

**Department Of Computer Science**

**National University Of Computer And Emerging Sciences.**

THE DECORUM

Team Members:

Syed Muhammad Aqeel Abbas

17k-3701

Muneeb Ul Hasan

17k-3717

Muhammad Furqan

17k-3836

Dr Fahad Samad – Signature

**ABSTRACT:**

This enables customers to search, detect and position 3D objects via AR technology in the real environment. This app will address the existing problem by which consumers had to go to showrooms and try to imagine how each product  would fit into their place. However, this our app will allow customers to insert their e-commerce products that looks lifelike into their places before buying the actual furniture.

1. **INTRODUCTION:**
2. **Need For Product**

Purchasing interior design products often has a problem that consumers may not be satisfied with the products they have bought because they cannot arrange them in their own place before buying them. The aim of this project is to develop an android application called 'DECORUM' using Augmented Reality technology for interior decoration that will help customers visualize how furniture and interior design products will look and fit (to scale) in their home and can also provide details of products to support customer decision-making.

1. **Associated CS Problems**

Instead of going to the physical store to purchase the items, the customer would be able to display the products virtually in their home structure. The customer's basic issue is that they need to measure and search for the correct size of the product that suits their home environment because our application only enables users to see the product they want to buy according to their specifications.

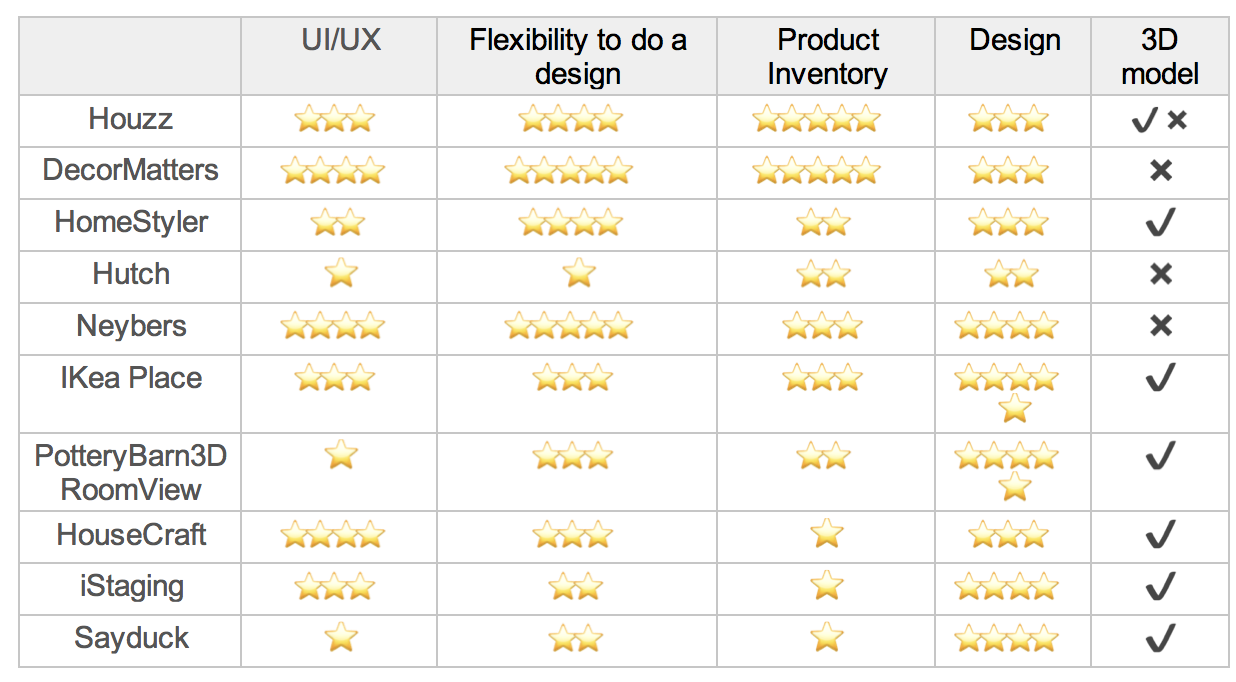
1. **Benefits to Users**

In the field of interior designing, the increased augmented reality could not fully take control. Today, people are familiar with the technology and operate smartphones that support AR. Thus, the idea of developing an interior design application leads the designer to be technologically advanced. Moreover, our app ‘DECORUM’ aims to achieve several other objectives which are as follows:

* **Improve customer’s experience in online shopping:** The project aims at providing the use of their smartphones to customers with a realistic picture of the furniture. Therefore, the user can then open the camera on the smartphone to see the selected product in AR from the list. Users can therefore immediately see how well the product is and whether it meets their needs and preferences, without the hassle of going to showrooms or travelling out.
* **Provide the featured product with realistic 3D view:** It is not enough to encourage users to display the products only from an angle so the application should be able to calibrate the 3D model with information provided for correct positioning of the products. Users will also switch around the products to see how it feels from different viewpoints. In addition, anchors should be positioned on the plane surface to the object on which the camera is aimed. This stops the movement of products from shifting with the smartphone thus making it impossible to watch from a certain perspective.
* **Provide the real-world atmosphere with correct live size:** The project must be able to view the items in a live format so that customers can ensure that the product blends into the actual space available. Therefore, having a live scale view helps create a practical environment for customers as if they had bought the items, however it disappears automatically when the application is closed.
* **Allow the application to be easily maintained:** Since the project is used on the online platform, developers need to be able to effectively perform software maintenance. Developers should also be able to add models to the application's database, which should automatically represent the application data retrieval. Therefore, users can instantly view new products and take a look at the product with AR.

1. **Gap Analysis With Existing Solution**

The application currently in use is slower to capture images, offers low resolution, further degrades user experience and issues and has to wait before an image is properly processed, even the graphics of the interior decor items are lower in resolution. However there are some applications with strong UI and UX, but 3d modelling and inventory aren't really good.

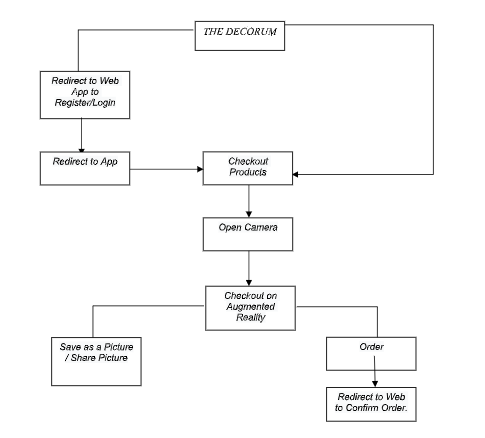


[fig. 1] Comparison of existing applications

**Our Application Features:**

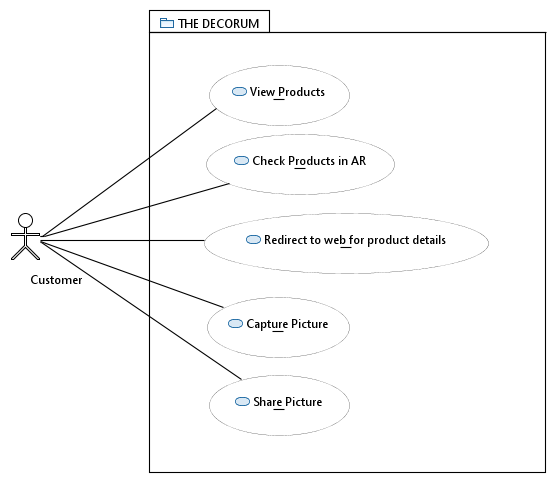
* 3d Object Tracking.
* Able to select angle of product.
* Take pictures while using AR.
* Share pictures.
* Select available color.
* Select available size.
* Search items.
* Click on product to see details of it on web.

