

클라우드네이티브+AI 전문가 양성 .

개발자 3.0 시대

최재규

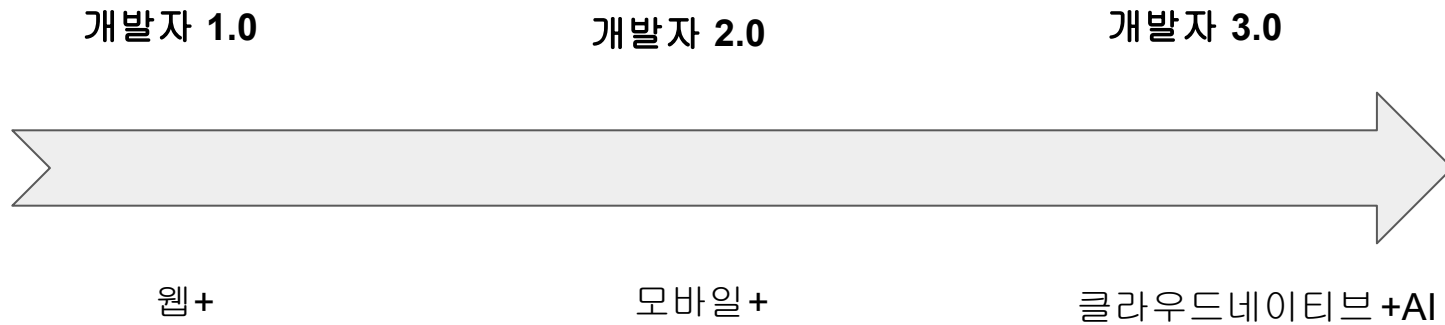
1. 개발자 3.0 이란?
2. 인공지능 기술의 진화
3. 클라우드 서비스의 진화
4. 어떤 개발자를 양성할 것인가?
5. 다음 시간에는...

클라우드네이티브+AI 전문가 양성 .

1. 개발자 3.0 이란?

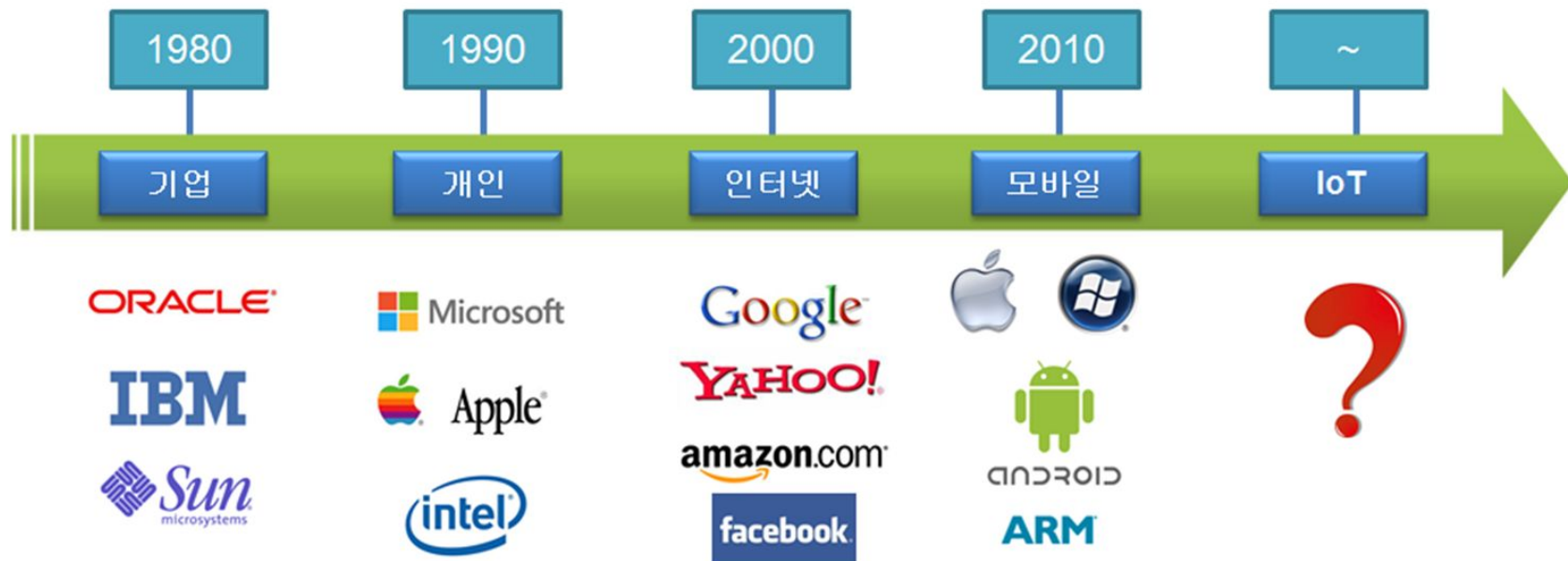
개발자 세대 구분

시장의 변화에 따라 개발자 정의가 변화되고 있음



기업 환경의 변화

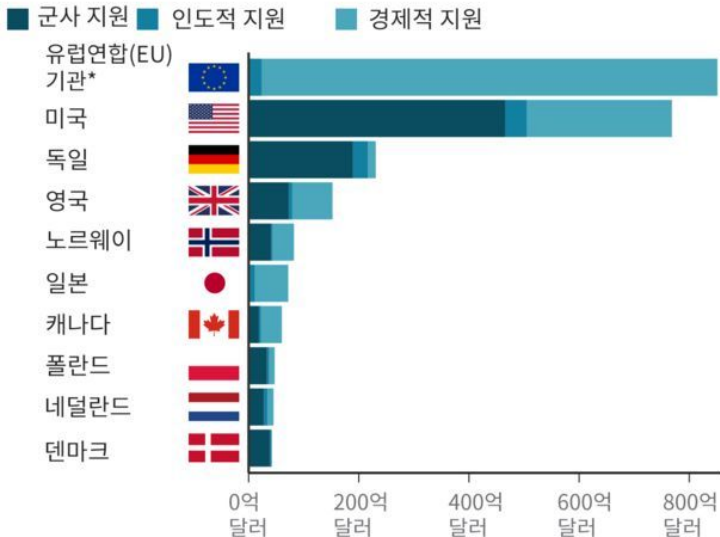
기업 전산화, 개인PC, 인터넷 서비스, 모바일 서비스, 초 개인화...



유럽과 중동에서 전쟁 발발...

EU와 미국은 가장 많은 우크라이나 원조 약속을 했다

2023년 8월까지 가장 많은 원조 약속을 한 10개국



*EU 이사회 및 EU 집행위원회가 약속한 지원액만 포함

출처: 독일 '킬 세계경제연구소'

B B C



김영은 기자 20231010

전쟁의 최대 수혜국 미국...

세계 10대 석유 생산국 하루 생산량 단위: 배럴, 2022년 5월 기준.

괄호는 세계 생산 비중(%) ★는 OPEC+ 국가들



자료: 미국 에너지정보청(EIA)

세계 밀 수출 순위

단위: t, 2020년 기준 수출량



자료: 유엔식량농업기구(FAO)

The JoongAng

혼란스러운 국제 정세

전쟁의 최대 수혜국 미국...



로고	기업	국가	시가총액 USD	시가총액 KRW
	애플 		24791 억달러	2856 조원
	마이크로소프트 		21816 억달러	2513 조원
	사우디 아람코 		18633 억달러	2146 조원
	아마존 		18441 억달러	2124 조원
	구글 		18157 억달러	2091 조원
	페이스북 		10485 억달러	1208 조원
	텐센트 		6558 억달러	755 조원
	버크셔 해서웨이 		6367 억달러	733 조원
	테슬라 		6197 억달러	714 조원
	알리바바 		5599 억달러	645 조원

- 2022년 기준

기업 환경의 변화

기술 혁신을 통한 산업 지형의 변화



기술 혁신을 통한 산업 지형의 변화



클라우드네이티브+AI 전문가 양성 .

2. 인공지능 기술의 진화



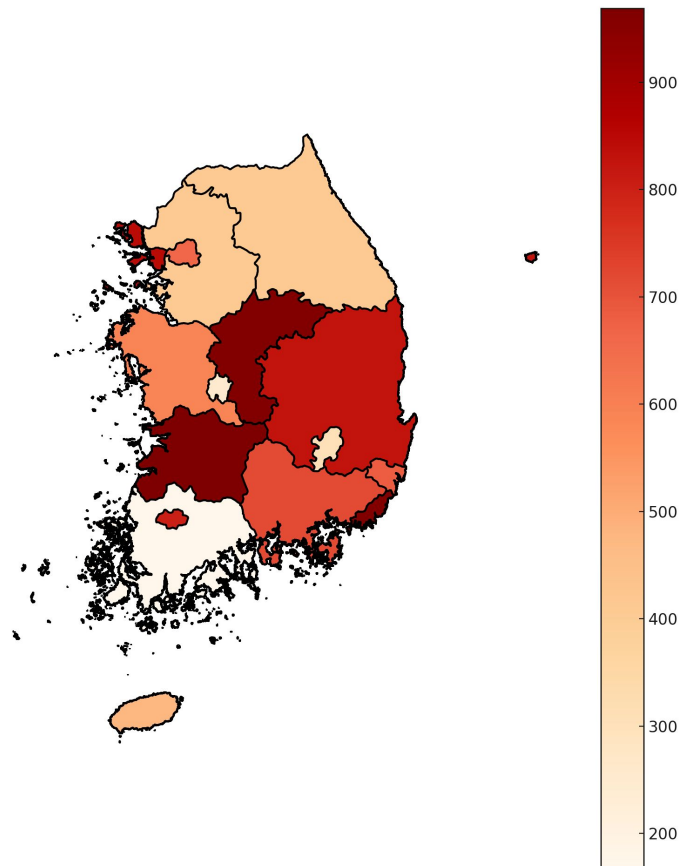
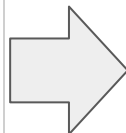
1. $\left(\frac{4}{2^{\sqrt{2}}}\right)^{2+\sqrt{2}}$ 의 값은? [2점]

- ① $\frac{1}{4}$ ② $\frac{1}{2}$ ③ 1 ④ 2 ⑤ 4



거대 언어 모델

1	광역시도(한글)	광역시도(영어)	매출(단위: 백만원)
2	부산	Busan	960
3	충청북도	Chungcheongbuk-do	962
4	충청남도	Chungcheongnam-do	589
5	대구	Daegu	303
6	대전	Daejeon	253
7	강원도	Gangwon-do	407

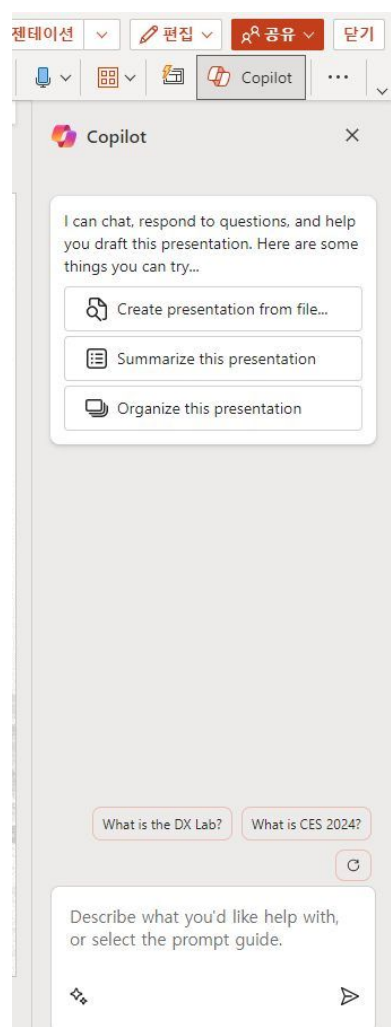


거대 언어 모델

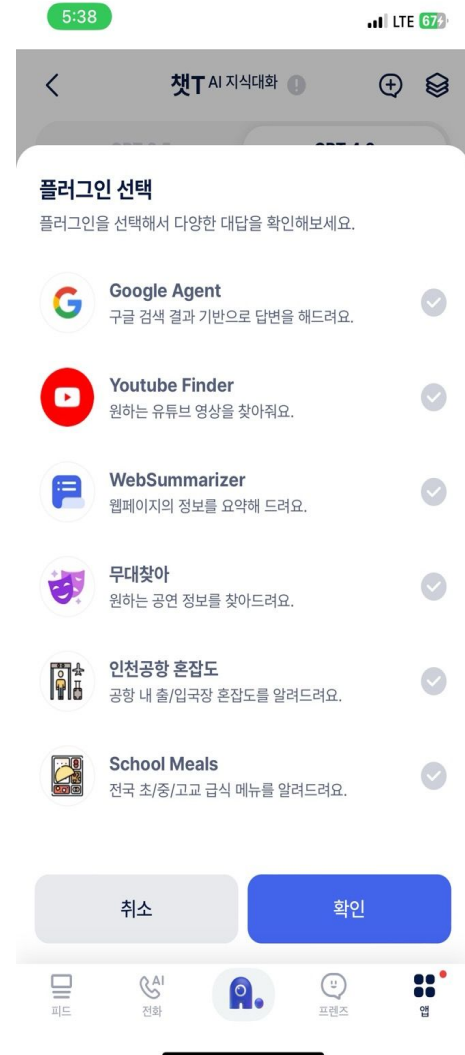
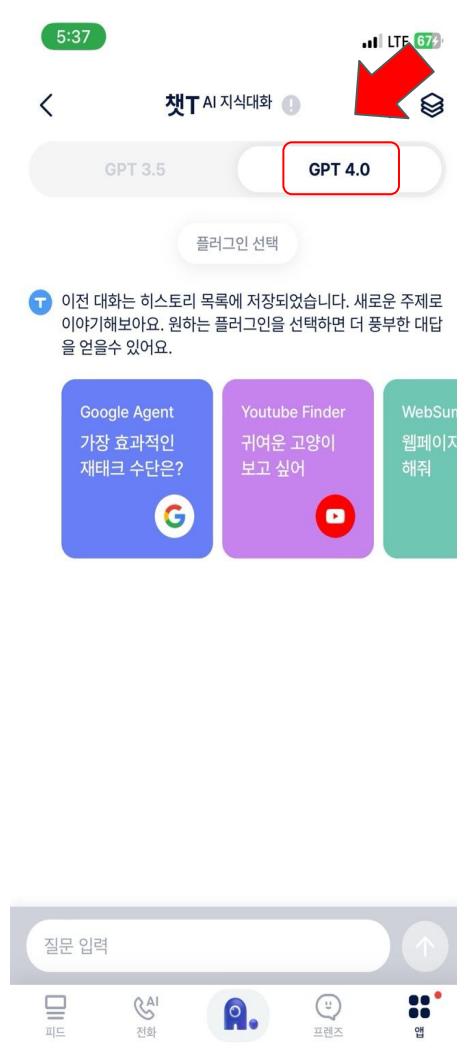
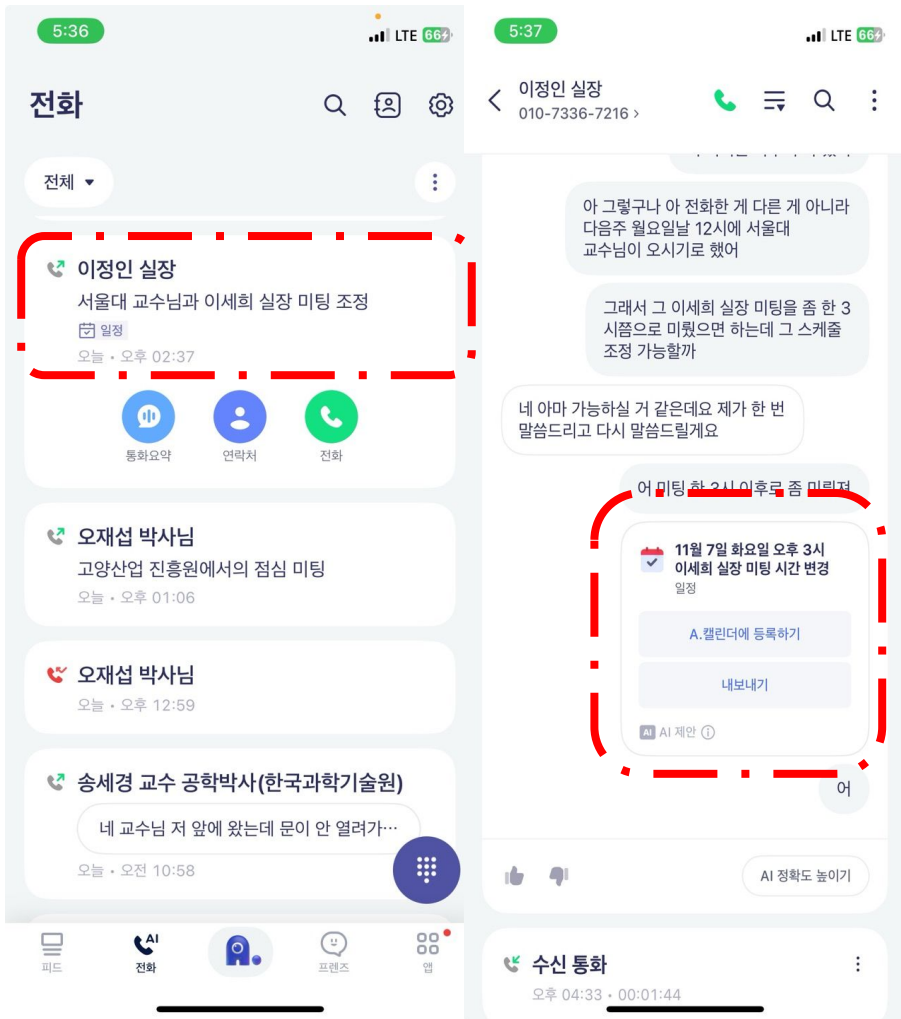
세계지도를 그리고 한국을 표시해줘.
아시아 지도를 그리고 한국을 표시해줘.
한국 지도를 그리고 서울을 표시해줘.
한국의 행정구역을 지도로 그려줘. 지도를 그려줘.
광역시도를 표시하고 경계선을 굵게 그려줘.
이 데이터의 매출을 지도에 표시해줘. 단위는 빼줘.

서울특별시의 인구변화를 **GIF** 애니메이션 그래프로 만들어줘.
GIF 애니메이션에서 모든 프레임을 추출해서 하나의 이미지로 보여줘.
이미지안에 프레임은 가로로 **3**개씩 들어가게 해줘.
2010년 값을 기준으로 **Y**축과 **X**축을 고정해서 다시 만들어줘.

