Week 2 Assignment – Blood Donation App

By Dyutin.R 20BCG10060

Main Activity:

```
package com.example.blooddonationapp
import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent

import androidx.compose.runtime.Composable

import androidx.navigation.NavHostController
import androidx.navigation.compose.NavHost
import androidx.navigation.compose.composable
import androidx.navigation.compose.rememberNavController


sealed class Destination(val route:String){
    object Main: Destination("Main")
    object Login: Destination("Login")
    object Signup: Destination("Signup")
    object Home: Destination("Home")
}
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            val navController = rememberNavController()
            NavigationAppHost(navController = navController)

        }
    }
}
@Composable
fun NavigationAppHost(navController: NavHostController){
    NavHost(navController =navController , startDestination = "Home"){
        composable(Destination.Main.route){
            Mainscreen()
        }
        composable(Destination.Login.route) {
            LoginScreen(navController)
        }
        composable(Destination.Signup.route) {
            SignUpScreen(navController)
        }
        composable(Destination.Home.route) {
            HomeScreen(navController)
        }
    }

}
```

Home Screen:

```
package com.example.blooddonationapp


import androidx.compose.foundation.Image
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Box
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.height
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.layout.size
import androidx.compose.foundation.text.KeyboardOptions
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.Visibility
import androidx.compose.material.icons.filled.VisibilityOff
import androidx.compose.material3.Button
import androidx.compose.material3.ButtonDefaults
import androidx.compose.material3.Card
import androidx.compose.material3.CardDefaults
import androidx.compose.material3.ExperimentalMaterial3Api
import androidx.compose.material3.Icon
import androidx.compose.material3.IconButton
import androidx.compose.material3.OutlinedTextField
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.runtime.getValue
import androidx.compose.runtime.mutableStateOf
import androidx.compose.runtime.remember
import androidx.compose.runtime.setValue
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Brush
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.text.ExperimentalTextApi
import androidx.compose.ui.text.SpanStyle
import androidx.compose.ui.text.TextStyle
import androidx.compose.ui.text.buildAnnotatedString
import androidx.compose.ui.text.font.FontFamily
import androidx.compose.ui.text.input.KeyboardType
import androidx.compose.ui.text.input.PasswordVisualTransformation
import androidx.compose.ui.text.input.VisualTransformation
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.text.withStyle
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp

import androidx.navigation.NavHostController

@OptIn(ExperimentalTextApi::class, ExperimentalMaterial3Api::class)
@Composable
fun HomeScreen(navController: NavHostController){

    Box(
        modifier = Modifier
```

```
            .fillMaxSize()
            .background(Color(0xFFB04759))
    )
    {
        Card(
        elevation = CardDefaults.cardElevation(
            defaultElevation = 10.dp
        ),
        modifier = Modifier.padding(30.dp).align(Alignment.Center),colors =
CardDefaults.cardColors(
            containerColor = Color(0xFFF6E7D8)
            )
    )
    {
        Column( modifier = Modifier.padding(horizontal = 20.dp, vertical =
0.dp),
            verticalArrangement = Arrangement.Center,
            horizontalAlignment = Alignment.CenterHorizontally) {
            Spacer(modifier = Modifier.height(60.dp))
            Image(painter = painterResource(id = R.drawable.blood),
contentDescription ="",
                modifier = Modifier.size(200.dp,100.dp ))
            Spacer(modifier = Modifier.height(10.dp))
            Text(text = "Donate Blood; Save Lives",
                style = TextStyle(
                    color = Color(0xFFB04759),
                    fontSize =30.sp,
                    fontFamily = FontFamily.Monospace,
                    textAlign = TextAlign.Center

                )
            )

            Spacer(modifier = Modifier.height(10.dp))
            Row() {
                Button(onClick = {
navController.navigate(Destination.Signup.route) },colors =
ButtonDefaults.buttonColors(Color(0xFFB04759) ),
                    modifier = Modifier
                        .fillMaxWidth()
                        .padding(horizontal = 20.dp, vertical = 20.dp)
                ) {
                    Text(text = "Sign Up ")
                }

            }

            Row() {
                Button(onClick = {
navController.navigate(Destination.Login.route) },colors =
ButtonDefaults.buttonColors(Color(0xFFB04759) ),
                    modifier = Modifier
                        .fillMaxWidth()
                        .padding(horizontal = 20.dp, vertical = 20.dp)
                ) {
                    Text(text = "Log In ")
                }

            }
        }
    }
```

```
    }
    }
```

Sign up screen:

```
package com.example.blooddonationapp


import androidx.compose.foundation.Image
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Box
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.height
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.layout.size
import androidx.compose.foundation.text.KeyboardOptions
