**A Web App Using ESP32 and Current Sensors for Water Flow and Power Forecasting**

An

Application Development Project

Presented to the Faculty of

**Mindoro State University Calapan City Campus**

Masipit, Calapan City

Oriental Mindoro

In Partial Fulfillment

of the Requirements for the Degree

Bachelor of Science in Information Technology

by:

**Aspi, John Rene**

**Empenio, Ella V.**

**Pineda, Kier Paolo**

**Young, Ken Mark G.**

**TABLE OF CONTENTS**

**LIST OF FIGURES**

**LIST OF TABLES**

**LIST OF APPENDICES**

**CHAPTER I**

**INTRODUCTION**

**Project Context**

**Introduction**

There has been a significant increase toward creating creative approaches that integrate smart technologies with sustainable energy sources as the need for renewable energy keeps rising. Because of its efficiency and dependability, hydropower has gained attention as an essential factor in supplying energy demands in recent years (Alnaqbi et al., 2022). Particularly in areas where renewable energy is prioritized, hydropower together with modern technology like the Internet of Things (IoT) presents new options to optimize energy production and consumption (Kougias et al., 2019).

Monitoring and controlling energy use with sensors based on the ESP32 is one of the most innovative and innovative concepts in this industry. These sensors have the potential to be extremely important in gathering data in real time, enabling less expensive energy use, and enhancing the overall efficiency of hydropower systems (Hussien et al., 2021). A low-cost microcontroller with integrated Bluetooth and Wi-Fi, the ESP32 is ideal for Internet of Things applications involving energy monitoring (Macheso & Thotho, 2022). We can effectively store electricity and optimize power use using advanced analytics by connecting it with hydropower systems.

Although hydropower and other renewable energy sources have great promise, many systems lack advanced analytics and real-time monitoring, which reduces system efficiency (Bhatti, 2021). Effective management and analysis of energy consumption is further limited by the absence of connection between power generation systems and IoT platforms (Shafi et al., 2021). Managing energy storage and consumption becomes difficult in the absence of reliable information, particularly in rural areas where hydropower is frequently the main energy source (Kuriqi et al., 2021).

Management and monitoring of renewable energy sources is

a problem that is addressed by many of present methods. For example, real-time tracking of energy efficiency and consumption has been made possible by IoT-based devices (Caneda & Calimpusan, 2022). By using sensors to track electrical factors, these systems give clients a thorough understanding of how much energy they are using. Nevertheless, a lot of these systems aren't designed with certain energy storage uses in mind, including pumped hydro energy storage (PHES), which has been recognized as a vital tool for regulating energy consumption (Alnaqbi et al., 2022).

The proposed method will make use of the most recent developments in energy analytics and IoT technologies, utilizing the information obtained from the ESP32 sensor to direct energy storage plans. The system will also work to maximize the storage capacity of hydro energy systems and minimize energy waste, making sure that extra energy produced during peak hours is saved and used during times of high demand (Amirul et al., 2022).

In order to enhance energy storage and provide real-time analytics, this project will create an integrated hydropower energy management system that incorporates the ESP32 current sensor with IoT technologies. With the extra advantage of data-driven insights, it will offer a more effective way to store and use electricity by filling in the gaps in energy monitoring and management.

**Objectives**

General Objectives: To develop a web application that monitors and manages water flow rates and power storage using ESP32 and current sensors, providing real-time data to optimize resource usage and battery performance.

Specific Objectives:

Develop a Communication System Between the ESP32 and the Web App:

Implement a reliable connection between the ESP32 microcontroller and the web app to collect and display real-time data on water flow and battery storage.

Monitor Battery Charge Levels in Real-time:

Enable the app to display the current battery charge, and estimate how long the remaining charge will last be based on current power consumption.

Display Water Flow Rate Data:

Show real-time water flow rate data gathered from current sensors to give users insights into system performance.

Estimate Battery Charging Time:

Provide an estimate of how long it will take to fully charge the battery based on the current charging rate.

Send Notifications and Alerts:

Notify users when the battery is low or when water flow rates are outside the expected range, helping them take timely action.

**Scope and Limitations**

The scope of this project includes creating an Internet of Things (IoT)-based energy management system that uses the ESP32 current sensor to monitor and store hydropower electricity in real time. A web-based analytics dashboard that presents information on energy output, consumption, and storage is one of the primary features. Other features include real-time tracking of energy flow and energy storage optimization via hydro energy storage systems. The ESP32 microcontroller will provide seamless data collection and transmission, enabling remote access and integration of IoT functions to boost energy management efficiency.

