# OWL.M A MATERIAL DESIGN STUDY APP

## ABSTRACT:

The present Covid-19 situation pandemic forces students to attend their classes through online. This becomes a difficult task for Students to listen classes as they are facing several distractions. It also leads to Lack of communication between Students and teachers. Even students can't take notes properly. Also, there will be a library in college which helps the students to prepare for their Exams. This Project helps Teachers to upload their subject related notes, videos or Books in the App. This project can help the students in a better way than other apps So Students will feel so easy to study for their exam and also helps to gain Knowledge regarding the Subject. It reduces the teacher's work as well. Also the app provides the students with live chat and chat assistant feature.

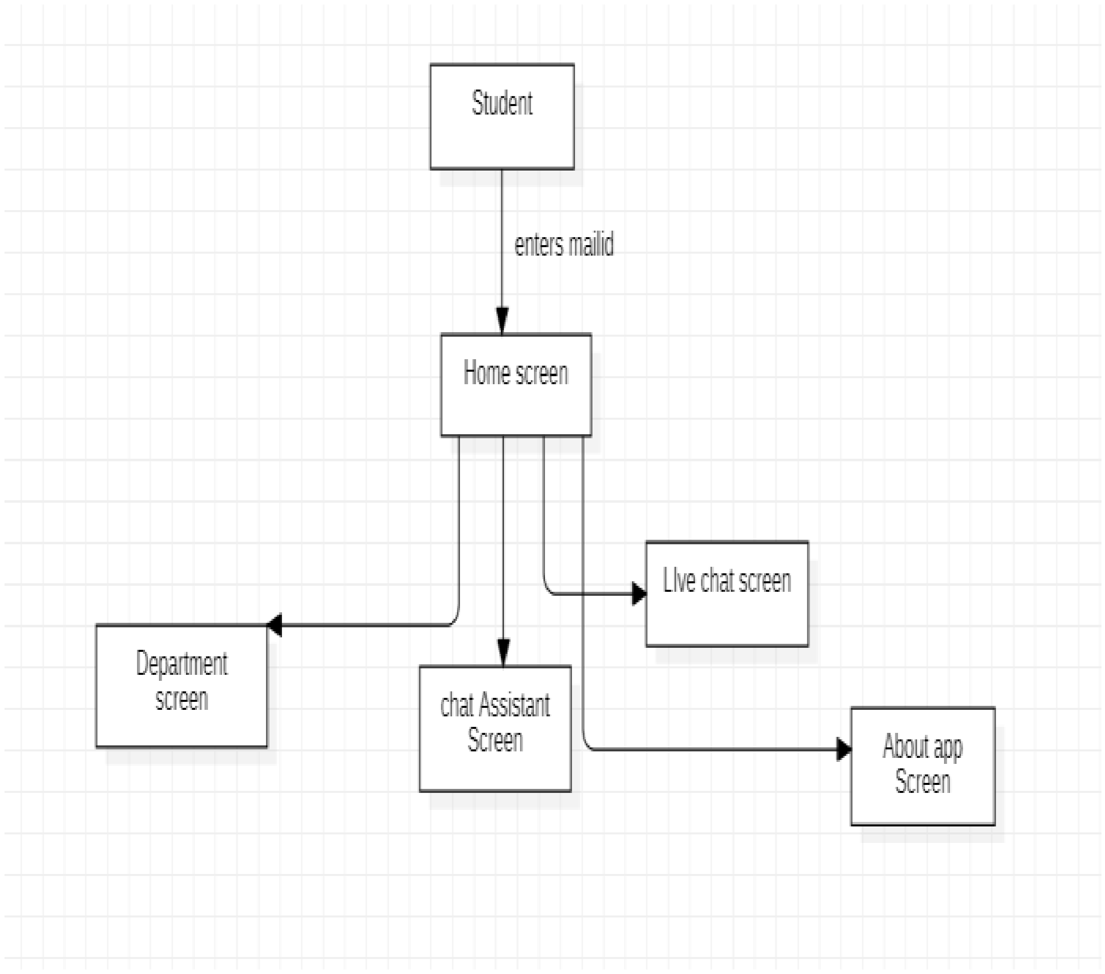
## INTRODUCTION

The Study Material Android Application is specifically designed for SCSVMV university students. This android application aims to help the students In their academics by providing them their required study materials which are uploaded by the staffs itself so that no need for students to spend their valuable time for searching the study materials online.

## PROPOSED SYSTEM

Since mobile devices became more and more powerful and distributive, mobile computing has greatly changed our lifestyle together of the foremost popular mobile operating systems, Android provides the tools and API for Android developer to develop Android applications. Mobile learning as an intersection of Mobile Computing and E-Learning providing resources which will be accessed in anywhere has capability in a superb searching system, rich interaction and full support towards an efficient learning and performance-based assessment. The proposed system aims to supply the scholars with their required study materials which are uploaded by the staffs using the database which is made by using Mongo DB and uploaded to API's. The front UI is meant using xml which allows the scholar to navigate to multiple screens. First initially a student will refer the study materials uploaded by the staff. just in case if they're not satisfied they will get help from the chat assistant which may solve the foremost of the doubts of the scholars. If the scholar wants to clarify the doubt with the staff directly they will use the live chat where the scholar and staff can chat with one another privately.

