A PERSONAL FINANCE MANAGEMENT APP

- **DHANUSHKUMAR J**
HARSHAVARADHAN G
KARTHIGA P
DIBAHAKAR B

DESCRIPTION

•A Personal Finance Management App is a mobile or web-based tool designed to help individuals organize, track, and manage their financial activities. These apps offer a comprehensive suite of features that users monitor income, expenses, savings, investments, debt, and financial goals—all in one place.

•It provides comprehensive tools for budgeting, saving, investing, and managing debt while offering personalize insights, reminders, and security its . With its emphasis on tracking expenses, setting goal, and making smarter financial decisions.

•The ability to track investments, monitor credit scores, and securely store sensitive data adds further value, especially for users with more complex financial needs.

•Electronic scoring software can be a powerful tool for evaluating personal finance management apps, allowing users to systematically score features like budgeting, expense tracking, security, and usability.

Whether you are an individual looking .

•whether they're looking to budget, save, invest, or pay off debt. With essential features like expense tracking, budgeting, goal setting, investment tracking, and security, along with advanced AI insights and integrations.The platform is designed to help individuals track income, expenses, savings, investments, set financial goals, and monitor credit scores, all in one place. It should provide insights and tools that empower users to make informed financial decisions and improve their financial health

MAIN ACTIVITY.KT

```
package com.example.myapplication

import ...

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            MyApplicationTheme {
                Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding ->
                    Greeting(
                        name = "Android",
                        modifier = Modifier.padding(innerPadding)
                    )
                }
            }
        }
    }
```

```
@Composable
fun Greeting(name: String, modifier: Modifier = Modifier) {
    Text(
        text = "Hello $name!",
        modifier = modifier
    )
}

@Preview(showBackground = true)
@Composable
fun GreetingPreview() {
    MyApplicationTheme {
        Greeting( name: "Android")
    }
}
```

ANDROID MAINFEST.XML

```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="Expenses Tracker"
        android:supportsRtl="true"
        android:theme="@style/Theme.ExpensesTracker"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:label="Expenses Tracker"
            android:theme="@style/Theme.MyApplication">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
```

```
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
<activity
    android:name=".RegisterActivity"
    android:exported="false"
    android:label="RegisterActivity"
    android:theme="@style/Theme.ExpensesTracker" />
<activity
    android:name=".ViewRecordsActivity"
    android:exported="false"
    android:label="ViewRecordsActivity"
    android:theme="@style/Theme.ExpensesTracker" />
<activity
    android:name=".SetLimitActivity"
    android:exported="false"
    android:label="SetLimitActivity"
    android:theme="@style/Theme.ExpensesTracker" />
<activity
    android:name=".AddExpensesActivity"
    android:exported="false"
    android:label="AddExpensesActivity"
    android:theme="@style/Theme.ExpensesTracker" />
```

```
        <activity
            android:name=".LoginActivity"
            android:exported="true"
            android:label="Expenses Tracker"
            android:theme="@style/Theme.ExpensesTracker">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

```xml
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="androidx.tracing" >

    <uses-sdk
        android:minSdkVersion="14"
        android:targetSdkVersion="30" />

</manifest>
```

```xml
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="androidx.test.runner" >

    <uses-sdk
        android:minSdkVersion="14"
        android:targetSdkVersion="30" />

    <queries>
        <package android:name="androidx.test.orchestrator" />
        <package android:name="androidx.test.services" />
        <package android:name="com.google.android.apps.common.testing.services" />
    </queries>

