

# **TRIAGE CLINIC**

## **ABSTRACT**

Clinics currently use a manual system for the management and maintenance of critical information. The current system requires numerous paper forms, with data stores spread throughout the clinic infrastructure. Often Information is incomplete or does not follow management standards. Forms are often lost in transit between departments requiring a comprehensive auditing process to ensure that no vital information is lost. To overcome this Problem we come up with a System Called “TRIAGE CLINIC” .It is an Mobile application. We develop the system using Android Programming. The system consists of Login Module, Add Patient details, Add queue details, Patient Report, Queue Display, Doctor Details. Triage Clinic designed for Any Clinic to replace their existing manual, paper based system. These services are to be provided in efficient, cost effective manner with the goal of reducing the time and resources currently required for such tasks. Now-a-days Application are widely used for many project areas. here we create an application for the small town hospitals to manage the patient details, queue, doctor details and patient reports. This project is developed to maintain the clinic management includes information about admit patients, outdoor patients (OPD) status, outdoor patients appointments. Maintaining these information manually is time consuming and not suitable for clinics where number of appointment increases.

## TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	<b>ABSTRACT</b>	
	<b>LIST OF FIGURES</b>	9
	<b>LIST OF SYMBOLS</b>	10
<b>1.</b>	<b>INTRODUCTION</b>	
	1.1.Problem Statement	11
	1.2.Motivation	11
	1.3.Objective	12
	1.3.1.Proposed System	12
	1.3.2.Advantages of Proposed system	13
<b>2.</b>	<b>TECHNOLOGIES LEARNT</b>	13
<b>3.</b>	<b>SYSTEM DESIGN</b>	
	3.1 System Architecture	15
	3.2 Module description	16
	3.3 System Specification	17
	3.3.1 Software Requirements	
	3..3.2 Hardware Requirements	
	3.4 Detailed Design	
	3.4.1 Use case Diagram	17
	3.4.2 Sequence Diagram	18
	3.4.3 Class Diagram	19
	3.4.4 Dataflow diagram	19
	3.4.5 Activity diagram	21

<b>4.</b>	<b>IMPLEMENTATION</b>	
	4.1 Implementation details	22
<b>5.</b>	<b>TEST RESULTS</b>	
	5.1 Test cases	26
<b>6.</b>	<b>RESULTS AND DISCUSSIONS</b>	30
<b>7.</b>	<b>CONCLUSION AND FUTURE WORK</b>	30
	7.1 Conclusion	
	7.2 Future Work	
<b>8.</b>	<b>REFERENCES</b>	

## LIST OF FIGURES

Figure No	Title	Page No
3.1	System Architecture	15
3.4.1	Use Case Diagram	17
3.4.2	Sequence Diagram	18
3.4.3	Class Diagram	19
3.4.4	Data flow Diagram Level-1,2,3	19
3.4.5	Activity Diagram	21

## LIST OF ABBREVIATIONS

ACRONYM	EXPANSION OPD
	OutDoor Patients
<b>JVM</b>	Java Virtual Machine
<b>SMS</b>	Short Message Sevice
<b>RAM</b>	Random Access Memory
<b>DFD</b>	Data Flow Diagram

## **1. INTRODUCTION**

### **1.1 PROBLEM STATEMENT**

Clinical Patient management system is enhanced from the traditional paper-based management system that has been using in the clinic. Based on the previous system, the patient who comes to the clinic for the first time is registered via the system. The assistant assist the patient by write down the personal detail in a form. The patient gets the treatment and information about the treatment is record in a file. The system manages the activities in the clinic but the previous system has cause problems to the user.

Problems of using paper to record down the records of patient:

- Only one copy, emergent consult problem
- Waste time to search the record
- Easy to lost record or duplicate record
- Waste money on purchase paper
- Waste space for store record

### **1.2 MOTIVATION**

These problems are so important because they will affect the operation of clinic cause decrease of patients visits, inefficiency and increase of cost. Clinical Patient management is developed to overcome the problems. The system has few modules such as patient registration, inventory module, medical certificate, disease history, patient record search, appointment, billing and reporting.

### **1.3 OBJECTIVE**

- The project aims to overcome the problem exists in the previous system.
- In order to overcome the problem exists in the previous system, we must determine the problems existing in previous system, find out the reason cause the problems in previous existing system and create a solution to solve the problems.
- Investigate on system/user request and define new requirements.
- To achieve this objective need to determine who the user is, understanding the user request, verify the request can be achieve or not.
- Make user easy to maintain record
- Determine what record is requiring in the system. All the records will be kept in database.
- Ensure the system useful to user as it help in daily activity in the clinic.

### **1.4 PROPOSED SYSTEM**

In the Proposed System Of Triage Clinic there are totally 7 modules. Login, Add Patient Details, Add Queue Details, Patient Report, Queue Display, Doctor Details, Payment Module And Signup and forgot page. This System reduce the time of patients on appointments and system is User friendly and easy to use. Clinic use paper for record the details of the patient. This System reduce the manual work and the System is very efficient. The Application includes:

- Maintaining Patient details.
- Providing Prescription, Precautions and Diet advice.
- Providing and maintaining all kinds of tests for a patient.
- Billing and Report generation.
- Advantages of proposed system:
- Patient Registration Details
- Medical Alert Details
- Doctor's Daily Schedule List

### **1.5 Advantages Of Proposed System**

- Patient can maintain their own profile by giving their respective details which is related to this criterion. Also there is a possibility that they can edit their own details in future if they want to change. Save lots of time for the doctor and patient, it saves the travelling time of patients etc.,
- With the help of Application the patient can book their appointment with the doctors through the mobile app, they don't have to go hospital for this. They can reschedule their appointments in case of unavailability. They can also have of chance of change to another doctor or else change the appointment to selected date with patient verification through mail or SMS. They can also view their different test report sent by the lab in the mobile phone at their home, they don't have to wait for the report at lab.
- Pay Patient Bill via Internet Banking, Credit Cards, Debit Cards and Cash. For In-Patient, There is an option to indicate the bed availability and available rooms at that current time.

