Module Interface Specification for Hairesthetics

Team 18
Charlotte Cheng
Marlon Liu
Senni Tan
Qiushi Xu
Hongwei Niu
Bill Song

April 5, 2023

1 Revision History

Date	Version	Notes
Jan 15	0.1	Initial Draft
Jan 16	0.2	Minor updates
Jan 17	1	Rev0 MIS
Apr 1	2	Final Doc Update
Apr 4	2.1	Minor update to modules

2 Symbols, Abbreviations and Acronyms

symbol	description
ML	Machine Learning
UI	User Interface
AI	Artificial Intelligence
AR	Augumented Reality
App	Application
API	Application programming interface
REST	Representational state transfer
RGB	Red, Green, Blue
macOS	Operating system developed by Apple Inc
MG	Module Guide
MIS	Module Interface Specification

See SRS Documentation at /docs/SRS/SRS.pdf

Contents

List of Tables

List of Figures

3 Introduction

The following document details the Module Interface Specifications for the Hairesthetics application. Hairesthetics is an application that simulates 3D hairstyles.

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at https://github.com/marlon4dashen/Hairesthetics.

4 Notation

The structure of the MIS for modules comes from HoffmanAndStrooper1995, with the addition that template modules have been adapted from GhezziEtAl2003. The mathematical notation comes from Chapter 3 of HoffmanAndStrooper1995. For instance, the symbol:= is used for a multiple assignment statement and conditional rules follow the form $(c_1 \Rightarrow r_1 | c_2 \Rightarrow r_2 | ... | c_n \Rightarrow r_n)$.

The following table summarizes the primitive data types used by the modules.

Data Type	**Notation**	**Description**
character	char	a single symbol or digit
integer	$\mathbb{Z}$	a number without a fractional component in (-∞, ∞)
natural number	$\mathbb{N}$	a number without a fractional component in [1, ∞)
real	$\mathbb{R}$	any number in (-∞, ∞)

The specification of our modules uses some derived data types: sequences, strings, and tuples. Sequences are lists filled with elements of the same data type. Strings are sequences of characters. Tuples contain a list of values, potentially of different types. In addition, our modules use functions, which are defined by the data types of their inputs and outputs. Local functions are described by giving their type signature followed by their specification.

5 Module Decomposition

This section provides an overview of the module design. Modules are summarized in a hierarchy decomposed by secrets in Table 1. The modules listed below, which are leaves in the hierarchy tree, are the modules that will actually be implemented.

M1: ~~Controller Module~~ App Module

M2: Server Module

M3: Hair Color Module

M4: Worker Module

M5: Image Worker Module

M6: Salon Recommendation Module

M7: Hair Artist Module

M8: Model Utils Module

M9: Image Utils Module

M10: Hair Color View Module

M11: Hair Style View Module

M12: Salon Recommendation View Module

M13: Home View Module

M14: ~~Error View Module~~ Footer Module

M15: ~~Camera Module~~ NavBar Module

M16: ~~Launch View Module~~ AR Canvas Module

M17: HairModel Module

M18: ThreeFiber Helper Module

Level 1	**Level 2**
Hardware-Hiding Module	~~M13~~ M1
Behaviour-Hiding Module	~~M1~~ M2 M3 M4 M5 M6 M7 M10 M11 M12 M13 M14 M15 M16 M17
Software Decision Module	~~M6~~ ~~M7~~ M8 M9 M18

