Fully Dressed Use Cases:

(Use Cases HIghlighted in Yellow)

Function: User Profile (Sarah Abbadi)

Use Case UC1: RegisterUserInfo (Personalized username and password)
Primary Actor: User
Stakeholders and interests:
User: wants secure and personalized login information
Preconditions: The user has the app downloaded and a unique username
Success guarantee: Username and password saved and assigned to the user. The number of users in the database is updated.
Main success scenario:

1. User downloads app from app store
2. The user is prompted to enter their personal information and creates a new user profile.
3. System records user profile and user's personal information in user database.
4. System signs user in and presents the home screen of the app.

Extensions:
2a. Username and password already exists with a user's account

1. System rejects new sign-up attempt
2. System prompts user to enter their username and password previously made
   2a. Invalid password associated with username: System rejects sign-in attempt. System prompts user to enter correct username and password. System gives user an option to send an email to user's email associated with previously made account with a link to change their password.
   Repeat 2a until correct username and password combination is entered by user.
3. User correctly inputs username and password
4. User signs in to system.
5. System presents user with home screen of the app.

2b. Username and password is unique and does not already exist

1. User inputs personal information including: first name, last name, date of birth, email (username), and password.
2. System prompts user to sign in with previously entered username and password.
3. User correctly inputs username and password
4. User signs in to system.
5. System presents user with home screen of the app

Special requirements:

- App must be downloaded in IOS operating system

Technology and data variation list:
1a. App is downloaded from the Apple app store using an Apple device
2a. User's personal information is entered on device keyboard by user
2b. User's username (email) must end with @gmail.com, @yahoo.com, @mail.com, @aol.com, or @outlook.com

2c. User’s password must be at least 12 characters long, have at least one uppercase letter, one lowercase letter, one number, and one symbol.
2d. User’s date of birth must be entered in this format: MM/DD/YYYY
4a. Home screen of app presented on app on user’s device
Frequency of Occurrence
Once for every user sign-in

Use Case UC2: RegisterPaymentInfo(Credit card information)
Primary Actor: User
Stakeholders and interests:
User: wants accurate and fast payment
Preconditions: The user has the app downloaded and a user profile created.
Success guarantee: Payment information is saved and assigned to the user. Payment information is confirmed by card issuer and is ready for use in the app.
Main success scenario:

1. User opens app and signs in
2. User navigates to user profile tab, and then to payment information
3. User presses edit payment information
4. System prompts user to enter credit card number, expiration date, and security code.
5. System validates credit card information with card issuer
6. System saves payment information to user’s profile for future use in the app.

Extensions:
4a. User enters incorrect credit card information

1. System rejects attempt to add credit card information
2. System prompts user to enter correct credit card information
   Repeat 4a until correct credit card information is entered by user.

4b. Credit card information already exists in system

1. User deletes previous credit card information.
2. User enter new credit card information (main success scenario)
   2a. If credit card information already exists, user cannot enter new credit card information (User must got to Step 1 under 4b)

5a. Credit card issuer rejects credit card information

1. Go to 4a.

Special requirements:

- App must be download in IOS operating system
- Credit card issuer validation responds within 30 seconds 90% of the time

Technology and data variation list:
1a. App is downloaded from the Apple app store using an Apple device
2a. User’s navigates to the user profile tab and to payment information section within the app interface
3a. User presses edit payment information using button in payment information section in app interface
4a. Credit card information is entered by user through device keyboard
4b. User’s credit card number must be 16 digits arranged in sets of four

4c. User’s credit card expiration date must be entered in this format: MM/YY
4d. User’s credit card security code must be at least 3 digits
6a. User’s credit card information is saved in the app
Frequency of Occurrence
Once for every time a user inputs new credit card information

Use Case UC3: DisplayPurchases (Purchase History)
Primary Actor: User
Stakeholders and interests:
User: wants to access their purchase history and view past items purchased through the app
Preconditions: The user has the app downloaded, a user profile created, updated credit card information in the app, and previous purchase history through the app.
Success guarantee: Purchased items information is saved and assigned to the user. Purchased items are displayed in the purchase history section of the user profile tab in the app.
Main success scenario:

1. User opens app and signs in
2. User navigates to user profile tab, and then to purchase history section
3. User views a list of the item(s) they have previously purchased with the date, name of item, vendor name, and picture of item listed with each of the items purchased
4. User can access each past purchase to view the purchase information by clicking on the item in the purchase history list
5. User accesses individual items by clicking on each item
6. Item information is displayed: item name, multiple item images, item purchase date, item price paid by user, item description, and item reviews
7. User can navigate back to purchase history section or to app home screen

Repeat Steps 4-7 until User is done with viewing purchasing history
Extensions:
3a. No items previously purchased

1. System displays the message “No items purchased” on purchase history screen in the app
2. User can navigate back to the app home page

3b. New item purchased

1. User purchases item in the app
2. System validates purchase
3. System updates purchase history list with new items purchased
   3a. User must exit app for system to update purchase history
4. User can view new items purchased in purchase history section in the app

Special requirements:

- App must be downloaded in IOS operating system
- The app must include a help button
- The app should contain adjustable text/color features to accommodate everyone

Technology and data variation list:
1a. App is downloaded from the Apple app store using an Apple device

2a. User’s navigates to the user profile tab and to the purchase history section within the app interface
4a. User presses each purchased item in the purchase history list by pressing on the image of the item in the purchase history section in app interface
6a. Item information is displayed in text format
Frequency of Occurrence
Every time User accesses purchase history section in the app.
Every time User purchases an item through the app.

Use Case UC4: RecordPreferences (Color scheme/theme preferences)
Primary Actor: User
Stakeholders and interests:
User: wants to cutomize the color scheme/themes of their rooms in the app
Preconditions: The user has the app downloaded, a user profile created, and a room or furniture category created
Success guarantee: Color scheme/theme preferences are saved to their respective rooms/furniture sections and saved in user’s profile. Chosen Color scheme/theme preferences are displayed under the preferences in the saved rooms/furniture sections in the Rooms tab or Furniture tab.
Main success scenario:

1. User opens app and signs in
2. User navigates to Rooms tab
3. User navigates to specific room
4. System displays options of: Color Scheme or Themes in respective room
5. User chooses from dropdown list in the respective option that they desire and saves choices
6. System saves User’s choices of preferences in the respective room/furniture category
7. User can navigate back to Rooms tab or to app home screen

Extensions:
2a. User navigates to Rooms tab

1. User chooses a Room that they have previously created
2. System displays room name and cover image of room for each room previously created in a list
3. User can choose desired room
4. System displays room details and preferences for each room
   4a. Room details include: room cover image (overview image of room), room name, house name (if applicable), furniture items already placed in room, and users that the room is shared with
   4b. Room preferences include the options to choose Color Sheme or Theme
    1. Color sheme options include: monochromatic, analogous, complementary, split complementary, triadic, square, and rectangle (or tetradic).

