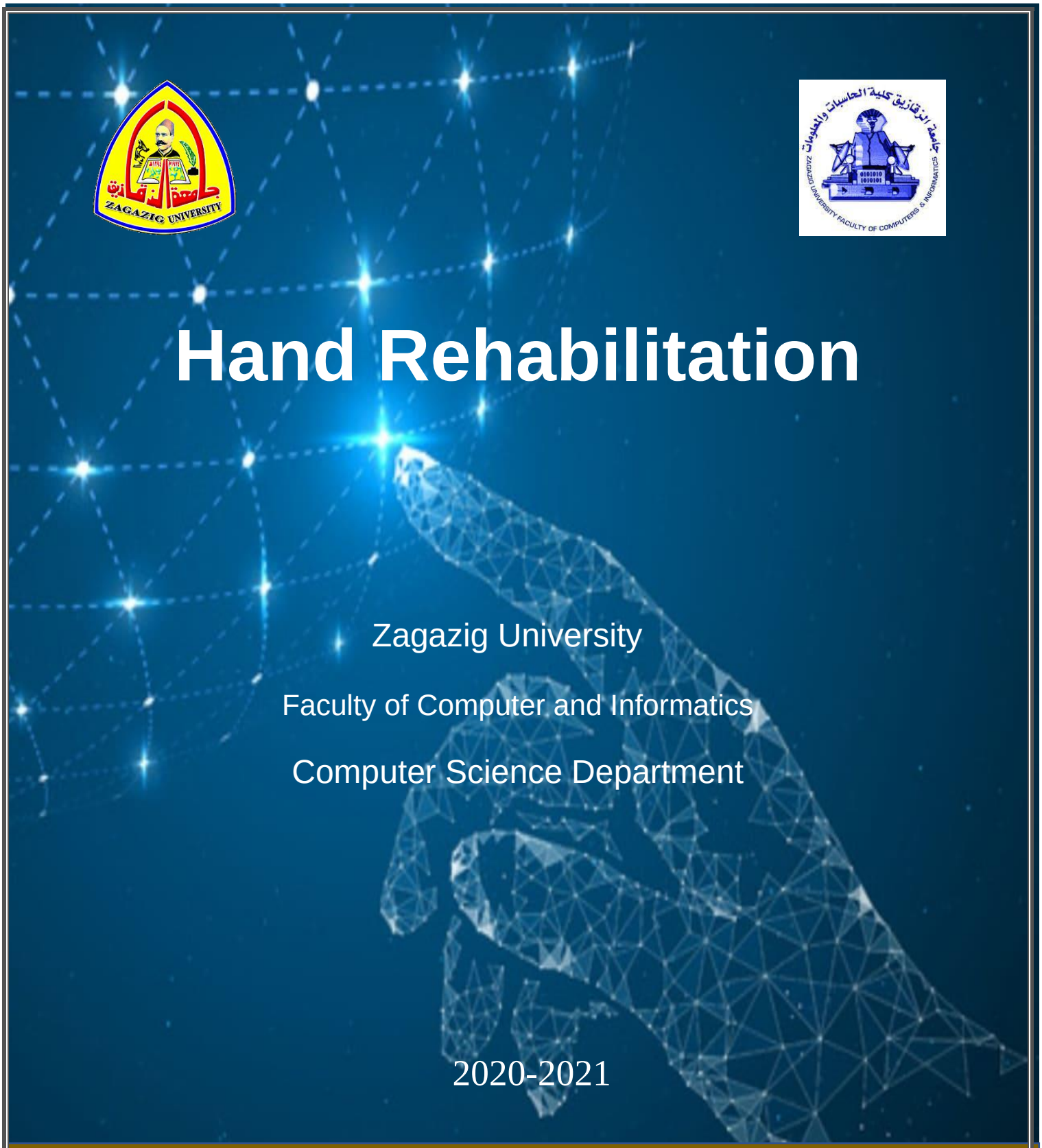


Hand Rehabilitation

Zagazig University

Faculty of Computer and Informatics

Computer Science Department

2020-2021

Supervisor

Prof.Dr. / Ahmed Salah.

Hand Rehabilitation

Professor Supervisor

Dr. / Ahmed Salah.

Assistant Supervisor

Eng. /Asma Abdelkader.

Computer Science Department

Graduation Project

2020 - 2021

Our team

- Ahmed Atef Abdelmonim Mohammed.
- Abdelaziz Elsayed Abdelaziz Eiwisha.
- Abdeladl Mounir Abdeladl Shaheen.
- Abdel Rahman mohamed mohamed shata.
- Eslam Khaled Reda Abo Alnaga.
- Gehad Abdelaziz Omran Amer.

Acknowledgments

First and foremost, praises and thanks to the God, the Almighty, for His showers of blessings throughout our work to complete the project successfully.

We would like to express our deep and sincere gratitude to our project supervisor, **Dr/ Ahmed Salah**, for giving us the opportunity to do this project under his supervision and providing invaluable guidance. His vision, sincerity and motivation have deeply inspired us. He has taught us the methodology to carry out the project and to present the works as clearly as possible. It was a great privilege and honor to work and study under his guidance. We are extremely grateful for what he has offered us.

Also, we would like to express our sincere gratitude to **Eng/ Asmaa Abdelkader** for the continuous support of our project, for her patience and motivation. Her guidance helped us in all the times of project.

Thanks also attached our team members for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had during doing the project.

Last but not the least, we are extremely grateful to our families for their love, prayers, caring and sacrifices for educating and preparing us for our futures and for supporting us spiritually throughout our lives.

Our thanks go to all the people who have supported us to complete the project directly or indirectly.

Abstract

"Hand Rehabilitation" is a mobile application to help doctors and patients for hand diagnosis using computer vision and machine learning. It is known that if a person is severely injured in his hands, he immediately goes to the doctor, the doctor then decides to perform a surgery to overcome that injury. Usually, if the hand is still sore and does not move well and normally, the doctor directs the patient to go and receive some physical therapy until his hands improve and return to the normal position they were in with their normal movements. The role of the physiotherapist in diagnosing the patient revolves around measuring the angles between the fingers of the patient's hand each time. The patient's diagnosis has gone through several stages. At first, the doctor used to measure these angles in a traditional way, by drawing the patient's hand on a sheet and starting to identify the lines on each finger, and then measuring the angle between them using the transferrer. The devices used began to evolve gradually, and later began using a device called the Manual Unitometer, which was used by placing it between two fingers

and selecting the angle directly, but it should be noted that the disadvantages of this device, in addition to the traditional method used, are that they consume very long time and little accuracy, which requires further development in this area. Recently, doctors have been using a device called a "digital gunmeter scale" that emits two laser-like beams on two fingers and calculates the angle directly through them.

Learned Skills

Skills Gained from This Project

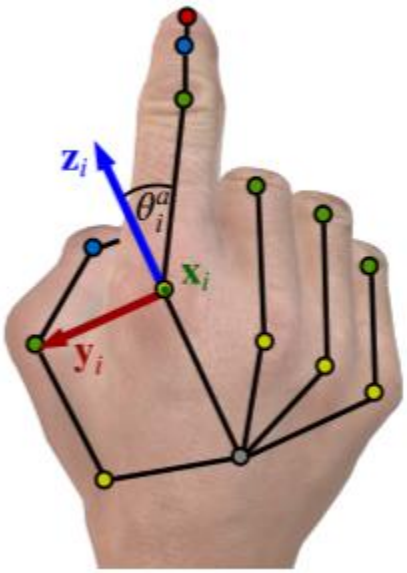
- How to collect information about the problem.
- How to analyze information.
- How to select sufficient information.
- How to solve the problem
- How to deal with external environment.
- Deal with new technologies.
- Increase our programming skills specially in machine learning.
- How to read international research and add your enhancement notes.
- How to be a member of teamwork, and we learned the experience of co-operation
- How to search about a specific information, tool, technique, or technology that may help us in achieving our project by reading books, exploring the internet, or asking anyone.

Table of Contents:

List of Figures:

Chapter One

Introduction



1-Introduction

The hand is the primary operating tool for humans. Therefore, its location, orientation and articulation in space is vital for many potential applications, for instance, object handover in robotics, learning from demonstration, sign language and gesture recognition, and using the hand as an input device for man-machine interaction. Full 3D hand pose estimation from single images is difficult because of many ambiguities, strong articulation, and heavy self-occlusion, even more so than for the overall human body.

1.1- The main problem that we are solving: -

Fingers and Hand pose estimation following the rehabilitation of hand nerve injuries from images or videos.

After surgery of hand the therapist want to discover the evolution of surgery, then therapist measure the angle between fingers.

1.2- The importance of this problem: -

The estimation of human pose and orientation opens the door to solving various real-life problems. In the following, we outline the importance of the addressed problems.

1.2-1. Fingers and hand pose estimation:

After peripheral nerve injuries, a follow-up is needed to estimate the nerve's degree of recovery. In hand and fingers injuries, the follow-up can be done by measuring the angles between spread fingers. The proposed method aims to automate the follow-up process by detecting the degree of recovery only from the image of the spread hand pose. Thus, the patient can do the follow-up from home using her/his mobile phone only.

1.3- Current solutions: -

The estimation is done manually by drawing lines around the hand on a paper and measure the angles or using special devices.

- Manual goniometer (Universal) as in figure 1.
- Digital goniometer as in figure 2.

❖ Disadvantages of Current Solutions:

- The current methods are manual, not very accurate, and slightly complex.
- The digital goniometer device is very expensive.

1.4- Proposed solutions: -

- We propose utilizing AI-based methods of pose estimation to address peripheral in mobile application.
- we will use a mobile application to capture an image of the hand pose and then analyze the image using AI-based pose estimation methods over time (i.e., several days) for the sake of reporting the recovery progress of finger nerves. As the degrees between two fingers increases, the recovery increases.

In our solution we will use:

1.5- Architecture: –

1. Mobile phone to capture the image.
2. A Server to carry out the pose estimation and machine learning analysis.

1.6- Integration: -

We develop a mobile phone application and a desktop application for collecting data (images). Then, this application will send the images to the server to carry out back-end computations using pose estimation and machine learning algorithm

1.7- Innovative components: -

We will integrate angles measuring to AI-based pose estimation methods for the problems.

1.8- development environment: -

- we use Anaconda and Jupiter notebook in addition to several Python libraries such as (TensorFlow, Pytorch, Pandas, and Numpy).
- Tools needed for our solutions are cameras attached to a mobile phone.

Chapter Two

Analysis

2.1 Project Requirements: -

The functional and non-functional requirements is the best thing that any project must have in order not to be doomed to failure. Any project's requirements need to be well thought out, balanced, and clearly understood by all involved, but perhaps of most importance is that they are not dropped or compromised halfway through the project.

2.1.1 Functional requirement:

Functional requirements will specify a behavior or function.

1. Register.

   The app allows the doctor and the Patient to log in.

2. Create a therapeutic file.

   The application allows the doctor to create a therapeutic file for the patient.

3. Take or Upload a photo.

   The app allows the doctor and the Patient to take a Photo or Upload a Photo from gallery.

4. Show results.

   The doctor and the patient can see the results immediately.

5. Save the Photo.

   The application allows the doctor and the patient to save the Photo in the patient's therapeutic file.

6. Search

The doctor can search in patients' records.

7. Open therapeutic file.

The application allows the doctor and the patient to open the therapeutic file of the patient and see his pictures.

2.1.2 Qualities, non-functional requirements can be divided into 4 main categories:

- Operational

  It can work on a mobile phone that support the android operating system.

  The mobile phone must connect to internet (Wi-Fi OR Data).

- Performance

  The app opens the camera by simply pressing the button (Speed).

  The app saves images in databases.

  Results appear by pressing the button (Speed).

  The app available for use 24 hours per day.

- Security

  Only the direct doctor and the patient can see the patient's therapeutic file.

  The app will ensure the protection of patient data.

- Cultural & political

  Protect the patient's personal information in accordance with the Data Protection Act.

  The app supports several different languages.

2.2 Use case.

A use case diagram is a dynamic or behavior diagram in UML. That is a set of actions, services, and functions that the system needs to perform. Use case diagrams specify how the system interacts with actors without worrying about the details of how that functionality is implemented.

- ### Actors.

  An actor is a person, organization, or external system that plays a role in one or more interactions with your system. Actors are drawn as stick figures.

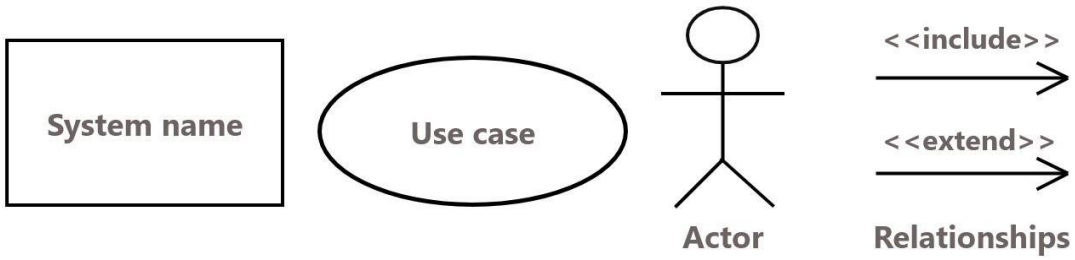
- ### Association.

  Associations between actors and use cases are indicated in use case diagrams by solid lines. An association exists whenever an actor is involved with an interaction described by a use case. Associations are modeled as lines connecting use cases and actors to one another, with an optional arrowhead on one end of the line. The arrowhead is often used to indicating the direction of the initial invocation of the relationship or to indicate the primary actor within the use case. The arrowheads are typically confused with data flow and as a result I avoid their use.

- System boundary boxes (optional).

You can draw a rectangle around the use cases, called the system boundary box, to indicate the scope of your system. Anything within the box represents functionality that is in scope and anything outside the box is not. System boundary boxes are rarely used, although on occasion I have used them to identify which use cases will be delivered in each major release of a system.

2.2.1 Use case diagram Component.



2.2.2 Use case diagram.

Figure 2.1

2.2.3 Use case tables.

Use case name: Login to app	**ID:** UC-1	**Priority:** high
Actor: doctor		
Description: This use case describes the scenario when doctor will be prompted to login with his account information before he can use the app.		
Trigger: User requests to login **Type:** External		
Preconditions: 1. The user has account. 2. The user is trying to log in with his account. 3. The user is already logged in		
Normal course: 1.0 User accesses the URL. 1. The system prompts the user for his account credentials. 2. The user enters his username and password. 3. The system authenticates the login. 4. The user gains access to the systems functionality		
Postconditions: 1. The user is logged in to the system. 2. The user has access to the functions of the system		
Exceptions: E1: Incorrect credentials 1. ensure the user have account on app. 2. if the user has not account, he will register.		

USE CASE NAME: Take a photo	ID: UC-2	PRIORITY: High

ACTOR: Doctor

DESCRIPTION: This use case describes the scenario when the doctor wants to take a photo/picture from the patient by opening the camera of the application.

TRIGGER: The scenario starts when the doctor opens the camera of the application.

PRECONDITIONS:

1- The doctor must login to the application.
2- The doctor must create a therapeutic file for the patient.
3- The doctor must open the camera of the application.

NORMAL COURSE:

1.0 Take a photo from the patient.

1- System asks for the preferred input method.
2- The doctor wants to take a photo from the patient.
3- The doctor takes the photo and gives permission to access photo system.
4- System uploads the photo and checks whether it is a valid photo file, whether it has size and dimension under limit.
5- The system is doing its operations to get the result**.**

ALTERNATIVE COURSE:

1.1 **Upload** a photo from the gallery.

1- System asks for the preferred input method.
2- The doctor wants to upload a photo from the gallery.
3- System asks for the path of the photo file.
4- The doctor selects the photo file and gives permission to access file system.
5- System uploads the photo and checks whether it is a valid photo file, whether it has size and dimension under limit.
6- The system is doing its operations to get the result.

POSTCONDITIONS:

When the doctor takes / upload the photo, The system is doing its operations to get the result and save it in a therapeutic file of the patient.

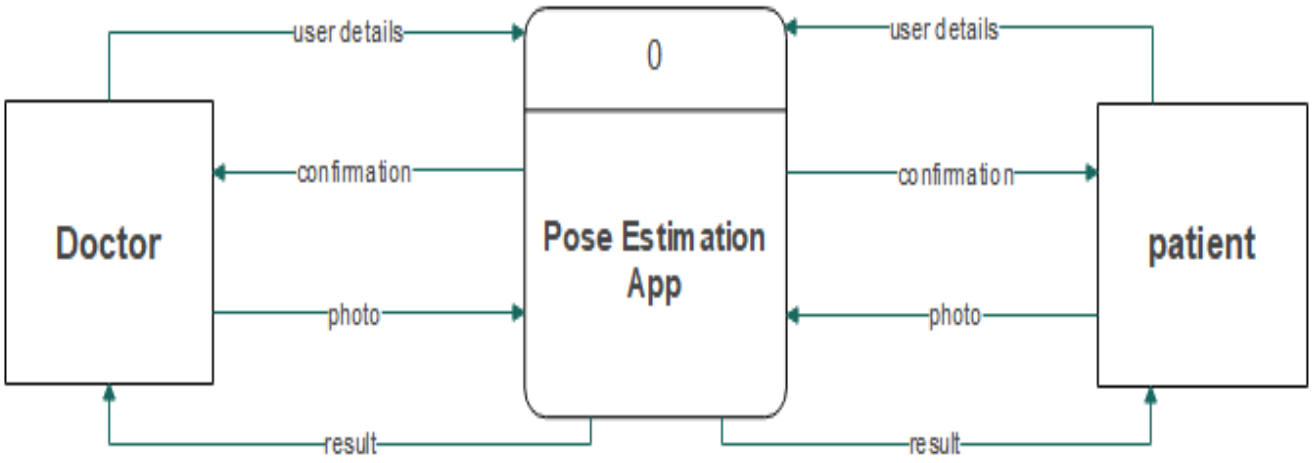
Business rules

- JPG or GIF images are supported only.
- Max image size 256 KB
- Max dimension 800X600
- Image shouldn't contain offensive materials.
- Image shouldn't be a copyrighted image.

2.3 Context Diagram: -

The context diagram defines how the business process or computer system interacts with its environment.

It is a basic overview of the whole modeled. It is designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts and developers. System or

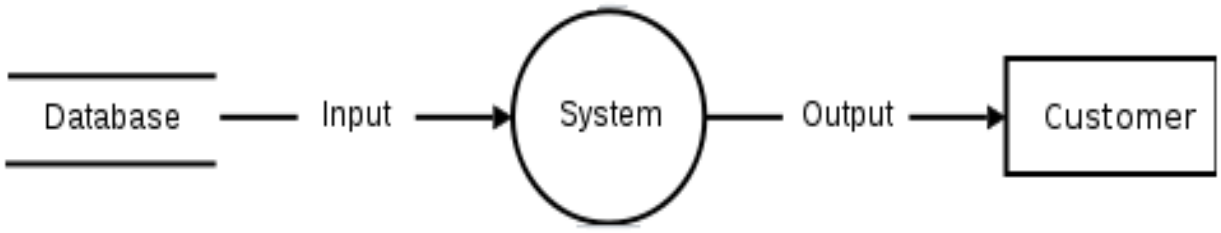


process being analyzed or

Figure 2.2

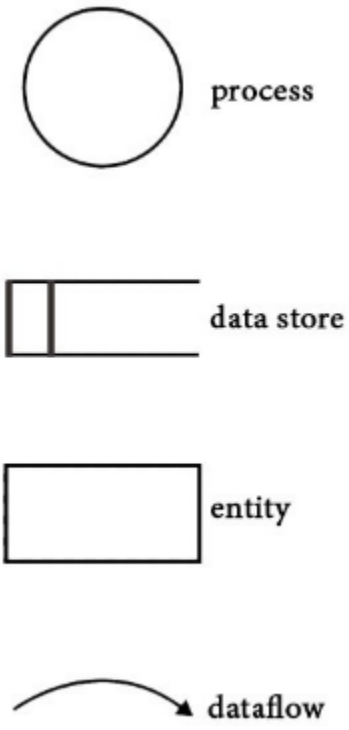
2.4 A data-flow diagram (DFD):

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually "say" things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO. That is why DFDs remain so popular after all these years. While they work well for data flow software and systems, they are less applicable nowadays to visualizing interactive, real-time, or database-oriented software or systems.



DFD with data storage, data flows.

2.4.1 A data-flow diagram Components:



the symbols depict the four components of data flow diagrams.

1. External entity: an outside system that sends or receives data, communicating with the system being diagrammed. They are the sources and destinations of information entering or leaving the system. They might be an outside organization or person, a computer system or a business system. They are also known as terminators, sources and sinks or actors. They are typically drawn on the edges of the diagram.

2. Process: any process that changes the data, producing an output. It might perform computations, or sort data based on logic, or direct the data flow based on business rules. A short label is used to describe the process.

3. Data store: files or repositories that hold information for later use, such as a database table or a membership form.
4. Data flow: the route that data takes between the external entities, processes, and data stores. It portrays the interface between the other components and is shown with arrows, typically labeled with a short data name.

" A data flow diagram can divide into progressively more detail by using levels and layers, zeroing in on a particular piece. DFD levels are numbered 0, 1 or 2, and occasionally go .to even Level 3 or beyond. The necessary level of detail depends on the scope of what you are trying to accomplish."

- DFD Level 0 provides a more detailed breakout of pieces of the Context Level Diagram. You will highlight the main functions carried out by the system, as you break down the high-level process of the Context Diagram into its sub processes.

