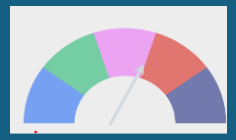
GEN AI LEARNING

Github Repository link --https://github.com/Abhishekrai129

$13.7 billion in 2023 to **$165 billion** by 2032.



CAGR: **40.2%** from 2021 to 2028.

75% of customers prefer AI solutions that integrate easily with existing system

60% are concerned about data privacy and security.

65% of customers expect solutions to provide instant responses

80% of users prefer tailored recommendations

Value Creation and Monetization in Gen AI

VALUE PROPOSITIONS

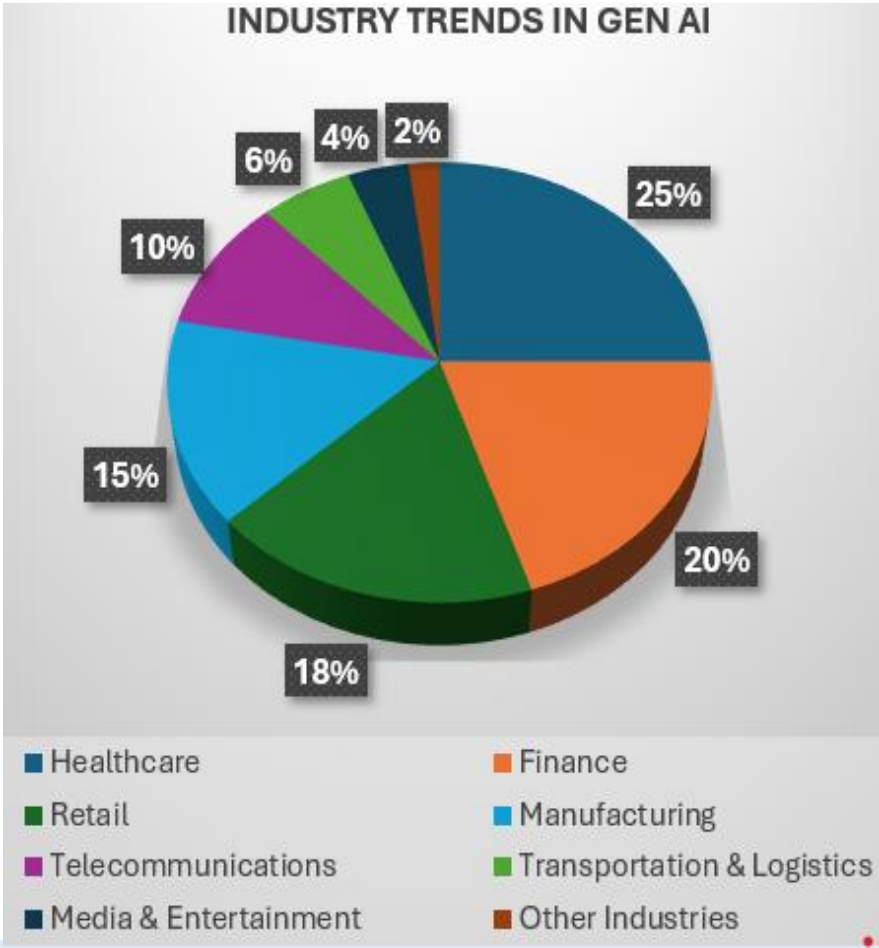
- **Data-Driven Insights**: 70% improve decision-making (McKinsey)
- **Increased Efficiency**: 40% time saved (Harvard Business Review)
- **Cost Reduction**: Up to 30% lower costs (Deloitte)
- **Personalized Experiences**: 80% customer satisfaction (Epsilon)
- **Scalability**: 50% faster growth (Gartner)

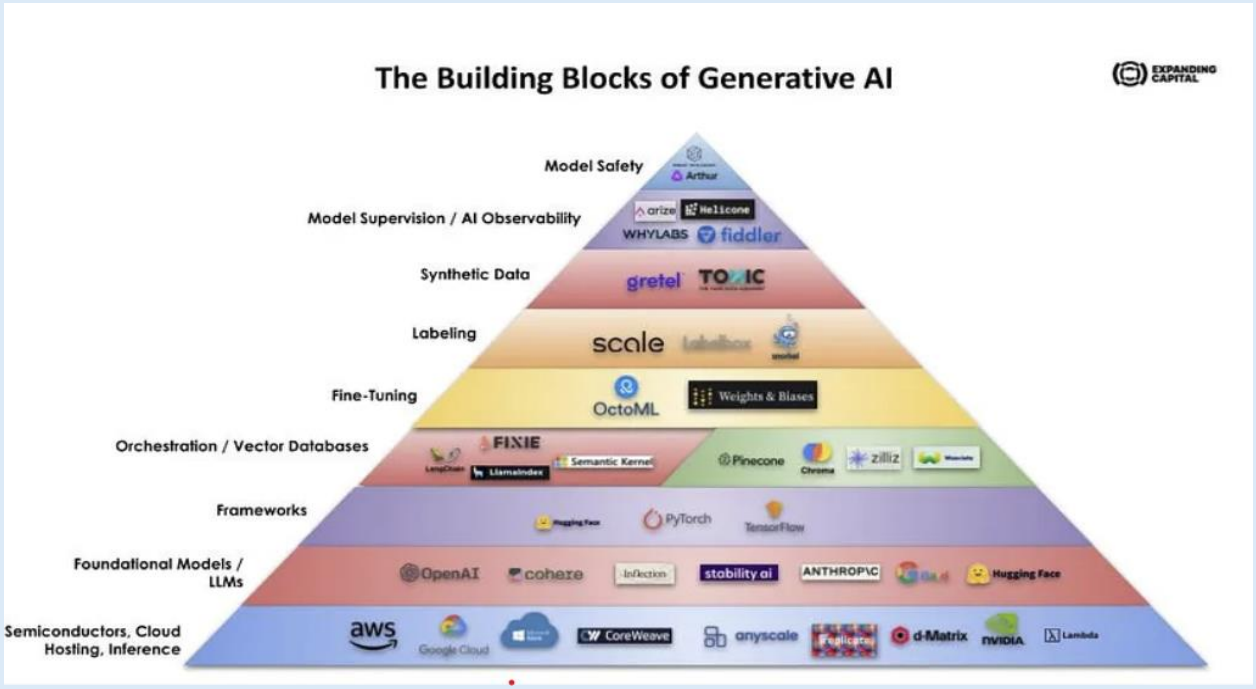
CUSTOMER SEGMENTS

- **Enterprises**: 85% using AI (PwC)
- **SMBs:** 40% adopting AI tools (Forrester)
- **Developers**: 60% prefer AI integrations (Stack Overflow)
- **Startups**: 70% leverage AI for growth (TechCrunch)
- **Public Sector:** 50% using AI solutions (Accenture)

REVENUE STREAMS

- **Subscription Fees:** $300 billion market (Statista)
- **Licensing:** 20% revenue from licensing (Bain & Company)
- **Consulting Services:** $200 billion industry (IBISWorld)
- **Training Programs**: 60% of clients prefer training (Learning Guild)
- **Data Monetization**: $1 trillion potential (McKinsey)





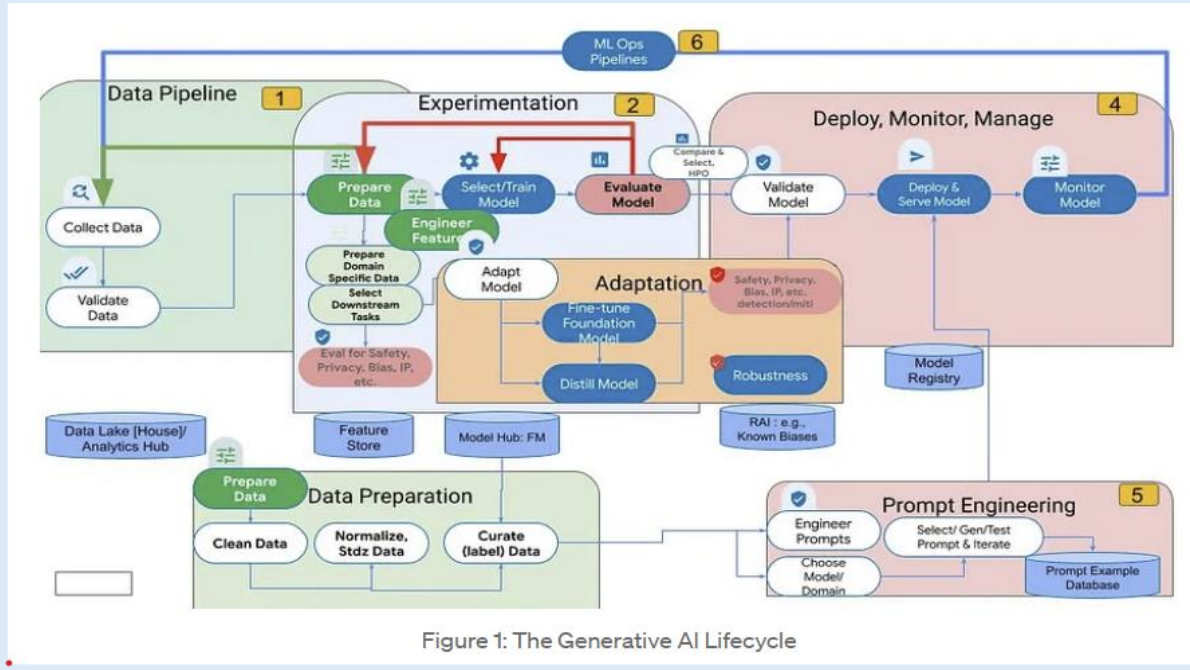
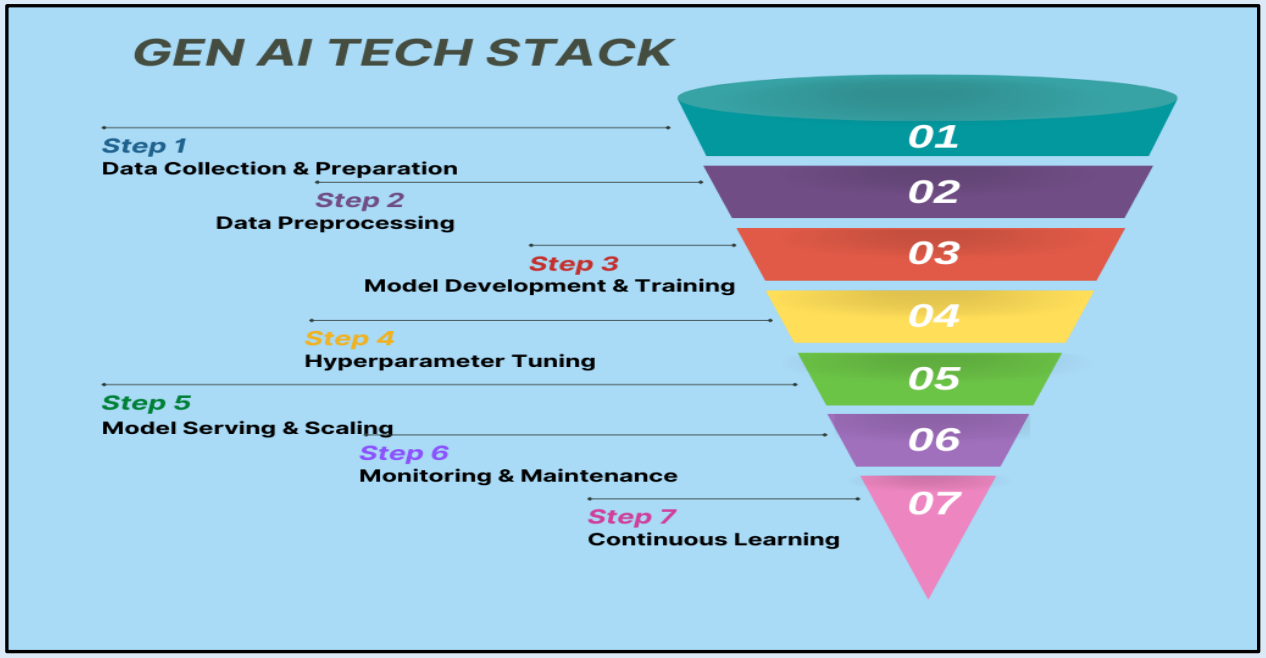


Figure 1: The Generative AI Lifecycle



🔍 **Data Collection**-*Gather and store datasets.*
Tools: Scrapy, Twitter API, AWS S3, Snowflake

🖌️ **Data Preprocessing**-*Clean and prepare data.*
Tools: Pandas, PySpark, SpaCy, NLTK

📊 **Model Development**-*Train machine learning models.*
Tools: TensorFlow, PyTorch, Hugging Face Transformers,