텍스트를 추출해서 알려줘.
웹사이트 주소, 휴대폰 번호, 이메일을 추출해서 엑셀로 정리해줘.
각 이미지의 해상도와 용량을 엑셀 열에 추가해줘

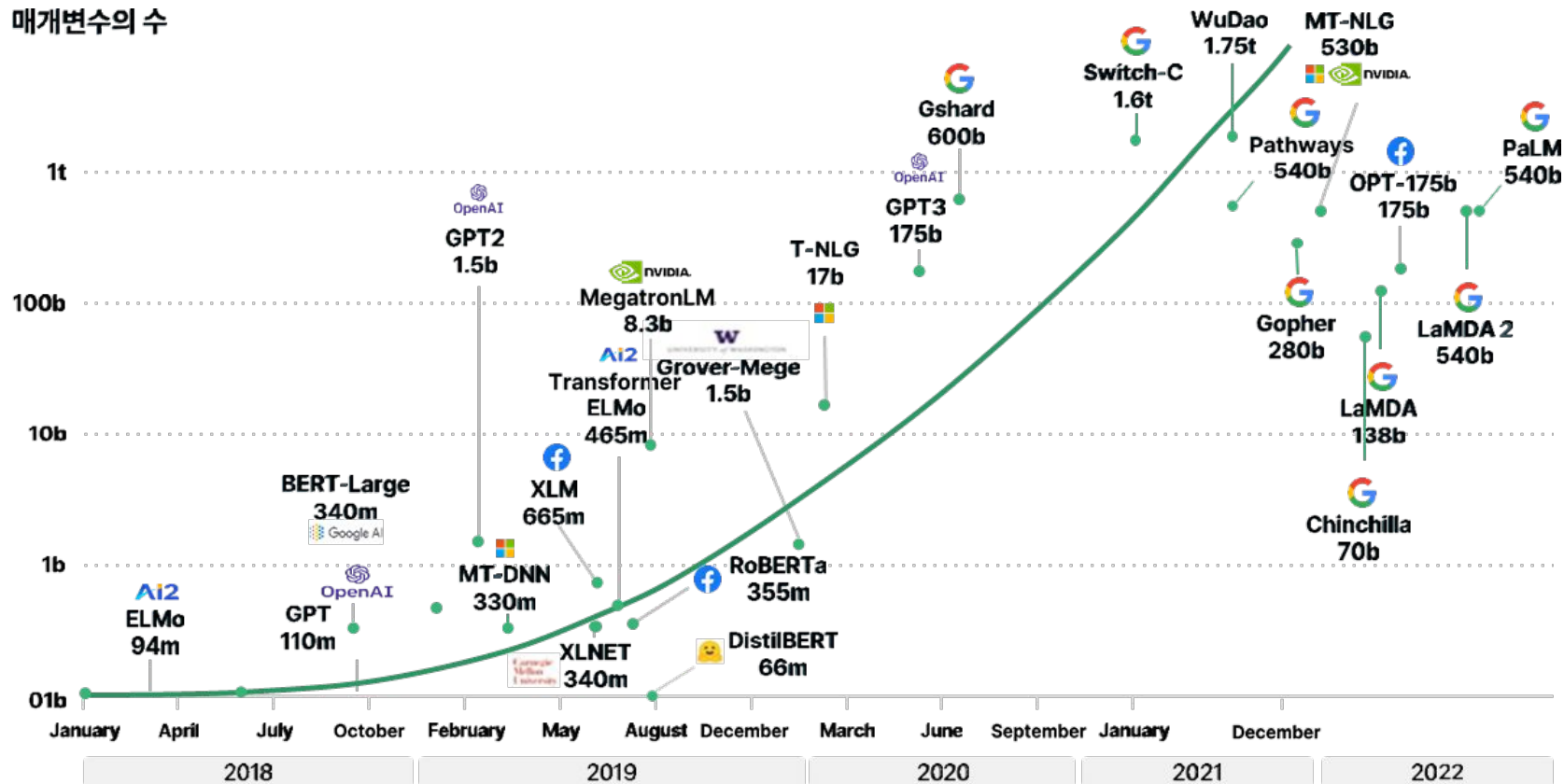






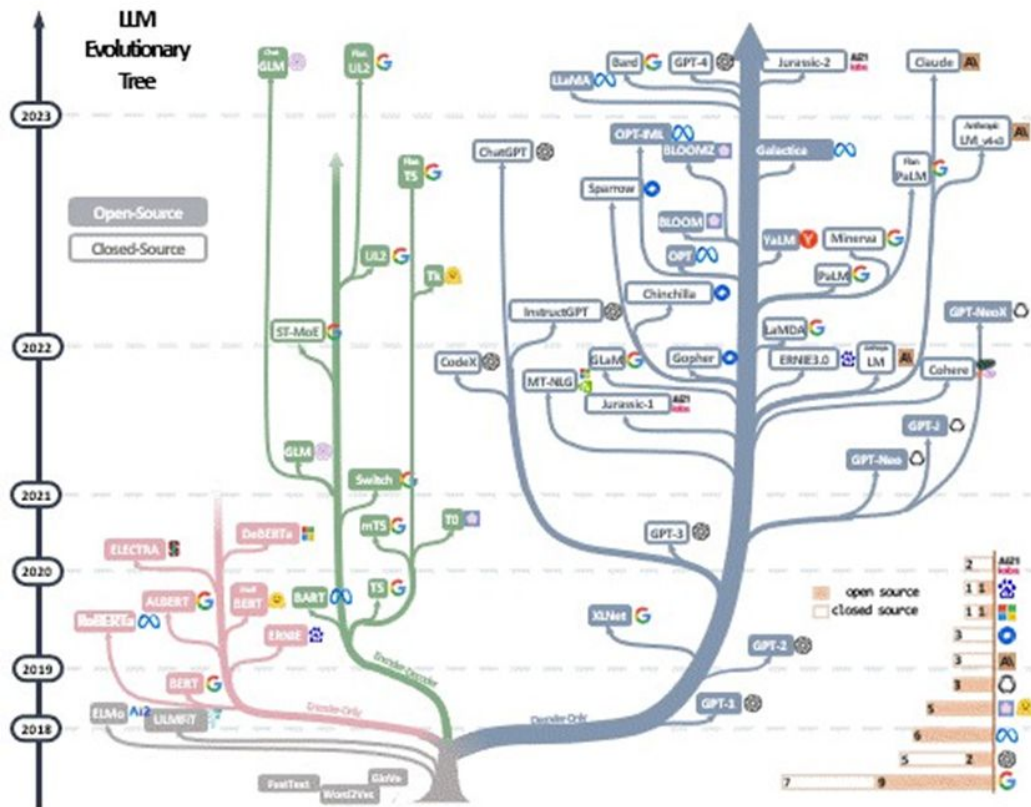
거대 언어 모델

매개변수의 수



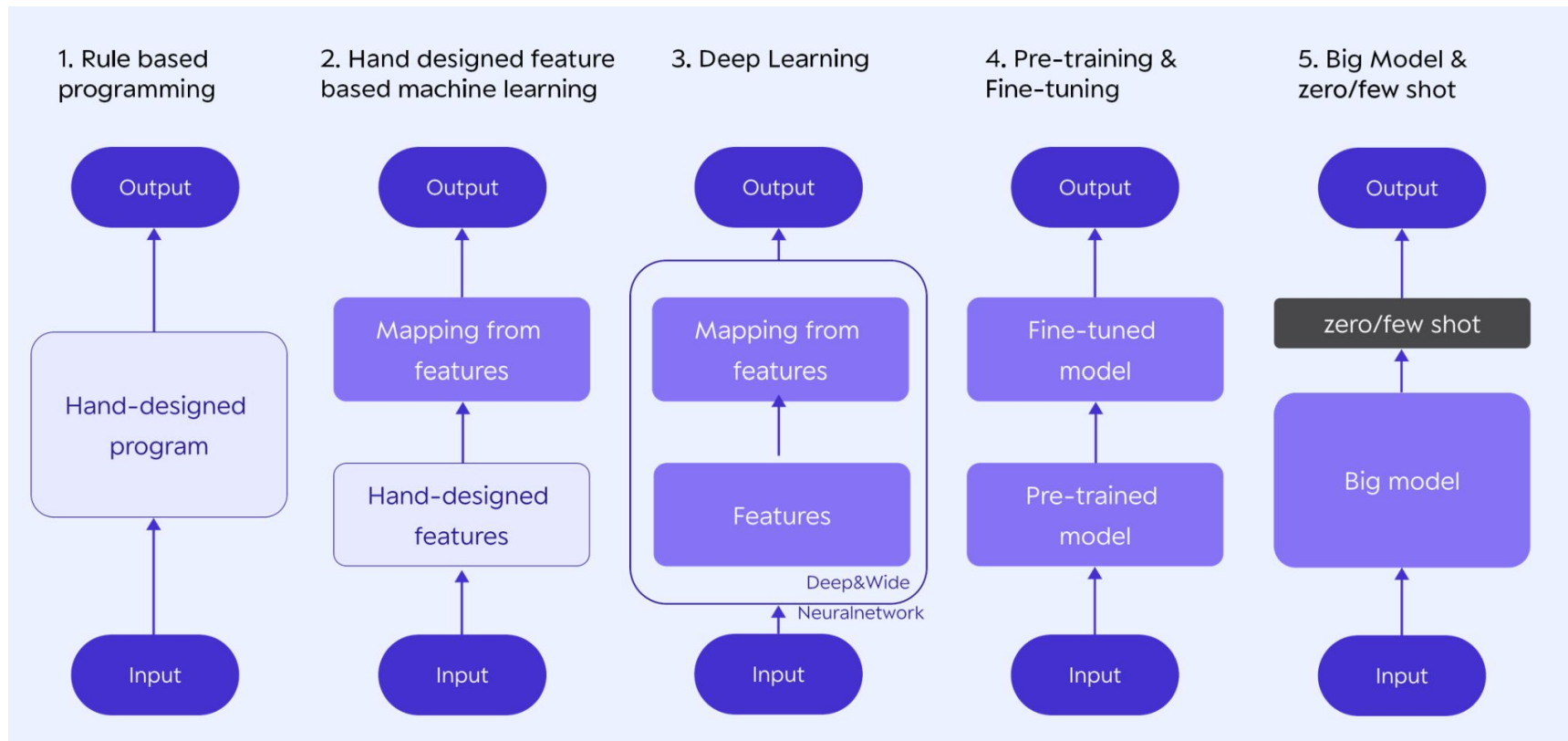
Gadi Singer, 2021, Google, 2022

언어 모델의 폭발적 진화



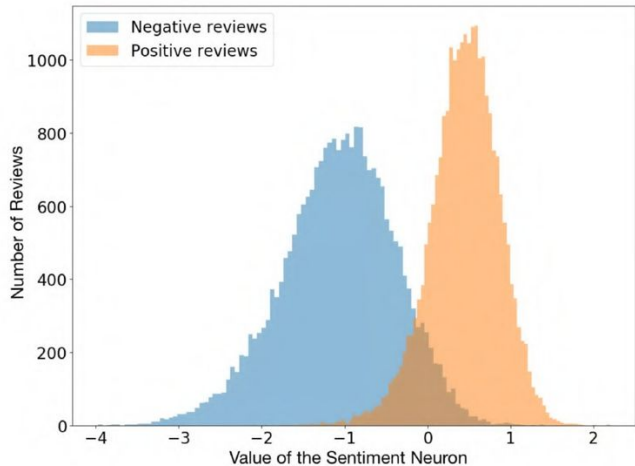
- 2018년
 - 트랜스포머 아키텍처 이후 급속한 발전
- 2020년
 - 거대 언어 모델의 특이점들 발견
- 2022년
 - 거대 언어 모델의 대중화 서비스 시작
 - ChatGPT... 더이상 말이 필요한가?

AI 개발 세대 분류



Emergence 2017년 4월

- OpenAI에서 Alec Radford가 언어 모델을 RNN으로 만
들고 있었음 (참고: Alec Radford는 GPT 논문의 1저자)
- 그런데, 특정 뉴런이 *감성 분석*을 하고 있음을 발견?! →
- 언어 모델링을 하다보면 의도하지 않은 능력이 생기게 되는
게 아닐까? → **"Emergence"**



<https://openai.com/research/unsupervised-sentiment-neuron>

Transformer 2017년 6월

- Transformer는 RNN, CNN과 유사한 아키텍처의 일종
- "Attention"이란 항목과 항목 사이의 연관성
- Transformer는 여러모로 성능이 좋았음! 계산 효율이 기존의 RNN 등에 비해 대단히 높았던게 이점, 게다가 결과의 품질도 더 좋음 → 이후 비전, 추천, 바이오 등 다른 모든 분야에서 쓰는 기술이 됨
- Alec Radford도 자연스럽게 Transformer를 가지고 실험하기 시작함

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

Lukasz Kaiser*
Google Brain
lukaszkaiser@google.com

Illia Polosukhin* ‡
illia.polosukhin@gmail.com

GPT 2018년 6월

- Alec Radford와 동료인 Ilya Sutskever 등이 RNN에서 Transformer로 넘어가며 출판 (참고: Ilya Sutskever를 어디에서 많이 들어봤다구요? AlexNet (이미지넷 우승작), TensorFlow (딥러닝 프레임워크), Dropout (학습 방법론), Seq2seq (NMT의 기원), AlphaGo 등의 저자)
- "*Generative pretraining* (GP)"을 하는 Transformer (T)
- *Pretraining-finetuning 패러다임*의 대표적인 논문:
 - 큰 스케일에서 언어 모델링을 통해 사전학습 모델을 만들고,
 - 이 모델을 파인튜닝하면 다양한 NLP 태스크에서 좋은 성능을 보이더라

Improving Language Understanding by Generative Pre-Training

Alec Radford
OpenAI
alec@openai.com

Karthik Narasimhan
OpenAI
karthikn@openai.com

Tim Salimans
OpenAI
tim@openai.com

Ilya Sutskever
OpenAI
ilyasu@openai.com

Abstract

Natural language understanding comprises a wide range of diverse tasks such as textual entailment, question answering, semantic similarity assessment, and document classification. Although large unlabeled text corpora are abundant, labeled data for learning these specific tasks is scarce, making it challenging for discriminatively trained models to perform adequately. We demonstrate that large gains on these tasks can be realized by *generative pre-training* of a language model on a diverse corpus of unlabeled text, followed by *discriminative fine-tuning* on each specific task. In contrast to previous approaches, we make use of task-aware input transformations during fine-tuning to achieve effective transfer while requiring minimal changes to the model architecture. We demonstrate the effectiveness of our approach on a wide range of benchmarks for natural language understanding. Our general task-agnostic model outperforms discriminatively trained models that use architectures specifically crafted for each task, significantly improving upon the state of the art in 9 out of the 12 tasks studied. For instance, we achieve absolute improvements of 8.9% on commonsense reasoning (Stories Cloze Test), 5.7% on question answering (RACE), and 1.5% on textual entailment (MultiNLI).

GPT-2 2019년 2월

- Ilya Sutskever가 오랫동안 주장했던 믿음은 "데이터를 많이 부어 넣고 모델의 크기를 키우면 신기한 일들이 일어난다"였음
- Transformer 전까지는 큰 모델의 학습을 어떻게 할 것인가가 문제였는데, Transformer가 계산 효율이 높아 스케일링에 유리
- 모델을 키우고 (117M → 1.5B) 데이터를 왕창 부음 (4GB → 40GB) → GPT-2의 탄생!
- 생성에 너무 뛰어나서 해당 모델이 가짜 정보를 다량 생성할 위험성이 크다고 판단, OpenAI는 GPT-2를 공개하지 않는다고 함 → ClosedAI 아니냐며 조롱을 받음
- 언어 생성을 아주 잘하게 될 뿐더러 emergence!가 또 보임

Language Models are Unsupervised Multitask Learners

GPT-2

Alec Radford^{*1}

Jeffrey Wu^{*1}

Rewon Child¹

David Luan¹

Dario Amodei^{**1}

Ilya Sutskever^{**1}

Abstract

Natural language processing tasks, such as question answering, machine translation, reading comprehension, and summarization, are typically approached with supervised learning on task-specific datasets. We demonstrate that language models begin to learn these tasks without any explicit supervision when trained on a new dataset of **millions of webpages called WebText**. When conditioned on a document plus questions, the answers generated by the language model reach 55 F1 on the CoQA dataset - matching or exceeding the performance of 3 out of 4 baseline systems without using the 127,000+ training examples. The capacity of the language model is essential

competent generalists. We would like to move towards more general systems which can perform many tasks - eventually without the need to manually create and label a training dataset for each one.

The dominant approach to creating ML systems is to collect a dataset of training examples demonstrating correct behavior for a desired task, train a system to imitate these behaviors, and then test its performance on independent and identically distributed (IID) held-out examples. This has served well to make progress on narrow experts. But the often erratic behavior of captioning models (Lake et al., 2017), reading comprehension systems (Jia & Liang, 2017), and image classifiers (Alcorn et al., 2018) on the diversity and variety of possible inputs highlights some of the shortcomings of this approach.

"Emergence": Zero-shot learning

- GPT2는 방대한 데이터를 기반으로 세상에 대해 많이 배운 모델
- Zero-shot learning
 - 예시를 전혀 보지 않고
 - 모델 업데이트 없이 새로운 태스크를 수행
 - "Unsupervised multitask learners"
 - "하나를 가르쳤는데 열을 아네"

- 독해, 번역, 요약, Q&A 등에 대해 zero-shot 능력이 꽤 있음!
- Zero-shot인데도 특정 태스크는 기존의 SOTA* 모델들을 짓눌러버림

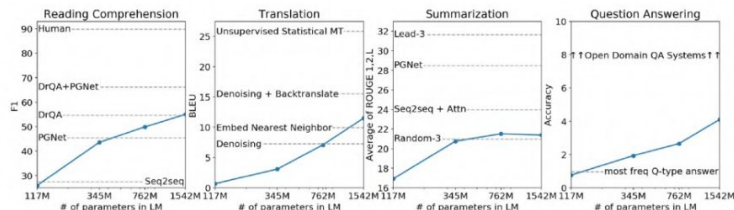


Figure 1. Zero-shot task performance of WebText LMs as a function of model size on many NLP tasks. Reading Comprehension results are on CoQA (Reddy et al., 2018), translation on WMT-14 Fr-En (Artetxe et al., 2017), summarization on CNN and Daily Mail (See et al., 2017), and Question Answering on Natural Questions (Kwiatkowski et al., 2019). Section 3 contains detailed descriptions of each result.

	LAMBADA (PPL)	LAMBADA (ACC)	CBT-CN (ACC)	CBT-NE (ACC)	WikiText2 (PPL)	PTB (PPL)	enwik8 (BPB)	text8 (BPB)	WikiText103 (PPL)	1BW (PPL)
SOTA	99.8	59.23	85.7	82.3	39.14	46.54	0.99	1.08	18.3	21.8
117M	35.13	45.99	87.65	83.4	29.41	65.85	1.16	1.17	37.50	75.20
345M	15.60	55.48	92.35	87.1	22.76	47.33	1.01	1.06	26.37	55.72
762M	10.87	60.12	93.45	88.0	19.93	40.31	0.97	1.02	22.05	44.575
1542M	8.63	63.24	93.30	89.05	18.34	35.76	0.93	0.98	17.48	42.16

Table 3. Zero-shot results on many datasets. No training or fine-tuning was performed for any of these results. PTB and WikiText-2 results are from (Gong et al., 2018). CBT results are from (Bajgar et al., 2016). LAMBADA accuracy result is from (Hoang et al., 2018) and LAMBADA perplexity result is from (Grave et al., 2016). Other results are from (Dai et al., 2019).

SOTA란 state-of-the-art, 즉 현존하는 제일 좋은 모델. 볼드체가 더 좋은 점수. ACC제외하고는 낮을수록 좋음.

GPT-3 2020년 6월

- 여기서 한 번 더 크기를 키운 것이 GPT-3 (aka. davinci)
 - 모델: 1.5B → 175B
 - 데이터: 40GB → 600GB+
- 많은 데이터로 pretraining해서 더욱 놀라운 **생성 능력**을 갖추게 됨
- 역시 여러 측면으로 "emergence"를 확인
 - 지식을 포함? (world knowledge)
 - 학습 없이 태스크를 배우는 능력? ("few-shot learners")

Language Models are Few-Shot Learners

Tom B. Brown*	Benjamin Mann*	Nick Ryder*	Melanie Subbiah*	
Jared Kaplan†	Prafulla Dhariwal	Arvind Neelakantan	Pranav Shyam	Girish Sastry
Amanda Askell	Sandhini Agarwal	Ariel Herbert-Voss	Gretchen Krueger	Tom Henighan
Rewon Child	Aditya Ramesh	Daniel M. Ziegler	Jeffrey Wu	Clemens Winter
Christopher Hesse	Mark Chen	Eric Sigler	Mateusz Litwin	Scott Gray
Benjamin Chess		Jack Clark	Christopher Berner	
Sam McCandlish	Alec Radford	Ilya Sutskever	Dario Amodei	

OpenAI

"Emergence": In-context learning

- Few-shot도 모델 파인튜닝 없이 되네...?
 - 프롬프트에 예시 몇 개("few-shot")를 넣어주면
 - 모델 업데이트 없이 새로운 태스크를 수행



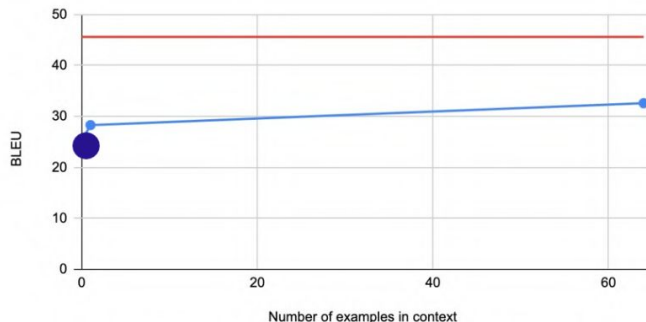
"Emergence": In-context learning

- Few-shot도 모델 파인튜닝 없이 되네...?
 - 프롬프트에 예시 몇 개("few-shot")를 넣어주면
 - 모델 업데이트 없이 새로운 태스크를 수행

