Compose Input: A Demonstration of Text Input and Validation with Android Compose

Team Leader :

S.Lokeswari

Team Members :

M.Aruna

S.Abisha

T.Bagathiswari

R.M.Shanmugapriya

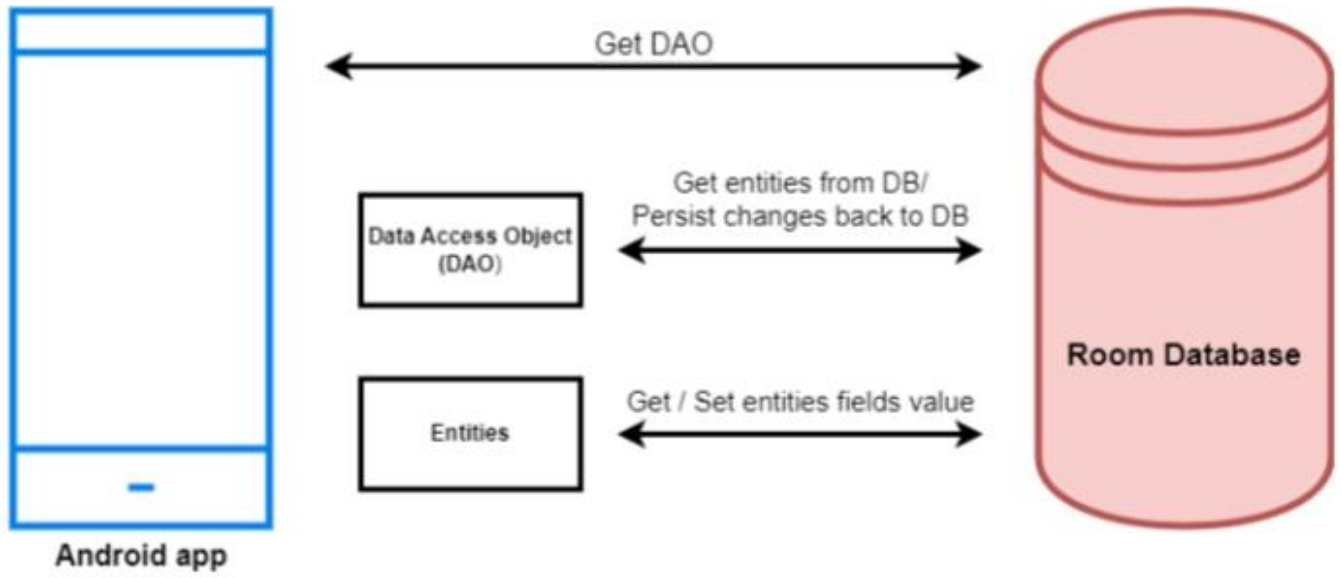
PROJECT REPORT

1. INTRODUCTION:

1.1 Overview

The app is a sample project that demonstrates how to use the Android Compose UI toolkit to build a survey app. The app allows the user to answer a series of questions. It showcases some of the key features of the Compose UI toolkit, data management, and user interactions.

Architecture:

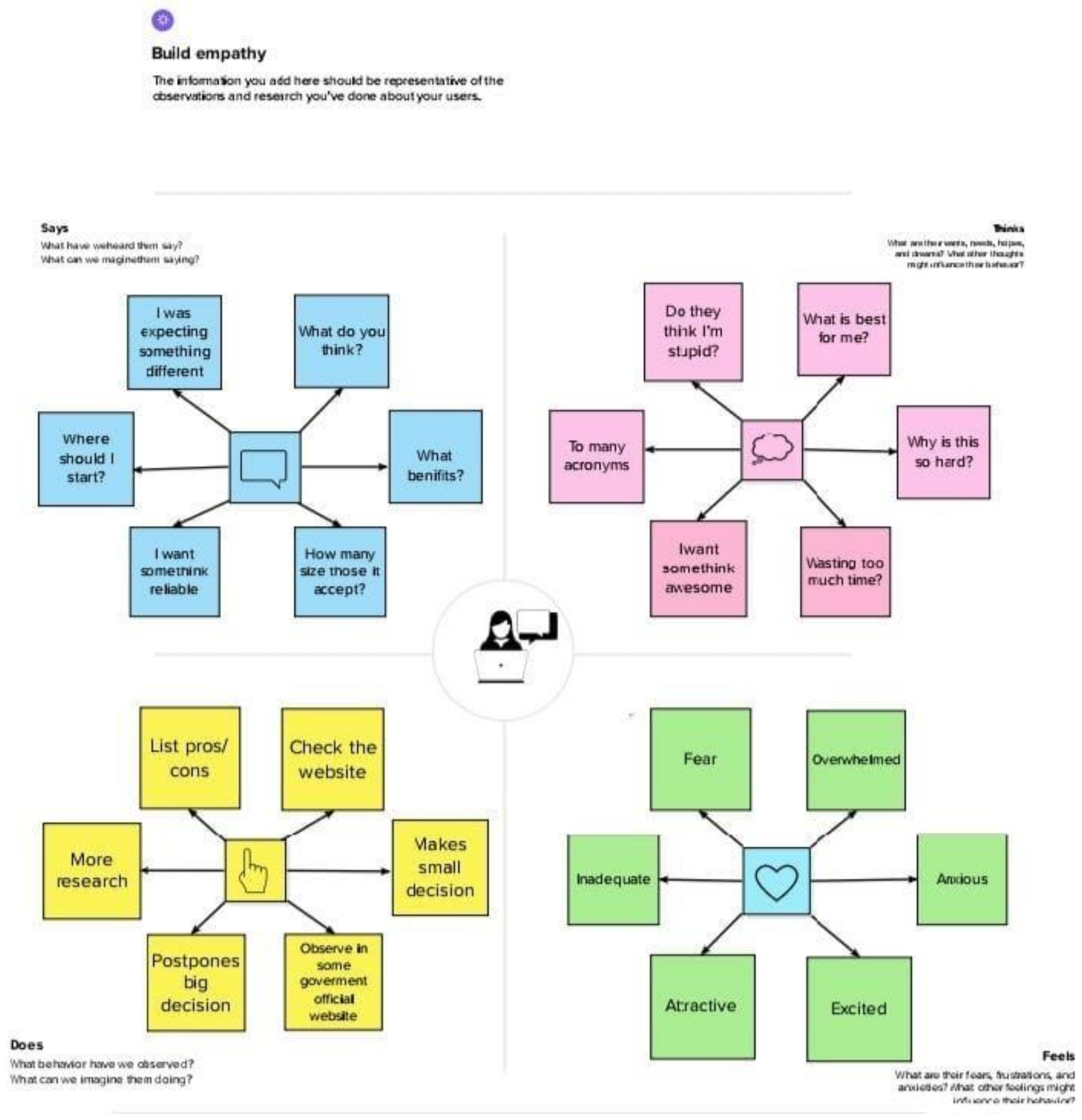

Get DAO
Data Access Object (DAO)
Get entities from DB/ Persist changes back to DB
Entities
Get / Set entities fields value
Room Database
Android app

