Generative AI for software Development:

Authors:
EL ALOUT Ismail, EL HABTI Ouassel, KARMAOUI Oussama
LAHMOUZ Zakaria, HALILOUA Othmane

Supervisor:
Mr CHAARI Lotfi

Plan

- Introduction
- Work Organization and task planning
- Development and versioning tools
- Dataset building
- File System Architecture and Data Pipelines
- Fine-tuning
- Deployment
- Demonstration
- Conclusion

Introduction

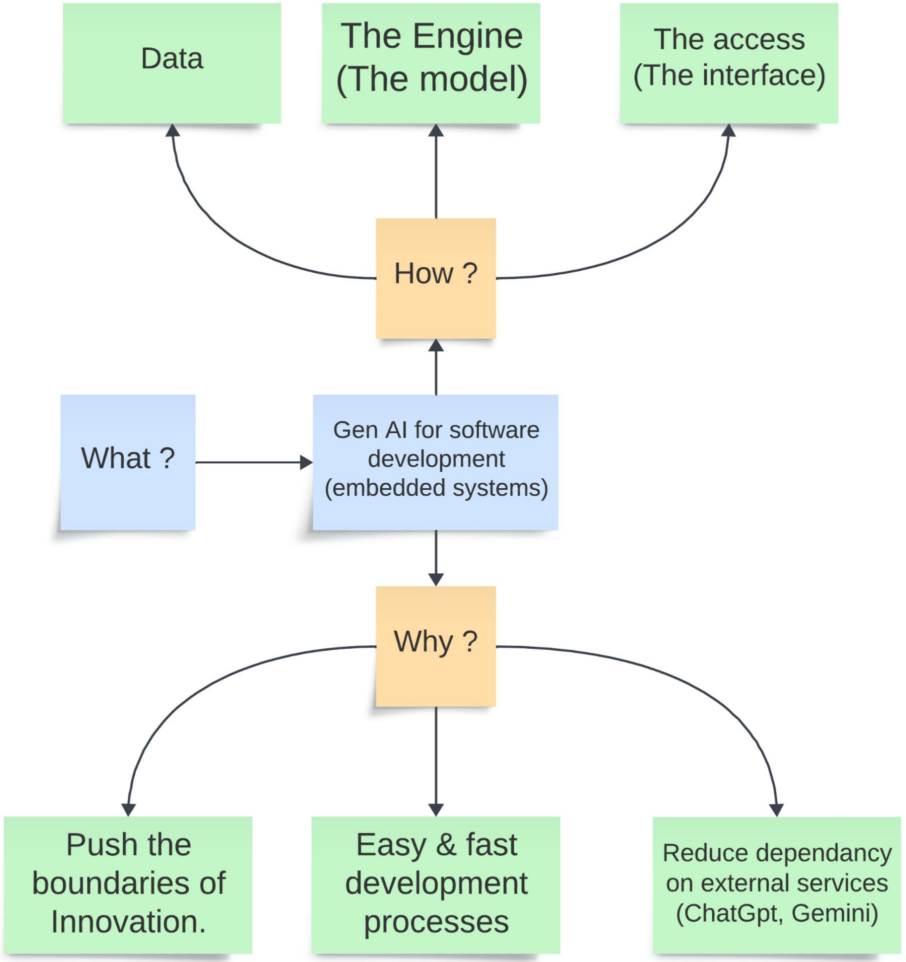
The What ? The why ? And The How ?

- AI has pushed the boundaries of what can be achieved.

- One would like to live up to the standards.

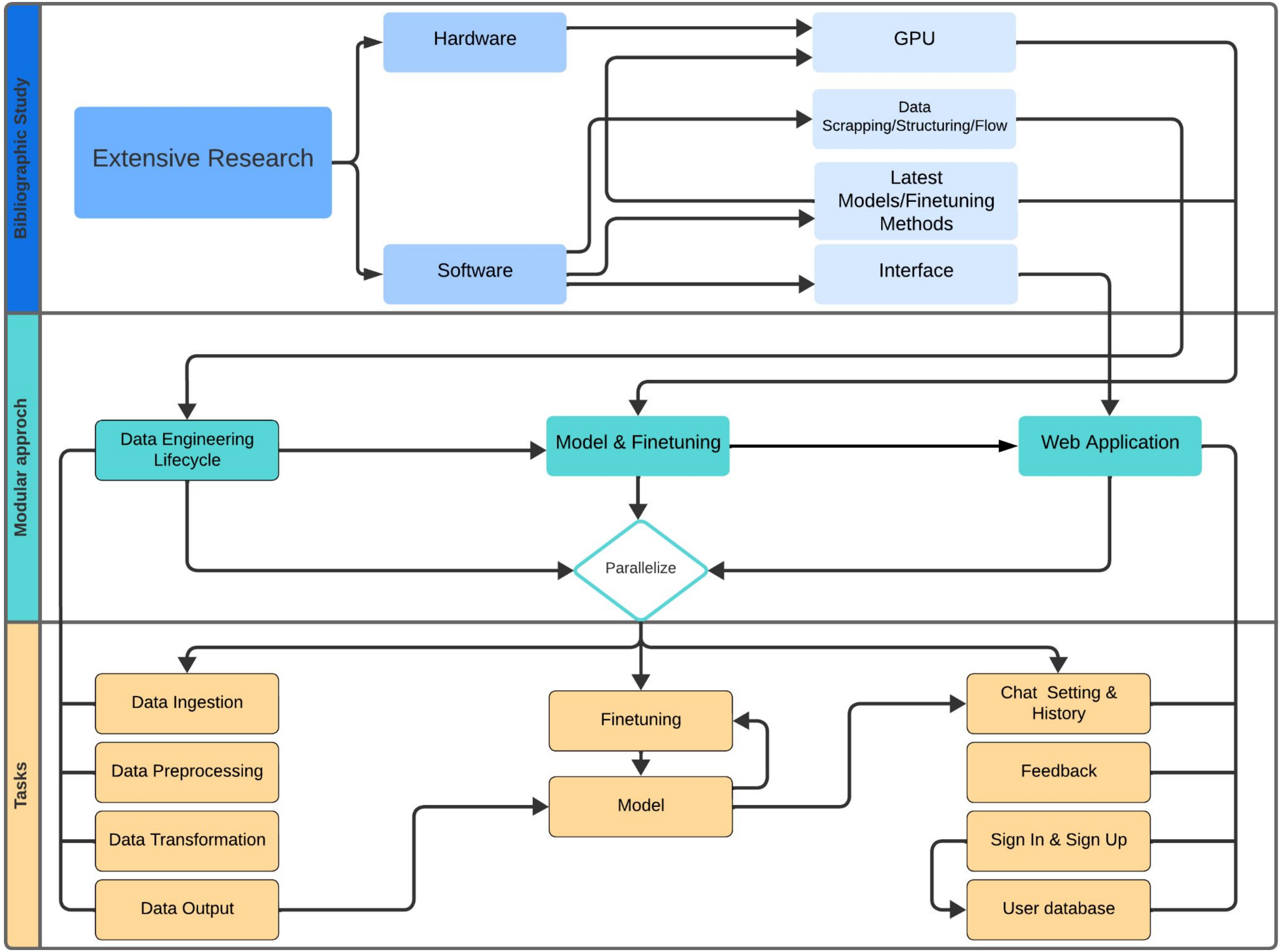
- One would want to be a protagonist in the innovation act, to better his work, and reduce his dependence.