1. **REQUIREMENTS ANALYSIS**
2. **Functional Requirements:**
   1. Functional Hierarchy



[fig. 2] Work Breakdown Structure of our Application

* 1. Use Cases



[fig. 3] Use Case Diagram of our App

|  |  |  |  |
| --- | --- | --- | --- |
| **<Use case Id: View Products>** | | | |
| **Use case Id:** | | 1 | |
| **Actors:**  Customer initiates the use case | | | |
| **Feature:** Customer can view the products | | | |
| **Scenarios** | | | |
| **Step#** | **Action** | | **Software Reaction** |
| **1.** | Customer opens the app. | | App will show the available products. |
| **Post Conditions** | | | |
| **Step#** | **Description** | | |
| **1.** | If customer wants to check product details he will be redirected to web. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **<Use case Id: Check product in AR>** | | | |
| **Use case Id:** | | 2 | |
| **Actors:**  Customer initiates the use case | | | |
| **Feature:** Customer can check the product in AR. | | | |
| **Pre-condition:** | | Customer should have clicked on a product he wishes to check in AR. | |
| **Scenarios** | | | |
| **Step#** | **Action** | | **Software Reaction** |
| **1.** | Customer clicks on the product. | | App will open camera and user will be able to drag and place the product anywhere. |
| **Post Conditions** | | | |
| **Step#** | **Description** | | |
| **1.** | If customer wants to checkout he will be redirected to web. | | |
| **2.** | If customer wants to check another product he can go to back to products list. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **<Use case Id: Redirect to web for product Details>** | | | |
| **Use case Id:** | | 3 | |
| **Actors:**  Customer initiates the use case | | | |
| **Feature:** Customer can view the products details | | | |
| **Scenarios** | | | |
| **Step#** | **Action** | | **Software Reaction** |
| **1.** | Customer clicks on the product details button | | App will redirect customer to web. |
| **Post Conditions** | | | |
| **Step#** | **Description** | | |
| **1.** | Customer can checkout from web. | | |
| **2.** | Comeback to web and view other products in AR. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **<Use case Id: Capture Pictures>** | | | |
| **Use case Id:** | | 4 | |
| **Actors:**  Customer initiates the use case | | | |
| **Feature:** Customer can capture picture. | | | |
| **Scenarios** | | | |
| **Step#** | **Action** | | **Software Reaction** |
| **1.** | Customer will click the capture button. | | App will click a picture. |
| **Post Conditions** | | | |
| **Step#** | **Description** | | |
| **1.** | Customer can share the picture. | | |
| **2.** | Save the picture and check it afterwards. | | |
| **3.** | Cancel and click another picture. | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **<Use case Id: Share Picture>** | | | | |
| **Use case Id:** | | 5 | | |
| **Actors:**  Customer initiates the use case | | | | |
| **Feature:** Customer can share the picture | | | | |
| **Pre-condition:** | | Customer should have captured a picture in order to share it. | | |
| **Scenarios** | | | | |
| **Step#** | **Action** | | | **Software Reaction** |
| **1.** | Customer will click the share button. | | | App will show share options. |
| **Post Conditions** | | | | |
| **Step#** | **Description** | | | |
| **1.** | Click another picture. | | | |
| **2.** | Cancel this activity and go to products list. | | | |
| **Use Case Cross referenced** | | | Use Case: Capture Picture should be done before doing this use case. | |

1. **Non-Functional Requirements:**
2. **Performance Requirements**

Performance will depend on the models that are loaded in the app. If the model is heavy it will take a little time to load.

1. User Documentation

FYP-I does not include this requirement however in case of time then we might implement this feature as well.

**In scope features of our Application:**

* 3d Object Tracking.
* Able to select angle of product.
* Take pictures while using AR.
* Share pictures.
* Select available color.
* Select available size.
* Search items.
* Click on product to see details of it on web.

