MINERAL DETECTION IN THE HUMAN BODY USING SALIVA THROUGH ELECTROCHEMICAL ANALYSIS

AIM:

This project introduces a non-invasive method for detecting minerals in the human body using saliva and electrochemical sensing technology. Traditional mineral testing methods, such as blood tests and urine analysis, are often invasive, time-consuming, and require laboratory facilities. To address these challenges, this system leverages potential difference measurements to identify and quantify essential minerals based on their unique electrical properties. The detected data is processed using an evaluation algorithm and presented to users through a interface, offering real-time insights into their mineral levels. The system operates in a three-stage process: first, the user provides a saliva sample, which is analyzed using an electrochemical sensor to detect voltage and conductivity variations corresponding to different minerals. The collected data is then processed and compared against predefined standard values to determine any deficiencies or excess levels. Finally, the results are displayed on a user-friendly interface, where users can track their mineral status, receive health insights, and get recommendations on dietary adjustments. By integrating advanced electrochemical sensing, real-time data analysis, and digital health monitoring, this project provides a fast, cost-effective, and accessible alternative to traditional mineral testing methods. It enables individuals to monitor their nutritional health conveniently and make informed decisions regarding their diet and supplementation. This innovation has the potential to revolutionize preventive healthcare and personalized nutrition management, making mineral analysis more efficient and widely accessible.

Objective of Project:

This project aims to enhance diagnostic efficiency by utilizing biosensors and potential difference measurement techniques to accurately identify mineral compositions. The collected data is then processed using advanced computational techniques, ensuring precise and reliable results. By integrating this technology with a cloud-based interface, the system enables real-time monitoring and personalized health tracking, allowing users to make informed dietary and medical decisions. Furthermore, the project seeks to bridge the gap between preventive healthcare and technology by providing a tool that can be widely used for early deficiency detection, nutrition planning, and overall wellness management. The long-term vision is to make mineral testing more accessible, scalable, and adaptable for both personal and clinical applications, contributing to the advancement of digital healthcare solutions.

Abstract:

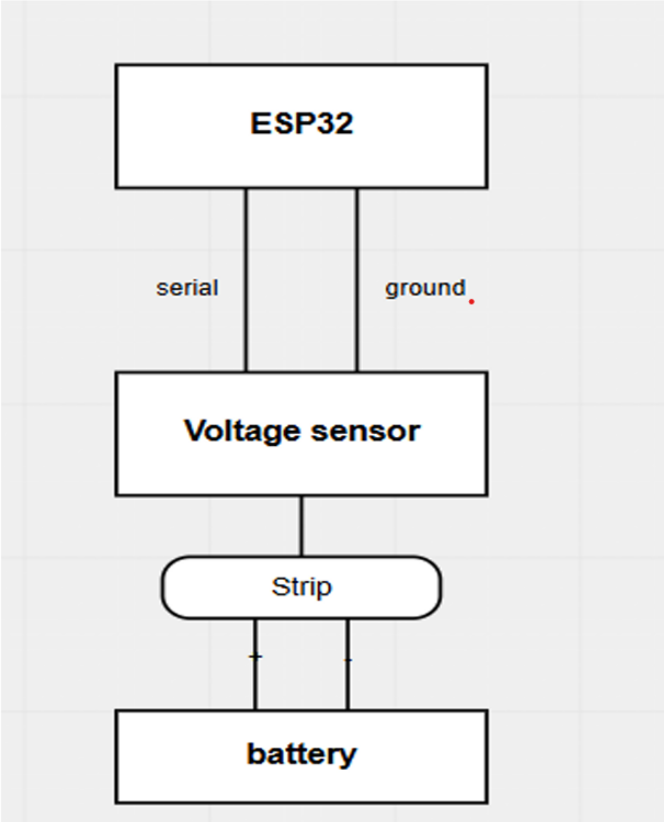
Minerals play a vital role in maintaining human health, and their deficiency or excess can lead to various medical conditions.

This project proposes a non-invasive method to detect the presence and concentration of essential minerals in the human body using saliva analysis.

The system utilizes the principle of potential difference (voltage variation) to identify and quantify minerals based on their electrical properties.

By leveraging biosensors and a microcontroller-based analysis system, the proposed solution provides a quick and cost-effective alternative to traditional blood tests for mineral detection.

Architecture Diagram:



Hardware Requirement:

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It should what the system do and not how it should be implemented. The minimum requirements to demonstrate the solution are

- Esp32
- Voltage sensor
- Battery
- Custom Slits
- Wires

Hardware Components and Their Roles

1. ESP32

- Acts as the microcontroller unit (MCU) for processing data.
- Handles sensor readings, data transmission, and wireless communication.