    2. Theme options include: Minimalism, Art Deco, Modern architecture, Shabby chic, Mid-century modern, Scandinavian design, Feng shui, Industrial style, French country, Boho, and Asian Zen
5. User chooses Color scheme and/or Theme for chosen room from two drop down menus
6. System saves User's choices
7. User navigates out of Rooms tab to app home screen

2b. User does not have any rooms made

1. System displays "Create room" after user navigates to Rooms tab
2. User can click "Create room button" to make a room
3. Follow extensions in "Create Room" scenario to make a room

Special requirements:

- App must be downloaded in IOS operating system
- The app must include a help button
- The app should contain adjustable text/color features to accommodate everyone

Technology and data variation list:

1a. App is downloaded from the Apple app store using an Apple device
2a. User navigates to Rooms tab through app interface
3a. User navigates to specific room using app interface
4a. System displays options of: Color Scheme or Themes in respective room by using "Color Scheme" or "Theme" buttons in a specific room in the app
5a. User chooses from dropdown list in the respective option that they desire and saves choices by using app interface
5b. All color scheme and them options must be in text format
6a. System saves User's choices of preferences in the respective room/furniture category by continuously saving User's choices
7a. User can navigate back to Rooms tab or to app home screen using app interface

Frequency of Occurrence

Every time User desires to change their room/furniture category color scheme/theme preferences.
Every time User desires to add room/furniture category color scheme/theme preferences.

Function: Purchasing Feature (Anushka)

Use Case UC1: ReviewWishlist

Primary Actor: user

Stakeholders and interests:

User: Wants to create a list of favorite items, track them conveniently, and easily transition these items into their shopping cart when ready to make a purchase.
App: Wants to provide a wishlist feature that enhances the user's shopping experience and encourages purchases.

Preconditions:

- User is authenticated
- User is browsing the product catalog and has found items they wish to add to their wishlist.
- The user has items in their wishlist.

Success guarantee:
User can successfully create and manage a wishlist, move items from the wishlist to the shopping cart, and complete purchases.

Main success scenario:

1. The user selects an item that they find interesting.
2. The user decides they are not ready to add the item to their shopping cart so they find the “Add to Wishlist” button
3. The user clicks the "Add to Wishlist" button for the selected item.
4. The item is added to the user's wishlist.
5. User views and manages their wishlist by clicking on the "Wishlist" section in their account.
6. The wishlist displays a list of items the user has favorited
7. The user can then remove the item from their wishlist, move the items to their shopping cart or view more details about the item
8. The user decides to move one from the wishlist to their shopping cart.
9. The user clicks the "Add to Cart" button next to the selected items.
10. The items are seamlessly transferred to the shopping cart.

Extensions:

3a. The User tries to add an item to the wishlist without logging in:

a. An error message appears saying that the user is not logged in
b. The user is directed to the login page

5a. The user wants to remove an item from the wishlist:

a. user clicks on a "Remove" button next to an item in the wishlist.
b. The item is removed from the wishlist

9a. user cannot move items from wishlist to shopping cart:

a. User clicks "Move to Cart" but a connectivity error pops up.
b. A message pops up telling the user to try again

Special requirements:

- Must use IOS
- Text should be accessible for everyone

Technology and data variation list:

- Should be integrated with the shopping cart functionality
- Should provide updates to the wishlist and cart

Frequency of Occurrence:

- Frequently:
  - Users use the wishlist option when they are not fully ready to purchase the item

Use Case UC2: ReviewAndPurchase:

Primary Actor: User

Stakeholders and interests:

- User: Wants to easily select and manage furniture items they intend to purchase for placement in their space using the AR furniture placement app.
- App: Wants to offer a user-friendly and efficient shopping cart feature that enhances the overall app experience.

Preconditions:

- User is authenticated
- User has already browsed furniture selection at least once on the app and has moved any desired wishlist items to the shopping cart
- User has at least one item in the shopping cart

Success guarantee:

The user successfully adds furniture to the shopping cart, makes it to the checkout page, and correctly inputs all information including name, contact information, payment information, billing address and shipping address. Payment is accepted and receipt and confirmation is sent to user.

Main success scenario:

1. User browses through furniture selection and selects furniture to be put into their shopping cart
2. Text is displayed confirming that item was added to the shopping cart
3. The user continues shopping until they are satisfied with their shopping cart
4. The user clicks on the shopping cart icon and selects the review cart option
5. The app navigates to the shopping cart page where the user can review the items in their shopping cart
6. The page displays the name, the price, and the quantity of each item
7. The user can choose to delete the item from their cart or change the quantity of the item in the cart
8. Each time an adjustment is made in the cart, the total price is updates at the bottom of the screen
9. The user decides to make no adjustments and clicks checkout
10. They will then be prompted with the payment information section
11. They will input their name, payment information, and billing address and then continue to the shipping address section
12. They will input their shipping information and click complete order and receive an email confirmation and receipt

Extensions

1a. User tries to add an item into their cart that is out of stock

   a. A pop up is shown that the item is out of stock
   b. The user continues shopping

7a. User decides to remove an item from the shopping cart:

   a. A pop up is showing asking the user if they are sure they want to delete the item
   b. If confirmed, the item is removed from the cart, and the total price is updated

7b. User decides to change the quantity of an item in the cart

   a. The user clicks the plus button next to the quantity of the item to increase the amount
   b. The total prices is updated based on the new quantity

9a. User decides to cancel the checkout process:

   a. After reviewing the items in the cart, the user decides not to proceed with the purchase.
   b. The user clicks on a "Cancel" button on the shopping cart page instead of the “Checkout” button
   c. The user is redirected back to the catalog to continue browsing.