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.Visibility
import androidx.compose.material.icons.filled.VisibilityOff
import androidx.compose.material3.Button
import androidx.compose.material3.ButtonDefaults
import androidx.compose.material3.Card
import androidx.compose.material3.CardDefaults
import androidx.compose.material3.ExperimentalMaterial3Api
import androidx.compose.material3.Icon
import androidx.compose.material3.IconButton
import androidx.compose.material3.OutlinedTextField
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.runtime.getValue
import androidx.compose.runtime.mutableStateOf
import androidx.compose.runtime.remember
import androidx.compose.runtime.setValue
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Brush
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.text.ExperimentalTextApi
import androidx.compose.ui.text.SpanStyle
import androidx.compose.ui.text.TextStyle
import androidx.compose.ui.text.buildAnnotatedString
import androidx.compose.ui.text.font.FontFamily
import androidx.compose.ui.text.input.KeyboardType
import androidx.compose.ui.text.input.PasswordVisualTransformation
import androidx.compose.ui.text.input.VisualTransformation
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.text.withStyle
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp

import androidx.navigation.NavHostController
```

```
@OptIn(ExperimentalTextApi::class, ExperimentalMaterial3Api::class)
@Composable
fun SignUpScreen(navController: NavHostController){

    var username by remember { mutableStateOf("") }
    var password by remember { mutableStateOf("") }
    var fullName by remember { mutableStateOf("") }
    var mobile by remember { mutableStateOf("") }
    var showPassword by remember { mutableStateOf(value = false) }
    Box(
        modifier = Modifier
            .fillMaxSize()
            .background(Color(0xFFB04759))
    )
    {
        Card(
        elevation = CardDefaults.cardElevation(
            defaultElevation = 10.dp
        ),
        modifier = Modifier.padding(30.dp),colors =
CardDefaults.cardColors(
            containerColor = Color(0xFFF6E7D8)
            )
    )
    {
        Column( modifier = Modifier.fillMaxSize().padding(horizontal =
20.dp, vertical = 0.dp),
            verticalArrangement = Arrangement.Top,
            horizontalAlignment = Alignment.CenterHorizontally) {
            Spacer(modifier = Modifier.height(60.dp))
            Image(painter = painterResource(id = R.drawable.blood),
contentDescription ="",
                modifier = Modifier.size(200.dp,100.dp ))
            Spacer(modifier = Modifier.height(10.dp))
            Text(text = "Sign Up",
                style = TextStyle(
                    color = Color(0xFFB04759),
                    fontSize =17.sp,
                    fontFamily = FontFamily.Monospace,
                    textAlign = TextAlign.Center

                )
            )

            Row() {
                OutlinedTextField(
                    value = mobile,
                    onValueChange = { mobile = it },
                    label = { Text(text = "Mobile Number or Email") },
                    modifier = Modifier.fillMaxWidth().padding(horizontal =
20.dp, vertical = 10.dp)
                )

            }
            Row() {
                OutlinedTextField(
                    value = fullName,
                    onValueChange = { fullName = it },
                    label = { Text(text = "FullName") },
                    modifier = Modifier.fillMaxWidth().padding(horizontal =
```

```
20.dp, vertical = 10.dp)
                )

            }
            Row() {
                OutlinedTextField(
                    value = username,
                    onValueChange = { username = it },
                    label = { Text(text = "Username") },
                    modifier = Modifier.fillMaxWidth().padding(horizontal =
20.dp, vertical = 10.dp)
                )

            }
            Row() {
                OutlinedTextField(
                    value = password,
                    onValueChange = { password = it },
                    label = { Text(text = "Password") },
                    visualTransformation = if (showPassword) {

                        VisualTransformation.None

                    } else {

                        PasswordVisualTransformation()

                    },
                    keyboardOptions = KeyboardOptions(keyboardType =
KeyboardType.Password),
                    trailingIcon = {
                        if (showPassword) {
                            IconButton(onClick = { showPassword = false })
{
                                Icon(
                                    imageVector = Icons.Filled.Visibility,
                                    contentDescription = "hide_password"
                                )
                            }
                        } else {
                            IconButton(
                                onClick = { showPassword = true }) {
                                Icon(
                                    imageVector =
Icons.Filled.VisibilityOff,
                                    contentDescription = "hide_password"
                                )
                            }
                        }
                    },
                    modifier = Modifier.fillMaxWidth().padding(horizontal =
20.dp, vertical = 10.dp)

                )
            }
            Text(text = buildAnnotatedString { append("People who use our
service may have uploaded your contact information to our Database.")
