Android JetPack Navigation Component

Problem Requirement: Use JetPack Navigation Component to pass data from one fragment to another Fragment.

Gradle Build system – Setup to use Navigation Components

Step 1: Create a new project with an empty activity and do the gradle setup

Inside build.gradle(Module) - Changes

1. Add it inside android {} to enable ViewBinding

```
buildFeatures{
 viewBinding true
}
```

2. Add the below two dependencies to use Navigation Component

```
implementation 'androidx.navigation:navigation-fragment-ktx:2.5.2'
implementation 'androidx.navigation:navigation-ui-ktx:2.5.2'
```

3. Add below plugins on the top

```
id 'androidx.navigation.safeargs'
```

Inside build.gradle(Project),

1. add the below lines on the top

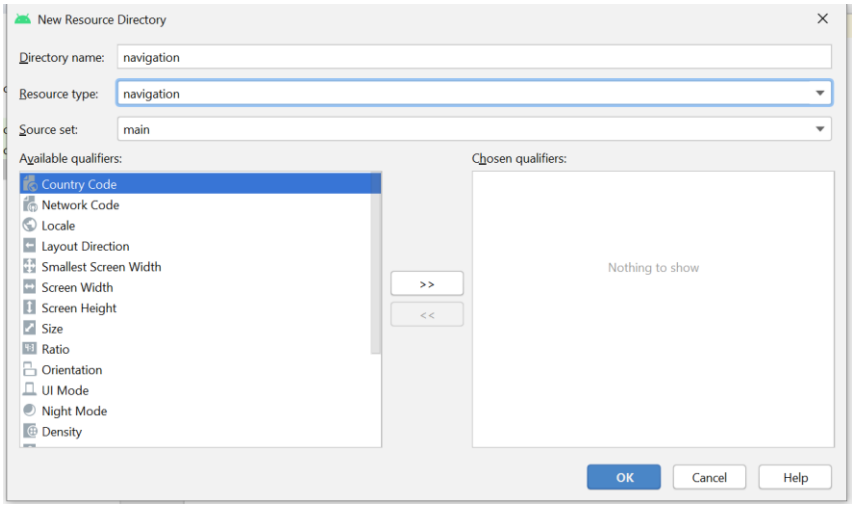
```
buildscript {
    dependencies {
        classpath("androidx.navigation:navigation-safe-args-gradle-plugin:2.5.2")
    }
}
```

Finally click Sync now and wait until Sync finished

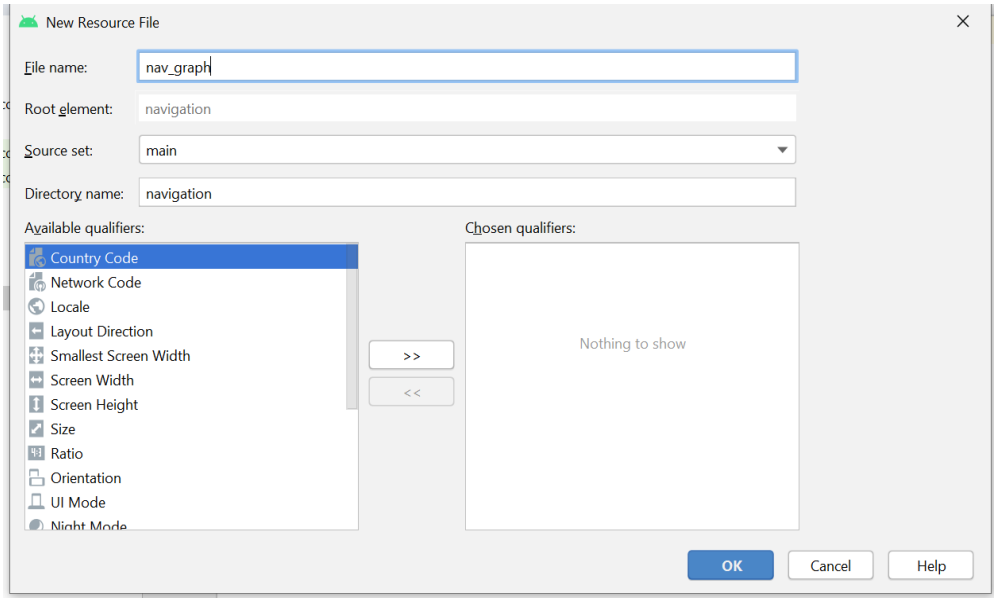
Follow the below steps to practice this concept.

Step 2: Adding navigation resource directory and nav_graph

- In the Project window, right-click on the res directory and select New →Android Resource Directory. The New Resource File dialog appears as like the given screenshot.
- Type a Directory name as navigation.
- Select Navigation from the Resource type drop-down list, and then click OK.