## **2. TECHNOLOGIES LEARNT**

We develop an App using Android Programming in Eclipse Tool.

### **Android Technology Is Essential**

- While developing great applications and the softwares for Android, it is absolutely necessary that the right technology to be used from the beginning.
- we already know that the Android is one of the most popular mobile devices on the market. Right now, Android is most used technology, when it comes to traditional smart phones, and it is constantly growing and changing in response to its top market rival, Apple.
- For that reason we used the best and most popular technology will be implemented in developing the android application

### **Uses and importance of java in Android:**

- ❖ Java is both robust and stable.
- ❖ In fact, Java is enjoying a resurgence because it is central to so many mobile applications that are developed today.



- ❖ For most purposes, Java is the fastest and cleanest way to produce an interactive mobile experience.
- ❖ Java is the technology of choice for building applications using managed code that can execute on mobile devices.
- ❖ Java is secure (no threat to security because nothing gets executed outside the JVM).
- ❖ Object oriented paradigms.
- ❖ Rich set of core features (java's core features are complete and vast. Also they are regularly updated and maintained by the oracle).

## **Eclipse**

Eclipse is a development environment that is used by the vast majority of Android developers. By using Eclipse, we can ensure that your Android application runs according to all of the latest standards and best practices in the fast-evolving world of the Android application. Eclipse is still a viable alternative and if your software project is one that can be completed, tested, and implemented quickly, it's very possible that your whole project will be developed in Eclipse.

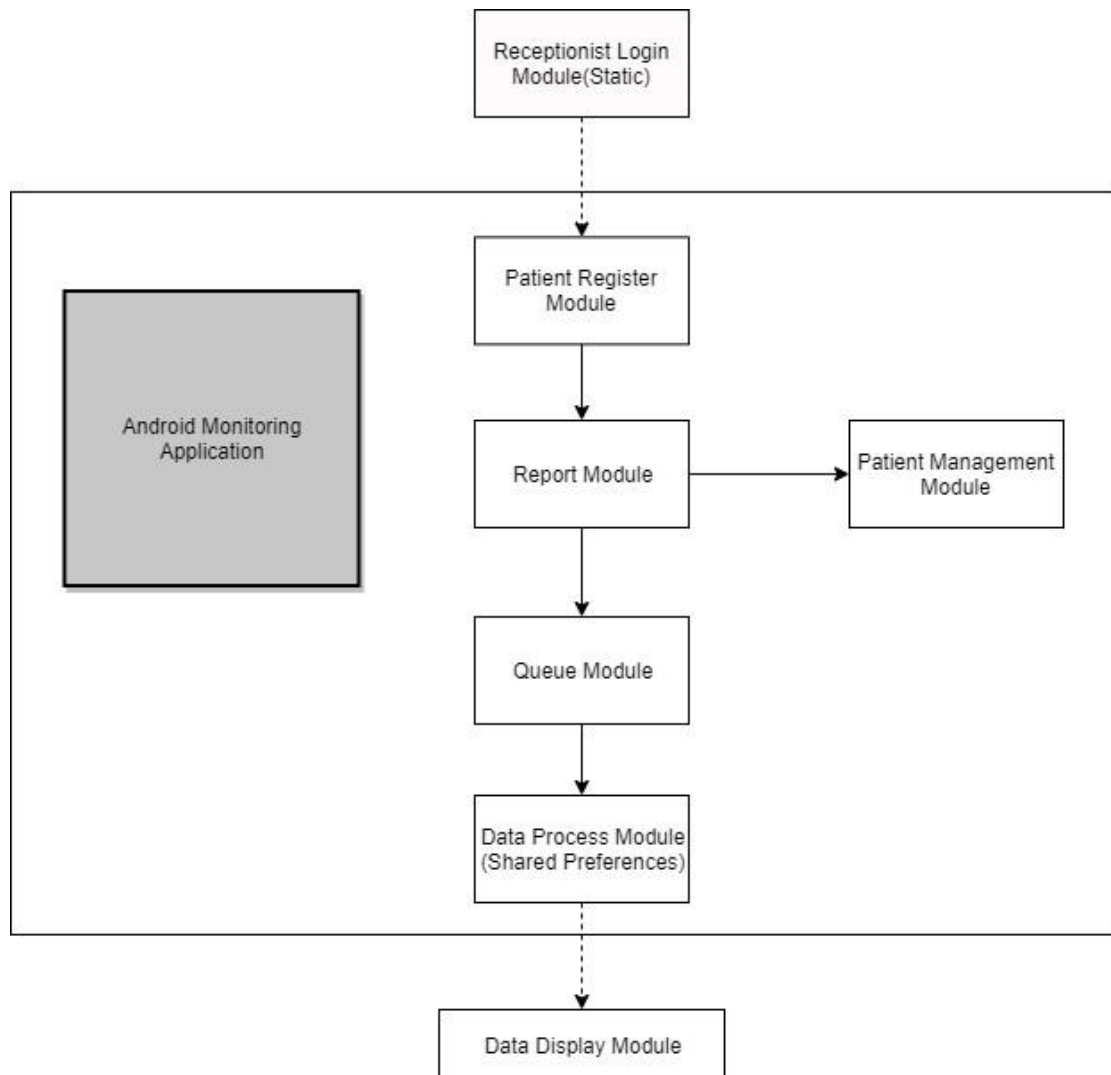
### **Uses and importance of Eclipse:**

- ❖ Reliable; you can expect that it will behave exactly as planned, and so will projects made in it.
- ❖ Familiar; developers have had time to get used to the interface and find elegant solutions in it.
- ❖ Compatible; unlike Android Studio, everything in Eclipse has been tested very extensively.