                withStyle(style = SpanStyle(color = Color.Blue)){
                    append(" Learn More")
                } } , Modifier.padding(horizontal = 20.dp, vertical =
0.dp))
```

```
            Text(text = buildAnnotatedString { append("\n" +
                    "By signing up, you agree to our ")
                withStyle(style = SpanStyle(color = Color.Blue )){
                    append("Terms , Privacy Policy ")
                }
                append(" and")
                withStyle(style = SpanStyle(color = Color.Blue)){
                    append(" Cookies Policy .")
                }                                   }
                , Modifier.padding(horizontal = 20.dp, vertical = 0.dp))
            Row() {
                Button(onClick = {
navController.navigate(Destination.Main.route) },colors =
ButtonDefaults.buttonColors(Color(0xFFB04759) ),
                    modifier = Modifier
                        .fillMaxWidth()
                        .padding(horizontal = 20.dp, vertical = 20.dp)
                ) {
                    Text(text = "Sign in ")
                }

            }
        }
    }
    }
    }
```

Login Screen:

```
package com.example.blooddonationapp


import androidx.compose.foundation.Image
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Box
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.height
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.layout.size
import androidx.compose.foundation.text.KeyboardOptions
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.Visibility
import androidx.compose.material.icons.filled.VisibilityOff
import androidx.compose.material3.Button
import androidx.compose.material3.ButtonDefaults
import androidx.compose.material3.Card
import androidx.compose.material3.CardDefaults
import androidx.compose.material3.ExperimentalMaterial3Api
import androidx.compose.material3.Icon
import androidx.compose.material3.IconButton
import androidx.compose.material3.OutlinedTextField
import androidx.compose.material3.Text
```

```
import androidx.compose.runtime.Composable
import androidx.compose.runtime.getValue
import androidx.compose.runtime.mutableStateOf
import androidx.compose.runtime.remember
import androidx.compose.runtime.setValue
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Brush
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.text.ExperimentalTextApi
import androidx.compose.ui.text.SpanStyle
import androidx.compose.ui.text.TextStyle
import androidx.compose.ui.text.buildAnnotatedString
import androidx.compose.ui.text.font.FontFamily
import androidx.compose.ui.text.input.KeyboardType
import androidx.compose.ui.text.input.PasswordVisualTransformation
import androidx.compose.ui.text.input.VisualTransformation
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.text.withStyle
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp

import androidx.navigation.NavHostController

@OptIn(ExperimentalTextApi::class, ExperimentalMaterial3Api::class)
@Composable
fun LoginScreen(navController: NavHostController){

    var username by remember { mutableStateOf("") }
    var password by remember { mutableStateOf("") }
    var showPassword by remember { mutableStateOf(value = false) }
    Box(
        modifier = Modifier
            .fillMaxSize()
            .background(Color(0xFFB04759))
    )
    {
        Card(
        elevation = CardDefaults.cardElevation(
            defaultElevation = 10.dp
        ),
            modifier =
Modifier.padding(30.dp).align(Alignment.Center),colors =
CardDefaults.cardColors(
            containerColor = Color(0xFFF6E7D8)
            )
    )
    {
        Column( modifier = Modifier.padding(horizontal = 20.dp, vertical =
0.dp),
            verticalArrangement = Arrangement.Top,
            horizontalAlignment = Alignment.CenterHorizontally) {
            Spacer(modifier = Modifier.height(60.dp))
            Image(painter = painterResource(id = R.drawable.blood),
contentDescription ="",
                modifier = Modifier.size(200.dp,100.dp ))
            Spacer(modifier = Modifier.height(10.dp))
            Text(text = "Login",
                style = TextStyle(
                    color = Color(0xFFB04759),
```

```
                    fontSize =24.sp,
                    fontFamily = FontFamily.Monospace,
                    textAlign = TextAlign.Center

                )
            )

            Row() {
                OutlinedTextField(
                    value = username,
                    onValueChange = { username = it },
                    label = { Text(text = "Username") },