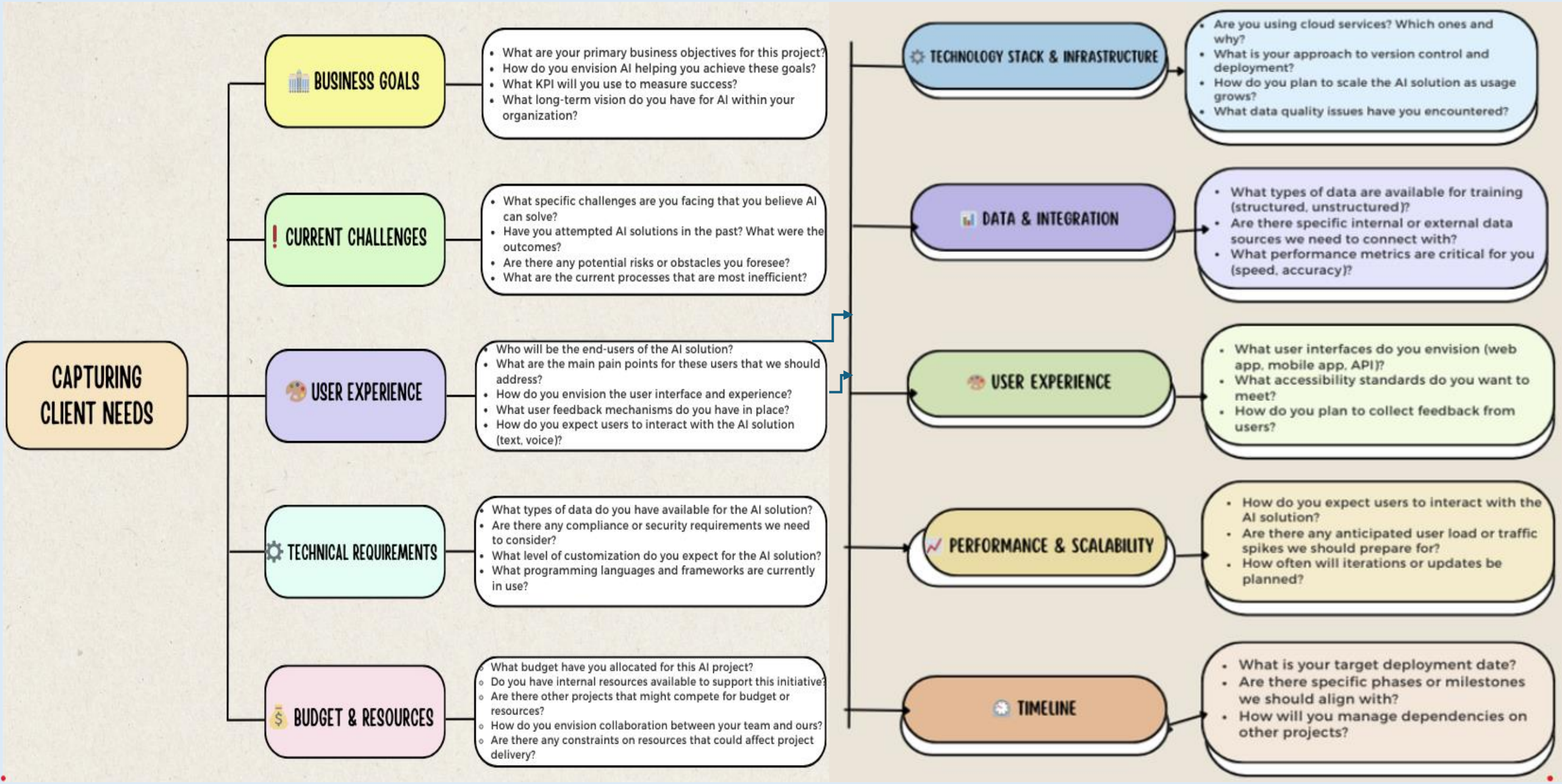
⚙️ **Hyperparameter Tuning**-*Optimize performance.*
Tools: Optuna, Ray Tune, Hyperopt

🚀 **Model Serving**-*Deploy models.*
Tools: AWS SageMaker, Docker, TensorFlow Serving, Kubernetes

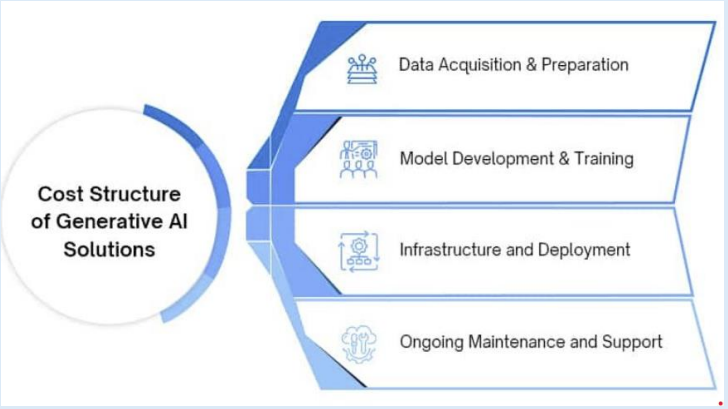
📈 **Monitoring=***Track model performance.*
Tools: Prometheus, Grafana, ELK Stack, Apache Kafka

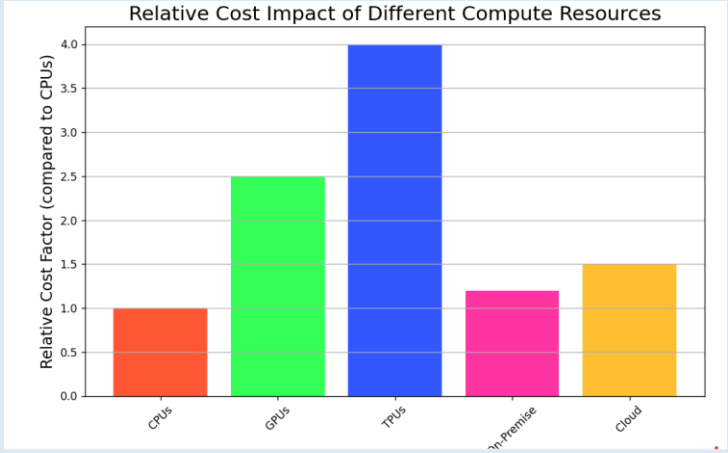
🔄 **Continuous Learning**=*Improve models continuously with new data.*
Tools: MLflow, DVC, A/B Testing (Optimizely)

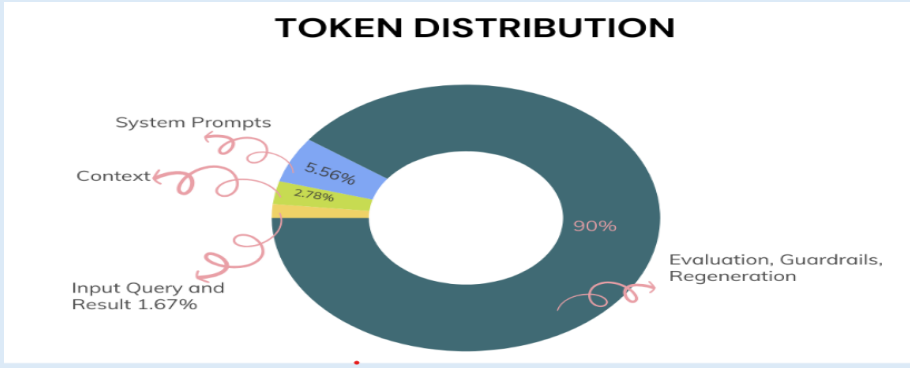
Understanding Client Needs for Generative AI Projects

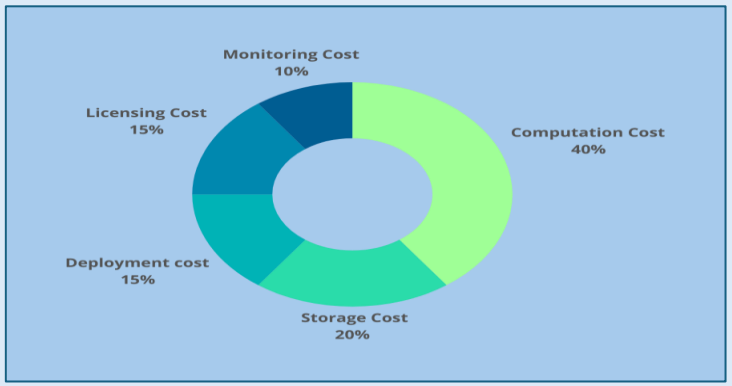


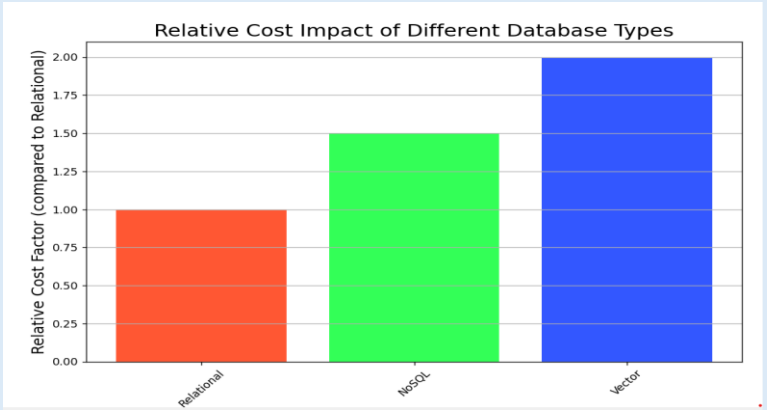
GEN AI Project COST



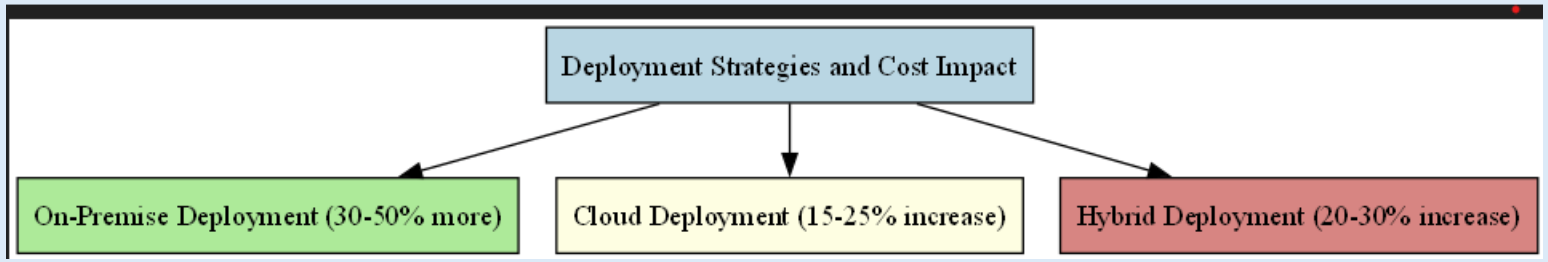








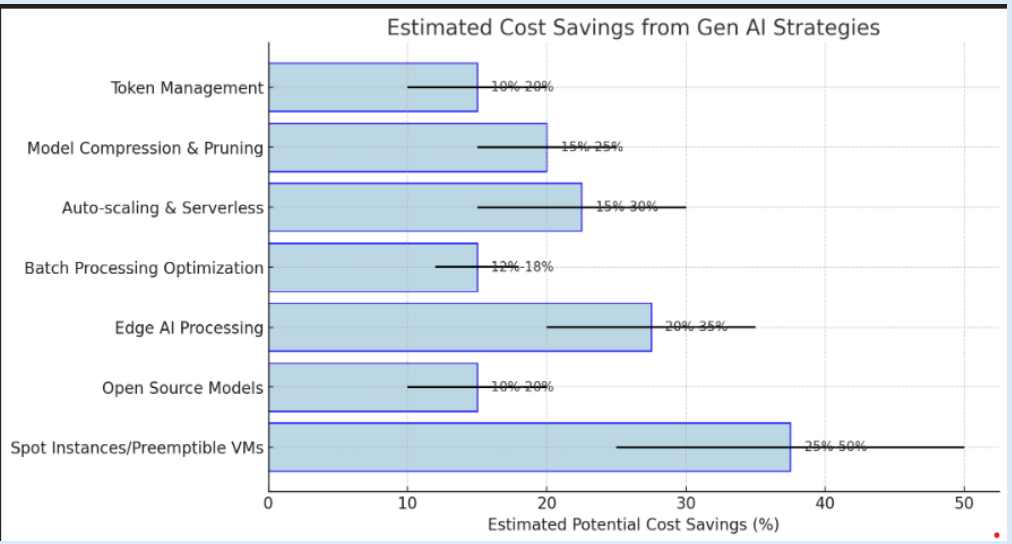
Component	Description	Example Tokens
Input/Output Tokens	Basic user queries and AI responses.	300
Context Tokens	Reference previous interactions for better responses.	500
Metadata/System Prompts	User info and instructions for task accuracy.	600
Evaluation Tokens	Guardrails for quality and safety.	100
Error Handling/Regeneration	Tokens for retries or clarification.	300+
Total Tokens per Interaction	Sum of all token costs.	1,800
Annual Cost (10 million interactions)	Cost for GPT-4 example.	$465,000