**DATAFLOW DIAGRAM**



**MODULE DESCRIPTION**

## User Interface Module

This module provides user’s Specific College ID to login into the app. Once the user is logged in, User can select the attributes which they want to perform in the App like viewing Documents, viewing Announcements, Live Chatting, Chatbot

## Login Module

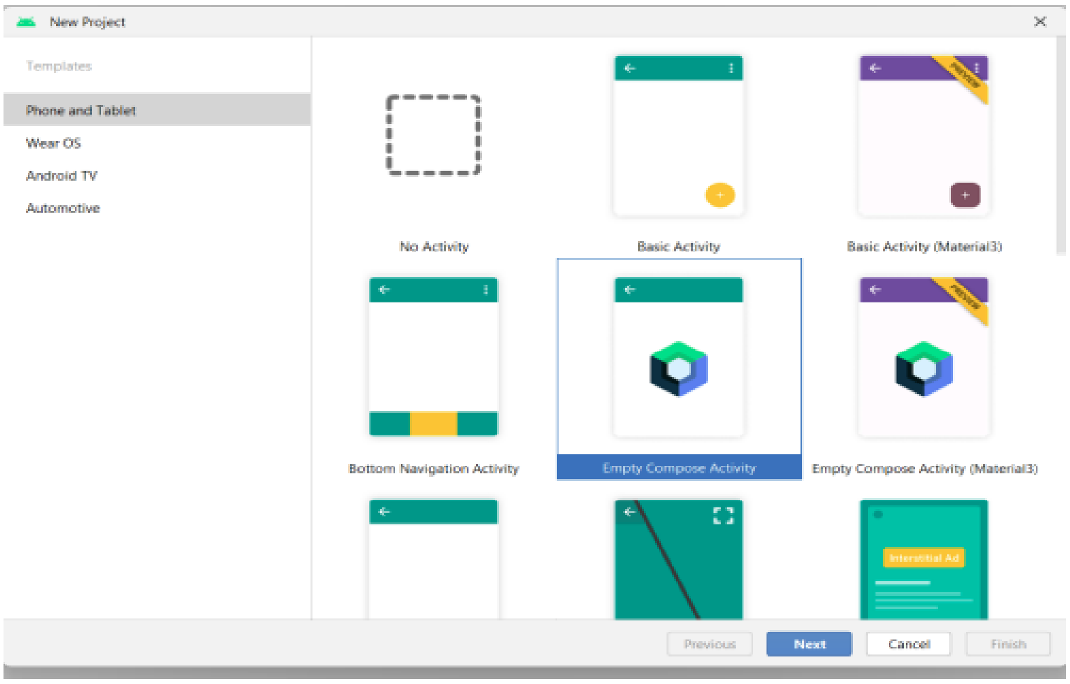
When the user of the system confronts the login page page, they will be required to enter their college ID

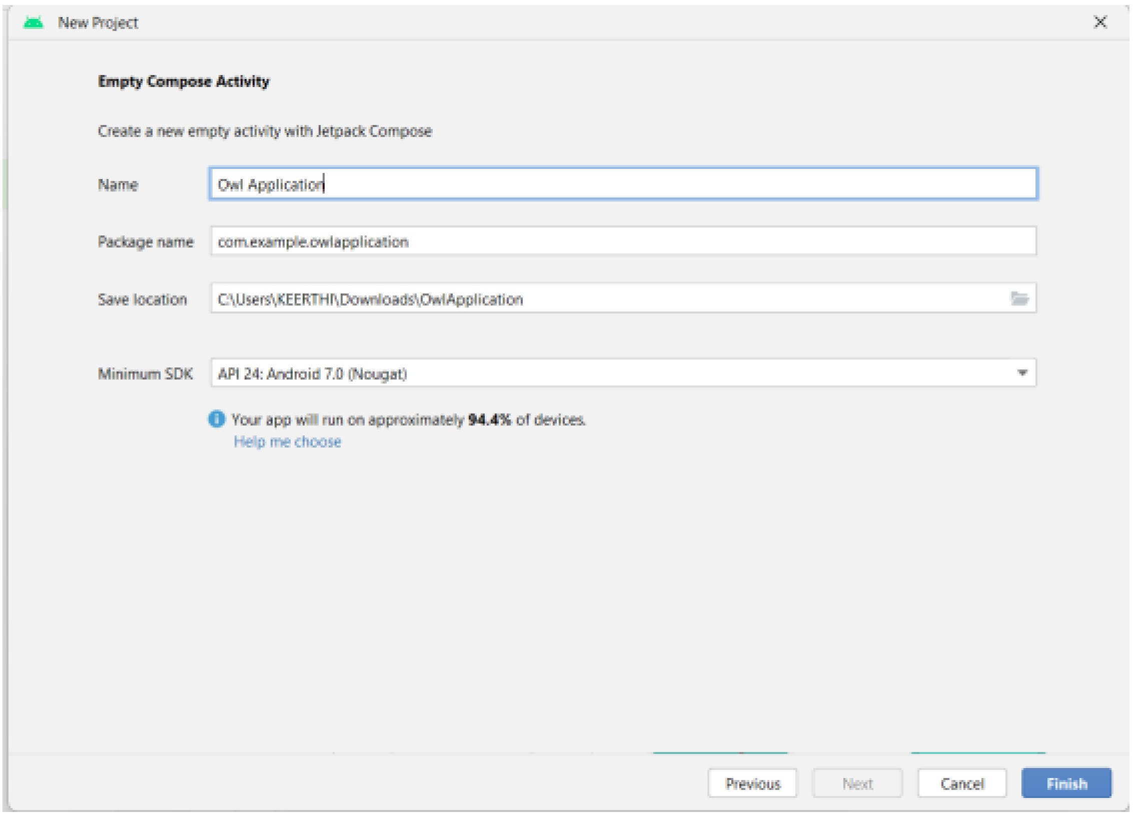
## Analysis Module

In this module the user specifies a particular attribute, specific Attribute will perform a specific task. It will be Discussed in the Result Section.

