POLYCYSTIC OVARY SYNDROME

A PROJECT REPORT

Submitted by

AMYSOJ J A EXSON JOSEPH (112719104004)

MANISH KUMAR JHA (112719104019)

SAI MADAN RAJ T M (112719104031)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

St. PETER'S COLLEGE OF ENGINEERING AND TECHNOLOGY

ANNA UNIVERSITY::CHENNAI 600 025

JUNE 22

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report "**POLYCYSTIC OVARY SYNDROME TESTING APPLICATION**" is the bonafide work of "AMYSOJ J A EXSON JOSEPH (112719104004), MANISH KUMAR JHA (112719104019) and SAI MADAN RAJ T M (112719104031)" who carried out the project work under mysupervision.

SIGNATURE,	**SIGNATURE,**
Ms. P.PREETHY REBECCA,	**Ms. MARY SELVAN**
HEAD OF THE DEPARTMENT	**SUPERVISOR**
	ASSOCIATEPROFESSOR
Department of Computer Science	Department of Computer Science
&Engineering,	& Engineering,
St. Peter's College of Engineering	St. Peter's College of Engineering
And Technology,	and Technology,
Avadi,Chennai-600054.	Avadi, Chennai-600 054.

Submitted for viva-voce held on…… **..................................**at St. Peter's College of Engineering and Technology.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We thank God Almighty for giving this opportunity and blessing us to update our career through **St. Peter's College of Engineering and Technology.**

We thank the Chairperson **Dr. Banumathi Thambidurai**, for having provided the necessary infrastructure that has helped to learn and shine in our academic career.

We thank **Dr. C. Jayakumar, M.E., Ph.D., Principal**, who has been a source of inspiration for us to renown our institute's reputation.

We would like to express our faithful thanks **Ms. P. Preethy Rebecca, M.E.,** Head of the Department of Computer Science and Engineering, for giving the opportunity to carry out this project and having extended all the department facilities without slightest hesitation and also for guiding in the project development and for rendering her valuable guidance, encouragement and support throughout the project work.

We extend our sincere gratitude to **Ms. Mary Selvan, M.E.,** Associate Professor, Department of Computer Science and Engineering, Project Supervisor for rendering her valuable guidance and for helping us throughout the project work.

We extend our sincere thanks to all **Teaching and Non- Teaching staffs of Department of Computer Science and Engineering**, for their help and support. Further we thank **Parents and Friends** who haveconstantly encouraged us throughout this course by extending the moral support to achieve a great success in life.

ABSTRACT

POLYCYSTIC OVARY SYNDROME (**PCOS**) is a disorder involving infrequent, irregular or prolonged menstrual periods, and often excess male hormone (androgen) levels.
The ovaries develop numerous small collections of fluid called follicles and may fail to regularly release eggs dataset contains all physical and clinical parameters to determine **PCOS** and infertility related issues.

Computational developments, in turn, allow the application of machine learning methods to detect "patterns" from the data that can predict a patient's condition.

By building a simple app and getting data from user, feeding it into the model to determine the presence of **PCOS**.

Software Required : Python , Flutter

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGENO
	ABSTRACT	4
1	INTRODUCTION	6
2	LITERATURE REVIEW	7
3	REQUIREMENT ANALYSIS (EXISTING/PROPSED SYSTEM & SOFTWARE REQUIREMENT)	8
4	PROJECT TITTLE • INTRODUCTION • MODULE EXPLANATION • ARCHITCTURE DAIGRAM	10
5	CONCLUSION & FUTURE ENHANCEMENT	13
6	SCREEN SHOTS & SAMPLE CODE	14
	REFERENCE	30

CHAPTER 1
INTRODUCTION

The objective behind developing this PCOS application project is to build a system which can assist the medical experts to detect PCOS by means of an Android based application. This application contain the features by using of which can easily find out whether have these symptoms or not. The PCOS application project will be Android based so the doctor can access the details of symptoms. This application save a lot of time and very helpful for doctors.

In this project, contains model from catboat algorithm in python that is hosted in cloud and will get output from there. This app will give the result in digit.