1.2 Purpose:

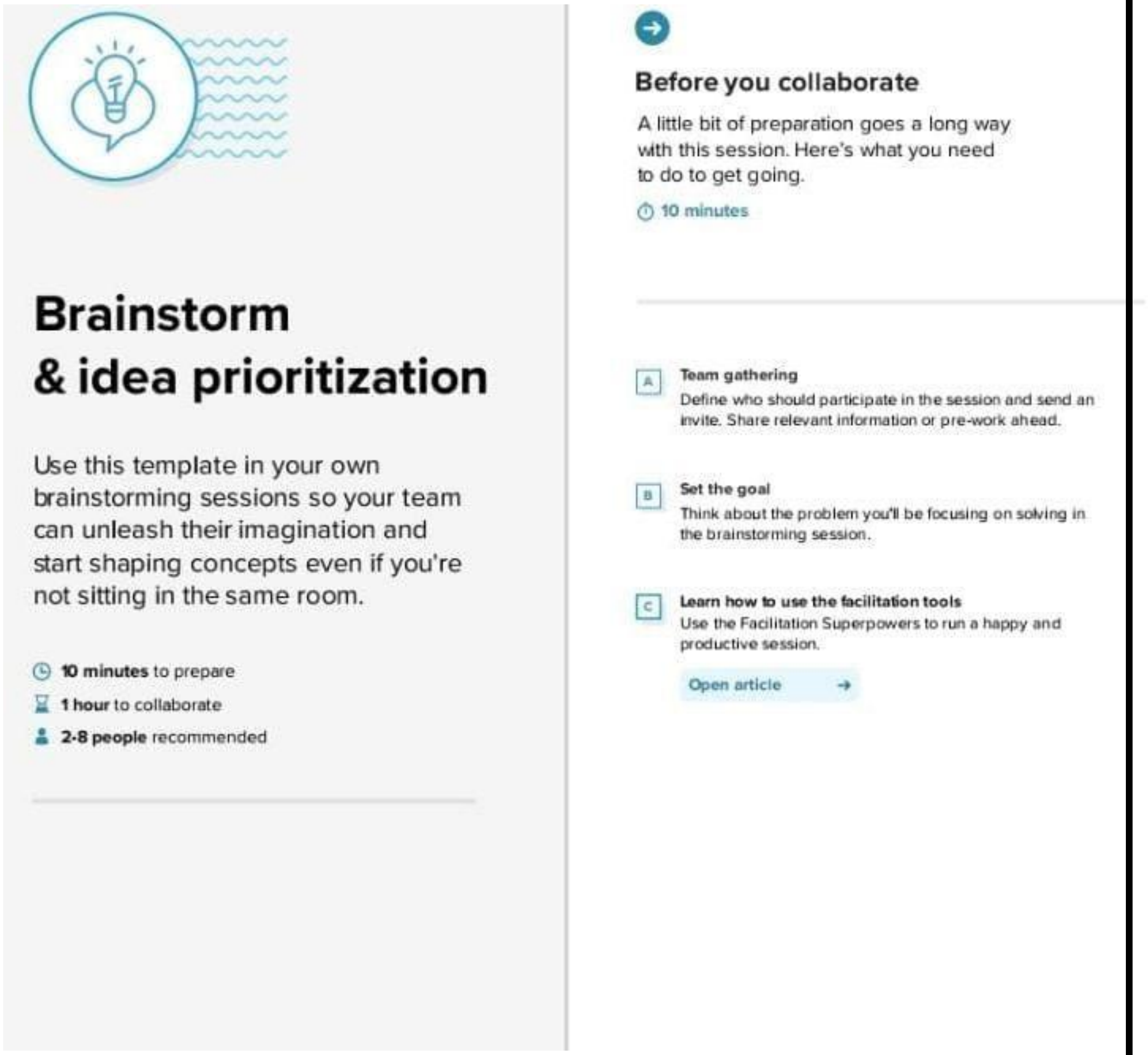
- Survey results provide insights on trends that health care providers can apply in their own practices and that the diabetes community can use to reach populations affected by diabetes. Data from the National Diabetes Survey may complement statistics on diabetes prevalence and cost collected by other organizations.

2. Problem Definition & Design Thinking

2.1 Empathy Map:


Build empathy
The information you add here should be representative of the observations and research you've done about your users.
Says
What have we heard them say?
What can we maginethem saying?
I was expecting something different
What do you think?
Where should I start?
What benifits?
I want somethink reliable
How many size those it accept?
Thinks
What are their wants, needs, hopes, and dreams? What other thoughts might influence their behavior?
Do they think I'm stupid?
What is best for me?
To many acronyms
Why is this so hard?
Iwant somethink awesome
Wasting too much time?
List pros/ cons
Check the website
More research
Makes small decision
Postpones big decision
Observe in some goverment official website
Does
What behavior have we observed?
What can we imagine them doing?
Fear
Overwhelmed
Inadequate
Anxious
Atractive
Excited
Feels
What are their fears, frustrations, and anxieties? What other feelings might influence their behavior?

2.2 Ideation & Brainstorming Map:



Brainstorm
& idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare

1 hour to collaborate

2-8 people recommended

Before you collaborate

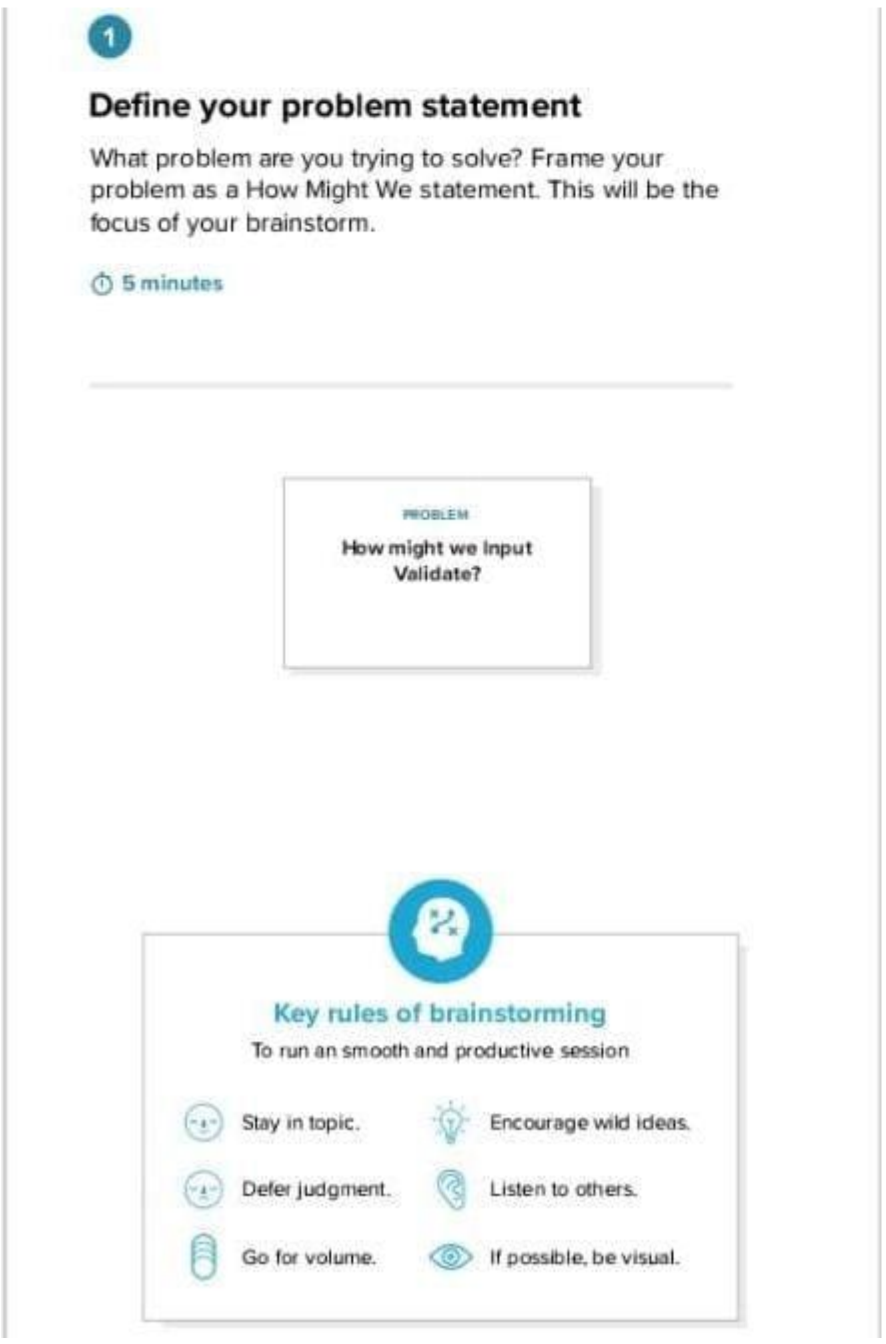
A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

A **Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B **Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.