Task 1:

Required initial steps :





**Empty Compose Activity**

Create a new empty activity with Jetpack Compose

Name: Owl Application

Package name: com.example.owlapplication

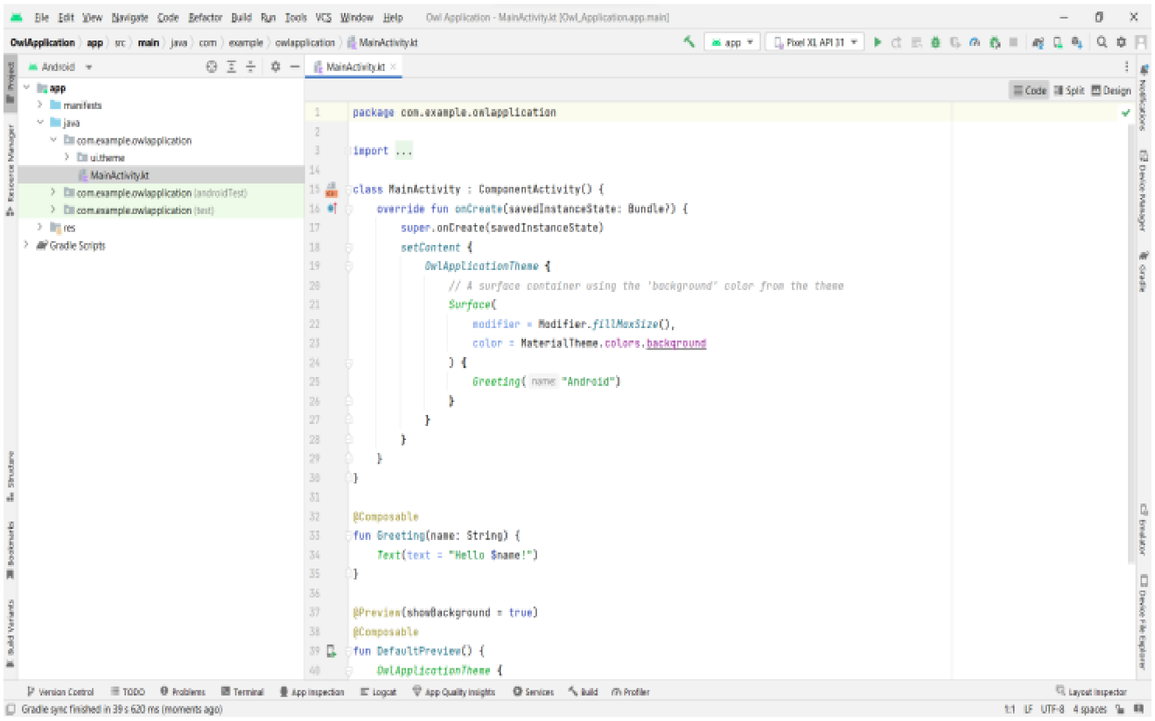
Save location: C:\Users\KEERTHI\Downloads\OwlApplication

Minimum SDK: API 24: Android 7.0 (Nougat)

Your app will run on approximately **94.4%** of devices.

Help me choose

Main activity file

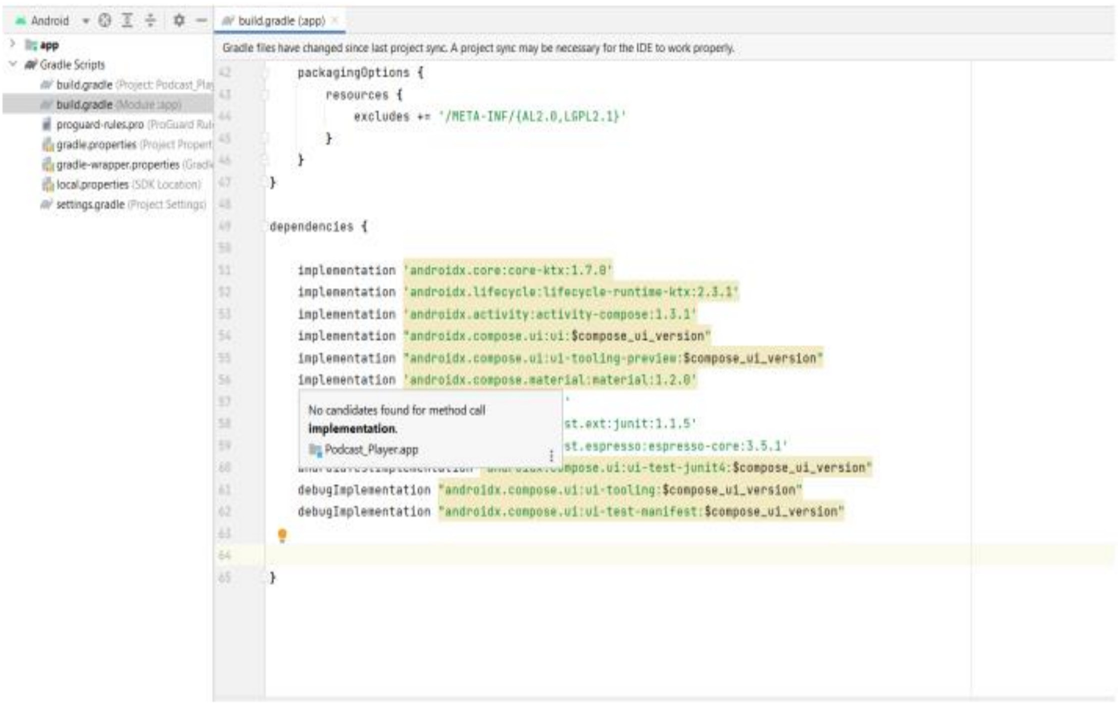


```
package com.example.owlapplication

import ...

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            OwlApplicationTheme {
                // A surface container using the 'background' color from the theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {
                    Greeting( name: "Android")
                }
            }
        }
    }
}

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

@Preview(showBackground = true)
@Composable
fun DefaultPreview() {
    OwlApplicationTheme {
```