CHAPTER 2

LITERATURE REVIEW

1. https://www.researchgate.net/publication/311866892_Current_aspects_of_polycystic_ovary_syndrome_A_literature_review
2. https://www.sciencedirect.com/science/article/abs/pii/S1701216316304029
3. https://nursinganswers.net/essays/literature-review-polycystic-ovarian-syndrome-health-and-social-care-essay.php
4. https://pubmed.ncbi.nlm.nih.gov/28001262/

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 EXISTING SYSTEM

In today’s modern world, we use data warehouse for exploring the data from wide sources and converging in to one while data mining is used for extracting the useful knowledge out from data warehouse. The previous works of various recordists in references indicate their research work of PCOS, their causes, ways to be detected and so on. The work of Amsy Denny et al, addresses the problem of early detection and prediction of PCOS through optimal and minimal parameters. The recordist had collected data through a survey from patients (around 500) during doctor consultation. The work of Subrato et al, explains the detection of PCOS through univariate feature selection algorithm which uses the ratio of Folliclesti mulating hormone and Luteinizing hormone as the most important attribute.

3.2 PROPSED SYSTEM

The proposed pcos application will help the Doctors. This app is very useful for the Doctors, in this app only input data has to be entered, after that machine learning algorithm will process it in the model and then generate the report. Heroku will work in this app. This app will give the result immediately after entering the input. This app will give the result in digit, it will tell whether it is a symptom or not.This app helps doctors as well as patients. This application contains model from catboost algorithm in python that is hosted in cloud and will get output from there. This app will give the result in digit, after that this app will convert the digit and it will tell whether it is a symptom or not. This app helps doctors as well as patients. The Pcos app is a health and wellbeing platform that supports women during their entire reproductive lives – from first menstruation to early motherhood and menopause. With the help of this app, doctors will be very easy to test and will be able to give results quickly and will give accurate results. This application contains model from catboost algorithm in python that is hosted in cloud and will get output from there. This app will give the result in digit, after that this app will convert the digit and it will tell whether it is a symptom or not. This app helps doctors as well as patients. The Pcos app is a health and wellbeing platform that supports women during their entire reproductive lives – from first menstruation to early motherhood and menopause. With the help of this app, doctors will be very easy to test and will be able to give results quickly and will give accurate results.

3.3 SOFTWARE REQUIREMENT

- Python
    - Matplotlib
    - Pandas
    - Seaboard
    - Flask
- Flutter
- Machine Learning
    - Catboost Algorithm
    - Sklearn
- Cloud
    - Heroku

CHAPTER 4

PROJECT TITTLE

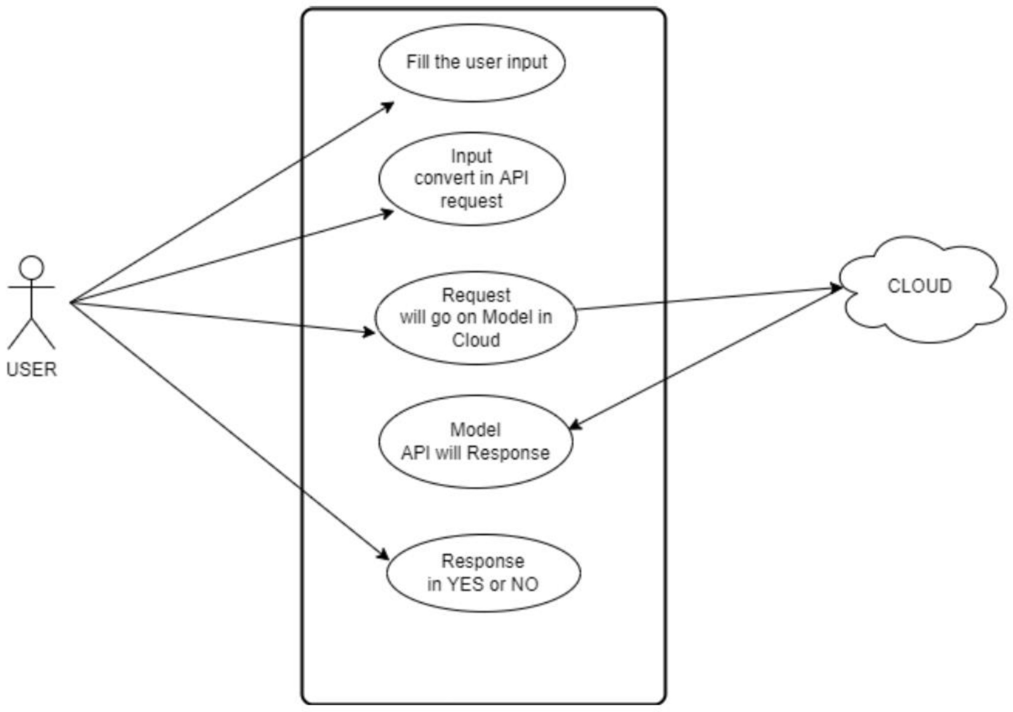
4.1 INTRODUCTION

The objective behind developing this pcos application project is to build a system which can assist the medical experts to detect PCOS by means of an Android based application.This application will contain the features by using of which can easily find out whether have these symptoms or not.The pcos application project will be Android based so the doctor can access the details of symptoms.This application will save a lot of time and very helpful for doctors

