added before or not. If it is not, the user will be returned to the Image Stitching Tool Screen and see the resulting image after stitching the new image with applyOperation() function. The user can remove the photos that s/he doesn't want to use anymore. When the user decides to remove a photo, it will be checked if the photo exist or not with photoExists() function to be safe while removing the photo. If the photo exists, it will be removed using removePhoto() function. There is a loop on the diagram to underline that these operations are repetitive. Going back to the Tool Selection Screen can be done by clicking cancel or continue. The interactions of the user with different screens and objects while stitching multiple images is shown in Fig. 9.

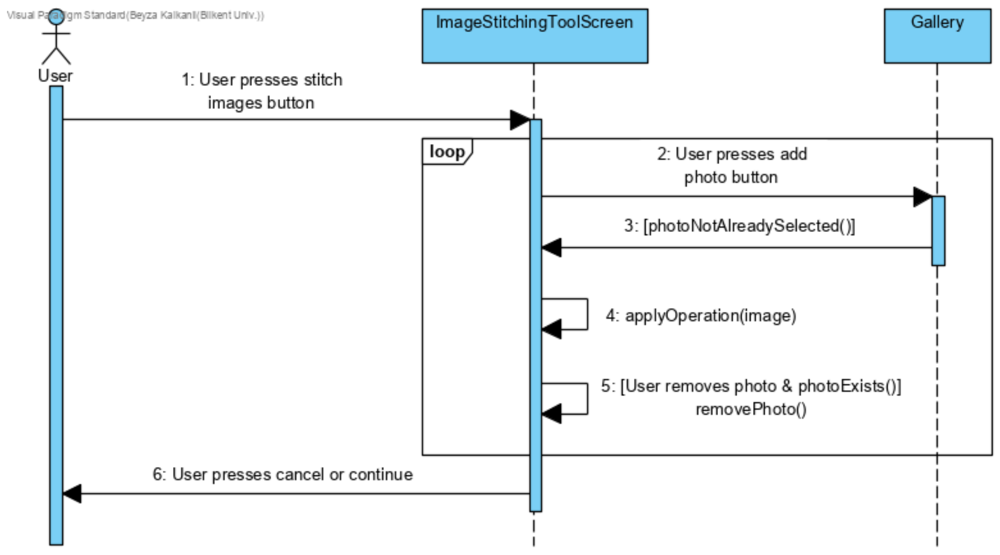

Visual Paradigm Standard(Beyza Kalkanli(Bilkent Univ.))
User
ImageStitchingToolScreen
Gallery
1: User presses stitch images button
loop
2: User presses add photo button
3: [photoNotAlreadySelected()]
4: applyOperation(image)
5: [User removes photo & photoExists()] removePhoto()
6: User presses cancel or continue

Figure 9: Sequence diagram for Image Stitching

2.5.4.1.5 Head Pose Modification

Pressing the change head pose button in the Tool Selection Screen will lead the user to the Head Pose Tool Screen. The process of choosing the person to change the pose of the head will be the same as the process on the face expression modification operation. After the person in the image to work on is decided, it will be checked whether the pose is changed by the user using the sphere below or not. If it has changed, then using the applyOperation() the new pose will be demonstrated to the user. This operation is in loop since the user can apply the change to the head pose of a person of different people. The user can go back to the Tool Selection Screen with cancel or continue buttons. In Fig. 10, the interactions of the user with different screens and relationships between different objects while changing the poses of heads in a photo is demonstrated.

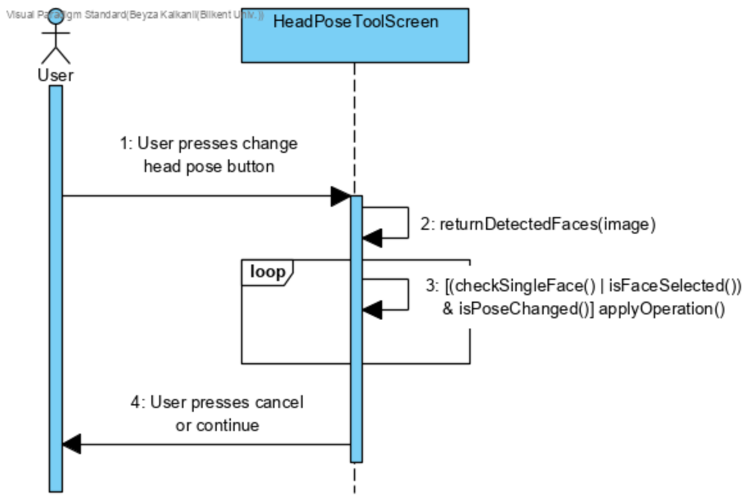

Visual Paradigm Standard(Beyza Kalkanli(Bilkent Univ.))
User
HeadPoseToolScreen
1: User presses change head pose button
2: returnDetectedFaces(image)
loop
3: [(checkSingleFace() | isFaceSelected()) & isPoseChanged()] applyOperation()
4: User presses cancel or continue

Figure 10: Sequence diagram for Head Pose Modification

2.5.4.1.6 Theme Transfer

The user can go to the Theme Transfer Tool Screen by pressing the transfer theme button in the Tool Selection Screen. The user will be able to change the transfer the color palette(theme) of a photo by using the theme of a different photo. The user can either choose the photo from the gallery or take a new photo using the camera. After this decision, according to the theme of the chosen or taken photo the theme of the content photo of the user will be modified using applyOperation() function. Since at one time only one style photo can be used, if the user decided to take a new photo or upload a new photo from the gallery the current style photo will be replaced with the new one. This part is shown as a loop because the application enables user to change the style photo multiple times. Pressing cancel or continue takes the user back to the Tool Selection Screen. The relationships of objects and the user while transferring theme of a photo to another one can be seen in Fig. 11 which is an illustration of what is explained above.

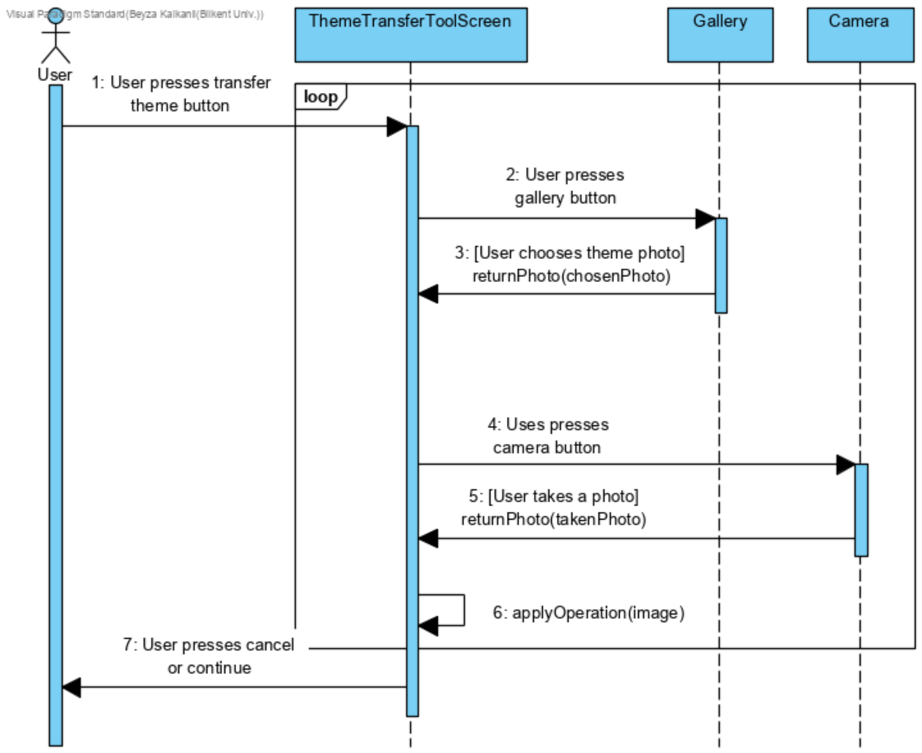

Visual Paradigm Standard(Beyza Kalkanlı(Bilkent Univ.))
User
ThemeTransferToolScreen
Gallery
Camera
loop
1: User presses transfer theme button
2: User presses gallery button
3: [User chooses theme photo] returnPhoto(chosenPhoto)
4: Uses presses camera button
5: [User takes a photo] returnPhoto(takenPhoto)
6: applyOperation(image)
7: User presses cancel or continue

Figure 11: Sequence diagram for Theme Transfer

2.5.4.1.7 Rain Removal

The user will be directed to the Rain Removal Tool Screen, if s/he presses the rain removal tool button in the Tool Selection Screen. In this screen the press of the user to the remove rain button triggers the applyOperation() function. Since this operation removes all the rain drops in the image, applying this operation only once is enough. Since the user will be able to see the original photo and the edited one, there is no need to put these operations into a loop. After the user presses the cancel or continue buttons, s/he can go back to the Tool Selection Screen. The sequence diagram of removing rain in a photo can be seen in Fig. 12.

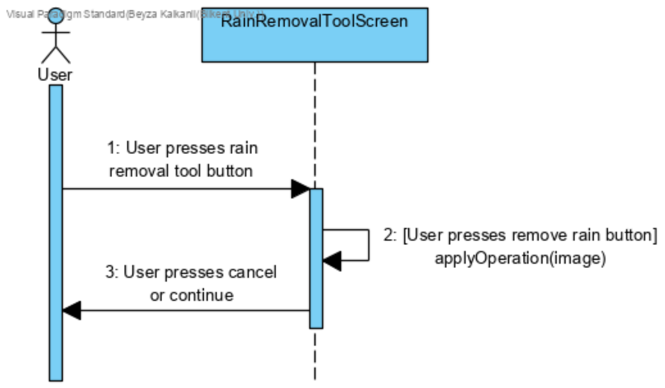

Visual Paradigm Standard(Beyza Kalkanli(Bilkent Univ.))
User
RainRemovalToolScreen
1: User presses rain removal tool button
2: [User presses remove rain button] applyOperation(image)
3: User presses cancel or continue

Figure 12: Sequence diagram for Rain Removal

2.5.4.2 State Diagram

In the state diagram, the behaviour of a single component of the application can be seen.