                    modifier = Modifier.fillMaxWidth().padding(horizontal =
20.dp, vertical = 10.dp)
                )

            }
            Row() {
                OutlinedTextField(
                    value = password,
                    onValueChange = { password = it },
                    label = { Text(text = "Password") },
                    visualTransformation = if (showPassword) {

                        VisualTransformation.None

                    } else {

                        PasswordVisualTransformation()

                    },
                    keyboardOptions = KeyboardOptions(keyboardType =
KeyboardType.Password),
                    trailingIcon = {
                        if (showPassword) {
                            IconButton(onClick = { showPassword = false })
{

                                Icon(
                                    imageVector = Icons.Filled.Visibility,
                                    contentDescription = "hide_password"
                                )
                            }
                        } else {
                            IconButton(
                                onClick = { showPassword = true }) {
                                Icon(
                                    imageVector =
Icons.Filled.VisibilityOff,
                                    contentDescription = "hide_password"
                                )
                            }
                        }
                    },
                    modifier = Modifier.fillMaxWidth().padding(horizontal =
20.dp, vertical = 10.dp)

                )
            }
            Text(text = buildAnnotatedString { append("People who use our
service may have uploaded your contact information to our Database.")
                withStyle(style = SpanStyle(color = Color.Blue)){
```

```
                    append(" Learn More")
                } } , Modifier.padding(horizontal = 20.dp, vertical =
0.dp))
            Text(text = buildAnnotatedString { append("\n" +
                    "By signing up, you agree to our ")
                withStyle(style = SpanStyle(color = Color.Blue )){
                    append("Terms , Privacy Policy ")
                }
                append(" and")
                withStyle(style = SpanStyle(color = Color.Blue)){
                    append(" Cookies Policy .")
                }                                   }
                , Modifier.padding(horizontal = 20.dp, vertical = 0.dp))
            Row() {
                Button(onClick = {
navController.navigate(Destination.Main.route) },colors =
ButtonDefaults.buttonColors(Color(0xFFB04759) ),
                    modifier = Modifier
                        .fillMaxWidth()
                        .padding(horizontal = 20.dp, vertical = 20.dp)
                ) {
                    Text(text = "Login ")
                }

            }
        }
    }
    }
    }
```

Main Screen:

```
package com.example.blooddonationapp


import androidx.compose.foundation.BorderStroke
import androidx.compose.foundation.Image
import androidx.compose.foundation.background
import androidx.compose.foundation.border
import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Box
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.ExperimentalLayoutApi
import androidx.compose.foundation.layout.PaddingValues
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.consumedWindowInsets
import androidx.compose.foundation.layout.fillMaxHeight
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.height
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.layout.size
import androidx.compose.foundation.layout.width
import androidx.compose.foundation.lazy.LazyColumn
import androidx.compose.foundation.lazy.items
import androidx.compose.foundation.rememberScrollState
```

```
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.foundation.verticalScroll
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.Send
import androidx.compose.material3.Button
import androidx.compose.material3.ButtonDefaults
import androidx.compose.material3.Card
import androidx.compose.material3.CardDefaults
import androidx.compose.material3.ExperimentalMaterial3Api
import androidx.compose.material3.Icon
import androidx.compose.material3.Scaffold
import androidx.compose.material3.Text
import androidx.compose.material3.TextField
import androidx.compose.material3.TextFieldDefaults
import androidx.compose.material3.TopAppBar
import androidx.compose.runtime.Composable
import androidx.compose.runtime.MutableState
import androidx.compose.runtime.mutableStateOf
