ABSTRACT

The AR Furniture App revolutionizes the way users engage with furniture shopping by integrating augmented reality technology to enhance decision-making. This innovative application allows users to visualize and customize furniture in their own living spaces through their smartphones or tablets.

By simply scanning their environment, users can place virtual furniture items, adjust sizes, and explore various designs and colors, ensuring that each piece harmoniously fits within their home's aesthetic. The app also features a user-friendly interface, comprehensive product information, and a seamless shopping experience, including direct links to purchase. This unique blend of practicality and convenience empowers consumers to make informed choices, ultimately transforming the furniture shopping experience into an interactive and personalized journey

Beyond individual purchases, AR simplifies room planning by allowing users to visualize multiple items together and create cohesive layouts for their spaces. This not only streamlines the buying process but also strengthens brand trust by offering transparency and reducing uncertainties. Ultimately, AR integration in furniture apps aims to improve user satisfaction, increase loyalty, and drive business growth by redefining the way customers shop for furniture online. Additionally, by closing the gap between the actual product and the expectations of the buyer, AR lowers return rates. Visualizations that are accurate reduce the amount of discontent brought on by improper sizing or mismatched aesthetics. Higher conversion rates result from buyers feeling more confident about their purchases as a result of this. Additionally, augmented reality technology gives the app a competitive edge by offering unique and user-friendly functions that set it apart in a crowded industry.

CHAPTER 1

INTRODUCTION

1.1 ABOUT THE PROJECT

Implementing Augmented Reality (AR) in an e-commerce app for furniture can significantly enhance the shopping experience for customers. Key features of such an application include virtual room visualization, where users can utilize their smartphone or tablet cameras to see how different furniture pieces will look in their living spaces, allowing them to place items like sofas, chairs, and tables in their rooms while adjusting their sizes and positions as needed. Additionally, the app can provide interactive 3D models that users can rotate and view from various angles, helping them better understand the dimensions and design of the furniture.

Customizable options allow users to personalize furniture pieces within the app by changing colors, fabrics, and finishes, thus enhancing their shopping experience. Incorporating measurement tools will enable users to ensure that the furniture will fit in their space by measuring distances directly with their devices. Furthermore, style integration through suggestions or style guides can show how different pieces work together in a space, perhaps featuring curated collections or themed setups. Lastly, including a feature for customer reviews and testimonials will let users view feedback and photos from others who have purchased the same items, particularly in their homes, providing additional reassurance. The benefits of an AR-focused e-commerce app are significant.

It improves customer decision-making by allowing users to visualize how furniture looks in their spaces, which reduces the chances of buyer's regret. An engaging AR experience can also keep users invested in the app for longer periods, leading to higher conversion rates. Moreover, when customers can accurately envision how the furniture will fit in their homes, the likelihood of returns decreases. Offering AR capabilities can help differentiate the app from competitors, making it more appealing to tech-savvy shoppers.

In conclusion, an AR-focused e-commerce app for furniture can revolutionize online shopping by providing customers with a more immersive and satisfying experience. By incorporating these features, retailers can enhance customer confidence and satisfaction while boosting sales and minimizing returns.

1.1 EXISTING SYSTEM

One major problem with many furniture e-commerce apps is that it can be difficult to determine the comfort and quality of objects without actually seeing them. Photographs and descriptions are frequently relied upon by users, although they might not always give a complete picture of the object. If the furniture is delivered with material quality, comfort, or general appearance below expectations, this can cause disappointment. Additionally, some buyers may be discouraged from making a purchase if they are unable to visually inspect the things before making a purchase.

1.2 DRAWBACK OF EXISTING SYSTEM

- Misjudging fit and aesthetic can lead to higher returns.
- Users may feel unsure about their purchasing decisions without visual confirmation.
- Users must rely on imagination and descriptions, increasing the effort needed to make choices.
- Lack of interactive features makes the app less appealing
- Missed opportunities for personalized recommendations based on users' unique environments.
- The absence of AR can significantly limit the effectiveness and user satisfaction of a furniture e-commerce app.

1.3 PROPOSED SYSTEM

In a furniture app, augmented reality (AR) provides a number of noteworthy advantages that improve customers' purchasing experiences. Using the cameras on their smartphones, AR first lets users see how furniture would appear in their own living areas by superimposing 3D models in real time. Users will be able to view how various styles and colors blend in with their current decor in addition to being able to visualize the product in context. In addition, AR offers a more precise scale and dimension measurement, assisting customers in figuring out whether a piece of furniture would fit in a certain room of their house. As a result, there is less chance that you would buy things that are too big or tiny for the space, which will eventually increase customer satisfaction and decrease returns.

1.4 ADVANTAGES OF THE PROPOSED SYSTEM

- AR capabilities make shopping more dynamic and interesting

- The integration of AR tools can provide valuable data on user interactions and Preference, helping brands refine their offerings and marketing strategies.

- Offering AR features can set a furniture app apart from competitors, attracting users who are looking for innovative and modern shopping experiences.

- Customers are more confident in their selections when they can picture the furniture in their room, which may lead to higher conversion rates.

SUMMARY

This chapter describes the existing system and its disadvantages, such as the fact how the furniture app currently without augmented reality which increases the struggle of buying goods online. The detailed information regarding the system analysis will be presented in the following chapter 2.

CHAPTER 2

SYSTEM ANALYSIS

2.1 IDENTIFICATION OF NEED

By enabling consumers to view furniture in their homes in an immersive and interactive way before making a purchase, augmented reality (AR) is transforming the furniture app market. With augmented reality, users may place 3D models of couches, tables, chairs, and other furniture in their rooms to see how well they match the existing layout. This feature helps customers make informed decisions by assessing the furniture's size, design, and color in an actual environment. Through experimentation with different layouts and configurations, users may use AR to create their ideal environments, doing away with the need for guesswork or physical trial and error. AR's accurate room scaling and spatial perception eliminate any uncertainty regarding furniture's compatibility with a given area. This innovative technology not only improves customer satisfaction but also helps furniture brands stand out in a competitive market by providing a cutting-edge solution that bridges the gap between online and offline shopping. Integrating AR into a furniture app is not just a feature it's a powerful tool to build trust, boost sales, and transform the way people shop for their homes.

2.2 FEASABILITY STUDY

A feasibility study is a complete analysis that is conducted to evaluate the practicality and viability of a proposed project. The prime goal of a feasibility study is to determine whether the project is worth pursuing and if it can be successfully implemented within the defined constraints. Each structure has to be thought of in the developing of the project, as it has to servethe end user in friendly manner. Three considerations involved in feasibility are

- Technical Feasibility
- Operational Feasibility
- Economic Feasibility

2.2.1 TECHNICAL FEASIBILITY

The Technical Feasibility assess the technical characteristics of the project, including the availability of technology, infrastructure, and expertise required for successful implementation. The proposed system will be platform independent since it is being coded in java, xml and firebase with the help of android studio to implement, which have several developing tools for developing the application.

The cost and benefits analysis may be concluded that computerized system is favorable intoday’s fast-moving world. The assessment of technical feasibility must be based on online designof the system requirements in terms of input, output, files, programs and procedures.

This application enables a limitless number of users to browse and purchase furniture products. Furthermore, the system is highly adaptable and user-friendly, ensuring a smooth and intuitive experience for all users, whether they are purchasing furniture or managing their listings.