4.2 MODULE EXPLANATION

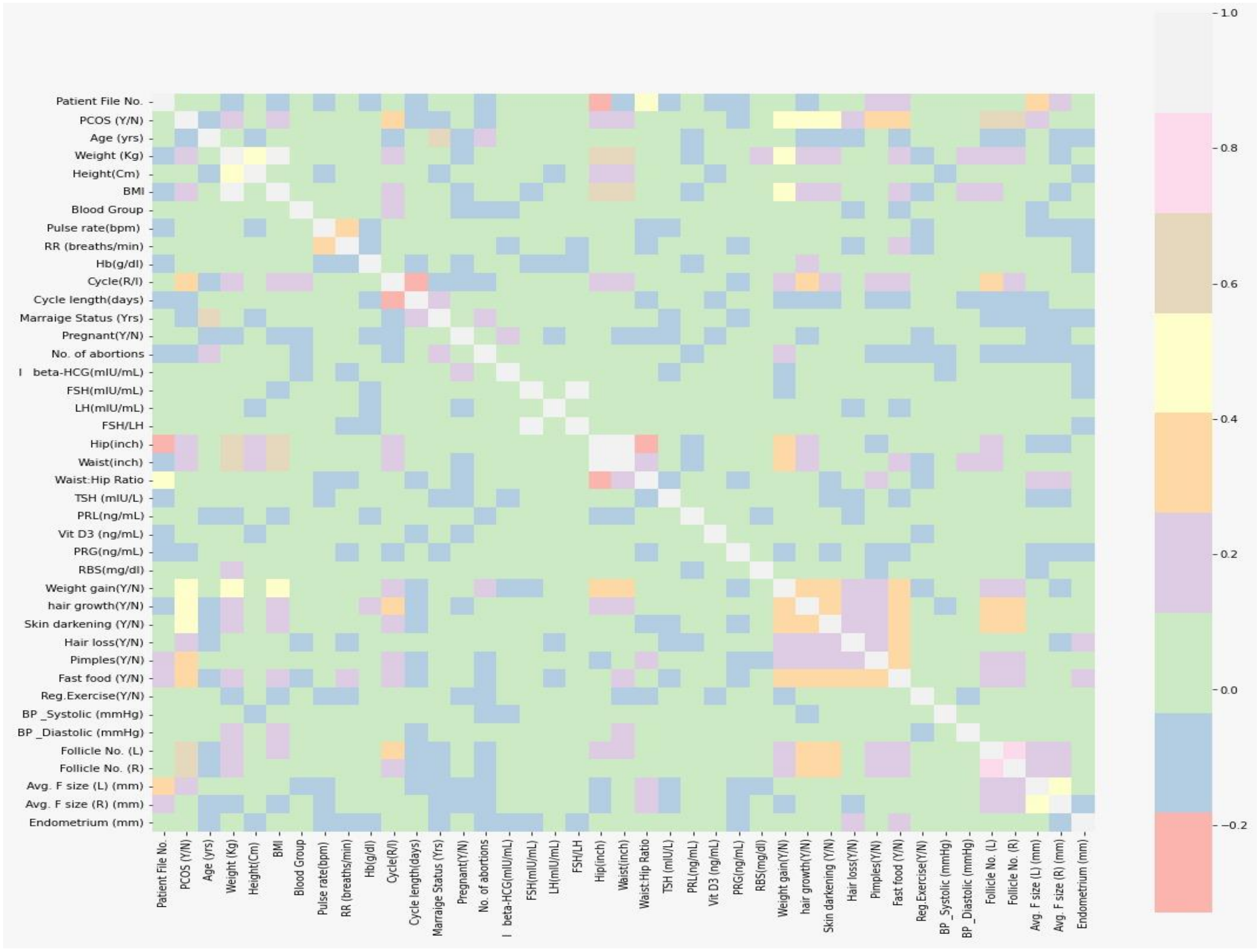
User module will be the most important part of the application as it will validate the new patient, will add the patient to the system, will add the details about patient **Name, Patient File No., Age, Weight, Height** into the system, will generate the report of the work. App give the result in digit, it will tell whether it is a symptom or not

4.2 ARCHITCTURE DAIGRAM



- Step 1 – User fill the inputs
- Step 2 – Input convert in API request
- Step 3 –Request will go on Model in Cloud
- Step 4 –Model API will Response
- Step 5–Response in YES OR NO

