CSC1097

FINAL YEAR PROJECT

ARrangeIt

Technical Specification

Jade Hudson - 21706905

Sruthi Santhosh - 21377986

Dr. Hyowon Lee

15/04/2024

TABLE OF CONTENTS

1. Introduction 3
2. Overview 3
3. Glossary 3
4. Motivation 3
5. Research 3
6. System Architecture/Design 4
    6.1 Frontend 4
        6.1.1 AR Implementation 4
        6.1.2 Catalogue Loading (Java) 4
        6.1.3 Object Placement in AR (Java & Firebase) 5
        6.1.4 Moving Objects (Java) 5
        6.1.5 Rotating Objects (Java) 5
        6.1.6 Deletion (Java) 6
        6.1.7 Saving Room Layout Screenshots (Java & Firebase) 6
        6.1.8 Measurement Tools 7
    6.2 Backend 7
        6.2.1 Firebase Authentication 7
        6.2.2 Firebase Realtime Database Architecture 8
        6.2.3 Cloud Storage 8
8. Distribution of Functions 9
9. High-Level Overview 9
    9.1. Data Flow Diagram 10
    9.2. Object Model 10
10. Problems Solved 11
11. Results 11
12. Future Work 11
13. Reused or 3rd Party Components 11
    13.1 Code Implementation 11
    13.2. 3D Models 12
14. Installation Guide 13
    14.1 Prerequisites and Requirements: 13
    14.2 Step-by-Step Installation: 14
    14.3 Dependencies List and Version 16

1. Introduction

ARrangeIt is an Augmented Reality (AR) mobile application designed to assist users to visualise furniture and decor items in their real world prior to purchase. By integrating ARCore and Firebase technologies, ARrangeIt offers a new way to shopping experiences by overlaying 3D furniture models into real-world using smartphone cameras. The application fills critical usability pitfalls of existing AR shopping utilities, including targeting Android, multi item placement, scale in real-time, and space planning.

2. Overview

ARrangeIt offers a practical solution for both customers and furniture businesses. The app enables users to browse a furniture catalogue, place and customise virtual items, measure space dimensions, and save layouts. Businesses can benefit from integration by uploading 3D models to enhance customer engagement and reduce return rates.

3. Glossary

AR	Augmented Reality
ARCore	Google’s AR development platform for Android
ARKit	Apple’s augmented reality (AR) development framework for iOS devices
3D Model	A digital representation of a physical object
Firebase	Google’s platform for mobile and web app development
UI	User Interface
SDK	Software Development Kit

4. Motivation

The app addresses a growing need for improved customer decision-making in online furniture shopping. Existing AR solutions are limited by platform compatibility, lack of intuitive controls, or support for only single-item placements. ARrangeIt was designed to fill these gaps by providing Android users with a seamless, flexible, and multi-functional AR experience.

5. Research

Market analysis revealed that AR features significantly increase engagement and sales. Studies from DigitalBridge and Harvard Business Review show consumers are more likely to purchase when they can preview items. Competitive analysis highlighted that solutions like IKEA Place offer good AR visualisation but lack support for Android or multiple item placement.

6. System Architecture/Design

ARrangeIt follows a modular architecture with clearly defined layers: User Interface, AR Processing, Backend Integration, and Data Management. The UI is designed for ease of use, while the ARCore engine manages environment detection, surface tracking, and object anchoring. Firebase handles secure storage and authentication.

The system is divided into modular components:

- ➢ UI Module: Login, catalogue browsing, AR view, and saved layouts
- ➢ AR Module: Manages AR functionalities (placement, room scanning, movement, rotation)
- ➢ Backend Module: Handles Firebase integration (authentication, data storage)
- ➢ Furniture Data Module: Handles catalogue data retrieval, filtering, searching
- ➢ System Services: Session management, error logging, notifications

6.1 Frontend

- ➢ Developed using Android Studio in Java
- ➢ AR view integrates with ARCore SDK
- ➢ Interfaces include login screen, catalogue, AR scene, Object placement, save/load saved layout, measurement tool.

6.1.1 AR Implementation

```
arFragment = (ArFragment)
   getSupportFragmentManager().findFragmentById(R.id.arFragment);
```

- ➢ Initializes the AR Fragment which provides the AR SceneView and handles ARCore session management.

