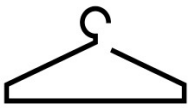
Bilkent University

Department of Computer Engineering

Senior Design Project

Project name: Dressy



Dressy.

*Your own virtual wardrobe

Low Level Design Report

Asuman Aydın, Doğukan Köse, Fatih Çakır, Muhammed Musab Okşaş, Şeyma Aybüke Ertekin

Supervisor: Uğur Güdükbay

Website Link: https://dress-y.github.io/

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

1 Introduction

The preference of online shopping is a huge trend in today's digital world because of the lack of time, the convenience of online shopping, the advantage to have access to other users' experiences, the existence of cost choices and richness in the product variety. By the beginning of 2020, 69% of Americans had an online shopping experience. Also, 25% of them were regular online shoppers who bought at least one item each month. Statistics also demonstrate that 47% of online shoppers buy clothing items. The online shopping trend is not limited to America. If we look at the worldwide statistics, the rate of online shopping in 2018 is 47.3%. While 72% of women had preferred to shop items online while 40% of these items, the largest percentage, were clothing. With the effect of the coronavirus, this preference has become even clearer. People have begun to prefer to buy most of their needs online rather than visiting crowded shopping malls and trying out clothes that they do not know who had worn previously. [1] However, shopping for clothes online has a downside. Shoppers do not have the advantage of trying clothes and therefore, it is difficult to make good choices in terms of looks and size of clothing.

Figure 1. Statistics on presentation of increase on online shopping after Covid-19

The purpose of our senior design project is to assist people who choose to buy their clothes online. We want to create an application that provides a virtual fitting room for them. Thereby, they can try the clothes they have chosen and see the clothes on themselves.

With this project specifications report, we aim to explain the description of our application, current systems, requirements, diagrams about our senior project, Dressy. Also, we will provide mockups of screens of our app.

1.1 Object design trade-offs

1.1.1 Functionality vs Usability

With the advanced technologies added to the products, the purpose of appealing to every audience can sometimes be inversely proportional. The clothing trial feature, a function we provide to users here, should be easily understandable to users. In this respect, functionality lags behind usability and is less important to us. This does not mean that functionality is not important. As for usability increases, space will be opened for further improvable functionality.

1.1.2 Maintainability vs Extensibility

Maintainability is, in many ways, one of the most important elements of this project for us. The main reason why it is compared with extendibility here is that ensuring that it has more active ingredients to reach more people will have a detrimental effect on the maintainability of the application. While we want it to be active, we think that expansion will not be a solid commercial and technological step without creating a solid structure. In this respect, maintainability often comes before extendibility.

1.1.3 Privacy vs Usability

To increase usability, the protection of every information received from the user within a privacy framework highlights the privacy element in this comparison. Although increasing user interactions is an important factor for an application, failure to provide privacy will be a major problem for Dressy. When considering the ml algorithm, which is trained with image and personal information, the information should be protected with great importance.

1.1.4 Reliability vs Cost

While doing trade-offs on the robustness of the application, we do not want to make a mistake by not considering what factor we have in terms of cost at the design stage. The fact that factors such as cost and time are the sources for many remaining factors for the development of the project causes the cost to come before most of the factors at this point.

1.2 Interface documentation guidelines

After a Class Diagram, a table as shown below will be used to define each class. The signup template can be taken as an example from the View section written by React Native. There are attributes in the class. These attributes correspond to the objects used in functions. Don't be fooled by the scarcity of functions, you can think of the render method as the main method, since there is no obvious function and main function structure in the React Native structure.

Class Sample	
This class is for Signing Up to the application	
Attributes	
string Name string Surname string Email	
Operations	
public void Signup()	: Signing up to the application with name, surname etc.
public void Render()	: Displaying view for the signup page jsx.
public void ComponentDidMount()	: Checks if the user signed up.

Figure 2. Example of Class Interfaces

1.3 Engineering standards

There will be use of UML guidelines on the descriptions of class interfaces, diagrams, use cases etc. IEEE standards will be used to fulfill the scientific and academic side of the report such as citations.

1.4 Definitions, acronyms, and abbreviations

IaaS	Infrastructure As Service with cloud data systems. It can be referred to as a transfer and store system for application. [2]
REST API	Application programming interface (API or web API) that conforms to the constraints of REST architectural style and allows for interaction with RESTful web services. [3]
MySQL	Type of database management system.
DoS	Attack meant to shut down a machine or network, making it inaccessible to its intended users. [4]
React Native	JavaScript framework for writing real, natively rendering mobile applications for iOS and Android

Figure 3. Definitions, Acronyms, Abbreviations

2 Packages

There are two types of packages in the Dressy application. These are called Client and Server. On the client side, there are three parts as Controller, View and Data. Controller package is used for routing and optimization applications within the application. View includes Javascript and CSS codes of the pages designed for UI. Structures will be presented as screen packages. The data package shows the controls in the reflection of the information used on the front-end side of the application.