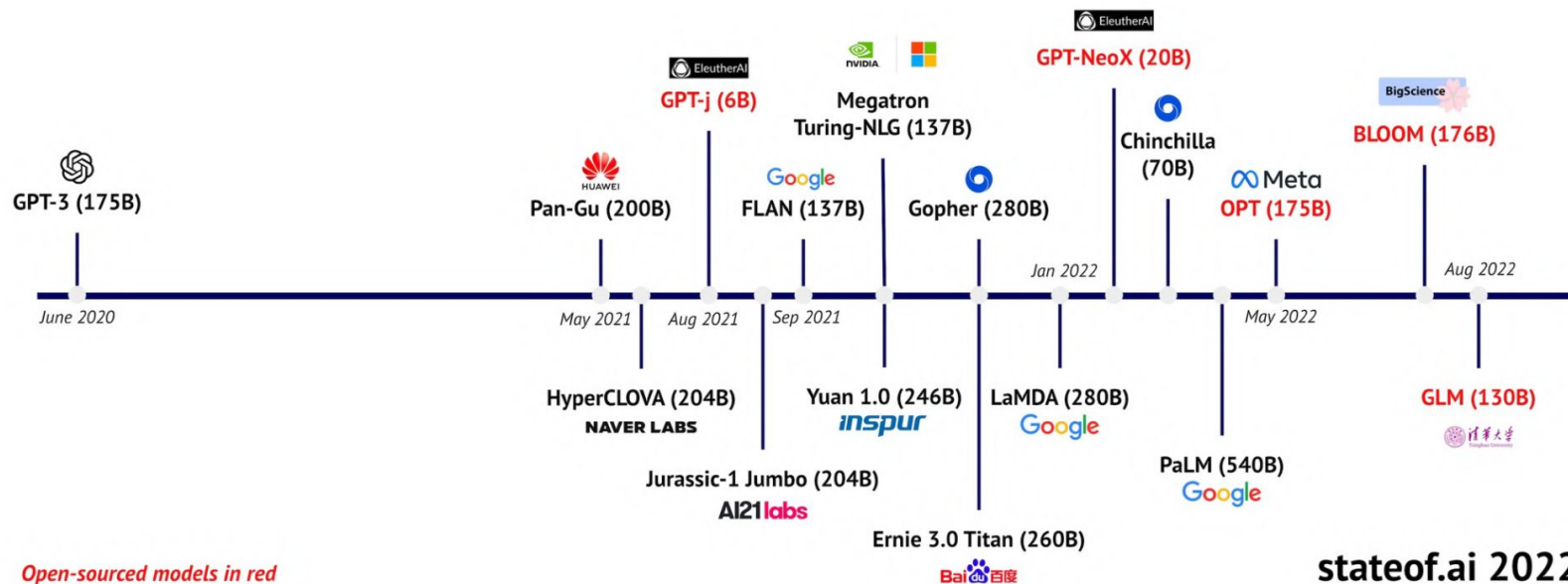
Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1 Translate English to French: ← task description
2 cheese => ..... ← prompt
```



This is a competition



GPT-4?!

- CLIP(2021년 1월): "zero-shot" 이미지 분류
- DALL-E(2021년 1월): 주어진 텍스트로부터 이미지 생성
- Codex(2021년 8월): 코드 생성을 위한 모델*
- InstructGPT(2022년 1월): 명령에 대한 파인튜닝과 강화학습
→ 이미 지식은 다 있다, 어떻게 뽑아낼 것인가

일반 언어 모델:

> ChatGPT에 대해 설명해줘

BERT에 대해 설명해줘

T5에 대해 설명해줘

GPT에 대해 설명해줘

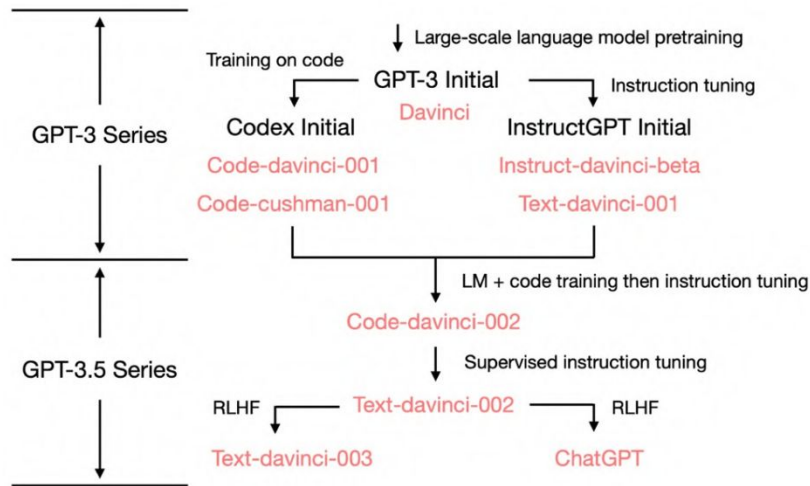
Instruction fine-tuning을 적용한 언어 모델:

> ChatGPT에 대해 설명해줘

ChatGPT는 OpenAI에서 개발한 대규모 언어 모델입니다. 텍스트와 코드의 방대한 데이터 세트로 학습되었으며 텍스트 생성, 언어 번역, 다양한 종류의 창의적인 콘텐츠 작성, 유익한 방식으로 질문에 답변할 수 있습니다. 아직 개발 중이지만 다음과 같은 다양한 작업을 수행하는 방법을 배웠습니다.

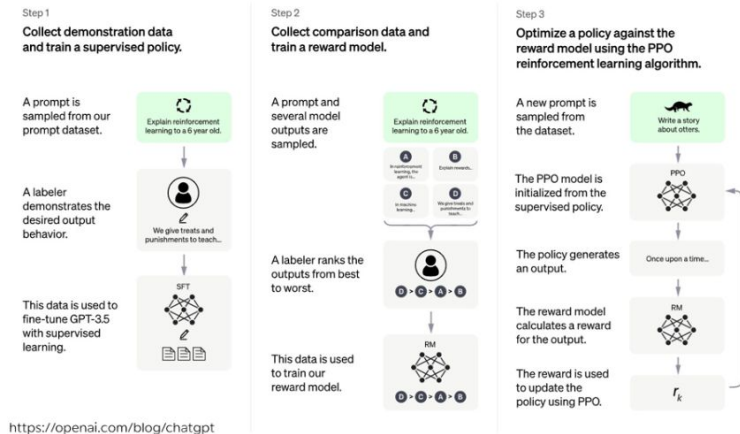
GPT-3.5 2022년 3월

- GPT-3.5 ~ GPT-3 + Code + Instruction fine-tuning
- Code data: 코드 생성을 위한 모델인 Codex(2021년 8월)에서 추론 능력과 긴 입력에 대한 이해가 올라가는 것을 관찰해서 코드 데이터 추가
- Instruction fine-tuning: 명령에 대한 파인튜닝과 강화학습을 하면 사용자의 의도를 더 잘 파악하고 답변한다는 것에 착안한 InstructGPT(2022년 1월)의 실험 방식을 가미
- 다 섞어서 짬뽕한게 GPT-3.5!



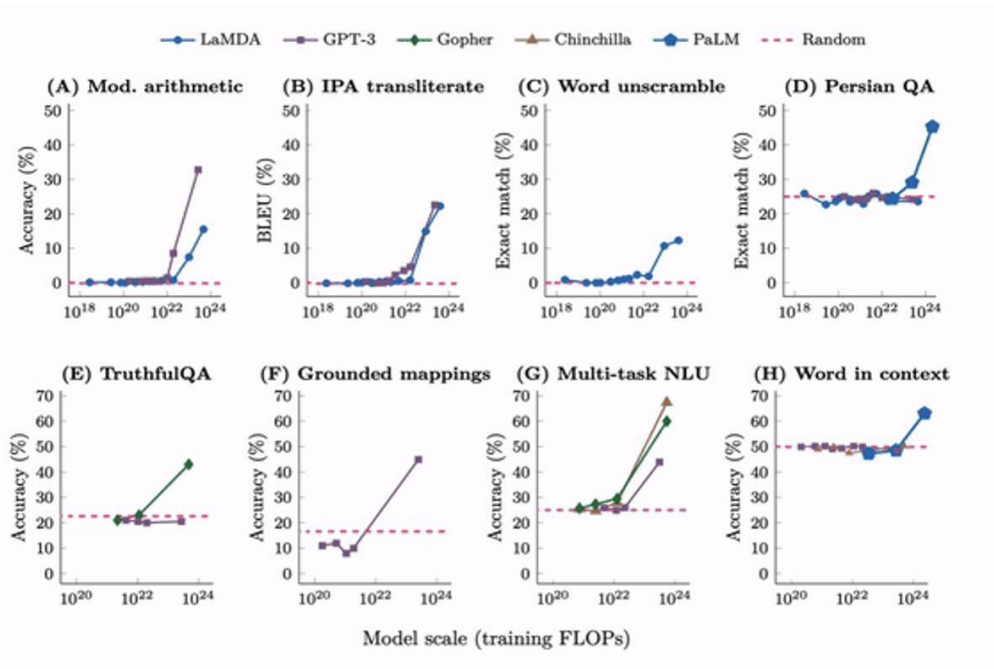
ChatGPT 2022년 11월

- GPT-3.5를 파인튜닝한 모델
- InstructGPT의 "sibling model"로, 학습 방식이 유사
- Demonstration data: 데이터를 대화형으로 바꿈
- 보상 모델(Reward model, RM): 유저의 선호도에 대한 모델
- 보상 모델을 활용해 ChatGPT를 강화학습(Reinforcement learning, RL)으로 업데이트



RNN부터 ChatGPT까지의 여정





• 모델 키우기

- 왜?
- 크면 해결 되는 일들이 있더라 .

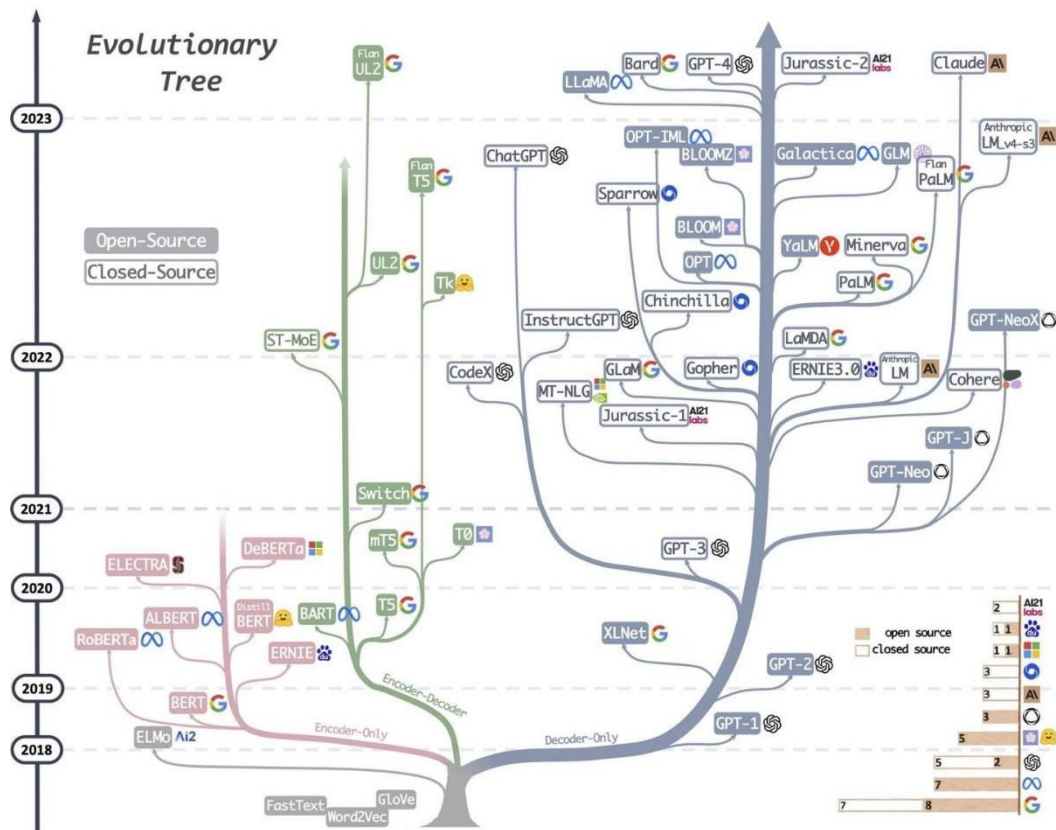
• 10B (100억 파라미터)

- 거대 언어 모델의 컨텍스트 인식 점프
- RLHF의 이득을 가장 많이 보는 구간

• 100B (1000억 파라미터)

- 거대 언어 모델의 동작을 가르는 지점

2023년 : 캄브리아 LLM 대폭발



2023년 5~7월 사이에

- 약 10,000여개 언어 모델 등장
- 2023년 9월 기준 약 15,000개

10여개의 사전 훈련 모델

100여개의 응용 모델

10000여개의 파인 튜닝 모델

- PaLM 2 (2023년 5월)
 - 구글 차세대 언어 모델
 - 한국어 및 일본어 특화 개발
 - Gemini 는 공개 안하는 것으로...
- Falcon LLM (2023년 6월)
 - UAE 자금력을 바탕으로 만든 거대 언어 모델
 - Falcon 180B : 공개 언어 모델 중 가장 거대
 - GPT 3.5 : 175B
- Claude V2 (2023년 7월)
 - 10만토큰 입력 가능...
- Llama 2 (2023년 7월)
 - 메타의 Llama 개선 모델
 - 사실상 상업적 용도 무제한 허용

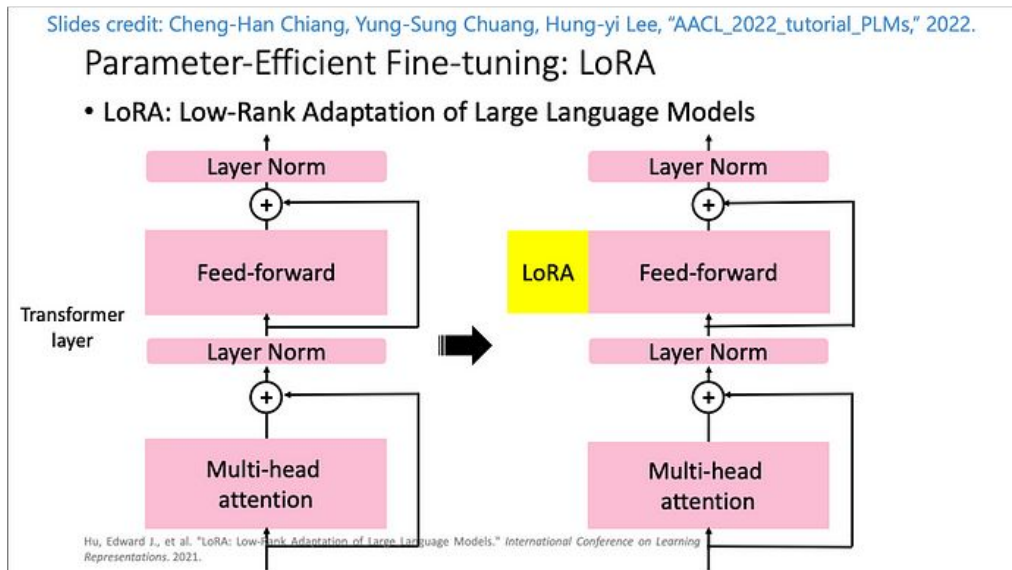
<https://ai.google/discover/palm2>

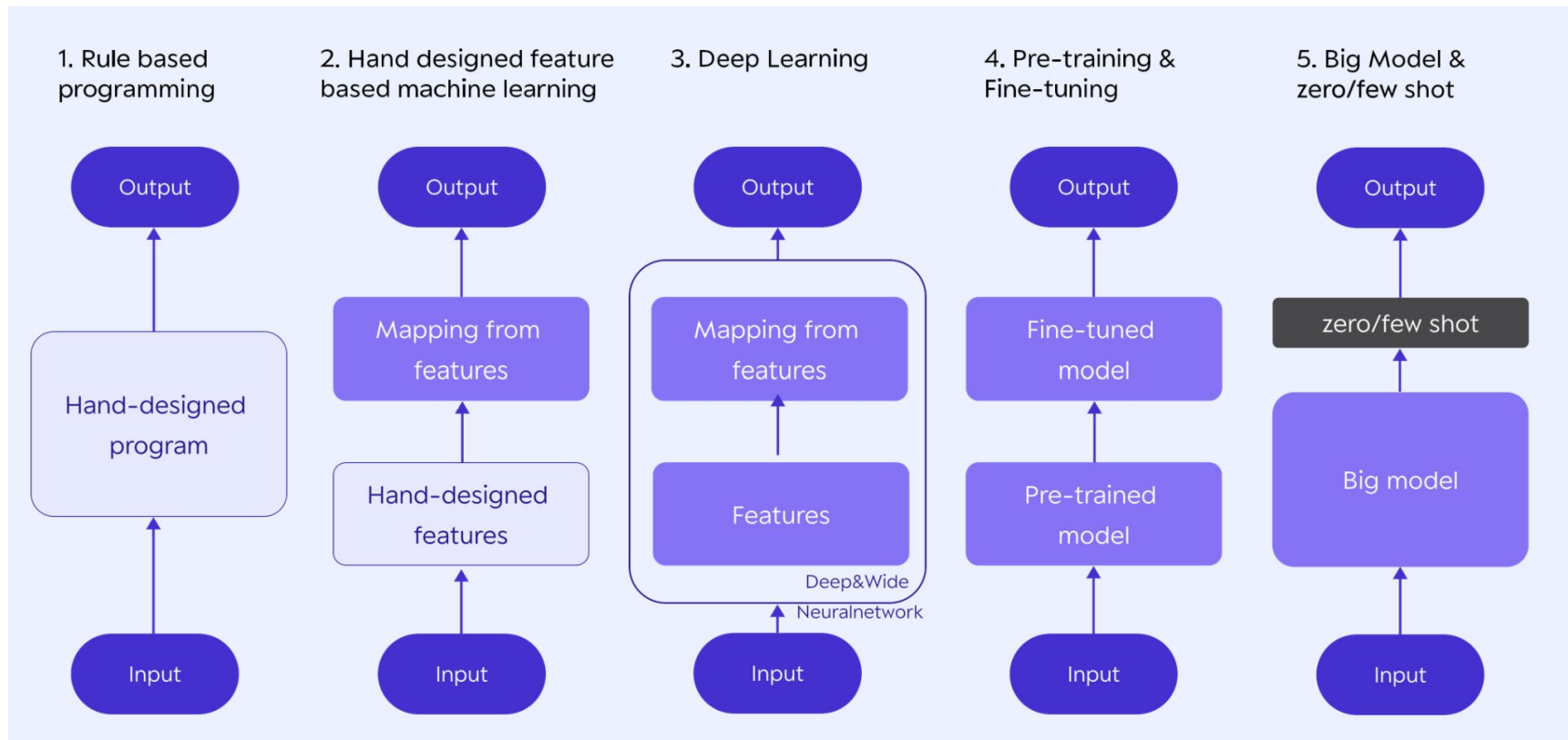
<https://falconllm.tii.ae>

<https://www.anthropic.com/index/introducing-claude>

<https://ai.meta.com/llama>

- LoRA(Low-Rank Adaptation of Large Language Models)
 - 사전 훈련된 모델 가중치는 고정
 - 훈련 가능한 레이어들을 별도로 붙이고 추가 훈련을 통해 학습





LLM 을 사용한 AI-Agent

1. Chat 기반 서비스



<https://www.langchain.com>

2. RPA 기반 서비스



<https://github.com/Significant-Gravitas/AutoGPT>

LLM 을 사용한 챗봇 서비스 만들기 - 챗GPT Playground

Playground

Chat Save View code Share ...

ChatGPT 역할 부여

SYSTEM
You are a helpful assistant.
너는 운세 전문가야

USER Enter a user message here.
유저 채팅

ASSISTANT Enter an assistant message here.
chatGPT

USER Enter a user message here.

+ Add message

Mode
Chat Beta

Model
gpt-3.5-turbo

Temperature 0.9

Maximum length 150

Top P 1

Frequency penalty 0

Presence penalty 0.6

무작위성

답변 길이

상위 확률 단어

빈도 패널티

존재 패널티

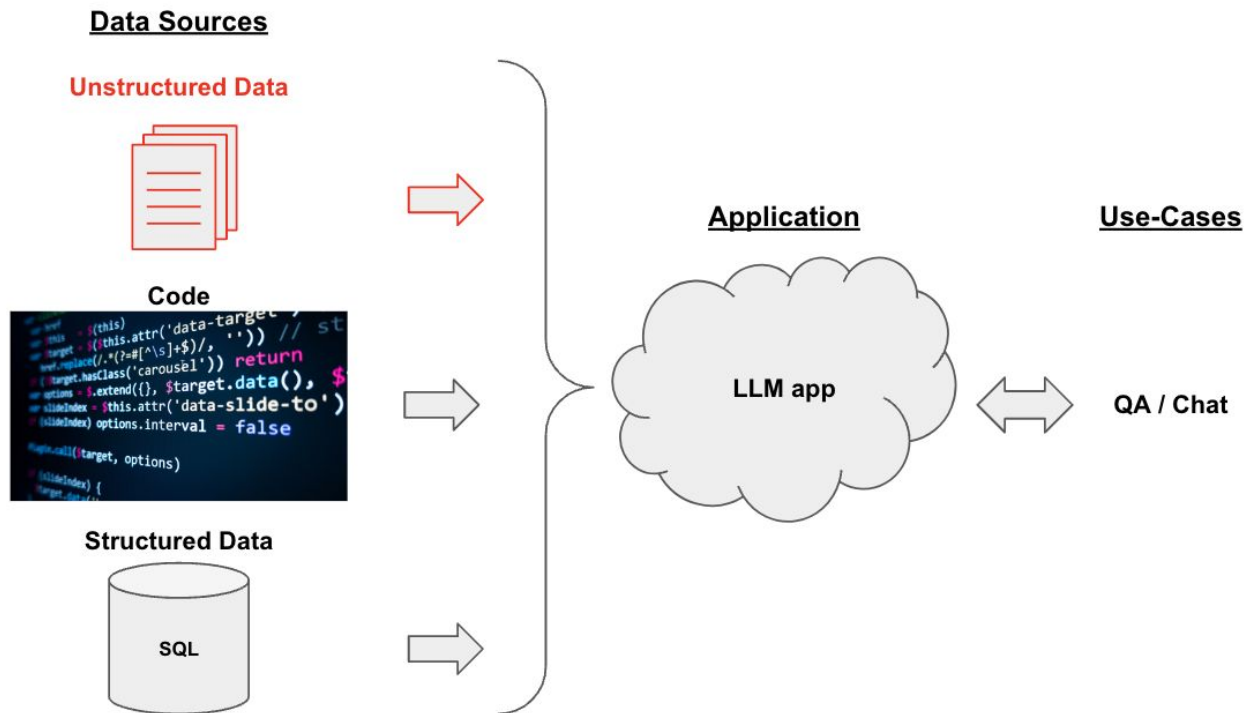
<https://platform.openai.com/playground>

LLM 을 사용한 챗봇 서비스 만들기 - langchain



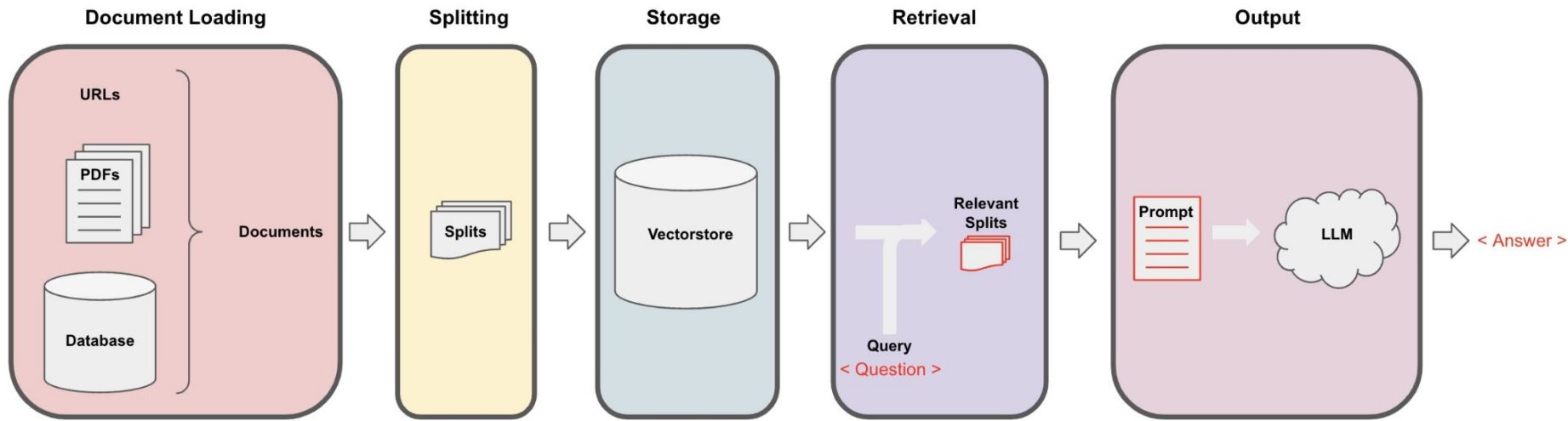
https://python.langchain.com/docs/get_started/introduction.html

LLM 을 사용한 챗봇 서비스 만들기 - langchain

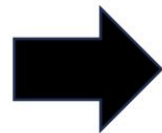
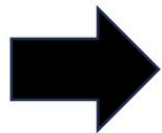


https://python.langchain.com/docs/use_cases/question_answering

LLM 을 사용한 챗봇 서비스 만들기 - langchain



https://python.langchain.com/docs/use_cases/question_answering



LLM 을 사용한 챗봇 서비스 만들기 - langchain

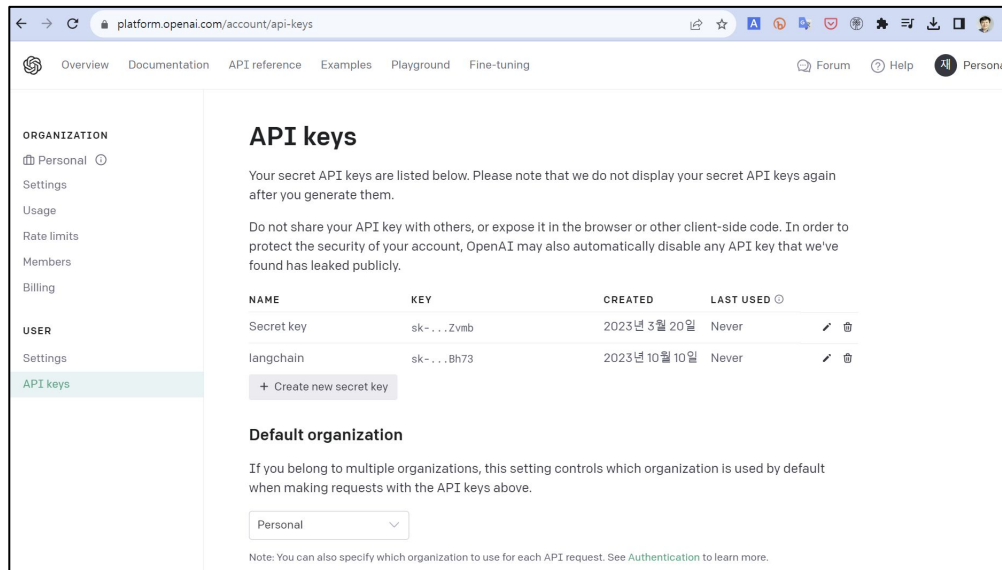
파이썬 환경에서 쉽게 개발 가능

pip install langchain

pip install openai

```
from langchain.llms import OpenAI

llm = OpenAI(openai_api_key="...")
```



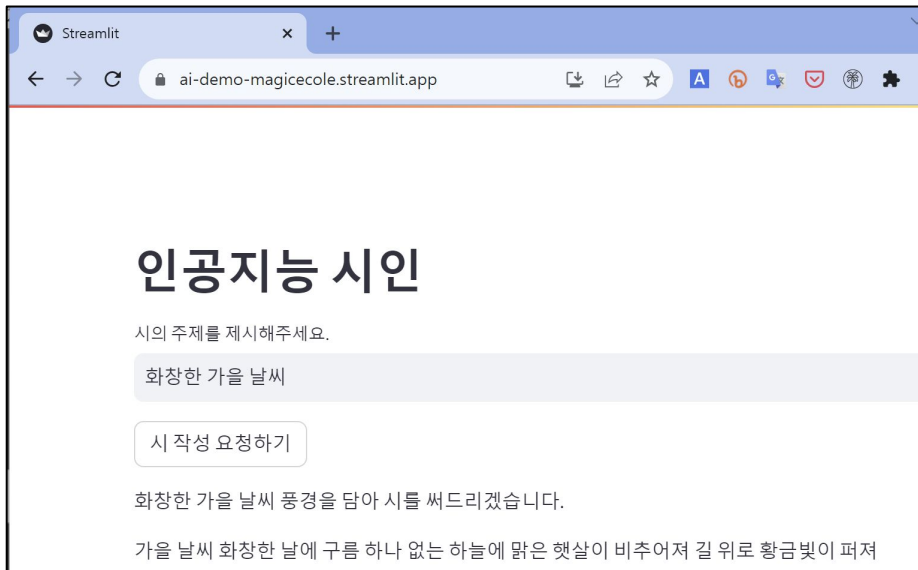
LLM 을 사용한 챗봇 서비스 만들기

```
from langchain.llms import OpenAI
from langchain.chat_models import ChatOpenAI

llm = OpenAI()
chat_model = ChatOpenAI()

llm.predict("hi!")
>>> "Hi"

chat_model.predict("hi!")
>>> "Hi"
```



LLM 을 사용한 챗봇 서비스 만들기 - LLaMA 2 경량 버전 사용하기