Gradle sync finished in 39 s 620 ms (moments ago)

Task 3 :

Adding required dependencies



Gradle files have changed since last project sync. A project sync may be necessary for the IDE to work properly.

```
    packagingOptions {
        resources {
            excludes += '/META-INF/{AL2.0,LGPL2.1}'
        }
    }
}

dependencies {

    implementation 'androidx.core:core-ktx:1.7.0'
    implementation 'androidx.lifecycle:lifecycle-runtime-ktx:2.3.1'
    implementation 'androidx.activity:activity-compose:1.3.1'
    implementation "androidx.compose.ui:ui:$compose_ui_version"
    implementation "androidx.compose.ui:ui-tooling-preview:$compose_ui_version"
    implementation 'androidx.compose.material:material:1.2.0'
                                           '
                                           st.ext:junit:1.1.5'
                                           st.espresso:espresso-core:3.5.1'
                                           mpose.ui:ui-test-junit4:$compose_ui_version"
    debugImplementation "androidx.compose.ui:ui-tooling:$compose_ui_version"
    debugImplementation "androidx.compose.ui:ui-test-manifest:$compose_ui_version"


}
```

No candidates found for method call **implementation**.
Podcast_Player.app

**CODE**

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.google.android.material.testapp.theme">

  <uses-sdk android:minSdkVersion="14"
    tools:overrideLibrary="androidx.test,      android.app,      androidx.test.rule,
androidx.test.espresso, androidx.test.espresso.idling"/>

  <application
    android:name="androidx.multidex.MultiDexApplication"
    android:supportsRtl="true"
    android:theme="@style/Theme.MaterialComponents.ViewInflaterTest">

    <activity
      android:name=".MaterialComponentsViewInflaterActivity"/>
  </application>
</manifest>
-->

<LinearLayout
xmlns:android="http://schemas.android.com/apk/res
/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    android:orientation="vertical">

    <TextView
        android:id="@+id/test_text_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/text1"/>
```

```
            <Button
              android:id="@+id/test_button"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:layout_margin="24dp"
              android:text="@string/text1"/>

            <RadioButton
              android:id="@+id/test_radiobutton"
              android:layout_width="match_parent"
              android:layout_height="wrap_content"
              android:background="@android:color/white"/>

            <CheckBox
              android:id="@+id/test_checkbox"
              android:layout_width="match_parent"
              android:layout_height="wrap_content"
              android:background="@android:color/white"/>

            <AutoCompleteTextView

        android:id="@+id/test_autocomplete_text_view"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:background="@android:color/white"
                android:hint="@string/hint"
                android:text="@string/text1"/>
        </LinearLayout>

<style name="Theme.MyApp" parent="Theme.AppCompat">

  <!-- Original AppCompat attributes. -->
  <item name="colorPrimary">@color/my_app_primary_color</item>
  <item name="colorSecondary">@color/my_app_secondary_color</item>
  <item name="android:colorBackground">@color/my_app_background_color</item>
  <item name="colorError">@color/my_app_error_color</item>

  <!-- New MaterialComponents attributes. -->
  <item name="colorPrimaryVariant">@color/my_app_primary_variant_color</item>
  <item name="colorSecondaryVariant">@color/my_app_secondary_variant_color</item>
  <item name="colorSurface">@color/my_app_surface_color</item>
  <item name="colorOnPrimary">@color/my_app_color_on_primary</item>
  <item name="colorOnSecondary">@color/my_app_color_on_secondary</item>
  <item name="colorOnBackground">@color/my_app_color_on_background</item>
  <item name="colorOnError">@color/my_app_color_on_error</item>
  <item name="colorOnSurface">@color/my_app_color_on_surface</item>
  <item name="scrimBackground">@color/mtrl_scrim_color</item>
  <item
name="textAppearanceHeadline1">@style/TextAppearance.MaterialComponents.Headlin
e1</item>
  <item
name="textAppearanceHeadline2">@style/TextAppearance.MaterialComponents.Headlin
e2</item>
  <item
name="textAppearanceHeadline3">@style/TextAppearance.MaterialComponents.Headlin
e3</item>
  <item
name="textAppearanceHeadline4">@style/TextAppearance.MaterialComponents.Headlin
e4</item>
  <item
name="textAppearanceHeadline5">@style/TextAppearance.MaterialComponents.Headlin
e5</item>
  <item
name="textAppearanceHeadline6">@style/TextAppearance.MaterialComponents.Headlin
e6</item>
  <item
name="textAppearanceSubtitle1">@style/TextAppearance.MaterialComponents.Subtitle1
</item>
```