2.1 Client Package

Client Package symbolizes the visible side of the application. The Controller, which makes sure that the data received by the server are displayed correctly on the screen, consists of a View that makes sure that it is displayed well and a Data Package that makes sure that the data are correct. The most important feature on the client side is to ensure that the information received by the Client is reflected to the Server side while modeling and that a robust and accurate model is created.

2.1.1 Controller

Controller classes manage the bridge between the client and the server side applications. This package provides registering, modifying and receiving data. Authentication and authorization services are used in this package. Additionally, permissions of clients' cameras, taking and sending videos are managed via this package.

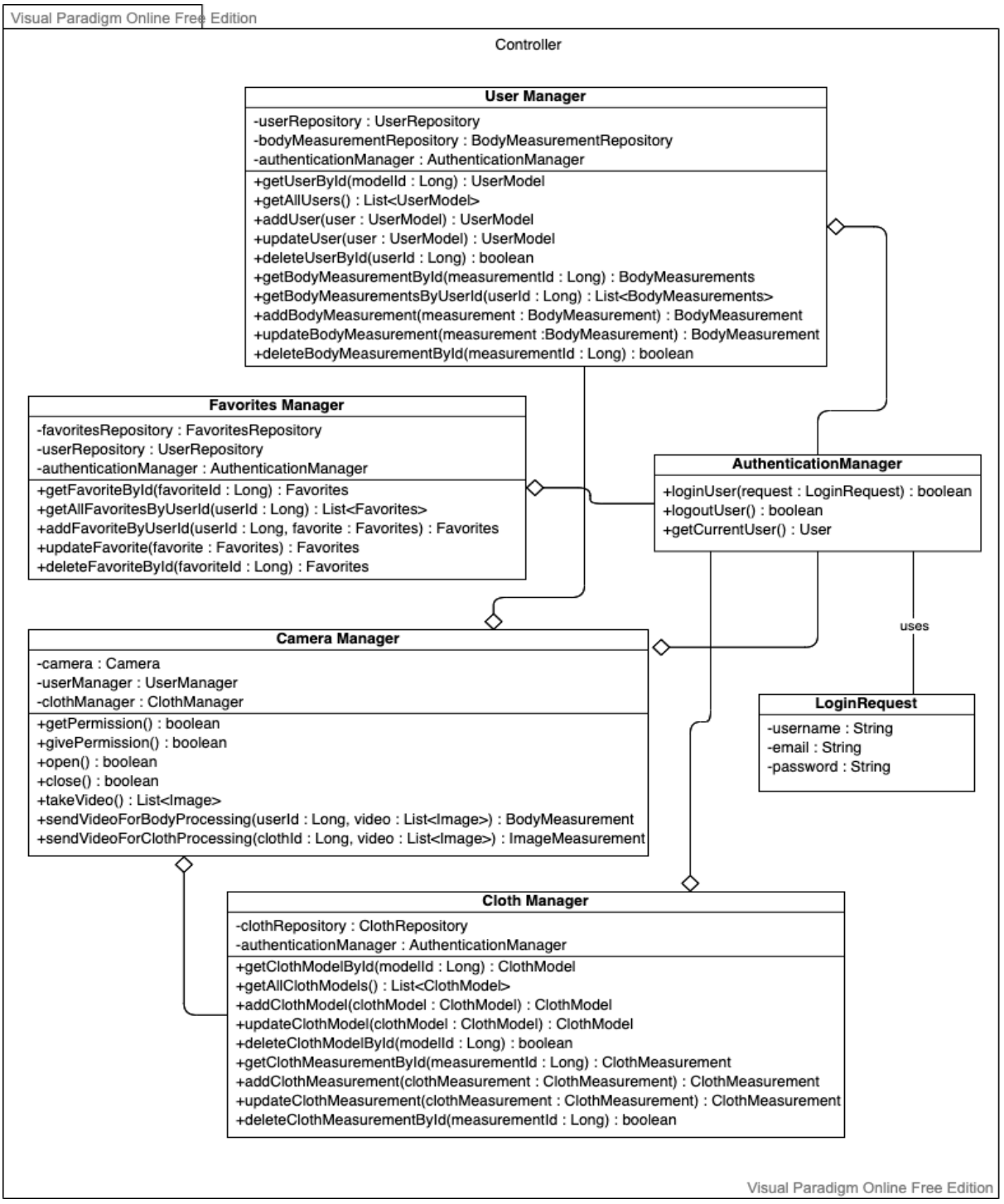


Figure 4. Controller Package

Authentication Manager

Authentication Manager handles security, authentication and authorization for dressy applications. It also provides current user information.

User Manager

User Manager handles user model related services. It provides create, read, update, delete services for user model and body measurements.

Favorites Manager

Favorites Manager handles user's favorites services. It provides create, read, update, delete services for the user's favorite clothes.

Cloth Manager

Cloth Manager handles cloth model related services. It provides create, read, update, delete services for cloth model and cloth measurements.

Camera Manager

Camera Manager handles permissions of clients' cameras, taking and sending videos for body and cloth measurements.

2.1.2 View

The reflections of the screens seen by the user in the application on the coding side are given in the view package. Their connections with each other will be specified in the class diagram. Missing pages may stand out, but they will be included in these packages as small classes or functions.

