BELLABEAT CASE STUDY PORTFOLIO

Table of Contents

Introduction

Bellabeat, a pioneering company specializing in health-focused technology for women, has rapidly established itself as a leader in the wellness sector since its inception in 2013. Under the leadership of Urška Sršen, co-founder and Chief Creative Officer, Bellabeat aims to leverage smart device fitness data to explore new growth avenues. Sršen has tasked the marketing analytics team with analyzing the usage data from Bellabeat's smart devices to understand consumer behavior. The goal is to generate actionable insights that can guide Bellabeat's marketing strategy.

My Role

As a junior data analyst on the marketing analytics team at Bellabeat, my responsibility is to examine the usage data of one of Bellabeat's products. My analysis will uncover how consumers interact with their smart devices, providing valuable insights to shape the company's marketing strategy. I will present my findings and strategic recommendations to the Bellabeat executive team to support their decision-making process.

The project adheres to the five key steps of the data analysis process: Ask, Prepare, Process, Analyze & Share, and Act.

Bellabeat Products

- Bellabeat App: Connects with Bellabeat's smart wellness products to offer comprehensive health insights.
- Leaf: A versatile wellness tracker wearable as a bracelet, necklace, or clip, monitoring activity and sleep.
- Time: A stylish wellness watch that tracks activity, sleep, and stress with smart technology.
- Spring: A smart water bottle that monitors daily water intake to help users stay hydrated.
- Bellabeat Membership: Provides 24/7 personalized guidance on nutrition, activity, sleep, health, and mindfulness-based on individual goals.

Phase 1: Ask

Business Task

The objective is to analyze smart device usage data to understand consumer behavior with non-Bellabeat smart devices. Based on these insights, select one Bellabeat product and apply the findings to enhance Bellabeat's marketing strategy.

Questions for Analysis:

I. What are the current trends in smart device usage?
II. How might these trends be relevant to Bellabeat customers?
III. In what ways can these trends inform and improve Bellabeat's marketing strategy?

Key Stakeholders:

- Urška Sršen: Co-founder and Chief Creative Officer of Bellabeat.
- Sando Mur: Co-founder, mathematician, and a key executive team member at Bellabeat.
- Bellabeat Marketing Analytics Team: Responsible for implementing and optimizing marketing strategies based on the analysis and recommendations from this study.

Phase 2: Prepare - Data Sources

For this case study, we utilized Fitbit fitness tracker data, sourced from Kaggle and provided by Mobius.

About the Dataset:

The dataset was collected through a survey distributed via Amazon Mechanical Turk between March 12, 2016, and May 12, 2016. It includes data from thirty eligible Fitbit users who agreed to share their personal tracker information. The dataset features minute-level output for physical activity, heart rate, and sleep monitoring.

Analysis Focus:

For this analysis, I will concentrate on daily and hourly data, rather than minute-by-minute performance details. To facilitate this, I will integrate several tables to provide a comprehensive data view.

Phase 3: Process - Data Preparation

For this project, I used MySQL Workbench to process and analyze the data and Power BI for visualization.

<u>Data Exploration and Quality Checks:</u>

I used the following tables in MySQL Workbench: Daily Activity_merged, Dailyactivity_sleep,Dailycalories_merged,Dailyintensities_merged,Dailysteps_merged, sleepday_merged, and weightloginfo_merged.

Step 1: Checking Unique IDs

I reviewed the tables for the number of unique user IDs. All tables had 33 unique user IDs, except sleepday_merged, which had 24 unique user IDs, and weightloginfo_merged, which had 8 unique user IDs.

```
#Checking the count for distinct id for each table
Select count(distinct(id))
From dailyactivity_merged;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

count(distinct(id))
33

```
Select count(distinct(id))
From weightloginfo_merged;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	count(distinct(id))
▶	8

```
Select count(distinct(id))
From sleepday_merged;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	count(distinct(id))
▶	24

```
Select count(distinct(id))
From dailyintensities_merged;

Select count(distinct(id))
From dailysteps_merged;

Select count(distinct(id))
From hourlyintensities_merged;

Select count(distinct(id))
From hourlycalories_merged;

Select count(distinct(id))
From hourlysteps_merged;
```

Step 2: Utilizing only Three Tables and Creating Staging Tables

The dailyActivity_merged dataset includes daily calories, intensities, and steps, making the separate datasets for these metrics redundant for this analysis. Therefore, I utilized only three tables for the analysis Daily Activity_merged, sleepday_merged, and weightloginfo_merged.

I created three staging tables to preserve the original data during the cleaning process. The staging tables created are:

- dailyactivity for Daily Activity_merged
- sleepday for sleepday_merged
- weightloginfo for weightloginfo_merged

These tables ensure that the original data remains intact while I perform data cleaning and other processing tasks.

