tA Personal Finance Management

1.INTRODUCTION

OVERVIEW:

At a very basic level, personal financial management simply means gaining an understanding of your financial situation in order to make the most of your assets in day-to-day life and in planning for your future.

What is the introduction of personal finance?

Personal finance, as a term, **covers the concepts of managing your money, saving, and investing**. It also includes banking, budgeting, mortgages, investments, insurance, retirement planning, and tax planning.

# Purpose

What Is Personal Finance?

Personal finance is the knowledge, instruments, and techniques used to

manage your finances. When you understand the principles and concepts behind personal finance, you can manage debt, savings, living expenses, and retirement savings.

There is a wide range of careers that relate to personal financial management and advice. If you’re passionate about any of the topics mentioned in this guide, you may want to consider a career in the industry.

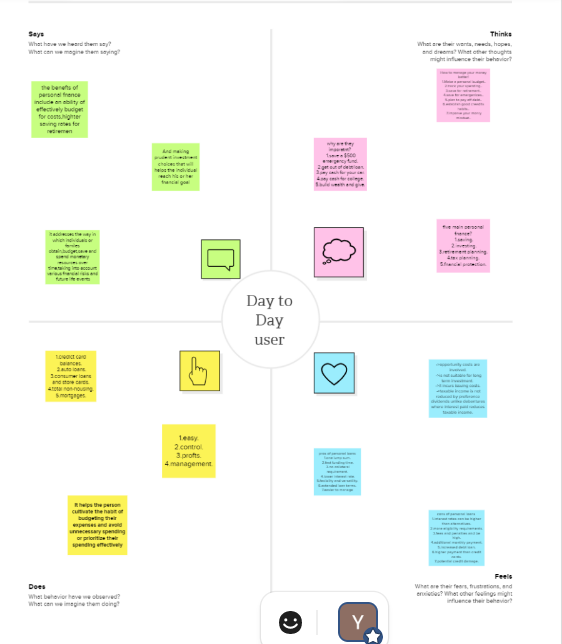
Some of the most common careers include:

* [Personal banker](https://corporatefinanceinstitute.com/resources/careers/jobs/personal-banker-job-description/)
* Wealth manager
* Investment advisor
* Insurance advisor
* Tax advisor

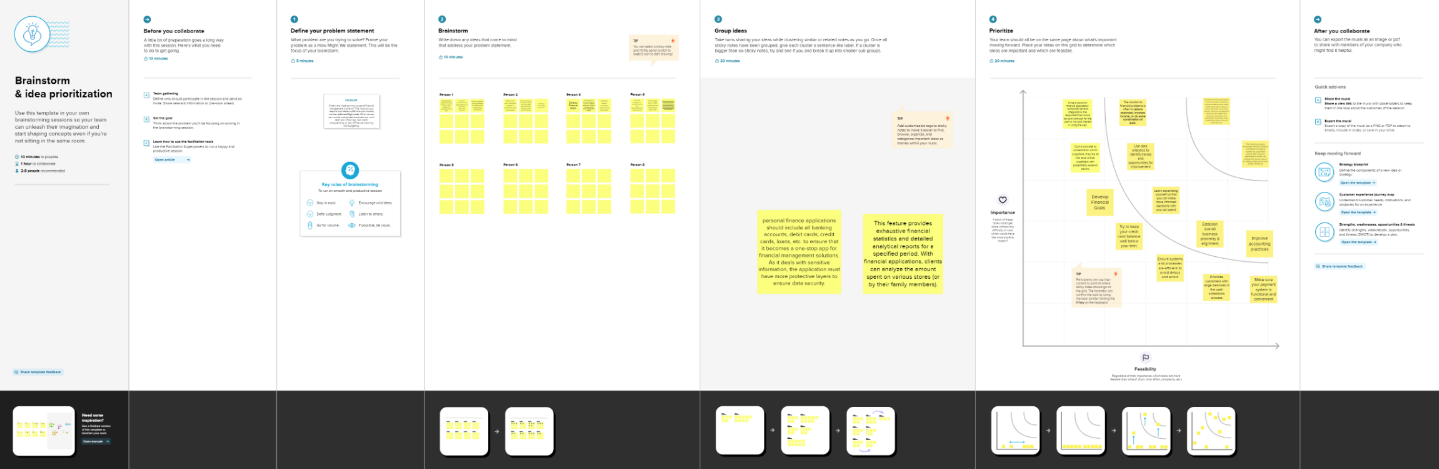


Problem Definition &Design Thinking

Empathy Map

******

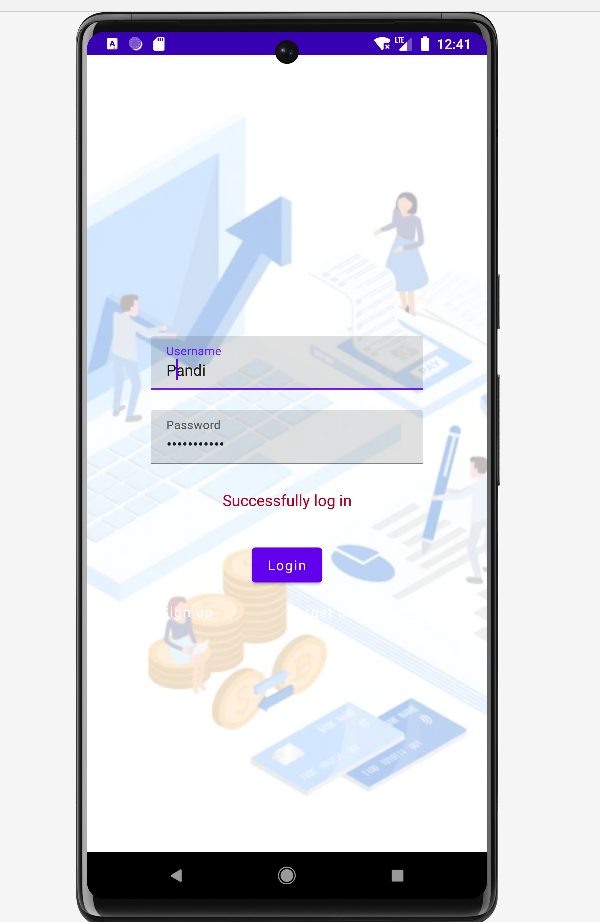
Ideation&Brainstorming Map

******

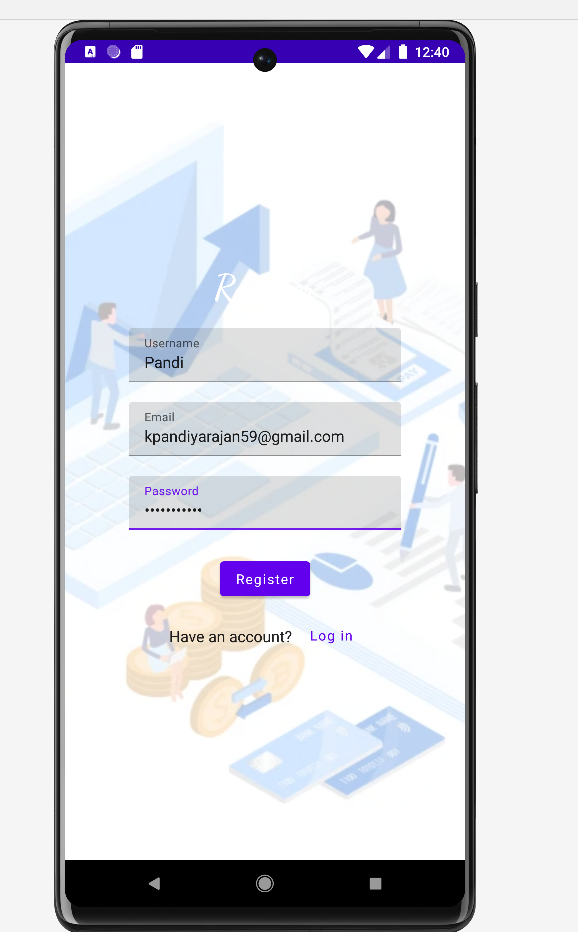
RESULT

Final findings(output)