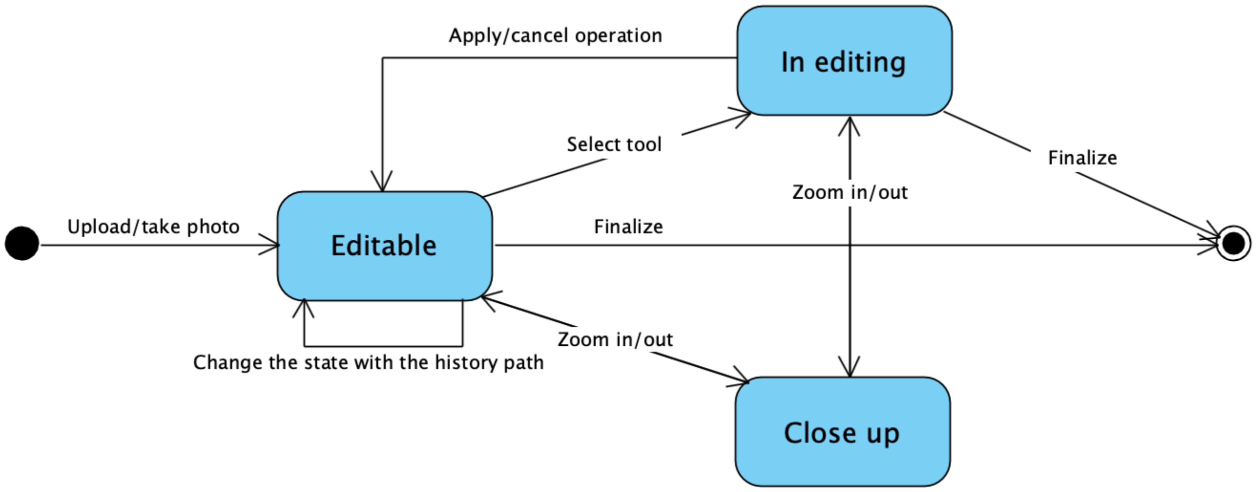
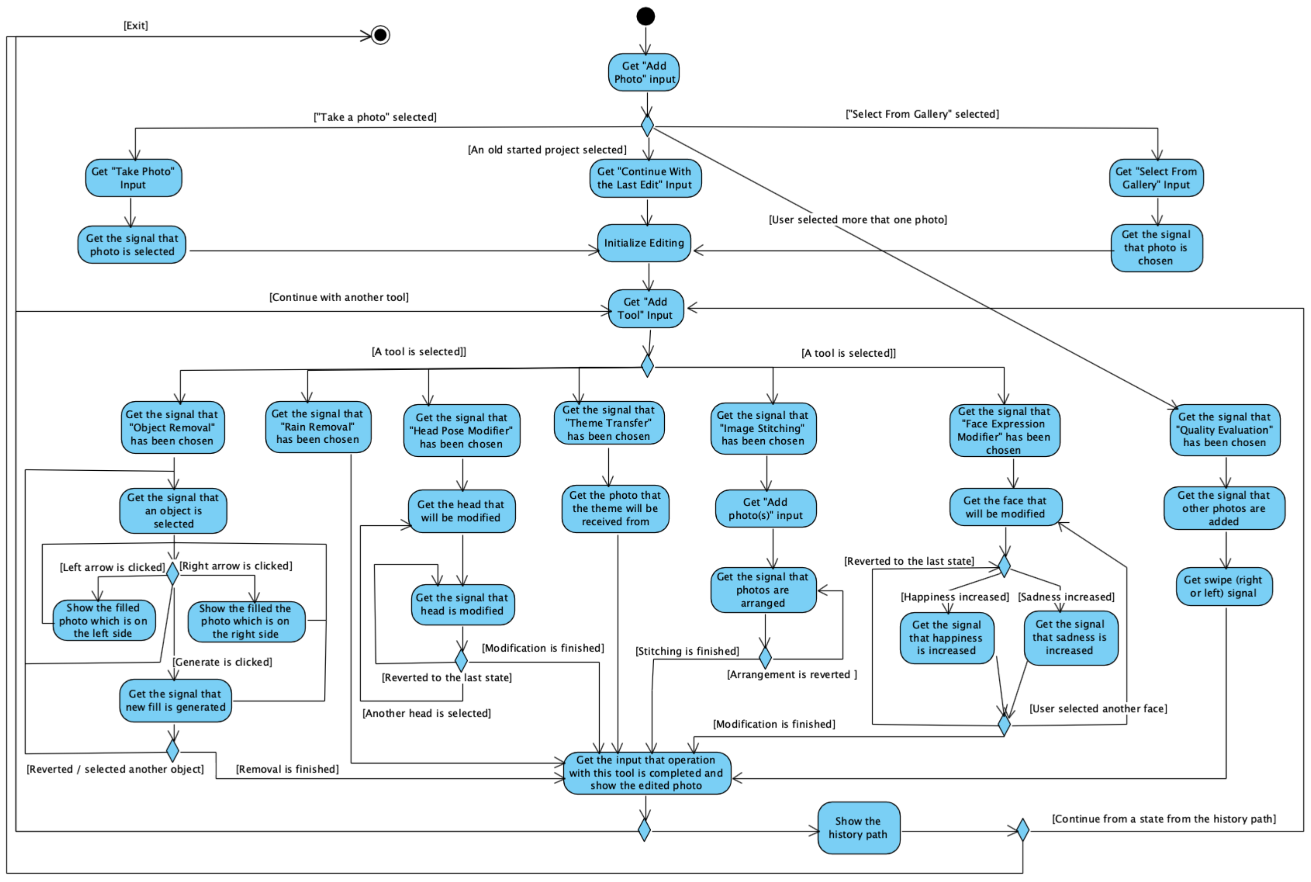


Figure 13: State Diagram for Photo

In the above diagram, the dynamic behaviour of a photo is shown. In the application, a photo which is uploaded or taken by the user will be editable. From this state, a user can select a tool and edit the photo. After editing, there are two options for the user, s/he can finalize the operations on the photo or s/he can can go back to the Editable state. To go back to that state, editing which is done in the previous states needs to be applied or cancelled. In the Editable state, the user can also see the history path and change the state of the operation. Also, it is possible to finalize the operation from the Editable state. Lastly, when user zooms in our our out, the photo will go to the Close up state and go back to the state where it came immediately.

2.5.4.3 Activity Diagram



[Exit]
Get "Add Photo" input
["Take a photo" selected]
["Select From Gallery" selected]
[An old started project selected]
Get "Take Photo" Input
Get "Continue With the Last Edit" Input
Get "Select From Gallery" Input
[User selected more that one photo]
Get the signal that photo is selected
Initialize Editing
Get the signal that photo is chosen
[Continue with another tool]
Get "Add Tool" Input
[A tool is selected]]
[A tool is selected]]
Get the signal that "Object Removal" has been chosen
Get the signal that "Rain Removal" has been chosen
Get the signal that "Head Pose Modifier" has been chosen
Get the signal that "Theme Transfer" has been chosen
Get the signal that "Image Stitching" has been chosen
Get the signal that "Face Expression Modifier" has been chosen
Get the signal that "Quality Evaluation" has been chosen
Get the signal that an object is selected
Get the head that will be modified
Get the photo that the theme will be received from
Get "Add photo(s)" input
Get the face that will be modified
Get the signal that other photos are added
[Left arrow is clicked]
[Right arrow is clicked]
[Reverted to the last state]
Show the filled photo which is on the left side
Show the filled the photo which is on the right side
Get the signal that head is modified
Get the signal that photos are arranged
[Happiness increased]
[Sadness increased]
Get the signal that happiness is increased
Get the signal that sadness is increased
Get swipe (right or left) signal
[Modification is finished]
[Stitching is finished]
[Generate is clicked]
[Reverted to the last state]
[Arrangement is reverted]
Get the signal that new fill is generated
[Another head is selected]
[User selected another face]
[Modification is finished]
[Reverted / selected another object]
[Removal is finished]
Get the input that operation with this tool is completed and show the edited photo
Show the history path
[Continue from a state from the history path]

Figure 14: Activity diagram

In the activity diagram, we see how the system works when the user use tools on photos. Firstly, the system gets an input from the user to initialize editing. The user has three alternatives to continue. S/he can take a, select a photo from gallery or continue with the last edit. If the user selects or takes more than one photo, Quality Evaluation tool will be automatically selected.

Otherwise, the system will initialize editing.

To begin to edit, user needs to choose a tool. There are six options for user. The program gets a signal to continue with a tool.

If the program gets the signal that Object Removal has been chosen, user will chose the object which will be removed. Then, another decision will provided to the user. The application will generate new fill type, when user clicks the generate button. Also, user will be able to see all the generated fills with a click to the left arrow or the right arrow button. After a tool is successfully completed, user can select another object to remove or finish the removal.

If it is a Rain Removal signal, it will automatically remove the rain on the selected photo and the application will finish the operation.

If it is a Head Pose Modifier signal, the program will get input head from user. After user selects the head, user will be able to modify it with the button on the button. User can revert the modifications on the head. Also, when the modification with a head is finished, selecting another head will be available.

If it is a Theme Transfer signal, user selects the another photo to apply its theme to the photo which is being edited. After the application gets the second photo, it will apply the transfer and finished the operation.

If it is an Image Stitching signal, the user will add another photo or photos. Then, the user will arrange the photos. After any arrangement, the user can revert them and the user can finish stitching.

Lastly, it the system gets Facial Expression signal, the user will select the face which will be modified. On the modification part, happiness and sadness will be able to increased. After a modification with a face is finished, the user can revert it, continue with another face or finish the operation.

If the user selected more that one photos at the beginning, the application will get the signal that Quality Evaluation has been chosen and it will show the photos with the quality rate. The user will be able to swipe the screen to see other selected photos.

When an operation with a tool is finished, the system will provide three options. The user can finalize the editing of the photo, s/he can select another tool and apply that or s/he can go to the History Path and see all states of the photo. After selecting the History Path, s/he can select a state and continue with it or s/he can finalize his/her work on that photo.

