Xml files

Login Activity:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/white"
    tools:context=".activities.LoginActivity">
    <!-- Header Part only the text view containing text login -->
    <RelativeLayout
        android:id="@+id/header_part_login"
        android:layout_alignParentTop="true"
        android:layout_width="match_parent"
        android:layout_height="@dimen/dp150">
        <TextView
            android:layout_centerInParent="true"
            android:fontFamily="monospace"
            android:textColor="@color/colorPrimary"
            android:text="@string/login"
            android:textSize="48dp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />

    </RelativeLayout>
    <!-- Login Credentials -->
    <LinearLayout
        android:layout_below="@id/header_part_login"
        android:layout_margin="@dimen/dp10"
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
        <LinearLayout
            android:orientation="vertical"
            android:layout_width="match_parent"
            android:layout_height="wrap_content">
            <TextView
                android:text="Email"
                android:textSize="@dimen/sp15"
                android:textColor="@color/text_color_dark"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content" />
            <EditText
                android:textColor="@color/text_color_dark"
                android:backgroundTint="@color/text_color_dark"
```

```
        android:inputType="textEmailAddress"
        android:id="@+id/user_email_edt_login"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    </LinearLayout>
    <LinearLayout
      android:orientation="vertical"
      android:layout_width="match_parent"
      android:layout_height="wrap_content">
      <TextView
        android:text="Username"
        android:textSize="@dimen/sp15"
        android:textColor="@color/text_color_dark"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
      <EditText
        android:textColor="@color/text_color_dark"
        android:backgroundTint="@color/text_color_dark"
        android:inputType="textEmailAddress"
        android:id="@+id/user_name_edt_login"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    </LinearLayout>
    <LinearLayout
      android:orientation="vertical"
      android:layout_width="match_parent"
      android:layout_height="wrap_content">
      <TextView
        android:text="Password"
        android:textSize="@dimen/sp15"
        android:textColor="@color/text_color_dark"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
      <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/etPasswordLayout"
        android:backgroundTint="@color/text_color_dark"
        app:passwordToggleTint="@color/text_color_dark"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:passwordToggleEnabled="true">

        <com.google.android.material.textfield.TextInputEditText
          android:id="@+id/user_password_edt_login"
          android:backgroundTint="@color/text_color_dark"
          android:drawableTint="@color/colorPrimaryDark"
          android:textColor="@color/text_color_dark"
          android:layout_width="match_parent"
          android:layout_height="wrap_content"
          android:inputType="textPassword"/>
      </com.google.android.material.textfield.TextInputLayout>
```

```
    </LinearLayout>
    <Button
      android:id="@+id/sign_in_login"
      android:layout_marginTop="@dimen/dp10"
      style="@style/Widget.AppCompat.Button.Borderless"
      android:background="@color/text_color_dark"
      android:text="@string/sign_in"
      android:onClick="signin"
      android:textColor="@color/white"
      android:textSize="@dimen/sp18"
      android:layout_width="match_parent"
      android:layout_height="wrap_content" />
    <Button
      android:id="@+id/sign_up_login"
      android:layout_marginTop="@dimen/dp10"
      android:onClick="signUp"
      style="@style/Widget.AppCompat.Button.Borderless"
      android:background="@color/text_color_dark"
      android:text="@string/sign_up"
      android:textColor="@color/white"
      android:textSize="@dimen/sp18"
      android:layout_width="match_parent"
      android:layout_height="wrap_content" />
    <TextView
      android:id="@+id/forgot_passwordTextView"
      android:padding="@dimen/dp8"
      android:text="Forgot Password?"
      android:textSize="@dimen/sp15"
      android:layout_gravity="end"
      android:textColor="@android:color/holo_red_dark"
      android:gravity="end|center_vertical"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"/>
  </LinearLayout>


</RelativeLayout>
```

Register Activity:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".activities.RegisterActivity">

  <LinearLayout
    android:layout_width="match_parent"
```

```
android:layout_height="wrap_content"
android:layout_margin="@dimen/dp10"
android:orientation="vertical">
<!-- User name -->
<LinearLayout
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:orientation="vertical">

   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@string/user_name"
      style="@style/TextAppearance.AppCompat.Title"
      android:textColor="@color/text_color_dark"
      android:textSize="@dimen/sp12" />

   <EditText
      android:id="@+id/user__name_edt_register"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:backgroundTint="@color/text_color_dark"
      android:inputType="text" />

</LinearLayout>
<!-- Full name Layout-->
<LinearLayout
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:orientation="vertical">

   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@string/full_name"
      style="@style/TextAppearance.AppCompat.Title"
      android:textColor="@color/text_color_dark"
      android:textSize="@dimen/sp12" />

   <EditText
      android:id="@+id/user_full_name_edt_register"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:backgroundTint="@color/text_color_dark"
      android:inputType="text" />

</LinearLayout>
<!-- Phone Number Layout -->
<LinearLayout
   android:layout_width="match_parent"
```

```
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/phone_no"
        android:textColor="@color/text_color_dark"
        android:textSize="@dimen/sp12" />

    <EditText
        android:id="@+id/user_phone_no_edt_register"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:backgroundTint="@color/text_color_dark"
        android:inputType="number" />

</LinearLayout>
<!-- Email Layout -->
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Email"
        android:textColor="@color/text_color_dark"
        android:textSize="@dimen/sp12" />

    <EditText
        android:id="@+id/user_email_edt_register"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:backgroundTint="@color/text_color_dark"
        android:inputType="textEmailAddress" />

</LinearLayout>
<!-- Password Layout-->
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/password"
        android:textColor="@color/text_color_dark"
```

```
        android:textSize="@dimen/sp12" />

    <EditText
        android:id="@+id/user_password_edt_register"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Min 8 Characters"
        android:backgroundTint="@color/text_color_dark"
        android:inputType="textWebPassword" />

</LinearLayout>
<!-- Confirm Password -->
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/confirm_password"
        android:textColor="@color/text_color_dark"
        android:textSize="@dimen/sp12" />

    <EditText
        android:id="@+id/user_confirm_password_edt_register"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Min 8 Characters"
        android:backgroundTint="@color/text_color_dark"
        android:inputType="textWebPassword" />

</LinearLayout>
<!-- Secret Key -->
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Secret Key "
        android:textColor="@color/text_color_dark"
        android:textSize="@dimen/sp12" />

    <EditText
        android:id="@+id/user_secret_password_edt_register"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
```

```
            android:hint="Min 16 Characters"
            android:backgroundTint="@color/text_color_dark"
            android:inputType="text" />

    </LinearLayout>
    <!-- Submit Button -->
    <Button
        android:id="@+id/submit_button"
        style="@style/Widget.AppCompat.Button.Borderless"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:clickable="true"
        android:onClick="register"
        android:layout_marginTop="@dimen/dp20"
        android:background="@color/text_color_dark"
        android:text="submit"
        android:textColor="@color/white"
        android:textSize="@dimen/sp18" />
  </LinearLayout>

</RelativeLayout>
```

Home Activity:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/white"
    android:orientation="vertical"
    android:weightSum="3"
    tools:context=".activities.HomeActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:orientation="horizontal"
        android:weightSum="2">
        <!-- Contacts -->
        <RelativeLayout
            android:id="@+id/contacts_rv"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_margin="2dp"
            android:layout_weight="1"
            android:background="@color/colorPrimary"
```

```
    android:onClick="navigate"
    tools:ignore="NestedWeights">

    <ImageView
        android:id="@+id/contacts_imageView"
        android:layout_width="@dimen/dp100"
        android:layout_height="@dimen/dp100"
        android:layout_centerInParent="true"
        android:background="@color/colorPrimary"
        android:onClick="navigate"
        android:src="@drawable/ic_contact_phone_black_24dp"
        android:tint="@color/white" />

    <TextView
        android:id="@+id/contacts_textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/contacts_imageView"
        android:layout_centerHorizontal="true"
        android:onClick="navigate"
        android:text="@string/contacts"
        android:textColor="@color/white"
        android:textSize="@dimen/sp18" />
</RelativeLayout>
<!-- Messages -->
<RelativeLayout
    android:id="@+id/messages_rv"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="2dp"
    android:layout_weight="1"
    android:background="@color/colorPrimary"
    android:onClick="navigate">

    <ImageView
        android:id="@+id/messages_imageView"
        android:layout_width="@dimen/dp100"
        android:layout_height="@dimen/dp100"
        android:layout_centerInParent="true"
        android:onClick="navigate"
        android:src="@drawable/messages"
        android:tint="@color/white" />

    <TextView
        android:id="@+id/messages_textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/messages_imageView"
        android:layout_centerHorizontal="true"
        android:onClick="navigate"
```

```
            android:text="@string/messages"
            android:textColor="@color/white"
            android:textSize="@dimen/sp18" />
    </RelativeLayout>


</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:weightSum="2">
    <!-- Notes -->
    <RelativeLayout
        android:id="@+id/notes_rv"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_margin="2dp"
        android:layout_weight="1"
        android:background="@color/colorPrimary"
        android:onClick="navigate">

        <ImageView
            android:id="@+id/notes-imageView"
            android:layout_width="@dimen/dp100"
            android:layout_height="@dimen/dp100"
            android:layout_centerInParent="true"
            android:background="@color/colorPrimary"
            android:onClick="navigate"
            android:src="@drawable/notes"
            android:tint="@color/white" />

        <TextView
            android:id="@+id/notes_tv"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@id/notes-imageView"
            android:layout_centerHorizontal="true"
            android:onClick="navigate"
            android:text="@string/notes"
            android:textColor="@color/white"
            android:textSize="@dimen/sp18" />
    </RelativeLayout>
    <!-- Files -->
    <RelativeLayout
        android:id="@+id/files_rv"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_margin="2dp"
```

```
        android:layout_weight="1"
        android:background="@color/colorPrimary"
        android:onClick="navigate">

        <ImageView
            android:id="@+id/files_imageView"
            android:layout_width="@dimen/dp100"
            android:layout_height="@dimen/dp100"
            android:layout_centerInParent="true"
            android:onClick="navigate"
            android:src="@drawable/ic_folder_white_24dp"
            android:tint="@color/white" />

        <TextView
            android:id="@+id/files_textView"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@id/files_imageView"
            android:layout_centerHorizontal="true"
            android:onClick="navigate"
            android:text="@string/files"
            android:textColor="@color/white"
            android:textSize="@dimen/sp18" />
    </RelativeLayout>

</LinearLayout>


<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:weightSum="2">
    <!-- Settings -->
    <RelativeLayout
        android:id="@+id/settings_rv"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_margin="2dp"
        android:layout_weight="1"
        android:background="@color/colorPrimary"
        android:onClick="navigate">

        <ImageView
            android:id="@+id/settings_imageView"
            android:layout_width="@dimen/dp100"
            android:layout_height="@dimen/dp100"
            android:layout_centerInParent="true"
            android:background="@color/colorPrimary"
            android:onClick="navigate"
```

```
            android:src="@drawable/ic_settings_white_24dp"
            android:tint="@color/white" />

        <TextView
            android:id="@+id/settings_textView"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@id/settings_imageView"
            android:layout_centerHorizontal="true"
            android:onClick="navigate"
            android:text="@string/settings"
            android:textColor="@color/white"
            android:textSize="@dimen/sp18" />
    </RelativeLayout>

    <RelativeLayout
        android:id="@+id/logout_rv"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_margin="2dp"
        android:layout_weight="1"
        android:background="@color/colorPrimary"
        android:onClick="navigate">

        <ImageView
            android:id="@+id/logout_imageView"
            android:layout_width="@dimen/dp100"
            android:layout_height="@dimen/dp100"
            android:layout_centerInParent="true"
            android:onClick="navigate"
            android:src="@drawable/logout"
            android:tint="@color/white" />

        <TextView
            android:id="@+id/logout_textView"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@id/logout_imageView"
            android:layout_centerHorizontal="true"
            android:onClick="navigate"
            android:text="@string/logout"
            android:textColor="@color/white"
            android:textSize="@dimen/sp18" />
    </RelativeLayout>

  </LinearLayout>

</LinearLayout>
```

Contacts activity:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".activities.subactivities.contacts.ContactsActivitiy">
    <ProgressBar
        android:layout_centerInParent="true"
        android:id="@+id/contacts_progress_bar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <androidx.recyclerview.widget.RecyclerView
        tools:listitem="@layout/contacts_item"
        app:layoutManager="androidx.recyclerview.widget.LinearLayoutManager"
        android:layout_alignParentTop="true"
        android:id="@+id/contacts_RecyclerView"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

    </androidx.recyclerview.widget.RecyclerView>
    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:layout_alignParentBottom="true"
        android:layout_alignParentEnd="true"
        android:layout_margin="@dimen/dp10"
        android:id="@+id/add_contact"
        android:onClick="pickaContact"
        android:scaleType="centerCrop"
        android:src="@drawable/ic_person_add_black_24dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```

Contacts Item:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="@dimen/dp5"
    android:background="@drawable/item_back"
    android:elevation="@dimen/dp10"
    android:padding="@dimen/dp5"
```

```
    android:orientation="vertical">


    <TextView
        android:paddingBottom="@dimen/dp5"
        android:paddingEnd="@dimen/dp5"
        android:paddingTop="@dimen/dp5"
        android:id="@+id/contact_name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:gravity="bottom"
        android:text="UnKnown"
        android:paddingStart="@dimen/dp5"
        android:textSize="@dimen/sp15"
        android:textColor="@color/text_color_dark" />

    <TextView
        android:layout_below="@id/contact_name"
        android:paddingTop="@dimen/dp5"
        android:paddingBottom="@dimen/dp5"
        android:paddingEnd="@dimen/dp5"
        android:paddingStart="@dimen/dp10"
        android:id="@+id/contact_number"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:textSize="@dimen/sp12"
        android:gravity="center_vertical"
        android:text="0123456789"
        android:textColor="@color/text_color_medium" />

    <ImageButton
        android:background="@color/white"
        android:layout_centerInParent="true"
        android:layout_alignParentEnd="true"
        android:src="@drawable/ic_delete_black_24dp"
        android:id="@+id/contact_delete_imageView"
        android:layout_width="@dimen/dp40"
        android:layout_height="@dimen/dp40"/>


</RelativeLayout>
```

Messages Activity:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
```

```
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".activities.subactivities.messages.MessagesActivity">
    <ScrollView

        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <TextView
            android:id="@+id/target"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />
    </ScrollView>
    <ProgressBar
        android:visibility="gone"
        android:layout_centerInParent="true"
        android:id="@+id/messages_progress_bar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <androidx.recyclerview.widget.RecyclerView
        tools:listitem="@layout/message_item"
        app:layoutManager="androidx.recyclerview.widget.LinearLayoutManager"
        android:layout_alignParentTop="true"
        android:id="@+id/messages_RecyclerView"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

    </androidx.recyclerview.widget.RecyclerView>
    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:layout_alignParentBottom="true"
        android:layout_alignParentEnd="true"
        android:layout_margin="@dimen/dp10"
        android:id="@+id/add_message"
        android:onClick="pickaMessage"
        android:scaleType="centerCrop"
        android:src="@drawable/ic_add_black_24dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```

Messages sub activity:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
```

```
    tools:context=".activities.subactivities.messages.MessageSubActivity">
    <ProgressBar
        android:visibility="gone"
        android:layout_centerInParent="true"
        android:id="@+id/messages_sub_progress_bar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <androidx.recyclerview.widget.RecyclerView
        tools:listitem="@layout/contacts_item"
        app:layoutManager="androidx.recyclerview.widget.LinearLayoutManager"
        android:layout_alignParentTop="true"
        android:id="@+id/messages_sub_RecyclerView"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

    </androidx.recyclerview.widget.RecyclerView>
</RelativeLayout>
```

Messages Item:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_margin="@dimen/dp5"
    android:padding="@dimen/dp10"
    android:background="@drawable/message_back"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <TextView
        android:id="@+id/message_title"
        android:textColor="@color/text_color_dark"
        android:textSize="@dimen/sp15"
        android:paddingStart="@dimen/dp5"
        android:text="@string/app_name"
        android:textStyle="bold"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <TextView
        android:id="@+id/message_body"
        android:maxLines="2"
        android:paddingStart="@dimen/dp10"
        android:textSize="@dimen/sp12"
        android:textColor="@color/text_color_light"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

</LinearLayout>
```

Notes Activity:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".activities.subactivities.notes.NotesActivity">
    <ProgressBar
        android:id="@+id/notes_progressBar"
        android:layout_centerInParent="true"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/notes_RecyclerView"
        tools:listitem="@layout/notes_item"
        app:layoutManager="androidx.recyclerview.widget.LinearLayoutManager"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

    </androidx.recyclerview.widget.RecyclerView>
<com.google.android.material.floatingactionbutton.FloatingActionButton
    android:layout_width="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_alignParentEnd="true"
    android:layout_margin="@dimen/dp10"
    android:onClick="addNotes"
    android:src="@drawable/ic_add_black_24dp"
    android:layout_height="wrap_content" />
</RelativeLayout>
```

Notes Item:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:background="@drawable/notes_back"
    android:layout_margin="@dimen/dp5"
    android:padding="@dimen/dp10"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <TextView
        android:id="@+id/notes_title_item_tv"
        android:text="title"
        style="@style/TextAppearance.AppCompat.Title"
        android:textSize="@dimen/sp18"
        android:textColor="#000000"
        android:paddingStart="@dimen/dp5"
        android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content" />
    <TextView
        android:id="@+id/message_body_item_tv"
        android:maxLines="2"
        android:paddingStart="@dimen/dp5"
        android:paddingEnd="@dimen/dp5"
        android:text="this is the body of the notes"
        android:textSize="@dimen/sp15"
        android:textColor="@color/text_color_medium"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

</LinearLayout>
```

Settings Activity:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".activities.subactivities.SettingsActivity">
    <LinearLayout
        android:orientation="vertical"
        android:layout_margin="@dimen/dp5"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
        <RelativeLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content">
            <TextView
                android:layout_alignParentStart="true"
                android:text="Contacts"
                android:textSize="@dimen/sp18"
                android:paddingStart="@dimen/dp10"
                android:textStyle="bold"
                android:id="@+id/contacts_setting_textView_title"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content" />
            <CheckBox
                android:shadowColor="#000080"
                android:id="@+id/contacts_switch"
                android:buttonTint="#000080"
                android:backgroundTint="#008000"
                android:layout_alignParentEnd="true"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content" />
```

```
    </RelativeLayout>
    <Button
        android:visibility="gone"
        android:onClick="setPasswordForContacts"
        android:id="@+id/contacts_password_btn_settings"
        android:text="@string/change_password"
        android:layout_marginStart="@dimen/dp10"
        style="@style/Widget.AppCompat.Button.Borderless"
        android:textColor="#000080"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
</LinearLayout>
<LinearLayout
    android:orientation="vertical"
    android:layout_margin="@dimen/dp5"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
        <TextView
            android:layout_alignParentStart="true"
            android:text="Messages"
            android:textSize="@dimen/sp18"
            android:paddingStart="@dimen/dp10"
            android:textStyle="bold"
            android:id="@+id/messages_setting_textView_title"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
        <CheckBox
            android:id="@+id/messages_switch"
            android:buttonTint="#000080"
            android:backgroundTint="#008000"
            android:layout_alignParentEnd="true"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
    </RelativeLayout>
    <Button
        android:visibility="gone"
        android:onClick="setPasswordForMessages"
        android:id="@+id/messages_password_btn_settings"
        android:text="@string/change_password"
        android:layout_marginStart="@dimen/dp10"
        style="@style/Widget.AppCompat.Button.Borderless"
        android:textColor="#000080"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
</LinearLayout>
<LinearLayout
    android:orientation="vertical"
```

```
    android:layout_margin="@dimen/dp5"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <RelativeLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content">
      <TextView
        android:layout_alignParentStart="true"
        android:text="Notes"
        android:textSize="@dimen/sp18"
        android:paddingStart="@dimen/dp10"
        android:textStyle="bold"
        android:id="@+id/notes_setting_textView_title"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
      <CheckBox
        android:id="@+id/notes_switch"
        android:buttonTint="#000080"
        android:backgroundTint="#008000"
        android:layout_alignParentEnd="true"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
    </RelativeLayout>
    <Button
      android:visibility="gone"
      android:onClick="setPasswordForNotes"
      android:id="@+id/notes_password_btn_settings"
      android:text="@string/change_password"
      android:layout_marginStart="@dimen/dp10"
      style="@style/Widget.AppCompat.Button.Borderless"
      android:textColor="#000080"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content" />
  </LinearLayout>
  <LinearLayout
    android:orientation="vertical"
    android:layout_margin="@dimen/dp5"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <RelativeLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content">
      <TextView
        android:layout_alignParentStart="true"
        android:text="Files"
        android:textSize="@dimen/sp18"
        android:paddingStart="@dimen/dp10"
        android:textStyle="bold"
        android:id="@+id/files_setting_textView_title"
        android:layout_width="wrap_content"
```