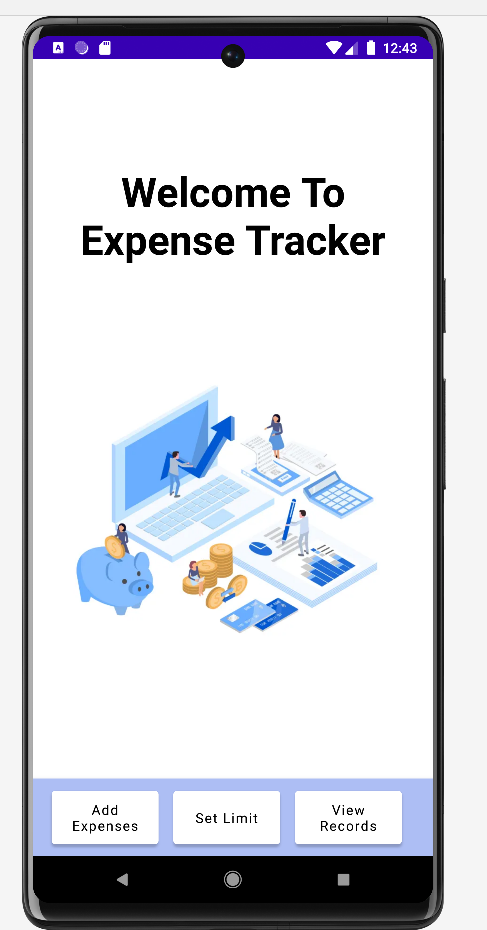
Login Page:

******

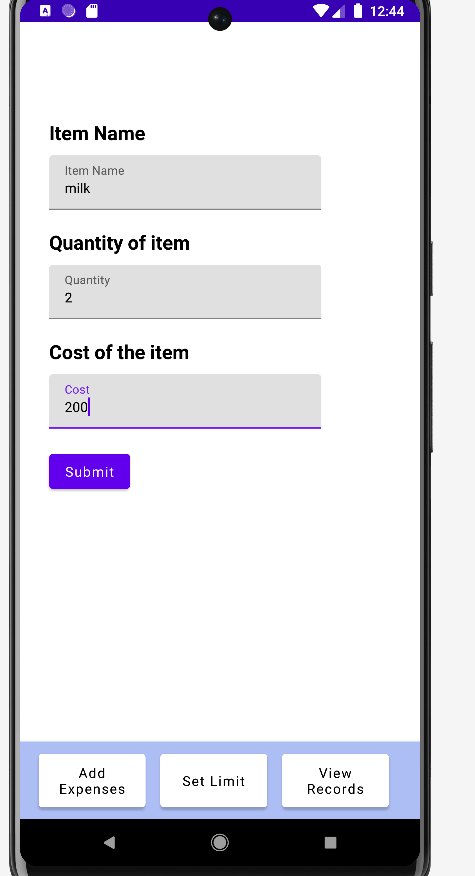
Register Page:

******

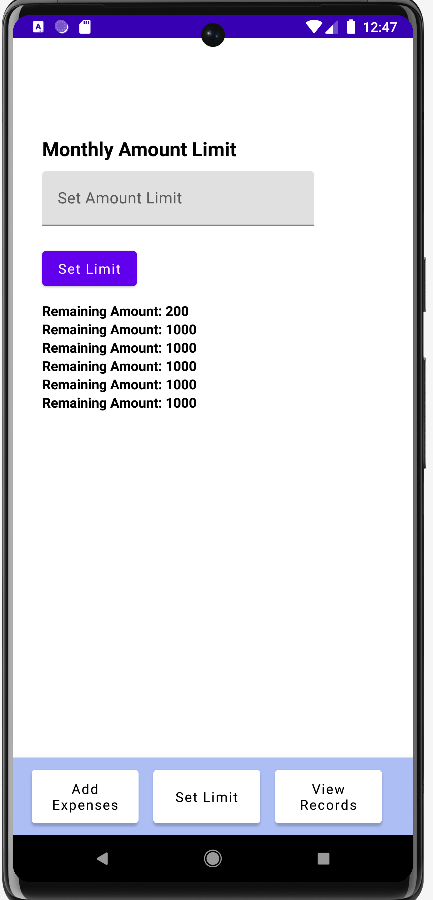
Main Page:

******

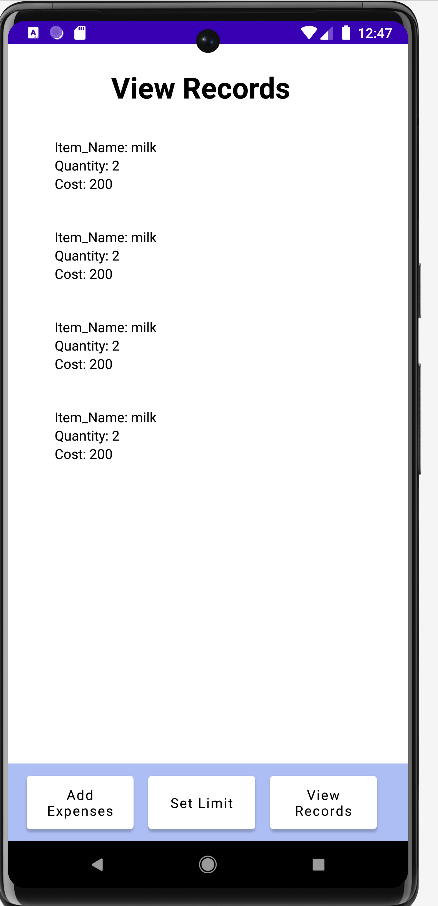
Add Expenses page:

******

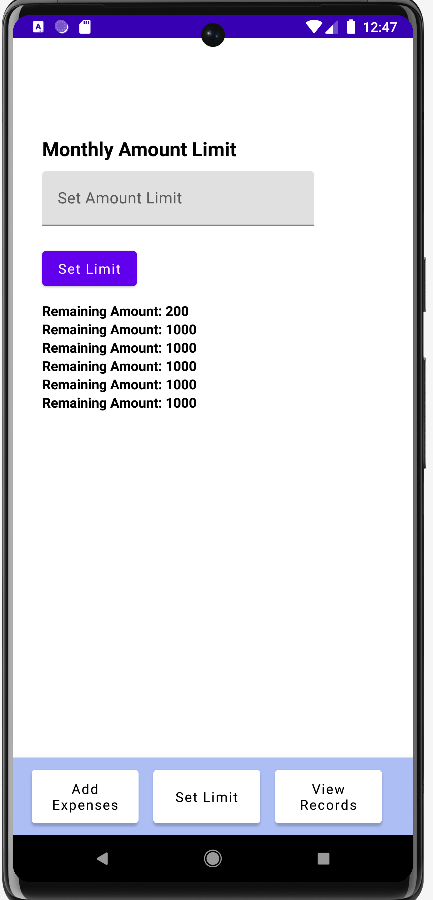
Set limit page before any data in expenses:

******

View records page:

******

Set limit page after expenses in add expense page

******

ADVANTAGES&DISADVANTAGES

###### Advantages of Financial Planning

Here are the advantages of financial planning:

* **Have significant financial control**Financial planning will help an individual have control over their expenses. If you have a goal set, your purpose behind saving will be clear. If there is a purpose, it will become easier for you to focus on it and move ahead towards achieving it.

* **Be prepared for unforeseen circumstances**Nobody is sure when there can be an emergency. Especially in your old age, you would not want to take other loans. Financial planning will keep you prepared for any unforeseen circumstances. Be it health requirements or any other unavoidable expenses, you should be ready. The future is unpredictable, but you can at least be prepared for anything that might come your way.

* **Define a purpose**If you have a purpose in life, the path towards reaching that goal becomes easy. Having a proper financial plan can help you stay focused and be ambitious in your career. This is a very prominent advantage of financial planning. If you are sure about your financial goals, it will show you the path towards a secure future.

* **Stay away from stress and anxiety about future finances**If you do not have to worry about your finances in the future, you can live a stress-free life now. One of the advantages of [**financial management**](https://www.tataaia.com/blogs/life-insurance/5-resolutions-to-become-financially-fit.html) is a stress-free present and a secure future. You can be at peace if you have your financial goals set. You do not need to worry about any uncertainties that may arise in the future.