Although the system has several of unique features, there are also significant drawbacks to consider. Firstly, the project's energy source is limited to hydropower which means without major adjustments, the system is unable to work with other renewable energy sources, including solar or wind. Furthermore, the processing capability of the ESP32 microcontroller is limited, which can limit the complexity of data analysis and storage capabilities. Finally, without additional research, the system's scalability to alternative energy storage methods is limited because its energy storage optimization is limited to a hydro energy storage setup.

**Definition of Terms**

ESP32 - A low-cost, low-power system on a chip (SoC) with integrated Wi-Fi and Bluetooth. It acts as the main microcontroller in this project to gather data from the sensors and communicate with the web app.

Current Sensor - A sensor used to detect and measure the flow of electric current, either in water flow systems or in electrical circuits for power management.

Power Storage Management - Refers to the process of monitoring and managing the charge and discharge of energy storage devices, such as batteries.

Water Flow Rate - The amount of water passing through a system, measured in units such as liters per second. It is monitored by current sensors in this project to provide information about system performance.

Battery Charge Level - The current amount of stored energy in a battery, expressed as a percentage. It indicates how much power is left for use before the battery needs recharging.

Battery Duration - The estimated time the battery can continue supplying power based on current usage patterns.

Real-time Monitoring - The continuous tracking and display of data as it happens, allowing the user to see live updates about battery levels and water flow rates through the web app.

**CHAPTER II**

**REQUIREMENTS SPECIFICATION**

**Hardware and Software Requirements**

|  |  |  |
| --- | --- | --- |
| **Hardware Component** | **Type** | **Specification** |
| Current Sensor | Sensor | ACS712 30A, Measures current up to 30A with high precision, suitable for real-time monitoring. |
| ESP32 Microcontroller | Microcontroller | ESP32-WROOM-32, 32-bit dual-core, 240MHz, Wi-Fi + Bluetooth enabled, 4MB Flash. |
| Mini Hydro 12V | Power Generator | DC Permanent Magnet Generator, generates 12V from water flow to power the system. |
| |  | | --- | | Lithium-Ion Battery |  |  | | --- | |  | | Battery | 18650, 3.7V, 2200mAh rechargeable battery for stable power storage and long operation. |
| Charging Module | Power module | TP4056, provides overcharge protection while charging lithium-ion batteries. |
| Battery Holder | Power Accessory | 18650 Battery Holder, Insulated holder for a single 18650 battery, secure fit. |

|  |  |  |
| --- | --- | --- |
| Step-up Converter | Power Module | DC-DC Boost Converter, Steps up voltage from 3.7V to 12V to power ESP32 and components. |
| USB Module | Interface Module | Standard wire gauge for efficient current flow and secure connections. |
| Wire | |  | | --- | | Electrical Wiring | | Standard wire gauge for efficient current flow and secure connections. |
| LCD 3.2-inch | Display Module | TFT LCD 480x320, Provides a clear, high-resolution display of voltage, current, and power data. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Software Component** | **Type** | **Minimum Specification** | **Recommended Specification** |
| Node.js | Backend Framework | |  | | --- | | Version 12.0 or higher | | |  | | --- | | Version 12.0 or higher | |
| Express.js | |  | | --- | | Web Framework | | |  | | --- | | Web Framework | | |  | | --- | | Web Framework | |

**Functional Requirements**

1. Real-time Data Transmission:
   * The system must collect voltage and current data from the ESP32 and send it via a POST request to a Node.js API.
   * The interface must display live updates of voltage and current by fetching real-time data from the backend API
2. Data Logging and Retrieval:
   * Voltage and current data must be logged in a database (if applicable) for analysis over time.
   * Users should be able to retrieve historical data through the web app.
3. Graph Representation and Visualization:
   * The web app must provide graphical representations of real-time and historical data for voltage and current.
   * Users can view data in various timeframes (seconds, minutes, hours, days, weeks, months, years).
4. Time Series Forecasting:
   * The system will analyze historical data and use time series forecasting to predict future voltage and power usage.
   * Forecast results should be visualized on the web app using graphs.
5. API Functionality:
   * A POST API must be created in Node.js to handle incoming data from the ESP32.
   * The web app should send GET requests to fetch real-time and historical data from the backend.
6. User Interface:
   * The Vue.js-based web app should offer a clean, intuitive interface for users to view real-time data, historical trends, and forecasts.

