

# Introduction

In an age of ever-evolving technology, the concept of life-logging precisely digital life-logging has emerged as a powerful tool for recording and analyzing the experiences that shape our lives. It extends beyond simply documenting daily events; it encompasses the meticulous collection and organization of a vast array of personal data. By harnessing the power of digital tools, life-logging allows us to create a comprehensive record of our existence, fostering self-reflection, habit formation, and a deeper understanding of ourselves within the context of our environment. Lifelogging is the practice of digitally recording and storing various aspects of one's life, typically using wearable devices like cameras, smartphones, or other technology. This can include tracking activities, health metrics, location, sleep patterns, social interactions, and more. The term "lifelogging" is derived from "life" and "logging," reflecting the continuous and comprehensive nature of recording one's experiences and activities.

Lifelogging applications allow individuals to collect data passively or actively throughout their day, creating a digital record or "lifelog" of their lives. These life logs can provide insights into patterns, behaviors, and trends over time, offering opportunities for self-reflection, behavior modification, and personal improvement.

The primary purposes of lifelogging vary from person to person but often include self-awareness, goal-setting, health monitoring, and lifestyle optimization. Lifelogging can also have practical applications in healthcare, research, and personal development.

However, lifelogging raises important ethical and privacy considerations, as it involves collecting and storing vast amounts of personal data. Users must weigh the benefits of lifelogging against potential risks to privacy and security. Additionally, there are ongoing discussions about data ownership, consent, and the responsible use of lifelogging technology.

## **Key aspects of lifelogging may include:**

Building upon the concept of digital life-logging, the application incorporates several key aspects that set it apart from traditional tools. Here's a breakdown of these crucial elements:

**Environmental Data Collection:**

* **Comprehensive Weather Data:** This application goes beyond basic temperature by capturing details like "Feels Like," wind speed, humidity, AQI, and pressure. This comprehensive picture allows for a deeper understanding of how weather conditions impact users' experiences and well-being.
* **Noise Level Monitoring:** By measuring sound levels, the application delves into the impact of surrounding noise on users. This data can be analyzed to identify potential disruptions to sleep, concentration, or overall mood.
* **Light Level Detection:** The inclusion of average brightness data provides insights into users' light exposure throughout the day. This information can be valuable for understanding circadian rhythms.

**User Health and Activity Tracking:**

* **Detailed Health Metrics:** In addition to steps and calories burned, this application captures active minutes, and heart rate minutes, and allows for weight input. This richer data set provides a more holistic view of users' physical activity and health.

**Advanced Media Analysis:**

* **Image, Video, and Audio Classification:** Moving beyond simple data capture, this application leverages machine learning to classify the content within images, videos, and audio recordings. This allows users to not only store these memories but also gain insights into their surroundings and activities.

**Integration with Existing Systems:**

* **Calendar Event Integration:** By incorporating calendar events, the application can create a more contextual understanding of users' daily routines and how they interact with environmental factors.

**Overall, these key aspects position your life-logging application to be a powerful tool for:**

* **Personalized Health Management:** By correlating environmental data with health metrics, users can identify potential triggers and make informed decisions to optimize their well-being.
* **Enhanced Self-awareness:** The application empowers users to gain deeper insights into their behavior patterns and how they interact with their surroundings. This knowledge can be used to cultivate positive changes and improve overall quality of life.

## **The core idea behind lifelogging is to:**

The core idea behind lifelogging can be approached from two main perspectives:

**1. Preserving and Recalling Experiences:**

* At its heart, life-logging is about creating a comprehensive record of our lives. This extends beyond simply documenting major events; it encompasses the meticulous collection of everyday experiences, sensory details, and personal reflections.
* Life-logging helps us remember everything, from the big moments to the everyday details, so we can relive them clearly whenever we want.

**2. Self-knowledge and Personal Growth:**

* Lifelogging transcends mere preservation. It's about using the collected data to gain a deeper understanding of ourselves and our place in the world.
* Through analysis of our routines, habits, and how we interact with our environment, life-logging empowers us to:
  + Identify patterns and connections we might have missed otherwise.
  + Gain insights into our strengths, weaknesses, and triggers.
  + Make informed choices to improve our well-being, productivity, and overall quality of life.

Essentially, life-logging bridges the gap between experience and self-awareness. It allows us to not only live our lives but also to learn from them and use that knowledge to shape a more fulfilling future.

# Previous Work History

Lifelogging has been a fascinating concept for decades, with research and development efforts spanning a wide range of disciplines. Here's a brief overview of the previous work history in this area:

**Early Pioneering Efforts:**

* **Vannevar Bush's "As We May Think" (1945):** This seminal article envisioned a hypothetical electromechanical microfilm device called the Memex that could store and retrieve personal information. It laid the groundwork for future concepts of personal information management.

Source 1: [Wikipedia](https://en.wikipedia.org/wiki/As_We_May_Think) | Source 2: [Wikipedia](https://en.wikipedia.org/wiki/Memex)

* **Steve Mann's Wearable Computing Experiments (1980s):** Steve Mann, also known as the "Father of Wearable Computing," documented his life continuously using wearable cameras and other sensors, paving the way for lifelogging technologies.