* **Reap the benefits of your hard work in your old age**Best retirement planning will ensure you live a peaceful life post-retirement. If you plan well in advance, you can have your retirement sorted. The advantage of financial planning is that you secure your present and future. It is also a good habit to save. You will not have to take any loans to meet your expenses when you retire.

* **Get tax\* benefits**You can get tax\* benefits if you plan your future finances. Buying a life insurance plan is one of the top ways to save taxes u/s 80C of the Indian Income Tax\* Act. 1961. Also, you can opt for a retirement plan such as NPS to save on taxes\*.

###### Disadvantages of Financial Planning

Here are the disadvantages of financial planning:

* **Time-consuming process**The limitation of financial planning is that it needs your time. You will not be able to see immediate results. The process of understanding your income, expenses, and savings is crucial. Be aware of your finances and future aspirations to set realistic financial goals. The whole process of developing a retirement plan is time-consuming. **Predictions and actuality might not be in line**You will determine your financial goal based on some predictions and forecasts now. These forecasts and predictions might not hold in the future. They are subject to various conditions you might or might not consider. Thus, your financial plan might go for a toss if the predictions do not align with reality. To be cautious, you might have enough room for alterations in your financial plan. Have an open mind.

* **You must keep track of it constantly**It is not enough to make a financial plan. You must keep track of the conditions around you and alter your plans if needed. Having a rigid financial plan might not be the idea. Market conditions need to be considered for your retirement plan to work out. Even when the conditions are favourable, track your progress. See how close you are to your financial goals. Plan your moves to achieve the goals in time and optimise your efforts.

5.APPLICATIONS

What is a personal finance application?

A personal finance app can **help you keep track of your expenses, savings, investments, and bill payments and update you on credit score changes**. You can connect the personal finance app to your financial institutions to see where your bank account money is being spent.

## **How to Build a Personal Finance Application?**

Building a personal finance application can be a complex process. However, with the help of a professional app development company and the right steps followed, you can gain ground in your FinTech app development project easily and efficiently. Here are some general steps to help you get started.

### **Step 1: Define Your Audience**

The first step in building a personal finance application is to define your target audience; and comprehend the pain points and problems you want to solve. Once you know the challenges your audience is facing, you can choose the required tech stack and features to develop an effective application.

### **Step 2: Set the Purpose of the Application**

Now, it is time to set the purpose of your application. Determine what financial information you want to track, and what features the application will provide. Consider if it’s just for budgeting or for investment tracking, loan calculators or debt reduction plans, or a combination of several features.

### **Step 3: Gather Requirements**

Once you have defined the purpose of your application, do market research and gather all the requirements for your app. It includes features, user interface design, data management, security, and any other necessary functionalities. You should research APIs to integrate data sources and [data visualization](https://appinventiv.com/blog/data-visualization-in-business/) libraries for data presentation.

### **Step 4: Design the User Interface**

[Designing the user interface (UI)](https://appinventiv.com/blog/proven-tips-to-enhance-mobile-app-design-by-improving-its-ui/) is a critical step to build a personal finance application. The user interface should be simple and easy to navigate. It should provide a good user experience, making it easy for users to access information, track expenses, and manage their finances. The UI should also be responsive, working seamlessly on different devices and platforms.

### **Step 5: Develop and Test**

After finalizing the UI design, the development process begins. Start by building the basic functionalities, including user registration and login, data input, and data management. You should also add features like charts and graphs to help users visualize their financial data. Once the development is complete, conduct thorough testing to ensure the application functions smoothly without any issues.

### **Step 6: Launch the Application**

After completing testing and fixing any issues, it’s time to [launch the application](https://appinventiv.com/blog/tips-successful-mobile-app-launch/). You can start by releasing the application to a small group of beta testers to receive feedback and make improvements before launching it publicly. You can also opt for an MVP (minimal viable product) at this stage. Once you get the satisfactory result from MVP, you can consider scaling up the app by introducing more customer-centric features.

Plan for a marketing strategy to promote the app, such as social media, paid ads, or content marketing, to attract new users and build a user base.

### **Step 7: Maintain and Update the Application**

Maintaining and updating the application is an ongoing process. You should continually monitor user feedback and make necessary changes to improve the user experience. Additionally, stay up-to-date on industry trends and security concerns to ensure your application is always relevant and secure.

6.CONCLUSION

Personal financial management is done by every individual on some level. The key is to strike the right balance between income, expenses, savings, and investments. This balance will ensure that the personal financial planning and management of the individual are optimum

All of that, were about its beneficial of financial planning, whatshould we do or how we can prevent when we face to financial problem and planningfor the better life style. For this personal financial planning is a kind of useful softskills in our future life, it will bring us improvement of standard of living and achieveor goals. After discussion in this assignment, we gain of understanding for financialplanning process, we also learn more about beneficial of using the personal financialplanning by influencing economic environment.For example, we can classify aboutour investment and planing our wealth for better when make our decision in futureplanning. We also will learn about the keys principle of the personal financial

Money management refers to the process of tracking and planning an individual or group’s use of capital. In personal finance, money management includes budgeting, spending, saving, and investing.

In corporate finance, money management covers the raising and use of capital. A firm’s budgeting is mainly influenced by its business strategies.

In financial markets, money management refers to portfolio management and investment management.

7.FUTURE SCOPE

## **The Future of Personal Finance Looks Toward Inclusion and Places a Focus on Improving Financial Health**

Personal finance is the management of your financial resources through different strategies such as planning, saving, investing, and spending. In other words, personal finance includes all aspects of managing your finances, whether they be short- or long-term, as well as how your financial condition changes throughout your lifetime.

By understanding the fundamentals of personal finance, you can better grasp possibilities and solutions to improve your finances. This knowledge can assist you in setting a budget for immediate requirements while making long-term financial plans. To give you a clearer idea, this article will offer key information about personal finance and how it can catalyze building a more equitable and balanced financial future.

### **Financial Literacy**

Understanding and being proficient in the use of different financial skills, such as investing, budgeting, and personal financial management, is known as financial literacy. The cornerstone of your relationship with money is laid out by financial literacy, which is a lifetime learning process. The earlier you begin, the better off you will be — education is the secret to financial success.

### **Understanding Personal Finance**

The greatest way to begin building a strong personal financial foundation is to improve financial literacy and better understand what personal finance is. It will be easier for you to identify ways to strengthen your own finances and achieve your short- and long-term financial goals once you have a better understanding of what they are.

Personal finance has many different components, yet they fall into one of five categories:

* **Income:** It’s made up of your cash flow, or the money you receive from all sources, and is the cornerstone of your personal finances. Your salary, pension, Social Security benefits, rental income, business income, and investment income are all included.
* **Spending:** It refers to the money that goes to covering all of your expenses, including food, rent, and even vacations, and other items in your life that need money.
* **Savings:** It’s any money from your income that you set aside for the future rather than for immediate spending. Savings is a sum of money that can be used to cover both anticipated and unexpected future expenses.
* **Investing:** Investments are expenditures that give you the chance to build savings or a future income. Purchases of mutual funds, equities, bonds, or real estate that you anticipate will provide a high rate of return are examples of investments.
* **Protection:**These are financial products that you purchase and in return, they can offer monetary safety or protection against unforeseen financial expenses. Annuities, property or casualty insurance, life insurance, and health insurance are just a few examples of financial products that can be used to manage financial risk.

### **Financial Inclusion**

Since personal finance is the management of anything financial, it encompasses:

* Budgeting
* Banking
* Insurance
* Mortgages
* Investments
* Retirement
* Tax
* Estate planning
* Daily necessities
* And more

To have solid personal finance management, you also need to understand what financial inclusion is. Financial inclusion refers to the availability to both individuals and businesses of useful and cost-effective financial goods and services, including payments, transactions, savings, credit, and insurance, that are provided sustainably and ethically. As a result, it’s important to have access to all of your financial transactions to better understand your financial inclusion.

