



Making LLMs useful

~~Massimo Re Ferrè~~
~~Director Product Management, AWS~~

Mark Rambow
Sr. Software Development Management

Why am I here

Have a look with me under the hood of Amazon Q Feature Development



Created with Stable Diffusion, through [PartyRock](#)

Can GenAI do my day job? From **WOW** to **WORK!**



Created with Stable Diffusion, through [PartyRock](#)

My first prompt

Please create a python program that calculates how much energy from my solar system could be self-used if I'd install a battery.

I have data from my solar panels in the following format:

FeedIn, Production, Consumption, SelfConsumption, Purchased

- FeedIn means I feed my produced energy to the grid. When the battery is not fully charged, this will be charging the battery until it reaches full capacity and only after feed in to the grid.

- When I consume energy I take it in this order:

1. from production
2. from battery (discharging its level)
3. from the grid when the battery is empty.

For the given JSON I want for each hour a one line in a CSV file with:
production, feedIn, selfConsumption and the battery level

Here is an example JSON: {"energyDetails": {"timeUnit": "HOUR", "unit": "Wh", "meters": [{"type": "Consumption", "values": [{"date": "2022-03-02 00:00:00", "value": 725.0}, {"date": "2022-03-02 01:00:00", "value": 795.0}, {"date": "2022-03-02 02:00:00", "value": 783.0}, {"date": "2022-03-02 03:00:00", "value": 748.0}, {"date": "2022-03-02 04:00:00", "value": 842.0.....

My first Prompt - It can code!!!

Decent, not super idiomatic though

We will get there!

Let's see if we can do it without ANY coding

```
# Input JSON
json_data = '{"energyDetails": {"timeUnit": "HOUR", "unit": "Wh", "meters": [...]}}'

# Parse JSON
data = json.loads(json_data)
meters = data['energyDetails']['meters']

# Extract data for each type
consumption = [entry['value'] for entry in meters[0]['values']]
purchased = [entry['value'] for entry in meters[1]['values']]
production = [entry.get('value', 0) for entry in meters[2]['values']]
self_consumption = [entry['value'] for entry in meters[3]['values']]
feed_in = [entry['value'] for entry in meters[4]['values']]

# Calculate battery level
battery_capacity = 10000 # Define battery capacity
battery_level = [0] * len(consumption)

for i in range(1, len(consumption)):
    available = production[i] + battery_level[i-1] - self_consumption[i]
    battery_level[i] = min(available, battery_capacity)

# Write results to CSV
with open('output.csv', 'w', newline='') as csvfile:
    fieldnames = ['production', 'feedIn', 'selfConsumption', 'batteryLevel']
    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
    writer.writeheader()

    for i in range(len(consumption)):
        writer.writerow({'production': production[i], 'feedIn': feed_in[i], 'selfConsumption': self_consumption[i], 'batteryLevel': battery_level[i]})

print("CSV file generated: output.csv")
```

My first prompt - Iterate

I apologize for the oversight. I realized I made a mistake in the previous code by not accounting for the time series of the data. Please see the corrected version of the

I understand, I'll modify the code to ensure that the battery level cannot be negative. In the case of battery discharge, the consumption will be from the grid. Here's the updated

< ----- Conversational experience to define the mini steps to achieve the final goal which include steering and code generation ----->



Starting point

Goal

Several hours later...

Maybe it is it all just a hype?

Let's start over, be the Engineering Manager...

- My team would
 - Disambiguate
 - Split tasks
 - Bring in ideas and perspectives
 - Would aim for an iterative approach
 - My team would get it done!

Let's start over – Planning removes ambiguity

Understand the *intent & context*

- Mark wants
 - to feed less power back to the grid
 - safe money
 - to know the ROI of a battery by size

Wasn't it all obvious?