Table 1: Module Hierarchy

5.1 UML Diagram

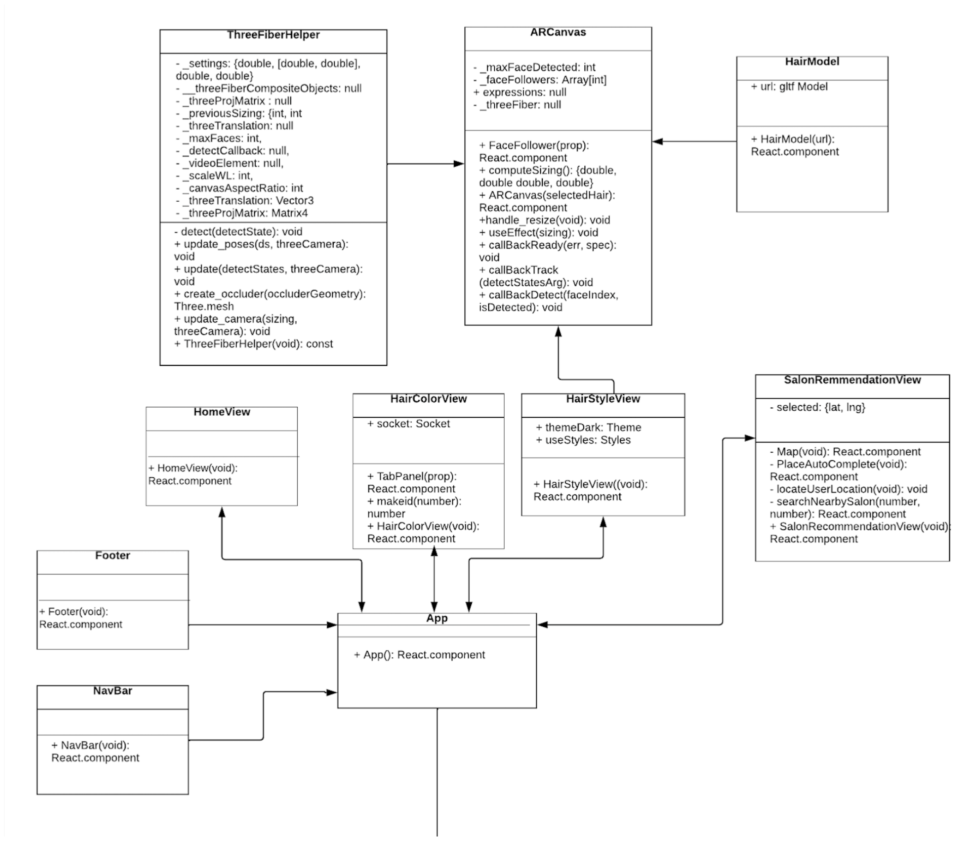
ThreeFiberHelper
- _settings: {double, [double, double], double, double}
- __threeFiberCompositeObjects: null
- _threeProjMatrix : null
- _previousSizing: {int, int
- _threeTranslation: null
- _maxFaces: int,
- _detectCallback: null,
- _videoElement: null,
- _scaleWL: int,
- _canvasAspectRatio: int
- _threeTranslation: Vector3
- _threeProjMatrix: Matrix4
- detect(detectState): void
+ update_poses(ds, threeCamera): void
+ update(detectStates, threeCamera): void
+ create_occluder(occluderGeometry): Three.mesh
+ update_camera(sizing, threeCamera): void
+ ThreeFiberHelper(void): const
ARCanvas
- _maxFaceDetected: int
- _faceFollowers: Array[int]
+ expressions: null
- _threeFiber: null
+ FaceFollower(prop): React.component
+ computeSizing(): {double, double double, double}
+ ARCanvas(selectedHair): React.component
+handle_resize(void): void
+ useEffect(sizing): void
+ callBackReady(err, spec): void
+ callBackTrack (detectStatesArg): void
+ callBackDetect(faceIndex, isDetected): void
HairModel
+ url: gltf Model
+ HairModel(url): React.component
HomeView
+ HomeView(void): React.component
HairColorView
+ socket: Socket
+ TabPanel(prop): React.component
+ makeid(number): number
+ HairColorView(void): React.component
HairStyleView
+ themeDark: Theme
+ useStyles: Styles
+ HairStyleView((void): React.component
SalonRemmendationView
- selected: {lat, lng}
- Map(void): React.component
- PlaceAutoComplete(void): React.component
- locateUserLocation(void): void
- searchNearbySalon(number, number): React.component
+ SalonRecommendationView(void): React.component
Footer
+ Footer(void): React.component
App
+ App(): React.component
NavBar
+ NavBar(void): React.component