- To that end, three pivotal components: Data, the model, the interface.


Data
The Engine (The model)
The access (The interface)
How ?
What ?
Gen AI for software development (embedded systems)
Why ?
Push the boundaries of Innovation.
Easy & fast development processes
Reduce dependancy on external services (ChatGpt, Gemini)

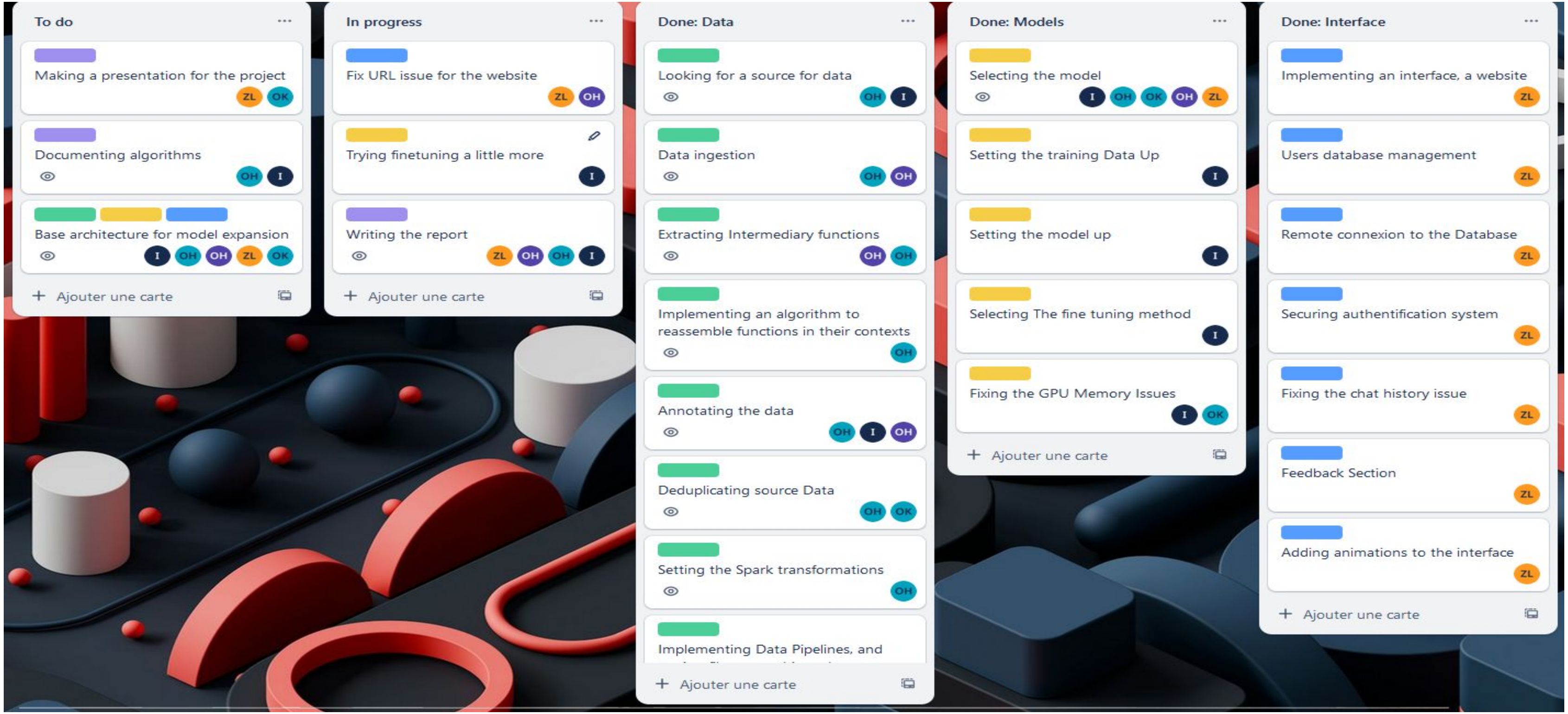
Work organization and task planning

TASK PLANNING & THOUGHT PROCESS:



PROJECT MANAGEMENT:

The use of Trello allowed us to better depict the tasks that require tackling, monitoring, refining or initiating. We used our modular development approach so that member can be assigned easily to corresponding tasks, with brackets to evaluate the completion rate.



To do

- Making a presentation for the project — ZL OK
- Documenting algorithms — OH I
- Base architecture for model expansion — I OH OH ZL OK

+ Ajouter une carte

In progress

- Fix URL issue for the website — ZL OH
- Trying finetuning a little more — I
- Writing the report — ZL OH OH I

+ Ajouter une carte

Done: Data

- Looking for a source for data — OH I
- Data ingestion — OH OH
- Extracting Intermediary functions — OH OH
- Implementing an algorithm to reassemble functions in their contexts — OH
- Annotating the data — OH I OH
- Deduplicating source Data — OH OK
- Setting the Spark transformations — OH
- Implementing Data Pipelines, and

+ Ajouter une carte

Done: Models

- Selecting the model — I OH OK OH ZL
- Setting the training Data Up — I
- Setting the model up — I
- Selecting The fine tuning method — I
- Fixing the GPU Memory Issues — I OK

