**1.Authentication System**

* The User class includes methods such as login() and multiFactorAuthentication() to handle the authentication process.
* The login(username, password) method takes a username and password as parameters and verifies them against the stored credentials.
* The multiFactorAuthentication() method implements multi-factor authentication for enhanced security.
* The User class includes an attribute -location, which represents the user's location information.
* This location attribute can store the user's current location or any other relevant location information associated with the user.

This representation ensures that the functionality of the authentication system is encapsulated within the User class and maintains a clear relationship with user account management. The login() method handles user authentication with a valid username and password, while the multiFactorAuthentication() method provides an additional layer of security through multi-factor authentication when needed.

Priority Level: Medium

Precondition: User has a registered account

**2. User Registration**

* The User class includes an attribute -registered, which indicates whether the user account has been registered and email verified.
* Methods such as register() and verifyEmail() are provided to handle the registration process.
* The register() method is responsible for creating a new account with a unique username, email, and password.
* The verifyEmail() method is responsible for verifying the user's email address for account activation.

This representation ensures that the functionality of user registration is encapsulated within the User class and maintains a clear relationship with user account management. The register() method handles the creation of new user accounts, while the verifyEmail() method manages the email verification process for account activation.

Priority Level: Medium

Precondition: User does not have an existing account/registered email address.

**3. Password Recovery**

* The PasswordRecovery class is introduced to handle password recovery functionality.
* The PasswordRecovery class includes attributes such as securityQuestions, verificationCode, and failedAttempts.

The User class has a one-to-one association with the PasswordRecovery class, indicating that each user has one associated password recovery instance.

Priority Level: Medium

Precondition: User has a registered email address.

**4. Logout Functionality**

* The User class includes attributes such as -loggedIn, representing whether the user is currently logged in, and -lastActiveTime, indicating the last time the user was active.
* Methods such as login(), logout(), and autoLogout() are provided to handle user authentication and session management.
* The login() method is responsible for logging in the user.
* The logout() method is responsible for logging out the user from the current device.
* The autoLogout() method checks for user inactivity and logs out the user automatically after a specified period of inactivity.

This representation ensures that the functionality of the logout process is encapsulated within the User class and maintains a clear relationship with the user's session management. The logout() method can handle the user's action of clicking on the "Logout" button, while the autoLogout() method ensures security by logging out users after a period of inactivity.

Priority Level: Medium Precondition: User is logged in.

**5. User Dashboard**

* The UserDashboard class represents the functionality of the user's personalized dashboard.
* It includes attributes such as -userId, -healthMetrics, -environmentalConditions, -recommendations, and -alerts.
* -userId links the dashboard data to the specific user.
* -healthMetrics stores the health metrics displayed on the dashboard.
* -environmentalConditions stores the environmental conditions displayed on the dashboard.
* -recommendations stores personalized recommendations for the user.
* -alerts stores alerts for the user.

This representation ensures that the functionality of the user dashboard is encapsulated within its own class and maintains a clear relationship with the User class. The UserDashboard class can provide users with a personalized view of their health metrics, environmental conditions, recommendations, and alerts upon successful login, enhancing the user experience and facilitating informed decision-making regarding their well-being.

**6. Profile Creation**

* The User class includes attributes such as -profilePicture and -healthGoals to represent the user's profile-related information.
* Methods such as createProfile() and updateProfile() are provided to handle profile creation and updating processes.
* The createProfile() method allows users to create their profile with personal information, including uploading a profile picture and setting health goals and preferences.
* The updateProfile() method enables users to update their profile information, including the profile picture, health goals, and preferences.

This representation ensures that the functionality of profile creation and updating is encapsulated within the User class and maintains a clear relationship with user account management. The createProfile() and updateProfile() methods provide users with the ability to manage their profile information conveniently within the application.

Priority Level: High

Precondition: User is logged in

**7. Health Profile Setup**

* The HealthProfile class is introduced to handle health profile setup functionality.
* The HealthProfile class includes attributes such as medicalHistory, allergies, and healthConditions.
* The User class has a one-to-one association with the HealthProfile class, indicating that each user has one associated health profile instance.

This representation ensures that the health profile setup functionality is encapsulated within its own class and maintains a clear relationship with the User class.

Priority Level: High Precondition: User is logged in.

**8. Real-time Health Monitoring**

* The HealthMonitor class is introduced to handle real-time health monitoring functionality.
* The HealthMonitor class includes attributes such as pulseRate, heartRate, and thermalConditions.
* The HealthMonitor class has a one-to-one association with the User class, indicating that each user has one associated health monitoring instance.