C **Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.



1

Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

How might we Input Validate?

Key rules of brainstorming

To run an smooth and productive session

Stay in topic.	Encourage wild ideas.
Defer judgment.	Listen to others.
Go for volume.	If possible, be visual.

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

S.Lokeswari

- More research for the websites
- Avoid taking too much time for the website processing
- Problem that may arise due to upper case and lower case

M.Aruna

- Adding some pop up windows
- Adding the people review session
- Problems arises if letter are given instead of numbers

S.Abisha

- Create a clear guidelines
- Content marketing visual search
- problem arise during uploading the document

R.M.Shanmugapriya

- Adding voice features
- Too many acronyms usingthe websites
- problem arise in updating user image

T.Bagathiswari

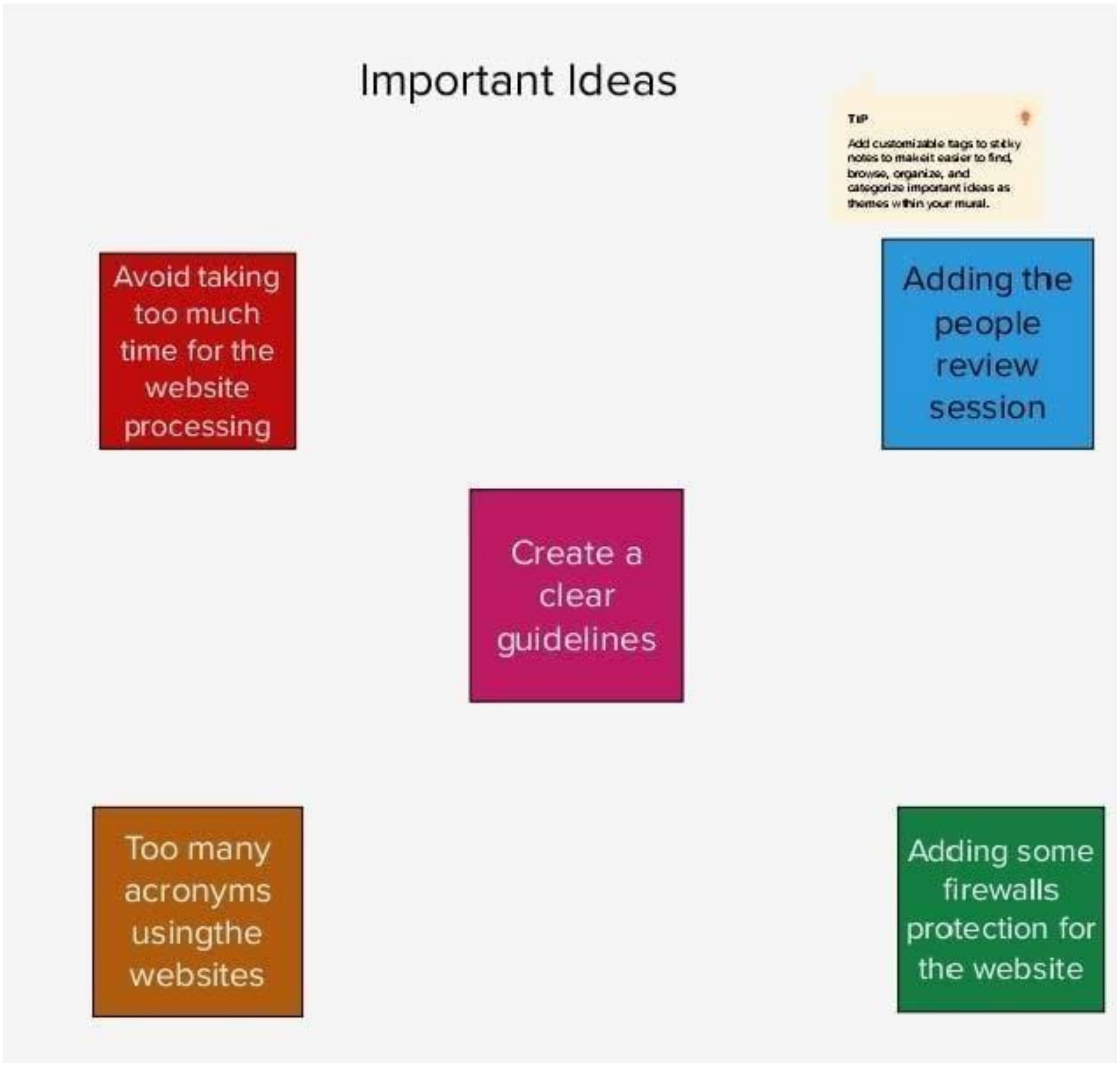
- Some problem arise when changing address of user
- Adding some firewalls protection for the website
- Adding some terms and conditions

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes



Important Ideas

TIP

Add customizable tags to sticky notes to makeit easier to find, browse, organize, and categorize important ideas as themes within your mural.

Avoid taking too much time for the website processing

Adding the people review session

Create a clear guidelines

Too many acronyms usingthe websites

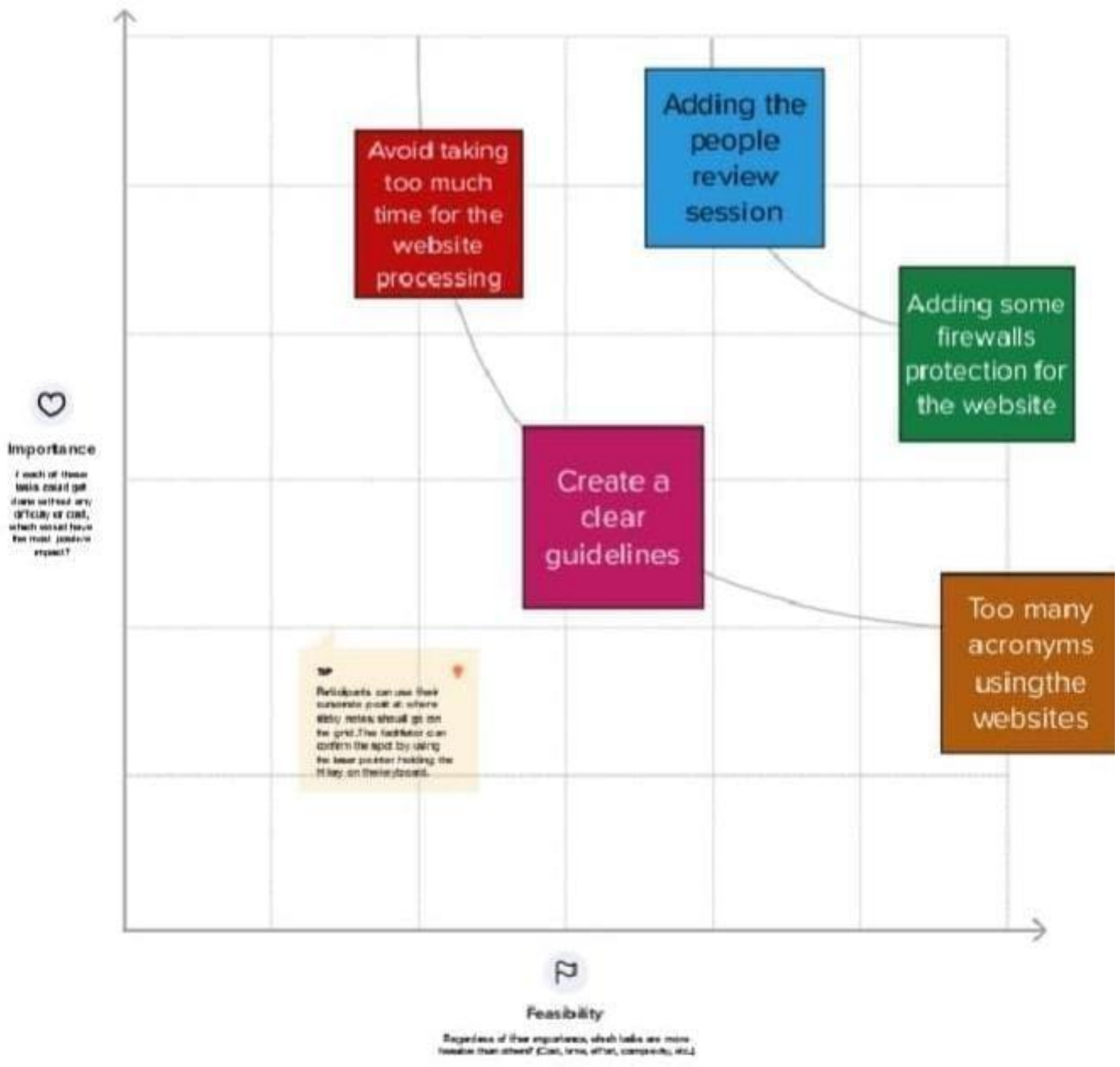
Adding some firewalls protection for the website