```
  <item
name="textAppearanceSubtitle2">@style/TextAppearance.MaterialComponents.Subtitle2
</item>
  <item
name="textAppearanceBody1">@style/TextAppearance.MaterialComponents.Body1</ite
m>
  <item
name="textAppearanceBody2">@style/TextAppearance.MaterialComponents.Body2</ite
m>
  <item
name="textAppearanceCaption">@style/TextAppearance.MaterialComponents.Caption</i
tem>
  <item
name="textAppearanceButton">@style/TextAppearance.MaterialComponents.Button</ite
m>
  <item
name="textAppearanceOverline">@style/TextAppearance.MaterialComponents.Overline<
/item>

</style>
<com.google.android.material.textfield.TextInputLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="@string/textfield_label">

  <com.google.android.material.textfield.TextInputEditText
      android:layout_width="match_parent"
      android:layout_height="wrap_content"/>
</com.google.android.material.textfield.TextInputLayout>
<com.google.android.material.textfield.TextInputLayout
    style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="@string/textfield_label">
  <com.google.android.material.textfield.TextInputEditText
      android:layout_width="match_parent"
      android:layout_height="wrap_content"/>
</com.google.android.material.textfield.TextInputLayout>

package com.google.android.material.testapp;

import androidx.appcompat.widget.Toolbar;
package com.google.android.material.testapp.base;
     import androidx.annotation.LayoutRes;

     /** Base activity type for all Material Components test fixtures. */
     public abstract class BaseTestActivity extends RecreatableAppCompatActivity {

       private boolean destroyed;

       @Override
       protected void onCreate(Bundle savedInstanceState) {
         super.onCreate(savedInstanceState);
         overridePendingTransition(0, 0);



     getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
         final int contentView = getContentViewLayoutResId();
         if (contentView > 0) {
           setContentView(contentView);
         }
         onContentViewSet();

     getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
       }
```

```
@Override
public void finish() {
  super.finish();
  overridePendingTransition(0, 0);
}

@LayoutRes
protected abstract int getContentViewLayoutResId();

protected void onContentViewSet() {}

@Override
protected void onDestroy() {
  super.onDestroy();
  destroyed = true;
}

@Override
public boolean isDestroyed() {
  return destroyed;
}
```

```
package com.google.android.material.testapp.base;
import android.annotation.SuppressLint;
import android.os.Bundle;
import androidx.appcompat.app.AppCompatActivity;
import androidx.annotation.Nullable;
import java.util.concurrent.CountDownLatch;

/**
 * Activity that keeps track of resume / destroy lifecycle events, as well as of the last instance
 * of itself.
 */
public class RecreatableAppCompatActivity extends AppCompatActivity {
  // These must be cleared after each test using clearState()
  @SuppressLint("StaticFieldLeak") // Not an issue because this is test-only and gets
cleared
  public static RecreatableAppCompatActivity activity;

  public static CountDownLatch resumedLatch;
  public static CountDownLatch destroyedLatch;

  public static void clearState() {
    activity = null;
    resumedLatch = null;
    destroyedLatch = null;
  }

  @Override
  protected void onCreate(@Nullable Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    activity = this;
  }

  @Override
  protected void onResume() {
    super.onResume();
    if (resumedLatch != null) {
      resumedLatch.countDown();
    }
  }

  @Override
  protected void onDestroy() {
    super.onDestroy();
    if (destroyedLatch != null) {
      destroyedLatch.countDown();
    }
  }
}



package com.google.android.material.testapp.custom;
import android.view.View;
import android.widget.TextView;
import androidx.coordinatorlayout.widget.CoordinatorLayout;
import com.google.android.material.snackbar.BaseTransientBottomBar;
import com.google.errorprone.annotations.CanIgnoreReturnValue;

/**
 * Sample code for a custom snackbar that shows two separate text views and two
images in the main
 * content area.
 */
public class CustomSnackbar extends BaseTransientBottomBar<CustomSnackbar>
{
  public CustomSnackbar(
      CoordinatorLayout parent,
      View content,
      BaseTransientBottomBar.ContentViewCallback contentViewCallback) {
    super(parent, content, contentViewCallback);
  }

  /** Sets the title of this custom snackbar. */
  @CanIgnoreReturnValue
  public CustomSnackbar setTitle(String title) {
    TextView titleView = getView().findViewById(R.id.custom_snackbar_title);
    titleView.setText(title);
    return this;
  }

  /** Sets the subtitle of this custom snackbar. */
  @CanIgnoreReturnValue
  public CustomSnackbar setSubtitle(String subtitle) {
    TextView                              subtitleView                              =
getView().findViewById(R.id.custom_snackbar_subtitle);
    subtitleView.setText(subtitle);
    return this;
  }
}


package com.google.android.material.testapp.custom;
import android.view.View;
import android.widget.TextView;
import androidx.coordinatorlayout.widget.CoordinatorLayout;
import com.google.android.material.snackbar.BaseTransientBottomBar;
import com.google.errorprone.annotations.CanIgnoreReturnValue;

/**
 * Sample code for a custom snackbar that shows two separate text views and two
images in the main
 * content area.
 */
public class CustomSnackbar extends BaseTransientBottomBar<CustomSnackbar>
{
  public CustomSnackbar(
      CoordinatorLayout parent,
      View content,
      BaseTransientBottomBar.ContentViewCallback contentViewCallback) {
```

```
    super(parent, content, contentViewCallback);
  }

  /** Sets the title of this custom snackbar. */
  @CanIgnoreReturnValue
  public CustomSnackbar setTitle(String title) {
    TextView titleView = getView().findViewById(R.id.custom_snackbar_title);
    titleView.setText(title);
    return this;
  }