Source: [Website](https://www.interaction-design.org/literature/book/the-encyclopedia-of-human-computer-interaction-2nd-ed/wearable-computing#:~:text=In%201981%20Mann%20designed%20and,for%20one%2Dhanded%20input).)

* **Robert Shields' Lifelog Project (1972-1997):** Shields manually recorded every 5 minutes of his life for 25 years, resulting in the longest diary ever written (37 million words). His work highlighted the potential of detailed self-documentation.



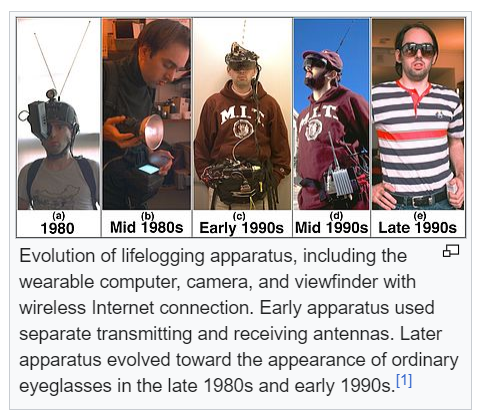


Image Source: [Wikipedia](https://en.wikipedia.org/wiki/Lifelog)

# What’s new?

# Previously, most lifelogging capturing devices were wearable devices to capture audio and video. Now, technology has changed massively. We are connected with our digital gadgets like mobile phones as well. In 2024, the number of smartphone users in the world today is 6.93 Billion, which translates to 85.68% of the world's population owning a smartphone. In total, the number of people who own a smart and feature phone is 7.41 Billion, making up 91.61% of the world's population. Source: [Website](https://www.bankmycell.com/blog/how-many-phones-are-in-the-world)

According to 2023 research from Data Reports, the average screen time for users around the world aged 16 to 64 – across different platforms and devices – is 6 hours 37 minutes per day.

Source: [Website](https://www.independent.co.uk/advisor/vpn/screen-time-statistics#:~:text=According%20to%202023%20research%20from,hours%2037%20minutes%20per%20day.)

According to Big Think, the average person globally sleeps just over 7 hours each night. However, factors like age, education, commute, gender, and geography can affect this average. For example, the Japanese and East Asians in general sleep less than the global average.

The National Sleep Foundation says most adults should aim for seven to nine hours of sleep or an average of eight hours. Experts recommend that adults sleep between 7 and 9 hours a night. Adults who sleep less than 7 hours a night may have more health issues than those who sleep 7 or more hours a night.

Source: [Website](https://bigthink.com/strange-maps/global-sleep-duration/#:~:text=While%20the%20world%20average%20sleep,varies%20by%20almost%20an%20hour.)

So, if we consider the above facts, With nearly 92% of the global population wielding smartphones and spending an average of 6.5 hours glued to their screens daily in their 17hours of active life within 24 hours, these devices have become more than just communication tools – they're powerful, personalized life-logging machines.

As per NTCIR18-Lifelog6 task,

The lifelog task appears to be a benchmark for evaluating how well systems can process and understand personal data collected through lifelogging applications. Here's a breakdown of the three subtasks according to NTCIR:

**1. Lifelog Semantic Access (LSAT):**

* Focuses on retrieving specific moments from a user's lifelog based on a query.
* The query can be in text format (written description), visual (image), or a combination of both.
* This subtask is similar to a traditional search engine but specifically targets personal data.

**2. Lifelog Insight (LIT):**

* Aims to uncover hidden patterns and insights from a user's lifelog data.
* This year's focus is on automatically detecting everyday activities (like sleeping, eating, exercising) from a user's 18-month lifelog data.
* The goal is to develop systems that can analyze large amounts of lifelog data and automatically identify recurring patterns in a user's daily routines.

**3. Lifelog Question Answering (LQAT):**

* Evaluates how well systems can answer questions directly based on a user's lifelog information.
* Participants are provided with a dataset containing questions and corresponding answers related to a user's 85-day lifelog data.
* The goal is to develop systems that can understand and answer questions about a user's experiences and activities as reflected in their lifelog data.

In essence, these subtasks explore different aspects of lifelog data analysis:

* LSAT - Targeted retrieval of specific moments.
* LIT - Uncovering broader patterns and insights.
* LQAT - Answering questions directly based on the data.

By tackling these subtasks, researchers can evaluate and improve the capabilities of systems designed to understand and utilize personal lifelog data.

[Source: [Website](http://lifelogsearch.org/ntcir-lifelog/)]

# Objective

# 

Through our life-logging application, we are focusing on collecting user data with user consent.