6.1.2 Catalogue Loading (Java)

```
if (fragmentContainer.getVisibility() == View.GONE) {
    // Shows catalogue as overlay
    getSupportFragmentManager().beginTransaction()
            .replace(R.id.fragment_container, new
   FurnitureCatalogueFragment())
            .commit();
    fragmentContainer.setVisibility(View.VISIBLE);
} else {
    // Hides catalogue
    getSupportFragmentManager().popBackStack();
    fragmentContainer.setVisibility(View.GONE);
}
```

- ➢ The catalogue appears as a separate UI panel while the AR camera continues to run behind it.
- ➢ Uses Android's FragmentManager to handle the overlay

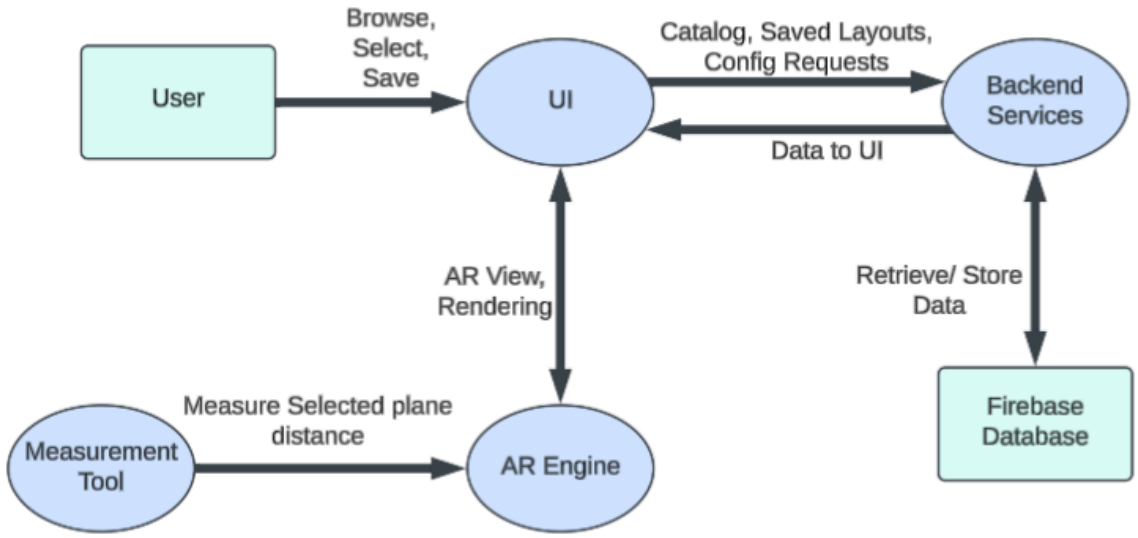
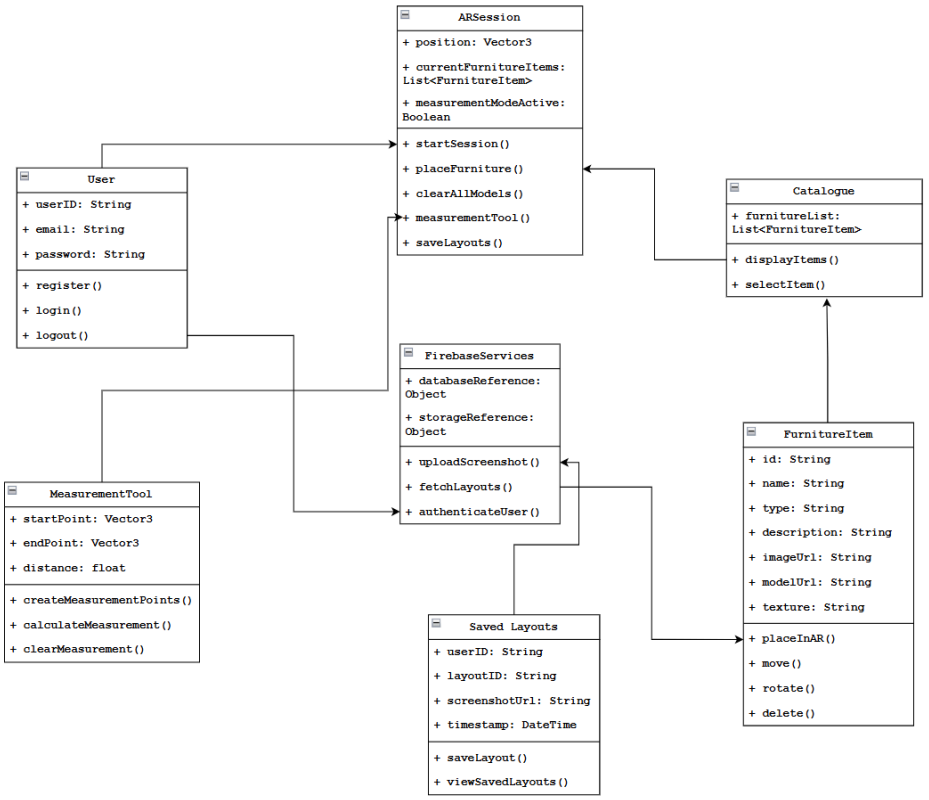
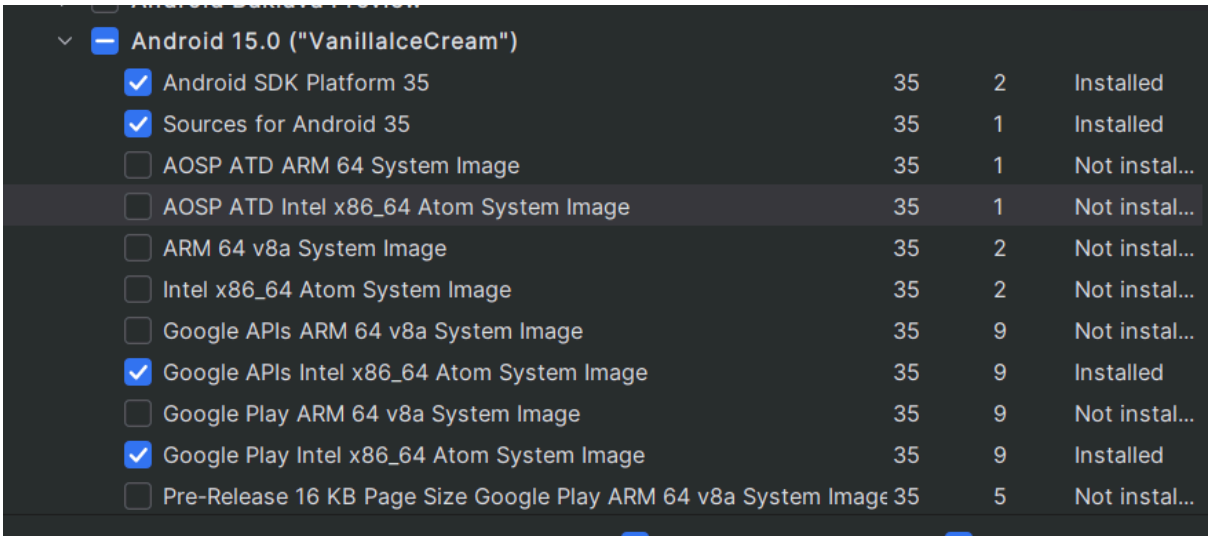
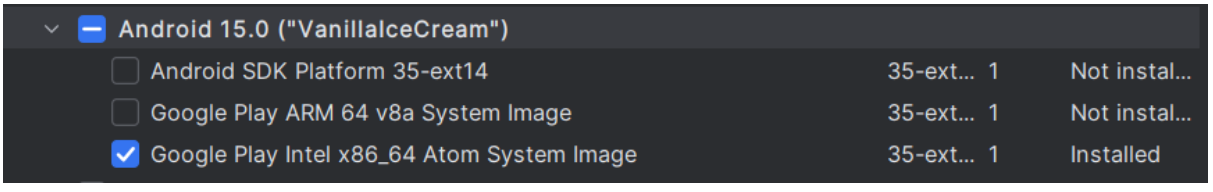
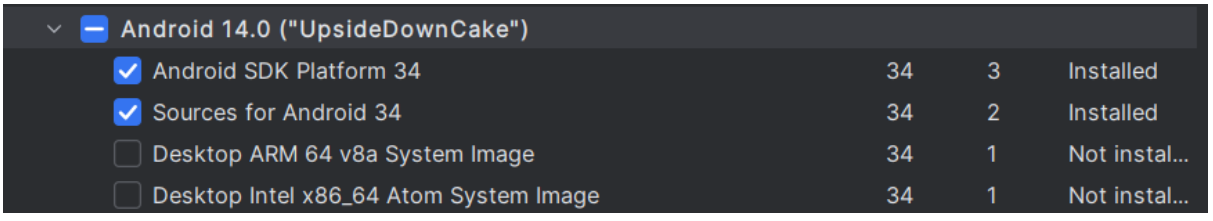
```
void loadFurnitureCatalogue() {
db.collection("furniture").get()
//loading the furniture details
}
```

