**Title:** Desain dan Implementasi Sistem IoT untuk Sensor yang Dapat Dipasang dan Dilepas

**Expected Duration (days):** 180

**Author:** hadrianjavas12@gmail.com

**Related components**: eFeeder, Sensor, Platform IoT

**Approvers:** Manajer Proyek, Firmware Engineer, Hardware Engineer

|  |  |  |
| --- | --- | --- |
| **Role** | **Yes|NotYet** | **Remarks** |
| Manajer Proyek | NotYet |  |
| Firmware Engineer | Yes |  |
| Hardware Engineer | NotYet |  |

# Summary

Proyek ini bertujuan untuk merancang dan mengimplementasikan sistem IoT yang memungkinkan berbagai sensor untuk dengan mudah dipasang dan dilepas. Sistem ini akan digunakan dalam ekosistem eFishery dan memungkinkan pengguna untuk mengumpulkan lebih banyak data dari lingkungan mereka.

# Problem & Motivation

Saat ini pada ekosistem eFishery hanya memiliki kapabilitas kontrol tebar pakan (eFeeder). Dengan sistem baru ini, kami ingin memperluas fungsionalitas sistem eFeeder dan memungkinkan pengguna untuk mengumpulkan lebih banyak data dari lingkungan mereka sehingga bisa mendapatkan manfaat lebih banyak dari ekosistem eFishery (produktifitas meningkat).

# Detailed Desain

Sistem akan menggunakan pendekatan berbasis modul, di mana setiap sensor bisa dihubungkan ke sistem melalui modul yang dapat dipasang dan dilepas. Teknologi yang akan digunakan antara lain adalah modbus RS485 rtu untuk komunikasi antara sensor dan sistem, dan platform IoT MQTT untuk akuisisi data.

## UML Class

|  |
| --- |
| Sensor |
| - \_sensorName: String |
| - \_unit: String |
| - \_rawValue: double |
| - \_m: double |
| - \_C: double |
| - Info: String |
| - \_err\_measure: float |
| - \_err\_estimate: float |
| - \_q: float |
| - \_current\_estimate: float |
| - \_last\_estimate: float |
| - \_current\_estimate: float |
| + Value: double |
| + Sensor(sensorName: String, unit: String) |
| + Sensor(sensorName: String, unit: String, mValue: double, CValue: double) |
| + Sensor(sensorName: String, unit: String, mValue: double, unit: String, mValue: double, CValue: double, mea\_e: float, est\_e: float, q: float) |
| + SetSensorParam(sensorName: String, unit: String): void |
| + SetCalibrationParam(mValue: double, CValue: double): void |
| + GetRawValue(): double |
| + SetRawValue(newRawValue: double): void |
| + UpdateRawValue(newRawValue: double): void |
| + SetFilter(mea\_e: float, est\_e: float, q: float): void |
| + GetSensorInfo(): String |
| + GetSensorFullParam(): String |

|  |
| --- |
| SensorSerial |
| - \_serial: SoftwareSerial |
| - \_sensorInfo: String |
| - \_inputString: String |
| - \_readComplete: bool |
| - \_countSensorCheck: int |
| - \_channel: int |
| + SensorSerial(rxPin: uint8\_t, channelNo: int) |
| + CollectSensorData(): void |
| + GetSensorInfo(): String |
| + GetChannelNumber(): int |

|  |
| --- |
| SensorCollector |
| - Channel1: SensorSerial |
| - Channel2: SensorSerial |
| - Channel3: SensorSerial |
| - ssid: const char\* |
| - password: const char\* |
| - mqtt\_server: const char\* |
| - espClient: WiFiClient |
| - client: PubSubClient |
| - MSG\_BUFFER\_SIZE: int |
| - msg: char[MSG\_BUFFER\_SIZE] |
| - outTopic: char[MSG\_BUFFER\_SIZE] |
| + setup\_wifi(): void |
| + reconnect(): void |
| + PublsihChannel(channel: int, message: String): void |
| + GetChannelNumber(): int |
| + setup(): void |
| + loop(): void |

|  |
| --- |
| SensorAdapter – Analog/Humidity/LoadCell |
| - sensorName: String |
| - unit: String |
| - m: double |
| - C: double |
| - sensorPin: int |
| - deliveryIntervalTime: int |
| - err\_measure: float |
| - err\_estimate: float |
| - q: float |
| - analogSensor: Sensor |
| - mySerial: SoftwareSerial |
| + setup(): void |
| + loop(): void |
| + taskSendSensorData(void \*): void |

# Dependencies

## Library "Sensor"

Library ini telah dibuat dengan tujuan memfasilitasi pengelolaan data sensor sebelum data tersebut dikirimkan. Library ini mencakup fungsi-fungsi seperti kalibrasi dan filter value untuk menghilangkan noise yang mungkin muncul pada data sensor. Dependensi pada library ini memastikan bahwa data sensor yang dikelola dan diproses sebelum pengiriman akan menjadi akurat dan dapat diandalkan.