```
# creating staging table

create table dailyactivity
like dailyactivity_merged;

insert dailyactivity
select *
from dailyactivity_merged;

SELECT * FROM bellabeat.dailyactivity;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id	ActivityDate	TotalSteps	TotalDistance	TrackerDistance	LoggedActivitiesDistance	VeryActiveDistance	ModeratelyActiveDistance	LightActiveDistance	SedentaryActiveDistance	VeryActiveMinutes
1503960366	2016-04-12	13162	8.5	8.5	0	1.879999995	0.550000012	6.059999943	0	25
1503960366	2016-04-13	10735	6.96999979	6.96999979	0	1.570000052	0.689999998	4.710000038	0	21
1503960366	2016-04-14	10460	6.739999771	6.739999771	0	2.440000057	0.400000006	3.910000086	0	30
1503960366	2016-04-15	9762	6.28000021	6.28000021	0	2.140000105	1.25999999	2.829999924	0	29
1503960366	2016-04-16	12669	8.159999847	8.159999847	0	2.710000038	0.409999996	5.039999962	0	36
1503960366	2016-04-17	9705	6.480000019	6.480000019	0	3.190000057	0.779999971	2.50999999	0	38

Step 3: Data Cleaning and Removing Duplicates

First, I checked for duplicates in the three new staging tables. Duplicates were found only in the sleepday table, while the other tables did not contain any duplicates.

```
Select count(*)
From

(Select id, activitydate, totalsteps, count(*)

from dailyactivity

Group by id, activitydate, totalsteps

Having count(*)>1 ) A;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

count(*)
0

```
Select count(*)

From

(Select id, sleepday, TotalMinutesAsleep, count(*)

From sleepday

Group by id, sleepday, TotalMinutesAsleep

Having count(*)>1 ) A;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

count(*)
3

After identifying duplicates, I created a new table called sleepday_new where I included only distinct values to remove duplicates from the sleepday table.

After creating the sleepday_new table with distinct values to remove duplicates, I rechecked for duplicates and confirmed that none exist in the new table.

```sql
-- Creating new sleepday_new table to remove duplicates by adding distinct ids and records TO THE NEW TABLE.
CREATE TABLE Sleepday_new

SELECT *

FROM

    (SELECT DISTINCT  id, sleepday, totalsleeprecords, totalminutesasleep, totaltimeinbed

    FROM sleepday_merged) A;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id	sleepday	totalsleeprecords	totalminutesasleep	totaltimeinbed
1503960366	2016-04-12	1	327	346
1503960366	2016-04-13	2	384	407
1503960366	2016-04-15	1	412	442
1503960366	2016-04-16	2	340	367
1503960366	2016-04-17	1	700	712

```sql
-- Checking if there is any duplicates in sleepday_new table:

Select count(*)

From

(Select id, sleepday, TotalMinutesAsleep, count(*)

From sleepday_new

Group by id, sleepday, TotalMinutesAsleep

Having count(*)>1 ) A;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

count(*)
0

Step 4: Checking Common Users Across Tables

I verified if the same users existed in all tables. All tables had a common user count of 33, except sleepday_merged2, which had 9 users not found in the other tables.

```
with base as
(select *,
case when id1=id2 then 'COMMON USERS' else 'NON COMMON USERS' end as flag
from (select distinct id as id1 from dailyactivity) A
left join (select distinct id as id2 from sleepday_new) B
on A.id1=B.id2)
select flag, count(*) as user
from base
group by flag
order by flag;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

flag	user
COMMON USERS	24
NON COMMON USERS	9

Step 5: Data Transformation and Integration

The date columns in all tables were initially in text format. I converted these columns from string to date format by following these steps:

```
select ActivityDate,
str_to_date(ActivityDate,'%m/%d/%Y')
from dailyactivity;

SET SQL_SAFE_UPDATES = 0;

