package com.example.firstproject  
  
import android.annotation.SuppressLint  
import android.os.Bundle  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.activity.enableEdgeToEdge  
import androidx.compose.foundation.Image  
import androidx.compose.foundation.background  
import androidx.compose.foundation.clickable  
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.WindowInsets  
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.offset  
import androidx.compose.foundation.layout.padding  
import androidx.compose.foundation.layout.size  
import androidx.compose.foundation.layout.width  
import androidx.compose.foundation.layout.wrapContentSize  
import androidx.compose.foundation.rememberScrollState  
import androidx.compose.foundation.shape.CornerSize  
import androidx.compose.foundation.shape.RoundedCornerShape  
import androidx.compose.foundation.text.KeyboardOptions  
import androidx.compose.foundation.verticalScroll  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.automirrored.filled.ArrowBack  
import androidx.compose.material.icons.filled.ArrowBack  
import androidx.compose.material.icons.filled.DateRange  
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.DatePicker  
import androidx.compose.material3.ExperimentalMaterial3Api  
import androidx.compose.material3.Icon  
import androidx.compose.material3.IconButton  
import androidx.compose.material3.MaterialTheme  
import androidx.compose.material3.OutlinedTextField  
import androidx.compose.material3.OutlinedTextFieldDefaults  
import androidx.compose.material3.Scaffold  
import androidx.compose.material3.Surface  
import androidx.compose.material3.Text  
import androidx.compose.material3.TopAppBar  
import androidx.compose.material3.TopAppBarColors  
import androidx.compose.material3.TopAppBarDefaults  
import androidx.compose.runtime.Composable  
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.TextStyle  
import androidx.compose.ui.text.font.FontWeight  
import androidx.compose.ui.text.style.TextAlign  
import androidx.compose.ui.tooling.preview.Preview  
import androidx.compose.ui.unit.dp  
import androidx.compose.ui.unit.sp  
import com.example.firstproject.ui.theme.FirstProjectTheme  
import androidx.compose.material3.rememberDatePickerState  
import androidx.compose.runtime.CompositionLocalProvider  
import androidx.compose.runtime.getValue  
import androidx.compose.runtime.mutableStateOf  
import androidx.compose.runtime.remember  
import androidx.compose.runtime.setValue  
import androidx.compose.ui.draw.clip  
import androidx.compose.ui.draw.shadow  
import androidx.compose.ui.graphics.RectangleShape  
import androidx.compose.ui.layout.AlignmentLine  
import androidx.compose.ui.layout.ContentScale  
import androidx.compose.ui.layout.onGloballyPositioned  
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.tooling.preview.Devices  
import androidx.compose.ui.window.Popup  
import androidx.navigation.NavController  
import androidx.navigation.compose.NavHost  
import androidx.navigation.compose.composable  
import androidx.navigation.compose.rememberNavController  
import java.text.SimpleDateFormat  
import java.util.Date  
import java.util.Locale  
  
class MainActivity : ComponentActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 enableEdgeToEdge()  
 setContent **{** FirstProjectTheme **{** navigation()  
  
  
 **}  
 }** }  
}  
  
@Composable  
fun navigation() {  
 val navController = rememberNavController()  
 NavHost(navController = navController, startDestination = "welcome") **{** composable(route = "welcome") **{** WelcomeScreen(navController)  
 **}** composable(route = "log in") **{** LogIn(navController)  
 **}** composable(route = "sign up") **{** NewAccount(navController)  
 **}** composable(route = "home screen") **{** HomeScreen(navController)  
 **}** composable(route = "set password") **{** SetPassword(navController)  
 **}  
 }**}  
  
@SuppressLint("UnusedMaterial3ScaffoldPaddingParameter")  
@OptIn(ExperimentalMaterial3Api::class)  
  
@Composable  
fun HomeScreen(navController: NavController) {  
 Scaffold(  
 topBar = **{** TopAppBar(title = **{** Row(  
 verticalAlignment = Alignment.CenterVertically,  
 horizontalArrangement = Arrangement.Center  
 ) **{** Image(painter = painterResource(id = R.drawable.vector),  
 contentDescription = null,  
  
 modifier = Modifier  
 .clickable **{** navController.popBackStack() **}** .size(20.dp)  
  
 //.align(alignment = Alignment.CenterVertically)  
 //popBackState()  
 )  
 Text(  
 text = "Home Screen",  
 style = TextStyle(  
 fontWeight = FontWeight.W600,  
 fontSize = 24.sp,  
 color = Color(0xFF2260FF)  
 ),  
 modifier = Modifier  
 .fillMaxWidth(),  
 //.align(alignment = Alignment.CenterVertically)  
 //,  
 textAlign = TextAlign.Center  
 )  
  