### 3. SYSTEM DESIGN

#### 3.1 SYSTEM ARCHITECTURE

#### 3.2



## **3.2 MODULE DESCRIPTION**

### **LOGIN PAGE**

Receptiniost wants to enter the username and password to access the attendance monitoring system. If the username and password matches they are successfully logged in. Otherwise it shows an error message.

### **PATIENT DETAILS**

Patient details consists of his personal details like patient name, id , Address , Mobile number, Blood Group, Medical Conditions etc. This will help to Know the doctors about the patient conditions.

### **PATIENT REPORT**

Patient report consists of Patient\_id, Name, Age, Height, weight, Blood Group, Last checkup date , Vital signs, diseases, and prescription.

### **DOCTOR DETAILS**

Doctor Details consists of information about the doctor like name, Mobile no, Available time, Appointment time, Specialist in etc.

### **QUEUE DETAILS**

In this module the receptionist add the patient name and the token number. If the patient visited the doctor they are going to delete from the queue

### 3.3 SYSTEM SPECIFICATION

#### SOFTWARE REQUIREMENTS

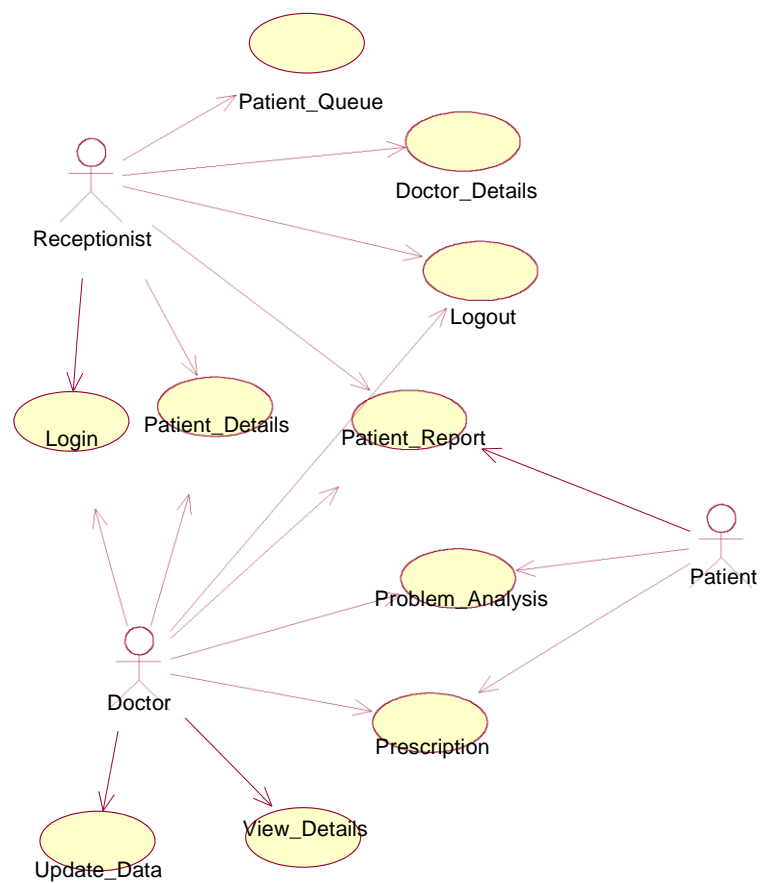
- Eclipse Tool
- Windows 10,8

#### HARDWARE REQUIREMENTS

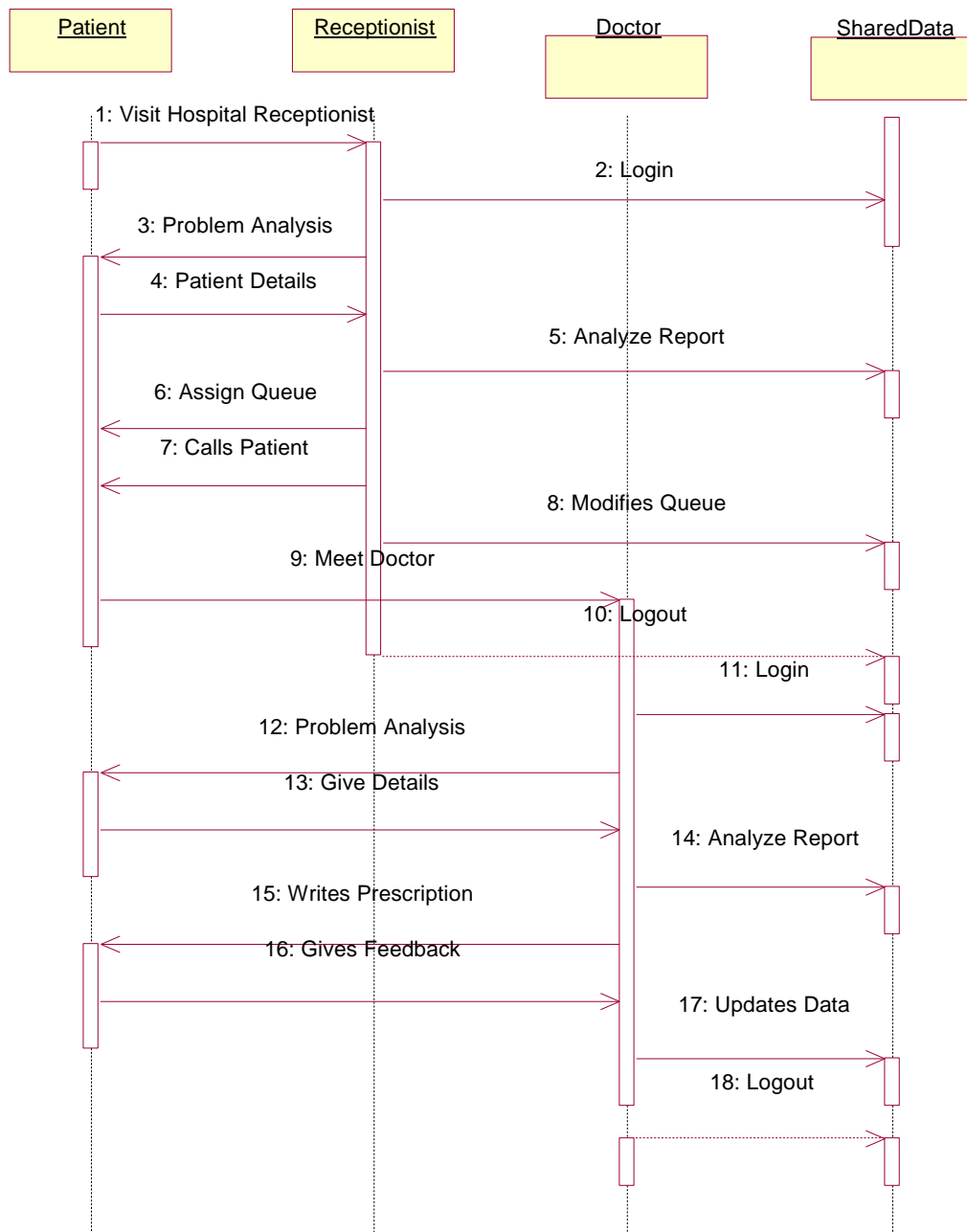
- 8.00 GB RAM
- 1 TB Hard Disk

### 3.4 DETAILED DESIGN

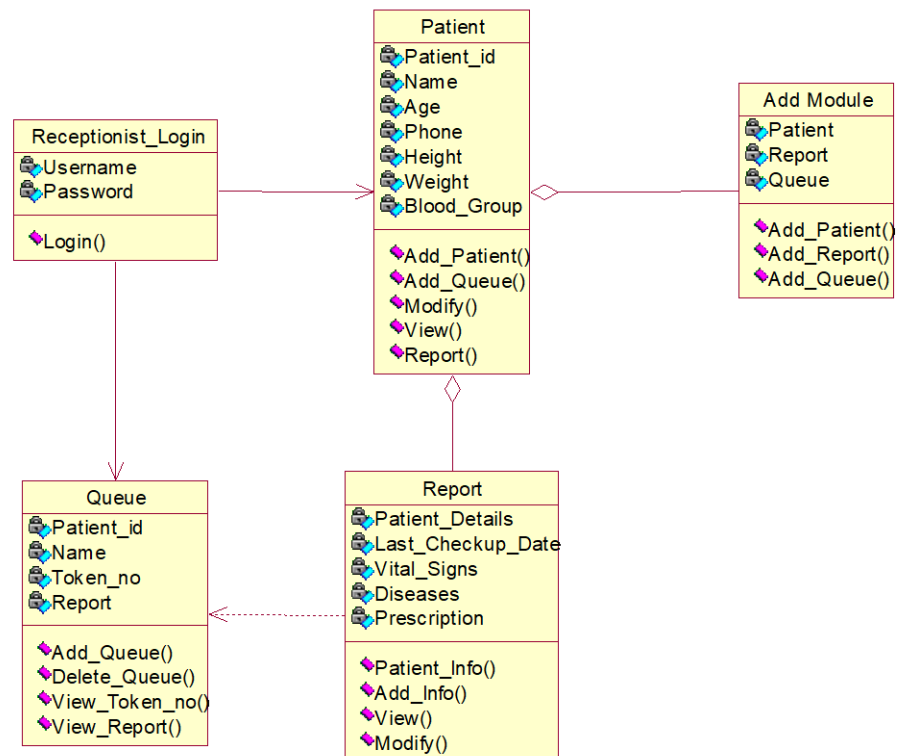
#### 3.4.1 USE CASE DIAGRAM



### 3.4.2 SEQUENCE DIAGRAM

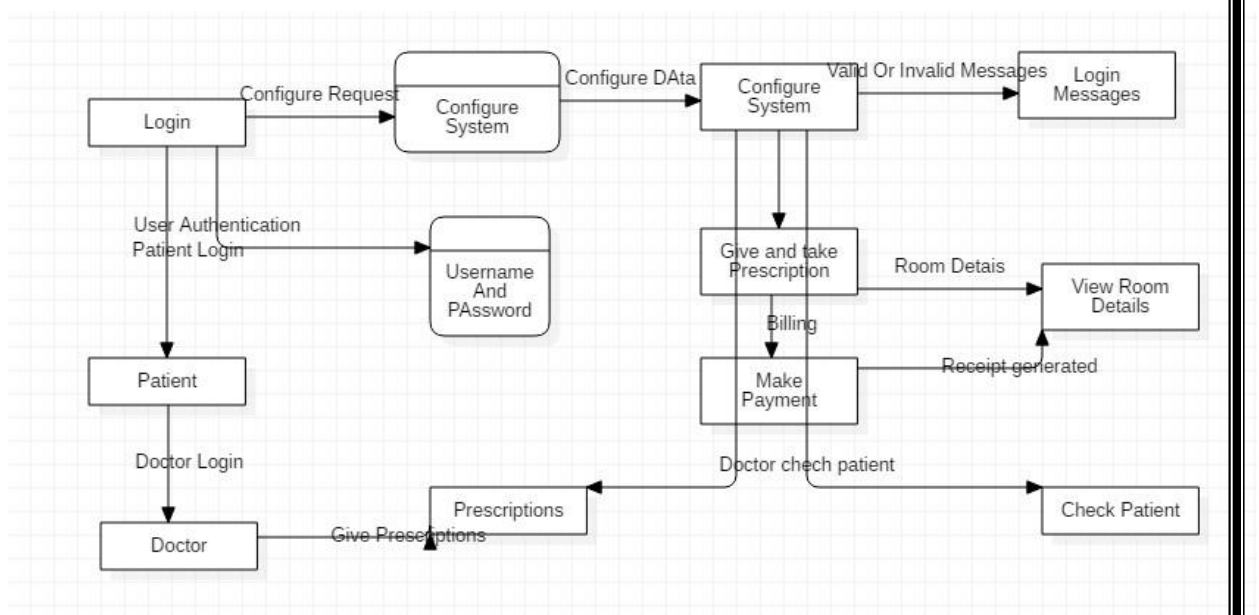


### 3.4.3 CLASS DIAGRAM

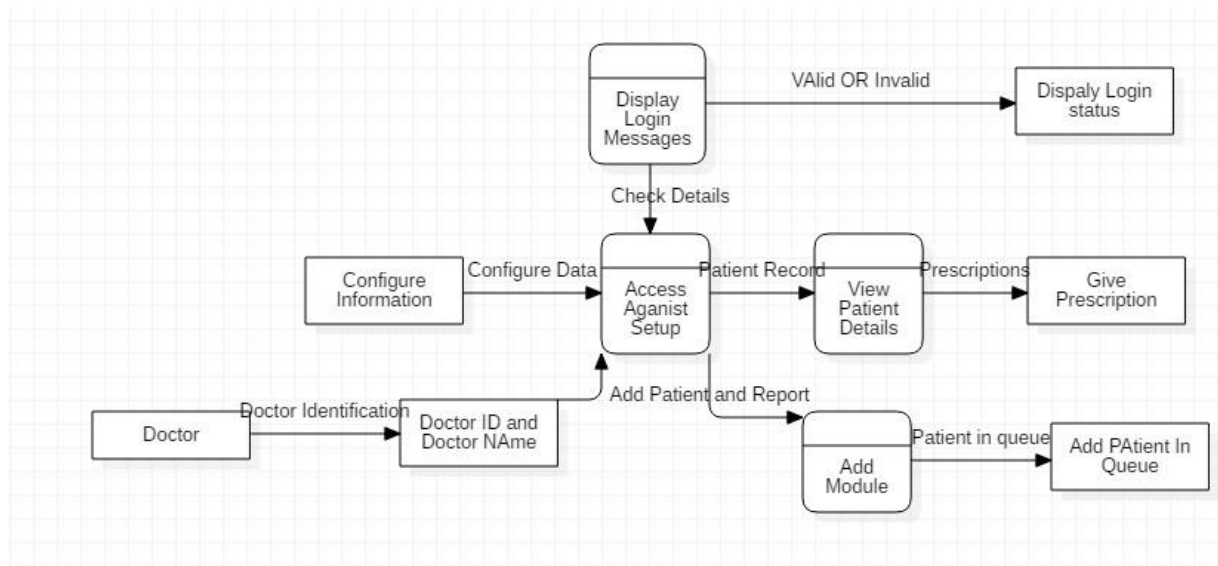


### 3.4.4 DATA FLOW DIAGRAM

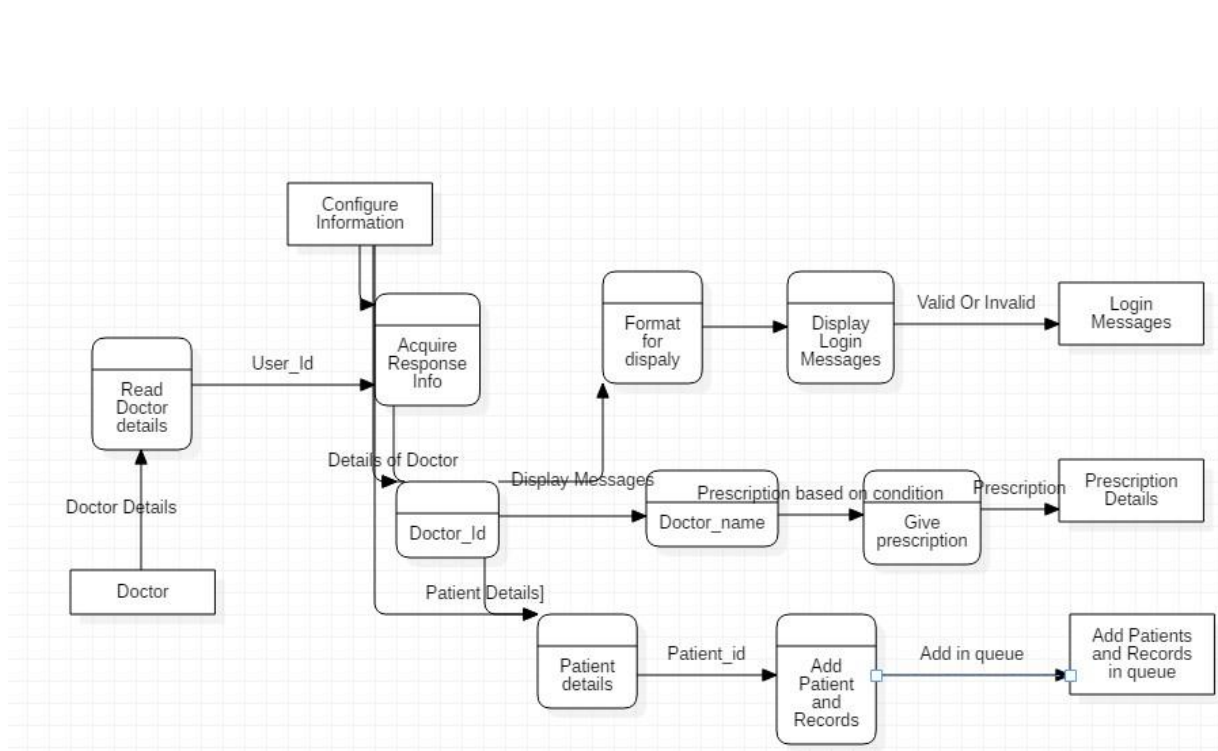
DFD 1:



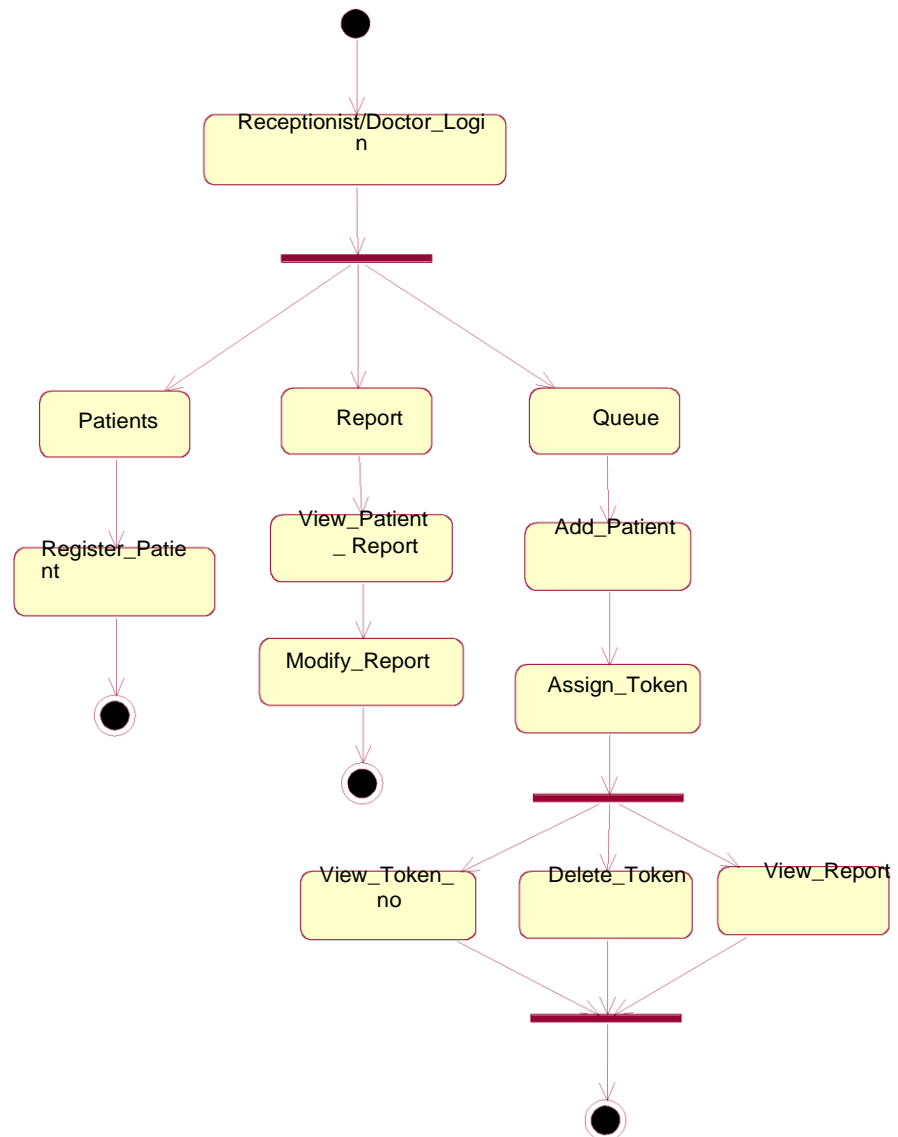
## DFD 2:



## DFD 3:



### 3.4.5 ACTIVITY DIAGRAM





## 4. IMPLEMENTATION

### 4.1 IMPLEMENTATION DETAILS

```
5. package com.example.hospital;
6.
7. import java.util.ArrayList;
8. import android.os.Bundle;
9. import android.app.Activity;
10.     import android.content.Intent;
11.     import android.content.SharedPreferences;
12.     import android.view.Menu;
13.     import android.view.View;
14.     import android.view.Window;
15.     import android.view.WindowManager;
16.     import android.widget.AdapterView;
17.     import android.widget.AdapterView.OnItemClickListener;
18.     import android.widget.ArrayAdapter;
19.     import android.widget.Button;
20.     import android.widget.EditText;
21.     import android.widget.Spinner;
22.     import android.widget.TextView;
23.     import android.widget.Toast;
24.     public class Queue_act extends Activity {
25.         TextView bac;
26.         Button adque;
27.         EditText checktime;
28.         Spinner sp;
29.         ArrayList<String> arr=new ArrayList<String>();
30.         @Override
31.         protected void onCreate(Bundle savedInstanceState)
32.         {
33.             super.onCreate(savedInstanceState);
34.             requestWindowFeature(Window.FEATURE_NO_TITLE);
35.             getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
36.                 WindowManager.LayoutParams.FLAG_FULLSCREEN);
37.             setContentView(R.layout.activity_queue_act);
38.             bac=(TextView)findViewById(R.id.bacak);
39.             adque=(Button)findViewById(R.id.queueadd);
40.             checktime=(EditText)findViewById(R.id.checkintime);
41.             sp=(Spinner)findViewById(R.id.spname);
42.             SharedPreferences
43.             sp21=getSharedPreferences("register1",MODE_PRIVATE);
44.             SharedPreferences
45.             sp2=getSharedPreferences("register",MODE_PRIVATE);
46.             SharedPreferences
47.             sp3=getSharedPreferences("register3",MODE_PRIVATE);
48.             SharedPreferences
49.             sp4=getSharedPreferences("register4",MODE_PRIVATE);
50.             SharedPreferences
51.             sp5=getSharedPreferences("register5",MODE_PRIVATE);
```

```

45.         SharedPreferences
    sp6=getSharedPreferences("register5",MODE_PRIVATE);
46.         SharedPreferences
    sp7=getSharedPreferences("register5",MODE_PRIVATE);
47.         String snam11=sp21.getString("patname",
    null);
48.         String snam1=sp2.getString("patname", null);
49.         String snam2=sp3.getString("patname", null);
50.         String snam3=sp4.getString("patname", null);
51.         String snam4=sp5.getString("patname", null);
52.         String snam5=sp6.getString("patname", null);
53.         String snam6=sp7.getString("patname", null);

54.         if(snam11!=null)
55.         {
56.             arr.add(snam11);
57.         }
58.         if(snam1!=null)
59.         {
60.             arr.add(snam1);
61.         }
62.         if(snam2!=null)
63.         {
64.             arr.add(snam2);
65.         }
66.         if(snam3!=null)
67.         {
68.             arr.add(snam3);
69.         }
70.         if(snam4!=null)
71.         {
72.             arr.add(snam4);
73.         }
74.         if(snam5!=null)
75.         {
76.             arr.add(snam5);
77.         }
78.         if(snam6!=null)
79.         {
80.             arr.add(snam6);
81.         }
82.         final ArrayAdapter<String> ada=new
    ArrayAdapter<String>(this,android.R.layout.simple_spinner_item,
    arr);
83.         ada.setDropDownViewResource(android.R.layout.simple_spinner_dro
    pdown_item);
84.         sp.setAdapter(ada);
85.         sp.setOnItemClickListener(new
    OnItemSelectedListener() {
86.             @Override
87.             public void onItemClick(AdapterView<?>
    parent, View arg1, int pos,
88.                 long arg3) {
89.                 String
    s=parent.getItemAtPosition(pos).toString();
90.                 Toast.makeText(parent.getContext(),"you
    selected "+s, Toast.LENGTH_LONG).show();
91.             }
92.             @Override