UPDATE dailyactivity
SET Activitydate = STR_TO_DATE(ActivityDate, '%m/%d/%Y');

ALTER TABLE dailyactivity
modify COLUMN ActivityDate DATE;

select *
from dailyactivity;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id	ActivityDate	TotalSteps	TotalDistance	TrackerDistance	LoggedActivitiesDistance	VeryActiveDistance	Mo
1503960366	2016-04-12	13162	8.5	8.5	0	1.879999995	0.5
1503960366	2016-04-13	10735	6.96999979	6.96999979	0	1.570000052	0.6
1503960366	2016-04-14	10460	6.739999771	6.739999771	0	2.440000057	0.4
1503960366	2016-04-15	9762	6.28000021	6.28000021	0	2.140000105	1.2

Table: dailyactivity

Columns:

Id	bigint
ActivityDate	date
TotalSteps	int
TotalDistance	double
TrackerDistance	double
LoggedActivitiesDistance	int
VeryActiveDistance	double
ModeratelyActiveDistance	double
LightActiveDistance	double
SedentaryActiveDistance	int
VeryActiveMinutes	int
FairlyActiveMinutes	int
LightlyActiveMinutes	int
SedentaryMinutes	int
Calories	int

Step 6: Creating Final Master Table

I created a final master_table by joining all three tables to consolidate the data for final use.

```sql
CREATE TABLE master_table AS
SELECT
    da.id,da.activitydate,da.TotalSteps,da.TrackerDistance,da.LoggedActivitiesDistance,da.VeryActiveDistance,da.ModeratelyActiveDistance,da.LightActiveDistance,
    da.veryactiveminutes,da.fairlyactiveminutes,da.lightlyactiveminutes,da.sedentaryminutes,da.calories,sd.totalminutesasleep,sd.totaltimeinbed,
    wi.weightKg,wi.weightpounds,wi.IsManualReport
FROM
    dailyactivity da
LEFT JOIN
    sleepday_new sd
ON da.id = sd.id AND da.activitydate = sd.sleepday
LEFT JOIN
    weightinfo wi
ON da.id = wi.id AND da.activitydate = wi.date;

select * from master_table;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id	activitydate	TotalSteps	TrackerDistance	LoggedActivitiesDistance	VeryActiveDistance	ModeratelyActiveDistance	LightActiveDistance	veryactiveminutes	fairlyactiveminutes	lightlyactiveminutes
1503960366	2016-04-12	13162	8.5	0	1.879999995	0.550000012	6.059999943	25	13	328
1503960366	2016-04-13	10735	6.96999979	0	1.570000052	0.689999998	4.710000038	21	19	217
1503960366	2016-04-14	10460	6.739999771	0	2.440000057	0.400000006	3.910000086	30	11	181
1503960366	2016-04-15	9762	6.28000021	0	2.140000105	1.25999999	2.829999924	29	34	209

Step 7: Checking for NULL Values in the Master Table

I checked for NULL values in the master_table to identify missing data. The purpose of the query below was to count the number of NULL values in each specified column, helping to determine how many records lack data in these columns.

Don't Limit

```sql
SELECT
    SUM(CASE WHEN veryactiveminutes IS NULL THEN 1 ELSE 0 END) AS veryactiveminutes_nulls,
    SUM(CASE WHEN fairlyactiveminutes IS NULL THEN 1 ELSE 0 END) AS fairlyactiveminutes_nulls,
    SUM(CASE WHEN lightlyactiveminutes IS NULL THEN 1 ELSE 0 END) AS lightlyactiveminutes_nulls,
    SUM(CASE WHEN sedentaryminutes IS NULL THEN 1 ELSE 0 END) AS sedentaryminutes_nulls,
    SUM(CASE WHEN calories IS NULL THEN 1 ELSE 0 END) AS calories_nulls,
    SUM(CASE WHEN totalminutesasleep IS NULL THEN 1 ELSE 0 END) AS totalminutesasleep_nulls,
    SUM(CASE WHEN totaltimeinbed IS NULL THEN 1 ELSE 0 END) AS totaltimeinbed_nulls,
    SUM(CASE WHEN weightKg IS NULL THEN 1 ELSE 0 END) AS weightKg_nulls,
    SUM(CASE WHEN weightpounds IS NULL THEN 1 ELSE 0 END) AS weightpounds_nulls,
    SUM(CASE WHEN ismanualreport IS NULL THEN 1 ELSE 0 END) AS ismanualreport_nulls
FROM master_table;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

veryactiveminutes_nulls	fairlyactiveminutes_nulls	lightlyactiveminutes_nulls	sedentaryminutes_nulls	calories_nulls	totalminutesasleep_nulls	totaltimeinbed_nulls	weightKg_nulls	weightpounds_nulls	ismanualreport_nu
0	0	0	0	0	0	0	0	0	0