Not In Scope Features:

* Product details.
* Add to cart products.
* Signup/Login

1. **DESIGN DETAILS**
   1. **Covering Process And Data Models**
   2. **ULS**
   3. **Code in the Database**
2. **IMPLEMENTATION DETAILS**
   1. **Development Tools**

* **Frontend :** Android Studio
* **Backend:** Firebase
  1. **Design Trade-Offs During Implementation**

1. **TESTING**
   1. **Test Data**
   2. **Unit Test**

|  |  |
| --- | --- |
| Test Case Description | App shows all the required information of a product. |
| Test Procedure | 1. Open the app. 2. App will display all the products and their info. |
| Test Data | Item Name  Price  Size |
| Expected Result | Users will be able to view all the products and their information. |
| Actual Result | Customers are able to view all the products. |
| Status | PASS |
| Priority | Major |
| Remarks | The test result contains the desired output and shows all the information of the products. |

|  |  |
| --- | --- |
| Test Case Description | Customer can check the product in Augmented Reality. |
| Test Procedure | 1. Open the app. 2. Long tap on the product that you wish to check. 3. Click on “Check in AR” button. |
| Test Data | 3d model |
| Expected Result | Users will be able to view the product in AR. |
| Actual Result | Customer is able to view the product in AR. |
| Status | PASS |
| Priority | Major |
| Remarks | The test result contains the desired output and shows the desired product in AR. |

|  |  |
| --- | --- |
| Test Case Description | Customer can download the product model and place it on the plane. |
| Test Procedure | 1. Choose the product that you want to see in AR. 2. Camera will open. 3. Click on “Download Model” button to download the model. 4. Place it on the plane. |
| Test Data | 3d model |
| Expected Result | Customers will be able to download the correct 3d model and model will be displayed. |
| Actual Result | Customers are able to download the model and view it. |
| Status | PASS |
| Priority | Major |
| Remarks | The test result contains the desired output and shows the 3d model. |

|  |  |
| --- | --- |
| Test Case Description | Customer click on the screenshot button. |
| Test Procedure | 1. Place the product where the hit point is. 2. Click on “Take Screenshot” button. 3. Screenshot will be taken. |
| Test Data | - |
| Expected Result | Users will be able to view the screenshot that they had taken. |
| Actual Result | Black screen appearing without the camera contents. |
| Status | FAIL |
| Priority | Major |
| Remarks | The test result does not contain the desired output. |

* 1. **System Test**

|  |  |
| --- | --- |
| Test Case Description | If the App launches properly with all relevant pages and logos. |
| Test Procedure | User launches the app. |
| Test Data | - |
| Expected Result | App should launch with all the working features. |
| Actual Result | App is launching with all features working fine. |
| Status | PASS |
| Priority | Major |
| Remarks | App is launching properly with all the features that are in working. |

**2)**

|  |  |
| --- | --- |
| Test Case Description | If Customers can see all the products. |
| Test Procedure | 1. Users first opens the app. 2. App will show all the products. |
| Test Data | - |
| Expected Result | Customers will be able to see all the products. |
| Actual Result | Customers can view all the products. |
| Status | PASS |
| Priority | Major |
| Remarks | This test is a success and shows the desired output. |

**3)**

|  |  |
| --- | --- |
| Test Case Description | If Customers can view all the products they can their desired product in AR. |
| Test Procedure | 1. Customer will open the app. 2. App will show all the products. 3. Customer can long tap on a product and click on “Check in AR” button. 4. App will open camera and show the product in AR. |
| Test Data | - |
| Expected Result | Customers will be able to check the product in AR. |
| Actual Result | Customers can view the product in AR. |
| Status | PASS |
| Priority | Major |
| Remarks | This test is a success and shows the desired output. |