```

```

93.                public void onNothingSelected(AdapterView<?>
    arg0) {
94.                    }
95.                });
96.                bac.setOnClickListener(new View.OnClickListener()
    {
97.                    @Override
98.                    public void onClick(View arg0) {
99.                        Intent ne=new
    Intent(Queue_act.this,Appear_act.class);
100.                        startActivity(ne);
101.                    }
102.                });
103.                adque.setOnClickListener(new
    View.OnClickListener() {
104.                    @Override
105.                    public void onClick(View arg0) {
106.                        String
    s1=sp.getSelectedItemAt().toString();
107.                        String
    s=checktime.getText().toString();
108.                        int a=Integer.parseInt(s);
109.                        if(a==1)
110.                        {
111.                            SharedPreferences
    spque1=getSharedPreferences("queue1",MODE_PRIVATE);
112.                            SharedPreferences.Editor
    edque1=spque1.edit();
113.
    edque1.putString("User",s1);
114.
    edque1.putString("check",s);
115.
    edque1.commit();

116.                        }
117.                        else if(a==2)
118.                        {
119.                            SharedPreferences
    spque2=getSharedPreferences("queue2",MODE_PRIVATE);
120.                            SharedPreferences.Editor
    edque2=spque2.edit();
121.
    edque2.putString("User",s1);
122.
    edque2.putString("check",s);
123.
    edque2.commit();

124.                        }
125.                        else if(a==3)
126.                        {
127.                            SharedPreferences
    spque3=getSharedPreferences("queue3",MODE_PRIVATE);
128.                            SharedPreferences.Editor
    edque3=spque3.edit();
129.
    edque3.putString("User",s1);
130.
    edque3.putString("check",s);
131.
    edque3.commit();

```

```

132.                }
133.                else if (a==4)
134.                {
135.                    SharedPreferences
spque4=getSharedPreferences("queue4",MODE_PRIVATE);
136.                    SharedPreferences.Editor
edque4=spque4.edit();
137.                    edque4.putString("User",s1);
138.                    edque4.putString("check",s);
139.                    edque4.commit();

140.                }
141.                else if (a==5)
142.                {
143.                    SharedPreferences
spque5=getSharedPreferences("queue5",MODE_PRIVATE);
144.                    SharedPreferences.Editor
edque5=spque5.edit();
145.                    edque5.putString("User",s1);
146.                    edque5.putString("check",s);
147.                    edque5.commit();

148.                }
149.                else if (a==6)
150.                {
151.                    SharedPreferences
spque6=getSharedPreferences("queue6",MODE_PRIVATE);
152.                    SharedPreferences.Editor
edque6=spque6.edit();
153.                    edque6.putString("User",s1);
154.                    edque6.putString("check",s);
155.                    edque6.commit();

156.                }
157.                else if (a==7)
158.                {
159.                    SharedPreferences
spque7=getSharedPreferences("queue7",MODE_PRIVATE);
160.                    SharedPreferences.Editor
edque7=spque7.edit();
161.                    edque7.putString("User",s1);
162.                    edque7.putString("check",s);
163.                    edque7.commit();

164.                }
165.                else
166.                {
167.                    SharedPreferences
spque8=getSharedPreferences("queue8",MODE_PRIVATE);
168.                    SharedPreferences.Editor
edque8=spque8.edit();