Step 8: Replacing NULL Values with 0

The UPDATE query is designed to replace NULL values in specific columns of the master_table with 0.

```
UPDATE master_table
SET
    veryactiveminutes = IFNULL(veryactiveminutes, 0),
    fairlyactiveminutes = IFNULL(fairlyactiveminutes, 0),
    lightlyactiveminutes = IFNULL(lightlyactiveminutes, 0),
    sedentaryminutes = IFNULL(sedentaryminutes, 0),
    calories = IFNULL(calories, 0),
    totalminutesasleep = IFNULL(totalminutesasleep, 0),
    totaltimeinbed = IFNULL(totaltimeinbed, 0),
    weightKg = IFNULL(weightKg, 0),
    weightpounds = IFNULL(weightpounds, 0),
    ismanualreport = IFNULL(ismanualreport, 0);
```

Step 9: Adding Week Number and Day of Week Columns

I added week number and day of week columns to the master_table to enable visualization of the data on a weekly basis.

```
ALTER TABLE master_table
ADD COLUMN day_of_week VARCHAR(10);

UPDATE master_table
SET day_of_week = DAYNAME(activitydate);

ALTER TABLE master_table
ADD COLUMN week_number INT;

UPDATE master_table
SET week_number = WEEK(activitydate, 1); -- Mode 1 considers Monday as the first day of the week
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

fairlyactiveminutes	lightlyactiveminutes	sedentaryminutes	calories	totalminutesasleep	totaltimeinbed	weightKg	weightpounds	ismanualreport	day_of_week	week_number	BMI	ActivityLevel
13	328	728	1985	327	346	0	0	0	Tuesday	15	22.65	VERY ACTIVE
19	217	776	1797	384	407	0	0	0	Wednesday	15	22.65	MODERATELY ACTIVE
11	181	1218	1776	0	0	0	0	0	Thursday	15	22.65	MODERATELY ACTIVE
34	209	726	1745	412	442	0	0	0	Friday	15	22.65	LIGHTLY ACTIVE
10	221	773	1863	340	367	0	0	0	Saturday	15	22.65	VERY ACTIVE

Step 10: Adding User Activity Level Column

I added a User ActivityLevel column to the master_table based on the number of steps taken each day. This column categorizes users' activity levels to provide better insights into their daily activity patterns.

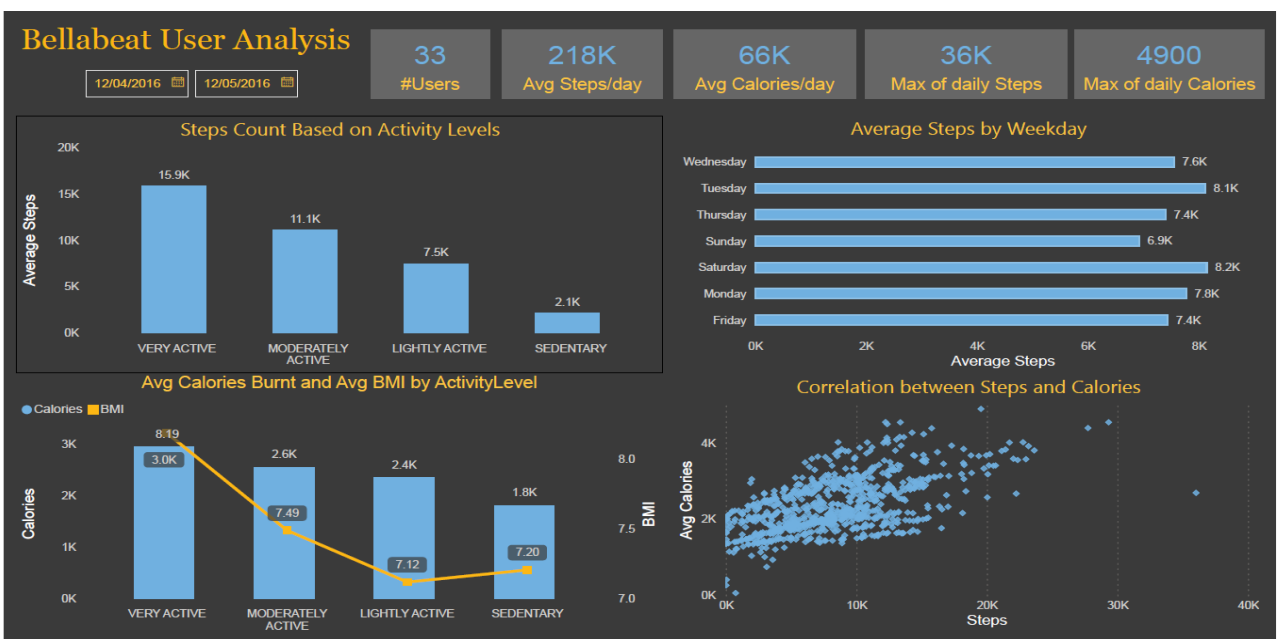
```sql
ALTER TABLE master_table ADD COLUMN ActivityLevel VARCHAR(20);
UPDATE master_table
SET ActivityLevel = CASE
    WHEN TotalSteps < 5000 THEN 'SEDENTARY'
    WHEN TotalSteps BETWEEN 5000 AND 9999 THEN 'LIGHTLY ACTIVE'