2. Voltage Sensor

• Measures potential differences in saliva samples to detect mineral concentrations.

• Converts electrical signals into readable data for the ESP32.

Ensures high accuracy in detecting variations in conductivity based on mineral presence.

3. Battery

• Provides power supply to the ESP32 and other hardware components.

• Ensures the system is portable and operates without external power sources.

• Supports long-duration usage with rechargeable and energy-efficient properties.

4. Custom Sensor Module

• Serves as the interface between saliva and electronic components.

• Features optimized geometry for consistent sample collection.

• Incorporates electrode pairs with specific spacing for measurement accuracy

5. Wires and Connectors

• Facilitate electrical connections between sensors, the ESP32, and the power source.

• Ensure stable and efficient signal transmission for accurate data collection.

• Provide flexibility in circuit design for better system integration.

Software Requirement:

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the teams and tracking the team's progress throughout the development activity

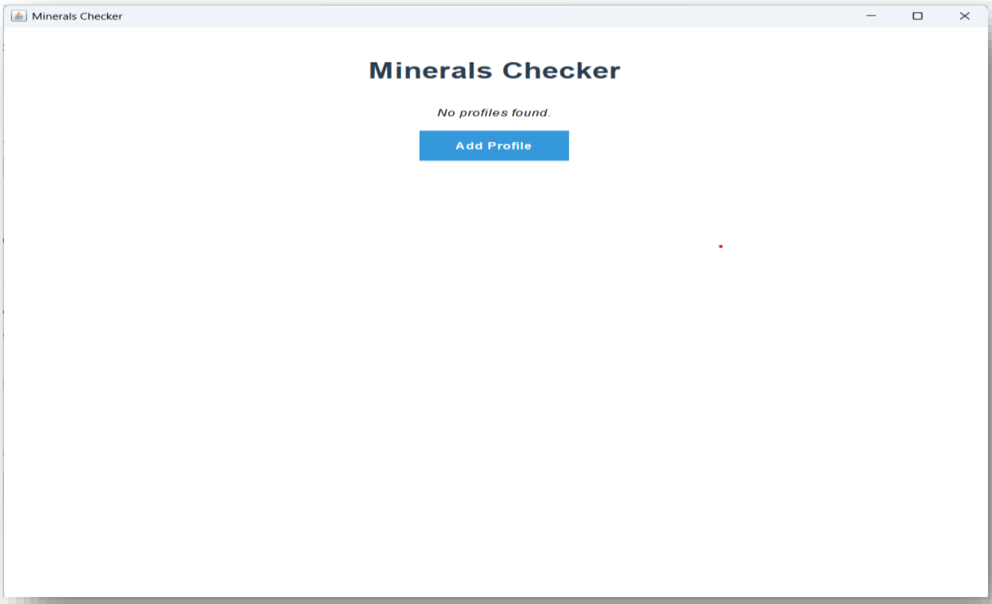
Operating System: Windows 7, 8, 10, or 11 (compatible with required development tools).