```
import streamlit as st
from langchain.llms import CTransformers

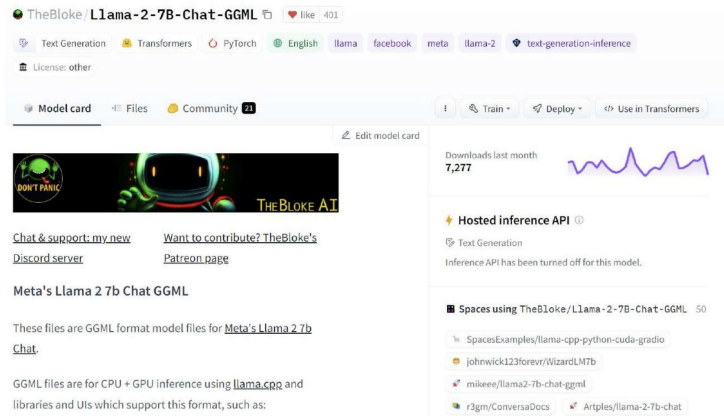
llm = CTransformers(
    model="llama-2-7b-chat.ggmlv3.q2_K.bin",
    model_type="llama"
)

st.title('인공지능 시인')

content = st.text_input('시의 주제를 제시해주세요.')

if st.button('시 작성 요청하기'):
    with st.spinner('시 작성 중...'):
        result = llm.predict("write a poem about " + content + ": ")
        st.write(result)
```

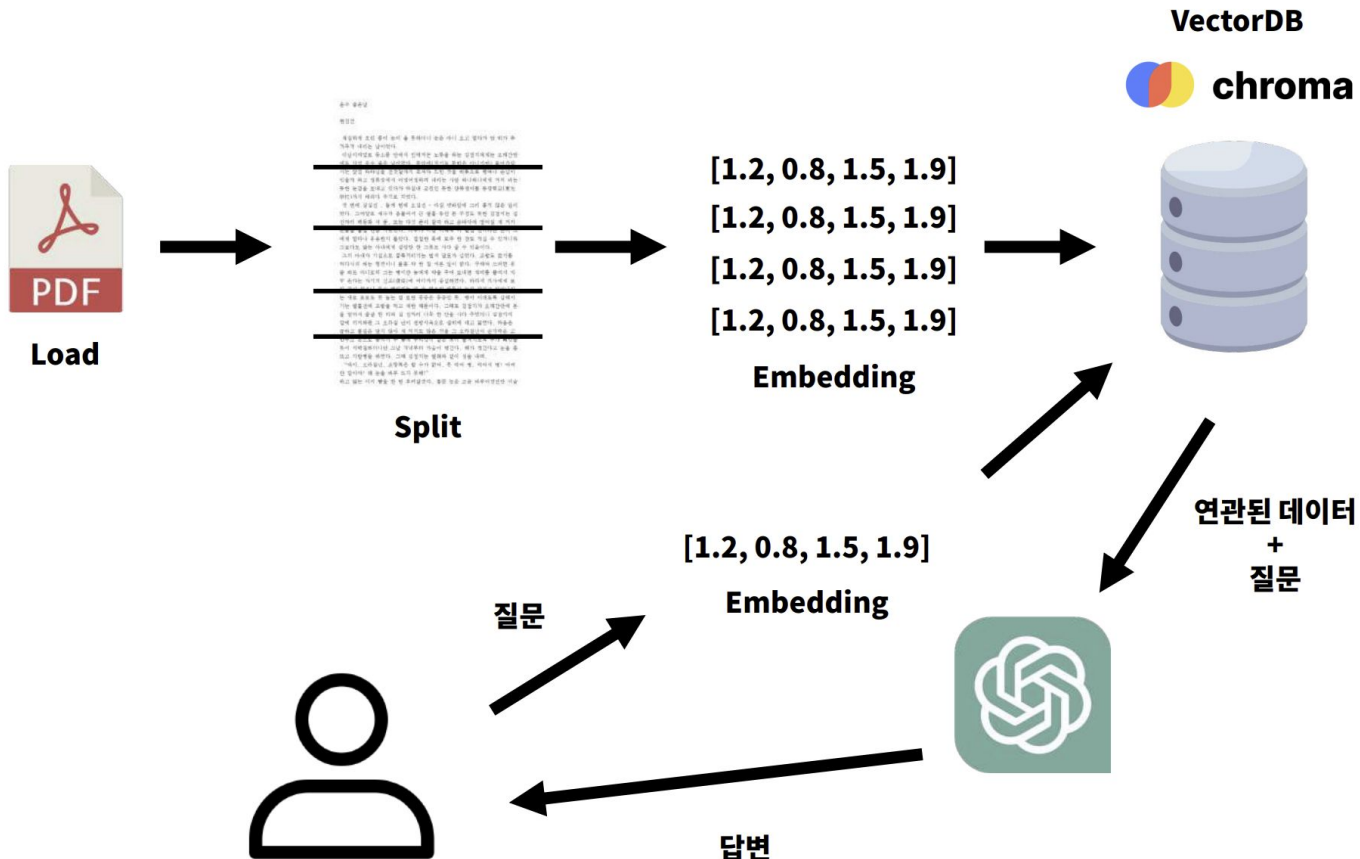
<https://huggingface.co/TheBloke/Llama-2-7B-Chat-GGML>



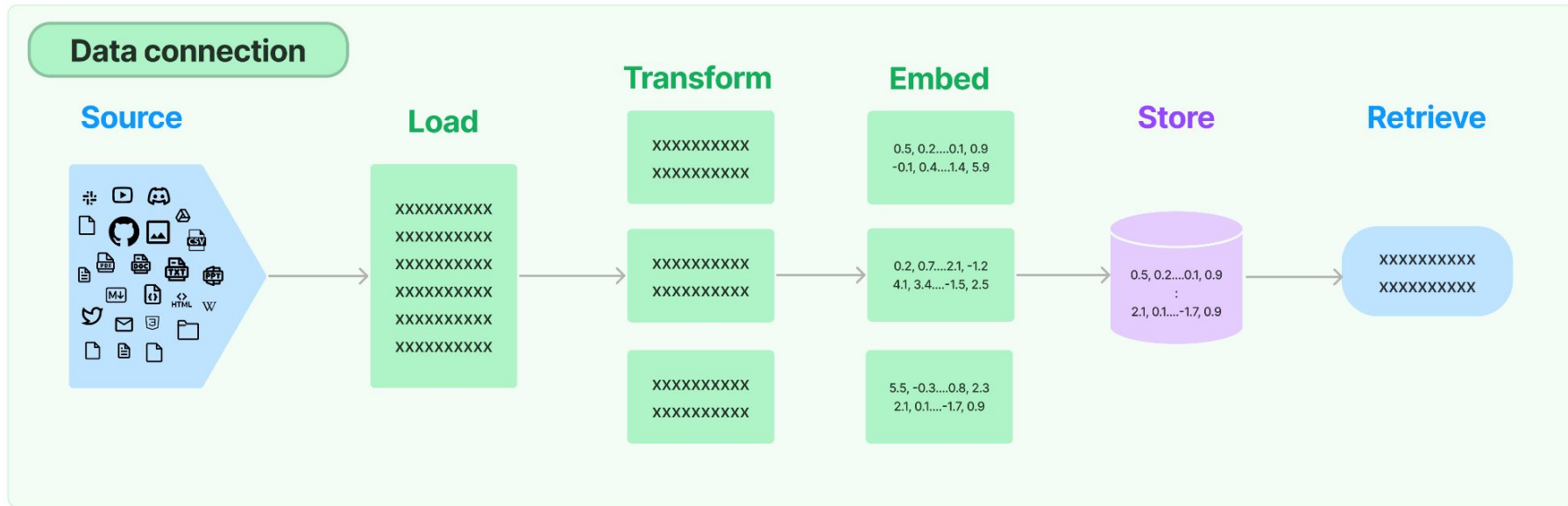
Georgi Gerganov + Machine Learning

<https://huggingface.co/TheBloke/Llama-2-7B-Chat-GGML>

LLM 을 사용한 챗봇 서비스 만들기 - ChatPDF 만들기



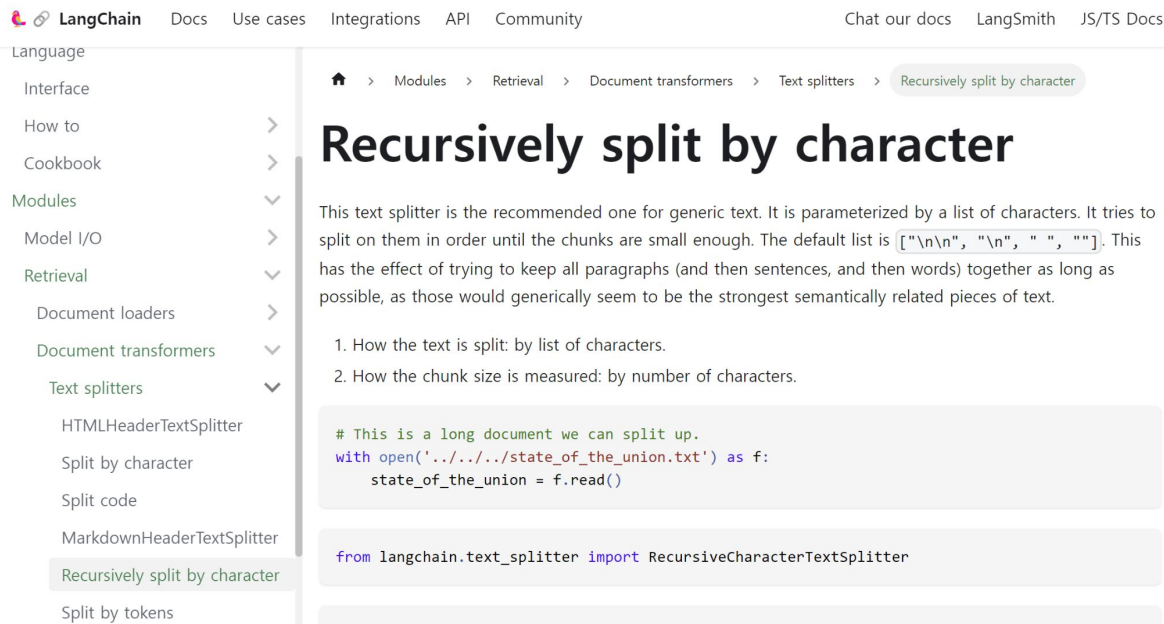
LLM 을 사용한 챗봇 서비스 만들기 - PDF 로딩



https://python.langchain.com/docs/modules/data_connection/document_loaders/pdf

pip install pypdf

https://python.langchain.com/docs/modules/data_connection/document_transformers/text_splitters/recursive_text_splitter



The screenshot shows the LangChain documentation website. The top navigation bar includes links for LangChain, Docs, Use cases, Integrations, API, Community, Chat our docs, LangSmith, and JS/TS Docs. A left sidebar contains a tree view of the documentation structure, with 'Text splitters' expanded under 'Modules'. The main content area is titled 'Recursively split by character' and includes a description of the text splitter, a list of two points (how text is split and how chunk size is measured), and two code snippets. The first snippet shows how to read a file, and the second shows how to import the RecursiveCharacterTextSplitter class.

LangChain Docs Use cases Integrations API Community Chat our docs LangSmith JS/TS Docs

Language
Interface
How to
Cookbook
Modules
Model I/O
Retrieval
Document loaders
Document transformers
Text splitters
HTMLHeaderTextSplitter
Split by character
Split code
MarkdownHeaderTextSplitter
Recursively split by character
Split by tokens

Modules > Retrieval > Document transformers > Text splitters > Recursively split by character

Recursively split by character

This text splitter is the recommended one for generic text. It is parameterized by a list of characters. It tries to split on them in order until the chunks are small enough. The default list is `["\n\n", "\n", " ", ""]`. This has the effect of trying to keep all paragraphs (and then sentences, and then words) together as long as possible, as those would generically seem to be the strongest semantically related pieces of text.

1. How the text is split: by list of characters.
2. How the chunk size is measured: by number of characters.

```
# This is a long document we can split up.  
with open('../../state_of_the_union.txt') as f:  
    state_of_the_union = f.read()
```

```
from langchain.text_splitter import RecursiveCharacterTextSplitter
```

LLM 을 사용한 챗봇 서비스 만들기 - embedding

Retrieval ▾

Document loaders >

Document transformers ▾

Text splitters ▾

HTMLHeaderTextSplitter

Split by character

Split code

MarkdownHeaderTextSplitter

Recursively split by character

Split by tokens

Post retrieval >

Text embedding models ▾

Caching

Vector stores

Retrievers >

Indexing

Chains >

Memory >

Agents >

for having these as two separate methods is that some embedding providers have different embedding methods for documents (to be searched over) vs queries (the search query itself).

G

Get started

Setup

To start we'll need to install the OpenAI Python package:

```
pip install openai
```

Accessing the API requires an API key, which you can get by creating an account and heading [here](#). Once we have a key we'll want to set it as an environment variable by running:

```
export OPENAI_API_KEY="..."
```

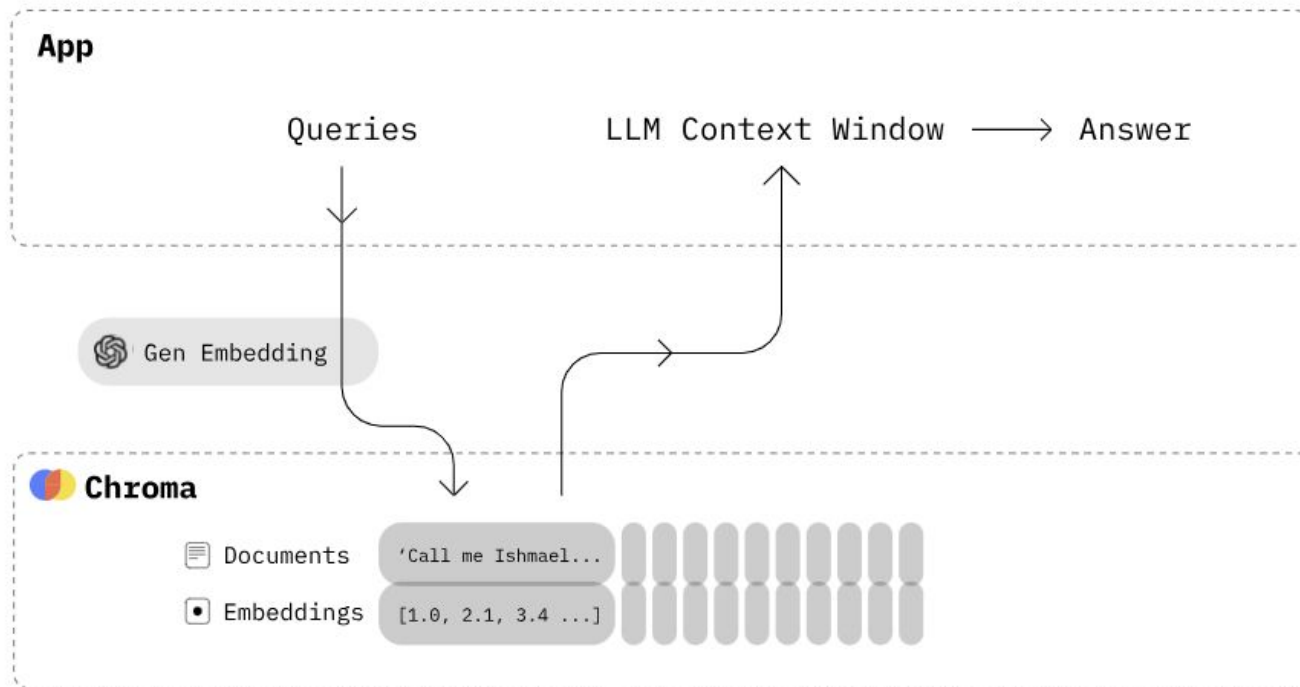
If you'd prefer not to set an environment variable you can pass the key in directly via the `openai_api_key` named parameter when initiating the OpenAI LLM class:

```
from langchain.embeddings import OpenAIEmbeddings

embeddings_model = OpenAIEmbeddings(openai_api_key="...")
```

pip install tiktoken

LLM 을 사용한 챗봇 서비스 만들기 - vectorDB



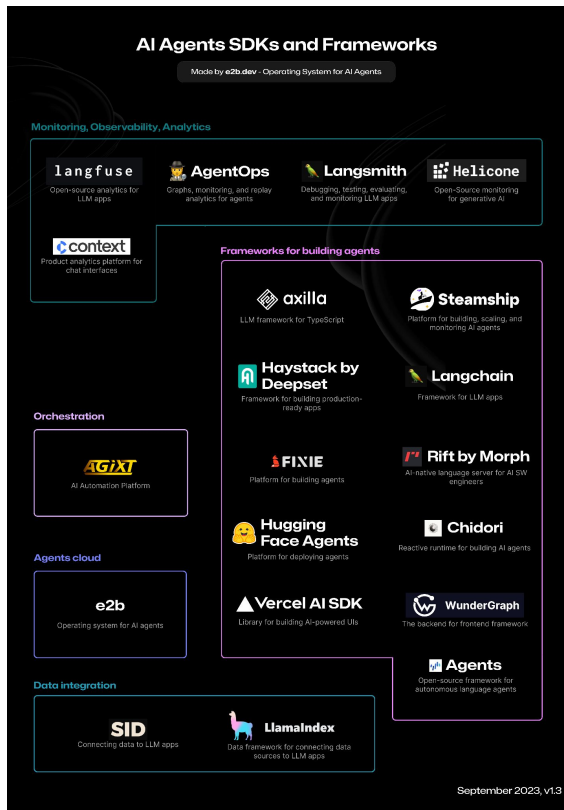
pip install chromadb

<https://docs.trychroma.com>

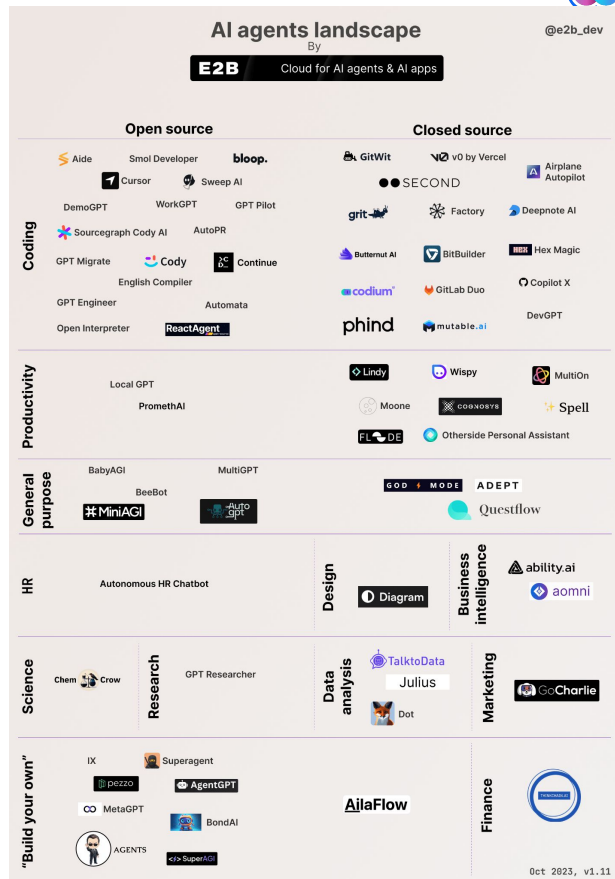


<https://github.com/Significant-Gravitas/AutoGPT>

AI Agent 생태계

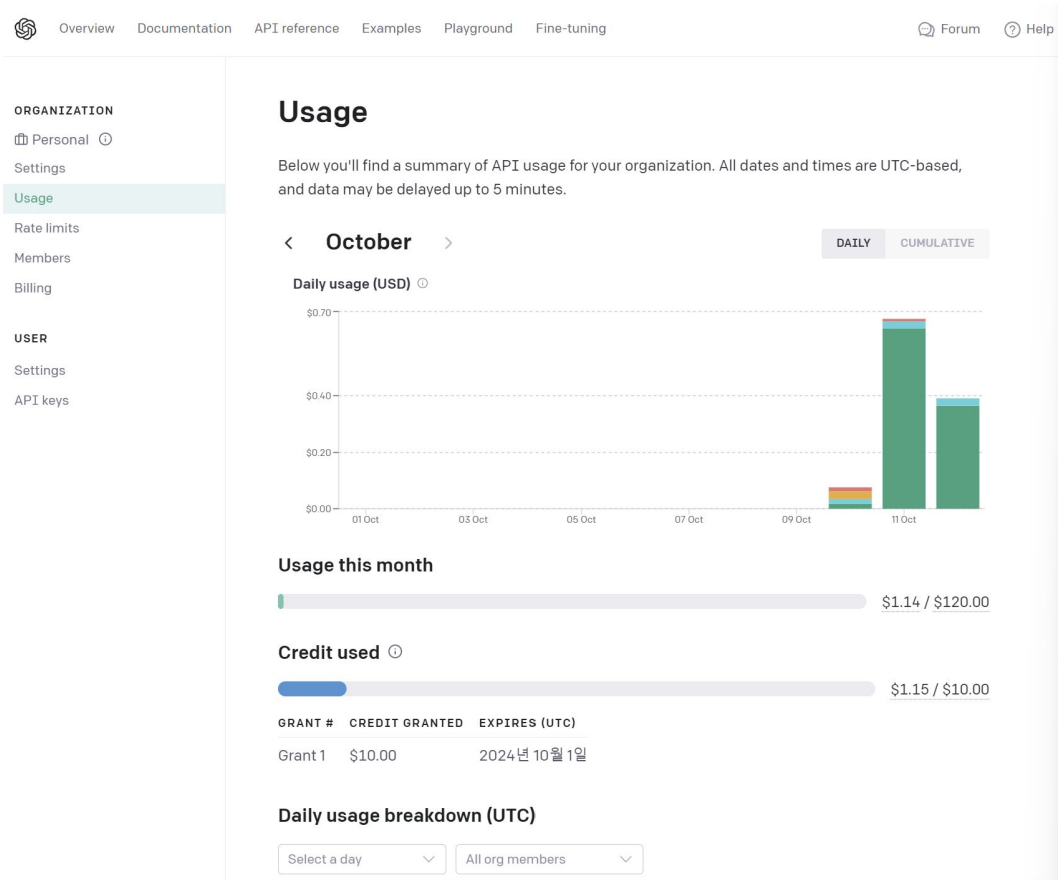


<https://github.com/e2b-dev/awesome-sdks-for-ai-agents>



<https://github.com/e2b-dev/awesome-ai-agents>

AutoGPT를 이용한 업무 자동화 - Reworkd



AgentGPT 배다

Interested in automating businesses with AI Agents? Apply here >

Free Trial

A small taste of what AgentGPT offers.

\$0 / month

Features:

- ✓ 5 demo agents a day using GPT-3.5-Turbo
- ✓ Limited plugin integrations
- ✓ Limited web search capabilities

[Continue](#)

PRO

Powerful AI agents at your fingertips.

\$40 / month Most Popular

Features:

- ✓ 30 Agents per day
- ✓ GPT-3.5-Turbo 16k access
- ✓ GPT-4 access*
- ✓ 25 loops per Agent
- ✓ Unlimited web search capabilities
- ✓ Access to the latest AgentGPT plugins
- ✓ Priority support

[Subscribe](#)

Enterprise

AI agents tailored to automate your business.

Custom / month

Features:

- ✓ Everything in the Pro plan
- ✓ Features based on enterprise requirements
- ✓ SAML single sign-on
- ✓ Dedicated account manager

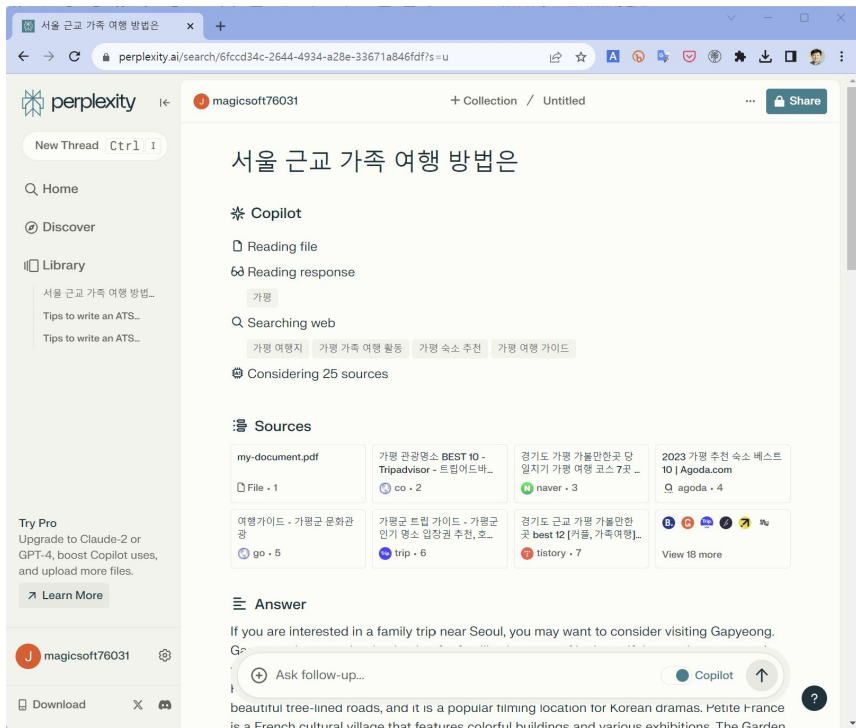
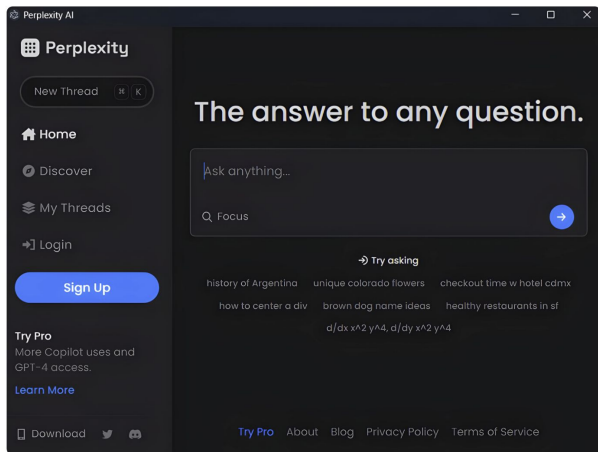
[Contact Us](#)

AutoGPT를 이용한 업무 자동화 - Perplexity



Perplexity AI Desktop App

A beautifully crafted desktop application for Perplexity AI, an advanced artificial intelligence (AI) platform created by Meta AI and brought to life by the Perplexity team. The app is built using Electron, offering seamless navigation between "perplexity.ai" and "labs.perplexity.ai".



<https://github.com/inulute/perplexity-ai-app>
<https://www.perplexity.ai/discover>

AI 보조시스템



PBT 시스템

Testing
(시험 · 평가)

Training
(LMS)

Tokening
(DT 역량 성장 리포트)

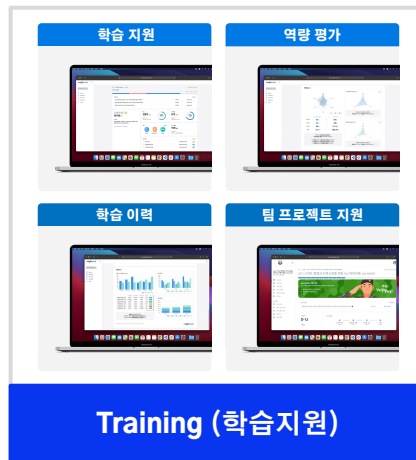
평가 문항 개발
Agent

학습자 지원
Agent

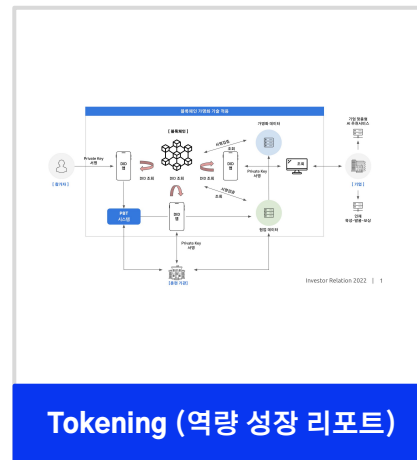
PBT 리포트 생성
Agent



Testing (평가)

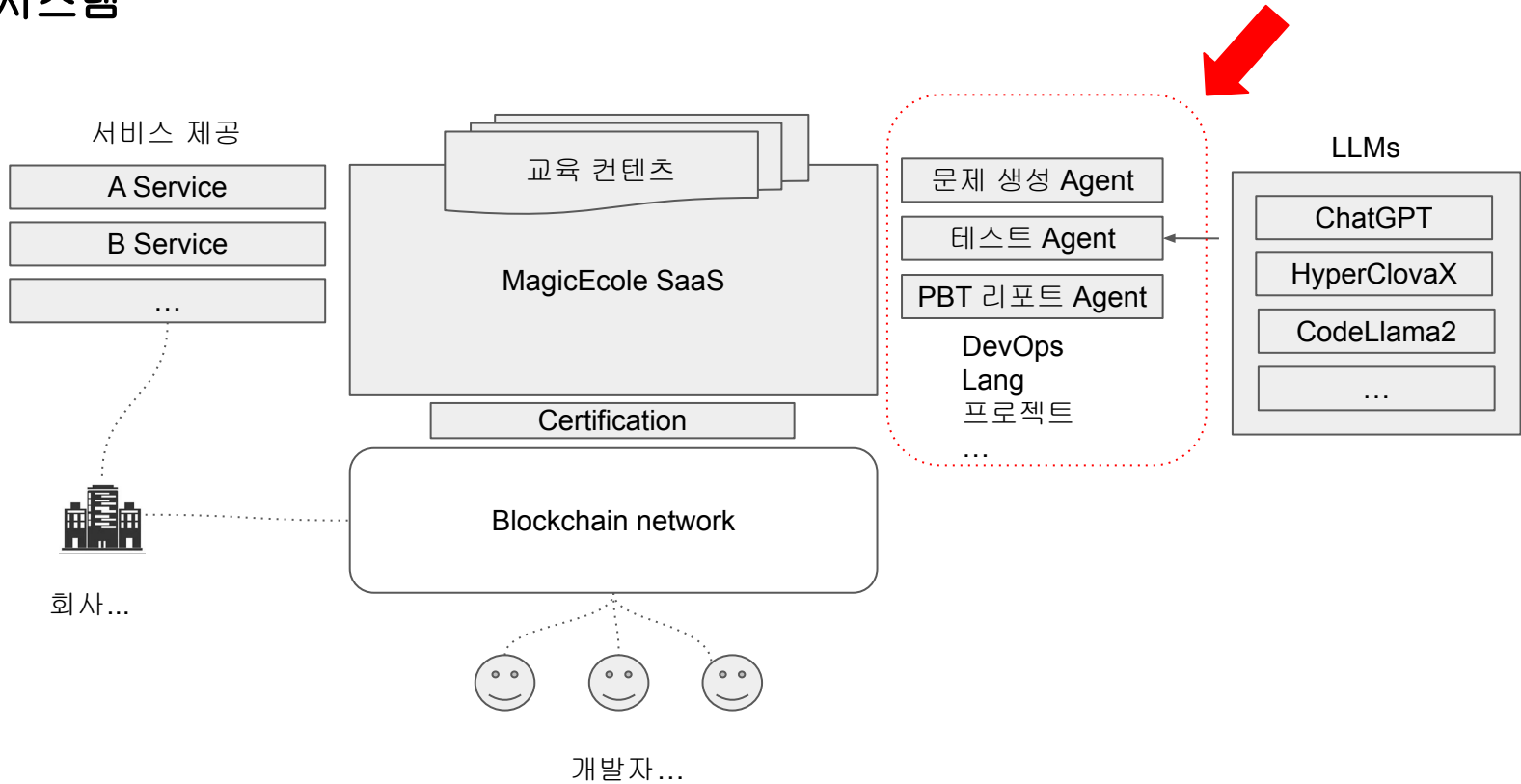


Training (학습지원)



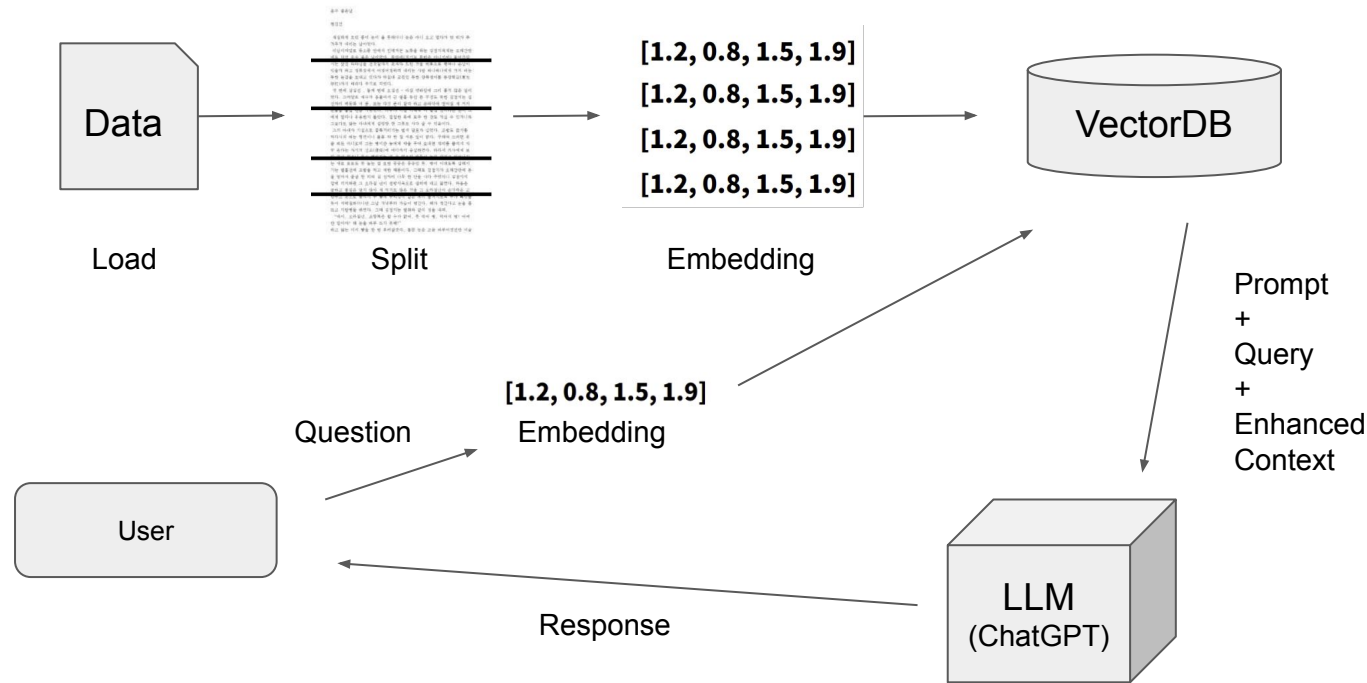
Tokening (역량 성장 리포트)

AI 보조시스템



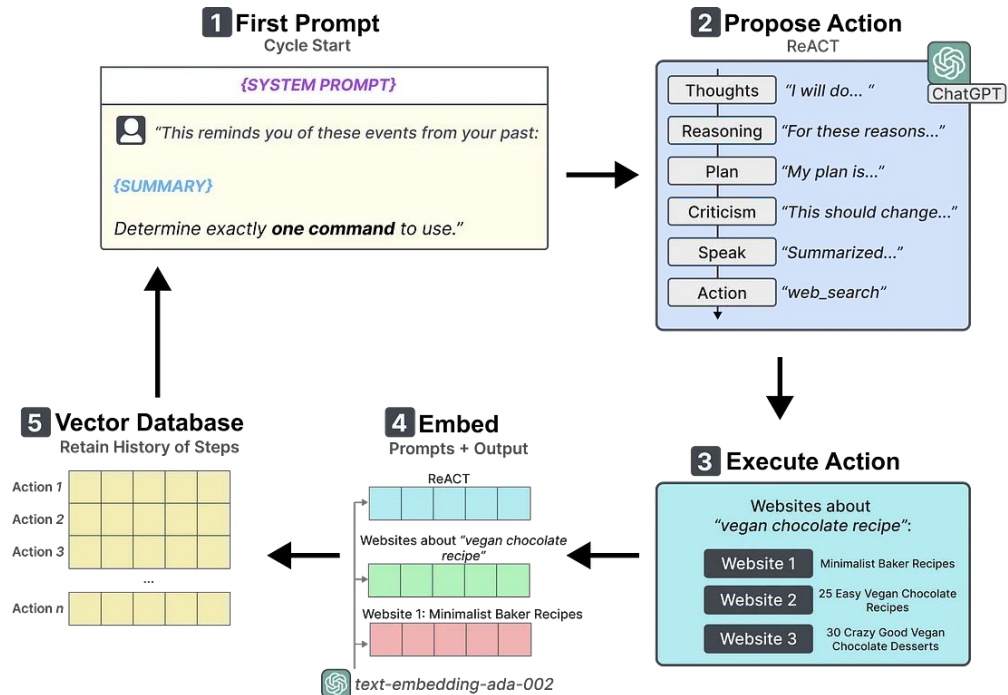
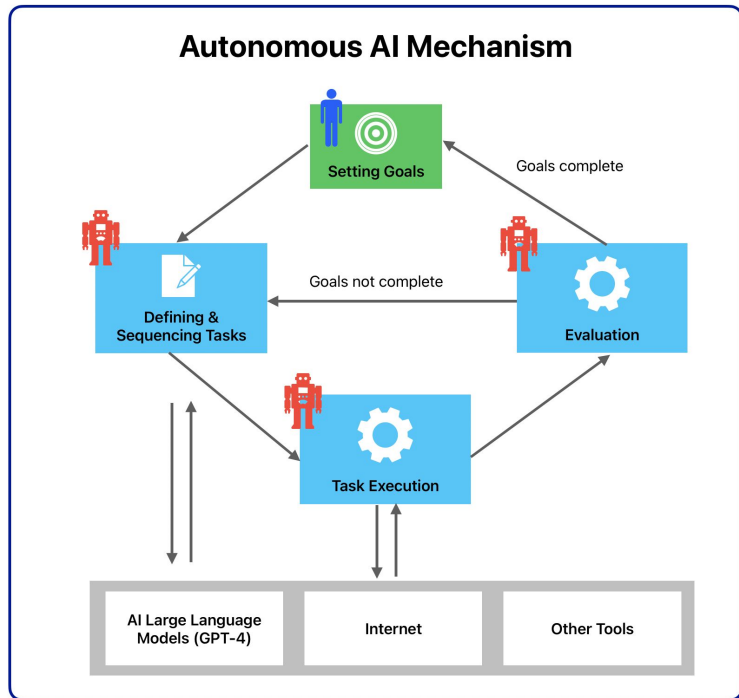
AI 보조시스템

학습자 지원1: 질의응답 시스템



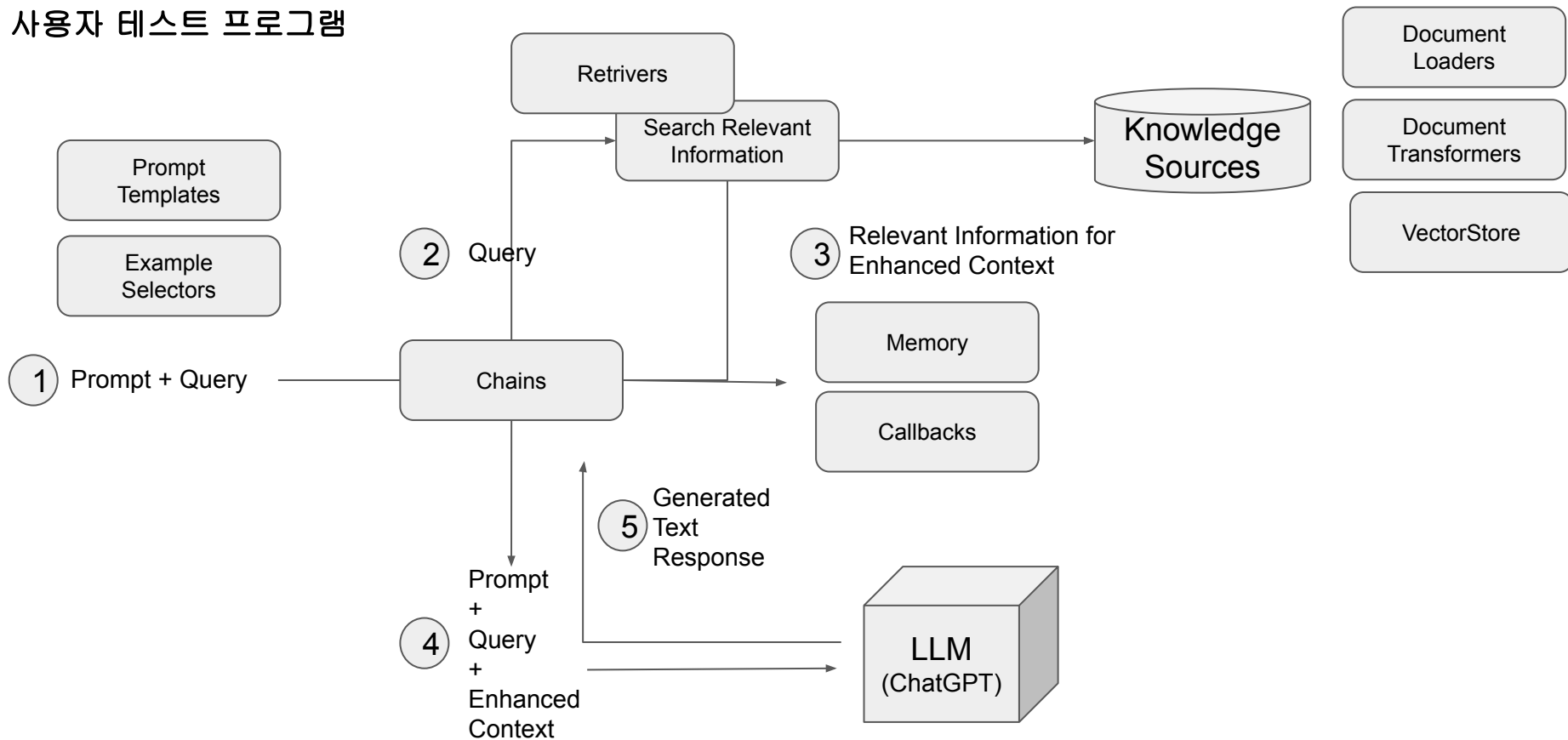
AI 보조시스템

AutoGPT 기반 문제 출제 시스템



AI 보조시스템

사용자 테스트 프로그램



AI 보조시스템

학습자 지원1: 질의응답 시스템

PDF 파일을 올려주세요!



Drag and drop file here

Limit 200MB per file • PDF

Browse files



[매직에꼴] 학생 사용 가이드.pdf 24.6MB



매직에꼴에 질문해보세요!!

질문을 입력하세요

매직에꼴 회원 가입 방법 설명해줘

질문하기

매직에꼴 회원 가입 방법은 다음과 같습니다:

1. 매직에꼴 사이트에 접속합니다.
2. 수강할 기관을 클릭한 뒤, 오른쪽 상단의 '회원가입' 버튼을 클릭합니다.
3. 매직에꼴 회원가입 약관을 확인하고 '동의'를 클릭합니다.

위의 단계를 따라 진행하면 매직에꼴 회원으로 가입할 수 있습니다.