**Non-Functional Requirements**

Operational Requirements:

* The system must continuously monitor and transmit data without downtime, allowing 24/7 access to the web app.
* The API should handle data requests efficiently and be scalable for potential future expansions.
* The web app should be optimized for Android platforms.

Performance Requirements:

* The system should process data and send it to the backend within 2-3 seconds of collection.
* The web app must display real-time data within 2 seconds of retrieval from the API.
* Time series forecasting should complete in less than 5 seconds based on the amount of historical data.

Security Requirements:

* Data transmission between the ESP32 and the Node.js API should be encrypted to ensure secure communication.
* API access should be protected by authentication and proper authorization mechanisms to prevent unauthorized data access.
* The database, if implemented, must secure stored data and restrict unauthorized access.

Cultural Requirements:

* The app’s interface should be adaptable to different languages to cater to diverse user groups.
* The UI/UX design should account for cultural preferences, including appropriate color schemes and data presentation styles.

**CHAPTER III**

**Design and Development Methodologies**

**System Design**

Architectural Diagram/ Block Diagram

A system architecture shows the representation and structure of the system

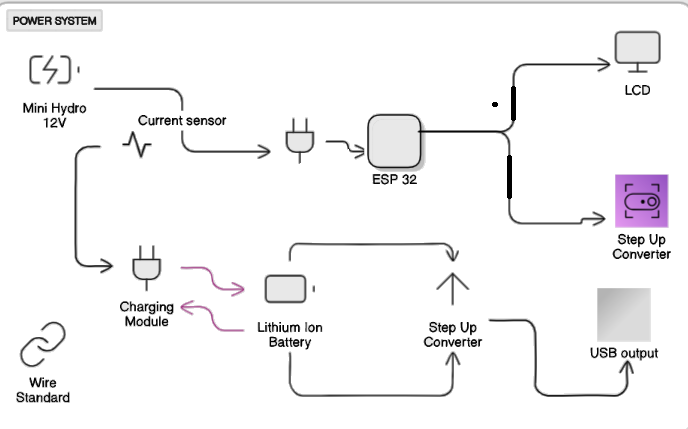
This figure illustrates the system architecture for storing electricity generated from water flow. A mini hydro generator converts water flow into electrical energy, which is monitored by a current sensor and stored in a lithium-ion battery via a charging module. The ESP32 microcontroller processes the data, displays it on an LCD screen, and transmits it to a web application for remote monitoring. A step-up converter ensures the components receive the correct voltage, while a USB output module allows the system to power external devices. This setup efficiently integrates energy generation, storage, and monitoring for renewable energy management.

Figure 1. System Architecture

DFD Level 0

Processes:

* 1.0 Data Collection from Sensors:

The ESP32 gathers data from current sensors, which monitor water flow rates and battery power levels.

* 2.0 Data Transmission to Web App:

The ESP32 sends the collected data (water flow rate, battery level, etc.) to the web application.

* 3.0 Data Display on Web App:

The web app receives the data and displays real-time information, such as remaining battery power, battery charge time, water flow rate, and power consumption.

* 4.0 User Notification and Alerts:

The app sends notifications or alerts to users when water flow rates or battery levels reach critical thresholds.

External Entities:

* Current Sensors:

These sensors provide real-time data about the water flow rate and battery charge levels to the ESP32.

* User:

The user interacts with the web app to view the real-time data, receive alerts, and monitor battery and water flow metrics.

Data Stores:

* Sensor Data Storage:

Temporarily stores data from the sensors for processing and transmission to the web app.