 **}  
  
 }** )  
 **}** ) **{** innerPadding **->** Text(  
 text = "WELCOME BACK !!",  
 textAlign = TextAlign.Center,  
 style = TextStyle(  
 fontWeight = FontWeight.W600,  
 fontSize = 24.sp,  
 color = Color(0xFF2260FF)  
 ),  
  
  
 )  
  
 **}**}  
  
  
@Composable  
fun WelcomeScreen(navController: NavController) {  
 Surface(  
  
 modifier = Modifier  
 .fillMaxSize()  
  
 ) **{** Column(  
 verticalArrangement = Arrangement.Center,  
 horizontalAlignment = Alignment.CenterHorizontally,  
 ) **{** Image(painter = painterResource(id = R.drawable.logo2), contentDescription = null)  
 Spacer(modifier = Modifier.height(100.dp))  
 Text(  
 text = "Lorem ipsum dolor sit amet, consectetur adipiscing elit,\n" +  
 " sed do eiusmod tempor incididunt ut labore et dolore \n" +  
 " magna aliqua. ",  
 style = TextStyle(fontSize = 12.sp, fontWeight = FontWeight.W300)  
 )  
 Spacer(modifier = Modifier.height(50.dp))  
 Button(  
 onClick = **{** navController.navigate(route = "log in") **}**,  
 modifier = Modifier  
 .width(207.dp)  
 .height(55.dp),  
 colors = ButtonDefaults.buttonColors(  
 containerColor = Color(0xFF2260FF)  
 ),  
  
  
 ) **{** Text(  
 text = "Log In ",  
 style = TextStyle(fontSize = 24.sp, fontWeight = FontWeight.W500)  
 )  
 **}** Spacer(modifier = Modifier.height(10.dp))  
 Button(  
 onClick = **{** navController.navigate(route = "sign up") **}**,  
 modifier = Modifier  
 .width(207.dp)  
 .height(55.dp),  
 colors = ButtonDefaults.buttonColors(  
 containerColor = Color(0xFFCAD6FF)  
 ),  
  
  
 ) **{** Text(  
 text = " Sign Up",  
 style = TextStyle(  
 fontSize = 24.sp,  
 fontWeight = FontWeight.W500,  
 color = Color(0xFF2260FF)  
 )  
 )  
 **}  
 }  
 }**}  
  
@Preview(device = Devices.PIXEL)  
@Composable  
private fun SignupPreview() {  
 SetPassword(navController = rememberNavController())  
}  
  
  
//@Preview(showBackground = true)  
@OptIn(ExperimentalMaterial3Api::class)  
@SuppressLint("UnusedMaterial3ScaffoldPaddingParameter", "RememberReturnType")  
@Composable  
fun LogIn(navController: NavController) {  
 var textFieldValue = remember **{** mutableStateOf("")  
 **}** var password = remember **{** mutableStateOf("")  
 **}** val passwordVisible = remember **{** mutableStateOf(false)  
 **}** Scaffold(  
 topBar = **{** TopAppBar(title = **{** Row **{** Image(painter = painterResource(id = R.drawable.vector),  
 contentDescription = null,  
  
 modifier = Modifier  
 .clickable **{** navController.popBackStack() **}** .size(20.dp)  
 .align(alignment = Alignment.CenterVertically)  
 //popBackState()  
 )  
 Text(  
 text = "Log In",  
 style = TextStyle(  
 fontWeight = FontWeight.W600,  
 fontSize = 24.sp,  
 color = Color(0xFF2260FF)  
 ),  
 modifier = Modifier  
 .fillMaxWidth()  
 .align(alignment = Alignment.CenterVertically),  
 textAlign = TextAlign.Center  
 )  
  
 **}  
  
 }** )  
 **}** ) **{** innerPadding **->** Column(modifier = Modifier.padding(innerPadding)) **{** Text(  
 text = "Welcome",  
 modifier = Modifier.padding(top = 10.dp, start = 30.dp),  
 style = TextStyle(  
 fontSize = 24.sp,  
 lineHeight = 22.08.sp,  
 fontWeight = FontWeight.W600,  
 color = Color(0xFF2260FF)  
 )  
 )  
  
 Spacer(modifier = Modifier.padding(5.dp))  
  
  
 Text(  
 text = "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do \nneiusmod tempor incididunt ut labore et dolore magna aliqua. ",  
 modifier = Modifier.padding(start = 30.dp, end = 6.dp),  
 style = TextStyle(  
 fontSize = 1.sp,  
 fontWeight = FontWeight.W300,  
 color = Color(0xFF070707)  
 )  
  
 )  
  