2.5.5 User Interface

2.5.5.1 Navigational Paths

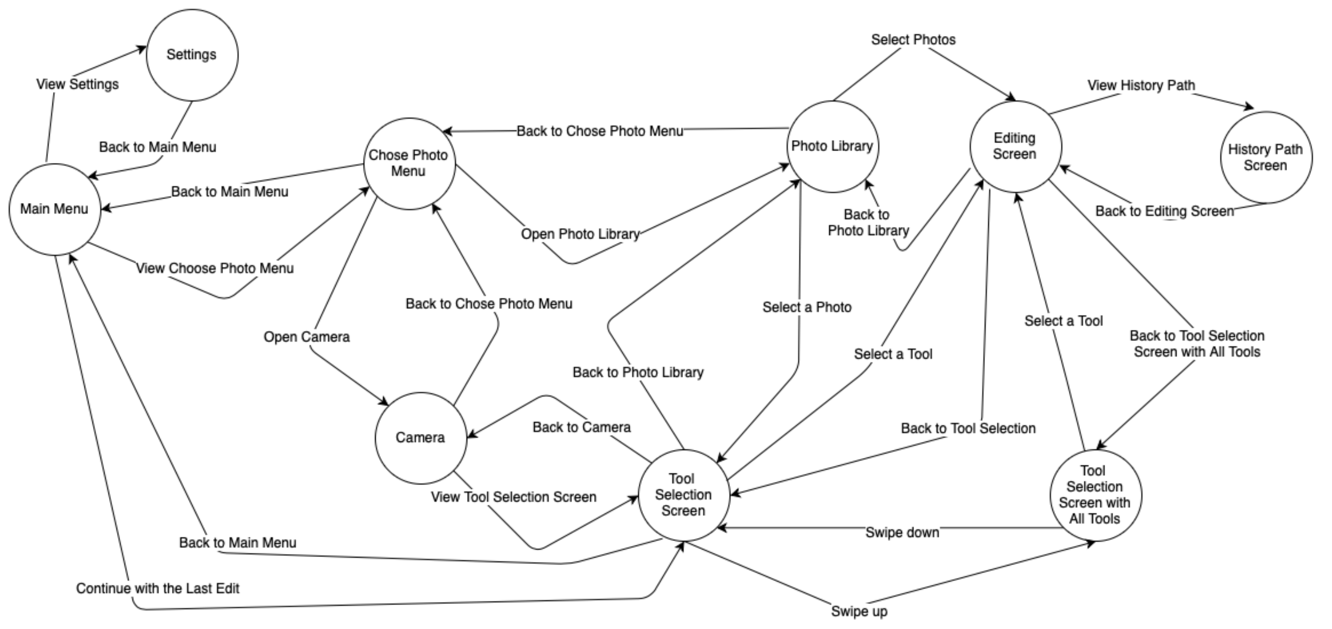


Figure 15: Navigational paths of Photonom

2.5.5.2 Main Menu

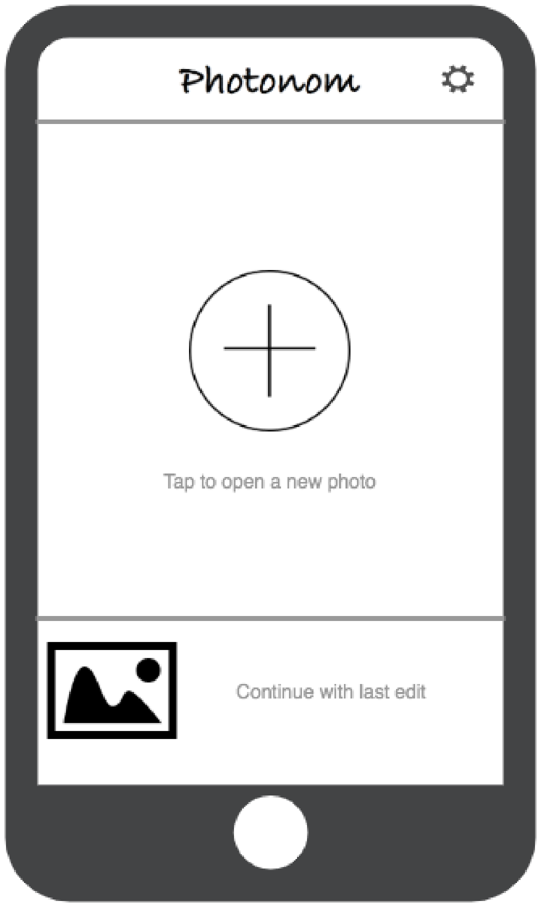


Figure 16: Mockup for main menu

Main Menu is the first page on the screen when the application is started. It includes settings, open a new photo button and continue button. If the user choose that the user chooses to continue, then they will continue editing the last photo they chose for editing.

2.5.5.3 Choose Photo Menu

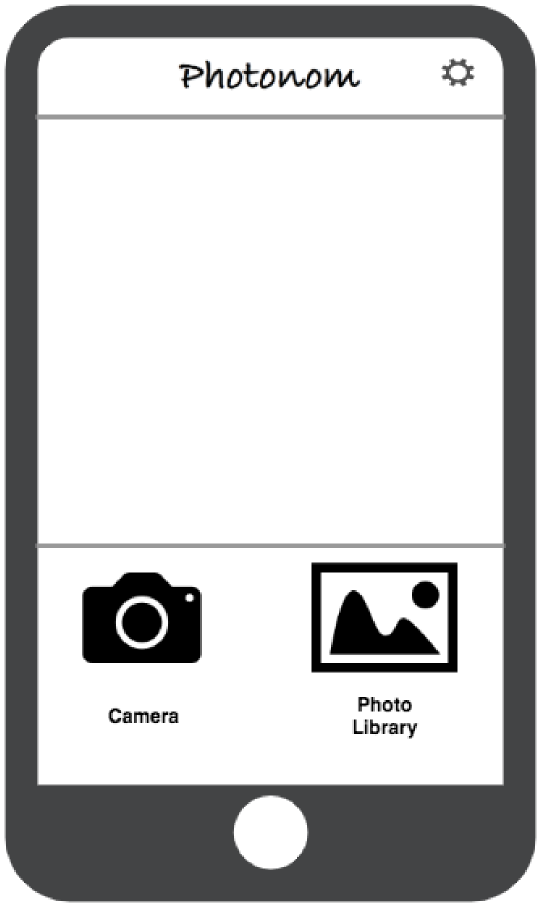


Figure 17: Mockup for Choose Photo Menu

Choose Photo Menu appears when the user presses the *Tap to open a new photo* button in the Main Menu. It includes Camera and Photo Library options buttons to take picture(s) and choose photo(s) respectively.

2.5.5.4 Camera

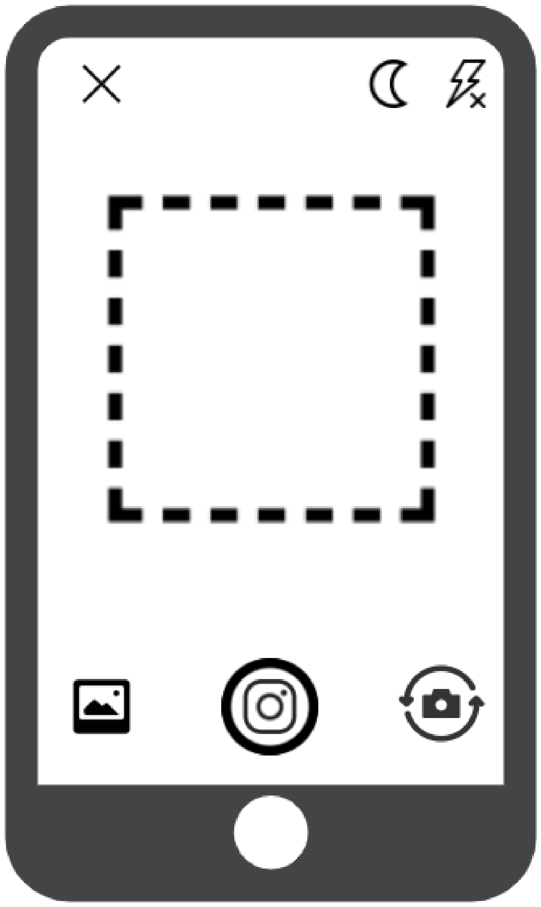


Figure 18: Mockup for Camera

Camera screen appears when the user presses the *Camera* button in the Choose Photo screen. If the user takes a single picture, then they will be directed to the Tool Selection screen. If they take multiple pictures then they will be directed to the Quality Evaluation Screen. The user will also be able to exit from Camera screen by pressing the X(close) button on the top left corner and will be directed to Choose Photo screen.

2.5.5.5 Photo Library

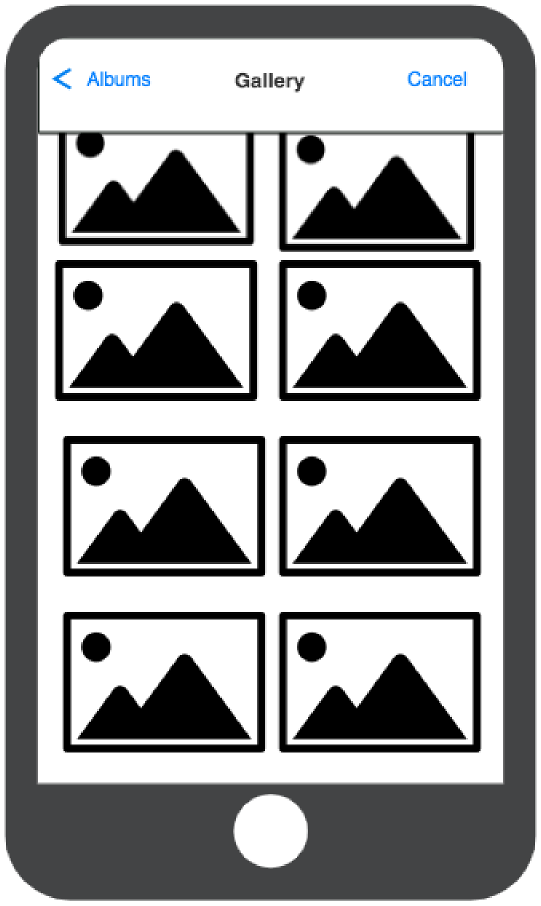


Figure 19: Mockup for Photo Library screen

Photo Library screen appears when the user presses the *Library* button in the Choose Photo Screen. The user will be able to choose their photo(s) from their local gallery. If the user chooses only one photo then they will be directed to the Tool Selection screen. However, if the user chooses more than one photo then they will be directed to the Quality screen. The user will also be able to exit from Photo Library screen by pressing the Cancel button on the top right corner and will be directed to Choose Photo screen.