This representation ensures that the real-time health monitoring functionality is encapsulated within its own class and maintains a clear relationship with the User class.

Priority Level: High

**9. Environmental Data Integration**

* The EnvironmentalDataIntegration class is introduced to handle the integration of environmental data.
* The EnvironmentalDataIntegration class includes attributes such as weatherCondition, airQualityIndex, pollenCount, and uvIndex.
* These attributes represent different environmental factors that can impact people's health.
* The EnvironmentalDataIntegration class has a one-to-one association with the User class, indicating that each user has one associated environmental data instance.

This representation ensures that the environmental data integration functionality is encapsulated within its own class and maintains a clear relationship with the User class. Additionally, it provides a more descriptive view of the attributes related to environmental factors.

Priority Level: High Precondition: User has selected a location or destination for health monitoring.

**10. Personalized Health Recommendations**

* These attributes provide a more comprehensive set of environmental measurements, including temperature, humidity, wind speed, air pressure, precipitation, visibility, pollution levels (such as ozone, carbon monoxide, nitrogen dioxide, sulfur dioxide), and particulate matter levels (PM10 and PM2.5).
* Including these attributes in the EnvironmentalDataIntegration class allows for a more detailed representation of environmental conditions, which can be valuable for correlating with health metrics and providing personalized recommendations to users.

Priority Level: High

Precondition: Weather forecast data for the selected time and location is available.

**11. Health Alerts and Warnings**

* The HealthRecommendations class represents the personalized health recommendations functionality.
* It includes attributes such as -userId, -selectedLocation, -weatherForecastData, and -healthInsights.
* -userId and -selectedLocation link the recommendations to the specific user and their chosen location for health monitoring, fulfilling the precondition.
* -weatherForecastData stores weather forecast data for the selected location and time, fulfilling the precondition.
* -healthInsights stores insights derived from real-time health and environmental data, which are used to generate tailored recommendations.
* The User class remains unchanged.

This representation ensures that the personalized health recommendations functionality is encapsulated within its own class and maintains a clear relationship with the User class. The HealthRecommendations class can utilize data from the EnvironmentalDataIntegration class and other sources to generate tailored recommendations for each user.

Priority Level: High

**12. Adaptive Clothing and Activity Suggestions**

* The HealthAlerts class represents the functionality of generating health alerts and warnings.
* It includes attributes such as -userId, -alertType, -alertMessage, and -precautionaryMeasures.
* -userId links the alert to the specific user experiencing the health risk.
* -alertType specifies the type of alert, such as "High UV Index", "Air Quality Warning", etc.
* -alertMessage provides details about the alert, including the nature of the health risk and any relevant information.
* -precautionaryMeasures offers guidance or precautionary measures to mitigate the health risk, such as staying indoors during poor air quality or applying sunscreen during high UV index days.

This representation ensures that the functionality of health alerts and warnings is encapsulated within its own class and maintains a clear relationship with the User class. The HealthAlerts class can utilize data from the EnvironmentalDataIntegration class and real-time health monitoring to generate alerts tailored to each user's health profile and environmental context.

Priority Level: Medium

**13. Food Habit Tracking**

* The FoodTracking class represents the functionality of tracking food habits.
* It includes attributes such as -userId, -loggedFoods, -nutritionInsights, and -dietRecommendations.
* -userId links the food tracking data to the specific user who logged the foods.
* -loggedFoods is an array that stores the daily food intake logged by the user.
* -nutritionInsights provides insights on nutrition derived from the logged food intake.
* -dietRecommendations offers recommendations for a balanced diet based on the user's food habits and nutritional needs.

This representation ensures that the functionality of food habit tracking is encapsulated within its own class and maintains a clear relationship with the User class. The FoodTracking class can provide users with valuable insights into their dietary habits and offer recommendations to help them maintain a healthy and balanced diet.

Priority Level: Medium

**14. Personalized Fitness Program**

* The ActivitySuggestions class represents the functionality of generating adaptive clothing and activity suggestions.
* It includes attributes such as -userId, -clothingRecommendation, and -activitySuggestion.
* -userId links the suggestions to the specific user for whom the recommendations are being provided.
* -clothingRecommendation provides suggestions for suitable clothing based on upcoming weather conditions.
* -activitySuggestion offers recommendations for activities aligned with individual health metrics and environmental factors.

This representation ensures that the functionality of adaptive clothing and activity suggestions is encapsulated within its own class and maintains a clear relationship with the User class. The ActivitySuggestions class can utilize data from the EnvironmentalDataIntegration class and real-time health monitoring to generate tailored recommendations that promote the user's well-being and comfort.