```
            android:layout_height="wrap_content" />
        <CheckBox
            android:id="@+id/files_switch"
            android:buttonTint="#000080"
            android:backgroundTint="#008000"
            android:layout_alignParentEnd="true"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
    </RelativeLayout>
    <Button
        android:visibility="gone"
        android:onClick="setPasswordForFiles"
        android:id="@+id/files_password_btn_settings"
        android:text="@string/change_password"
        android:layout_marginStart="@dimen/dp10"
        style="@style/Widget.AppCompat.Button.Borderless"
        android:textColor="#000080"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
  </LinearLayout>

</LinearLayout>
```

Files Activity:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".activities.FilesEncDecActivity">

  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:gravity="center">

    <LinearLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_margin="@dimen/dp2"
      android:orientation="horizontal">
      <RelativeLayout
        android:onClick="openImages"
        android:id="@+id/images_folder_rv"
        android:layout_weight="1"
        android:gravity="center"
```

```
        android:layout_width="@dimen/dp150"
        android:layout_height="@dimen/dp150">
        <ImageView
            android:onClick="openImages"
            android:layout_centerInParent="true"
            android:id="@+id/images_folder_imageView"
            android:src="@drawable/ic_folder_white_24dp"
            android:tint="@color/text_color_dark"
            android:layout_width="@dimen/dp100"
            android:layout_height="@dimen/dp100" />
        <TextView
            android:onClick="openImages"
            android:id="@+id/images_folder_textView"
            android:layout_centerHorizontal="true"
            android:text="@string/images"
            android:textSize="@dimen/sp15"
            android:textColor="@color/text_color_dark"
            android:layout_below="@id/images_folder_imageView"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
    </RelativeLayout>
    <RelativeLayout
        android:onClick="openDocs"
        android:id="@+id/docs_folder_rv"
        android:layout_weight="1"
        android:gravity="center"
        android:layout_width="@dimen/dp150"
        android:layout_height="@dimen/dp150">
        <ImageView
            android:onClick="openDocs"
            android:layout_centerInParent="true"
            android:id="@+id/docs_folder_imageView"
            android:src="@drawable/ic_folder_white_24dp"
            android:tint="@color/text_color_dark"
            android:layout_width="@dimen/dp100"
            android:layout_height="@dimen/dp100" />
        <TextView
            android:onClick="openDocs"
            android:id="@+id/docs_folder_textView"
            android:layout_centerHorizontal="true"
            android:text="@string/documents"
            android:textSize="@dimen/sp15"
            android:textColor="@color/text_color_dark"
            android:layout_below="@id/docs_folder_imageView"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
    </RelativeLayout>
</LinearLayout>
<LinearLayout
    android:layout_width="match_parent"
```

```
android:layout_height="wrap_content"
android:layout_margin="@dimen/dp2"
android:orientation="horizontal">
<RelativeLayout
   android:onClick="openAudio"
   android:id="@+id/audios_folder_rv"
   android:layout_weight="1"
   android:gravity="center"
   android:layout_width="@dimen/dp150"
   android:layout_height="@dimen/dp150">
   <ImageView
      android:onClick="openAudio"
      android:layout_centerInParent="true"
      android:id="@+id/audios_folder_imageView"
      android:src="@drawable/ic_folder_white_24dp"
      android:tint="@color/text_color_dark"
      android:layout_width="@dimen/dp100"
      android:layout_height="@dimen/dp100" />
   <TextView
      android:onClick="openAudio"
      android:id="@+id/audios_folder_textView"
      android:layout_centerHorizontal="true"
      android:text="@string/audios"
      android:textSize="@dimen/sp15"
      android:textColor="@color/text_color_dark"
      android:layout_below="@id/audios_folder_imageView"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content" />
</RelativeLayout>
<RelativeLayout
   android:onClick="openVideos"
   android:id="@+id/videos_folder_rv"
   android:layout_weight="1"
   android:gravity="center"
   android:layout_width="@dimen/dp150"
   android:layout_height="@dimen/dp150">
   <ImageView
      android:onClick="openVideos"
      android:layout_centerInParent="true"
      android:id="@+id/videos_folder_imageView"
      android:src="@drawable/ic_folder_white_24dp"
      android:tint="@color/text_color_dark"
      android:layout_width="@dimen/dp100"
      android:layout_height="@dimen/dp100" />
   <TextView
      android:onClick="openVideos"
      android:id="@+id/videos_folder_textView"
      android:layout_centerHorizontal="true"
      android:text="@string/videos"
      android:textSize="@dimen/sp15"
```

```
            android:textColor="@color/text_color_dark"
            android:layout_below="@id/videos_folder_imageView"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
        </RelativeLayout>
    </LinearLayout>
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="@dimen/dp2"
        android:orientation="horizontal">
        <RelativeLayout
            android:onClick="openZip"
            android:id="@+id/zip_folder_rv"
            android:layout_weight="1"
            android:gravity="center"
            android:layout_width="@dimen/dp150"
            android:layout_height="@dimen/dp150">
            <ImageView
                android:onClick="openZip"
                android:layout_centerInParent="true"
                android:id="@+id/zip_folder_imageView"
                android:src="@drawable/ic_folder_white_24dp"
                android:tint="@color/text_color_dark"
                android:layout_width="@dimen/dp100"
                android:layout_height="@dimen/dp100" />
            <TextView
                android:onClick="openZip"
                android:id="@+id/zip_folder_textView"
                android:layout_centerHorizontal="true"
                android:text="@string/zip_files"
                android:textSize="@dimen/sp15"
                android:textColor="@color/text_color_dark"
                android:layout_below="@id/zip_folder_imageView"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content" />
        </RelativeLayout>

    </LinearLayout>
</LinearLayout>

</RelativeLayout>
```

Images Activity:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
```

```
    android:layout_height="match_parent"
    tools:context=".activities.ImagesActivity">

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/decryptedImage"
        />

</RelativeLayout>
```

Images files:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    tools:context=".activities.subactivities.files.ImageFilesActivity">
    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">
    <ImageView
        android:padding="@dimen/dp5"
        android:layout_margin="@dimen/dp5"
        android:id="@+id/decryptedImage"
        android:src="@drawable/default_image"
        android:layout_width="match_parent"
        android:layout_height="300dp"/>

    <androidx.recyclerview.widget.RecyclerView
        android:layout_below="@id/decryptedImage"
        android:id="@+id/imageFilesRecyclerView"
        app:layoutManager="androidx.recyclerview.widget.GridLayoutManager"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

    </androidx.recyclerview.widget.RecyclerView>

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/addAnImage"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:elevation="@dimen/dp8"
        android:layout_margin="@dimen/dp8"
        android:src="@drawable/ic_add_black_24dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
```

```
    </RelativeLayout>

</ScrollView>
```

Video Activity:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".activities.subactivities.files.VideoFilesActivity">
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/videoFilesRecyclerView"
        app:layoutManager="androidx.recyclerview.widget.GridLayoutManager"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

    </androidx.recyclerview.widget.RecyclerView>

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/addVideo"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:elevation="@dimen/dp8"
        android:layout_margin="@dimen/dp8"
        android:src="@drawable/ic_add_black_24dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
</RelativeLayout>
```

Documents Activity:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".activities.subactivities.files.DocumentsActivity">
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/docFilesRecyclerView"
        app:layoutManager="androidx.recyclerview.widget.GridLayoutManager"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

    </androidx.recyclerview.widget.RecyclerView>
```

```
    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/addDoc"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:elevation="@dimen/dp8"
        android:layout_margin="@dimen/dp8"
        android:src="@drawable/ic_add_black_24dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>

</RelativeLayout>
```

Audio Activity:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".activities.subactivities.files.AudioFilesActivity">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/audioFilesRecyclerView"
        app:layoutManager="androidx.recyclerview.widget.GridLayoutManager"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

    </androidx.recyclerview.widget.RecyclerView>

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/addAnAudio"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:elevation="@dimen/dp8"
        android:layout_margin="@dimen/dp8"
        android:src="@drawable/ic_add_black_24dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>

</RelativeLayout>
```

Zip files Activity:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
```

```
    android:layout_height="match_parent"
    tools:context=".activities.subactivities.files.ZipFilesActivity">
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/zipFilesRecyclerView"
        app:layoutManager="androidx.recyclerview.widget.GridLayoutManager"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

    </androidx.recyclerview.widget.RecyclerView>

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/addZipFile"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:elevation="@dimen/dp8"
        android:layout_margin="@dimen/dp8"
        android:src="@drawable/ic_add_black_24dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>

</RelativeLayout>
```

Enter password Activity:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:background="@drawable/enterpassword_back"
    android:layout_width="match_parent"
    android:padding="@dimen/dp10"
    android:layout_height="wrap_content">
    <TextView
        android:paddingStart="@dimen/dp5"
        android:text="Enter the Code"
        android:textSize="@dimen/sp18"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>
    <EditText
        android:id="@+id/password_edit"
        android:hint="Security Code"
        android:paddingStart="@dimen/dp5"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>

</LinearLayout>
```

Change Password:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <EditText
      android:id="@+id/old_password_edt"
      android:hint="@string/old_password"
      android:paddingStart="@dimen/dp10"
      android:layout_width="match_parent"
      android:layout_height="wrap_content" />
   <EditText
      android:id="@+id/new_password"
      android:hint="@string/new_password"
      android:paddingStart="@dimen/dp10"
      android:layout_width="match_parent"
      android:layout_height="wrap_content" />
   <EditText
      android:id="@+id/confirm_password"
      android:hint="@string/confirm_password"
      android:paddingStart="@dimen/dp10"
      android:layout_width="match_parent"
      android:layout_height="wrap_content" />

</LinearLayout>
```

JAVA FILES

Login Activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.ViewModelProvider;

import android.Manifest;
import android.annotation.TargetApi;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.net.ConnectivityManager;
import android.os.Build;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
```

```
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.android.material.snackbar.Snackbar;
import com.google.firebase.FirebaseApp;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;


import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.SettingsActivity;
import www.srit.ac.in.project.mobile.encryption.self.models.User;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;

import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.SECRETKEY;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_EMAIL;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_FULLNAME;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_NAME;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PASS;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PHONE;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;
import static www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.target;

public class LoginActivity extends AppCompatActivity {
    private Button signUp, signIn;
    private EditText userEmail,userName;
    private EditText userPassword;
    FirebaseAuth mAuth;
    MSFViewModel viewModel;
    private SharedPreferences preferences;
    private DatabaseReference reference;
    private ValueEventListener valueEventListener;
    private TextView forgotPasswordTextView;

    String[] permissions = {
            Manifest.permission.READ_CONTACTS,
```

```
            Manifest.permission.WRITE_CONTACTS,
            Manifest.permission.READ_EXTERNAL_STORAGE,
            Manifest.permission.WRITE_EXTERNAL_STORAGE,
            Manifest.permission.READ_SMS,
            Manifest.permission.WRITE_CONTACTS
    };


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        //Initialization of Views
        signIn = findViewById(R.id.sign_in_login);
        signUp = findViewById(R.id.sign_up_login);
        userEmail = findViewById(R.id.user_email_edt_login);
        userPassword = findViewById(R.id.user_password_edt_login);
        forgotPasswordTextView=findViewById(R.id.forgot_passwordTextView);
        userName=findViewById(R.id.user_name_edt_login);
        viewModel = new ViewModelProvider(this).get(MSFViewModel.class);
        FirebaseApp.initializeApp(this);
        mAuth = FirebaseAuth.getInstance();
        checkPermissions();
        // Intent that takes to you to register Activity
        preferences = LoginActivity.this.getSharedPreferences(USER_PREF,
Context.MODE_PRIVATE);
        String gmail = preferences.getString(USER_EMAIL, null);
        String uName=preferences.getString(USER_NAME,null);
        if (gmail!=null) {
            userEmail.setText(gmail);
            userName.setText(uName);
        }
        forgotPasswordTextView.setOnClickListener(view -> {
            AlertDialog.Builder builder=new AlertDialog.Builder(this);
            final EditText editText=new EditText(LoginActivity.this);
            LinearLayout.LayoutParams lp = new LinearLayout.LayoutParams(
                    LinearLayout.LayoutParams.MATCH_PARENT,
                    LinearLayout.LayoutParams.MATCH_PARENT);
            builder.setTitle("Enter Email id");
            editText.setLayoutParams(lp);
            builder.setView(editText);
            builder.setPositiveButton("SUBMIT", (dialogInterface, i) -> {
                String password=editText.getText().toString().trim();
                if (TextUtils.isEmpty(password)){
                    Toast.makeText(LoginActivity.this, "Please Enter Email",
Toast.LENGTH_SHORT).show();
                }else{
                  mAuth.sendPasswordResetEmail(password).addOnCompleteListener(this, new
OnCompleteListener<Void>() {
                    @Override
```

```
                public void onComplete(@NonNull Task<Void> task) {
                    dialogInterface.dismiss();
                    Toast.makeText(LoginActivity.this, "Please Check your Email",
Toast.LENGTH_SHORT).show();
                }
            });
        }
    }).create().show();

  });
}


@TargetApi(Build.VERSION_CODES.M)
private void checkPermissions() {
  requestPermissions(permissions, 10);
}

private void getUserData(){
  valueEventListener=new ValueEventListener() {
    @Override
    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
      User user=dataSnapshot.getValue(User.class);
      saveUserData(user);
    }

    @Override
    public void onCancelled(@NonNull DatabaseError databaseError) {

    }
  };
  reference.addValueEventListener(valueEventListener);

}
public void saveUserData(User user){
  if (preferences.getString(USER_PASS,null)==null){
    viewModel.insertSetting(new
www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings(0, "CONTACTS",
user.getUserPassword(), false));
    viewModel.insertSetting(new
www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings(1, "MESSAGES",
user.getUserPassword(), false));
    viewModel.insertSetting(new
www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings(2, "NOTES",
user.getUserPassword(), false));
    viewModel.insertSetting(new
www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings(3, "FILES",
user.getUserPassword(), false));

  }
```

```
        SharedPreferences.Editor myEditor=preferences.edit();
        myEditor.putString(USER_EMAIL, user.getUserEmail());
        myEditor.putString(USER_PASS, user.getUserPassword());
        myEditor.putString(USER_FULLNAME,user.getUserFullName());
        myEditor.putString(USER_NAME,user.getUserName());
        myEditor.putString(USER_PHONE, user.getUserPhone());
        myEditor.putString(SECRETKEY,user.getUserSecretKey());
        myEditor.apply();

         }
    public void signin(View view) {
        String usernameStr=userName.getText().toString().trim();
        String email = userEmail.getText().toString();
        String enterpass = userPassword.getText().toString();
        reference=
FirebaseDatabase.getInstance().getReference().child(target).child(usernameStr.toUpperCase(
));
        if (TextUtils.isEmpty(email)){
            Toast.makeText(this, "Please Enter Valid Email",
Toast.LENGTH_SHORT).show();
        }else if (TextUtils.isEmpty(usernameStr)){
            Toast.makeText(this, "Please Enter Valid Username",
Toast.LENGTH_SHORT).show();
        }else if (TextUtils.isEmpty(enterpass)){
            Toast.makeText(this, "Please Enter Password", Toast.LENGTH_SHORT).show();
        }else if (isNetworkConnected()) {
            ProgressDialog dialog = ProgressDialog.show(this, "Authentication", "Please Wait
we are Processing your Request");
            mAuth.signInWithEmailAndPassword(email,
enterpass).addOnCompleteListener(this, task -> {
                if (task.isSuccessful()) {
                    reference.child("userPassword").setValue(enterpass);
                    reference=
FirebaseDatabase.getInstance().getReference().child(target).child(usernameStr.toUpperCase(
));
                    getUserData();
                    dialog.dismiss();
                    startActivity(new Intent(LoginActivity.this, HomeActivity.class));
                    Toast.makeText(this, "Logged In Successful",
Toast.LENGTH_SHORT).show();
                    finish();
                } else {
                    dialog.dismiss();
                    Toast.makeText(this, "LogIn UnSuccessful",
Toast.LENGTH_SHORT).show();
                }
            }).addOnFailureListener(this, e -> {
                dialog.dismiss();
                Snackbar.make(signIn, "Password didn't match",
Snackbar.LENGTH_SHORT).show();
```

```
            });
        } else {
            noInternetAlert();
        }
    }

    private void noInternetAlert() {
        AlertDialog.Builder alertBuilder = new AlertDialog.Builder(this);
        alertBuilder.setTitle("No Connection")
                .setMessage("Please Check your Internet Connection and try again")
                .setPositiveButton("OK", null)
                .setNegativeButton("Close", null).create().show();
    }

    @Override
    protected void onRestart() {
        super.onRestart();
        preferences = LoginActivity.this.getSharedPreferences(USER_PREF,
Context.MODE_PRIVATE);
        String gmail = preferences.getString(USER_EMAIL, null);
        String username=preferences.getString(USER_NAME,null);
        if (gmail!=null) {
            userEmail.setText(gmail);
            userName.setText(username);
        }
    }

    public void signUp(View view) {
        // Intent that takes to you to register Activity
        SharedPreferences preferences =
LoginActivity.this.getSharedPreferences(USER_PREF, Context.MODE_PRIVATE);
        String password = preferences.getString(USER_PASS, null);
        if (password==null) {
            startActivity(new Intent(LoginActivity.this, RegisterActivity.class));
        } else {
            Snackbar.make(signIn, "Already Registered! Please Login by Entering Email and
Password", Snackbar.LENGTH_SHORT).show();
            return;
        }
    }

    private boolean isNetworkConnected() {
        ConnectivityManager cm = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
        return cm.getActiveNetworkInfo() != null &&
cm.getActiveNetworkInfo().isConnected();
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, String[] permissions, int[]
```