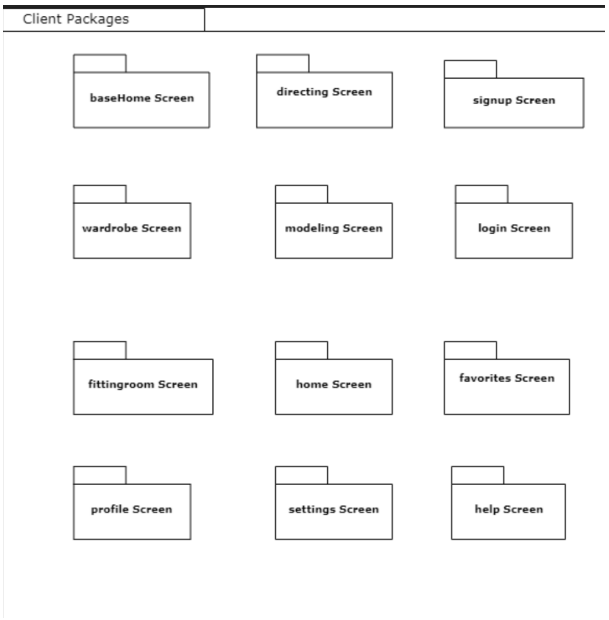


Figure 5. View Package

BaseHome Screen:

This view is responsible for opening of applications.

Directing Screen:

This view is responsible for directing the base home page to the modeling page for users to open the camera and model themselves.

Favorites Screen:

This view is responsible for favoring clothes and materials to try out at the fitting room.

Fitting Room Screen:

This view is responsible for trying out clothes, liking them and also favoring them.

Help Screen:

This view is responsible for explaining the application and also directing to the contact information for help.

Home Screen:

This view is responsible for the main page display with buttons to direct to the other views.

Login Screen:

This view is responsible for log in to the home screen and validation of the user.

Modeling Screen:

This view is responsible for having the model of the user as a whole body and using it to try out the clothes in the fitting room.

Profile Screen:

This view is responsible for displaying user information in detail and maybe directing to the settings so that users can change the settings and information of the application.

Settings Screen:

This view is responsible for changing the settings or making updates on the application.

Signup Screen:

This view is responsible for signing up to the home screen and validation of the user.

Wardrobe Screen:

This view is responsible for having clothes and materials to try out at the fitting room.

2.2 Application Package

2.2.1 DensePoseLogic

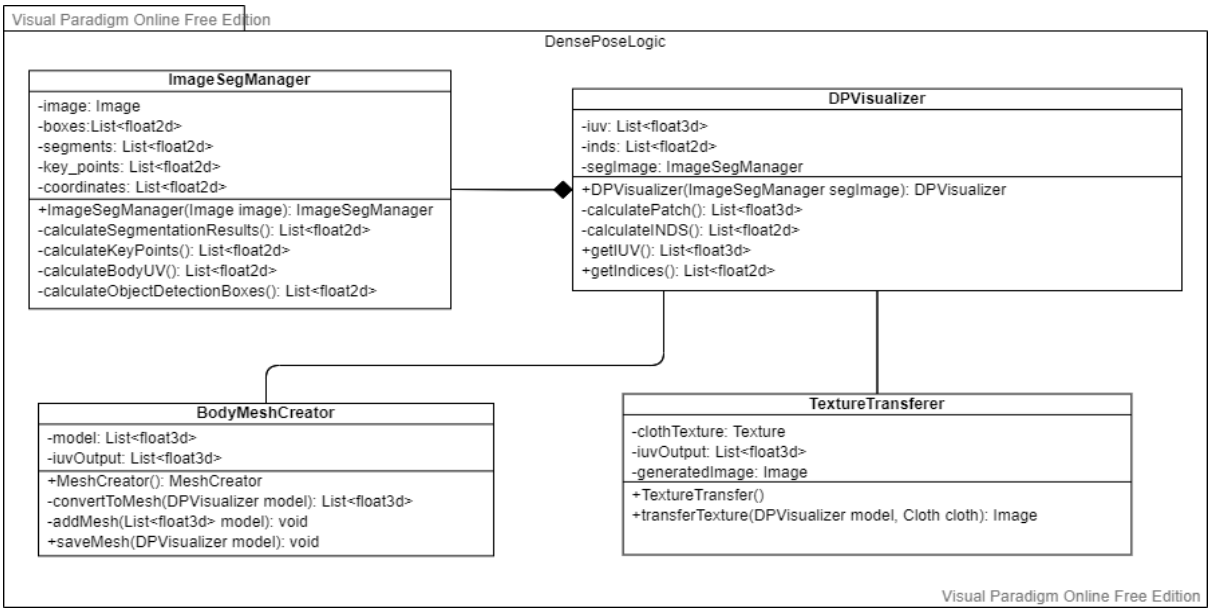


Figure 6. DensePose Logic

DensePoseLogic Package gets camera images and processes them. The purpose of this package is to map a cloth texture to a human body and create a body mesh of a human and save it into the database in order to be able to make size estimations. This package communicates with the Controller package to get the camera views and Data Package to save body mesh into the database. This package contains 4 classes which are shortly explained below.

