```
//Libraries for SD card
#include "WiFi.h"
#include "FS.h"
#include "SD.h"
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>


// Define deep sleep options
uint64_t uS_TO_S_FACTOR = 1000000;  // Conversion factor for micro seconds to seconds
// Sleep for 10 minutes = 600 seconds
uint64_t TIME_TO_SLEEP = 600;

// Replace with your network credentials
const char* ssid     = "esp32";
const char* password = "asdfg1234";

// Define CS pin for the SD card module
#define SD_CS 5
#define SD_MOSI 23
#define SD_MISO 19
#define SD_SCK 18


#define SEALEVELPRESSURE_HPA (1013.25)

Adafruit_BME280 bme; // I2C
//Adafruit_BME280 bme(BME_CS); // hardware SPI
//Adafruit_BME280 bme(BME_CS, BME_MOSI, BME_MISO, BME_SCK); // software SPI


String dataMessage="";
float tempMessage;

void setup() {
  // Start serial communication for debugging purposes
  Serial.begin(115200);

  // Connect to Wi-Fi network with SSID and password
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.softAP(ssid, password);

  IPAddress IP = WiFi.softAPIP();
  Serial.print("AP IP address: ");
  Serial.println(IP);
```

```
Serial.println("");
Serial.println("WiFi connected.");


// Initialize SD card
Serial.println("Initializing SD card...");
SD.begin(SD_CS);   //SD_CS,SD_MOSI,SD_MISO,SD_SCK || chipSel, mosi, miso, sck
//if(!SD.begin(SD_CS)) {
 // Serial.println("Card Mount Failed");
 // return;
 // }
uint8_t cardType = SD.cardType();
if(cardType == CARD_NONE) {
  Serial.println("No SD card attached");
  return;
}

Serial.println(F("BME280 test"));
bool status;

// default settings
// (you can also pass in a Wire library object like &Wire2)
status = bme.begin();
if (!status) {
  Serial.println("Could not find a valid BME280 sensor, check wiring!");
  while (1);
}


// If the data.txt file doesn't exist
// Create a file on the SD card and write the data labels
File file = SD.open("/data.txt");
if(!file) {
  Serial.println("File doens't exist");
  Serial.println("Creating file...");
  writeFile(SD, "/data.txt", "Reading ID, Date, Hour, Temperature \r\n");
}
else {
  Serial.println("File already exists");
}
file.close();

// Enable Timer wake_up
esp_sleep_enable_timer_wakeup(TIME_TO_SLEEP * uS_TO_S_FACTOR);

logSDCard();


// Start deep sleep
```

```
  Serial.println("DONE! Going to sleep now.");
  esp_deep_sleep_start();
}

void loop() {
  // The ESP32 will be in deep sleep
  // it never reaches the loop()
}


// Write the sensor readings on the SD card
void logSDCard() {
  dataMessage = "sd card test from rudra..";
  Serial.print("Save data: ");
  Serial.println(dataMessage);
  appendFile(SD, "/data.txt", dataMessage.c_str());
  appendFile(SD, "/data.txt", "BME: Temp. Value");
  tempMessage=bme.readTemperature();
  //appendFile(SD, "/data.txt",tempMessage);

}

// Write to the SD card (DON'T MODIFY THIS FUNCTION)
void writeFile(fs::FS &fs, const char * path, const char * message) {
  Serial.printf("Writing file: %s\n", path);

  File file = fs.open(path, FILE_WRITE);
  if(!file) {
    Serial.println("Failed to open file for writing");
    return;
  }
  if(file.print(message)) {
    Serial.println("File written");
  } else {
    Serial.println("Write failed");
  }
  file.close();
}

// Append data to the SD card (DON'T MODIFY THIS FUNCTION)
void appendFile(fs::FS &fs, const char * path, const char * message) {
  Serial.printf("Appending to file: %s\n", path);

  File file = fs.open(path, FILE_APPEND);
  if(!file) {
    Serial.println("Failed to open file for appending");
    return;
  }
  if(file.print(message)) {
```

```
    Serial.println("Message appended");
  } else {
    Serial.println("Append failed");
  }
  file.close();
}
```