Figure 1: UML Diagram Part 1

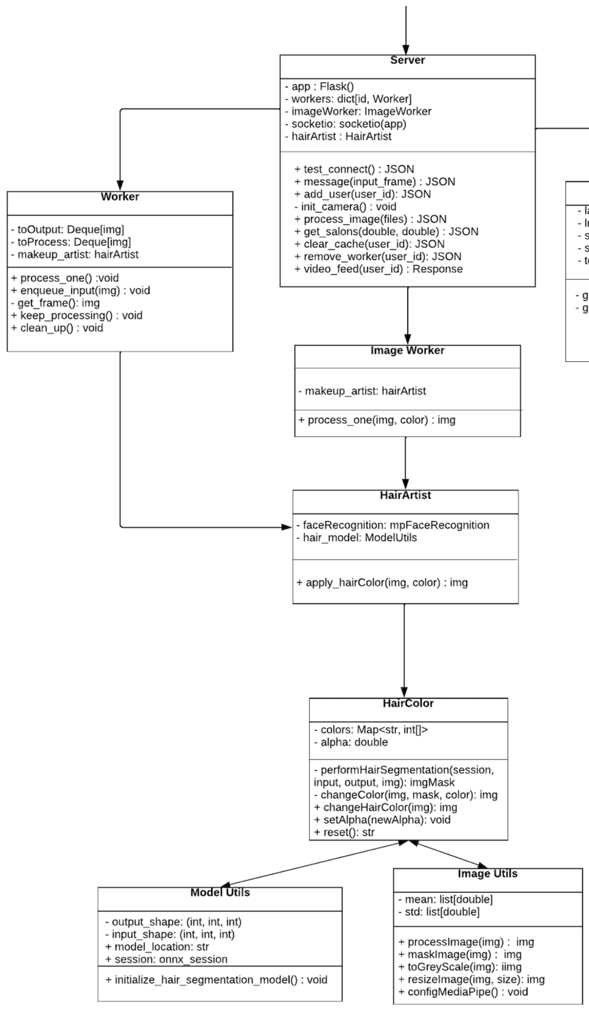
Server
- app : Flask()
- workers: dict[id, Worker]
- imageWorker: ImageWorker
- socketio: socketio(app)
- hairArtist : HairArtist
+ test_connect() : JSON
+ message(input_frame) : JSON
+ add_user(user_id): JSON
- init_camera() : void
+ process_image(files) : JSON
+ get_salons(double, double) : JSON
+ clear_cache(user_id): JSON
+ remove_worker(user_id): JSON
+ video_feed(user_id) : Response
Worker
- toOutput: Deque[img]
- toProcess: Deque[img]
- makeup_artist: hairArtist
+ process_one() :void
+ enqueue_input(img) : void
- get_frame(): img
+ keep_processing() : void
+ clean_up() : void
SalonRemmendation
- lat: double
- lng: double
- salons: [dict]
- size: int
- total_average_rating: double
- getSize() : int
- getNearbySalons() : [salon]
Image Worker
- makeup_artist: hairArtist
+ process_one(img, color) : img
HairArtist
- faceRecognition: mpFaceRecognition
- hair_model: ModelUtils
+ apply_hairColor(img, color) : img
HairColor
- colors: Map<str, int[]>
- alpha: double
- performHairSegmentation(session, input, output, img): imgMask
- changeColor(img, mask, color): img
+ changeHairColor(img): img
+ setAlpha(newAlpha): void
+ reset(): str
Model Utils
- output_shape: (int, int, int)
- input_shape: (int, int, int)
+ model_location: str
+ session: onnx_session
+ initialize_hair_segmentation_model() : void
Image Utils
- mean: list[double]
- std: list[double]
+ processImage(img) : img
+ maskImage(img) : img
+ toGreyScale(img): iimg
+ resizeImage(img, size): img
+ configMediaPipe() : void

Figure 2: UML Diagram Part 2

6 ~~MIS of Controller Module~~App Module

6.1 Module

M1 - App Module
Abstract Data Type Module

6.2 Uses

Server Module (M2)
HairColorView (M10)
HairStyleView (M11)
SalonRecommendationView (M12)
HomeView (M13)
Footer Module (M14)
NavBar Module (M15)

6.3 Syntax

6.3.1 Exported Constants

6.3.2 Exported Access Programs

Name	In	Out	Exceptions
App	-	-	-

6.4 Semantics

6.4.1 State Variables

~~hairColorScreen := HairColorView~~
~~hairStyleScreen := HairStyleView~~
~~salonRecomScreen := SalonRecommendationView~~
~~camera := Camera~~
~~homeScreen := HomeView~~
~~launchScreen := launchView~~
~~errorScreen := errorView~~
~~facialRecognitionModel := FacialRecognition~~
~~hairColorModel := HairColor~~
~~hairStyleModel := HairStyle~~
~~salonRecommendationModel := SalonRecommendation~~
~~currentView := homeScreen~~

6.4.2 Environment Variables

6.4.3 Assumptions

6.4.4 Access Routine Semantics

App():

- transition:
  app := react.component()
- output:
- exception:

6.4.5 Local Functions

7 MIS of Server Module

7.1 Module

M2 - Server
Abstract Object Module

7.2 Uses

Worker (M4)
Hair Artist (M7)
Salon Recommendation (M6)
Flask
SocketIO

7.3 Syntax

7.3.1 Exported Constants

7.3.2 Exported Access Programs

Name	In	Out	Exceptions
test_connect		JSON	
message	JSON		
add_user	int	JSON	
init_camera			
process_images	File	JSON	
get_salons	double, double	JSON	
clear_cache	int	JSON	KeyError
remove_worker	int	JSON	KeyError
video_feed	int	JSON	KeyError

7.4 Semantics

7.4.1 State Variables

app := Flask()
workers := dict()
imageWorker := null
socketio := SocketIO(app)
hairArtist := null
salonRecommendation := null

7.4.2 Environment Variables

7.4.3 Assumptions

7.4.4 Access Routine Semantics

test_connect():

- transition:
- output:
  return output - JSON with 200 status connection success
- exception:

message(img, color, user_id):

- transition: worker[user_id].enqueue_input = (img, color)
- output: JSON with 200 status success
- exception:

add_user(user_id):

- transition: hairArtist = HairArtist() if hairArtist == none
  worker[user_id] = Worker(hairArtist)
- output: JSON with 200 status success
- exception:

init_camera(user_id):