    <application />

</manifest>
```

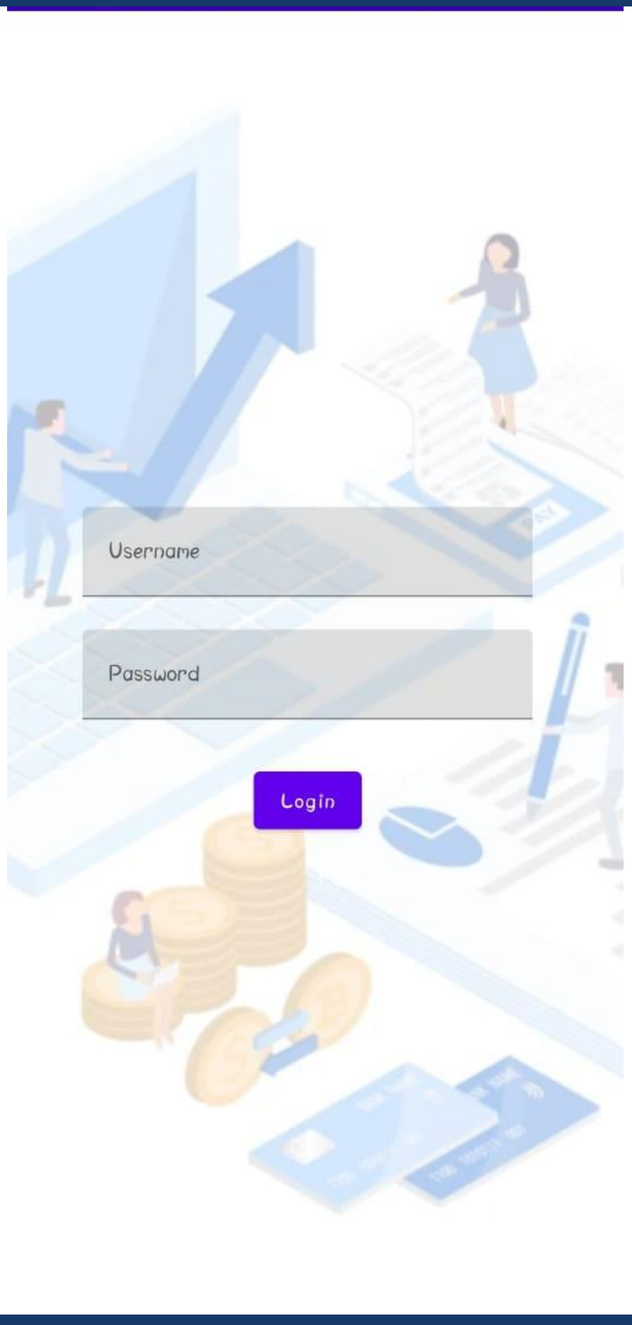
```
Manifest-Version: 1.0
Implementation-Title: kotlin-stdlib-jdk8
Kotlin-Runtime-Component: Main
Kotlin-Version: 1.8