2.2.2 OPERATIONAL FEASIBILITY

The developed application is user friendly for the customers because they can easily recognize the purpose of this application. The purpose of operational feasibility isto govern whether the new system will be used if it developed and implemented or will there be resistance from the users that will take the possible application benefits. Also, it includes the levelof acceptance of the system by the user.

The proposed device having access to system to solves issues what happened in existing device. The current day-to-day operations of the organization can be fit into the system. Mainly operational feasibility should include on analysis of how the proposed system will affect the organizational structure and procedures. The developed app is user-friendly to the customers because they can easily view the product.

2.2.3 ECONOMIC FEASIBILITY

This application is developed by java, xml and firebase, which is one of the open source language and it can be used anywhere easily. The economic feasibility is carried out to check the economic impact that the system will have on the organization. Thus, the developedsystem is well within the budget and this was achieved because most of the technologies used arefreely available.

One of the key economic benefits of AR in e-commerce is its ability to reduce product return rates. Customers can make better-informed decisions when they see how a product will look or fit in their space, minimizing mismatched expectations. This not only saves operational costs associated with returns but also enhances profitability. Additionally, AR provides a competitive edge, differentiating the app in a crowded market and attracting tech-savvy customers, which can lead to market share expansion.

2.3 SOFTWARE REQUIREMENT SPECIFICATION

A declaration that describes the capability that a system needs in order to satisfy the user's requirements is known as a system requirement. The system requirements for specific machines, software or business operations are general. Taking it all the way down to the hardware and coding that operates the software. System requirements are the most efficient wayto address user needs while lowering implementation costs.

2.3.1 Hardware Requirements

The hardware for the system is selected considering the factors such as CPU processing speed, memory access, peripheral channel access speed, printed speed; seek time& relational dataof hard disk and communication speed etc.

Processer	:	AMD
Ryzen 5RAM	:	8 GB
Monitor	:	16 inch
Disk	:	512 GB
Keyboard	:	Optical

2.3.2 Software Requirements

The software for the project is selected considering the factors such as working frontend environment, flexibility in the coding language, database knowledge of enhances in backend technology etc.

FrontEnd	:	React Native
BackEnd	:	Java
Database	:	Firebase
Tools	:	VS Code

Frontend

A React Native front-end app is a mobile application built using the React framework and JavaScript. It enables developers to create cross-platform apps for iOS and Android with a single codebase, offering near-native performance. Components like Views, Text, and Buttons are used to design the UI, while state management tools like Redux or Context API handle app logic. React Native supports third-party libraries, APIs, and native modules for advanced functionality. Hot reloading allows developers to see real-time updates during development. Its flexibility, faster development cycle, and reusability make React Native a popular choice for mobile app development.

Features of React Native

The key features of React Native:

- **Cross-Platform Development**: Write one codebase that works on both iOS and Android.
- **Reusable Components:** Build UI components that can be reused across the app.
- **Hot Reloading:** Instantly see updates during development without recompiling the app.
- **Native Performance:** Combines native code with JavaScript for smooth, high-performance apps.

- **Modular Architecture**: Easy to maintain and update due to its modular structure.
- **Third-Party Plugin Support:** Extend app capabilities with plugins and libraries.
- **Large Community:** Access extensive documentation, tutorials, and community support.
- **Rich Ecosystem:** Seamless integration with APIs and tools like Redux for state management.
- **Live Reloading:** Reflects code changes in real time, improving development speed.
- **Cost-Effective**: Reduces development time and resources with shared code.

Backend

Java is a high-level, object-oriented programming language known for its platform independence, enabling programs to run on any system with a Java Virtual Machine (JVM). Designed for versatility and reliability, it supports features like automatic memory management, strong type-checking, and built-in exception handling. Java is widely used for app development, mobile apps (especially Android), enterprise systems, and big data processing, thanks to its rich APIs and frameworks. Its emphasis on security, scalability, and multithreading makes it suitable for applications ranging from simple desktop programs to large-scale distributed systems.

Features of Java

- **Object-Oriented:** Java is based on object-oriented programming (OOP) principles, such as encapsulation, inheritance, polymorphism, and abstraction. Everything in Java revolves around objects and classes, which makes it modular, reusable, and easier to maintain.

- **Platform-Independent:** Java follows the "Write Once, Run Anywhere" (WORA) philosophy.
- Java programs are compiled into bytecode, which can run on any device with a **Java Virtual Machine (JVM)**, regardless of the underlying operating system or hardware.

- **Open Source:** Java is open source, making it freely accessible to developers and supported by a vast community that continuously improves the language and its tools.

- **Dynamic and Extensible**: Java is designed to adapt to evolving environments by dynamically linking new classes, libraries, and methods during runtime. This makes it suitable for applications that need to update or extend functionality without restarting.

- **Multithreaded**: Java supports multithreading, allowing programs to perform multiple tasks simultaneously within a single program. It simplifies the development of highly interactive and responsive applications.

- **Secure**: Java is designed with security features like bytecode verification, secure class loading, and runtime checks. It provides a secure environment for running untrusted code, such as applets or applications downloaded from the internet.

- **Garbage Collection**: Java automatically manages memory through garbage collection, which eliminates unused objects and frees developers from manually handling memory allocation and deallocation.

Database

Firebase began as a YC11 startup and is now a Backend-as-a-Service. It developed into a cutting-edge Google Cloud Platform app development platform. A list of items can be stored in a tree format using Firebase, a No SQL JSON database that operates in real time. Data can be synchronize devices. The Google- backed application development tool Google Firebase enables developers to create apps for IOS, Android, and other platforms. Firebase offers several capabilities to manage statistics, report and fix app errors, and create marketing and product experiments. A real-time database, user authentication, and hosting are Firebase's three primary services. With the aid of the Firebase IOS SDK, we can use these services to develop apps without having to write any server code. One of Firebase's standout features is its real-time data synchronization, where any updates to the database are instantly reflected across all connected clients, eliminating the need for manual polling and reducing latency. Offline capabilities further enhance the user experience by allowing data changes to be stored locally when offline and synchronized once the app reconnects. Firebase also integrates with other services such as Authentication for secure user access, Cloud Functions for backend logic, and Analytics for tracking user behavior. With robust security rules, scalability, and cross-platform support, Firebase is an excellent choice for modern applications requiring dynamic, real-time, and user-friendly database solutions.

Features of Fire base

- **Real time Database:** A Nosql cloud database that stores data as JSON. Enables real-time synchronization across all connected clients Supports offline data access by caching changes locally and syncing them when reconnected.

- **Cloud Firestore:** A scalable, flexible Nosql database that stores data in collections and documents Supports complex queries, hierarchical data structures, and real-time updates.

- **Remote Configuration:** Enables dynamic app updates without requiring a new release. Allows developers to change app behavior and appearance in real-time.

- **Integration and Cross-Platform Support:** Firebase SDKs support Android, ios, Web, and Unity platforms. Seamlessly integrates with Google Cloud and third-party tools.

SUMMARY

The system requirements are fully explained in detail for each piece of hardware and software that was used to meet the user and system requirements The detailed information regarding the system design will be presented in the following chapter 3.

CHAPTER 3

SYSTEM DESIGN

3.1 MODULE DESCRIPTION

A module description provides detailed information about the module and its supported components, which is accessible in different manners.