## Library "Sensor Serial"

Library ini digunakan untuk mengambil data dari sensor adapter, yang akan berperan penting dalam pengoperasian sensor collector. Dengan dependensi pada library "Sensor Serial," data yang diterima dari sensor adapter dapat diakses dan diproses dengan efisien oleh sensor collector untuk kemudian diteruskan ke ekosistem IoT eFishery.

# Milestone/Deployment Strategy

Proyek ini akan dibagi menjadi beberapa fase, termasuk desain, implementasi, testing, dan deployment. Setiap fase akan memiliki milestone dan deliverables yang jelas seperti yang dijelaskan pada dokumen probrem discovery

## Timeline

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Item | | Phase | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| Development | Sensor Interface Research (Sensor Adapter, Modbus RTU RS485, Sensor Collector) |  |  |  |  |  |  |  |  |  |
| Data Comunication (Sensor Collector) |  |  |  |  |  |  |  |  |  |
| IoT Integration (Integration wirh eFishery Ecosystem) |  |  |  |  |  |  |  |  |  |
| Staging | Testing and Validation |  |  |  |  |  |  |  |  |  |
| Deployment | Design for Master Production |  |  |  |  |  |  |  |  |  |
| Quality Control |  |  |  |  |  |  |  |  |  |
| Mass Production |  |  |  |  |  |  |  |  |  |
| Quality Control |  |  |  |  |  |  |  |  |  |
| Business | Deliver to Users |  |  |  |  |  |  |  |  |  |
| After Sales | Bug Fixes |  |  |  |  |  |  |  |  |  |
| Add New Features |  |  |  |  |  |  |  |  |  |

# Drawbacks/Risks

Salah satu risiko utama dalam proyek ini adalah kompatibilitas dengan berbagai jenis sensor, yang mungkin memiliki spesifikasi teknis yang berbeda dan mungkin memerlukan protokol komunikasi yang berbeda.

## **Penyesuaian Format Komunikasi**

Salah satu kekurangan yang mungkin timbul adalah kebutuhan untuk membakukan format komunikasi antara perangkat sensor adapter, sensor collector, dan ekosistem IoT eFishery. Jika ada sistem tambahan yang memerlukan format komunikasi yang berbeda, hal ini dapat mengharuskan pembaruan firmware pada perangkat tersebut. Ini mungkin mengakibatkan waktu tambahan dan upaya yang diperlukan untuk pengembangan dan pengujian.

## **Biaya Tambahan untuk Setiap Sensor**

Karena perangkat sensor adapter dan sensor collector dipisahkan, diperlukan perangkat keras tambahan pada setiap sensor. Hal ini dapat mengakibatkan biaya tambahan untuk setiap sensor yang akan diintegrasikan ke dalam sistem. Biaya tambahan ini perlu dipertimbangkan dalam perencanaan anggaran proyek.

## Kompleksitas Hardware

Adanya perangkat keras tambahan pada setiap sensor juga dapat meningkatkan kompleksitas fisik sistem. Hal ini dapat memerlukan manajemen tambahan dalam hal pemeliharaan dan penggantian perangkat keras jika diperlukan.

# Alternatives

## Penggunaan Protokol Komunikasi Lain

Sebagai alternatif untuk Modbus RTU RS485, kita dapat mempertimbangkan penggunaan protokol komunikasi lain seperti I2C atau CAN, tergantung pada jenis sensor yang digunakan. Pemilihan protokol komunikasi yang berbeda dapat bermanfaat jika kita menghadapi kebutuhan komunikasi khusus atau memerlukan fleksibilitas dalam metode komunikasi sensor.

## Kombinasi Sensor Adapter dan Sensor Collector:

Sebagai alternatif untuk memisahkan sensor adapter dan sensor collector, kita dapat mempertimbangkan untuk menggabungkannya menjadi satu perangkat yang dapat mendukung beberapa jenis sensor yang menghasilkan sinyal dasar seperti Voltage, milliVolt, atau mA. Dengan cara ini, fungsi dasar perangkat akuisisi data sensor dapat terpenuhi dalam satu perangkat yang lebih sederhana.

# Unresolved/Future

Possibilities Salah satu kemungkinan masa depan adalah integrasi dengan platform analitik data untuk memungkinkan pengguna untuk menganalisis data yang dikumpulkan oleh sensor.