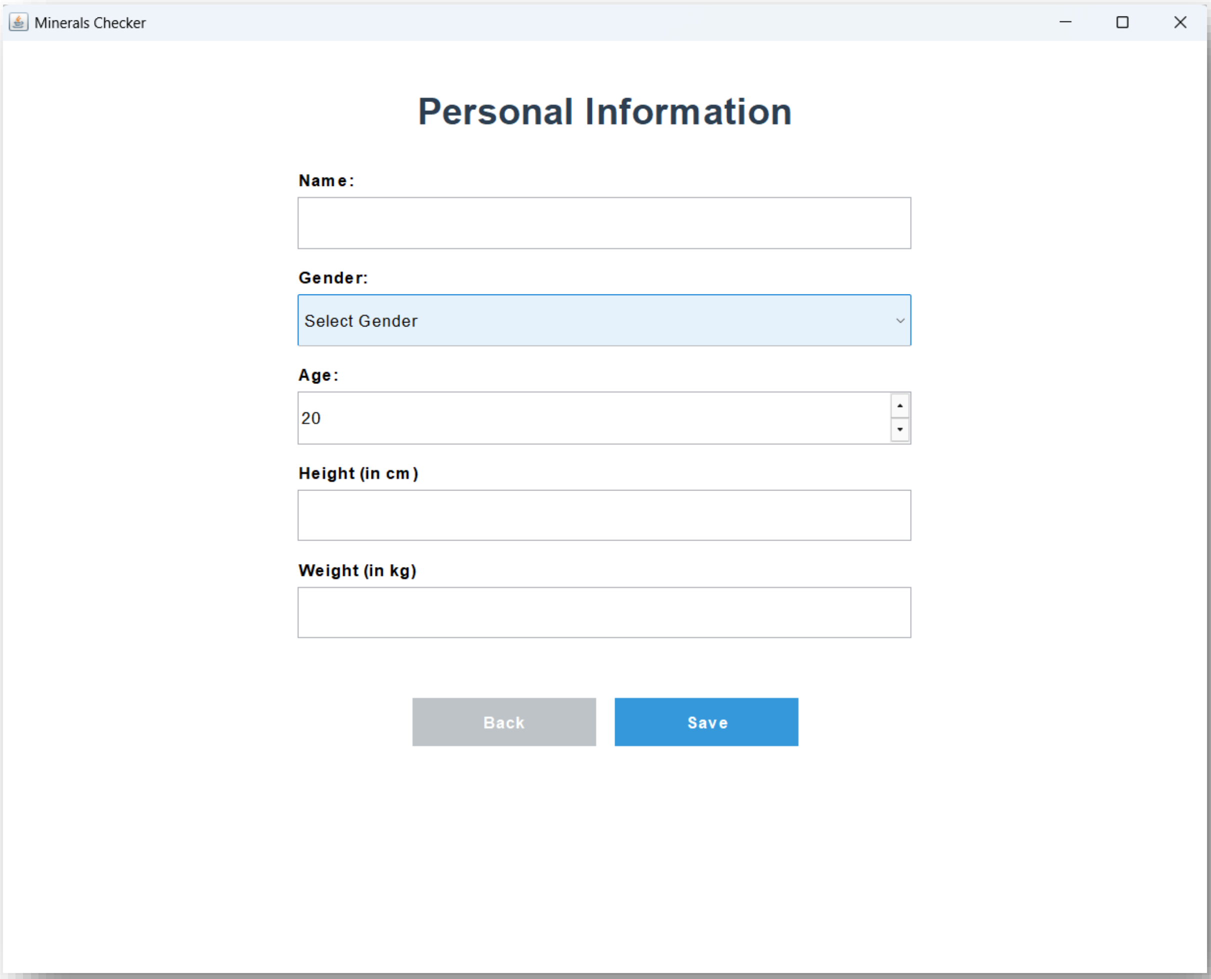
Additional Requirement:

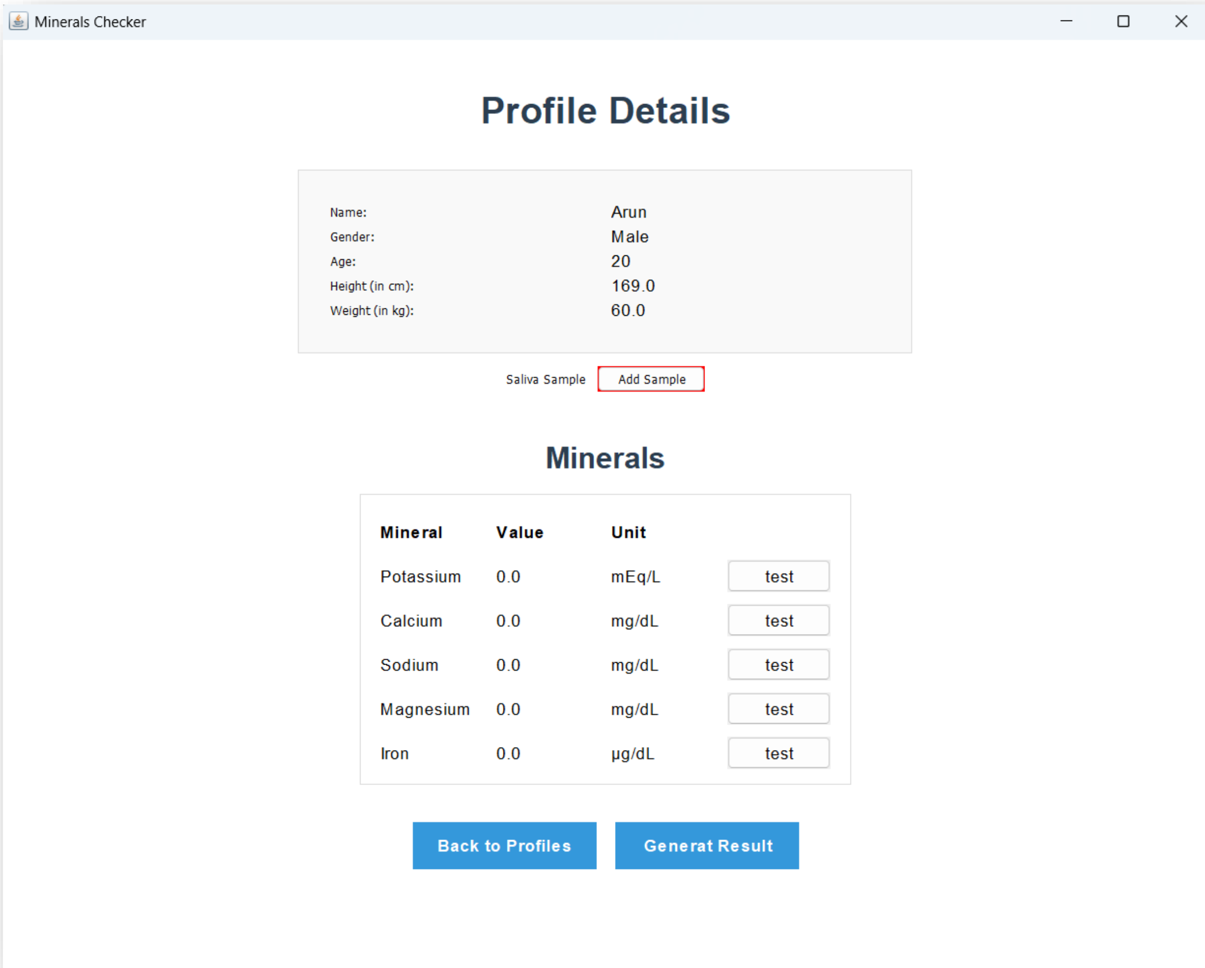
Chemicals:

1. Boric Acid: A compound in aqueous solution used to react with saliva for calculating potassium level in blood.
2. Hydrogen Peroxide: A compound in aqueous solution used to react with saliva for calculating calcium level in blood.

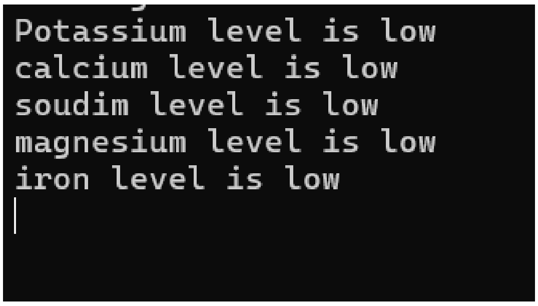
Input:


Minerals Checker
Minerals Checker
No profiles found.
Add Profile


Minerals Checker
Personal Information
Name:
Gender:
Select Gender
Age:
20
Height (in cm)
Weight (in kg)
Back
Save


Minerals Checker
Profile Details
Name:
Arun
Gender:
Male
Age:
20
Height (in cm):
169.0
Weight (in kg):
60.0
Saliva Sample
Add Sample
Minerals
Mineral
Value
Unit
Potassium
0.0
mEq/L
test
Calcium
0.0
mg/dL
test
Sodium
0.0
mg/dL
test
Magnesium
0.0
mg/dL
test
Iron
0.0
µg/dL
test
Back to Profiles
Generat Result

Output:


Potassium level is low
calcium level is low
soudim level is low
magnesium level is low
iron level is low

Application Development Phase:

Frontend Development: Java (Swing & AWT)

In this project, Java Swing and AWT (Abstract Window Toolkit) are used for developing the graphical user interface (GUI), providing an interactive and user-friendly experience.

- **User Interface (UI) Design**: Swing components like JFrame, JPanel, JButton, JTextField, and JTable are used to build an intuitive interface.
- **Data Visualization**: JLabel and JTextArea display mineral analysis results, while charts or graphs can be integrated using Java libraries.
- **User Input Handling**: Swing and AWT handle user interactions like entering data, selecting options, and navigating between screens.
- **Cross-Platform Compatibility**: The Java-based UI ensures the application runs on various Windows versions (7-11) without dependency issues.

Using Swing and AWT, the frontend is designed to be responsive, lightweight, and efficient, ensuring smooth operation and seamless user interaction.

Embedded Development: C++ for IoT (ESP32)

In this project, C++ is used for programming the ESP32 microcontroller, enabling communication between electrochemical sensors and the software system.