```
main.py M X  openal.py
main.py > ...
26
27 #파일 업로드
28 uploaded_file = st.file_uploader("PDF 파일을 올려주세요!", type=['pdf'])
29 st.write("----")
30
31 def pdf_to_document(uploaded_file):
32     temp_dir = tempfile.TemporaryDirectory()
33     temp_filepath = os.path.join(temp_dir.name, uploaded_file.name)
34     with open(temp_filepath, "wb") as f:
35         f.write(uploaded_file.getvalue())
36     loader = PyPDFLoader(temp_filepath)
37     pages = loader.load_and_split()
38     return pages
39
40 #업로드 되면 동작하는 코드
41 if uploaded_file is not None:
42     pages = pdf_to_document(uploaded_file)
43
44     #Split
45     text_splitter = RecursiveCharacterTextSplitter(
46         # Set a really small chunk size, just to show.
47         chunk_size = 300,
48         chunk_overlap = 20,
49         length_function = len,
50         is_separator_regex = False,
51     )
52     texts = text_splitter.split_documents(pages)
53
54     #Embedding
55     embeddings_model = OpenAIEmbeddings(openai_api_key=openai_key)
56
57 PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 2
58
59 openai.error.InvalidRequestError: The model `gpt-4.0` does not exist
60 ^C Stopping...
61 magic@magic5:~/work/ai-chatdemo$ streamlit run main.py
62
63 Collecting usage statistics. To deactivate, set browser.gatherUsageStats to False.
64
65 You can now view your Streamlit app in your browser.
66
67 Network URL: http://172.28.195.95:8501
68 External URL: http://175.193.74.224:8501
```

AI 보조시스템

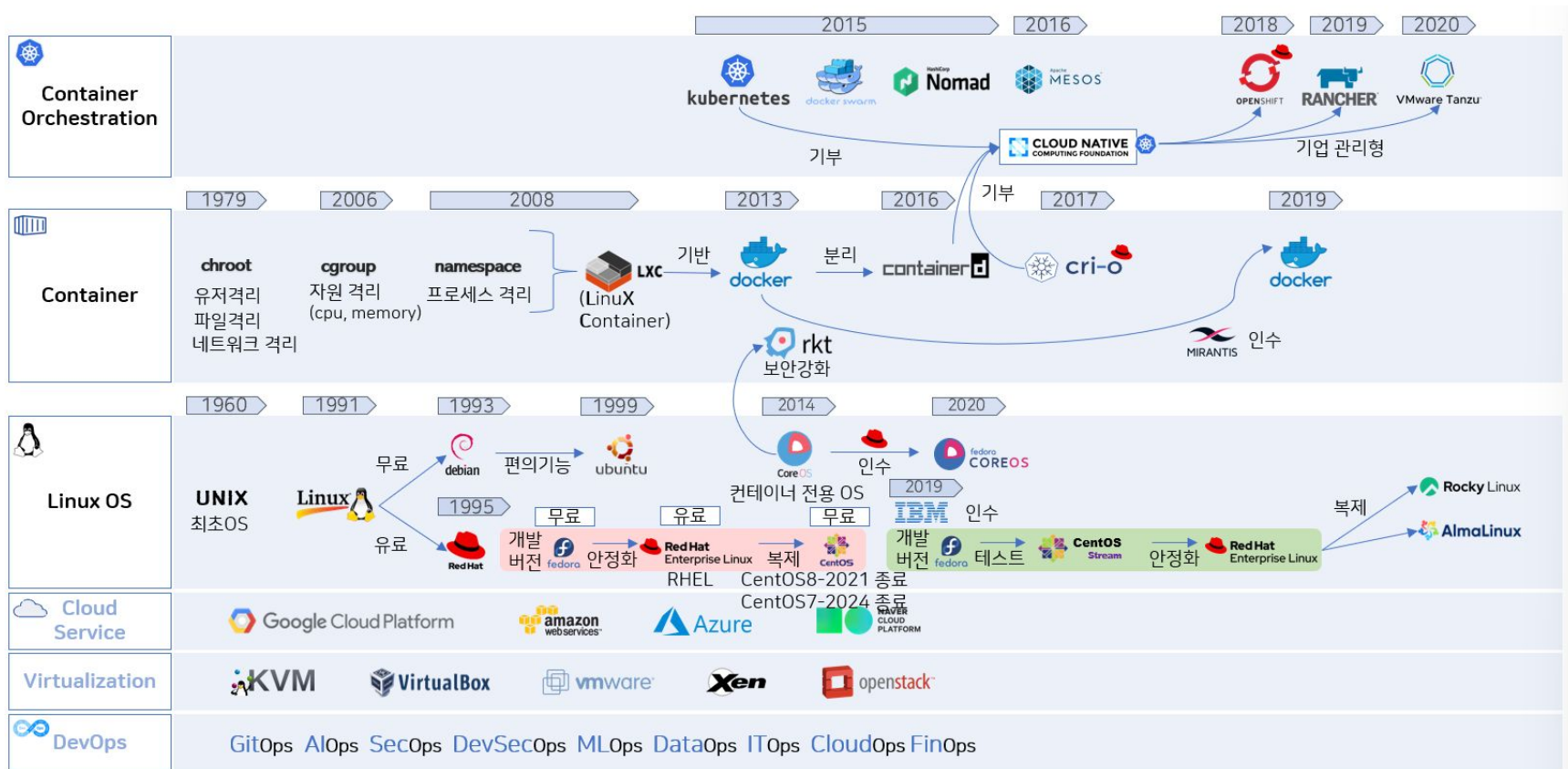
AutoGPT 기반 문제 출제 시스템

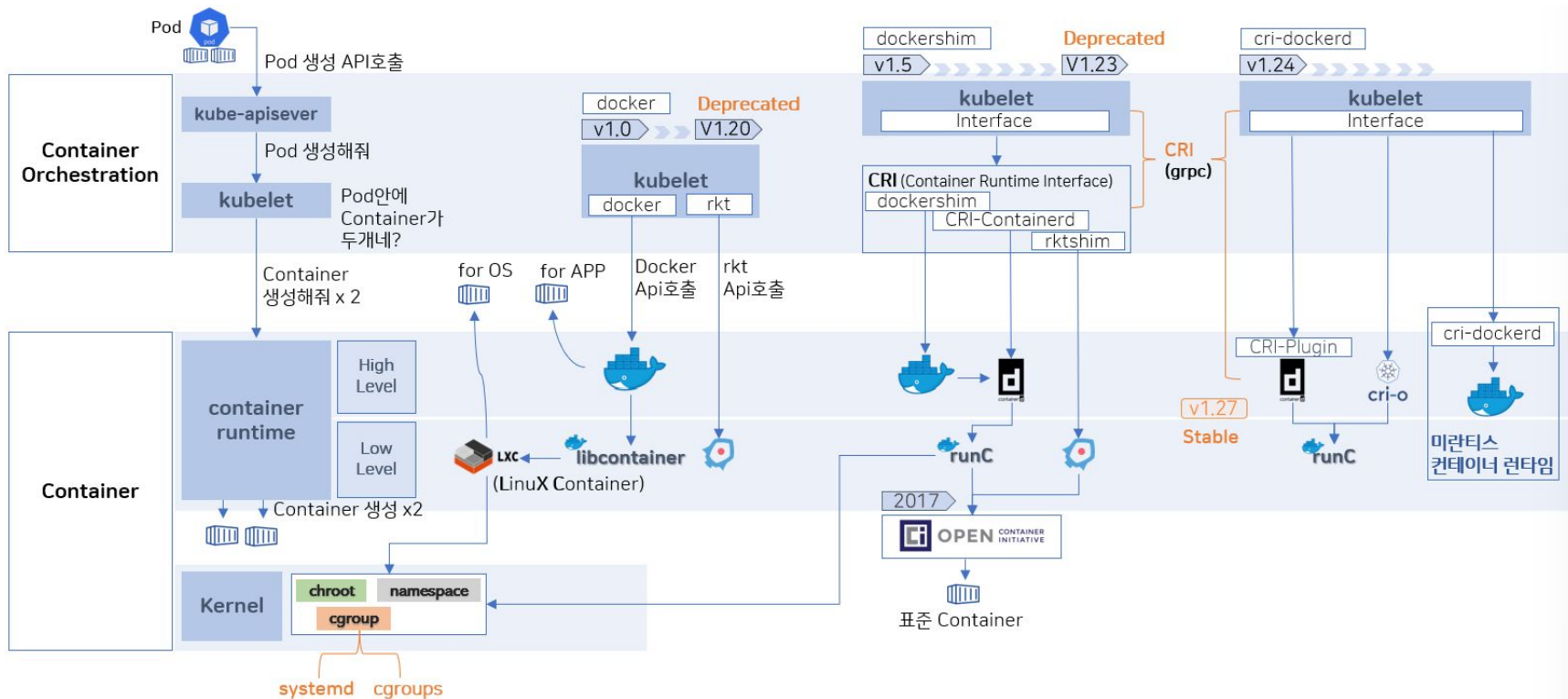
The screenshot displays the MagicEcole web application interface. On the left is a sidebar with a menu including '코딩 테스트 GPT', '개발자 인터뷰 GPT', 'IT 컨설턴트', '개발자 인터뷰 GPT', 'DevOps GPT', 'DevOps GPT', 'DevOps GPT', 'TravelGPT', '초등학교 선생님', '채용담당자', '면접관', '면접관', '며저과', 'Pages', 'Home', 'Templates', 'Settings', and a user profile 'cjk'. The main area shows a task log for 'Embarking on a new goal' (자바 언어 알고리즘 개발 능력을 검증할 코딩 테스트 문제 3개 작성). The log includes: 'Task Added', 'Starting task', 'Generating response...', 'Executing' (with a detailed instruction to create 3 Java algorithm test questions), 'Finished', and another 'Executing' step for 'Response for '자료 구조 중심의 알고리즘 문제로 작성해줘''. At the bottom, there is a 'Thinking' indicator and a text input field '에이전트에 추가로 명령어 입력...'. The right sidebar, titled 'Current tasks', shows the active task: '자바 언어 알고리즘 개발 능력을 검증할 코딩 테스트 문제 3개 작성'. The bottom right corner features buttons for '사용자 작업' and 'Add'.

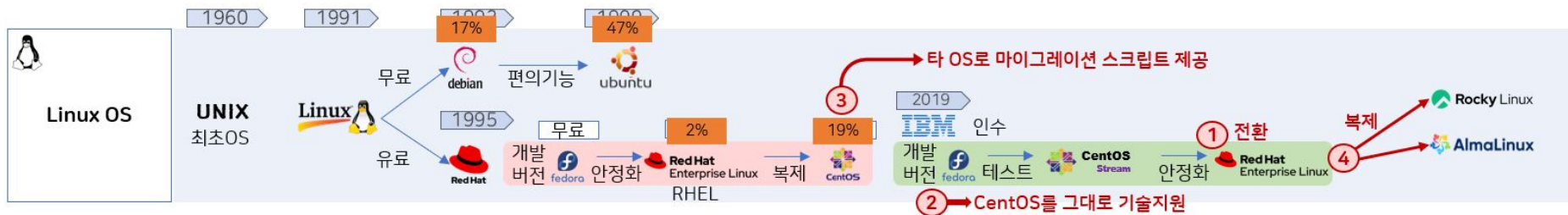
클라우드네이티브+AI 전문가 양성 .

3. 클라우드 서비스의 진화

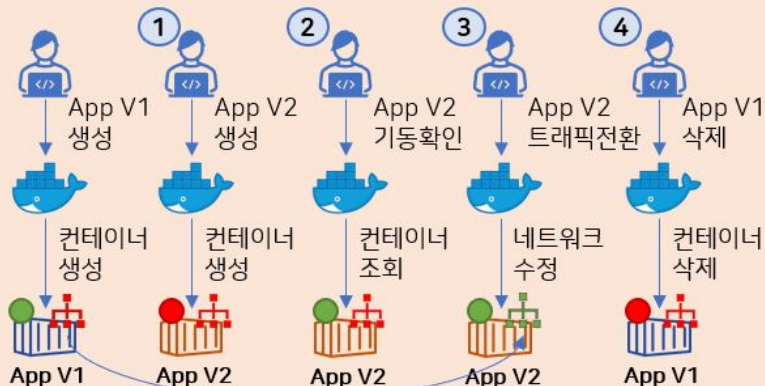
클라우드네이티브



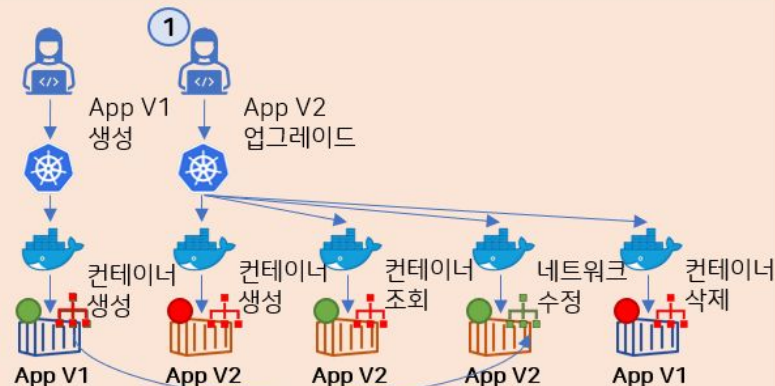




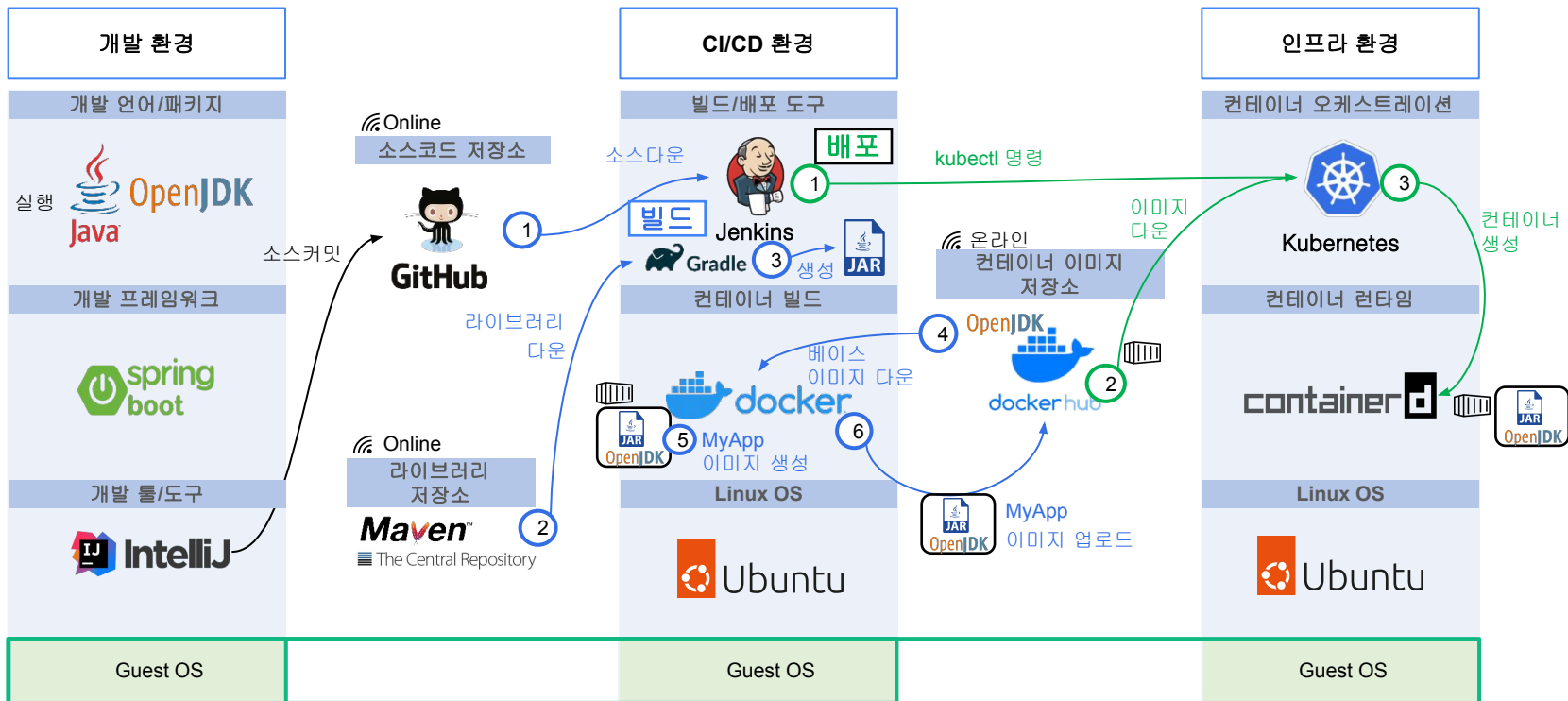
Container



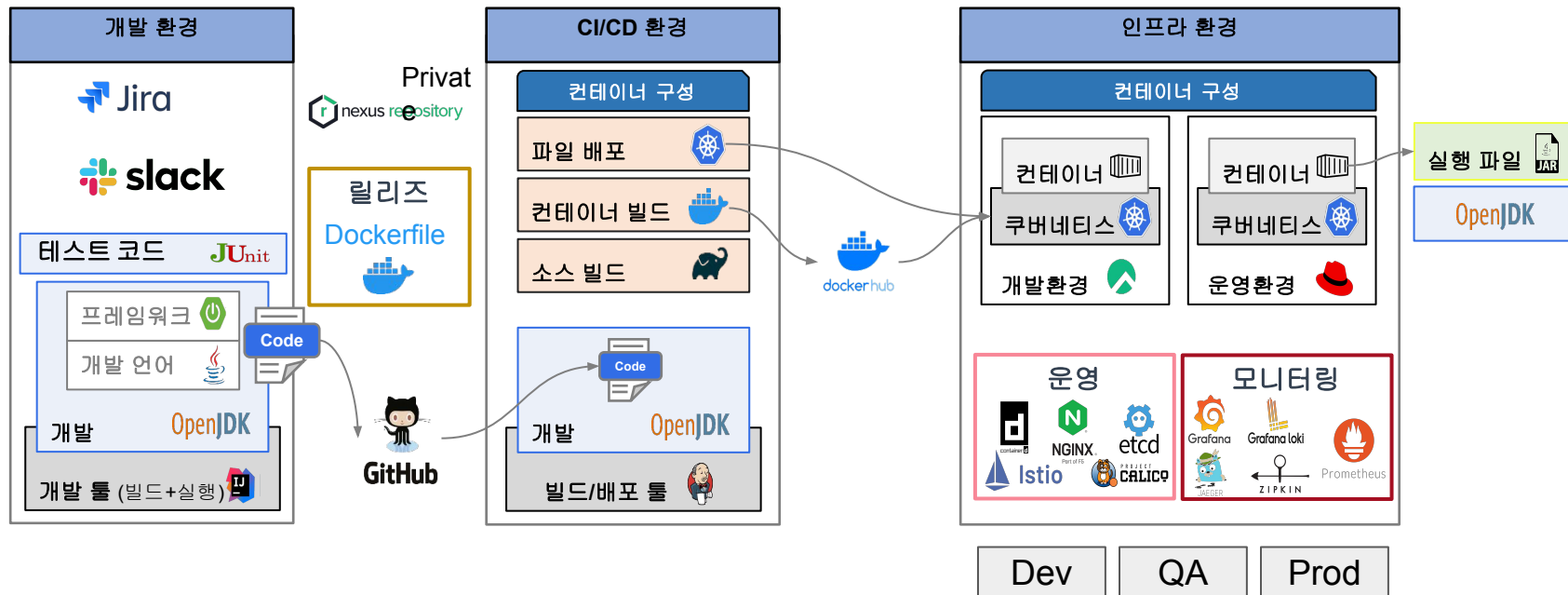
Container Orchestration



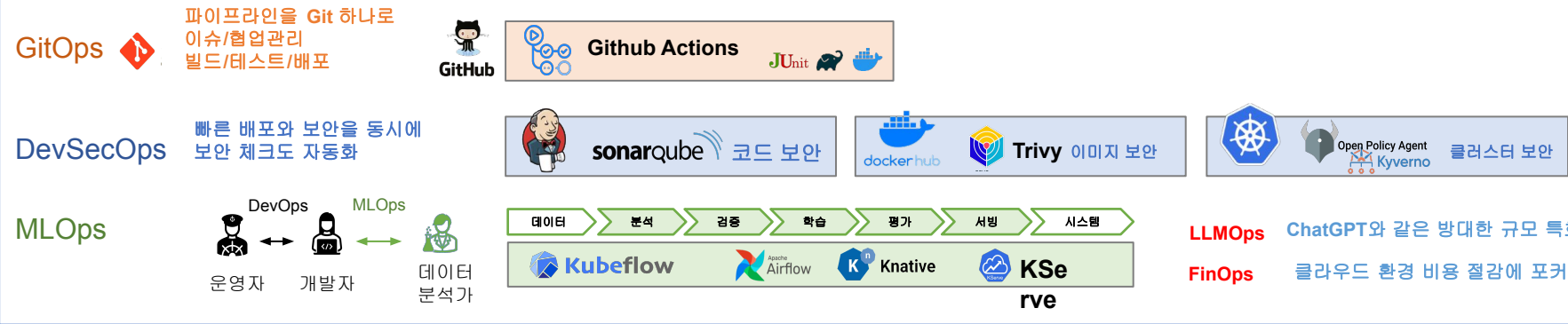
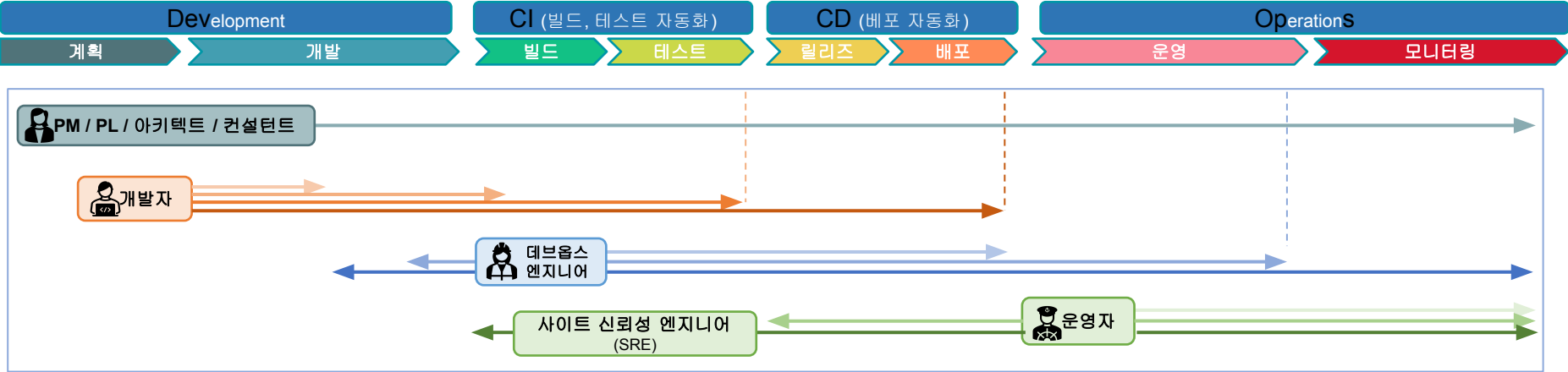
DevOps 구성도



DevOps 구성도



DevOps 구성도



클라우드네이티브+AI 전문가 양성 .

4. 어떤 개발자를 양성할 것인가

도메인 전문가 (내부 전문가)

- IT 리터러시
- 실제 프로젝트 경험

IT 전문가 (기존 개발자)

- 생성형 AI 기술
- 클라우드 네이티브

신입 개발자

- 미취업자(?)

학습할 내용이
너무 많아...

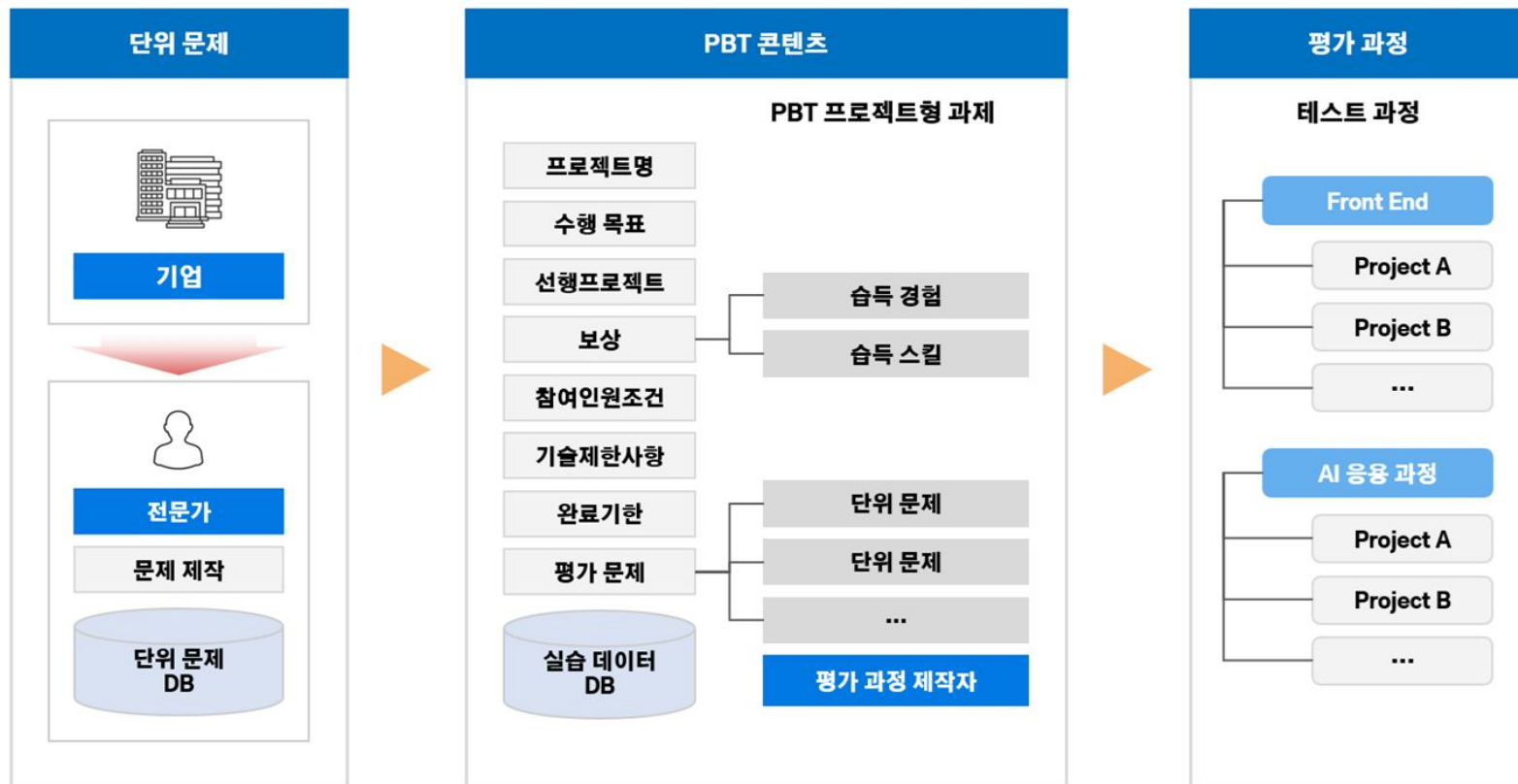
실제 사례를
바탕으로
필요한 것
중심으로

단계별
역량 측정

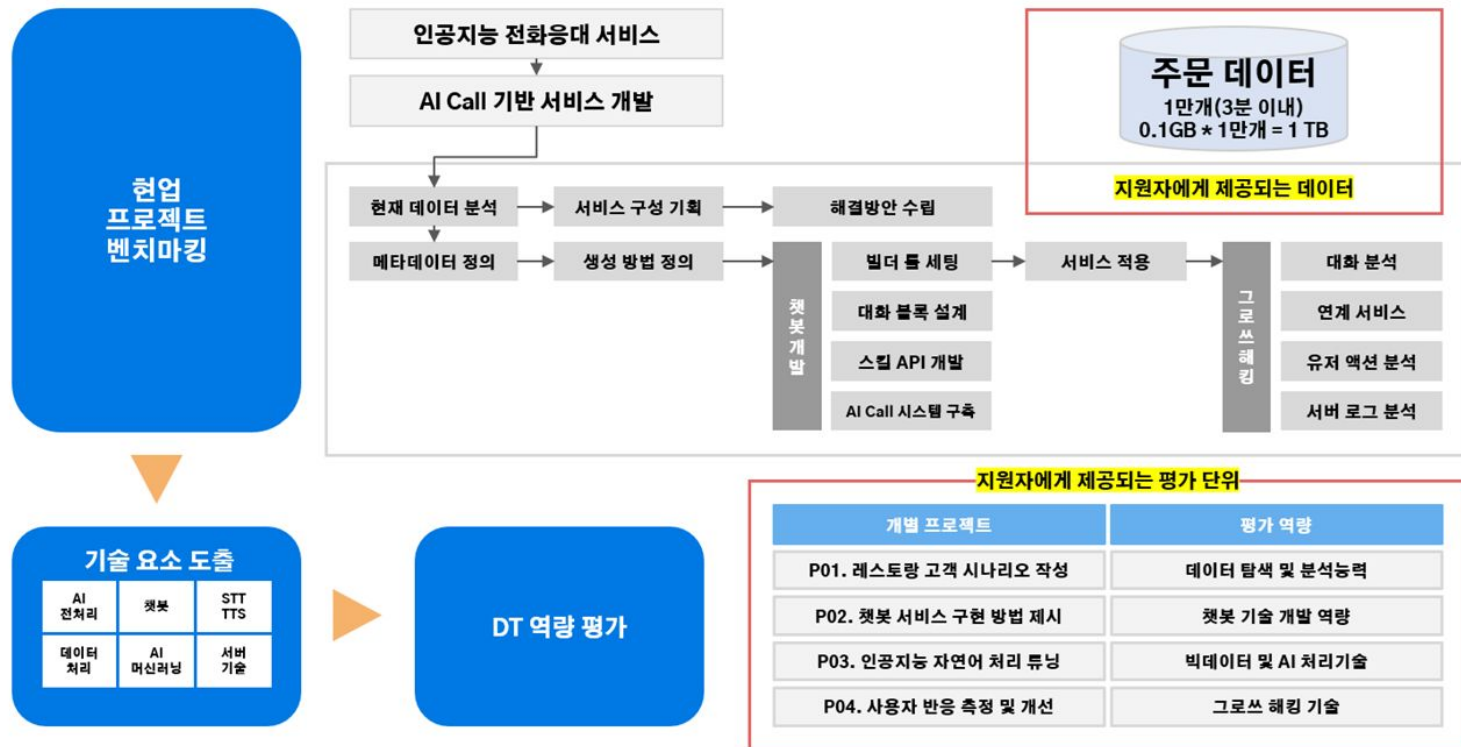
학습 목표를 정확하게 정의

진행 단계	“ DT 직무 역량 정의 ”	“ PBT 평가 문항 개발 ”	“ 역량 평가 및 추천 ”
단계의 특징	기업 DT 업무 수행을 위한 필요 역량 조사 및 관련 기술 지식 정의	DT 직무 실행에 적합한 프로젝트 문제를 정의 하고 평가할 수 있는 시나리오 작성	기술 구현 경험과 Soft-skill을 종합하여 프로젝트형 과제 평가 기준 개발
산출물	가) 기술 지식 List	가) 경험(Tech-skill) 체계도 나) PBL형 과제 샘플	가) Soft-skill 체계도 나) 구현해 봐야 할 경험 List 다) 과제 평가 기준
구조			
역량 수준	<div>Lv1</div> <div>DT 기술 역량 평가</div> <div>DT 문제 해결 평가</div>	<div>Lv2</div> <div>습득한 지식을 바탕으로 문제 해결</div> <div>습득한 지식을 바탕으로 기술 구현</div>	<div>Lv3</div> <div>경험과 지식을 활용하여 주어진 과제 수행</div> <div>Lv4</div> <div>주어진 환경에 맞게 모델 최적화 수행</div> <div>Lv5</div> <div>...</div>

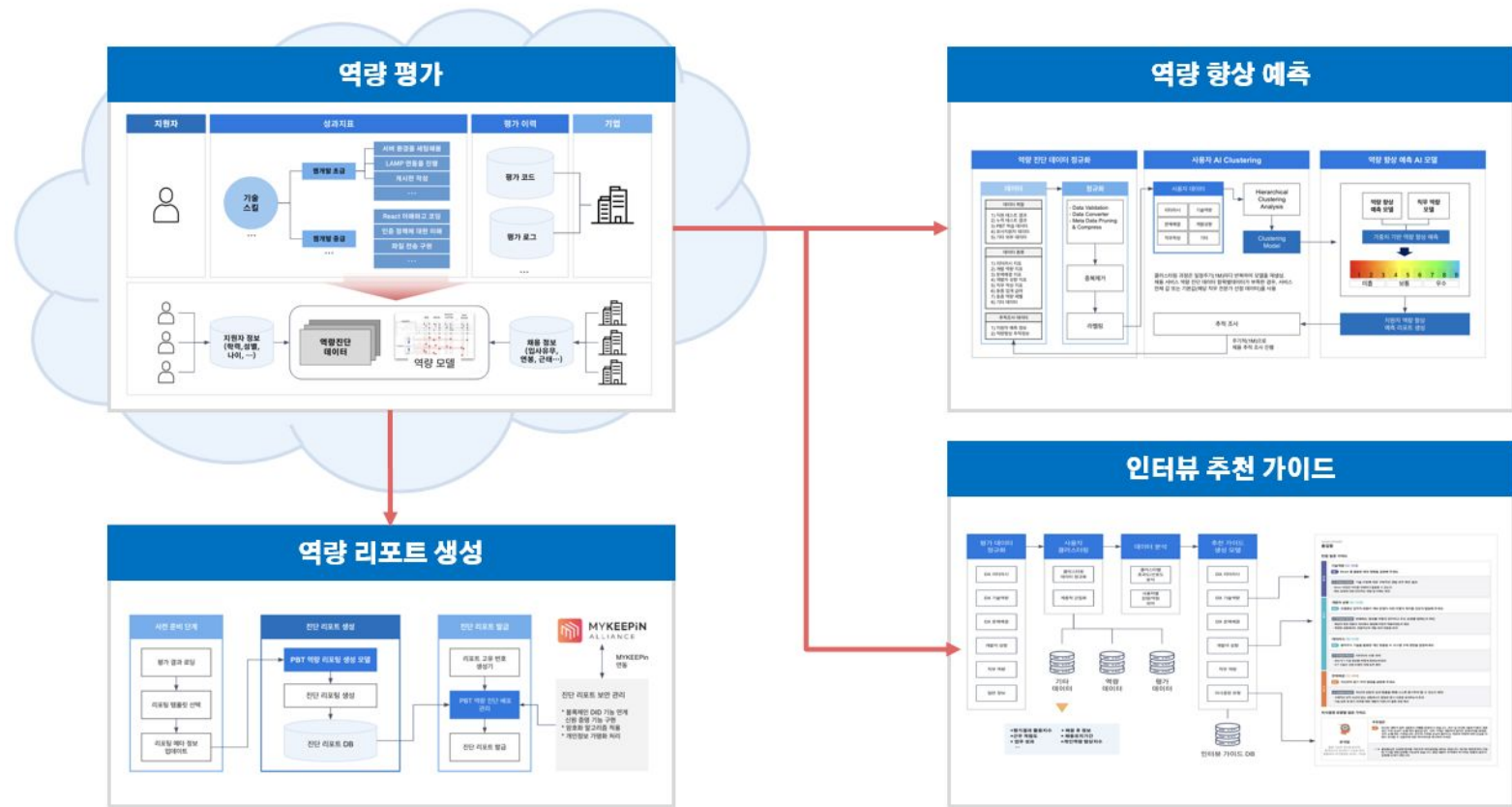
실제 사례를 프로젝트로



네이버클라우드 AlaaS 프로젝트 PBT 문제 도출 및 평가 단위

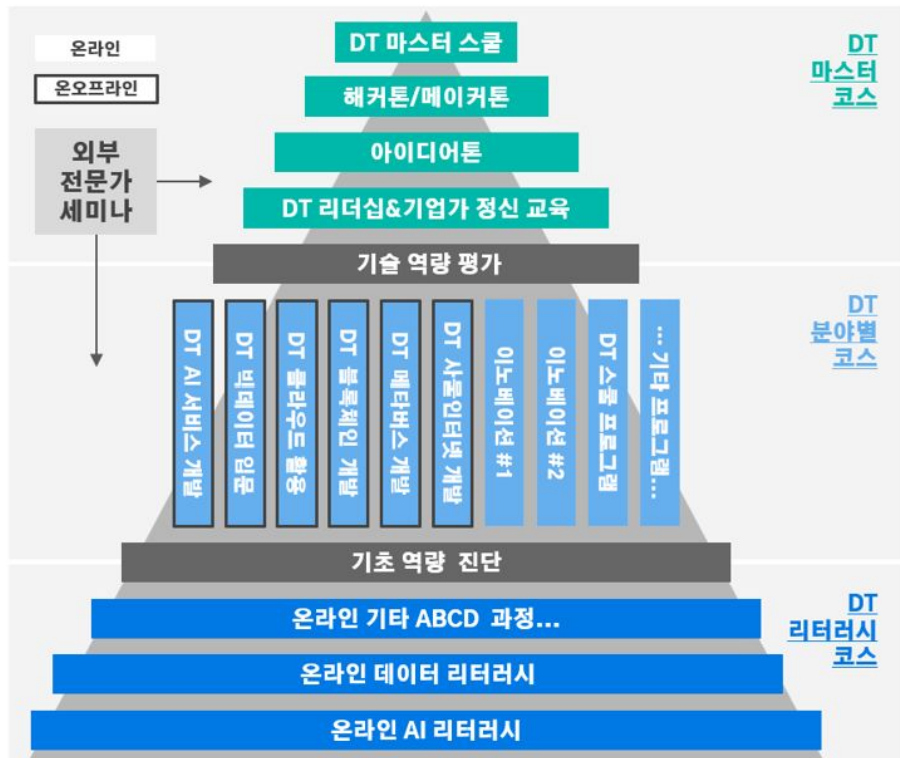


학습 역량을 평가하고 추천하기...



인재 양성 프로세스

DX 리터러시 코스(기초 역량 강화) ⇒ DX 분야별 코스(프로젝트) ⇒ DX 마스터 코스(전문가 발굴)을 통한 체계적인 역량 성장

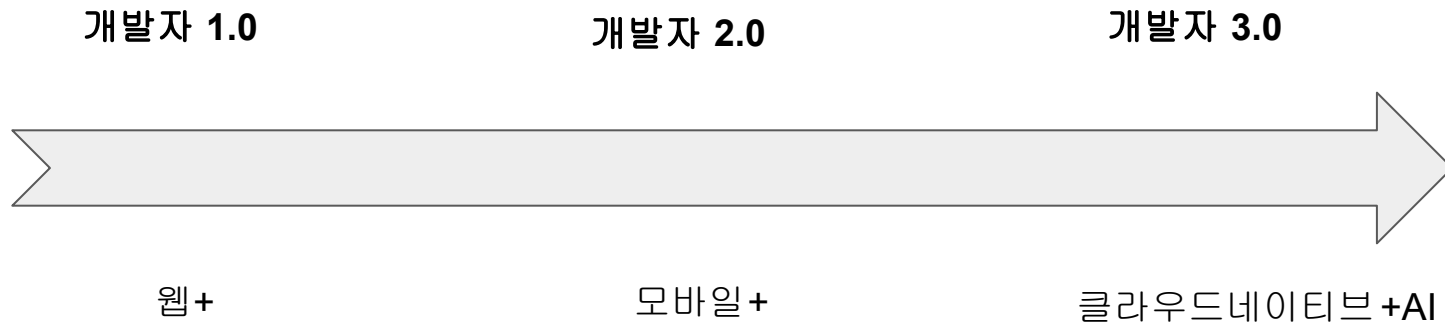


클라우드네이티브+AI 전문가 양성 .

5. 개발자 양성 전략

개발자 세대 구분

시장의 변화에 따라 개발자 정의가 변화되고 있음



에콜42 방식이란?

러닝 피라미드



에콜42 방식이란?

 시장·고객  니즈  제품·서비스



에콜42 방식이란?

기술 지식 카테고리 및 정의



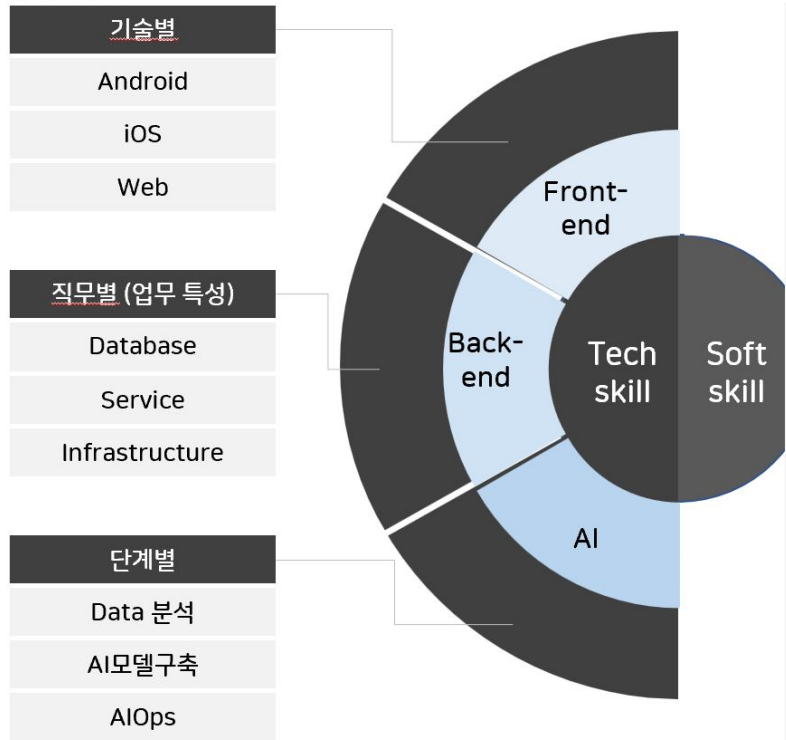
	기본지식: 언어
	목표하는 인터페이스, API, 소스 개발하기 위해 프로그램 작성시 사용하는 기호 체계
	개발 도구
	개발 도구 및 플랫폼을 활용하여 코딩, 디버그, 컴파일, 배포를 수행할 수 있는 환경
	프레임워크
	응용 프로그램의 표준 구조를 구현하는 클래스와 라이브러리의 모임
	라이브러리
	시간과 자원을 절약하며, 원하는 산출물을 빠른 시간안에 구현하도록 돕는 부품의 모음
	관련지식 (개발 리터러시)
	목표하는 결과물을 구현하기 위해 필요한 소프트웨어 공학, 운영체제, 네트워크 등 기초 지식

기술 지식 List

AI					
역량(지식/스킬)	언어	도구(인프라, 개발도구)	프레임워크	라이브러리	리터러시
정의	Tensorflow 등 프레임워크를 활용하여 학습모델을 설계하기 위해 필요한 코드를 잘 만들어내는 능력	개발자의 생산성을 높이기 위한 도구를 사용할 수 있는 능력	플랫폼이 제공하는 기능을 적절히 사용해서 개발하기 쉽고 유지보수가 용이한 앱을 만들기 위한 최적의 동적 구조를 설계할 수 있는 능력	적절한 라이브러리를 활용하여 시간과 자원을 절약하며, 원하는 산출물을 빠른 시간안에 구현할 수 있는 능력	목표하는 결과물을 구현하기 위해 필요한 소프트웨어 공학, 운영체제, 네트워크 등 기초 지식을 활용할 수 있는 능력