The project contains the following module:

- Product
- Category
- Filter
- Cart
- Payment

Product Module:

This is the most basic and the vital module for AR Ecommerce app involves. It allows sellers to list their products with detailed information such as Product Name, Category, Dimensions, Material, Price, and a Detailed Description of the product. The module also enables sellers to upload high- resolution images and 3D models to showcase the furniture effectively, helping customers make informed purchasing decisions.

Category Module:

Creating a menu module involves designing a user-friendly navigation system that allows users to easily browse and access various features and sections of the platform. The menu should organize furniture into main categories like **Sofas, Tables, Beds, Chairs, and Storage** for easy browsing. Each category can include subcategories (e.g., material type, style, or price range) to refine searches further.

Filter Module:

The Filter Module is a core feature designed to enhance product discovery and help customers effortlessly find the perfect furniture to suit their needs. This module provides an intuitive interface with advanced filtering options, allowing users to refine search results based on specific preferences.

Key filtering options include category filters, which allow customers to browse by furniture type such as sofas, beds, dining tables, chairs, and storage units. Users can also explore furniture by style and aesthetic, including modern, contemporary, traditional, rustic, industrial, or bohemian designs. The module supports filtering by dimensions, such as length, width, height, or specific size ranges tailored for small or large spaces.

Cart Module:

Implementing a cart module involves creating a seamless and user- friendly experience for buyers to add, manage, and complete their purchases. Enable buyers to specifythe quantity of each product they want to purchase. Provide a dedicated page where users can review the contents of their cart. Allow users to modify the quantity or remove items from their cart. Associate the cart with user accounts to ensure persistence across sessions and devices. Automatically update the cart total and product quantities as users make changes.

Payment Module:

Implementing a secure and reliable payment module is crucial for the success of an AR Ecommerce app. Secure Payment Gateways Integrate reliable and secure payment options, such as credit cards, digital wallets, or bank transfers. Ensure users receiveconfirmation of successful payments. Implement a messaging system to facilitate communication between buyers and sellers. Keep users informed about order status, new products, and important updates.

3.2 DATAFLOW DIAGRAM

The Data Flow Diagram provides information about the inputs and outputs of each entity and process itself.

LEVEL 0

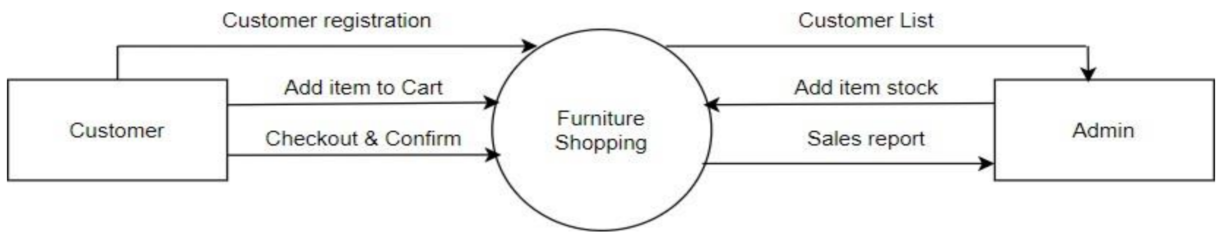


Figure 3.1
Dataflow Diagram Level 0

In Figure 3.1 customer and admin can Login to the application and make use of their respective process in the system. In this, admin can able to manage the product details. User can view the product description and can add the product to cart and can buy the farm product and they can make payment.

LEVEL 1

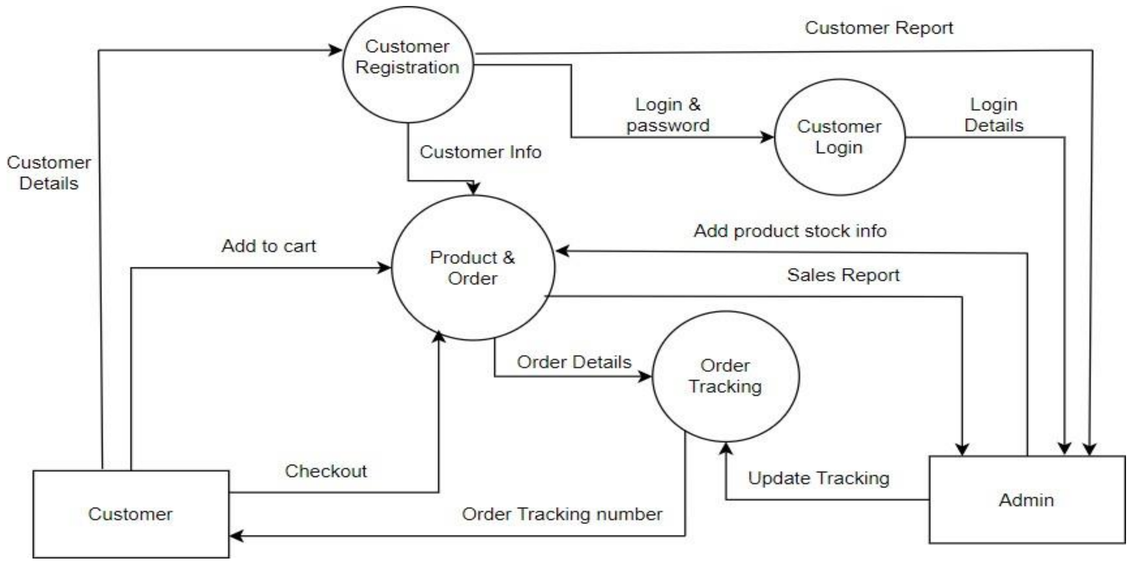
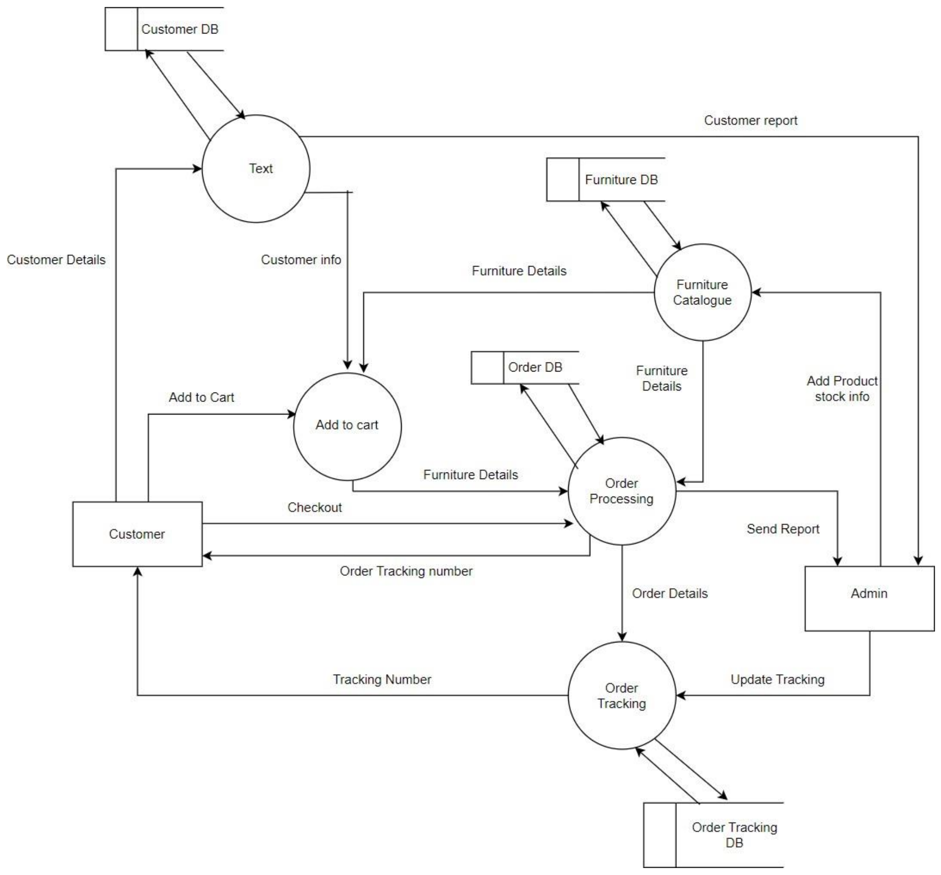


Figure 3.2
Dataflow Diagram Level 1

In Figure 3.2 Customer should register the application and after register customer can login by entering the valid details. After login user can add the product to cart. Here the admin has the previlage of tracking the order and having the customer report. Login details and product details are stored in firebase.