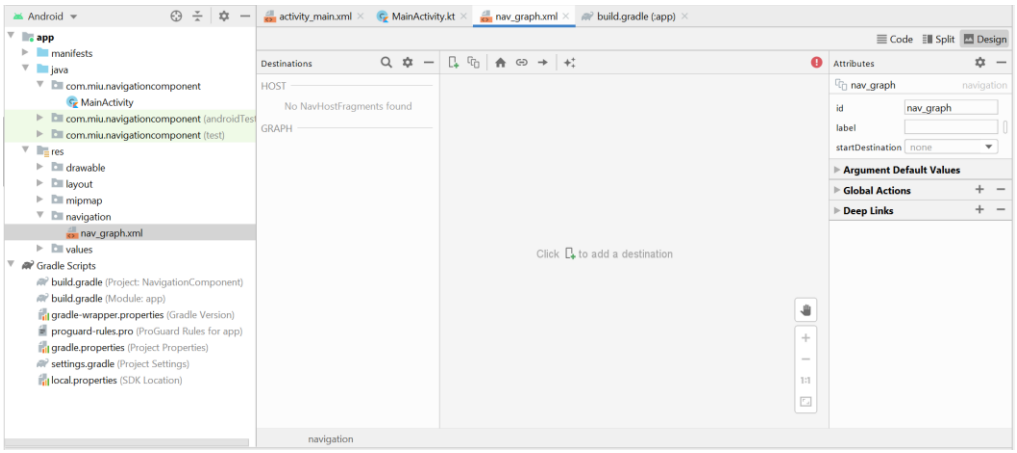
Step 3: Go to res→ Right click navigation directory → New→ Navigation Resource File, then give the file name as nav_graph.



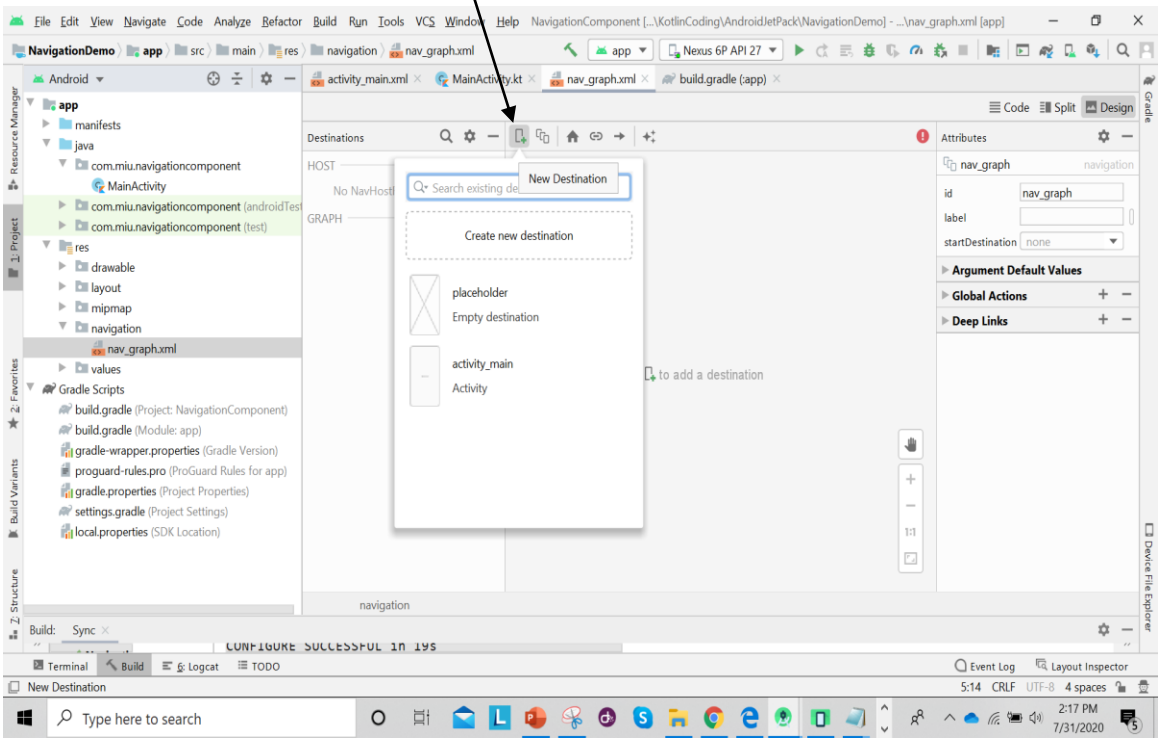
Step 4: Need to add the two fragments by doing any of the given below approaches.

Approach 1: Create Blank Fragments in the regular way for making Navigation graph destinations.