We are collecting data in different slabs:

1. Personal Data - Here we will ask the user the following information:
   1. **First Name**: It is required to introduce the person.
   2. **Last Name**: It is required to introduce the person.
   3. **Date of Birth**: To calculate the user's current age. Age informs lifelogging insights, tracking health, activity, nutrition, and disease across life stages.
   4. **Gender**: Gender influences data interpretation in lifelogging for tailored insights and understanding diverse health patterns.
   5. **Weight**: Weight impacts health tracking in lifelogging for fitness and wellness monitoring.

| Height (ft) | Height (cm) | Weight Range (kg) for Men | Weight Range (kg) for Women |
| --- | --- | --- | --- |
| 5'2" | 157 | 43 - 54 | 41 - 50 |
| 5'4" | 163 | 46 - 58 | 43 - 53 |
| 5'6" | 168 | 49 - 62 | 46 - 56 |
| 5'8" | 173 | 52 - 66 | 49 - 59 |
| 5'10" | 178 | 55 - 70 | 52 - 62 |
| 6'0" | 183 | 59 - 74 | 55 - 65 |

1. Weather - Here we will get the weather information of the user’s surroundings:
2. **Date & Time:** Crucial for tracking historical weather patterns and comparing current conditions with forecasts.
3. **Latitude & Longitude:** Provides context for the general climate zone (tropical, temperate, etc.) based on proximity to the equator.
4. **City & Country:** Offers general location details to understand typical weather patterns for that area.
5. **Temperature & Feels Like** Measure how hot or cold it is and how it feels on your skin, considering factors like wind and humidity.
6. **Wind Speed & Direction:** Indicates wind strength and direction, which can affect comfort level, potential hazards, and even storm strength.
7. **Humidity:** Measures moisture content in the air, influencing precipitation chances and how hot it feels.

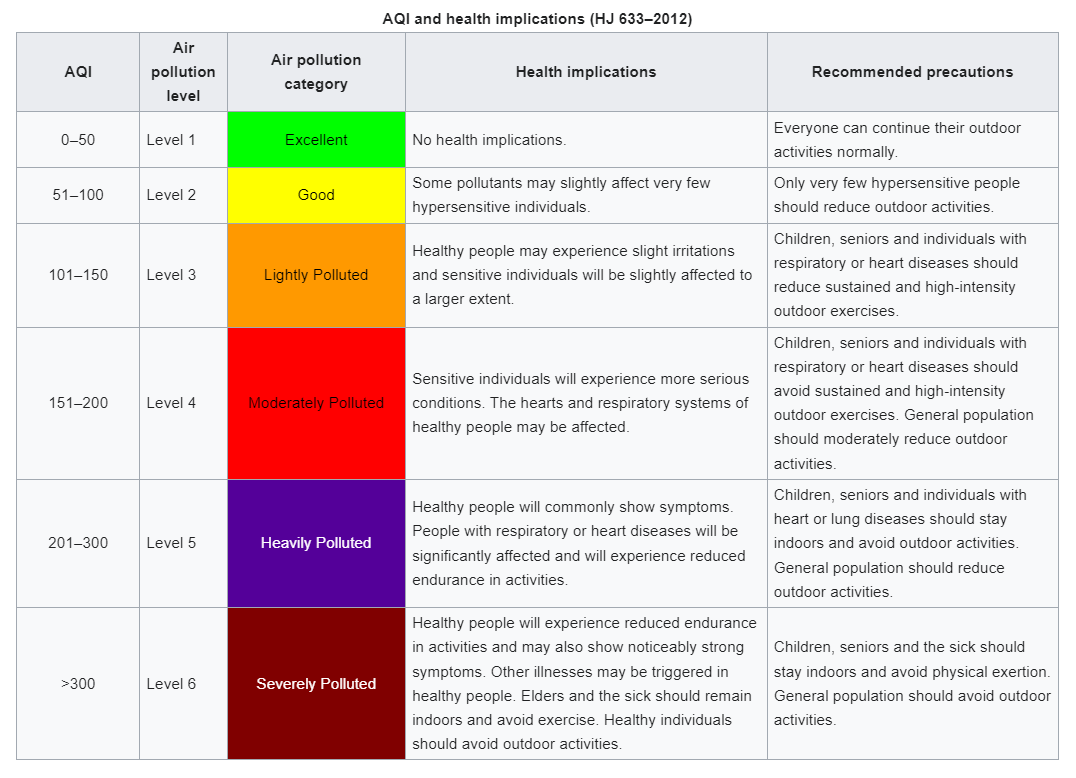
The ideal humidity level for comfort and health falls within a range, rather than having a single "best" value. Here's a breakdown:

* **Poor:** Below 30% RH (relative humidity) - This is too dry. It can cause dry skin, respiratory problems, and static electricity.
* **Good:** 30-40% RH - This is acceptable but on the drier side.
* **Better:** 40-50% RH - This is considered the ideal range for most people. It promotes comfort and minimizes health risks.
* **Not Ideal (can be good for some):** 50-60% RH - This is still acceptable for most people, but can feel muggy or stuffy.
* **Poor:** Above 60% RH - This is too high. It can lead to mold growth, dust mite proliferation, and discomfort.