- ➢ Queries the furniture collection in Firebase Firestore
- ➢ Retrieves all documents in one batch

6.1.3 Object Placement in AR (Java & Firebase)

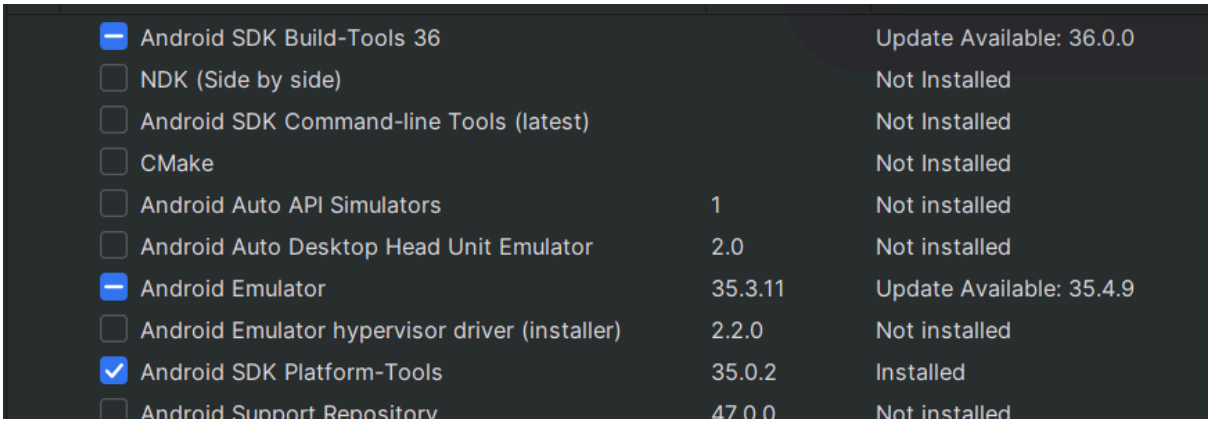
```
```private void loadModelFromFirebase(String modelUrl)
private void buildModel(File file)
```
```

- ➢ Download GLB models from Firebase Storage
- ➢ Creates Sceneform Renderables

```
```if (selectedFurnitureRenderable == null) return;

    Anchor anchor = hitResult.createAnchor();
    AnchorNode anchorNode = new AnchorNode(anchor);
    anchorNode.setParent(arFragment.getArSceneView().getScene());

    currentFurnitureNode = new
   TransformableNode(arFragment.getTransformationSystem());
    currentFurnitureNode.setParent(anchorNode);
    currentFurnitureNode.setRenderable(selectedFurnitureRenderable);
    currentFurnitureNode.setLocalScale(new Vector3(0.5f, 0.5f, 0.5f));
```
```

- ➢ Creates an anchor at the tapped position
- ➢ Attaches a TransformableNode to the anchor that holds the 3D model
- ➢ Uses Sceneform's built-in transformation system for manipulation

6.1.4 Moving Objects (Java)

```
```private void setRotateMode(boolean rotateMode) {
    isRotateMode = rotateMode;
    if (currentFurnitureNode != null) {
        // Enable translation (moving)

   currentFurnitureNode.getTranslationController().setEnabled(!rotateMode
   );
        // ... other controllers disabled
    }
}
```
```

- ➢ Uses Sceneform's bulti-in translation controller
- ➢ Activated when not in rotate mode

6.1.5 Rotating Objects (Java)

```
```currentFurnitureNode = new
   TransformableNode(arFragment.getTransformationSystem()) {
    @Override
    public boolean onTouchEvent(HitTestResult hitTestResult, MotionEvent
   motionEvent) {
        if (isRotateMode && motionEvent.getPointerCount() == 1) {
            // Handle rotation based on touch movement
```
```

```
            float deltaX = currentX - lastTouchX[0];
            Quaternion additionalRotation = Quaternion.axisAngle(
                new Vector3(0, 1, 0), // Rotate around Y axis
                rotationDegrees
            );
            setLocalRotation(Quaternion.multiply(currentRotation,
   additionalRotation));
            return true;
        }
        return super.onTouchEvent(hitTestResult, motionEvent);
    }
};
```

- ➢ Custom rotation implementation that rotates based on horizontal touch movement
- ➢ Uses quaternions for smooth around the Y-axis

```
private void showManipulationButtons()
private void setRotateMode()
```

