**SLEEP TRACKING**

# ABSTRACT:

# Sleep plays a vital role in maintaining physical and mental health, yet modern lifestyles often disrupt healthy sleep patterns. This project aims to design a comprehensive sleep-tracking system that monitors sleep duration, quality, and patterns through non-intrusive methods. Utilizing sensors, wearable devices, or smartphone-based technologies, the system collects data such as movement, heart rate, and sound to provide detailed insights into sleep cycles. Advanced algorithms analyze this data to detect anomalies, identify trends, and recommend personalized strategies for improved sleep hygiene. By integrating user-friendly interfaces and actionable feedback, the solution empowers individuals to better understand and optimize their sleep, contributing to enhanced overall well-being.

# SYSTEM REQUIREMENTS:

### HARDWARE REQUIREMENTS:

* **Sensors**:
  + Accelerometer and gyroscope (motion detection).
  + Heart rate sensor and pulse oximeter (health metrics).
  + Temperature sensor (body/ambient temperature).
  + Microphone (sound/snoring detection).
* **Processing Unit**:
  + Microcontroller (e.g., Arduino, ESP32) or microprocessor (e.g., Raspberry Pi).
* **Power Supply**:
  + Rechargeable battery and power management components.
* **Communication Modules**:
  + Bluetooth or Wi-Fi for data transfer.
* **Display/Interface** (optional):
  + Small OLED/LED display or smartphone integration for user feedback.

### SOFTWARE REQUIREMENTS:

 **Data Collection**: Firmware to read sensor data and process signals.

 **Sleep Analysis**: Algorithms for sleep stage detection and health metrics.

 **App Interface**: Mobile or web app for displaying sleep data and trends.

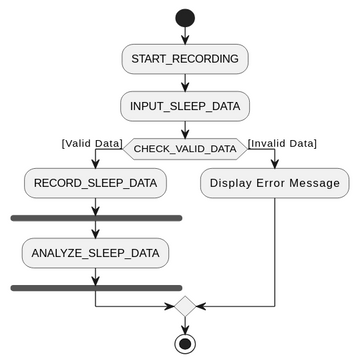
 **Data Storage**: Local (SQLite) and optional cloud storage (e.g., Firebase).

###  ****Visualization****: Graphs, charts, and feedback features for user insights.

### TOOLS AND VERSIONS:

* **C/C++**: For microcontroller firmware.
* **Java/Kotlin**: For Android apps.
* **Swift**: For iOS apps.
* **JavaScript (ES6+)**: For web-based solutions.

# FLOWCHART:



1. **CODE IMPLEMENTATION:**

Mainactivity.kt

package com.example.projectone

import android.content.Context

import android.content.Intent

import android.icu.text.SimpleDateFormat

import android.os.Bundle

import androidx.activity.ComponentActivity

import androidx.activity.compose.setContent

import androidx.compose.foundation.Image

import androidx.compose.foundation.layout.\*

import androidx.compose.material.Button

import androidx.compose.material.MaterialTheme

import androidx.compose.material.Surface

import androidx.compose.material.Text

import androidx.compose.runtime.\*

import androidx.compose.ui.Alignment

import androidx.compose.ui.Modifier

import androidx.compose.ui.draw.alpha

import androidx.compose.ui.layout.ContentScale

import androidx.compose.ui.res.painterResource

import androidx.compose.ui.unit.dp

import androidx.core.content.ContextCompat

import com.example.projectone.ui.theme.ProjectOneTheme

import java.util.\*

class MainActivity : ComponentActivity() {

private lateinit var databaseHelper: TimeLogDatabaseHelper

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

databaseHelper = TimeLogDatabaseHelper(this)

databaseHelper.deleteAllData()

setContent {

ProjectOneTheme {

// A surface container using the 'background' color from the theme

Surface(

modifier = Modifier.fillMaxSize(),

color = MaterialTheme.colors.background

) {

MyScreen(this,databaseHelper)