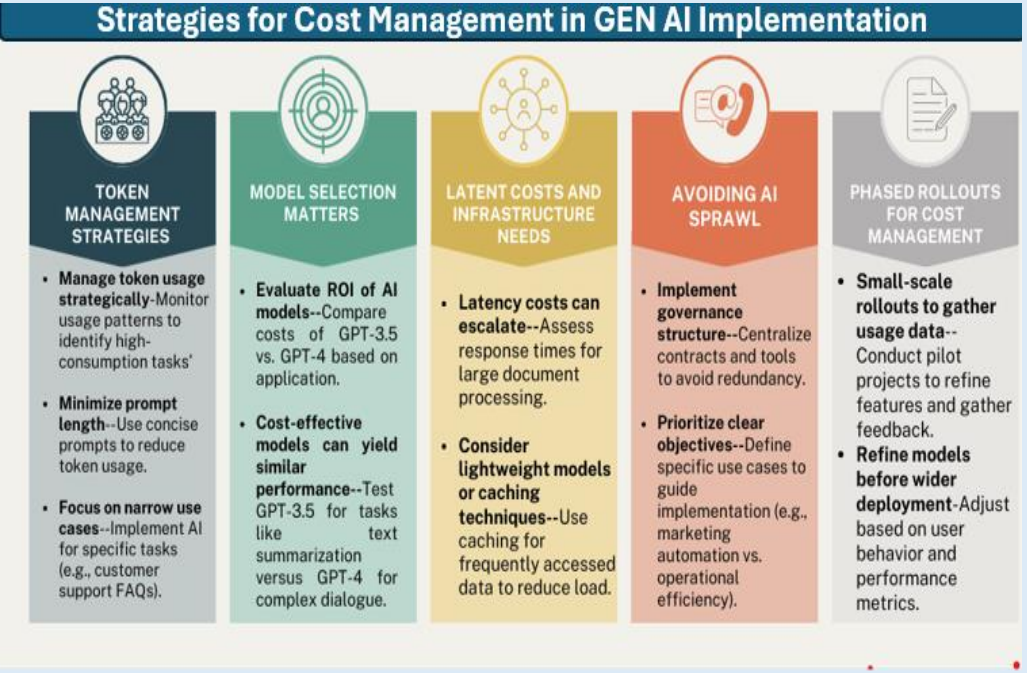


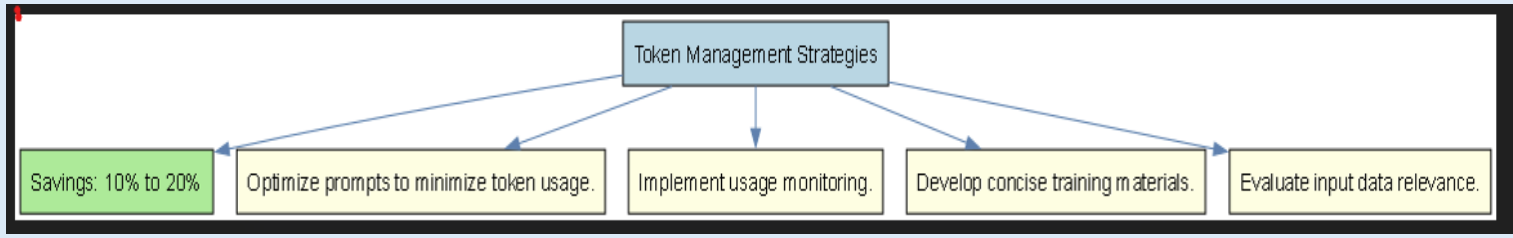
Available GPUs (by cloud provider)

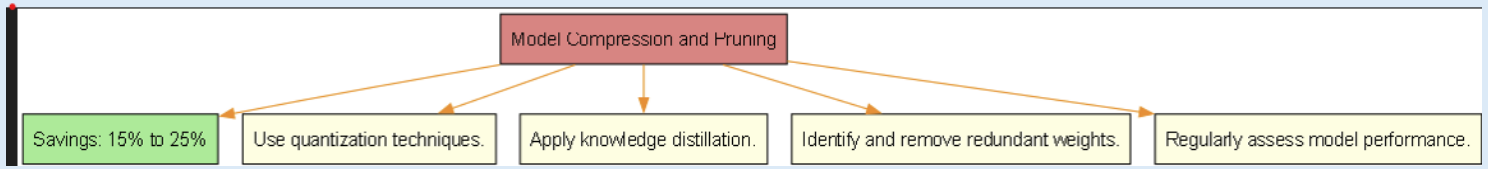
Cloud Service Provider (CSP)	Available GPUs	Pricing Indicator (A100 40gb, on-demand)
AWS	A100, V100, M60, T4, A10, Trainium, Inferentia, Gaudi, V520	$4.10/h (p4d.24xlarge)
Azure	H100, A100, V100, P100, K80, M60, P40, T4, A10, MI25	$3.40/h (ND96asr A100 v4)
Oracle	A100, V100, P100, A10	$3.05/h (BM.GPU4.8)
Google	A100, V100, P100, K80, T4, P4, TPUv4	$2.93/h (NVIDIA A100 40GB)
Coreweave	H100, A100, V100, A40, A6000, A5000, A4000, Quadro RTX 4k/5k	$2.06/h (A100 40GB NVLINK)
Lambda Labs	A100, V100, A6000, A10, Quadro RTX 6k	$1.10/h (1x NVIDIA A100)
FluidStack	A100, V100, A40, A6000, A5000, QUADRO RTX 5K/4K, RTX 3090, RTX 3080, RTX 2080Ti	$1.73/h (A100 40 GB)