import androidx.compose.runtime.remember
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.drawBehind
import androidx.compose.ui.geometry.Offset
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.text.input.TextFieldValue
import androidx.compose.ui.text.input.VisualTransformation
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import android.app.DatePickerDialog
import android.widget.DatePicker
import androidx.compose.foundation.layout.*
import androidx.compose.material.*
import androidx.compose.ui.platform.LocalContext

import java.util.Calendar
import java.util.Date

data class DataModel(val value:String, val detail:String)


@Composable
fun Mainscreen() {
    CustomScaffold()
}

@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun CustomScaffold() {
    Scaffold(topBar = { CustomTopBar() },
        content = { pad -> MainContent(pad) }
    )
}

@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun CustomTopBar() {
```

```
    TopAppBar(
        title = {
            Row(
                verticalAlignment = Alignment.CenterVertically,
horizontalArrangement =
                Arrangement.Center, modifier = Modifier
                    .fillMaxWidth()
                    .padding(
                        end = 50
                            .dp
                    )
            ) {
                Text(text = "Donation App")
                Spacer(modifier = Modifier)

            }
        },
        modifier = Modifier.drawBehind {
            drawLine(
                Color.LightGray,
                Offset(0f, size.height),
                Offset(size.width, size.height),
                5f
            )
        }
    )
}


@OptIn(ExperimentalLayoutApi::class)
@Composable
fun MainContent(padding: PaddingValues)
{
    val primaryTextColor = remember {
        mutableStateOf(Color(115, 115, 115))
    }
    remember {
        mutableStateOf(Color(0, 55, 107))
    }
    val tertiaryTextColor = remember {
        mutableStateOf(Color.Black)
    }
    remember {
        mutableStateOf(Color(0, 149, 246))
    }
    remember {
        mutableStateOf(Color(0, 149, 246))
    }
    remember {
        mutableStateOf(Color(62, 174, 247, 255))
    }
    remember {
        mutableStateOf(Color(255, 255, 255))
    }
    remember {
        mutableStateOf(Color.White)
    }
    remember {
        mutableStateOf(Color.White)
    }
    val textFieldColor = remember {
```

```
        mutableStateOf(Color(250, 250, 250))
    }
    remember {
        mutableStateOf(Color(219, 219, 219))
    }
    val mobile = remember {
        mutableStateOf(TextFieldValue())
    }

    val fullName = remember {
        mutableStateOf(TextFieldValue())
    }

    val email = remember {
        mutableStateOf(TextFieldValue())
    }

    val address = remember {
        mutableStateOf(TextFieldValue())
    }

    val bloodType = remember {
        mutableStateOf(TextFieldValue())
    }
    val centre = remember {
        mutableStateOf(TextFieldValue())
    }


    val myDataList = mutableListOf<DataModel>()

    val showList = remember {
        mutableStateOf(false)
    }
    // Fetching the Local Context
    val mContext = LocalContext.current

    // Declaring integer values
    // for year, month and day
    val mYear: Int
    val mMonth: Int
    val mDay: Int

    // Initializing a Calendar
    val mCalendar = Calendar.getInstance()

    // Fetching current year, month and day
    mYear = mCalendar.get(Calendar.YEAR)
    mMonth = mCalendar.get(Calendar.MONTH)
    mDay = mCalendar.get(Calendar.DAY_OF_MONTH)

    mCalendar.time = Date()

    // Declaring a string value to
    // store date in string format
    val mDate = remember { mutableStateOf("") }
    Box(
        modifier = Modifier
            .fillMaxSize()
            .background(Color(0xFFF6E7D8))
    )
```

```
    {
        Card(
            elevation = CardDefaults.cardElevation(
                defaultElevation = 0.dp
            ),
            modifier = Modifier
                .padding(0.dp)
                .fillMaxHeight()
                .fillMaxSize()
                .fillMaxWidth(), colors = CardDefaults.cardColors(
                containerColor = Color(0xFFF6E7D8)
            )
        )
        {
            Column(
                modifier = Modifier
                    .fillMaxSize()
                    .padding(horizontal = 20.dp, vertical = 0.dp)
                    .verticalScroll(
                        rememberScrollState()
                    ),
                verticalArrangement = Arrangement.Top,
                horizontalAlignment = Alignment.CenterHorizontally