11a. User has a problem with their payment informatio:

   a. On the payment information page, the user encounters an error, such as an invalid/expired credit card number or invalid billing address
   b. The app displays an error message that something is incorrect and tells the user to re input their information

12a. User encounters an issue during the shipping address input:

   a. After entering payment information, the user puts in their shipping address.
   b. The user encounters an error, such as an incomplete or invalid address.
   c. The app displays an error message that the information provided is incorrect and tells the user to re input their information

Special requirements:

- Must use IOS

- Text should be accessible for everyone

Technology and data variation list:

- Integration with payment processor

Frequency of Occurence:

- Frequently:
  - Adding items and reviewing a shopping cart is done often by users

Use Case UC3: TrackOrder

Primary Actor: User

Stakeholders and interests:

- User: Wants to monitor the status of their orders
- App: Wants user to have a seamless experience while tracking their order
- Shipping Courier: Wants to provide efficient and timely shipping updates

Preconditions:

- User is authenticated
- User has placed an order on the app
- The order has been shipped
- The app has a successfully integrated shipping tracking system

Success guarantee:

The user can successfully track their shipped item on the app. The user can see where the order is on each day and what date the item is supposed to be delivered

Main success scenario:

1. User goes to the "Order History" section.
2. The app displays a list of the user's recent orders, including those with shipped items.
3. User selects the order they want to track.
4. The app shows detailed information about the selected order's shipment status and expected delivery date, including where the package is currently
5. The user has the option to copy the tracking number, contact the courier, or user service
6. user is satisfied with the provided information.

Extensions

3a. User doesn't have any shipped orders

a. The app displays that there are no shipped orders for the user to view
b. The user leaves the order history page

5a. User decides to contact the courier

a. There is an issue with the tracking system so the user calls the courier

5b. User decides to contact user service

a. There is an issue with the shipped order so the user calls user service

Special requirements:

- Must use IOS
- Text should be accessible for everyone

Technology and data variation list:

- Integration with shipping courier APIs to obtain tracking information.

Frequency of Occurence:

- Frequently:
  - users want to stay informed about the status and delivery dates of their orders

Use Case UC4: InitiateReturn

Primary Actor: User

Stakeholders and interests:

- User: Wants a straightforward and transparent process for returning purchased items,
- App: Wants to provide a reliable return process to maintain user satisfaction and comply with return policies and regulations.

Preconditions:

- User has made a purchase.
- The return policy is available and accessible to the user.
- The user's purchase is eligible for return based on the return policy.

Success guarantee:

user can successfully initiate returns, track the status of their returns, and receive refunds according to the app's return policy.

Main success scenario:

1. User goes to the "Return" section.
2. The system displays a list of the user's recent orders
3. User selects the item they want to return.
4. User specifies the reason for the return
5. The system generates a return request number and provides instructions for packaging and shipping the return.
6. user follows the provided instructions, packages the items securely, and sends it to the designated return address.
7. user can track the status of their return through the app.
8. user receives an email notification confirming the refund, along with the refunded amount and the expected time frame for the funds to appear in their account.

Extensions:

3a. user attempts to initiate a return for an item not eligible for return:

a. The system displays a message informing the user that the selected item(s) cannot be returned based on the return policy.

b. user is advised to review the return policy or contact user support for further assistance

Special requirements:

- Must use IOS
- Text should be accessible for everyone

Technology and data variation list:

- Should provide tracking of return status

Frequency of Occurrence:

- Somewhat frequent:
    - The return process occurs when user has an issue with their product

Function: AR Furniture Placement (Ameya)

Use Case UC1: Capture Space

Primary Actor: App/ System

Stakeholders and Interests

1. The user wants to save their 3D space in their profile

Preconditions

1. The user's environment is well-lit
2. The user has an iOS device that has a functioning LiDAR Scanner featuring current AR support
3. User opens the furniture app
4. User browses the furniture catalog and selects a furniture item
5. User clicks the AR button in the item page
6. The system fetches the AR image
7. User selects the Live AR View option
8. Systems prompts the user to move their iOS device around the room in order for the system to scan the environment
9. User selects a spot to place the AR image by dragging it and the system maps the image to the desired spot

Success Guarantee

1. The app successfully creates a 3D scan of the user's environment including the AR furniture piece placed by the user.

Main Success Scenario

1. System detects that the AR image has been placed and displays button to capture the space
2. User clicks the Capture Space Button
3. System prompts the User to move their iOS device around the room to create a scan of the environment containing the AR furniture item and prompts the user that the scan is complete
4. System display options for the user to save the captured space to the profile, share the space, redo the scan
5. User selects save the capture space to the profile
6. Systems saves space to the profile and exits the capture space view

Extensions

1a. System fails to detect the AR image

1. System does not display the button to capture the space
2. System prompts the user that no AR image was found
3. System displays button to exit the view or rescan
    1. User clicks to rescan the environment
        1. System repeats 1 in the main success scenario and rescans
    2. User clicks to exit the view
        1. System exits the current view

3a. System cannot complete scan

1. System alerts user that the scan is incomplete
2. Systems displays button to exit the view or rescan
    1. User clicks to rescan the environment
        1. Systems repeats 3 in the main success scenario and rescans
    2. User click to exit the view
        1. System exits the current view

4a. User clicks share space

1. System uses default apple share menu to share the AR space with friends
2. User shares space with friend and clicks exit share menu and exits current view
3. System exists menu and current view

4b. User clicks redo the scan

1. Systems repeats 3 in success scenario

5a. User does not have profile

1. Systems prompts user to create a profile
2. User creates profile
3. Systems saves space to profile

Special Requirements

1. Scan must be completed in < 30 seconds

Technology and Data Variation List

1. App is supported on any iPhone or iPad that came out after 2017
2. System can use RoomPlan API by Apple or any API allowing for room detection

Frequency of Occurence

Sometimes, The User will only use this feature if they choose to share and save spaces with others

Use Case UC2: Detect Collision

Primary Actor: App/System

Stakeholders and Interests

1. The user wants to make sure that the AR furniture piece is properly aligned and doesn't collide with other physical objects in the room
2. The developers want to have more accurate mappings of the AR Image to places in the User's environment

Preconditions

1. The user is in a well-lit environment
2. The user has an iOS device that has a functioning LiDAR Scanner featuring current AR support
3. User opens the furniture app
4. User browses the furniture catalog and selects a furniture item
5. User clicks the AR button in the item page
6. The system fetches the AR image
7. User selects the Live AR View option
8. Systems prompts the user to move their iOS device around the room in order for the system to scan the environment

Success Guarantee

1. The system detects collisions with physical objects in the user's environment when the user attempts to place the AR furniture piece in the same place

Main Success Scenario

1. The user drags the AR Image around the room in their desired location
2. The system detects whether or not the AR Image collides with any of the physical objects detected during the 3D room scan
3. User selects a location that has no collisions
4. Systems maps the AR Image to the desire location

Extensions

2a. The Systems could not detect certain objects

1. The system allows user to place AR Image that collides with an object that was not detected

3a. User selects a location that has collisions

1. Systems prompts user to move AR Image as a collision is detected

Special Requirements

1. System should detect most common furniture objects
2. Systems should detect objects up to 5 meters away from the device

Technology and Data Variation List

1. App is supported on any iPhone or iPad that came out after 2017
2. System can use RoomPlan API by Apple or any API allowing for room detections
3. Should use the LiDAR sensor on iOS devices

Frequency of Occurence

1. Often, Every time the user wants to display the furniture item in their environment, the collision detection feature is in use.