 Spacer(modifier = Modifier.padding(25.dp))  
 Text(  
 text = "Email or Mobile Number",  
 modifier = Modifier.padding(start = 30.dp),  
 style = TextStyle(  
 fontSize = 20.sp,  
 lineHeight = 18.08.sp,  
 fontWeight = FontWeight.W500,  
 color = Color(0xFF000000)  
 )  
 )  
 OutlinedTextField(  
 value = textFieldValue.value,  
 onValueChange = **{** textFieldValue.value = it **}**,  
 placeholder = **{** Text(  
 text = "example@example.com",  
 style = TextStyle(  
 fontSize = 20.sp,  
 fontWeight = FontWeight.W400,  
 lineHeight = 18.4.sp,  
 color = Color(0xFF809CFF)  
 )  
 )  
 **}**,  
 modifier = Modifier  
 .fillMaxWidth()  
 .padding(start = 30.dp, end = 30.dp, top = 10.dp),  
 colors = OutlinedTextFieldDefaults.colors(  
 focusedBorderColor = Color(0xFFECF1FF),  
 unfocusedBorderColor = Color(0xFFECF1FF),  
 focusedContainerColor = Color(0xFFECF1FF),  
 unfocusedContainerColor = Color(0xFFECF1FF)  
  
 ),  
  
 maxLines = 1,  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
  
 ),  
 )  
  
 Spacer(modifier = Modifier.padding(15.dp))  
 Text(  
 text = "Password",  
 modifier = Modifier.padding(start = 30.dp),  
 style = TextStyle(  
 fontSize = 20.sp,  
 lineHeight = 18.08.sp,  
 fontWeight = FontWeight.W500,  
 color = Color(0xFF000000)  
 )  
 )  
 OutlinedTextField(  
 value = password.value,  
 onValueChange = **{** password.value = it **}**,  
 placeholder = **{** Text(  
 text = "Password ",  
 style = TextStyle(  
 fontSize = 20.sp,  
 fontWeight = FontWeight.W400,  
 lineHeight = 18.4.sp,  
 color = Color(0xFF809CFF)  
 )  
 )  
 **}**,  
 visualTransformation = if (passwordVisible.value) VisualTransformation.None  
 else PasswordVisualTransformation(),  
 trailingIcon = **{** var image = if (passwordVisible.value)  
 Icons.Filled.Visibility  
 else  
 Icons.Filled.VisibilityOff  
 val description = if (passwordVisible.value) "show Password"  
 else "Hide Pssword"  
 IconButton(onClick = **{** passwordVisible.value = !passwordVisible.value **}**) **{** Icon(imageVector = image, contentDescription = null)  
 **}  
 }**,  
 modifier = Modifier  
 .fillMaxWidth()  
 .padding(start = 30.dp, end = 30.dp, top = 10.dp),  
 colors = OutlinedTextFieldDefaults.colors(  
 focusedBorderColor = Color(0xFFECF1FF),  
 unfocusedBorderColor = Color(0xFFECF1FF),  
 focusedContainerColor = Color(0xFFECF1FF),  
 unfocusedContainerColor = Color(0xFFECF1FF)  
  
 ),  
 maxLines = 1,  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 ),  
 )  
 Text(text = "Forget Password", style = TextStyle(  
 fontWeight = FontWeight.W500,  
 fontSize = 12.sp,  
 color = Color(0xFF2260FF)  
 ),  
 modifier = Modifier  
 .padding(start = 235.dp, top = 5.dp)  
 .clickable **{** navController.navigate(route = "set password") **}**)  
  
 Spacer(modifier = Modifier.height(50.dp))  
  
 Button(  
 onClick = **{** navController.navigate(route = "home screen") **}**,  
 modifier = Modifier  
 .width(207.dp)  
 .height(55.dp)  
 .align(alignment = Alignment.CenterHorizontally),  
 colors = ButtonDefaults.buttonColors(  
 containerColor = Color(0xFF2260FF)  
 ),  
 ) **{** Text(  
 text = "Log In ", style = TextStyle(  
 fontSize = 24.sp, fontWeight = FontWeight.W500  
 )  
 )  
 **}** Spacer(modifier = Modifier.height(20.dp))  
 Text(  
 text = "or sign up with",  
 style = TextStyle(  
 fontSize = 12.sp,  
 fontWeight = FontWeight.W300,  
 color = Color(0xFF070707),  
 textAlign = TextAlign.Center,  
 ),  
 modifier = Modifier.align(alignment = Alignment.CenterHorizontally)  
 )  
 Spacer(modifier = Modifier.height(20.dp))  
 Image(  
 painter = painterResource(id = R.drawable.group\_99), contentDescription = null,  
 modifier = Modifier.align(alignment = Alignment.CenterHorizontally)  
 )  
  