**Additional factors to consider:**

1. **Season:** Your comfort level may vary depending on the season. In winter, drier air might feel more comfortable than in summer.
2. **Health conditions:** People with allergies or respiratory problems may be more sensitive to humidity levels.
3. **Air Pressure:** Helps predict weather patterns as high-pressure associates with clear skies and low pressure with storms.
4. **Visibility:** Indicates how far you can see, impacted by factors like fog or haze.
5. **Weather Type & Description:** Provides a clear picture of the current weather conditions (e.g., sunny, rainy, snowy).
6. **AQI:** Measures air quality, essential for health awareness, especially for people with respiratory issues.

An air quality index (AQI) is an indicator developed by government agencies to communicate to the public how polluted the air is or how contaminated it is forecast to become. As air pollution levels rise, so does the AQI, along with the associated public health risk. Children, the elderly and individuals with respiratory or cardiovascular problems are typically the first groups affected by poor air quality. When the AQI is high, governmental bodies generally encourage people to reduce physical activity outdoors, or even avoid going out altogether. When wildfires result in a high AQI, the use of a mask (such as an N95 respirator) outdoors and an air purifier (incorporating both HEPA and activated carbon filters) indoors are also encouraged. [**Source**: [Website](https://en.wikipedia.org/wiki/Air_quality_index)]



3. Fitness - Here we will get user physical activity and health-related information.

1. **Step counts:** When tracked, step counts can be used to **indicate** overall activity level, with higher counts **being associated** with better fitness and health benefits. They serve as a good starting point for **tracking** daily movement and **setting goals** for improvement.

According to 10000steps.org, the goal of 10,000 steps is the recommended daily step target for healthy adults to achieve health benefits. With continual advances in technology and our lives becoming more sedentary it now takes a concerted effort to make active choices. We can all make the effort to move more in small ways across the day, by parking further from the entrance to shops, taking stairs instead of escalators, standing while at work, and walking to socialize with friends and family.

Activity trackers provide data that enables you to become aware of your physical activity levels, work towards a goal, and monitor progress. Studies using the 10,000 steps per day goal have shown weight loss, improved glucose tolerance, and reduced blood pressure from increased physical activity toward achieving this goal. The following pedometer indices have been developed to provide a guideline on steps and activity levels:

* Sedentary is less than 5,000 steps per day
* Low activity is 5,000 to 7,499 steps per day
* Somewhat active is 7,500 to 9,999 steps per day
* Active is more than 10,000 steps per day
* Highly active is more than 12,500

Although the program promotes the goal of reaching 10,000 steps each day for healthy adults, this goal is not universally appropriate across all ages and physical functions. There are some groups where the goal of 10,000 steps may not be accurate, such as the elderly and children. Your individual step goal should be based on current activity levels and overall health and fitness.

Source: [Website](https://www.10000steps.org.au/articles/healthy-lifestyles/counting-steps/)

According to Wikipedia, we can also classify this in other ways:

| **Activity level** | **Steps per day** |
| --- | --- |
| Basal | Below 2,500 |
| Limited | 2,500–4,999 |
| Low | 5,000–7,499 |
| Somewhat active | 7,500–9,999 |
| Active | 10,000-12,499 |
| Very active | over 12,500 |

**Source**: [Wikipedia](https://www.healthline.com/health/how-many-steps-a-day#how-many-steps-per-day)

| **Age** | **Step Count** | **Activity Level** | **Notes** |
| --- | --- | --- | --- |
| Children (6-12): | No Specific Recommendation | - | Children are naturally active. Focus on encouraging active play throughout the day. |
| Teenagers (13-18): | No Specific Recommendation | - | Teenagers are also naturally active. However, screen time and sedentary habits can creep in. Encourage participation in sports or active hobbies. |
| Young Adults (19-39): | \* Men: Below 5,000\* | Sedentary | Aim to increase activity throughout the day. |
| Women: Below 4,000 | Sedentary | Aim to increase activity throughout the day. |  |
| \* Men: 5,000 - 7,499\* | Low Active | Start incorporating more movement into your routine. |  |
| Women: 4,000 - 7,499 | Low Active | Start incorporating more movement into your routine. |  |
| Both: 7,500 - 9,999 | Somewhat Active | A good baseline for maintaining health benefits. |  |
| \*Both: 10,000+ | Active | Aim for this for additional health benefits. |  |
| Middle Adults (40-59): | \* Men: Below 4,000\* | Sedentary | Increase activity levels gradually. |
| Women: Below 3,000 | Sedentary | Increase activity levels gradually. |  |
| Both: 4,000 - 6,999 | Low Active | Gradually increase movement towards a more active goal. |  |
| Both: 7,000 - 9,999 | Somewhat Active | A good baseline for maintaining health benefits. |  |
| \*Both: 10,000+ | Active | Aim for this for additional health benefits. |  |
| Older Adults (60+): | \* Below 3,000\* | Sedentary | Focus on gentle activity for improved mobility and health. |
| 3,000 - 4,000 | Low Active | A good starting point, gradually increase if able. |  |
| 4,000 - 6,000 | Somewhat Active | Aim for this for maintaining health benefits. |  |
| \*Above 6,000+ | Active | Great achievement, continue focusing on regular movement. |  |

1. **Active Minutes:** The amount of time spent in moderate-to-vigorous activity, measured by active minutes, is a more important measure of fitness than steps alone. This is because active minutes **highlight** the time dedicated to activities that **elevate** heart rate and **improve** cardiovascular health.