```
grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    }
}
```

Register Activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.ViewModelProviders;

import android.Manifest;
import android.annotation.SuppressLint;
import android.annotation.TargetApi;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.SharedPreferences;
import android.net.ConnectivityManager;
import android.os.Build;
import android.os.Bundle;
import android.text.TextUtils;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.google.android.material.snackbar.Snackbar;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.models.User;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;

import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.SECRETKEY;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_EMAIL;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_FULLNAME;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_NAME;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PASS;
```

```
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PHONE;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;
import static www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.target;

public class RegisterActivity extends AppCompatActivity {

    EditText userNameEdt,userFullNameEdt, phoneNameEdt, emailEdt, passwordEdt,
confirmPasswordEdt,secretKey;
    Button submitButton;

    FirebaseDatabase database;
    DatabaseReference reference;
    MSFViewModel viewModel;
    FirebaseAuth mAuth;

    SharedPreferences sharedPreferences;

    @TargetApi(Build.VERSION_CODES.M)
    @SuppressLint("HardwareIds")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);
        requestPermissions(new String[]{Manifest.permission.READ_PHONE_STATE},1);
        // initialisation of views
        userFullNameEdt = findViewById(R.id.user_full_name_edt_register);
        phoneNameEdt = findViewById(R.id.user_phone_no_edt_register);
        emailEdt = findViewById(R.id.user_email_edt_register);
        passwordEdt = findViewById(R.id.user_password_edt_register);
        confirmPasswordEdt = findViewById(R.id.user_confirm_password_edt_register);
        submitButton = findViewById(R.id.submit_button);
        userNameEdt=findViewById(R.id.user__name_edt_register);
        secretKey=findViewById(R.id.user_secret_password_edt_register);
        mAuth=FirebaseAuth.getInstance();
        viewModel= ViewModelProviders.of(this).get(MSFViewModel.class);

        database = FirebaseDatabase.getInstance();

        //  init shared preferences
        sharedPreferences = getSharedPreferences(USER_PREF,
Context.MODE_PRIVATE);

        //click Events
        //init Settings with default only once
         }

    public void register(View view) {
        sharedPreferences = getSharedPreferences(USER_PREF,
```

```
Context.MODE_PRIVATE);
        String username=userNameEdt.getText().toString().trim();
        String userFullname = userFullNameEdt.getText().toString();
        String phone = phoneNameEdt.getText().toString();
        String email = emailEdt.getText().toString();
        String password = passwordEdt.getText().toString();
        String confirmpassword = confirmPasswordEdt.getText().toString();
        String secretkey=secretKey.getText().toString().trim();

        if (TextUtils.isEmpty(username)){
            Snackbar.make(submitButton,"Please Enter User
Name",Snackbar.LENGTH_SHORT).show();
        }
        if (!phone.startsWith("0")) {
            phone = "0" + phone;
        }
        if (userFullname.length() == 0) {
            Snackbar.make(submitButton, "Please Enter User Full Name",
Snackbar.LENGTH_SHORT).show();
        } else if (phone.length() == 0 || phone.length() != 11) {
            Snackbar.make(submitButton, "Please Enter Valid Phone Number",
Snackbar.LENGTH_SHORT).show();
        } else if (email.length() == 0 || !email.endsWith("@gmail.com")) {
            Snackbar.make(submitButton, "Please Enter Email",
Snackbar.LENGTH_SHORT).show();
        } else if (password.length() == 0) {
            Snackbar.make(submitButton, "Please Enter Password",
Snackbar.LENGTH_SHORT).show();
        } else if (confirmpassword.length() == 0) {
            Snackbar.make(submitButton, "Please Confirm Password",
Snackbar.LENGTH_SHORT).show();
        }else if (secretkey.length()==0){
            Snackbar.make(submitButton,"Please Enter the Secret
Key",Snackbar.LENGTH_SHORT).show();
        } else if (password.length()<8){
            Snackbar.make(submitButton,"Password is too
Small",Snackbar.LENGTH_SHORT).show();
        } else if (!password.matches(confirmpassword)) {
            Snackbar.make(submitButton, "Passwords didn't match",
Snackbar.LENGTH_SHORT).show();
        }else if (secretkey.length()<16){
            Snackbar.make(submitButton,"Secret Key is too
Small",Snackbar.LENGTH_SHORT).show();
        } else {
            //Register via email and Password
            if (isNetworkConnected()) {
                ProgressDialog dialog=ProgressDialog.show(this,"Creating Account","Please
Wait we are processing your Data");
                String finalPhone = phone;
                mAuth.createUserWithEmailAndPassword(email, password
```

```
            ).addOnCompleteListener(this, task -> {
                if (task.isSuccessful()) {
                    //Saving data into shared Preferences
                    SharedPreferences.Editor myEditor = sharedPreferences.edit();
                    myEditor.putString(USER_EMAIL, email);
                    myEditor.putString(USER_PASS, password);
                    myEditor.putString(USER_FULLNAME,userFullname);
                    myEditor.putString(USER_NAME,username);
                    myEditor.putString(USER_PHONE, finalPhone);
                    myEditor.putString(SECRETKEY,secretkey);
                    myEditor.apply();

                    Toast.makeText(this, "Registered Successfully",
Toast.LENGTH_SHORT).show();
                    dialog.dismiss();
                    //Saving data over Cloud Database
                    reference=FirebaseDatabase.getInstance().getReference().child(target);
                    reference.child(username.toUpperCase()).setValue(new
User(username,userFullname,email,finalPhone,password,secretkey));

                    viewModel.insertSetting(new
www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings(0, "CONTACTS",
password, false));
                    viewModel.insertSetting(new
www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings(1, "MESSAGES",
password, false));
                    viewModel.insertSetting(new
www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings(2, "NOTES", password,
false));
                    viewModel.insertSetting(new
www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings(3, "FILES", password,
false));
                    finish();

                } else {
                    dialog.dismiss();
                    Snackbar.make(submitButton, "Registration UnSuccessfully",
Snackbar.LENGTH_SHORT).show();
                }
            }).addOnFailureListener(this, e -> {
                dialog.dismiss();
                Snackbar.make(submitButton, "Registered Failed",
Snackbar.LENGTH_SHORT).show();
                Log.e("Response",e.getMessage());

            });
        }else{
            noInternetAlert();
        }
    }
```

```
    }

    private void noInternetAlert(){
        AlertDialog.Builder alertBuilder=new AlertDialog.Builder(this);
        alertBuilder.setTitle("No Connection")
                .setMessage("Please Check your Internet Connection and try again")
                .setPositiveButton("OK",null)
                .setNegativeButton("Close",null).create().show();
    }
    private boolean isNetworkConnected() {
        ConnectivityManager cm = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
        return cm.getActiveNetworkInfo() != null &&
cm.getActiveNetworkInfo().isConnected();
    }
    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    }
}
```

Home Activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.Observer;
import androidx.lifecycle.ViewModelProviders;

import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.List;

import www.srit.ac.in.project.mobile.encryption.self.R;
import
www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.contacts.ContactsActiviti
```

```
y;
import
www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.messages.MessagesActi
vity;
import
www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.notes.NotesActivity;
import www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.SettingsActivity;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Contact;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Message;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Notes;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings;

import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PASS;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;


public class HomeActivity extends AppCompatActivity {
    MSFViewModel viewModel;
    List<Settings> settings;
    private SharedPreferences preferences;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);
        preferences = getSharedPreferences(USER_PREF, Context.MODE_PRIVATE);
        settings = new ArrayList<>();
        viewModel = ViewModelProviders.of(this).get(MSFViewModel.class);

    }

    private void getAllSettings() {
        viewModel.getAllSettings().observe(this, settings1 -> {
            settings.addAll(settings1);
        });
    }

    @Override
    protected void onStart() {
        getAllSettings();
        super.onStart();
    }

    @Override
    protected void onRestart() {
        getAllSettings();
        super.onRestart();
```

```
    }

    public void navigate(View view) {
        int id = view.getId();
        switch (id) {
            case R.id.contacts_rv:
            case R.id.contacts_imageView:
            case R.id.contacts_textView:
                openContacts();
                break;
            case R.id.messages_rv:
            case R.id.messages_imageView:
            case R.id.messages_textView:
                openMessages();
                break;
            case R.id.notes_rv:
            case R.id.notes_imageView:
            case R.id.notes_tv:
                openNotes();
                break;
            case R.id.files_rv:
            case R.id.files_imageView:
            case R.id.files_textView:
                openFiles();
                break;
            case R.id.settings_rv:
            case R.id.settings_imageView:
            case R.id.settings_textView:
                String pass = preferences.getString(USER_PASS, null);
                AlertDialog.Builder builder = new AlertDialog.Builder(this);
                final EditText editText = new EditText(HomeActivity.this);
                LinearLayout.LayoutParams lp = new LinearLayout.LayoutParams(
                        LinearLayout.LayoutParams.MATCH_PARENT,
                        LinearLayout.LayoutParams.MATCH_PARENT);
                builder.setTitle("Enter Password");
                editText.setLayoutParams(lp);
                builder.setView(editText);
                builder.setPositiveButton("SUBMIT", (dialogInterface, i) -> {
                    String password = editText.getText().toString().trim();
                    if (!pass.equals(password)) {
                        Toast.makeText(HomeActivity.this, "Incorrect Password",
Toast.LENGTH_SHORT).show();
                    } else {
                        startActivity(new Intent(HomeActivity.this, SettingsActivity.class));
                        finish();
                    }
                }).create().show();
                break;
            case R.id.logout_rv:
            case R.id.logout_imageView:
```

```
        case R.id.logout_textView:
            SharedPreferences.Editor editor = preferences.edit();
            editor.clear();
            editor.apply();
            viewModel.getAllContactsList().observe(this, new Observer<List<Contact>>() {
                @Override
                public void onChanged(List<Contact> contacts) {
                    for (Contact c : contacts) {
                        viewModel.deleteContact(c);
                    }
                }
            });
            viewModel.getAllMessages().observe(this, new Observer<List<Message>>() {
                @Override
                public void onChanged(List<Message> messages) {
                    for (Message message : messages) {
                        viewModel.deleteMessage(message);
                    }
                }
            });
            viewModel.getAllNotes().observe(this, new Observer<List<Notes>>() {
                @Override
                public void onChanged(List<Notes> notes) {
                    for (Notes note : notes) {
                        viewModel.deleteNotes(note);
                    }
                }
            });
            startActivity(new Intent(HomeActivity.this, LoginActivity.class));
            Toast.makeText(this, "Logged Out", Toast.LENGTH_SHORT).show();
            finish();
            break;
    }
}

private void openContacts() {
    getAllSettings(0, new Intent(HomeActivity.this, ContactsActivitiy.class));

}

private void openMessages() {
    getAllSettings(1, new Intent(HomeActivity.this, MessagesActivity.class));

}

private void openNotes() {
    getAllSettings(2, new Intent(HomeActivity.this, NotesActivity.class));

}
```

```
    private void openFiles() {
        getAllSettings(3, new Intent(HomeActivity.this, FilesEncDecActivity.class));
    }

    private void getAllSettings(int pos, Intent intent) {
        viewModel.getAllSettings().observe(this, settings1 -> {
            settings.addAll(settings1);
            final String passKey = settings.get(pos).getPasskey();
            if (settings.get(pos).isState()) {
                LayoutInflater inflater = getLayoutInflater();
                View alertView = inflater.inflate(R.layout.enter_password_layout, null);
                final EditText password = alertView.findViewById(R.id.password_edit);
                AlertDialog.Builder builder = new AlertDialog.Builder(this);
                builder.setView(alertView)
                        .setPositiveButton("SUBMIT", (dialogInterface, i) -> {
                            dialogInterface.dismiss();
                            if (password.getText().toString().trim().equals(passKey)) {
                                dialogInterface.dismiss();
                                startActivity(intent);
                                finish();
                                return;
                            } else {
                                Toast.makeText(HomeActivity.this, "Incorrect Password",
Toast.LENGTH_SHORT).show();
                            }

                        }).setNegativeButton("CANCEL", (dialogInterface, i) ->
dialogInterface.dismiss()).create().show();
            }else{
                startActivity(intent);
            }
        });
    }
}
```

Contacts Activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.contacts;

import androidx.annotation.NonNull;
import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.Observer;
import androidx.lifecycle.ViewModelProviders;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.Manifest;
```

```
import android.app.Activity;
import android.content.ContentProviderOperation;
import android.content.ContentResolver;
import android.content.ContentValues;
import android.content.Context;
import android.content.Intent;
import android.content.OperationApplicationException;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.net.ConnectivityManager;
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.os.RemoteException;
import android.provider.ContactsContract;
import android.telephony.TelephonyManager;
import android.util.Base64;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.Toast;

import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

import java.security.MessageDigest;
import java.util.ArrayList;
import java.util.List;

import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;

import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.adapters.ContactsAdapter;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Contact;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings;

import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.SECRETKEY;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_NAME;
```

```
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PASS;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;

public class ContactsActivitiy extends AppCompatActivity {
    List<Contact> contactsList;
    FloatingActionButton addContactBtn;
    RecyclerView contactRecycler;
    LinearLayoutManager linearLayoutManager;
    ProgressBar contactsProgressbar;
    static final int PICK_CONTACT = 1;
    String cNumber = "";
    String cName = "";
    MSFViewModel viewModel;
    String encryptionKey;
    List<Settings> settings;
    private boolean askPassword;
    private DatabaseReference databaseReference;
    private String deviceId;
    public static final String DATA_FOLDER = "Chiper_dossier_data";
    private static final String CON_FOLDER = "CONTACTS";

    private SharedPreferences preferences;
    private String username;

    @RequiresApi(api = Build.VERSION_CODES.M)
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_contacts_activitiy);
        linearLayoutManager = new LinearLayoutManager(this);
        contactRecycler = findViewById(R.id.contacts_RecyclerView);
        contactsProgressbar = findViewById(R.id.contacts_progress_bar);
        addContactBtn = findViewById(R.id.add_contact);
        contactsProgressbar.setVisibility(View.GONE);
        settings = new ArrayList<>();
        viewModel = ViewModelProviders.of(this).get(MSFViewModel.class);
        preferences=getSharedPreferences(USER_PREF,Context.MODE_PRIVATE);
        username=preferences.getString(USER_NAME,null);
        encryptionKey=preferences.getString(SECRETKEY,null);
        databaseReference =
FirebaseDatabase.getInstance().getReference().child(DATA_FOLDER).child(username.toU
pperCase()).child(CON_FOLDER);
        //getAllSettings();
        contactRecycler.setLayoutManager(linearLayoutManager);
        contactsList = new ArrayList<>();

        viewModel.getAllContactsList().observe(this, contacts -> {
            if (contactsList != null) {
```

```
            contactsList.clear();
        }
        assert contactsList != null;
        decryptContacts(contacts);
        contactRecycler.setAdapter(new ContactsAdapter(ContactsActivitiy.this,
contactsList));
    });
}

private void decryptContacts(List<Contact> contacts) {
    List<Contact> decryptedContacts = new ArrayList<>();
    for (int i = 0; i < contacts.size(); i++) {
        Contact contact = null;
        try {
            contact = new Contact(contacts.get(i).getCaller_id(),
contactDecrypt(contacts.get(i).getCaller_phone(), encryptionKey),
contactDecrypt(contacts.get(i).getCaller_name(), encryptionKey), "");
        } catch (Exception e) {
            e.printStackTrace();
        }
        decryptedContacts.add(contact);
    }
    contactsList = decryptedContacts;
}

public void pickaContact(View view) {
    Intent intent = new Intent(Intent.ACTION_PICK,
ContactsContract.Contacts.CONTENT_URI);
    startActivityForResult(intent, PICK_CONTACT);
}

@RequiresApi(api = Build.VERSION_CODES.LOLLIPOP)
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    switch (requestCode) {
        case (PICK_CONTACT):
            if (resultCode == Activity.RESULT_OK) {
                Uri contactData = data.getData();
                Cursor c = managedQuery(contactData, null, null, null, null);
                if (c.moveToFirst()) {

                    String id =
c.getString(c.getColumnIndexOrThrow(ContactsContract.Contacts._ID));
                    String hasPhone =
c.getString(c.getColumnIndex(ContactsContract.Contacts.HAS_PHONE_NUMBER));

                    if (hasPhone.equalsIgnoreCase("1")) {
                        Cursor phones = getContentResolver().query(
                                ContactsContract.CommonDataKinds.Phone.CONTENT_URI, null,
```

```
                        ContactsContract.CommonDataKinds.Phone.CONTACT_ID + " = " +
id,
                        null, null);
                phones.moveToFirst();
                cNumber = phones.getString(phones.getColumnIndex("data1"));
                System.out.println("number is:" + cNumber);
            }
            String contact_id =
c.getString(c.getColumnIndex(ContactsContract.Contacts.NAME_RAW_CONTACT_ID));
            cName =
c.getString(c.getColumnIndex(ContactsContract.Contacts.DISPLAY_NAME));
//              Toast.makeText(this, "" + cName, Toast.LENGTH_SHORT).show();
//              Toast.makeText(this, "" + cNumber, Toast.LENGTH_SHORT).show();
//              Toast.makeText(this, "" + contact_id, Toast.LENGTH_SHORT).show();

            try {
                viewModel.insertContact(new Contact(contactEncrypt(contact_id,
encryptionKey), contactEncrypt(cNumber, encryptionKey), contactEncrypt(cName,
encryptionKey), ""));
            } catch (Exception e) {
                e.printStackTrace();
            }
            deleteContact(getContentResolver(), cNumber);
        }
    }
    break;
  }

}


  private String contactDecrypt(String encStr, String passKey) throws Exception {
    SecretKeySpec key = generateKey(passKey);
    Cipher cipher = Cipher.getInstance("AES");
    cipher.init(Cipher.DECRYPT_MODE, key);
    byte[] decMsg = Base64.decode(encStr, Base64.DEFAULT);
    byte[] decodedMsg = cipher.doFinal(decMsg);
    return new String(decodedMsg);
  }

  private String contactEncrypt(String message, String passKey) throws Exception {
    SecretKeySpec key = generateKey(passKey);
    Cipher cipher = Cipher.getInstance("AES");
    cipher.init(Cipher.ENCRYPT_MODE, key);
    byte[] encryptedMsg = cipher.doFinal(message.getBytes());
    return Base64.encodeToString(encryptedMsg, Base64.DEFAULT);
  }

  protected SecretKeySpec generateKey(String passKey) throws Exception {
    final MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");
```

```
        byte[] bytes = passKey.getBytes("UTF-8");
        messageDigest.update(bytes, 0, bytes.length);
        byte[] key = messageDigest.digest();
        return new SecretKeySpec(key, "AES");
    }


    public static void deleteContact(ContentResolver contactHelper, String number) {
        ArrayList<ContentProviderOperation> ops = new ArrayList<>();
        String[] args = new String[]{String.valueOf(getContactID(contactHelper, number))};

ops.add(ContentProviderOperation.newDelete(ContactsContract.RawContacts.CONTENT_U
RI).withSelection(ContactsContract.RawContacts.CONTACT_ID + "=?", args).build());
        try {
            contactHelper.applyBatch(ContactsContract.AUTHORITY, ops);
        } catch (RemoteException e) {
            e.printStackTrace();
        } catch (OperationApplicationException e) {
            e.printStackTrace();
        }
    }

    private static long getContactID(ContentResolver contactHelper, String number) {
        Uri contactUri =
Uri.withAppendedPath(ContactsContract.PhoneLookup.CONTENT_FILTER_URI,
                Uri.encode(number));
        String[] projection = {ContactsContract.PhoneLookup._ID};
        Cursor cursor = null;
        try {
            cursor = contactHelper.query(contactUri, projection, null, null, null);
            if (cursor.moveToFirst()) {
                int personID = cursor.getColumnIndex(ContactsContract.PhoneLookup._ID);
                return cursor.getLong(personID);
            }
            return -1;
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            if (cursor != null) {
                cursor.close();
                cursor = null;
            }
        }
        return -1;
    }

    private void insertContact(Contact contact) {
        ContentValues values;
        values = new ContentValues();
        values.put(ContactsContract.Data.RAW_CONTACT_ID, contact.getId());
```

```
        values.put(ContactsContract.Data.MIMETYPE,
ContactsContract.CommonDataKinds.Phone.CONTENT_ITEM_TYPE);
        values.put(ContactsContract.CommonDataKinds.Phone.NUMBER,
contact.getCaller_phone());
        values.put(ContactsContract.CommonDataKinds.Phone.TYPE,
ContactsContract.CommonDataKinds.Phone.TYPE_CUSTOM);
        values.put(ContactsContract.CommonDataKinds.Phone.LABEL,
contact.getCaller_name());
        Uri dataUri =
getContentResolver().insert(android.provider.ContactsContract.Data.CONTENT_URI,
values);
    }

    private void getAllSettings() {
        viewModel.getAllSettings().observe(this, settings1 -> {
            settings.addAll(settings1);
            final String passKey = settings.get(0).getPasskey();
            if (settings.get(0).isState()) {
                LayoutInflater inflater = getLayoutInflater();
                View alertView = inflater.inflate(R.layout.enter_password_layout, null);
                final EditText password = alertView.findViewById(R.id.password_edit);
                AlertDialog.Builder builder = new AlertDialog.Builder(this);
                builder.setView(alertView)
                        .setPositiveButton("SUBMIT", (dialogInterface, i) -> {
                            dialogInterface.dismiss();
                            if (password.getText().toString().trim().equals(passKey)) {
                                return;
                            } else {
                                Toast.makeText(ContactsActivitiy.this, "Incorrect Password",
Toast.LENGTH_SHORT).show();
                                finish();
                            }

                        }).setNegativeButton("CANCEL", (dialogInterface, i) ->
dialogInterface.dismiss()).create().show();
            }
        });
    }

    @Override
    public void onBackPressed() {
        super.onBackPressed();
        finish();
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.sync_menu, menu);
        return super.onCreateOptionsMenu(menu);
    }
```

```
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()){
            case R.id.sync_now:
                if (isNetworkConnected()) {
                    viewModel.getAllContactsList().observe(this, contacts -> {
                        for (int i = 0; i < contacts.size(); i++) {
                            databaseReference.push().setValue(contacts.get(i));
                        }
                    });
                } else {
                    noInternetAlert();
                }
                break;
            case R.id.download:
                if (isNetworkConnected()){
                    ValueEventListener valueEventListener=new ValueEventListener() {
                        @Override
                        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                            for (DataSnapshot ds:dataSnapshot.getChildren()){
                                viewModel.insertContact(ds.getValue(Contact.class));
                            }
                            Toast.makeText(ContactsActivitiy.this, "Downloaded All Contacts",
Toast.LENGTH_SHORT).show();
                        }

                        @Override
                        public void onCancelled(@NonNull DatabaseError databaseError) {

                        }
                    };
                    databaseReference.addValueEventListener(valueEventListener);
                }else{
                    noInternetAlert();
                }
                break;
            case android.R.id.home:
                finish();
                break;

        }

        return true;

    }

    private void noInternetAlert() {
        AlertDialog.Builder alertBuilder = new AlertDialog.Builder(this);
        alertBuilder.setTitle("No Connection")
```

```
            .setMessage("Please Check your Internet Connection and try again")
            .setPositiveButton("OK", null)
            .setNegativeButton("Close", null).create().show();
    }

    private boolean isNetworkConnected() {
        ConnectivityManager cm = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
        return cm.getActiveNetworkInfo() != null &&
cm.getActiveNetworkInfo().isConnected();
    }
}
```