### **Digitalization: Improved Access**

The integration of various technologies and methods is referred to as finance digitalization, which enables the finance function to provide value in the digital era. It has the potential to increase the safety, convenience, and accessibility of financial services, all of which would be advantageous for consumers’ and enterprises’ financial health.

Digitalization is a top priority investment for the finance function. It improves processing times, boosts productivity, and lowers expenses. Given how important digitalization is for your finances, you’ll need a reliable source, such as[Money Fit](https://www.moneyfit.org/).

They provide consumers with a range of debt relief solutions to assist them in building a bright financial future through the development of strong fundamental personal finance competencies. With Money Fit, you’ll have improved access to all of your financial transactions, understand your financial situation better, and have strong and advanced financial inclusion.

### **Essential Drivers for a Balanced and Fair Financial Future**

The most common practices you may hear or observe from others in the effort to create a balanced and fair financial future are increasing your source of income, saving more money, and investing. You should work hard to secure a bright financial future, but how can you take these actions effectively? One of the best ways is innovation and working with an institution that has goals similar to yours.

### **Innovation**

One of the best things to happen to humanity in the modern era is innovation. Most people’s lives are considerably improved as a result. But innovation is also a major factor in the finance industry. Because of financial innovation, ordinary people’s lives have improved. Numerous categories can be used to categorize financial innovation. There are advancements in corporate banking and personal banking.

Here are a few examples of financial innovations:

* Automated Teller Machine (ATM)
* Credit Cards
* Electronic Banking
* SWIFT Messaging System
* Lockbox
* Credit Default Swaps
* Collateralized Debt Obligations
* Securitization
* Many more

Along with many other advances, financial innovation also includes better risk management, risk transfer, credit generation, and equity generation. With this type of financial innovation, you can have customized investments and improved wealth management. You’ll be more financially literate and sophisticated with your insurance premium calculations, debt management, and investment planning, which will help you create a solid financial future for yourself.

### **Partnerships With Institutions**

One of the best ways to achieve your financial objectives and ensure a more fair and balanced financial future is to partner with organizations and institutions that share your financial goals. Here are some trustworthy companies and institutions with which you can partner.

### **Banking Institutions**

One excellent strategy to ensure a stable and fair financial future is to collaborate with banking institutions like[JP Morgan Chase](https://www.jpmorganchase.com/impact/our-approach/financial-health). They are utilizing the power of technology to find and expand the appeal of the newest financial technology products and services that cater to the demands of people in need. They offer technological innovation and digitization that can help you on your path to achieving your financial goals. A bank company like JP Morgan Chase offers:

* **Financial Solutions:** To develop, foster, and scale up novel concepts that improve the financial well-being of low- to moderate-income consumers and historically underserved communities.
* **Financial Inclusion:** Foster creative, technologically-enabled solutions to address the financial well-being of consumers.
* **Catalyst Fund:**BFA Global’s inclusive fintech accelerator improves financial well-being for underprivileged people worldwide, with a focus on Kenya, Nigeria, South Africa, India, and Mexico.

Other bank institutions also offer great innovation and digitalization with programs that not only help you achieve your financial goals but also help other people around the world.

### **Nonprofit Organizations**

A nonprofit company like[Money Fit](https://www.moneyfit.org/) by DRS Inc. provides consumers with a range of debt relief services to assist them in overcoming their current difficulties and building a strong financial future.

The nonprofit organization’s main goals are to prevent financial problems and to aid people in acquiring solid fundamental knowledge of personal finance. Money Fit Academy offers free access to its credit counseling services and personal finance education programs. The fees for their debt management program are often affordable and worthwhile for consumers who are eligible to enroll in our debt relief program, which consolidates unsecured debt into a single monthly payment.

### **Community Members**

Along with collaborating with financial institutions and non-profit organizations, community members are one of the best parties to think about partnering with. For example, a financial corporation such as[Capital One](https://www.capitalone.com/about/our-commitments/) is a community member that is focused on creating a world where everyone has an equal opportunity to prosper, particularly in finance.

They help community relief efforts by promoting economic resilience through direct financial transfers. Additionally, they support a variety of projects such as making affordable housing more accessible, helping new businesses by assisting owners with the right pitch, enabling employers to provide financial safety nets, and many others.

By working with community members who share your objective for a secure and fair financial future, you can achieve your financial goal and help others achieve theirs too.

### 

8.APPENDIX

1.Creating the database classes for user login and registration

Step 1 : Create User data class

packagecom.example.expensestracker

importandroidx.room.ColumnInfo

importandroidx.room.Entity

importandroidx.room.PrimaryKey

@Entity(tableName ="user\_table")

data classUser(

@PrimaryKey(autoGenerate =true) valid:Int?,

@ColumnInfo(name ="first\_name") valfirstName:String?,

@ColumnInfo(name ="last\_name") vallastName:String?,

@ColumnInfo(name ="email") valemail:String?,

@ColumnInfo(name ="password") valpassword:String?,

)

Step 2 : Create an UserDao interface

packagecom.example.expensestracker

importandroidx.room.\*

@Dao

interfaceUserDao{

@Query("SELECT \* FROM user\_table WHERE email = :email")

suspendfungetUserByEmail(email:String): User?

@Insert(onConflict =OnConflictStrategy.REPLACE)

suspendfuninsertUser(user:User)

@Update

suspendfunupdateUser(user:User)

@Delete

suspendfundeleteUser(user:User)

}

Step 3 : Create an UserDatabase class

packagecom.example.expensestracker

importandroid.content.Context

importandroidx.room.Database

importandroidx.room.Room

importandroidx.room.RoomDatabase

@Database(entities =[User::class], version =1)

abstractclassUserDatabase: RoomDatabase() {

abstractfunuserDao(): UserDao

companionobject{

@Volatile

privatevarinstance:UserDatabase?=null

fungetDatabase(context:Context): UserDatabase{

returninstance ?:synchronized(this) {

valnewInstance =Room.databaseBuilder(

context.applicationContext,

UserDatabase::class.java,

"user\_database"

).build()

instance =newInstance

newInstance

}

}

}

}

Step 4 : Create an UserDatabaseHelper class

packagecom.example.expensestracker

importandroid.annotation.SuppressLint

importandroid.content.ContentValues

importandroid.content.Context

importandroid.database.Cursor

importandroid.database.sqlite.SQLiteDatabase

importandroid.database.sqlite.SQLiteOpenHelper

classUserDatabaseHelper(context:Context) :

SQLiteOpenHelper(context, DATABASE\_NAME, null, DATABASE\_VERSION) {

companionobject{

privateconstvalDATABASE\_VERSION=1

privateconstvalDATABASE\_NAME="UserDatabase.db"

privateconstvalTABLE\_NAME="user\_table"

privateconstvalCOLUMN\_ID="id"

privateconstvalCOLUMN\_FIRST\_NAME="first\_name"

privateconstvalCOLUMN\_LAST\_NAME="last\_name"

privateconstvalCOLUMN\_EMAIL="email"

privateconstvalCOLUMN\_PASSWORD="password"

}

overridefunonCreate(db:SQLiteDatabase?) {

valcreateTable ="CREATE TABLE $TABLE\_NAME("+

"$COLUMN\_IDINTEGER PRIMARY KEY AUTOINCREMENT, "+

"$COLUMN\_FIRST\_NAMETEXT, "+

"$COLUMN\_LAST\_NAMETEXT, "+

"$COLUMN\_EMAILTEXT, "+

"$COLUMN\_PASSWORDTEXT"+

")"

db?.execSQL(createTable)

}

overridefunonUpgrade(db:SQLiteDatabase?, oldVersion:Int, newVersion:Int) {

db?.execSQL("DROP TABLE IF EXISTS $TABLE\_NAME")

onCreate(db)

}

funinsertUser(user:User) {

valdb =writableDatabase

valvalues =ContentValues()

values.put(COLUMN\_FIRST\_NAME, user.firstName)

values.put(COLUMN\_LAST\_NAME, user.lastName)

values.put(COLUMN\_EMAIL, user.email)

values.put(COLUMN\_PASSWORD, user.password)

db.insert(TABLE\_NAME, null, values)

db.close()