ImageSegManager:

This class calculates bounding boxes, key points, and UV coordinates of a given image using machine learning models.

DPVisualizer:

This is a DensePose visualizer class. This class converts boxes, key points, and UV coordinates into a visualizable format. It adds indices, patches, to UV coordinates and creates IUV output. It also calculates INDS output of the image.

TextureTransferer:

This class transfers cloth texture to a human body. It uses IUV coordinates to be able to do this.

BodyMeshCreator:

This class creates a human body mesh using SMPL model standards and IUV coordinates of a human body image.

2.2.2 SizeEstimationManager

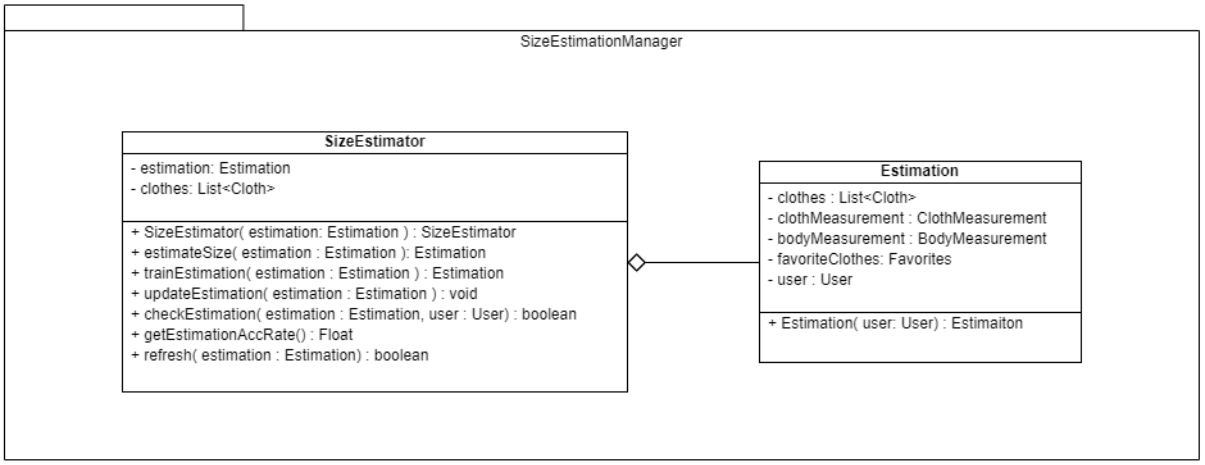


Figure 7. SizeEstimationManager Package

Size estimation manager is responsible for estimating and recommending clothes with convenient size according to the user's favourite clothes. There is an Estimation class and SizeEstimator class in this package.

2.2.3 3DClothManager

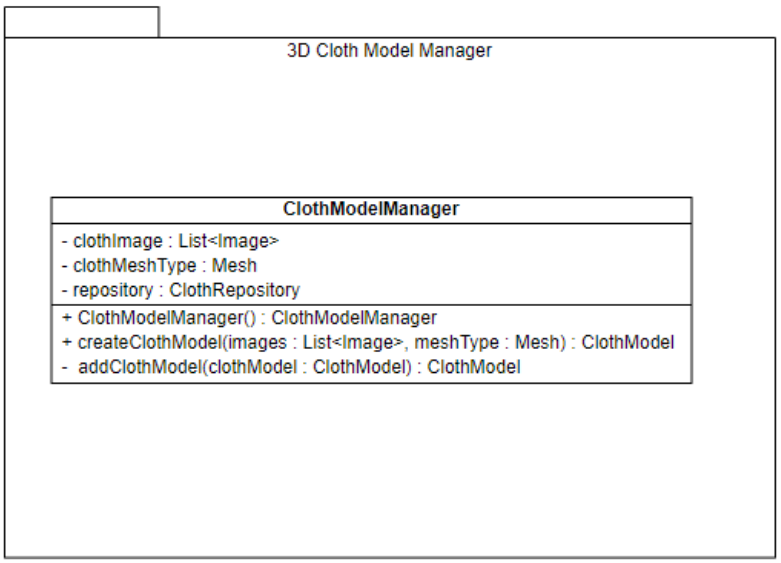


Figure 8. Cloth Model Package

The 3D Cloth model manager package will be used for creating clothes on the application. It has only one class. ClothModelManager class takes cloth images and its mesh type. It extracts texture from an image and maps it on given mesh type. Then, it saves the model to the database.

2.3 Data Package

Data Package manages interactions with the database. It will communicate with the Application Package and Controller Package to give the requested data. This package contains 2 main data classes which are user and cloth.