- DFD Level 1 then goes one step deeper into parts of Level 1. It may require more text to reach the necessary level of detail about the system's functioning.

2.4.2 Data Flow Diagram Level 0:

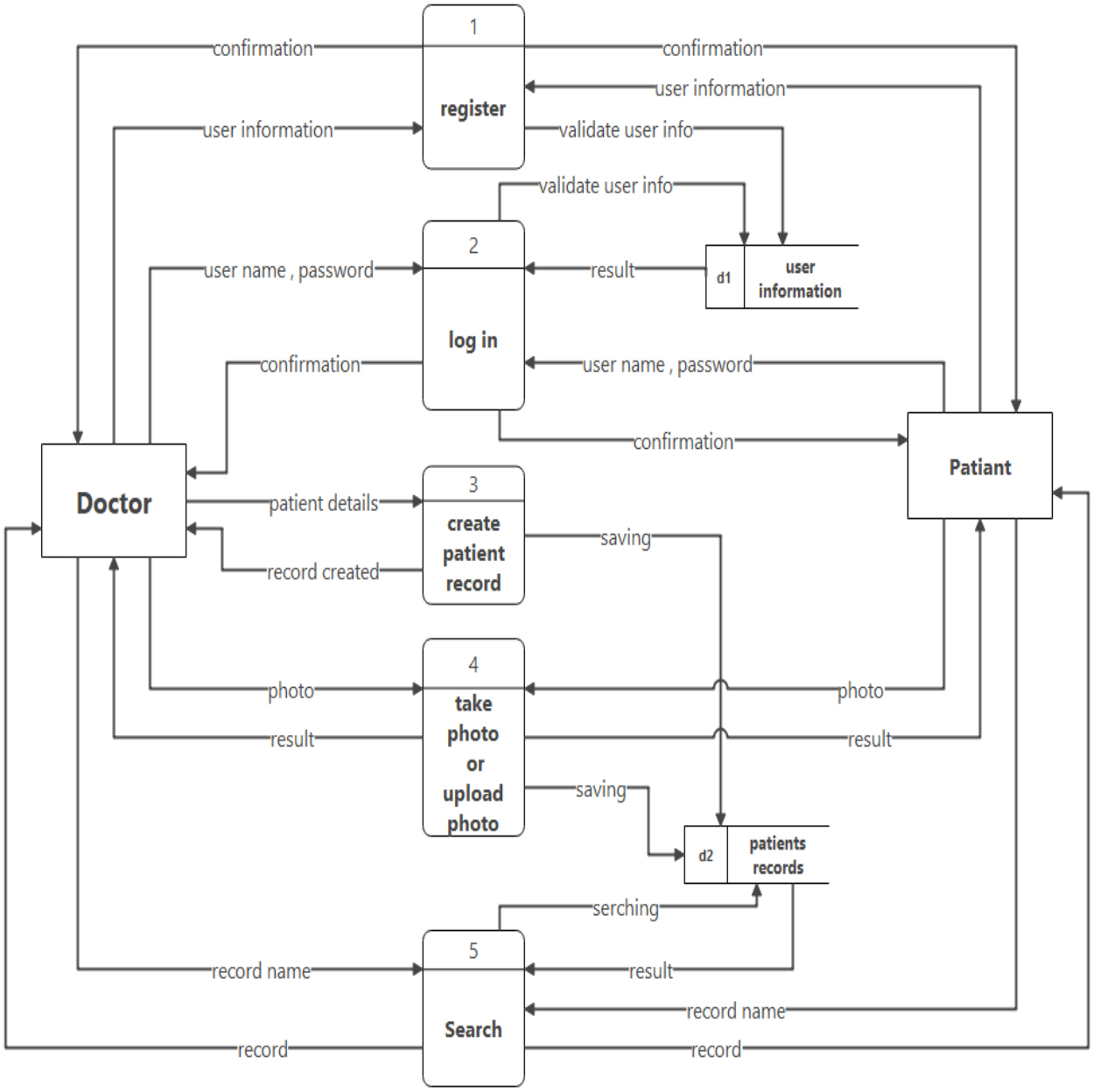


Figure 2.3

2.4.3 Data Flow Diagram Level 1:

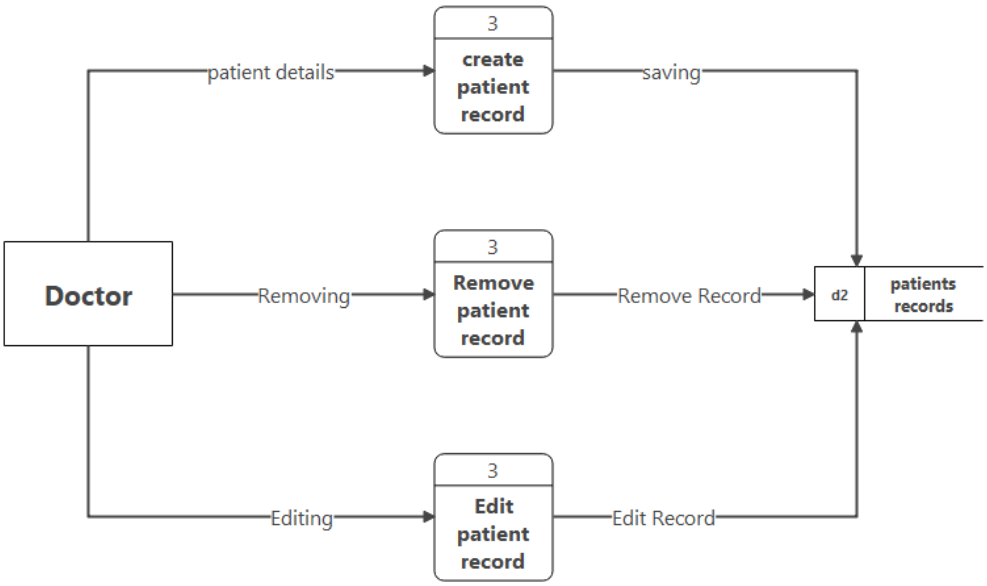


Figure 2.4

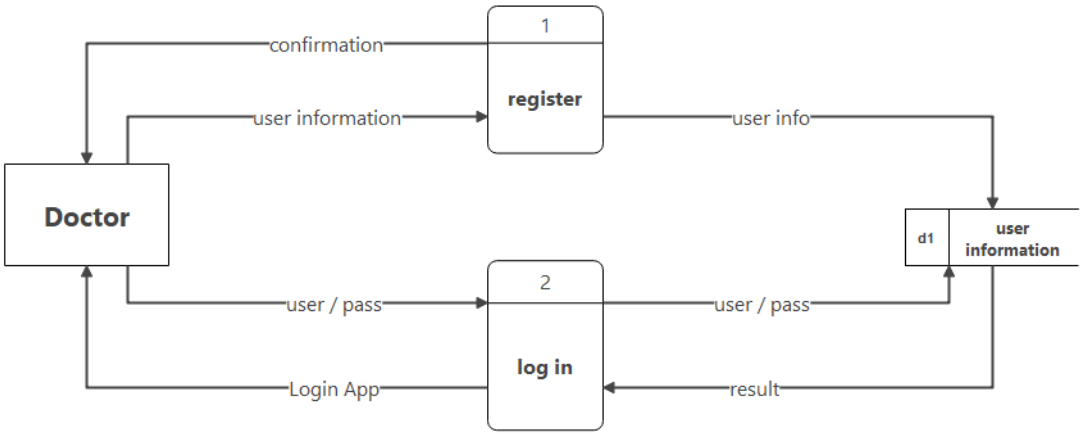


Figure 2.5

2.5 Entity Relationship Diagram:

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define its properties. By defining the entities, their attributes, and showing the relationships between them, an ER diagram illustrates the logical structure of databases. ER diagrams are used to sketch out the design of a database.

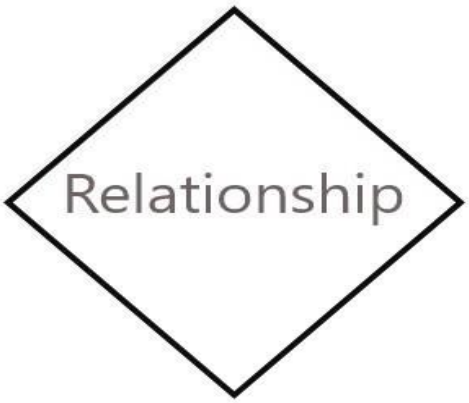
2.5.1 Entity Relationship Diagram Symbols:

There are five main components of an ERD:

Entity: - An entity can be a person, place, event, or object that is relevant to a given system. For example, a school system may include students, teachers, major courses, subjects, fees, and other items. Entities are represented in ER diagrams by a rectangle and named using singular nouns.

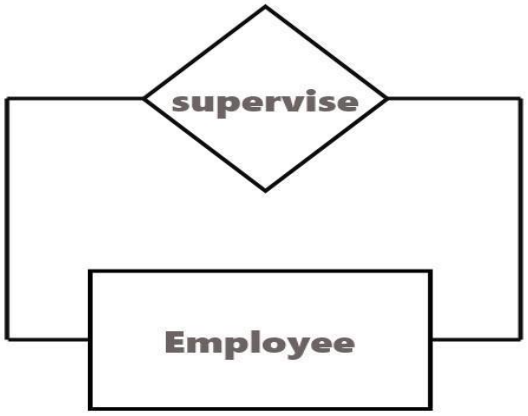
Weak Entity: - A weak entity is an entity that depends on the existence of another entity. In more technical terms it can be defined as an entity that cannot be identified by its own attributes. It uses a foreign key combined with its attributed to form the primary key. An entity like order item is a good example for this. The order item will be meaningless without an order so it depends on the existence of the order.

Actions, which are represented by diamond shapes, show how two entities share information in the database.



Action's symbol.

In some cases, entities can be self-linked. For example, employees can supervise other employees.



Entities can be self-linked.

Attribute: - An attribute is a property, trait, or characteristic of an entity, relationship, or another attribute. For example, the attribute Inventory Item Name is an attribute of the entity Inventory Item. An entity can have as many attributes as necessary. Meanwhile, attributes can also have their own specific attributes. For example, the attribute "customer address" can have the attributes number, street, city, and state. These are called composite attributes. Note that some top-level ER diagrams do not show attributes for the sake of simplicity. In those that do; however, attributes are represented by oval shapes.

Attribute symbol

A multivalued attribute can have more than one value. For example, An employee entity can have multiple skill values.

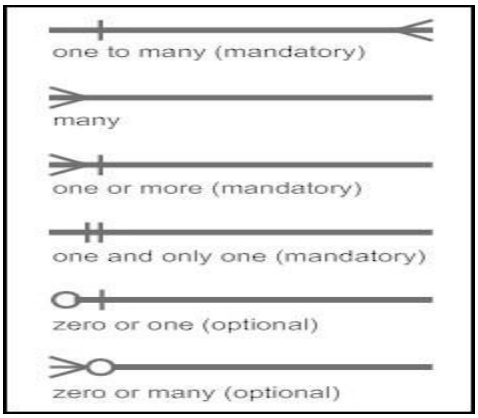
Multivalued attribute symbol.

cardinality: the cardinality of a relationship is the number of instances of one entity type that may be associated with each instance of the other entity type.

One to one (1-1): one instance in an entity (parent) refers to one and only one instance in the related entity (child).

One to many (1-m): one instance in an entity (parent) refers to one or more instances in the related entity (child).

Many to many (m-n): exists when one instance of the first entity (parent) Can relate to many instances of the second entity (child), and one instance of the second entity can relate to many instances of the first entity.



How to Draw ER Diagrams

Below points show how to go about creating an ER diagram.

1. Identify all the entities in the system. An entity should appear only once in a particular diagram. Create rectangles for all entities and name them properly.

2. Identify relationships between entities. Connect them using a line and add a diamond in the middle describing the relationship.

3. Add attributes for entities. Give meaningful attribute names so they can be understood easily.

Benefits of ER diagrams

ER diagrams constitute a very useful framework for creating and manipulating databases. First, ER diagrams are easy to understand and do not require a person to undergo extensive training to be able to work with it efficiently and accurately. This means that designers can use ER diagrams to easily communicate with developers, customers, and end users, regardless of their IT proficiency. Second, ER diagrams are readily translatable into relational tables which can be used to quickly build databases. In addition, ER diagrams can directly be used by database developers as the blueprint for implementing data in specific software applications. Lastly, ER diagrams may be applied in other contexts such as describing the different relationships and operations within an organization.

2.5.2 ERD of the project:

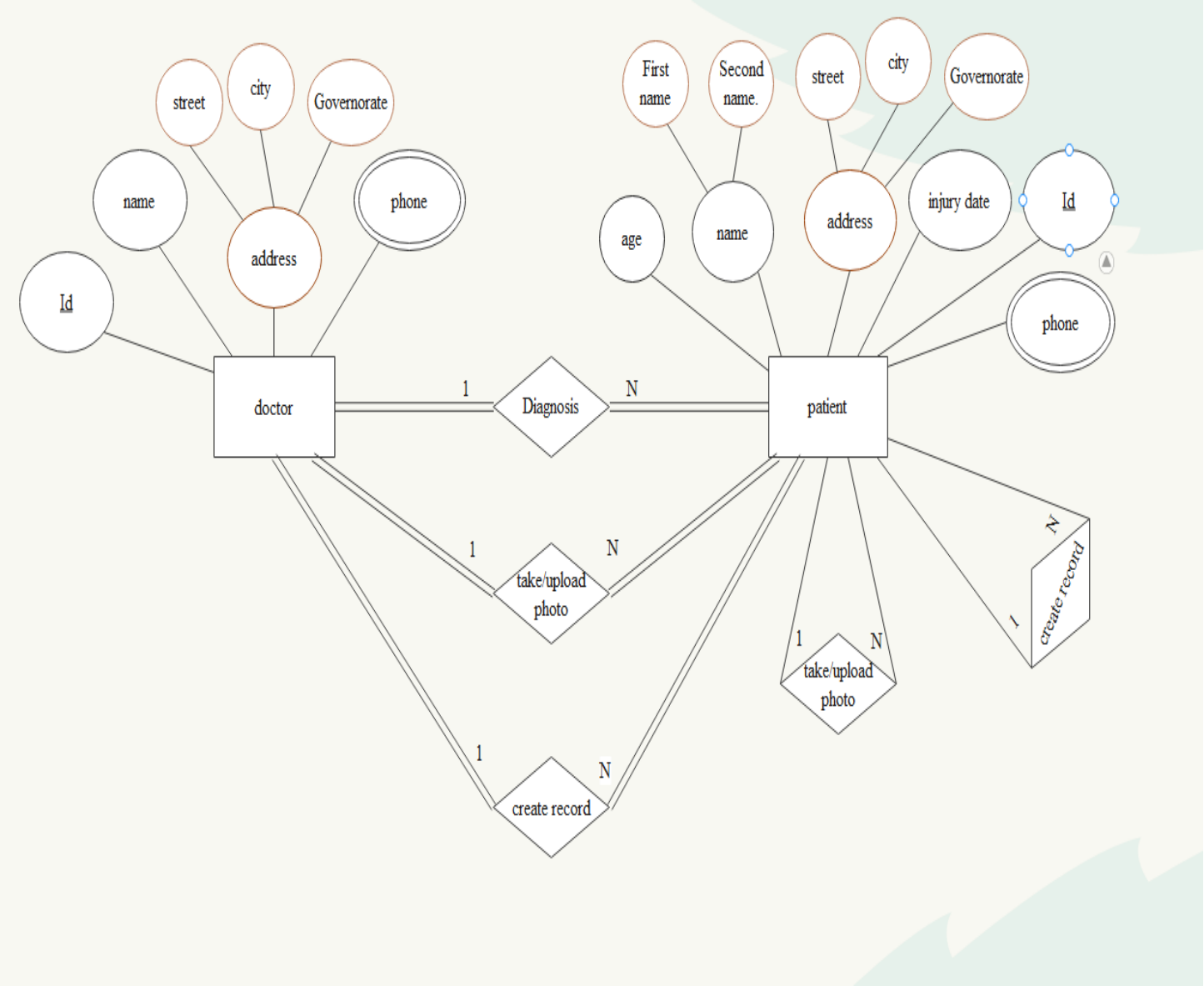
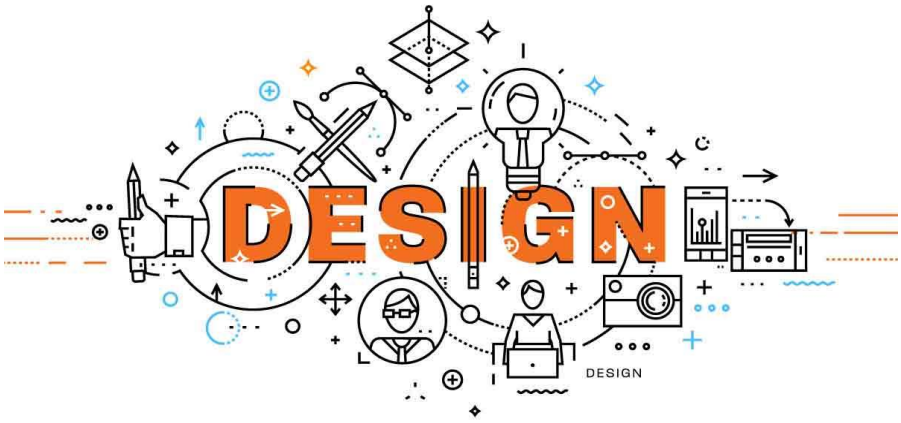