According to Google, To keep your heart healthy, the [American Heart Association](http://www.heart.org/en/healthy-living/fitness/fitness-basics/aha-recs-for-physical-activity-in-adults) (AHA) encourages you to stay active. Each week, they recommend you do at least 150 minutes of moderate activity or 75 minutes of vigorous activity. To help you follow these recommendations, Google Fit tracks your exercise in the form of Heart Points and Steps.

1. **Calories Expended:** Calories expended refer to the total estimated calories burned for the day. Tracking calories expended can help monitor weight management and understand how activity level impacts energy expenditure.
2. **Heart-minute Count:** Heart-minute count likely refers to the amount of time spent exercising at a moderate intensity, specifically targeting heart rate. This metric **serves as** a good indicator of cardiovascular health, as regular moderate-intensity exercise strengthens the heart and improves circulation.

4. Image Classification: Image classification can reveal activities, objects, and scenes in life logs, providing insights into daily routines and experiences.

1. **Timestamp**: It is useful for tasks like understanding lighting changes or analyzing activity over time.
2. **Prediction**: In image classification, predictions tell you the likely category (e.g., cat, car) an image belongs to.
3. **Brightness**: Brightness alone in image classification can indicate illumination variations

4. Audio classification: Audio classification in life logging helps understand activities and environments, providing a richer context for logged experiences.

1. **Max\_amplitude\_in\_audio**: It likely indicates the peak volume level of the audio and doesn't directly tell us about audio classification.
2. **Max\_amplitude\_in\_decible**: It can tell you about the loudness of sounds within an audio clip, but it might not directly determine audio classification.

The decibel (dB) is a unit used to measure the intensity of sound. The decibel scale is logarithmic, so a 10 dB increase in sound level is perceived as a doubling of loudness. Here's a table showing some common sounds and their decibel levels:

| **Sound** | **Decibel Level (dB)** | **Effects** |
| --- | --- | --- |
| Rustling leaves | 20 dB | Very quiet, barely audible. |
| Whisper | 30 dB | Quiet, can be easily heard in a quiet room. |
| Normal conversation | 60 dB | Moderate, normal conversation level. |
| Hair dryer | 80 dB | Loud, can be disruptive over time. |
| City traffic | 85 dB | Very loud, prolonged exposure can damage hearing. |
| Motorcycle engine | 95 dB | Extremely loud, can cause hearing damage. |
| Live music | 100-110 dB | Very loud, and can cause immediate discomfort and long-term hearing damage. |
| Car horn | 110 dB | Very loud, and can cause immediate discomfort and long-term hearing damage. |
| Fireworks | 140-160 dB | Extremely loud, can cause permanent hearing damage. |

Long or repeated exposure to loud noises (above 85 dB) can damage hearing. If one is regularly exposed to loud noises, it is important to wear hearing protection.

**Source**: Gemini

1. **Prediction**: In audio classification, predictions tell you the likely category of an audio clip (e.g., speech, music, genre).

Based on these collected data, the main objectives of this application are to find out the following information that is relevant to tracking the user’s activity and improving the user’s daily lifestyle.

1. **Environmental Data Collection**
   1. Analysis of User’s presence with Timestamp along with latitude and longitude.
   2. Analysis of Temperature with Timestamp
   3. Analysis of Feels Like with Timestamp
   4. Analysis of Humidity with Timestamp
   5. Analysis of Visibility and AQI with Timestamp
   6. Analysis of Weather type and AQI with Timestamp

Combining user location data (timestamps, latitude, longitude) with weather information (temperature, feels like, humidity, visibility, AQI) can be incredibly helpful for doctors from several perspectives:

**1. Personalized Diagnosis and Treatment:**

* **Environmental Triggers:** By correlating a patient's location with weather changes, doctors can identify potential environmental triggers for their condition. For example, a rise in asthma attacks coincides with high pollen counts or increased allergy symptoms during periods of high humidity.
* **Targeted Treatment:** This information allows doctors to tailor treatment plans based on the patient's environment. For example, adjusting medication dosages for patients with respiratory issues during periods of low visibility or high pollution.
* **Remote Monitoring:** Doctors can use this data to remotely monitor patients with chronic conditions, especially those sensitive to weather changes. For instance, tracking a patient's location and the local AQI to see if they've been exposed to unhealthy air quality.

**2. Improved Prognosis and Prevention:**

* **Predictive Risk Assessment:** Doctors can leverage weather forecasts and historical data to predict potential health risks for their patients based on location. This allows for proactive interventions, like advising patients with heart conditions to avoid strenuous activity during extreme heat or recommending flu vaccinations before peak flu season.
* **Long-Term Management:** Understanding how weather patterns affect a patient's health over time can help doctors develop long-term management strategies. For example, identifying recurring flare-ups during specific seasons can prompt preventative measures ahead of time.