Level 2


Customer DB
Customer report
Text
Furniture DB
Customer Details
Customer info
Furniture Details
Furniture Catalogue
Order DB
Furniture Details
Add Product stock info
Add to Cart
Add to cart
Furniture Details
Order Processing
Checkout
Customer
Send Report
Order Tracking number
Order Details
Admin
Tracking Number
Order Tracking
Update Tracking
Order Tracking DB

3.3 DATABASE DESIGN

Database design is the organization of data according to a database model. The designer determines what data must be stored and how the data elements interrelate. With this information, they can begin to fit the data to the database model. Database managementsystem manages the data accordingly. The term database design can be used to describe many different parts of the design of an overall database system.

Principally, and most correctly, it can be thought of as the logical design of the basedata structure used to store the data. In an object database the entities and relationships map directly to object classes and named relationships. However, the term database design couldalso be used to apply to the overall process of designing, not just the base data structure, butalso the forms and queries used as part of the overall database application within the database management system.

Data Integration

In a database, information from several files is coordinated, accessed and operated upon as though it is in a single file. Logically, the information is centralized, physically, the data may be located on different devices, connected through data communication facilities. Data integrity means storing all data in one place only and determining how each applicationhas to access it. This approach results in more consistent information, one update being sufficient to achieve a new record status for all applications., which use it. This leads to less data redundancy, a reduction in the direct access storage requirement.

Data Independence

Data independence is the insulation of application programs from changing aspects of physical data organization. This objective seeks to allow changes in the content and organization of physical data without reprogramming of applications and to allow modifications to application program without recognizing the physic

If the database changes and expands over time, it is very important that the changes in one level should not affect the data at other levels of the database. The table needed for each module were designed and the specification of each and every column was given basedon the records and details collected during record specification of the system study, is shownin the below.

Data Security

Data security refers to the process of protecting data from unauthorized access and data corruption throughout its lifecycle. Data security includes data encryption, hashing, tokenization and key management practices that protect data across all applications and platforms. Data security means protecting digital data, such as those in a database, from destructive forces and form the unwanted actions of unauthorized users, such as cyberattack or a data breach.

Security requirements placed restrictions on the use of this application by the adminand customers of wireless Lan communicator only, control access to the data, provide different kinds of requirements to different people, require the use of passwords. Organizations around the globe are investing heavily in information technology, cyber security capabilities to protect their critical assets.

Whether an enterprise needs to protect a brand, intellectual capital and customer information or provide controls for critical infrastructure, that means for incident detection and response to protecting organizational interests have three common elements: people, process and technology. Solution that provides end-to-end encryption for email and mobile messaging, keeping personally identifiable information and personal health information secure and private.

Protects sensitive data captured at the browser from the point to the customer enterscardholder or personal data and keeps it protected through the ecosystem to the trusted hostdestination. Solution that provides an end-to-end data-centric approach to enterprise data pro

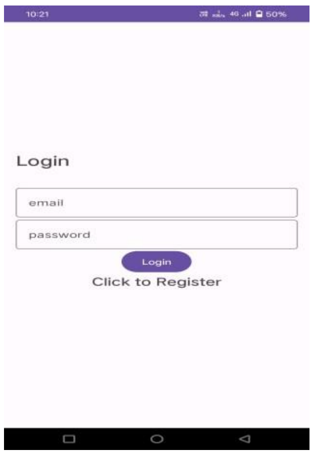
3.4 INPUT DESIGN

Input design is the process of converting user-originated inputs to a computer understandable format. Input design is one of the most expensive phase of the operation of computerized system and is often the major problem of a system. A large number of problems with a system can usually be tracked back to fault input design and method.

Every moment of input design should be analyzed and designed with utmost care. The decisions made during the input design are the project gives the low time consumption to make sensitive application made simple. Thus, the developed system is well within the budget. This was achieved because most of the technologies used are freely available. Only the customized product had to be purchased. In the project, the forms are designed with easy- to-use option. The coding is being done such that proper validation are made to get the prefect input. No error inputs are accepted.

Login form

This a user login form, the user can login with their registered mail id and passwordto access the application. After completing the registration, the user should give their correctregister details else the user cannot able to login.



The Figure 3.4 is user login form. Login form has mail id and password. The username should be in proper email format. If the fields are not filled, form will not be submitted. Password must contain at least 6 digit or characters. If the form is submitted successfully the page redirected towards home page.

Product page:

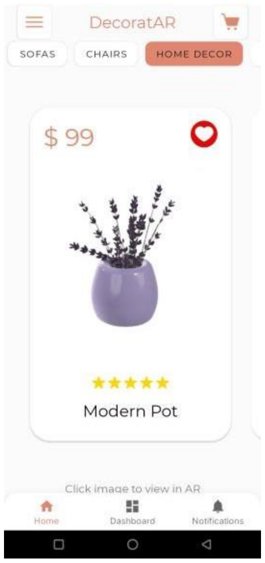


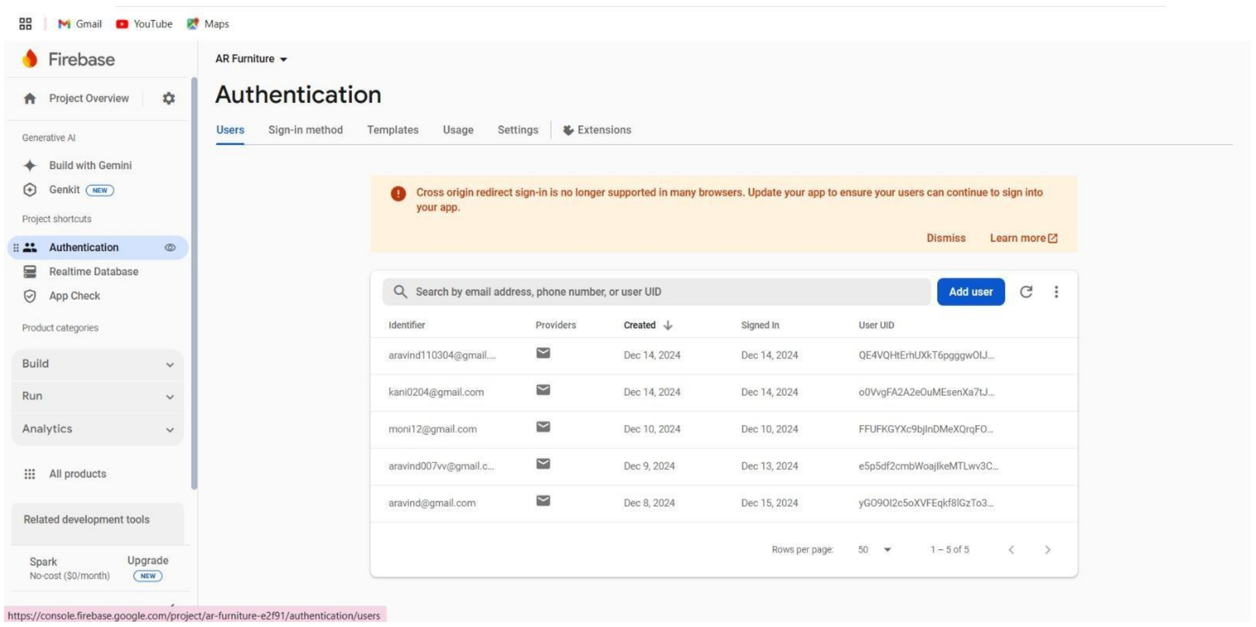
Figure 3.5 Product Page

The Figure 3.5 is user has to register to the page by giving the required details. After register user has to login with the authorized mail id so that the users can be redirected to the new product page. User can now able to view product details like product name, quantity, price and description and buy products

3.5 OUTPUT DESIGN

Output design generally refers to the results and information that are generated by the system for many end-users; it should be understandable with the enhanced format. Computer output is most important direct source of information to the user. Output design deals with form design. Efficient output design should improve the interfacing with user. The term output applies to any information produced by an information system in terms of data displayed. When analyst design system output, they identify the specific output that is needed to meet the requirements of end user.

Previewing the output reports by the user is extremely important because the user isthe ultimate judge of the quality of the output and in turn, the success of the system. When designing output, system analysis accomplishes more things like, to determine what applications, website or documents are blocked or allowed. The output is designed in such way that is attractive, convenient and informative.



Authentication

Cross origin redirect sign-in is no longer supported in many browsers. Update your app to ensure your users can continue to sign into your app.

Identifier	Providers	Created	Signed In	User UID
aravind110304@gmail....	✉	Dec 14, 2024	Dec 14, 2024	QE4VQHtErhUXkT6pgggwOlJ...
kani0204@gmail.com	✉	Dec 14, 2024	Dec 14, 2024	o0VvgFA2A2eOuMEsenXa7tJ...
moni12@gmail.com	✉	Dec 10, 2024	Dec 10, 2024	FFUFKGYXc9bjInDMeXQrqFO...
aravind007vv@gmail.c...	✉	Dec 9, 2024	Dec 13, 2024	e5p5df2cmbWoajIkeMTLwv3C...
aravind@gmail.com	✉	Dec 8, 2024	Dec 15, 2024	yGO9OI2c5oXVFEqkf8lGzTo3...

Rows per page: 50 1 – 5 of 5

https://console.firebase.google.com/project/ar-furniture-e2f91/authentication/users

Figure 3.6 Storing in Firebase

The Figure 3.6 shows the information like details of login, user details. In firebase data is stored as documents, and documents are grouped in collections.

CHAPTER 4

IMPLEMENTATION

4.1 CODE DESCRIPTION

Code description can be used to summarize code or to explain the programmer's intent. Good comments don't repeat the code or explain it. They clarify its intent. Comments are sometimes processed in various ways to generate documentation external to the source code itself by document generator or used for integration with systems and other kinds of external programming tools. I have chosen xml as front end and xml as back end.

4.2 STANDARDIZATION OF THE CODING

Coding standards define a programming style. A coding standard does not usually concern itself with wrong or right in a more abstract sense. It is simply a set of rules and guidelines for the formatting of source code. The other common type of coding standard is the one used in or between development teams. Professional code performs a job in such a way that is easy to maintain and debug. All the coding standards are followed while creating this project. Coding standards become easier, the earlier you start. It is better to do a neat job than cleaning up after all is done. Every coder will have a unique pattern than he adheres to such a style might include the conventions he uses to name variables and functions and how he comments his work. When the said pattern and style is standardized, it pays off the effort well in the long.

4.3 ERROR HANDLING

Exception handling is a process or method used for handling the abnormal statements in the code and executing them. It also enables to handle the flow control of the code/program. For handling the code, various handlers are used that process the exception and execute the code. In many cases there are many corner cases which must be checking during an execution but "if-else" can only handle the defined conditions. In if-else, conditions are manually generated based on the task.

An error is a serious problem than an application doesn’t usually get pass without incident. Errors cause an application to crash, and ideally send an error message offering somesuggestion to resolve the problem and return to a normal operating state, there is no way to dealwith errors “live” or in production the only solution is to detect them via error monitoring and bug tracking and dispatch a developer or two to sort out the code.

USER INTERFACE DESIGN

Input design is the process of converting user originated inputs to a computer understandable format. Input design is one of the most expensive phases of the operation of computerized system and is often the major problem of a system. A large number of problems with a system can usually be tracked backs to fault input design and method. Every moment of input design should be analyzed and designed with utmost care. To provide cost effective method of input. To achieve the highest possible level of accuracy. To ensure that the input is understand by the user. System analysis decide the following input design details like, what datato input, what medium to use, how the data should be arranged or coded, data items and transactions needing validations to detect errors and at last the dialogue to guide user in providing input. Input data of a system may not be necessarily is raw data captured in the systemfrom scratch. These can also be the output of another system or subsystem.

SUMMARY

In this chapter shows explained that the purpose of a code description is to summarise the code or to clarify the programmer’s intent. Good comments don’t repeat or explain the code. A programming style is defined by coding standards. A coding standard isn’t usually concerned with what’s proper or bad in a broader sense. Exception handling is a method or process for dealing with and executing anomalous statements in code. Next Chapter 5 is shown about the Testin

CHAPTER 5

TESTING AND RESULTS

5.1 TESTING

Software testing serves as the final assessment of specifications, designs, and coding and is a crucial component of software quality assurance. The system is tested throughout the testingphase utilizing diverse test data. The preparation of test data is essential to the system testing process. The system under study is tested after the test data preparation. Once the source code iscomplete, relevant data structures should be documented. The finished project must go throughtesting and validation, when errors are explicitly targeted and attempted to be found.

The project developer is always in charge of testing each of the program's separateunits, or modules. Developers frequently also perform integration testing, which is the testing phase that results in the creation of the complete program structure.

This project has undergone the following testing procedures to ensure its correctness

- Unit testing
- Integration Testing
- Validation Testing

5.1.1 Unit Testing

A testing strategy known as unit and integration testing has been used to check that the system behaves as expected. The testing strategy was based on the functionality and the requirements of the system. In unit checking out, we have to check the applications making up the device.

This enables, to stumble on mistakes in coding and common sense which might be contained with the module alone. The checking out became completed at some stage in programming level itself.

Test Case 1

Module	: User Login
Input	: Username and Password
Event	: Button click
Output	: Logged in successfully

Test 1

Module	: User Login
Username	: xyzgmail.com
Password	: 989898
Output	: Logged in successfully
Event	: Button click
Analysis	: Username and Password has been verified

Test 2

Module	: User Login
Username	: xyzgmail.com
Output	: Enter the password
Event	: Button click
Analysis	: Username and Password has been checked and error shown

5.1.2 Integration Testing

Integration testing is done to test itself if the individual modules work together as one single unit. In integration testing, the individual modules that are to be integrated are available for testing. Thus, the manual test data that used to test the interfaces replaced by that which in generated automatically from the various modules. It can be used for testing how the module would actually interact with the proposed system. The modules are integrated and tested to reveal the problem interfaces.