Figure 2.6

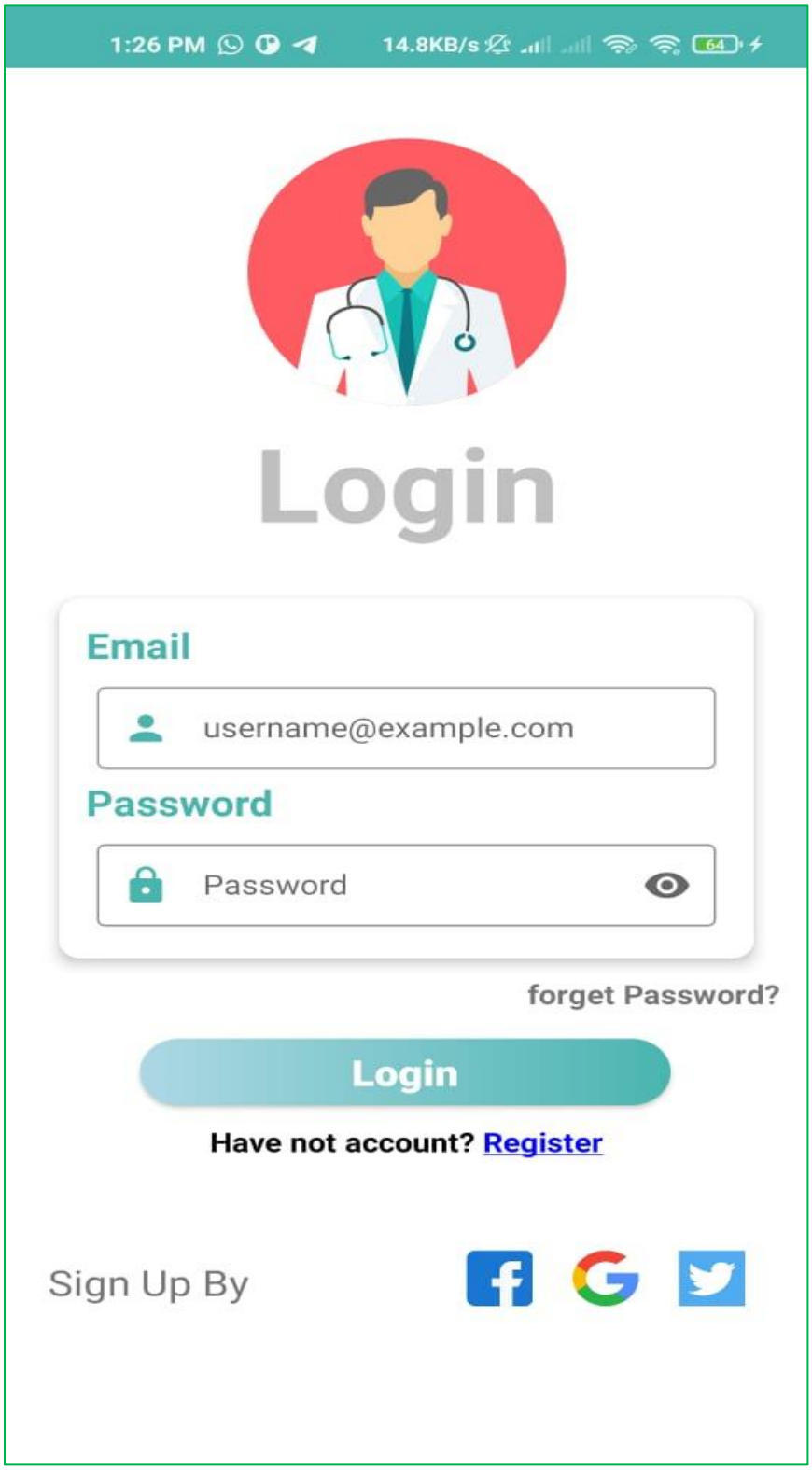
Chapter three

Design

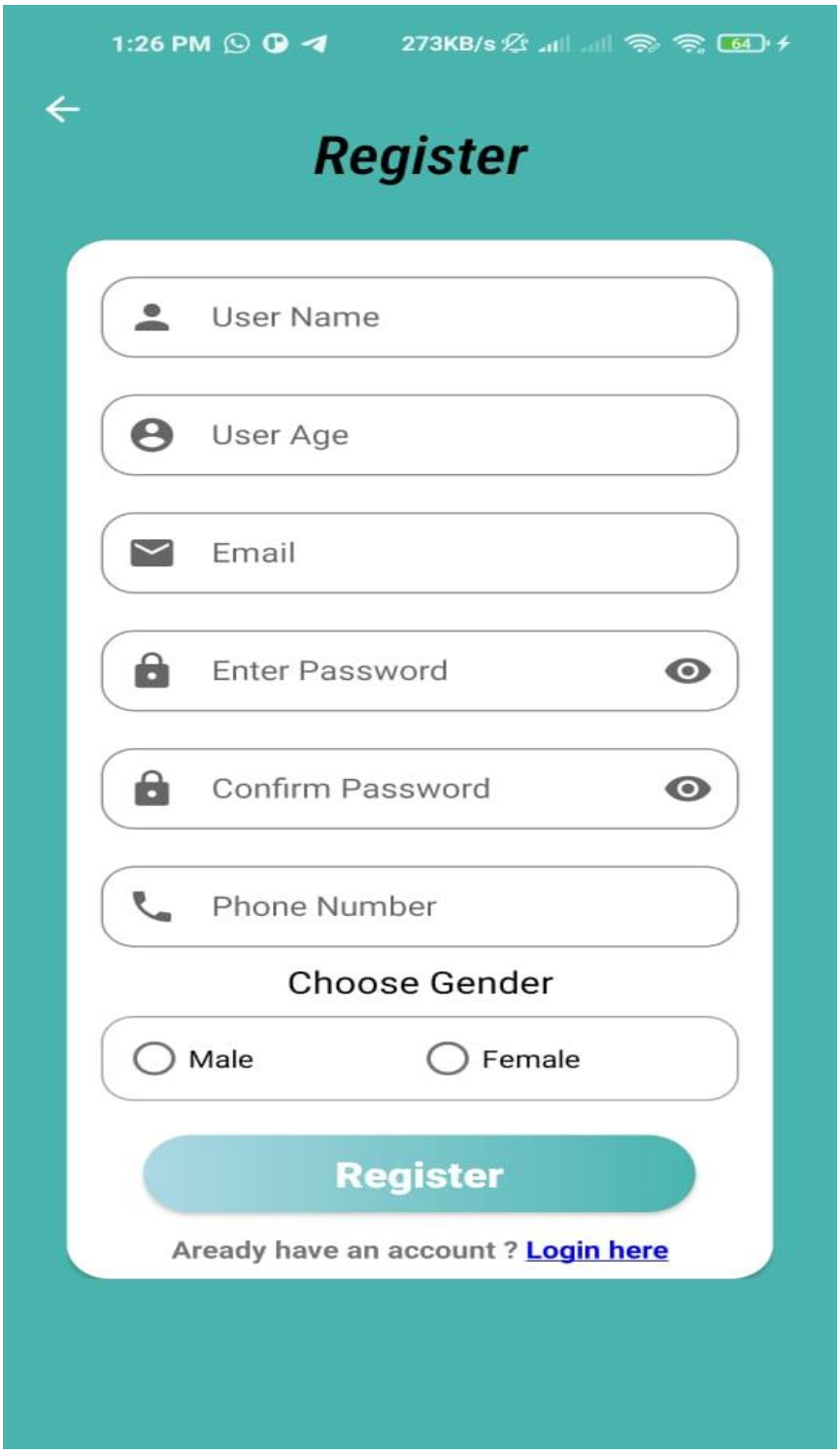


3.1- User type selection page

3.2- Login Page

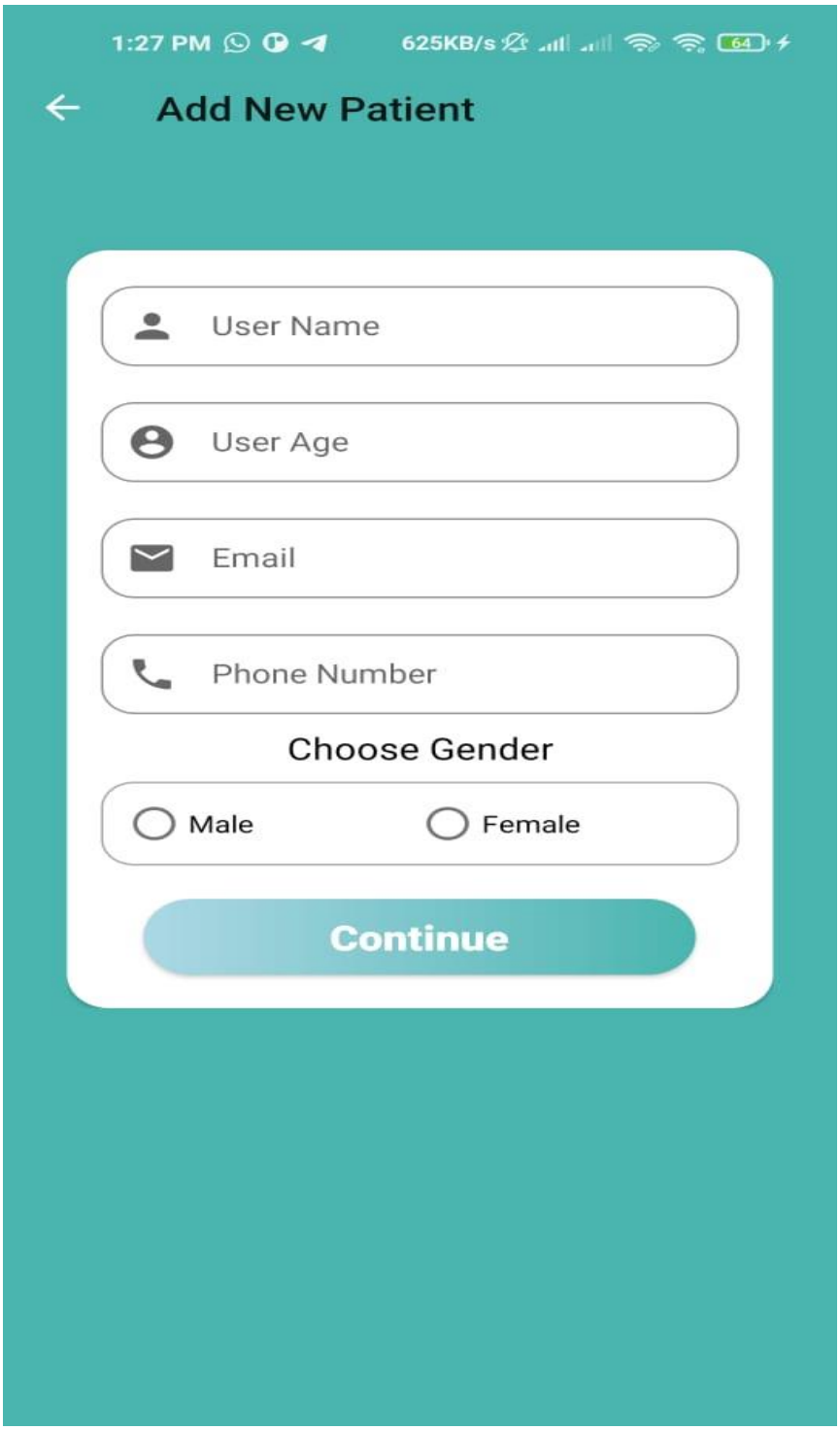


3.3- Register Page

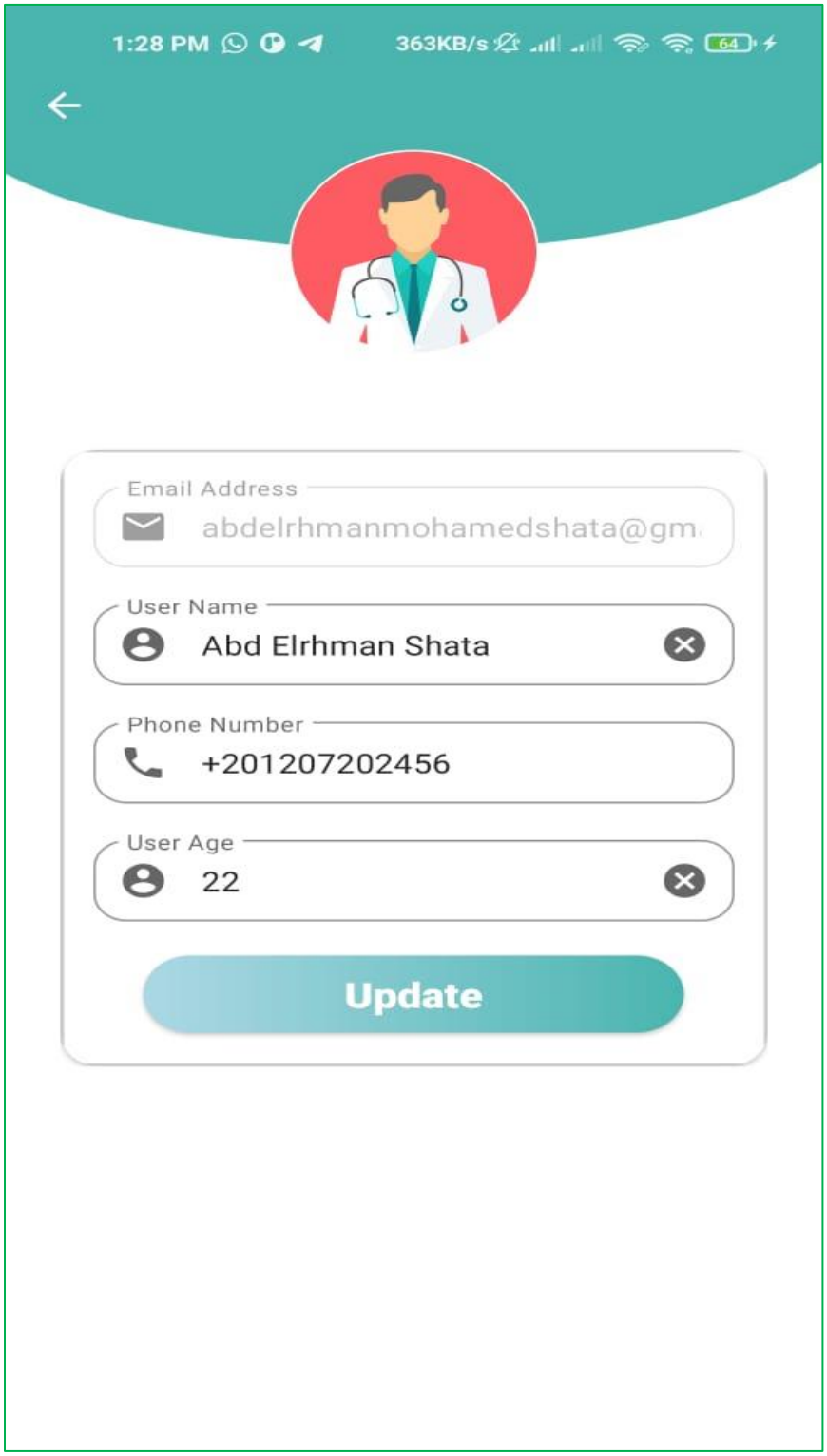


3.4- Doctor Home Page.

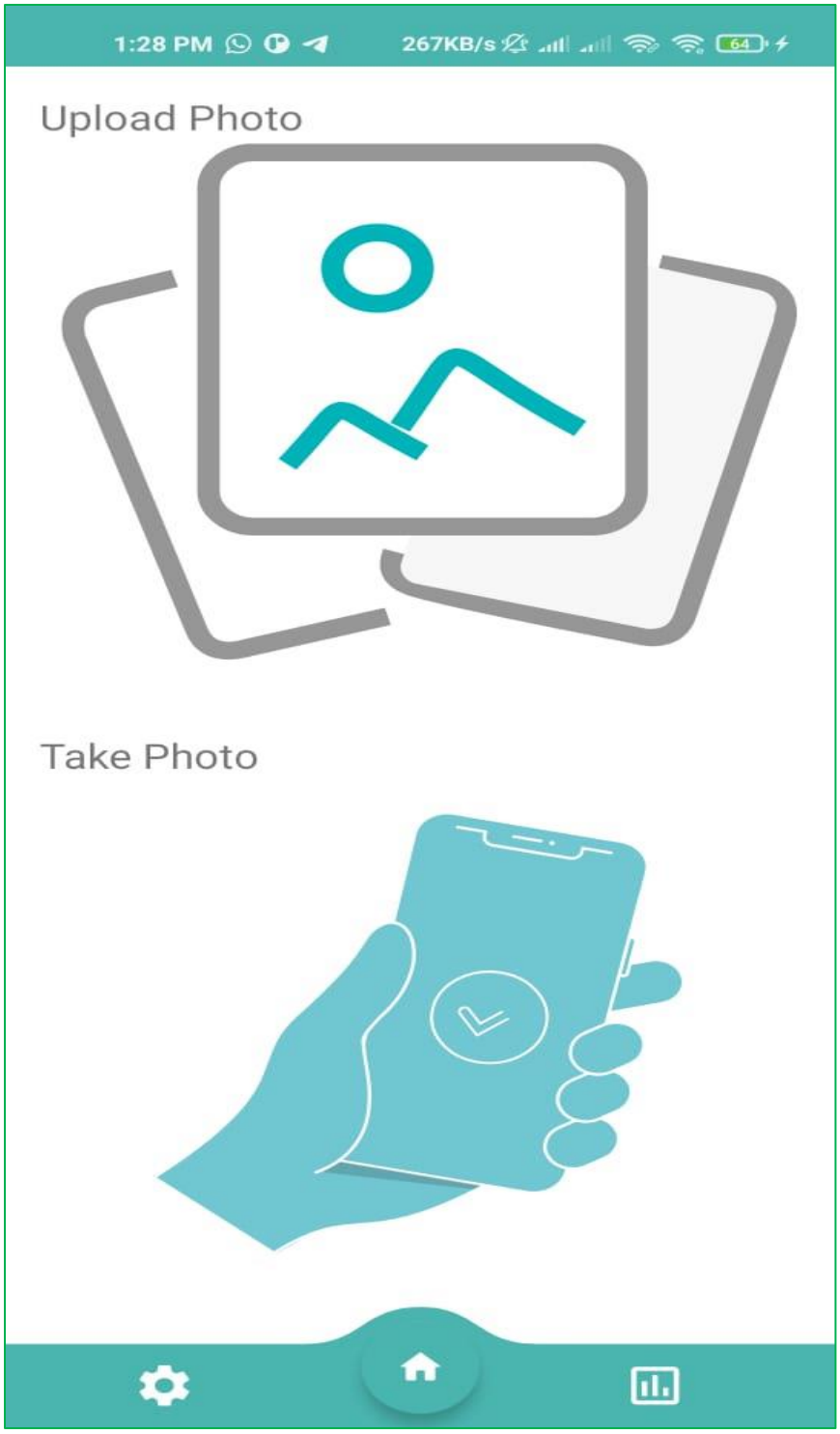
3.5- Add New Patient Page.



3.6- Doctor Profile.

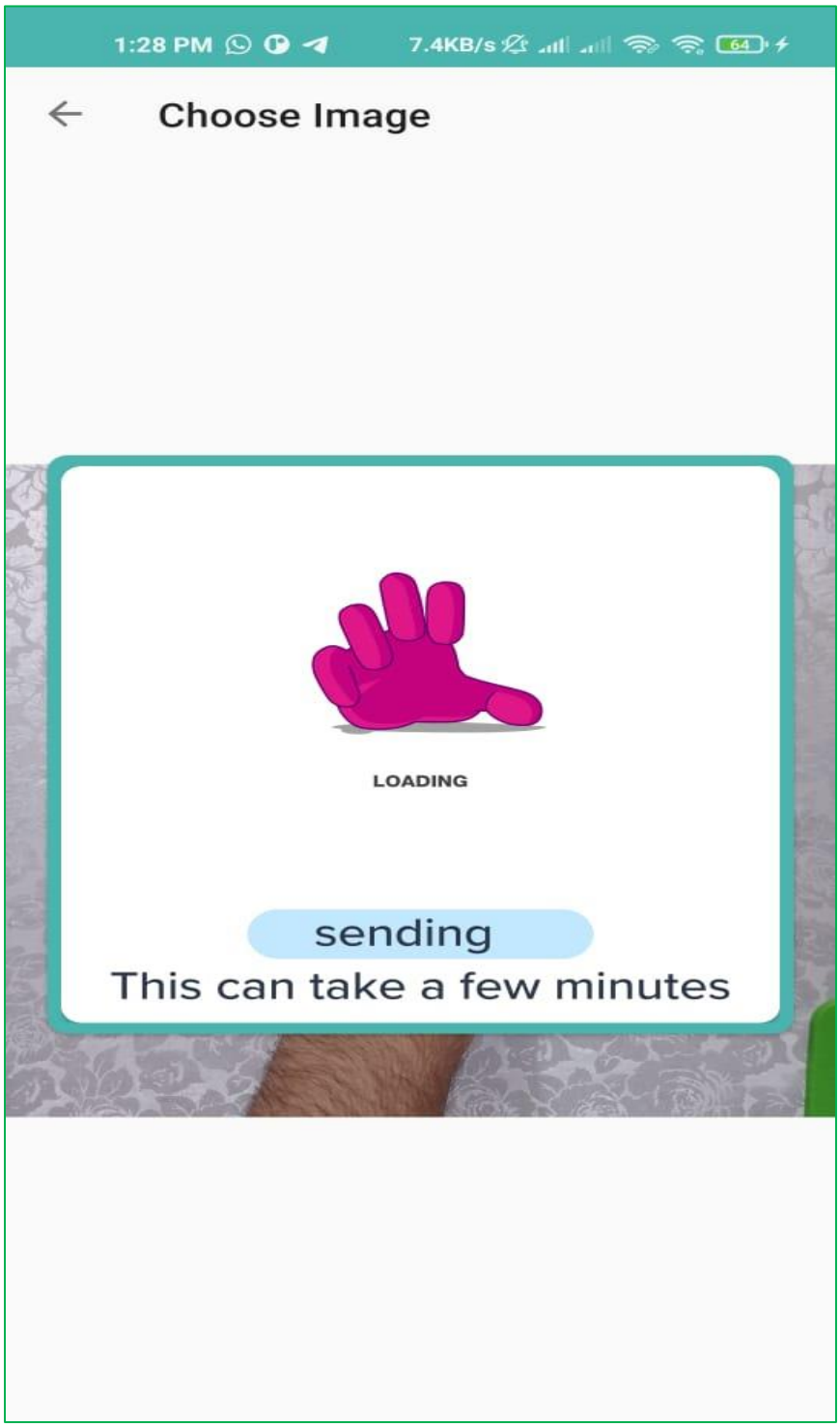


3.7- Patient Home Page

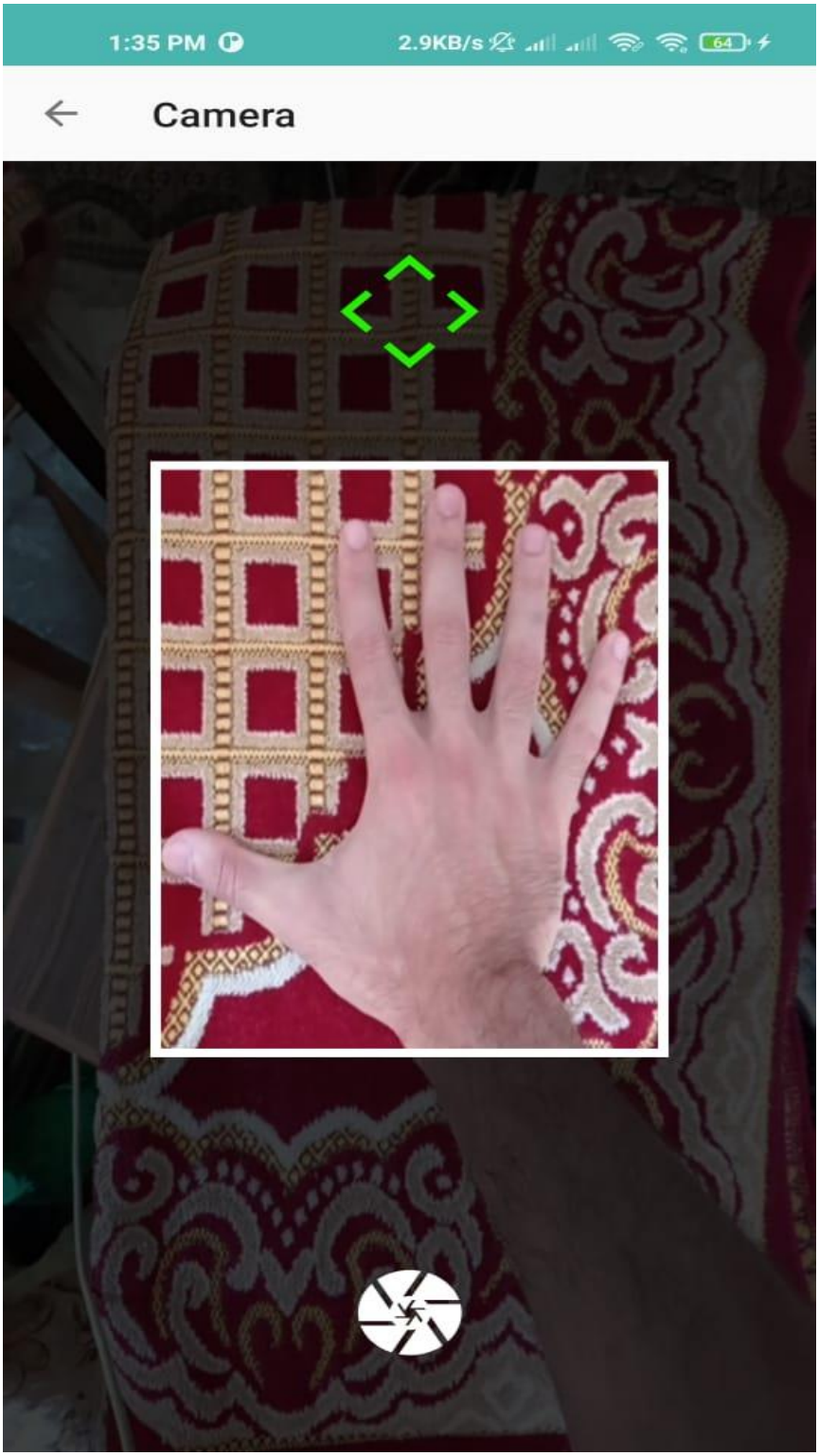


3.8- Choose image from Gallery.