- transition: imageWorker = imageWorker(hairArtist)
- output: JSON with 200 status success
- exception:

process_images(File, color, user_id):

- transition: res = imageWorker.process_one(files.toImage(), color)
- output: JSON with processed frame status 200 success
- exception:

get_salons(lat, lng):

- input: lat - latitude in double, lng - longitude in double

- transition:
  salonRecommendation = salonRecommendation(lat, lng)
  salons = salonRecommendation.get_nearby_salons()
- output: salons - JSON with nearby salon informations and 200 status code
- exception:

clear_cache(user_id):

- transition:
  if user_id exists:
  workers[user_id].clean_up()
- output: JSON with 200 status code success
- exception:

remove_worker(user_id):

- transition:
  if user_id exists:
  delete workers[user_id]
- output: JSON with 200 status code success
- exception:

video_feed(user_id):

- transition:
  if user_id exists:
  resp = gen(user_id)
- output: Response with form data containing the processed frames
- exception:

7.4.5 Local Functions

The generator function that generates the frame for different users
gen(user_id):

- transition: frame = worker[user_id].get_frame()
- output: yield formdata containing the frame

8 MIS of Hair Color Module

8.1 Module

M3 - HairColor Module
Abstract Object Module

8.2 Uses

Model Utils (M8)
Image Utils (M9)
~~MLModel (M6)~~
~~Utility (M7)~~

8.3 Syntax

8.3.1 Exported Constants

8.3.2 Exported Access Programs

Name	In	Out	Exceptions
changeHairColor	image, string	image	KeyErrorException
setAlpha	double		
reset		string	

8.4 Semantics

8.4.1 State Variables

colors - Map<str, int[]> - a mapping between the name of color and their rgb values
alpha - double - represents the ratio between the original image and the masked image
min_confidence_value - double - the minimum confidence interval for machine learning model

8.4.2 Environment Variables

8.4.3 Assumptions

8.4.4 Access Routine Semantics

changeHairColor(image, color):

- input:
  image - the copy of an original image
  color - the chosen hair color
- transition: N/A

- output:
  hairModelSession = utility.getHairModel()
  mask = performHairSegmentation(hairModelSession, hairModelSession.inputName, hairModelSession.output image) - the image where the hair detected by the model is masked.
  outputImg = changeColor(image, mask, color)
  return outputImg - an image where the hair color of each person is changed to the specified color

- exception: InterruptException - the prediction and masking process is interrupted by the user

setAlpha(newAlpha):

- input: newAlpha - double - input alpha value for update

- transition: alpha := newAlpha - update the alpha value

- output: N/A

- exception: N/A

reset():

- transition:

- output: message => MLModel.reset(hair)

- exception:

8.4.5 Local Functions

performHairSegmentation(session, input, output, image):

- input: session - the onnx inference session that contains the input model
  input - list of integer - the input shape of the image
  output - list of integer - the expected output shape
  image - the copy of an original image

- transition: N/A

- output: mask - hair mask.

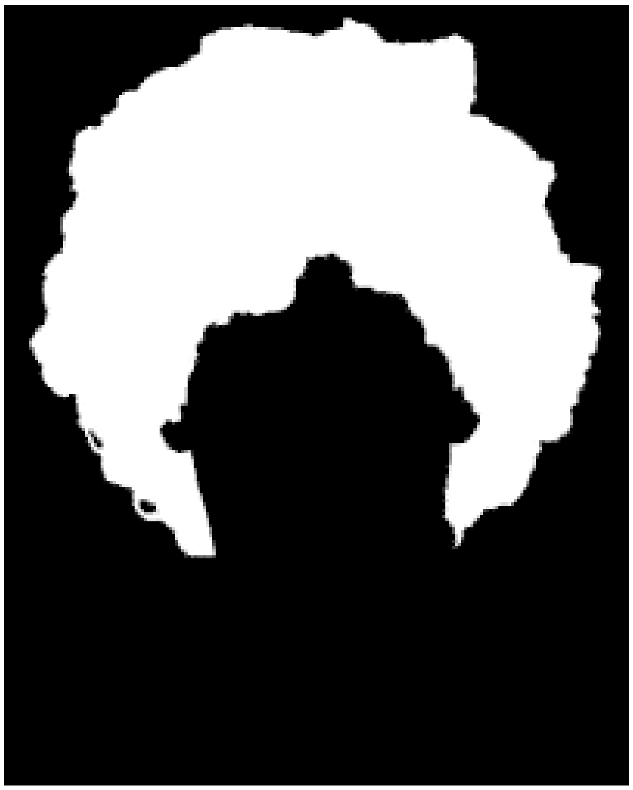

Figure 3: Hair Mask after running the pre-trained hair segmentation model

- exception: KeyErrorException - the specified color is not in the color map

changeColor(img, mask, color):