Use Case UC3: ShowDimensions

Primary Actor: App/System

Stakeholders and Interests

1. User wants to view the dimensions of the furniture item

Preconditions

1. The user is in a well-lit environment
2. The user has an iOS device that has a functioning LiDAR Scanner featuring current AR support
3. User opens the furniture app
4. User browses the furniture catalog and selects a furniture item
5. User clicks the AR button in the item page
6. The system fetches the AR image
7. User selects the Live AR View option
8. Systems prompts the user to move their iOS device around the room in order for the system to scan the environment

Success Guarantee

1. The systems displays the dimensions of the AR Image in the user's environment

Main Success Scenario

1. The user drags the AR Image into their desired location
2. The System maps the AR Image to their desired location

3. System displays option to view dimensions of the furniture piece
4. User Clicks button to view dimensions
5. The system fetches dimension sizes of the piece
6. The system outlines the dimensions of the furniture piece around the object in the user’s environment

Extensions

5a. If the Furniture API does not contain dimension information

1. System will use apple’s proprietary measuring tool and displays the dimensions

Special Requirements

1. The dimensions should be displayed outlining the furniture piece without covering other objects in the user’s environment

Technology and Data Variation List

1. The system should use the dimensions specified by the furniture manufacturer or the dimensions provided by the apple measuring tool

Frequency of Occurence

1. Often, The user will wants to measure and display the dimensions of the furniture piece often in order to compare it to other furniture items in their environment

Use Case 4 UC4: FetchARImage

Primary Actor: App/System

Stakeholders and Interests

1. The furniture companies want to continuously add objects to the furniture API
2. Developers want to store all of the furniture item AR objects in the API

Preconditions

1. The user is in a well-lit environment
2. The user has an iOS device that has a functioning LiDAR Scanner featuring current AR support
3. User opens the furniture app
4. User browses the furniture catalog and selects a furniture item

Success Guarantee

1. When the AR Button is clicked, the system successfully fetches the desired AR Image from the furniture API

Main Success Scenario

1. The User selects the AR button on the furniture item they have selected in the catalog
2. The Systems fetches the AR Image the user selected from the furniture API
3. The user can place the AR Image in their environment

Extensions

2a. The item the user selected does not have a corresponding AR object in the furniture API currently

1. The system should display that the AR Image is currently not available

Special Requirements

1. The system should fetch images from the API and display them to the user < 10 seconds

Technology and Data Variations List

1. The system uses furniture items from many different stores that are constantly updated

Frequency Occurence

1. Very Often, Every time the user wants to view the item in their environment and presses the AR button, the system will fetch AR objects from the furniture API

Function: AR Furniture View and Sharing (Shreyas Sakhalkar)

Use Case UC1: Preview Furniture in AR Mode
Primary Actor: User
Stakeholders and interests:

- User: Wants to visualize how a furniture item would fit and look in their real environment before making a purchase.
- Furniture Store: Wants the user to have a realistic experience of how the furniture will fit in their home, leading to confident purchasing decisions.
- App Developer: Wants to provide a user-friendly experience and reduce potential issues related to furniture returns or customer dissatisfaction.

Preconditions:

- User has the AR furniture app installed on a compatible device.
- The room in which the user wants to preview the furniture is well-lit and the device's camera can clearly detect the floor and surrounding space.

Success guarantee: User successfully previews the furniture in the desired location within their space using the AR mode.

Main success scenario:

- User opens the AR furniture app.
- User selects a furniture item they wish to preview.
- User grants the necessary permissions for the app to access the device's camera.
- The AR mode is activated, and the selected furniture item is displayed in the room through the device's camera view.
- User can drag the furniture item to move it around the room.
- User positions the furniture item in the desired location within their space.
- User is satisfied with the preview and can make a more informed decision about purchasing the furniture.

Extensions: User's device is not compatible with the AR functionality.

- The app notifies the user about the compatibility issue and suggests alternative devices or methods to preview the furniture.

User tries to resize the furniture item.

- The app displays a notification that resizing is not possible and only moving the furniture around the room is allowed.

Special requirements:

- The AR mode must be able to detect the floor and walls accurately to place the furniture item realistically.
- The app should have a user-friendly interface to allow easy dragging and positioning of the furniture item.

Technology and data variation list:

- The app should support multiple device types and operating systems to cater to a wide range of users.
- The app should be able to handle different furniture data (e.g., 3D models, textures) for accurate representation in AR mode.

Frequency of Occurrence:

- Often, whenever a user is considering purchasing a new furniture item and wants to preview it in their space using the AR functionality.

Use Case UC2: Toggle Between Object View and Live AR View

Primary Actor: User

Stakeholders and interests:

- User: Wants the flexibility to switch between a standard object view and a live augmented reality (AR) view of a furniture item to better understand its design, details, and how it fits within their space.
- Furniture Store: Wants to provide the user with a comprehensive experience of the furniture, allowing them to see both the detailed design and the item in context.
- App Developer: Seeks to offer a seamless user experience when toggling between views, ensuring clarity and ease of use.

Preconditions:

- User has the AR furniture app installed on a compatible device.
- User has selected a specific furniture item within the app.

Success guarantee: User can effortlessly switch between the object view and the live AR view of the selected furniture item.
Main success scenario:

- User opens the AR furniture app.
- User browses and selects a furniture item they are interested in.
- By default, the furniture item is displayed in the object view, showcasing its design and details.
- User finds an option/button labeled "Switch to AR View" or a similar call to action.
- Upon selecting the option, the app requests access to the device's camera (if not already granted).
- The live AR view is activated, showing the selected furniture item in the room through the device's camera view.
- User sees an option/button labeled "Switch to Object View" or a similar call to action within the AR mode.
- User selects the option, and the app reverts back to the standard object view of the furniture item.
- User can toggle between the two views as desired.

Extensions: User denies camera access for the AR view.

- The app displays a notification informing the user that the AR view requires camera access and provides steps to grant permissions.

Special requirements:

- The transition between the two views should be smooth and intuitive.
- Both views should be rendered in high quality, ensuring that the furniture item's details are clearly visible.