}

@SuppressLint("Range")

fungetUserByUsername(username:String): User?{

valdb =readableDatabase

valcursor:Cursor=db.rawQuery("SELECT \* FROM $TABLE\_NAMEWHERE $COLUMN\_FIRST\_NAME= ?", arrayOf(username))

varuser:User?=null

if(cursor.moveToFirst()) {

user =User(

id =cursor.getInt(cursor.getColumnIndex(COLUMN\_ID)),

firstName =cursor.getString(cursor.getColumnIndex(COLUMN\_FIRST\_NAME)),

lastName =cursor.getString(cursor.getColumnIndex(COLUMN\_LAST\_NAME)),

email =cursor.getString(cursor.getColumnIndex(COLUMN\_EMAIL)),

password =cursor.getString(cursor.getColumnIndex(COLUMN\_PASSWORD)),

)

}

cursor.close()

db.close()

returnuser

}

@SuppressLint("Range")

fungetUserById(id:Int): User?{

valdb =readableDatabase

valcursor:Cursor=db.rawQuery("SELECT \* FROM $TABLE\_NAMEWHERE $COLUMN\_ID= ?", arrayOf(id.toString()))

varuser:User?=null

if(cursor.moveToFirst()) {

user =User(

id =cursor.getInt(cursor.getColumnIndex(COLUMN\_ID)),

firstName =cursor.getString(cursor.getColumnIndex(COLUMN\_FIRST\_NAME)),

lastName =cursor.getString(cursor.getColumnIndex(COLUMN\_LAST\_NAME)),

email =cursor.getString(cursor.getColumnIndex(COLUMN\_EMAIL)),

password =cursor.getString(cursor.getColumnIndex(COLUMN\_PASSWORD)),

)

}

cursor.close()

db.close()

returnuser

}

@SuppressLint("Range")

fungetAllUsers(): List<User> {

valusers =mutableListOf<User>()

valdb =readableDatabase

valcursor:Cursor=db.rawQuery("SELECT \* FROM $TABLE\_NAME", null)

if(cursor.moveToFirst()) {

do{

valuser =User(

id =cursor.getInt(cursor.getColumnIndex(COLUMN\_ID)),

firstName =cursor.getString(cursor.getColumnIndex(COLUMN\_FIRST\_NAME)),

lastName =cursor.getString(cursor.getColumnIndex(COLUMN\_LAST\_NAME)),

email =cursor.getString(cursor.getColumnIndex(COLUMN\_EMAIL)),

password =cursor.getString(cursor.getColumnIndex(COLUMN\_PASSWORD)),

)

users.add(user)

} while(cursor.moveToNext())

}

cursor.close()

db.close()

returnusers

}

}

2.Creating the database classes for item name, quantity and cost.

Step 1 : Create Items data class

packagecom.example.expensestracker

importandroidx.room.ColumnInfo

importandroidx.room.Entity

importandroidx.room.PrimaryKey

@Entity(tableName ="items\_table")

data classItems(

@PrimaryKey(autoGenerate =true) valid:Int?,

@ColumnInfo(name ="item\_name") valitemName:String?,

@ColumnInfo(name ="quantity") valquantity:String?,

@ColumnInfo(name ="cost") valcost:String?,

)

Step 2 : Create ItemsDao interface

packagecom.example.expensestracker

importandroidx.room.\*

@Dao

interfaceItemsDao{

@Query("SELECT \* FROM items\_table WHERE cost= :cost")

suspendfungetItemsByCost(cost:String): Items?

@Insert(onConflict =OnConflictStrategy.REPLACE)

suspendfuninsertItems(items:Items)

@Update

suspendfunupdateItems(items:Items)

@Delete

suspendfundeleteItems(items:Items)

}

Step 3 : Create ItemsDatabse class

packagecom.example.expensestracker

importandroid.content.Context

importandroidx.room.Database

importandroidx.room.Room

importandroidx.room.RoomDatabase

@Database(entities =[Items::class], version =1)

abstractclassItemsDatabase: RoomDatabase() {

abstractfunItemsDao(): ItemsDao

companionobject{

@Volatile

privatevarinstance:ItemsDatabase?=null

fungetDatabase(context:Context): ItemsDatabase{

returninstance ?:synchronized(this) {

valnewInstance =Room.databaseBuilder(

context.applicationContext,

ItemsDatabase::class.java,

"items\_database"

).build()

instance =newInstance

newInstance

}

}

}

}

Step 4 : Create ItemsDatabaseHelper class

packagecom.example.expensestracker

importandroid.annotation.SuppressLint

importandroid.content.ContentValues

importandroid.content.Context

importandroid.database.Cursor

importandroid.database.sqlite.SQLiteDatabase

importandroid.database.sqlite.SQLiteOpenHelper

classItemsDatabaseHelper(context:Context) :

SQLiteOpenHelper(context, DATABASE\_NAME, null,DATABASE\_VERSION){

companionobject{

privateconstvalDATABASE\_VERSION=1

privateconstvalDATABASE\_NAME="ItemsDatabase.db"

privateconstvalTABLE\_NAME="items\_table"

privateconstvalCOLUMN\_ID="id"

privateconstvalCOLUMN\_ITEM\_NAME="item\_name"

privateconstvalCOLUMN\_QUANTITY="quantity"

privateconstvalCOLUMN\_COST="cost"

}

overridefunonCreate(db:SQLiteDatabase?) {

valcreateTable ="CREATE TABLE $TABLE\_NAME("+

"${COLUMN\_ID}INTEGER PRIMARY KEY AUTOINCREMENT, "+

"${COLUMN\_ITEM\_NAME}TEXT,"+

"${COLUMN\_QUANTITY}TEXT,"+

"${COLUMN\_COST}TEXT"+

")"

db?.execSQL(createTable)

}

overridefunonUpgrade(db:SQLiteDatabase?, oldVersion:Int, newVersion:Int) {

db?.execSQL("DROP TABLE IF EXISTS $TABLE\_NAME")

onCreate(db)

}

funinsertItems(items:Items) {

valdb =writableDatabase

valvalues =ContentValues()

values.put(COLUMN\_ITEM\_NAME, items.itemName)

values.put(COLUMN\_QUANTITY, items.quantity)

values.put(COLUMN\_COST, items.cost)

db.insert(TABLE\_NAME, null, values)

db.close()

}

@SuppressLint("Range")

fungetItemsByCost(cost:String): Items?{

valdb =readableDatabase

valcursor:Cursor=db.rawQuery("SELECT \* FROM $TABLE\_NAMEWHERE $COLUMN\_COST= ?", arrayOf(cost))

varitems:Items?=null

if(cursor.moveToFirst()) {

items =Items(

id =cursor.getInt(cursor.getColumnIndex(COLUMN\_ID)),

itemName =cursor.getString(cursor.getColumnIndex(COLUMN\_ITEM\_NAME)),

quantity =cursor.getString(cursor.getColumnIndex(COLUMN\_QUANTITY)),

cost =cursor.getString(cursor.getColumnIndex(COLUMN\_COST)),

)

}

cursor.close()

db.close()

returnitems

}

@SuppressLint("Range")

fungetItemsById(id:Int): Items?{

valdb =readableDatabase

valcursor:Cursor=db.rawQuery("SELECT \* FROM $TABLE\_NAMEWHERE $COLUMN\_ID= ?", arrayOf(id.toString()))

varitems:Items?=null

if(cursor.moveToFirst()) {

items =Items(

id =cursor.getInt(cursor.getColumnIndex(COLUMN\_ID)),

itemName =cursor.getString(cursor.getColumnIndex(COLUMN\_ITEM\_NAME)),

quantity =cursor.getString(cursor.getColumnIndex(COLUMN\_QUANTITY)),

cost =cursor.getString(cursor.getColumnIndex(COLUMN\_COST)),

)

}

cursor.close()

db.close()

returnitems

}

@SuppressLint("Range")

fungetAllItems(): List<Items> {

valitem =mutableListOf<Items>()

valdb =readableDatabase

valcursor:Cursor=db.rawQuery("SELECT \* FROM $TABLE\_NAME", null)

if(cursor.moveToFirst()) {

do{

valitems =Items(

id =cursor.getInt(cursor.getColumnIndex(COLUMN\_ID)),

itemName =cursor.getString(cursor.getColumnIndex(COLUMN\_ITEM\_NAME)),

quantity =cursor.getString(cursor.getColumnIndex(COLUMN\_QUANTITY)),

cost =cursor.getString(cursor.getColumnIndex(COLUMN\_COST)),

)

item.add(items)

} while(cursor.moveToNext())

}

cursor.close()

db.close()

returnitem

}

}

3.Creating the database classes for an amount.