- ➢ Shows/ hides the manipulation buttons
- ➢ Toggles between move/rotate modes

6.1.6 Deletion (Java)

```
private void deleteCurrentModel() {
    if (currentFurnitureNode != null) {
        AnchorNode parentAnchor = (AnchorNode)
   currentFurnitureNode.getParent();
        arFragment.getArSceneView().getScene().removeChild(parentAnchor);
        parentAnchor.setAnchor(null);
        currentFurnitureNode = null;
    }
}
```

- ➢ Removes the node from the scene
- ➢ Cleans up the anchor

6.1.7 Saving Room Layout Screenshots (Java & Firebase)

```
private void takeScreenshot(String layoutName) {
    Bitmap bitmap = Bitmap.createBitmap(
            arFragment.getArSceneView().getWidth(),
            arFragment.getArSceneView().getHeight(),
            Bitmap.Config.ARGB_8888);

    PixelCopy.request(arFragment.getArSceneView(), bitmap, copyResult -> {
        if (copyResult == PixelCopy.SUCCESS) {
            saveScreenshotToStorage(bitmap, layoutName);
        }
    }, new Handler());
}
```

- ➢ Uses Android's PixelCopy API to capture the AR view

➢ Saves to storage and Firebase

6.1.8 Measurement Tools

```
```private void handleMeasurementTap(HitResult hitResult, float screenX,
   float screenY) {
    Anchor anchor = hitResult.createAnchor();
    AnchorNode anchorNode = new AnchorNode(anchor);

    if (!isFirstPointSet) {
        firstAnchorNode = anchorNode;
        markerLineView.setFirstPoint(new PointF(screenX, screenY));
    } else {
        secondAnchorNode = anchorNode;
        markerLineView.setSecondPoint(new PointF(screenX, screenY));
        calculateDistance(); // Computes 3D distance between points
    }
}

private void calculateDistance() {
    Vector3 firstPos = firstAnchorNode.getWorldPosition();
    Vector3 secondPos = secondAnchorNode.getWorldPosition();
    float distance = Vector3.subtract(firstPos, secondPos).length();
    markerLineView.setDistanceText(String.format("%.1f cm", distance *
   100));
}
```
```

➢ Uses MarkerLineView for visual line
➢ It computes distance between two points

6.2 Backend

Firebase Realtime Database, Authentication, and Cloud Storage

6.2.1 Firebase Authentication

```
```firebaseAuth.signInWithEmailAndPassword(email, password)
                        .addOnCompleteListener(new
   OnCompleteListener<AuthResult>() {
                    @Override
                    public void onComplete(@NonNull Task<AuthResult> task)
   {
                        if(task.isSuccessful()) {
                            Toast.makeText(MainActivity.this, "Login
   Successful!", Toast.LENGTH_SHORT).show();
                            Intent intent = new Intent(MainActivity.this,
   ARCorePage.class);
                            startActivity(intent);
                            finish();
                        }
```
```

```
```firebaseAuth.createUserWithEmailAndPassword(email,
   password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
                    @Override
```
```

```
                    public void onComplete(@NonNull Task<AuthResult> task)
    {
                        if (task.isSuccessful()) {
                            Toast.makeText(RegisterPage.this,
    "Registration Successful!", Toast.LENGTH_SHORT).show();
                            Intent intent = new Intent(RegisterPage.this,
    MainActivity.class);
                            startActivity(intent);
                            finish();
                        }
```

- ➢ Manages user login/signup
- ➢ Email/Password authentication
- ➢ Automatically persists login state
- ➢ Provides UID for user-specific data partitioning

6.2.2 Firebase Realtime Database Architecture

```
rules_version = '2';
service firebase.storage {
  match /b/{bucket}/o {
    match /models/{allPaths=**} {
      allow read: if request.auth != null;
    }
    match /screenshots/{userId}/{allPaths=**} {
      allow read, write: if request.auth != null && request.auth.uid ==
   userId;
    }
  }
}
```

- ➢ Utilises a NoSQL, tree-structured database to store and sync data in real time across clients.
- ➢ Enables real-time syncing, ensuring updates (e.g. furniture placement) reflect instantly across user devices.
- ➢ Implements security rules to restrict access:
    - ○ Users can only read/write their own data under users and layouts.
    - ○ Catalogue data under catalogue is read-only for all users.