- input: img - the original image, mask - the masked image generated from hair segmentation, color - color's name as a string
- transition: N/A
- output: an image where the original image is mixed with the masked image.
- exception: KeyErrorException - the specified color is not in the color map

9 ~~MIS of Hair Style Module~~

9.1 Module

M9 - FacialRecognition
Abstract Object Module

9.2 Uses

MLModel (M6)
Utility (M7)

9.3 Syntax

9.3.1 Exported Constants

9.3.2 Exported Access Programs

Name	In	Out	Exceptions
computeHairCoordinate	list[double], list[double]	list[double]	
computeRotationDegree	list[double], list[double]	list[double]	

9.4 Semantics

9.4.1 State Variables

9.4.2 Environment Variables

9.4.3 Assumptions

9.4.4 Access Routine Semantics

computeHairCoordinate(basePosition, facialCoordinates):

- input: basePosition - the basePosition of the camera setting in a tuple
  facialCoordinates - a list of coordinates of the facial features
- transition: N/A
- output: output the desired position to place the hairstyle centered at a coordinate, computed based on the base position and facial coordinates.
- exception: InterruptException := action terminated by the user

computeRotationDegree(basePosition, facialCoordinates):

- input: basePosition - the basePosition of the camera setting in a tuple
  facialCoordinates - a list of coordinates of the facial features
- transition: N/A
- output: output the desired rotation of the hairstyle when being placed on the user's face, computed based on the base position and facial coordinates.

9.4.5 Local Functions

10 MIS of Worker Module

10.1 Module

M4 - Worker Module
Abstract Data Type Module

10.2 Uses

Hair Artist (M7)
Threading

10.3 Syntax

10.3.1 Exported Constants

10.3.2 Exported Access Programs

Name	**In**	**Out**	**Exceptions**
__init__	HairArtist	Worker	
enqueue_input	img, tuple		
get_frame		img	
clean_up			

10.4 Semantics

10.4.1 State Variables

thread - Thread - daemon thread processing images in the background
to_process - Deque - Double ended queue that stores input images (FIFO)
to_output - Deque - Double ended queue that stores output images (FIFO)
makeup_artist - HairArtist - HairArtist instance that used for image processing

10.4.2 Environment Variables

10.4.3 Assumptions

10.4.4 Access Routine Semantics

__init__(hairArtist):

- input: hairArtist - HairArtist instance that used for image processing

- transition:
  to_process := Deque([])
  to_output := Deque([])
  makeup_artist := hairArtist
  thread = Thread(target=keep_processing())
- output:
- exception:

enqueue_input(img, color):

- input: img - the input image from user
  tuple(color) - the input color to change hair to
- transition:
  to_process.append((img, color))
- output:
- exception:

get_frame():

- input:
- transition:
  frame = to_output.popleft(img)
- output:
  return - frame - processed image frame in order
- exception:

clean_up():

- input:
- transition:
  to_output.clear()
  to_process.clear()
- output:
- exception:

10.4.5 Local Functions

keep_processing():

- input:
- transition:
  while !to_process().empty():
  process_one()
- output:
- exception:

process_one():

- input:
- transition:
  img, color = to_process.popleft()
  res = hair_artist.apply_hair_color(img, color)
  to_output.append(res)
- output:
- exception:

11 MIS of Image Worker Module

11.1 Module

M5 - Image Worker Module
Abstract Data Type Module

11.2 Uses

Hair Artist (M7)

11.3 Syntax

11.3.1 Exported Constants

11.3.2 Exported Access Programs

Name	In	Out	Exceptions
__init__	HairArtist	ImageWorker	
process_one	img, tuple		

11.4 Semantics

11.4.1 State Variables

makeup_artist - HairArtist - HairArtist instance that used for image processing

11.4.2 Environment Variables

11.4.3 Assumptions

11.4.4 Access Routine Semantics

__init__(hairArtist):

- input: hairArtist - HairArtist instance that used for image processing

- transition:
  to_process := Deque([])
  to_output := Deque([])
  makeup_artist := hairArtist
  thread = Thread(target=keep_processing())
- output:
- exception:

process_one(img, color):

- input: img - input image for hair segmentation
  color - input color to change the hair

- transition:
  res = hair_artist.apply_hair_color(img, color)

- output:

- exception:

11.4.5 Local Functions

12 MIS of Salon Recommendation Module

12.1 Module

M5: Salon Recommendation Module
Abstract Object Module

12.2 Uses

12.3 Syntax

12.3.1 Exported Constants

12.3.2 Exported Access Programs

Name	In	Out	Exceptions
get_size		int	
getNearbySalons		list[salons]	

12.4 Semantics

12.4.1 State Variables

lat - double - latitude
lng - double - longitude
salons - list[dict] - contains the information of salons
size - int - size of nearby salons
total_average_rating - double - the average rating among all salons