 Spacer(modifier = Modifier.height(40.dp))  
 Row(horizontalArrangement = Arrangement.Center, modifier = Modifier.fillMaxWidth()) **{** Text(  
 text = "Don't have an account?",  
 style = TextStyle(  
 fontSize = 12.sp, fontWeight = FontWeight.W300  
 ),  
 )  
 Text(text = "Sign Up", style = TextStyle(  
 fontSize = 12.sp, fontWeight = FontWeight.W500, color = Color(0xFF2260FF)  
 ),  
 modifier = Modifier.clickable **{** navController.navigate(route = "sign up") **}**)  
 **}  
 }  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
//@Preview(device = Devices.PIXEL)  
  
@Composable  
fun NewAccount(navController: NavController) {  
 Scaffold(  
 topBar = **{** TopAppBar(title = **{** Row **{** Image(painter = painterResource(id = R.drawable.vector),  
 contentDescription = null,  
  
 modifier = Modifier  
 .clickable **{** navController.popBackStack() **}** .size(20.dp)  
 .align(alignment = Alignment.CenterVertically)  
 //popBackState()  
 )  
 Text(  
 text = "New Account",  
 style = TextStyle(  
 fontWeight = FontWeight.W600,  
 fontSize = 24.sp,  
 color = Color(0xFF2260FF)  
 ),  
 modifier = Modifier  
 .fillMaxWidth()  
 .align(alignment = Alignment.CenterVertically),  
 textAlign = TextAlign.Center  
 )  
  
 **}  
  
 }** )  
 **}** ) **{** innerPadding **->** Column(  
 modifier = Modifier  
 .padding(innerPadding)  
 .verticalScroll(rememberScrollState())  
 ) **{** Spacer(modifier = Modifier.padding(5.dp))  
 var name = remember **{** mutableStateOf("")  
 **}** val aPassword = remember **{** mutableStateOf("")  
 **}** val Email = remember **{** mutableStateOf("")  
 **}** val mNumber = remember **{** mutableStateOf("")  
 **}** val dob = remember **{** mutableStateOf("")  
 **}** val PasswordVisible = remember **{** mutableStateOf(false)  
 **}** Text(  
 text = "Full Name ",  
 modifier = Modifier.padding(start = 30.dp),  
 style = TextStyle(  
 fontSize = 20.sp,  
 lineHeight = 18.08.sp,  
 fontWeight = FontWeight.W500,  
 color = Color(0xFF000000)  
 )  
 )  
  
  
 OutlinedTextField(  
 value = name.value,  
 onValueChange = **{** name.value = it **}**,  
 placeholder = **{** Text(  
 text = "example@example.com",  
 style = TextStyle(  
 fontSize = 20.sp,  
 fontWeight = FontWeight.W400,  
 lineHeight = 18.4.sp,  
 color = Color(0xFF809CFF)  
 ),  
  
 )  
 **}**,  
 modifier = Modifier  
 .fillMaxWidth()  
 .padding(start = 30.dp, end = 30.dp, top = 10.dp),  
 colors = OutlinedTextFieldDefaults.colors(  
 focusedBorderColor = Color(0xFFECF1FF),  
 unfocusedBorderColor = Color(0xFFECF1FF),  
 focusedContainerColor = Color(0xFFECF1FF),  
 unfocusedContainerColor = Color(0xFFECF1FF)  
  
 ),  
  
 maxLines = 1,  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
  
 ),  
 )  
  
 Spacer(modifier = Modifier.padding(7.dp))  
 Text(  
 text = "Password",  
 modifier = Modifier.padding(start = 30.dp),  
 style = TextStyle(  
 fontSize = 20.sp,  
 lineHeight = 18.08.sp,  
 fontWeight = FontWeight.W500,  
 color = Color(0xFF000000)  
 )  
 )  
  
 OutlinedTextField(  
 value = aPassword.value,  
 onValueChange = **{** aPassword.value = it **}**,  
 placeholder = **{** Text(  
 text = "Password ",  
 style = TextStyle(  
 fontSize = 20.sp,  
 fontWeight = FontWeight.W400,  
 lineHeight = 18.4.sp,  
 color = Color(0xFF809CFF)  
 )  
 )  
 **}**,  
 visualTransformation = if (PasswordVisible.value) VisualTransformation.None  
 else PasswordVisualTransformation(),  
 trailingIcon = **{** var image = if (PasswordVisible.value)  
 Icons.Filled.Visibility  
 else  
 Icons.Filled.VisibilityOff  
 val description = if (PasswordVisible.value) "show Password"  
 else "Hide Pssword"  
 IconButton(onClick = **{** PasswordVisible.value = !PasswordVisible.value **}**) **{** Icon(imageVector = image, contentDescription = null)  
 **}  
 }**,  
 modifier = Modifier  
 .fillMaxWidth()  
 .padding(start = 30.dp, end = 30.dp, top = 10.dp),  
 colors = OutlinedTextFieldDefaults.colors(  
 focusedBorderColor = Color(0xFFECF1FF),  
 unfocusedBorderColor = Color(0xFFECF1FF),  
 focusedContainerColor = Color(0xFFECF1FF),  
 unfocusedContainerColor = Color(0xFFECF1FF)  
  