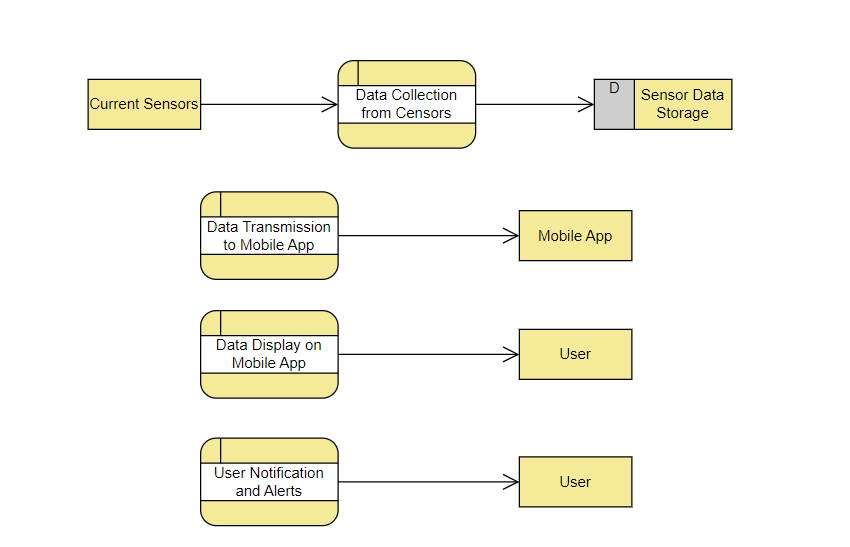
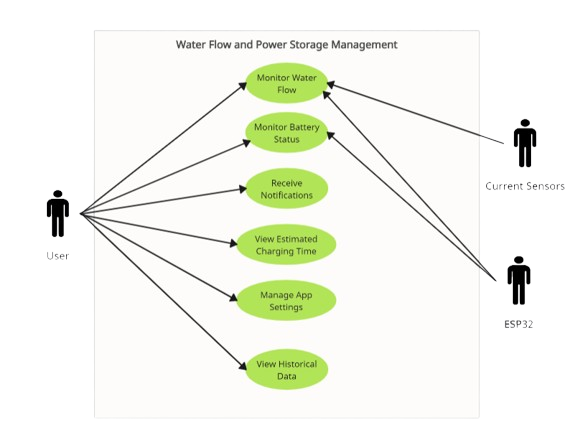


Figure 2. Data Flow Diagram

UML Use-case Diagram

The use case diagram for the Water Flow and Power Storage Management System illustrates the interactions between the user and the system components, including the ESP32 and Current Sensors. The User interacts with the web application to monitor essential functions such as water flow, battery status, and estimated charging time. They can also receive notifications, manage app settings, and view historical data to track the system’s performance over time.

The Current Sensors provide real-time data on water flow, which is monitored by the system. This information is sent to the ESP32 microcontroller, which processes both the sensor data and battery status. The ESP32 then relays this processed information to the web application, enabling the user to access live updates and notifications about the system’s status. This use case diagram effectively captures the core functionalities of the system, highlighting how the user utilizes the web app to manage water flow and power storage while relying on the ESP32 and current sensors for accurate and timely data.



Sample Mock-up

The mock-up design for the Web App Using ESP32 and Current Sensors for Water Flow and Power Storage Management provides a user-friendly interface that focuses on real-time monitoring and data visualization. The main screen prominently displays key metrics such as current, voltage, and power consumption, enabling users to easily track water flow and power storage. An interactive line graph illustrates usage trends over various timeframes, including by the minute, hour, day, week, and month. Users can switch between these intervals to analyze patterns and predict future usage through time series forecasting. The design incorporates a clean layout with blue and green tones, representing water and energy. This mock-up highlights the core functionality of the app, allowing users to efficiently manage their system's performance.