2.5.5.6 Tool Selection

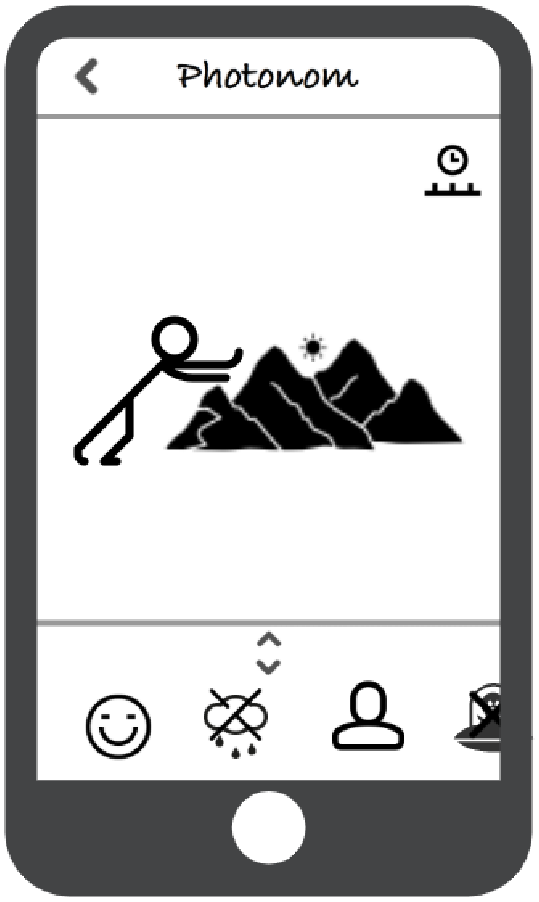


Figure 20: Mockup for Tool Selection screen

After the user selects their photo, they will be directed to this page where they can see their photo and the tools they can use for editing. If the user swipes up the tool screen below then the All Tools screen will appear and they can close the tool screen by swiping down. There is also a back button to go back to the Choose Photo screen to choose another photo. Moreover, this screen includes History Path button which will direct the user to the History Path Screen.

2.5.5.7 All Tools

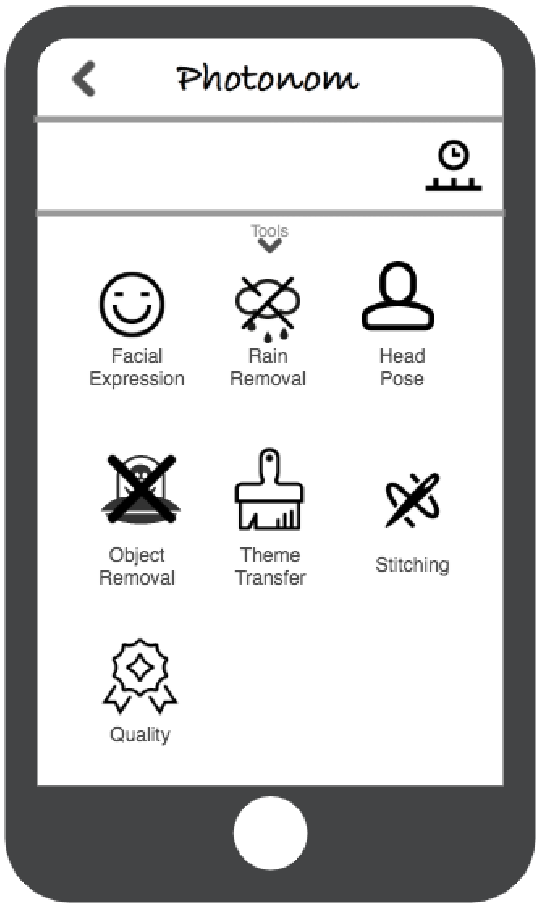


Figure 21: Mockup for All Tools screen

This screen includes all 6 tools for editing. The tools are Facial Expression, Rain Removal, Head Pose, Object Removal, Theme Transfer, Stitching. The user will choose their tool by pressing it's respective button.

2.5.5.8 Object Removal

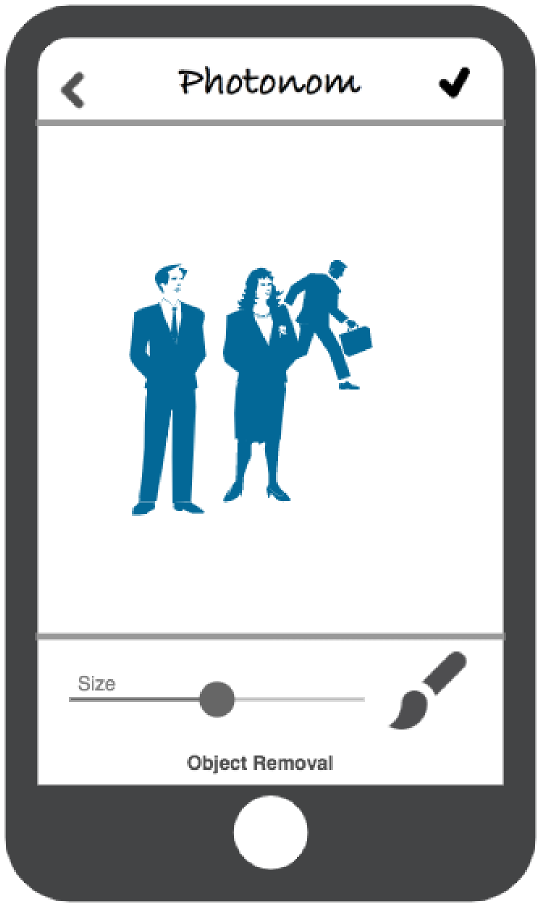


Figure 22: Mockup for Object Removal

This screen appears when the user chooses the Object Removal tool from the All Tool screen. In this screen, the user will be able adjust the brush size that they will use to select the object they want to remove.

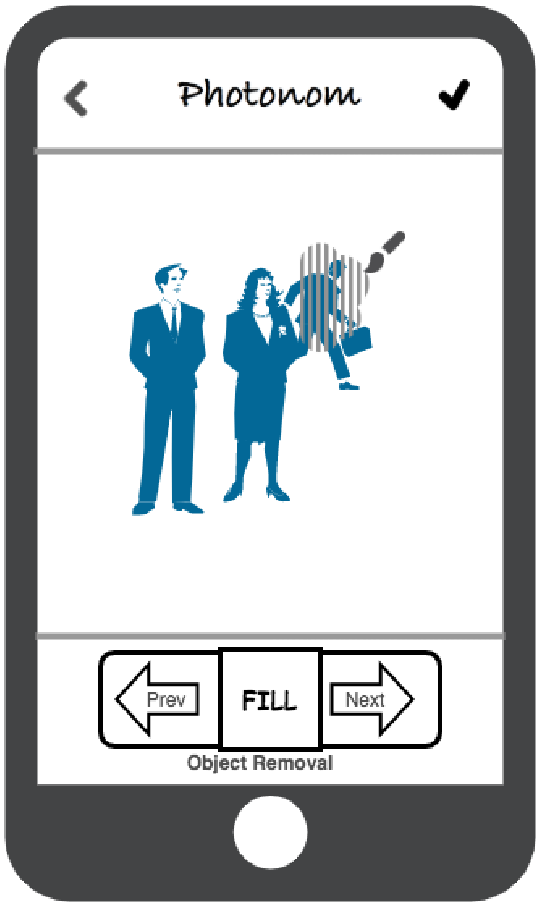


Figure 23: Mockup for Object Removal and Fill

This screen appears when the user drag the brush towards the picture. After the user paints the object to be removed, they will be able to fill the removed area by pressing the button at the bottom of the screen. The user can undo filling by pressing the Prev button and refill by pressing the Next button. The user will approve the changes by pressing the check button on the top right corner.

2.5.5.9 Rain Removal

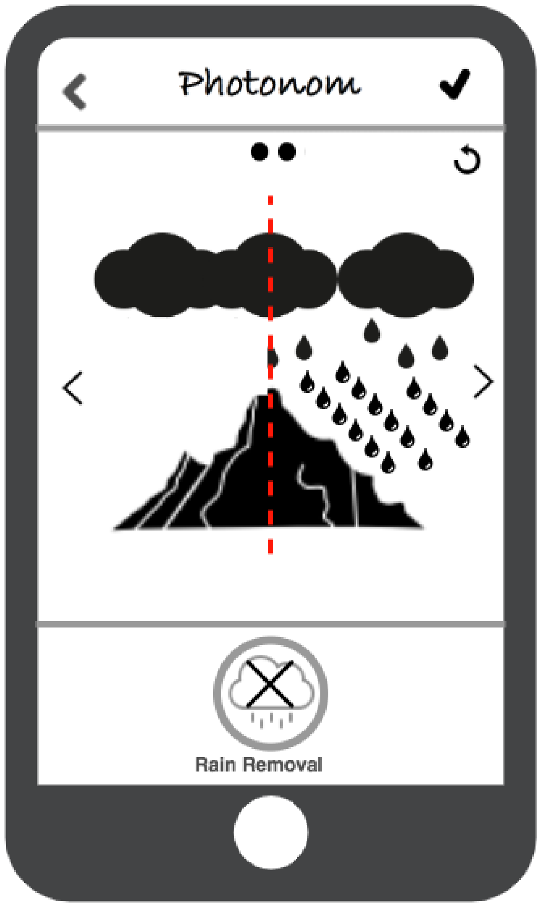


Figure 24: Mockup for Rain Removal

This screen appears when the user chooses the Rain Removal tool from the All Tools screen. In this screen, the user will be able to remove raindrops by pressing the button below. Moreover, the user will be able to swipe right and left to see both the original and the edited photo. The user will approve the changes by pressing the check button on the top right corner.