Test Case 1

Module	: User Login
Input	: Username and Password
Output	: Redirect to New product page

Test 1

Module	: User Login
Username	: xyzgmail.com
Password	989898
Output	: Redirected to New Product page
Analysis	: Username and Password has been verified

Test 2

Module	: User Login
Username	: xyzgmail.com
Password	989898
Output	: Enter the correct username
Analysis	: Username and Password has been checked and error shown

5.1.3 Validation Testing

Verification and validation checking out are critical tests, which might be achieved earlier than the product has been surpassed over to the customer. This make sure, that the software program checking out lifestyles cycle begins off evolved early. The intention of eachverification and validation is to make certain that the product is made in step with the necessities of the customer and does certainly fulfil the supposed purpose.

Test case 1

Module	: User Registration
Input	: Password
Output	: Password must be at least 6 numbers or characters

Test 1

Module	: User Registration
Password	989898
Output	: Registered successfully
Analysis	: The given password is 6 numbers. So, it is validated.

Test 2

Module	: User Registration
Password	989898
Output	: Password must be at least 6 characters or number
Analysis	: Only 6 digit numbers or characters should be validated.

SUMMARY

The preceding chapter discusses the many types of testing, including unit testing, integration testing and system testing. During the system evaluation following the completion of the source code, it is documented as associated data structures. The final project must go through testing and validation, which includes both subtle and overt attempts to find problems.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

The project AR in Ecommerce App transforms the furniture shopping experience by leveraging augmented reality to offer users an interactive, personalized, and efficient way to visualize and customize products within their own spaces. By enhancing decision-making, streamlining room planning, and providing accurate visualizations, the app not only improves user satisfaction but also reduces return rates and fosters brand loyalty. The integration of AR technology gives the app a competitive advantage, allowing it to stand out in a crowded market while driving business growth. Ultimately, this innovative approach empowers consumers to make confident, informed furniture choices, revolutionizing the way furniture is purchased online.

Customers are looking for more immersive and customized purchasing experiences, and this technology helps the app stand out from other online retailers. The software sets a new benchmark in the market by providing a distinctive, user-friendly layout and integrating real-time customization possibilities, surpassing contemporary customer expectations. Additionally, business growth will probably be fueled by the app's capacity to raise conversion rates, lower acquisition friction, and enhance customer delight. The app's use of augmented reality technology makes shopping more fun and interesting, which encourages users to return. This innovative strategy not only transforms the online furniture buying experience but also opens the door for further advancements in e-commerce, making it a vital resource for both customers and companies.

6.2 FUTURE ENHANCEMENT

The package was designed in such a way that future modification can be done easily.Automation of the entire system improves the efficiency. It provides a friendly graphical user interface which proves to be better when compared to the existing system.

There is a lot of room for improvement in the AR Furniture App's future, which might lead to an even better user experience. Personalized design recommendations based on a user's preferences, space size, and past selections could be one such enhancement with the incorporation of artificial intelligence (AI). Customers might find furniture possibilities they hadn't thought of before thanks to this, making the purchasing experience more personalized and easy. Additionally, by adding AI-powered assistants or virtual consultations with interior designers, the app might enhance its functionality and provide users with real-time expert guidance while guiding them through the design process. By enabling users to share their virtual room ideas with friends or family for comments and suggestions, the app might further improve its social elements.

APPENDICES

A. SAMPLE CODING

MainActivity.java

```
package com.example.decoratar;

import android.app.ActionBar;
import android.os.Build;
import android.os.Bundle;
import android.view.View;
import android.view.WindowManager;

import com.google.android.material.bottomnavigation.BottomNavigationView;

import androidx.appcompat.app.AppCompatActivity;
import androidx.navigation.NavController;
import androidx.navigation.Navigation;
import androidx.navigation.ui.AppBarConfiguration;
import androidx.navigation.ui.NavigationUI;

import com.example.decoratar.databinding.ActivityMain2Binding;

public class MainActivity2 extends AppCompatActivity {

    private ActivityMain2Binding binding;

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

        binding = ActivityMain2Binding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        BottomNavigationView navView = findViewById(R.id.nav_view);
        // Passing each menu ID as a set of Ids because each
        // menu should be considered as top level destinations.
        AppBarConfiguration appBarConfiguration = new
AppBarConfiguration.Builder(
                R.id.navigation_home, R.id.navigation_dashboard,
R.id.navigation_notifications)
                .build();
//        NavController navController = Navigation.findNavController(this,
R.id.nav_host_fragment_activity_main2);
//        NavigationUI.setupActionBarWithNavController(this, navController,
appBarConfiguration);
//        NavigationUI.setupWithNavController(binding.navView, navController);
    }

    @Override
    protected void onResume() {
        super.onResume();


getWindow().addFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN);

        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.P) {
            getWindow().getAttributes().layoutInDisplayCutoutMode =
WindowManager.LayoutParams.LAYOUT_IN_DISPLAY_CUTOUT_MODE_SHO
RT_EDGES;
        }
    }
}
```

Android Manifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.decoratar">

    <uses-permission android:name="android.permission.CAMERA" />
    <uses-permission android:name="android.permission.INTERNET" />

    <uses-feature
        android:glEsVersion="0x00030000"
        android:required="true" />
    <uses-feature
        android:name="android.hardware.camera.ar"
        android:required="true" />

    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/logo3"
        android:label="@string/app_name"
        android:roundIcon="@drawable/logo3"
        android:supportsRtl="true"
        android:theme="@style/Theme.DecoratAR">
        <activity
```

```
            android:name=".MainActivity2"
            android:exported="false"
            android:label="@string/title_activity_main2" />
        <activity
            android:name=".SplashScreen"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <meta-data
            android:name="com.google.ar.core"
            android:value="required" />

        <activity
            android:name=".MainActivity"
            android:exported="true" />
    </application>