            ) {
                Column(
                    modifier = Modifier
                        .padding(20.dp)
                        .padding(padding)
                        .consumedWindowInsets(padding),
                    horizontalAlignment = Alignment.CenterHorizontally
                ) {
                    Image(
                        painterResource(id = R.drawable.blood),
                        contentDescription =
                        "instagram logo",
                        modifier = Modifier.size(width = 220.dp, height =
100.dp)
                    )
                    Spacer(modifier = Modifier.height(30.dp))
                    Text(
                        "Register Now! ",
                        color = primaryTextColor.value,
                        fontFamily =
                        FontFamily.SansSerif,
                        fontSize = 20.sp,
                        fontWeight = FontWeight.SemiBold,
                        textAlign = TextAlign.Center
                    )

                    Spacer(modifier = Modifier.height(40.dp))
                    CustomTextField(
                        modifier = Modifier.fillMaxWidth(),
                        mutableValue =
                        fullName,
                        label = "Full Name",
                        focusedColor = primaryTextColor.value,
                        textColor = tertiaryTextColor.value,
                        conColor = textFieldColor.value

                    )
                    Spacer(modifier = Modifier.height(10.dp))
```

```
                CustomTextField(
                    modifier = Modifier.fillMaxWidth(),
                    mutableValue =
                    email,
                    label = "Email",
                    focusedColor = primaryTextColor.value,
                    textColor = tertiaryTextColor.value,
                    conColor = textFieldColor.value
                )
                Spacer(modifier = Modifier.height(10.dp))

                CustomTextField(
                    modifier = Modifier
                        .fillMaxWidth()
                        .height(100.dp),
                    mutableValue =
                    address,
                    label = "Address",
                    focusedColor = primaryTextColor.value,
                    textColor = tertiaryTextColor.value,
                    conColor = textFieldColor.value
                )
                Spacer(modifier = Modifier.height(10.dp))

                CustomTextField(
                    modifier = Modifier.fillMaxWidth(),
                    mutableValue =
                    mobile,
                    label = "Mobile Number",
                    focusedColor = primaryTextColor.value,
                    textColor = tertiaryTextColor.value,
                    conColor = textFieldColor.value
                )

                Spacer(modifier = Modifier.size(10.dp))

                CustomTextField(
                    modifier = Modifier.fillMaxWidth(),
                    mutableValue =
                    bloodType,
                    label = "Blood Type",
                    focusedColor = primaryTextColor.value,
                    textColor = tertiaryTextColor.value,
                    conColor = textFieldColor.value
                )

                Spacer(modifier = Modifier.size(10.dp))

                CustomTextField(
                    modifier = Modifier.fillMaxWidth(),
                    mutableValue =
                    centre,
                    label = "Donation Centre",
                    focusedColor = primaryTextColor.value,
                    textColor = tertiaryTextColor.value,
                    conColor = textFieldColor.value
                )

                Spacer(modifier = Modifier.size(10.dp))
```

```
                    val mDatePickerDialog = DatePickerDialog(
                        mContext,
                        { _: DatePicker, mYear: Int, mMonth: Int,
mDayOfMonth: Int ->
                            mDate.value = "$mDayOfMonth/${mMonth+1}/$mYear"
                        }, mYear, mMonth, mDay
                    )

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

                        // Creating a button that on
                        // click displays/shows the DatePickerDialog
                        Button(onClick = {
                            mDatePickerDialog.show()
                        } ) {
                            Text(text = "Open Date Picker", color =
Color.White)
                        }