+ Ajouter une carte

Done: Interface

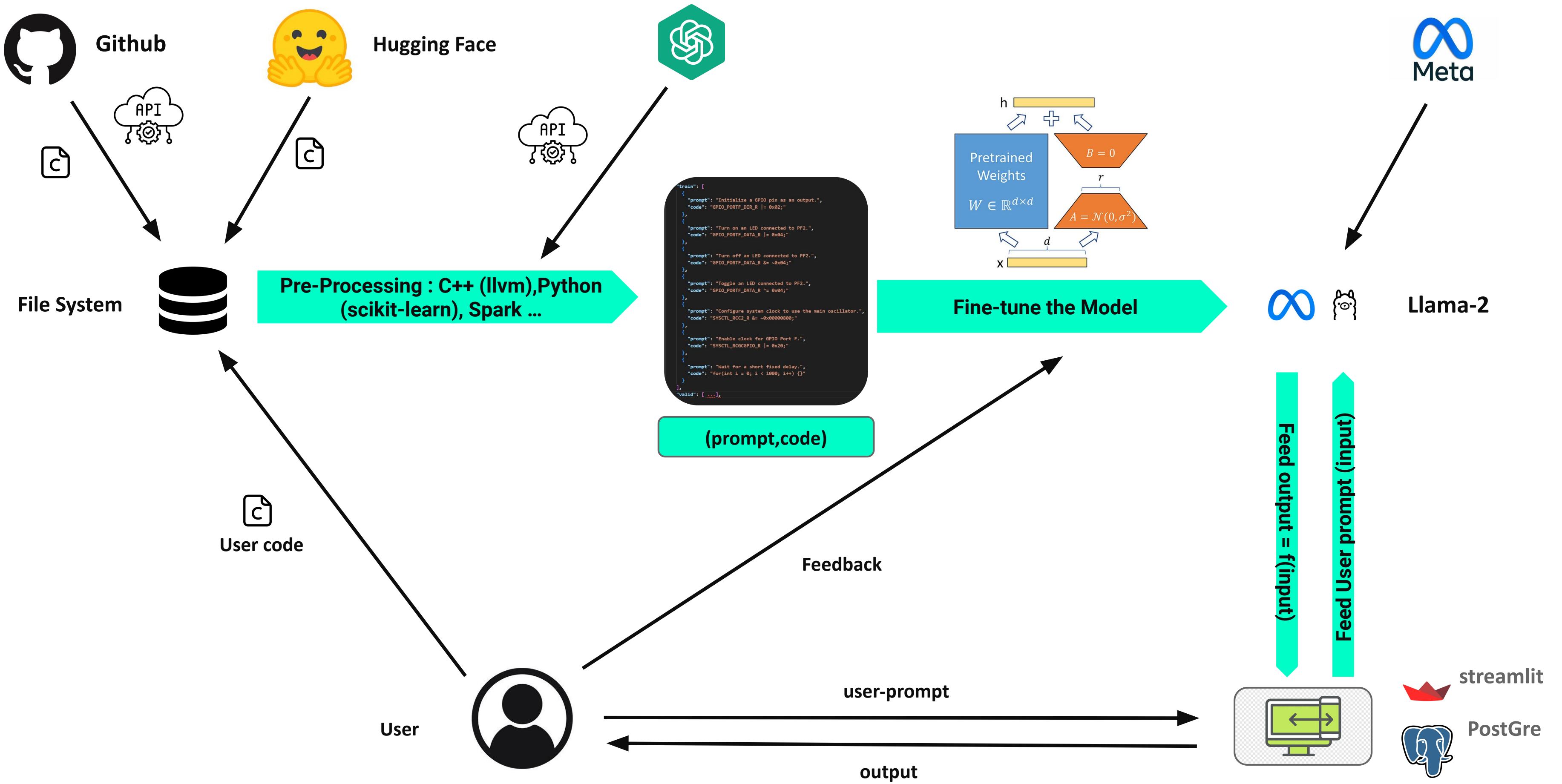
- Implementing an interface, a website — ZL
- Users database management — ZL
- Remote connexion to the Database — ZL
- Securing authentification system — ZL
- Fixing the chat history issue — ZL
- Feedback Section — ZL
- Adding animations to the interface — ZL

+ Ajouter une carte

Development & Versioning Tools

Development:

Python	Versatility, readability, and a rich ecosystem of libraries for all development modules.
C++	Efficient thanks to its low-level control to parse and manipulate code snippets effectively.
Spark	Data preprocessing and transformation tasks, harnessing its distributed computing capabilities to handle massive datasets efficiently.
Pytorch	Dynamic computational graph construction and a high degree of flexibility.
Streamlit	Hosting the chatbot interface, providing an intuitive and interactive platform.
PostGre	Robust database management system, facilitating efficient storage and retrieval of user data within the application.
CSS	Used to style the web application, enhancing user experience and interface aesthetics.



```
"train": [
  {
    "prompt": "Initialize a GPIO pin as an output.",
    "code": "GPIO_PORTF_DIR_R |= 0x02;"
  },
  {
    "prompt": "Turn on an LED connected to PF2.",
    "code": "GPIO_PORTF_DATA_R |= 0x04;"
  },
  {
    "prompt": "Turn off an LED connected to PF2.",
    "code": "GPIO_PORTF_DATA_R &= ~0x04;"
  },
  {
    "prompt": "Toggle an LED connected to PF2.",
    "code": "GPIO_PORTF_DATA_R ^= 0x04;"
  },
  {
    "prompt": "Configure system clock to use the main oscillator.",
    "code": "SYSCTL_RCC2_R &= ~0x00000800;"
  },
  {
    "prompt": "Enable clock for GPIO Port F.",
    "code": "SYSCTL_RCGCGPIO_R |= 0x20;"
  },
  {
    "prompt": "Wait for a short fixed delay.",
    "code": "for(int i = 0; i < 1000; i++) {}"
  }
],
"valid": [ ...],
```

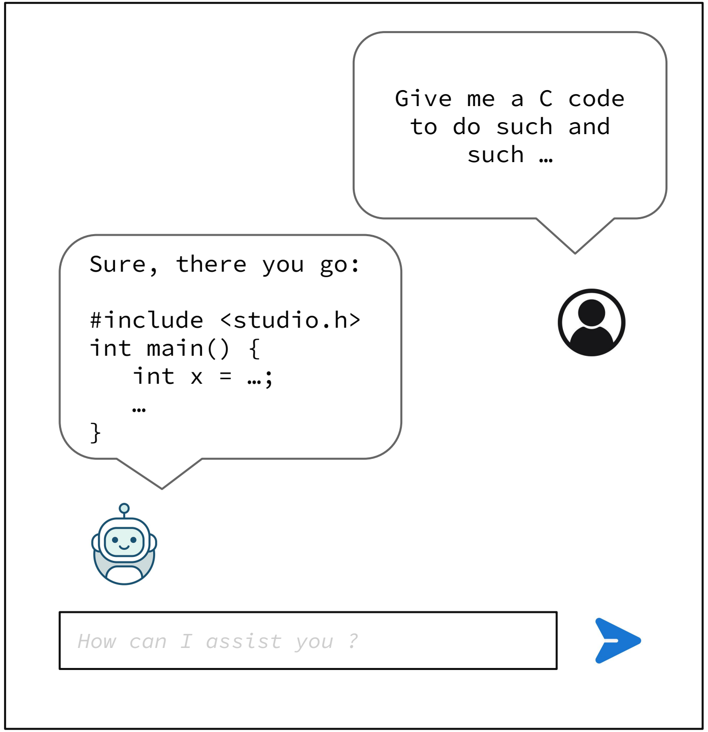
Versioning:

Github	❑ Allows for parallel development and smoother integration of changes into the main project.
Google Drive	❑ Flexibility with Google Colab which was used during Fine-tuning. ❑ Allows to manage and transfer large files. (the model’s configuration files)

Dataset building

Data Scraping:

First of all, what do we need ?



Give me a C code to do such and such …

Sure, there you go:

```
#include <studio.h>
int main() {
   int x = …;
   …
}
```

The AI must understand the prompt (the user’s request), and the code provided (the chatbot answer).