Created with Stable Diffusion, through [PartyRock](#)

Let's start over - Design

Divide and Conquer

- Parse the JSON
- Calculate the battery impact per hour
- Generate a CSV with new hourly values
- Variables
 - Battery size and price/wh
 - Grid price



Created with Stable Diffusion, through [PartyRock](#)

Let's start over

Review and Iterate
Review and Iterate
Review and Iterate
Review and Iterate
Review and Iterate
Review and Iterate
Review and Iterate
Review and Iterate
Review and Iterate
Review and Iterate
Review and Iterate
Review and Iterate



Created with Stable Diffusion, through [PartyRock](#)

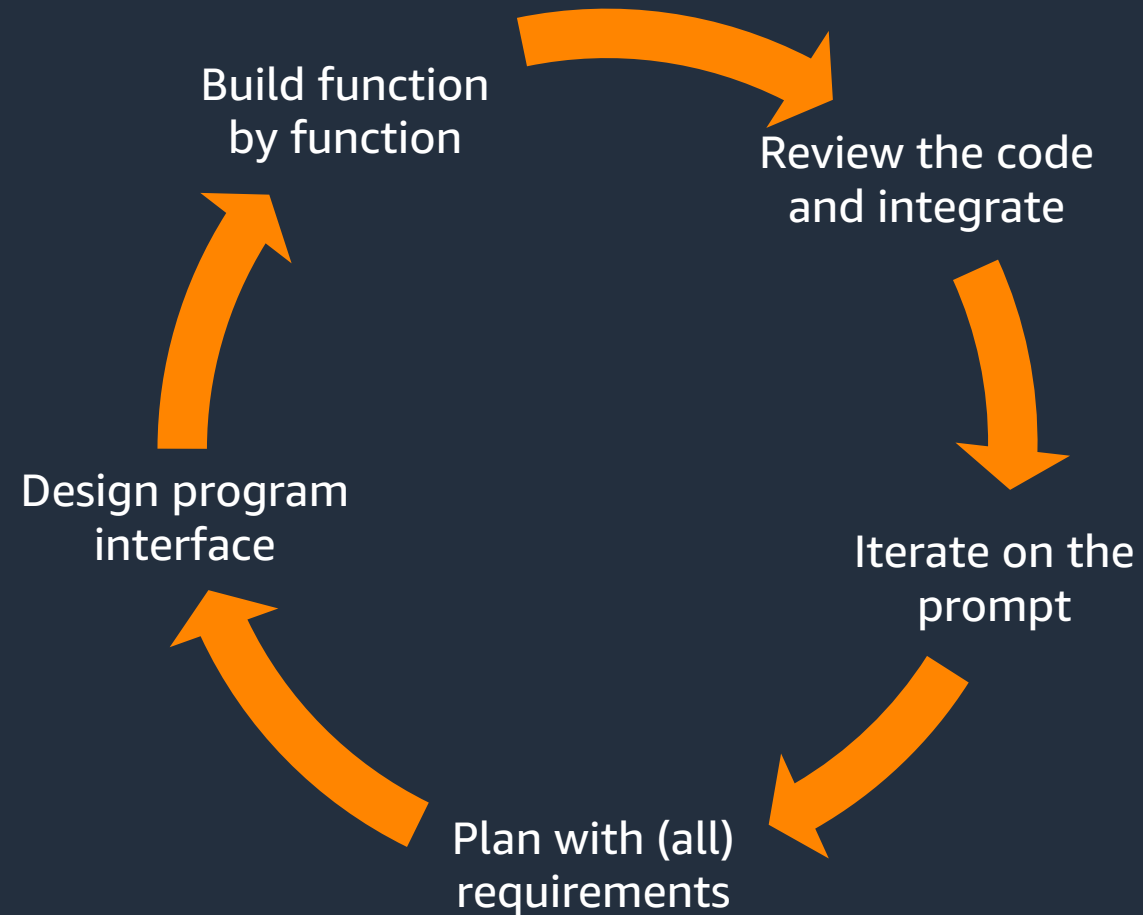
Finally... it worked

I found out
A battery is not a smart investment :-/



Created with Stable Diffusion, through [PartyRock](#)

LLM supported Development



How do you usually get there?



Human language to code

- Exciting potential for coding
- Natural Language to code is complex
- Let's break it down



Created with Stable Diffusion, through [PartyRock](#)

Understanding Semantics – The Enigma of Intent

- The intent is not just keywords
- Context is crucial

Strengthen security

Improve performance

Reduce complexity

Refactor the code

Improve scalability

Make the UI faster



Created with Stable Diffusion, through [PartyRock](#)

Code Generation – An Art and Craft

It works!