2.5.5.10 Theme Transfer

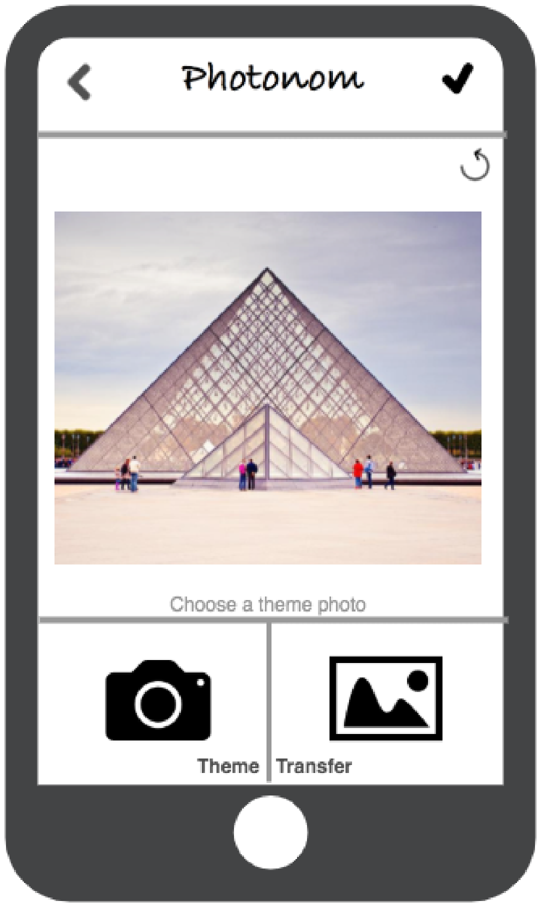


Figure 25: Mockup for Theme Transfer

This screen appears when the user chooses the Theme Transfer tool from the All Tool screen. In this screen, the user will be able to choose a theme photo either from their gallery ot by taking a photo.

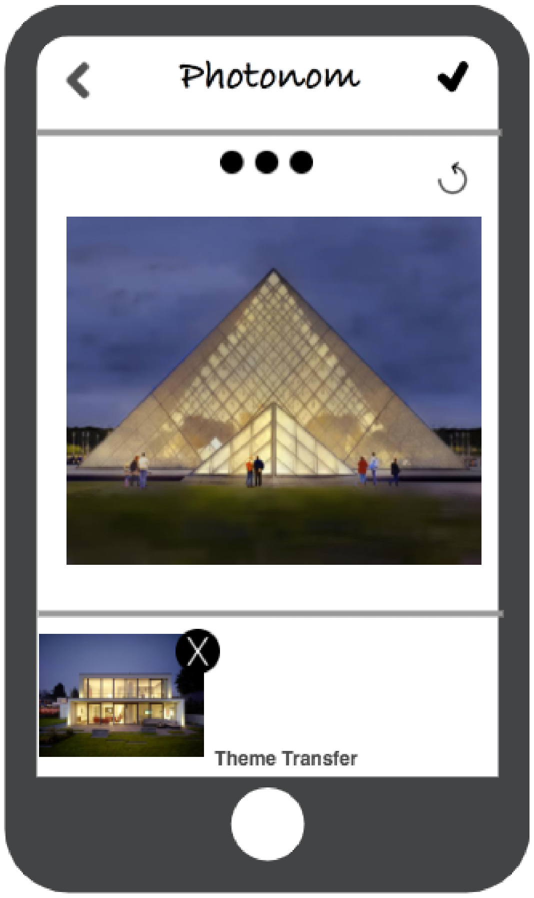


Figure 26: Mockup for After Theme Transfer

After the user chooses the theme photo, they will be directed to this page. The product of theme transfer will appear in the middle of the screen. The theme photo is on the bottom left of the screen. If the user cancels the theme photo by pressing the X button on its top right corner, then the theme transfer will be undone and the photo will be shown in its original format. Moreover, by swiping right and left the product and the original photo can be seen. The user will approve the changes by pressing the check button on the top right corner.

2.5.5.11 Image Stitching

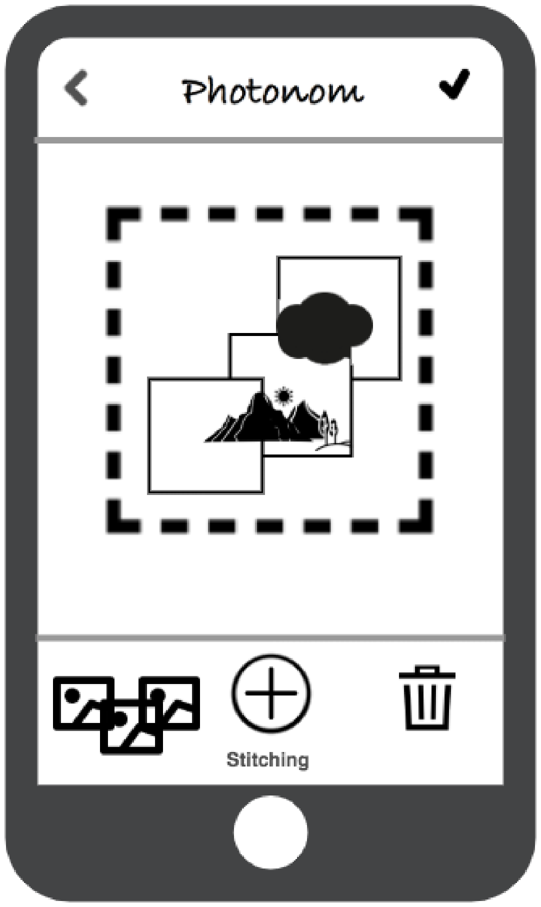


Figure 27: Mockup for Stitching

This screen appears when the user chooses the Stitching tool from the All Tool screen. The user will be able to add more photos to be stitched and will also be able to remove them by dragging the photos to the trash located at the bottom right corner of the screen. Photos that were selected for stitching will appear on the bottom left corner of the screen. The user will approve the changes by pressing the check button on the top right corner.

2.5.5.12 Face Expression Modification

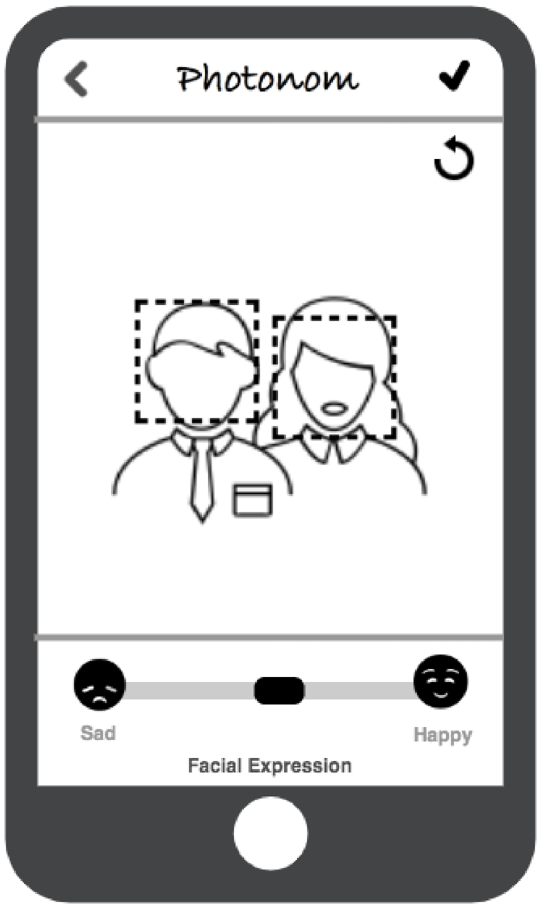


Figure 28: Mockup for Facial Expression screen

This screen appears when the user chooses the Facial Expression tool from the All Tools screen. In this screen, the user will be able to choose a face from the detected faces and will be able to edit more than one face. After the face is selected, the user will be able to change facial expression by moving the slider, located at the bottom of the screen, which goes from sad to happy expression. The user can undo their actions from the undo button. The user will approve the changes by pressing the check button on the top right corner.

2.5.5.13 Quality Evaluation

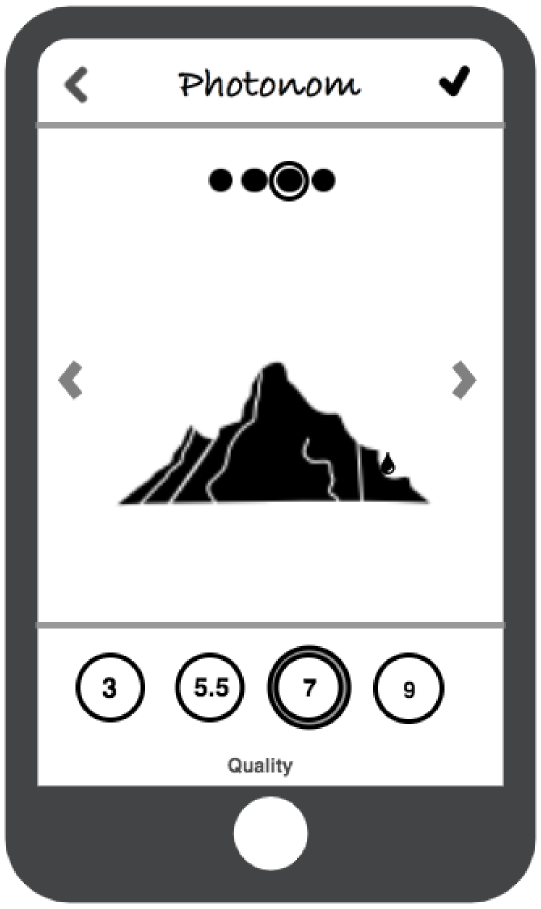


Figure 29: Mockup for Quality Evaluation

This screen appears when the user chooses multiple photos. In this screen, the user will be able to see scores of their uploaded photo at the bottom of the screen and they will be able to see their uploaded photos by swiping right and left. The user will approve the changes by pressing the check button on the top right corner.