                        // Adding a space of 100dp height
                        Spacer(modifier = Modifier.size(10.dp))

                        // Displaying the mDate value in the Text
                        Text(text = "Selected Date: ${mDate.value}",
fontSize = 20.sp, textAlign = TextAlign.Center)
                    }


                    Spacer(modifier = Modifier.height(10.dp))

                    CustomButton(
                        buttonText = "Register", onClick = {
                            myDataList.add(DataModel(fullName.value.text,
"Name: "))
                            myDataList.add(DataModel(email.value.text,
"Email: "))
                            myDataList.add(DataModel(address.value.text,
"Address: "))
                            myDataList.add(DataModel(mobile.value.text,
"Mobile No: "))
                            myDataList.add(DataModel(bloodType.value.text,
"Blood Type: "))
                            myDataList.add(DataModel(centre.value.text,
"Donation Centre: "))
                            myDataList.add(DataModel(mDate.value, "Date:
"))

                            showList.value = false
                            showList.value = true

                        }, isLogo =
                        false
                    )



                    Spacer(modifier = Modifier.height(40.dp))
                    if (showList.value) {
```

```
                        CustomList(myList = myDataList)
                    }


                }
            }
        }
    }
}



@Composable
fun CustomList(myList: MutableList<DataModel>) {
    Spacer(modifier = Modifier.height(20.dp))
    Text(text = "Registered Successfully", fontSize = 20.sp, textAlign =
TextAlign.Center)
    Spacer(modifier = Modifier.height(20.dp))
    LazyColumn(
        modifier = Modifier
            .fillMaxWidth()
            .height(300.dp)
    ) {

        items(myList) { model ->
            CustomLazyItem(model = model)
        }

    }
}

@Composable
fun CustomLazyItem(model: DataModel) {
    Column(
        horizontalAlignment = Alignment.CenterHorizontally, modifier =
Modifier
            .fillMaxWidth()
            .height(50.dp)
            .border(
                BorderStroke(0.2.dp, Color.Black), shape =
                RoundedCornerShape(4.dp)
            ), verticalArrangement = Arrangement.Center
    ) {
        Text(text = model.detail + model.value, textAlign =
TextAlign.Center)

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

@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun CustomTextField(
    modifier: Modifier = Modifier,
    mutableValue: MutableState<TextFieldValue>, label: String,
    placeholder: String
    = label,
    focusedColor: Color, conColor: Color = Color(250, 250, 250),
    isHideVal: Boolean = false, textColor: Color
) {
    TextField(
```

```
        modifier = modifier.border(
            BorderStroke(0.2.dp, focusedColor), RoundedCornerShape
                (4.dp)
        ),
        value = mutableValue.value,
        onValueChange = { mutableValue.value = it },
        label = { Text(text = label) },
        placeholder = { Text(text = placeholder) },
        colors = TextFieldDefaults.outlinedTextFieldColors(
            focusedBorderColor = Color.Transparent,
            focusedLabelColor = focusedColor,
            placeholderColor = focusedColor,
            textColor = textColor,
            unfocusedBorderColor = Color.Transparent,
            unfocusedLabelColor = focusedColor,
            unfocusedLeadingIconColor = focusedColor,
            focusedLeadingIconColor = focusedColor,
            containerColor = conColor,

            ),

        visualTransformation = if (isHideVal) PasswordVisualTransformation(
            mask =
            '\u2022'
        ) else VisualTransformation.None,

        )
}

@Composable
fun CustomButton(
    buttonText: String,
    textColor: Color = Color.White,
    backgroundColor: Color = Color(0, 149, 246),
    onClick: () -> Unit = {},
    isLogo: Boolean = false
) {
    Button(
        onClick = onClick,
        shape = RoundedCornerShape(10.dp),
        colors = ButtonDefaults.buttonColors(backgroundColor),
        modifier = Modifier.fillMaxWidth()
    ) {
        if (isLogo) {
            Icon(
                Icons.Filled.Send, contentDescription =
                "facebook logo", modifier = Modifier.size(25.dp)
            )
        }
        Spacer(modifier = Modifier.width(10.dp))
        Text(
            buttonText,
            color = textColor,
            fontSize = 16.sp,
            fontWeight = FontWeight.Bold
        )
}


}
```