도구 활용		클라우드 서비스 (AWS, Azure, GCP) 컨테이너 서비스 (EKS, GKE, AKS)		Pandas, Numpy, Scikit-learn, Matplotlib	
Front-end					
역량(지식/스킬)	언어	도구(인프라, 개발도구)	프레임워크	라이브러리	리터러시
정의	사용자에게 제공하는 UI를 사용자제로 작성할 수 있으며, 플랫폼이 제공하는 적절한 기능을 사용하는 능력	개발자의 생산성을 높이기 위한 도구를 사용할 수 있는 능력	플랫폼이 제공하는 기능을 적절히 사용해서 개발하기 쉽고 유지보수가 용이한 앱을 만들기 위한 능력	기존 플랫폼에 맞게 설계할 수 있는 능력	목표하는 결과물을 구현하기 위해 필요한 소프트웨어 공학, 운영체제, 네트워크 등 기초 지식을 활용할 수 있는 능력
응용	Android	Kotlin / Java		Firestore, Google Maps, Image Library (Glide)	Component Architecture, Network, User Interface
Back-end					
역량(지식/스킬)	언어	도구(인프라, 개발도구)	프레임워크	라이브러리	리터러시
정의	개발 도구를 활용하여 코딩, 디버그, 컴파일, 배포를 수행할 수 있는 능력	개발 도구를 활용하여 코딩, 디버그, 컴파일, 배포를 수행할 수 있는 능력	응용 프로그램의 표준 구조를 구현하는 클래스와 라이브러리의 모임을 사용하여 목표 결과를 산출하는 능력	적절한 라이브러리를 활용하여 시간과 자원을 절약하며, 원하는 산출물을 빠른 시간안에 구현할 수 있는 능력	목표하는 결과물을 구현하기 위해 필요한 소프트웨어 공학, 운영체제, 네트워크 등 기초 지식을 활용할 수 있는 능력
API	파이썬 / JSP / PHP / Ruby on rails / JavaScript	Visual Studio Code / 이클립스	django / Flask / Spring / Laravel / NODE.js	jQuery, Ajax	API구성 언어 (CSV, HTML, XML, JSON, YAML) API형식 (RESTful API, GraphQL)
Database	SQL	RDBMS(SQL Server / Oracle / My SQL / Maria DB) / NoSQLDBS(Mongo DB) / Hadoop			
인프라운영		클라우드 서비스 (AWS, Azure, GCP) 컨테이너 서비스 (EKS, GKE, AKS)			
공통		Git			

에콜42 방식이란?

경험 체계도

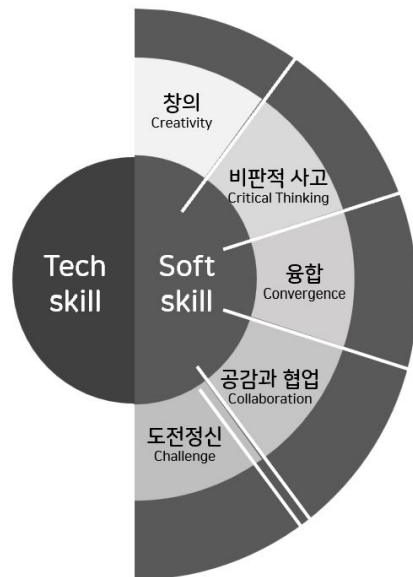


경험 list

[Front-end] Web		
난이도	경험	지식/스킬
1	HTML Array에 있는 데이터를 Adapter class를 이용해서 View에 연결한다.	data adaptor
1	Layout editor에서 제공하는 기본 기능을 통해서 단순한 View를 구성한다.	layout
1	종차적인 항목을 linear layout으로 구성한다.	layout-layout
1	Notification을 화면에 표시한다.	Notifications
1	drawable에 이미지 파일을 넣고 프로그램에서 불러올 수 있다.	resource
1	복제/붙여넣기 Android 기본 View 손작을 한다.	views
1	image view의 drawable을 표시한다.	views
1	edit text를 통해서 사용자 입력 값 기록을 가져올 수 있다.	views
1	체크박스를 이용해서 사용자 입력 값을 받을 수 있다.	views
1	spinner를 이용해서 값을 받을 수 있다.	views
2	Translate/Slide의 Opacity/Scale과 같은 기본적인 동적용 적용하여 View의 변형을 적용한다.	views
2	opacity를 바꿀 수 있다.	views
2	View Holder를 이용해서 항목을 재사용하는 방법을 이해하고 사용한다.	views
2	작성된 화면의 View의 속성을 변경하거나 일부 간단한 배치 변경을 한다.	views
2	scroll view를 통해서 가로/세로의 긴 콘텐츠를 자유롭게 다룰 수 있다.	views
2	Grid layout을 이용해서 만들 수 있다.	views
2	relative layout을 이용해서 상대적으로 views를 배치한다.	views
2	Android Studio로 프로젝트 생성시 Material Design 기본 요소를 적용한다. (Floating Button, Navigation, ...)	views
2	Icon을 변경하거나 View를 이용해서 화면 배치를 변경해서 여러 유형의 Notification을 적용한다.	views
2	텍스트를 소스 코드로 넣지 않고 string 리소스로 분리한다.	views
2	View의 property와 method를 사용한다.	views
3	다수의 또는 연속적인 기본 Animation을 혼합해서 적용한다.	views
3	recyclerview의 view holder를 이용해서 항목의 내용을 제공할 수 있다.	views
3	ViewAdapter와 관동해서 화면의 요소를 보고 이벤트를 받아서 처리한다.	views
3	recyclerview를 이용해서 연속적인 데이터를 배치한다.	views
3	recyclerview를 이용해서 리스트의 개별 항목을 만들 수 있다.	views
3	recyclerview의 view holder를 이용해서 항목의 내용을 제공할 수 있다.	views
3	다수의 또는 연속적인 기본 Animation을 혼합해서 적용한다.	views
3	여러개의 linear layout을 계층적으로 사용한다.	views
3	Material Design의 기본 요소를 프로젝트에 적용한다.	views
3	일정이나, 작업버튼(링크 열기, ... 등)을 추가한다.	views
3	Notification에 이미지를 표현한다.	views
3	long Notification을 표현한다.	views
3	일정버튼을 추가한다.	views
3	작업버튼 (링크 열기/ 기록의 업데이트) 기능을 삽입한다.	views
3	channel을 사용한다.	views
3	API에 따라서 구성(데이터베이스)을 달린다.	views
3	필요한 기능을 탑재하거나 타인의 소인이나 코드를 참조해서 View의 향상된 기능을 사용한다.	views
3	text views를 다양한 크기와 색상으로 표현한다.	views
3	text views를 활용해서 특정한 부분을 강조한다.	views
3	상대적으로 복잡한 Translate 동작을 적용한다.	views
3	Header/Footer를 적용해서 데이터의 화면을 구성한다.	views
3	필요에 따라서 Binding Adapter를 만들어서 적용한다.	views
3	recyclerview를 이용해서 header/footer를 구성한다.	views
3	원하는 Views를 자유 자재로 배치한다.	views
3	Shared elements를 이용한 화면 전환 효과를 적용한다.	views
3	카드와 체크메 지은 다양한 Channel을 적용해서 Notification을 사용한다.	views
3	ViewGroup, ViewPager를 통해서 다수의 View를 다룰 수 있다.	views
3	View에 Theme, Style 등을 통해서 UI를 관리한다.	views
3	수학적 함수를 적용한 특정한 Animation을 개발한다.	views
3	다양한 View Type을 생체해 데이터의 화면을 구성한다.	views
3	Inverse Binding Adapter를 만들어서 적용한다.	views
3	Layout의 특성에 따라 다양한 features 등을 이용할 줄 알고 View Rendering performance를 감안하여 최적화한다.	views
3	Notification의 action에 따라서 동적용 적용하거나 서버로 전송하는 등의 기능을 적용한다.	views
3	CustomView를 생성해서 사용한다.	views

에콜42 방식이란?

Soft-skill 체계도

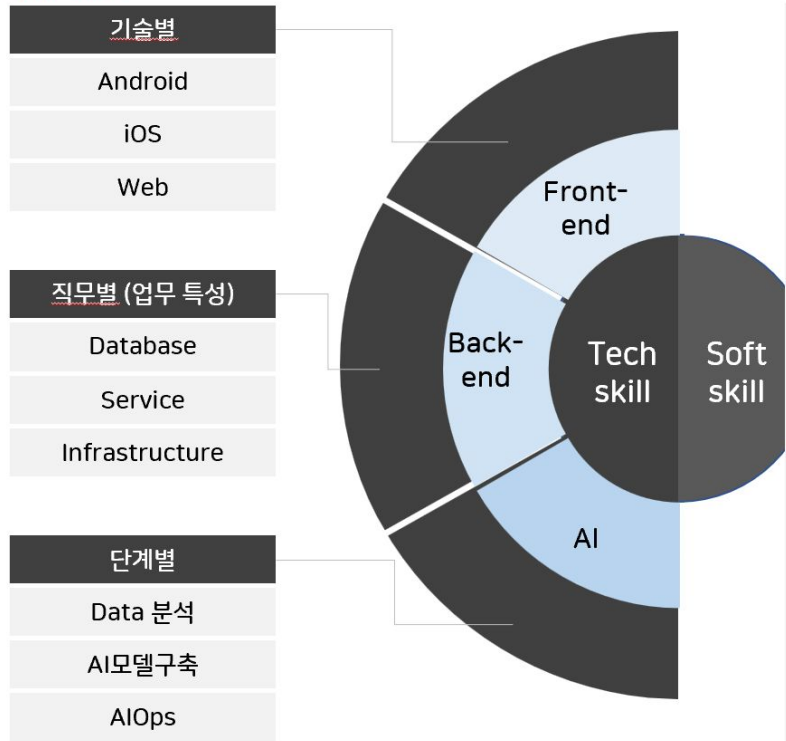


Skill 별 세부 역량

창의	융합	도전정신
새로운 관점으로 결과 해석	지속적 학습	끈기 있는 노력
새로운 아이디어 제안	트렌드 민감성 유지	변화 흐름 모니터링
혁신적 문제 접근	전문 지식의 확장	주도적 변화 실행
비즈니스 아이템 발굴	폭넓은 지식 활용	도전적 목표 설정
비판적 사고	공감과 협업	
정보간 관계 분석	협력적 분위기 조성	
핵심 정보 파악	공동의 의사결정	
근본 원인 규명	의사결정의 질과 속도 관리	
문제 해결방안 모색	회의 결과물 명확화	
올바른 대안 제안 및 수용	체계적 자료 관리	
분석결과 도출	계획의 재조정	
	약속 준수	
	정보 공유	
	공동의 책임감 발휘	

에콜42 방식이란?

경험 체계도



경험 list

[Front-end] Web		
난이도	경험	지식/스킬
1	HTML Array에 있는 데이터를 Adapter class를 이용해서 View에 연결한다.	data adaptor
1	Layout editor에서 제공하는 기본 기능을 통해서 단순한 View를 구성한다.	layout
1	종자적인 항목을 linear layout으로 구성한다.	layout-layout
1	Notification을 화면에 표시한다.	Notifications
1	drawable에 이미지 파일을 넣고 프로그램에서 불러올 수 있다.	resource
1	복제/붙여넣기/삭제/Android 기본 View 손해를 한다.	views
1	image view의 drawable을 표시한다.	views
1	edit text를 통해서 사용자 입력 값 기록을 가져올 수 있다.	views