6.2.3 Cloud Storage

```
FirebaseFirestore.getInstance().collection("savedLayouts")
    .add(layoutData)
    .addOnSuccessListener(documentReference -> {
        Toast.makeText(this, "Layout saved!", Toast.LENGTH_SHORT).show();
    });
```

```
FirebaseFirestore.getInstance().collection("savedLayouts")
    .whereEqualTo("userId", user.getUid())
```

```
        .orderBy("timestamp", Query.Direction.DESCENDING)
        .get()
        .addOnSuccessListener(queryDocumentSnapshots -> {
            for (DocumentSnapshot document : queryDocumentSnapshots) {
                ScreenshotItem item = new ScreenshotItem();
                item.setImageUrl(document.getString("screenshotUrl"));
                item.setName(document.getString("layoutName"));

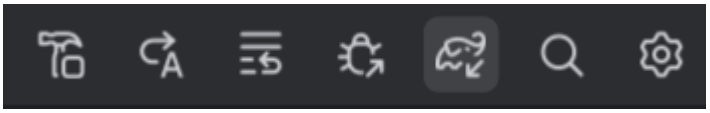
                // Convert Firestore timestamp to date
                Timestamp timestamp = document.getTimestamp("timestamp");
                item.setDate(new SimpleDateFormat("MMM dd, HH:mm")
                    .format(timestamp.toDate()));
            }
            adapter.notifyDataSetChanged();
        });
```

- ➢ Stores 3D models and layout screenshots
- ➢ GLB format recommended for AR (compact binary)
- ➢ Cache models locally after first download
- ➢ Use Firebase Storage Security Rules

8. Distribution of Functions

- ➢ Authentication: Login, account creation and deletion.
- ➢ Catalogue Browsing: View Furniture, Filter and search furniture.
- ➢ AR Placement: Place, move, rotate and delete furniture.
- ➢ Measurement: Calculate/Measure distance between two selected points.
- ➢ Saving Layouts: Saves and Store multiple layout designs.

9. High-Level Overview

- ➢ UI Module: Handles user input (touch, gestures), displays furniture, receives data from Catalogue and AR modules.
- ➢ Catalogue Module: Retrieves and filters furniture items from Firebase.
- ➢ AR Module: Manages placement, scaling, and interaction of 3D objects in AR.
- ➢ Backend Module: Tracks user layouts, manages session data, and communicates with Firebase.
- ➢ Firebase Backend: Central cloud storage for user data, catalogue, and layout sessions.
- ➢ ARCore Engine: Detects surfaces, renders 3D models.

9.1. Data Flow Diagram



9.2. Object Model



10. Problems Solved

- Android Compatibility: Built for ARCore-supported Android devices, ensuring wide device support and seamless AR functionality.
- Multi-Item AR Placement: Enables users to place, arrange, and interact with multiple 3D furniture models in a single AR session.
- Model Placement Precision: Utilised ARCore hit-testing and anchors to ensure accurate and stable object placement on detected surfaces.
- Scalable Object Interaction: Implemented gesture-based interactions (move, rotate), allowing smooth manipulation of multiple objects..
- Performance: Reduced memory footprint and load times by optimising 3D asset sizes and managing rendering efficiently.
- Catalogue Overlay UI: Integrated an in-app catalogue overlay for browsing and selecting furniture models without leaving the AR view.
- Layout Persistence: Implemented layout saving functionality, allowing users to store and review their AR room arrangements at a later time.

11. Results

- Fully functional prototype completed with multi-item placement, measurement tool, and real-time object manipulation (including movement and rotation).
- Positive feedback from usability tests indicating high engagement and intuitive interaction.
- Demonstrated strong potential to reduce user uncertainty and enhance decision-making during furniture selection and interior planning.

12. Future Work

- **iOS Support**: Extend compatibility to iOS devices by integrating Apple's ARKit framework.
- **AI-Powered Layout Suggestions**: Implement machine learning algorithms to recommend optimal furniture arrangements based on room dimensions and style preferences.
- **E-commerce Integration**: Allow users to purchase selected furniture directly through partnerships with online retailers.
- **Room Scanning via LiDAR**: Leverage LiDAR sensors (on supported devices) for enhanced spatial understanding and automatic room mapping.
- **Real-Time Collaboration**: Enable multiple users to co-design and edit room layouts simultaneously in a shared AR session.
- **Editable Saved Layouts**: Evolve the current screenshot-based saved layouts into fully interactive, editable AR scenes, allowing users to revisit and modify previously arranged furniture.