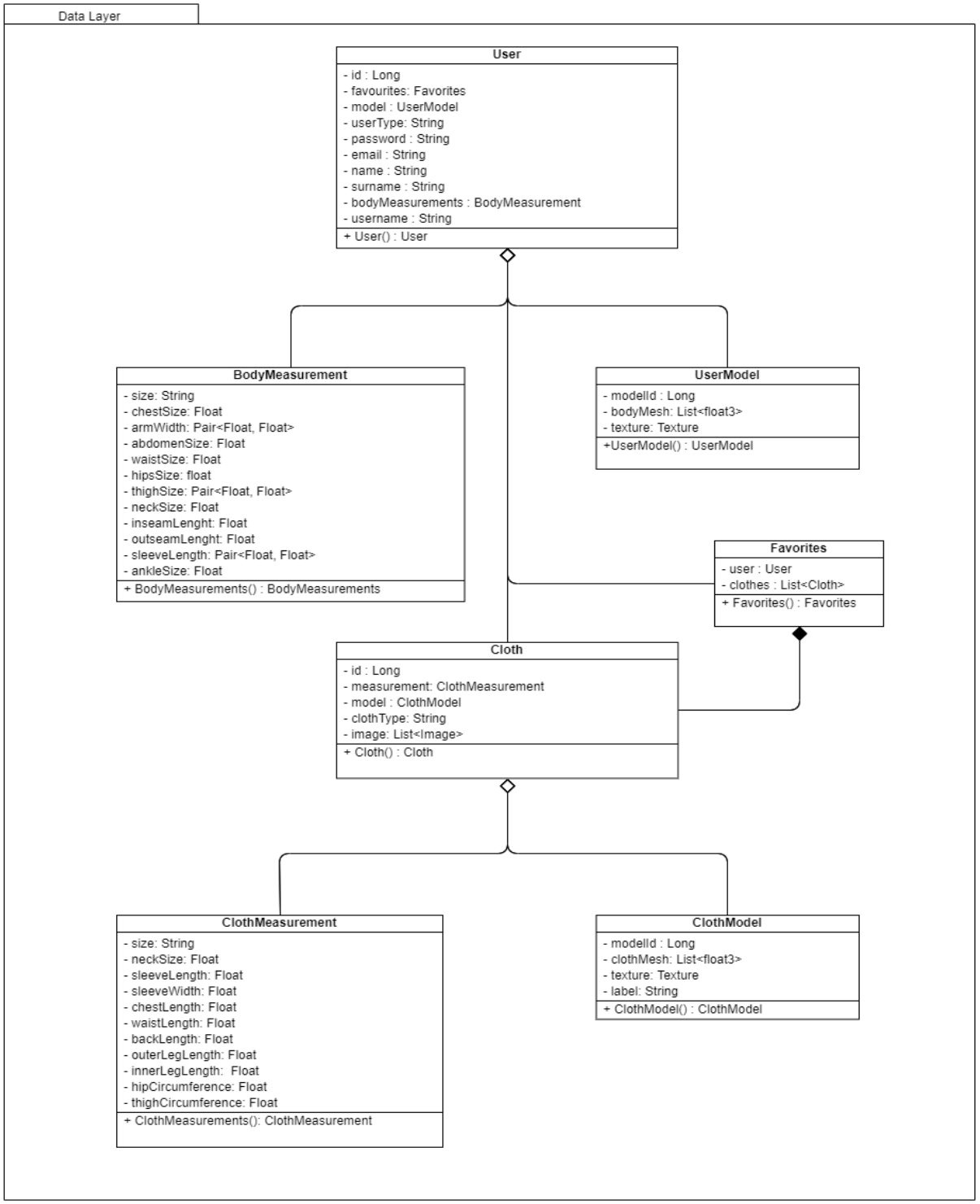


Figure 9. Data Package

User:

User entities will be used both in the controller and application layer in this project for the whole process.

Body Measurement:

Body Measurement contains size information about the user body. This information will be taken both from the user as an input and processed mesh model. So that created body model can represent in the most realistic way.

User Model:

The User Model keeps necessary information about the body of the user.

Favorites:

Favorites store the favourite clothes of the user.

Cloth:

These clothes will be shown in the client and will be used by the application layer to fit a cloth on a user body.

Cloth Measurement:

Cloth Measurement contains size information about a cloth. The information will be used for giving advice to a user. So that users can select the best fitting clothes for themselves.

Cloth Model:

Cloth Model keeps necessary information for fitting a cloth on a user.

3 Class Interfaces

3.1 Client

3.1.1 Controller

Class AuthenticationManager
This class is for handling security, authentication and authorization for dressy application. It also provides current user information.
Operations
public loginUser(LoginRequest request) : Logins given user public logoutUser() : Logouts current user public getCurrentUser() : Returns the current user if anybody is logged in

Class UserManager
This class is for handling user model related services. It provides create, read, update, delete services for user model and body measurements.
Attributes
private UserRepository userRepository private BodyMeasurementRepository bodyMeasurementRepository private AuthenticationManager authenticationManager
Operations
public getUserById(Long id) : Returns the user model by given id public getAllUsers() : Returns all user models public addUser(UserModel userModel) : Adds given user model to the database public updateUser(UserModel userModel) : Update corresponding user model public deleteUserById(Long id) : Deletes the user model by given id public getBodyMeasurementById(Long measurementId) : Returns the body measurement by given id public getBodyMeasurementsByUserId(Long userId) : Returns the body measurements by given user id public updateBodyMeasurement(BodyMeasurement measurement) : Update corresponding body measurement public deleteBodyMeasurementById(Long measurementId) : Deletes the body measurement by given id