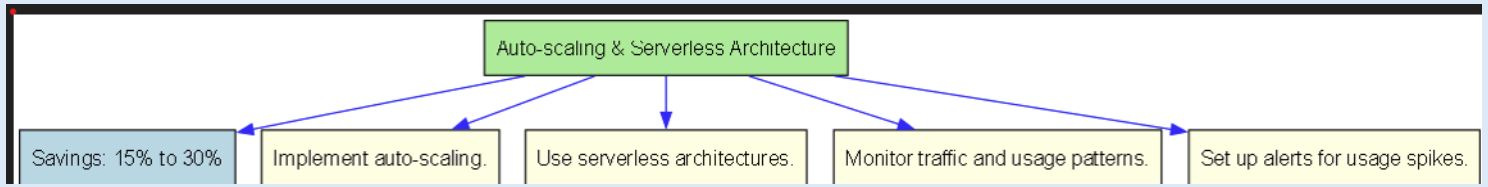
Strategies for optimizing Costs

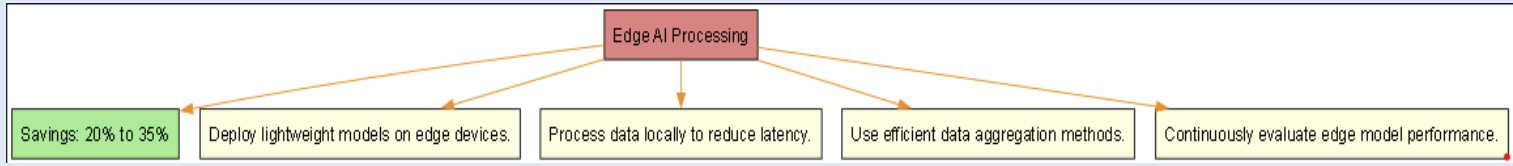


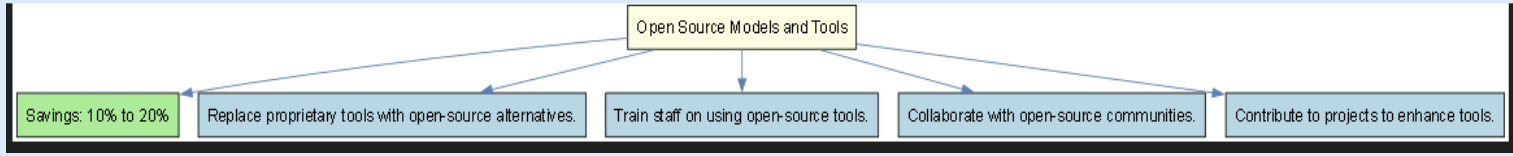


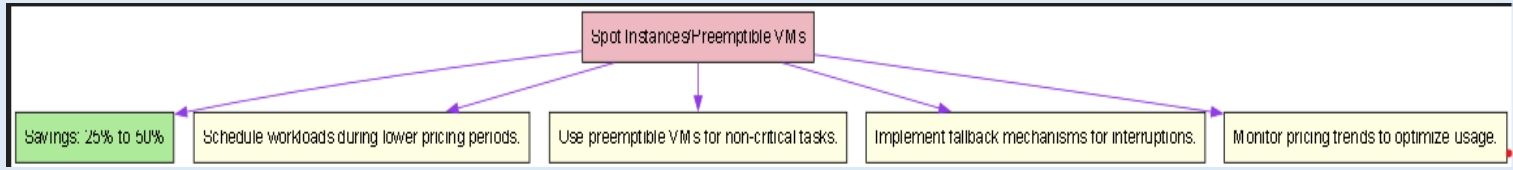




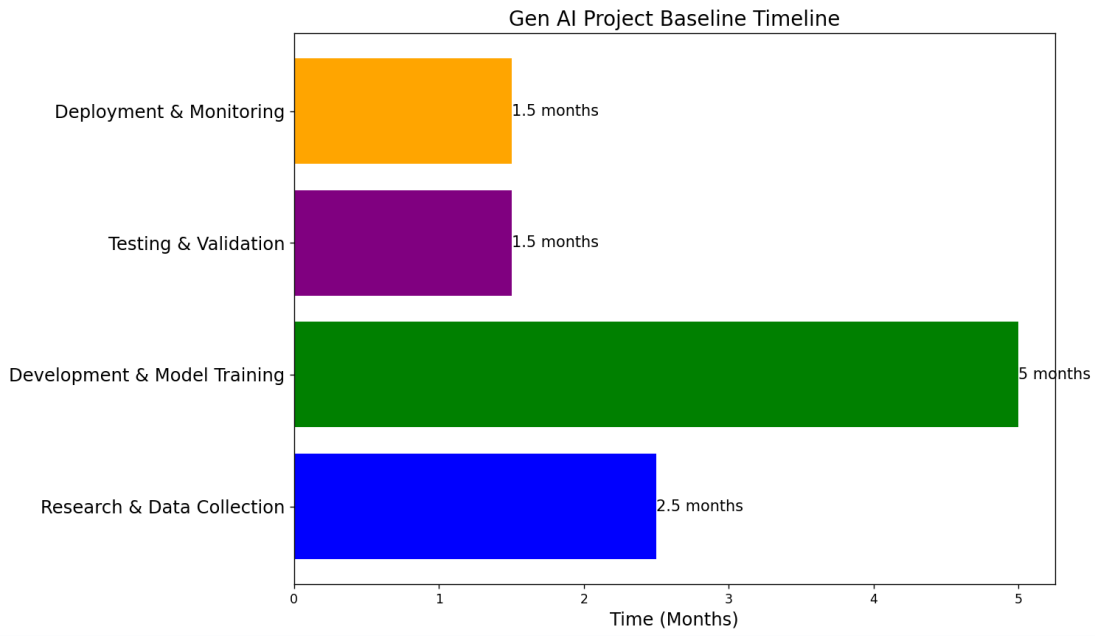


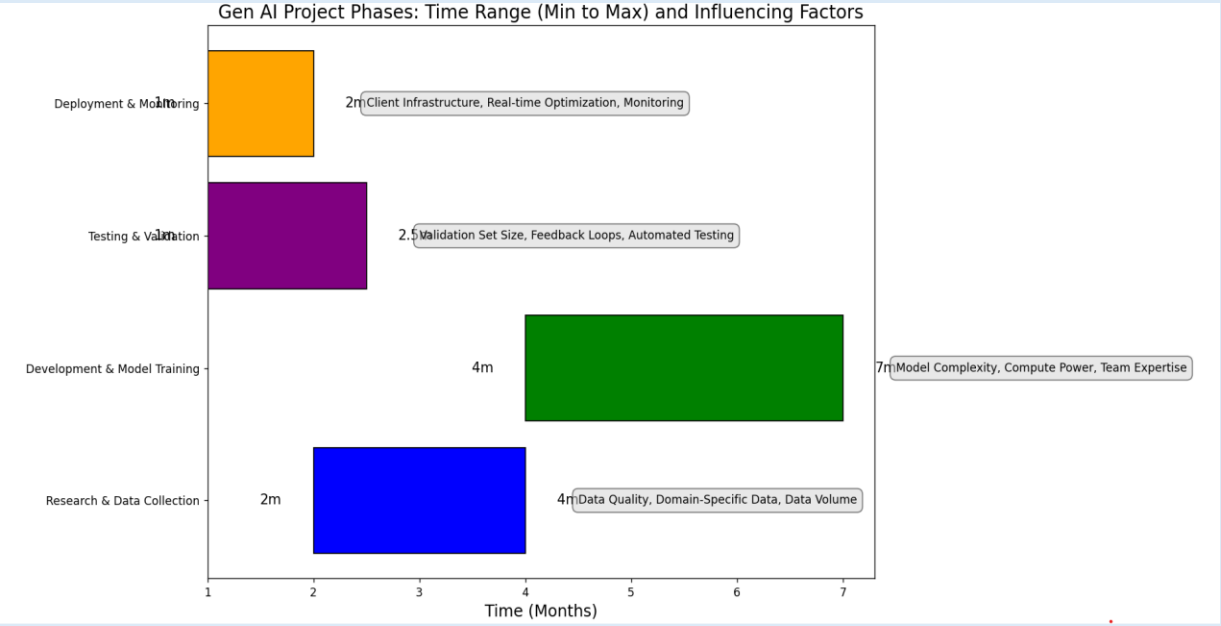


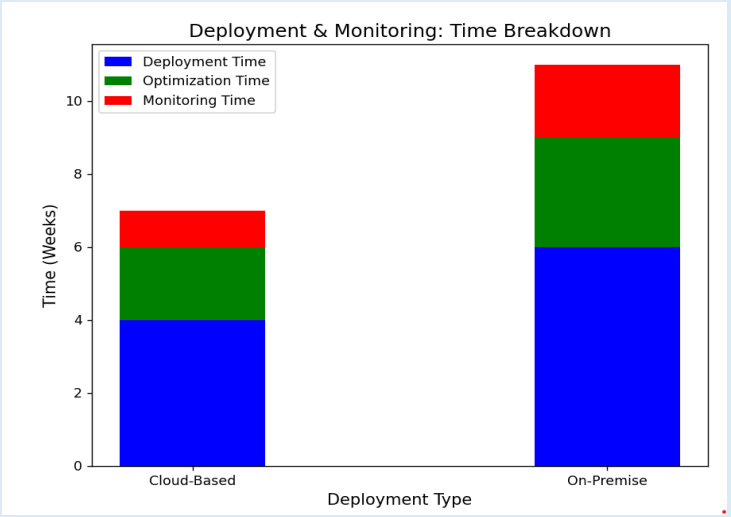


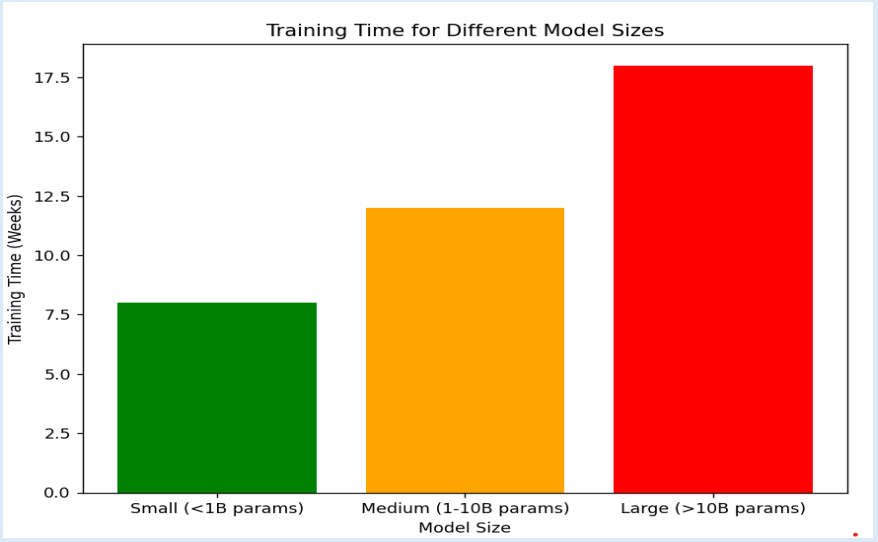


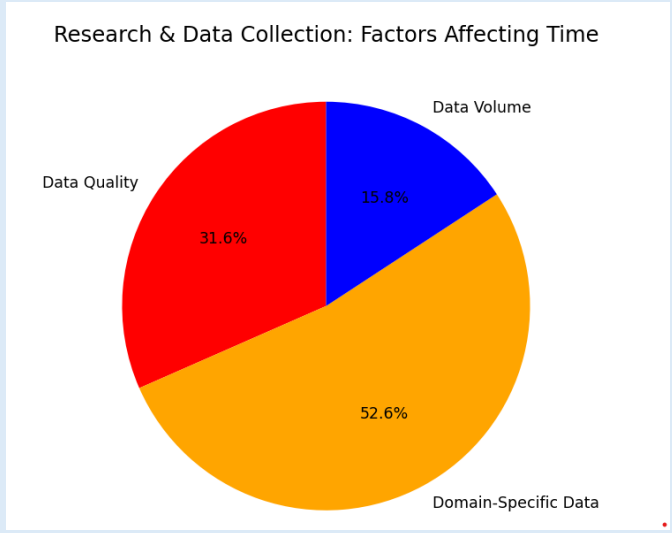
GEN AI SCHEDULE



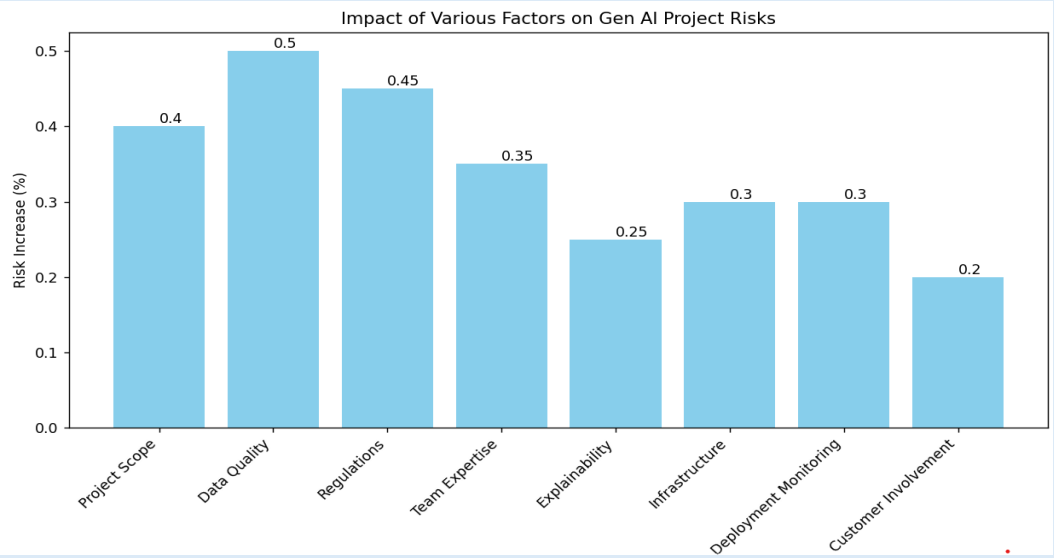


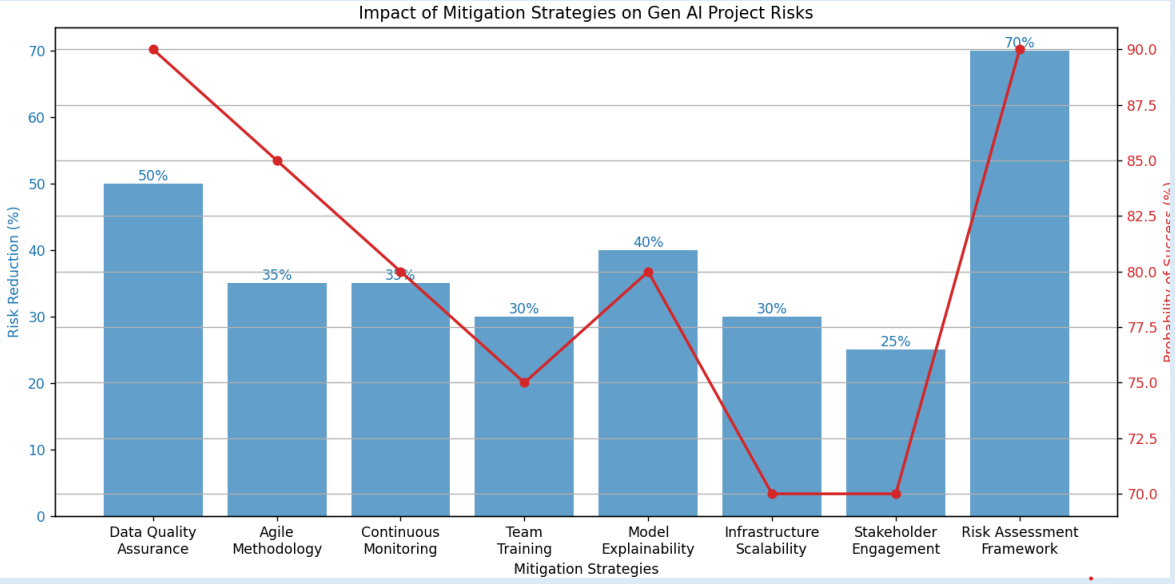






GEN AI RISK





Different generative AI use cases are associated with different kinds of risk.

Primary risk

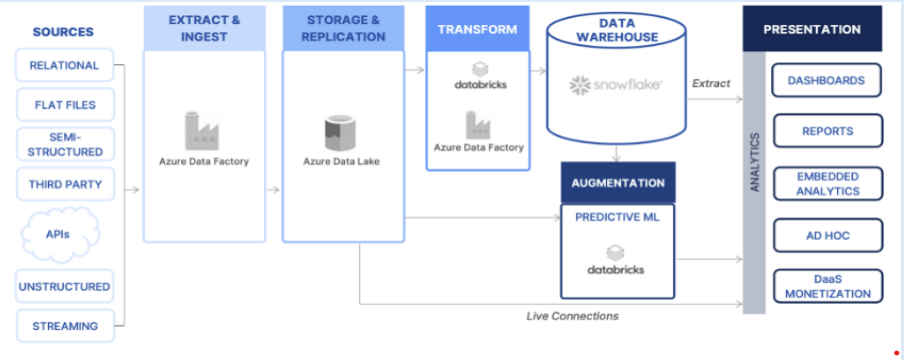
Generative AI use case	Impaired fairness	IP[1] infringement	Data privacy and quality	Malicious use	Security threats	Performance and 'explainability'	Strategic
Customer journeys *(eg, chatbots for customer services)*	✓		✓			✓	✓
Concision *(eg, generating content summaries)*	✓	✓				✓	
Coding *(eg, generating or debugging code)*		✓		✓	✓	✓	
Creative content *(eg, developing marketing content)*	✓	✓		✓		✓	

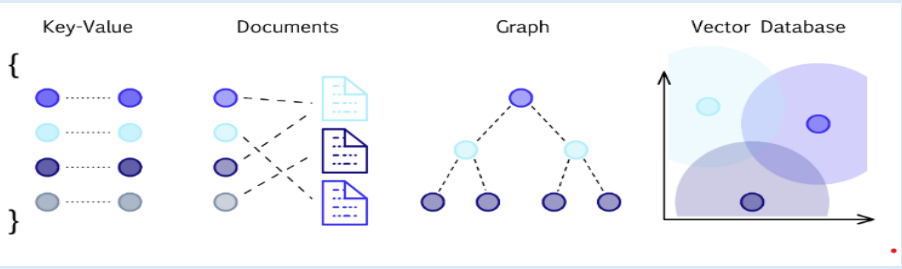
Generative AI risk can be mitigated at multiple points across a user interaction.