**3. Enhanced Patient Communication:**

* **Empowering Patients:** Doctors can educate patients about how weather affects their health and offer personalized advice on managing their condition during different weather scenarios. This empowers patients to take charge of their health and make informed decisions about their daily activities.

**4. Research and Public Health:**

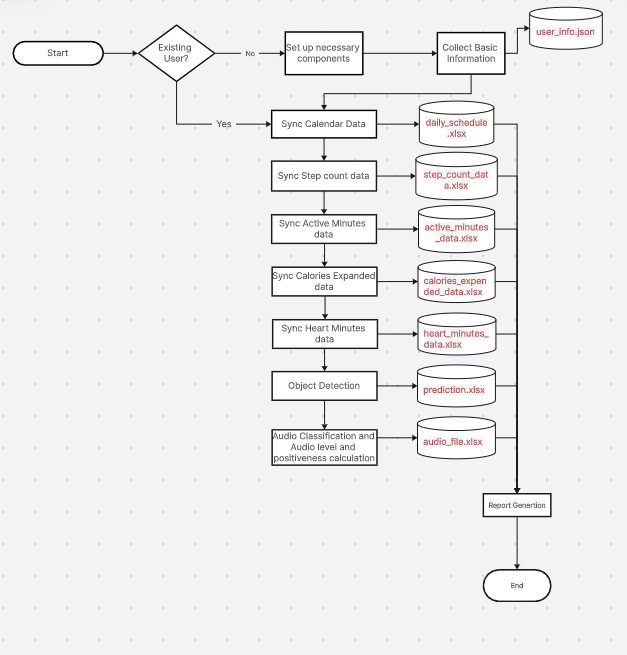
* **Identifying Trends:** Analyzing large datasets of patient location and weather data can contribute to research efforts. This may reveal previously unknown correlations between weather patterns and specific health conditions.
* **Public Health Initiatives:** Large-scale data analysis can inform public health initiatives related to weather-related health risks. For example, issuing air quality advisories or targeting vaccination campaigns based on weather forecasts and population data.

Overall, this application can empower doctors to provide more personalized and preventative care, considering the influence of weather on their patients' health.

1. **User Health and Activity Tracking**
   1. Analysis of Heart Minutes with Timestamp
   2. Analysis of Active Minutes with Timestamp
   3. Analysis of Step Count with Timestamp
   4. Analysis of Expended Calories with Timestamp
   5. Average Heart Minutes of Daily/Weekly/ Monthly Basis
   6. Average Active Minutes of Daily/Weekly/ Monthly Basis
   7. Average Step Count of Daily/Weekly/ Monthly Basis
   8. Average Expended Calories of Daily/Weekly/ Monthly Basis
2. **Advanced Media Analysis**
   1. Analysis of Predicted objects with timespan
   2. Analysis of Brightness with Timespan
   3. Analysis of Noise Level with Timespan
   4. Analysis of Text Emotion Prediction with Timespan
3. **Integration with Existing Systems**
   1. Number of Activities performed by users daily.
   2. Activity Status of users daily.
   3. Activity Status of users with temperature and humidity
   4. Activity Status of users with AQI
4. **Cross Examine**
   1. Analysis of Average Noise Level with Average Heart Minutes and Average Status of Activity on Daily Views.
   2. Analysis of Average Brightness Level with Average Heart Minutes and Average Status of Activity on Daily Views.
   3. Analysis of AQI with Average Heart Minutes and Average Status of Activity on Daily Views.
   4. Analysis of Humidity with Average Heart Minutes and Average Status of Activity on Daily Views.
   5. What percentage of events are completed on time vs What percentage of events are not completed on time vs What percentage of my events are canceled?
   6. Average Sleeping Time daily.

# Functionality:

# 



## User Information Collection

**Import Statements:**

* os: Provides functions for interacting with the operating system (used for file path verification).
* JSON: Enables working with JSON data format for storing user information.
* google.colab (optional): Used for mounting Google Drive if the code is intended for a Google Colab environment.
* tabulate: Used to present user information in a clear tabular format for better readability.
* IPython.display.Audio (optional): Potentially used for audio output (not implemented in this code).

get\_user\_information Function:

* This function interacts with the user to collect their basic information:
  + First and Last Name
  + Date of Birth (separated by Day, Month, and Year)
  + Weight
  + Gender
* The information is stored in a Python dictionary (user\_info) with clear labels.

save\_user\_information Function:

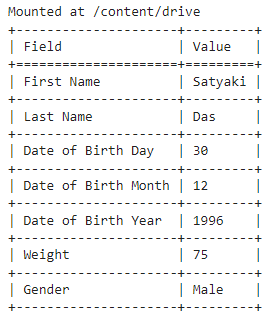
* This function takes the user\_info dictionary as input.
* It defines a file path (file\_path) within Google Drive (assuming you're using Google Colab) to store the user information.
* It opens the file for writing ('w') and uses json.dump to convert the dictionary into a JSON format and save it to the file.
* Upon successful saving, a confirmation message is printed.

print\_user\_information Function:

* This function takes the user\_info dictionary as input.
* It converts the dictionary into a list of lists (data) suitable for the tabulate library.
* It iterates through the user information dictionary, handling nested structures (like Date of Birth) and creating a well-formatted list for the table.
* It uses tabulate to print the user information in a visually appealing table with headers ("Field" and "Value").