12.4.2 Environment Variables

12.4.3 Assumptions

12.4.4 Access Routine Semantics

get_size():

- input:
- transition:
- output:
  return - size - the size of nearby salons
- exception:

getNearbySalons():

- input:
- transition: nearby_salons = fetchNearbySalons(lat, lng)
  foreach salon in nearby_salons: details = fetch_place_details
  salons.append(details)
  rank_salons(salons)
- output:
  return - salons - the sorted nearby salon details list
- exception:

12.4.5 Local Functions

fetchNearbySalons():

- input:
- transition:
- output:
  return nearby_salons - response from google map API.
- exception:

fetchPlaceDetails(place_id):

- input:
- transition:
- output:
  return salon_info - the salon information with the given place_id
- exception:

rank_salons():

- input:
- transition: sort the salons with their average ratings generated by the bayesian algorithm
- output:
- exception:

13 MIS of Hair Artist Module

13.1 Module

M7 - HairArtist Module
Abstract Data Type Module

13.2 Uses

Model Utils Module (M8)
Image Utils Module (M9)
Hair Color Module (M3)

13.3 Syntax

13.3.1 Exported Constants

13.3.2 Exported Access Programs

Name	In	Out	Exceptions
apply_hair_color	img, color	img	

13.4 Semantics

13.4.1 State Variables

face_recognition - mediapipe instance of face recognition
hair_model - ModelUtils instance with session that performs hair segmentation

13.4.2 Environment Variables

13.4.3 Assumptions

13.4.4 Access Routine Semantics

apply_hair_color(img, color):

- input: img - input image for hair segmentation
  color - input color to change the hair
- transition:
  res = hair_color.change_hair_color(img, color, hair_model, face_recognition)
- output: res - img - processed frame with hair color changed as desired
- exception:

13.4.5 Local Functions

14 MIS of ModelUtils Module

14.1 Module

M8 - Model Utils Module
Abstract Data Type Module

14.2 Uses

Onnx (External Module)

14.3 Syntax

14.3.1 Exported Constants

14.3.2 Exported Access Programs

Name	In	Out	Exceptions
initialize_hair_segmentation_model		FileNotFoundException	

14.4 Semantics

14.4.1 State Variables

minConfidenceLevel - 0.5 - minimum confidence accuracy for output
inputShape - tuple - represent model input shape
outputShape - tuple - represent model output shape
modelFilePath - filePath (path to the pre-trained model)
model_session - ML model with active session

14.4.2 Environment Variables

14.4.3 Assumptions

14.4.4 Access Routine Semantics

initialize_hair_segmentation_model():

- transition:
  model_session = onnx_inference_session(modelFilePath, minConfidenceLevel)
  inputShape = model_session.input_shape
  outputShape = model_session.output_shape

- output:
- exception:

14.4.5 Local Functions

15 MIS of Image Utils Module

15.1 Module

M9 - Utility Module
Library

15.2 Uses

OpenCV (External Module)
Numpy (External Module)
Pillow (External Module)
BytesIO (External Module)
base64 (External Module)

15.3 Syntax

15.3.1 Exported Constants

15.3.2 Exported Access Programs

Name	In	Out	Exceptions
processImage	image, list[int]	tensor	illegalArgumentException
maskImage	image, image	image	illegalArgumentException
toGreyScale	image	image	
resizeImage	image, list[int]	image	illegalArgumentException
cv2_image_to_base64	img	string	illegalArgumentException
base64_to_cv2_image	string	img	illegalArgumentException

15.4 Semantics

15.4.1 State Variables

mean - list[double] - the mean values of trained images, used to normalize the images
std - list[double] - the standard deviation values of trained images, used to normalize the images

15.4.2 Environment Variables

15.4.3 Assumptions

15.4.4 Access Routine Semantics

processImage(image, input_size):

- input: image - the original input image in the form of 3-dimensional array, input_size - a tuple represents the input size the ML model requires
- transition: N/A
- output:
  OpenCV.convertColor(image, BGR2RGB) - convert the image to RGB format
  resizeImage(image, input_size) - convert image to input size
  image = (image / 255 - mean) / std - normalize the image
  Numpy.expandDimension(image, axis=0) - expand one dimension to a tensor
  output a image tensor ready for process with the model
- exception: illegalArgumentException - illegal input size for resizing

maskImage(original_img, mask):

- input: original_img - the original image in the form of 3-dimensional array, mask - the masked image in the form of 3-dimensional array with same dimension as original
- transition: N/A
- output:
  OpenCV.bitwise_or(original_img, original_img, mask) - apply masking to the original image with the given mask.
  output a masked image.
- exception: illegalArgumentException - original image has different size from the masked image.

toGreyScale(image):

- input: image - the input image to be converted to grey scale
- transition: N/A
- output:
  OpenCV.convertColor(image, BGR2GRAY) - convert the input image to an grey scale image
  output the greyscaled image
- exception:

resizeImage(image, shape):