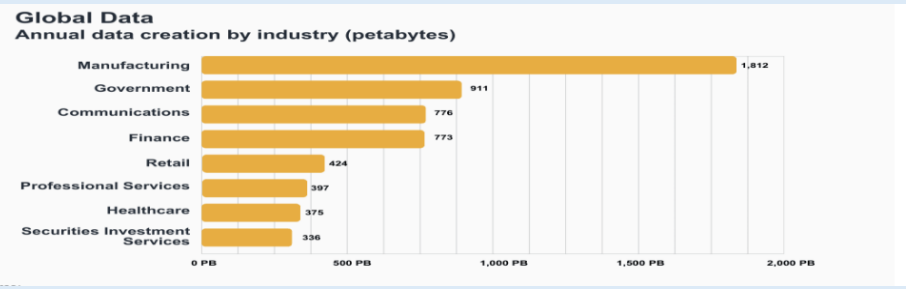
Sample HR chatbot interaction with built-in checkpoints to catch potential misfires

1 User asks question	2 Chatbot clarifies question	3 Prompt is enriched	4 Bot searches relevant knowledge	5 Bot generates response	6 Answer is consumed
User asks, "What are my vision care benefits?"	Model seeks to better understand user's needs by asking for more data: • Permanent or temporary employee? • Eyewear or checkups? • State of residence? • Benefits plan subscription?	Model enriches query by retrieving templates from a prompt library (all templates have been tested against performance benchmarking); after receiving updated prompt, user is asked to confirm that query has been parsed accurately	Model searches for relevant information to answer the prompt	Model generates response (explicitly citing HR documents used to compose it) and conducts final set of risk and quality checks	User receives response and, thanks to citations, can easily verify answer and dig deeper if necessary

DATABASE







Critical capability	Why this matters	SingleStore Cloud OLTP+OLAP vector capable	Pinecone vector database	Weaviate vector database	Milvus vector database	MongoDB Atlas NoSQL OLTP vector capable	Cosmos DB NoSQL vector capable	ElasticSearch full-text search
SPEED	**Immediate insights; Responsive application experience for end users**							
In-memory performance	Responsive applications and instantaneous analytics	●	◔	◔	●	○	○	○
Streaming data ingestion	Immediate queryability of streaming data from multiple sources (files, Kafka, Spark, HDFS or object stores such as S3)	●	◔	◔	●	◒	●	○
Columnstore	Low-latency (~10s of milliseconds) on complex queries	●	○	○	◒	○	○	○
Analytics on semistructured data	High-performance analytics on relational and semi-structured data in the same database engine	◕	◔	◔	◒	◒	○	●
SCALE	**Adapt to growing needs**							
Horizontal scalability	Distributed (shared-nothing) architecture that decouples storage and compute to allow scaling using low-cost infrastructure	●	◕	●	●	●	●	●
Read replicas/ multi-region deployments	Run multiple workloads and scale compute instances across shared databases	●	◕	◕	●	●	●	●
Resiliency	Run critical applications and workloads; Mitigate risks on business operations and reputation.	●	◕	◕	◒	●	●	◔
Deploy anywhere	Ability to deploy both as a fully managed cloud service or self-managed on-premises	●	○	●	●	●	○	●
SIMPLICITY	**Minimize complexity and costs**							
SQL-powered OLTP + OLAP with Zero ETL	Minimize data movement and duplication; minimize complexity and costs emanating from sprawl; power and simplicity of SQL for CRUD and rich query operations.	●	○	○	○	○	○	◒
Multi-model	Ability to store and query multiple secondary data formats (full-text search, JSON, time series, geospatial, etc.)	●	○	○	○	◒	●	◔
Vector search engine	Efficiently handle large amounts of vector data for vector similarity search	◒	●	●	●	◒	◔	◕
Open- source software	Community-developed software that's typically free to use and distribute	○	○	●	●	◒	◕	◕

1. Pure Vector Databases

Milvus · zilliz · Pinecone · vespa · marqo · qdrant · Weaviate · Chroma · Vald · LanceDB

2. Full-text search databases

This category includes databases such as Elastic/Lucene, OpenSearch and Solr.

elasticsearch · APACHE LUCENE · OpenSearch · Apache Solr

4. Vector-capable NoSQL databases

DataStax Astra · cassandra · Azure Cosmos DB · MongoDB. · neo4j · redis · ROCKSET

5. Vector-capable SQL databases

SingleStore · ClickHouse · kinetica

GEN AI KEY METRICS

Model Type	Success Metrics
1. NLP & LLM (Natural Language Processing)	- Accuracy - Precision - Recall - F1 Score - BLEU Score - ROUGE Score - Perplexity - AUC-ROC
2. Computer Vision	- Mean Average Precision (mAP) - Intersection over Union (IoU) - F1 Score - Accuracy - Recall - Precision
3. Audio and Speech Processing	- Word Error Rate (WER) - Signal-to-Noise Ratio (SNR) - F1 Score - Precision and Recall - Mel-Frequency Cepstral Coefficients (MFCCs)
4. Generative Models (GANs, VAEs, etc.)	- Inception Score (IS) - Fréchet Inception Distance (FID) - Mean Squared Error (MSE) - Visual Turing Test
5. Reinforcement Learning & Robotics	- Cumulative Reward - Average Reward per Episode - Success Rate - Learning Curve - Task Completion Rate

LLM

Requests Per Minute
Maximum concurrent requests the service can handle per minute. Critical for large workloads.

Time To First Token (TTFT)
Time before the service returns the first token. Important for interactive applications.

Inter-Token Latency (ITL)
Average time between consecutive tokens. Affects responsiveness.

End-to-End Latency
Total time to complete a request. Key overall speed measure.

Cost Per Request
Critical to assess cost-effectiveness for production usage.

Computer Vision

Frames Per Second (FPS)
Measures the number of frames the system can process per second. Essential for real-time video processing.

Inference Time
The time taken to run inference on a given image. Important for applications needing fast results.

Model Accuracy
Evaluates the correctness of the model's predictions. Often measured using metrics like precision, recall, and F1-score.

Throughput
The number of images processed per second. Critical for high-throughput applications like surveillance.

Cost Per Inference
Calculating the cost-effectiveness per image processed. Important for budgeting and scaling.

Image or Content Gen.

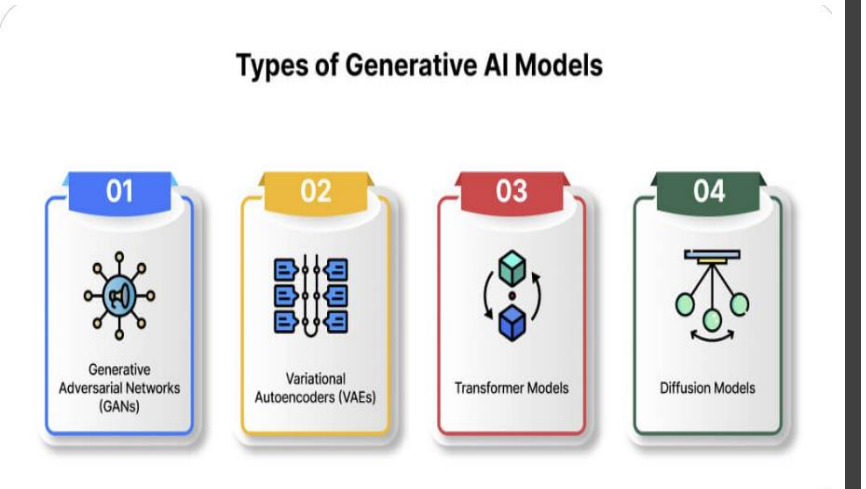
Content Quality
Often assessed through human evaluations or automated metrics like BLEU or ROUGE (for text), or Inception Score and FID (for images).

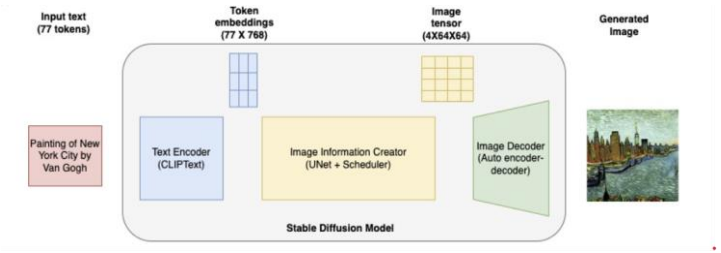
Generation Speed
Time taken to generate an image or a text segment. Impacts usability in time-sensitive applications.

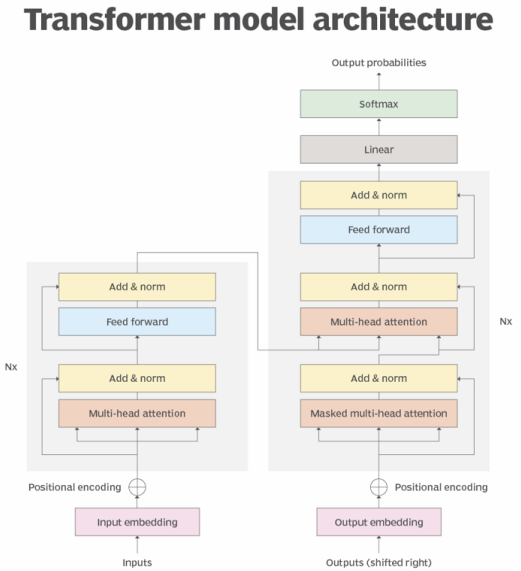
Model Creativity
Measures the diversity and novelty of outputs. Evaluated through user studies or variety indices.

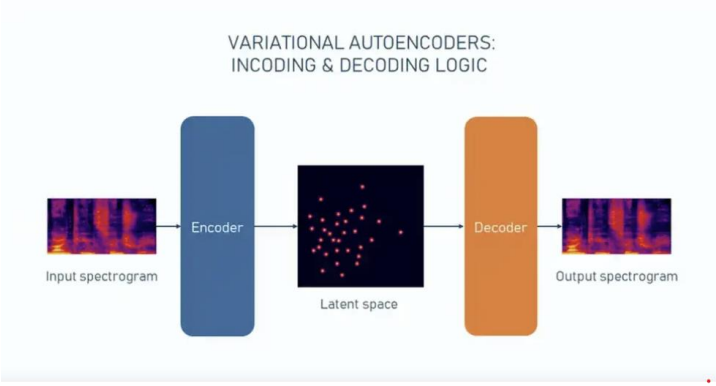
Resource Utilization
The computational resources required during generation, including CPU, GPU, and memory.

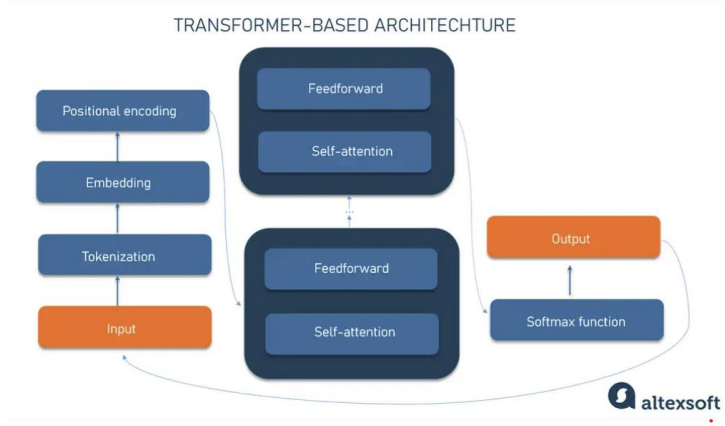
Cost Per Generation
Analyzes the cost-effectiveness of generating each instance, crucial for large-scale deployment.

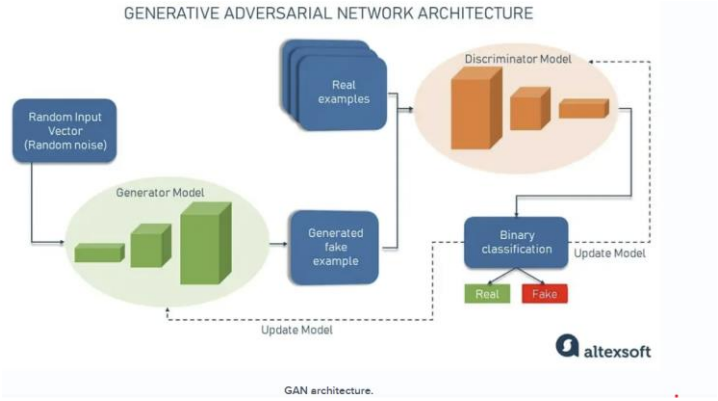

Types of Generative AI Models
01
Generative Adversarial Networks (GANs)
02
Variational Autoencoders (VAEs)
03
Transformer Models
04
Diffusion Models


Input text (77 tokens)
Token embeddings (77 X 768)
Image tensor (4X64X64)
Generated Image
Painting of New York City by Van Gogh
Text Encoder (CLIPText)
Image Information Creator (UNet + Scheduler)
Image Decoder (Auto encoder-decoder)
Stable Diffusion Model


Transformer model architecture
Output probabilities
Softmax
Linear
Add & norm
Feed forward
Add & norm
Multi-head attention
Add & norm
Masked multi-head attention
Nx
Add & norm
Feed forward
Add & norm
Multi-head attention
Nx
Positional encoding
Positional encoding
Input embedding
Output embedding
Inputs
Outputs (shifted right)


VARIATIONAL AUTOENCODERS:
INCODING & DECODING LOGIC
Input spectrogram
Encoder
Latent space
Decoder
Output spectrogram