Step 1 : Create Expense data class

packagecom.example.expensestracker

importandroidx.room.ColumnInfo

importandroidx.room.Entity

importandroidx.room.PrimaryKey

@Entity(tableName ="expense\_table")

data classExpense(

@PrimaryKey(autoGenerate =true) valid:Int?,

@ColumnInfo(name ="amount") valamount:String?,

)

Step 2 : Create ExpenseDao interface

packagecom.example.expensestracker

importandroidx.room.\*

@Dao

interfaceExpenseDao{

@Query("SELECT \* FROM expense\_table WHERE amount= :amount")

suspendfungetExpenseByAmount(amount:String): Expense?

@Insert(onConflict =OnConflictStrategy.REPLACE)

suspendfuninsertExpense(items:Expense)

@Update

suspendfunupdateExpense(items:Expense)

@Delete

suspendfundeleteExpense(items:Expense)

}

Step 3 : Create ExpenseDatabase class

packagecom.example.expensestracker

importandroid.content.Context

importandroidx.room.Database

importandroidx.room.Room

importandroidx.room.RoomDatabase

@Database(entities =[Items::class], version =1)

abstractclassExpenseDatabase: RoomDatabase() {

abstractfunExpenseDao(): ItemsDao

companionobject{

@Volatile

privatevarinstance:ExpenseDatabase?=null

fungetDatabase(context:Context): ExpenseDatabase{

returninstance ?:synchronized(this) {

valnewInstance =Room.databaseBuilder(

context.applicationContext,

ExpenseDatabase::class.java,

"expense\_database"

).build()

instance =newInstance

newInstance

}

}

}

}

Step 4 : Create ExpenseDatabaseHelper class

packagecom.example.expensestracker

importandroid.annotation.SuppressLint

importandroid.content.ContentValues

importandroid.content.Context

importandroid.database.Cursor

importandroid.database.sqlite.SQLiteDatabase

importandroid.database.sqlite.SQLiteOpenHelper

classExpenseDatabaseHelper(context:Context) :

SQLiteOpenHelper(context, DATABASE\_NAME, null,DATABASE\_VERSION){

companionobject{

privateconstvalDATABASE\_VERSION=1

privateconstvalDATABASE\_NAME="ExpenseDatabase.db"

privateconstvalTABLE\_NAME="expense\_table"

privateconstvalCOLUMN\_ID="id"

privateconstvalCOLUMN\_AMOUNT="amount"

}

overridefunonCreate(db:SQLiteDatabase?) {

valcreateTable ="CREATE TABLE $TABLE\_NAME("+

"${COLUMN\_ID}INTEGER PRIMARY KEY AUTOINCREMENT, "+

"${COLUMN\_AMOUNT}TEXT"+

")"

db?.execSQL(createTable)

}

overridefunonUpgrade(db1:SQLiteDatabase?, oldVersion:Int, newVersion:Int) {

db1?.execSQL("DROP TABLE IF EXISTS $TABLE\_NAME")

onCreate(db1)

}

funinsertExpense(expense:Expense) {

valdb1 =writableDatabase

valvalues =ContentValues()

values.put(COLUMN\_AMOUNT, expense.amount)

db1.insert(TABLE\_NAME, null, values)

db1.close()

}

funupdateExpense(expense:Expense) {

valdb =writableDatabase

valvalues =ContentValues()

values.put(COLUMN\_AMOUNT, expense.amount)

db.update(TABLE\_NAME, values, "$COLUMN\_ID=?", arrayOf(expense.id.toString()))

db.close()

}

@SuppressLint("Range")

fungetExpenseByAmount(amount:String): Expense?{

valdb1 =readableDatabase

valcursor:Cursor=db1.rawQuery("SELECT \* FROM ${ExpenseDatabaseHelper.TABLE\_NAME}WHERE ${ExpenseDatabaseHelper.COLUMN\_AMOUNT}= ?", arrayOf(amount))

varexpense:Expense?=null

if(cursor.moveToFirst()) {

expense =Expense(

id =cursor.getInt(cursor.getColumnIndex(COLUMN\_ID)),

amount =cursor.getString(cursor.getColumnIndex(COLUMN\_AMOUNT)),

)

}

cursor.close()

db1.close()

returnexpense

}

@SuppressLint("Range")

fungetExpenseById(id:Int): Expense?{

valdb1 =readableDatabase

valcursor:Cursor=db1.rawQuery("SELECT \* FROM $TABLE\_NAMEWHERE $COLUMN\_ID= ?", arrayOf(id.toString()))

varexpense:Expense?=null

if(cursor.moveToFirst()) {

expense =Expense(

id =cursor.getInt(cursor.getColumnIndex(COLUMN\_ID)),

amount =cursor.getString(cursor.getColumnIndex(COLUMN\_AMOUNT)),

)

}

cursor.close()

db1.close()

returnexpense

}

@SuppressLint("Range")

fungetExpenseAmount(id:Int): Int?{

valdb =readableDatabase

valquery ="SELECT $COLUMN\_AMOUNTFROM $TABLE\_NAMEWHERE $COLUMN\_ID=?"

valcursor =db.rawQuery(query, arrayOf(id.toString()))

varamount:Int?=null

if(cursor.moveToFirst()) {

amount =cursor.getInt(cursor.getColumnIndex(COLUMN\_AMOUNT))

}

cursor.close()

db.close()

returnamount

}

@SuppressLint("Range")

fungetAllExpense(): List<Expense> {

valexpenses =mutableListOf<Expense>()

valdb1 =readableDatabase

valcursor:Cursor=db1.rawQuery("SELECT \* FROM $TABLE\_NAME", null)

if(cursor.moveToFirst()) {

do{

valexpense =Expense(

id =cursor.getInt(cursor.getColumnIndex(COLUMN\_ID)),

amount =cursor.getString(cursor.getColumnIndex(COLUMN\_AMOUNT)),

)

expenses.add(expense)

} while(cursor.moveToNext())

}

cursor.close()

db1.close()

returnexpenses

}

 }

Building application UI and connecting to database.

Step 1: Creating LoginActivity.kt with database

packagecom.example.expensestracker

importandroid.content.Context

importandroid.content.Intent

importandroid.os.Bundle

importandroidx.activity.ComponentActivity

importandroidx.activity.compose.setContent

importandroidx.compose.foundation.Image

importandroidx.compose.foundation.layout.\*

importandroidx.compose.material.\*

importandroidx.compose.runtime.\*

importandroidx.compose.ui.Alignment

importandroidx.compose.ui.Modifier

importandroidx.compose.ui.graphics.Color

importandroidx.compose.ui.layout.ContentScale

importandroidx.compose.ui.res.painterResource

importandroidx.compose.ui.text.font.FontFamily

importandroidx.compose.ui.text.font.FontWeight

importandroidx.compose.ui.text.input.PasswordVisualTransformation

importandroidx.compose.ui.text.input.VisualTransformation

importandroidx.compose.ui.tooling.preview.Preview

importandroidx.compose.ui.unit.dp

importandroidx.compose.ui.unit.sp

importandroidx.core.content.ContextCompat

importcom.example.expensestracker.ui.theme.ExpensesTrackerTheme

classLoginActivity: ComponentActivity() {

privatelateinitvardatabaseHelper:UserDatabaseHelper

overridefunonCreate(savedInstanceState:Bundle?) {

super.onCreate(savedInstanceState)

databaseHelper =UserDatabaseHelper(this)

setContent {

ExpensesTrackerTheme{

//A surface container using the 'background' color from the theme

Surface(

modifier =Modifier.fillMaxSize(),

color =MaterialTheme.colors.background

) {

LoginScreen(this, databaseHelper)