```

```

169.         edque8.putString("User",s1);
170.         edque8.putString("check",s);
171.                                     edque8.commit();

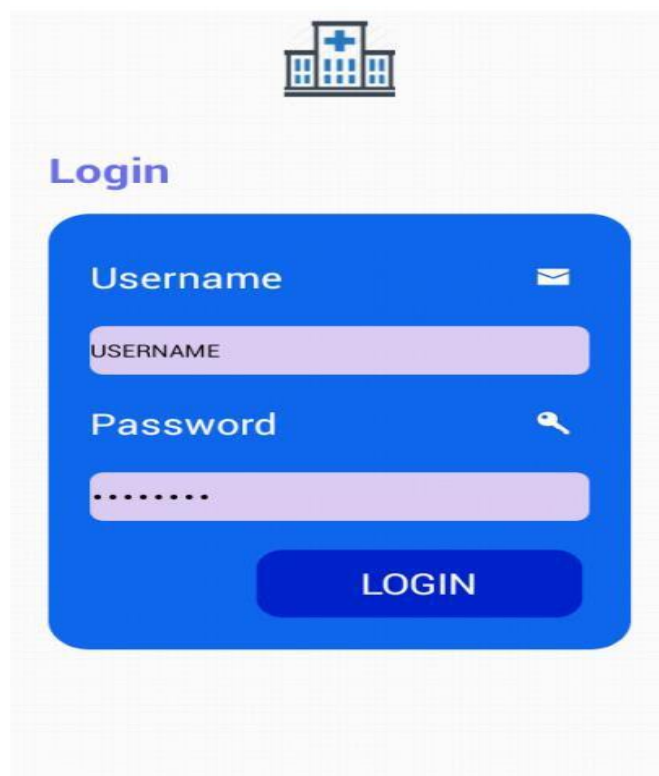
172.                                     }
173.                                     Toast.makeText(Queue_act.this,
"Token Number Added Sucessfully",Toast.LENGTH_SHORT).show();
174.                                     Intent ne=new
Intent(Queue_act.this,Appear_act.class);
175.                                     startActivity(ne);
176.                                     }
177.                                     });
178.                                     }
179.                                     @Override
180.                                     public boolean onCreateOptionsMenu(Menu menu) {
181.                                     getMenuInflater().inflate(R.menu.activity_queue_act,
menu);

182.                                     return true;
183.                                     }
184.                                     }
185.

```

## **5.TEST RESULTS**

### **5.1 TEST CASES**



## Register Patient

Patient Id

Name

Age

Phone Number

Height

Weight

Blood Group

Register Patient

Hospital




Patients

Reports

siva

prabhakaran



Token Number

Back

Patient name

siva


siva

prabhakaran

Add

Hospital

+



Patients

Reports

siva

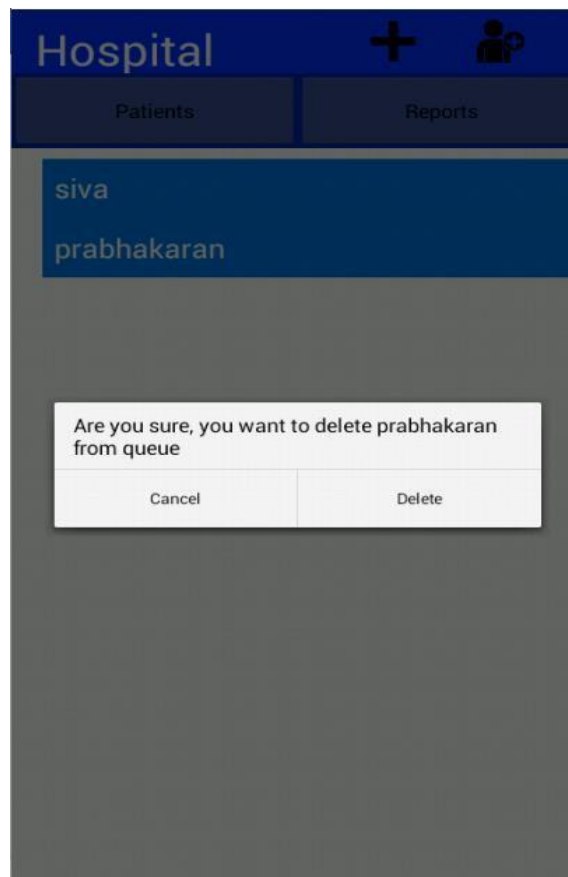
prabhakaran

prabhakaran

Token Number 1

Delete

Patient report



The screenshot shows a mobile application interface for a patient report. At the top, there is a white header with a hospital icon and the text "Patient Report". Below the header, there is a blue rounded rectangle containing a form. The form has the following fields:

- Patient Id: 2
- Name: Prabhakaran
- Age: 24
- Phone Number: 9087564312
- Height: 5.6
- weight: 56
- Blood Group: O+



weight  
56

Blood Group  
O+

Last CheckUp Date  
23/09/19

vital Signs  
fever

Diseases  
Just fever

Prescription  
Doloplus

Back

## **6 RESULTS AND DISCUSSIONS**

I developed an app for Clinic called Triage Clinic using the Android Programming. It is very efficient for the patient and saves the time of the patient. Doctor can view the patient report it consists of Height, Weight, Blood Pressure etc. According to that they can give their prescription. This app can fit for any clinic and reduce the manual work. Data stores in a secured manner and it is available to the authenticated person.

## **7 CONCLUSION AND FUTURE WORK**

### **7.1 CONCLUSION**

Most health-related appointments with telemedicine technology are "almost" not far from one day, and the election is ultimately more than an exception. Telemedicine is an exciting technology and promises to transform healthcare for the benefit of all. As the population grows faster than the optimal development of available physicians and facilities (institutional beds, hearing labs, day care centers, etc.), this

technique should be adapted to ensure that all caregivers are minimally receptive. Provided, or better. and more people in an organization will always be able to physically visit, regardless of providers and their happy where patients "go" from, and need constant monitoring in search of effective measures and, of course , To lead a situation that they do not want only to visit their doctors. This will help solve many problems currently challenging the system. Facilities will reduce the burden on patients, doctors will have to travel less, and more time and effort will be devoted to patients who need such care. Telemedicine will definitely improve communication and satisfaction. Now put pressure on yourself. It is not easy to change anything. Replacing the old order and using the new one can be difficult and painful, and is a process that is initially forced to deal with varying degrees of resistance. However, it is very unwise to keep things "fresh". Once rotted, things get "old" and soon all 112 go bad, which is undesirable at any time. Consequently, upgrades are necessary. However, to make it a common place, a long pregnancy is required. This is particularly the case in the health industry, where ideas and ways of doing things are protected from jealousy and strongly opposed.

## **7.2FUTURE WORK**

As far as our project is concern , we have just finished our front end of the project that is in the form of a mobile application. In the future, we have planned to develop a database and merge the database with the front end application.

## **8. REFERENCES**

1. <https://codelabs.developers.google.com/android-training/>
2. <https://developer.android.com/>