  /** Sets the subtitle of this custom snackbar. */
  @CanIgnoreReturnValue
  public CustomSnackbar setSubtitle(String subtitle) {
    TextView                              subtitleView                              =
getView().findViewById(R.id.custom_snackbar_subtitle);
    subtitleView.setText(subtitle);
    return this;
  }
}


package com.google.android.material.testapp.custom;
import android.content.Context;
import android.util.AttributeSet;
import androidx.core.view.WindowInsetsCompat;
import com.google.android.material.navigation.NavigationView;

/** Expose hasSystemWindowInsets() for testing. */
public class NavigationTestView extends NavigationView {

  boolean hasSystemWindowInsets;
```

```
 public NavigationTestView(Context context) {
   this(context, null);
 }

 public NavigationTestView(Context context, AttributeSet attrs) {
   this(context, attrs, 0);
 }

 public NavigationTestView(Context context, AttributeSet attrs, int defStyleAttr) {
   super(context, attrs, defStyleAttr);
 }

 @Override
 protected void onInsetsChanged(WindowInsetsCompat insets) {
   super.onInsetsChanged(insets);
   hasSystemWindowInsets = insets.hasSystemWindowInsets();
 }

 public boolean hasSystemWindowInsets() {
   return hasSystemWindowInsets;
 }
}
```

```
apply plugin:
'com.android.application'
```

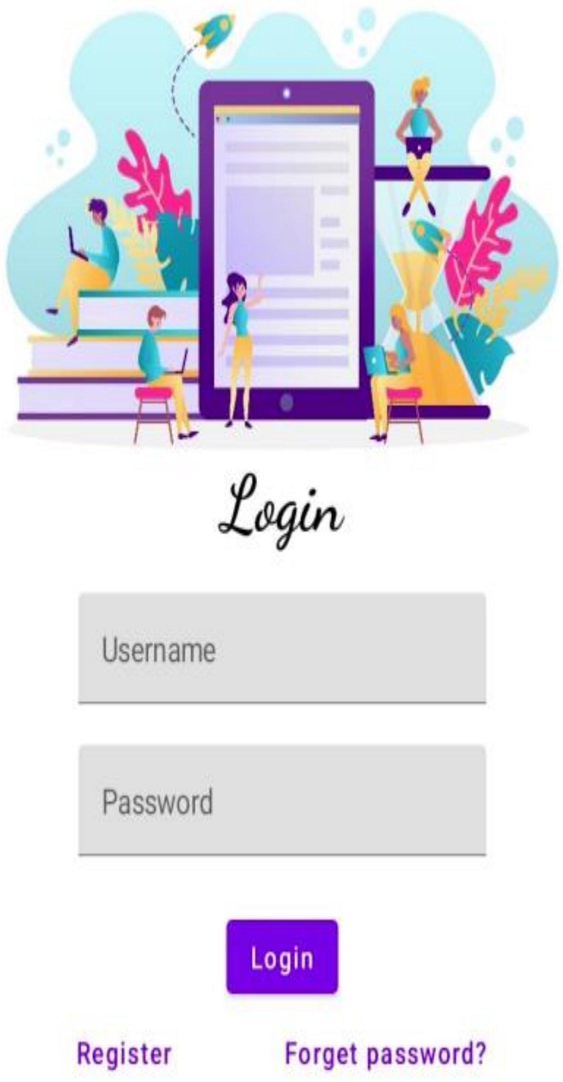
```
dependencies {
  api compatibility("annotation")
  api compatibility("appcompat")

  api project(fromPath("lib"))
  api
project(fromPath("testing/java/com/google/android/material/t
estapp/base"))
  api
project(fromPath("testing/java/com/google/android/material/t
estapp/custom"))

  api 'androidx.multidex:multidex:2.0.0'
}

android {
  defaultConfig {
    multiDexEnabled true
    minSdkVersion 14
    targetSdkVersion 33
  }

  sourceSets {
    main.manifest.srcFile 'AndroidManifest.xml'
    main.java.srcDirs = [ '.' ]
    main.java.excludes = [
      '**/build/**',
    ]
    // Only include things in this directory, not subdirectories
    main.java.includes = [ '*.java' ]
    main.res.srcDirs = [ 'res' ]
  }

  buildTypes {
    debug {
      pseudoLocalesEnabled true
    }
  }
```

```
}

package com.google.android.material.testapp;
import androidx.appcompat.widget.Toolbar;
import
com.google.android.material.testapp.base.Base
TestActivity;

/** Activity with an AppBar that contains
horizontally-scrolling content. */
public class
AppBarHorizontalScrollingActivity extends
BaseTestActivity {
  @Override
  protected int getContentViewLayoutResId() {
    return
R.layout.design_appbar_horizontal_scrolling;
  }

  @Override
  protected void onCreate(Bundle
savedInstanceState) {
    super.onCreate(savedInstanceState);

    Toolbar toolbar = (Toolbar)
findViewById(R.id.toolbar);
    setSupportActionBar(toolbar);
  }
}

