Documentation of the project CLOUDOC

ABSTRACT

The most widely discussed problem in any industry is security

breach, so it’s better to have a personalized system for our work.

As we are well aware about the health camps that take place in

rural areas, but unfortunately they don’t have proper system and

resources to manage the system, so we provide them a system

where they can manage the records of the patients without any

possibility of security breach, and effectively send the details of

the patient to them.

**Project Title**: CLOUDOC

Date: 28 MAY 2019 Prepared By: 1 JUNE 2019

**Project Justification:**

This project is typically building GUI based application, so that

admin, nurses and doctors don’t face any kind of irregularities

regarding the system.

**Project Requirements:**

Java jdk swing GUI

Cloud instance

Secured database

Checking the availability of the doctors

Cross checking the doctor’s details

ACKNOWLEDGEMENT

We would like to express my deepest appreciation to all

those who provided us the possibility to complete this

project. A special gratitude we give to our project trainer

Mr. BINAYAK , whose contribution in stimulating

Suggestions and encouragement.

We are highly indebted to “ASPIRE VISION” for their

Guidance and constant supervision as well as providing

Necessary information regarding the project &amp; also for

their support.

However, it would not have been possible without the

kind support and help of many individuals and

organization. I would like to extend my sincere thanks to

all of them.

INTRODUCTION

**PURPOSE:**

The main purpose of software requirements specification document is to describe the precise manner of all the

capabilities that will provided by the software application

system ”CLOUDOC” and external requirements for the health

camps. It also describes the interfaces for the system.

**SCOPE:**

The proposed software product is the CLOUDOC. This system will

be used in any hospital, clinic, camps, dispensary to get

information from the patient and then storing the data for future

usages.

The current system which is in use is a traditional based system

or paper based system. It is too slow and cannot provide update

list of patients within a reasonable timeframe. The intention of

the system is to reduce overtime pay and increase the number of

patients that can be treated accurately.

**MODULES:**

The cloudoc project has been divided into two modules. They are

 Admin module

 Doctor module

Admin module:

 Entering patient’s data.

 Getting corresponding doctors details.

 Sending patients information to doctor.

 Checking whether the treatment is done or not.

**Doctor module:**

 Checking whether they got notification or not.

 To agree the rules.

 Giving virtual treatment/precautions.

**Software Design**

INTRODUCTION

HEALTH CAMP system will become an important tool in the area of ensuring security of the client, this system allows us to

* Create two files about the patient details one is disposable and another one is non disposable.

In disposable files the patient details will always be rewritten i.e. the patient details once sent out in email, it will be again a normal file.

In non-disposable files, the patient details will be there permanently, for data storage data and for future sake.

* Ensuring effective treatment:

After the mail is sent the doctor receives it along with it doc receives a unique token number, which is very important, because once the doctor gives treatment via text he has to enter this token number in order to match with the same patient.

* Rules and regulations:

Once the doctor checks for the notifications, he has to agree to the terms and conditions, to give treatment in mentioned time or else he will be blocked away from the list of doctors.

* Facilities for time management:

Both doctors and admin has to benefit from the system so that their precious time is saved. So we avail an option to check if doc received any notification yet from admin and admin a check if he received any notifications from doctors about patient treatment.

We have an option called check in both the domains of the doctor and admin.

DATA FLOW DIAGRAMS

**SUMMARY**

We now understand the effectiveness of security and honesty we get from using this application, and to sum up we can tell that

**Accuracy:** - The level of accuracy in the proposed system will be higher. All operation would be done correctly and it ensures that whatever information is coming from the centre is accurate.

**Reliability:** - The reliability of the proposed system will be high due to the above stated reasons. The reason for the increased reliability of the system is that now there would be proper storage of information.

**No Redundancy:** - In the proposed system utmost care would be that no information is repeated anywhere, in storage or otherwise. This would assure economic use of storage space and consistency in the data stored.

**Immediate retrieval of information**: - The main objective of proposed system is to provide for a quick and efficient retrieval of information. Any type of information would be available whenever the user requires.

**Immediate storage of information**: - In manual system there are many problems to store the largest amount of information.

**Easy to Operate**: - The system should be easy to operate and should be such that it can be developed within a short period of time and fit in the limited budget of the user.

**Software Code**

**INTRODUCTION**

Our java code included

* using loops
* callings functions
* getting input from user
* Designing the GUI using swing
* File system

**CODE**

import java.awt.\*;

import javax.swing.\*;

import java.awt.event.\*;

import java.io.\*;

import java.util.\*;

import java.util.\*;

import java.io.File;

import java.io.FileNotFoundException;

import java.util.Scanner;

import java.io.\*;

import java.util.Scanner;

public class gui{

public static void main(String[] args) {

Frame f=new Frame("cloudoc");

JButton b2=new JButton("Doctor");

b2.setBounds(600,300,200,30);

JButton b3=new JButton("Admin");

b3.setBounds(600,100,200,30);

JButton b6=new JButton("OK");

b6.setBounds(150,100,200,30);

JButton b5=new JButton("Open");

b5.setBounds(800,100,200,30);

JButton b7=new JButton("Submit");

b7.setBounds(100,600,200,30);

JTextField te=new JTextField("");

te.setBounds(600,150, 200,30);

JTextField to=new JTextField("Enter token num");

to.setBounds(100,550, 200,30);

b3.setBounds(600,350,200,30);

f.add(b2);f.add(b3);

b2.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e1) {

final Choice c1=new Choice();

c1.setBounds(600,100, 200,75);

c1.add("Dermatolagist");

c1.add("Gynacolagist");

c1.add("Dentist");

c1.add("Pedeatristian");

c1.add("Opthomolagist");

c1.add("Oncolagist");

f.add(c1);f.add(te);

