**Assignment 1**

**Step 1**

The main requirement for this project is to design an Automated Food Dispenser for pets for a local animal shelter. The basic functionality of this system will be as below

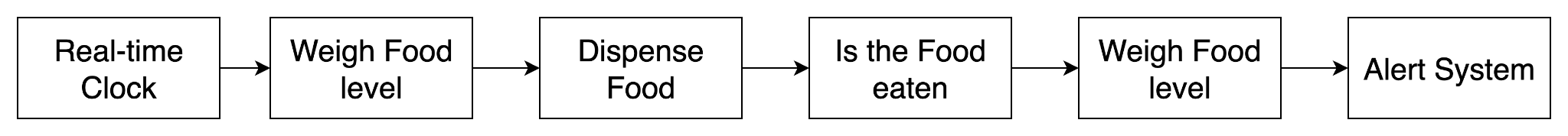
* Scheduled Feeding Time
* Auto-dispense food with a specific size
* Sensor to monitor Food and availability
* Monitor food consumption

The system will have certain Inputs and Outputs. For Example, the Inputs will include a food weight level detector to detect the dispensing and consumption of Food. A real-time clock, to manage the schedules.

The outputs will be a display of when the food should be dispensed according to the schedule, a motor to dispense the right amount of food. An alert system to alert the user if the food is not dispensed.

Focusing on the Operational parameters (assumptions), the feeder will dispense food between 25g to 150g, depending on the pet and their diet. The weight sensor will have to be very accurate so that there is minimal chance of error. The real-time clock will also be maintained accurately as this manages the number of times the food is being dispensed. These features together will ensure that the Automated Food Dispenser will be a pet-friendly experience.

Below is a simple block diagram to represent the flow from inputs such as a timer and a weight measure sensor, and then outputs like an alert system and motor control.



**Step 2**

| Type | Parameter | Sample |
| --- | --- | --- |
| Input | Clock | 7:00AM, 1:00PM |
| Input | Food Sensor | 75% |
| Input | Weight Sensor | 250g |

| Type | Parameter | Sample |
| --- | --- | --- |
| Output | Alert Notification | Low Food |
| Output | Motor Control | Motor Issues |
| Output | Display | Food Level, Time left |

Features:

* Automated monitoring of food times and bowl weight
* Alert Notifications

Inputs:

* Scheduled Feeding times (Ex. 7AM, 1PM)
* Measure Food in the bowl

Outputs:

* Low Food alerts
* Reports Motor fault

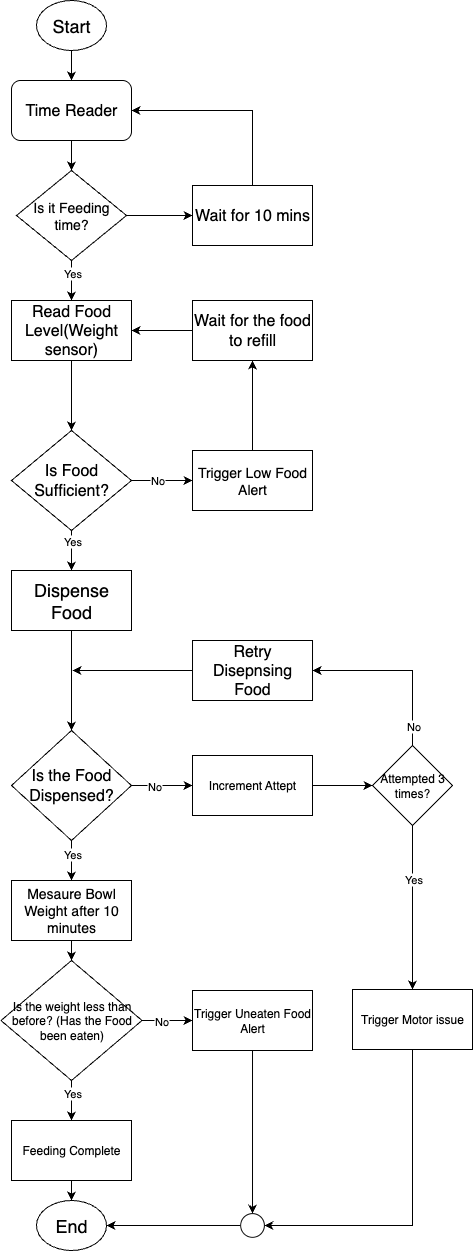
Assumptions:

* Sensors are accurate
* Food is stocked already

Expected Outcome:

* Food is dispensed at the scheduled time.
* The exact amount of food is dispensed

**Step 3:**

****

**Step 4**

* Time Reader displays if it is feeding time.
* If it is feeding time, check if there is enough food in stock, If there is not, trigger low food alert
* Dispense pre-decided amount of food
* If the Food is not dispensed, trigger an alert regarding the issue with the Motor
* Mesaure the bowl weight after a while, to see if the food is eaten by pets, if it is not, trigger an alert that Food is not eaten
* Display Feeding Complete

**Step 5:**

| Scenario | Input | Exepcted Output |
| --- | --- | --- |
| Pet eats as expected | Clock is at 8:AM, Food bowl is 100% filled | Food should dispense and the pet should eat |
| Pet does not eat | After the scheduled time, food bowl weighs the same as before | Motor dispenses food, but pet did not eat, send alert |
| Food bin is empty | At the scheduled feeding time, the food bowl is empty | Alert sent to refill the food stock |
| Motor fault during dispensing | Motor could not dispense food during the scheduled feeding time | Alert sent informing the motor issue |
| Overfeeding prevention | Next feed is scheduled at 1PM. Until then the food bowl remains empty | No more food is dispensed |

### Refinements & Improvements

* Adjustable alert timings — allow owner to set how long before uneaten food triggers an alert.
* Consumption history logs — track how much each pet eats over time to identify health changes.
* Redundant sensors — e.g., second bin sensor to confirm food availability and reduce false “empty” alerts.
* Remote manual override — caretaker can trigger feeding from a mobile app in emergencies.

[**https://github.com/AdiShanbhag/u3315922\_Assignment1**](https://github.com/AdiShanbhag/u3315922_Assignment1)

**Part 3: AI Integration**

**Prompt:**

I have a system that is an automatic pet feeder. I have done the flow chart of the same and completed the Word coding, which is as follow Time Reader displays if it is feeding time. If it is feeding time, check if there is enough food in stock, If there is not, trigger low food alert Dispense pre-decided amount of food If the Food is not dispensed, trigger an alert regarding the issue with the Motor Mesaure the bowl weight after a while, to see if the food is eaten by pets, if it is not, trigger an alert that Food is not eaten Display Feeding Complete I want your assistance to enhance the word coding with one statement each