Dataset visualization



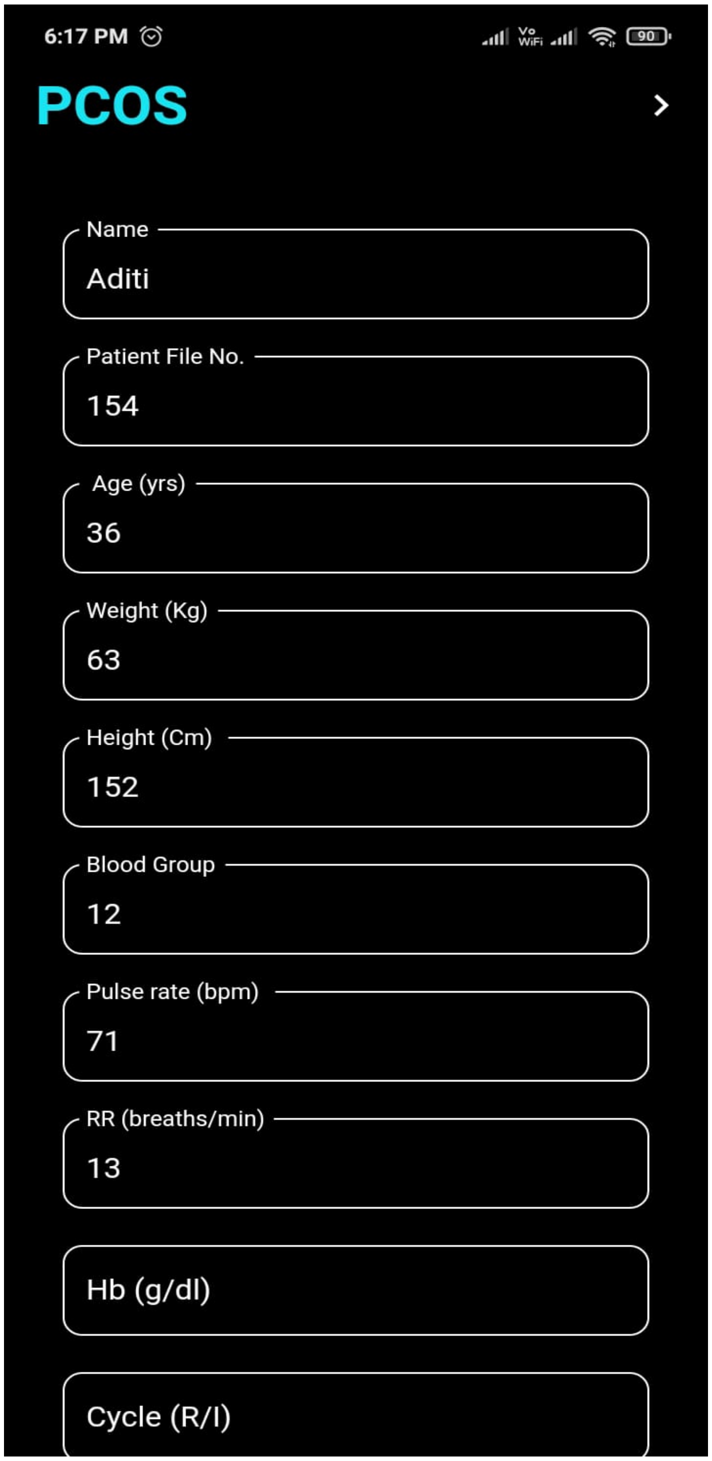
CHAPTER 5

CONCLUSION

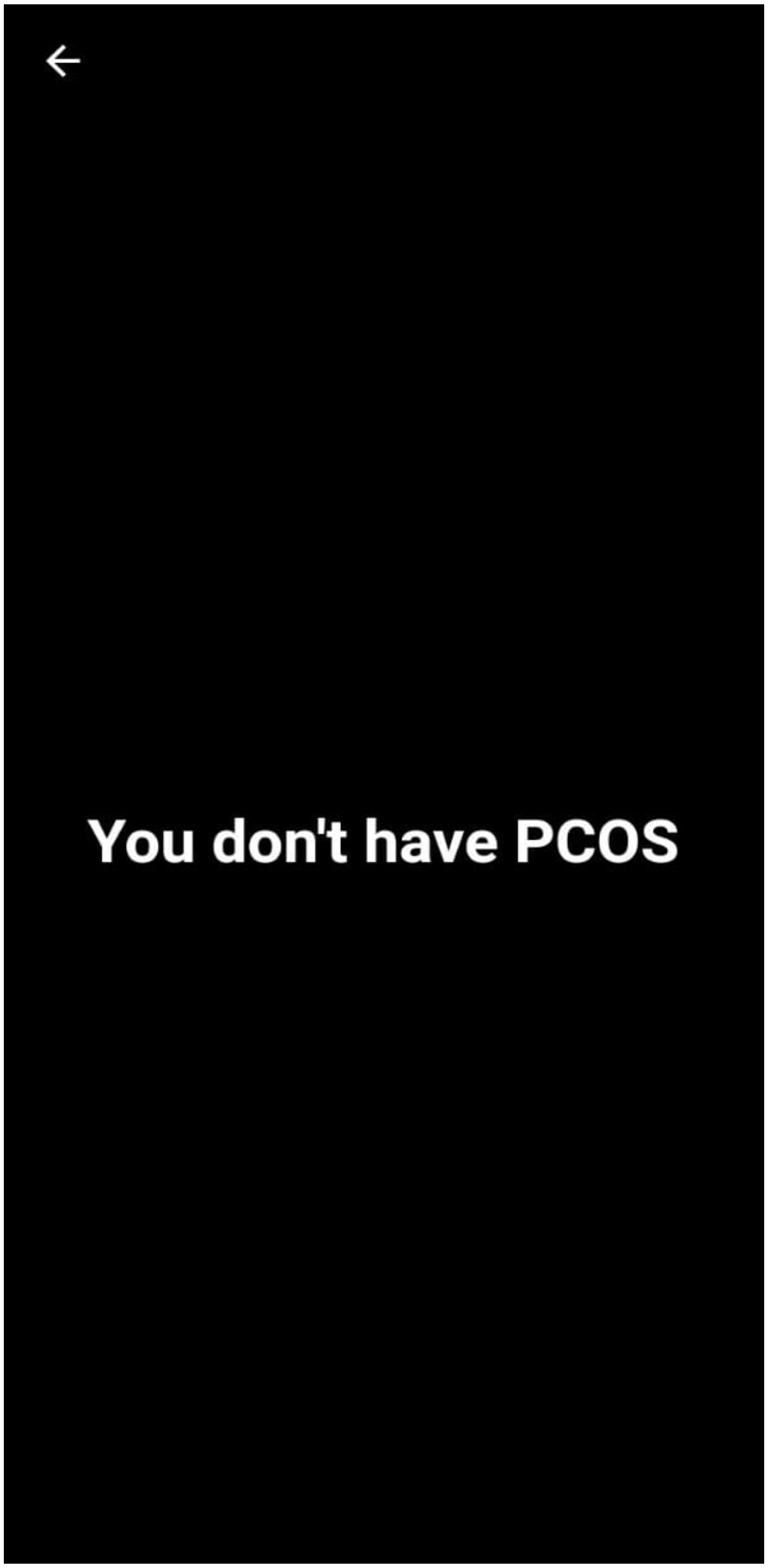
In conclusion, PCOS is becoming a more prevalent disorder among women of reproductive age with lifelong complications. One of the most challenging aspects of this syndrome is its ambiguous diagnostic criteria and vast complexity of characteristics. In the future, more research in the genetics and pathophysiology of PCOS is needed to determine preventative risk factors as well as successful treatment modalities for this syndrome.

CHAPTER 6

SCREEN SHOTS & SOURCE CODE



Name	Aditi
Patient File No.	154
Age (yrs)	36
Weight (Kg)	63
Height (Cm)	152
Blood Group	12
Pulse rate (bpm)	71
RR (breaths/min)	13
Hb (g/dl)	
Cycle (R/I)	