}

}

}

}

}

@Composable

funLoginScreen(context:Context, databaseHelper:UserDatabaseHelper) {

Image(

painterResource(id =R.drawable.img\_1), contentDescription ="",

alpha =0.3F,

contentScale =ContentScale.FillHeight,

)

varusername by remember { mutableStateOf("") }

varpassword by remember { mutableStateOf("") }

varerror by remember { mutableStateOf("") }

Column(

modifier =Modifier.fillMaxSize(),

horizontalAlignment =Alignment.CenterHorizontally,

verticalArrangement =Arrangement.Center

) {

Text(

fontSize =36.sp,

fontWeight =FontWeight.ExtraBold,

fontFamily =FontFamily.Cursive,

color =Color.White,

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") },

modifier =Modifier.padding(10.dp)

.width(280.dp),

visualTransformation =PasswordVisualTransformation()

)

if(error.isNotEmpty()) {

Text(

text =error,

color =MaterialTheme.colors.error,

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

)

}

Button(

onClick ={

if(username.isNotEmpty() &&password.isNotEmpty()) {

valuser =databaseHelper.getUserByUsername(username)

if(user !=null&&user.password ==password) {

error ="Successfully log in"

context.startActivity(

Intent(

context,

MainActivity::class.java

)

)

//onLoginSuccess()

}

else{

error ="Invalid username or password"

}

} else{

error ="Please fill all fields"

}

},

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

) {

Text(text ="Login")

}

Row{

TextButton(onClick ={context.startActivity(

Intent(

context,

RegisterActivity::class.java

)

)}

)

{ Text(color =Color.White,text ="Sign up") }

TextButton(onClick ={

})

{

Spacer(modifier =Modifier.width(60.dp))

Text(color =Color.White,text ="Forget password?")

}

}

}

}

privatefunstartMainPage(context:Context) {

valintent =Intent(context, MainActivity::class.java)

ContextCompat.startActivity(context, intent, null)

}

Step 2 : Creating RegisterActivity.kt with database

packagecom.example.expensestracker

importandroid.content.Context

importandroid.content.Intent

importandroid.os.Bundle

importandroidx.activity.ComponentActivity

importandroidx.activity.compose.setContent

importandroidx.compose.foundation.Image

importandroidx.compose.foundation.layout.\*

importandroidx.compose.material.\*

importandroidx.compose.runtime.\*

importandroidx.compose.ui.Alignment

importandroidx.compose.ui.Modifier

importandroidx.compose.ui.graphics.Color

importandroidx.compose.ui.layout.ContentScale

importandroidx.compose.ui.res.painterResource

importandroidx.compose.ui.text.font.FontFamily

importandroidx.compose.ui.text.font.FontWeight

importandroidx.compose.ui.text.input.PasswordVisualTransformation

importandroidx.compose.ui.tooling.preview.Preview

importandroidx.compose.ui.unit.dp

importandroidx.compose.ui.unit.sp

importandroidx.core.content.ContextCompat

importcom.example.expensestracker.ui.theme.ExpensesTrackerTheme

classRegisterActivity: ComponentActivity() {

privatelateinitvardatabaseHelper:UserDatabaseHelper

overridefunonCreate(savedInstanceState:Bundle?) {

super.onCreate(savedInstanceState)

databaseHelper =UserDatabaseHelper(this)

setContent {

ExpensesTrackerTheme{

//A surface container using the 'background' color from the theme

Surface(

modifier =Modifier.fillMaxSize(),

color =MaterialTheme.colors.background

) {

RegistrationScreen(this,databaseHelper)

}

}

}

}

}

@Composable

funRegistrationScreen(context:Context, databaseHelper:UserDatabaseHelper) {

Image(

painterResource(id =R.drawable.img\_1), contentDescription ="",

alpha =0.3F,

contentScale =ContentScale.FillHeight,

)

varusername by remember { mutableStateOf("") }

varpassword by remember { mutableStateOf("") }

varemail by remember { mutableStateOf("") }

varerror by remember { mutableStateOf("") }

Column(

modifier =Modifier.fillMaxSize(),

horizontalAlignment =Alignment.CenterHorizontally,

verticalArrangement =Arrangement.Center

) {

Text(

fontSize =36.sp,

fontWeight =FontWeight.ExtraBold,

fontFamily =FontFamily.Cursive,

color =Color.White,

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") },

modifier =Modifier

.padding(10.dp)

.width(280.dp),

visualTransformation =PasswordVisualTransformation()

)

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()) {

valuser =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"

}

},

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(text ="Log in")

}

}

}

}

privatefunstartLoginActivity(context:Context) {

valintent =Intent(context, LoginActivity::class.java)

ContextCompat.startActivity(context, intent, null)

}

Step 3 : Creating MainActivity.kt file

packagecom.example.expensestracker

importandroid.annotation.SuppressLint

importandroid.content.Intent

importandroid.os.Bundle

importandroidx.activity.ComponentActivity

importandroidx.activity.compose.setContent

importandroidx.compose.foundation.Image

importandroidx.compose.foundation.layout.\*

importandroidx.compose.material.\*

importandroidx.compose.runtime.\*

importandroidx.compose.ui.Alignment

importandroidx.compose.ui.Modifier

importandroidx.compose.ui.graphics.Color

importandroidx.compose.ui.res.painterResource

importandroidx.compose.ui.text.font.FontWeight

importandroidx.compose.ui.text.style.TextAlign

importandroidx.compose.ui.tooling.preview.Preview

importandroidx.compose.ui.unit.dp

importandroidx.compose.ui.unit.sp

importcom.example.expensestracker.ui.theme.ExpensesTrackerTheme

classMainActivity: ComponentActivity() {

@SuppressLint("UnusedMaterialScaffoldPaddingParameter")

overridefunonCreate(savedInstanceState:Bundle?) {

super.onCreate(savedInstanceState)

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 ={

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

)

)

},

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

)

)

},

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

)

}

}

)

}

) {

MainPage()

}

}

}

}

@Composable

funMainPage() {

Column(

modifier =Modifier.padding(20.dp).fillMaxSize(),

verticalArrangement =Arrangement.Center,

horizontalAlignment =Alignment.CenterHorizontally

) {

Text(text ="Welcome To Expense Tracker", fontSize =42.sp, fontWeight =FontWeight.Bold,

textAlign =TextAlign.Center)

Image(painterResource(id =R.drawable.img\_1), contentDescription ="", modifier =Modifier.size(height =500.dp, width =500.dp))

}

}

Step 4 : Creating AddExpensesActivity.kt file

packagecom.example.expensestracker

importandroid.annotation.SuppressLint

importandroid.content.Context

importandroid.content.Intent

importandroid.os.Bundle

importandroid.widget.Toast

importandroidx.activity.ComponentActivity

importandroidx.activity.compose.setContent

importandroidx.compose.foundation.layout.\*

importandroidx.compose.material.\*

importandroidx.compose.runtime.\*

importandroidx.compose.ui.Alignment

importandroidx.compose.ui.Modifier

importandroidx.compose.ui.graphics.Color

importandroidx.compose.ui.platform.LocalContext

importandroidx.compose.ui.text.font.FontWeight

importandroidx.compose.ui.text.style.TextAlign

importandroidx.compose.ui.unit.dp

importandroidx.compose.ui.unit.sp

classAddExpensesActivity: ComponentActivity() {

privatelateinitvaritemsDatabaseHelper:ItemsDatabaseHelper

privatelateinitvarexpenseDatabaseHelper:ExpenseDatabaseHelper

@SuppressLint("UnusedMaterialScaffoldPaddingParameter")

overridefunonCreate(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 ={

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

)

)

},

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

)

)

},

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")

@Composable

funAddExpenses(context:Context, itemsDatabaseHelper:ItemsDatabaseHelper, expenseDatabaseHelper:ExpenseDatabaseHelper) {

Column(

modifier =Modifier

.padding(top =100.dp, start =30.dp)

.fillMaxHeight()