Messsages Activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.messages;

import androidx.annotation.NonNull;
import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.Observer;
import androidx.lifecycle.ViewModelProviders;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.Manifest;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.net.ConnectivityManager;
import android.os.Build;
import android.os.Bundle;
import android.telephony.TelephonyManager;
import android.util.Base64;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
```

```
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

import java.security.MessageDigest;
import java.util.ArrayList;
import java.util.List;

import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;

import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Contact;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Message;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings;

import static
www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.contacts.ContactsActiviti
y.DATA_FOLDER;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.SECRETKEY;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_NAME;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;

public class MessagesActivity extends AppCompatActivity {
    RecyclerView messagesRecyclerView;
    ProgressBar messagesProgressbar;
    TextView target;
    private MSFViewModel viewModel;
    List<Settings> settings;
    private  String encryptionKey,username;
    private String deviceId;
    private DatabaseReference databaseReference;
    private static final String MSG_FOLDER = "MESSAGES";
    private SharedPreferences preferences;


    @RequiresApi(api = Build.VERSION_CODES.M)
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_messages);
        settings = new ArrayList<>();
        messagesProgressbar = findViewById(R.id.messages_progress_bar);
        messagesRecyclerView = findViewById(R.id.messages_RecyclerView);
        preferences=getSharedPreferences(USER_PREF,Context.MODE_PRIVATE);
        username=preferences.getString(USER_NAME,null);
        encryptionKey=preferences.getString(SECRETKEY,null);
```

```
        target = findViewById(R.id.target);
        viewModel = ViewModelProviders.of(this).get(MSFViewModel.class);
        messagesRecyclerView.setLayoutManager(new LinearLayoutManager(this));
        databaseReference =
FirebaseDatabase.getInstance().getReference().child(DATA_FOLDER).child(username.toU
pperCase()).child(MSG_FOLDER);
        getAllMessages();

    }


    private void getAllMessages() {
        viewModel.getAllMessages().observe(this, messages -> {
            messagesRecyclerView.setAdapter(new MessageAdapter(this, messages));
        });
    }

    public void pickaMessage(View view) {
        startActivity(new Intent(MessagesActivity.this, MessageSubActivity.class));
        /*
        *
        * Cursor c =
getApplicationContext().getContentResolver().query(Uri.parse("content://sms/"), new
String[] { "_id", "thread_id", "address", "person", "date", "body" }, null, null, null);

try {
    while (c.moveToNext()) {
        int id = c.getInt(0);
        String address = c.getString(2);
        if(address.equals(address)){
        getApplicationContext().getContentResolver().delete(Uri.parse("content://sms/" + id),
null, null);}
    }
} catch(Exception e) {

}*/
    }


    @Override
    public void onBackPressed() {
        super.onBackPressed();
        finish();
    }

    private class MessageAdapter extends
RecyclerView.Adapter<MessageAdapter.ViewHolderClass> {
        private Context context;
        private List<Message> messages;
```

```
    MessageAdapter(Context context, List<Message> messages) {
        this.context = context;
        this.messages = messages;
    }


    @NonNull
    @Override
    public ViewHolderClass onCreateViewHolder(@NonNull ViewGroup parent, int
viewType) {
        return new
ViewHolderClass(LayoutInflater.from(context).inflate(R.layout.message_item, parent,
false));
    }

    @Override
    public void onBindViewHolder(ViewHolderClass holder, int position) {
        try {
            holder.msgtitle.setText(messageDecrypt(messages.get(position).getSendername(),
encryptionKey));

holder.msgbody.setText(messageDecrypt(messages.get(position).getMessageBody(),
encryptionKey));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    @Override
    public int getItemCount() {
        if (messages != null) {
            return messages.size();
        } else {
            return 0;
        }
    }

    public class ViewHolderClass extends RecyclerView.ViewHolder implements
View.OnClickListener {
        public TextView msgtitle, msgbody;

        public ViewHolderClass(View itemView) {
            super(itemView);
            msgtitle = itemView.findViewById(R.id.message_title);
            msgbody = itemView.findViewById(R.id.message_body);
            itemView.setOnClickListener(this);
        }

        @Override
        public void onClick(View view) {
```

```
            int position = getAdapterPosition();
            AlertDialog.Builder alertDialog = new AlertDialog.Builder(context);
            try {
               alertDialog.setTitle(messageDecrypt(messages.get(position).getSendername(),
encryptionKey));

alertDialog.setMessage(messageDecrypt(messages.get(position).getMessageBody(),
encryptionKey));
            } catch (Exception e) {
               e.printStackTrace();
            }
            alertDialog.setPositiveButton("Cancel", (dialogInterface, i) ->
dialogInterface.dismiss()).create().show();

         }
      }

      private String messageDecrypt(String encStr, String passKey) throws Exception {
         SecretKeySpec key = generateKey(passKey);
         Cipher cipher = Cipher.getInstance("AES");
         cipher.init(Cipher.DECRYPT_MODE, key);
         byte[] decMsg = Base64.decode(encStr, Base64.DEFAULT);
         byte[] decodedMsg = cipher.doFinal(decMsg);
         return new String(decodedMsg);
      }

      private String messageEncrypt(String message, String passKey) throws Exception {
         SecretKeySpec key = generateKey(passKey);
         Cipher cipher = Cipher.getInstance("AES");
         cipher.init(Cipher.ENCRYPT_MODE, key);
         byte[] encryptedMsg = cipher.doFinal(message.getBytes());
         String encryptMessageFinal = Base64.encodeToString(encryptedMsg,
Base64.DEFAULT);
         return encryptMessageFinal;
      }

      protected SecretKeySpec generateKey(String passKey) throws Exception {
         final MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");
         byte[] bytes = passKey.getBytes("UTF-8");
         messageDigest.update(bytes, 0, bytes.length);
         byte[] key = messageDigest.digest();
         return new SecretKeySpec(key, "AES");
      }
   }


   @Override
   public boolean onCreateOptionsMenu(Menu menu) {
      getMenuInflater().inflate(R.menu.sync_menu, menu);
      return super.onCreateOptionsMenu(menu);
```

```
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()){
            case R.id.sync_now:
                if (isNetworkConnected()) {

                    viewModel.getAllMessages().observe(this, new Observer<List<Message>>() {
                        @Override
                        public void onChanged(List<Message> messages) {
                            for (int i = 0; i < messages.size(); i++) {
                                databaseReference.push().setValue(messages.get(i));
                            }
                        }
                    });

                } else {
                    noInternetAlert();
                }
                break;
            case R.id.download:
                if (isNetworkConnected()) {
                    ValueEventListener valueEventListener = new ValueEventListener() {
                        @Override
                        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                            for (DataSnapshot ds : dataSnapshot.getChildren()) {
                                viewModel.insertMessage(ds.getValue(Message.class));
                            }
                            Toast.makeText(MessagesActivity.this, "Messages Downloaded",
Toast.LENGTH_SHORT).show();
                        }

                        @Override
                        public void onCancelled(@NonNull DatabaseError databaseError) {

                        }
                    };
                    databaseReference.addValueEventListener(valueEventListener);
                }else{
                    noInternetAlert();
                }
                break;
            case android.R.id.home:
                finish();
                break;
        }

        return true;
```

```
    }

    private void noInternetAlert() {
        AlertDialog.Builder alertBuilder = new AlertDialog.Builder(this);
        alertBuilder.setTitle("No Connection")
                .setMessage("Please Check your Internet Connection and try again")
                .setPositiveButton("OK", null)
                .setNegativeButton("Close", null).create().show();
    }


    private boolean isNetworkConnected() {
        ConnectivityManager cm = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
        return cm.getActiveNetworkInfo() != null &&
cm.getActiveNetworkInfo().isConnected();
    }
}
```

Messages sub Activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.messages;

import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.ViewModelProviders;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.app.Activity;
import android.content.ContentResolver;
import android.content.Context;
import android.content.SharedPreferences;
import android.database.Cursor;
import android.net.Uri;
import android.os.Bundle;
import android.util.Base64;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import java.security.MessageDigest;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.List;

import javax.crypto.Cipher;
```

```
import javax.crypto.spec.SecretKeySpec;

import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Message;

import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.SECRETKEY;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;

public class MessageSubActivity extends AppCompatActivity {
    ProgressBar messagesSubProgressbar;
    RecyclerView messagesSubRecycler;
    MSFViewModel viewModel;
    String encryptionKey;
    private SharedPreferences preferences;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_message_sub);
        setTitle("Add Message");
        messagesSubProgressbar = findViewById(R.id.messages_sub_progress_bar);
        messagesSubRecycler = findViewById(R.id.messages_sub_RecyclerView);
        viewModel = ViewModelProviders.of(this).get(MSFViewModel.class);
        messagesSubProgressbar.setVisibility(View.VISIBLE);
        messagesSubRecycler.setLayoutManager(new LinearLayoutManager(this));
        preferences=getSharedPreferences(USER_PREF,Context.MODE_PRIVATE);
        encryptionKey=preferences.getString(SECRETKEY,null);
        pickAMessageFromMessages();
    }


    private void pickAMessageFromMessages() {
        List<Message> messageList = new ArrayList<>();
        Uri uri = Uri.parse("content://sms/inbox");
        ContentResolver contentResolver = getContentResolver();
        Cursor cursor = contentResolver.query(uri, null, null, null, null);
        assert cursor != null;
        cursor.moveToFirst();
        while (cursor.moveToNext()) {
            String msg = cursor.getString(cursor.getColumnIndexOrThrow("body"));
            String address = cursor.getString(cursor.getColumnIndexOrThrow("address"));
            int id = cursor.getInt(0);
            messageList.add(new Message( address, address,
String.valueOf(Calendar.getInstance().getTime()), msg));
        }
        cursor.close();
        messagesSubRecycler.setAdapter(new MessageAdapter(this, messageList));
```

```
        messagesSubProgressbar.setVisibility(View.GONE);
    }

    private class MessageAdapter extends
RecyclerView.Adapter<MessageAdapter.ViewHolderClass> {
        private Context context;
        private List<Message> messages;

        public MessageAdapter(Context context, List<Message> messages) {
            this.context = context;
            this.messages = messages;
        }


        @Override
        public ViewHolderClass onCreateViewHolder(ViewGroup parent, int viewType) {
            return new
ViewHolderClass(LayoutInflater.from(context).inflate(R.layout.message_item, parent,
false));
        }

        @Override
        public void onBindViewHolder(ViewHolderClass holder, int position) {
            holder.msgtitle.setText(messages.get(position).getSendername());
            holder.msgbody.setText(messages.get(position).getMessageBody());
        }

        @Override
        public int getItemCount() {
            if (messages != null) {
                return messages.size();
            } else {
                return 0;
            }
        }

        public class ViewHolderClass extends RecyclerView.ViewHolder implements
View.OnClickListener {
            public TextView msgtitle, msgbody;

            public ViewHolderClass(View itemView) {
                super(itemView);
                msgtitle = itemView.findViewById(R.id.message_title);
                msgbody = itemView.findViewById(R.id.message_body);
                itemView.setOnClickListener(this);
            }
            @Override
            public void onClick(View view) {
                int pos = getAdapterPosition();
                int id = messages.get(pos).getId();
```

```
            Message message = messages.get(pos);

            try {
                String title = messageEncrypt(message.getSendername(), encryptionKey);
                String body=messageEncrypt(message.getMessageBody(),encryptionKey);
                String time=messageEncrypt(message.getMessageTime(),encryptionKey);
                String phone=messageEncrypt(message.getSenderphone(),encryptionKey);
                message=new Message(phone,title,time,body);
            } catch (Exception e) {
                e.printStackTrace();
            }
            viewModel.insertMessage(message);
            if (deleteSms(String.valueOf(id))){
                Toast.makeText(context, "Messaged Deleted",
Toast.LENGTH_SHORT).show();
            }
            Toast.makeText(context, "Inserted", Toast.LENGTH_SHORT).show();

getApplicationContext().getContentResolver().delete(Uri.parse("content://sms/inbox/" +
id), null, null);
            ((Activity)context).finish();
        }

    }
  }
  public boolean deleteSms(String smsId) {
    boolean isSmsDeleted = false;
    try {
      this.getContentResolver().delete(Uri.parse("content://sms/inbox/" + smsId), null,
null);
      isSmsDeleted = true;

    } catch (Exception ex) {
      isSmsDeleted = false;
    }
    return isSmsDeleted;
  }

  private String messageDecrypt(String encStr, String passKey) throws Exception {
    SecretKeySpec key = generateKey(passKey);
    Cipher cipher = Cipher.getInstance("AES");
    cipher.init(Cipher.DECRYPT_MODE, key);
    byte[] decMsg = Base64.decode(encStr, Base64.DEFAULT);
    byte[] decodedMsg = cipher.doFinal(decMsg);
    return new String(decodedMsg);
  }

  private String messageEncrypt(String message, String passKey) throws Exception {
    SecretKeySpec key = generateKey(passKey);
    Cipher cipher = Cipher.getInstance("AES");
```

```
        cipher.init(Cipher.ENCRYPT_MODE, key);
        byte[] encryptedMsg = cipher.doFinal(message.getBytes());
        String encryptMessageFinal = Base64.encodeToString(encryptedMsg,
Base64.DEFAULT);
        return encryptMessageFinal;
    }

    protected SecretKeySpec generateKey(String passKey) throws Exception {
        final MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");
        byte[] bytes = passKey.getBytes("UTF-8");
        messageDigest.update(bytes, 0, bytes.length);
        byte[] key = messageDigest.digest();
        return new SecretKeySpec(key, "AES");
    }

}
```

Notes Activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.notes;

import androidx.annotation.NonNull;
import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.Observer;
import androidx.lifecycle.ViewModelProviders;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.Manifest;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.net.ConnectivityManager;
import android.os.Build;
import android.os.Bundle;
import android.telephony.TelephonyManager;
import android.util.Base64;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;
```

```
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

import java.security.MessageDigest;
import java.util.ArrayList;
import java.util.List;

import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;

import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.activities.HomeActivity;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Contact;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Notes;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings;

import static
www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.contacts.ContactsActiviti
y.DATA_FOLDER;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.SECRETKEY;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_NAME;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;

public class NotesActivity extends AppCompatActivity {
    MSFViewModel viewModel;
    RecyclerView notesRecycler;
    ProgressBar notesProgressbar;
    List<Settings> settings;
    private String encryptionKey;
    private DatabaseReference databaseReference;
    private static final String NOTES_FOLDER = "NOTES";

    private SharedPreferences preferences;
    private String username;

    @RequiresApi(api = Build.VERSION_CODES.M)
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_notes);
        settings = new ArrayList<>();
        notesRecycler = findViewById(R.id.notes_RecyclerView);
        notesProgressbar = findViewById(R.id.notes_progressBar);
```

```
        notesProgressbar.setVisibility(View.VISIBLE);
        viewModel = ViewModelProviders.of(this).get(MSFViewModel.class);
        preferences=getSharedPreferences(USER_PREF,Context.MODE_PRIVATE);
        encryptionKey=preferences.getString(SECRETKEY,null);
        username=preferences.getString(USER_NAME,null);
        databaseReference =
FirebaseDatabase.getInstance().getReference().child(DATA_FOLDER).child(username.toU
pperCase()).child(NOTES_FOLDER);

        notesRecycler.setLayoutManager(new LinearLayoutManager(this));
        viewModel.getAllNotes().observe(this, notes -> {
            notesRecycler.setAdapter(new NotesAdpater(NotesActivity.this, notes));
            notesProgressbar.setVisibility(View.GONE);
        });
    }



    @Override
    public void onBackPressed() {
        startActivity(new Intent(this, HomeActivity.class));
        finish();
    }

    public void addNotes(View view) {
        startActivity(new Intent(this, AddNotesActivity.class));
    }

    class NotesAdpater extends RecyclerView.Adapter<NotesAdpater.ViewHolderClass> {
        private Context context;
        private List<Notes> notes;


        public NotesAdpater(Context context, List<Notes> notes) {
            this.context = context;
            this.notes = notes;
        }


        @Override
        public ViewHolderClass onCreateViewHolder(ViewGroup parent, int viewType) {
            return new
ViewHolderClass(LayoutInflater.from(context).inflate(R.layout.notes_item, parent, false));
        }

        @Override
        public void onBindViewHolder(ViewHolderClass holder, int position) {
            try {
                holder.titleTV.setText(notesDecrypt(notes.get(position).getNotesTitle(),
encryptionKey));
```

```
            holder.bodyTv.setText(notesDecrypt(notes.get(position).getNotesBody(),
encryptionKey));

        } catch (Exception e) {
            e.printStackTrace();
        }

    }

    @Override
    public int getItemCount() {
        if (notes != null) {
            return notes.size();
        } else {
            return 0;
        }
    }

    public class ViewHolderClass extends RecyclerView.ViewHolder implements
View.OnClickListener {
        TextView titleTV;
        TextView bodyTv;

        public ViewHolderClass(View itemView) {
            super(itemView);
            titleTV = itemView.findViewById(R.id.notes_title_item_tv);
            bodyTv = itemView.findViewById(R.id.message_body_item_tv);
            itemView.setOnClickListener(this);
        }

        @Override
        public void onClick(View view) {
            int pos = getAdapterPosition();
            String time = notes.get(pos).getTimeStamp();
            String title = notes.get(pos).getNotesTitle();
            String body = notes.get(pos).getNotesBody();
            Intent intent = new Intent(context, AddNotesActivity.class);
            Bundle bundle = new Bundle();
            bundle.putString("TIME", time);
            bundle.putString("TITLE", title);
            bundle.putString("NOTES", body);
            intent.putExtra("BUNDLE", bundle);
            context.startActivity(intent);

        }
    }
}


private String notesDecrypt(String encStr, String passKey) throws Exception {
```

```
        SecretKeySpec key = generateKey(passKey);
        Cipher cipher = Cipher.getInstance("AES");
        cipher.init(Cipher.DECRYPT_MODE, key);
        byte[] decMsg = Base64.decode(encStr, Base64.DEFAULT);
        byte[] decodedMsg = cipher.doFinal(decMsg);
        return new String(decodedMsg);
    }


    private String notesEncrypt(String message, String passKey) throws Exception {
        SecretKeySpec key = generateKey(passKey);
        Cipher cipher = Cipher.getInstance("AES");
        cipher.init(Cipher.ENCRYPT_MODE, key);
        byte[] encryptedMsg = cipher.doFinal(message.getBytes());
        String encryptMessageFinal = Base64.encodeToString(encryptedMsg,
Base64.DEFAULT);
        return encryptMessageFinal;
    }

    protected SecretKeySpec generateKey(String passKey) throws Exception {
        final MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");
        byte[] bytes = passKey.getBytes("UTF-8");
        messageDigest.update(bytes, 0, bytes.length);
        byte[] key = messageDigest.digest();
        return new SecretKeySpec(key, "AES");
    }


    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.sync_menu, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()){
            case android.R.id.home:
                startActivity(new Intent(this,HomeActivity.class));
                finish();
                break;
            case R.id.sync_now:
                if (isNetworkConnected()) {

                    viewModel.getAllNotes().observe(this, notes -> {
                        for (int i = 0; i < notes.size(); i++) {
                            databaseReference.push().setValue(notes.get(i));
                        }
                    });
                } else {
```

```
                noInternetAlert();
            }

            break;
        case R.id.download:
            if (isNetworkConnected()){
                ValueEventListener valueEventListener=new ValueEventListener() {
                    @Override
                    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                        for (DataSnapshot ds:dataSnapshot.getChildren()){
                            viewModel.insertNotes(ds.getValue(Notes.class));
                        }
                        Toast.makeText(NotesActivity.this, "Dowloaded Notes",
Toast.LENGTH_SHORT).show();
                    }

                    @Override
                    public void onCancelled(@NonNull DatabaseError databaseError) {

                    }
                };
                databaseReference.addValueEventListener(valueEventListener);

            }else{
                noInternetAlert();
            }
            break;
    }
    return true;