Class FavoritesManager
This class is for handling a user's favorites service. It provides create, read, update, delete services for the user's favorite clothes.
Attributes
private FavoritesRepository favoritesRepository private BodyMeasurementRepository bodyMeasurementRepository private AuthenticationManager authenticationManager
Operations
public getFavoritesById(Long id) : Returns the favorites list by given id public getAllFavoritesByUserId(Long userId) : Returns the favorites list by given user id public addFavoriteByUserId(Long userId, Favorites favorite) : Adds given favorites list for given user id to the database public updateFavorite(Long id) : Update corresponding favorites list public deleteFavoriteById(Long id) : Deletes the favorites list by given id

Class ClothManager
This class is for handling cloth model related services. It provides create, read, update, delete services for cloth model and cloth measurements.
Attributes
private ClothRepository clothRepository private AuthenticationManager authenticationManager
Operations
public getClothModelById(Long id) : Returns the cloth model by given id public getAllClothModels() : Returns all cloth models public addClothModel(ClothModel clothModel) : Adds given cloth model to the database public updateClothModel(ClothModel clothModel) : Update corresponding cloth model public deleteClothModelById(Long id) : Deletes the cloth model by given id public getClothMeasurementById(Long id) : Returns the cloth measurement by given id public addClothMeasurement(ClothMeasurement clothMeasurement) : Adds cloth measurement to the database public updateClothMeasurement(ClothMeasurement clothMeasurement) : Update corresponding cloth measurement public deleteClothMeasurementById(Long id) : Deletes the cloth measurement by given id

Class CameraManager
This class is for handling permissions of clients' cameras, taking and sending videos for body and cloth measurements.
Attributes
private Camera camera private UserManager userManager private ClothManager clothManager
Operations
public getPermission() : Gets the current permission status of the client public givePermission() : Gives camera usage permission for the current client public open() : Opens the camera of the client public close() : Closes the camera of the client public takeVideo() : Records and saves a video for processing public sendVideoForBodyProcessing(Long userId, List<Image> video) : Send video for processing and returns body measurement result for given user id public sendVideoForClothProcessing(Long clothId, List<Image> video) : Send video for processing and returns body measurement result for given cloth id

3.1.2 View

View Class Diagram is as follows. The baseHome and modeling classes, which are specified as starting classes, are used as pages that carry the user to the next steps to make modeling and login or signup. It progresses with connections and directing within themselves, without having any obvious attributes. When the home screen is reached, there are many attributes in the class that are transferred to and taken from other classes. It is not always mentioned here because it enables these attributes to be moved to each other in Navigation React Native.

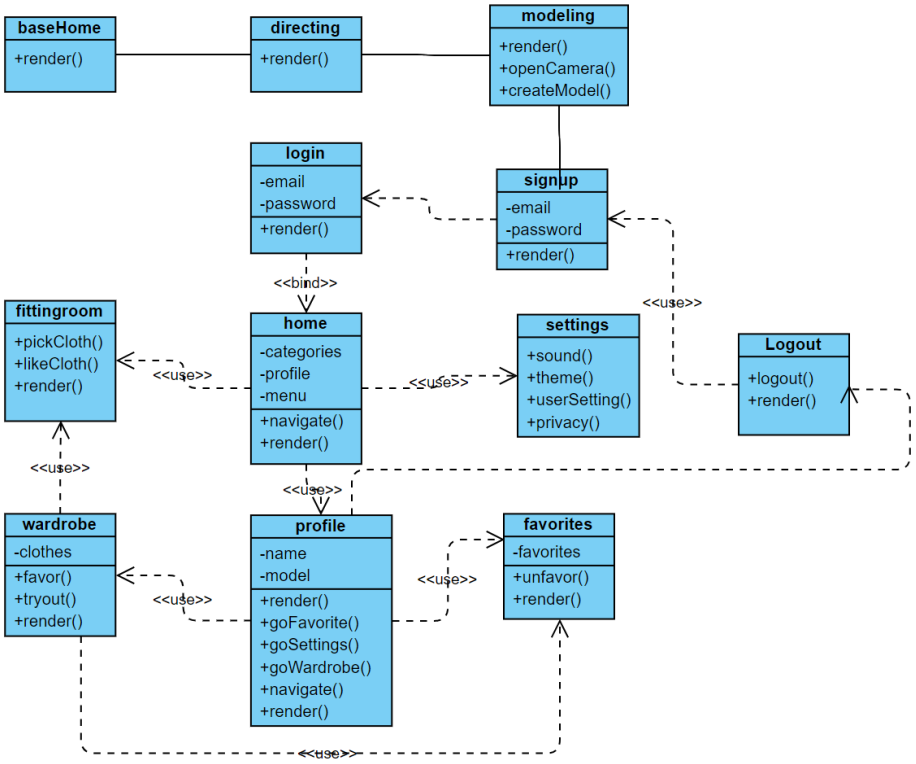


Figure 10. View Class Diagram

Class Sign Up	
This class is for Signing Up to the application	
Attributes	
string Name string Surname string Email	
Operations	
public void Signup()	: Signing up to the application with name, surname
and public int Render()	: Displaying view for the signup page jsx.
public boolean ComponentDidMount()	: Checks if the user signed up.