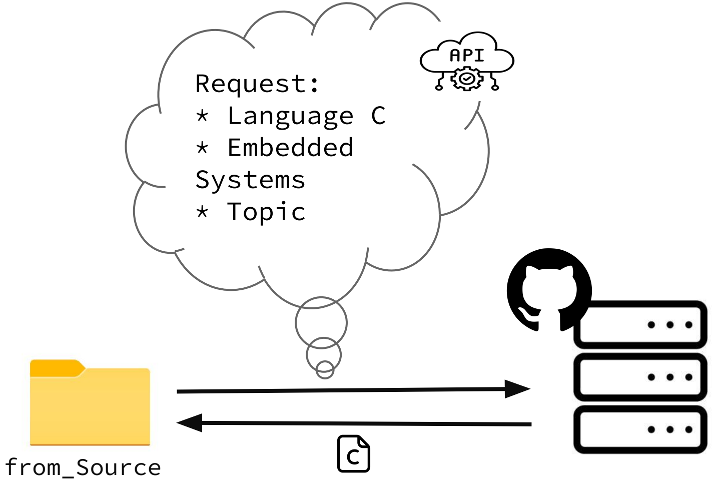
Our Dataset must be a set of coupled Prompts and corresponding codes.

Data Output form

(“write a C code that does such…”,”#include<studio.h>\nint main…”)

Code Snippets:

- Go for Open Source platforms: rich in code, no cost.
- Automating the downloading process: The use of GitHub’s Python API.
- We need a Storage System: The exploitation of an external Hard Drive.



Request:
⋆ Language C
⋆ Embedded Systems
⋆ Topic

Topics:

Embedded Networking

Real Time Operating Systems …

Prompts:

We need the prompt that would make an appropriate request for each code snippet. ChatGpt was of great assistance, we have no expertise in Embedded Systems. Again requires automation, can’t be done manually.

Data Pipelines & File System Architecture

Data Ingestion

Data ingestion is the process of integrating data into the service upon which the development in ongoing.

The initial was simple. Use a physical Storage system for data ingestion.

The development was carried on independently of the specifics of physical storage systems, all that was needed was the location of the input.

Fertile ground to migrate to cloud solutions, for more, speed flexibility, and fault tolerance.

DATA PREPROCESSING:

Having thousands of raw C code snippets, how can we clean up ?

- Deduplication: No need to have two codes that do the same thing.
- Code cleaning: Comments removal, empty lines removal… (comments can be used ?)
- Deriving more insights from each code: For more specific requests.
- Reconstruction of derived codes in the context of the father code.

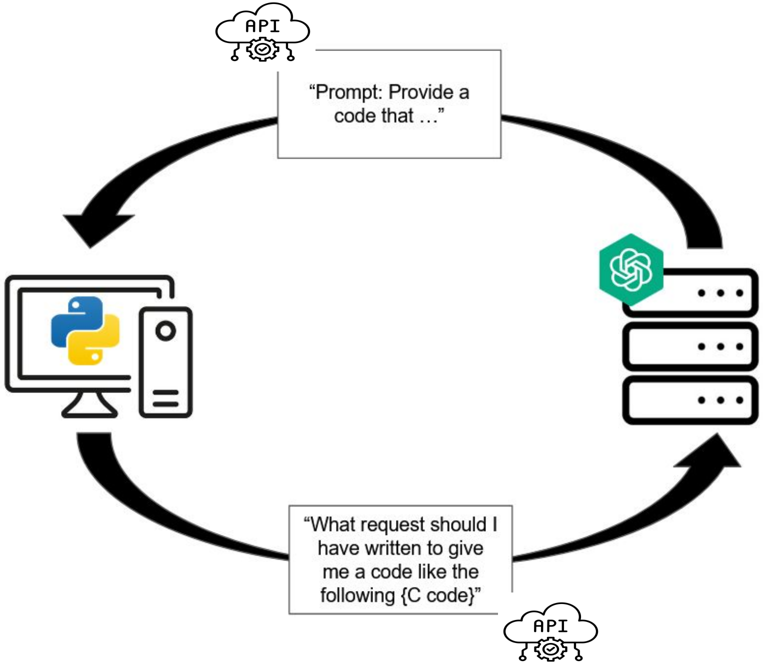
Data Transformation

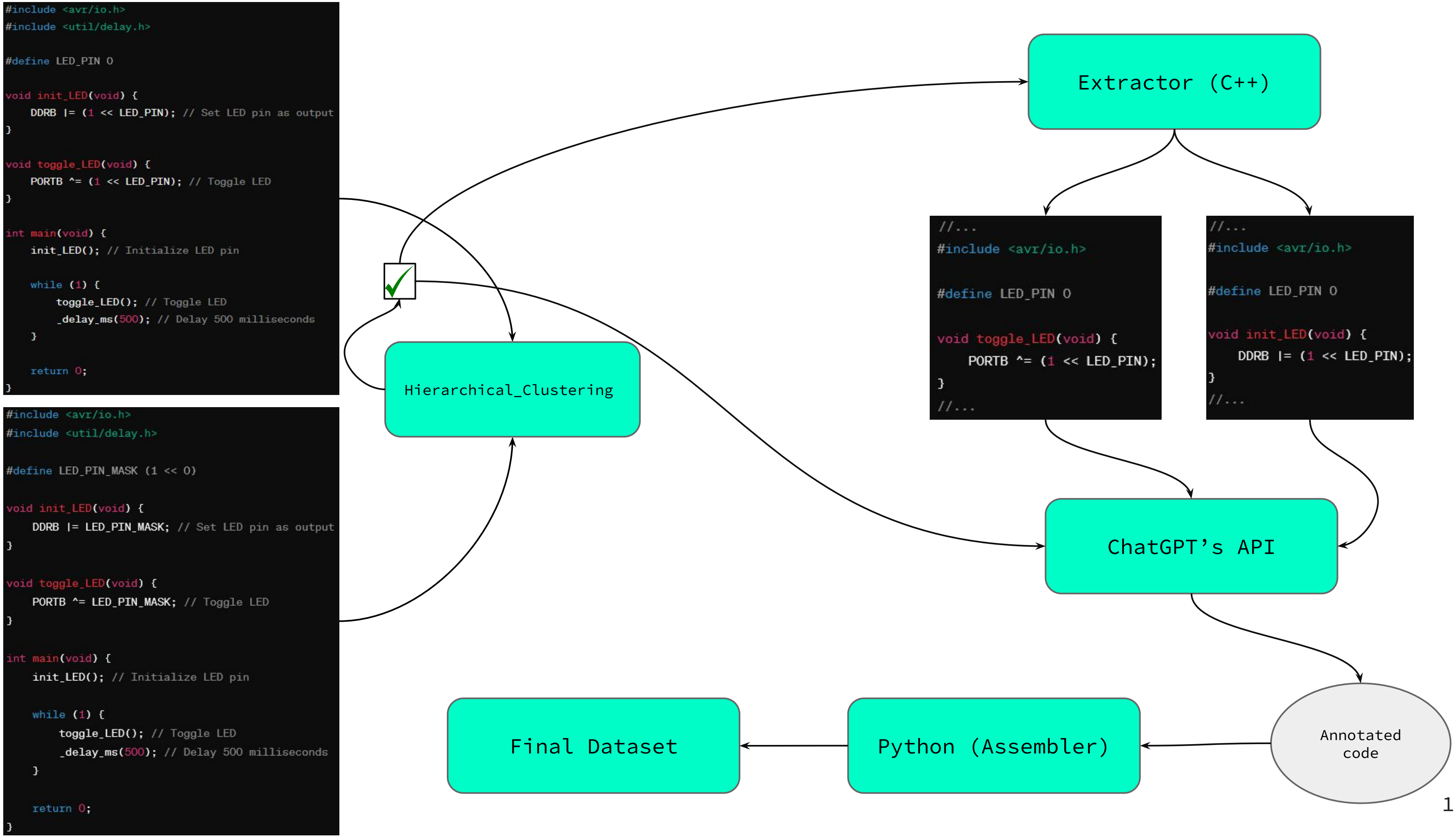
- C code:“““//…

```
unsigned int fec_golay2412_decode_symbol(unsigned int _sym_enc){

        if (_sym_enc >= (1<<15)) {

                fprintf(stderr,"error, fec_golay2412_decode(), input
        symbol too large\\n");

                 exit(1);

        }

        return 0;}

//…”””
```