</manifest>
```

Main Activity.Xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/container"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#3EEDEFEF"

    >
    <androidx.cardview.widget.CardView
        android:id="@+id/menu"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toTopOf="@id/nav_host_fragment_activity_main2"
        app:layout_constraintStart_toStartOf="parent"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_marginStart="20dp"
        android:layout_marginTop="20dp"
        android:elevation="10dp"
        android:padding="10dp"
    app:cardCornerRadius="6dp"
    >

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:src="@drawable/menu"
        />
</androidx.cardview.widget.CardView>

<TextView
    android:id="@+id/tvName"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="DecoratAR"
    android:textColor="@color/orange"
    android:fontFamily="@font/montserrat_regular"
    android:textSize="24sp"
    app:layout_constraintStart_toStartOf="@id/menu"
    app:layout_constraintEnd_toEndOf="@id/cart"
    app:layout_constraintTop_toTopOf="@id/menu"
    app:layout_constraintBottom_toBottomOf="@id/menu"
    />

<androidx.cardview.widget.CardView
    android:id="@+id/cart"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintBottom_toTopOf="@id/nav_host_fragment_activity_main2"
    app:layout_constraintEnd_toEndOf="parent"
    android:layout_width="50dp"
    android:layout_height="50dp"
    android:elevation="10dp"
    android:layout_marginTop="20dp"
    android:layout_marginEnd="20dp"
    android:padding="10dp"
    app:cardCornerRadius="6dp"
    >

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:src="@drawable/cart"
        />
</androidx.cardview.widget.CardView>
```

```
<!--    <TextView-->
<!--        android:id="@+id/tvHeading_home"-->
<!--        android:layout_width="wrap_content"-->
<!--        android:layout_height="wrap_content"-->
<!--        app:layout_constraintStart_toStartOf="parent"-->
<!--        app:layout_constraintEnd_toEndOf="parent"-->
<!--        app:layout_constraintTop_toTopOf="@id/menu"-->
<!--        android:fontFamily="@font/montserrat_regular"-->
<!--        android:textStyle="bold"-->
<!--
app:layout_constraintBottom_toTopOf="@id/nav_host_fragment_activity_main2"-->
<!--        android:textSize="24sp"-->
<!--        android:layout_marginTop="10dp"-->
<!--        android:text="@string/best_furniture_for_your_house"-->
<!--        android:textColor="@color/black"-->
<!--        />-->

    <fragment
        android:id="@+id/nav_host_fragment_activity_main2"
        android:name="androidx.navigation.fragment.NavHostFragment"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        app:defaultNavHost="true"
        app:layout_constraintBottom_toTopOf="@id/nav_view"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        android:layout_marginTop="60dp"
        app:navGraph="@navigation/mobile_navigation" />

    <com.google.android.material.bottomnavigation.BottomNavigationView
        android:id="@+id/nav_view"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="0dp"
        android:layout_marginEnd="0dp"
        android:background="@drawable/navbar_white"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:menu="@menu/bottom_nav_menu"
        app:elevation="30dp"
        />



</androidx.constraintlayout.widget.ConstraintLayout>
```

LoginActivity.java

```
package com.example.myapplication;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ProgressBar;
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.textfield.TextInputEditText;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

public class LoginActivity extends AppCompatActivity {
    TextInputEditText editTextEmail, editTextPassword;
    Button buttonLogin;
    FirebaseAuth mAuth;
    ProgressBar progressBar;
    TextView textView;

    @Override
    public void onStart() {
        super.onStart();
        FirebaseUser currentUser = mAuth.getCurrentUser();
        if(currentUser != null){
            Intent intent = new Intent(getApplicationContext(), MainActivity.class);
            startActivity(intent);
            finish();
        }
    }

    @SuppressLint("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        mAuth= FirebaseAuth.getInstance();
        editTextEmail=findViewById(R.id.email);
        editTextPassword=findViewById(R.id.password);
        buttonLogin = findViewById(R.id.btn_login);
        progressBar = findViewById(R.id.progressBar);
        textView =findViewById(R.id.RegisterNow);
        textView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent =new Intent(getApplicationContext(), RegisterActivity.class);
                startActivity(intent);
                finish();
            }
        });

        buttonLogin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                progressBar.setVisibility(View.VISIBLE);
                String email, password;
                email = String.valueOf(editTextEmail.getText());
                password =String.valueOf(editTextPassword.getText());

                if(TextUtils.isEmpty(email)){
                    Toast.makeText(LoginActivity.this, "Enter email",
Toast.LENGTH_SHORT).show();
                    return;
                }

                if(TextUtils.isEmpty(password)){
                    Toast.makeText(LoginActivity.this, "Enter password",
Toast.LENGTH_SHORT).show();
                    return;
                }

                mAuth.signInWithEmailAndPassword(email, password)
                        .addOnCompleteListener(new OnCompleteListener<AuthResult>() {
                            @Override
                            public void onComplete(@NonNull Task<AuthResult> task) {
                                progressBar.setVisibility(View.GONE);
                                if (task.isSuccessful()) {
                                    Toast.makeText(getApplicationContext(), "Login Successful",
Toast.LENGTH_SHORT).show();
                                    Intent intent = new Intent(getApplicationContext(),
MainActivity.class);
                                    startActivity(intent);
                                    finish();
                        } else {

                            Toast.makeText(LoginActivity.this, "Authentication failed.",
                                    Toast.LENGTH_SHORT).show();

                        }
                    }
                });
            }
        });
    }
}
```

RegisterActivity.java

```
package com.example.myapplication;

import static androidx.constraintlayout.helper.widget.MotionEffect.TAG;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ProgressBar;
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.textfield.TextInputEditText;
import com.google.firebase.Firebase;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

public class RegisterActivity extends AppCompatActivity {

    TextInputEditText editTextEmail, editTextPassword;
    Button buttonReg;
    FirebaseAuth mAuth;
    ProgressBar progressBar;
    TextView textView;

    @Override
    public void onStart() {
        super.onStart();
        FirebaseUser currentUser = mAuth.getCurrentUser();
        if(currentUser != null){
            Intent intent = new Intent(getApplicationContext(), MainActivity.class);
            startActivity(intent);
            finish();
        }
    }

    @SuppressLint("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);
        mAuth= FirebaseAuth.getInstance();
        editTextEmail=findViewById(R.id.email);
        editTextPassword=findViewById(R.id.password);
        buttonReg = findViewById(R.id.btn_register);
        progressBar = findViewById(R.id.progressBar);
        textView =findViewById(R.id.loginNow);
        textView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent =new Intent(getApplicationContext(), LoginActivity.class);
                startActivity(intent);
                finish();
            }
        });

        buttonReg.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                progressBar.setVisibility(View.VISIBLE);
                String email, password;
                email = String.valueOf(editTextEmail.getText());
                password =String.valueOf(editTextPassword.getText());

                if(TextUtils.isEmpty(email)){
                    Toast.makeText(RegisterActivity.this, "Enter email",
Toast.LENGTH_SHORT).show();
                    return;
                }
                if(TextUtils.isEmpty(password)){
                    Toast.makeText(RegisterActivity.this, "Enter password",
Toast.LENGTH_SHORT).show();
                    return;
                }

                mAuth.createUserWithEmailAndPassword(email, password)
                        .addOnCompleteListener(new OnCompleteListener<AuthResult>() {
                            @Override
                            public void onComplete(@NonNull Task<AuthResult> task) {
                                progressBar.setVisibility(View.GONE);
                                if (task.isSuccessful()) {
                                    Toast.makeText(RegisterActivity.this, "Account created.",
                                            Toast.LENGTH_SHORT).show();
                                    Intent intent =new Intent(getApplicationContext(),
LoginActivity.class);
                                    startActivity(intent);
                                    finish();

                                } else {
                                    // If sign in fails, display a message to the user.
                                    Toast.makeText(RegisterActivity.this, "Authentication failed.",
                                            Toast.LENGTH_SHORT).show();

                                }
                            }
                        });

            }
        });

    }
}
```