TRANSFORMER-BASED ARCHITECHTURE
Positional encoding
Embedding
Tokenization
Input
Feedforward
Self-attention
Feedforward
Self-attention
Output
Softmax function
altexsoft

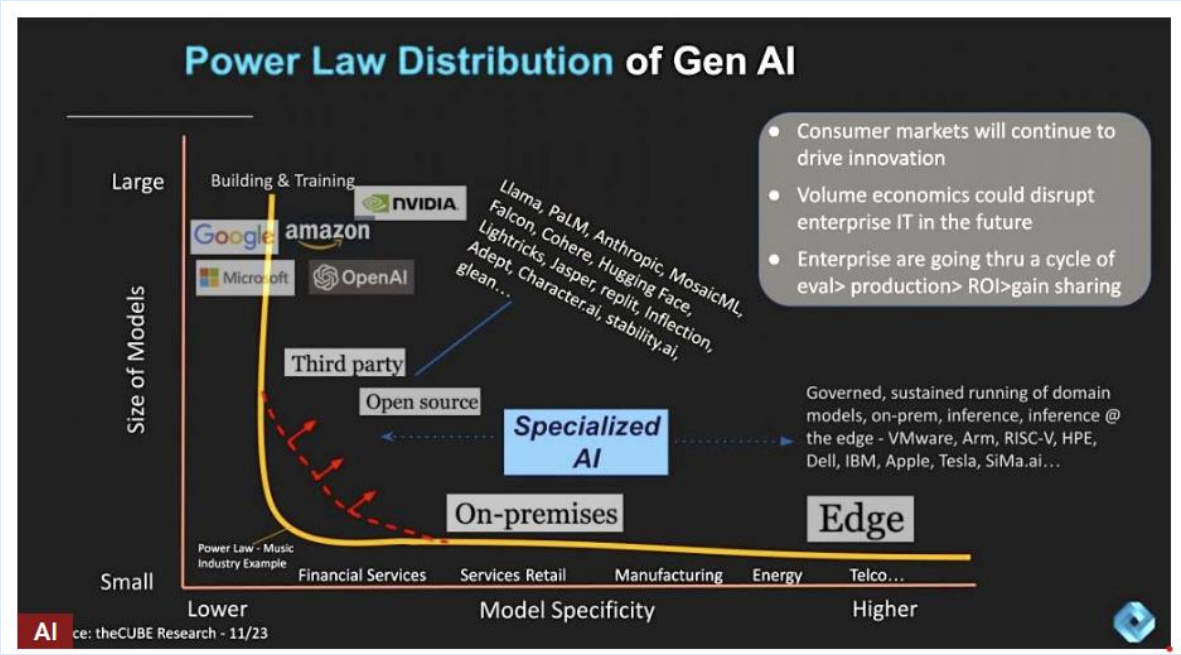

GENERATIVE ADVERSARIAL NETWORK ARCHITECTURE
Random Input Vector (Random noise)
Generator Model
Generated fake example
Real examples
Discriminator Model
Binary classification
Update Model
Real
Fake
Update Model
altexsoft
GAN architecture.

GEN AI MODELS AND ARCHITECTURES

Icon	Architecture	Description	Notable Models/Variants	Use Cases
	Transformer Architecture	Designed for sequence-to-sequence tasks; backbone of state-of-the-art models in NLP and beyond.	BERT, GPT-2, GPT-3, T5, BART	Language translation, text summarization
	Generative Adversarial Networks (GANs)	Consist of two neural networks (generator and discriminator) that compete, enabling realistic data generation.	StyleGAN, CycleGAN, Pix2Pix, cGAN	Image generation, video synthesis
	Variational Autoencoders (VAEs)	Type of autoencoder that learns data representation in latent space and enables data generation by sampling from this space.	-	Image generation, anomaly detection
	Diffusion Models	Generate data by gradually transforming a simple distribution into complex data distributions through diffusion steps.	DALL-E 2, Stable Diffusion	Image generation, denoising
	Recurrent Neural Networks (RNNs)	Used for generating sequences, especially in time-series forecasting and text generation, although somewhat overshadowed by Transformers.	LSTM, GRU	Speech recognition, text generation
	Multi-Modal Architectures	Process and generate data across multiple modalities (e.g., text, images, audio) simultaneously.	CLIP, DALL-E	Content generation, multi-modal learning
	Reinforcement Learning from Human Feedback (RLHF)	Involves training models based on human feedback, enhancing the ability to generate desired outputs.	-	Dialogue systems, content generation

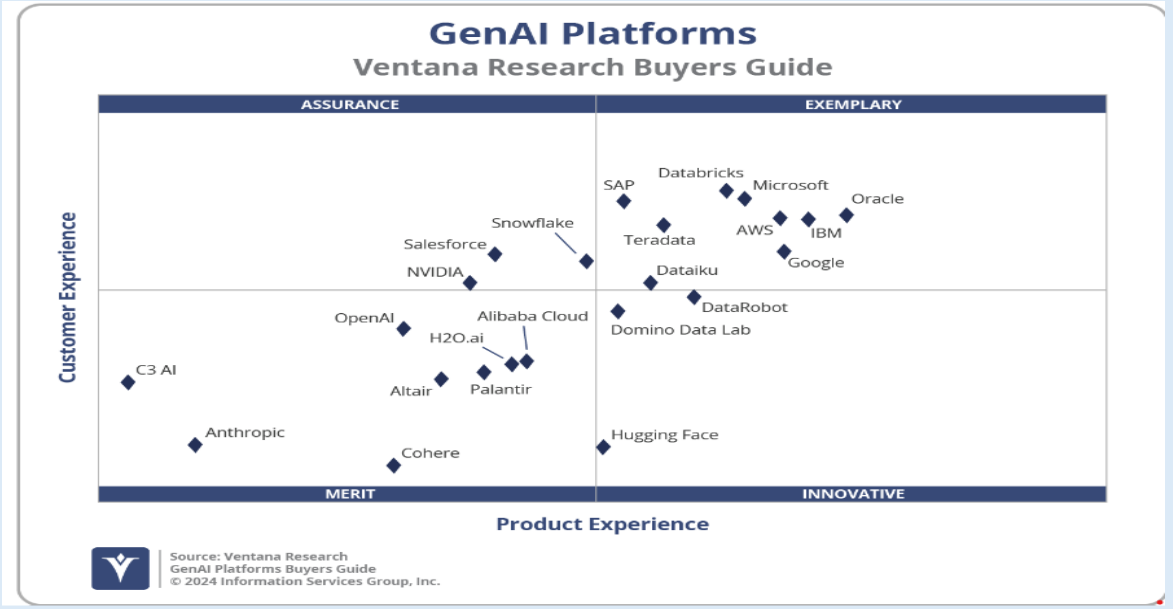
Model	Key Tasks	Frameworks/Libraries	Platforms
Transformers	NLP: text classification, language modeling, machine translation	Hugging Face Transformers, PyTorch, TensorFlow	Hugging Face Platform
GPT	Natural language generation, text summarization, dialogue systems	OpenAI's GPT models, Hugging Face Transformers	Hugging Face Platform, Langchain
BERT	NLP: question answering, named entity recognition, sentiment analysis	Hugging Face Transformers, PyTorch, TensorFlow	Hugging Face Platform
GANs	Image generation, text generation, data augmentation	PyTorch, TensorFlow, Keras	-
ResNet	Computer vision: image classification, object detection	PyTorch, TensorFlow, Keras	-
YOLO	Real-time object detection	PyTorch, TensorFlow, Darknet	-
RNNs	Sequence modeling, speech recognition	PyTorch, TensorFlow, Keras	-
VAEs	Generative modeling, dimensionality reduction	PyTorch, TensorFlow, Keras	-
U-Net	Image segmentation, particularly in medical imaging	PyTorch, TensorFlow, Keras	-
CLIP	Cross-modal learning, image-text retrieval	OpenAI's CLIP, Hugging Face Transformers	Hugging Face Platform
DALL-E / Stable Diffusion	Text-to-image generation	OpenAI's DALL-E, Hugging Face Stable Diffusion	Hugging Face Platform
RAG	Open-domain question answering	Hugging Face Transformers, PyTorch	Hugging Face Platform
Neural Style Transfer	Artistic image generation, photo filters	PyTorch, TensorFlow, Keras	-
Deepfake Technology	Face swapping, video manipulation	PyTorch, TensorFlow, Keras	-
AGG	Enhanced text generation, multimodal content creation	PyTorch, TensorFlow, Hugging Face Transformers	Hugging Face Platform

GEN AI MODELS



Generative foundation models

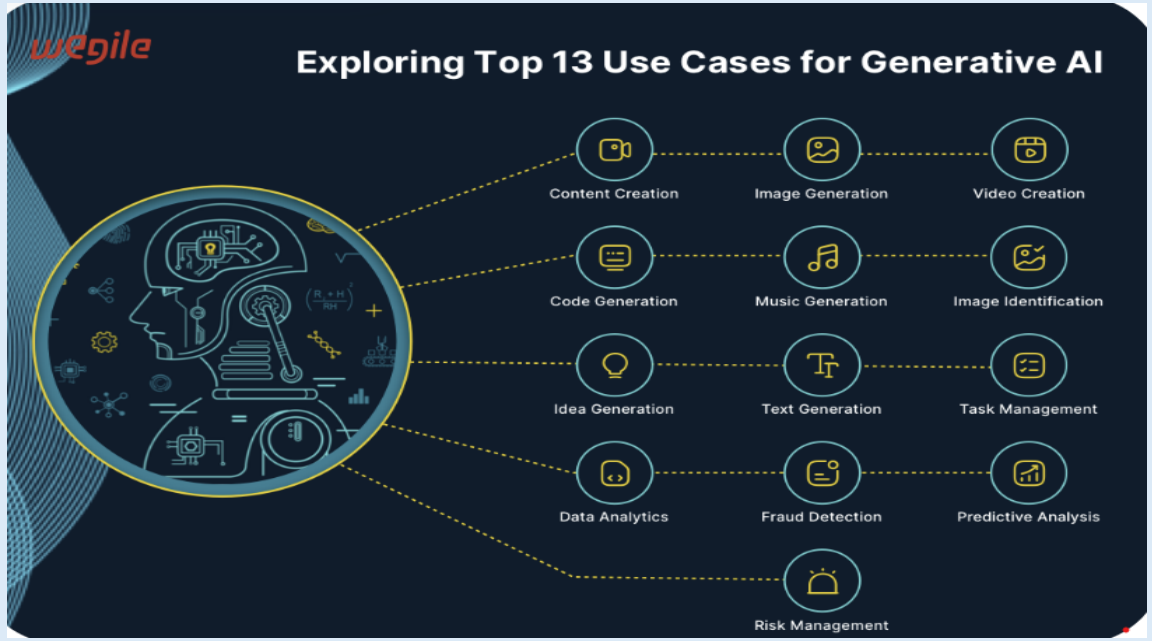
	OpenAI	Meta AI	Google	Other
Text	GPT-4	LLaMA	PaLM 2 Bard	Falcon Dolly Alpaca
Image	CLIP DALL·E 2	ImageBind SAM	Imagen	Stable Diffusion Midjourney
Audio	Whisper	Voicebox	MusicLM	