Implementation-Version: 1.8.20-release-327(1.8.20)
Multi-Release: true
Implementation-Vendor: JetBrains
```

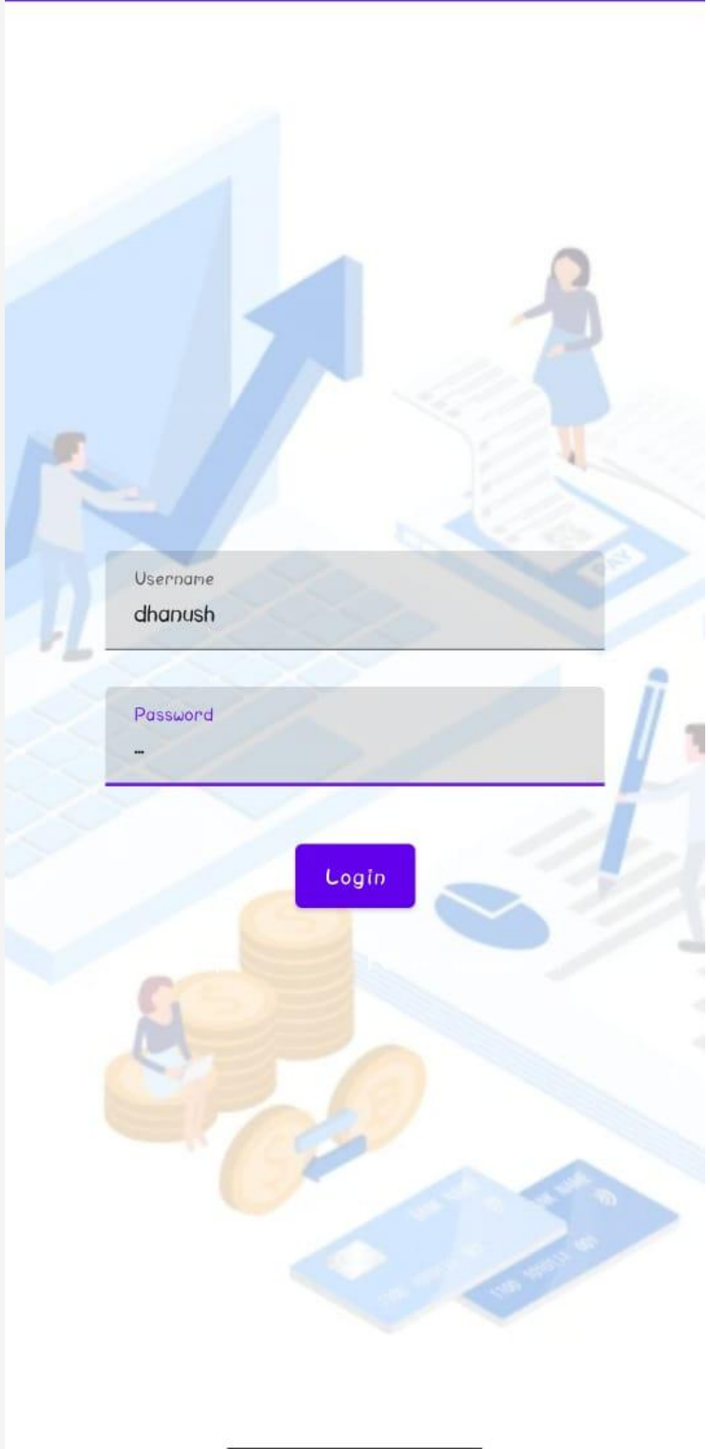
```kotlin
package com.example.myapplication