 ),  
 maxLines = 1,  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 ),  
  
 )  
 Spacer(modifier = Modifier.padding(top = 7.dp, bottom = 7.dp))  
 Text(  
 text = "Email",  
 modifier = Modifier.padding(start = 30.dp),  
 style = TextStyle(  
 fontSize = 20.sp,  
 lineHeight = 18.08.sp,  
 fontWeight = FontWeight.W500,  
 color = Color(0xFF000000)  
 )  
 )  
  
 Spacer(modifier = Modifier.padding(start = 7.dp, bottom = 7.dp))  
  
 OutlinedTextField(  
 value = Email.value,  
 onValueChange = **{** Email.value = it **}**,  
 placeholder = **{** Text(  
 text = "example@example.com",  
 style = TextStyle(  
 fontSize = 20.sp,  
 fontWeight = FontWeight.W400,  
 lineHeight = 18.4.sp,  
 color = Color(0xFF809CFF)  
 ),  
  
 )  
 **}**,  
 modifier = Modifier  
 .fillMaxWidth()  
 .padding(start = 30.dp, end = 30.dp, top = 10.dp),  
 colors = OutlinedTextFieldDefaults.colors(  
 focusedBorderColor = Color(0xFFECF1FF),  
 unfocusedBorderColor = Color(0xFFECF1FF),  
 focusedContainerColor = Color(0xFFECF1FF),  
 unfocusedContainerColor = Color(0xFFECF1FF)  
  
 ),  
  
 maxLines = 1,  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
  
 ),  
 )  
  
  
 Spacer(modifier = Modifier.padding(7.dp))  
  
 Text(  
 text = "Phone Number",  
 modifier = Modifier.padding(start = 30.dp),  
 style = TextStyle(  
 fontSize = 20.sp,  
 lineHeight = 18.08.sp,  
 fontWeight = FontWeight.W500,  
 color = Color(0xFF000000)  
 )  
 )  
  
  
 OutlinedTextField(  
 value = mNumber.value,  
 onValueChange = **{** mNumber.value = it **}**,  
 placeholder = **{** Text(  
 text = "example@example.com",  
 style = TextStyle(  
 fontSize = 20.sp,  
 fontWeight = FontWeight.W400,  
 lineHeight = 18.4.sp,  
 color = Color(0xFF809CFF)  
 ),  
  
 )  
 **}**,  
 modifier = Modifier  
 .fillMaxWidth()  
 .padding(start = 30.dp, end = 30.dp, top = 10.dp),  
 colors = OutlinedTextFieldDefaults.colors(  
 focusedBorderColor = Color(0xFFECF1FF),  
 unfocusedBorderColor = Color(0xFFECF1FF),  
 focusedContainerColor = Color(0xFFECF1FF),  
 unfocusedContainerColor = Color(0xFFECF1FF)  
  
 ),  
  
 maxLines = 1,  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Number  
  
 ),  
 )  
  
 Spacer(modifier = Modifier.padding(7.dp))  
  
 Text(  
 text = "Date Of Birth ",  
 modifier = Modifier.padding(start = 30.dp),  
 style = TextStyle(  
 fontSize = 20.sp,  
 lineHeight = 18.08.sp,  
 fontWeight = FontWeight.W500,  
 color = Color(0xFF000000)  
 )  
 )  
  
  
 OutlinedTextField(  
 value = dob.value,  
 onValueChange = **{** dob.value = it **}**,  
 placeholder = **{** Text(  
 text = "DD / MM / YYY",  
 style = TextStyle(  
 fontSize = 20.sp,  
 fontWeight = FontWeight.W400,  
 lineHeight = 18.4.sp,  
 color = Color(0xFF809CFF)  
 ),  
  
 )  
 **}**,  
 modifier = Modifier  
 .fillMaxWidth()  
 .padding(start = 30.dp, end = 30.dp, top = 10.dp),  
 colors = OutlinedTextFieldDefaults.colors(  
 focusedBorderColor = Color(0xFFECF1FF),  
 unfocusedBorderColor = Color(0xFFECF1FF),  
 focusedContainerColor = Color(0xFFECF1FF),  
 unfocusedContainerColor = Color(0xFFECF1FF)  
  