You don't have PCOS

SOURCE CODE

APP:

```
from flask import Flask,request, jsonify
from flask_cors import CORS
from catboost import CatBoostClassifier
import pickle

app = Flask(__name__)
CORS(app)

model = pickle.load(open('PCOS_model.sav', 'rb'))

cols = ['Patient File No.', ' Age (yrs)', 'Weight (Kg)', 'Height(Cm) ', 'BMI', 'Blood Group', 
'Pulse rate(bpm) ', 'RR (breaths/min)', 'Hb(g/dl)', 'Cycle(R/I)', 'Cycle length(days)', 'Marraige Status 
(Yrs)', 'Pregnant(Y/N)', 'No. of abortions', '  I   beta-HCG(mIU/mL)', 'II    beta-HCG(mIU/mL)', 
'FSH(mIU/mL)', 'LH(mIU/mL)', 'FSH/LH', 'Hip(inch)', 'Waist(inch)', 'Waist:Hip Ratio', 'TSH 
(mIU/L)', 'AMH(ng/mL)', 'PRL(ng/mL)', 'Vit D3 (ng/mL)', 'PRG(ng/mL)', 'RBS(mg/dl)', 'Weight 
gain(Y/N)', 'hair growth(Y/N)', 'Skin darkening (Y/N)', 'Hair loss(Y/N)', 'Pimples(Y/N)', 'Fast food 
(Y/N)', 'Reg.Exercise(Y/N)', 'BP _Systolic (mmHg)', 'BP _Diastolic (mmHg)', 'Follicle No. (L)', 
'Follicle No. (R)', 'Avg. F size (L) (mm)', 'Avg. F size (R) (mm)', 'Endometrium (mm)']

@app.route('/predict_api',methods=['POST'])
def predict_api():
   intdata = request.json['data']
   data=[]
   for v in intdata:
      if v=='Y':
         data.append(1)
      elif v=='y':
         data.append(1)
      elif v=='N':
         data.append(0)
      elif v=='n':
         data.append(0)
      else:
         data.append(v)
   print(data)
   prediction = model.predict(data)

   if prediction == 0:
      prediction = "You don't have PCOS"

   elif prediction == 1:
```

```
            prediction = "You have PCOS"
        return jsonify({'output':f'{prediction}'})


    if __name__ == '__main__':
        app.run()
```