1. **OUTPUTS OF THE PRODUCT**
   1. **Screenshots**

**A picture containing graphical user interface

Description automatically generated**

**Graphical user interface, application, PowerPoint

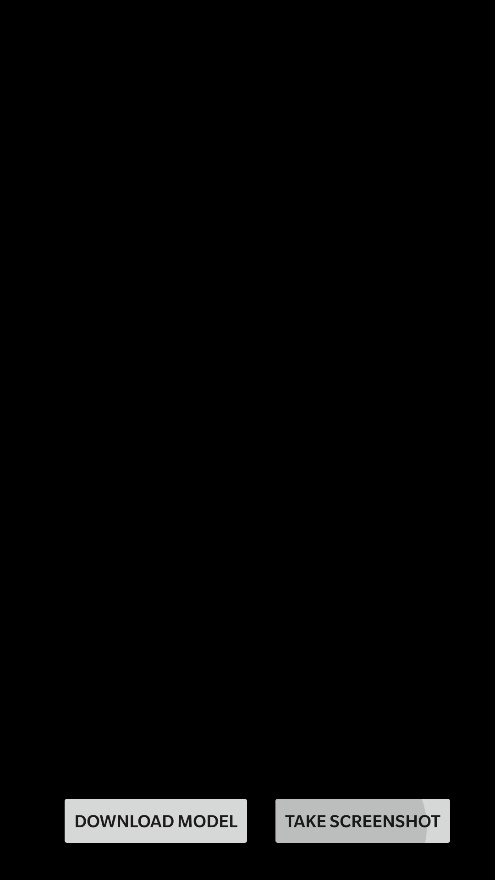
Description automatically generated**

**A picture containing text, indoor

Description automatically generated**

**A picture containing text

Description automatically generated**

****

* 1. **Use-Case Data**

1. **CODE**

***MAIN ACTIVITY:***

**package** com.example.secondar;  
  
**import** androidx.appcompat.app.AppCompatActivity;  
**import** androidx.recyclerview.widget.LinearLayoutManager;  
**import** androidx.recyclerview.widget.RecyclerView;  
  
**import** android.net.Uri;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
  
**import** com.google.ar.core.Anchor;  
**import** com.google.ar.sceneform.AnchorNode;  
**import** com.google.ar.sceneform.rendering.ModelRenderable;  
**import** com.google.ar.sceneform.ux.ArFragment;  
**import** com.google.ar.sceneform.ux.TransformableNode;  
  
**import** java.util.ArrayList;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
  
 **private** ArFragment arFragment;  
 **private** ArrayList<Integer> imagesPath = **new** ArrayList<Integer>();  
 **private** ArrayList<String> namesPath = **new** ArrayList<>();  
 **private** ArrayList<String> modelNames = **new** ArrayList<>();  
 AnchorNode anchorNode;  
 **private** Button btnRemove;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_main);  
  
 arFragment = (ArFragment)getSupportFragmentManager().findFragmentById(R.id.fragment);  
 btnRemove = (Button)findViewById(R.id.remove);  
 getImages();  
  
 arFragment.setOnTapArPlaneListener((hitResult, plane, motionEvent) -> {  
  
 Anchor anchor = hitResult.createAnchor();  
  
 ModelRenderable.builder()  
 .setSource(**this**,Uri.parse(Common.model))  
 .build()  
 .thenAccept(modelRenderable -> addModelToScene(anchor,modelRenderable));  
  
 });  
  
  
 btnRemove.setOnClickListener(view -> removeAnchorNode(anchorNode));  
 }  
  
 **private void** getImages() {  
  
 imagesPath.add(R.drawable.table);  
 imagesPath.add(R.drawable.bookshelf);  
 imagesPath.add(R.drawable.lamp);  
 imagesPath.add(R.drawable.odltv);  
 imagesPath.add(R.drawable.clothdryer);  
 imagesPath.add(R.drawable.chair);  
 namesPath.add(**"Table"**);  
 namesPath.add(**"BookShelf"**);  
 namesPath.add(**"Lamp"**);  
 namesPath.add(**"Old Tv"**);  
 namesPath.add(**"Cloth Dryer"**);  
 namesPath.add(**"Chair"**);  
 modelNames.add(**"table.sfb"**);  
 modelNames.add(**"model.sfb"**);  
 modelNames.add(**"lamp.sfb"**);  
 modelNames.add(**"tv.sfb"**);  
 modelNames.add(**"cloth.sfb"**);  
 modelNames.add(**"chair.sfb"**);  
  
 initaiteRecyclerview();  
 }  
  
 **private void** initaiteRecyclerview() {  
  
 LinearLayoutManager layoutManager = **new** LinearLayoutManager(**this**,LinearLayoutManager.HORIZONTAL,**false**);  
 RecyclerView recyclerView = (RecyclerView)findViewById(R.id.recyclerview);  
 recyclerView.setLayoutManager(layoutManager);  
 RecyclerviewAdapter adapter = **new** RecyclerviewAdapter(**this**,namesPath,imagesPath,modelNames);  
 recyclerView.setAdapter(adapter);  
  