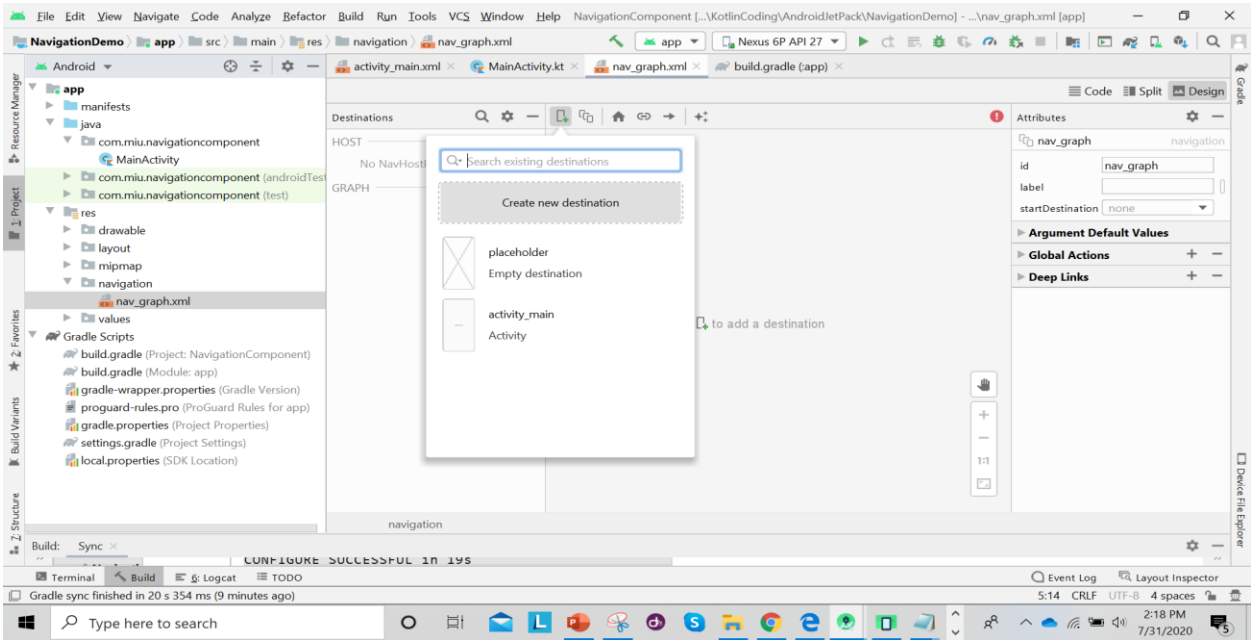
Approach 2: Click nav_graph and open it in a design mode. You will see the screen as below



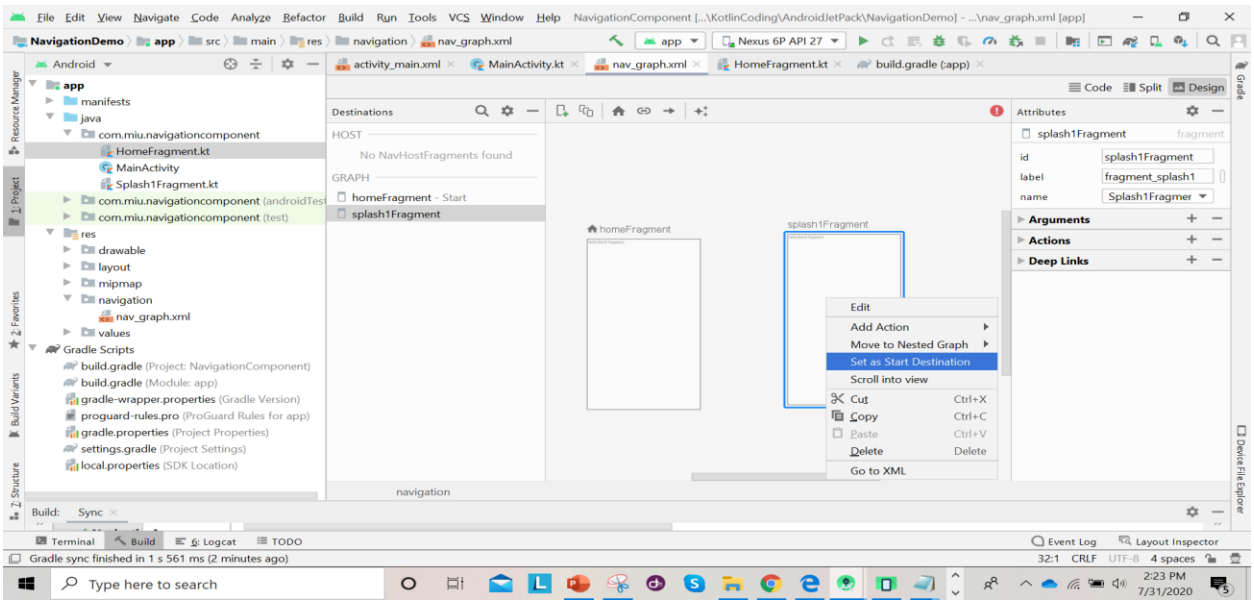
- Click on the small icon  to add the Fragments, here called as New Destination.



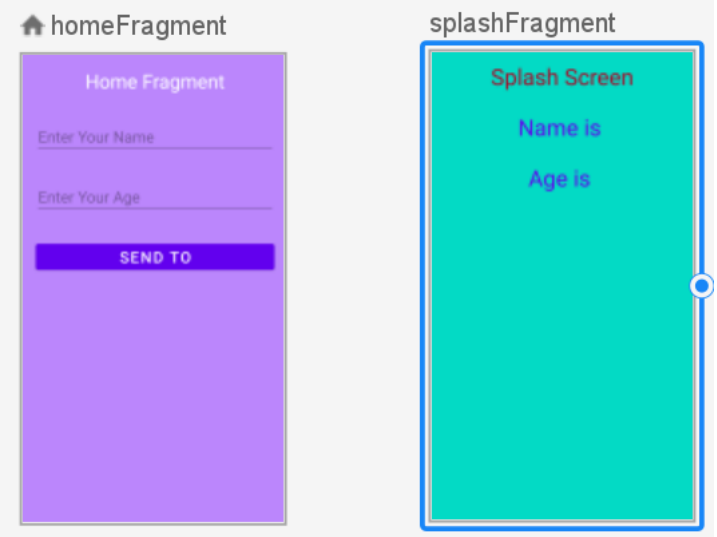
- If you have Existing fragments you can select from the list, if you want to create a new Fragments click Create new destination. By following the Approach 2 and create two Fragments like HomeFragmnet and Splash1Fragment.