- Prompt: (provided by chatGPT)

“Can you provide me with a function to decode a received symbol encoded with this code? I need to ensure that the input symbol size is within the correct range for processing.”


API
“Prompt: Provide a code that …”
“What request should I have written to give me a code like the following {C code}”
API



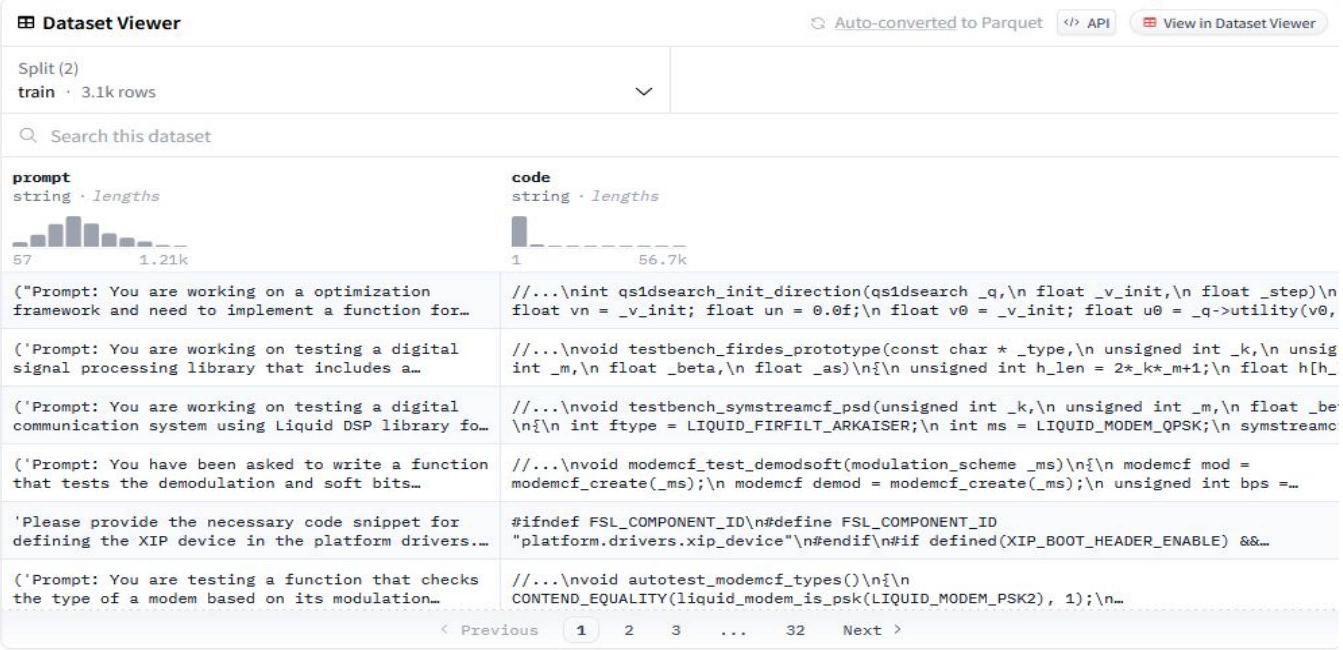
```
#include <avr/io.h>
#include <util/delay.h>

#define LED_PIN 0

void init_LED(void) {
    DDRB |= (1 << LED_PIN); // Set LED pin as output
}

void toggle_LED(void) {
    PORTB ^= (1 << LED_PIN); // Toggle LED
}

int main(void) {
    init_LED(); // Initialize LED pin

    while (1) {
        toggle_LED(); // Toggle LED
        _delay_ms(500); // Delay 500 milliseconds
    }

    return 0;
}
```

```
#include <avr/io.h>
#include <util/delay.h>

#define LED_PIN_MASK (1 << 0)

void init_LED(void) {
    DDRB |= LED_PIN_MASK; // Set LED pin as output
}

void toggle_LED(void) {
    PORTB ^= LED_PIN_MASK; // Toggle LED
}

int main(void) {
    init_LED(); // Initialize LED pin

    while (1) {
        toggle_LED(); // Toggle LED
        _delay_ms(500); // Delay 500 milliseconds
    }

    return 0;
}
```

```
//...
#include <avr/io.h>

#define LED_PIN 0

void toggle_LED(void) {
    PORTB ^= (1 << LED_PIN);
}
//...
```

```
//...
#include <avr/io.h>

#define LED_PIN 0

void init_LED(void) {
    DDRB |= (1 << LED_PIN);
}
//...
```

Data output



Split (2)
train · 3.1k rows

prompt string · *lengths* 57 1.21k	code string · *lengths* 1 56.7k
("Prompt: You are working on a optimization framework and need to implement a function for…	//...\nint qs1dsearch_init_direction(qs1dsearch _q,\n float _v_init,\n float _step)\n float vn = _v_init; float un = 0.0f;\n float v0 = _v_init; float u0 = _q->utility(v0,
('Prompt: You are working on testing a digital signal processing library that includes a…	//...\nvoid testbench_firdes_prototype(const char * _type,\n unsigned int _k,\n unsig int _m,\n float _beta,\n float _as)\n{\n unsigned int h_len = 2*_k*_m+1;\n float h[h_
('Prompt: You are working on testing a digital communication system using Liquid DSP library fo…	//...\nvoid testbench_symstreamcf_psd(unsigned int _k,\n unsigned int _m,\n float _be \n{\n int ftype = LIQUID_FIRFILT_ARKAISER;\n int ms = LIQUID_MODEM_QPSK;\n symstreamc
('Prompt: You have been asked to write a function that tests the demodulation and soft bits…	//...\nvoid modemcf_test_demodsoft(modulation_scheme _ms)\n{\n modemcf mod = modemcf_create(_ms);\n modemcf demod = modemcf_create(_ms);\n unsigned int bps =…
'Please provide the necessary code snippet for defining the XIP device in the platform drivers.…	#ifndef FSL_COMPONENT_ID\n#define FSL_COMPONENT_ID "platform.drivers.xip_device"\n#endif\n#if defined(XIP_BOOT_HEADER_ENABLE) &&…
('Prompt: You are testing a function that checks the type of a modem based on its modulation…	//...\nvoid autotest_modemcf_types()\n{\n CONTEND_EQUALITY(liquid_modem_is_psk(LIQUID_MODEM_PSK2), 1);\n…