- **Sensor Data Acquisition**: Reads electrical signals from the voltage sensor to measure potential differences in the saliva sample.
- **Data Processing**: Converts raw sensor data into meaningful values representing mineral concentrations.

- **Microcontroller Control**: Manages ESP32 operations, including sensor calibration, data transmission, and power management.
- **Wireless Communication**:
  - Uses Wi-Fi/Bluetooth to send processed data to the Java-based frontend for real-time monitoring.
- **Real-Time Processing & Alerts**: Implements logic to detect mineral imbalances and trigger notifications or alerts when necessary.

C++ ensures efficient execution, real-time performance, and low power consumption, making it ideal for IoT-based mineral detection and monitoring.

main.cpp

C: > Users > ramch > OneDrive > Desktop > batch 10 project files > Source code > device code > main.cpp

```cpp
const int VoltPin=32; // Pin to read voltage

// Intialize WiFi
void Start_Wifi(){
  WiFi.mode(WIFI_AP);
  WiFi.softAP(ssid, password, 1 , 0, 1);
  Serial.println("WiFi AP started");
}

// Connect to the pplication
void AppConnect(){
  Serial.println("Waiting for application to connect...");
  while(app_client.connected() == false)
  {
    app_client.connect("192.168.4.2", 80);
  }

  if(app_client.connected())
  {
    Serial.println("Application conected");
  }
  else
  {
    Serial.println("Application not connected");
    app_client.stop();
  }
}

// Function to read volatage
int getVolt() {
  return analogRead(VoltPin);
}

// Setup (first function to run)
void setup(){
  Serial.begin(115200);
  pinMode(VoltPin,OUTPUT);
  Start_Wifi();
}

// Called repeatedly after setup
void loop(){
  AppConnect();
```

```java
public class App extends JFrame {

    public App() {
        setTitle("Minerals Checker");
        setSize(1000, 800);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        // Initialize components
        cardLayout = new CardLayout();
        mainPanel = new JPanel(cardLayout);
        mainPanel.setBackground(Color.WHITE);
        profiles = new ArrayList<>();

        // Load existing profiles
        DataManager.loadProfiles(profiles);
        JavaServer server = new JavaServer();

        // Create panels
        WelcomePanel welcomePanel = new WelcomePanel(cardLayout, mainPanel, profiles, server);
        PersonalInfoPanel personalInfoPanel = new PersonalInfoPanel(cardLayout, mainPanel, welcomePanel, profiles);

        // Add panels to main panel
        mainPanel.add(welcomePanel, "welcome");
        mainPanel.add(personalInfoPanel, "personalInfo");

        // Add main panel to frame
        add(mainPanel);

        // Show first card
        cardLayout.show(mainPanel, "welcome");

        // Close server listener
        this.addWindowListener(new WindowListener() {
            @Override
            public void windowOpened(WindowEvent e) {}

            @Override
            public void windowClosing(WindowEvent e) {
                server.stop();
            }

            @Override
```

Testing:

S.No	Test Case ID	Test Case Description	Module Name	Condition	Expected Output	Obtained Output	Status
1	TC_001	Saliva sample detection	Sample Collection	Valid input	Saliva detected	Saliva detected	Pass
2	TC_002	Voltage sensing accuracy	Electrochemical Sensing	Correct sample applied	Voltage detected	Voltage detected	Pass
3	TC_003	Mineral composition analysis	Data Processing	Correct input signals	Correct mineral levels displayed	Correct mineral levels displayed	Pass
4	TC_004	Output display on user interface	Web Interface	Data received	Results displayed on GUI	Results displayed on GUI	Pass
5	TC_005	Alert for mineral deficiency	Web Interface	Low mineral level detected	Warning message displayed	Warning message displayed	Pass
6	TC_006	Cloud data storage	Web Interface	Internet connected	Data successfully stored in the cloud	Data successfully stored in the cloud	Pass
7	TC_007	Sensor calibration accuracy	Electrochemical Sensing	Predefined test input applied	Expected voltage range detected	Expected voltage range detected	Pass
8	TC_008	User authentication	Web Interface	Valid login credentials entered	User successfully logged in	User successfully logged in	Pass
9	TC_009	System response time	Data Processing	Input given within normal conditions	Results displayed within 5 sec	Results displayed within 5 sec	Pass
10	TC_010	Power management test	Hardware Module	Device powered on	System starts without issues	System starts without issues	Pass

Module Design:

- C++ for device
- Java for Application

Credentials (for testing)

WiFi Access Point SSID: ESP32-MC-DEVICE

Password: ESP32-1234