Created with Stable Diffusion, through [PartyRock](#)

Integration – Flawless Fusion

And it works in style



Created with Stable Diffusion, through [PartyRock](#)

A Holistic View: Beyond Individual Changes

- Solving each problem is a challenge
- Consider the interplay between
 - Intent
 - code generation and
 - integration of all puzzle pieces



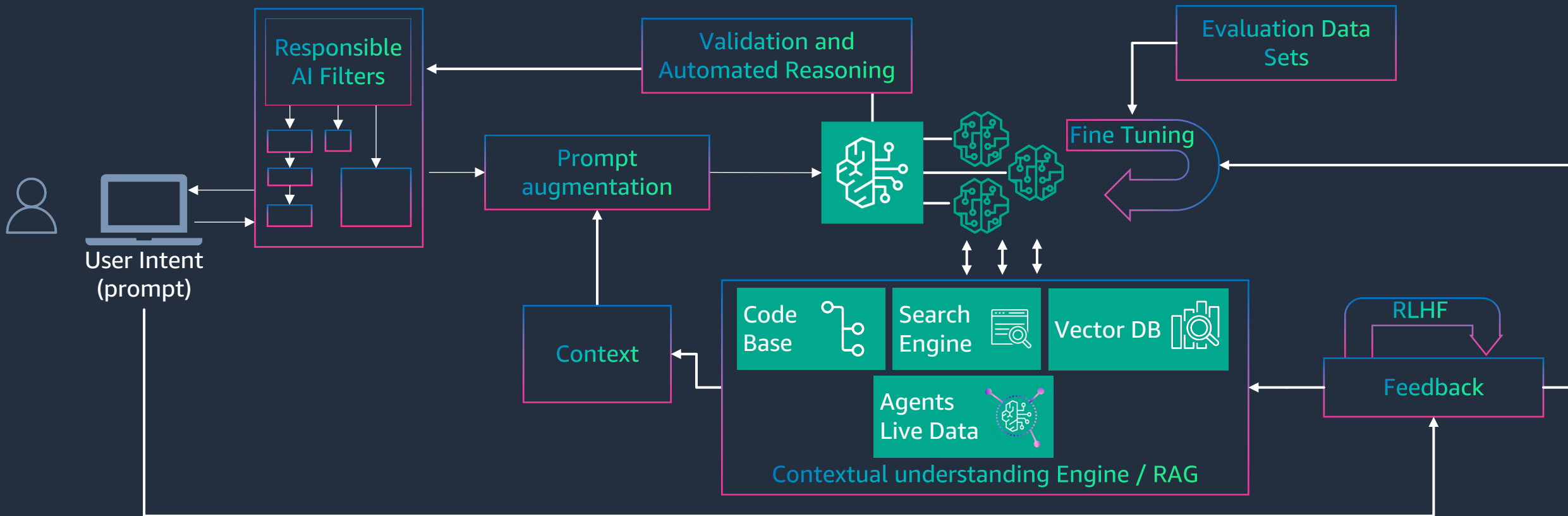
Created with Stable Diffusion, through [PartyRock](#)

From Puzzle To Solution

- Understanding Semantics
- Context awareness
- Responsible AI
- Current information
- Code validation
- Agents
- Domain knowledge
- Fighting hallucinations
- Adversarial models
- RLHF and Fine Tuning