2.5.5.14 History Path

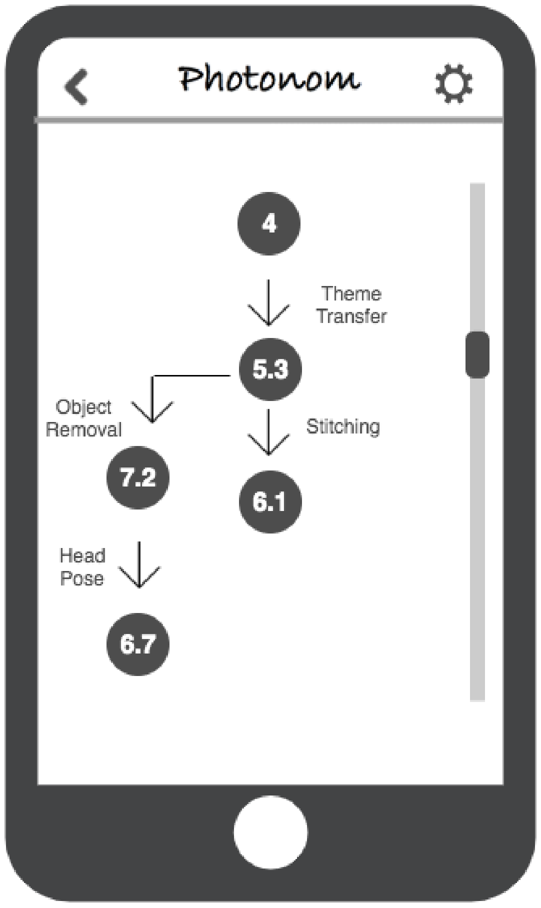


Figure 30: Mockup for HistoryPath

The user can reach this screen from the Tool Selection and All Tools screen by pressing the History button. In this screen, each node represents the score of the photo in its each state. For example, in this screen, the top node represents the original photo and the one under that node represents photo after theme transfer editing. The user will be able to select and continue from the desired state.

2.5.5.15 Head Pose Modifier

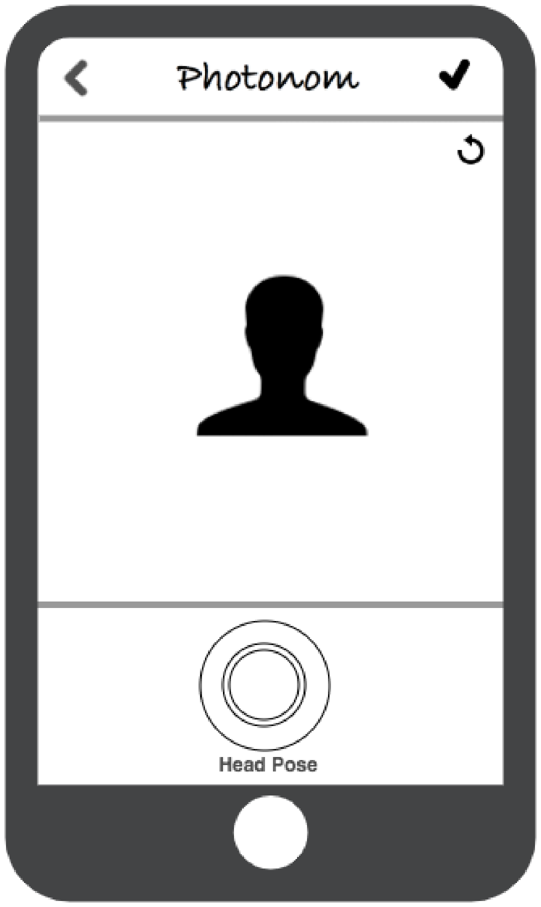


Figure 31: Mockup for Head Pose

This screen appears when the user chooses the Head Pose tool from the All Tools screen. The user will be able to modify the selected head pose from the sphere at the bottom of the screen. The user will also be able to undo their actions by pressing the undo arrow or go back to the All Tools screen by pressing the back button on the top left corner. The user will approve the changes by pressing the check button on the top right corner.

3 Other Analysis Elements

3.1 Consideration of Various Factors

While deciding the functionalities of the end product, various factors are considered related to the demands and needs of the users. In addition to affecting which functionality is going to be added to the application, these factors also affected the design of the project. The said factors can be seen in Table 1. Since the implementation of the project is an iterative process, the factors not considered before may affect the project.

	Effect Level	Effect
Global factors	5/5	Since one of the prerequisites of standing out in the global market is including global properties, language support functionality is added to the design.
Technological factors	5/5	As the effect of the social media applications on the society increases, the people value the photographs they post more. Our application can help improve these photographs by providing editing tools. Furthermore, by considering the effect of social media, the in-app social media sharing functionality is added.

Social factors	3/5	While adding the object removal functionality, the privacy of the people on the background is also considered. It might be disturbing for them to be shown in a photograph of a stranger. In order to avoid this situation, the object removal is added to the design of the project.
Public welfare	4/5	The increase of a public welfare enables people to spend more time for social activities such as managing social media accounts. Since more people are involved in this, the application should reach more people. That's why, during the design of the project usability is one of our priorities.

Cultural factors	4/5	In the cases where the photo someone took in front of a historical building but some parts of it is missing, we wanted to ensure people would still be able to look back on those photos and remember that historical structure in its true form. As a result we added image stitching during analysis. Cultural factors were effective in this decision since we wanted to conserve the nature of the culture being reflected on the photo.

Table 1: Factors that affected analysis and design

3.2 Risks and Alternatives

The possible risks that might come up during the project will be specified in this section. Then each of these risks will be quantified by considering the likelihood of that risk occurring and the effect the occurrence will have on the project. Each risk will be given a score out of 5 for the likelihood of the risk happening and it's affect on the project. Then a solution or plan b for each of these risks will be explained.

The risks related to this project are the following:

1. The papers we plan on using or the dataset we will be utilising for implementation may not be accessible.

This possible risk is related to the accessibility of the resources, papers and dateset, we plan on using for our project. This scenario is likely to happen since some organisations or researchers prefer to keep their data to themselves as having data for research is an advantage amongst different organisations and researchers. In addition, they might be less likely to share their data with us because we are undergarudate students. However, there are also a lot of organisations that are comfortable with sharing their data. The likelihood of this risk will be given a score of 4 out of 5.

The occurrence of this risk will have a significant effect on the project because not being able to access the dataset that we will use in order to implement the features of our project will drastically effect the success of our project. So the effect of this risk quantifies as 5 out of 5.

There are multiple measures that can be implemented in order to decrease the possibility of this risk happening and to decrease it's effects on the project. First of all, the research of papers and dataset should be limited to open sources resources. Furthermore, we should find at least 2 possible resources so that we will have alternative resources. To decrease the effect of this risk happening, we should make sure that we contact the responsible individuals as soon as possible so that we will have the time to find other datasets.

2. Overly optimistic schedule.

Before starting the work on the project, there is a possibility of us underestimating the amount of work some tasks might need. This faulty statement might cause our schedule to be overly optimistic and eventually might cause some trouble with delivery of the products. However given the fact that we have some experience with the required deliverables of this project such as reports, and for those deliverable we will probably make accurate assumptions. Same can not be said, however, for the tasks we are unfamiliar with. So the likelihood of this risk occurring could be countrified with the score 3 out of 5.

Overly optimistic schedule might effect the project by causing delays in the delivery of the products. Since this is a project with strict deadlines, this will have drastic effects on the project. Thus, the effect of this risk occurring will be given a score of 5 out of 5.

In order to ensure this risk does not occur or to ensure it has minimal effect on the project, we should prepare or schedules with time buffers to ensure possible underestimates will take away from these buffers instead of the delivery time of the product. Furthermore, group meetings for the assessment of the schedule should be organised at least monthly but preferably every other week to ensure our schedule is up to date and we address possible underestimates the moment we become aware of them.

3. Group members might use different platforms to implement the parts they are responsible of which might cause some issues during the merging of the project.

   The likelihood of this risk occurring is not very high, since from the beginning we decided on using Overleaf to merge our work on the reports and GitHub to merge our code. Therefore the likelihood of this risk will be given a score of 1 out of 5.

   Not being able to merge our work properly would have some significant effect on the project. However given the fact that we have experience of group projects and therefore merging the work of multiple people, the effect of this risk on the project will be given a scour of 2 out of 5.

   The best measure to avoid this risk is to agree on the tools that will be used during implementation and to ensure everyone is using these tools.

4. During the implementation the group members might have more immediate assignments they

will have to work on.

Since all of the group members are university students, it is very likely that during the implementation we will have other projects or assignment that we have to work on. Therefore the likelihood of this risk will be given a score of 5 out of 5.

Whilst the occurrence of this risk might cause momentarily delays on the completion of certain tasks, it will not have significant effect on the project. This is due to all of us being senior students who are, at this point, pretty used to having multiple responsibilities and deadlines at the same time. So the effect on the project of this risk will be scored 3 out of 5.

In order to ensure this risk does not hinder the process of this project, the scheduling should be updated regularly with attention to other responsibilities group members might have.

5. Conflicts inside the group might affect the speed of implementation.

   This risk is related to the dynamics within the project group. The likelihood of this risk happening is not high because all of the group members know each other outside of the project. Furthermore, as mentioned before we already worked with each other before and did not face such problem. Therefore the likelihood of this risk occurring will be given a score of 1 out of 5.

   If this risk was to occur, it might have some effect on the project since conflicts inside the group might cause some mistakes in communication and it is not comfortable work in a tense environment. However, all of the group members agreed beforehand on the expectations they have from this project so it is very likely that each member will be able to have a professional point of view towards personal conflicts. Therefore, the effect of this risk on the project will be given a score of 2 out of 5.

In order to ensure conflicts do not arise inside the group, opportunities for each member to speak their mind should be created. This would ensure possible conflicts are addressed before they turn into big problems.

6. We might be in need of additional computation power during implementation.

   There is a possibility that we will need additional computation power during implementation. This is due to the fact that the operations performed by our system are computationally heavy. Therefore, the likelihood of this risk occurring will be given a score of 3 out of 5.

   The occurrence of this risk will hinder the implementation since insufficient computation power will slow down the application which is overall undesirable. Therefore, the effect of this risk on the project will be given a score of 4 out of 5.

   The best measure to implement the effect of this risk on the project is to ensure we have a plan B in case we do need additional computation power. The possible plan B for this scenario is using the free credits of Google Cloud Platform or Amazon Web Services [7] [6].

Summary of the points made in this section can be seen in Table 2.

Risk	Likelihood	Effect on the Project	Plan B
Resources we plan on using might not be accessible	4	5	Contact the owners of the dataset and detect alternatives.
Overly optimistic schedule	3	5	Add time buffers to schedule and update the schedule immediately when necessary

Use of different tools by group members	1	2	Agree on tools to use before the implementation starts
Other responsibilities of the group members may arise	5	3	Update the schedule immediately when necessary
Conflicts inside the group.	1	2	Ensure healthy communication amongst group members
Additional computational power	3	4	Use free credits of Google Cloud Platform or Amazon Web Services.

Table 2: Risks and Alternatives (Scores for Likelihood and Effect are given out of 5)

3.3 Project Plan

In this section our project plan for the next months will be discussed.