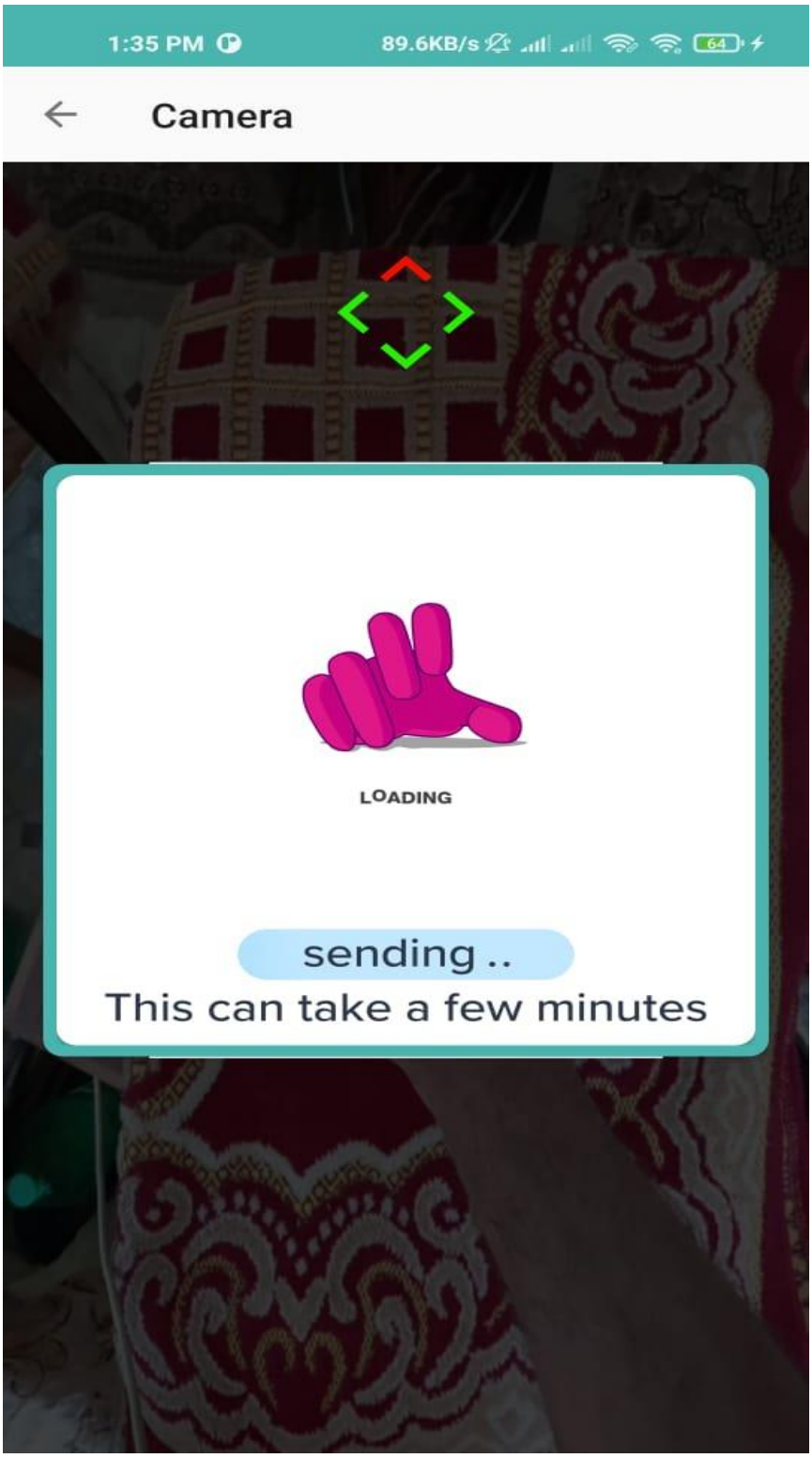
3.9- Sending Image to Server.



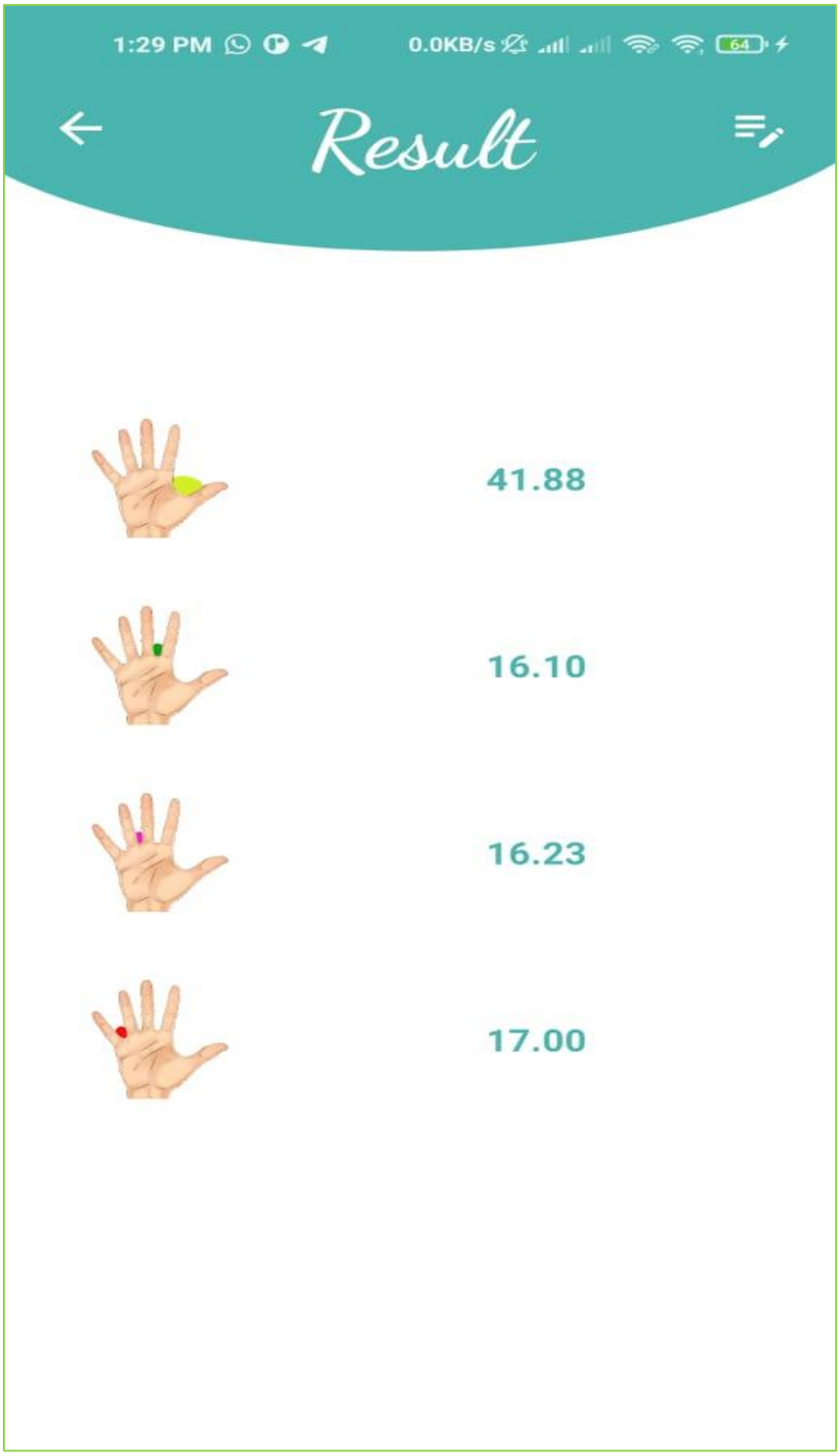
3.10- Take Hand Image



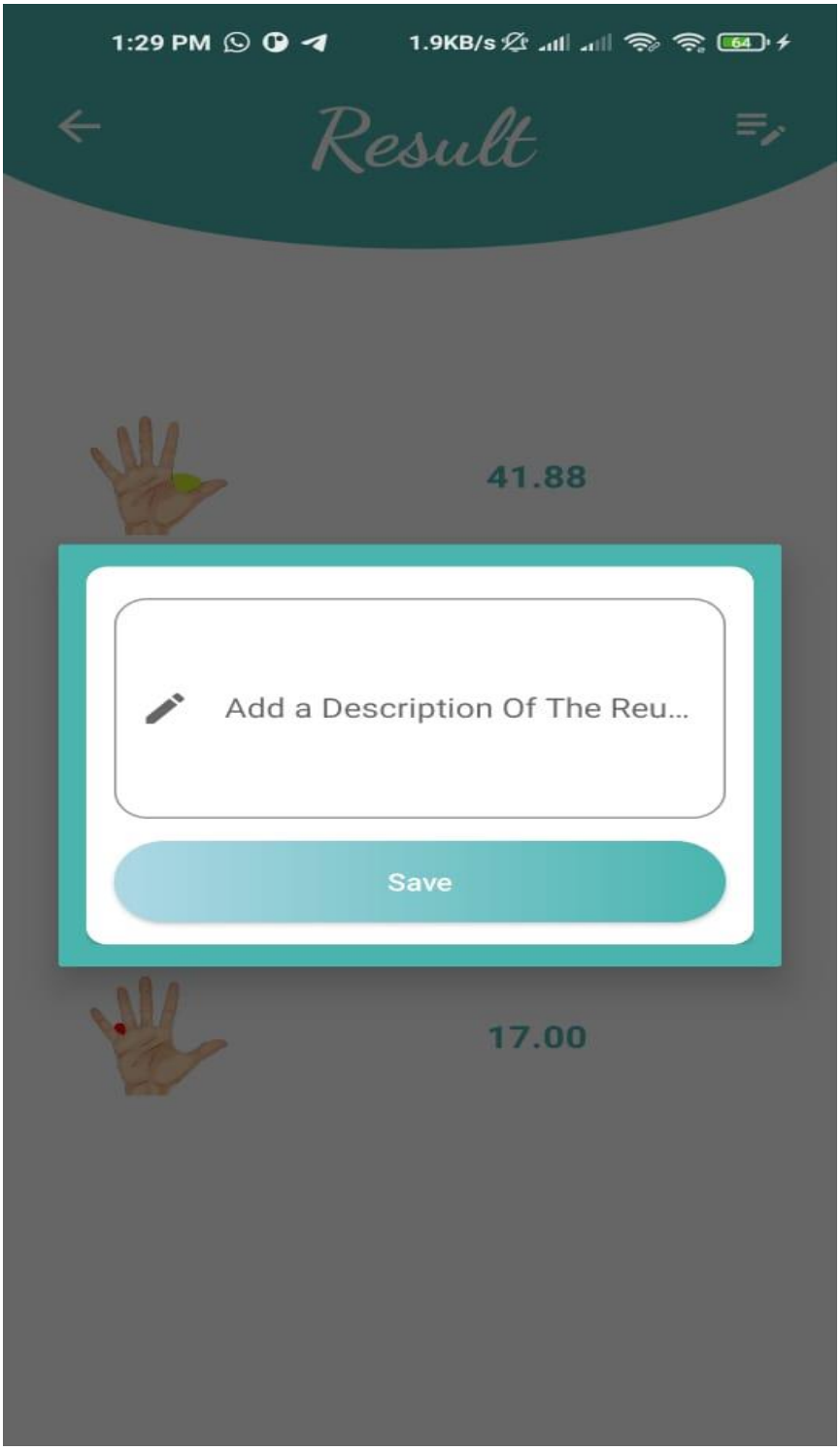
3.11- sending hand image to server



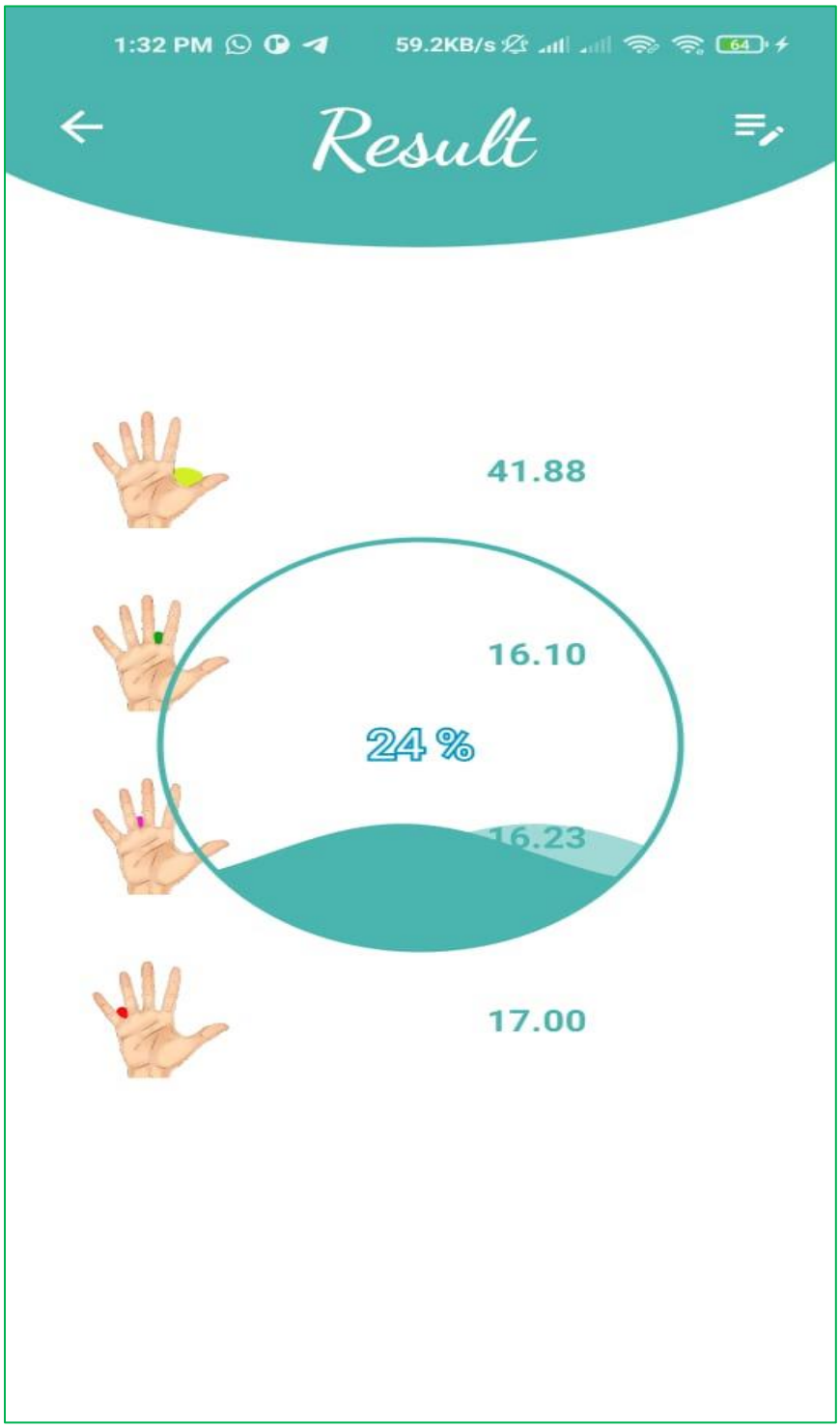
3.12- result hand angle page



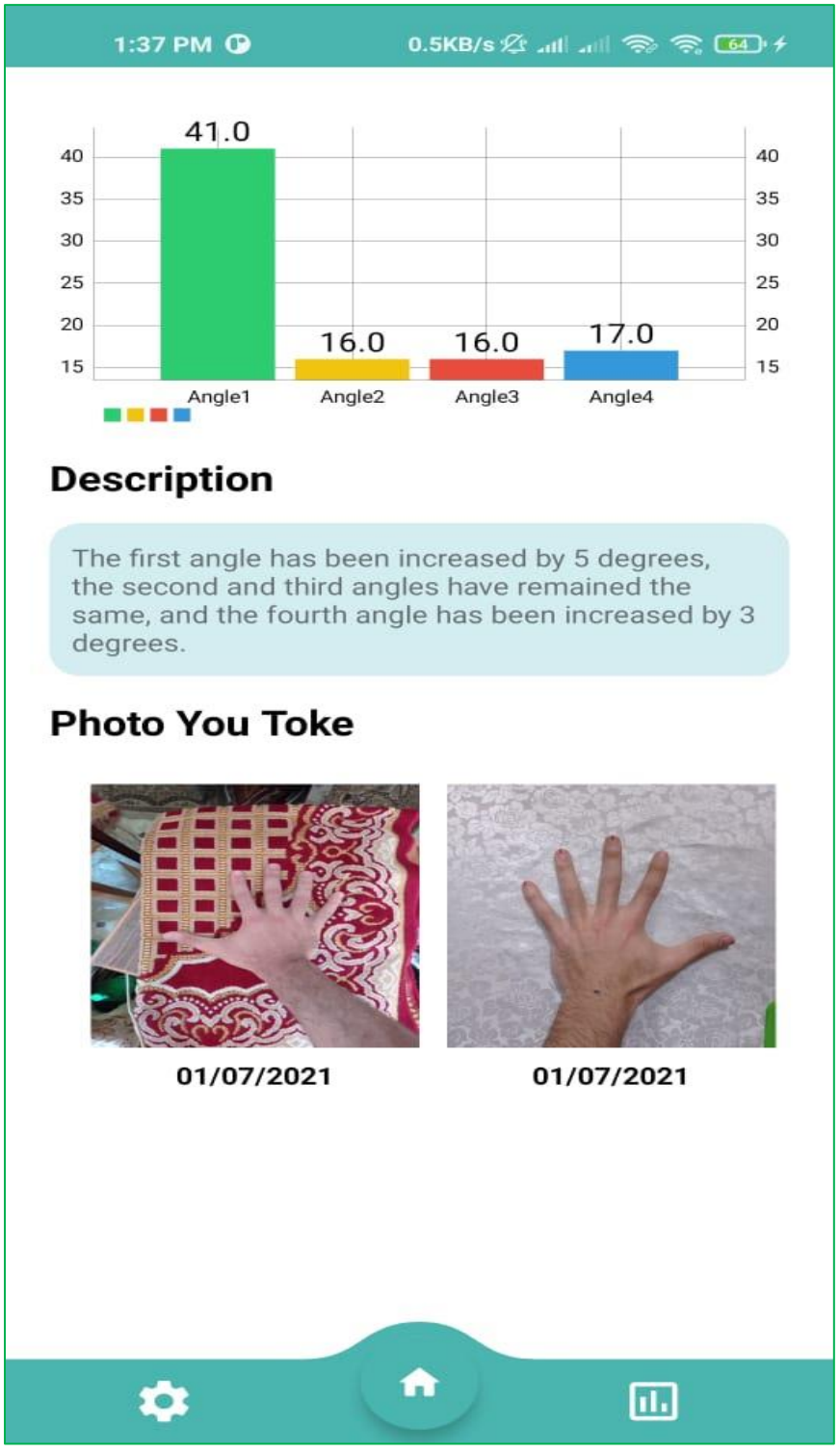
3.13- doctor can write description of result and save image and result



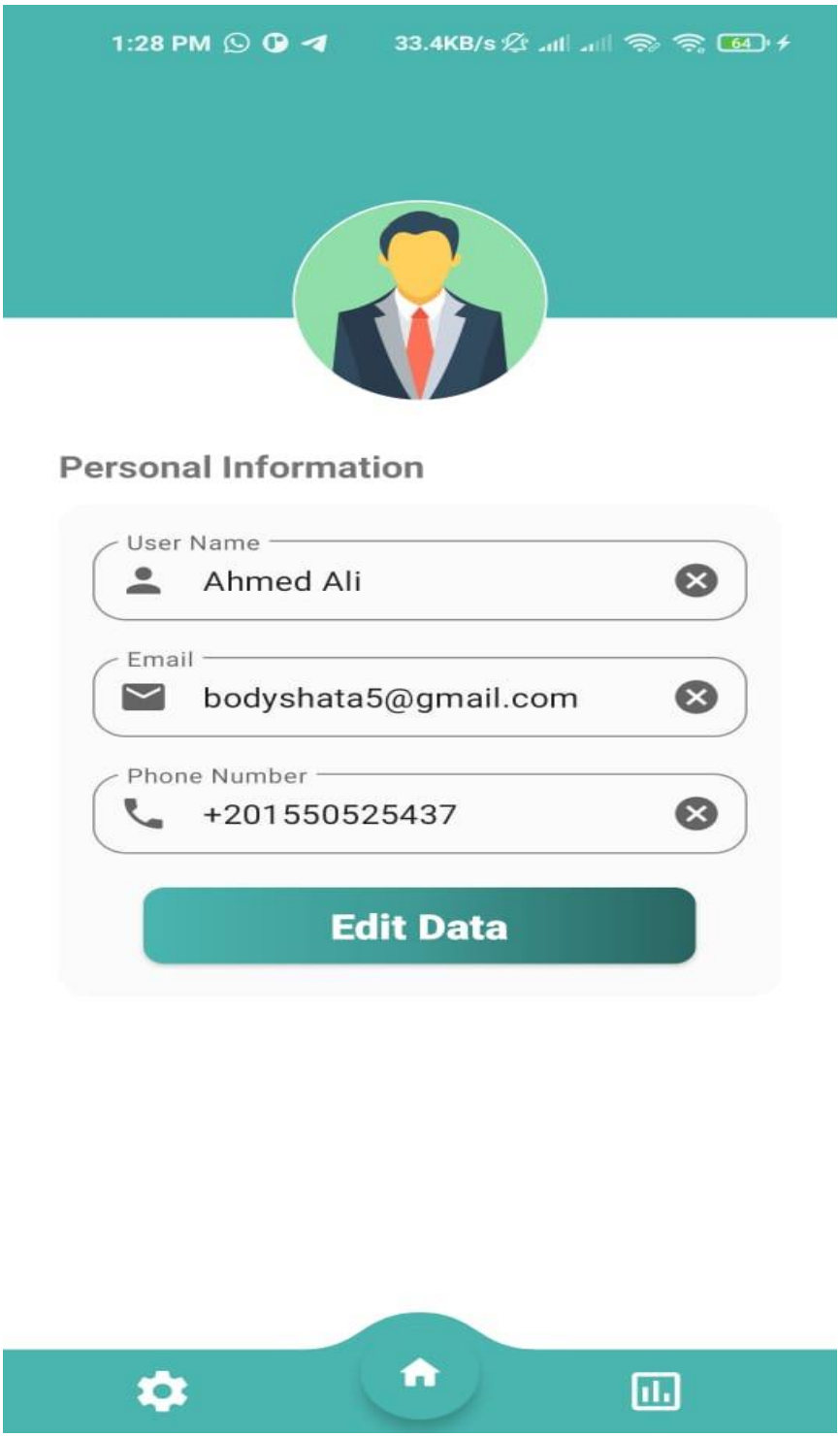
3.14- saving image



3.15- patient progress page



3.16- patient profile page



Chapter Four

Experiment and Results

4.1 Implementation:

We have created an application that computes the angles between

the fingers like in Figure (3).