JButton b4=new JButton("check");

b4.setBounds(600,200,200,30);

f.add(b4);

JTextField tr=new JTextField("Welcome to CLOUDOC");

tr.setBounds(100,75, 400,30);

f.add(tr);

JTextField trr=new JTextField("PLEASE ENTER YOUR TREATEMENT HERE");

trr.setBounds(100,150, 400,400);

f.add(trr);

b4.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e1) {

String word=c1.getItem(c1.getSelectedIndex());

try{

File file = new File("G:\\admin.txt");

Scanner scan = new Scanner(file);

while(scan.hasNext()){

String s=scan.nextLine();

if(s.contains(word)){

te.setText("You have a notification");

f.add(tr); f.add(b6); f.add(b7); f.add(to);

tr.setText("Rules and policies: 1.Give the treatement atleast before 1 hour of zoning notification");

}

else {te.setText("no notification yet");}

}

scan.close();}catch(Exception e){System.out.print(e);}

}});

b6.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e1) {

b7.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e1) {

String treat =trr.getText();

String tok=to.getText();

try{

BufferedWriter fr=new BufferedWriter(new FileWriter("G:\\doc.txt"));

fr.write(tok+"\r\n");

fr.write(treat+"\r\n");

fr.close();

}catch(Exception e){System.out.print("Error");}

}});

}});

}});

b3.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e1) {

final Choice ch=new Choice();

ch.setBounds(600,150,200,75);

ch.add("FEMALE");

ch.add("MALE");

f.add(ch);

final Choice c=new Choice();

c.setBounds(600,100, 200,75);

c.add("Dermatolagist");

c.add("Gynacolagist");

c.add("Dentist");

c.add("Pedeatristian");

c.add("Opthomolagist");

c.add("Oncolagist");

f.add(c);

Label l1=new Label("Patient Name");

Label l2=new Label("Age");

Label l3=new Label("Token");

Label l4=new Label("Blood group");

Label l5=new Label("height ");

Label l6=new Label("weight");

Label l7=new Label("Symptoms");

Label l8=new Label("BP");

Label l9=new Label("Pulse rate");

JTextField tf=new JTextField("");

JTextField t1=new JTextField("");

JTextField t2=new JTextField("");

JTextField t3=new JTextField("");

JTextField t4=new JTextField("");

JTextField t5=new JTextField("");

JTextField t6=new JTextField("");

JTextField t7=new JTextField("");

JTextField t8=new JTextField("");

JTextField t9=new JTextField("");

JTextField t10=new JTextField("");

tf.setBounds(250,50, 200,30);

t1.setBounds(250,100,200,30);

t2.setBounds(250,150,200,30);

t3.setBounds(250,200,200,30);

t4.setBounds(250,250,200,30);

t5.setBounds(250,300,200,30);

t6.setBounds(250,350,200,30);

t7.setBounds(250,400,200,30);

t8.setBounds(250,450,200,30);

t9.setBounds(250,500,200,30);

l1.setBounds(50,50,200,30);

l2.setBounds(50,100,200,30);

l3.setBounds(50,150,200,30);

l4.setBounds(50,200,200,30);

l5.setBounds(50,250,200,30);

l6.setBounds(50,300,200,30);

l7.setBounds(50,350,200,30);

l8.setBounds(50,400,200,30);

l9.setBounds(50,450,200,30);

JButton b=new JButton("Submit");

b.setBounds(50,600,200,30);

JButton b1=new JButton("Send");

b1.setBounds(50,650,200,30);

TextField tok=new TextField("");

tok.setBounds(300,600, 200,30);

JButton bc=new JButton("Check");

bc.setBounds(300,650,200,30);

Hashtable<String,String> map=new Hashtable<String,String>();

map.put("Dermatolagist","hakijune@gmail.com");

map.put("Gynacolagist","veeralluri@yahoo.com");

map.put("Opthomolagist","harikaalluri16@gmail.com");

map.put("Dentist","alluri\_harika@srmap.edu.in");

map.put("pediatristian","veeralluri@yahoo.com");

map.put("Oncolagist","harikaalluri16@gmail.com");

b.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e3) {

String dat = c.getItem(c.getSelectedIndex());

try{

BufferedWriter fr=new BufferedWriter(new FileWriter("G:\\admin.txt",true));

fr.write(dat+"\r\n");

fr.close();

}catch(Exception e){System.out.print("Error");}

}

});

b.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e1) {

String s1=tf.getText();

String s2=t1.getText();

String s3=t2.getText();

String s4=t3.getText();

String s5=t4.getText();

String s6=t5.getText();

String s7=t6.getText();

String s8=t7.getText();

String s9=t8.getText();

try{

BufferedWriter fr=new BufferedWriter(new FileWriter("G:\\patient.txt"));

fr.write(s1+"\r\n");

fr.write(s2+"\r\n");

fr.write(s3+"\r\n");

fr.write(s4+"\r\n");

fr.write(s5+"\r\n");

fr.write(s6+"\r\n");

fr.write(s7+"\r\n");

fr.write(s8+"\r\n");

fr.write(s9+"\r\n");

fr.close();

}catch(Exception e){System.out.print("Error");}

}

});

b.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e11) {

String s1=tf.getText();

String s2=t1.getText();

String s3=t2.getText();

String s4=t3.getText();

String s5=t4.getText();

String s6=t5.getText();

String s7=t6.getText();

String s8=t7.getText();