Class Directing	
This class is for Directing to the modeling screen	
Attributes	
Operations	
public int Render()	: Displaying view for the Directing page jsx.

Class Modeling	
This class is for Modeling the user body.	
Attributes	
Operations	
public boolean OpenCamera() public int Render() public void CreateModel() body.	: Signing up to the application with name, surname and : Displaying view for the modeling page jsx. : creates a model from the image processing of the user

Class BaseHome	
This class is for BaseHome which page welcoming to the application	
Attributes	
Operations	
public int Render()	: Displaying view for the baseHome page jsx.

Class Login	
This class is for Login to the application	
Attributes	
string Name string Surname string Email	
Operations	
public boolean login() public int Render()	:login to the application with name, surname and email. : Displaying view for the login page jsx.

Class Home	
This class is for main page display for the application	
Attributes	
<string> categories Node* Profile <string> menu	
Operations	
public string navigate() public int Render()	: directing users to the other pages. : Displaying view for the home page jsx.

Class Profile	
This class is for User Profile	
Attributes	
string Name Object Model string Setting	
Operations	
public string navigate() public int Render()	: navigating the one page chosen from the user side. : Displaying view for the profile page jsx.

Class FittingRoom	
This class is for tying out the clothes for the user.	
Attributes	
string Name string Surname string Email	
Operations	
public void likeCloth() public int Render()	: when user pushes the button of like for the cloth : Displaying view for the fitting room page jsx.

There are similar functionalities and the attributes that repeat for each class. In react native structure, one attribute can be used in other classes accessing with prop components. So the repeating attributes are not specified in the class interfaces.

3.2 Application

3.2.1 DensePose Logic

Class ImageSegManager
This class segments a given image.
Attributes
private Image image private List<float2d> boxes private List<float2d> segments private List<float2d> key_points private List<float2d> coordinates
Operations
public ImageSegManager ImageSegManager(Image image) : The constructor of ImageSegManager. Constructor gets an image as a parameter and calls private functions of the class to segmentate the image. private List<float2d> calculateSegmentationResults(): This function applies a mask to bounding boxes of the objects. private List<float2d> calculateKeyPoints(): This function infers keypoint poses of the human body using machine learning models. private List<float2d> calculateBodyUV(): This function predicts UV coordinates of the image using machine learning models.

private List<float2d> calculateObjectDetectionBoxes(): This function detects bounding boxes of the objects and returns it. “getter methods are ignored”

Class DPVisualizer
This class calculates visualizable IUV and INDS output of an image.
Attributes
private List<float2d> inds private List<float3d> iuv private ImageSegManager segImage
Operations
public DPVisualizer DPVisualizer(ImageSegManager segImage): The constructor takes an ImageSegManager instance and calculates IUV and INDS output of an image using its key points, bounding boxes, and UV coordinates. It uses private methods of the class. private List<float2d> calculatePatch(): This function calculates third dimension, I, of an IUV output. private List<float2d> calculateINDS(): This function calculates the INDS output of an image. public List<float3d> getIUV(): Getter function for IUV output. public List<float2d> getInds(): Getter function for INDS output.

Class BodyMeshCreator
This class creates the mesh of a human body using IUV output of the DensePose for a human body and saves this mesh to the database in order to be used for size estimations. This is a utility class.
Attributes
private List<float3d> model private List<float3d> iuvOutput
Operations
public MeshCreator MeshCreator(): The constructor of the class. Because this is a utility class, it will be enough to initialize only one instance of TextureTransferer using this constructor. private List<float3d> convertToMesh(DPVisualizer model): This function converts IUV

output of DPVisualizer class into a mesh.

private void addMesh(List<float3d> model): This class saves given body mesh into the database.

public void saveMesh(DPVisualizer model): This class converts IUV output into a mesh and saves it into the database using above functions.

Class TextureTransferer
This class is a utility class which gets the texture of a cloth and maps the texture to the human body using IUV output of DensePose for the human body.
Attributes
private Texture clothTexture private List<float3d> iuvOutput private Image generatedImage
Operations
public TextureTransfer TextureTransfer(): The constructor of the class. Because this is a utility class, it will be enough to initialize only one instance of TextureTransferer using this constructor. public Image transferTexture(DPVisualizer model, Cloth cloth): This method maps a cloth texture to the human body using IUV output of DPVisualizer class.