1	체크박스를 이용해서 사용자 입력 값을 받을 수 있다.	views
1	spinner를 이용해서 값을 받을 수 있다.	views
2	Translate/Slide의 Opacity/Scale과 같은 기본적인 동적용 적용하여 View의 변형을 적용한다.	views
2	opacity를 바꿀 수 있다.	views
2	View Holder를 이용해서 항목을 재사용하는 방법을 이해하고 사용한다.	views
2	작성된 화면의 View의 속성을 변경하거나 일부 간단한 배치 변경을 한다.	views
2	scroll view를 통해서 가로/세로의 긴 콘텐츠를 자유롭게 다룰 수 있다.	views
2	Grid layout을 이용해서 만들 수 있다.	views
2	relative layout을 이용해서 상대적으로 views를 배치한다.	views
2	Android Studio로 프로젝트 생성시 Material Design 기본 요소를 적용한다. (Floating Button, Navigation, ...)	views
2	icon을 변경하거나 View를 이용해서 화면 배치를 변경해서 여러 유형의 Notification을 적용한다.	views
2	텍스트를 소스 코드로 넣지 않고 string 리소스로 분리한다.	views
2	View의 property와 method를 사용한다.	views
3	다수의 또는 연속적인 기본 Animation을 혼합해서 적용한다.	views
3	recyclerview의 view holder를 이용해서 항목의 내용을 제공할 수 있다.	views
3	ViewModel과 관동해서 화면의 데이터를 주고 이관되는 양에서 처리한다.	views
3	recyclerview를 이용해서 연속적인 데이터를 배치한다.	views
3	recyclerview를 이용해서 리스트의 개별 항목을 만들 수 있다.	views
3	recyclerview의 view holder를 이용해서 항목의 내용을 제공할 수 있다.	views
3	서버로부터 데이터를 가져와서 recyclerview의 내용을 제공할 수 있다.	views
3	다른 사람의 sample code나 인터넷 검색을 이용한다.	views
3	여러개의 linear layout을 계층적으로 사용한다.	views
3	Material Design의 기본 원칙을 이해하고 적용한다.	views
3	일정이나, 작업버튼(원고물기, ...) 등을 추가한다.	views
3	Notification에 이미지를 표현한다.	views
3	long Notification을 표현한다.	views
3	일정표를 생성한다.	views
3	작업버튼 (원고물기/ 기록의 일정으로 기능)을 삽입한다.	views
3	channel을 사용한다.	views
3	API에 따라서 구성(데이터베이스)을 달린다.	views
3	필요한 기능을 이해하거나 타인의 조언이나 코드를 참조해서 View의 향상된 기능을 사용한다.	views
3	text views를 다양한 크기와 색상으로 표현한다.	views
3	text views를 활용해서 특정한 부분을 강조한다.	views
4	실제로서 복제된 Translate 동적용을 적용한다.	views
4	Header/Footer를 적용해서 데이터의 화면을 구성한다.	views
4	필요에 따라서 Binding Adapter를 만들어서 적용한다.	views
4	recyclerview를 이용해서 header/footer를 구성한다.	views
4	원하는 Views를 자유 자재로 배치한다.	views
4	Shared elements를 이용한 화면 전환 효과를 적용한다.	views
4	카드와 체크메 지은 다양한 Channel을 적용해서 Notification을 사용한다.	views
4	ViewGroup, ViewPager를 통해서 다수의 View를 다룰 수 있다.	views
4	View에 Theme, Style 등을 통해서 UI를 관리한다.	views
4	수학적 함수를 적용한 특정한 Animation을 개발한다.	views
4	다양한 View Type을 생성해서 데이터의 화면을 구성한다.	views
4	Inverse Binding Adapter를 만들어서 적용한다.	views
4	Layout의 특성에 따라 다양한 features 등을 이용할 줄 알고 View Rendering performance를 감안하여 최적화한다.	views
4	Notification의 action에 따라서 동적용을 적용하거나 서버로 전송하는 등의 기능을 적용한다.	views
4	CustomView를 생성해서 사용한다.	views

에콜42 방식이란?

① 개요

② 기초지식

③ 실습

④ 디자인 씽킹

⑤ 팀 미션

⑥ 발표/회고

#	주제	학습 테마	내용
1	우리동네 커피숍	데이터 수집-분석-시각화-예측	서울시내 스타벅스 매장 정보 분석을 시작으로, 우리동네 커피숍을 차릴 경우 입지 분석에 대한 내용을 AI 모델로 개발하고, 이를 지도에 표시하며 데이터 시각화에 대한 내용을 파악
2	AI를 이용한 주식 투자	<u>머신러닝</u> 기초와 데이터 분석	주가 예측을 위한 자료 수집과 분석을 통해, 예측형 AI 모델 개발 역량을 향상시키고, 실제 주식 정보를 바탕으로 주가 예측 AI 모델을 개발
3	농작물 병충해 분석을 위한 AI	딥러닝 CNN 기술의 이해 전이학습 PBL 학습	토마토 등의 농작물에 발생하는 병충해 종류를 파악하고, 해당 문제 해결을 위한 데이터를 수집해 정리한 후, 병충해 분석 및 해결을 위한 AI 모델을 개발
4	숨은 그림 찾기 AI 봇	딥러닝 CNN 기술의 이해 <u>AutoML</u> 서비스 이해	'월리를 찾아라' 게임을 AI 기술을 사용해 새롭게 정의하고, 다양한 분야 응용을 위한 역량 향상을 위한 PBL 학습 능력을 함양

에콜42 방식이란?

우리동네 커피숍



학습 목표

- 데이터 수집과 분석을 통한 예측형 AI 개발 능력 함양
- 부동산 관련 데이터 수집과 분석을 통한 흥미 유발
- 스타벅스 커피숍 입지 분석 실습을 바탕으로 커피숍 창업 지원용 AI 예측 모델 개발

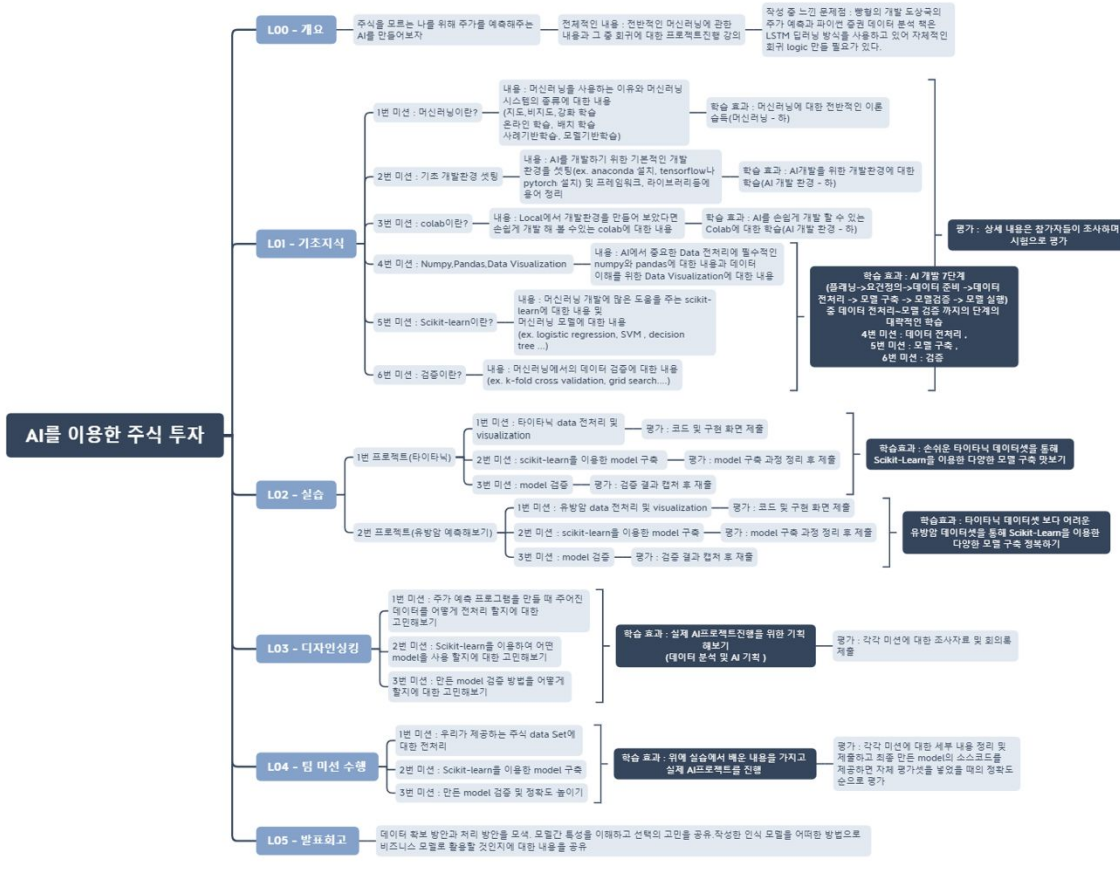
제공 범위

- 스타벅스 및 커피숍 AI 구현을 위한 데이터셋 일체
- 관련 실습 프로그램 진행을 위한 소스코드 및 매뉴얼 일체

프로젝트 문제 구성 SET

1레벨 (기초지식)	2레벨 (실습)	3레벨 (디자인 씽킹)	4레벨 (미션수행)	5레벨 (미션수행)
1. <u>파이썬</u> 기초 문제	1. 스타벅스 데이터 수집	1. 스타벅스 매장수와 인구수 비교	1. 커피숍 입지 추천 AI 설계하기	1. 공공데이터 취합
2. 데이터 <u>크롤링</u> 기초	2. 스타벅스 데이터 <u>전처리</u>	2. 스타벅스 매장수와 <u>사업체</u> 수 비교	2. 프로그램 UI/UX 설계	2. 커피숍 입지 추천 AI 모델 개발하기
3. 웹 데이터 크롤링	3. 스타벅스 데이터 분석	3. 커피숍 입지 분석 요소 파악하기	3. 수요를 통한 부동산 분석 (인구수, 경제활동 등)	3. 추천 AI 웹 개발
4. 웹 사이트 만들기	4. 데이터 시각화	4. 입지 예측에 적합한 AI 알고리즘 선정하기	4. 공급을 통한 부동산 분석(공공시설, 기업체 등)	4. 발표 및 회고
	5. 스타벅스 매장 지도 시각화			

에콜42 방식이란?



에콜42 방식이란?

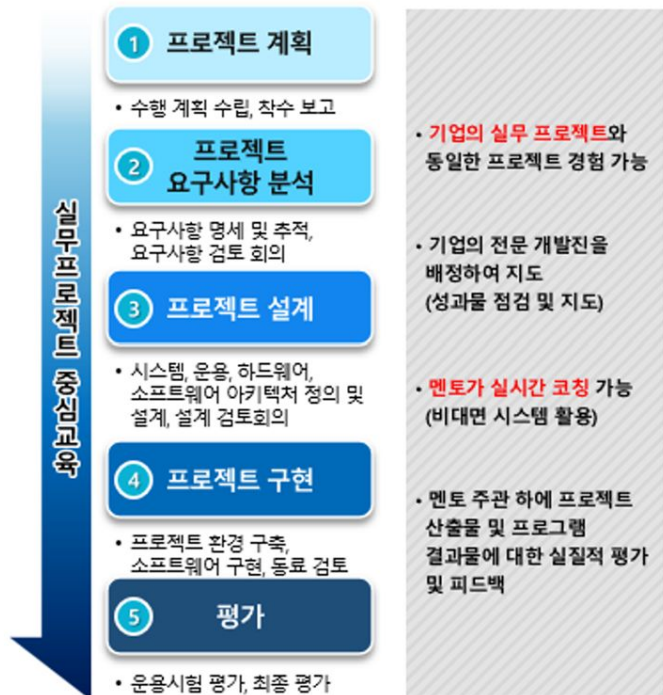
학습 속성 및 효과

*AI 개발 역량 모델 Edu1~Edu4 수준 설계

구분	미션 속성	학습수준	역량 수준	학습 효과
1레벨	기초지식	하	Edu1	<ul style="list-style-type: none">- 딥러닝 기초 학습- Keras를 활용하기- 딥러닝 오버피팅 및 학습 파라미터 이해- AI 이미지 인식 처리 기술의 이해
2레벨	실습	중	Edu1	<ul style="list-style-type: none">- MNIST를 활용한 이미지 인식 기술의 이해- Keras 기반 AI 모델 구축과 검증의 이해
3레벨	디자인 씽킹	중	Edu2	<ul style="list-style-type: none">- 숨은 그림 찾기 AI 봇 데이터 분석 및 활용- 모델 개발시에 주요 고려사항 파악- AutoML 서비스 연계 방안의 이해
4레벨	미션수행	고	Edu3	<ul style="list-style-type: none">- 숨은 그림 찾기 데이터 전처리 기법의 이해- 모델 구축과 검증 능력 함양
5레벨	발표	고	Edu4	<ul style="list-style-type: none">- 웹 및 오픈소스하드웨어 버전 개발- 발표 및 회고를 통한 성장

에콜42 방식이란?

시장의 변화에 따라 개발자 정의가 변화되고 있음



웹 프로젝트

- 게시판 기반 서비스 구축 프로젝트
- 외부 API와 공공 데이터 활용한 병원 찾기 서비스 프로젝트
- PT 이용권 관리 서비스 프로젝트
- 훈련생별로 팀 구성 3개 주제를 과정 중 모두 수행

DevOps 프로젝트

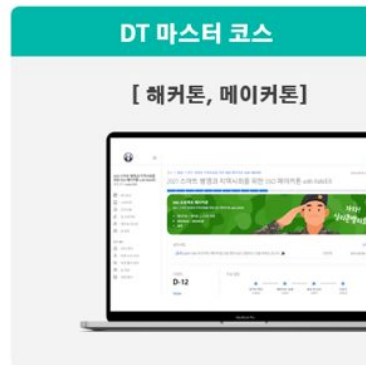
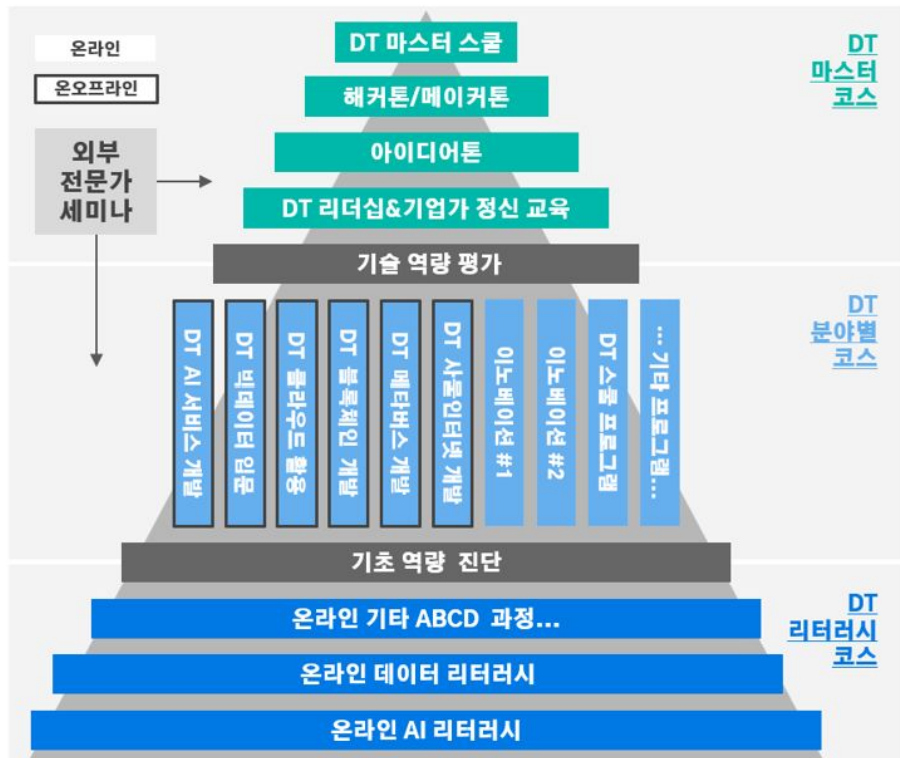
- 코드를 통한 인프라 관리 프로젝트
- 지속적 배포/지속적 통합 구현 프로젝트
- 애플리케이션 성능 테스트 및 모니터링 프로젝트
- 훈련생별로 팀 구성 3개 주제를 과정 중 모두 수행

MSA 기반 프로젝트

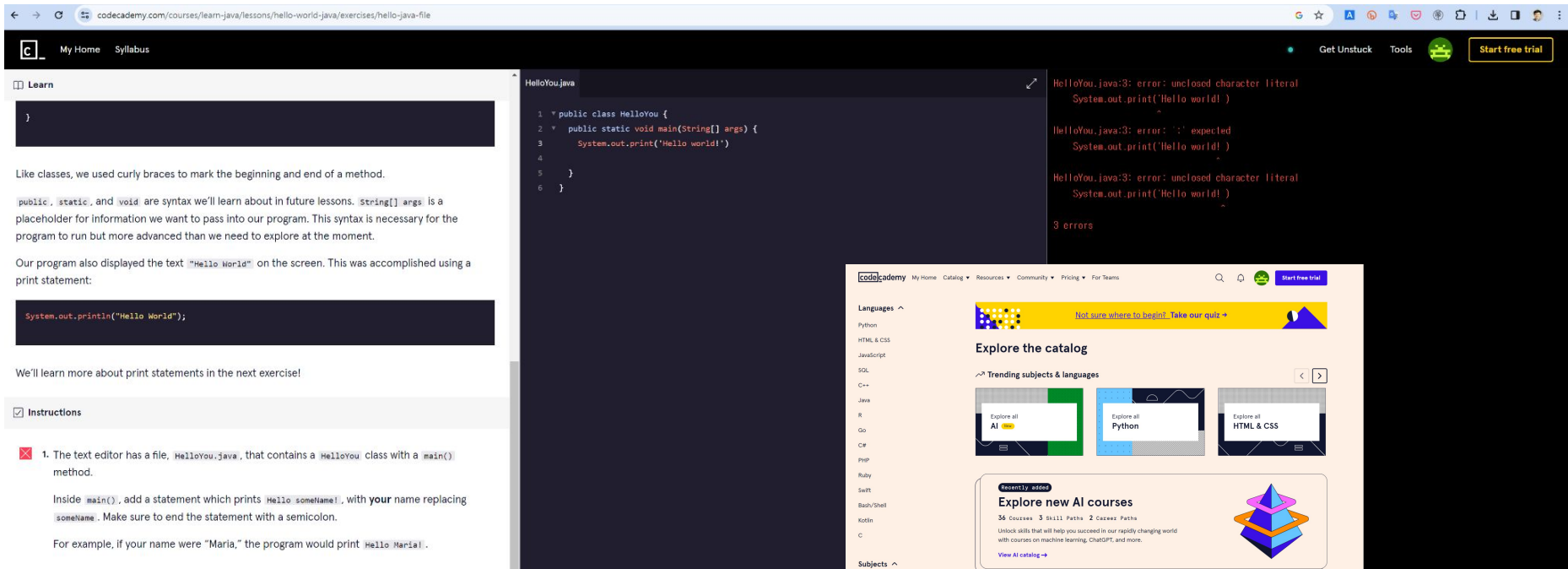
- MSA 기반 구글 캘린더 클론 프로젝트
- MSA 기반 블록체인 NFT 서비스 개발 프로젝트
- MSA 기반 대용량 트래픽을 고려한 이커머스 프로젝트
- 훈련생별로 팀 구성, 3개 주제 중 1개 선택

인재 양성 프로세스

DX 리터러시 코스(기초 역량 강화) ⇒ DX 분야별 코스(프로젝트) ⇒ DX 마스터 코스(전문가 발굴)을 통한 체계적인 역량 성장



리터러시



<https://www.codecademy.com/catalog>