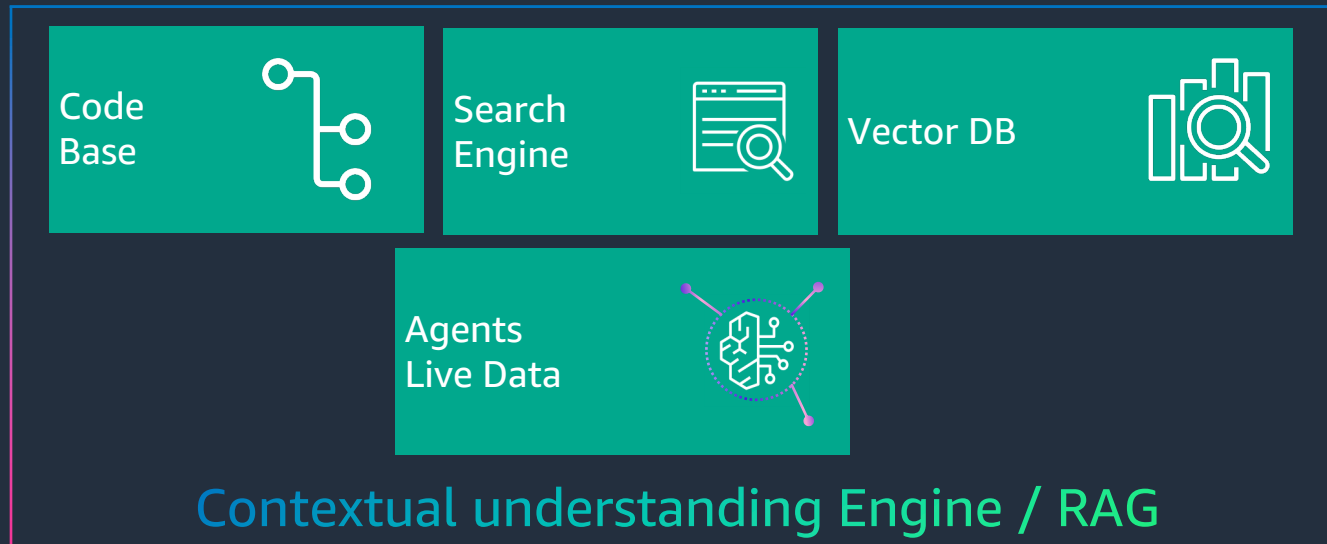
Making LLMs useful, takes a village...

Powerful Large Language Models alone don't cut it
Understanding a person's intent needs context



Let me explain, don't make things up

How do you start with a new codebase?



Created with Stable Diffusion, through [PartyRock](#)