String s9=t8.getText();

try{

BufferedWriter fr=new BufferedWriter(new FileWriter("G:\\patie.txt",true));

fr.write("NAME: "+s1+"\r\n");

fr.write("AGE: "+s2+"\r\n");

fr.write("Token: "+s3+"\r\n");

fr.write("BloodGroup: "+s4+"\r\n");

fr.write("Height: "+s5+"\r\n");

fr.write("Weight: "+s6+"\r\n");

fr.write("Symptoms: "+s7+"\r\n");

fr.write("BP: "+s8+"\r\n");

fr.write("Pulse Rate: "+s9+"\r\n");

//fr.write("Gender:"+dat+"\r\n");

fr.close();

}catch(Exception e){System.out.print("Error");}

}

});

bc.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e12) {

String wo=tok.getText();

try{

File file = new File("G:\\doc.txt");

Scanner scan = new Scanner(file);

while(scan.hasNext()){

String s=scan.nextLine();

if(s.contains(wo)){

tok.setText("You have a notification");

}

else {tok.setText("no notification yet");}

}

scan.close();}catch(Exception e){System.out.print(e);}

}});

b.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String data = c.getItem(c.getSelectedIndex());

for(Map.Entry m:map.entrySet()){

if(e.getSource()==b){

String l=(m.getKey().toString());

int k=l.compareTo(data);

if(k==0){

String result=m.getValue().toString();

t9.setText(result);}

}

}

}

});

f.add(b);f.add(tf); f.add(t1);f.add(t2);f.add(b1);f.add(l1);f.add(l2);f.add(l3);f.add(l7);f.add(l4);f.add(l5);f.add(l6);f.add(l8);f.add(l9);f.add(t1);f.add(t2);f.add(t3);f.add(t4);f.add(t5);f.add(t6);

f.add(t7);f.add(t8);f.add(t9);f.add(l9);f.add(bc);f.add(tok);}});

f.setSize(1000,1000);

f.setLayout(null);

f.setVisible(true);

}

}

SUMMARY

**User Characteristics:**

Every user should be:

• Comfortable of working with computer.

• He must have knowledge in medical field.

• He must also have basic knowledge of English too.

Constraints:

• GUI is only in English.

• Login and password is used for identification of user and there is no facility for guest.

