MONEY MATTERS: A PERSONAL FINANCE MANAGEMENT APP

N.NAVIKA,
710422243033.

"Money Matters: A Personal Finance Management App" is designed to help individuals take control of their financial lives through an intuitive and user-friendly platform. The app enables users to track their income, expenses, and savings, set and manage budgets, and plan for future financial goals. By providing tools for automatic transaction syncing, bill reminders, debt management, and investment tracking, the app offers a comprehensive approach to personal finance. Through insightful reports and actionable recommendations, it empowers users to make informed decisions, improve financial habits, and work toward achieving their long-term financial objectives. With a focus on security, privacy, and accessibility, "Money Matters" aims to simplify financial management, reduce stress, and enhance users' overall financial well-being. Whether you're a student, professional, or retiree, this app provides the necessary tools to achieve greater financial freedom and stability.

```
package com.example.expensestrackerimport
android.annotation.SuppressLintimport android.content.Contextimport
android.content.Intentimport android.os.Bundleimport
android.widget.Toastimport androidx.activity.ComponentActivityimport
androidx.activity.compose.setContentimport
androidx.compose.foundation.layout.*import
androidx.compose.material.*import androidx.compose.runtime.*import
androidx.compose.ui.Alignmentimport androidx.compose.ui.Modifierimport
androidx.compose.ui.graphics.Colorimport
androidx.compose.ui.platform.LocalContextimport
androidx.compose.ui.text.font.FontWeightimport
androidx.compose.ui.text.style.TextAlignimport
androidx.compose.ui.unit.dpimport androidx.compose.ui.unit.sp
```

```kotlin
class AddExpensesActivity : ComponentActivity()
{
   private lateinit var itemsDatabaseHelper: ItemsDatabaseHelper    private lateinit var expenseDatabaseHelper:
ExpenseDatabaseHelper    @SuppressLint("UnusedMaterialScaffoldPaddingParameter")
   override fun onCreate(savedInstanceState: Bundle?)
{
      super.onCreate(savedInstanceState)
   itemsDatabaseHelper = ItemsDatabaseHelper(this)        expenseDatabaseHelper =
ExpenseDatabaseHelper(this)
      setContent {
        Scaffold(
 // in scaffold we are specifying top bar.
           bottomBar = {
  // inside top bar we are specifying
      // background color.
           BottomAppBar(backgroundColor = Color(0xFFadbef4),
            modifier = Modifier.height(80.dp),
      // along with that we are specifying
      // title for our top bar.
           content = {
```

```kotlin
Spacer(modifier = Modifier.width(15.dp))