- input: image - the input image
  shape - a tuple represents the width / height to be reshaped into.
- transition: N/A
- output:
  Numpy.reshape(image, shape) - reshape the image
  output an reshaped image
- exception: illegalArgumentException - illegal input size for resizing

cv2_image_to_base64(image):

- input: image - the input image
- transition: N/A
- output:
  img = PIL.Image.fromarray(img)
  base64.b64encode(img)
- exception: illegalArgumentException - input array can not be transformed into image

base64_to_cv2_image(string):

- input: string - image string encoded in base64
- transition: N/A
- output:
  str = base64.b64decode(string)
  img = PIL.Image.open(str)
- exception: illegalArgumentException - input string can not be transformed into image

15.4.5 Local Functions

None

16 MIS of Hair Color View Module

16.1 Module

M8 - HairColorView
Abstract Object Module

16.2 Uses

None

16.3 Syntax

16.3.1 Exported Constants

None

16.3.2 Exported Access Programs

Name	In	Out	Exceptions
TabPanel	prop	React.component	-
makeid	number	number	-
HairColorView	void	React.component	-

16.4 Semantics

16.4.1 State Variables

socket : Socket.io [io socket connected on localhost/5001 —SS]

16.4.2 Environment Variables

Screen, Camera

16.4.3 Assumptions

None

16.4.4 Access Routine Semantics

TabPanel(prop):

- transition: None
- output: the tab panel for the user to select different hair colors with the given React properties.

- exception: None

makeid(length):

- transition: None
- output: generate a random id number with the given length.
- exception: None

HairColorView():

- transition: None
- output: generate the HairColorView page.
- exception: None

17 MIS of Hair Style View Module

17.1 Module

M9 - HairStyleView
Abstract Object Module

17.2 Uses

HairStyleModule

17.3 Syntax

17.3.1 Exported Constants

None

17.3.2 Exported Access Programs

Name	In	Out	Exceptions
HairStyleView	void	React.component	-

17.4 Semantics

17.4.1 State Variables

themeDark : Theme [MUI Theme for frontend UI View —SS]
useStyles: Styles [MUI styles for frontend UI View —SS]

17.4.2 Environment Variables

Screen, Camera

17.4.3 Assumptions

None

17.4.4 Access Routine Semantics

HairStyleView():

- transition: None
- output: generate the HairStyleView Page
- exception: None

18 MIS of Salon Recommendation View Module

18.1 Module

M10 - SalonRecommendationView
Abstract Object Module

18.2 Uses

None

18.3 Syntax

18.3.1 Exported Constants

None

18.3.2 Exported Access Programs

Name	In	Out	Exceptions
SalonRecommendataionView	void	React.component	-

18.4 Semantics

18.4.1 State Variables

selected: {lat, lng} [selected user location by latitude and longitude —SS]

18.4.2 Environment Variables

Screen

18.4.3 Assumptions

None

18.4.4 Access Routine Semantics

SalonRecommendationView():

- transition: None
- output: generate the SalonRecommendationView Page
- exception: None

18.4.5 Local Functions

Map():

- transition: None
- output: generate the GoogleMap React Component
- exception: None

PlaceAutoComplete():

- transition: None
- output: generate the autocomplete search bar react component
- exception: None

locateUserLocation():

- transition: selected := located user location
- output: None
- exception: None

searchNearbySalon(lat, lng):

- transition: None
- output: use google map api to get nearby salons by the given latitude and longitude
- exception: None

19 MIS of Home View Module

19.1 Module

M11 - HomeView Module

19.2 Uses

None

19.3 Syntax

19.3.1 Exported Constants

None

19.3.2 Exported Access Programs

Name	In	Out	Exceptions
HomeView	void	React.component	-

19.4 Semantics

19.4.1 State Variables

None

19.4.2 Environment Variables

Screen

19.4.3 Assumptions

None

19.4.4 Access Routine Semantics

HomeView():

- transition: None
- output: generate the HomeView Page
- exception: None

20 MIS of Footer Module

20.1 Module

M14 - Footer
Abstract Object Module

20.2 Uses

react, react-bootstrap, react-icons

20.3 Syntax

20.3.1 Exported Constants

None

20.3.2 Exported Access Programs

Name	In	Out	Exceptions
Footer	void	React.component	-

20.4 Semantics

20.4.1 State Variables

None

20.4.2 Environment Variables

None

20.4.3 Assumptions

None

20.4.4 Access Routine Semantics

Footer():

- transition: None
- output: generate the Footer component.
- exception: None

21 MIS of NavBar Module

21.1 Module

M15 - NavBar
Abstract Object Module

21.2 Uses

react, react-bootstrap, react-router-dom

21.3 Syntax

21.3.1 Exported Constants

None

21.3.2 Exported Access Programs

Name	In	Out	Exceptions
NavBar	void	React.component	-

21.4 Semantics

21.4.1 State Variables

None

21.4.2 Environment Variables

None

21.4.3 Assumptions

None

21.4.4 Access Routine Semantics

Footer():