**Software Interface**

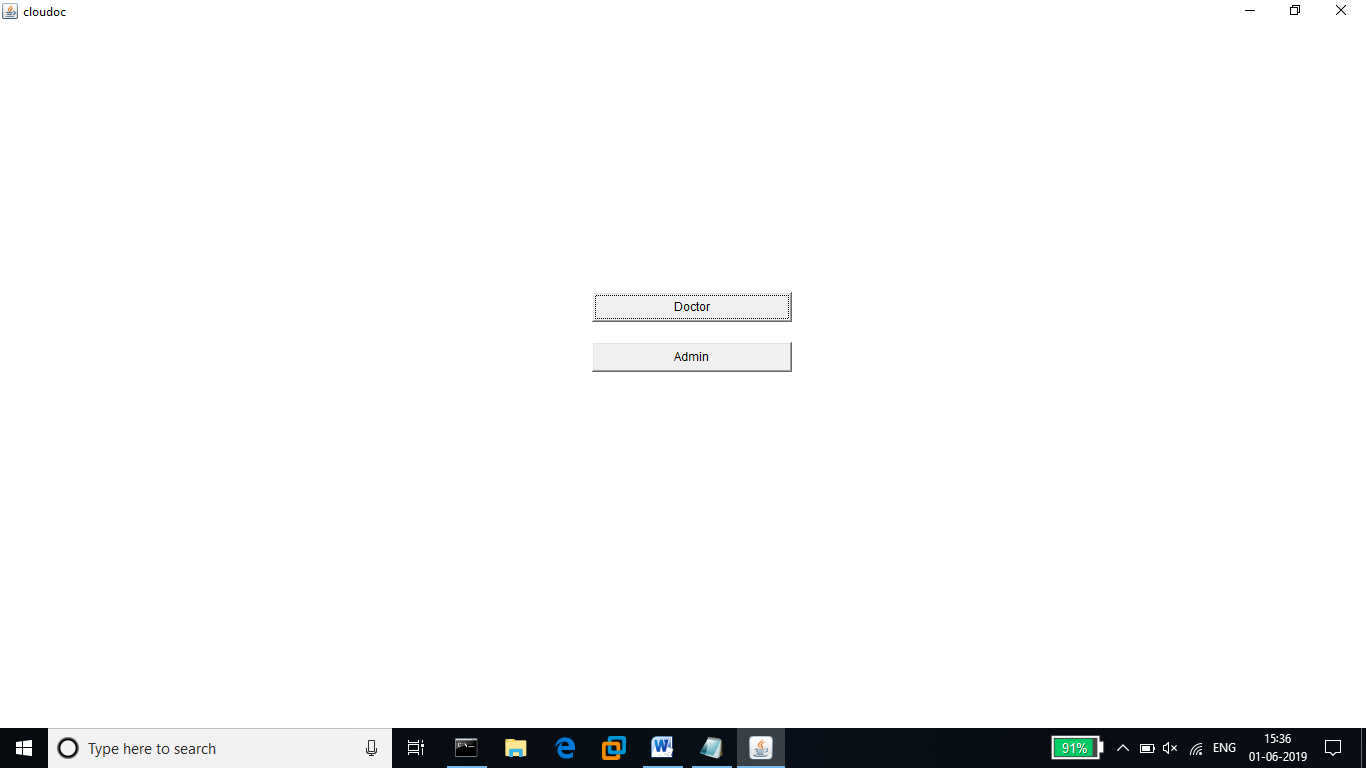
INTRODUCTION

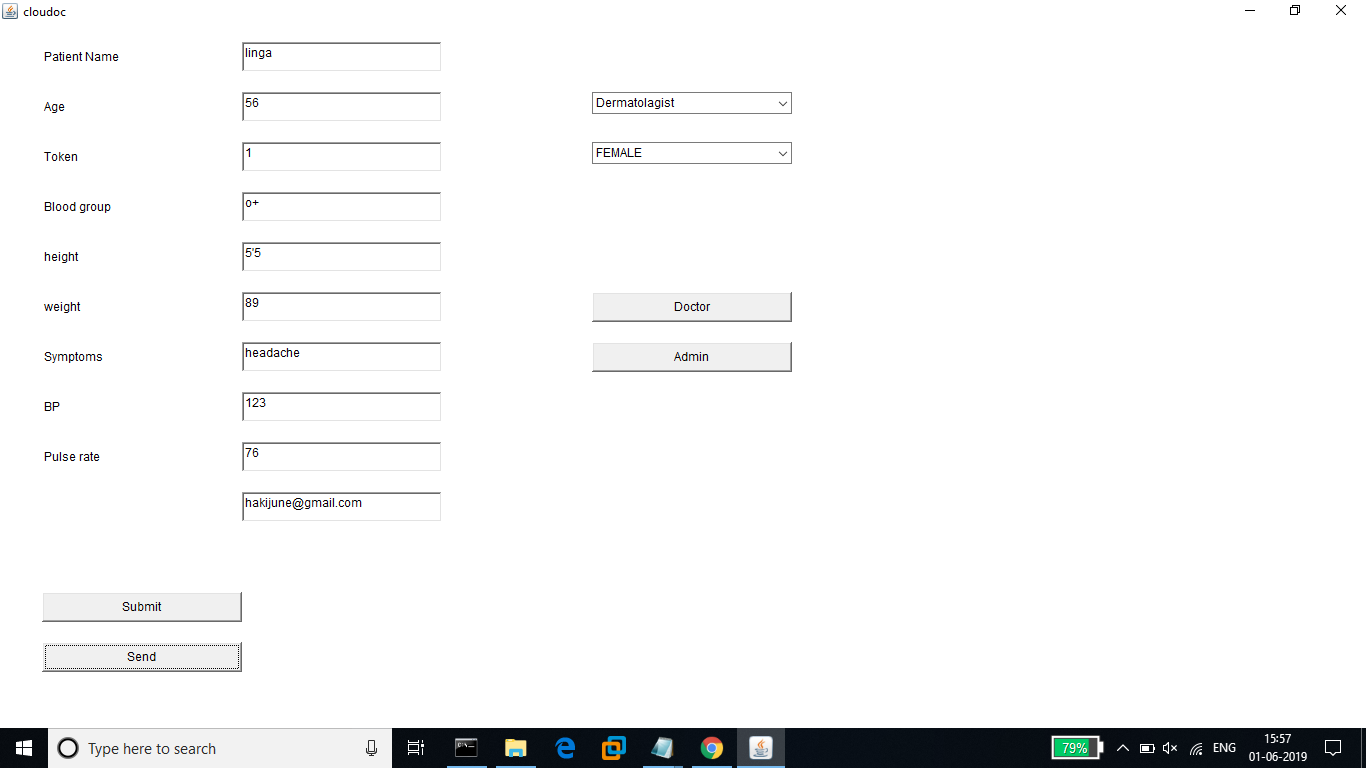
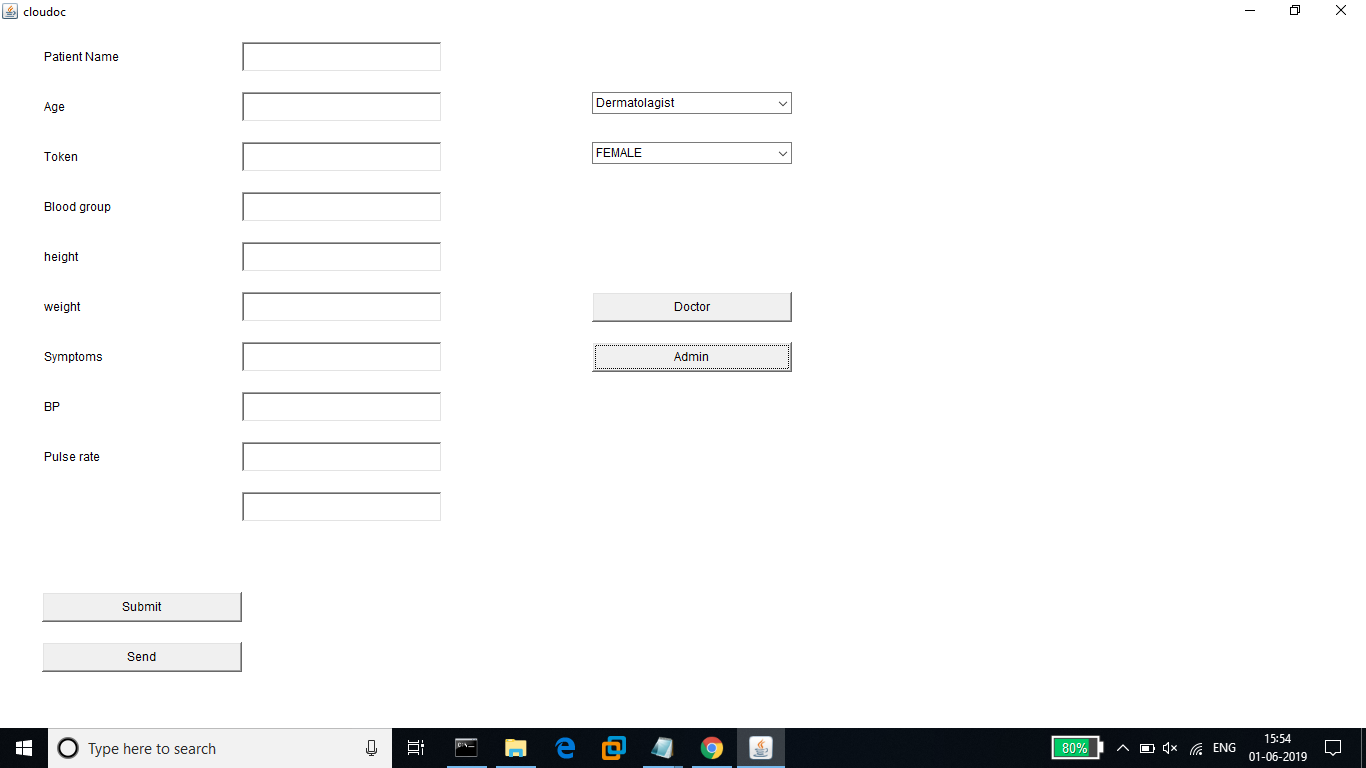
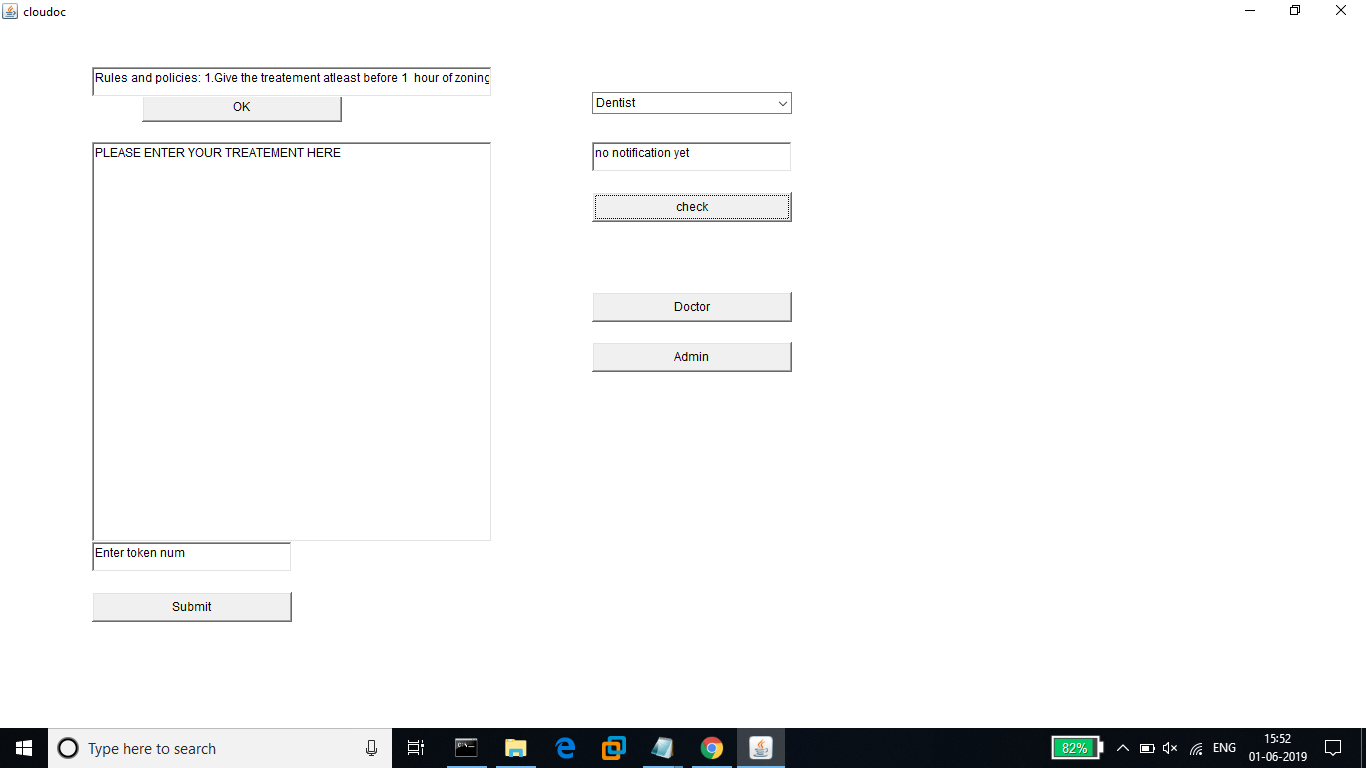
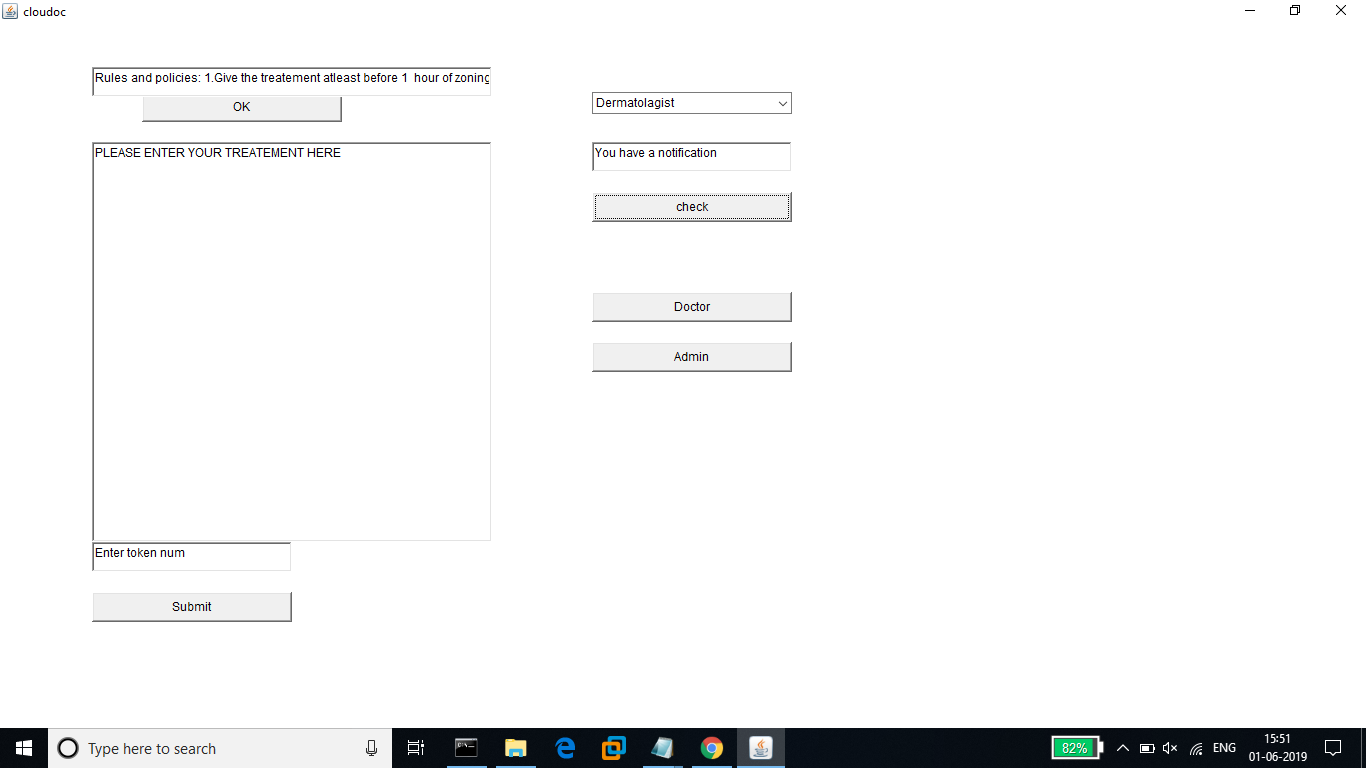
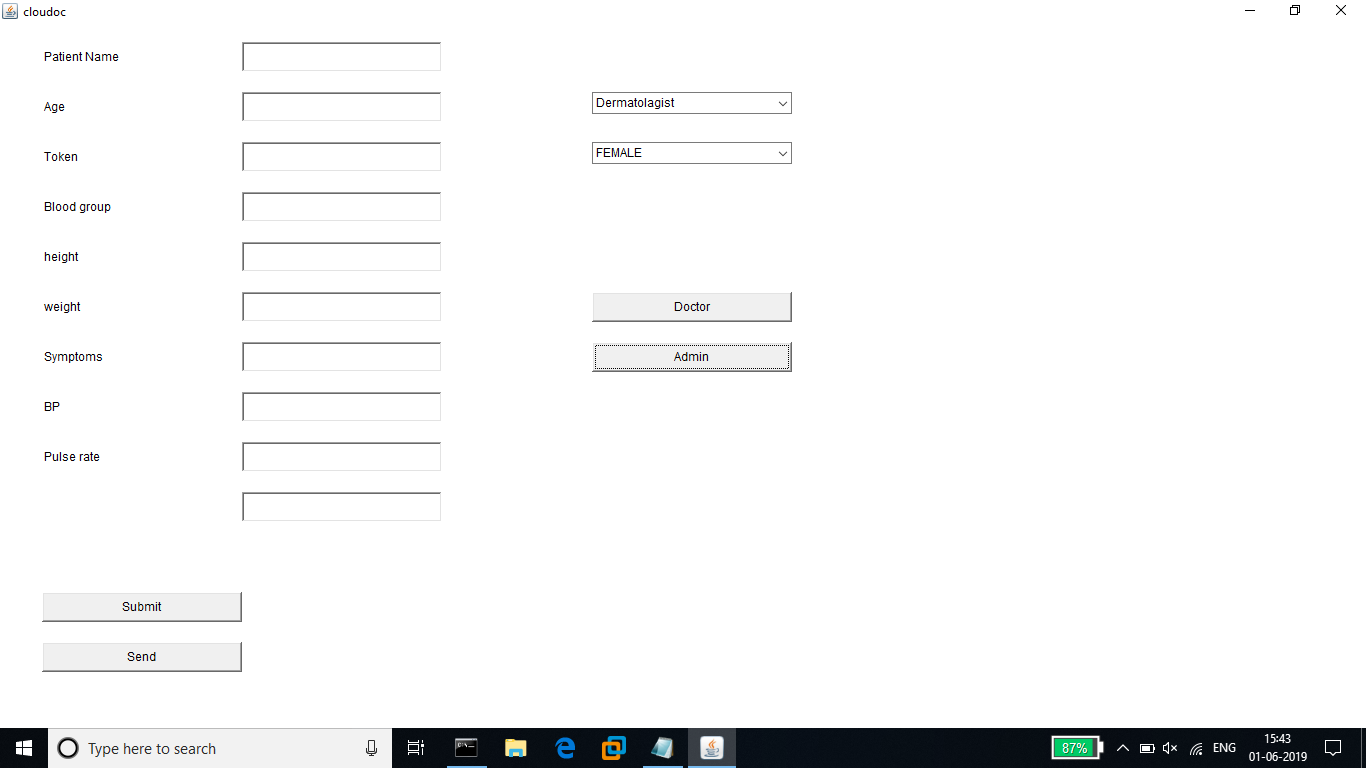