13. Reused or 3rd Party Components

13.1 Code Implementation

- ARCore SDK (Google) – AR functionality

- Sceneform Implementation tutorial (Youtube) - https://www.youtube.com/watch?v=2YtlIiUKNdA&t=26s
- Firebase SDK – Authentication and database services

13.2. 3D Models

Public domain /Creative Commons licensed assets

- Aidian Corner Storage Sofa Bed, Night Blue: "Aidian Corner Storage Sofa Bed, Night Blue" (https://skfb.ly/6ALVR) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Alana Bedside Table, Brushed Brass: "Alana Bedside Table, Brushed Brass" (https://skfb.ly/6xSR6) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Arm chair / Furniture: "Arm chair / Furniture" (https://skfb.ly/o6MMo) by maxsbond.work is licensed under Creative Commons Attribution (http://creativecommons.org/licenses/by/4.0/).
- Austin Floor Lamp, Copper: "Austin Floor Lamp, Copper" (https://skfb.ly/6xSRt) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Bed: "Bed" (https://skfb.ly/6Un8t) by Maxmalow is licensed under Creative Commons Attribution (http://creativecommons.org/licenses/by/4.0/).
- Bed: "Bed" (https://skfb.ly/6ZJuI) by Susidko Studio is licensed under Creative Commons Attribution (http://creativecommons.org/licenses/by/4.0/).
- Boone Dining Table, Concrete Resin Top: "Boone Dining Table, Concrete Resin Top" (https://skfb.ly/6xSSS) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Branagh Large Ottoman, Pearl Grey: "Branagh Large Ottoman, Pearl Grey" (https://skfb.ly/6zSTu) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Desk: "Desk" (https://skfb.ly/6zPwZ) by charles.cla is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Edelweiss Extending Dining Table, Ash And White: "Edelweiss Extending Dining Table, Ash And White" (https://skfb.ly/6ywCz) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Elias Chest, White and pine: "Elias Chest, White and pine" (https://skfb.ly/6FY6T) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Elona Sideboard, Charcoal And Brass: "Elona Sideboard, Charcoal And Brass" (https://skfb.ly/6ywAJ) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Elona Wide Chest Of Drawers, Charcoal And Brass: "Elona Wide Chest Of Drawers, Charcoal And Brass" (https://skfb.ly/6ywAE) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).

- Esme Compact Media Unit, Ash: "Esme Compact Media Unit, Ash" (https://skfb.ly/6ywvt) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Fedor_SAT_IZBA_v7(v4)_BEZ_LED: "Fedor_SAT_IZBA_v7(v4)_BEZ_LED" (https://skfb.ly/oNvFw) by KRONZI is licensed under Creative Commons Attribution (http://creativecommons.org/licenses/by/4.0/).
- Helena Sofabed, Plush Teal Velvet: "Helena Sofabed, Plush Teal Velvet" (https://skfb.ly/6ywID) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- IKEA "ALEX" Drawer: "IKEA "ALEX" Drawer" (https://skfb.ly/o6IDC) by Hekxyit is licensed under Creative Commons Attribution (http://creativecommons.org/licenses/by/4.0/).
- Kolton Rocking Chair, Marl Grey: "Kolton Rocking Chair, Marl Grey" (https://skfb.ly/6yw8J) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Kubrick Wingback Armchair: "Kubrick Wingback Armchair" (https://skfb.ly/6xK6n) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Matryoshka Chest,White: "Matryoshka Chest,White" (https://skfb.ly/6YFQG) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Reclaimed TV Stand / Furniture: "Reclaimed TV Stand / Furniture" (https://skfb.ly/o6NPC) by maxsbond.work is licensed under Creative Commons Attribution (http://creativecommons.org/licenses/by/4.0/).
- Roscoe King Size Bed, Cool Grey: "Roscoe King Size Bed, Cool Grey" (https://skfb.ly/6A7vE) by MADE.COM is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Simple dining table: "Simple dining table" (https://skfb.ly/6Xnzp) by DailyArt is licensed under Creative Commons Attribution-NonCommercial (http://creativecommons.org/licenses/by-nc/4.0/).
- Standing Lamp: "Standing Lamp." (https://skfb.ly/6FVJT) by Pippa is licensed under Creative Commons Attribution (http://creativecommons.org/licenses/by/4.0/).
- Table: "Table" (https://skfb.ly/oRHYn) by Keskin is licensed under Creative Commons Attribution (http://creativecommons.org/licenses/by/4.0/).
- Wooden Sofa: "Wooden Sofa" (https://skfb.ly/oxHUP) by Watndit is licensed under Creative Commons Attribution (http://creativecommons.org/licenses/by/4.0/).