To give a better hierarchical view of our work packages, the list of work packages and the work package leaders are given below. The work packages written in bold are general work packages.

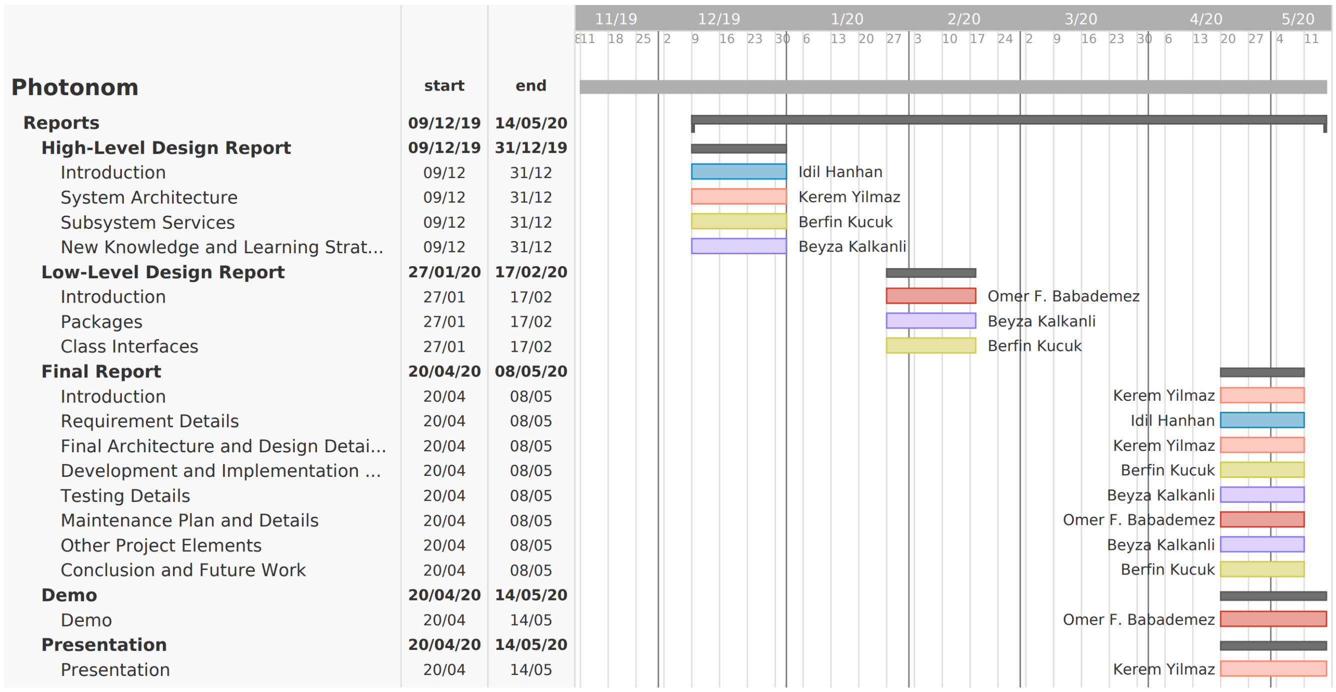
WP#	Work Package Title	Leader	Members Involved
WP1	**Reports**	Idil Hanhan	All Team
WP2	High Level Design Report	Idil Hanhan	Idil, Kerem, Berfin, Beyza
WP3	Low Level Design Report	Kerem Yilmaz	Omer, Beyza, Berfin
WP4	Final Report	Omer F. Babademez	All Team
WP5	Demo	Berfin Kucuk	Omer
WP6	Presentation	Beyza Kalkanli	Kerem
WP7	**Development**	Kerem Yilmaz	All Team
WP8	Research	Beyza Kalkanli	All Team
WP9	Server	Omer F. Babademez	Idil, Kerem, Omer
WP10	Mobile Application	Berfin Kucuk	All Team

Table 3: List of work packages

As it can be seen in Table 3 we have two main work packages which contain sub-work packages. We separated our work packages into two main groups: Reports and Development. These can be seen as the general work packages that help us categorize the main work packages. Now each work package will be explained.

3.3.1 Reports

This general work package contains five different work packages each representing a deliverable that is listed in CS491/492 course web page [14]. The deadlines given are matched with the deadlines given on the web page. The objectives and deliverables of the general work packages are defined by the work packages they contain. For the sake of simplicity, the same objectives and deliverables will not be repeated in the general work packages.



Photonom	start	end
Reports	**09/12/19**	**14/05/20**
High-Level Design Report	**09/12/19**	**31/12/19**
Introduction	09/12	31/12
System Architecture	09/12	31/12
Subsystem Services	09/12	31/12
New Knowledge and Learning Strat...	09/12	31/12
Low-Level Design Report	**27/01/20**	**17/02/20**
Introduction	27/01	17/02
Packages	27/01	17/02
Class Interfaces	27/01	17/02
Final Report	**20/04/20**	**08/05/20**
Introduction	20/04	08/05
Requirement Details	20/04	08/05
Final Architecture and Design Detai...	20/04	08/05
Development and Implementation ...	20/04	08/05
Testing Details	20/04	08/05
Maintenance Plan and Details	20/04	08/05
Other Project Elements	20/04	08/05
Conclusion and Future Work	20/04	08/05
Demo	**20/04/20**	**14/05/20**
Demo	20/04	14/05
Presentation	**20/04/20**	**14/05/20**
Presentation	20/04	14/05

Figure 32: Reports Work Package Gantt

3.3.2 High Level Design Report

The Gantt chart for this work package is given in figure 32. The aim of this work package is to fulfil the requirements for the High Level Design Report as given in the course web page [14]. It is also aimed to have a precise high level design for our project so that during the development stage we will not have design discussions that cause considerable amount of change.

The tasks for this work package are created upon reflection on the report overview given in our course web page [14].

The deliverable of this work package is a complete high level design report.

3.3.3 Low Level Design Report

The Gantt chart for this work package is given in figure 32. The aim of this work package is to fulfil the requirements for the Low Level Design Report as given in the course web page [14]. It is also aimed to have a precise low level design for our project so that during the development stage we will not have design discussions that cause considerable amount of change.

The tasks for this work package are created upon reflection on the report overview given in our course web page [14].

The deliverable of this work package is a complete low level design report.

3.3.4 Final Report

The Gantt chart for this work package is given in figure 32. The aim of this work package is to fulfil the requirements for the Final Report as given in the course web page [14].

The tasks for this work package are created upon reflection on the report overview given in our course web page [14].

The deliverable of this work package is a complete final report.

3.3.5 Demo

The Gantt chart for this work package is given in figure 32. The demo work package is a straightforward but important work package since it allows the team to demonstrate the work done in the past two semesters in a comparably short amount of time. To inflict its importance, it is wrapped as a separate work package.

The deliverable of this work package is to make a complete demonstration of our end product.

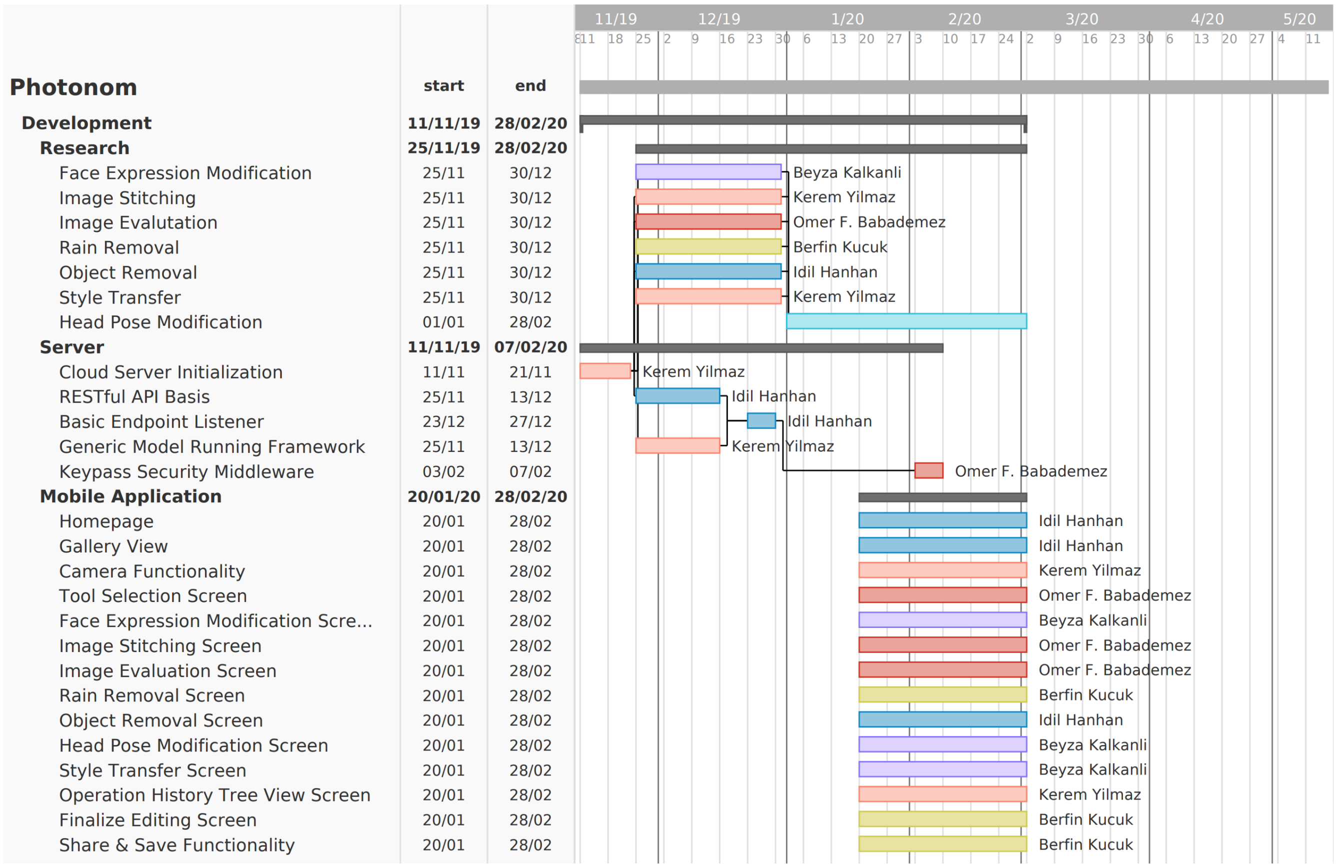
3.3.6 Presentation

The Gantt chart for this work package is given in figure 32. The demo work package is a straightforward but important work package since it allows the team to present the work done in the past two semesters in a comparably short amount of time. To inflict its importance, it is wrapped as a separate work package.