LoginActivity.Xml

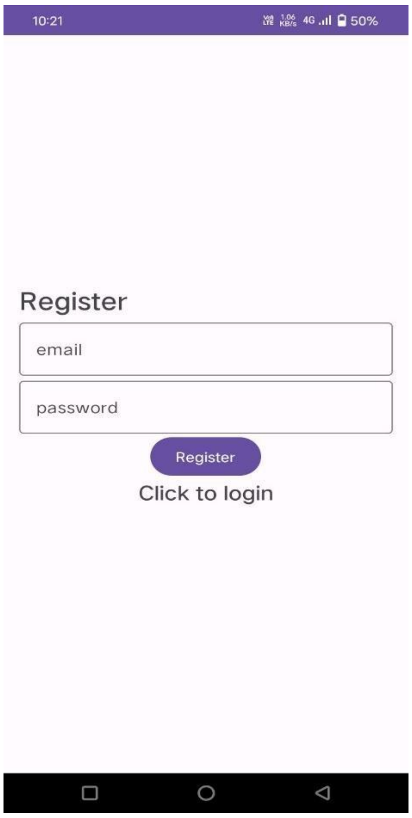
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    android:layout_height="match_parent"
    android:padding="15dp"
    tools:context=".LoginActivity">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginBottom="2sp"
        android:text="@string/login"
        android:textSize="25sp"
        android:textStyle="bold" />

    <com.google.android.material.textfield.TextInputLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
        <com.google.android.material.textfield.TextInputEditText
            android:id="@+id/email"
            android:hint="@string/email"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"/>
    </com.google.android.material.textfield.TextInputLayout>

    <com.google.android.material.textfield.TextInputLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
        <com.google.android.material.textfield.TextInputEditText
            android:id="@+id/password"
            android:hint="@string/password"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"/>
    </com.google.android.material.textfield.TextInputLayout>

    <ProgressBar
        android:id="@+id/progressBar"
        android:visibility="gone"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>

    <Button
        android:id="@+id/btn_login"
        android:text="@string/login1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>

    <TextView
        android:layout_marginBottom="20dp"
        android:textSize="20sp"
        android:textStyle="bold"
        android:gravity="center"
        android:id="@+id/RegisterNow"
        android:text="@string/click_to_register"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

Register activity.xml

```
<?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:orientation="vertical"
    android:gravity="center"
    android:layout_height="match_parent"
    android:padding="15dp"
    tools:context=".LoginActivity">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginBottom="2sp"
        android:text="@string/register"
        android:textSize="25sp"
        android:textStyle="bold" />

    <com.google.android.material.textfield.TextInputLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
        <com.google.android.material.textfield.TextInputEditText
            android:id="@+id/email"
            android:hint="@string/email"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"/>
    </com.google.android.material.textfield.TextInputLayout>

    <com.google.android.material.textfield.TextInputLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
        <com.google.android.material.textfield.TextInputEditText
            android:id="@+id/password"
            android:hint="@string/password"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"/>
    </com.google.android.material.textfield.TextInputLayout>

    <ProgressBar
        android:id="@+id/progressBar"
        android:visibility="gone"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>

    <Button
        android:id="@+id/btn_register"
        android:text="@string/register"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>

    <TextView
        android:layout_marginBottom="20dp"
        android:textSize="20sp"
        android:textStyle="bold"
        android:gravity="center"
        android:id="@+id/loginNow"
        android:text="@string/click_to_login"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

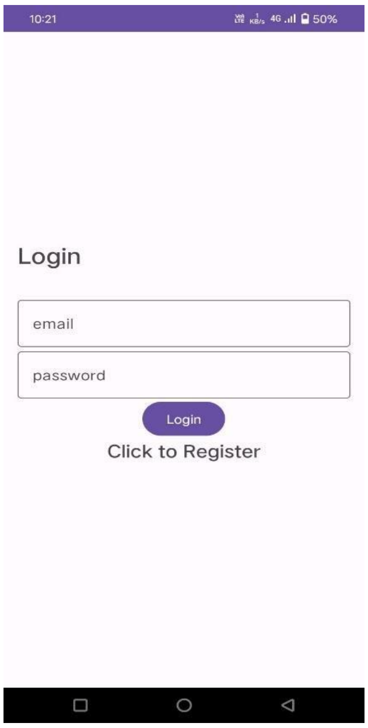
SCREENSHOTS

B.1REGISTER PAGE



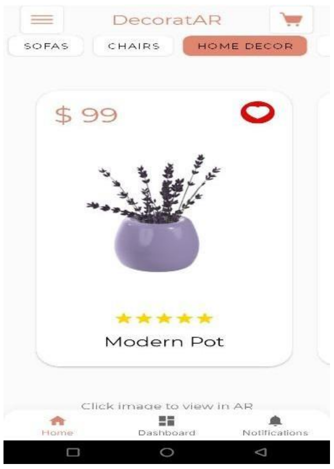
The figure B1 is the first screen users see upon opening the app. Providing a friendly introduction and guiding them to register

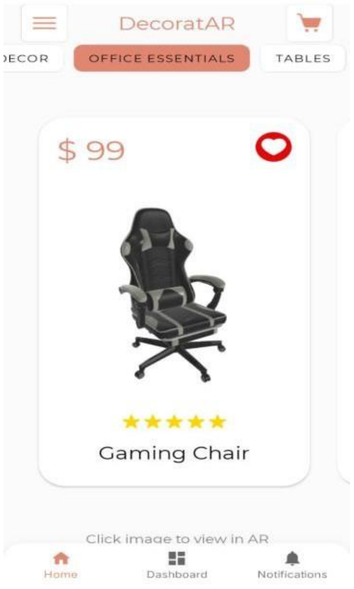
B.2 LOGIN PAGE

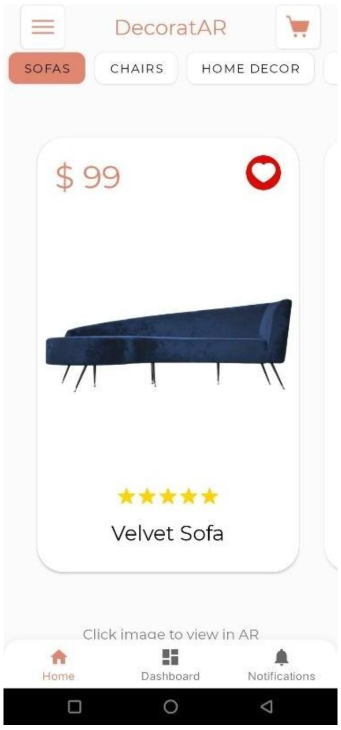


The figure B2 of the ar in ecommerce app allows users to securely sigin in with their email and password It includes validation for correction and options for password recovery or new account creation

B.3 PRODUCT VIEW







The figure B3 provides users with an overview of the product, ratings and price

REFERENCES

BOOK REFERENCE

[1] "Core Java Volume I—Fundamentals" by Horstmann, Cay S., and Cornell, Gary (2022, Volume 1,12th Edition)

[2] "Effective Java" by Bloch, Joshua (2018, 3rd Edition)

[3] "XSLT: Mastering XML Transformations" by Tidwell, Doug (2008, 2nd Edition)

[4] "Learning XML" by Nerino, Erik T. Ray (2003, 2nd Edition)

[5] "Mastering Firebase for Android Development" by Patel, Praveen Kumar (2018, 1st Edition)

[6] "Firebase Essentials: Real-Time Database, Cloud Functions, and Authentication" (2022, 1st Edition)

WEBSITE REFERENCE

[1] https://www.w3schools.com/java/

[2] https://www.javatpoint.com/java-tutorial

[3] https://www.geeksforgeeks.org/java/

[4] https://www.w3schools.com/xml/

[5] https://www.geeksforgeeks.org/xml-tutorial/

[6] https://www.geeksforgeeks.org/firebase-tutorial/

[7] https://www.coursera.org/learn/learn-firebase