    WHEN TotalSteps BETWEEN 10000 AND 12588 THEN 'MODERATELY ACTIVE'
    ELSE 'VERY ACTIVE'
END;
SELECT * FROM master_table;
```

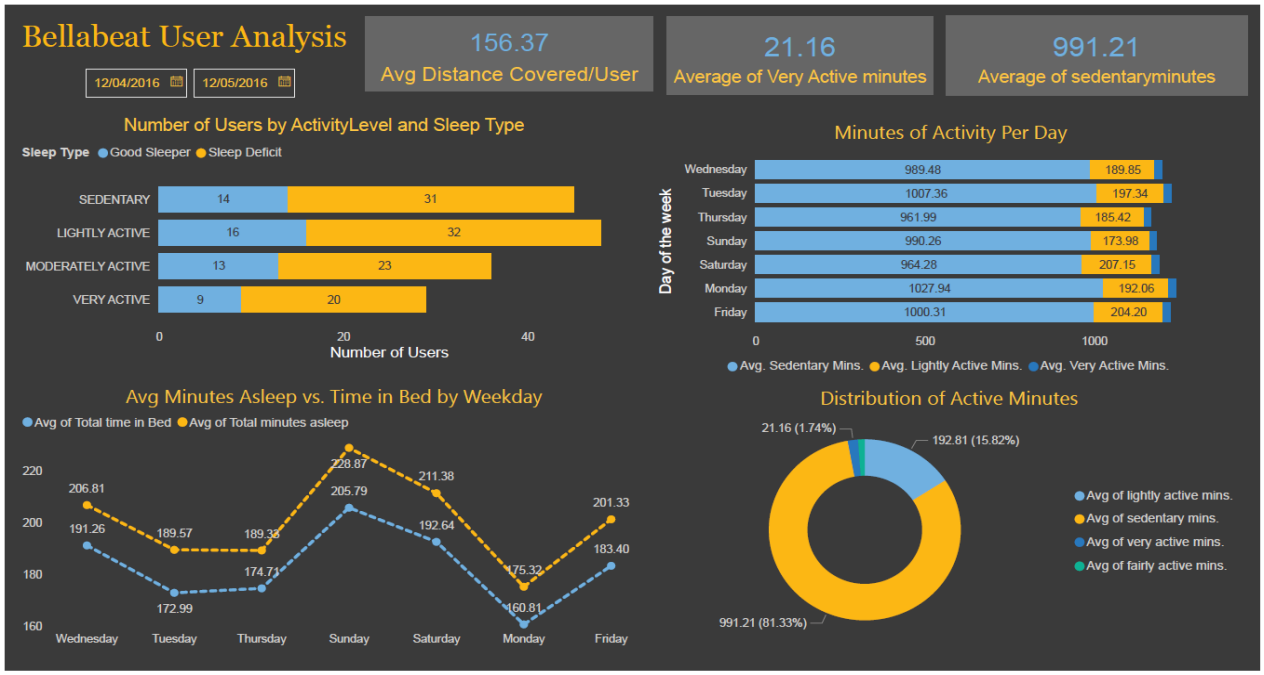
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

fairlyactiveminutes	lightlyactiveminutes	sedentaryminutes	calories	totalminutesasleep	totaltimeinbed	weightKg	weightpounds	ismanualreport	day_of_week	week_number	BMI	ActivityLevel
13	328	728	1985	327	346	0	0	0	Tuesday	15	22.65	VERY ACTIVE
19	217	776	1797	384	407	0	0	0	Wednesday	15	22.65	MODERATELY ACTIVE
11	181	1218	1776	0	0	0	0	0	Thursday	15	22.65	MODERATELY ACTIVE
34	209	726	1745	412	442	0	0	0	Friday	15	22.65	LIGHTLY ACTIVE
10	221	773	1863	340	367	0	0	0	Saturday	15	22.65	VERY ACTIVE

The final master_table was exported from MySQL as a CSV file, and a dashboard was created using Power BI for data visualization.

Phase 4: Visualization and Findings and Key Insights





Key Insights

- Sedentary Behavior: Sedentary minutes dominate most participants' days and remain consistent throughout the week.
- Sleep and Activity Patterns: On average, participants sleep the most and take the fewest steps on Sundays. Conversely, the most steps are taken on Tuesdays and Saturdays.
- Activity Levels: Very active Participants take the most steps. However, the average of active minutes is significantly lower than sedentary minutes, indicating that most participants are lightly active.
    - Average Sedentary Time: 991 minutes per day (over 16 hours), suggesting a need for reduction.
- Steps and Calories: There is a direct correlation between the number of steps taken and the calories burnt.
- Sleep Quality: Few participants are classified as good sleepers, with a higher number experiencing sleep deficits.

Phase 5: Strategic Recommendations

Founded in 2014, Bellabeat has pioneered wearables designed specifically for women, and now offers a range of digital health products. Based on the analysis of FitBit fitness data—covering activity, steps, calories burned, intensity, and sleep patterns—several key trends have emerged:

- Correlation: A positive relationship exists between the number of steps taken and calories burned, as well as between minutes slept and time spent in bed.