The deliverable of this work package is to make a complete presentation of our end product.

3.3.7 Development

This general work package contains three different work packages each representing a main component of our senior design project. The objectives and deliverables of the general work packages are defined by the work packages they contain. For the sake of simplicity, the same objectives and deliverables will not be repeated in the general work packages.



Photonom	start	end
Development	**11/11/19**	**28/02/20**
Research	**25/11/19**	**28/02/20**
Face Expression Modification	25/11	30/12
Image Stitching	25/11	30/12
Image Evalutation	25/11	30/12
Rain Removal	25/11	30/12
Object Removal	25/11	30/12
Style Transfer	25/11	30/12
Head Pose Modification	01/01	28/02
Server	**11/11/19**	**07/02/20**
Cloud Server Initialization	11/11	21/11
RESTful API Basis	25/11	13/12
Basic Endpoint Listener	23/12	27/12
Generic Model Running Framework	25/11	13/12
Keypass Security Middleware	03/02	07/02
Mobile Application	**20/01/20**	**28/02/20**
Homepage	20/01	28/02
Gallery View	20/01	28/02
Camera Functionality	20/01	28/02
Tool Selection Screen	20/01	28/02
Face Expression Modification Scre...	20/01	28/02
Image Stitching Screen	20/01	28/02
Image Evaluation Screen	20/01	28/02
Rain Removal Screen	20/01	28/02
Object Removal Screen	20/01	28/02
Head Pose Modification Screen	20/01	28/02
Style Transfer Screen	20/01	28/02
Operation History Tree View Screen	20/01	28/02
Finalize Editing Screen	20/01	28/02
Share & Save Functionality	20/01	28/02

Figure 33: Development Work Package Gantt

3.3.8 Research

The Gantt chart for this work package is given in figure 33. The aim of this package is to test and verify that there are open source, usable, and working projects that satisfy the requirements regarding the features specified in the specifications report. If the testing and verification processes are completed and the results are satisfying, it is required that the assignee should deliver a working model for the server.

The tasks of this work package are as follows.

- **Face Expression Modification:** Main aim is to modify the smile of people. The assignee should deliver a working product with satisfactory outputs within the restrictions of the cloud server.

- **Image Stitching:** Main aim is to be able to stitch multiple images regardless of the photos' color schemes. The assignee should deliver a working product with satisfactory outputs within the restrictions of the cloud server.

- **Image Evaluation:** Main aim is to be able to grade images by taking into account many traits. The assignee should deliver a working product with satisfactory outputs within the restrictions of the cloud server.

- **Rain Removal:** Main aim is to remove the effects of rain in the images taken. The assignee should deliver a working product with satisfactory outputs within the restrictions of the cloud server.

- **Object Removal:** Main aim is to remove the object the user painted and fill the missing parts while at the same time keeping the image natural. The assignee should deliver a working product with satisfactory outputs within the restrictions of the cloud server.

- **Style Transfer:** Main aim is to transfer the theme, color scheme of an image to the other one. The assignee should deliver a working product with satisfactory outputs within the restrictions of the cloud server.
- **Head Pose Modification:** This task will be started if time permits since its requirements and the complexity are problematic. If time permits and we start this task, the assignee should deliver a working product with satisfactory outputs within the restrictions of the cloud server.

3.3.9 Server

The Gantt chart for this work package is given in figure 33. The aim of this package is to develop a complete server that:

- can listen to specific ports to receive commands,
- can apply the requested operations using a generic framework,
- can handle requests from multiple users at the same time without confusion,
- has a secure connection between the user's application since the communication packages contain delicate information regarding the user.

The tasks of this work package are given as follows.

- **Cloud Server Initialization:** Main aim is to have a cloud server that lets all the team to work on and sets an endpoint for the mobile application to communicate with. The assignee is should deliver a working server with the required packages and minimal satisfying computing power while at the same time considering the costs.

- **RESTful API Basis:** Main aim is to have a minimalist server that can listen and send requests.
- **Basic Endpoint Listener:** Main aim is to implement an endpoint listener and trigger the necessary events regarding that endpoint. If necessary it should also return a meaningful response to the requester.
- **Generic Model Running Framework:** Main aim is to have a generic framework that can run the models of the research components easily. It should be customizable and easily extendable for additional features.
- **Keypass Security Middleware:** Main aim is to provide a secure communication medium for the mobile app and server.

3.3.10 Mobile Application

The Gantt chart for this work package is given in figure 33. The aim of this package is to develop a mobile application that:

- can be run on both iOS and Android,
- has a fluent user interface,
- has a user experience that users can get used to quickly.

The tasks of this work package are given as follows.

- **Homepage:** Main aim is to have a simplistic user interface that matches with the related mockups. The assignee should deliver a functional and complete user interface.

- **Gallery View:** Main aim is to have the users access their gallery in such a way that complies with our features and requirements. The assignee should deliver a functional and complete user interface.

- **Camera Functionality:** Main aim is to have the users access their camera in such a way that complies with our features and requirements. The assignee should deliver a functional and complete user interface.

- **Tool Selection Screen:** Main aim is to have a simplistic user interface that matches with the related mockups. The assignee should deliver a functional and complete user interface.

- **Face Expression Modification Screen:** Main aim is to have a simplistic user interface that matches with the related mockups. The assignee should deliver a functional and complete user interface.

- **Image Stitching Screen:** Main aim is to have a simplistic user interface that matches with the related mockups. The assignee should deliver a functional and complete user interface.

- **Image Evaluation Screen:** Main aim is to have a simplistic user interface that matches with the related mockups. The assignee should deliver a functional and complete user interface.

- **Rain Removal Screen:** Main aim is to have a simplistic user interface that matches with the related mockups. The assignee should deliver a functional and complete user interface.

- **Object Removal Screen:** Main aim is to have a simplistic user interface that matches with the related mockups. The assignee should deliver a functional and complete user interface.

- **Head Pose Modification Screen:** Main aim is to have a simplistic user interface that matches with the related mockups. The assignee should deliver a functional and complete user interface.

- **Style Transfer Screen:** Main aim is to have a simplistic user interface that matches with the related mockups. The assignee should deliver a functional and complete user interface.
- **Operation History Tree View Screen:** Main aim is to have a simplistic user interface that matches with the related mockups. The assignee should deliver a functional and complete user interface.
- **Finalize Editing Screen:** Main aim is to have a simplistic user interface that matches with the related mockups. The assignee should deliver a functional and complete user interface.
- **Share and Save Functionality:** Main aim is to give the ability of sharing the outcome or saving it to the user. The assignee should deliver a functional and complete user interface.

3.4 Ensuring Proper Team Work

In this section our plans to ensure proper team work will be specified.

- The project is separated into work packages.
- Each work package, or sub-package is organised in sprints.
- For each sprint we will have a meeting during which tasks will be specified and the time required for each of the tasks will be determined. We try to ensure that total sum of hours for each member during each sprint is more or less equal.
- Each work package has a leader who will make sure each group member knows their responsibility.

- Each member is the leader of at least one work package.
- Each group member has at least one research area they are responsible of.

3.5 Ethics and Professional Responsibilities

In this section the ethical and professional responsibilities of this project will be specified.

- The user should consent to Photonom accessing and modifying their photo. However, we cannot assure that the user of Photonom is the actual owner of the photo or whether the consent of the owner is taken before photo is uploaded. Photonom does not take further responsibility of taking consent of the owner and it is assumed that the user has taken the consent of the owner.
- Pictures will not be stored in our servers after the user has completed editing. However, editing will be done in our servers, in the meantime, security for the user's data will be provided. User's data will not be shared with other applications or people without user's consent.
- The user will be able to share and save the processed picture so it is the user's responsibility to abide by the copyright of pictures.
- Licenses for third-party APIs and libraries will be checked before usage.

3.6 New Knowledge and Learning Strategies

During our senior design project, our team is required to learn many things regarding both the research and the development work packages. In order to learn the necessary subjects we've come up with two different strategies. The learning items are as follows.

For the research related subjects, we are required to read a few general articles or research papers regarding the subject. After we do so, we need to find an open source project and examine its code to have a somewhat engineering perspective to it. After these steps are completed, we should be on-boarded enough to focus the main task.

For the development related subjects, we've come to notice that the best way to learn a new framework is to start using it. The strategy is as follows. We should first use the original quick start tutorial if it exists, otherwise we should follow a comprehensive tutorial to set up the framework and learn about its philosophies. Then, the team members should focus on the main focus of the assigned task. Along the way, if necessary, purpose focused tutorials should help us learn the tools we will be using while at the same time making some progress regarding our senior design project, Photonom.

4 Glossary

Photonom: A word derived from the merging of two words: photoshop and autonom.

References

[1] "Instagram by the numbers (2019): Stats, demographics & fun facts," 2019. [Online]. Available: https://www.omnicoreagency.com/instagram-statistics/

[2] J. Brucculieri, "4 instagrammers show us how many photos they took before nailing 'the shot'," 2018. [Online]. Available: https://www.huffpost.com/entry/instagramphotocamerarolls_n_5ac4ed48e4b063ce2e58131f

[3] N. Murray, L. Marchesotti, and F. Perronnin, "Ava: A large-scale database for aesthetic visual analysis."

[4] 2019. [Online]. Available: https://gdpr.eu/

[5] 2019. [Online]. Available: https://www.docker.com/

[6] 2019. [Online]. Available: https://aws.amazon.com/

[7] 2019. [Online]. Available: https://cloud.google.com/

[8] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[9] 2019. [Online]. Available: https://github.com/pathak22/context-encoder#6-paris-street-view-dataset

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[11] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image

database for scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[12] [Online]. Available: http://cbcsl.ece.ohio-state.edu/cvpr16.pdf

[13] [Online]. Available: http://www.ponomarenko.info/tid2013.htm

[14] 2019. [Online]. Available: http://www.cs.bilkent.edu.tr/CS491-2/CS491.html