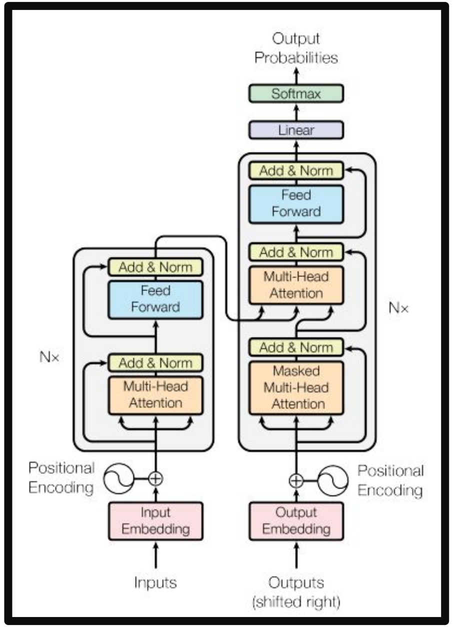
3879 instances of prompt code

FINE-TUNING

MODEL SELECTION: LLAMA-2

- ❏ Llama-2 is a family of large language model released by Meta, based on the Transformer Decoder-only architecture.
- ❏ Decoder-only means that the model utilizes only the decoder part of the Transformer architecture.
- ❏ There are 3 released versions by Meta with different parameter sizes: 7B, 13B and 70B.

SELECTED MODEL: Llama-2 7B because of limited Hardware resources.



The Transformer architecture

What is Fine-tuning ?

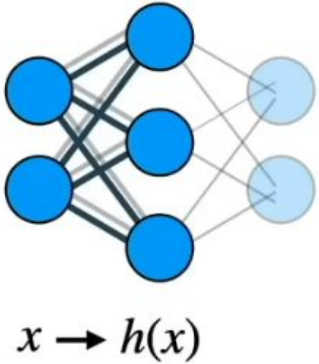
MAIN IDEA: Customize a pre-trained large language model for a particular task.

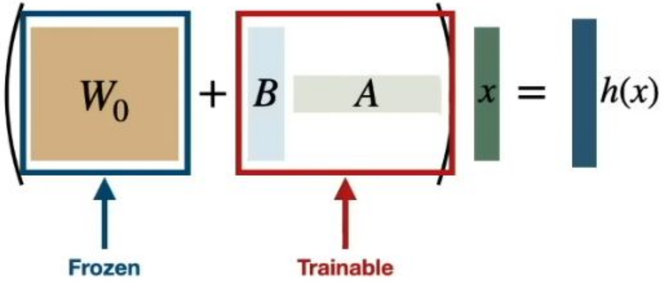
Full Fine-tuning: Train each layer of the model on the target dataset.	***Parameter efficient Fine-tuning (PEFT):*** Freeze some layers and only adjust the weights of a smaller set of the last layers.
✗ Very expensive computationally. (Needs to allocate around 112GB for a 7B model). ✗ Long training time.	✓ Less computational and storage costs. ✓ Less training time. ✓ Reach the same accuracy as a full fine-tuning.

PEFT approach: LoRA adapter

Fine-tunes the model by incorporating trainable rank decomposition matrices into each layer (specifically Feed-Forward layers).

$$h(x) = W_0 x + \Delta W x \qquad \Delta W = BA$$
$$= W_0 x + BAx$$





$$d = 1{,}000 \quad k = 1{,}000 \quad r = 2 \implies (d \times r) + (r \times k) = \mathbf{4{,}000}$$

trainable parameters

$$W_0, \Delta W \in R^{d \times k}$$
$$B \in R^{d \times r}$$
$$A \in R^{r \times k}$$
$$h(x) \in R^{d \times 1}$$

Fine-tuning Results:

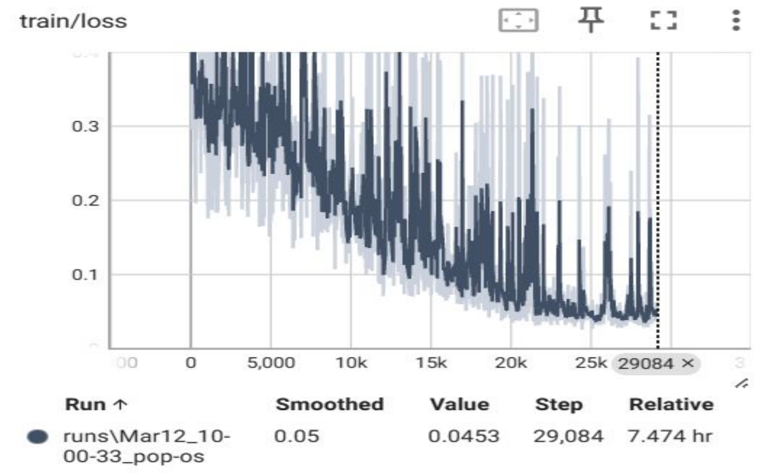
❑ Number of Epochs: 20 epochs.

❑ Batch size: 1

❑ Training time: approximately 19 hours.

❑ Training Loss: 0.045

Note: This fine-tuning requires at least a GPU with 24GB VRAM. It cannot be done with the free Google Colab version.