4

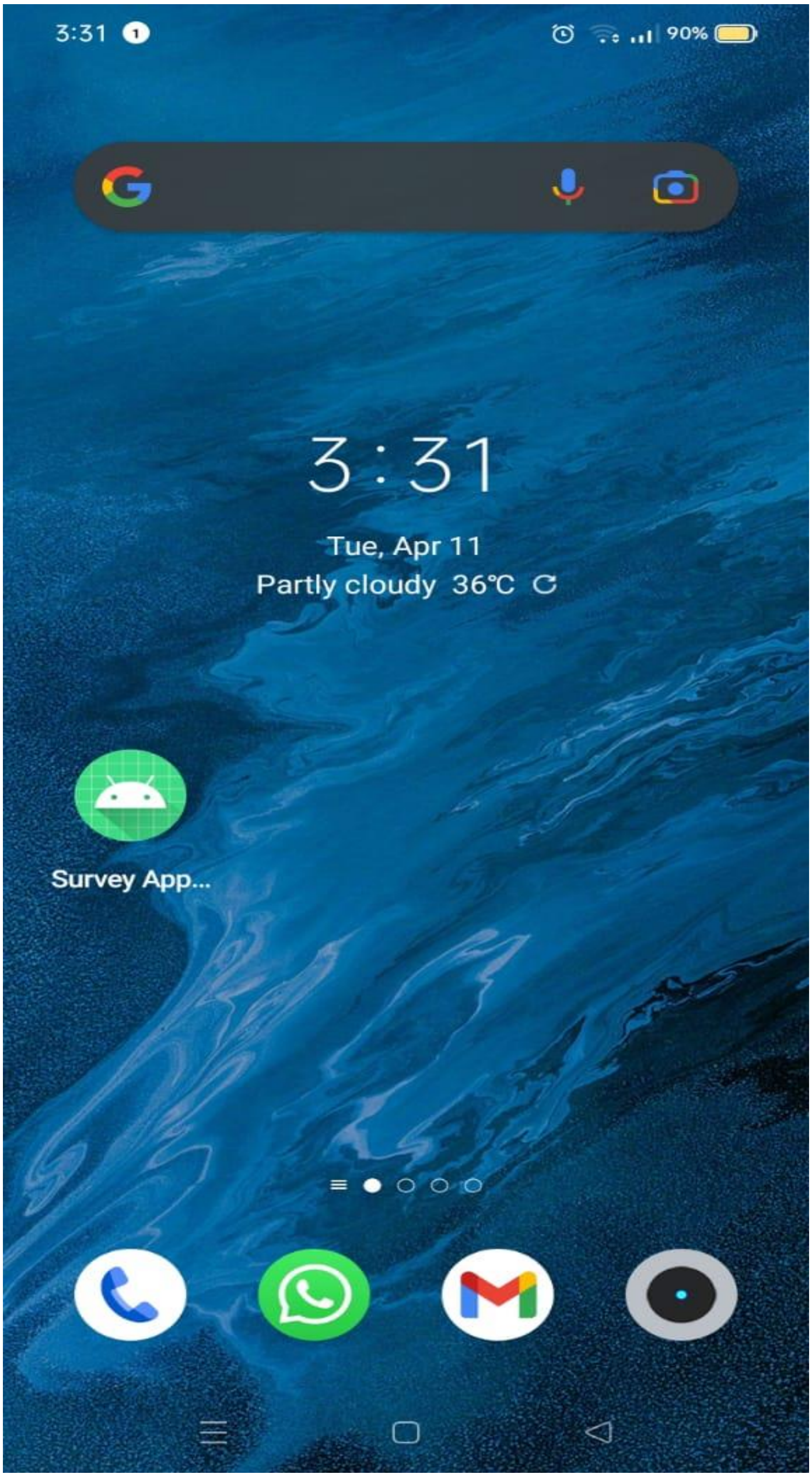
Prioritize

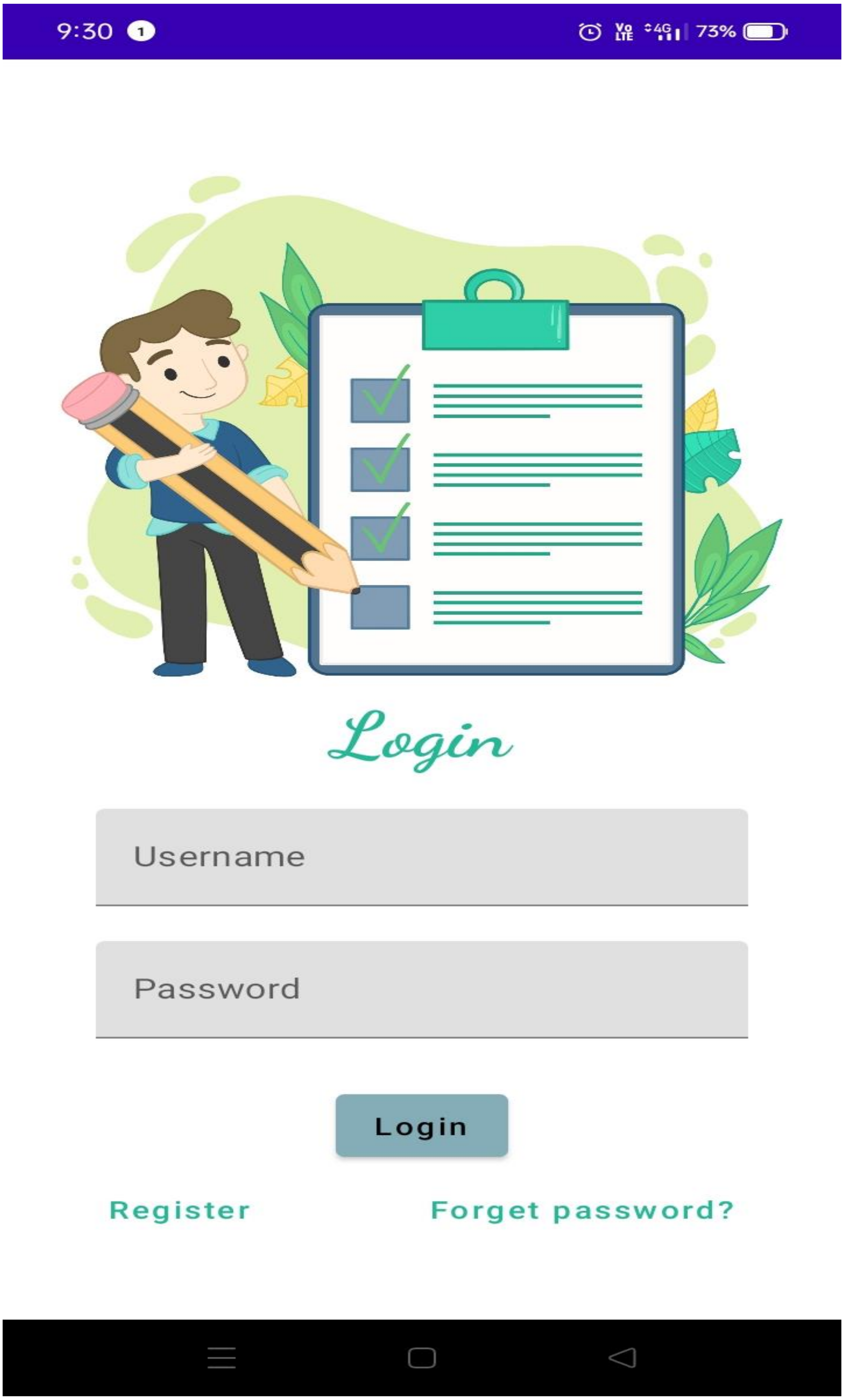
Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