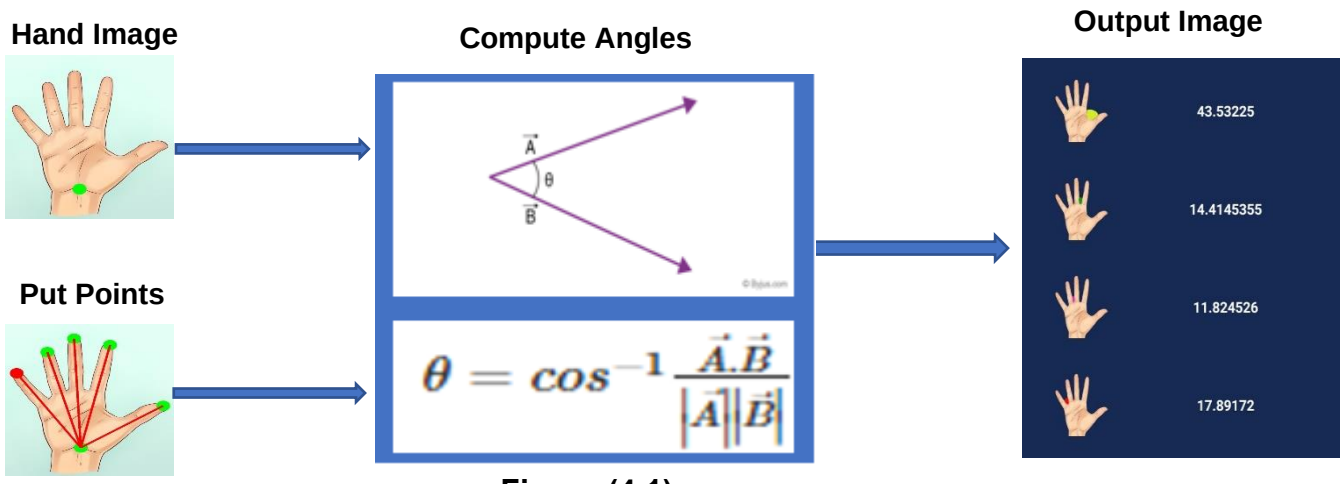


Figure (4.1)

Upload Photo

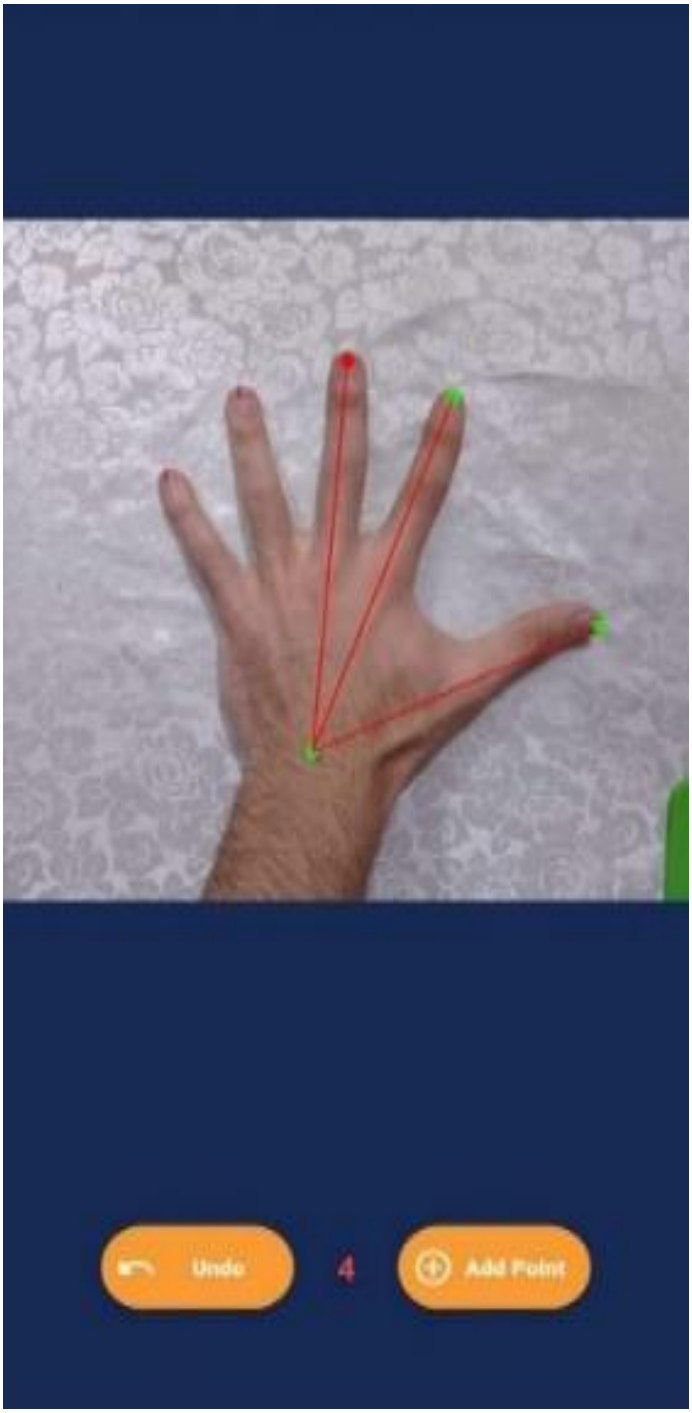
Select

Center Point

First Point

Second Point

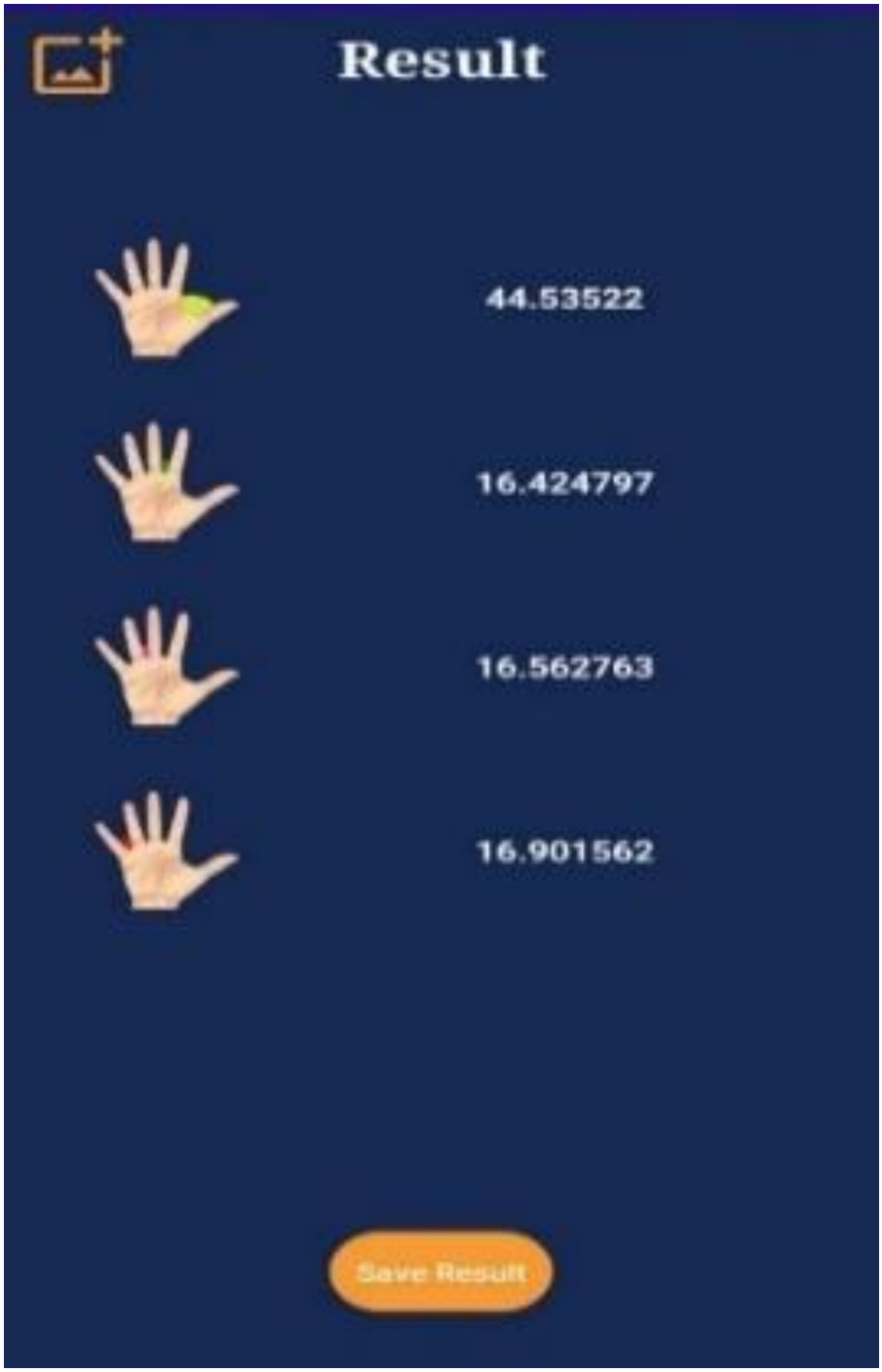
Third Point



Forth Point

Last Point

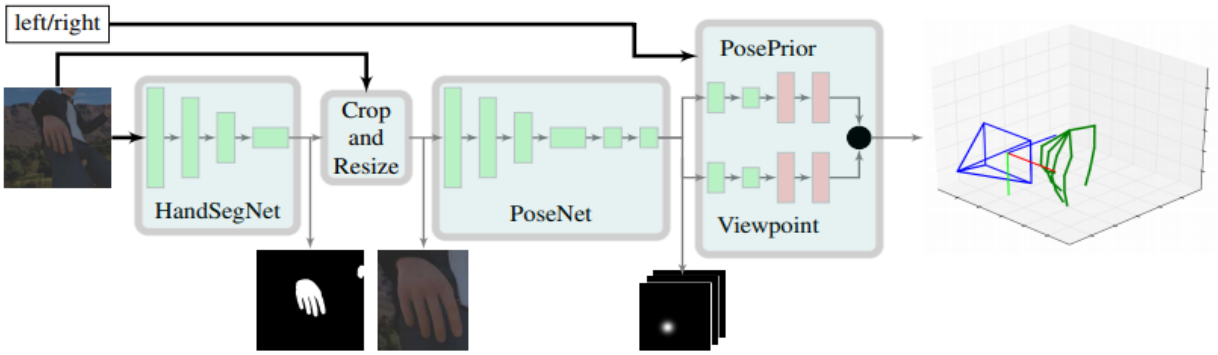
Result



4.2 Machine Learning

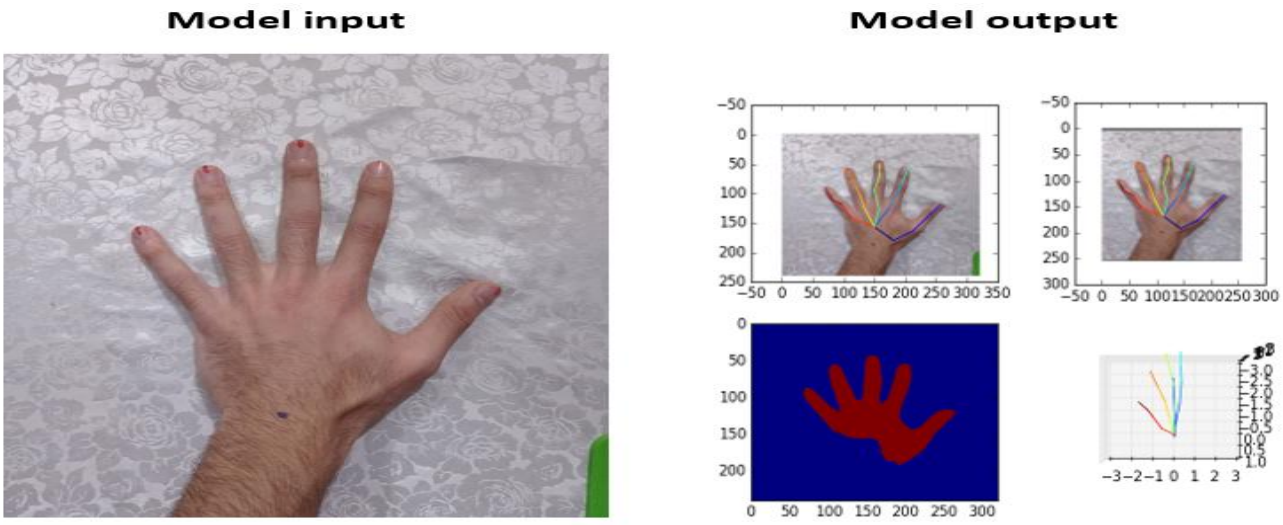
- we use a model consists of Convolutional Neural Network estimating 2D Hand Pose from a single RGB Image.
- The main purpose of using this model is to collect a set of hand key points like in figure (4).

Figure (4.2) Key points that model do them.



Figure(4.3): Our approach consists of three building blocks:

- The hand is localized within the image by a segmentation network (HandSegNet).
- Accordingly, to the hand mask, the input image is cropped and serves as input to the Pose Net.
- This localizes a set of hand key points represented as score maps c. Subsequently, the Pose Prior network estimates the most likely 2D structure conditioned on the score maps.



4.3 Dataset:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
X1	Y1	X2	Y2	X3	Y3	X4	Y4	Angle	X5	Y5	X6	Y6	X7	Y7	X8	Y8	Angle nur	RightorLeft
0.860019	-3.09649	0.753413	-2.53052	0.629331	-1.96242	0.361692	-1.22791	15.95801	0.456454	-2.8526	0.565661	-2.18847	0.568863	-1.59394	0.363601	-0.90782	2	0
0.456454	-2.8526	0.565661	-2.18847	0.568863	-1.59394	0.363601	-0.90782	16.42408	-0.46249	-2.03149	-0.12555	-1.5283	0.12385	-1.02663	0.238231	-0.51058	3	0
-0.46249	-2.03149	-0.12555	-1.5283	0.12385	-1.02663	0.238231	-0.51058	15.63518	-0.91526	-0.8673	-0.55691	-0.61907	-0.34712	-0.38529	-0.00611	-0.00794	4	0
0.106369	-2.18694	0.116188	-1.52086	0.142727	-0.70365	0.068778	0.170585	26.29968	0.381094	-3.31663	0.390072	-2.75407	0.427469	-2.13141	0.336888	-1.26616	1	0
0.381094	-3.31663	0.390072	-2.75407	0.427469	-2.13141	0.336888	-1.26616	15.05076	-0.26297	-3.25648	-0.09447	-2.59454	0.073604	-1.95445	0.088474	-1.11149	2	0
-0.26297	-3.25648	-0.09447	-2.59454	0.073604	-1.95445	0.088474	-1.11149	13.80723	-0.97094	-2.59504	-0.68358	-2.04217	-0.37603	-1.47279	-0.1901	-0.80085	3	0
-0.97094	-2.59504	-0.68358	-2.04217	-0.37603	-1.47279	-0.1901	-0.80085	14.9022	-1.49357	-1.45745	-1.15112	-1.15929	-0.9292	-0.89032	-0.5499	-0.3524	4	0
-0.30409	-1.99758	-0.21596	-1.30763	-0.1169	-0.45409	-0.06902	0.41662	15.05121	0.055419	-3.59259	0.131205	-2.97367	0.252867	-2.28604	0.242098	-1.32952	1	0
0.055419	-3.59259	0.131205	-2.97367	0.252867	-2.28604	0.242098	-1.32952	12.91501	-0.34493	-3.76696	-0.13743	-2.99723	0.053303	-2.26987	0.066792	-1.30821	2	0
-0.34493	-3.76696	-0.13743	-2.99723	0.053303	-2.26987	0.066792	-1.30821	7.8242	-0.90573	-3.18521	-0.60451	-2.52013	-0.29118	-1.86147	-0.1612	-1.0634	3	0
-0.90573	-3.18521	-0.60451	-2.52013	-0.29118	-1.86147	-0.1612	-1.0634	14.90128	-1.37806	-2.05701	-1.03313	-1.68632	-0.8128	-1.35043	-0.48936	-0.65571	4	0
-1.82141	-1.59195	-1.51389	-0.96289	-1.07434	-0.20566	-0.48062	0.548351	14.79674	-1.08156	-3.52222	-0.86576	-2.89304	-0.59247	-2.22524	-0.42759	-1.29182	1	0
-1.08156	-3.52222	-0.86576	-2.89304	-0.59247	-2.22524	-0.42759	-1.29182	13.30765	-0.83885	-3.73217	-0.53402	-3.00394	-0.27865	-2.29333	-0.17347	-1.31125	2	0
-0.83885	-3.73217	-0.53402	-3.00394	-0.27865	-2.29333	-0.17347	-1.31125	7.63281	-0.74586	-3.15151	-0.42393	-2.51601	-0.12764	-1.90133	-0.02003	-1.08894	3	0
0.079858	-2.27585	0.110173	-1.61019	0.123588	-0.80592	0.055486	0.068419	19.27549	0.432835	-3.1703	0.423602	-2.64078	0.433069	-2.04898	0.32526	-1.23626	1	0
0.432835	-3.1703	0.423602	-2.64078	0.433069	-2.04898	0.32526	-1.23626	13.1989	-0.26078	-3.12814	-0.06063	-2.45803	0.095771	-1.81487	0.097707	-1.02213	2	0
-0.26078	-3.12814	-0.06063	-2.45803	0.095771	-1.81487	0.097707	-1.02213	18.07126	-1.02828	-2.40304	-0.70209	-1.8667	-0.36454	-1.30497	-0.16287	-0.68006	3	0
-1.02828	-2.40304	-0.70209	-1.8667	-0.36454	-1.30497	-0.16287	-0.68006	16.15592	-1.49923	-1.21723	-1.12328	-0.93689	-0.89487	-0.67946	-0.49993	-0.20723	4	0
0.129745	-2.30122	0.084896	-1.61252	0.098763	-0.79981	0.025109	0.082421	30.13132	0.581242	-3.27618	0.532582	-2.69367	0.509752	-2.07271	0.357651	-1.24972	1	0
0.581242	-3.27618	0.532582	-2.69367	0.509752	-2.07271	0.357651	-1.24972	16.4133	-0.03022	-3.13964	0.079306	-2.47409	0.187384	-1.82852	0.130153	-1.044	2	0
-0.03022	-3.13964	0.079306	-2.47409	0.187384	-1.82852	0.130153	-1.044	14.47935	-0.77297	-2.5293	-0.51949	-1.91895	-0.26159	-1.33507	-0.13929	-0.70757	3	0
-0.77297	-2.5293	-0.51949	-1.91895	-0.26159	-1.33507	-0.13929	-0.70757	15.98332	-1.37119	-1.33517	-1.03376	-1.00637	-0.82905	-0.73151	-0.49395	-0.24838	4	0
-0.176	-2.27689	-0.12006	-1.5642	-0.04634	-0.67771	-0.03253	0.273766	16.44489	0.128192	-3.58549	0.208429	-2.97406	0.327548	-2.2958	0.306496	-1.36865	1	0
0.128192	-3.58549	0.208429	-2.97406	0.327548	-2.2958	0.306496	-1.36865	14.65065	0.32868	-3.65627	0.11691	-3.88745	0.078274	-3.17226	0.08759	-1.35057	2	0