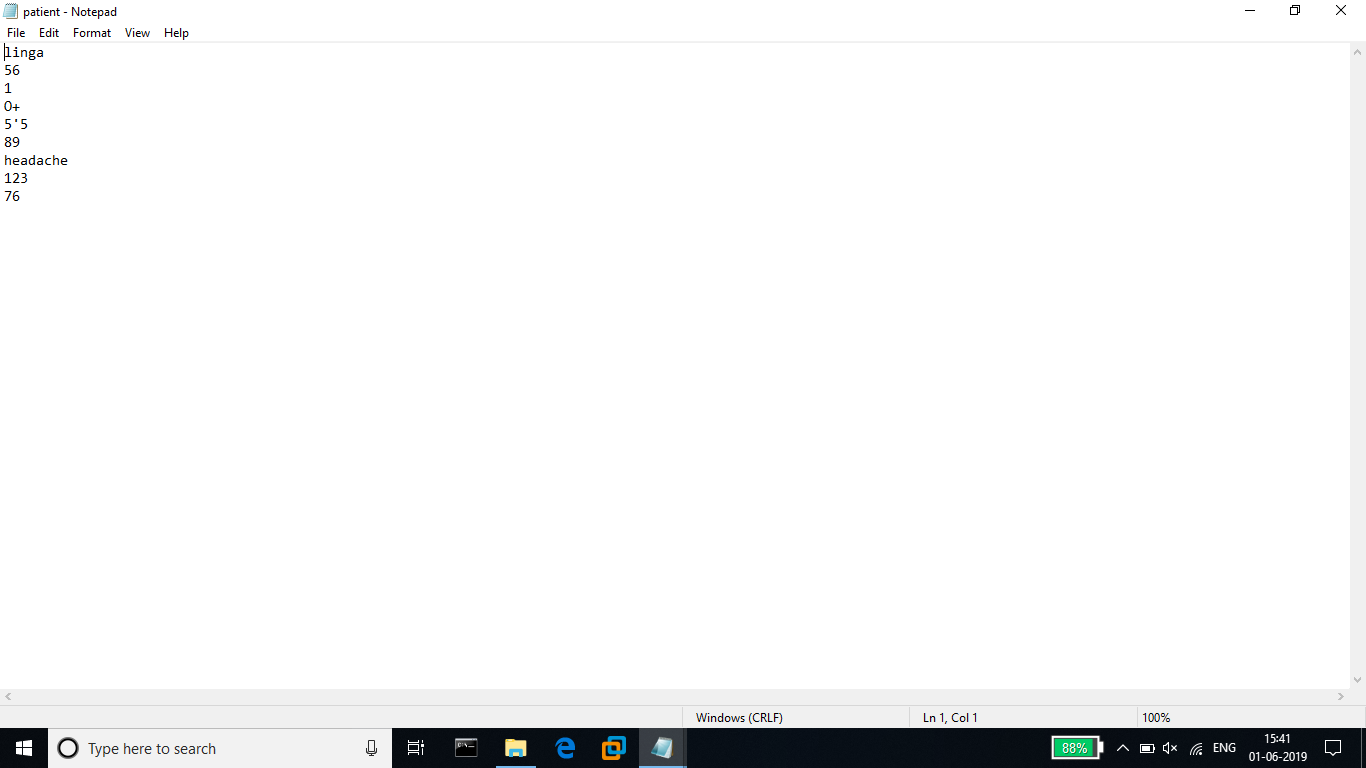
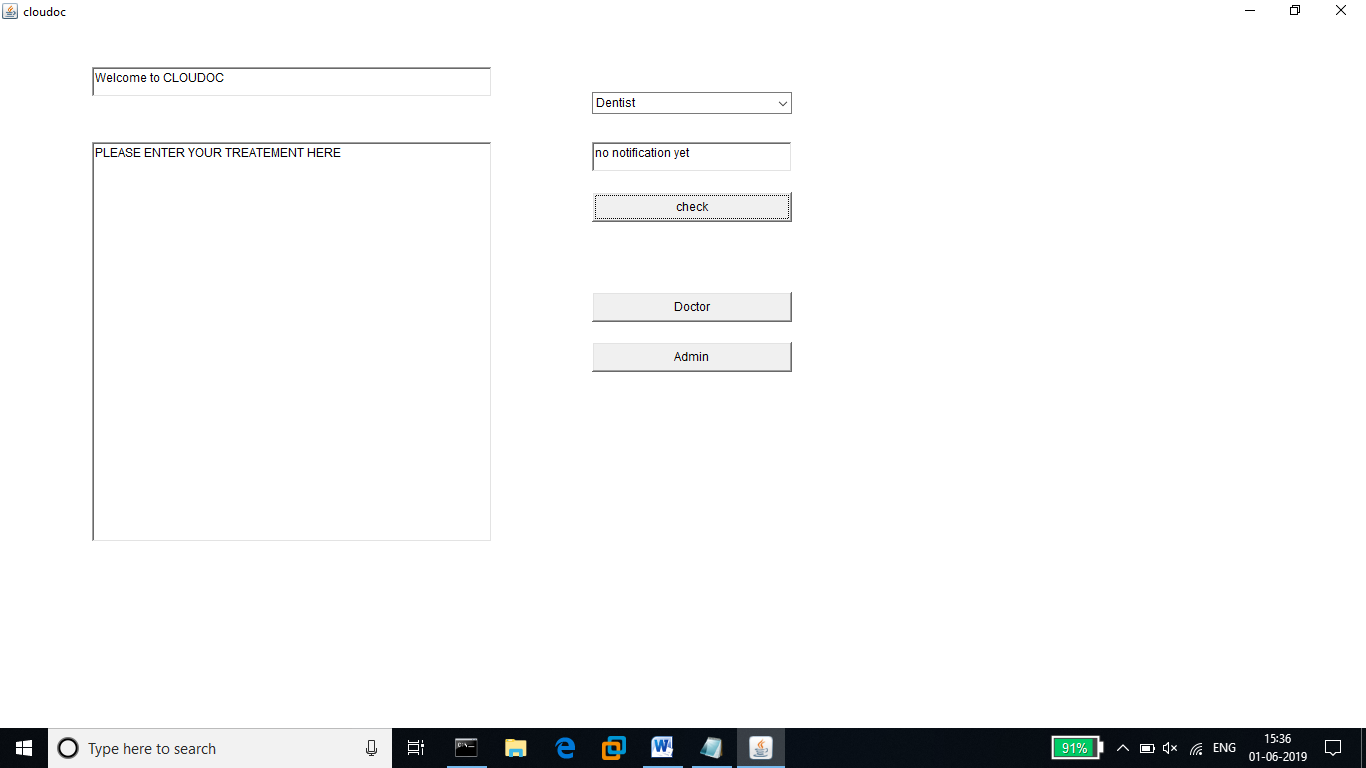
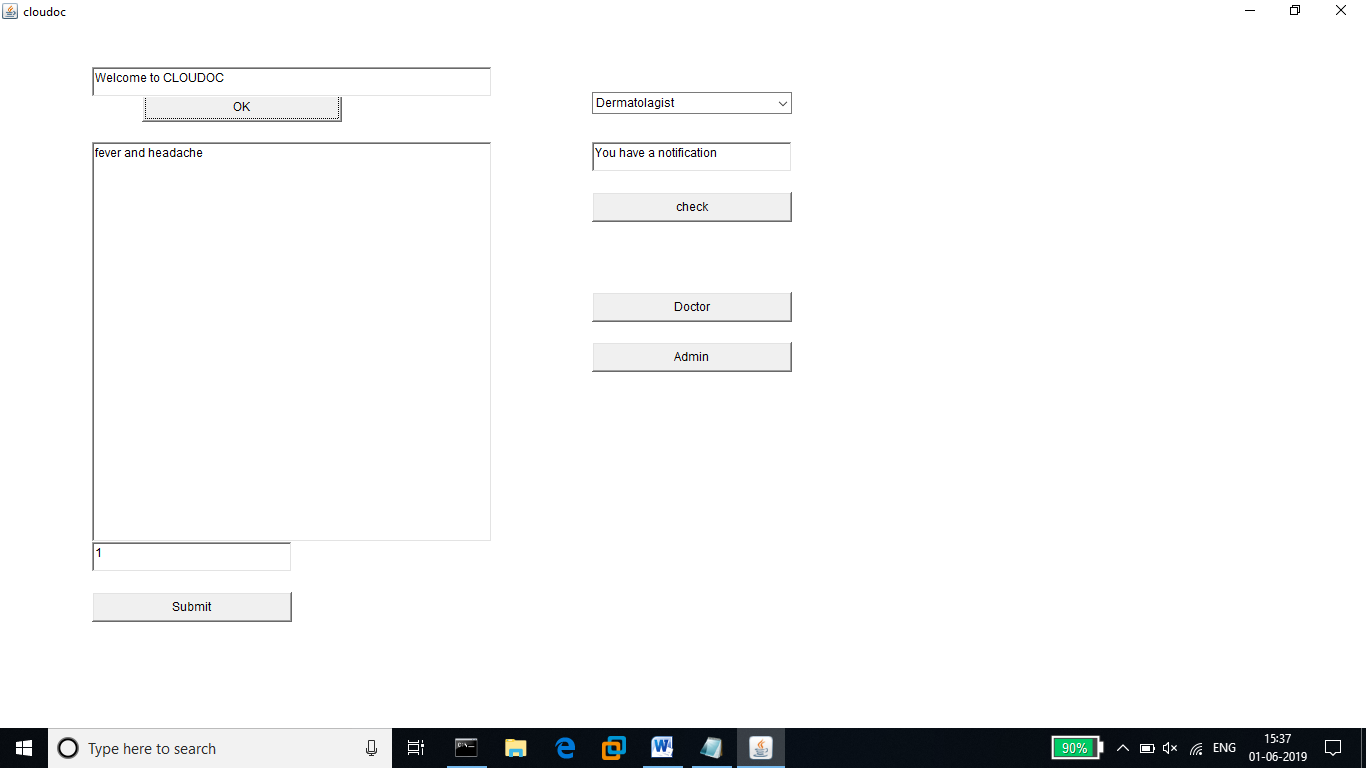
**Software interfaces** (programming **interfaces**) are the languages, codes and messages that programs use to communicate with each other and to the hardware. Examples are the Windows, Mac and Linux operating systems, SMTP email, IP network protocols and the **software** drivers that activate the peripheral devices.

**I**n our system we used windows OS, and SMTP email and cloud instances i.e. on google cloud platform.

This system can be possible in any OS, but only constraint is it should have SMTP EMAIL, and jdk 11.0.1 and a cloud platform.

Snapshots





**Summary**

cloudoc supports interface of mac os, Windows, and Linux for automatic upload of lab results into the test results in cloudoc. cloudoc generates tokens for samples for usage in this machines. This feature that reduces human errors in manual entry of results. Over 30 mail id’s of doctors have interfaced with cloudoc.

**Testing Phase**

**INTRODUCTION**

TESTING PHASE:

Testing is done in various ways such as testing the algorithm, programming code, sample data debugging is also one of following the above.