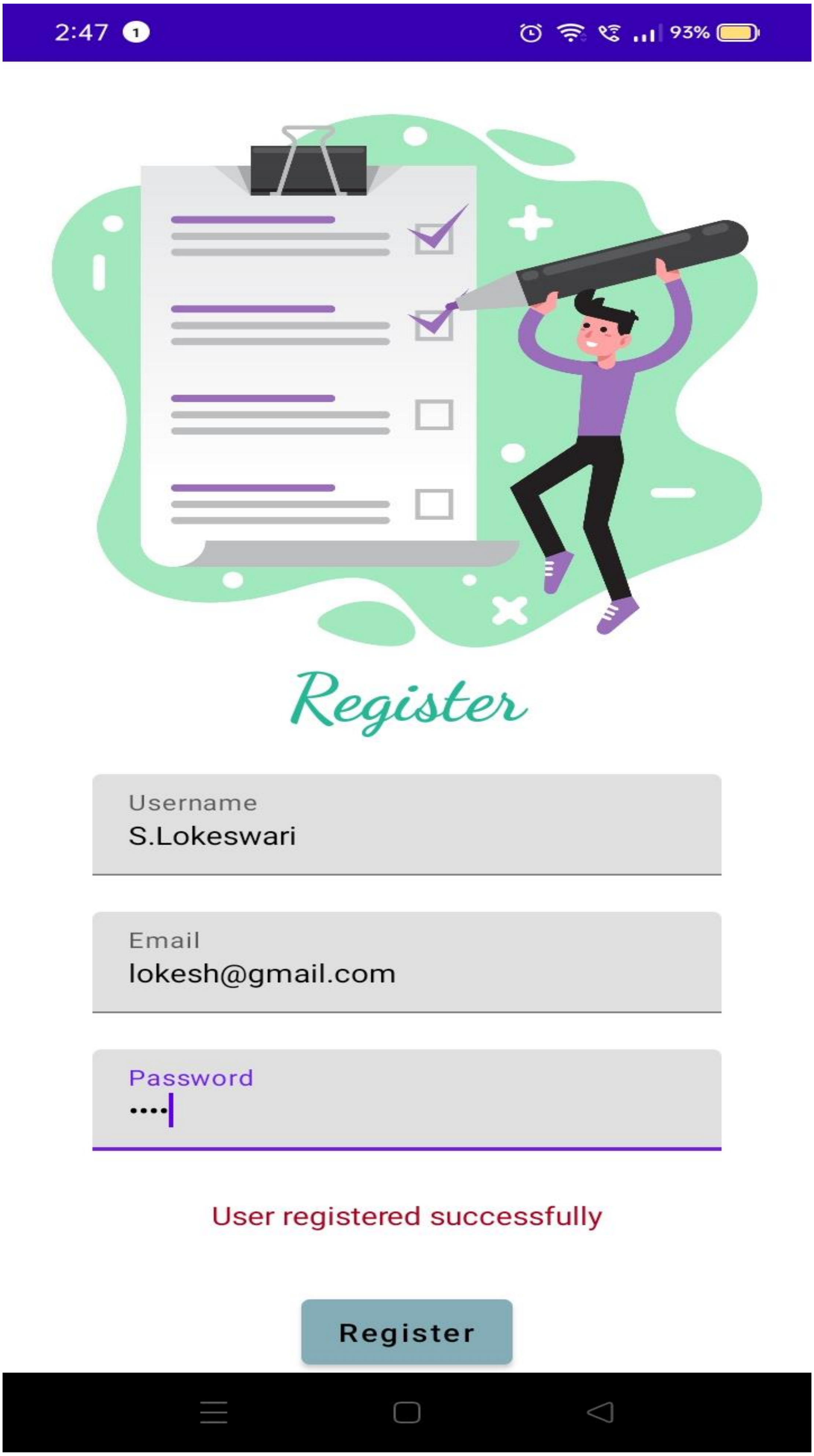
20 minutes


Avoid taking too much time for the website processing
Adding the people review session
Adding some firewalls protection for the website
Create a clear guidelines
Too many acronyms usingthe websites
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
Tip
Participants can use their cursor to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the L key on their keyboard.
Feasibility
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

3. RESUL




9:30
73%
Login
Username
Password
Login
Register
Forget password?