- transition: None
- output: generate the NavBar component.
- exception: None

22 MIS of ARCanvas Module

22.1 Module

M16 - ARCanvas Module
Abstract Object Module

22.2 Uses

react, @react-three/fiber, @react-three/drei, facefilter, ThreeFiberHelper, HairModel

22.3 Syntax

22.3.1 Exported Constants

None

22.3.2 Exported Access Programs

Name	In	Out	Exceptions
FaceFollower	props	React.component	
ThreeGrabber	props	React.component	
compure_sizing		list[double]	
ARCanvas	selectedHair	React.component	
handle_resize			
do_resize			
useEffect	sizing		
callbackReady	errCode, spec		
callbackTrack	detectStatesArg		
callbackDetect	faceIndex, isDetected		
ARCanvas	url	React.component	-

22.4 Semantics

22.4.1 State Variables

_maxFaceDetected: int
_faceFollowers: Array[int]
expressions: null
_threeFiber: null

22.4.2 Environment Variables

window

22.4.3 Assumptions

None

22.4.4 Access Routine Semantics

FaceFollower(props):

- transition: None
- output: Generate the FaceFollower React component
- exception: None

compute_sizing():

- transition: None
- output: {width, height, top, left}
- exception: None

ARCanvas(selectedHair):

- transition: selectedHair := event.id
- output: Generate the FaceFollower React component
- exception: None

handle_resize():

- transition: _timerResize := setTimeout(do_resize, 200)
- output: None
- exception: None

do_resize():

- transition: _timerResize := null, sizing := newSizing
- output: None
- exception: None

useEffect(sizing):

- transition: _timerResize := null, sizing := newSizing
- output: None
- exception: None

callbackReady(errCode, spec):

- transition: _timerResize := null, sizing := newSizing
- output: None
- exception: None

callbackTrack(detectStatesArg):

- transition: detectStates := detectStatesArg.length
- output: None
- exception: None

callbackDetect(faceIndex, isDetected):

- transition: None
- output: None
- exception: None

23 MIS of HairModel Module

23.1 Module

M17 - HairModel Module Abstract Object Module

23.2 Uses

react, @react-three/drei

23.3 Syntax

23.3.1 Exported Constants

None

23.3.2 Exported Access Programs

Name	**In**	**Out**	**Exceptions**
HairModel	url	React.component	-

23.4 Semantics

23.4.1 State Variables

nodes: Three.mesh.geometry
materials: Three.mesh.material

23.4.2 Environment Variables

Screen

23.4.3 Assumptions

None

23.4.4 Access Routine Semantics

HairModel(url):

- transition: None
- output: Import the gltf model from url and pre-load the Hair Model as React component
- exception: None

24 MIS of ThreeFiberHelper Module

24.1 Module

M18 - ThreeFiberHelper Module Abstract Object Module

24.2 Uses

Three

24.3 Syntax

24.3.1 Exported Constants

None

24.3.2 Exported Access Programs

Name	**In**	**Out**	**Exceptions**
update_poses	number[], ThreeCamera	-	-
update	number[], ThreeCamera	-	-
create_occluder	any	Meshany, ShaderMaterial	-
update_camera	{width, height}, ThreeCamera	-	-
ThreeFiberHelper	-	React.component	-

24.4 Semantics

24.4.1 State Variables

_settings: double, [double, double], double, double: _threeFiberCompositeObjects: null
_threeProjMatrix : null
_previousSizing: int, int
_threeTranslation: null
_maxFaces: int
_detectCallback: null
_videoElement: null
_scaleWL: int
_canvasAspectRatio: int
_threeTranslation: Three.Vector3
_threeProjMatrix: Three.Matrix4

24.4.2 Environment Variables

threeCamera

24.4.3 Assumptions

None

24.4.4 Access Routine Semantics

update_poses(ds, threeCamera):

- transition: halfTanFOVX := $tan(threeCamera.aspect * threeCamera.fov * pi/360)$
- output: None
- exception: None

update(detectStates, threeCamera):

- transition: None
- output: None
- exception: None

create_occluder(occluderGeometry):

- transition:
- output: occluderMesh := Three.Mesh(occluderGeometry)
- exception: None

update_camera(sizing, threeCamera):

- transition: threeCamera.aspect := _canvasAspectRatio, threeCamera.fov := fov, threeCamera.view := null
- output: None
- exception: None

ThreeFiberHelper():

- transition: None
- output: ThreeFiber object that finished setup and face detection
- exception: None

24.4.5 Local Functions

detect(detectState):

- input: None
- transition: detect face and set up AR using ThreeFiber library on state variable _settings
- output: None
- exception:

25 Appendix