 }  
  
 **private void** addModelToScene(Anchor anchor, ModelRenderable modelRenderable) {  
  
 anchorNode = **new** AnchorNode(anchor);  
 TransformableNode node = **new** TransformableNode(arFragment.getTransformationSystem());  
 node.setParent(anchorNode);  
 node.setRenderable(modelRenderable);  
 arFragment.getArSceneView().getScene().addChild(anchorNode);  
 node.select();  
 }  
  
 **public void** removeAnchorNode(AnchorNode nodeToremove) {  
 **if** (nodeToremove != **null**) {  
 arFragment.getArSceneView().getScene().removeChild(nodeToremove);  
 nodeToremove.getAnchor().detach();  
 nodeToremove.setParent(**null**);  
 nodeToremove = **null**;  
 }  
 }  
}

***RECYCLERVIEW ADAPTER:***

**package** com.example.secondar;  
  
**import** android.content.Context;  
**import** android.graphics.drawable.Drawable;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.ImageView;  
**import** android.widget.TextView;  
  
**import** androidx.annotation.NonNull;  
**import** androidx.recyclerview.widget.RecyclerView;  
  
**import** java.util.ArrayList;  
  
**public class** RecyclerviewAdapter **extends** RecyclerView.Adapter<RecyclerviewAdapter.ViewHolder>{  
  
 **private** ArrayList<String> **textNames** = **new** ArrayList<>();  
 **private** ArrayList<Integer> **imagesPath** = **new** ArrayList<>();  
 **private** Context **context**;  
 **private** ArrayList<String> **modelNames** = **new** ArrayList<>();  
  
  
 **public** RecyclerviewAdapter(Context context,ArrayList<String> textNames, ArrayList<Integer> imagesPath,ArrayList<String> modelNames) {  
 **this**.**textNames** = textNames;  
 **this**.**imagesPath** = imagesPath;  
 **this**.**modelNames** = modelNames;  
 **this**.**context** = context;  
 }  
  
 @NonNull  
 @Override  
 **public** ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, **int** viewType) {  
  
 View view = LayoutInflater.*from*(parent.getContext()).inflate(R.layout.***layout\_list\_item***,parent,**false**);  
 **return new** ViewHolder(view);  
 }  
  
 @Override  
 **public void** onBindViewHolder(@NonNull ViewHolder holder, **int** position) {  
 holder.**imageView**.setImageResource(**imagesPath**.get(position));  
 holder.**textView**.setText(**textNames**.get(position));  
  
 holder.**imageView**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 Common.*model* = **modelNames**.get(position);  
 }  
 });  
 }  
  
 @Override  
 **public int** getItemCount() {  
 **return imagesPath**.size();  
 }  
  
 **public class** ViewHolder **extends** RecyclerView.ViewHolder{  
  
 ImageView **imageView**;  
 TextView **textView**;  
  
 **public** ViewHolder(@NonNull View itemView) {  
 **super**(itemView);  
  
 **imageView** = itemView.findViewById(R.id.***imageview***);  
 **textView** = itemView.findViewById(R.id.***text***);  
 }  
 }  
}

* 1. **Limitation of the System:**
  2. **Future Work:**

In future our “DECORUM” dataset and scope will be scalable. The user might not only be able to try out different interior designing products but they can also try out this application by trying on other e-commerce products too like clothing, electronic items, watches etc. It can also be used in various industries like in auto mobile, medical science etc. New technology may come into existence in future that will help in developing 3D models automatically.

* 1. **Conclusion:**