Here is the screen with two Fragments after completing the previous step 5. By default HomeFragment is the start destination, if you want to make other Fragment as a Start destination by right click on the respective Fragment and select Set as Start Destination as below.



Step 5: After creating the Fragments you will automatically get the layout files fragment_home.xml and fragment_splash1.xml. Design according to your requirements.



Refer the given code for each Fragment

fragment_home.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
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"
    android:orientation="vertical"
    android:background="@color/purple_200"
    tools:context=".HomeFragment">

    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_margin="20dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="30sp"
        android:gravity="center"
        android:textColor="@color/white"
        android:text="Home Fragment" />
    <EditText
        android:layout_margin="20dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/etName"
```

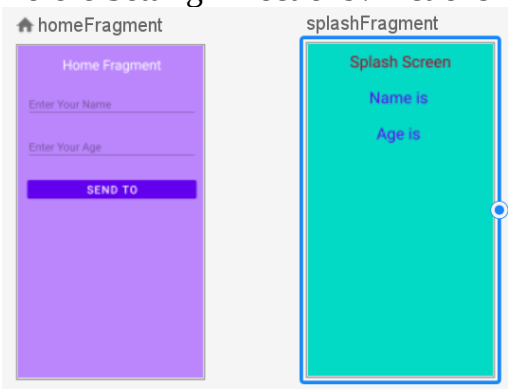
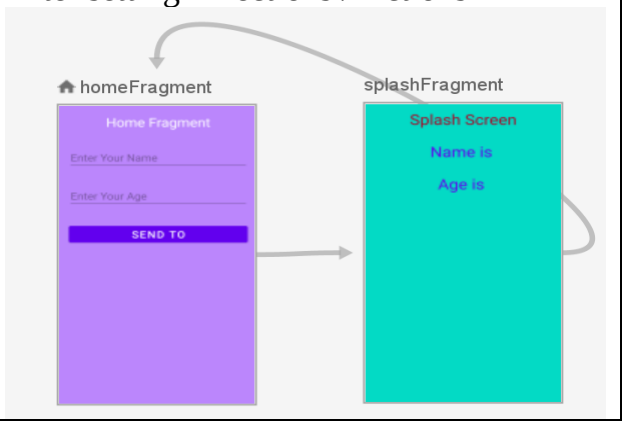
```
        android:hint="Enter Your Name"
        android:inputType="textPersonName"
        android:textSize="24sp"/>
    <EditText
        android:layout_margin="20dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/etAge"
        android:inputType="number"
        android:hint="Enter Your Age"
        android:textSize="24sp"/>
    <Button
        android:layout_margin="20dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/btnSend"
        android:text="Send To"
        android:textSize="24sp"/>
</LinearLayout>
```

fragment_splash.xml

```
<LinearLayout
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"
    android:orientation="vertical"
    android:background="@color/teal_200"
    tools:context=".SplashFragment">
    <TextView
        android:id="@+id/tvSplash"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="35sp"
        android:gravity="center"
        android:textColor="@color/design_default_color_error"
        android:text="Splash Screen" />
    <Space
        android:layout_width="match_parent"
        android:layout_height="16dp"/>
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/tvName"
        android:textSize="30sp"
        android:textColor="@color/purple_500"/>
    <Space
```