**15. Behavioral Insights Exploration**

* The BehavioralInsights class represents the functionality of exploring behavioral insights.
* It includes attributes such as -userId and -consentGiven.
* -userId links the behavioral insights data to the specific user for whom the analysis is conducted.
* -consentGiven indicates whether the user has provided consent for behavioral data analysis.

Other attributes related to behavioral data analysis, such as machine learning algorithms and insights, can be included as needed within the class.

This representation ensures that the functionality of exploring behavioral insights is encapsulated within its own class and maintains a clear relationship with the User class. The BehavioralInsights class can analyze user responses to health recommendations, identify behavior patterns, and provide adaptive suggestions to help users improve their health habits, provided they have given consent for such analysis.

Priority Level: High

Precondition: User has provided consent for behavioral data analysis.

**16. Integration of Mental and Physical Health**

* The MentalPhysicalIntegration class represents the functionality of integrating mental and physical health.
* It includes attributes such as -userId and -mentalHealthMetricsLogged.
* -userId links the mental and physical health integration data to the specific user.
* -mentalHealthMetricsLogged indicates whether the user has actively logged mental health metrics, fulfilling the precondition for this feature.

This representation ensures that the functionality of integrating mental and physical health is encapsulated within its own class and maintains a clear relationship with the User class. The MentalPhysicalIntegration class can provide users with a holistic approach to well-being by addressing both mental and physical health needs, including stress management and strategies for mental health improvement.

Priority Level: High

Precondition: User has actively logged mental health metrics.

**17. Integration of Telehealth Services**

* The TelehealthIntegration class represents the functionality of integrating telehealth services.
* It includes attributes such as -userId, -scheduledConsultations, and -secureChannels.
* -userId links the telehealth integration data to the specific user.
* -scheduledConsultations stores information about scheduled virtual consultations with healthcare professionals.
* -secureChannels provides secure channels for confidential telehealth interactions.

This representation ensures that the functionality of integrating telehealth services is encapsulated within its own class and maintains a clear relationship with the User class. The TelehealthIntegration class can offer users the option to schedule virtual consultations with healthcare professionals and ensure secure and confidential interactions to address their health needs.

Priority Level: Medium

**18. Emergency Health Support**

* The EmergencySupport class represents the functionality of emergency health support.
* It includes attributes such as -userId, -emergencyContactInfo, and -locationTrackingEnabled.
* -userId links the emergency support data to the specific user.
* -emergencyContactInfo stores the emergency contact information provided by the user.
* -locationTrackingEnabled indicates whether location tracking is enabled for quick access to emergency services.

This representation ensures that the functionality of emergency health support is encapsulated within its own class and maintains a clear relationship with the User class. The EmergencySupport class can offer users quick access to emergency services, location tracking, and provide emergency response protocols to ensure immediate assistance during critical situations.

Priority Level: High

Precondition: User has provided emergency contact information.

**19. Collaboration with Wearable Devices**

* The WearableDeviceIntegration class represents the functionality of collaborating with wearable devices.
* It includes attributes such as -userId, -wearableDevices, and -realTimeSyncEnabled.
* -userId links the wearable device integration data to the specific user.
* -wearableDevices stores information about the wearable devices that are paired and connected to the app.
* -realTimeSyncEnabled indicates whether real-time syncing of health metrics is enabled for enhanced tracking accuracy.

This representation ensures that the functionality of collaboration with wearable devices is encapsulated within its own class and maintains a clear relationship with the User class. The WearableDeviceIntegration class can seamlessly integrate with wearable devices, sync real-time health metrics, and enhance tracking accuracy to provide users with comprehensive health insights.

Priority Level: High

Precondition: Wearable devices are paired and connected to the app.

**20. Comprehensive Health History Record**

* The HealthHistory class represents the functionality of managing the comprehensive health history record.
* It includes attributes such as -userId, -healthMetricsArchive, -environmentalArchive, and -recommendationsArchive.
* -userId links the health history data to the specific user.
* -healthMetricsArchive stores a detailed archive of past health metrics recorded by the user within the app.
* -environmentalArchive stores a detailed archive of past environmental conditions recorded by the app.
* -recommendationsArchive stores a detailed archive of past recommendations provided to the user.

This representation ensures that the functionality of maintaining a comprehensive health history record is encapsulated within its own class and maintains a clear relationship with the User class. The HealthHistory class can provide users with insights into their health trends over time, environmental conditions, and recommendations received, allowing for reflection and informed decision-making regarding their well-being.

Priority Level: High

Precondition: User has a history of recorded health metrics within the app