}

}

}

}

}

@Composable

fun MyScreen(context: Context, databaseHelper: TimeLogDatabaseHelper) {

var startTime by remember { mutableStateOf(0L) }

var elapsedTime by remember { mutableStateOf(0L) }

var isRunning by remember { mutableStateOf(false) }

val imageModifier = Modifier

Image(

painterResource(id = R.drawable.sleeptracking),

contentScale = ContentScale.FillHeight,

contentDescription = "",

modifier = imageModifier

.alpha(0.3F),

)

Column(

modifier = Modifier.fillMaxSize(),

horizontalAlignment = Alignment.CenterHorizontally,

verticalArrangement = Arrangement.Center

) {

if (!isRunning) {

Button(onClick = {

startTime = System.currentTimeMillis()

isRunning = true

}) {

Text("Start")

//databaseHelper.addTimeLog(startTime)

}

} else {

Button(onClick = {

elapsedTime = System.currentTimeMillis()

isRunning = false

}) {

Text("Stop")

databaseHelper.addTimeLog(elapsedTime,startTime)

}

}

Spacer(modifier = Modifier.height(16.dp))

Text(text = "Elapsed Time: ${formatTime(elapsedTime - startTime)}")

Spacer(modifier = Modifier.height(16.dp))

Button(onClick = { context.startActivity(

Intent(

context,

TrackActivity::class.java

)

) }) {

Text(text = "Track Sleep")

}

}

}

private fun startTrackActivity(context: Context) {

val intent = Intent(context, TrackActivity::class.java)

ContextCompat.startActivity(context, intent, null)

}

fun getCurrentDateTime(): String {

val dateFormat = SimpleDateFormat("yyyy-MM-dd HH:mm:ss", Locale.getDefault())

val currentTime = System.currentTimeMillis()

return dateFormat.format(Date(currentTime))

}

fun formatTime(timeInMillis: Long): String {

val hours = (timeInMillis / (1000 \* 60 \* 60)) % 24

val minutes = (timeInMillis / (1000 \* 60)) % 60

val seconds = (timeInMillis / 1000) % 60

return String.format("%02d:%02d:%02d", hours, minutes, seconds)

}

# PROJECT HURDLES:

# Ensuring sensor accuracy and minimizing noise interference in data collection.

# Managing power consumption to optimize battery life for portable devices.

# Simplifying algorithms for efficient sleep stage detection and data processing.

# Protecting user privacy and securely storing sensitive health data.

# Designing a user-friendly interface to boost engagement and usability.

# Controlling hardware costs while maintaining quality.

# Ensuring scalability for future features or improvements.

# OUTPUT:

# FIG: Register and login page

# 

# 

# CONCLUSION:

Sleep tracking is an essential tool for improving sleep health and overall well-being. This project demonstrates how sensor technology, data processing, and user-friendly design can effectively monitor and analyze sleep patterns. By addressing challenges like data accuracy, power efficiency, and user engagement, the system empowers users to make informed decisions about their sleep hygiene. With further advancements, such solutions have the potential to revolutionize sleep monitoring, offering affordable and accessible ways to enhance quality of life.

# FUTURE SCOPE:

1. **Advanced Sleep Analysis**: Incorporating machine learning to detect complex sleep disorders like sleep apnea and insomnia.
2. **Integration with Smart Devices**: Seamless connectivity with smart home ecosystems to optimize sleep environments (e.g., lighting, temperature).
3. **Enhanced Wearables**: Development of lightweight, non-invasive devices for continuous monitoring.
4. **Cloud-Based Insights**: Leveraging cloud computing for detailed trend analysis and personalized recommendations.
5. **Health Ecosystem Expansion**: Integration with broader health platforms to link sleep data with overall health metrics.
6. **Real-Time Feedback**: Providing instant insights and actionable advice during sleep preparation phases.
7. **Global Accessibility**: Making the technology more affordable and adaptable for diverse user groups worldwide.Top of FormBottom of Form