14. Installation Guide

14.1 Prerequisites and Requirements:

- Android Studio (latest)
- Android device with ARCore support (Pixel 6a and Samsung Galaxy A22 tested)
- Java SDK 11+
- Firebase project setup
- AR Apps in Android Emulator: https://developers.google.com/ar/develop/c/emulator

14.2 Step-by-Step Installation:

1. Open Android Studio and the run the following command to clone the repository from ARrangeIt repository on Gitlab

```git clone https://gitlab.computing.dcu.ie/hudsonj5/2025-csc1097-hudsonj-5-santhos-2.git ```

Alternately, use a terminal to clone this repository and open the folder in Android Studio.

2. Once the repository is cloned and opened in Android Studio, make sure the following Packages are installed.

SDK Platform

- ❖ Android 15.0 (“VanillaCream”)
  - ➢ Android SDK Platform 35
  - ➢ Sources for Android 35
  - ➢ Google APIs Intel x86_64 Atom System Image
  - ➢ Google Play Intel x86_64 Atom System Image
- ❖ Android 14.0 (“UpsideDownCake”)
  - ➢ Android SDK Platform 34
  - ➢ Sources for Android 34






```

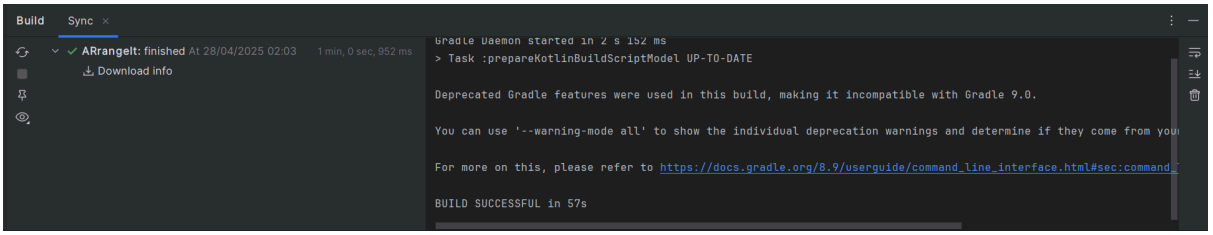
SDK Tools

- ➢ Android SDK Build-Tools 36
- ➢ Android Emulator
- ➢ Android SDK Platform-Tools

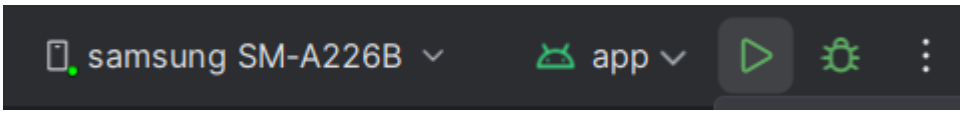


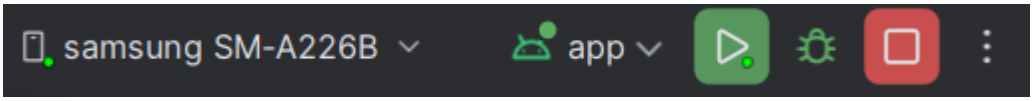
3. Sync Gradle and resolve dependencies using "**Sync Project with Gradle files**" button.

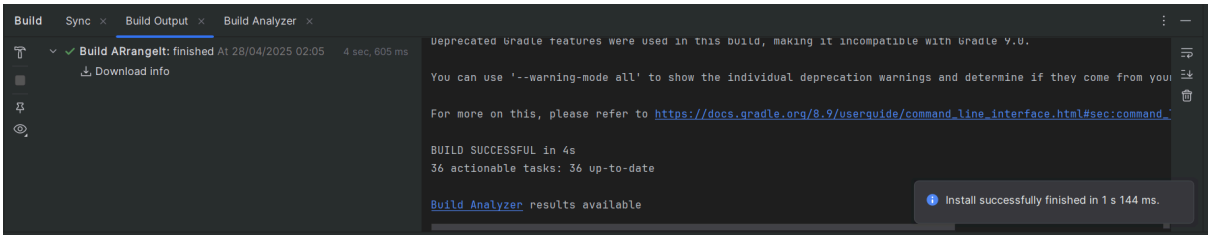




4. Deploy APK to Android device using "**Run app**" button.







5. To run AR Apps in Android Emulator to test AR test scenarios without a physical devices follow the instructions in https://developers.google.com/ar/develop/c/emulator.

14.3 Dependencies List and Version

1. ARCore SDK: v1.31.0
2. Firebase Authentication: v21.0.1
3. Firebase Realtime Database: v20.0.5
4. Android Gradle Plugin: v7.2.1
5. Java SDK: v11+