package com.google.android.material.testapp;
import android.view.View;
import androidx.annotation.VisibleForTesting;
```

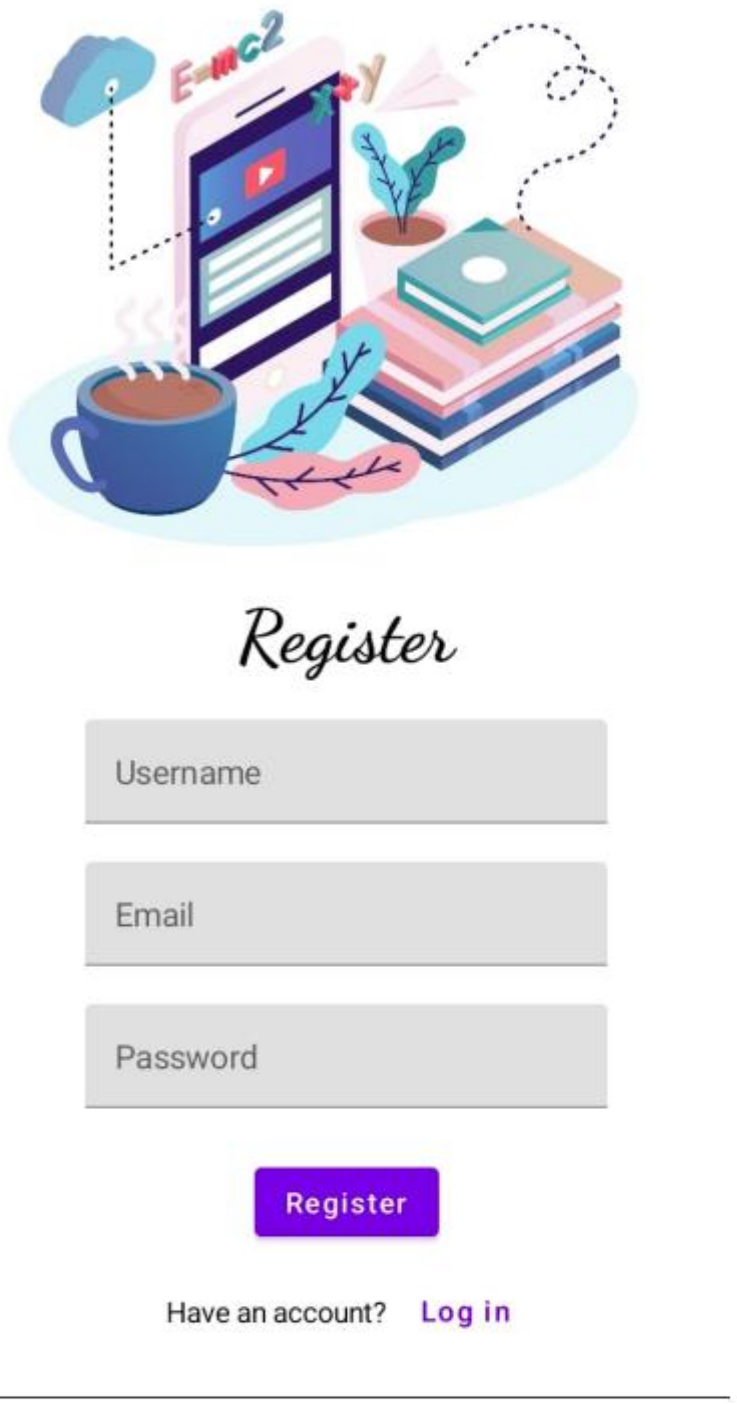
```
public class BottomSheetBehaviorWithInsetsActivity
extends BottomSheetBehaviorActivity {

  @VisibleForTesting public View mBottomSheetContent;

  @Override
  protected int getContentViewLayoutResId() {
    return
R.layout.test_design_bottom_sheet_behavior_with_insets;
  }

  @Override
  protected void onContentViewSet() {
    super.onContentViewSet();
    mBottomSheetContent =
findViewById(R.id.bottom_sheet_content);
  }
}

package com.google.android.material.testapp;
import android.widget.FrameLayout;
import androidx.annotation.VisibleForTesting;
import
androidx.coordinatorlayout.widget.CoordinatorLayout;
import
com.google.android.material.testapp.base.BaseTestActivity;

public class CoordinatorLayoutActivity extends
BaseTestActivity {

  @VisibleForTesting public FrameLayout mContainer;
  @VisibleForTesting          public          CoordinatorLayout
mCoordinatorLayout;

  @Override
  protected int getContentViewLayoutResId() {
    return R.layout.activity_coordinator_layout;
  }