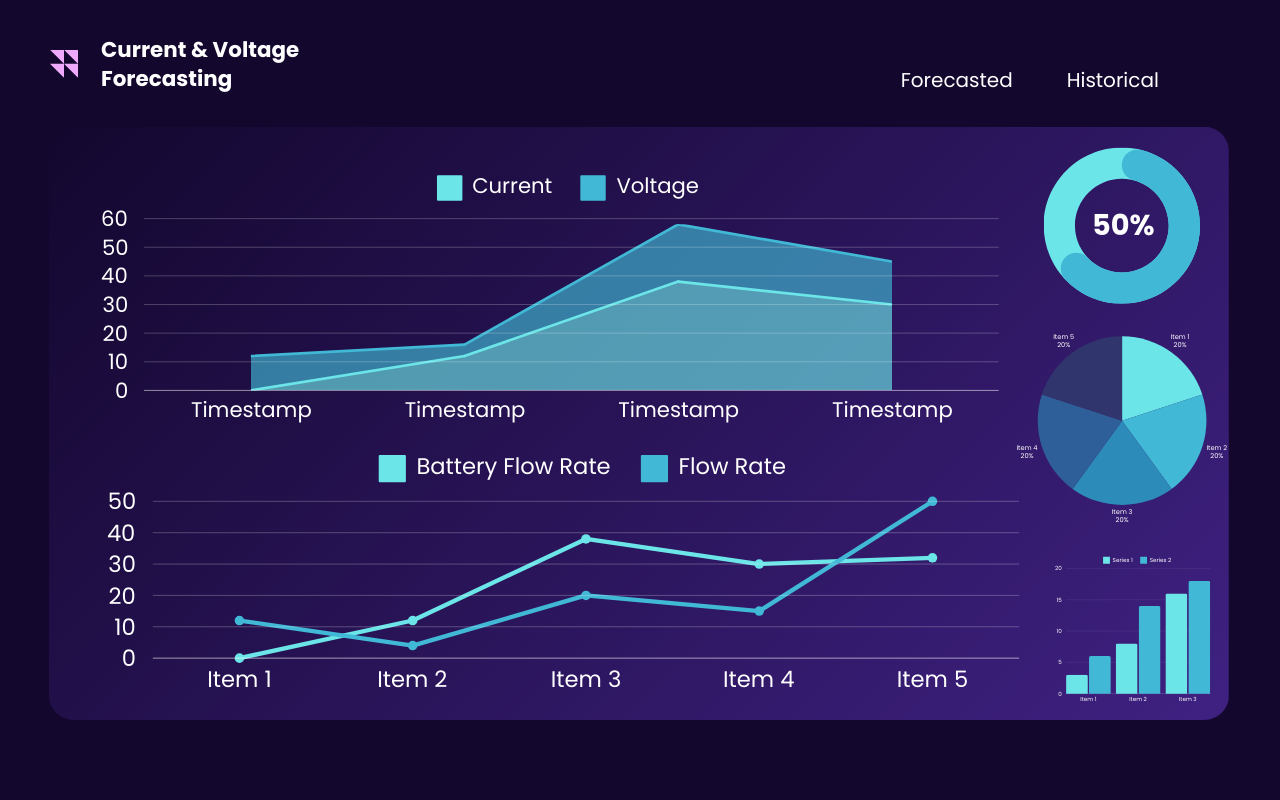


Figure 3. Sampled Mock-up

Alnaqbi, S. A., Alasad, S., Aljaghoub, H., Alami, A. H., Abdelkareem, M. A., & Olabi, A. G. (2022). Applicability of Hydropower Generation and Pumped Hydro Energy Storage in the Middle East and North Africa. *Energies*, *15*(7). https://doi.org/10.3390/en15072412

Amirul, A., Mohd, A., & Roslan, H. (2022). *Prototype Output from Renewable Energy Power Sources and Its Application to Some Electrical Equipment Controlled Using IoT*. *3*(2), 463–480.

Bhatti, N. (2021). *Minahil Ejaz (170326) Iqra Bibi (170302)*. *170326*, 1–60.

Caneda, J. D., & Calimpusan, R. A. C. O. (2022). Design and Development of IoT-based SMART Monitoring System for Hydro-Powered Generator. *8th International Conference on Engineering and Emerging Technologies, ICEET 2022*, *October 2022*. https://doi.org/10.1109/ICEET56468.2022.10007109

Hussien, N. A., Al Magsoosi, A. A. D., Al Magsoosi, A. A. D., Salim AlRikabi, H. T., & Abed, F. T. (2021). Monitoring the Consumption of Electrical Energy Based on the Internet of Things Applications. *International Journal of Interactive Web Technologies*, *15*(7), 17–29. https://doi.org/10.3991/ijim.v15i07.20183

Kougias, I., Aggidis, G., Avellan, F., Deniz, S., Lundin, U., Moro, A., Muntean, S., Novara, D., Pérez-Díaz, J. I., Quaranta, E., Schild, P., & Theodossiou, N. (2019). Analysis of emerging technologies in the hydropower sector. *Renewable and Sustainable Energy Reviews*, *113*(June), 109257. https://doi.org/10.1016/j.rser.2019.109257

Kuriqi, A., Pinheiro, A. N., Sordo-Ward, A., Bejarano, M. D., & Garrote, L. (2021). Ecological impacts of run-of-river hydropower plants—Current status and future prospects on the brink of energy transition. *Renewable and Sustainable Energy Reviews*, *142*, 1–45. https://doi.org/10.1016/j.rser.2021.110833

Macheso, P. S., & Thotho, D. (2022). ESP32 Based Electric Energy Consumption Meter. *International Journal of Computer Communication and Informatics*, *4*(1), 23–35. https://doi.org/10.34256/ijcci2213

Pereira, G. P., Chaari, M. Z., & Daroge, F. (2023). IoT-Enabled Smart Drip Irrigation System Using ESP32. *Internet of Things*, *4*(3), 221–243. https://doi.org/10.3390/iot4030012

Shafi, S., Narayan Sahu, D., Pandey, D., Kumar, S., Alaskar, K., & Professor, A. (2021). IOT based web application developed for water and power monitoring in residential building. *Volatiles & Essent. Oils*, *8*(5), 6434–6441.