HOME.DART

```
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:http/http.dart' as http;
import 'dart:convert';

import 'package:pcos_testing_application/result.dart';

class Requestoutput {
  final String data;

  const Requestoutput({required this.data});

  factory Requestoutput.fromJson(Map<String, dynamic> json) {
    var loc = json['output'];
    return Requestoutput(data: loc);
  }
}

class MyHomePage extends StatefulWidget {
  const MyHomePage({Key? key}) : super(key: key);

  @override
  State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  List output = [];
  String finaloutput = '';
  onsubmit() {
    output = [
      c2.text,
      c3.text,
      c4.text,
      c5.text,
      c6.text,
      c7.text,
      c8.text,
      c9.text,
      c10.text,
      c11.text,
      c12.text,
      c13.text,
      c14.text,
      c15.text,
      c16.text,
      c17.text,
      c18.text,
      c19.text,
      c20.text,
      c21.text,
      c22.text,
      c23.text,
      c24.text,
      c25.text,
      c26.text,
      c27.text,
      c28.text,
      c29.text,
      c30.text,
      c31.text,
      c32.text,
      c33.text,
      c34.text,
      c35.text,
      c36.text,
      c37.text,
      c38.text,
      c39.text,
      c40.text,
      c41.text,
      c42.text,
      c43.text,
    ];

    if (output.contains('')) {
      ScaffoldMessenger.of(context).showSnackBar(const SnackBar(
          backgroundColor: Colors.red,
          content: Text(
            "Fill all Fields",
            style: TextStyle(
              color: Colors.white,
            ),
          )));
    } else {
      api(output);
    }
  }
```

```
api(List output) async {
  final response =
      await http.post(Uri.parse('https://pcos-api.herokuapp.com/predict_api'),
          headers: <String, String>{
            'Content-Type': 'application/json; charset=UTF-8',
            "Access-Control-Allow-Origin": "*",
          },
          body: jsonEncode({'data': output}));
  if (response.statusCode == 200) {
    setState(() {
      finaloutput = Requestoutput.fromJson(jsonDecode(response.body)).data;
    });
    Navigator.push(
      context,
      MaterialPageRoute(
          builder: (context) => OutputScreen(prediction: finaloutput)),
    );
    return Requestoutput.fromJson(jsonDecode(response.body));
  } else {
    throw Exception('Failed to load album');
  }
}

TextEditingController c1 = TextEditingController();
TextEditingController c2 = TextEditingController();
TextEditingController c3 = TextEditingController();
TextEditingController c4 = TextEditingController();
TextEditingController c5 = TextEditingController();
TextEditingController c6 = TextEditingController();
TextEditingController c7 = TextEditingController();
TextEditingController c8 = TextEditingController();
TextEditingController c9 = TextEditingController();
TextEditingController c10 = TextEditingController();
TextEditingController c11 = TextEditingController();
TextEditingController c12 = TextEditingController();
TextEditingController c13 = TextEditingController();
TextEditingController c14 = TextEditingController();
TextEditingController c15 = TextEditingController();
TextEditingController c16 = TextEditingController();
TextEditingController c17 = TextEditingController();
TextEditingController c18 = TextEditingController();
TextEditingController c19 = TextEditingController();
TextEditingController c20 = TextEditingController();
TextEditingController c21 = TextEditingController();
TextEditingController c22 = TextEditingController();
TextEditingController c23 = TextEditingController();
TextEditingController c24 = TextEditingController();
TextEditingController c25 = TextEditingController();
TextEditingController c26 = TextEditingController();
TextEditingController c27 = TextEditingController();
TextEditingController c28 = TextEditingController();
TextEditingController c29 = TextEditingController();
TextEditingController c30 = TextEditingController();
TextEditingController c31 = TextEditingController();
TextEditingController c32 = TextEditingController();
TextEditingController c33 = TextEditingController();
TextEditingController c34 = TextEditingController();
TextEditingController c35 = TextEditingController();
TextEditingController c36 = TextEditingController();
TextEditingController c37 = TextEditingController();
TextEditingController c38 = TextEditingController();
TextEditingController c39 = TextEditingController();
TextEditingController c40 = TextEditingController();
TextEditingController c41 = TextEditingController();
TextEditingController c42 = TextEditingController();
TextEditingController c43 = TextEditingController();
Widget texfield(String lable, TextEditingController inputvalue,
    List<TextInputFormatter> tf, TextInputType inputtype) {
  return Padding(
    padding: const EdgeInsets.all(10.0),
    child: Container(
      height: 50,
      alignment: Alignment.center,
      child: TextFormField(
        inputFormatters: tf,
        style: const TextStyle(
          color: Colors.white,
        ),
        cursorColor: Colors.white,
        keyboardType: inputtype,
        controller: inputvalue,
        decoration: InputDecoration(
          focusedBorder: OutlineInputBorder(
            borderSide: const BorderSide(color: Colors.white, width: 1.0),
            borderRadius: BorderRadius.circular(10.0),
          ),
          border: OutlineInputBorder(
            borderSide: const BorderSide(color: Colors.white, width: 1.0),
            borderRadius: BorderRadius.circular(10.0),
          ),
          label: Text(
            lable,
            style: const TextStyle(
              color: Colors.white,
            ),
          ),
          enabledBorder: OutlineInputBorder(
            borderSide: const BorderSide(color: Colors.white, width: 1.0),
            borderRadius: BorderRadius.circular(10.0),
          ),
        ),
      ),
    ),
  );
}

@override
Widget build(BuildContext context) {
  var size = MediaQuery.of(context).size;
  return Scaffold(
    appBar: AppBar(
      elevation: 0,
      title: const Text(
        "PCOS",
        style: TextStyle(
            color: Color(0xFF18E4F2),
            fontSize: 30,
            fontWeight: FontWeight.bold),
      ),
      actions: [
        IconButton(
            onPressed: () {
              onsubmit();
            },
            icon: const Icon(Icons.keyboard_arrow_right_sharp))