**Output**

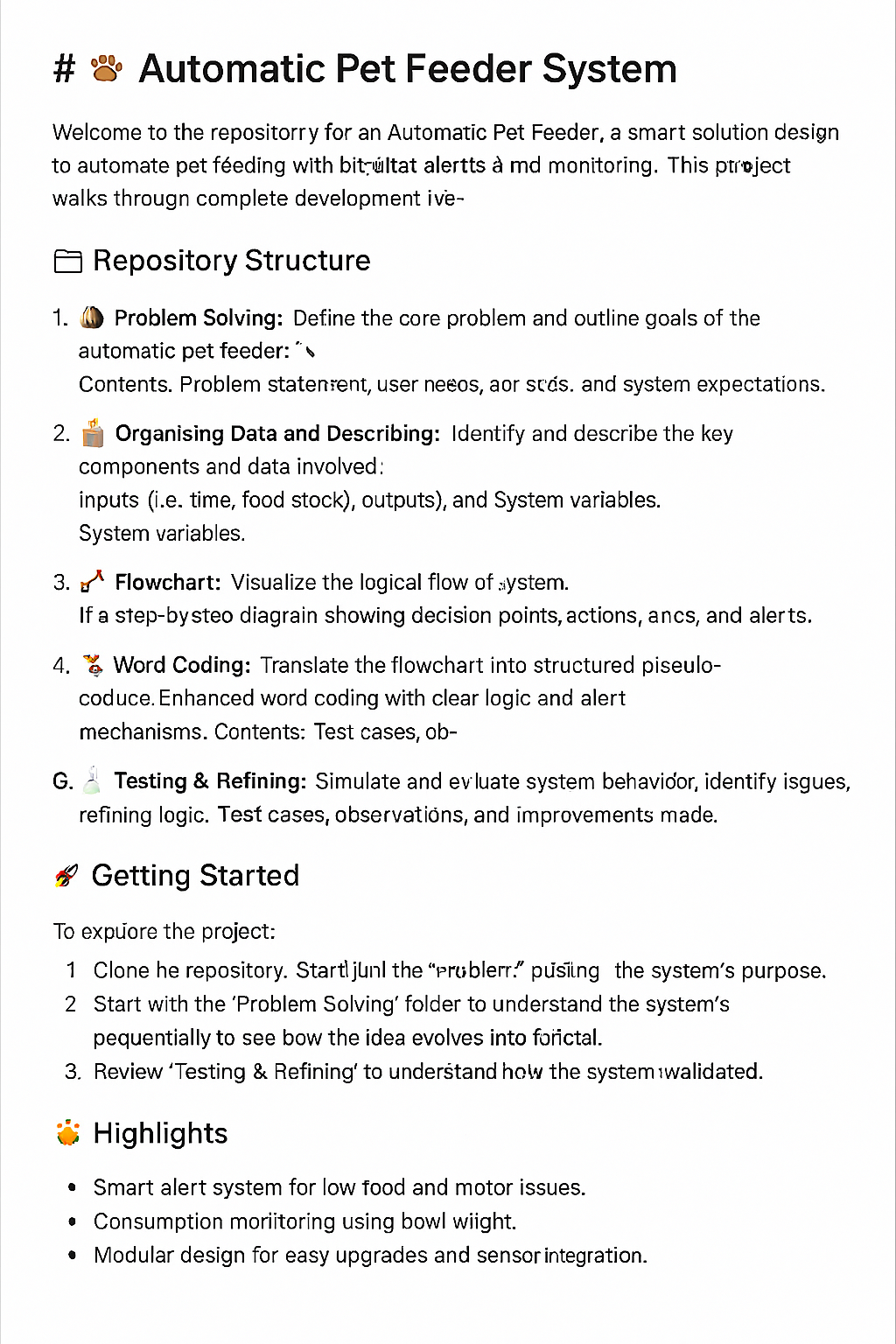
## Enhanced Word Coding – Automatic Pet Feeder

1. Check Feeding Time → Time Reader checks and displays whether it's the scheduled feeding time.
2. Verify Food Stock → If it is feeding time, check if food stock is sufficient.
3. Trigger Low Food Alert → If food stock is low or empty, activate the low food alert and halt feeding.
4. Dispense Food → If food is available, dispense the pre-decided quantity using the motor.
5. Check Dispensing Status → If food is not dispensed successfully, trigger a motor malfunction alert.
6. Monitor Consumption → After a set delay, measure bowl weight to determine if food was eaten.
7. Trigger Uneaten Food Alert → If bowl weight remains unchanged, trigger a food-not-eaten alert.
8. Complete Feeding Cycle → Display "Feeding Complete" message and log the event.