2:47

93%

Register

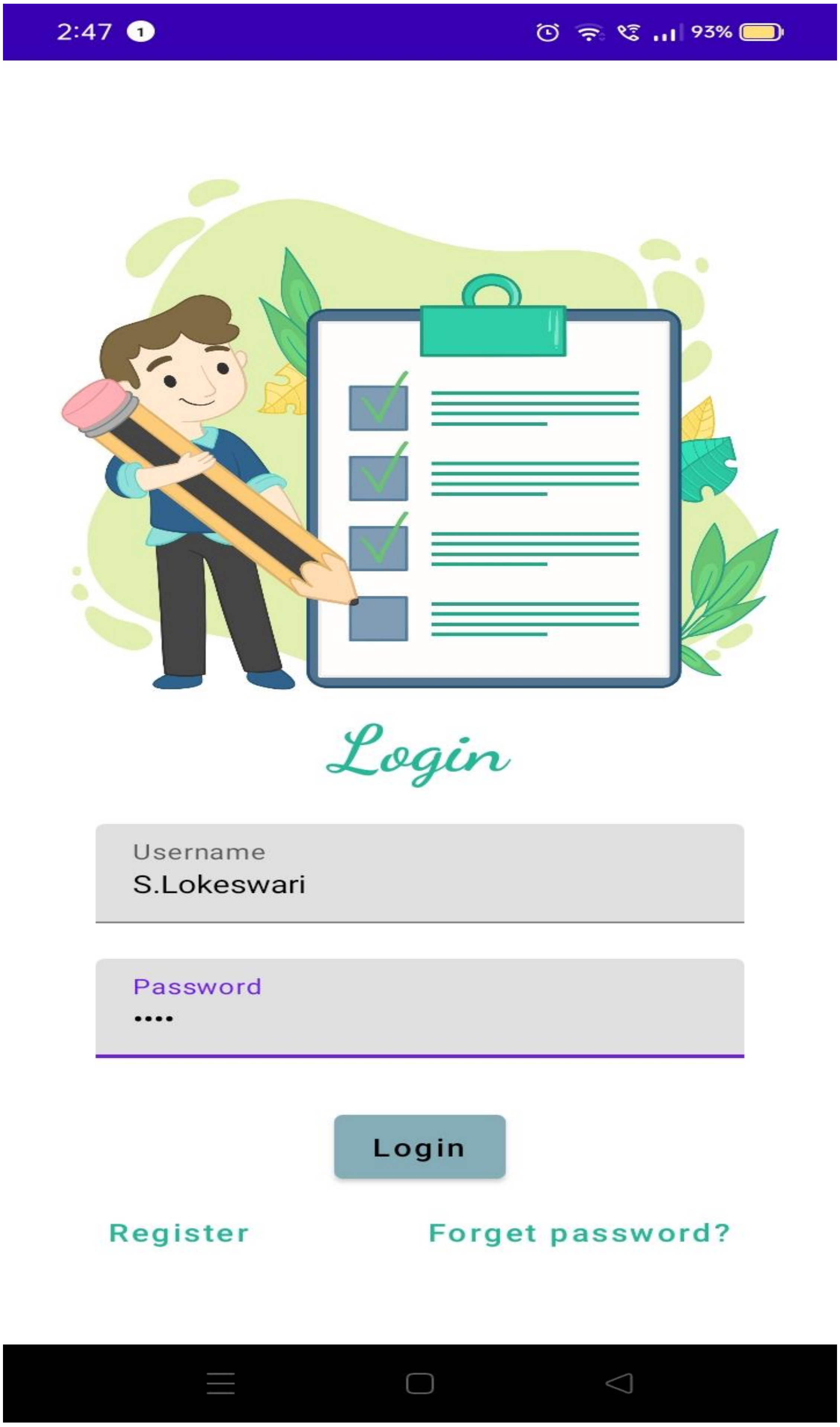
Username
S.Lokeswari

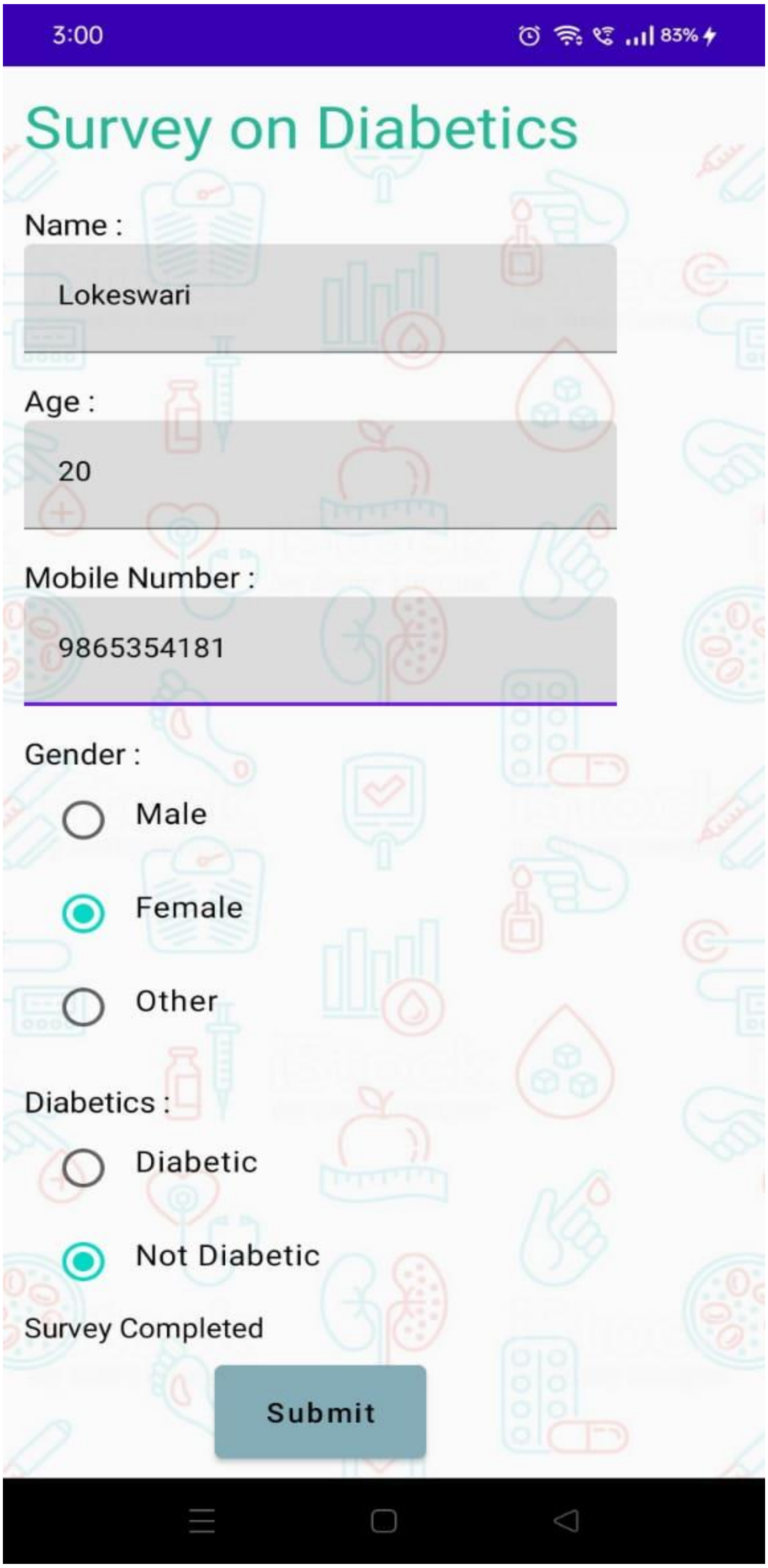
Email
lokesh@gmail.com

Password
••••

User registered successfully

Register


2:47
93%
Login
Username
S.Lokeswari
Password
••••
Login
Register
Forget password?