- Sedentary Time: The average sedentary time exceeds 16 hours per day, which is concerning.

The available data has a small sample size and lacks demographic details, which could lead to potential biases.

Recommendations:

I. **Step Notifications:** Encourage users to achieve at least 8,000 steps daily, as recommended by the CDC, by sending notifications about the health benefits of meeting this goal.

II. **Sleep Time Alerts:** Introduce a feature to notify users of their recommended sleep duration and send reminders a few minutes before bedtime to help them prepare for sleep.

III. **Reduce Sedentary Behavior:** Implement reminders or prompts for regular movement and stretching exercises to decrease prolonged periods of inactivity.

IV. **Promote Physical Activity on Sundays:** Since users tend to be less active on Sundays, encourage light physical activities or wellness challenges on this day to balance rest and activity.

V. **Encourage Consistent Sleep Patterns:** Highlight the importance of consistent and adequate sleep by incorporating sleep health tips and tracking features to promote good sleep hygiene.

VI. **Highlight Benefits of Active Minutes:** Promote the advantages of "very active minutes" through success stories, personalized fitness goals, and rewards for achieving specific activity milestones.

VII. **Emphasize Step Count Importance:** Develop initiatives to increase daily step counts, such as step challenges, virtual walking events, or integrating step tracking with other wellness apps.

VIII. **Tailor Health Interventions Based on BMI:** Use BMI data to customize health interventions and coaching. Encourage users with higher BMIs to engage in more physical activities and adopt healthier lifestyles.

IX. **Personalized Fitness Plans:** Offer personalized fitness plans and recommendations based on individual activity levels and health data, ensuring users receive tailored advice.

X. **Integration with Wearable Tech:** Expand integrations with other popular health and fitness apps to provide a more comprehensive view of users' health data and progress.

By implementing these recommendations, Bellabeat can enhance user engagement, promote healthier lifestyles, and improve overall well-being among its users.