```
        android:layout_width="match_parent"
        android:layout_height="16dp"/>
    <TextView
        android:id="@+id/tvAge"
        android:textSize="30sp"
        android:textColor="@color/purple_500"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>
    <Space
        android:layout_width="match_parent"
        android:layout_height="32dp"/>
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/home"
        android:textAllCaps="false"
        android:textSize="30sp"
        android:text="Home"/>
</LinearLayout>
```

Step 6: Open nav_graph.xml in the design mode. Now need to apply directions. We are going to navigate from HomeScreen to Splash1 Screen and vice versa. Click the dotted circle from the homeFragment to splashFragment and vice versa

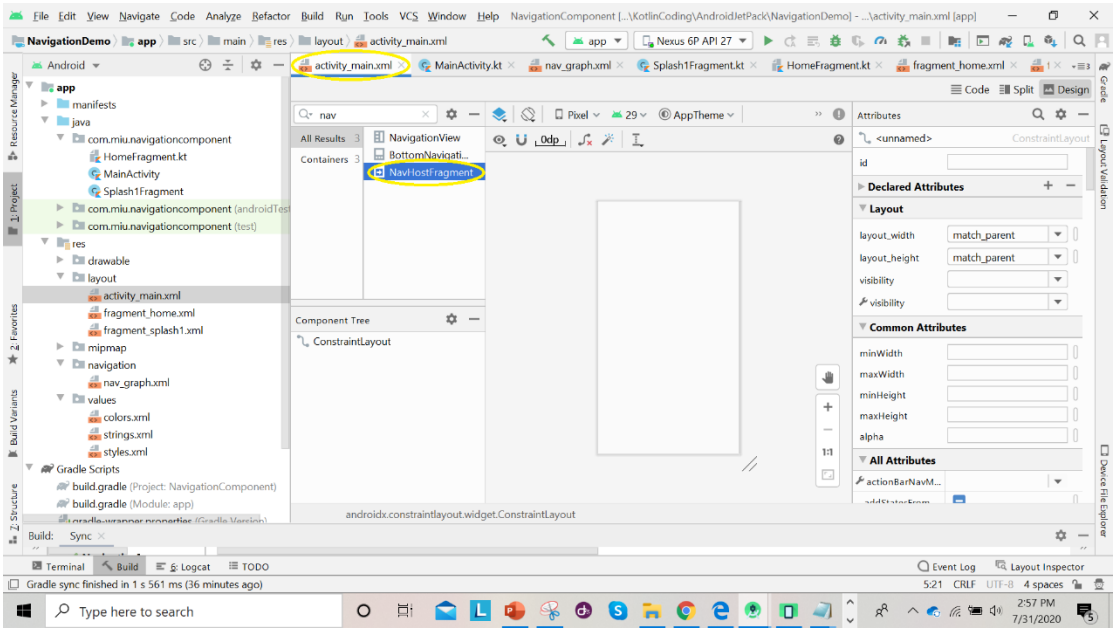
Before Setting Directions / Actions	After setting Directions / Actions
	

Once you create a directions/actions will generate xml code inside the nav_graph as mentioned below,

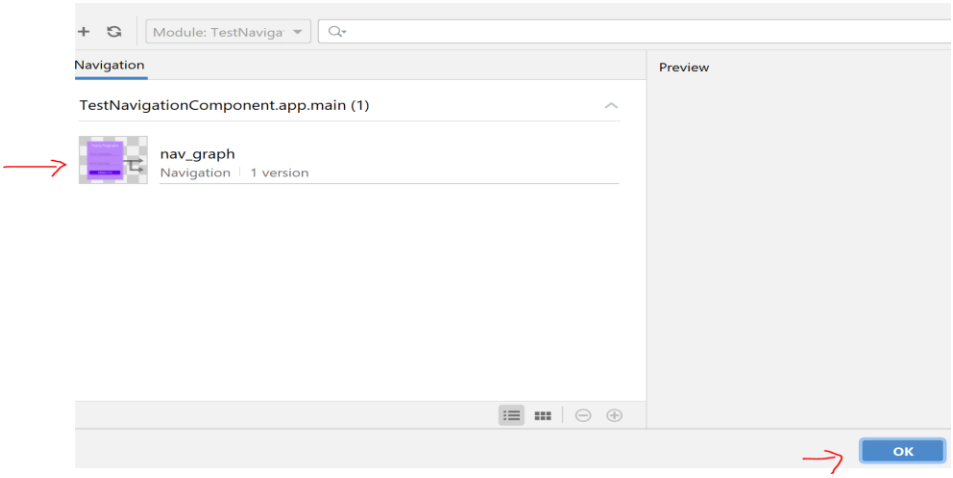
```
<fragment
    android:id="@+id/homeFragment"
    android:name="com.miu.testnavigationcomponent.HomeFragment"
    android:label="fragment_home"
    tools:layout="@layout/fragment_home" >
    <action
        android:id="@+id/action_homeFragment_to_splashFragment"
        app:destination="@id/splashFragment" />
```