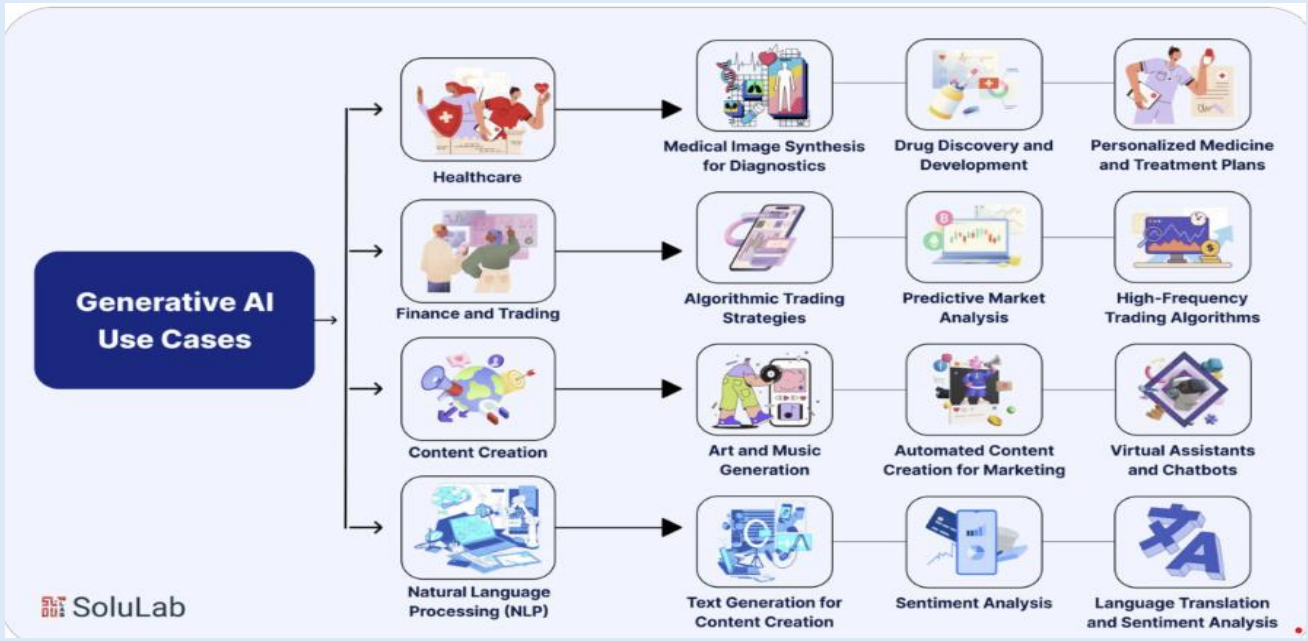


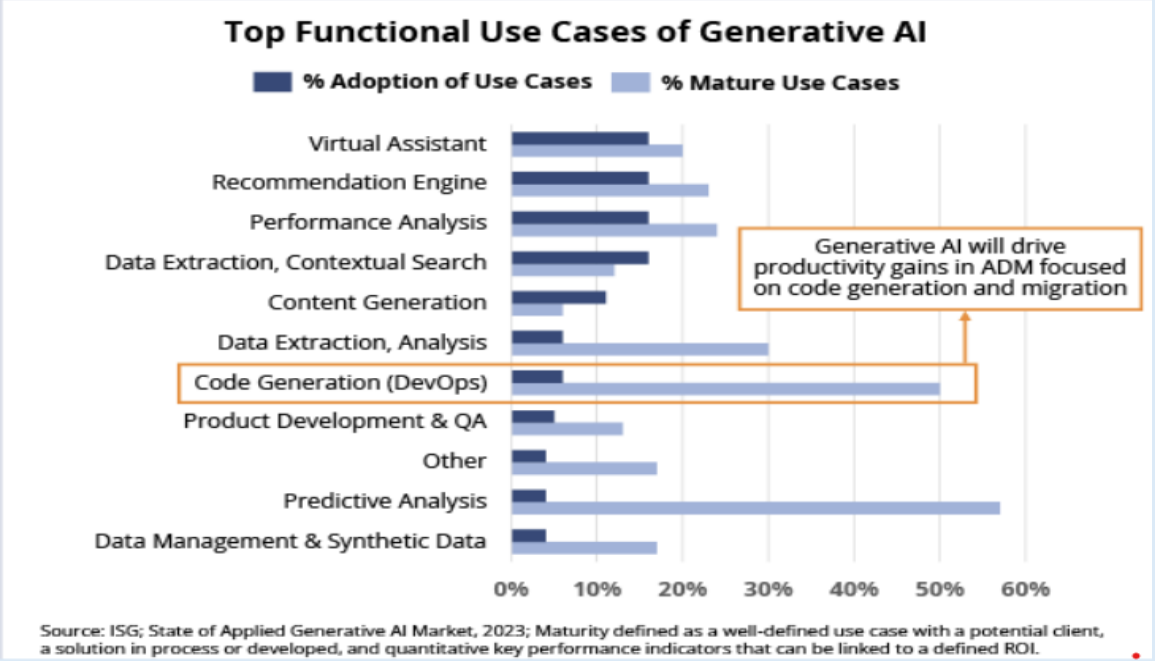
Model	Type	Use Cases
BERT	Free	Question Answering, Sentiment Analysis, NER
RoBERTa		Text Classification, Sentiment Analysis
DistilBERT		Sentiment Analysis, Text Classification
ALBERT		Sentence Classification, Sentiment Analysis
mBERT (Multilingual)		Multilingual Sentiment Analysis, Text Classification

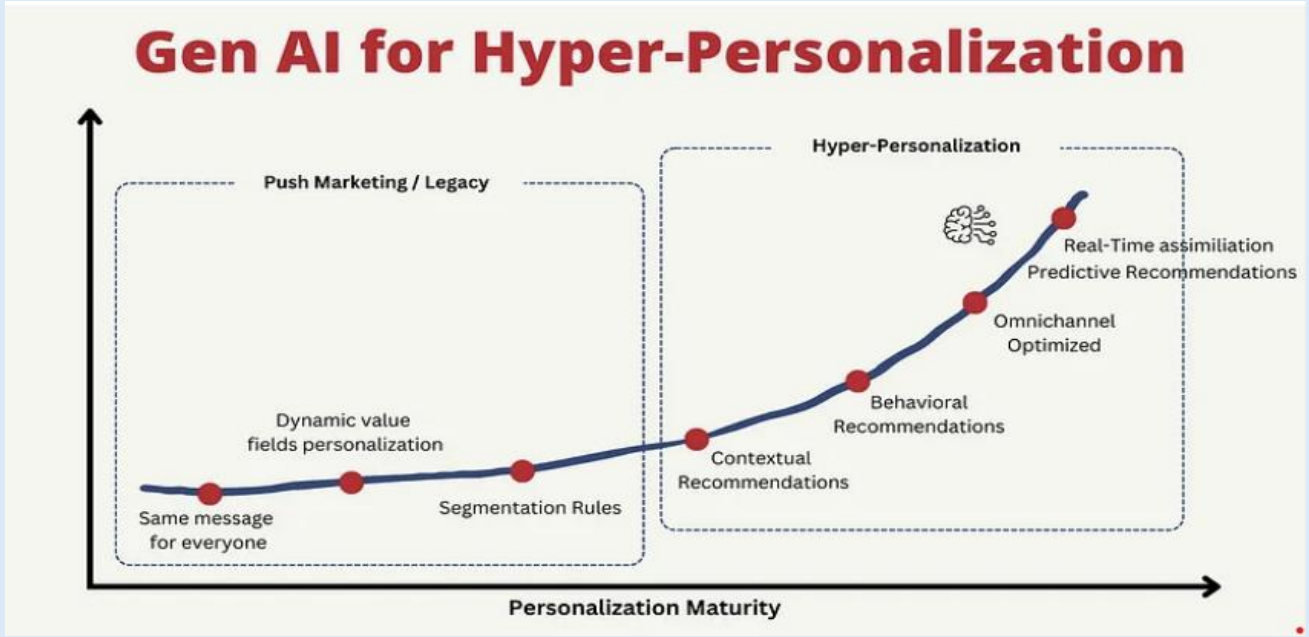
Model	Cost	Use Cases
GPT-4 (OpenAI)	Paid API (Subscription or per-token)	Text Generation, Chatbots, Summarization, QA
Claude 2 (Anthropic)		Text Generation, Summarization, Conversations
LLaMA 2 (Meta)		Language Modeling, Text Understanding, QA
BLOOM		Multilingual Text Generation, Summarization
T5 (Text-to-Text Transfer)		Summarization, Translation, Sentiment Analysis

GEN AI USE CASES





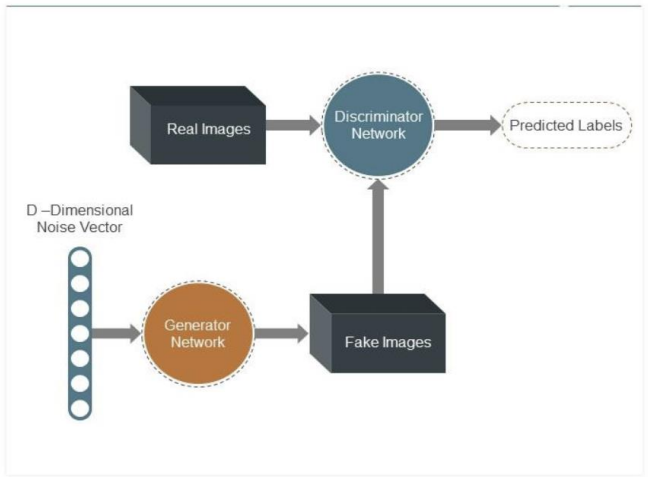




GAN

Architecture of generative adversarial networks (GANs)

This slide provides information regarding the framework of generative adversarial networks (GANs) which comprises of generator and discriminator. The network focuses on generating new data points based on similar distribution.



- Generative modeling is a potential method where
  - Network learns the distribution of input data
  - Focuses to generate new data points based on similar distribution

01

- Generative Adversarial Networks (GANs) comprise of two neural networks
  - Generator "G"
  - Discriminator "D"

02

- Generator (G) network learns distribution of training data

03

- After providing random noise as input, it generates synthetic data to replicate training samples

04

- Discriminator designate label- Real or Fake to data generated by generator (G)

05

Essential steps for effective implementation of GAN models

This slide provides information regarding essential steps for the deployment of GAN models in terms of data preparation and preprocessing, and development of generator & discriminator networks.

Step 1

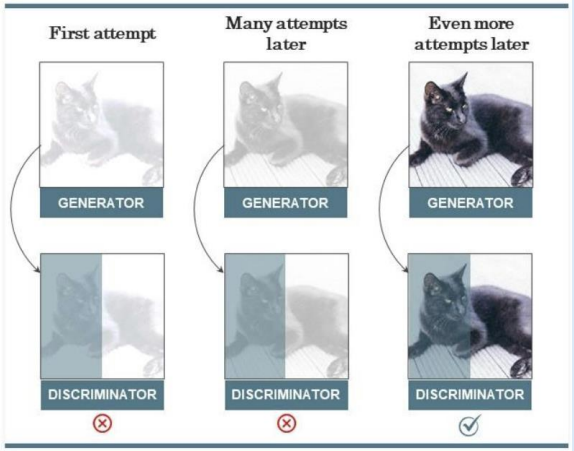
Data Preparation and Preprocessing

- Preparation of training data through cleaning, normalizing
- Suitable formatting of data for GAN model training

Step 2

Development of Generator and Discriminator Networks

- Develop architecture of generator and discriminator networks
- Factors to consider such as layer sizes, activation functions, normalization approaches



Essential steps for effective implementation of GAN models cont.

This slide provides information regarding essential steps for the deployment of GAN models in terms of data GAN models training along with evaluation and monitoring.

Step 3

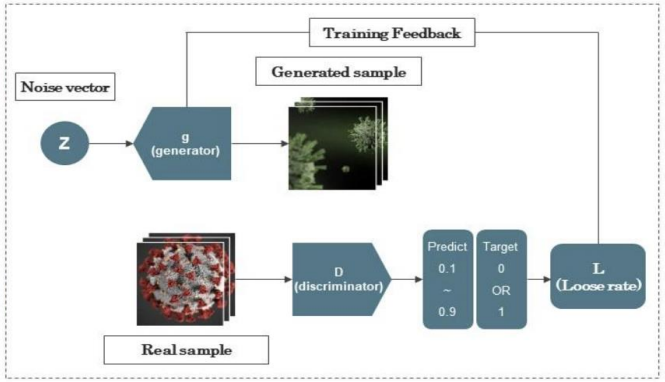
GAN models training

- Iterative GAN model training
- Optimize the generator and discriminator simultaneously
- Techniques include mini-batch discrimination, feature matching, label smoothing, etc.

Step 4

Evaluation and monitoring

- Assess performance of GAN model through metrices such as
  - Inception score
  - Frechet Inception Distance (FID)



Popular GAN frameworks: PyTorch enabled Torch-GAN

This slide provides information regarding popular GAN framework as PyTorch enabled Torch-GAN suitable for building short and easy-to-manage codes while development of GANs, effective performance visualization, extensibility or flexibility of framework.

01 TorchGAN is PyTorch-based framework

02 Suitable for writing short and easy-to-manage codes for building GANs

03 Package comprises of several generative adversarial networks along with utilities essential for deployment

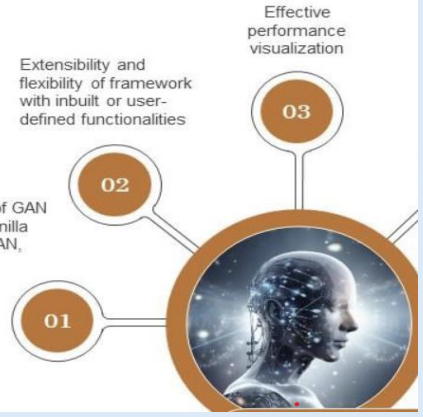
04 Torch GAN mimics GANs design through simple API that enables customization of components when required

05 Facilitates interaction among GAN components through highly flexible trainer that adopts user-defined model and losses

Effective performance visualization

Extensibility and flexibility of framework with inbuilt or user-defined functionalities

Support to wide range of GAN architecture such as Vanilla GAN, DCGAN, CycleGAN, conditional GAN, etc.