Technology and data variation list:

- The app should be optimized for various device screen sizes and resolutions to ensure that the furniture item is displayed correctly in both views.
- The app should incorporate advanced AR technologies for realistic placement and rendering of the furniture item in the live AR view.

Frequency of Occurrence:

- Frequently, as users will often want to inspect the design details of a furniture item and then visualize it within their own space before making a purchasing decision.

Use Case UC3: AR Image Sharing
Primary Actor: User
Stakeholders and interests:

- User: Wants to share a view of the furniture item in AR with friends to get feedback or suggestions.
- Friends: Interested in viewing the furniture item in their own space using AR to offer advice or consider for their own purchase.

- Furniture Store: Seeks to expand its reach and influence potential customers through shared AR experiences.
- App Developer: Aims to ensure the sharing feature is seamless and provides an authentic AR experience to the receiver.

Preconditions:

- User has the AR furniture app installed on a compatible device.
- User has selected a furniture item and previewed it in AR mode.
- Recipient (friend) has a device capable of displaying the shared AR experience.

Success guarantee: User successfully shares the AR view of the furniture item, and the recipient can place and view the item in their own space.

Main success scenario:

- User selects a furniture item and views it in AR mode in their space.
- Within the AR mode, the user sees a "Share" button or icon.
- User clicks on the "Share" button.
- The app generates a unique link or package that encapsulates the AR experience of the selected furniture item.
- User shares the generated link/package through their desired method (e.g., messaging apps, email, social media).
- Recipient receives the shared link/package and clicks on it.
- If the recipient has the AR furniture app installed, the app opens directly to the AR view of the shared furniture item. If not, they are prompted to download the app.
- Once in the app, the recipient can place and view the furniture item in their own space using AR.

Extensions: Recipient does not have the AR furniture app installed.

- Recipient is directed to the app store to download and install the AR furniture app.
- After installation, the recipient can access the shared AR experience of the furniture item.

Special requirements:

- The shared AR package should maintain the accuracy and quality of the furniture item's representation.
- The app should provide a user-friendly interface for both the sender and recipient during the sharing process.

Technology and data variation list:

- The AR sharing feature should be compatible with various messaging apps, email platforms, and social media channels to facilitate sharing.
- The app should be able to handle different furniture data (e.g., 3D models, textures) to ensure that the shared AR experience is consistent with the original view.

Frequency of Occurrence:

- Regularly, especially when users are uncertain about a purchase and seek feedback from friends or when they find a piece of furniture that they believe a friend might like.

Use Case UC4: Collection Sharing

Primary Actor: User

Stakeholders and interests:

- User: Wants to share their favorited furniture collections and spaces with friends and family for feedback, suggestions, or simply to showcase their choices.
- Friends and Family: Interested in viewing the shared collections and spaces to offer feedback or even consider similar styles for their own spaces.
- Furniture Store: Seeks to expand its influence and potentially attract more customers through shared collections and spaces.
- App Developer: Strives to ensure that the sharing functionality is intuitive, smooth, and offers a true representation of the user's collections and spaces.

Preconditions:

- User has the AR furniture app installed on a compatible device.
- User has favorited furniture collections and/or spaces saved in their account.

Success guarantee: User successfully shares their favorited collections and spaces, and the recipients can view the shared items in their entirety.

Main success scenario:

- User opens the AR furniture app and navigates to their account menu.
- Within the account menu, user views tiles representing their favorited collections and spaces.
- User long-presses on the tile of the collection or space they wish to share.
- A sharing menu pops up, providing options for sharing via messaging apps, email, social media, or generating a shareable link.
- User selects their desired sharing method and completes the sharing process.
- Recipient receives the shared collection or space and can click on the link or notification to view the shared items in the AR furniture app.
- If the recipient does not have the AR furniture app, they are prompted to download it to view the shared collection or space.

Extensions: Recipient does not have the AR furniture app installed.

- Recipient is directed to the app store to download and install the AR furniture app.
- After installation, the recipient can access and view the shared collection or space.

Special requirements:

- The sharing functionality should ensure that shared collections and spaces maintain their visual integrity and layout when viewed by recipients.
- User's privacy settings and preferences should be respected, ensuring that only intended items are shared and no personal information is unintentionally disclosed.

Technology and data variation list:

- The collection sharing feature should be integrated with popular messaging apps, email platforms, and social media channels to facilitate diverse sharing options.
- The app should handle different furniture data sets and styles, ensuring consistent representation when collections or spaces are shared.

Frequency of Occurrence:

- Occasionally, when users are particularly proud of their curated collections or spaces, or when they want feedback from trusted friends and family members before making purchasing decisions.

Function: Product Search Feature (Alexander Emory)

Use Case UC1: SearchProducts (Search through product catalog)

Primary Actor: User

Stakeholders and Interests:

User: wants fast and relevant search results

Furniture retailer: wants the user to view and purchase their products

App owner (us): wants to provide relevant results to encourage sale

Preconditions:

User has successfully arrived at the search landing page.

Success Guarantee:

User search is valid. Search results are saved. Relevant products displayed.

Main Success Scenario:

1. User arrives at the search page
2. User types search query into search bar
3. User submits search query
4. System handles query and displays relevant products
5. User browses results
6. User selects the desired product and is taken to the product's page

Extensions:

4a. No results found

1. "No results found" page is displayed with alternative search suggestions
2. User responds to new page:
   - 2a. User selects alternative search suggestions
     - 1. New products are displayed for User
       - *Proceed to step 5*
   - 2b. User types and submits new search query
     - *Return to step 4*

5a. Desired product not found

1. User narrows results or performs a new search
   - 1a. User narrows search results:
     - 1. User sorts results, filters results, or turns on product recommendations
       - 1a. User uses sort feature (outlined in Use Case UC2)
         - 1. Sorted results are displayed
       - 1b. User uses filter feature (outlined in Use Case UC3)
         - 1. Filtered results are displayed
       - 1c. User uses recommendations feature (outlined in Use Case UC4)
         - 1. Recommended products are displayed
       - *Return to step 5*
   - 1b. User performs new search
     - *Return to step 4*

Special Requirements:

- App must be downloaded on iOS device

Technology and Data Variation List:

1a. User is on officially supported iOS device

2a. Search query entered with any iOS supported input device
Frequency of Occurrence:
At least once per purchase

Use Case UC2: SortProducts (Sort product search results)
Primary Actor: User
Stakeholders and Interests:
User: wants fast and relevant search results
Furniture retailer: wants the user to view and purchase their products
App owner (us): wants to provide relevant results to encourage sale
Preconditions:
User is viewing product search results.
Success Guarantee:
Search results are saved. Products are correctly sorted and displayed.
Main Success Scenario:

1. User begins by viewing product search results
2. User selects their preferred sort-method (product price, popularity, or relevance)
3. System handles query and displays results
4. User browses newly sorted results

Extensions:
2a. Incorrect sorting method selected

   1. User selects the wrong sort method (ie. “price” instead of “popularity”)
   2. Results are sorted according to incorrect selection
   3. User selects correct sorting method
   4. Results are re-sorted according to newly selected method

Special Requirements:

   - App must be downloaded on iOS device

Technology and Data Variation List:
1a. User is on officially supported iOS device
2a. Sort method can be selected with cursor, touchscreen, or iOS supported accessibility devices
Frequency of Occurrence:
Any number of times per search, expected to be near once per search

Use Case UC3: FilterProducts (filter product search results)
Primary actor: User
Stakeholders and Interests:
User: wants to quickly filter search results to narrow down product results
Furniture retailer: wants the user to view and purchase their products
App owner (us): wants to provide relevant results to encourage sale
Preconditions:
User is viewing product search results.
Success Guarantee:
Search results are saved. Products are correctly filtered and displayed.

Main Success Scenario:

1. User begins by viewing product search results
2. User selects their preferred filter settings (price, rating, size, color, or availability)
3. User clicks “Apply Filters” to apply the selected search filters
4. System handles query and displays results
5. User browses newly filtered results

Extensions:

4a. Incorrect filter applied

1. System displays incorrectly filtered products
2. User updates filter settings to desired settings
3. User clicks “Apply Filters” to apply the newly updated filters
4. Correctly filtered product results are displayed

    Proceed to step 5

Special Requirements:

- App must be downloaded on iOS device

Technology and Data Variation List:

1a. User is on officially supported iOS device

2a. Filter settings can be selected with cursor, touchscreen, or iOS supported accessibility devices

3a. “Apply Filters” button can be selected with cursor, touchscreen, or iOS supported accessibility devices

Frequency of Occurrence:

Any number of times per search, expected to be near once per search

Use Case UC4: Recommend Products

Primary actor: User

Stakeholders and Interests:

User: wants product recommendations based on their search

Furniture retailer: wants the user to view and purchase their products

App owner (us): wants to provide relevant results to encourage sale

Preconditions:

User is viewing product search results.

Success Guarantee:

Search results are saved. Recommended products are displayed first with a “recommended” icon.

Main Success Scenario:

1. User begins by viewing product search results
2. User toggles on product recommendations
3. System retrieves and displays recommended products first with a “recommended” icon.
4. User browses results with recommendations visible.

Extensions:

3a. No recommended products found

1. “No recommendations found for ‘<User search query>’” message displayed
2. Product results displayed as normal, with no recommendations

Special Requirements:

- App must be downloaded on iOS device

Technology and Data Variation List:

1a. User is on officially supported iOS device
2a. Recommendation settings can be selected with cursor, touchscreen, or iOS supported accessibility devices

Frequency of Occurrence:

Generally 0-1 times per purchase

Function: Furniture Catalog (Alexander Emory)

Use Case 1: Furniture Catalog

Primary Actor: Customer

Stakeholders and Interests:

- Customer: Convenient to open
- Customer: Easy to use
- Customer: Large variety
- Customer: Conforms to other app features easily

Preconditions:

- Availability of furniture
- Tasteful combinations of furniture stored on primary server

Postconditions:

- Customer has a better idea of what to shop for
- Customer has furniture saved in shopping cart feature

Main Success Scenario:

- Customer opens catalog
- Customer browses catalog
    - Customer finds furniture collection they find appealing
    - Customer opens said collection
    - Customer selects all relevant furniture items they'd like
    - Furniture goes to cart
- Customer browses catalog
    - Customer does not find collections they like
    - Customer exits catalog mode

Use Case 2: Color and Style Match

Primary Actor: App

Stakeholders and Interests:

- Customer: Convenient to view similar furniture

Preconditions:

- Availability of furniture
- Furniture descriptions and images of said furniture

Postconditions:

- Customer has a better idea of what to shop for

- Customer can better match furniture

Main Success Scenario:

- App receives furniture information
- App parses through and finds similar furniture styles and colors
- App fuses them into collections in the catalog
- Customer browses catalog

Use Case 3: Select Furniture

Primary Actor: Customer

Stakeholders and Interests:

- Customer: Convenient way to add furniture to cart

Preconditions:

- Availability of furniture
- Tasteful combinations of furniture stored on primary server

Postconditions:

- Customer has furniture in cart to purchase

Main Success Scenario:

- Customer opens catalog
- Customer browses catalog
    - Customer finds furniture collection they find appealing
    - Customer opens said collection
    - Customer selects all relevant furniture items they'd like
    - Furniture goes to cart
- Customer browses catalog
    - Customer does not find collections they like
    - Customer exits catalog mode

Use Case 4: Filter Catalog

Primary Actor: Customer

Stakeholders and Interests:

- Customer: Allows customer to more effectively find certain styles and colors

Preconditions:

- Availability of furniture
- Tasteful combinations of furniture stored on primary server

Postconditions:

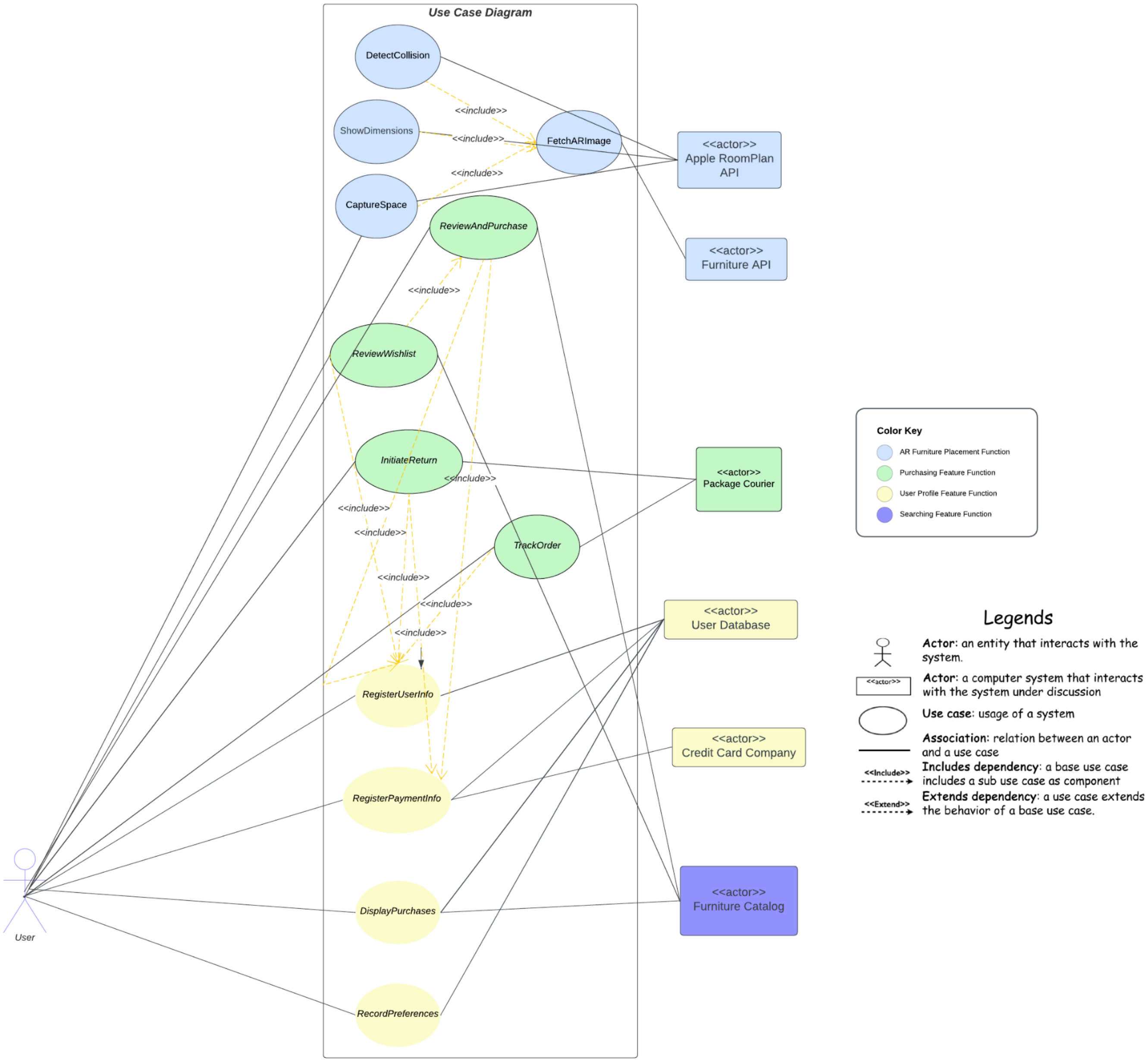
- Customer has a specially-curated list of furniture for them
- Customer has furniture saved in shopping cart feature

Main Success Scenario:

- Customer opens catalog
- Customer browses catalog
- Customer adds in desired filter
    - Customer finds furniture collection they find appealing
    - Customer opens said collection
    - Customer selects all relevant furniture items they'd like

    - Furniture goes to cart
- Customer browses catalog
    - Customer does not find collections they like
    - Customer exits catalog mode

Use Case Diagrams:



Color Key

- 🔵 AR Furniture Placement Function
- 🟢 Purchasing Feature Function
- 🟡 User Profile Feature Function
- 🟣 Searching Feature Function

Legends

Actor: an entity that interacts with the system.

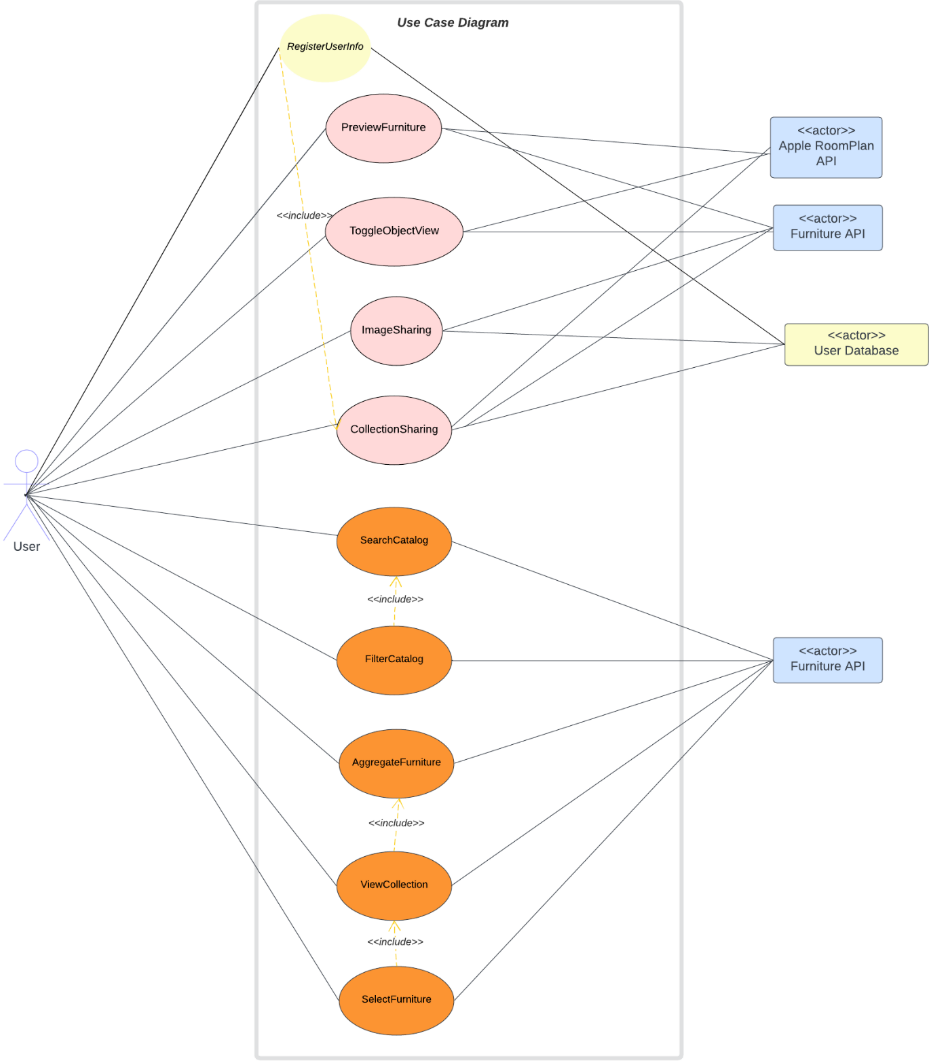
Actor: a computer system that interacts with the system under discussion

Use case: usage of a system

Association: relation between an actor and a use case

<<Include>> **Includes dependency**: a base use case includes a sub use case as component

<<Extend>> **Extends dependency**: a use case extends the behavior of a base use case.