```
</fragment>

<fragment
    android:id="@+id/splashFragment"
    android:name="com.miu.testnavigationcomponent.SplashFragment"
    android:label="fragment_splash"
    tools:layout="@layout/fragment_splash" >
    <action
        android:id="@+id/action_splashFragment_to_homeFragment"
        app:destination="@id/homeFragment" />
</fragment>
```

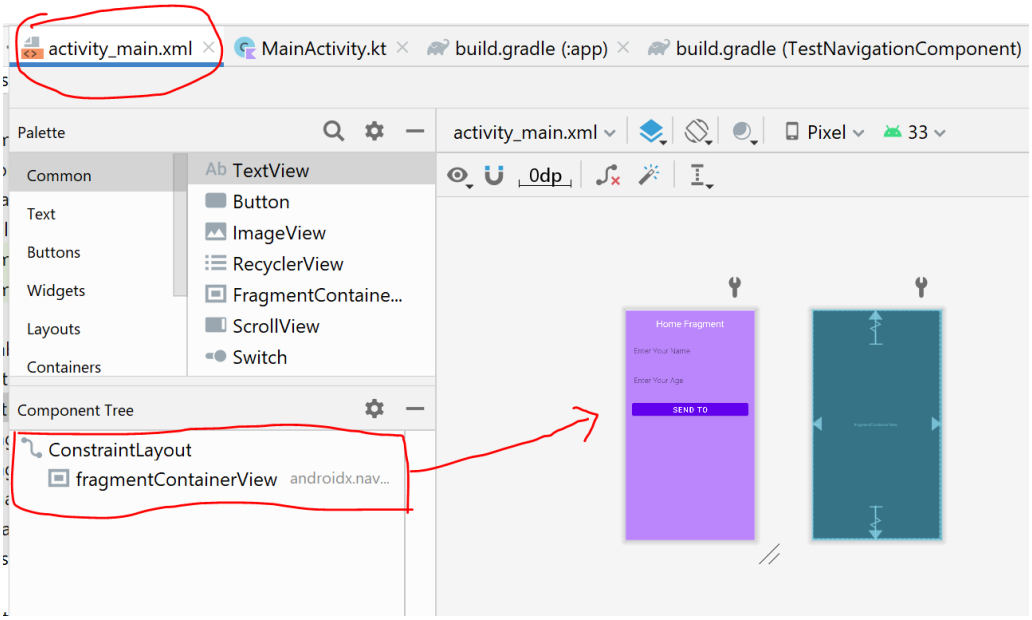
Step 7: Add NavHostFragment in the activity_main.xml. Remove the existing TextView and drag the NavHostFragment as highlighted below.



Once you drag it you will get the below screen. Select the nav_graph and click Ok.



It will automatically bring the HomeScreen as a Start destinations as below.



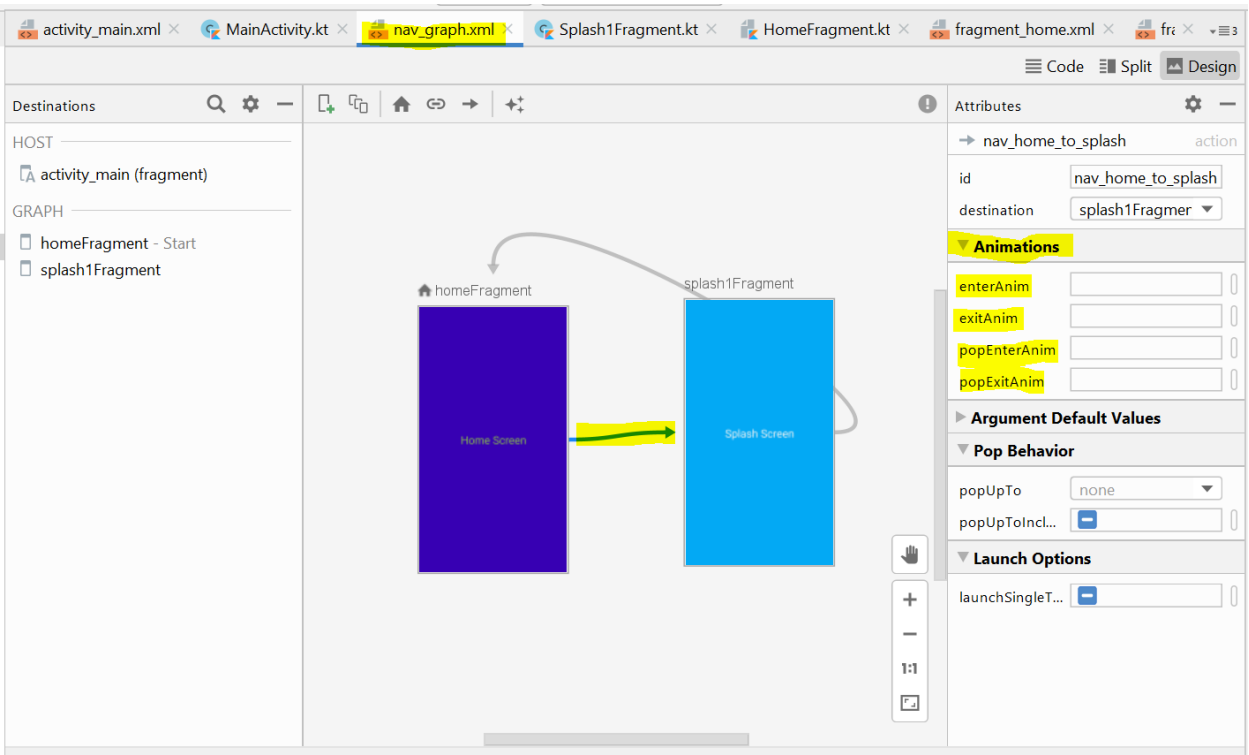
activity_main.xml

```
<androidx.constraintlayout.widget.ConstraintLayout
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=".MainActivity">

    <androidx.fragment.app.FragmentContainerView
        android:id="@+id/fragmentContainerView"
        android:name="androidx.navigation.fragment.NavHostFragment"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginTop="1dp"
        android:layout_marginBottom="1dp"
        app:defaultNavHost="true"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:navGraph="@navigation/nav_graph" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Step 8: Applying Animations

Select the nav_graph.xml and click on the arrow or action, then select Animation attribute on the right side and click the desired enterAnim and exitAnim.

Learn more about https://developer.android.com/guide/navigation/navigation-animate-transitions



Four Types of Transition

- Entering a destination
- Exiting a destination
- Entering a destination via a *pop action(Press back button)*, an action that pops additional destinations off of the back stack when navigating.
- Exiting a destination via a pop action