3:00 83%

Survey on Diabetics

Name : Lokeswari

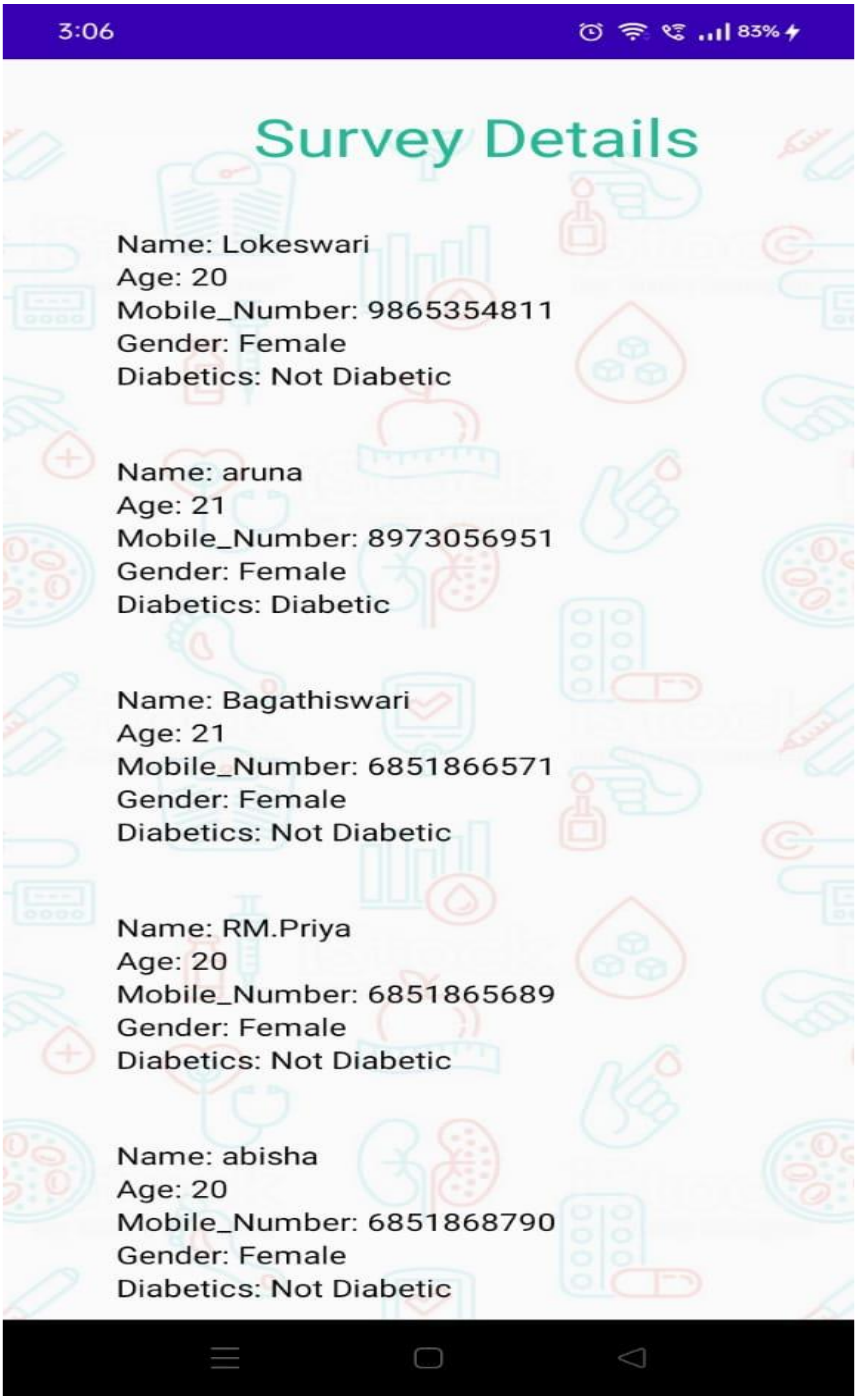
Age : 20

Mobile Number : 9865354181

Gender :
Male [✓ Female] Other

Diabetics :
Diabetic [✓ Not Diabetic]

Survey Completed


3:06 83%

Survey Details

Name: Lokeswari
Age: 20
Mobile_Number: 9865354811
Gender: Female
Diabetics: Not Diabetic

Name: aruna
Age: 21
Mobile_Number: 8973056951
Gender: Female
Diabetics: Diabetic

Name: Bagathiswari
Age: 21
Mobile_Number: 6851866571
Gender: Female
Diabetics: Not Diabetic

Name: RM.Priya
Age: 20
Mobile_Number: 6851865689
Gender: Female
Diabetics: Not Diabetic

Name: abisha
Age: 20
Mobile_Number: 6851868790
Gender: Female
Diabetics: Not Diabetic

4. ADVANTAGES & DISADVANTAGES

Advantages:

1. Easy to find out how many diabetics patients there are.
2. If the number of diabetic patients is high, steps should be taken to control it.
3. Diabetic passions are easy to categorize by age.
4. If the number of diabetic patients is high, create awareness videos and take steps to control them

Disadvantages:

1. No diabetic monitor facility.
2. No features for diabetic control.

3.If diabetics are high, there is no facility like automatically sending message to the government.

4.There is no facility to submit the diabetics report.

5.APPLICATIONS:

1.**Hospital** - Applied to get the information about the patient.

2.**Airport** - Applied old-age, adults those whom are willing to travel in aeroplane.

3.**School -** Apply the school to give the special consideration for the students.

4. **Employment office** - Apply the employment office to give the special consideration for the employee.

6.CONCLUSION:

*We have created a digital survey application project to easily survey how many people are affected by diabetes. Government can use this survey application in digital format and take many steps. Many facilities can be added in this survey application

* Diabetic patient counts can be easily stored in digital format. This application will be useful for the government to survey the diabetic patient and therefore it is easy to think about what steps can be taken to control it.

7.FUTURE SCOPE:

1. Add the facility to the monitor diabetic patient.

2.Add the features for diabetic control.

3.If diabetics are high, there is add the facility like automatically sending message to the government.

4. Add the facility to submit the diabetics report.

8.APPENDIX:

***AndroidManifest.xml:**

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/Theme.SurveyApplication"
        tools:targetApi="31">
        <activity
            android:name=".RegisterActivity"
            android:exported="false"
            android:label="@string/title_activity_register"
            android:theme="@style/Theme.SurveyApplication" />
        <activity
            android:name=".MainActivity"
            android:exported="false"
            android:label="MainActivity"
            android:theme="@style/Theme.SurveyApplication" />
        <activity
            android:name=".AdminActivity"
            android:exported="false"
            android:label="@string/title_activity_admin"
            android:theme="@style/Theme.SurveyApplication" />
        <activity
            android:name=".LoginActivity"
            android:exported="true"
            android:label="@string/app_name"
            android:theme="@style/Theme.SurveyApplication">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

*RegisterActivity:

```
package com.example.surveyapplication
import android.content.Context
import android.content.Intent
import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.Image
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.*
import androidx.compose.material.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.text.font.FontFamily
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.text.input.PasswordVisualTransformation
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import androidx.core.content.ContextCompat

class RegisterActivity : ComponentActivity() {
    private lateinit var databaseHelper: UserDatabaseHelper
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        databaseHelper = UserDatabaseHelper(this)
        setContent {


            RegistrationScreen(this,databaseHelper)


        }
    }
}

@Composable
fun RegistrationScreen(context: Context, databaseHelper: UserDatabaseHelper) {
```

```
var username by remember { mutableStateOf("") }
var password by remember { mutableStateOf("") }
var email by remember { mutableStateOf("") }
var error by remember { mutableStateOf("") }

Column(
    modifier = Modifier.fillMaxSize().background(Color.White),
    horizontalAlignment = Alignment.CenterHorizontally,
    verticalArrangement = Arrangement.Center
) {

    Image(painterResource(id = R.drawable.survey_signup), contentDescription = "")

    Text(
        fontSize = 36.sp,
        fontWeight = FontWeight.ExtraBold,
        fontFamily = FontFamily.Cursive,
        color = Color(0xFF25b897),
        text = "Register"
    )

    Spacer(modifier = Modifier.height(10.dp))
    TextField(
        value = username,
        onValueChange = { username = it },
        label = { Text("Username") },
        modifier = Modifier
            .padding(10.dp)
            .width(280.dp)

    )

    TextField(
    value = email,
    onValueChange = { email = it },
    label = { Text("Email") },
    modifier = Modifier
        .padding(10.dp)
        .width(280.dp)
)

TextField(
    value = password,
    onValueChange = { password = it },
    label = { Text("Password") },
    visualTransformation = PasswordVisualTransformation(),
    modifier = Modifier
        .padding(10.dp)
        .width(280.dp)
)



if (error.isNotEmpty()) {
    Text(
        text = error,
        color = MaterialTheme.colors.error,
        modifier = Modifier.padding(vertical = 16.dp)
    )
}

Button(
    onClick = {
        if (username.isNotEmpty() && password.isNotEmpty() && email.isNotEmpty()) {
            val user = User(
                id = null,
                firstName = username,
                lastName = null,
                email = email,
                password = password
            )
            databaseHelper.insertUser(user)
            error = "User registered successfully"
            // Start LoginActivity using the current context
            context.startActivity(
                Intent(
                    context,
                    LoginActivity::class.java
                )
            )

        } else {
            error = "Please fill all fields"
        }
    },
    colors = ButtonDefaults.buttonColors(backgroundColor = Color(0xFF84adb8)),
    modifier = Modifier.padding(top = 16.dp),

    ) {
    Text(text = "Register")
}
Spacer(modifier = Modifier.width(10.dp))
Spacer(modifier = Modifier.height(10.dp))

Row {
    Text(
        modifier = Modifier.padding(top = 14.dp), text = "Have an account?"
    )
    TextButton(onClick = {
        context.startActivity(
                Intent(
                    context,
                    LoginActivity::class.java
                )
            )
        })

        {
            Spacer(modifier = Modifier.width(10.dp))
            Text( color = Color(0xFF25b897),text = "Log in")
        }
    }
  }
}
private fun startLoginActivity(context: Context) {
    val intent = Intent(context, LoginActivity::class.java)
    ContextCompat.startActivity(context, intent, null)
}
```