      ],
    ),
    body: GestureDetector(
      onTap: () {
        FocusScope.of(context).unfocus();
      },
      child: Container(
        color: Colors.black,
        height: size.height,
        width: size.width,
        child: Padding(
          padding: const EdgeInsets.symmetric(horizontal: 20.0),
          child: SingleChildScrollView(
              child: Column(
            children: [
              const SizedBox(
                height: 30,
              ),
              texfield(
                  "Name",
                  c1,
                  [FilteringTextInputFormatter.allow(RegExp(r"[a-zA-Z. ]"))],
                  TextInputType.name),
              texfield(
                'Patient File No.',
                c2,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                ' Age (yrs)',
                c3,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Weight (Kg)',
                c4,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Height (Cm) ',
                c5,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'BMI',
                c6,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Blood Group',
                c7,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Pulse rate (bpm) ',
                c8,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'RR (breaths/min)',
                c9,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Hb (g/dl)',
                c10,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Cycle (R/I)',
                c11,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Cycle length (days)',
                c12,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Marraige Status (Yrs)',
                c13,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                  'Pregnant (Y/N)',
                  c14,
                  [
                    LengthLimitingTextInputFormatter(1),
                    FilteringTextInputFormatter.allow(RegExp("y|n|Y|N"))
                  ],
                  TextInputType.name),
              texfield(
                'No. of abortions',
                c15,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'I   beta-HCG (mIU/mL)',
                c16,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'II    beta-HCG (mIU/mL)',
                c17,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'FSH (mIU/mL)',
                c18,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'LH (mIU/mL)',
                c19,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'FSH/LH',
                c20,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Hip (inch)',
                c21,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Waist (inch)',
                c22,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Waist:Hip Ratio',
                c23,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'TSH (mIU/L)',
                c24,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'AMH (ng/mL)',
                c25,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'PRL (ng/mL)',
                c26,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Vit D3 (ng/mL)',
                c27,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'PRG (ng/mL)',
                c28,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'RBS (mg/dl)',
                c29,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                  'Weight gain (Y/N)',
                  c30,
                  [
                    LengthLimitingTextInputFormatter(1),
                    FilteringTextInputFormatter.allow(RegExp("y|n|Y|N"))
                  ],
                  TextInputType.name),
              texfield(
                  'hair growth (Y/N)',
                  c31,
                  [
                    LengthLimitingTextInputFormatter(1),
                    FilteringTextInputFormatter.allow(RegExp("y|n|Y|N"))
                  ],
                  TextInputType.name),
              texfield(
                  'Skin darkening (Y/N)',
                  c32,
                  [
                    LengthLimitingTextInputFormatter(1),
                    FilteringTextInputFormatter.allow(RegExp("y|n|Y|N"))
                  ],
                  TextInputType.name),
              texfield(
                  'Hair loss (Y/N)',
                  c33,
                  [
                    LengthLimitingTextInputFormatter(1),
                    FilteringTextInputFormatter.allow(RegExp("y|n|Y|N"))
                  ],
                  TextInputType.name),
              texfield(
                  'Pimples (Y/N)',
                  c34,
                  [
                    LengthLimitingTextInputFormatter(1),
                    FilteringTextInputFormatter.allow(RegExp("y|n|Y|N"))
                  ],
                  TextInputType.name),
              texfield(
                  'Fast food (Y/N)',
                  c35,
                  [
                    LengthLimitingTextInputFormatter(1),
                    FilteringTextInputFormatter.allow(RegExp("y|n|Y|N"))
                  ],
                  TextInputType.name),
              texfield(
                  'Reg.Exercise (Y/N)',
                  c36,
                  [
                    LengthLimitingTextInputFormatter(1),
                    FilteringTextInputFormatter.allow(RegExp("y|n|Y|N"))
                  ],
                  TextInputType.name),
              texfield(
                'BP _Systolic (mmHg)',
                c37,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'BP _Diastolic (mmHg)',
                c38,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Follicle No. (L)',
                c39,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Follicle No. (R)',
                c40,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Avg. F size (L) (mm)',
                c41,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Avg. F size (R) (mm)',
                c42,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
              texfield(
                'Endometrium (mm)',
                c43,
                [FilteringTextInputFormatter.allow(RegExp(r"[0-9.]"))],
                TextInputType.number,
              ),
            ],
          )),
        ),
      ),
    ),
  );
 }
}
```