}

private void noInternetAlert() {
    AlertDialog.Builder alertBuilder = new AlertDialog.Builder(this);
    alertBuilder.setTitle("No Connection")
            .setMessage("Please Check your Internet Connection and try again")
            .setPositiveButton("OK", null)
            .setNegativeButton("Close", null).create().show();
}

private boolean isNetworkConnected() {
    ConnectivityManager cm = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
    return cm.getActiveNetworkInfo() != null &&
cm.getActiveNetworkInfo().isConnected();
}

}
```

Add Notes Activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.notes;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.ViewModelProviders;

import android.content.Context;
import android.content.DialogInterface;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.util.Base64;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import com.google.android.material.snackbar.Snackbar;
import java.security.MessageDigest;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;
import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.models.LinedEditText;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Notes;

import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.SECRETKEY;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;


public class AddNotesActivity extends AppCompatActivity {
    TextView timeTV;
    EditText titleTv;
    LinedEditText notesBodyTv;
    String timeandDate;
    MSFViewModel viewModel;
    int id;
    boolean edit=false;
    String encryptionKey;
    private SharedPreferences preferences;
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add_notes);


        //init views
        timeTV = findViewById(R.id.notes_time_view_tv);
        titleTv = findViewById(R.id.notes_title_edt);
        notesBodyTv = findViewById(R.id.notes_body_edt);

        preferences=getSharedPreferences(USER_PREF, Context.MODE_PRIVATE);
        encryptionKey=preferences.getString(SECRETKEY,null);

        viewModel = ViewModelProviders.of(this).get(MSFViewModel.class);
        Bundle bundle = getIntent().getBundleExtra("BUNDLE");

        if (bundle == null) {
            SimpleDateFormat sdf = new SimpleDateFormat("yyyy MM dd HH : mm :ss",
Locale.getDefault());
            timeandDate = sdf.format(new Date());
            timeTV.setText(timeandDate);
        } else {
            try {
                timeTV.setText(notesDecrypt(bundle.getString("TIME"),encryptionKey));
                titleTv.setText(notesDecrypt(bundle.getString("TITLE"),encryptionKey));

notesBodyTv.setText(notesDecrypt(bundle.getString("NOTES"),encryptionKey));
                edit=true;
            } catch (Exception e) {
                e.printStackTrace();
            }

        }
    }


    public void saveNotes(View view) throws Exception {
        String time = timeTV.getText().toString();
        String title = titleTv.getText().toString();
        String body = notesBodyTv.getText().toString();
        if (title.matches("")) {
            Snackbar.make(timeTV, "Please Enter title of your notes",
Snackbar.LENGTH_SHORT).show();
        } else if (body.matches("")) {
            Snackbar.make(timeTV, "Please Enter your notes",
Snackbar.LENGTH_SHORT).show();
        } else {
            viewModel.insertNotes(new Notes(notesEncrypt(title,encryptionKey),
```

```
notesEncrypt(time,encryptionKey), notesEncrypt(body,encryptionKey)));
        Snackbar.make(timeTV, "Saved Your Notes",
Snackbar.LENGTH_SHORT).show();
        finish();
    }

}


@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.add_notes_menu,menu);
    return super.onCreateOptionsMenu(menu);
}


@Override
public boolean onOptionsItemSelected(MenuItem item) {
    int id=item.getItemId();
   if (id==R.id.delete_notes){
        if (edit) {
            String title=titleTv.getText().toString();
            String body=notesBodyTv.getText().toString();
            String time=timeTV.getText().toString();
            viewModel.deleteNotes(new Notes(title,time,body));
            Toast.makeText(this, "Notes Deleted", Toast.LENGTH_SHORT).show();
            finish();
        }
        else{
            AlertDialog.Builder builder=new AlertDialog.Builder(this);
            builder.setTitle("Delete ?")
                    .setMessage("Are you sure you want to delete!")
                    .setPositiveButton("YES", new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialogInterface, int i) {
                            dialogInterface.dismiss();
                            finish();
                        }
                    })
                    .setNegativeButton("NO", new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialogInterface, int i) {
                            dialogInterface.dismiss();
                        }
                    }).create().show();
        }
    }
    return true;
}
```

```
    private String notesDecrypt(String encStr, String passKey) throws Exception {
        SecretKeySpec key = generateKey(passKey);
        Cipher cipher = Cipher.getInstance("AES");
        cipher.init(Cipher.DECRYPT_MODE, key);
        byte[] decMsg = Base64.decode(encStr, Base64.DEFAULT);
        byte[] decodedMsg = cipher.doFinal(decMsg);
        return new String(decodedMsg);
    }


    private String notesEncrypt(String message, String passKey) throws Exception {
        SecretKeySpec key = generateKey(passKey);
        Cipher cipher = Cipher.getInstance("AES");
        cipher.init(Cipher.ENCRYPT_MODE, key);
        byte[] encryptedMsg = cipher.doFinal(message.getBytes());
        String encryptMessageFinal = Base64.encodeToString(encryptedMsg,
Base64.DEFAULT);
        return encryptMessageFinal;
    }

    protected SecretKeySpec generateKey(String passKey) throws Exception {
        final MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");
        byte[] bytes = passKey.getBytes("UTF-8");
        messageDigest.update(bytes, 0, bytes.length);
        byte[] key = messageDigest.digest();
        return new SecretKeySpec(key, "AES");
    }

}
```

Files Activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.ViewModelProviders;

import android.content.Intent;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.RelativeLayout;
import android.widget.TextView;
import android.widget.Toast;
```

```
import java.util.ArrayList;
import java.util.List;

import www.srit.ac.in.project.mobile.encryption.self.R;
import
www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.files.AudioFilesActivity;
import
www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.files.DocumentsActivity;
import
www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.files.ImageFilesActivity;
import
www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.files.VideoFilesActivity;
import
www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.files.ZipFilesActivity;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings;

public class FilesEncDecActivity extends AppCompatActivity {
    RelativeLayout imgRv,audRv,vidRv,docsRv,zipRv;
    ImageView imgIMG,audIMG,vidIMG,docsIMG,zipIMG;
    TextView imgTv,audTv,vidTv,docsTv,zipTv;
    List<Settings> settings;
    MSFViewModel viewModel;

    @Override
    public void onBackPressed() {
        startActivity(new Intent(this,HomeActivity.class));
        finish();
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_files_enc_dec);
        settings=new ArrayList<>();
        viewModel= ViewModelProviders.of(this).get(MSFViewModel.class);

        imgRv=findViewById(R.id.images_folder_rv);
        imgIMG=findViewById(R.id.images_folder_imageView);
        imgTv=findViewById(R.id.images_folder_textView);

        audRv=findViewById(R.id.audios_folder_rv);
        audIMG=findViewById(R.id.audios_folder_imageView);
        audTv=findViewById(R.id.audios_folder_textView);

        vidRv=findViewById(R.id.videos_folder_rv);
        vidIMG=findViewById(R.id.videos_folder_imageView);
        vidTv=findViewById(R.id.videos_folder_textView);
```

```
        docsRv=findViewById(R.id.docs_folder_rv);
        docsIMG=findViewById(R.id.docs_folder_imageView);
        docsTv=findViewById(R.id.docs_folder_textView);

        zipRv=findViewById(R.id.zip_folder_rv);
        zipIMG=findViewById(R.id.zip_folder_imageView);
        zipTv=findViewById(R.id.zip_folder_textView);

    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        if (item.getItemId()==android.R.id.home){
            startActivity(new Intent(this,HomeActivity.class));
            finish();
        }
        return true;
    }

    public void openImages(View view) {
        Intent intent=new Intent(this,ImageFilesActivity.class);
        startActivity(intent);
    }

    public void openDocs(View view) {
        Intent intent=new Intent(this,DocumentsActivity.class);
        startActivity(intent);
    }

    public void openAudio(View view) {
        Intent intent=new Intent(this,AudioFilesActivity.class);
        startActivity(intent);
    }

    public void openVideos(View view) {
        Intent intent=new Intent(this,VideoFilesActivity.class);
        startActivity(intent);
    }

    public void openZip(View view) {
        Intent intent=new Intent(this, ZipFilesActivity.class);
        startActivity(intent);
    }
}
```

Image Files activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.files;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.Observer;
import androidx.lifecycle.ViewModelProvider;
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.annotation.SuppressLint;
import android.app.ProgressDialog;
import android.content.ContentResolver;
import android.content.ContentUris;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.database.Cursor;
import android.graphics.Bitmap;
import android.graphics.drawable.BitmapDrawable;
import android.graphics.drawable.Drawable;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Build;
import android.os.Bundle;
import android.os.Environment;
import android.os.Handler;
import android.provider.MediaStore;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;

import com.google.android.gms.tasks.OnSuccessListener;
import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;
import com.google.firebase.storage.UploadTask;

import java.io.BufferedInputStream;
import java.io.ByteArrayInputStream;
import java.io.ByteArrayOutputStream;
```

```
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.URL;
import java.net.URLConnection;
import java.security.InvalidAlgorithmParameterException;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Random;

import javax.crypto.NoSuchPaddingException;

import www.srit.ac.in.project.mobile.encryption.self.MainActivity;
import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.adapters.ImageAdapter;
import www.srit.ac.in.project.mobile.encryption.self.models.Link;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Image;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings;
import www.srit.ac.in.project.mobile.encryption.self.service.FileEncrypter;

import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.SECRETKEY;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_NAME;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;


public class ImageFilesActivity extends AppCompatActivity {

    private static final int PICK_IMAGE = 11;
    private RecyclerView imageRecyclerView;
    FloatingActionButton addImageBtn;
    private String encryptionKey;
    private MSFViewModel viewModel;
    List<Image> images;
    File myDir;
    private String keystr;
    private String specStr;
    private SharedPreferences preferences;

    private File decryptedFile;
    private List<Uri> imageUris = new ArrayList<>();
```

```
    private Bitmap bitmap;
    private DatabaseReference reference;
    private int currentFileNo = 0;
    private ProgressDialog progressDialog;

    private List<Link> downloadUrl = new ArrayList<>();

    @SuppressLint("WrongThread")
    @RequiresApi(api = Build.VERSION_CODES.N)
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_image_files);
        initViews();
        preferences = getSharedPreferences(USER_PREF, Context.MODE_PRIVATE);
        keystr = preferences.getString(SECRETKEY, "xtVkg0knCiDc9k80");
        specStr = preferences.getString(SECRETKEY, "BentH1dIPoOEawVa");
        Drawable drawable = getResources().getDrawable(R.drawable.default_image);
        BitmapDrawable bitmapDrawable = (BitmapDrawable) drawable;
        bitmap = bitmapDrawable.getBitmap();

        clickEvents();
    }


    @RequiresApi(api = Build.VERSION_CODES.N)
    private void initViews() {
        //Create a Folder to Store the Encrypted Images
        myDir = new File(Environment.getExternalStorageDirectory().toString() +
"/Chiper_Dossier/Images");
        if (!myDir.exists()) {
            myDir.mkdirs();
        }
        decryptedFile = new File(myDir + "/decrypt.png");
        try (FileOutputStream out = new FileOutputStream("f")) {
            bitmap.compress(Bitmap.CompressFormat.PNG, 100, out); // bmp is your Bitmap
instance
            // PNG is a lossless format, the compression factor (100) is ignored
        } catch (IOException e) {
            e.printStackTrace();
        }
        images = new ArrayList<>();
        imageRecyclerView = findViewById(R.id.imageFilesRecyclerView);
        addImageBtn = findViewById(R.id.addAnImage);
        viewModel = new ViewModelProvider(this).get(MSFViewModel.class);
        viewModel.getAllSettings().observe(this, settings -> encryptionKey =
settings.get(3).getPasskey());
        populateImageFiles();
        reference =
FirebaseDatabase.getInstance().getReference().child("user_data_files").child("images");
```

```
        ValueEventListener valueEventListener = new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                for (DataSnapshot ds : dataSnapshot.getChildren()) {
                    downloadUrl.add(ds.getValue(Link.class));
                    /*
progressDialog=ProgressDialog.show(ImageFilesActivity.this,"Downloading...","Please
Wait while Downloading is in Progress");
                    new DownLoadFile().execute(ds.getValue(Link.class).toString());*/
                }

            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {

            }
        };
        reference.addValueEventListener(valueEventListener);
    }


    private void clickEvents() {
        addImageBtn.setOnClickListener(view -> {
            pickAnImageFromGallery();
        });

    }


    private void populateImageFiles() {
        images = new ArrayList<>();
        File[] files = myDir.listFiles();
        if (files.length != 0) {
            // loops through the array of files, outputing the name to console
            for (int ii = 0; ii < files.length; ii++) {
                String fileOutput = files[ii].toString();
                imageUris.add(Uri.fromFile(files[ii]));
                images.add(new Image(files[ii].getName(), null));
            }
        }
        imageRecyclerView.setLayoutManager(new GridLayoutManager(this, 4));
        imageRecyclerView.setAdapter(new ImageAdapter(this, images));
    }


    private void pickAnImageFromGallery() {
        Intent intent = new Intent();
        intent.setType("image/*");
        intent.setAction(Intent.ACTION_GET_CONTENT);
```

```
        startActivityForResult(Intent.createChooser(intent, "Select Picture"), PICK_IMAGE);
    }


    @Override
    protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data)
{
        super.onActivityResult(requestCode, resultCode, data);
        if (requestCode == PICK_IMAGE) {
            if (resultCode == RESULT_OK) {
                ProgressDialog progressDialog = ProgressDialog.show(this, "Encrypting....",
"Please wait it take a moment to Encrypt");
                Bitmap bitmap = null;
                try {
                    bitmap = MediaStore.Images.Media.getBitmap(getContentResolver(),
data.getData());
                    encryptImage(bitmap, "File" + images.size());
                } catch (IOException e) {
                    e.printStackTrace();
                }
                populateImageFiles();
                progressDialog.dismiss();
            }
        }
    }


    public static String random() {
        Random generator = new Random();
        StringBuilder randomStringBuilder = new StringBuilder();
        int randomLength = generator.nextInt(8);
        char tempChar;
        for (int i = 0; i < randomLength; i++) {
            tempChar = (char) (generator.nextInt(96) + 32);
            randomStringBuilder.append(tempChar);
        }
        return randomStringBuilder.toString();
    }


    private void encryptImage(Bitmap bitmap, String ENC_FILE_NAME) {
        ByteArrayOutputStream stream = new ByteArrayOutputStream();
        bitmap.compress(Bitmap.CompressFormat.PNG, 100, stream);
        InputStream inputStream = new ByteArrayInputStream(stream.toByteArray());
        File out = new File(myDir, ENC_FILE_NAME);
        try {
            FileEncrypter.encryptToFile(keystr, specStr, inputStream, new
FileOutputStream(out));
            Toast.makeText(ImageFilesActivity.this, "Encrypted",
Toast.LENGTH_SHORT).show();
```

```
        } catch (IOException | NoSuchAlgorithmException | InvalidKeyException |
InvalidAlgorithmParameterException | NoSuchPaddingException e) {
            e.printStackTrace();
        }
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.sync_menu, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.sync_now:
                reference =
FirebaseDatabase.getInstance().getReference().child("user_data_files").child("images");
                StorageReference folder =
FirebaseStorage.getInstance().getReference().child("users_data").child(preferences.getStri
ng(USER_NAME, null)).child("images");
                for (int i = 0; i < imageUris.size(); i++) {
                    String fname = "File" + i;
                    StorageReference filename = folder.child(fname);
                    filename.putFile(imageUris.get(i)).addOnSuccessListener(taskSnapshot ->
filename.getDownloadUrl().addOnSuccessListener(uri -> {
                        DatabaseReference file = reference.child(fname).child("link");
                        file.setValue(String.valueOf(uri));
                    }));
                }
                Toast.makeText(this, "Uploaded", Toast.LENGTH_SHORT).show();
                break;
            case R.id.download:
                for (int i = 0; i < downloadUrl.size(); i++) {
                    currentFileNo = i;

progressDialog=ProgressDialog.show(this,"Downloading("+(i+1)+"/"+downloadUrl.size(
)+")","Please Wait while Downloading is in Progress");
                    new Handler().postDelayed(() -> {
                        new DownLoadFile().execute(downloadUrl.get(currentFileNo).getLink());
                    },2000);

                }
               // Toast.makeText(this, "Download Complete", Toast.LENGTH_SHORT).show();
                break;
            case android.R.id.home:
                finish();
                break;
        }
        return true;
```

```
}

class DownLoadFile extends AsyncTask<String, String, String> {

    @Override
    protected String doInBackground(String... strings) {

        int count;
        try {
            URL url = new URL(strings[0]);
            URLConnection conection = url.openConnection();
            conection.connect();

            // this will be useful so that you can show a tipical 0-100%
            // progress bar
            int lenghtOfFile = conection.getContentLength();

            // download the file
            InputStream input = new BufferedInputStream(url.openStream(),
                    8192);

            // Output stream
            OutputStream output = new FileOutputStream(myDir + "/File" + currentFileNo);

            byte data[] = new byte[1024];

            long total = 0;

            while ((count = input.read(data)) != -1) {
                total += count;
                // publishing the progress....
                // After this onProgressUpdate will be called
                publishProgress("" + (int) ((total * 100) / lenghtOfFile));

                // writing data to file
                output.write(data, 0, count);
            }

            // flushing output
            output.flush();

            // closing streams
            output.close();
            input.close();

        } catch (Exception e) {
            Log.e("Error: ", e.getMessage());
        }

        return null;
```

```
        }

        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            Toast.makeText(ImageFilesActivity.this, "Download Started",
Toast.LENGTH_SHORT).show();
        }

        @Override
        protected void onPostExecute(String s) {
            super.onPostExecute(s);
            progressDialog.dismiss();
        }


    }

}
```

Video Files Activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.files;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.ViewModelProvider;
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Environment;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Toast;

import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
```

```
import com.google.firebase.database.ValueEventListener;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;

import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.URL;
import java.net.URLConnection;
import java.security.InvalidAlgorithmParameterException;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;

import javax.crypto.NoSuchPaddingException;

import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.adapters.FilesAdapter;
import www.srit.ac.in.project.mobile.encryption.self.adapters.ImageAdapter;
import www.srit.ac.in.project.mobile.encryption.self.models.FileModel;
import www.srit.ac.in.project.mobile.encryption.self.models.Link;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Image;
import www.srit.ac.in.project.mobile.encryption.self.service.FileEncrypter;

import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.SECRETKEY;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_NAME;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;

public class VideoFilesActivity extends AppCompatActivity {

    private static final int VIDEO_REQ = 101;
    private RecyclerView videoRecyclerView;
    private FloatingActionButton addVideo;
    private String encryptionKey;
    private File myDir;
    private MSFViewModel viewModel;
    private String keystr ;
    private String specStr ;
    private List<FileModel> videos;
```

```
    //14-04-2020
    private SharedPreferences preferences;
    private List<Uri> videosUris;

    private List<Link> downloadUrl=new ArrayList<>();
    private DatabaseReference reference;
    private ProgressDialog progressDialog;
    private int currentFileNo;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_video_files);
        preferences=getSharedPreferences(USER_PREF, Context.MODE_PRIVATE);
        keystr=preferences.getString(SECRETKEY,null);
        specStr=preferences.getString(SECRETKEY,null);
        initViews();
        clickEvents();
        populateVideos();
    }
    private void initViews(){
        videos=new ArrayList<>();
        videoRecyclerView=findViewById(R.id.videoFilesRecyclerView);
        addVideo=findViewById(R.id.addVideo);
        myDir = new File(Environment.getExternalStorageDirectory().toString() +
"/Chiper_Dossier/Videos");
        if (!myDir.exists()) {
            myDir.mkdirs();
        }
        viewModel = new ViewModelProvider(this).get(MSFViewModel.class);
        videosUris=new ArrayList<>();
        reference=
FirebaseDatabase.getInstance().getReference().child("user_data_files").child("video");
        ValueEventListener valueEventListener=new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                for (DataSnapshot ds:dataSnapshot.getChildren()){
                    downloadUrl.add(ds.getValue(Link.class));
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {

            }
        };
        reference.addValueEventListener(valueEventListener);

    }
    private void populateVideos() {
```

```
        videos = new ArrayList<>();
        File[] files = myDir.listFiles();
        if (files.length != 0) {
            // loops through the array of files, outputing the name to console
            for (int ii = 0; ii < files.length; ii++) {
                String fileOutput = files[ii].toString();
                videosUris.add(Uri.fromFile(files[ii]));
                videos.add(new FileModel(files[ii].getName(), files[ii].getAbsolutePath()));
            }
        }
        videoRecyclerView.setLayoutManager(new GridLayoutManager(this, 4));
        videoRecyclerView.setAdapter(new FilesAdapter(this, videos,".mp4"));
    }
    public static String random() {
        Random generator = new Random();
        StringBuilder randomStringBuilder = new StringBuilder();
        int randomLength = generator.nextInt(8);
        char tempChar;
        for (int i = 0; i < randomLength; i++) {
            tempChar = (char) (generator.nextInt(96) + 32);
            randomStringBuilder.append(tempChar);
        }
        return randomStringBuilder.toString();
    }