3.2.2 Size Estimation Manager

Class Estimation
This class is responsible for storing information about a size estimation.
Attributes
private List<Cloth> clothes private ClothMeasurement clothMeasurement private BodyMeasurement bodyMeasurement private Favorites favoriteClothes private User user
Operations
public Estimation() : Constructor class for Estimation “Getter and Setter methods are ignored ”

Class SizeEstimator
This class is responsible for the size estimation process.
Attributes
private Estimation estimation private List<Clothes> clothes
Operations
public SizeEstimator(Estimation e) : Constructor class for Size Estimator public estimateSize(Estimation e) : Estimates the most convenient size for user private trainEstimation(Estimation e) : Trains the ML model of estimation. private updateEstimation(Estimation e) : Updates estimation according to changes. private checkEstimation(Estimation e) : Checks if the estimation is valid or not. private getEstimationAccRate() : Returns accuracy of the estimation. private refresh(Estimation e) : Resets an estimation. “Getter and Setter methods are ignored ”

3.2.3 Cloth Model Manager

Class ClothModelManager
This class is responsible for constructing new cloth models for the application.
Attributes
private List<Image> clothImage private Mesh clothMesh private ClothRepository repository
Operations
public ClothModelManager : Constructor class public createClothModel(List<Image> images, Mesh clothMesh) : This function extracts texture from a given cloth image according to the given mesh. private addClothModel(ClothModel clothmodel) : This function saves cloth models to the database. It is private because it is called inside the createClothModel() function. “Getter and Setter methods are ignored ”

3.3 Data

Class User
This class is responsible for storing information about the user.
Attributes
private Long id private Favorites favorites private UserModel model private String userType private String password private String username private String email private String name private String surname private BodyMeasurement bodyMeasurements
Operations
public User() : Constructor class for User “Getter and Setter methods are ignored ”

Class BodyMeasurement
This class is responsible for storing necessary information about the body measurements of the user.
Attributes
private String size private Float chestSize private Pair<Float, Float> armWidth private Float abdomenSize private Float waistSize private Float hipsSize private Pair<Float, Float> thighSize private Float neckSize private Float inseamLength private Float outseamLength private Pair<Float, Float> sleeveLength private Float ankleSize
Operations
public BodyMeasurement() : Constructor class for BodyMeasurement. “Getter and Setter methods are ignored ”

Class UserModel
This class is responsible for storing the 3D model of the body of the user.
Attributes
private Long modelId private List<float3d> bodyMesh private Texture texture
Operations
public UserModel() : Constructor class for UserModel. “Getter and Setter methods are ignored ”

Class Favorites
This class is responsible for storing favourite clothes of a user.
Attributes
private User user private List<Cloth> clothes

Operations
public Favorites() : Constructor class for Favorites. “Getter and Setter methods are ignored ”

Class Cloth
This class is responsible for storing information about cloth.
Attributes
private Long id private ClothMeasurement measurement private ClothModel model private String clothType private List<Image> image
Operations
public Cloth() : Constructor class for Cloth “Getter and Setter methods are ignored ”

Class ClothMeasurement
This class is responsible for storing necessary size information about cloth.
Attributes
private String size private Float neckSize private Float sleeveLength private Float sleeveWidth private Float chestLength private Float waistLength private Float backLength private Float outerLegLength private Float innerLegLength private Float hipCircumference private Float thighCircumference
Operations
public ClothMeasurement() : Constructor class for ClothMeasurement “Getter and Setter methods are ignored”

Class ClothModel
This class is responsible for storing 3D model data of a cloth.
Attributes
private Long modelId private List<float3> clothMesh private Texture texture private String label
Operations
public ClothModel() : Constructor class for ClothModel “Getter and Setter methods are ignored”

4 Glossary

Component	Component is an object structure used in React Native. The functions in the components are combined with data and view and transmitted to the html side as jsx code and displayed on the screen. Components can include arrays, nodes, and objects. It is a React Native structure that includes the state and prop structure.
Router	Router structure is a container structure that keeps track of data transferred between pages and stats. It also provides a communication between Client and Server.
Jupyter	Jupyter is an open-source application that developers can create and share a notebook including live code, documents, visual attachments etc.
Dense Pose	Dense Pose is Facebook's real-time human pose estimation approach by 2D RGB images.
Redux	Redux structure is a structure that provides server connection by combining with Router in React Native. By connecting structures such as Firebase and SQL to the application, it provides an internal consistency.

5 References

[1] "COVID-19 has changed online shopping forever, survey shows," *Unctad.org*. [Online]. Available: https://unctad.org/news/covid-19-has-changed-online-shopping-forever-survey-shows. [Accessed: 21-Nov-2020].

[2] Cloud & Data Center Platform: Sparkle. (n.d.). Retrieved December 27, 2020, from https://www.tisparkle.com/our-platform/cloud-data-center-platform

[3] What is a REST API? (n.d.). Retrieved December 27, 2020, from https://www.redhat.com/en/topics/api/what-is-a-rest-api

[4] Read, 3. (n.d.). What is a denial of service attack (DoS) ? Retrieved December 27, 2020, from https://www.paloaltonetworks.com/cyberpedia/what-is-a-denial-of-service-attack-dos

[5] "Learning React Native," *O'Reilly Online Learning*. [Online]. Available: https://www.oreilly.com/library/view/learning-react-native/9781491929049/ch01.html. [Accessed: 05-Feb-2021].