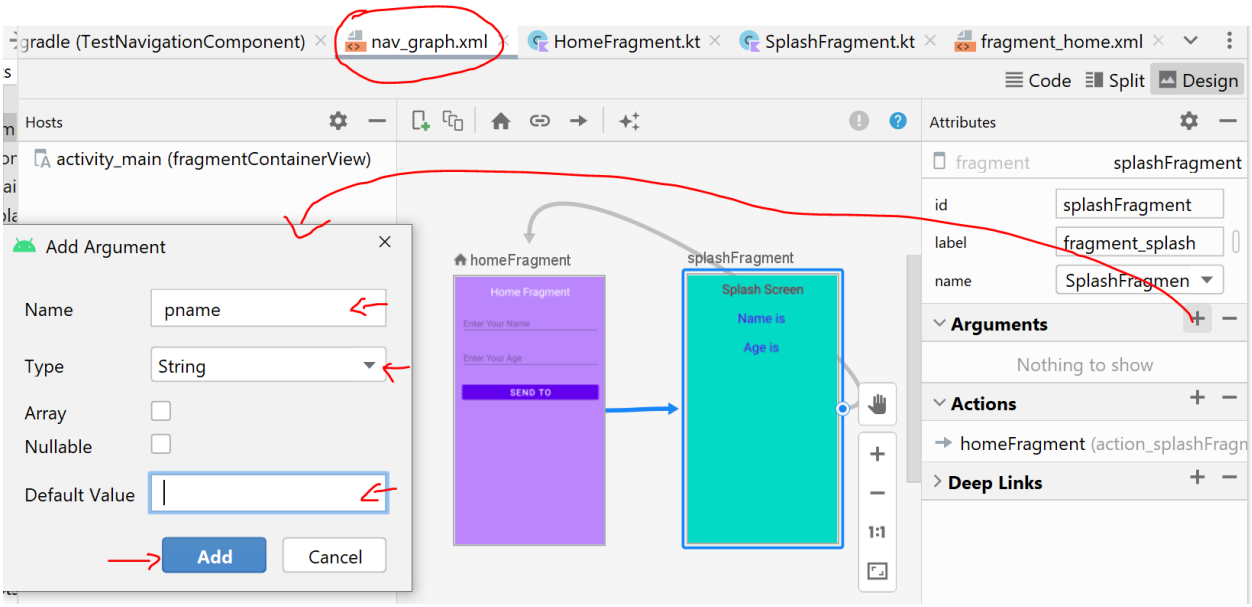
The complete code for the nav_graph.xml

```
<navigation
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:id="@+id/nav_graph"
    app:startDestination="@id/homeFragment">
    <fragment
        android:id="@+id/homeFragment"
        android:name="com.miu.testnavigationcomponent.HomeFragment"
        android:label="fragment_home"
```

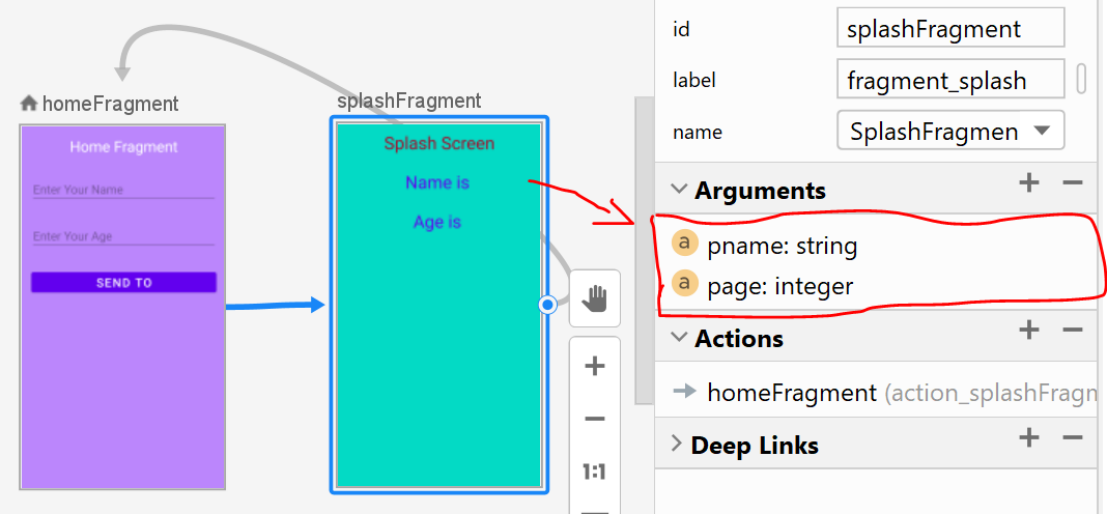
```
        tools:layout="@layout/fragment_home" >
        <action
            android:id="@+id/action_homeFragment_to_splashFragment"
            app:destination="@id/splashFragment"
            app:enterAnim="@android:anim/fade_in"
            app:exitAnim="@android:anim/fade_out" />
    </fragment>
    <fragment
        android:id="@+id/splashFragment"

android:name="com.miu.testnavigationcomponent.SplashFragment"
        android:label="fragment_splash"
        tools:layout="@layout/fragment_splash" >
        <action
            android:id="@+id/action_splashFragment_to_homeFragment"
            app:destination="@id/homeFragment" />
    </fragment>
</navigation>
```

Step 9: Use Safeargs, which helps to pass data between one Fragment to another. Here we are passing a name and age from HomeFragment to SpalshFragment. So need to add arguments for the SplashFragment by following the given below steps from the nav_graph.xml design view.

- Click the Spalsh1Fragment and select the Arguments attribute on the right side, then click + to add argument name with its type and optional default value. You have to add two arguments as a String and Int type. Default value is optional.



After adding you will get arguments as mentioned below,



Step 10: You have to rebuild your project after adding safe args.

Go to Build → Rebuild Project. Rebuild create generated classes to use for Navigations like View Binding classes.