    @Override
    protected void onRestart() {
        super.onRestart();
        populateVideos();
    }

    private void clickEvents(){
        addVideo.setOnClickListener(view -> {
            Intent intent = new Intent();
            intent.setType("video/*");
            intent.setAction(Intent.ACTION_GET_CONTENT);
            startActivityForResult(Intent.createChooser(intent, "Select an Document File"),
VIDEO_REQ);


        });
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data)
{
        super.onActivityResult(requestCode, resultCode, data);
        if (requestCode==VIDEO_REQ){
            if (resultCode==RESULT_OK){
                assert data != null;
```

```
            Uri uri = data.getData();
            InputStream inputStream = null;
            try {
                assert uri != null;
                inputStream = getContentResolver().openInputStream(uri);
            } catch (FileNotFoundException e) {
                e.printStackTrace();
            }
            File out = new File(myDir, "File"+videos.size());
            try {
                assert inputStream != null;
                FileEncrypter.encryptToFile(keystr, specStr, inputStream, new
FileOutputStream(out));
                Toast.makeText(VideoFilesActivity.this, "Encrypted",
Toast.LENGTH_SHORT).show();
            } catch (IOException | InvalidAlgorithmParameterException |
NoSuchPaddingException | InvalidKeyException | NoSuchAlgorithmException e) {
                e.printStackTrace();
            }
        }
    }
}
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.sync_menu,menu);
    return super.onCreateOptionsMenu(menu);
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.sync_now:
            reference=
FirebaseDatabase.getInstance().getReference().child("user_data_files").child("video");
            StorageReference folder=
FirebaseStorage.getInstance().getReference().child("users_data").child(preferences.getStri
ng(USER_NAME,null)).child("video");
            for (int i = 0; i< videosUris.size(); i++){
                String fname="File"+i;
                StorageReference filename=folder.child(fname);
                filename.putFile(videosUris.get(i)).addOnSuccessListener(taskSnapshot ->
filename.getDownloadUrl().addOnSuccessListener(uri -> {
                    DatabaseReference file=reference.child(fname).child("link");
                    file.setValue(String.valueOf(uri));
                }));
            }
            Toast.makeText(this, "Uploaded", Toast.LENGTH_SHORT).show();
            break;
        case R.id.download:
            for (int i = 0; i < downloadUrl.size(); i++) {
```

```
                currentFileNo = i;

progressDialog=ProgressDialog.show(this,"Downloading("+(i+1)+"/"+downloadUrl.size(
)+")","Please Wait while Downloading is in Progress");
                new DownLoadFile().execute(downloadUrl.get(i).getLink());
            }
            break;
        case android.R.id.home:
            finish();
            break;
    }
    return true;
  }
  class DownLoadFile extends AsyncTask<String, String, String> {

    @Override
    protected String doInBackground(String... strings) {

        int count;
        try {
            URL url = new URL(strings[0]);
            URLConnection conection = url.openConnection();
            conection.connect();

            // this will be useful so that you can show a tipical 0-100%
            // progress bar
            int lenghtOfFile = conection.getContentLength();

            // download the file
            InputStream input = new BufferedInputStream(url.openStream(),
                    8192);

            // Output stream
            OutputStream output = new FileOutputStream(myDir + "/File" + currentFileNo);

            byte data[] = new byte[1024];

            long total = 0;

            while ((count = input.read(data)) != -1) {
                total += count;
                // publishing the progress....
                // After this onProgressUpdate will be called
                publishProgress("" + (int) ((total * 100) / lenghtOfFile));

                // writing data to file
                output.write(data, 0, count);
            }

            // flushing output
```

```
            output.flush();

            // closing streams
            output.close();
            input.close();

        } catch (Exception e) {
            Log.e("Error: ", e.getMessage());
        }

        return null;
    }

    @Override
    protected void onPreExecute() {
        super.onPreExecute();
        Toast.makeText(VideoFilesActivity.this, "Download Started",
Toast.LENGTH_SHORT).show();
    }

    @Override
    protected void onPostExecute(String s) {
        super.onPostExecute(s);
        progressDialog.dismiss();
    }


  }
}
```

Documents Activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.files;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.ViewModelProvider;
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Environment;
```

```
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;

import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;

import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.URL;
import java.net.URLConnection;
import java.security.InvalidAlgorithmParameterException;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;

import javax.crypto.NoSuchPaddingException;

import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.adapters.FilesAdapter;
import www.srit.ac.in.project.mobile.encryption.self.models.FileModel;
import www.srit.ac.in.project.mobile.encryption.self.models.Link;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;
import www.srit.ac.in.project.mobile.encryption.self.service.FileEncrypter;

import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.SECRETKEY;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_NAME;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;

public class DocumentsActivity extends AppCompatActivity {
    private static final int DOC_REQ = 101;
    private RecyclerView docRecyclerView;
    private FloatingActionButton addDoc;
```

```
    //private String encryptionKey;
    private File myDir;
    private MSFViewModel viewModel;
    private String keystr;
    private String specStr;
    private List<FileModel> docs;

    //14-04-2020
    private SharedPreferences preferences;
    private List<Uri> docUris;

    private List<Link> downloadUrl=new ArrayList<>();
    private DatabaseReference reference;
    private ProgressDialog progressDialog;
    private int currentFileNo;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_documents);
        preferences = getSharedPreferences(USER_PREF, Context.MODE_PRIVATE);
        keystr = preferences.getString(SECRETKEY, null);
        specStr = preferences.getString(SECRETKEY, null);
        initViews();
        clickEvents();
    }

    private void initViews() {
        docs = new ArrayList<>();
        docRecyclerView = findViewById(R.id.docFilesRecyclerView);
        addDoc = findViewById(R.id.addDoc);
        myDir = new File(Environment.getExternalStorageDirectory().toString() +
"/Chiper_Dossier/Documents");

        if (!myDir.exists()) {
            myDir.mkdirs();
        }
        viewModel = new ViewModelProvider(this).get(MSFViewModel.class);
        docUris = new ArrayList<>();
        reference=
FirebaseDatabase.getInstance().getReference().child("user_data_files").child("docs");
        ValueEventListener valueEventListener=new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                for (DataSnapshot ds:dataSnapshot.getChildren()){
                    downloadUrl.add(ds.getValue(Link.class));
                }
            }

            @Override
```

```
        public void onCancelled(@NonNull DatabaseError databaseError) {

        }
    };
    reference.addValueEventListener(valueEventListener);
    populateDocs();
}

private void populateDocs() {
    docs = new ArrayList<>();
    File[] files = myDir.listFiles();
    if (files.length != 0) {
        // loops through the array of files, outputing the name to console
        for (int ii = 0; ii < files.length; ii++) {
            String fileOutput = files[ii].toString();
            docs.add(new FileModel(files[ii].getName(), files[ii].getAbsolutePath()));
            docUris.add(Uri.fromFile(files[ii]));
        }
    }
    docRecyclerView.setLayoutManager(new GridLayoutManager(this, 4));
    docRecyclerView.setAdapter(new FilesAdapter(this, docs, ".pdf"));
}

private void clickEvents() {
    addDoc.setOnClickListener(view -> {
        Intent intent = new Intent();
        intent.setType("application/pdf");
        intent.setAction(Intent.ACTION_GET_CONTENT);
        startActivityForResult(Intent.createChooser(intent, "Select an Document File"),
DOC_REQ);

    });
}

public static String random() {
    Random generator = new Random();
    StringBuilder randomStringBuilder = new StringBuilder();
    int randomLength = generator.nextInt(8);
    char tempChar;
    for (int i = 0; i < randomLength; i++) {
        tempChar = (char) (generator.nextInt(96) + 32);
        randomStringBuilder.append(tempChar);
    }
    return randomStringBuilder.toString();
}

@Override
protected void onRestart() {
    super.onRestart();
    populateDocs();
```

```
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data)
{
        super.onActivityResult(requestCode, resultCode, data);
        if (requestCode == DOC_REQ) {
            if (resultCode == RESULT_OK) {
                assert data != null;
                Uri uri = data.getData();
                InputStream inputStream = null;
                try {
                    assert uri != null;
                    inputStream = getContentResolver().openInputStream(uri);
                } catch (FileNotFoundException e) {
                    e.printStackTrace();
                }
                File out = new File(myDir, "File"+docs.size());
                try {
                    assert inputStream != null;
                    FileEncrypter.encryptToFile(keystr, specStr, inputStream, new
FileOutputStream(out));
                    Toast.makeText(DocumentsActivity.this, "Encrypted",
Toast.LENGTH_SHORT).show();
                } catch (IOException | InvalidAlgorithmParameterException |
NoSuchPaddingException | InvalidKeyException | NoSuchAlgorithmException e) {
                    e.printStackTrace();
                }
            }
        }
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.sync_menu, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.sync_now:
                reference=
FirebaseDatabase.getInstance().getReference().child("user_data_files").child("docs");
                StorageReference folder=
FirebaseStorage.getInstance().getReference().child("users_data").child(preferences.getStri
ng(USER_NAME,null)).child("docs");
                for (int i = 0; i< docUris.size(); i++){
                    String fname="File"+i;
                    StorageReference filename=folder.child(fname);
```

```
                filename.putFile(docUris.get(i)).addOnSuccessListener(taskSnapshot ->
filename.getDownloadUrl().addOnSuccessListener(uri -> {
                    DatabaseReference file=reference.child(fname).child("link");
                    file.setValue(String.valueOf(uri));
                }));
            }
            Toast.makeText(this, "Uploaded", Toast.LENGTH_SHORT).show();
            break;
        case R.id.download:
            for (int i = 0; i < downloadUrl.size(); i++) {
                currentFileNo = i;

progressDialog=ProgressDialog.show(this,"Downloading("+(i+1)+"/"+downloadUrl.size(
)+")","Please Wait while Downloading is in Progress");
                new DownLoadFile().execute(downloadUrl.get(i).getLink());
            }
            break;
        case android.R.id.home:
            finish();
            break;
    }
    return true;
  }
  class DownLoadFile extends AsyncTask<String, String, String> {

    @Override
    protected String doInBackground(String... strings) {

        int count;
        try {
            URL url = new URL(strings[0]);
            URLConnection conection = url.openConnection();
            conection.connect();

            // this will be useful so that you can show a tipical 0-100%
            // progress bar
            int lenghtOfFile = conection.getContentLength();

            // download the file
            InputStream input = new BufferedInputStream(url.openStream(),
                    8192);

            // Output stream
            OutputStream output = new FileOutputStream(myDir + "/File" + currentFileNo);

            byte data[] = new byte[1024];

            long total = 0;

            while ((count = input.read(data)) != -1) {
```

```
                total += count;
                // publishing the progress....
                // After this onProgressUpdate will be called
                publishProgress("" + (int) ((total * 100) / lenghtOfFile));

                // writing data to file
                output.write(data, 0, count);
            }

            // flushing output
            output.flush();

            // closing streams
            output.close();
            input.close();

        } catch (Exception e) {
            Log.e("Error: ", e.getMessage());
        }

        return null;
    }

    @Override
    protected void onPreExecute() {
        super.onPreExecute();
        Toast.makeText(DocumentsActivity.this, "Download Started",
Toast.LENGTH_SHORT).show();
    }

    @Override
    protected void onPostExecute(String s) {
        super.onPostExecute(s);
        progressDialog.dismiss();
    }


  }
}
```

Audio files Activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.files;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.Observer;
import androidx.lifecycle.ViewModelProvider;
```

```
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Environment;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Toast;

import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;

import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.URL;
import java.net.URLConnection;
import java.security.InvalidAlgorithmParameterException;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;

import javax.crypto.NoSuchPaddingException;

import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.adapters.FilesAdapter;
import www.srit.ac.in.project.mobile.encryption.self.adapters.ImageAdapter;
import www.srit.ac.in.project.mobile.encryption.self.models.FileModel;
import www.srit.ac.in.project.mobile.encryption.self.models.Link;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;
```

```
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Image;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings;
import www.srit.ac.in.project.mobile.encryption.self.service.FileEncrypter;

import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.SECRETKEY;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_NAME;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;

public class AudioFilesActivity extends AppCompatActivity {

    private static final int AUDIO_REQ = 101;
    private RecyclerView audioRecyclerView;
    private FloatingActionButton addAudio;
    //private String encryptionKey;
    private File myDir;
    private MSFViewModel viewModel;
    private String keystr;
    private String specStr;
    private List<FileModel> audioes;

    //14-04-2020
    private SharedPreferences preferences;
    private List<Uri> audioUris;


    private List<Link> downloadUrl = new ArrayList<>();
    private DatabaseReference reference;
    private ProgressDialog progressDialog;
    private int currentFileNo;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_audio_files);
        preferences = getSharedPreferences(USER_PREF, Context.MODE_PRIVATE);
        keystr = preferences.getString(SECRETKEY, null);
        specStr = preferences.getString(SECRETKEY, null);
        initViews();
        clickEvents();
        populateImageFiles();
    }

    private void populateImageFiles() {
        audioes = new ArrayList<>();

        File[] files = myDir.listFiles();
        if (files.length != 0) {
```

```
            // loops through the array of files, outputing the name to console
            for (int ii = 0; ii < files.length; ii++) {
                String fileOutput = files[ii].toString();
                audioUris.add(Uri.fromFile(files[ii]));
                audioes.add(new FileModel(files[ii].getName(), files[ii].getAbsolutePath()));
            }
        }
        audioRecyclerView.setLayoutManager(new GridLayoutManager(this, 4));
        audioRecyclerView.setAdapter(new FilesAdapter(this, audioes, ".mp3"));
    }

    private void initViews() {
        audioRecyclerView = findViewById(R.id.audioFilesRecyclerView);
        addAudio = findViewById(R.id.addAnAudio);
        //Create a Folder to Store the Encrypted Images
        myDir = new File(Environment.getExternalStorageDirectory().toString() +
"/Chiper_Dossier/Audio");
        if (!myDir.exists()) {
            myDir.mkdirs();
        }
        viewModel = new ViewModelProvider(this).get(MSFViewModel.class);
        audioUris = new ArrayList<>();
        reference =
FirebaseDatabase.getInstance().getReference().child("user_data_files").child("audio");
        ValueEventListener valueEventListener = new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                for (DataSnapshot ds : dataSnapshot.getChildren()) {
                    downloadUrl.add(ds.getValue(Link.class));
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {

            }
        };
        reference.addValueEventListener(valueEventListener);
    }

    private void clickEvents() {
        addAudio.setOnClickListener(view -> {
            Intent intent = new Intent();
            intent.setType("audio/*");
            intent.setAction(Intent.ACTION_GET_CONTENT);
            startActivityForResult(Intent.createChooser(intent, "Select an Audio File"),
AUDIO_REQ);
        });
    }
```

```
    public static String random() {
        Random generator = new Random();
        StringBuilder randomStringBuilder = new StringBuilder();
        int randomLength = generator.nextInt(8);
        char tempChar;
        for (int i = 0; i < randomLength; i++) {
            tempChar = (char) (generator.nextInt(96) + 32);
            randomStringBuilder.append(tempChar);
        }
        return randomStringBuilder.toString();
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data)
{
        super.onActivityResult(requestCode, resultCode, data);
        if (requestCode == AUDIO_REQ) {
            if (resultCode == RESULT_OK) {
                assert data != null;
                Uri uri = data.getData();
                InputStream inputStream = null;
                try {
                    assert uri != null;
                    inputStream = getContentResolver().openInputStream(uri);
                } catch (FileNotFoundException e) {
                    e.printStackTrace();
                }
                File out = new File(myDir, "File" + audioes.size());
                try {
                    assert inputStream != null;
                    FileEncrypter.encryptToFile(keystr, specStr, inputStream, new
FileOutputStream(out));
                    Toast.makeText(AudioFilesActivity.this, "Encrypted",
Toast.LENGTH_SHORT).show();
                } catch (IOException | InvalidAlgorithmParameterException |
NoSuchPaddingException | InvalidKeyException | NoSuchAlgorithmException e) {
                    e.printStackTrace();
                }
                populateImageFiles();
            }
        }
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.sync_menu, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
```

```
    public boolean onOptionsItemSelected(MenuItem item) {

        switch (item.getItemId()) {
            case R.id.sync_now:
                reference =
FirebaseDatabase.getInstance().getReference().child("user_data_files").child("audio");
                StorageReference folder =
FirebaseStorage.getInstance().getReference().child("users_data").child(preferences.getStri
ng(USER_NAME, null)).child("audio");
                for (int i = 0; i < audioUris.size(); i++) {
                    String fname = "File" + i;
                    StorageReference filename = folder.child(fname);
                    filename.putFile(audioUris.get(i)).addOnSuccessListener(taskSnapshot ->
filename.getDownloadUrl().addOnSuccessListener(uri -> {
                        DatabaseReference file = reference.child(fname).child("link");
                        file.setValue(String.valueOf(uri));
                    }));
                }
                Toast.makeText(this, "Uploaded", Toast.LENGTH_SHORT).show();
                break;
            case R.id.download:
                for (int i = 0; i < downloadUrl.size(); i++) {
                    currentFileNo = i;
                    progressDialog = ProgressDialog.show(this, "Downloading(" + (i + 1) + "/" +
downloadUrl.size() + ")", "Please Wait while Downloading is in Progress");
                    new DownLoadFile().execute(downloadUrl.get(i).getLink());
                }
                Toast.makeText(this, "Downloaded", Toast.LENGTH_SHORT).show();
                break;
            case android.R.id.home:
                finish();
                break;
        }
        return true;
    }

    class DownLoadFile extends AsyncTask<String, String, String> {

        @Override
        protected String doInBackground(String... strings) {

            int count;
            try {
                URL url = new URL(strings[0]);
                URLConnection conection = url.openConnection();
                conection.connect();

                // this will be useful so that you can show a tipical 0-100%
                // progress bar
                int lenghtOfFile = conection.getContentLength();
```

```
            // download the file
            InputStream input = new BufferedInputStream(url.openStream(),
                    8192);

            // Output stream
            OutputStream output = new FileOutputStream(myDir + "/File" + currentFileNo);

            byte data[] = new byte[1024];

            long total = 0;

            while ((count = input.read(data)) != -1) {
                total += count;
                // publishing the progress....
                // After this onProgressUpdate will be called
                publishProgress("" + (int) ((total * 100) / lenghtOfFile));

                // writing data to file
                output.write(data, 0, count);
            }

            // flushing output
            output.flush();

            // closing streams
            output.close();
            input.close();

        } catch (Exception e) {
            Log.e("Error: ", e.getMessage());
        }

        return null;
    }

    @Override
    protected void onPreExecute() {
        super.onPreExecute();
        Toast.makeText(AudioFilesActivity.this, "Download Started",
Toast.LENGTH_SHORT).show();
    }

    @Override
    protected void onPostExecute(String s) {
        super.onPostExecute(s);
        progressDialog.dismiss();
    }
```