.fillMaxWidth(),

horizontalAlignment =Alignment.Start

) {

valmContext =LocalContext.current

varitems by remember { mutableStateOf("") }

varquantity by remember { mutableStateOf("") }

varcost by remember { mutableStateOf("") }

varerror 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))

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()) {

valitems =Items(

id =null,

itemName =items,

quantity =quantity,

cost =cost

)

vallimit=expenseDatabaseHelper.getExpenseAmount(1)

valactualvalue =limit?.minus(cost.toInt())

//Toast.makeText(mContext, actualvalue.toString(), Toast.LENGTH\_SHORT).show()

valexpense =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")

}

}

}

Step 5 : Creating SetLimitActivity.kt file

packagecom.example.expensestracker

importandroid.annotation.SuppressLint

importandroid.content.Context

importandroid.content.Intent

importandroid.os.Bundle

importandroid.util.Log

importandroidx.activity.ComponentActivity

importandroidx.activity.compose.setContent

importandroidx.compose.foundation.layout.\*

importandroidx.compose.foundation.lazy.LazyColumn

importandroidx.compose.foundation.lazy.LazyRow

importandroidx.compose.foundation.lazy.items

importandroidx.compose.material.\*

importandroidx.compose.runtime.\*

importandroidx.compose.ui.Alignment

importandroidx.compose.ui.Modifier

importandroidx.compose.ui.graphics.Color

importandroidx.compose.ui.text.font.FontWeight

importandroidx.compose.ui.text.style.TextAlign

importandroidx.compose.ui.unit.dp

importandroidx.compose.ui.unit.sp

importcom.example.expensestracker.ui.theme.ExpensesTrackerTheme

classSetLimitActivity: ComponentActivity() {

privatelateinitvarexpenseDatabaseHelper:ExpenseDatabaseHelper

@SuppressLint("UnusedMaterialScaffoldPaddingParameter")

overridefunonCreate(savedInstanceState:Bundle?) {

super.onCreate(savedInstanceState)

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 ={

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

)

)

},

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

)

)

},

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

)

}

}

)

}

) {

valdata=expenseDatabaseHelper.getAllExpense();

Log.d("swathi",data.toString())

valexpense =expenseDatabaseHelper.getAllExpense()

Limit(this, expenseDatabaseHelper,expense)

}

}

}

}

@Composable

funLimit(context:Context, expenseDatabaseHelper:ExpenseDatabaseHelper, expense:List<Expense>) {

Column(

modifier =Modifier

.padding(top =100.dp, start =30.dp)

.fillMaxHeight()

.fillMaxWidth(),

horizontalAlignment =Alignment.Start

) {

varamount by remember { mutableStateOf("") }

varerror by remember { mutableStateOf("") }

Text(text ="Monthly Amount Limit", fontWeight =FontWeight.Bold, fontSize =20.sp)

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

TextField(value =amount, onValueChange ={ amount =it },

label ={ Text(text ="Set Amount Limit ") })

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

if(error.isNotEmpty()) {

Text(

text =error,

color =MaterialTheme.colors.error,

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

)

}

Button(onClick ={

if(amount.isNotEmpty()) {

valexpense =Expense(

id =null,

amount =amount

)

expenseDatabaseHelper.insertExpense(expense)

}

}) {

Text(text ="Set Limit")

}

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

LazyRow(

modifier =Modifier

.fillMaxSize()

.padding(top =0.dp),

horizontalArrangement =Arrangement.Start

) {

item {

LazyColumn{

items(expense) { expense ->

Column(

) {

Text("Remaining Amount: ${expense.amount}", fontWeight =FontWeight.Bold)

}

}

}

}

}

}

}

//@Composable

//fun Records(expense: List<Expense>) {

//Text(text = "View Records", modifier = Modifier.padding(top = 24.dp, start = 106.dp, bottom = 24.dp ), fontSize = 30.sp)

//Spacer(modifier = Modifier.height(30.dp))

//LazyRow(

//modifier = Modifier

//.fillMaxSize()

//.padding(top = 80.dp),

//

//horizontalArrangement = Arrangement.SpaceBetween

//){

//item {

//

//LazyColumn {

//items(expense) { expense ->

//Column(modifier = Modifier.padding(top = 16.dp, start = 48.dp, bottom = 20.dp)) {

//Text("Remaining Amount: ${expense.amount}")

//}

//}

//}

//}

//

//}

//}

Step 6 : Creating ViewRecordsActivity.kt file

packagecom.example.expensestracker

importandroid.annotation.SuppressLint

importandroid.content.Intent

importandroid.os.Bundle

importandroid.util.Log

importandroidx.activity.ComponentActivity

importandroidx.activity.compose.setContent

importandroidx.compose.foundation.ScrollState

importandroidx.compose.foundation.layout.\*

importandroidx.compose.foundation.lazy.LazyColumn

importandroidx.compose.foundation.lazy.LazyRow

importandroidx.compose.foundation.lazy.items

importandroidx.compose.foundation.verticalScroll

importandroidx.compose.material.\*

importandroidx.compose.runtime.Composable

importandroidx.compose.ui.Modifier

importandroidx.compose.ui.graphics.Color

importandroidx.compose.ui.text.font.FontWeight

importandroidx.compose.ui.text.style.TextAlign

importandroidx.compose.ui.tooling.preview.Preview

importandroidx.compose.ui.unit.dp

importandroidx.compose.ui.unit.sp

importcom.example.expensestracker.ui.theme.ExpensesTrackerTheme

classViewRecordsActivity: ComponentActivity() {

privatelateinitvaritemsDatabaseHelper:ItemsDatabaseHelper

@SuppressLint("UnusedMaterialScaffoldPaddingParameter", "SuspiciousIndentation")

overridefunonCreate(savedInstanceState:Bundle?) {

super.onCreate(savedInstanceState)

itemsDatabaseHelper =ItemsDatabaseHelper(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 ={

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

)

)

},

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

)

)

},

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

)

}

}

)

}

) {

valdata=itemsDatabaseHelper.getAllItems();

Log.d("swathi",data.toString())

valitems =itemsDatabaseHelper.getAllItems()

Records(items)

}

}

}

}

@Composable

funRecords(items:List<Items>) {

Text(text ="View Records", modifier =Modifier.padding(top =24.dp, start =106.dp, bottom =24.dp ), fontSize =30.sp, fontWeight =FontWeight.Bold)

Spacer(modifier =Modifier.height(30.dp))

LazyRow(

modifier =Modifier

.fillMaxSize()

.padding(top =80.dp),

horizontalArrangement =Arrangement.SpaceBetween

){

item {

LazyColumn{

items(items) { items ->

Column(modifier =Modifier.padding(top =16.dp, start =48.dp, bottom =20.dp)) {

Text("Item\_Name: ${items.itemName}")

Text("Quantity: ${items.quantity}")

Text("Cost: ${items.cost}")

}

}

}

}

}

}

Modifying AndroidManifest.xml

<?xml version="1.0" encoding="utf-8"?>

<manifestxmlns: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.ExpensesTracker"

tools:targetApi="31">

<activity

android:name=".RegisterActivity"

android:exported="false"

android:label="@string/title\_activity\_register"

android:theme="@style/Theme.ExpensesTracker"/>

<activity

android:name=".MainActivity"

android:exported="false"

android:label="MainActivity"

android:theme="@style/Theme.ExpensesTracker"/>

<activity

android:name=".ViewRecordsActivity"

android:exported="false"

android:label="@string/title\_activity\_view\_records"

android:theme="@style/Theme.ExpensesTracker"/>

<activity

android:name=".SetLimitActivity"

android:exported="false"

android:label="@string/title\_activity\_set\_limit"

android:theme="@style/Theme.ExpensesTracker"/>

<activity

android:name=".AddExpensesActivity"

android:exported="false"

android:label="@string/title\_activity\_add\_expenses"

android:theme="@style/Theme.ExpensesTracker"/>

<activity

android:name=".LoginActivity"

android:exported="true"

android:label="@string/app\_name"

android:theme="@style/Theme.ExpensesTracker">

<intent-filter>

<actionandroid:name="android.intent.action.MAIN"/>

<categoryandroid:name="android.intent.category.LAUNCHER"/>

</intent-filter>

</activity>

</application>

</manifest>