        Button(
    onClick =
{
startActivity(Intent(applicationContext,AddExpensesActivity::class.java))},
             colors = ButtonDefaults.buttonColors(backgroundColor = Color.White),
         modifier = Modifier.size(height = 55.dp, width = 110.dp)


     )
         {
        Text(
          text = "Add Expenses", color = Color.Black, fontSize = 14.sp,                    textAlign = TextAlign.Center
    )
         }
        Spacer(modifier = Modifier.width(15.dp))
         Button(
         onClick = {
              startActivity(
                 Intent(
               applicationContext,
                SetLimitActivity::class.java
                   )
```

```kotlin
        },
        colors = ButtonDefaults.buttonColors(backgroundColor = Color.White),
                modifier = Modifier.size(height = 55.dp, width = 110.dp)
                    )
            {
        Text(
                text = "Set Limit", color = Color.Black, fontSize = 14.sp,
            textAlign = TextAlign.Center
                    )
    }
            Spacer(modifier = Modifier.width(15.dp))
                    Button(
            onClick = {
          startActivity(
                    Intent
                            applicationContext,
                        ViewRecordsActivity::class.java
                    )
            )
            },
```

```kotlin
            colors = ButtonDefaults.buttonColors(backgroundColor = Color.White),                    modifier = Modifier.size(height = 55.dp, width = 110.dp)
                )
        {

                    Text(
                    text = "View Records", color = Color.Black, fontSize = 14.sp,
                        textAlign = TextAlign.Center                    )                    }
            }
                )
            }
        )
    {

            AddExpenses(this, itemsDatabaseHelper, expenseDatabaseHelper)

        }
        }
    }}
@SuppressLint("Range")@Composablefun AddExpenses(context: Context, itemsDatabaseHelper: ItemsDatabaseHelper, expenseDatabaseHelper:
ExpenseDatabaseHelper)

{
    Column(
        modifier = Modifier
            .padding(top = 100.dp, start = 30.dp)
.fillMaxHeight()
        .fillMaxWidth(),
            horizontalAlignment = Alignment.Start    ) {
```

```kotlin
val mContext = LocalContext.current
    var items by remember { mutableStateOf("") }
   var quantity by remember { mutableStateOf("") }
   var cost by remember { mutableStateOf("") }
   var error by remember { mutableStateOf("") }
 Text(text = "Item Name", fontWeight = FontWeight.Bold, fontSize = 20.sp)      Spacer(modifier =
Modifier.height(10.dp))