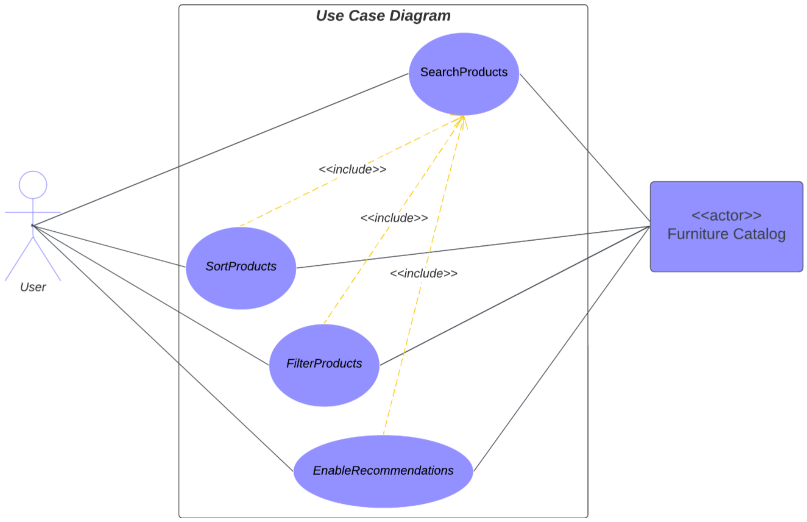


Color Key

- ● AR Furniture Placement Function
- ● AR Furniture View and Sharing Function
- ● User Profile Feature Function
- ● Furniture Catalog Function

Legends

- **Actor**: an entity that interacts with the system.
- <<actor>> **Actor**: a computer system that interacts with the system under discussion
- **Use case**: usage of a system
- **Association**: relation between an actor and a use case
- <<Include>> **Includes dependency**: a base use case includes a sub use case as component
- <<Extend>> **Extends dependency**: a use case extends the behavior of a base use case.



Color Key

● Searching Feature Function

Legends

Actor: an entity that interacts with the system.

<<actor>> **Actor**: a computer system that interacts with the system under discussion

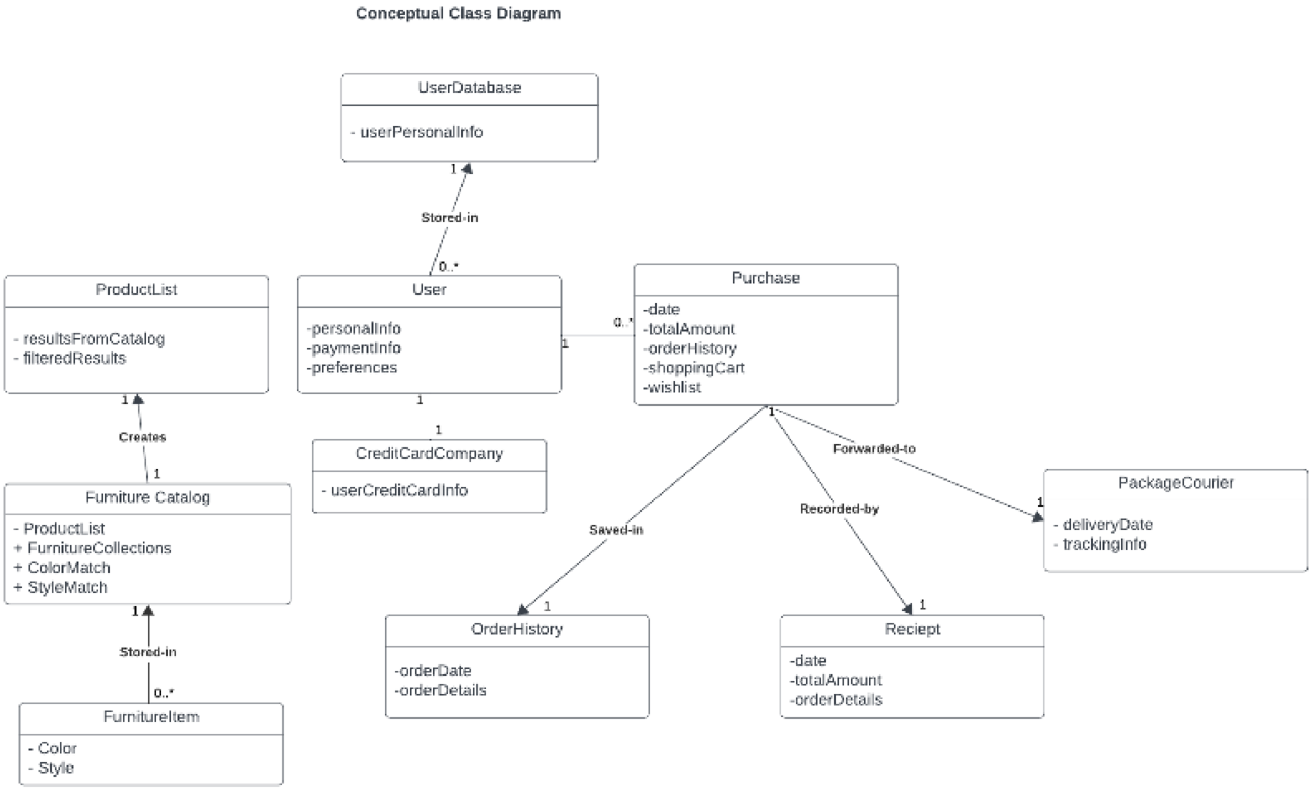
Use case: usage of a system

Association: relation between an actor and a use case

<<Include>> **Includes dependency**: a base use case includes a sub use case as component

<<Extend>> **Extends dependency**: a use case extends the behavior of a base use case.

Conceptual Class Diagram:



Supplemental Specifications:

Non-functional Requirements

Usability:

1. The app must include a help button:
    - This help button will be a link to a phone number that will call a call center to help the user answer any questions they might have.
2. App text size must scale with iOS device accessibility options
    - Selecting a different text size in the user's device settings will scale the text size in the app.

Performance

3. The app is required to load AR images into the user's environment rapidly, with a time limit of 10 seconds.
4. The app must finish scanning the user's environment for room and collision detection in less than 30 seconds

Implementation:

5. User: would be restricted to certain devices to use the app
6. Hiring team: can hire developers who specialize in iOS development
7. The app is designed to only support iOS to limit the scope of the project initially. This allows for a more effective starting product so that the development team can more easily focus on the bulk of the application, as opposed to cross-platform difficulties.

GitHub Repository Link:

https://github.com/anushkar123/CS3704Group1