Figure 4.4

Models performance comparison using all attributes

Model	RMSE (Training)	RMSE (Test)
Linear regression	**5.67**	**5.80**
Ridge regression	6.12	6.20
Lasso regression	7.64	7.71
RFA	1.88	5.06
SVR	5.96	6.13

Models performance comparison using all attributes without (LR&AN)

Model	RMSE (Training)	RMSE (Test)
Linear regression	5.87	6.05
RFA	**1.89**	**5.03**
SVR	5.99	6.19

Models performance comparison (Angle 1(R))

Model	RMSE (Training)	RMSE (Test)
Linear regression	4.51	4.86
SVR	5.72	5.40

Models performance comparison (Angle 2(R))

Model	RMSE (Training)	RMSE (Test)
Linear regression	3.86	4.05
SVR	4.55	4.62

Models performance comparison (Angle 3(R))

Model	RMSE (Training)	RMSE (Test)
Linear regression	3.20	4.40
SVR	4.0	4.14

Models performance comparison (Angle 4(R))

Model	RMSE (Training)	RMSE (Test)
Linear regression	3.58	3.67
SVR	4.45	4.63

Models performance comparison (Angle 5(L))

Model	RMSE (Training)	RMSE (Test)
Linear regression	6.80	7.46
SVR	10.27	9.69

Models performance comparison (Angle 6(L))

Model	RMSE (Training)	RMSE (Test)
Linear regression	4.16	4.17
SVR	4.59	4.49

Models performance comparison (Angle 7(L))

Model	RMSE (Training)	RMSE (Test)
Linear regression	3.08	3.08
SVR	3.93	3.83

Models performance comparison (Angle 8(L))

Model	RMSE (Training)	RMSE (Test)
Linear regression	3.97	4.08
SVR	4.80	5.07

Models performance comparison (Angle 1(RL))

Model	RMSE (Training)	RMSE (Test)
Linear regression	6.79	8.01
SVR	7.92	9.46

Models performance comparison (Angle 2(RL))

Model	RMSE (Training)	RMSE (Test)
Linear regression	4.21	4.54
SVR	4.59	4.75

Models performance comparison (Angle 3(RL))

Model	RMSE (Training)	RMSE (Test)
Linear regression	3.60	3.67
SVR	3.94	4.11

Models performance comparison (Angle 4(RL))

Model	RMSE (Training)	RMSE (Test)
Linear regression	4.22	4.29
SVR	4.78	4.73

Models (RFC & Logistic Reg & SVM) to know (Right or Left) performance comparison

In [7]:
```
Data = train_set.drop( RightorLeft , axis=1) # drop labels for training set
pre_label = train_set["RightorLeft"].copy()
```

In [10]:
```
from sklearn.ensemble import RandomForestClassifier
classifier_ran = RandomForestClassifier(n_estimators=100, max_depth=12, random_state=42)
classifier_ran.fit(Data,pre_label)
classifier_ran.score(X,y)*100
```

Out[10]: 99.84025559105432

In [12]:
```
from sklearn.linear_model import LogisticRegression
classifier_log = LogisticRegression(random_state=42)
classifier_log.fit(Data,pre_label)
classifier_log.score(X,y)*100
```

```
C:\Users\Ahmed Atef\Anaconda\envs\tf\lib\site-packages\sklearn\linear_model\_logistic.py:940: ConvergenceWarning: lbfgs failed
to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

Out[12]: 98.40255591054313

In [13]:
```
from sklearn import svm
SVM = svm.LinearSVC()
SVM.fit(Data,pre_label)

SVM.score(X,y)*100
```

```
C:\Users\Ahmed Atef\Anaconda\envs\tf\lib\site-packages\sklearn\svm\_base.py:947: ConvergenceWarning: Liblinear failed to conver
ge, increase the number of iterations.
  "the number of iterations.", ConvergenceWarning)
```


Out[13]: 99.40666362391603

4.4 Security

- Passwords stored is Base64 Encoded & dm5 hashed.
-

4.5 Database

WE used SQLite data base and Firebase:

SQLite is an in-process library that implements a self contained, serverless, zero-configuration, transactional SQL database engine. The code for SQLite is in the public domain and is thus free for use for any purpose, commercial or private. SQLite is the most widely deployed database in the world with more applications than we can count, including several high-profile projects.

Firebase is a Backend-as-a-Service (Baas). It provides developers with a variety of tools and services to help them develop quality apps, grow their user base, and earn profit. It is built on Google's infrastructure.

Firebase is categorized as a NoSQL database program, which stores data in JSON-like documents.

4.6 Environment and Tools

These are the tools and the environment that we used during the life cycle of the project. Our choice for the development environments and tools was based on the budget, ease of use, experience and keeping up to date with what is currently used for the market.

Operating system:

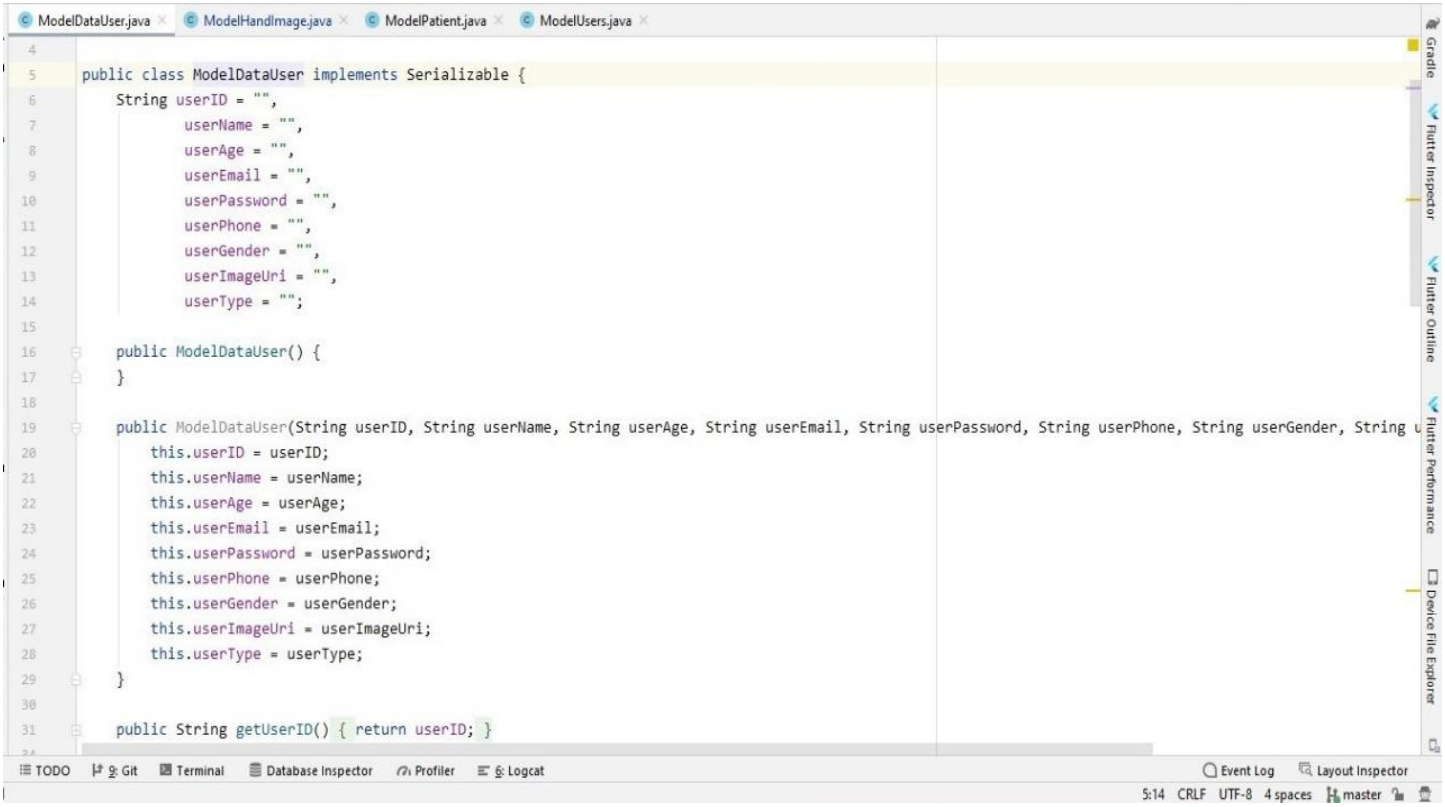
• **Windows**: for the availability of all the tools specially designing and prototyping tools

Frontend (Design)

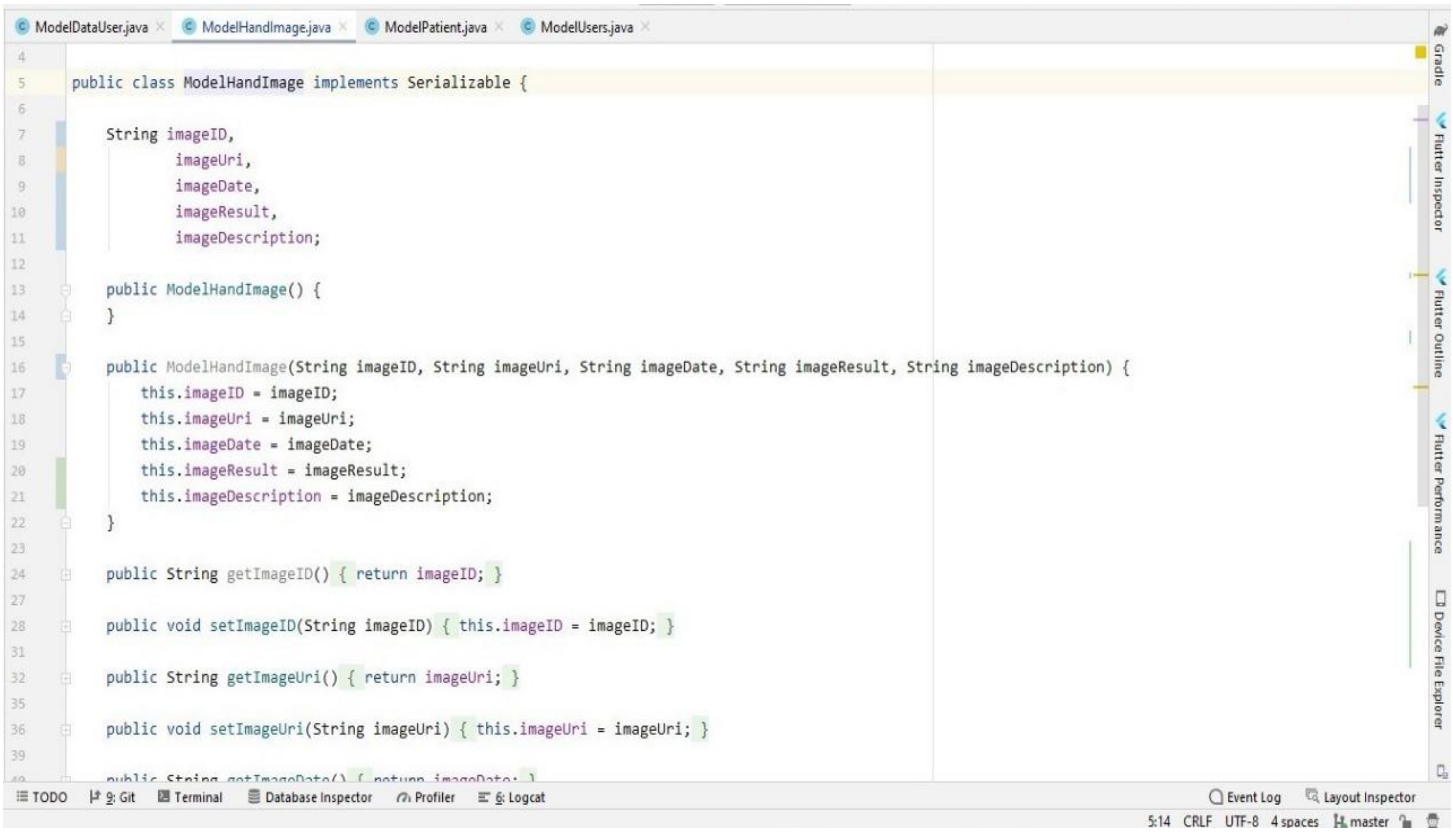
- UI UX
- Adobe XD
- XML
- EdrawMax

- **Backend (Development)**
  - Android Studio
  - Java
  - SQLite
  - Firebase
- **ML (Python)**
  - Tensorflow
  - Scipy
  - Matplotlib
  - Numpy
  - Pandas

4.7 Some of Design Code.



```
public class ModelDataUser implements Serializable {
    String userID = "",
            userName = "",
            userAge = "",
            userEmail = "",
            userPassword = "",
            userPhone = "",
            userGender = "",
            userImageUri = "",
            userType = "";

    public ModelDataUser() {
    }

    public ModelDataUser(String userID, String userName, String userAge, String userEmail, String userPassword, String userPhone, String userGender, String u
        this.userID = userID;
        this.userName = userName;
        this.userAge = userAge;
        this.userEmail = userEmail;
        this.userPassword = userPassword;
        this.userPhone = userPhone;
        this.userGender = userGender;
        this.userImageUri = userImageUri;
        this.userType = userType;
    }

    public String getUserID() { return userID; }
```



```
public class ModelHandImage implements Serializable {

    String imageID,
            imageUri,
            imageDate,
            imageResult,
            imageDescription;

    public ModelHandImage() {
    }

    public ModelHandImage(String imageID, String imageUri, String imageDate, String imageResult, String imageDescription) {
        this.imageID = imageID;
        this.imageUri = imageUri;
        this.imageDate = imageDate;
        this.imageResult = imageResult;
        this.imageDescription = imageDescription;
    }

    public String getImageID() { return imageID; }

    public void setImageID(String imageID) { this.imageID = imageID; }

    public String getImageUri() { return imageUri; }

    public void setImageUri(String imageUri) { this.imageUri = imageUri; }
```

ModelDataUser.java | ModelHandImage.java | ModelPatient.java | ModelUsers.java

```java
public class ModelPatient implements Serializable {

    String patientID = "",
           patientName = "",
           patientAge = "",
           patientEmail = "",
           patientPhone = "",
           patientGender = "",
           patientImageUri = "";

    public ModelPatient() {
    }

    public ModelPatient(String patientID, String patientName, String patientAge, String patientEmail, String patientPhone, String patientGender, String patier
        this.patientID = patientID;
        this.patientName = patientName;
        this.patientAge = patientAge;
        this.patientEmail = patientEmail;
        this.patientPhone = patientPhone;
        this.patientGender = patientGender;
        this.patientImageUri = patientImageUri;
    }

    public String getPatientID() { return patientID; }

    public void setPatientID(String patientID) { this.patientID = patientID; }
```

TODO | 9: Git | Terminal | Database Inspector | Profiler | 6: Logcat | Event Log | Layout Inspector

5:14 CRLF UTF-8 4 spaces master

Add_Patient_Activity.java | DocterActivity.java | DocterProfileActivity.java | LoginActivity.java | RegisterActivity.java

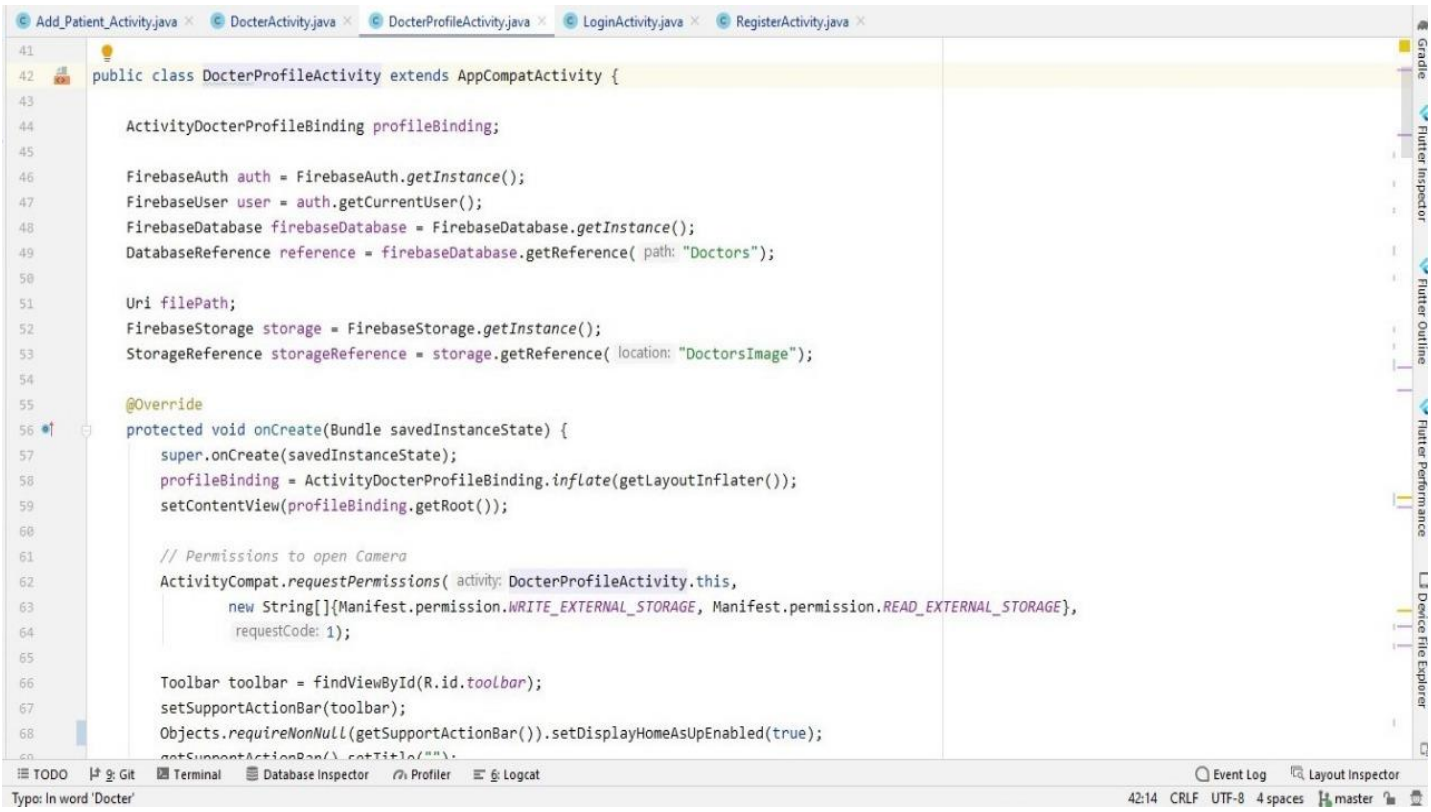
```java
public class DocterActivity extends AppCompatActivity implements Adapter_All_Patient.OnItemClickListener, SwipeRefreshLayout.OnRefreshListener {

    public static String NameSearch = "";

    ActivityDocterBinding docterBinding;
    FirebaseDatabase database = FirebaseDatabase.getInstance();
    DrawerLayout drawerLayout;
    FirebaseAuth firebaseAuth = FirebaseAuth.getInstance();
    FirebaseUser user = firebaseAuth.getCurrentUser();
    DatabaseReference mDatabaseRef = database.getReference( path: "DoctorPatients");
    DatabaseReference reference = database.getReference( path: "Doctors");
    ValueEventListener mDBListener;
    List<ModelPatient> mAllPatients;
    Adapter_All_Patient allPatientAdapter;
    // This Method is for Search in RealTime
    private final TextWatcher searchTextWatcher = new TextWatcher() {
        @Override
        public void beforeTextChanged(CharSequence s, int start, int count, int after) {
        }

        @Override
        public void onTextChanged(final CharSequence s, int start, int before, int count) {
            mDBListener = mDatabaseRef
                    .child(user.getUid())
                    .orderByChild("patientName")
                    .addValueEventListener(new ValueEventListener() {
                        @Override
```

TODO | 9: Git | Terminal | Database Inspector | Profiler | 6: Logcat | Event Log | Layout Inspector

Typo: In word 'Docter'

48:14 CRLF UTF-8 4 spaces master

```
Add_Patient_Activity.java

public class Add_Patient_Activity extends AppCompatActivity {

    RadioButton selectedGender;
    ActivityAddPatientBinding patientBinding;
    FirebaseAuth firebaseAuth = FirebaseAuth.getInstance();
    FirebaseUser user = firebaseAuth.getCurrentUser();
    FirebaseDatabase DataBase_Patient = FirebaseDatabase.getInstance();
    DatabaseReference mDatabaseRef_Patient = DataBase_Patient.getReference( path: "DoctorPatients/" + user.getUid());

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        patientBinding = ActivityAddPatientBinding.inflate(getLayoutInflater());
        setContentView(patientBinding.getRoot());

        // Toolbar
        patientBinding.toolbar.setTitle("Add New Patient");
        setSupportActionBar(patientBinding.toolbar);
        Objects.requireNonNull(getSupportActionBar()).setDisplayHomeAsUpEnabled(true);
        getSupportActionBar().setDisplayShowHomeEnabled(true);

        patientBinding.AddDataPatient.setOnClickListener(v -> continueRegister());
    }

    public void continueRegister() {
        patientBinding.progressCircle.setVisibility(View.VISIBLE);
```