import ...

/**
 * Instrumented test, which will execute on an Android device.
 *
 * See [testing documentation](http://d.android.com/tools/testing).
 */
@RunWith(AndroidJUnit4::class)
class ExampleInstrumentedTest {
    @Test
    fun useAppContext() {
        // Context of the app under test.
        val appContext = InstrumentationRegistry.getInstrumentation().targetContext
        assertEquals( expected: "com.example.myapplication", appContext.packageName)
    }
}
```

ITEMS DAO.KT

```
package com.example.expensestracker

import androidx.room.*

@Dao
interface ItemsDao {

    @Query("SELECT * FROM items_table WHERE  cost= :cost")
    suspend fun getItemsByCost(cost: String): Items?

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    suspend fun insertItems(items: Items)

    @Update
    suspend fun updateItems(items: Items)

    @Delete
    suspend fun deleteItems(items: Items)
}
```

```kotlin
package com.example.expensestracker

import androidx.room.*

@Dao
interface ExpenseDao {

    @Query("SELECT * FROM expense_table WHERE  amount= :amount")
    suspend fun getExpenseByAmount(amount: String): Expense?

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    suspend fun insertExpense(items: Expense)

    @Update
    suspend fun updateExpense(items: Expense)

    @Delete
    suspend fun deleteExpense(items: Expense)
}
```


Username
Password
Login


Username
dhanush
Password
...
Login

Welcome To Expense Tracker

Add Expenses | Set Limit | View Records

Item Name

Item Name

bag

Quantity of item

Quantity

1

Cost of the item

Cost

850

Submit

Add Expenses

Set Limit

View Records

Monthly Amount Limit

Set Amount Limit

5000

Set Limit

Remaining Amount: 150
Remaining Amount: 5000
Remaining Amount: 5000
Remaining Amount: 5000
Remaining Amount: 5000
Remaining Amount: 5000
Remaining Amount: 5000
Remaining Amount: 1500
Remaining Amount: 1500
Remaining Amount: 1500
Remaining Amount: 1500
Remaining Amount: 1500
Remaining Amount: 1500
Remaining Amount: 1500
Remaining Amount: 1500

Add Expenses | Set Limit | View Records

View Records

Item_Name: ghh
Quantity: hnj
Cost: 500

Item_Name: ghh
Quantity: hnj
Cost: 500

Item_Name: ghh
Quantity: hnj
Cost: 500

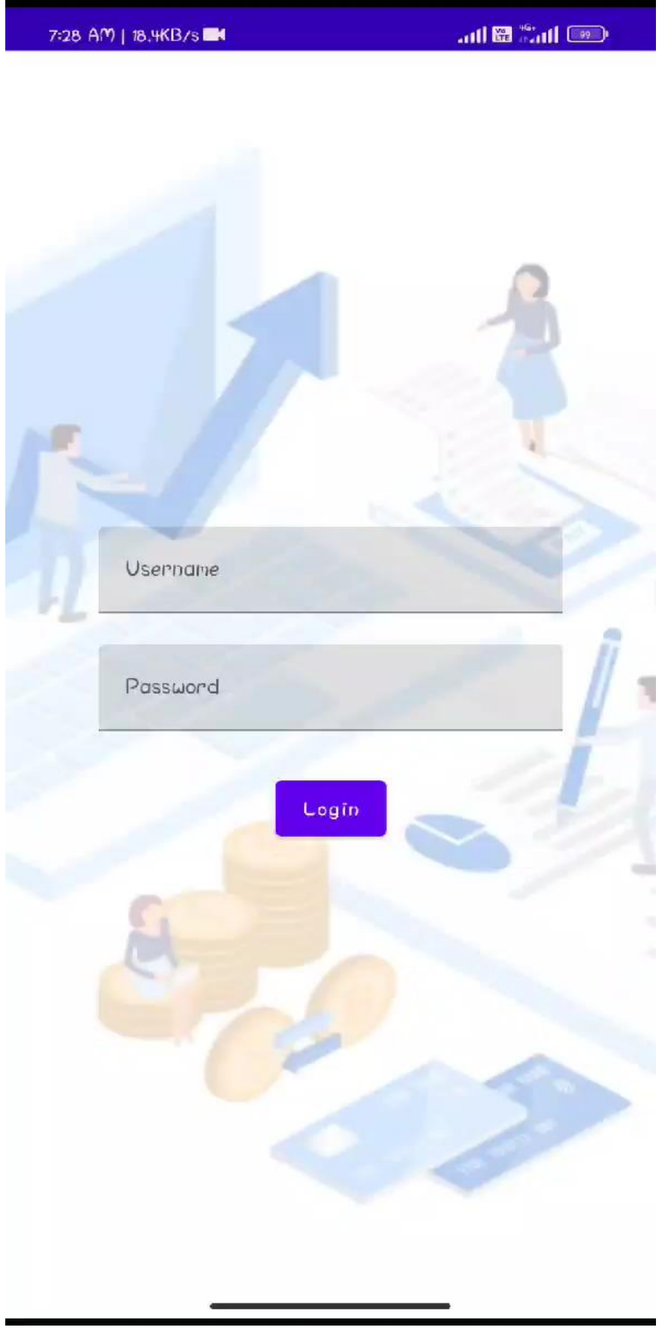
Item_Name: ghh
Quantity: hnj
Cost: 500

Item_Name: ghh
Quantity: hnj
Cost: 500

Item_Name: ghh
Quantity: hnj
Cost: 500

Item_Name: ghh
Quantity: hnj
Cost: 500

Add Expenses | Set Limit | View Records


7:28 AM | 18.4KB/s
Username
Password
Login