Here is the xml code after adding safeargs for nav_graph.xml

```
<navigation
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:id="@+id/nav_graph"
    app:startDestination="@id/homeFragment">
    <fragment
        android:id="@+id/homeFragment"
        android:name="com.miu.testnavigationcomponent.HomeFragment"
        android:label="fragment_home"
        tools:layout="@layout/fragment_home" >
        <action
            android:id="@+id/action_homeFragment_to_splashFragment"
            app:destination="@id/splashFragment"
            app:enterAnim="@android:anim/fade_in"
            app:exitAnim="@android:anim/fade_out" />
    </fragment>
    <fragment
        android:id="@+id/splashFragment"

android:name="com.miu.testnavigationcomponent.SplashFragment"
        android:label="fragment_splash"
        tools:layout="@layout/fragment_splash" >
        <action
            android:id="@+id/action_splashFragment_to_homeFragment"
```

```
            app:destination="@id/homeFragment" />
        <argument
            android:name="pname"
            app:argType="string" />
        <argument
            android:name="page"
            app:argType="integer" />
    </fragment>
</navigation>
```

Step 11: Applying Navigation action to move from Home to Splash screen and vice versa by passing data of Name and Age.

You can provide the implementation inside the fragments class either `override fun onCreateView()` or you lot of implementation code, can `override fun onViewCreated()methods.`

Need to add the highlighted code in your existing HomeFragment.kt to pass the name and age to SplashFragment.

```
import android.os.Bundle
import androidx.fragment.app.Fragment
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import androidx.navigation.fragment.findNavController
import
com.miu.testnavigationcomponent.databinding.FragmentHomeBinding

class HomeFragment : Fragment() {
    private lateinit var binding: FragmentHomeBinding
   override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
                // Inflate the layout for this fragment
       binding = FragmentHomeBinding.inflate(inflater, container,
false)
       binding.apply {
           btnSend.setOnClickListener {
               // Navigate through Generated Directions through the
action Home Fragment to SplashFragment
               val directions
=HomeFragmentDirections.actionHomeFragmentToSplashFragment(etName.t
ext.toString(),etAge.text.toString().toInt())
           // Calling this on a Fragment to navigate your
Directions
```

```
                findNavController().navigate(directions)
            }
        }

        return binding.root
    }
}
```

Step 12: Need to add the highlighted code in your existing SplashFragment.kt to retrieve name and age to SplashFragment.

```
class SplashFragment : Fragment() {
    // Need to declare an object to receive the Navigation
arguments from the Generated Args class.
    private val nargs : SplashFragmentArgs by navArgs()

    private lateinit var binding:FragmentSplashBinding
            override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // Inflate the layout for this fragment
                binding = FragmentSplashBinding.inflate(inflater,
container, false)

                return binding.root
    }

    override fun onViewCreated(view: View, savedInstanceState:
Bundle?) {
        super.onViewCreated(view, savedInstanceState)
        // Retrieve and set the Arguments received from the Home
Fragment
           binding.tvName.text = "Name is ${nargs.pname}"
        // To show Fragments Toast
        //  Toast.makeText(activity,"Name is
${nargs.pname}",Toast.LENGTH_LONG).show()
              binding.tvAge.text = "Age is
${nargs.page.toString()}"
        // Once the user click the Splash Screen TextView will take
to HomeFragement
// Action to return Home Fragment using action directions id
binding.home.setOnClickListener {

findNavController().navigate(R.id.action_splashFragment_to_homeFrag
ment)
}
```

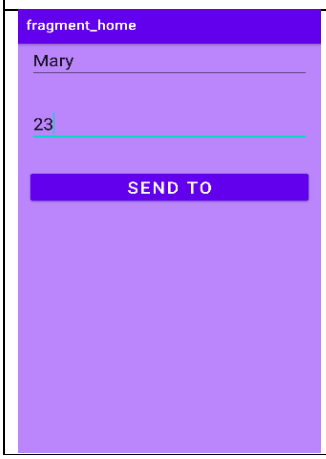
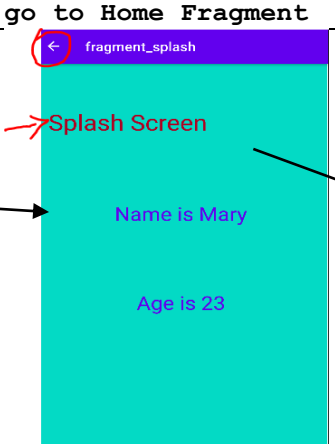
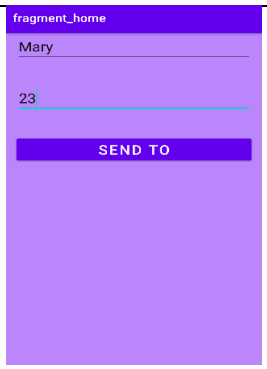
```
        }
    }

}
```

Step 13: Need to add the code to show the NavigateUP feature on the ActionBar inside MainActivity.kt

Include the highlighted code in MainActivity.kt

```
class MainActivity : AppCompatActivity() {
    // Declare Navigation Controller Object
    lateinit var mnavController: NavController
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        val navHostFragment =
supportFragmentManager.findFragmentById(R.id.fragmentContainerView)
as NavHostFragment
        mnavController = navHostFragment.navController
        // Code to link the navigation controller to the app bar
  NavigationUI.setupActionBarWithNavController(this,mnavController)
    }
    // override the onSupportNavigateUp() method to call
navigateUp() in the navigation controller
    override fun onSupportNavigateUp(): Boolean {
        return mnavController.navigateUp()
    }
}
```

After Executing the App the flow looks like below

Home Fragment	Splash Fragment Click back arrow or Splash Screen Text, will go to Home Fragment	Home Fragment
		

Difference between navigate up and back buttons

https://stuff.mit.edu/afs/sipb/project/android/docs/design/patterns/navigation.html