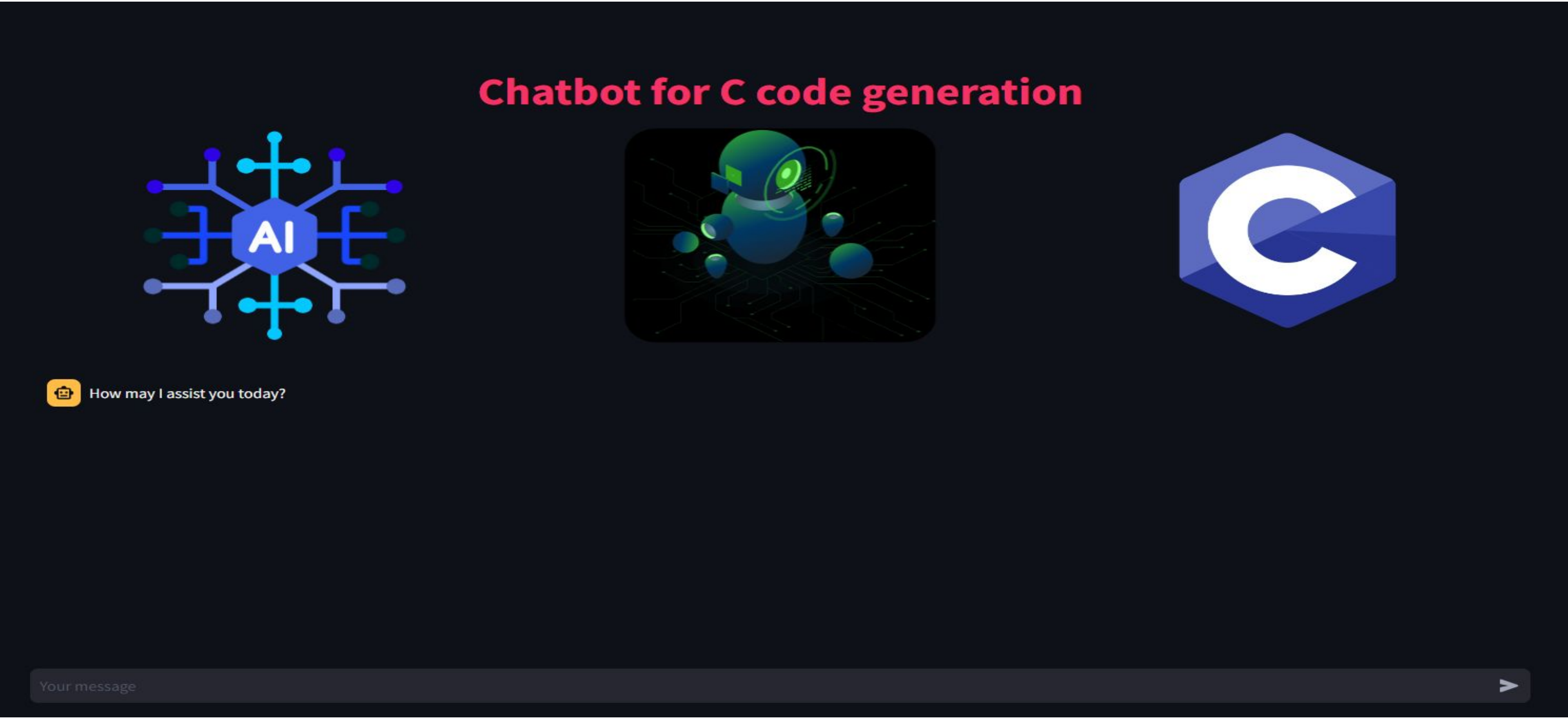


Run ↑	Smoothed	Value	Step	Relative
● runs\Mar12_10-00-33_pop-os	0.05	0.0453	29,084	7.474 hr

Interpretation: The fluctuations are common for small batch size choices. The overall trend which shows that the loss is decreasing over time means that the model is still learning.

Deployment

The model is deployed in a ***web server*** using ***Streamlit***.

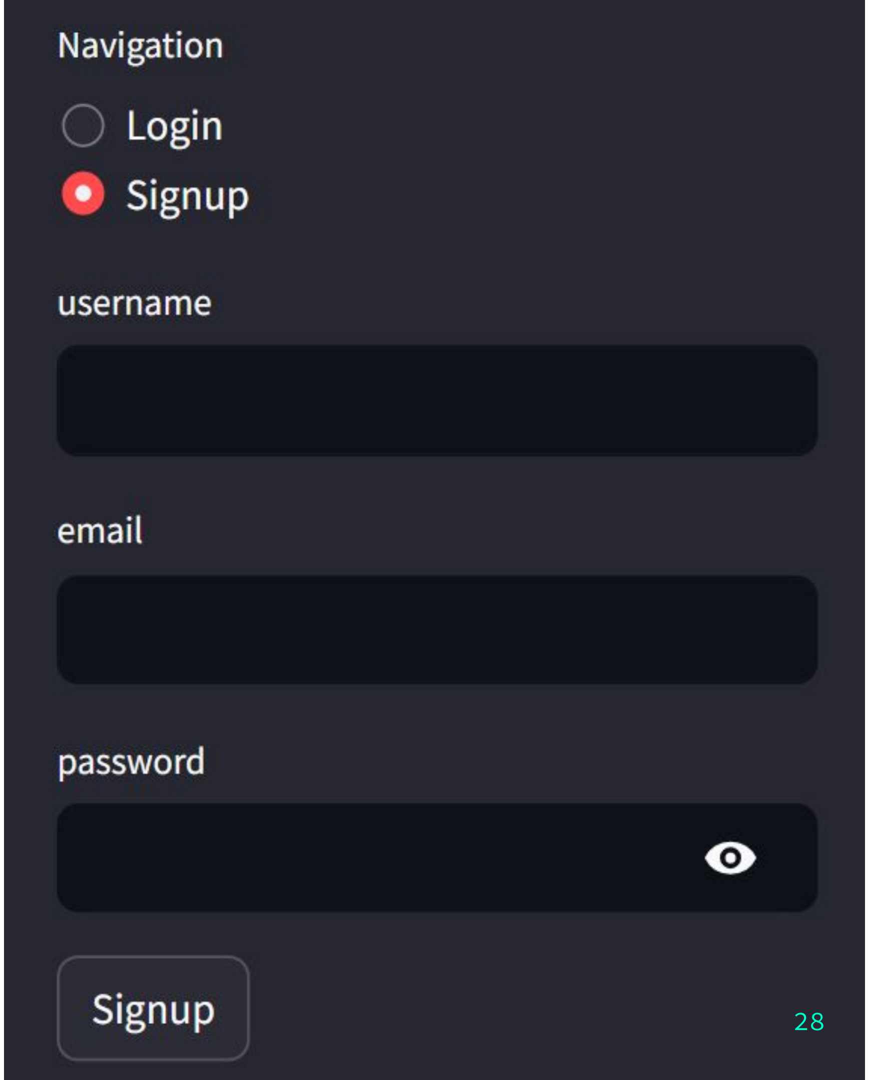


Chatbot for C code generation

How may I assist you today?

Creating an authentication system is essential:

- Enhance security
- Communicate with users
- Multiaccess of the app
- Subscription management