```
    }
}
```

Zip files Activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities.subactivities.files;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.ViewModelProvider;
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Environment;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Toast;

import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;

import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.URL;
import java.net.URLConnection;
import java.security.InvalidAlgorithmParameterException;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
```

```
import java.util.ArrayList;
import java.util.List;
import java.util.Random;

import javax.crypto.NoSuchPaddingException;

import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.adapters.FilesAdapter;
import www.srit.ac.in.project.mobile.encryption.self.adapters.ImageAdapter;
import www.srit.ac.in.project.mobile.encryption.self.models.FileModel;
import www.srit.ac.in.project.mobile.encryption.self.models.Link;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Image;
import www.srit.ac.in.project.mobile.encryption.self.service.FileEncrypter;

import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.SECRETKEY;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_NAME;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;

public class ZipFilesActivity extends AppCompatActivity {

    private static final int ZIP_REQ = 101;
    private RecyclerView zipRecyclerView;
    private FloatingActionButton addZipFile;
    private String encryptionKey;
    private File myDir;
    private MSFViewModel viewModel;
    private String keystr ;
    private String specStr ;
    private List<FileModel> zipFiles;

    //14-04-2020
    private SharedPreferences preferences;
    private List<Uri> zipUris;

    private List<Link> downloadUrl=new ArrayList<>();
    private DatabaseReference reference;
    private ProgressDialog progressDialog;
    private int currentFileNo;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_zip_files);
        preferences=getSharedPreferences(USER_PREF, Context.MODE_PRIVATE);
        keystr=preferences.getString(SECRETKEY,null);
        specStr=preferences.getString(SECRETKEY,null);
```

```
        initViews();
        clickEvents();
        populateZipFiles();
    }
    private void initViews(){
        zipFiles=new ArrayList<>();
        zipUris=new ArrayList<>();
        zipRecyclerView=findViewById(R.id.zipFilesRecyclerView);
        addZipFile=findViewById(R.id.addZipFile);
        myDir = new File(Environment.getExternalStorageDirectory().toString() +
"/Chiper_Dossier/Zip");

        if (!myDir.exists()) {
            myDir.mkdirs();
        }
        viewModel = new ViewModelProvider(this).get(MSFViewModel.class);
        viewModel.getAllSettings().observe(this, settings -> encryptionKey =
settings.get(3).getPasskey());
        reference=
FirebaseDatabase.getInstance().getReference().child("user_data_files").child("zip");
        ValueEventListener valueEventListener=new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                for (DataSnapshot ds:dataSnapshot.getChildren()){
                    downloadUrl.add(ds.getValue(Link.class));
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {

            }
        };
        reference.addValueEventListener(valueEventListener);

    }
    private void clickEvents(){
        addZipFile.setOnClickListener(view -> {
            Intent intent = new Intent();
            intent.setType("application/*");
            intent.setAction(Intent.ACTION_GET_CONTENT);
            startActivityForResult(Intent.createChooser(intent, "Select an Document File"),
ZIP_REQ);
        });

    }
    private void populateZipFiles() {
        zipFiles = new ArrayList<>();
        File[] files = myDir.listFiles();
        if (files.length != 0) {
```

```
            // loops through the array of files, outputing the name to console
            for (int ii = 0; ii < files.length; ii++) {
                String fileOutput = files[ii].toString();
                zipUris.add(Uri.fromFile(files[ii]));
                zipFiles.add(new FileModel(files[ii].getName(), files[ii].getAbsolutePath()));
            }
        }
        zipRecyclerView.setLayoutManager(new GridLayoutManager(this, 4));
        zipRecyclerView.setAdapter(new FilesAdapter(this, zipFiles,".zip"));
    }
    public static String random() {
        Random generator = new Random();
        StringBuilder randomStringBuilder = new StringBuilder();
        int randomLength = generator.nextInt(8);
        char tempChar;
        for (int i = 0; i < randomLength; i++) {
            tempChar = (char) (generator.nextInt(96) + 32);
            randomStringBuilder.append(tempChar);
        }
        return randomStringBuilder.toString();
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data)
{
        super.onActivityResult(requestCode, resultCode, data);
        if (requestCode==ZIP_REQ){
            if (resultCode==RESULT_OK){
                assert data != null;
                Uri uri = data.getData();
                InputStream inputStream = null;
                try {
                    assert uri != null;
                    inputStream = getContentResolver().openInputStream(uri);
                } catch (FileNotFoundException e) {
                    e.printStackTrace();
                }
                File out = new File(myDir, "File"+zipFiles.size());
                try {
                    assert inputStream != null;
                    FileEncrypter.encryptToFile(keystr, specStr, inputStream, new
FileOutputStream(out));
                    Toast.makeText(ZipFilesActivity.this, "Encrypted",
Toast.LENGTH_SHORT).show();
                } catch (IOException | InvalidAlgorithmParameterException |
NoSuchPaddingException | InvalidKeyException | NoSuchAlgorithmException e) {
                    e.printStackTrace();
                }
            }
        }
```

```
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.sync_menu,menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.sync_now:
                reference=
FirebaseDatabase.getInstance().getReference().child("user_data_files").child("zip");
                StorageReference folder=
FirebaseStorage.getInstance().getReference().child("users_data").child(preferences.getStri
ng(USER_NAME,null)).child("zip");
                for (int i = 0; i< zipUris.size(); i++){
                    String fname="File"+i;
                    StorageReference filename=folder.child(fname);
                    filename.putFile(zipUris.get(i)).addOnSuccessListener(taskSnapshot ->
filename.getDownloadUrl().addOnSuccessListener(uri -> {
                        DatabaseReference file=reference.child(fname).child("link");
                        file.setValue(String.valueOf(uri));
                    }));
                }
                Toast.makeText(this, "Uploaded", Toast.LENGTH_SHORT).show();
                break;
            case R.id.download:
                for (int i = 0; i < downloadUrl.size(); i++) {
                    currentFileNo = i;

progressDialog=ProgressDialog.show(this,"Downloading("+(i+1)+"/"+downloadUrl.size(
)+")","Please Wait while Downloading is in Progress");
                    new DownLoadFile().execute(downloadUrl.get(i).getLink());
                }
                break;
            case android.R.id.home:
                finish();
                break;
        }
        return true;
    }
    class DownLoadFile extends AsyncTask<String, String, String> {

        @Override
        protected String doInBackground(String... strings) {

            int count;
            try {
```

```
            URL url = new URL(strings[0]);
            URLConnection conection = url.openConnection();
            conection.connect();

            // this will be useful so that you can show a tipical 0-100%
            // progress bar
            int lenghtOfFile = conection.getContentLength();

            // download the file
            InputStream input = new BufferedInputStream(url.openStream(),
                    8192);

            // Output stream
            OutputStream output = new FileOutputStream(myDir + "/File" + currentFileNo);

            byte data[] = new byte[1024];

            long total = 0;

            while ((count = input.read(data)) != -1) {
                total += count;
                // publishing the progress....
                // After this onProgressUpdate will be called
                publishProgress("" + (int) ((total * 100) / lenghtOfFile));

                // writing data to file
                output.write(data, 0, count);
            }

            // flushing output
            output.flush();

            // closing streams
            output.close();
            input.close();

        } catch (Exception e) {
            Log.e("Error: ", e.getMessage());
        }

        return null;
    }

    @Override
    protected void onPreExecute() {
        super.onPreExecute();
        Toast.makeText(ZipFilesActivity.this, "Download Started",
Toast.LENGTH_SHORT).show();
    }
```

```
    @Override
    protected void onPostExecute(String s) {
        super.onPostExecute(s);
        progressDialog.dismiss();
    }


  }
}
```

Settings Activity:

```
package www.srit.ac.in.project.mobile.encryption.self.activities.subactivities;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.Observer;
import androidx.lifecycle.ViewModelProviders;

import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.List;

import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.activities.HomeActivity;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Contact;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings;

public class SettingsActivity extends AppCompatActivity {

    CheckBox contactsSwitch, messagesSwitch, notesSwitch, filesSwitch;
    private MSFViewModel viewModel;
    private boolean cState, mState, nState, fState;
    private String tarcpass, tarmpass, tarnpass, tarfpass;
    List<Settings> mySettings=new ArrayList<>();
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_settings);
    contactsSwitch = findViewById(R.id.contacts_switch);
    messagesSwitch = findViewById(R.id.messages_switch);
    notesSwitch = findViewById(R.id.notes_switch);
    filesSwitch = findViewById(R.id.files_switch);
    setTitle("Settings");

    viewModel = ViewModelProviders.of(this).get(MSFViewModel.class);

    viewModel.getAllSettings().observe(this, settings -> {
        mySettings=new ArrayList<>();
        mySettings=settings;
        cState = settings.get(0).isState();
        tarcpass = settings.get(0).getPasskey();
        mState = settings.get(1).isState();
        tarmpass = settings.get(1).getPasskey();
        nState = settings.get(2).isState();
        tarnpass = settings.get(2).getPasskey();
        fState = settings.get(3).isState();
        tarfpass = settings.get(3).getPasskey();
        setChanges();

    });

    contactsSwitch.setOnCheckedChangeListener((compoundButton, b) -> {
        cState = b;
        viewModel.updateSetting(new Settings(0, "CONTACTS", tarcpass, cState));
        //finish();
    });
    messagesSwitch.setOnCheckedChangeListener((compoundButton, b) -> {
        mState = b;
        viewModel.updateSetting(new Settings(1, "MESSAGES", tarmpass, mState));
        //finish();
    });
    notesSwitch.setOnCheckedChangeListener((compoundButton, b) -> {
        nState = b;
        viewModel.updateSetting(new Settings(2, "NOTES", tarnpass, nState));
        //finish();
    });
    filesSwitch.setOnCheckedChangeListener((compoundButton, b) -> {
        fState = b;
        viewModel.updateSetting(new Settings(3, "FILES", tarfpass, fState));
        //finish();
    });
}
private void setChanges(){
```

```
        cState=mySettings.get(0).isState();
        mState=mySettings.get(1).isState();
        nState=mySettings.get(2).isState();
        fState=mySettings.get(3).isState();
        contactsSwitch.setChecked(cState);
        messagesSwitch.setChecked(mState);
        notesSwitch.setChecked(nState);
        filesSwitch.setChecked(fState);

    }

    @Override
    public void onBackPressed() {
        startActivity(new Intent(this, HomeActivity.class));
        finish();
    }

    public void setPasswordForContacts(View view) {
        LayoutInflater inflater = getLayoutInflater();
        View alertView = inflater.inflate(R.layout.change_password, null);
        final EditText pass = alertView.findViewById(R.id.old_password_edt);
        final EditText newpass = alertView.findViewById(R.id.new_password);
        final EditText confirmpass = alertView.findViewById(R.id.confirm_password);
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle("Change Password")
                .setView(alertView)
                .setPositiveButton("CHANGE", (dialogInterface, i) -> {
                    String oldpas = pass.getText().toString();
                    String newpas = newpass.getText().toString();
                    String conpas = confirmpass.getText().toString();
                    if (oldpas.matches("")) {
                        Toast.makeText(SettingsActivity.this, "Please Enter Valid Password",
Toast.LENGTH_SHORT).show();
                    } else if (!newpas.matches(conpas) || newpas.matches("") || conpas.matches(""))
{
                        Toast.makeText(SettingsActivity.this, "Please Enter Valid Passwords",
Toast.LENGTH_SHORT).show();
                    } else if (!oldpas.matches(tarcpass)) {
                        Toast.makeText(SettingsActivity.this, "Sorry, Passwords didn't match",
Toast.LENGTH_SHORT).show();
                        pass.setText("");
                        newpass.setText("");
                        confirmpass.setText("");
                    } else if (newpas.matches(conpas)) {
                        tarcpass = newpas;
                        viewModel.insertSetting(new Settings(0, "CONTACTS", newpas, cState));
                        Toast.makeText(SettingsActivity.this, "Passwords Changed Successfully",
Toast.LENGTH_SHORT).show();
                        dialogInterface.dismiss();
                    }
```

```
            })
            .setNegativeButton("CANCEL", (dialogInterface, i) ->
dialogInterface.dismiss()).create().show();

    }

    public void setPasswordForMessages(View view) {
        LayoutInflater inflater = getLayoutInflater();
        View alertView = inflater.inflate(R.layout.change_password, null);
        final EditText pass = alertView.findViewById(R.id.old_password_edt);
        final EditText newpass = alertView.findViewById(R.id.new_password);
        final EditText confirmpass = alertView.findViewById(R.id.confirm_password);
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle("Change Password")
                .setView(alertView)
                .setPositiveButton("CHANGE", (dialogInterface, i) -> {
                    String oldpas = pass.getText().toString();
                    String newpas = newpass.getText().toString();
                    String conpas = confirmpass.getText().toString();
                    if (oldpas.matches("")) {
                        Toast.makeText(SettingsActivity.this, "Please Enter Valid Password",
Toast.LENGTH_SHORT).show();
                    } else if (!newpas.matches(conpas) || newpas.matches("") || conpas.matches(""))
{
                        Toast.makeText(SettingsActivity.this, "Please Enter Valid Passwords",
Toast.LENGTH_SHORT).show();
                    } else if (!oldpas.matches(tarmpass)) {
                        Toast.makeText(SettingsActivity.this, "Sorry, Passwords didn't match",
Toast.LENGTH_SHORT).show();
                        pass.setText("");
                        newpass.setText("");
                        confirmpass.setText("");
                    } else if (newpas.matches(conpas)) {
                        viewModel.insertSetting(new Settings(1, "MESSAGES", newpas, mState));
                        Toast.makeText(SettingsActivity.this, "Passwords Changed Successfully",
Toast.LENGTH_SHORT).show();
                        dialogInterface.dismiss();
                    }


                })
                .setNegativeButton("CANCEL", (dialogInterface, i) ->
dialogInterface.dismiss()).create().show();
    }

    public void setPasswordForNotes(View view) {
        LayoutInflater inflater = getLayoutInflater();
        View alertView = inflater.inflate(R.layout.change_password, null);
```

```
        final EditText pass = alertView.findViewById(R.id.old_password_edt);
        final EditText newpass = alertView.findViewById(R.id.new_password);
        final EditText confirmpass = alertView.findViewById(R.id.confirm_password);
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle("Change Password")
                .setView(alertView)
                .setPositiveButton("CHANGE", (dialogInterface, i) -> {
                    String oldpas = pass.getText().toString();
                    String newpas = newpass.getText().toString();
                    String conpas = confirmpass.getText().toString();
                    if (oldpas.matches("")) {
                        Toast.makeText(SettingsActivity.this, "Please Enter Valid Password",
Toast.LENGTH_SHORT).show();
                    } else if (!newpas.matches(conpas) || newpas.matches("") || conpas.matches(""))
{
                        Toast.makeText(SettingsActivity.this, "Please Enter Valid Passwords",
Toast.LENGTH_SHORT).show();
                    } else if (!oldpas.matches(tarnpass)) {
                        Toast.makeText(SettingsActivity.this, "Sorry, Passwords didn't match",
Toast.LENGTH_SHORT).show();
                        pass.setText("");
                        newpass.setText("");
                        confirmpass.setText("");
                    } else if (newpas.matches(conpas)) {
                        viewModel.insertSetting(new Settings(2, "NOTES", newpas, nState));
                        Toast.makeText(SettingsActivity.this, "Passwords Changed Successfully",
Toast.LENGTH_SHORT).show();
                        dialogInterface.dismiss();
                    }
                })
                .setNegativeButton("CANCEL", (dialogInterface, i) ->
dialogInterface.dismiss()).create().show();
    }


    public void setPasswordForFiles(View view) {
        LayoutInflater inflater = getLayoutInflater();
        View alertView = inflater.inflate(R.layout.change_password, null);
        final EditText pass = alertView.findViewById(R.id.old_password_edt);
        final EditText newpass = alertView.findViewById(R.id.new_password);
        final EditText confirmpass = alertView.findViewById(R.id.confirm_password);
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle("Change Password")
                .setView(alertView)
                .setPositiveButton("CHANGE", (dialogInterface, i) -> {
                    String oldpas = pass.getText().toString();
                    String newpas = newpass.getText().toString();
                    String conpas = confirmpass.getText().toString();
                    if (oldpas.matches("")) {
                        Toast.makeText(SettingsActivity.this, "Please Enter Valid Password",
```

```
Toast.LENGTH_SHORT).show();
                } else if (!newpas.matches(conpas) || newpas.matches("") || conpas.matches(""))
{
                    Toast.makeText(SettingsActivity.this, "Please Enter Valid Passwords",
Toast.LENGTH_SHORT).show();
                } else if (!oldpas.matches(tarfpass)) {
                    Toast.makeText(SettingsActivity.this, "Sorry, Passwords didn't match",
Toast.LENGTH_SHORT).show();
                    pass.setText("");
                    newpass.setText("");
                    confirmpass.setText("");
                } else if (newpas.matches(conpas)) {
                    viewModel.insertSetting(new Settings(3, "FILES", newpas, cState));
                    Toast.makeText(SettingsActivity.this, "Passwords Changed Successfully",
Toast.LENGTH_SHORT).show();
                    dialogInterface.dismiss();
                }
            })
            .setNegativeButton("CANCEL", (dialogInterface, i) ->
dialogInterface.dismiss()).create().show();
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        if (item.getItemId()==android.R.id.home){
            startActivity(new Intent(this,HomeActivity.class));
            finish();
        }
        return true;
    }
}
```

Contacts Adapter:

```
package www.srit.ac.in.project.mobile.encryption.self.adapters;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageButton;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.ViewModelProviders;
import androidx.recyclerview.widget.RecyclerView;
```

```
import com.tomash.androidcontacts.contactgetter.entity.ContactData;
import com.tomash.androidcontacts.contactgetter.entity.PhoneNumber;
import com.tomash.androidcontacts.contactgetter.main.ContactDataFactory;
import com.tomash.androidcontacts.contactgetter.main.contactsSaver.ContactsSaverBuilder;

import java.util.ArrayList;
import java.util.List;

import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.room.MSFViewModel;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Contact;

public class ContactsAdapter extends
RecyclerView.Adapter<ContactsAdapter.ViewHolderClass> {
    private Context context;
    private List<Contact> contactList;

    public ContactsAdapter(Context context, List<Contact> contactList) {
        this.context = context;
        this.contactList = contactList;
    }

    @Override
    public ViewHolderClass onCreateViewHolder(ViewGroup parent, int viewType) {
        return new
ViewHolderClass(LayoutInflater.from(context).inflate(R.layout.contacts_item, parent,
false));
    }

    @Override
    public void onBindViewHolder(@NonNull ViewHolderClass holder, int position) {
        if (contactList.get(position) != null) {
            holder.contactNameTv.setText(contactList.get(position).getCaller_name());
            holder.contactNumberTv.setText(contactList.get(position).getCaller_phone());
        }
        holder.deleteBtn.setOnClickListener(view -> {
            deleteContact(contactList.get(position));
            notifyDataSetChanged();

        });

    }

    @Override
    public int getItemCount() {
        if (contactList != null) {
            return contactList.size();
        } else {
            return 0;
        }
```

```
    }

    public class ViewHolderClass extends RecyclerView.ViewHolder  {
        TextView contactNameTv, contactNumberTv;
        private ImageButton deleteBtn;


        public ViewHolderClass(View itemView) {
            super(itemView);
            contactNameTv = itemView.findViewById(R.id.contact_name);
            contactNumberTv = itemView.findViewById(R.id.contact_number);
            deleteBtn = itemView.findViewById(R.id.contact_delete_imageView);
        }

    }


    private void deleteContact(Contact contact) {
        MSFViewModel viewModel =
ViewModelProviders.of((AppCompatActivity)context).get(MSFViewModel.class);
        viewModel.deleteContact(contact);

        ContactData contactData = ContactDataFactory.createEmpty();
        contactData.setCompositeName(contact.getCaller_name());
        List<PhoneNumber> phones = new ArrayList<>();
        phones.add(new PhoneNumber(context, contact.getCaller_phone()));
        contactData.setPhoneList(phones);
        int id = new ContactsSaverBuilder(context)
                .saveContact(contactData);
        Toast.makeText(context, "Contact is Restored Successfully",
Toast.LENGTH_SHORT).show();
    }

}
```

Files Adapter:

```
package www.srit.ac.in.project.mobile.encryption.self.adapters;

import android.content.Context;
import android.content.SharedPreferences;
import android.os.Environment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import android.widget.Toast;
```

```
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.security.InvalidAlgorithmParameterException;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.util.List;

import javax.crypto.NoSuchPaddingException;

import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.models.FileModel;
import www.srit.ac.in.project.mobile.encryption.self.service.FileEncrypter;

import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.SECRETKEY;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;

public class FilesAdapter extends RecyclerView.Adapter<FilesAdapter.ViewHolderClass>
{
    private Context context;
    private List<FileModel> filesNames;
    private String fileType;

    public FilesAdapter(Context context, List<FileModel> filesNames, String fileType) {
        this.context = context;
        this.filesNames = filesNames;
        this.fileType = fileType;
    }

    @NonNull
    @Override
    public ViewHolderClass onCreateViewHolder(@NonNull ViewGroup parent, int
viewType) {
        return new
ViewHolderClass(LayoutInflater.from(context).inflate(R.layout.image_item, parent, false));
    }

    @Override
    public void onBindViewHolder(@NonNull ViewHolderClass holder, int position) {
        holder.nameTv.setText(filesNames.get(position).getFilename());

    }
```

```
    @Override
    public int getItemCount() {
        if (filesNames != null) {
            return filesNames.size();
        } else {
            return 0;
        }
    }

    public class ViewHolderClass extends RecyclerView.ViewHolder implements
View.OnClickListener {
        TextView nameTv;

        public ViewHolderClass(@NonNull View itemView) {
            super(itemView);
            nameTv = itemView.findViewById(R.id.image_item_textView);
            itemView.setOnClickListener(this);

        }

        @Override
        public void onClick(View view) {
            int position = getAdapterPosition();
            String filename = filesNames.get(position).getFilename();
            File myDir;
            if (fileType.equals(".pdf")) {
              myDir = new File(Environment.getExternalStorageDirectory().toString() +
"/Chiper_Dossier/Documents");
            }else if (fileType.equals(".mp3")){
                myDir = new File(Environment.getExternalStorageDirectory().toString() +
"/Chiper_Dossier/Audio");
            }else if (fileType.equals(".mp4")){
                myDir = new File(Environment.getExternalStorageDirectory().toString() +
"/Chiper_Dossier/Videos");
            }else{
                myDir = new File(Environment.getExternalStorageDirectory().toString() +
"/Chiper_Dossier/Zip");
            }
            File encFile = new File(myDir, filename);
            File decFile = new File(myDir, "decrypt" + fileType);
            SharedPreferences preferences = context.getSharedPreferences(USER_PREF,
Context.MODE_PRIVATE);
            String key = preferences.getString(SECRETKEY, null);
            try {
                if (!filename.endsWith(fileType)) {
                    FileEncrypter.decryptToFile(key, key, new FileInputStream(encFile), new
FileOutputStream(decFile));
                    Toast.makeText(context, "Decrypted", Toast.LENGTH_SHORT).show();
                }
            } catch (IOException e) {
```

```
            e.printStackTrace();
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        } catch (InvalidKeyException e) {
            e.printStackTrace();
        } catch (InvalidAlgorithmParameterException e) {
            e.printStackTrace();
        } catch (NoSuchPaddingException e) {
            e.printStackTrace();
        }

    }
  }
}
```