```
Add_Patient_Activity.java | DocterActivity.java | DocterProfileActivity.java | LoginActivity.java | RegisterActivity.java

public class DocterProfileActivity extends AppCompatActivity {

    ActivityDocterProfileBinding profileBinding;

    FirebaseAuth auth = FirebaseAuth.getInstance();
    FirebaseUser user = auth.getCurrentUser();
    FirebaseDatabase firebaseDatabase = FirebaseDatabase.getInstance();
    DatabaseReference reference = firebaseDatabase.getReference( path: "Doctors");

    Uri filePath;
    FirebaseStorage storage = FirebaseStorage.getInstance();
    StorageReference storageReference = storage.getReference( location: "DoctorsImage");

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        profileBinding = ActivityDocterProfileBinding.inflate(getLayoutInflater());
        setContentView(profileBinding.getRoot());

        // Permissions to open Camera
        ActivityCompat.requestPermissions( activity: DocterProfileActivity.this,
                new String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE, Manifest.permission.READ_EXTERNAL_STORAGE},
                 requestCode: 1);

        Toolbar toolbar = findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
        Objects.requireNonNull(getSupportActionBar()).setDisplayHomeAsUpEnabled(true);
```

PatientActivity.java | HomePatientFragment.java | ProfilePatientFragment.java | ProgressPatientFragment.java

```java
public class PatientActivity extends AppCompatActivity {

    public static ModelPatient patient = null;
    SpaceNavigationView spaceNavigationView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_patient);

        patient = (ModelPatient) getIntent().getSerializableExtra( name: "PatientSelected");
        spaceNavigationView = (SpaceNavigationView) findViewById(R.id.space);
        Home();
        spaceNavigationView.setCentreButtonIcon(R.drawable.ic_round_home_24);
        spaceNavigationView.initWithSaveInstanceState(savedInstanceState);
        spaceNavigationView.addSpaceItem(new SpaceItem( itemName: "", R.drawable.ic_baseline_settings_24));
        spaceNavigationView.addSpaceItem(new SpaceItem( itemName: "", R.drawable.ic_outline_assessment_24));
        spaceNavigationView.setSpaceOnClickListener(new SpaceOnClickListener() {
            @Override
            public void onCentreButtonClick() {
                Home();
                spaceNavigationView.setCentreButtonSelectable(true);
            }

            @Override
            public void onItemClick(int itemIndex, String itemName) {

                switch (itemIndex) {
```

TODO | 9: Git | Terminal | Database Inspector | Profiler | 6: Logcat | Event Log | Layout Inspector

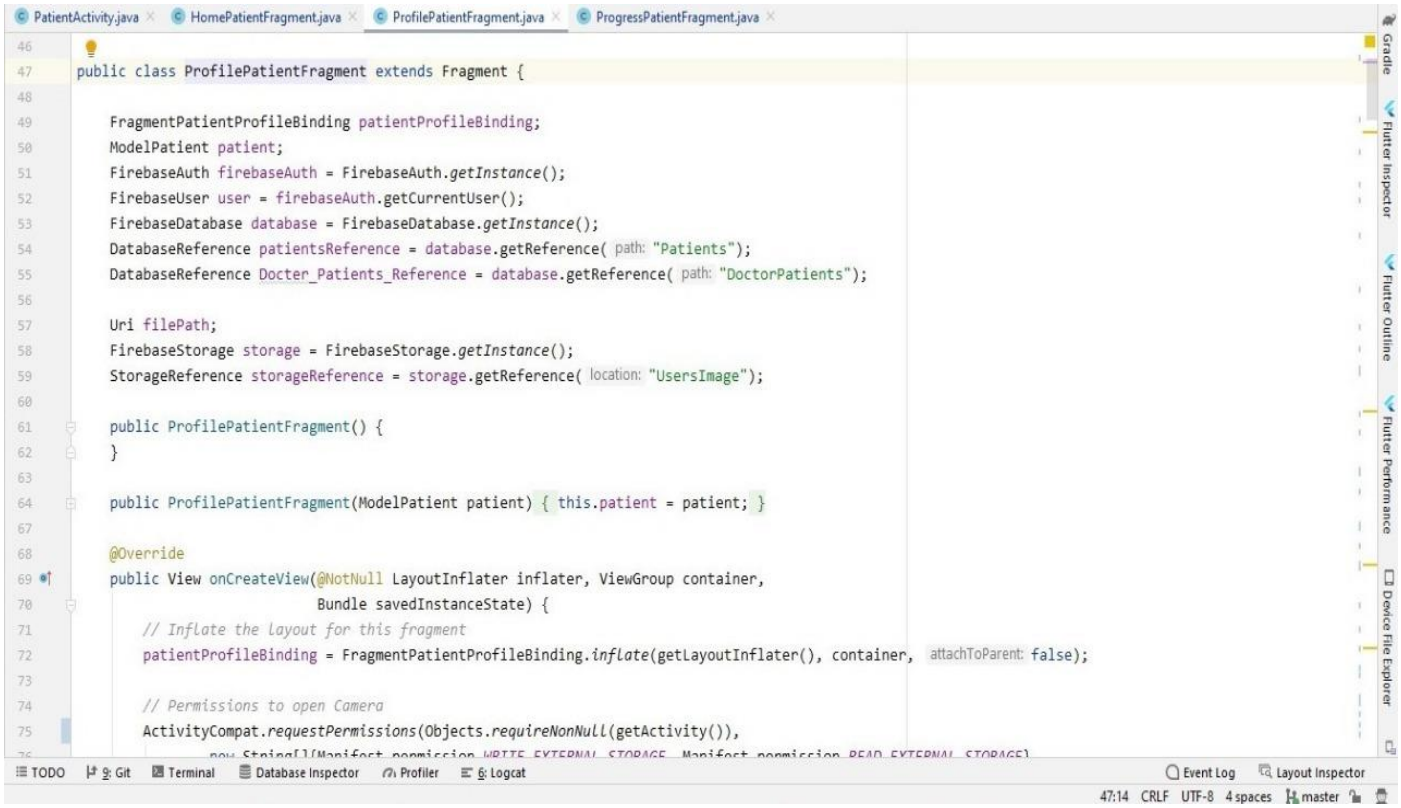
20:14 CRLF UTF-8 4 spaces master

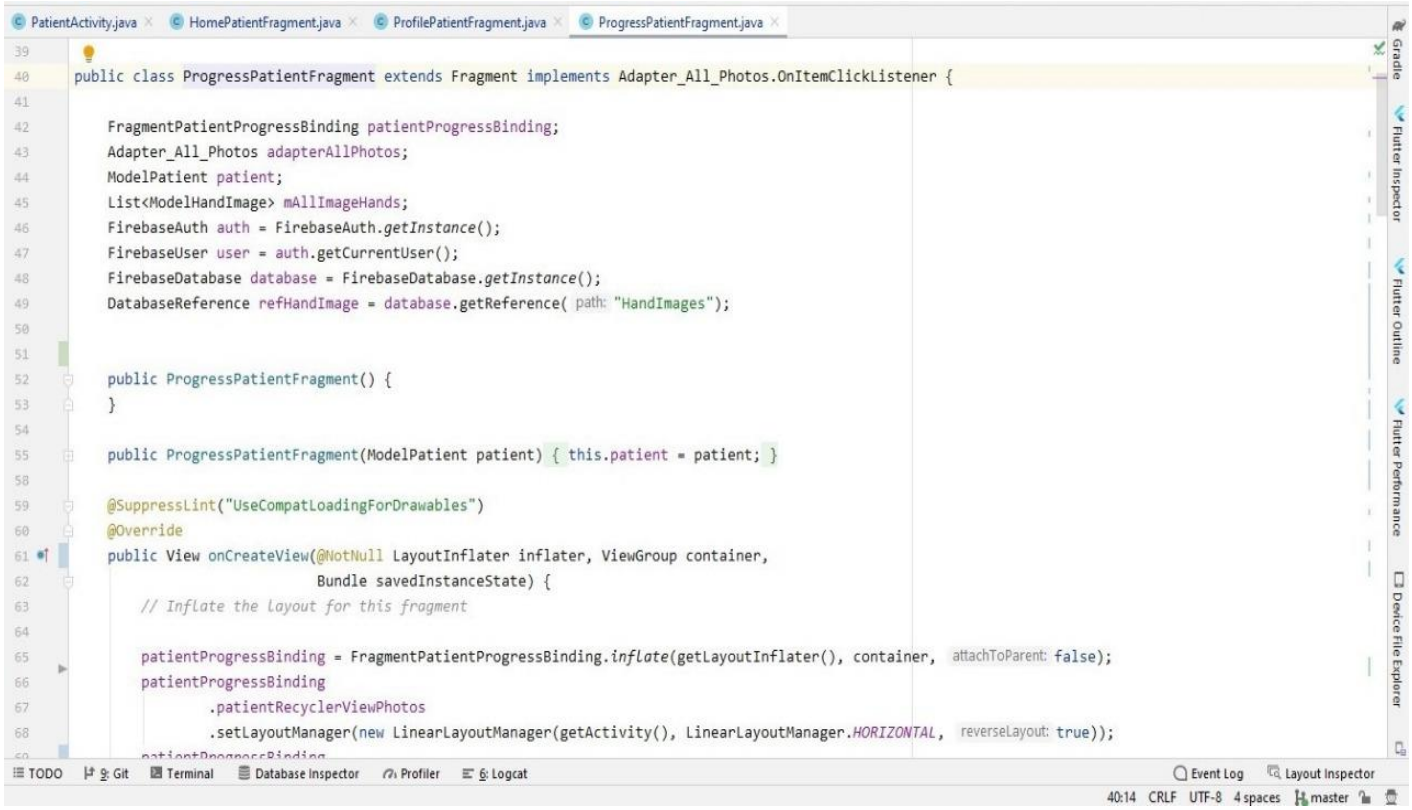
PatientActivity.java | HomePatientFragment.java | ProfilePatientFragment.java | ProgressPatientFragment.java

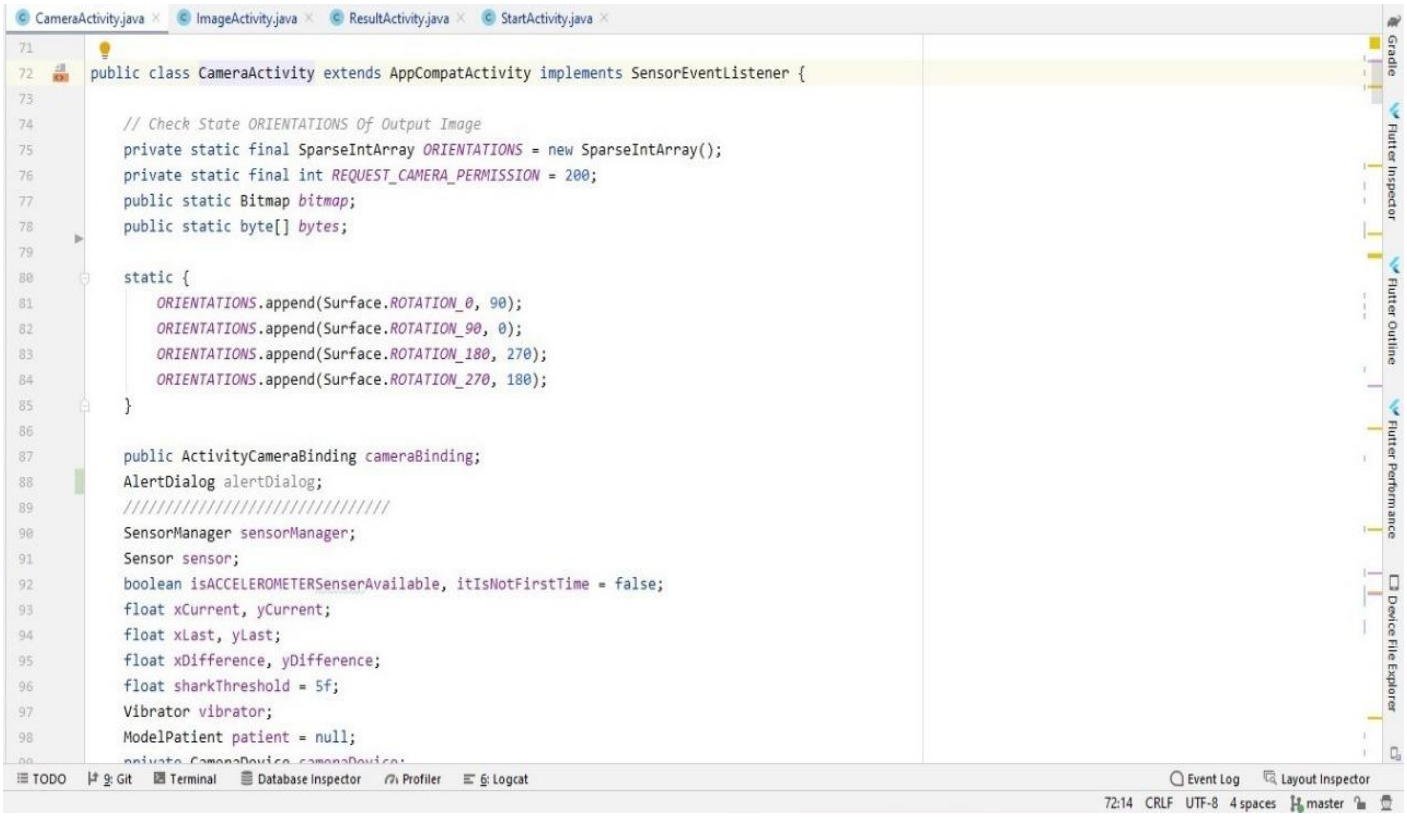
```java
public class HomePatientFragment extends Fragment {

    FragmentPatientHomeBinding patientHomeBinding;
    ModelPatient patient = null;

    public HomePatientFragment() {
    }

    public HomePatientFragment(ModelPatient patient) { this.patient = patient; }

    @Override
    public View onCreateView(@NotNull LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {

        // Inflate the layout for this fragment
        patientHomeBinding = FragmentPatientHomeBinding.inflate(getLayoutInflater(), container,  attachToParent: false);
        Initialize_variables();
        return patientHomeBinding.getRoot();
    }

