תיעוד מלא ESP32 – מערכת רישום מתח 🔸

סקירה כללית 🗐

מערכת מתקדמת לרישום נתוני מתח בעזרת ESP32, מערכת

- שליחת נתונים לשרת FastAPI
 - PostgreSQL שמירתם במסד
 - ניתוח וחישוב ממוצעים יומיים
 - מעקב אחר טריגרים מהשטח •

קישורים חשובים 🌐

כתובות שרת:

- : https://esp32-voltage-logger.onrender.com שרת ראשי
 - Swagger: /docs
 - ReDoc: /redoc •

(PostgreSQL):מסד נתונים

- URL:
- postgresql://postgres_voltage_logger:QhQvlgtV2svJj5CwmBKxDthLfS5GzUxY@dpg-d1n4p8ndiees73eoc7h0-a.oregon-postgres.render.com/voltage_logger
 - : Renderשירות
 - : Oregon, USA אזור

ארכיטקטורה 🔀

חלק 1: בקר(++) ESP32

מבנה תיקיות:

esp32/	
include/	
├— config.h, wifi_manager.h,	
├— main.cpp, http_client.cpp,	
└─ platformio.ini	
TRIGGER_PIN_START = 15	
TRIGGER PIN EVENT = 4	

WIFI_RESET_PIN = 16

VOLTAGE_PIN = 34

 $LED_PIN = 2$

SAMPLING_DELAY_MS = 1000

SAMPLING_START_DELAY_MS = 24000

TOTAL_TIMER_DURATION_MS = 30000

MAX_SAMPLES = 6

הגדרות זמנים: 📀

🕴 הגדרות פינים:

מהליך עבודה:

- 1. אתחול רכיבים
 - 2. קבלת טריגר
- 3. התחלת טיימר
- 4. דגימה כל שנייה (אחרי 24 שניות)
- 5. שליחת הנתון הגבוה ביותר לשרת
- 6. שליחת אירועים מיידיים בטריגר נוסף

חלק 2: שרת(Python) חלק

מבנה תיקיות:

בקודות קצה 📝 API –

תיאור	נתיב	שיטה תחום
שליחת נתון	/api/v1/voltage/	POST מתח
כל הנתונים	/api/v1/voltage/	GET
לפי מזהה	/api/v1/voltage/{id}	GET
האחרון	/api/v1/voltage/latest/	GET
שליחת אירוע	/api/v1/event/	אירועים POST
כל האירועים	/api/v1/event/	GET
לפי מזהה	/api/v1/event/{id}	GET
רשימת זמנים	/api/v1/event/times/	GET
כל הממוצעים	/api/v1/daily-average/	GET ממוצעיו
ממוצע אחרון	/api/v1/daily- average/latest/	GET

תיאור	נתיב	שיטה תחום
חישוב ידני	/api/v1/daily- average/calculate/	POST
בדיקת חיבור	/	כללי GET
בדיקת בריאות	/health	GET
Swagger	/docs	GET
ReDoc	/redoc	GET

דוגמאותJSON

```
שליחת נתון מתח:
{
"timestamp": "2025-01-06T12:34:56",
"max_voltage": 3.45
}
                                                                                  שליחת אירוע: 👲
{
"event_time": "2025-01-06T12:34:56"
}
                                                                                        מגובה:
{
"status": "success",
"message": "Voltage data saved successfully",
"data": {
 "id": 1,
 "timestamp": "2025-01-06T12:34:56",
 "max_voltage": 3.45,
 "created_at": "2025-01-06T12:34:56"
}
}
```

התקנה והגדרה 🕸

דרישות:

- Python 3.11+ •
- Docker & Docker Compose
 - PlatformIO •
 - ESP32 לוח

ESP32:-ל WiFi 📶

- 1. חבר את הבקר לחשמל
- 2. התחבר ל ESP32_VoltageLogge
 - 3. סיסמה: 12345678
 - 4. דף הגדרה ייפתח אוטומטית

.env: קובץ 🔐

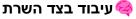
DATABASE_URL=postgresql://...

APP_ENVIRONMENT=production

APP_DEBUG=false

זרימת נתונים 🔄

- 1. טריגר
- 2. דגימה
- 3. חישוב
- 4. שליחה
- 5. שמירה במסד



JSON אימות \leftarrow שמירה \rightarrow תגובה ב-

ממוצע יומי 🚃

- תזמון יומי: 00:01
 - חישוב ממוצע •
- שמירה בטבלה נפרדת

ביצועים ואבטחה 📈

ביצועים: 🔸

- דגימה כל שנייה לאחר 24 שניות מהטריגר, שליחה לאחר 30 שניות של ערך המקסימום.
 - שרת תומך במאות קריאות לדקה.
 - אינדוקס על שדות חשובים במסד.

אבטחה:

- CORS בקרת גישה עם
- טיפול בשגיאות ולוגים

פתרון בעיות 🐛

† ESP32:

- לא מתחבר \leftarrow בדוק רשת וסיסמה WiFi
 - 34 אין דגימה → בדוק פין \bullet
 - אין שליחה \leftarrow ודא כתובת שרת תקינה •

:שרת

- DATABASE_URL לא שומר למסדightarrow בדוק הגדרת
 - שגיאות → API בדוק לוגים •
 - middleware בדוק CORS → בעיות

תמיכה 📞

- מפתח: דניאל שמש 🕯 🔸
- ds5121h@gmail.com : אימייל
 - github.com/Ds8200 :GitHub

עודכן לאחרונה: 15/07/2025