Image Adapter:

```
package www.srit.ac.in.project.mobile.encryption.self.adapters;

import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.net.Uri;
import android.os.Environment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.security.InvalidAlgorithmParameterException;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.util.List;

import javax.crypto.NoSuchPaddingException;

import www.srit.ac.in.project.mobile.encryption.self.R;
import www.srit.ac.in.project.mobile.encryption.self.activities.ImagesActivity;
```

```
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Image;
import www.srit.ac.in.project.mobile.encryption.self.service.FileEncrypter;

import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.SECRETKEY;
import static
www.srit.ac.in.project.mobile.encryption.self.constants.KeyConstants.USER_PREF;

public class ImageAdapter extends
RecyclerView.Adapter<ImageAdapter.ViewHolderClass> {

   private Context context;
   private List<Image> images;

   public ImageAdapter(Context context, List<Image> images) {
      this.context = context;
      this.images = images;
   }

   @NonNull
   @Override
   public ViewHolderClass onCreateViewHolder(@NonNull ViewGroup parent, int
viewType) {
      return new
ViewHolderClass(LayoutInflater.from(context).inflate(R.layout.image_item,parent,false));
   }

   @Override
   public void onBindViewHolder(@NonNull ViewHolderClass holder, int position) {
      holder.nameTv.setText(images.get(position).getImageName());


   }

   @Override
   public int getItemCount() {
      if (images!=null){
         return images.size();
      }
      return 0;
   }

   public class ViewHolderClass extends RecyclerView.ViewHolder implements
View.OnClickListener {
      TextView nameTv;
      ImageView imageView;

      public ViewHolderClass(@NonNull View itemView) {
         super(itemView);
         imageView=itemView.findViewById(R.id.image_item_imageView);
```

```
            nameTv=itemView.findViewById(R.id.image_item_textView);
            itemView.setOnClickListener(this);
        }

        @Override
        public void onClick(View view) {
            String filename=images.get(getAdapterPosition()).getImageName();
            SharedPreferences
preferences=context.getSharedPreferences(USER_PREF,Context.MODE_PRIVATE);
            String key=preferences.getString(SECRETKEY,null);
            File myDir=new
File(Environment.getExternalStorageDirectory().toString()+"/Chiper_Dossier/Images");
            if (!myDir.exists()){
                myDir.mkdirs();
            }
            File  decryptedFile= new File(myDir ,"decrypt.png");
            File encFile=new File(myDir,filename);
            try {
                if (filename.endsWith(".png")){
                    context.startActivity(new Intent(context, ImagesActivity.class));
                }else {
                    FileEncrypter.decryptToFile(key, key, new FileInputStream(encFile), new
FileOutputStream(decryptedFile));
                    Toast.makeText(context, "Decrypted", Toast.LENGTH_SHORT).show();
                    context.startActivity(new Intent(context, ImagesActivity.class));
                }
            } catch (IOException e) {
                e.printStackTrace();
            } catch (NoSuchAlgorithmException e) {
                e.printStackTrace();
            } catch (InvalidKeyException e) {
                e.printStackTrace();
            } catch (InvalidAlgorithmParameterException e) {
                e.printStackTrace();
            } catch (NoSuchPaddingException e) {
                e.printStackTrace();
            }
        }
    }


}
```

File Model:

```
package www.srit.ac.in.project.mobile.encryption.self.models;

public class FileModel {
```

```
    private String filename;
    private String filePath;

    public FileModel(String filename, String filePath) {
        this.filename = filename;
        this.filePath = filePath;
    }

    public String getFilename() {
        return filename;
    }

    public void setFilename(String filename) {
        this.filename = filename;
    }

    public String getFilePath() {
        return filePath;
    }

    public void setFilePath(String filePath) {
        this.filePath = filePath;
    }
}
```

Edit Text:

```
package www.srit.ac.in.project.mobile.encryption.self.models;

import android.annotation.SuppressLint;
import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Paint;
import android.graphics.Rect;
import android.util.AttributeSet;
import android.widget.EditText;

import www.srit.ac.in.project.mobile.encryption.self.R;


@SuppressLint("AppCompatCustomView")
public class LinedEditText extends EditText {

    Rect rect= new Rect();
    Paint paint=new Paint();

    public LinedEditText(Context context, AttributeSet attrs) {
        super(context, attrs);
```

```
    }


    @Override
    protected void onDraw(Canvas canvas) {
        paint.setStyle(Paint.Style.STROKE);
        paint.setColor(getResources().getColor(R.color.text_color_light));
        int height=getHeight();
        int lineheight=getLineHeight();
        int count=height/lineheight;
        if(lineheight>count){
            count=lineheight;
        }
        int linebounds=getLineBounds(0,rect);
        for (int i=1;i<count;i++){
            canvas.drawLine(rect.left,linebounds+1,rect.right,linebounds+1,paint);
            linebounds+=lineheight;
        }
        super.onDraw(canvas);
    }
}
```

ROOM:

```
package www.srit.ac.in.project.mobile.encryption.self.room.tables;

import androidx.room.ColumnInfo;
import androidx.room.Entity;
import androidx.room.PrimaryKey;
@Entity
public class Contact {
    @PrimaryKey(autoGenerate = true)
    @ColumnInfo(name = "caller_row_id")
    private int id;

    @ColumnInfo(name = "caller_id")
    private String caller_id;

    @ColumnInfo(name = "caller_name")
    private String caller_name;

    @ColumnInfo(name = "caller_phone")
    private String caller_phone;

    @ColumnInfo(name = "caller_email")
    private String caller_email;
```

```
    public Contact(){

    }
    public Contact(String caller_id, String caller_name, String caller_phone, String
caller_email) {
        this.caller_id = caller_id;
        this.caller_name = caller_name;
        this.caller_phone = caller_phone;
        this.caller_email = caller_email;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getCaller_id() {
        return caller_id;
    }

    public void setCaller_id(String caller_id) {
        this.caller_id = caller_id;
    }

    public String getCaller_name() {
        return caller_name;
    }

    public void setCaller_name(String caller_name) {
        this.caller_name = caller_name;
    }

    public String getCaller_phone() {
        return caller_phone;
    }

    public void setCaller_phone(String caller_phone) {
        this.caller_phone = caller_phone;
    }

    public String getCaller_email() {
        return caller_email;
    }

    public void setCaller_email(String caller_email) {
        this.caller_email = caller_email;
```

```
    }
}
```

```
package www.srit.ac.in.project.mobile.encryption.self.room.tables;

import androidx.annotation.NonNull;
import androidx.room.Entity;
import androidx.room.PrimaryKey;
import androidx.room.Query;

@Entity
public class Image {
    @NonNull
    @PrimaryKey(autoGenerate = true)
    private int id;
    private String imageName;
    private String imageUri;

    public Image(String imageName, String imageUri) {
        this.imageName = imageName;
        this.imageUri = imageUri;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getImageName() {
        return imageName;
    }

    public void setImageName(String imageName) {
        this.imageName = imageName;
    }

    public String getImageUri() {
        return imageUri;
    }

    public void setImageUri(String imageUri) {
        this.imageUri = imageUri;
    }
}
```

```
package www.srit.ac.in.project.mobile.encryption.self.room.tables;
```

```
import androidx.room.ColumnInfo;
import androidx.room.Entity;
import androidx.room.PrimaryKey;
@Entity
public class Message {
    @PrimaryKey(autoGenerate = true)
    @ColumnInfo(name = "message_id")
    private int id;
    @ColumnInfo(name = "sender_phone_no")
    private String senderphone;
    @ColumnInfo(name = "sender_name")
    private String sendername;
    @ColumnInfo(name = "message_time_stamp")
    private String messageTime;
    @ColumnInfo(name = "message_body")
    private String messageBody;

    public Message(){

    }
    public Message( String senderphone, String sendername, String messageTime, String
messageBody) {
        this.senderphone = senderphone;
        this.sendername = sendername;
        this.messageTime = messageTime;
        this.messageBody = messageBody;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getSenderphone() {
        return senderphone;
    }

    public void setSenderphone(String senderphone) {
        this.senderphone = senderphone;
    }

    public String getSendername() {
        return sendername;
    }

    public void setSendername(String sendername) {
        this.sendername = sendername;
```

```
    }

    public String getMessageTime() {
        return messageTime;
    }

    public void setMessageTime(String messageTime) {
        this.messageTime = messageTime;
    }

    public String getMessageBody() {
        return messageBody;
    }

    public void setMessageBody(String messageBody) {
        this.messageBody = messageBody;
    }
}
```

```
package www.srit.ac.in.project.mobile.encryption.self.room.tables;

import androidx.annotation.NonNull;
import androidx.room.ColumnInfo;
import androidx.room.Entity;
import androidx.room.PrimaryKey;

import com.google.firebase.database.annotations.NotNull;


@Entity
public class Notes {
    @ColumnInfo(name = "notes_title")
    private String notesTitle;

    @PrimaryKey
    @NonNull
    @ColumnInfo(name = "time_stamp")
    private String timeStamp;
    @ColumnInfo(name = "notes_body")
    private String notesBody;

    public Notes(String notesTitle, String timeStamp, String notesBody) {
        this.notesTitle = notesTitle;
        this.timeStamp = timeStamp;
        this.notesBody = notesBody;
    }


    public Notes() {

    }
```

```
    public String getNotesTitle() {
        return notesTitle;
    }

    public void setNotesTitle(String notesTitle) {
        this.notesTitle = notesTitle;
    }


    public String getTimeStamp() {
        return timeStamp;
    }

    public void setTimeStamp(String timeStamp) {
        this.timeStamp = timeStamp;
    }

    public String getNotesBody() {
        return notesBody;
    }

    public void setNotesBody(String notesBody) {
        this.notesBody = notesBody;
    }
}
```

```
package www.srit.ac.in.project.mobile.encryption.self.room.tables;

import androidx.room.ColumnInfo;
import androidx.room.Entity;
import androidx.room.PrimaryKey;
@Entity
public class Settings {
    @PrimaryKey(autoGenerate = false)
    @ColumnInfo(name = "id")
    private int id;
    @ColumnInfo(name = "settings_name")
    private String name;
    @ColumnInfo(name = "settings_password")
    private String passkey;
    @ColumnInfo(name = "settings_state")
    private boolean state;

    public Settings(int id, String name, String passkey, boolean state) {
        this.id = id;
        this.name = name;
        this.passkey = passkey;
        this.state = state;
    }
```

```
    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getPasskey() {
        return passkey;
    }

    public void setPasskey(String passkey) {
        this.passkey = passkey;
    }

    public boolean isState() {
        return state;
    }

    public void setState(boolean state) {
        this.state = state;
    }
}
```

Dao:

```
package www.srit.ac.in.project.mobile.encryption.self.room;

import androidx.lifecycle.LiveData;
import androidx.room.Dao;
import androidx.room.Delete;
import androidx.room.Insert;
import androidx.room.OnConflictStrategy;
import androidx.room.Query;
import androidx.room.Update;

import java.util.List;

import www.srit.ac.in.project.mobile.encryption.self.room.tables.Contact;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Image;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Message;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Notes;
```

```
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings;
@Dao
public interface MSFDao {
    @Insert(onConflict = OnConflictStrategy.REPLACE)
    void insertContact(Contact contact);

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    void insertMessage(Message message);

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    void insertNotes(Notes notes);

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    void insertSetting(Settings settings);


    @Delete
    void deleteContact(Contact contact);

    @Delete
    void deleteNotes(Notes notes);

    @Delete
    void deleteMessage(Message message);
    @Update
    void updateNotes(Notes notes);

    @Query("SELECT * FROM Settings")
    LiveData<List<Settings>> getAllSettings();

    @Query("SELECT * FROM Contact")
    LiveData<List<Contact>> getAllContacts();

    @Query("SELECT * FROM Message")
    LiveData<List<Message>> getAllMessages();

    @Query("SELECT * FROM Notes")
    LiveData<List<Notes>> getAllNotes();



    @Update
    void updateSettings(Settings settings);

    @Query("DELETE from Contact")
    void delteAllContacts();

    @Query("DELETE from Message")
    void deleteAllMessages();
```

```
    @Query("DELETE from Notes")
    void deleteAllNotes();

}
```

Database:

```
package www.srit.ac.in.project.mobile.encryption.self.room;

import android.content.Context;

import androidx.annotation.NonNull;
import androidx.room.Database;
import androidx.room.Delete;
import androidx.room.Room;
import androidx.room.RoomDatabase;
import androidx.sqlite.db.SupportSQLiteDatabase;

import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

import www.srit.ac.in.project.mobile.encryption.self.room.tables.Contact;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Image;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Message;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Notes;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings;

@Database(entities = {
        Contact.class,
        Message.class,
        Notes.class,
        Settings.class},
        version = 2, exportSchema = false)
public abstract class MSFDatabase extends RoomDatabase {
    public abstract MSFDao msfDao();

    private static volatile MSFDatabase INSTANCE;
    private static final int NUMBER_OF_THREADS = 10;
    static final ExecutorService databaseWriteExecutor =
            Executors.newFixedThreadPool(NUMBER_OF_THREADS);

    static MSFDatabase getDatabase(final Context context) {
        if (INSTANCE == null) {
            synchronized (MSFDatabase.class) {
                if (INSTANCE == null) {
                    INSTANCE = Room.databaseBuilder(context.getApplicationContext(),
                            MSFDatabase.class, "msf_database")
                            .addCallback(sRoomDatabaseCallback)
```

```
                    .build();
            }
        }
    }
    return INSTANCE;
}

private static RoomDatabase.Callback sRoomDatabaseCallback = new
RoomDatabase.Callback() {
    @Override
    public void onOpen(@NonNull SupportSQLiteDatabase db) {
        super.onOpen(db);

        // If you want to keep data through app restarts,
        // comment out the following block
        databaseWriteExecutor.execute(() -> {
            // Populate the database in the background.
            // If you want to start with more words, just add them.
            MSFDao dataDao = INSTANCE.msfDao();
        });
    }
};
}
```

Repository:

```
package www.srit.ac.in.project.mobile.encryption.self.room;

import android.app.Application;

import androidx.lifecycle.LiveData;

import java.util.List;

import www.srit.ac.in.project.mobile.encryption.self.room.tables.Contact;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Image;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Message;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Notes;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings;

class MSFRepository {
    private MSFDao msfDao;
    private LiveData<List<Contact>> allContactsList;
    private LiveData<List<Settings>> allSettings;
    private LiveData<List<Message>> allMessages;
    private LiveData<List<Notes>> allNotes;
```

```
MSFRepository(Application application){
    MSFDatabase database=MSFDatabase.getDatabase(application);
    msfDao=database.msfDao();
    allContactsList=msfDao.getAllContacts();
    allSettings=msfDao.getAllSettings();
    allMessages=msfDao.getAllMessages();
    allNotes=msfDao.getAllNotes();
    //allImages=msfDao.getAllImages();

}

LiveData<List<Contact>> getAllContactsList(){
    return allContactsList;
}
LiveData<List<Settings>> getAllSettings(){
    return allSettings;
}
LiveData<List<Message>> getAllMessages(){
    return allMessages;
}
LiveData<List<Notes>> getAllNotes(){
    return allNotes;
}
void insertContact(Contact contact) {
    MSFDatabase.databaseWriteExecutor.execute(() -> {
        msfDao.insertContact(contact);
    });
}
void insertSetting(Settings settings){
    MSFDatabase.databaseWriteExecutor.execute(()->{
        msfDao.insertSetting(settings);
    });
}
void insertMessage(Message message){
    MSFDatabase.databaseWriteExecutor.execute(()->{
        msfDao.insertMessage(message);
    });
}
void insertNotes(Notes notes){
    MSFDatabase.databaseWriteExecutor.execute(()->{
        msfDao.insertNotes(notes);
    });
}
void updateSetting(Settings settings){
    MSFDatabase.databaseWriteExecutor.execute(()->{
        msfDao.updateSettings(settings);
    });
}
void deleteContact(Contact contact){
    MSFDatabase.databaseWriteExecutor.execute(()->{
```

```
            msfDao.deleteContact(contact);
        });
    }
    void deleteNotes(Notes notes){
        MSFDatabase.databaseWriteExecutor.execute(()->{
            msfDao.deleteNotes(notes);
        });
    }
    void deleteMessage(Message message){
        MSFDatabase.databaseWriteExecutor.execute(()->{
            msfDao.deleteMessage(message);
        });
    }
    void updateNotes(Notes notes){
        MSFDatabase.databaseWriteExecutor.execute(()->{
            msfDao.updateNotes(notes);
        });
    }

}
```

View Model:

```
package www.srit.ac.in.project.mobile.encryption.self.room;

import android.app.Application;

import androidx.lifecycle.AndroidViewModel;
import androidx.lifecycle.LiveData;

import java.util.List;

import www.srit.ac.in.project.mobile.encryption.self.room.tables.Contact;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Image;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Message;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Notes;
import www.srit.ac.in.project.mobile.encryption.self.room.tables.Settings;

public class MSFViewModel extends AndroidViewModel {
    private MSFRepository msfRepository;
    private LiveData<List<Contact>> allContactsList;
    private LiveData<List<Settings>> allSettings;
    private LiveData<List<Message>> allMessages;
    private LiveData<List<Notes>> allNotes;
   // private LiveData<List<Image>> allImages;

    public MSFViewModel(Application application) {
        super(application);
```

```
        msfRepository=new MSFRepository(application);
        allContactsList=msfRepository.getAllContactsList();
        allSettings=msfRepository.getAllSettings();
        allMessages=msfRepository.getAllMessages();
        allNotes=msfRepository.getAllNotes();
        //allImages=msfRepository.getAllImages();
    }
    public LiveData<List<Contact>> getAllContactsList(){
        return allContactsList;
    }
    public LiveData<List<Settings>> getAllSettings(){
        return allSettings;
    }
    public LiveData<List<Message>> getAllMessages(){
        return allMessages;
    }
    public LiveData<List<Notes>> getAllNotes(){
        return allNotes;
    }
    /*public LiveData<List<Image>> getAllImages(){
        return allImages;
    }
    public void insertImage(Image image){msfRepository.insertImage(image);}
    public void deleteImage(Image image){msfRepository.deleteImage(image);}*/
    public void insertContact(Contact contact){
        msfRepository.insertContact(contact);
    }
    public void insertSetting(Settings settings){
        msfRepository.insertSetting(settings);
    }
    public void insertMessage(Message message)
    {
        msfRepository.insertMessage(message);
    }
    public void insertNotes(Notes notes){
        msfRepository.insertNotes(notes);
    }
    public void deleteContact(Contact contact){
        msfRepository.deleteContact(contact);
    }
    public void deleteNotes(Notes notes){
        msfRepository.deleteNotes(notes);
    }
    public void deleteMessage(Message message){
        msfRepository.deleteMessage(message);
    }
    public void updateSetting(Settings settings){
        msfRepository.updateSetting(settings);
    }
    public void updatNotes(Notes notes){
```

```
        msfRepository.updateNotes(notes);
    }
}
```