 ),  
  
 maxLines = 1,  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Password  
  
 ),  
 )  
 Spacer(modifier = Modifier.padding(7.dp))  
  
 Text(  
 text = "By continuing, you agree to", style = TextStyle(  
 fontSize = 12.sp, fontWeight = FontWeight.W300, lineHeight = 11.04.sp,  
 ),  
 modifier = Modifier.align(alignment = Alignment.CenterHorizontally)  
 )  
 Row(horizontalArrangement = Arrangement.Center, modifier = Modifier.fillMaxWidth()) **{** Text(  
 text = "Term of Use ", style = TextStyle(  
 fontSize = 12.sp, fontWeight = FontWeight.W500, color = Color(0xFF2260FF)  
 ),  
 modifier = Modifier.padding(start = 2.dp)  
 )  
  
 Text(  
 text = "and ",  
 style = TextStyle(  
 fontSize = 12.sp, fontWeight = FontWeight.W300  
 ),  
 modifier = Modifier.padding(start = 2.dp, end = 2.dp)  
 )  
  
 Text(  
 text = "Privacy Policy.", style = TextStyle(  
 fontSize = 12.sp, fontWeight = FontWeight.W500, color = Color(0xFF2260FF)  
 ),  
 modifier = Modifier.padding(end = 0.dp)  
 )  
 **}** Spacer(modifier = Modifier.padding(top = 7.dp, bottom = 7.dp))  
 Button(  
 onClick = **{** navController.navigate(route = "home screen") **}**,  
 modifier = Modifier  
 .width(207.dp)  
 .height(55.dp)  
 .align(alignment = Alignment.CenterHorizontally),  
 colors = ButtonDefaults.buttonColors(  
 containerColor = Color(0xFF2260FF)  
 ),  
 ) **{** Text(  
 text = "Sign Up ", style = TextStyle(  
 fontSize = 24.sp, fontWeight = FontWeight.W500  
 )  
 )  
 **}** Spacer(modifier = Modifier.height(20.dp))  
 Text(  
 text = "or sign up with",  
 style = TextStyle(  
 fontSize = 12.sp,  
 fontWeight = FontWeight.W300,  
 color = Color(0xFF070707),  
 textAlign = TextAlign.Center,  
 ),  
 modifier = Modifier.align(alignment = Alignment.CenterHorizontally)  
  
 )  
 Spacer(modifier = Modifier.height(20.dp))  
 Image(  
 painter = painterResource(id = R.drawable.group\_99), contentDescription = null,  
 modifier = Modifier.align(alignment = Alignment.CenterHorizontally)  
 )  
  
 Spacer(modifier = Modifier.height(25.dp))  
 Row(  
 horizontalArrangement = Arrangement.Center, modifier = Modifier.fillMaxWidth()  
 ) **{** Text(  
 text = "Already have an account?",  
 style = TextStyle(  
 fontSize = 12.sp, fontWeight = FontWeight.W300  
 ),  
 )  
 Text(text = "Log in", style = TextStyle(  
 fontSize = 12.sp, fontWeight = FontWeight.W500, color = Color(0xFF2260FF)  
 ),  
 modifier = Modifier.clickable **{** navController.navigate(route = "log in") **}**)  
  
 **}  
 }  
 }**}  
  
@SuppressLint("UnusedMaterial3ScaffoldPaddingParameter")  
@OptIn(ExperimentalMaterial3Api::class)  
  
@Composable  
fun SetPassword(navController: NavController) {  
 var findHight by remember **{** mutableStateOf(0.dp)  
 **}** Scaffold(  
 //containerColor = Color.Magenta,  
 topBar = **{** TopAppBar(  
 // windowInsets = WindowInsets(0.dp, 0.dp, 24.dp, 0.dp),  
 //colors = TopAppBarDefaults.topAppBarColors(containerColor = Color.Cyan),  
 navigationIcon = **{** Card(  
  
 modifier = Modifier  
 .clickable **{** navController.popBackStack() **}** .padding(start = 5.dp)  
 //.clip(shape = RoundedCornerShape(corner = CornerSize(122.dp)))  
 // .size(height = 40.dp, width = 40.dp)  
 ,  
 shape = RectangleShape,  
  
// colors = CardDefaults.cardColors(  
// containerColor = MaterialTheme.colorScheme.background  
// )  
 ) **{** Image(painter = painterResource(id = R.drawable.vector),  
 contentDescription = null,  
 // contentScale = ContentScale.Fit,  
  
 modifier = Modifier  
 .clickable **{** navController.popBackStack() **}** .padding(12.dp)  
 .size(20.dp)  
 //.align(alignment = Alignment.CenterVertically)  
 //popBackState()  
 )  
 **}  
 }**,  
  
 actions = **{** Card(  
  
 modifier = Modifier  
 .clickable **{** navController.popBackStack() **}** .padding(end = 5.dp)  
 .onGloballyPositioned **{** findHight =with()**{}  
 }** //.size(width = 20.dp , height = )  
  