Main Script:

* The script defines the file path for user information storage in Google Drive.
* It checks if the file already exists using os.path.exists.
* If the file exists:
  + It loads the existing user information from the JSON file using json.load.
  + You can uncomment the commented-out message to greet the returning user. (Optional)
  + It calls the print\_user\_information function to display the loaded user data.
* If the file doesn't exist:
  + It calls the get\_user\_information function to collect new user information.
  + It calls the save\_user\_information function to save the newly collected user data to the specified file path.
  + You can uncomment the commented-out message to greet the new user. (Optional)
  + It constructs a personalized welcome message using string formatting and the user's first name.



## Data Collection (Sync) and Integration for Life-Logging

Imports:

* csv: Used for working with CSV files.
* threading (not used in this code snippet): Used for creating threads.
* time (not used in this code snippet): Used for time-related operations.
* datetime: Used for working with dates and times.
* google.colab (not used in this code snippet): Used for interacting with Google Colab.
* requests: Used for making HTTP requests to APIs.

API Keys and URLs:

* api\_key: API key for OpenWeatherMap weather data.
* city: City name for weather data (set to "Kolkata" here).
* base\_url: Base URL for OpenWeatherMap weather API.
* url: URL for retrieving weather data for Kolkata.
* AQIapi\_key: API key for Air Quality Index data.
* urlAQI: URL for retrieving AQI data for Kolkata.

File Paths:

* weather\_data\_file\_path: Path to the CSV file where weather data is stored.
* calendar\_data\_file\_path: Path to the Excel file where calendar data is stored.
* step\_count\_data\_file\_path: Path to the Excel file where step count data is stored.
* active\_minutes\_data\_file\_path: Path to the Excel file where active minutes data is stored.
* calories\_expended\_data\_file\_path: Path to the Excel file where calories expended data is stored.
* heart\_minutes\_data\_file\_path: Path to the Excel file where heart minutes data is stored.

Functions:

* write\_to\_excel: Writes data to an Excel file.
* column\_names: List of column names for the weather data CSV file.
* getCurrentDate: Gets the current date.
* getCurrentTime: Gets the current time (adjusted for time zone difference).
* getLatitude: Gets the user's latitude using geolocation.
* getLongitude: Gets the user's longitude using geolocation.
* getCity: Gets the user's city using geolocation (or defaults to "Kolkata" here).
* getCountry: Gets the user's country using geolocation (or defaults to "IN" here).
* getTemperature: Gets the temperature from the weather data (in Celsius).
* getFeelsLike: Gets the feels-like temperature from the weather data (in Celsius).
* getWindSpeed: Gets the wind speed from the weather data.
* getHumidity: Gets the humidity from the weather data.
* getPressure: Gets the pressure from the weather data.
* getVisibility: Gets the visibility from the weather data.
* getWeatherType: Gets the weather type from the weather data.
* getWeatherDescription: Gets the weather description from the weather data.
* getWindDegree: Gets the wind direction (in degrees) from the weather data.
* aqi\_checker: Gets the Air Quality Index (AQI) for the city.
* write\_to\_csv: Writes data to a CSV file.
* sync\_weather\_data: Synchronizes weather data by calling the relevant functions and writing the data to a CSV file.
* sync\_Calendar\_data (not fully implemented): Synchronizes calendar data (likely from an API) and updates a user on their schedule.
* get\_step\_count: Retrieves step count data from a Nocode API.
* sync\_step\_count: Synchronizes step count data by calling get\_step\_count and writing the data to an Excel file.
* Similar functions exist for sync\_active\_minutes, sync\_Calories\_Expended, and sync\_heart\_minutes\_count to retrieve and store data from Nocode APIs for active minutes, calories expended, and heart minutes respectively.

Main Section:

* The script checks if it's the first time running for today (sync\_weather\_data).
* If it's not the first time, it skips getting weather data.
* Otherwise, it retrieves weather data using the defined functions and writes it to the CSV file.
* The commented-out section (sync\_Calendar\_data) seems to be for retrieving and updating calendar events.
* The rest of the code calls functions to get and store data for step count, active minutes, calories expended, and heart minutes.

Overall, this script automates the process of collecting weather data, fitness data, and potentially calendar data, and stores it in Google Drive for easy access and analysis.

## Image Processing and Object Detection with Brightness Level Checking

**Imports:**

* Necessary libraries for image processing (OpenCV, NumPy), object detection (MediaPipe), working with Google Colab, and data manipulation (pandas).