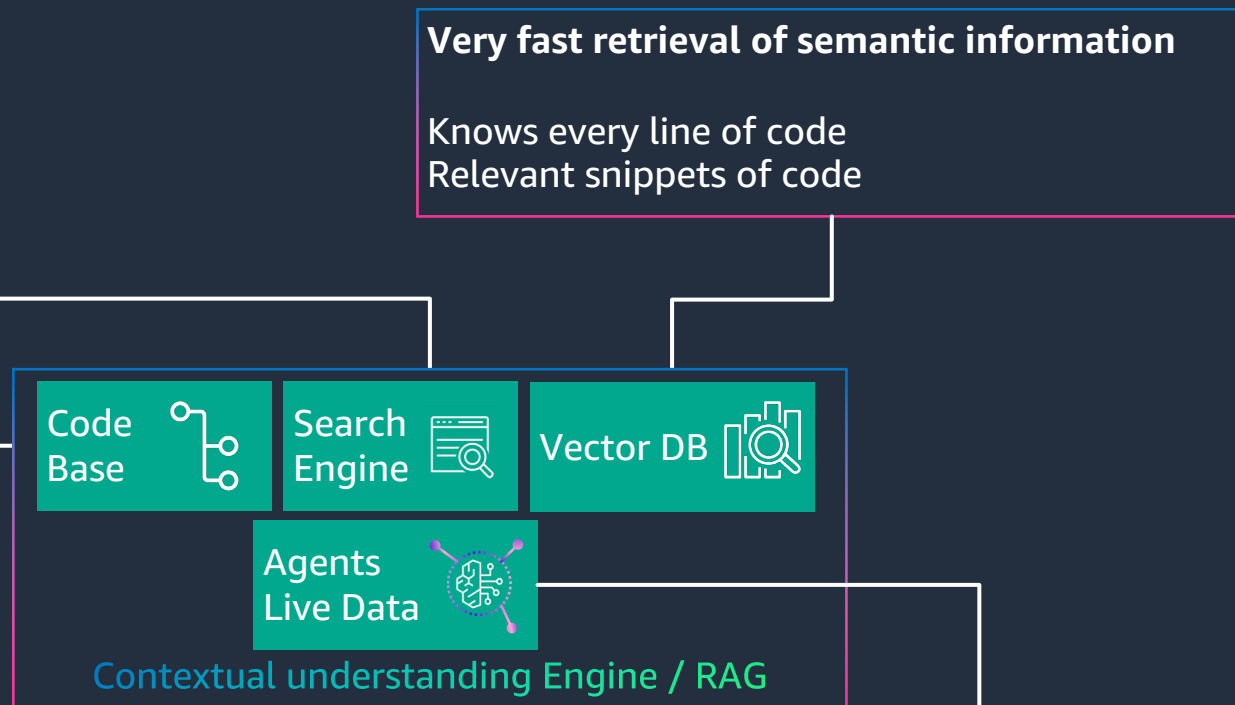
Zoom in

RAG is the detective

Relevant information and Domain Knowledge
Reasons about them
Bridges the gap between Natural language, the domain and code base

Code Base - the *Where* and *How*

Dealing with Limitations (context window, tokens cost)



Very fast retrieval of semantic information

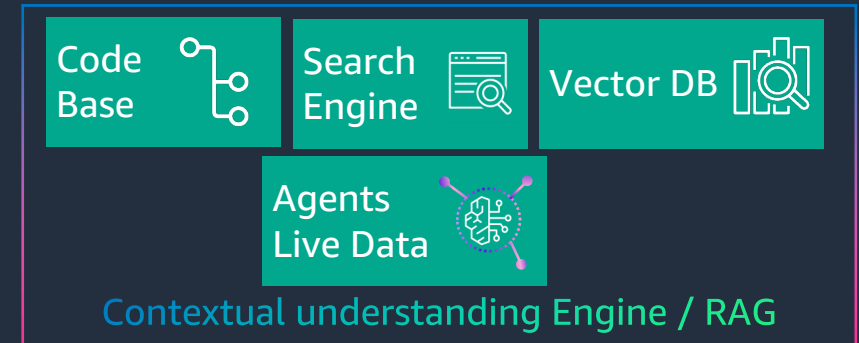
Knows every line of code
Relevant snippets of code

Action!

Look up the latest API description
The Code Whisperers
Try and Error
Build, Execute

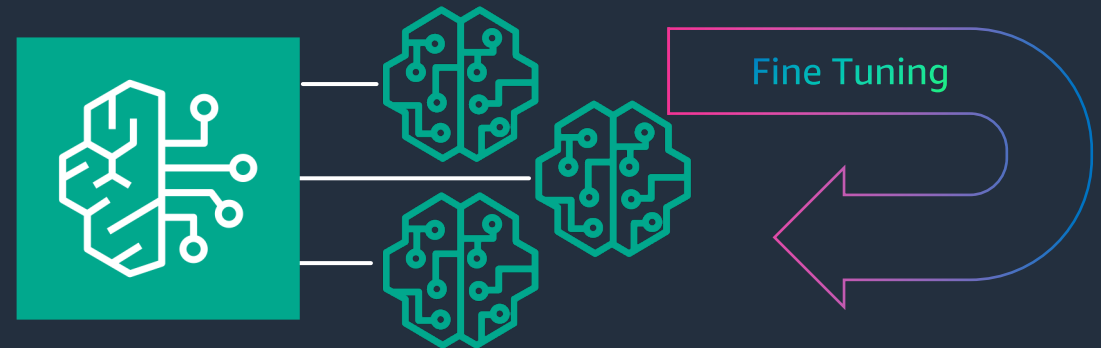
Rewind

- **Accuracy:** RAG helps to translate natural language into code that actually works, minimizing errors
- **Efficiency:** VectorDBs allow fast access to relevant code and boost the speed of RAG
- **Agents:** Test code in action, provide immediate feedback and ensure it works



A Symphony of Models

- LLMs are powerful, but for our complex task, they need help
- Combine several smaller, specialized models



Building Trustworthy Code

- Correct
- Everything tested?
- Automated Reasoning
 - Rust developers?

Validation and
Automated Reasoning

AI won't make developers irrelevant, but increase quality

**We all want better software
and there is a new promising tool for it**



Thank you!

Mark Rambow
Rambow@amazon.de



Please complete the session survey.