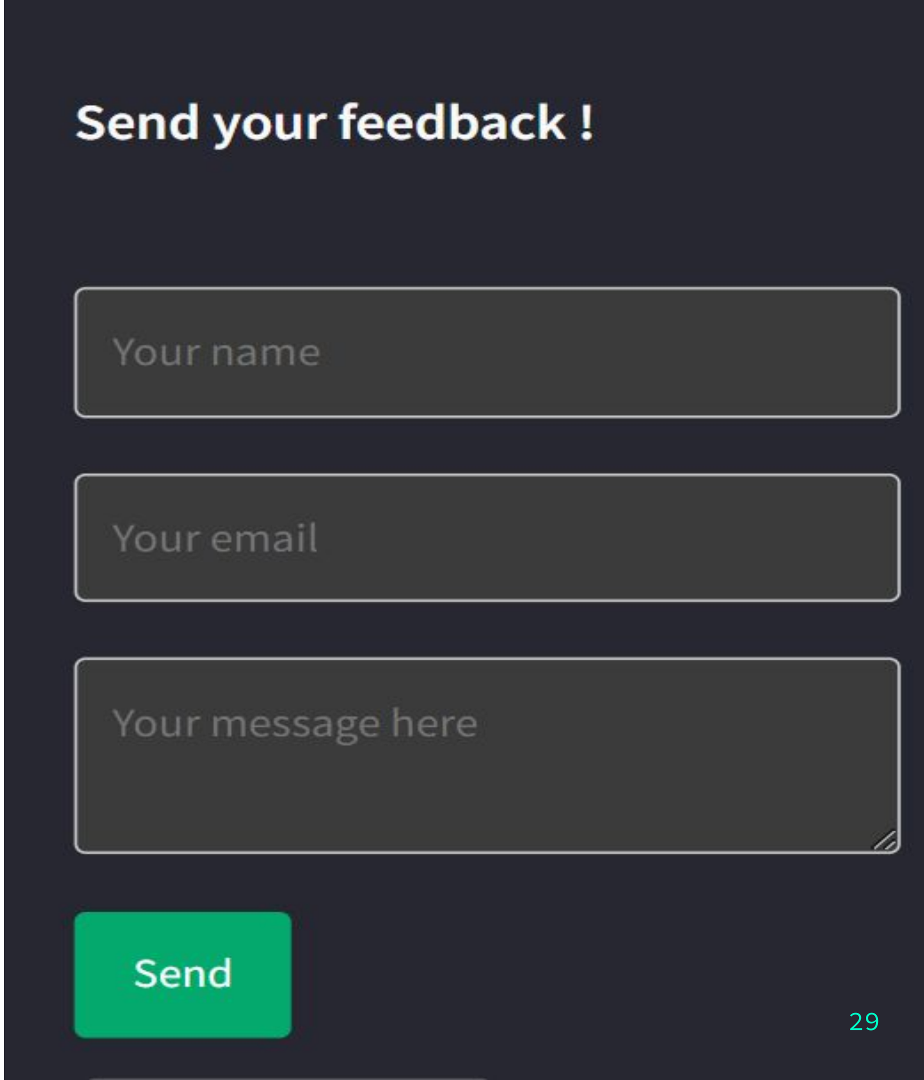


Navigation

○ Login

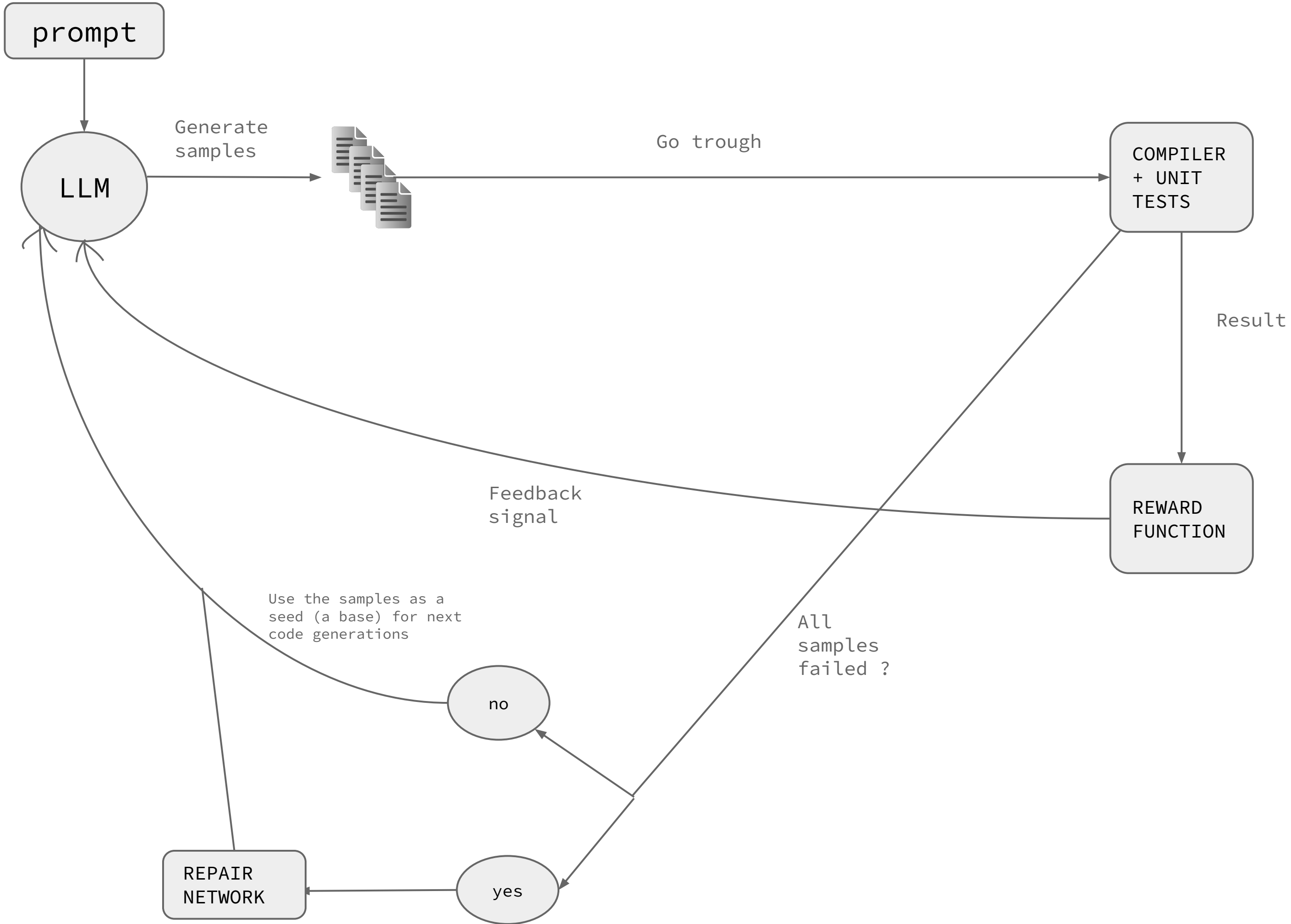
● Signup

Improve the user experience



Demonstration

Reinforcement learning integration


prompt
LLM
Generate samples
Go trough
COMPILER + UNIT TESTS
Result
REWARD FUNCTION
Feedback signal
All samples failed ?
Use the samples as a seed (a base) for next code generations
no
yes
REPAIR NETWORK

Conclusion

- Realised the ability to build the required dataset for our use case from scratch.
- Successful adaptation of the LoRA method to our model.
- Interactive Web Interface.
- Future improvement through Reinforcement learning.
- Soft skills: Communication, Adaptability, Flexibility, Initiative taking.

References

- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen: "LoRA: Low-Rank Adaptation of Large Language Models" https://arxiv.org/abs/2106.09685
- Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, Colin Raffel: "Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning” https://arxiv.org/abs/2205.05638
- https://www.entrypointai.com/blog/lora-fine-tuning/
- https://www.leewayhertz.com/parameter-efficient-fine-tuning/
- https://arxiv.org/pdf/2207.01780.pdf
- Getting Started with Streamlit for Data Science: Create and deploy Streamlit web applications from scratch in Python:Tyler Richards