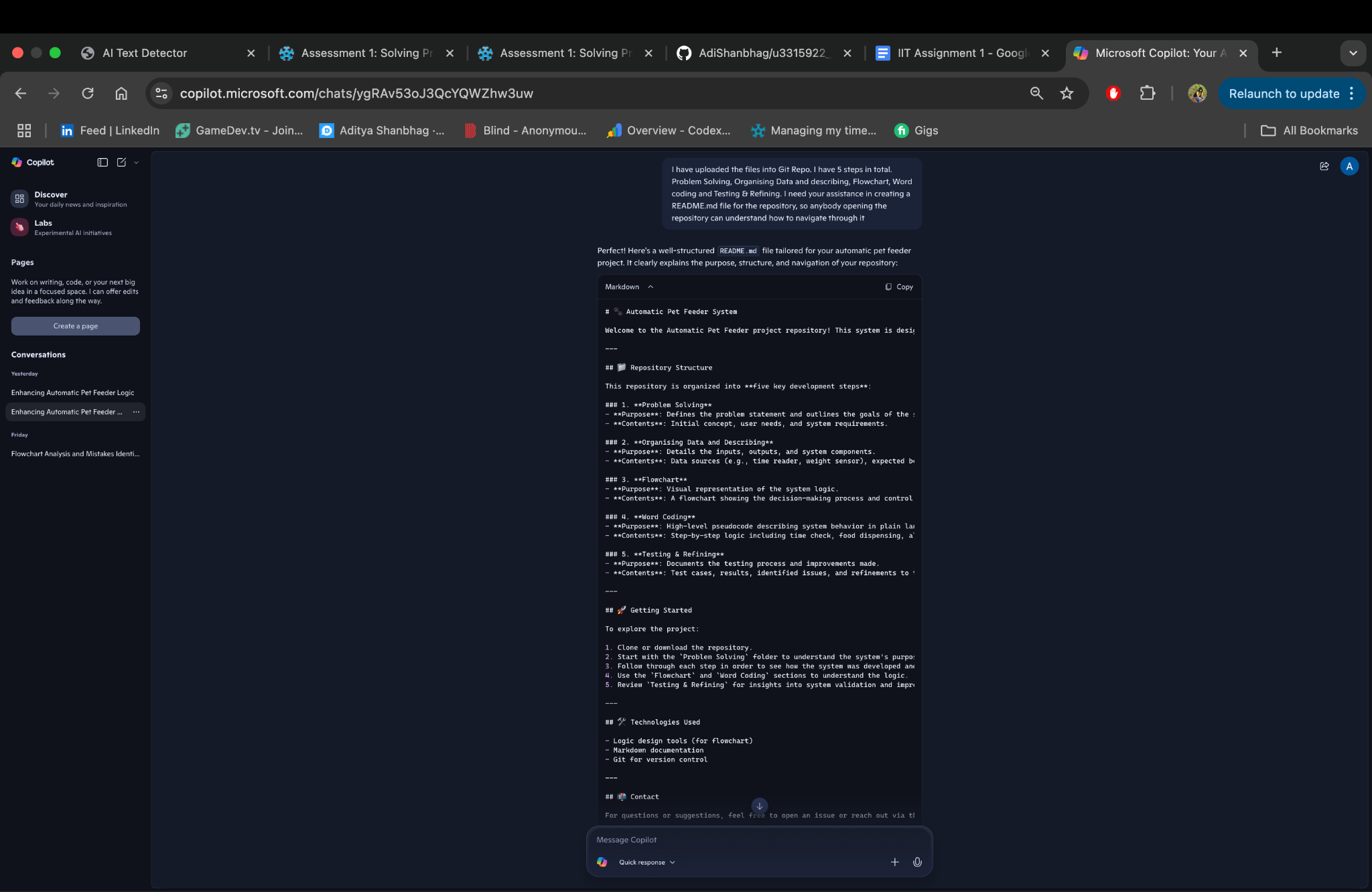
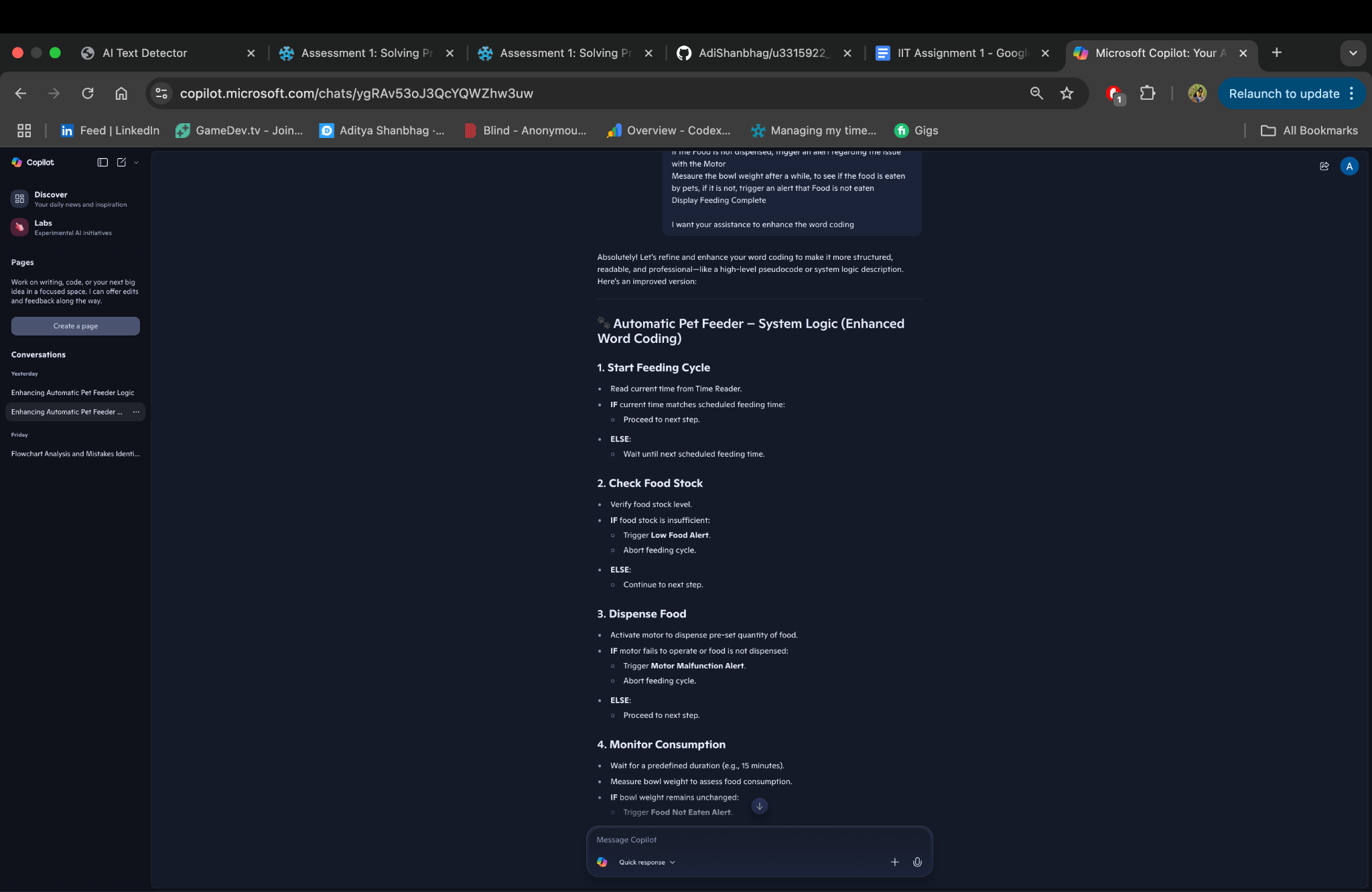
**Prompt:**

I have uploaded the files into Git Repo. I have 5 steps in total. Problem Solving, Organising Data and describing, Flowchart, Word coding and Testing & Refining. I need your assistance in creating a README.md file for the repository, so anybody opening the repository can understand how to navigate through it

**Output**



**Screenshots of the conversation with the co-pilot**

****

**Insights & Influence of CoPilot**

The AI agent has assisted significantly in enhancing my logical reasoning by suggesting enhancements such as the introduction of retry mechanisms, which has made the code more efficient and reliable in general.

Moreover, its significance of the concept of coding provided me with a clearer concept of how to portray the inherent logic in a structured and comprehensible manner.

This not only reinforced my knowledge of programming practice but also enabled me to write a readable and explanatory README file for my GitHub repository.