    TextField(value = items, onValueChange = { items = it },
        label = { Text(text = "Item Name") })
   Spacer(modifier = Modifier.height(20.dp))
     Text(text = "Quantity of item", fontWeight = FontWeight.Bold, fontSize = 20.sp)
     Spacer(modifier = Modifier.height(10.dp))
   TextField(value = quantity, onValueChange = { quantity = it },
        label = { Text(text = "Quantity") })
    Spacer(modifier = Modifier.height(20.dp))
      Text(text = "Cost of the item", fontWeight = FontWeight.Bold, fontSize = 20.sp)
     Spacer(modifier = Modifier.height(10.dp))
    TextField(value = cost, onValueChange = { cost = it },
   label = { Text(text = "Cost") })
     Spacer(modifier = Modifier.height(20.dp))
```
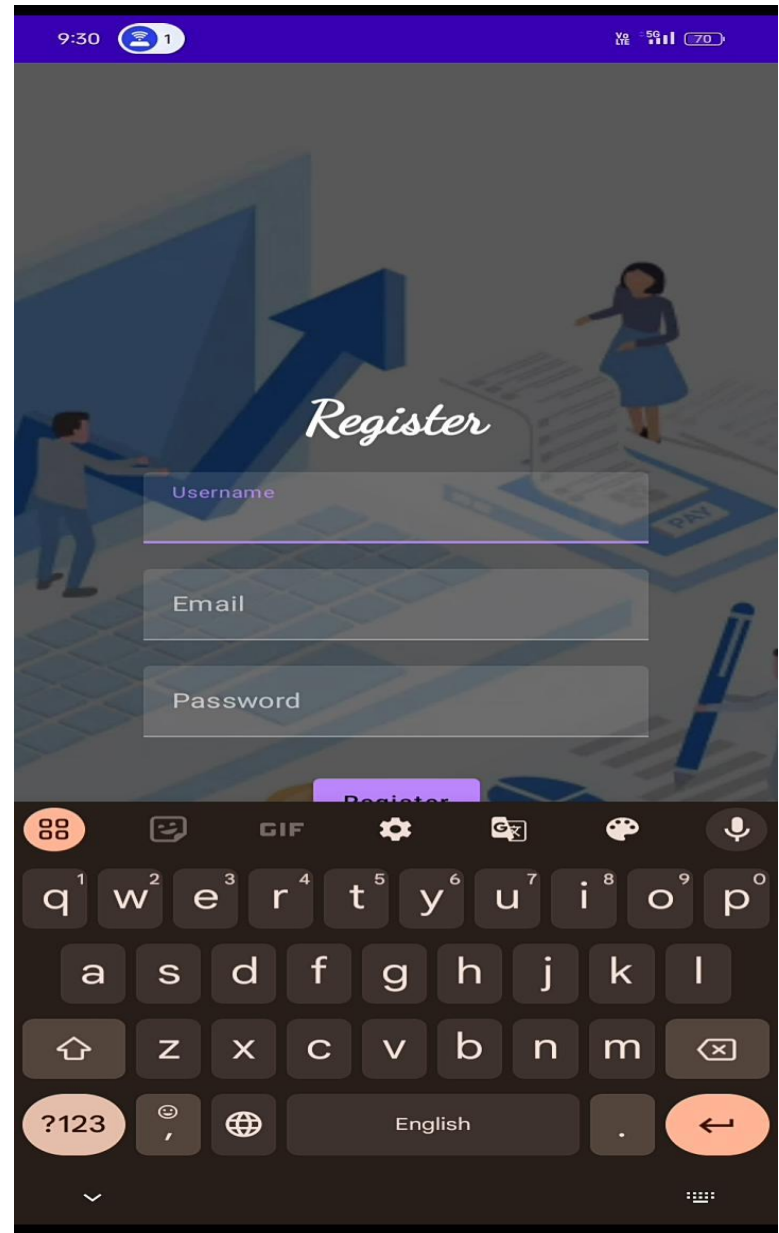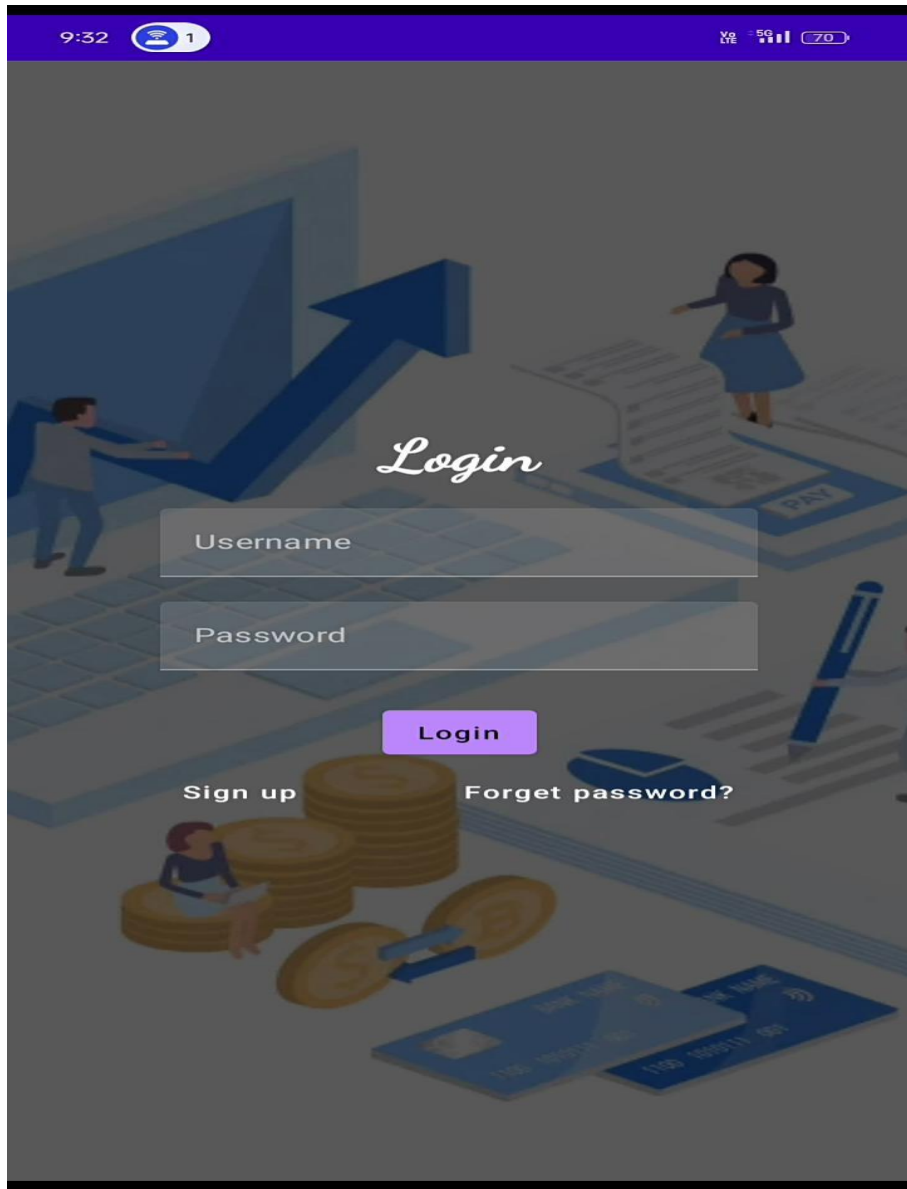
```kotlin
if (error.isNotEmpty()) {

    Text(

    text = error,

        color = MaterialTheme.colors.error,

      modifier = Modifier.padding(vertical = 16.dp)

    )      }

    Button(onClick = {

      if (items.isNotEmpty() && quantity.isNotEmpty() && cost.isNotEmpty()) {          val items = Items(

      id = null,

          itemName = items,

          quantity = quantity,

          cost = cost

      )

   val limit= expenseDatabaseHelper.getExpenseAmount(1)

      val actualvalue = limit?.minus(cost.toInt())

    // Toast.makeText(mContext, actualvalue.toString(), Toast.LENGTH_SHORT).show()

       val expense = Expense(

       id = 1,

          amount = actualvalue.toString()

 )           if (actualvalue != null) {               if (actualvalue < 1) {                 Toast.makeText(mContext, "Limit Over", Toast.LENGTH_SHORT).show()
} else {               expenseDatabaseHelper.updateExpense(expense)              itemsDatabaseHelper.insertItems(items)          }           }          }      })
{        Text(text = "Submit")       }    }}
```

```
)
    if (actualvalue != null) {
            if (actualvalue < 1) {
                Toast.makeText(mContext, "Limit Over", Toast.LENGTH_SHORT).show()
            }
else
{

            expenseDatabaseHelper.updateExpense(expense)
itemsDatabaseHelper.insertItems(items)
    }
    }
    }
 })
 {

    Text(text = "Submit))
```

Item Name

Item Name
apple

Quantity of item

Quantity
5

Cost of the item

Cost
240

Submit

View Records

Item_Name: apple
Quantity: 5
Cost: 50

Item_Name: apple
Quantity: 5
Cost: 50

Item_Name: apple
Quantity: 5
Cost: 50

Item_Name: apple
Quantity: 5
Cost: 50

Item_Name: apple
Quantity: 5
Cost: 50

Item_Name: apple
Quantity: 5
Cost: 50

Item_Name: apple
Quantity: 5
Cost: 50

Item_Name: apple
Quantity: 5
Cost: 50

Add Expenses | Set Limit | View Records