 //.clip(shape = RoundedCornerShape(corner = CornerSize(122.dp)))  
 .size(width = 40.dp, height = 0.dp)  
 ,  
 // shape = RectangleShape,  
  
  
// colors = CardDefaults.cardColors(  
// containerColor = MaterialTheme.colorScheme.background  
// )  
 )  
 **{** Image(painter = painterResource(id = R.drawable.vector),  
 contentDescription = null,  
 // contentScale = ContentScale.Fit,  
  
 modifier = Modifier  
 .clickable **{** navController.popBackStack() **}** .padding(12.dp)  
 .size(20.dp)  
 //.align(alignment = Alignment.CenterVertically)  
 //popBackState()  
 )  
  
 **}  
 }**,  
 title = **{**// Row(  
// horizontalArrangement = Arrangement.SpaceBetween,  
// modifier = Modifier.fillMaxSize()  
// .background(Color.Red)  
// ) {  
// Image(painter = painterResource(id = R.drawable.vector),  
// contentDescription = null,  
//  
// modifier = Modifier  
// .clickable { navController.popBackStack() }  
// .size(20.dp)  
// //.align(alignment = Alignment.CenterVertically)  
// //popBackState()  
// )  
 Text(  
 text = "Set Password",  
 style = TextStyle(  
 fontWeight = FontWeight.W600,  
 fontSize = 24.sp,  
 color = Color(0xFF2260FF)  
 ),  
 modifier = Modifier  
 .fillMaxWidth()  
  
 ,  
// .align(alignment = Alignment.CenterVertically),  
 textAlign = TextAlign.Center  
 )  
  
  
  
// Spacer(  
// modifier = Modifier  
// .size(20.dp)  
// )  
//  
// }  
  
 **}** )  
 **}**,  
 modifier = Modifier.background(Color.Green)  
 ) **{** innerPading **->** Column(modifier = Modifier.padding((innerPading))) **{** val PASSWORD = remember **{** mutableStateOf("")  
 **}** val VISIBLEPASSWORD = remember **{** mutableStateOf(false)  
 **}** Text(  
 text = "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do \nneiusmod tempor incididunt ut labore et dolore magna aliqua. ",  
 modifier = Modifier.padding(start = 30.dp , ),  
 style = TextStyle(  
 fontSize = 12.sp,  
 fontWeight = FontWeight.W300,  
 color = Color(0xFF070707)  
 )  
 )  
 Spacer(modifier = Modifier.padding(25.dp))  
 Text(  
 text = "Password",  
 modifier = Modifier.padding(start = 30.dp),  
 style = TextStyle(  
 fontSize = 20.sp,  
 lineHeight = 18.08.sp,  
 fontWeight = FontWeight.W500,  
 color = Color(0xFF000000)  
 )  
 )  
  
 OutlinedTextField(  
 value = PASSWORD.value,  
 onValueChange = **{** PASSWORD.value = it **}**,  
 placeholder = **{** Text(  
 text = "Password ",  
 style = TextStyle(  
 fontSize = 20.sp,  
 fontWeight = FontWeight.W400,  
 lineHeight = 18.4.sp,  
 color = Color(0xFF809CFF)  
 )  
 )  
 **}**,  
 visualTransformation = if (VISIBLEPASSWORD.value) VisualTransformation.None  
 else PasswordVisualTransformation(),  
 trailingIcon = **{** var image = if (VISIBLEPASSWORD.value)  
 Icons.Filled.Visibility  
 else  
 Icons.Filled.VisibilityOff  
 val description = if (VISIBLEPASSWORD.value) "show Password"  
 else "Hide Pssword"  
 IconButton(onClick = **{** VISIBLEPASSWORD.value = !VISIBLEPASSWORD.value **}**) **{** Icon(imageVector = image, contentDescription = null)  
 **}  
 }**,  
 modifier = Modifier  
 .fillMaxWidth()  
 .padding(start = 30.dp, end = 30.dp, top = 10.dp),  
 colors = OutlinedTextFieldDefaults.colors(  
 focusedBorderColor = Color(0xFFECF1FF),  
 unfocusedBorderColor = Color(0xFFECF1FF),  
 focusedContainerColor = Color(0xFFECF1FF),  
 unfocusedContainerColor = Color(0xFFECF1FF)  
  