  @Override
  protected void onContentViewSet() {
    mContainer = findViewById(R.id.container);
    mCoordinatorLayout = findViewById(R.id.coordinator);
  }
```

**OUTPUT**

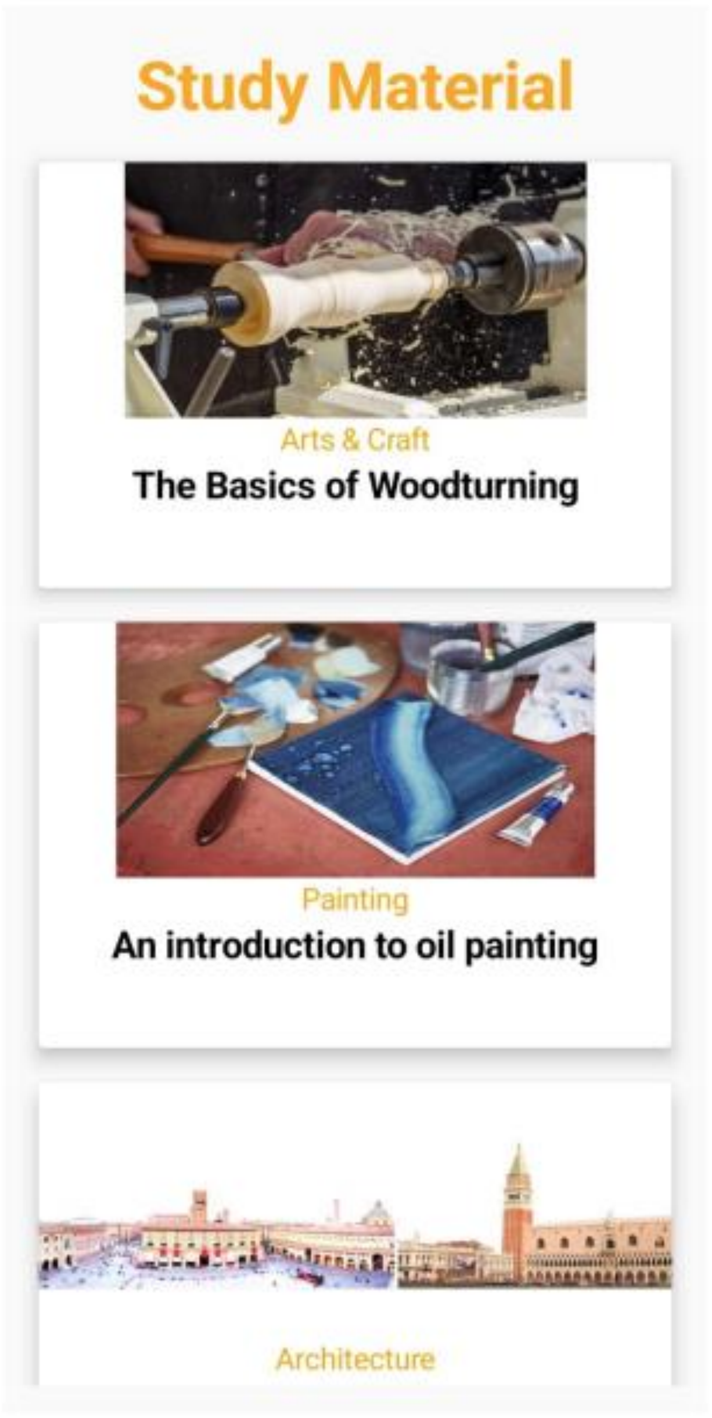
**Login Page :**



**RegisterPage :**



**MainPage :**



Study Material

Arts & Craft

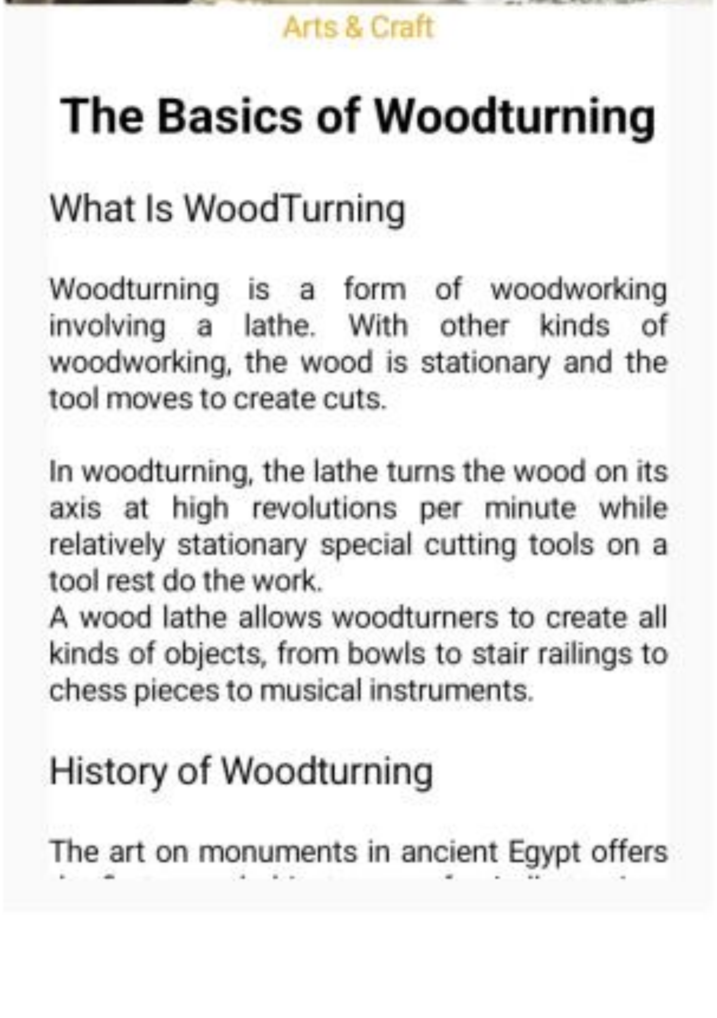
The Basics of Woodturning

Painting

An introduction to oil painting

Architecture

**Book page :**



Arts & Craft

# The Basics of Woodturning

## What Is WoodTurning

Woodturning is a form of woodworking involving a lathe. With other kinds of woodworking, the wood is stationary and the tool moves to create cuts.

In woodturning, the lathe turns the wood on its axis at high revolutions per minute while relatively stationary special cutting tools on a tool rest do the work.
A wood lathe allows woodturners to create all kinds of objects, from bowls to stair railings to chess pieces to musical instruments.

## History of Woodturning

The art on monuments in ancient Egypt offers

## CONCLUSIONS

Today's Internet user expects to experience personalized interaction with websites. If the company fails to deliver they run the risk of losing a potential customer forever. An important aspect of creating interactive web forms to collect information from users is to be able to check that the information entered is valid, therefore; information submitted through these forms should be extensively validated. Validation could be performed using client script where errors are detected when the form is submitted to the server and if any errors are found the submission of the form to the server is cancelled and all errors displayed to the user. This allows the user to correct their input before re-submitting the form to the server. We can not underestimate the importance of input validation which ensures that the application is robust against all forms of input data obtained from the user.