    private void Initialize_variables() {

        patientHomeBinding.layoutTakePhoto.setOnClickListener(v -> {
            startActivity(new Intent(getActivity().getApplicationContext(), CameraActivity.class).putExtra( name: "ObjectPatient", patient));
            getActivity().overridePendingTransition(R.anim.slide_in_right, R.anim.slide_out_left);
            //getActivity().finish();
        });
        patientHomeBinding.layoutUploadPhoto.setOnClickListener(v -> {
```

TODO | 9: Git | Terminal | Database Inspector | Profiler | 6: Logcat | Event Log | Layout Inspector

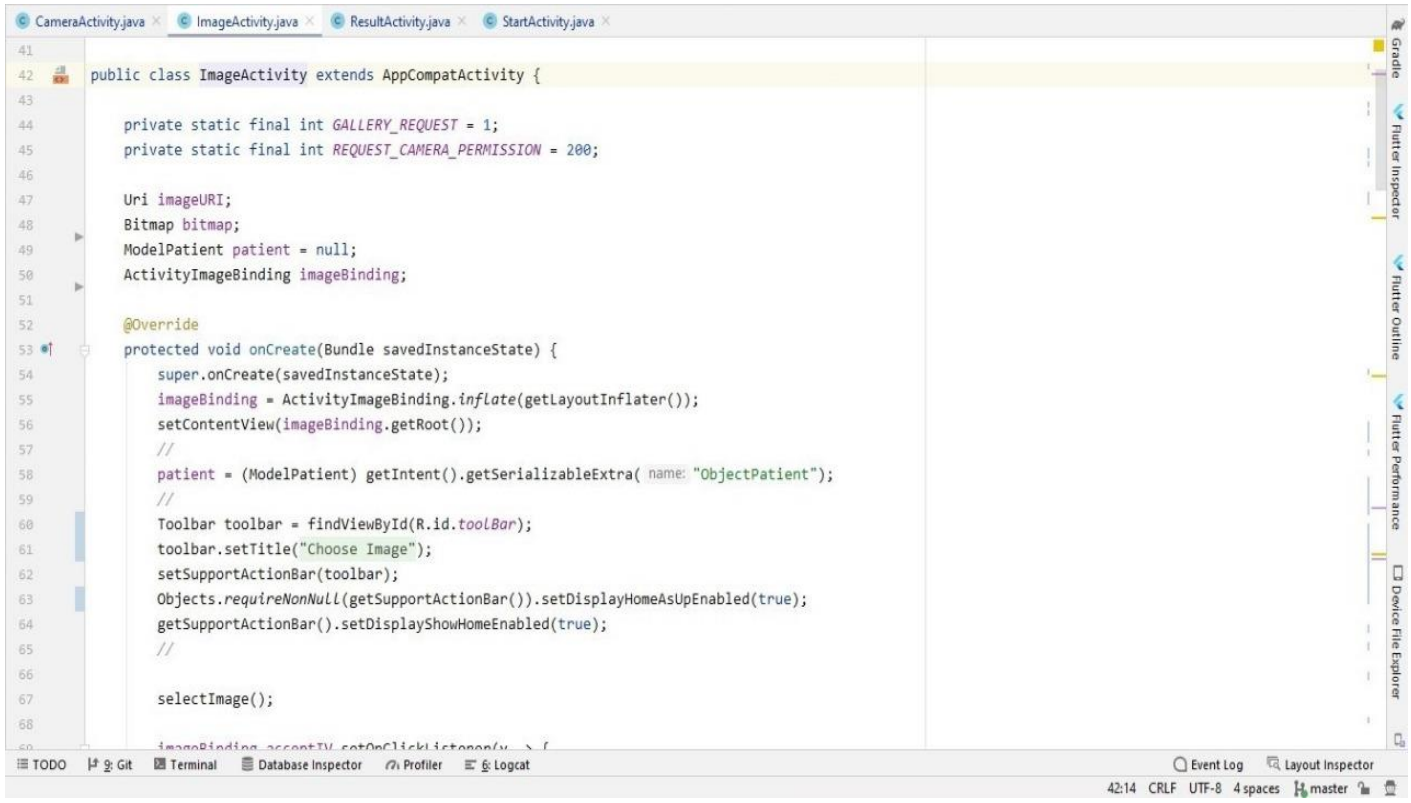
21:14 CRLF UTF-8 4 spaces master


PatientActivity.java
HomePatientFragment.java
ProfilePatientFragment.java
ProgressPatientFragment.java
public class ProfilePatientFragment extends Fragment {
FragmentPatientProfileBinding patientProfileBinding;
ModelPatient patient;
FirebaseAuth firebaseAuth = FirebaseAuth.getInstance();
FirebaseUser user = firebaseAuth.getCurrentUser();
FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference patientsReference = database.getReference(path: "Patients");
DatabaseReference Docter_Patients_Reference = database.getReference(path: "DoctorPatients");
Uri filePath;
FirebaseStorage storage = FirebaseStorage.getInstance();
StorageReference storageReference = storage.getReference(location: "UsersImage");
public ProfilePatientFragment() {
}
public ProfilePatientFragment(ModelPatient patient) { this.patient = patient; }
@Override
public View onCreateView(@NotNull LayoutInflater inflater, ViewGroup container,
Bundle savedInstanceState) {
// Inflate the layout for this fragment
patientProfileBinding = FragmentPatientProfileBinding.inflate(getLayoutInflater(), container, attachToParent: false);
// Permissions to open Camera
ActivityCompat.requestPermissions(Objects.requireNonNull(getActivity()),
TODO
9: Git
Terminal
Database Inspector
Profiler
6: Logcat
Event Log
Layout Inspector
47:14 CRLF UTF-8 4 spaces master

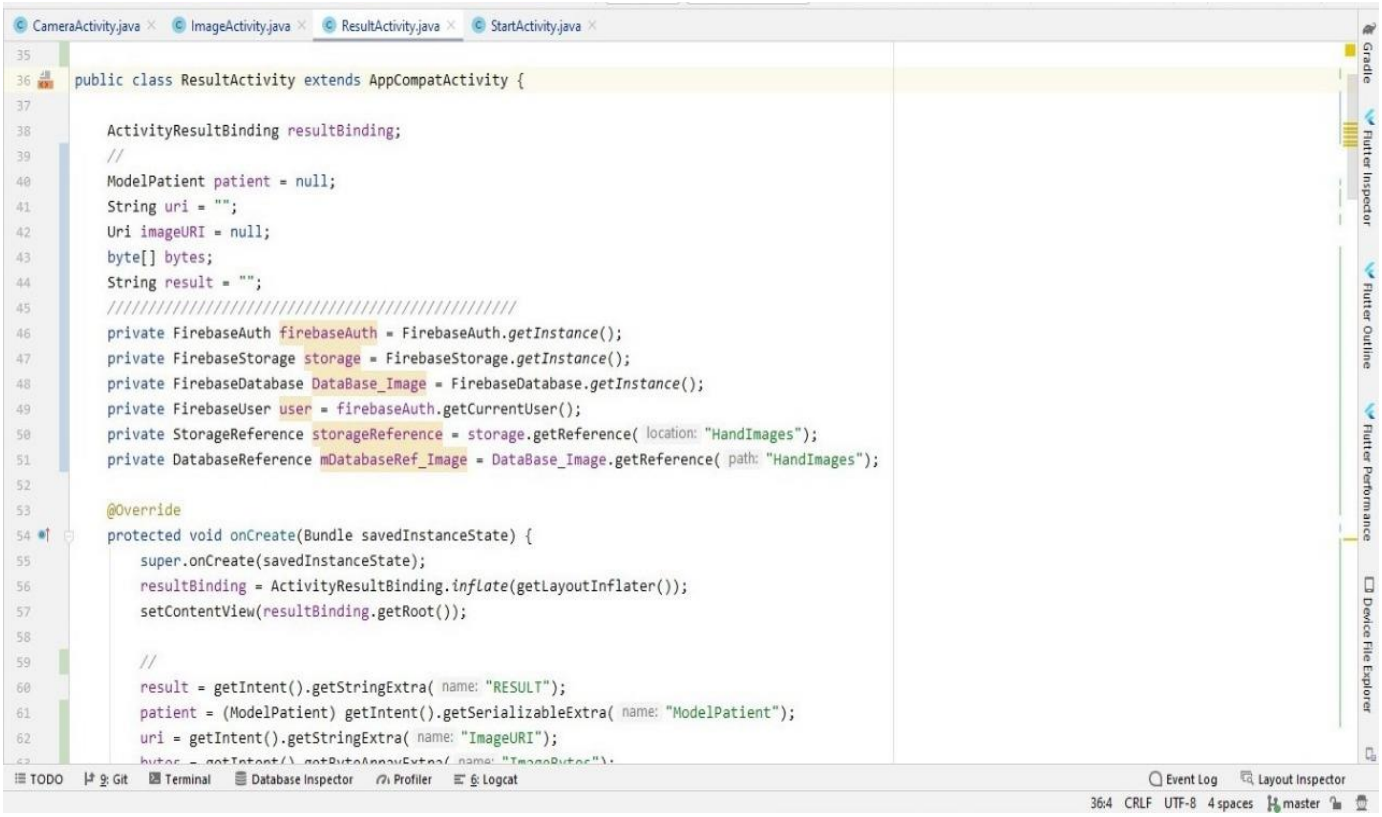

PatientActivity.java
HomePatientFragment.java
ProfilePatientFragment.java
ProgressPatientFragment.java
public class ProgressPatientFragment extends Fragment implements Adapter_All_Photos.OnItemClickListener {
FragmentPatientProgressBinding patientProgressBinding;
Adapter_All_Photos adapterAllPhotos;
ModelPatient patient;
List<ModelHandImage> mAllImageHands;
FirebaseAuth auth = FirebaseAuth.getInstance();
FirebaseUser user = auth.getCurrentUser();
FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference refHandImage = database.getReference(path: "HandImages");
public ProgressPatientFragment() {
}
public ProgressPatientFragment(ModelPatient patient) { this.patient = patient; }
@SuppressLint("UseCompatLoadingForDrawables")
@Override
public View onCreateView(@NotNull LayoutInflater inflater, ViewGroup container,
Bundle savedInstanceState) {
// Inflate the layout for this fragment
patientProgressBinding = FragmentPatientProgressBinding.inflate(getLayoutInflater(), container, attachToParent: false);
patientProgressBinding
.patientRecyclerViewPhotos
.setLayoutManager(new LinearLayoutManager(getActivity(), LinearLayoutManager.HORIZONTAL, reverseLayout: true));
TODO
9: Git
Terminal
Database Inspector
Profiler
6: Logcat
Event Log
Layout Inspector
40:14 CRLF UTF-8 4 spaces master



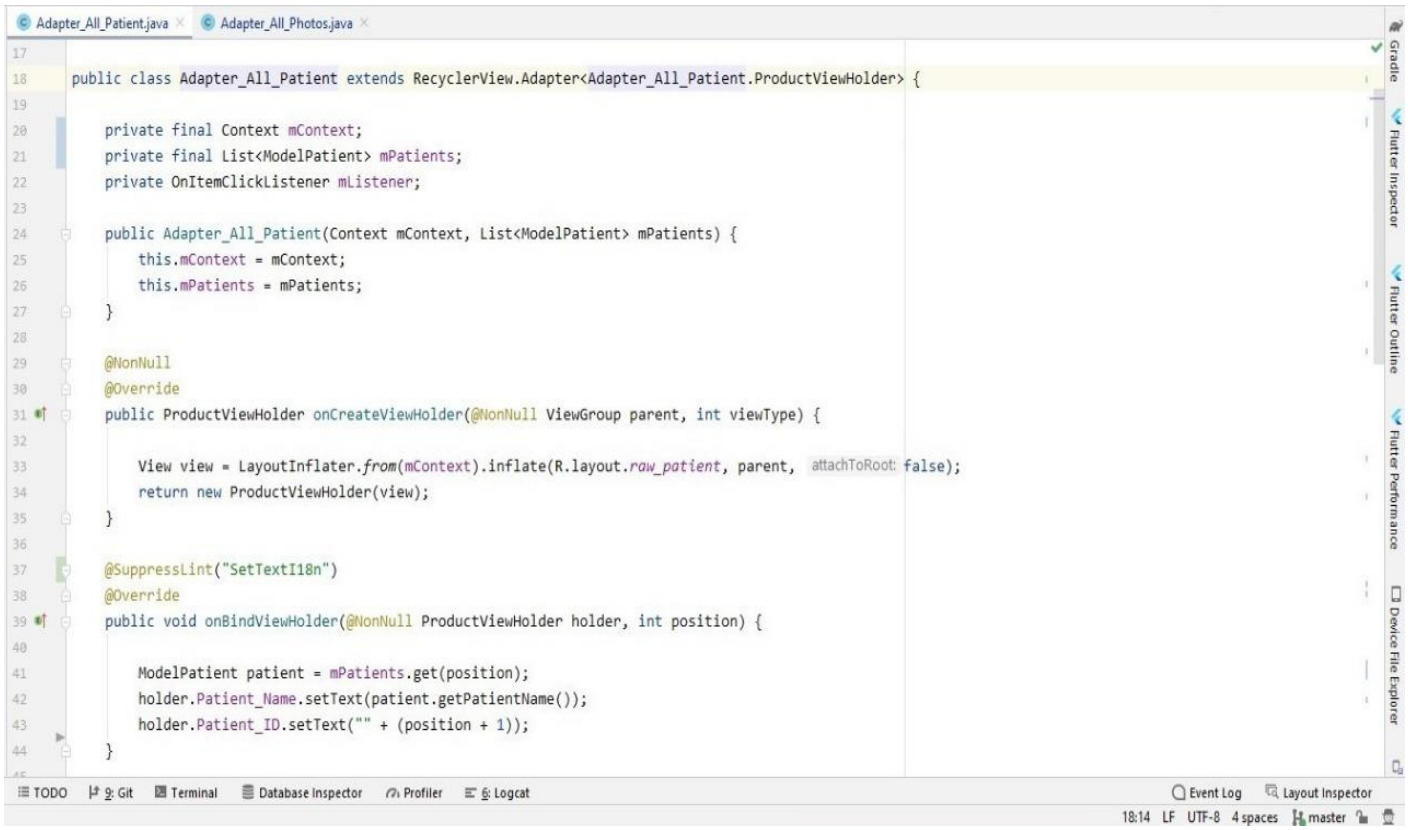
```
CameraActivity.java | ImageActivity.java | ResultActivity.java | StartActivity.java

public class CameraActivity extends AppCompatActivity implements SensorEventListener {

    // Check State ORIENTATIONS Of Output Image
    private static final SparseIntArray ORIENTATIONS = new SparseIntArray();
    private static final int REQUEST_CAMERA_PERMISSION = 200;
    public static Bitmap bitmap;
    public static byte[] bytes;

    static {
        ORIENTATIONS.append(Surface.ROTATION_0, 90);
        ORIENTATIONS.append(Surface.ROTATION_90, 0);
        ORIENTATIONS.append(Surface.ROTATION_180, 270);
        ORIENTATIONS.append(Surface.ROTATION_270, 180);
    }

    public ActivityCameraBinding cameraBinding;
    AlertDialog alertDialog;
    ////////////////////////////////
    SensorManager sensorManager;
    Sensor sensor;
    boolean isACCELEROMETERSenserAvailable, itIsNotFirstTime = false;
    float xCurrent, yCurrent;
    float xLast, yLast;
    float xDifference, yDifference;
    float sharkThreshold = 5f;
    Vibrator vibrator;
    ModelPatient patient = null;

TODO | 9: Git | Terminal | Database Inspector | Profiler | 6: Logcat | Event Log | Layout Inspector
72:14 CRLF UTF-8 4 spaces master
```



```
CameraActivity.java | ImageActivity.java | ResultActivity.java | StartActivity.java

public class ImageActivity extends AppCompatActivity {

    private static final int GALLERY_REQUEST = 1;
    private static final int REQUEST_CAMERA_PERMISSION = 200;

    Uri imageURI;
    Bitmap bitmap;
    ModelPatient patient = null;
    ActivityImageBinding imageBinding;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        imageBinding = ActivityImageBinding.inflate(getLayoutInflater());
        setContentView(imageBinding.getRoot());
        //
        patient = (ModelPatient) getIntent().getSerializableExtra( name: "ObjectPatient");
        //
        Toolbar toolbar = findViewById(R.id.toolBar);
        toolbar.setTitle("Choose Image");
        setSupportActionBar(toolbar);
        Objects.requireNonNull(getSupportActionBar()).setDisplayHomeAsUpEnabled(true);
        getSupportActionBar().setDisplayShowHomeEnabled(true);
        //

        selectImage();

TODO | 9: Git | Terminal | Database Inspector | Profiler | 6: Logcat | Event Log | Layout Inspector
42:14 CRLF UTF-8 4 spaces master
```



```
CameraActivity.java | ImageActivity.java | ResultActivity.java | StartActivity.java

public class ResultActivity extends AppCompatActivity {

    ActivityResultBinding resultBinding;
    //
    ModelPatient patient = null;
    String uri = "";
    Uri imageURI = null;
    byte[] bytes;
    String result = "";
    ////////////////////////////////////////////////
    private FirebaseAuth firebaseAuth = FirebaseAuth.getInstance();
    private FirebaseStorage storage = FirebaseStorage.getInstance();
    private FirebaseDatabase DataBase_Image = FirebaseDatabase.getInstance();
    private FirebaseUser user = firebaseAuth.getCurrentUser();
    private StorageReference storageReference = storage.getReference( location: "HandImages");
    private DatabaseReference mDatabaseRef_Image = DataBase_Image.getReference( path: "HandImages");

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        resultBinding = ActivityResultBinding.inflate(getLayoutInflater());
        setContentView(resultBinding.getRoot());

        //
        result = getIntent().getStringExtra( name: "RESULT");
        patient = (ModelPatient) getIntent().getSerializableExtra( name: "ModelPatient");
        uri = getIntent().getStringExtra( name: "ImageURI");
```



```
Adapter_All_Patient.java | Adapter_All_Photos.java

public class Adapter_All_Patient extends RecyclerView.Adapter<Adapter_All_Patient.ProductViewHolder> {

    private final Context mContext;
    private final List<ModelPatient> mPatients;
    private OnItemClickListener mListener;

    public Adapter_All_Patient(Context mContext, List<ModelPatient> mPatients) {
        this.mContext = mContext;
        this.mPatients = mPatients;
    }

    @NonNull
    @Override
    public ProductViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {

        View view = LayoutInflater.from(mContext).inflate(R.layout.raw_patient, parent,  attachToRoot: false);
        return new ProductViewHolder(view);
    }

    @SuppressLint("SetTextI18n")
    @Override
    public void onBindViewHolder(@NonNull ProductViewHolder holder, int position) {

        ModelPatient patient = mPatients.get(position);
        holder.Patient_Name.setText(patient.getPatientName());
        holder.Patient_ID.setText("" + (position + 1));
    }
```

CameraActivity.java | ImageActivity.java | ResultActivity.java | StartActivity.java

```
public class StartActivity extends AppCompatActivity {

    ActivityStartBinding startBinding;
    FirebaseAuth mAuth = FirebaseAuth.getInstance();
    FirebaseUser firebaseUser = mAuth.getCurrentUser();
    FirebaseDatabase database = FirebaseDatabase.getInstance();
    DatabaseReference databaseReference = database.getReference( path: "Users");

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        startBinding = ActivityStartBinding.inflate(getLayoutInflater());
        setContentView(startBinding.getRoot());

        if (firebaseUser != null && firebaseUser.getUid() != null) {
            startBinding.constraintLayoutStart.setBackgroundColor(getResources().getColor(R.color.white));
            startBinding.layoutWhoLogging.setVisibility(View.GONE);

            Thread welcome = new Thread(() -> {
                try {
                    Thread.sleep( millis: 1000);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }

                databaseReference
                        .child(firebaseUser.getUid())
```

TODO | 9: Git | Terminal | Database Inspector | Profiler | 6: Logcat | Event Log | Layout Inspector

27:14 CRLF UTF-8 4 spaces master

Adapter_All_Patient.java | Adapter_All_Photos.java

```
public class Adapter_All_Photos extends RecyclerView.Adapter<Adapter_All_Photos.ProductViewHolder> {

    private final Context mContext;
    private final List<ModelHandImage> handImages;
    private OnItemClickListener mListener;

    public Adapter_All_Photos(Context mContext, List<ModelHandImage> handImages) {
        this.mContext = mContext;
        this.handImages = handImages;
    }

    @NonNull
    @Override
    public ProductViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(mContext).inflate(R.layout.item_image, parent,  attachToRoot: false);
        return new ProductViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull ProductViewHolder holder, int position) {
        ModelHandImage imageHand = handImages.get(position);

        try {
            Picasso
                    .get() Picasso
                    .load( path: imageHand.getImageUri().trim() + "") RequestCreator
                    .fit()
```

TODO | 9: Git | Terminal | Database Inspector | Profiler | 6: Logcat | Event Log | Layout Inspector

20:14 LF UTF-8 4 spaces master

4.8 Some of model code.

Model1 LinearReg

In [54]:
```
final_rmse
```

Out[54]: 5.806596073732237

In [55]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[55]: array([5.43405513, 6.15663551])

In [56]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 4.4
Mean squared error(MSE) = 33.72
Median absolute error = 3.49
Explain variance score = 0.62
R2 score = 0.62
```

In [65]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [67]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
28.259528795582117
```

Model2 LinearReg(Without AN, RL)

In [23]:
```
final_rmse
```

Out[23]: 6.059408828885999

In [24]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[24]: array([5.69018043, 6.40739553])

In [25]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 4.68
Mean squared error(MSE) = 36.72
Median absolute error = 3.88
Explain variance score = 0.59
R2 score = 0.59
```

In [26]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [27]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
30.513436979107865
```

RidgeReg

In [19]:
```
final_rmse
```

Out[19]:
```
6.2022737117029845
```

In [20]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[20]:
```
array([5.79093513, 6.58797911])
```

In [21]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 4.67
Mean squared error(MSE) = 38.47
Median absolute error = 3.66
Explain variance score = 0.57
R2 score = 0.57
```

In [22]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [23]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
28.445101465005294
```

Lasso

In [20]:
```
final_rmse
```

Out[20]:
```
7.71344547756273
```

In [21]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[21]:
```
array([7.099645 , 8.2818792])
```

In [22]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 5.61
Mean squared error(MSE) = 59.5
Median absolute error = 4.43
Explain variance score = 0.33
R2 score = 0.33
```

In [23]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [24]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
32.70546729557658
```

Angle1 LinearReg

In [26]:
```
final_rmse
```

Out[26]:
```
4.867189323220134
```

In [27]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[27]:
```
array([3.05529539, 6.1680008 ])
```

In [28]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 3.42
Mean squared error(MSE) = 23.69
Median absolute error = 2.49
Explain variance score = 0.69
R2 score = 0.69
```

In [29]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [30]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
11.794598928094786
```

Angle2 LinearReg

In [25]:
```
final_rmse
```

Out[25]:
```
4.055621292072628
```

In [26]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[26]:
```
array([3.42025342, 4.604128  ])
```

In [27]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 3.17
Mean squared error(MSE) = 16.45
Median absolute error = 2.65
Explain variance score = 0.28
R2 score = 0.28
```

In [28]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [29]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
25.425447640946626
```

Angle3 LinearREG

In [22]:
```
final_rmse
```

Out[22]:
```
4.405278715627274
```

In [23]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[23]:
```
array([3.4210118 , 5.20669179])
```

In [24]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 3.2
Mean squared error(MSE) = 19.41
Median absolute error = 2.49
Explain variance score = -0.1
R2 score = -0.11
```

In [25]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [26]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
35.23855848282859
```

In [22]:
```
final_rmse
```

Out[22]:
```
3.6728393237241805
```

In [23]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[23]:
```
array([3.04021431, 4.21148363])
```

In [24]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 2.81
Mean squared error(MSE) = 13.49
Median absolute error = 2.21
Explain variance score = 0.54
R2 score = 0.54
```

In [25]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [26]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
23.63945584825226
```

Angle5 LinearReg

In [23]:
```
final_rmse
```

Out[23]: 7.463240694016823

In [24]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[24]: array([5.56497185, 8.96833383])

In [25]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 5.37
Mean squared error(MSE) = 55.7
Median absolute error = 3.73
Explain variance score = 0.52
R2 score = 0.52
```

In [26]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [27]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
20.917513835052908
```

Angle6 LinearReg

In [22]:
```
final_rmse
```

Out[22]: 4.179339235893598

In [23]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[23]: array([3.56859726, 4.7115673])

In [24]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 3.27
Mean squared error(MSE) = 17.47
Median absolute error = 2.76
Explain variance score = 0.28
R2 score = 0.27
```

In [25]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [26]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
25.110591294941404
```

In [51]:
```
final_rmse
```

Out[51]:
```
3.083109326984033
```

In [52]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[52]:
```
array([2.61290053, 3.49054108])
```

In [53]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 2.4
Mean squared error(MSE) = 9.51
Median absolute error = 1.83
Explain variance score = 0.42
R2 score = 0.42
```

In [54]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [55]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
23.034132656190828
```

Angle8 LinearReg

In [22]:
```
final_rmse
```

Out[22]: 4.083118802777449

In [23]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[23]: array([3.41378877, 4.65722713])

In [24]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 3.18
Mean squared error(MSE) = 16.67
Median absolute error = 2.48
Explain variance score = 0.51
R2 score = 0.5
```

In [25]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [26]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
21.298730058953467
```

In [48]:
```
final_rmse
```

Out[48]:
```
8.01802168143638
```

In [49]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[49]:
```
array([7.08610453, 8.85237064])
```

In [50]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 6.28
Mean squared error(MSE) = 64.29
Median absolute error = 5.3
Explain variance score = 0.5
R2 score = 0.5
```

In [51]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [52]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
22.3929801184978
```

All Angle2 LinearReg(R&L)

In [48]:
```
final_rmse
```

Out[48]: 4.545703032111956

In [49]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[49]: array([4.05496905, 4.98839233])

In [50]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 3.66
Mean squared error(MSE) = 20.66
Median absolute error = 3.13
Explain variance score = 0.17
R2 score = 0.17
```

In [51]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [52]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
28.991826923454276
```

In [46]:
```
final_rmse
```

Out[46]: 3.6767660167272274

In [47]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[47]: array([3.26296994, 4.04848661])

In [48]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 2.85
Mean squared error(MSE) = 13.52
Median absolute error = 2.18
Explain variance score = 0.28
R2 score = 0.28
```

In [49]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [50]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
26.30438798601308
```

In [47]:
```
final_rmse
```

Out[47]: 4.298930281654213

In [48]:
```
from scipy import stats

confidence = 0.95
squared_errors = (final_predictions - y_test) ** 2
np.sqrt(stats.t.interval(confidence, len(squared_errors) - 1,
                         loc=squared_errors.mean(),
                         scale=stats.sem(squared_errors)))
```

Out[48]: array([3.85555662, 4.7006687])

In [50]:
```
import sklearn.metrics as sm

print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, final_predictions), 2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, final_predictions), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, final_predictions), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, final_predictions), 2))
print("R2 score =", round(sm.r2_score(y_test, final_predictions), 2))
```

```
Regressor model performance:
Mean absolute error(MAE) = 3.4
Mean squared error(MSE) = 18.48
Median absolute error = 3.0
Explain variance score = 0.34
R2 score = 0.34
```

In [51]:
```
def mean_absolute_percentage_error(y_test, final_predictions):
    y_test, final_predictions = np.array(y_test), np.array(final_predictions)
    return np.mean(np.abs((y_test - final_predictions) / final_predictions)) * 100
```

In [52]:
```
print(mean_absolute_percentage_error(y_test, final_predictions))
```

```
23.639533509851248
```

Chapter Five

Conclusion and future work

5.1 Conclusions:

We decided to solve the problem that some of doctor's face in their work.

our solution will use a mobile application to capture an image of the hand pose and then analyze the image using AI-based pose estimation methods over time (i.e., several days) for the sake of reporting the recovery progress of finger nerves. As the degrees between two fingers increases, the recovery increases. For the problem, the automated solutions are totally new, as there is no software doing this task.

5.2 Future Work:

We are planning to develop our app in the future by adding:

- **Foot Pronation, Over-pronation & Supination detection:**

  Feet pose estimation while running can be used to detect foot over-pronation or supination. In this work, we propose measuring the degree of foot over-pronation or supination using an AI-based method of pose estimation. The proposed solution can help the end-user in several ways such as selecting proper footwear to avoid joints injuries and adjust the sportive equipment (e.g., treadmill) to fit the detected degree of foot over-pronation or supination.

6 References:

1.
2.
3.