 ),  
 maxLines = 1,  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 ),  
  
 )  
 Spacer(modifier = Modifier.padding(top = 25.dp, bottom = 3.dp))  
 Text(  
 text = "Conform Password",  
 modifier = Modifier.padding(start = 30.dp),  
 style = TextStyle(  
 fontSize = 20.sp,  
 lineHeight = 18.08.sp,  
 fontWeight = FontWeight.W500,  
 color = Color(0xFF000000)  
 )  
 )  
  
  
 Spacer(modifier = Modifier.padding(top = 5.dp))  
  
  
 OutlinedTextField(  
 value = PASSWORD.value,  
 onValueChange = **{** PASSWORD.value = it **}**,  
 placeholder = **{** Text(  
 text = "Password ",  
 style = TextStyle(  
 fontSize = 20.sp,  
 fontWeight = FontWeight.W400,  
 lineHeight = 18.4.sp,  
 color = Color(0xFF809CFF)  
 )  
 )  
 **}**,  
 visualTransformation = if (VISIBLEPASSWORD.value) VisualTransformation.None  
 else PasswordVisualTransformation(),  
 trailingIcon = **{** var image = if (VISIBLEPASSWORD.value)  
 Icons.Filled.Visibility  
 else  
 Icons.Filled.VisibilityOff  
 val description = if (VISIBLEPASSWORD.value) "show Password"  
 else "Hide Pssword"  
 IconButton(onClick = **{** VISIBLEPASSWORD.value = !VISIBLEPASSWORD.value **}**) **{** Icon(imageVector = image, contentDescription = null)  
 **}  
 }**,  
 modifier = Modifier  
 .fillMaxWidth()  
 .padding(start = 30.dp, end = 30.dp, top = 10.dp),  
 colors = OutlinedTextFieldDefaults.colors(  
 focusedBorderColor = Color(0xFFECF1FF),  
 unfocusedBorderColor = Color(0xFFECF1FF),  
 focusedContainerColor = Color(0xFFECF1FF),  
 unfocusedContainerColor = Color(0xFFECF1FF)  
  
 ),  
 maxLines = 1,  
 singleLine = true,  
 keyboardOptions = KeyboardOptions(  
 keyboardType = KeyboardType.Text,  
 ),  
  
 )  
 Spacer(modifier = Modifier.padding(top = 45.dp))  
 Button(  
 onClick = **{** navController.navigate(route = "log in") **}**,  
 modifier = Modifier  
 .width(307.dp)  
 .height(55.dp)  
 .align(alignment = Alignment.CenterHorizontally),  
 colors = ButtonDefaults.buttonColors(  
 containerColor = Color(0xFF2260FF)  
 ),  
 ) **{** Text(text = "Create New Password ", style = TextStyle(  
 fontSize = 24.sp, fontWeight = FontWeight.W500  
 ),  
 modifier = Modifier.clickable **{** navController.navigate(route = "log in") **}**)  
 **}  
 }  
  
  
 }**}  
  
@OptIn(ExperimentalMaterial3Api::class)  
//@Preview  
@Composable  
fun DatePickerDocked() {  
 var showDatePicker by remember **{** mutableStateOf(false) **}** val datePickerState = rememberDatePickerState()  
 val selectedDate = datePickerState.selectedDateMillis?.let **{** convertMillisToDate(it)  
 **}** ?: ""  
  
 Box(  
 modifier = Modifier.fillMaxWidth()  
 ) **{** OutlinedTextField(  
 value = selectedDate,  
 onValueChange = **{ }**,  
 label = **{** Text("DOB") **}**,  
 readOnly = true,  
 trailingIcon = **{** IconButton(onClick = **{** showDatePicker = !showDatePicker **}**) **{** Icon(  
 imageVector = Icons.Default.DateRange,  
 contentDescription = "Select date"  
 )  
 **}  
 }**,  
 modifier = Modifier  
 .fillMaxWidth()  
 .height(64.dp)  
 )  
  
 if (showDatePicker) {  
 Popup(  
 onDismissRequest = **{** showDatePicker = false **}**,  
 alignment = Alignment.TopStart  
 ) **{** Box(  
 modifier = Modifier  
 .fillMaxWidth()  
 .offset(y = 64.dp)  
 .shadow(elevation = 4.dp)  
 .background(MaterialTheme.colorScheme.surface)  
 .padding(16.dp)  
 ) **{** DatePicker(  
 state = datePickerState,  
 showModeToggle = false  
 )  
 **}  
 }** }  
 **}**}  
  
fun convertMillisToDate(millis: Long): String {  
 val formatter = SimpleDateFormat("MM/dd/yyyy", Locale.getDefault())  
 return formatter.format(Date(millis))  
}