*LoginActivity:

```
package com.example.surveyapplication


import android.content.Context
import android.content.Intent
import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.Image
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.*
import androidx.compose.material.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.text.font.FontFamily
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.text.input.PasswordVisualTransformation
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import androidx.core.content.ContextCompat

class LoginActivity : ComponentActivity() {
    private lateinit var databaseHelper: UserDatabaseHelper
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        databaseHelper = UserDatabaseHelper(this)
        setContent {


            LoginScreen(this, databaseHelper)


        }
    }
}

@Composable
fun LoginScreen(context: Context, databaseHelper: UserDatabaseHelper) {

    var username by remember { mutableStateOf("") }
    var password by remember { mutableStateOf("") }
    var error by remember { mutableStateOf("") }

    Column(
        modifier = Modifier.fillMaxSize().background(Color.White),
        horizontalAlignment = Alignment.CenterHorizontally,
        verticalArrangement = Arrangement.Center
    ) {

        Image(painterResource(id = R.drawable.survey_login), contentDescription = "")

        Text(
            fontSize = 36.sp,
            fontWeight = FontWeight.ExtraBold,
            fontFamily = FontFamily.Cursive,
            color = Color(0xFF25b897),
            text = "Login"
        )
        Spacer(modifier = Modifier.height(10.dp))

        TextField(
            value = username,
            onValueChange = { username = it },
            label = { Text("Username") },
            modifier = Modifier
                .padding(10.dp)
                .width(280.dp)
        )

        TextField(
            value = password,
            onValueChange = { password = it },
            label = { Text("Password") },
            visualTransformation = PasswordVisualTransformation(),
            modifier = Modifier
                .padding(10.dp)
                .width(280.dp)
```

```
)

if (error.isNotEmpty()) {
    Text(
        text = error,
        color = MaterialTheme.colors.error,
        modifier = Modifier.padding(vertical = 16.dp)
    )
}

Button(
    onClick = {
        if (username.isNotEmpty() && password.isNotEmpty()) {
            val user = databaseHelper.getUserByUsername(username)
            if (user != null && user.password == password) {
                error = "Successfully log in"
                context.startActivity(
                    Intent(
                        context,
                        MainActivity::class.java
                    )
                )
                //onLoginSuccess()
            }
            if (user != null && user.password == "admin") {
                error = "Successfully log in"
                context.startActivity(
                    Intent(
                        context,
                        AdminActivity::class.java
                    )
                )
            }
```

```
                else {
                    error =  "Invalid username or password"
                }

            } else {
                error = "Please fill all fields"
            }
        },
        colors = ButtonDefaults.buttonColors(backgroundColor = Color(0xFF84adb8)),
        modifier = Modifier.padding(top = 16.dp)
    ) {
        Text(text = "Login")
    }
    Row {
        TextButton(onClick = {context.startActivity(
            Intent(
                context,
                RegisterActivity::class.java
            )
        )}
        )
        { Text(color = Color(0xFF25b897),text = "Register") }
        TextButton(onClick = {
        })

        {
            Spacer(modifier = Modifier.width(60.dp))
            Text(color = Color(0xFF25b897),text = "Forget password?")
        }
    }
  }
}
private fun startMainPage(context: Context) {
  val intent = Intent(context, MainActivity::class.java)
  ContextCompat.startActivity(context, intent, null)
}
```

*MainActivity:

```
package com.example.surveyapplication


import android.content.Context
import android.content.Intent
import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.Image
import androidx.compose.foundation.layout.*
import androidx.compose.material.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.layout.ContentScale
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp


class MainActivity : ComponentActivity() {
    private lateinit var databaseHelper: SurveyDatabaseHelper
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        databaseHelper = SurveyDatabaseHelper(this)
        setContent {
            FormScreen(this, databaseHelper)
        }
    }
}

@Composable
fun FormScreen(  context: Context, databaseHelper: SurveyDatabaseHelper) {

    Image(
        painterResource(id = R.drawable.background), contentDescription = "",
        alpha =0.1F,
        contentScale = ContentScale.FillHeight,
        modifier = Modifier.padding(top = 40.dp)
    )




    // Define state for form fields
    var name by remember { mutableStateOf("") }
    var age by remember { mutableStateOf("") }
    var mobileNumber by remember { mutableStateOf("") }
    var genderOptions = listOf("Male", "Female", "Other")
    var selectedGender by remember { mutableStateOf("") }
    var error by remember { mutableStateOf("") }
    var diabeticsOptions = listOf("Diabetic", "Not Diabetic")
    var selectedDiabetics by remember { mutableStateOf("") }

    Column(
        modifier = Modifier.padding(10.dp),
        horizontalAlignment = Alignment.Start,
        verticalArrangement = Arrangement.SpaceEvenly
    ) {
```

```
Text(
    fontSize = 30.sp,
    textAlign = TextAlign.Center,
    text = "Survey on Diabetics",
    color = Color(0xFF25b897)
)

Spacer(modifier = Modifier.height(20.dp))

Text(text = "Name :", fontSize = 15.sp)
TextField(
    value = name,
    onValueChange = { name = it },
)

Spacer(modifier = Modifier.height(14.dp))

Text(text = "Age :", fontSize = 15.sp)
TextField(
    value = age,
    onValueChange = { age = it },
)

Spacer(modifier = Modifier.height(14.dp))

Text(text = "Mobile Number :", fontSize = 15.sp)
TextField(
    value = mobileNumber,
    onValueChange = { mobileNumber = it },
)

Spacer(modifier = Modifier.height(14.dp))
```

```
        Text(text = "Gender :", fontSize = 15.sp)
        RadioGroup(
            options = genderOptions,
            selectedOption = selectedGender,
            onSelectedChange = { selectedGender = it }
        )

        Spacer(modifier = Modifier.height(14.dp))

        Text(text = "Diabetics :", fontSize = 15.sp)
        RadioGroup(
            options = diabeticsOptions,
            selectedOption = selectedDiabetics,
            onSelectedChange = { selectedDiabetics = it }
        )

        Text(
            text = error,
            textAlign = TextAlign.Center,
            modifier = Modifier.padding(bottom = 10.dp)
        )
        // Display Submit button
        Button(
            onClick = {  if (name.isNotEmpty() && age.isNotEmpty() && mobileNumber.isNotEmpty()
&& genderOptions.isNotEmpty() && diabeticsOptions.isNotEmpty()){
                val survey = Survey(
                    id = null,
                    name = name,
                    age = age,
                    mobileNumber = mobileNumber,
                    gender = selectedGender,
                    diabetics = selectedDiabetics
                )
                databaseHelper.insertSurvey(survey)
                error="Survey Completed"
            context.startActivity(
                Intent(
                    context,
                    AdminActivity::class.java
                )
            )

            } else {
                "Please fill all fields"
            }
        },
        colors = ButtonDefaults.buttonColors(backgroundColor = Color(0xFF84adb8)),
        modifier = Modifier.padding(start = 90.dp).size(height = 900.dp, width = 100.dp)
    ) {
        Text(text = "Submit")
    }
  }
}
@Composable
fun RadioGroup(
  options: List<String>,
  selectedOption: String?,
  onSelectedChange: (String) -> Unit
) {
  Column {
    options.forEach { option ->
      Row(
        Modifier
          .fillMaxWidth()
          .padding(horizontal = 4.dp)
      ) {
            RadioButton(
               selected = option == selectedOption,
               onClick = { onSelectedChange(option) }
            )
            Text(
               text = option,
               style = MaterialTheme.typography.body1.merge(),
               modifier = Modifier.padding(top = 10.dp),
               fontSize = 15.sp
            )
         }

      }
   }
}
```

Thank-you