MAIN:

```
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:pcos_testing_application/home.dart';
```

```
void main() {
  WidgetsFlutterBinding.ensureInitialized();
  SystemChrome.setPreferredOrientations([DeviceOrientation.portraitUp])
      .then((_) {
    runApp(const MyApp());
  });
}

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Flutter Demo',
      theme: ThemeData(
        primarySwatch: primaryColor,
      ),
      home: const MyHomePage(),
    );
  }
}

const MaterialColor primaryColor = MaterialColor(
  _primaryValue,
  <int, Color>{
    50: Color(0xFF000000),
    100: Color(0xFF000000),
    200: Color(0xFF000000),
    300: Color(0xFF000000),
    400: Color(0xFF000000),
    500: Color(_primaryValue),
    600: Color(0xFF000000),
    700: Color(0xFF000000),
    800: Color(0xFF000000),
    900: Color(0xFF000000),
  },
);
const int _primaryValue = 0xFF000000;
```

RESULT:

```
import 'package:flutter/material.dart';
import 'package:pcos_testing_application/home.dart';

class OutputScreen extends StatefulWidget {
  const OutputScreen({Key? key, required this.prediction}) : super(key: key);
  final String prediction;
  @override
  State<OutputScreen> createState() => _OutputScreenState();
}

class _OutputScreenState extends State<OutputScreen> {
  @override
  Widget build(BuildContext context) {
    var size = MediaQuery.of(context).size;

    return Scaffold(
      appBar: AppBar(),
      body: Container(
          height: size.height,
          width: size.width,
          color: Colors.black,
          child: Center(
            child: Text(
              widget.prediction,
              style: const TextStyle(
                  color: Colors.white,
                  fontSize: 30,
                  fontWeight: FontWeight.bold),
            ),
          )),
    );
  }
}
```

REFERENCE

1. **https://www.kaggle.com/datasets/prasoonkottarathil/polycystic-ovary-syndrome-pcos**
2. **https://catboost.ai/en/docs/**
3. **https://openapi-generator.tech/docs/generators/python-flask/**
4. **https://devcenter.heroku.com/**