**Functions:**

* FinalPrediction: Takes a list of lists containing object classifications, flattens it, finds the most common object using Counter, and returns it.
* visualize: Takes an image and a list of detections, draws bounding boxes with labels and scores on the image, and displays it.
* ImageClassification: Creates an object detector using the provided model path (efficientdet.tflite), converts an image to the required format, detects objects, and returns a list of category names found in the image.
* js\_to\_image: Converts a base64 encoded image string from JavaScript to an OpenCV image.
* bbox\_to\_bytes: Converts a NumPy array representing a bounding box to a base64 encoded byte string for overlay on the video stream.
* video\_stream: Defines JavaScript code to create a live video stream using the webcam and provides functions to capture frames and display them with labels and bounding boxes.
* video\_frame: Calls the video\_stream function to capture a frame, optionally provides a label and bounding box, and returns the captured image data.
* write\_to\_excel: Saves data (either a Pandas DataFrame or a list) to an Excel file, handling cases where the file already exists.
* measure\_brightness: Converts an image to grayscale and calculates the average brightness using NumPy.
* assess\_brightness: Takes the brightness value and assigns a category ("High", "Moderate", or "Low") based on thresholds.

**Main Script:**

1. **Imports:** Includes necessary libraries.
2. **Mount Google Drive:** Mounts Google Drive to access the specified Excel file path.
3. **Start Video Stream:** Calls the video\_stream function to start capturing video from the webcam.
4. **Initialize Variables:** Sets initial label and bounding box for the video stream.
5. **Capture Video Frames:**
   * Loops to capture multiple frames (controlled by i counter).
   * Initializes empty lists for storing classifications and brightness data.
   * Captures a frame using video\_frame.
   * Converts the captured image to OpenCV format.
   * Creates an empty transparent overlay for the bounding box.
   * Converts the image to grayscale for face detection.
   * Performs object classification using ImageClassification and stores the results.
   * Measures the brightness of the image using measure\_brightness and stores the value.
   * Detects faces using the Haar Cascade face detector and updates the bounding box overlay if a face is found.
   * Converts the bounding box overlay to a base64 encoded byte string.
   * Increments the counter i.
6. **Process Captured Data:**
   * Prints the captured classifications.
   * Uses FinalPrediction to find the most common object class from all captured frames.
   * Calculates the average brightness across captured frames.
   * Creates a dictionary with timestamp, predicted object class, and average brightness.
   * Appends the dictionary to a list for storing all results.
7. **Exception Handling:** Catches any errors during video capture and prints the error message.
8. **Convert Data to DataFrame:** Converts the list of dictionaries containing results to a Pandas DataFrame.
9. **Write Data to Excel:** Calls the write\_to\_excel function to save the DataFrame to the specified Excel file.

Overall, this script demonstrates how to use various libraries to capture video, perform object classification with a pre-trained model, detect faces, measure image brightness, and organize the results in an Excel spreadsheet.

## Audio Classification

**Libraries:**

* Imports various libraries for audio processing (librosa, pydub, soundfile), speech recognition (speech\_recognition), data manipulation (pandas, numpy), file handling (os), and Google Colab functionalities (drive).

**Functions:**

* convert\_to\_mp3\_pydub: Converts a WAV file to MP3 using pydub (requires an external encoder like ffmpeg).
* convert\_mp3\_to\_WAV: Converts an MP3 file to WAV using soundfile.
* wav\_to\_Text: Uses speech\_recognition to recognize text from a WAV audio file.
* wav\_to\_mp3\_to\_wav\_Text: Combines the previous functions to convert WAV to MP3, then back to WAV, and finally performs speech recognition.
* record: Uses Javascript to access the microphone, record audio for a specified duration, save it as a WAV file, and return the filename.
* calculate\_decibels: Calculates the decibel level from the audio amplitude.
* surroundingNoiseLevel:
  + Mounts Google Drive (if necessary) to access the pre-trained YAMNet model for sound classification.
  + Loads the YAMNet model from Google Drive or downloads it if not found.
  + Records audio for 10 seconds.
  + Loads the recorded audio and prepares it for analysis.
  + Uses YAMNet to predict the type of surrounding noise (e.g., traffic, speech, music).
  + Calculates the maximum amplitude and decibel level of the audio.
  + Returns a list containing the maximum amplitude, decibel level, and predicted noise type.
* save\_to\_excel: Saves data (a list of dictionaries) to an Excel file, handling cases where the file already exists.
* store\_data\_continuously: Continuously records audio in a loop:
  + Calls surroundingNoiseLevel to analyze the audio.
  + Gets the current timestamp and performs speech recognition on the audio.
  + Combines the results (timestamp, amplitude, decibel level, prediction, text) into a dictionary.
  + Saves the data to the specified Excel file.
  + Waits for a short period (commented out) before repeating.

**Example Usage:**

* Defines the path to the Excel file where the data will be stored.
* Calls store\_data\_continuously to start recording and storing data indefinitely (until interrupted).

**Overall**, this script demonstrates how to perform various audio-related tasks using Python libraries and Google Colab. It can be used to monitor surrounding noise levels, convert speech to text, and store the results for further analysis.

## Future Aspects of Life-logging

## Usage of Life-logging

## Conclusion