VAEs

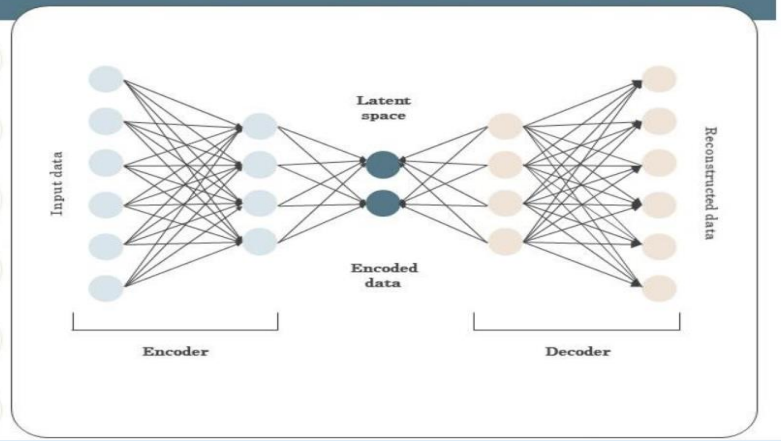
Variational autoencoders (VAEs) as essential part of generative AI model

This slide provides information regarding variational autoencoders as a kind of generative model that builds upon conventional autoencoders. They are potential tools for learning compact and relevant latent representations of data while generating new data samples.

- Variational autoencoder (VAE) is a kind of neural network that learns to regenerate its input and maps data to latent space
- Such autoencoders have regularized training to limit overfitting and enable latent space have good properties to enable generative process

Architecture

- Architecture comprises of encoder and decoder which is trained to minimize reconstruction error among encoded-decoded data along with initial data
- Input encoding is done as distribution instead of single point over latent space for introducing regularization
- Regularization enables VAEs to generate new data samples that seamlessly interpolated among training data points
- Enable VAEs as a potential tools for building new data samples similar to training data
- Regularization also enables VAE to limit the decoder network from reconstructing the input data perfectly
- Decoder network is forced to learn wider general data representation, that helps in enhancing competency of VAE to build new data samples



Pros of VAEs

- Competent in generating new data samples by sampling from learned latent distribution and sample decoding
- VAEs helps in reducing overfitting making them robust to handle complex datasets

Training process of Variational Autoencoders (VAEs)

This slide provides information regarding training process of variational autoencoders which are suitable in generating new samples by learning from training dataset. The model learn from latent representation of data and sampling from latent space to create new samples.

VAEs are suitable to utilized as generative model, and are competent to generate new samples in context with training dataset

It is attained by learning a latent representation of data and sampling from the latent space to develop new samples

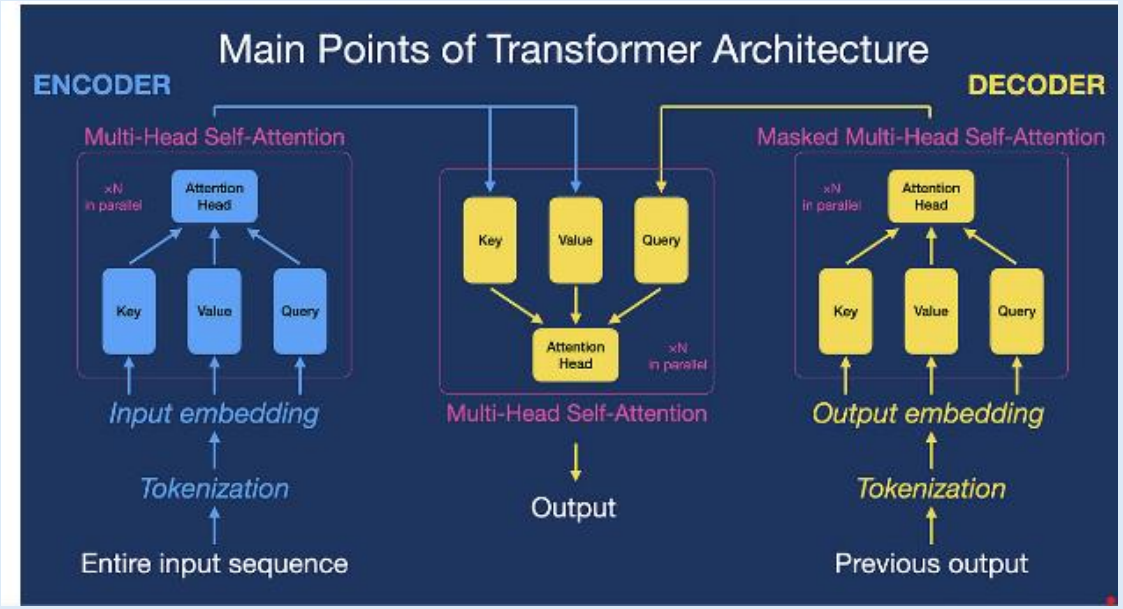
Process of generating new samples through VAE by training model

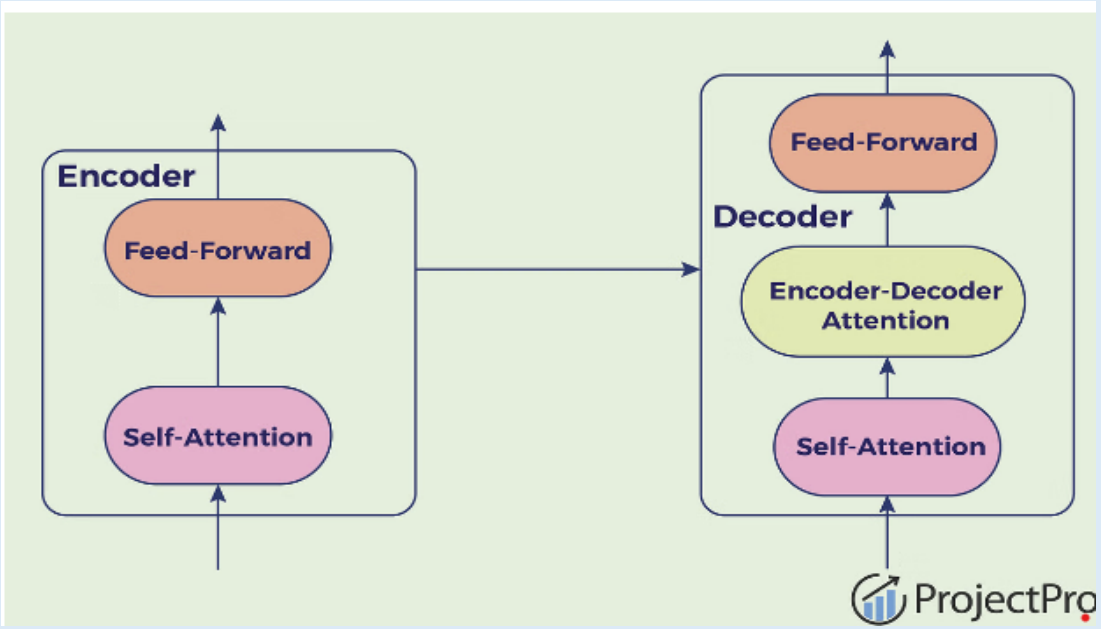
- Step 1: Encode a given input sample into latent space
- Step 2: Sample a new latent point from distribution defined by encoded sample
- Step 3: Latent point is passed through decoder network to build output sample
- Step 4: Reconstruction error is backpropagated across network as it learns to map latent points to data distribution on consistent basis
- Step 5: Interpolating among latenet points to generate new samples for smooth variations of input data

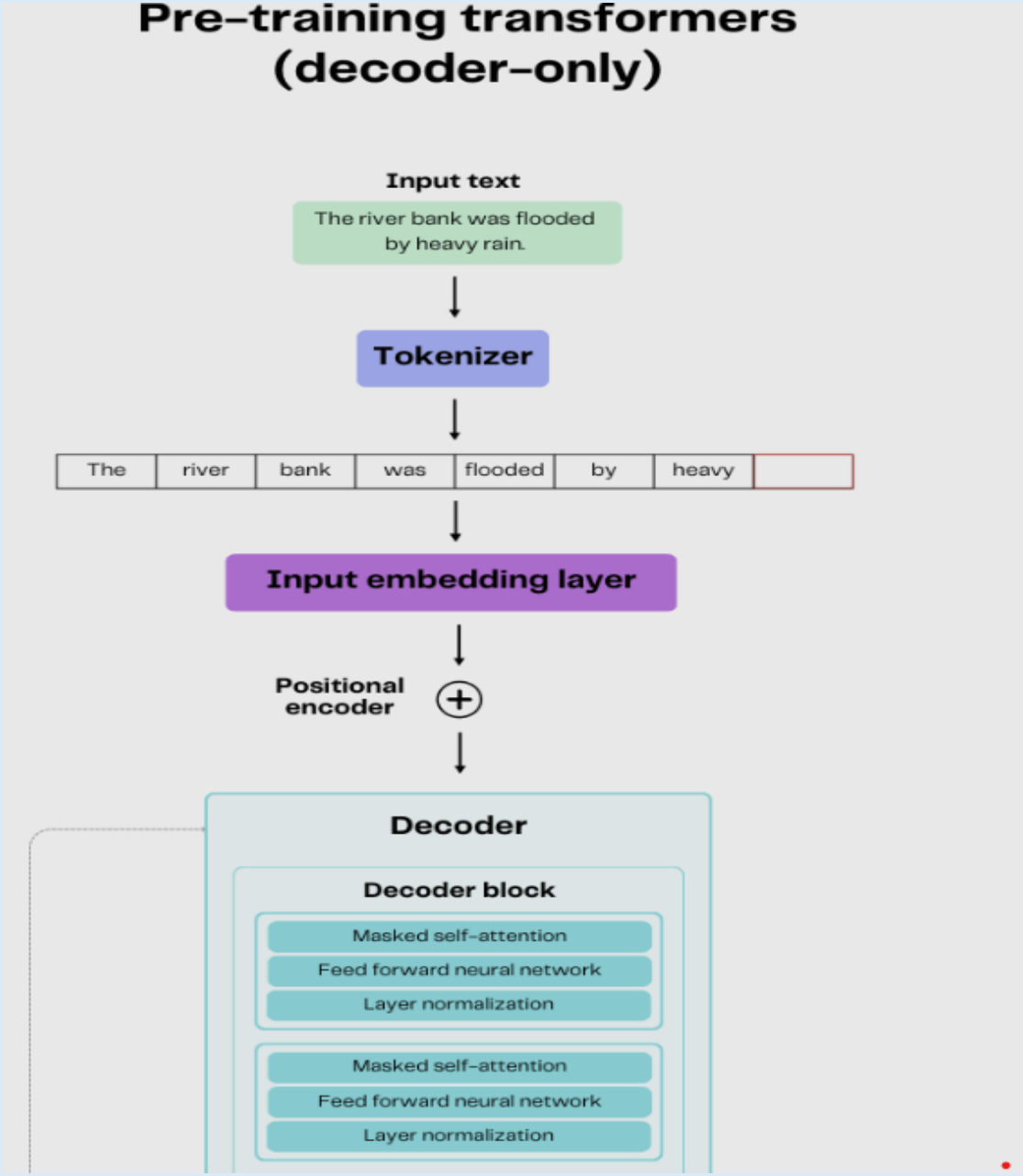
Cons of VAEs

- VAEs might generate blurry reconstructions
- Complex training process

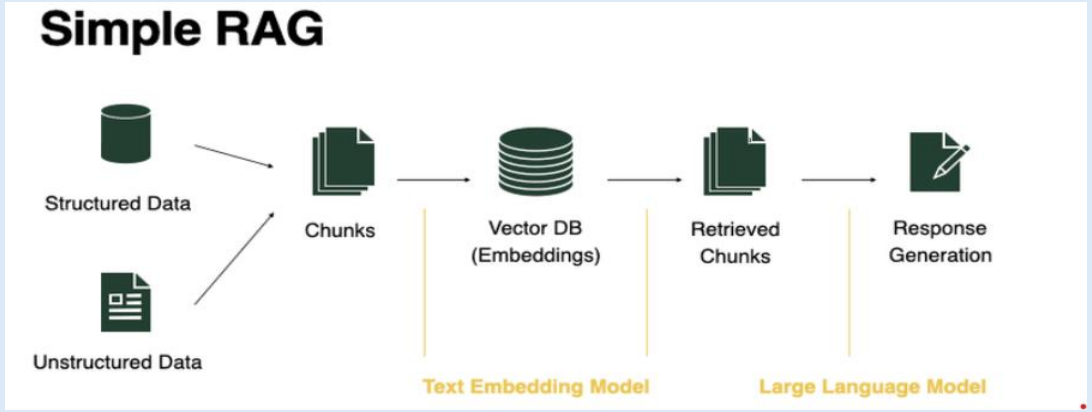
TRANSFORMER

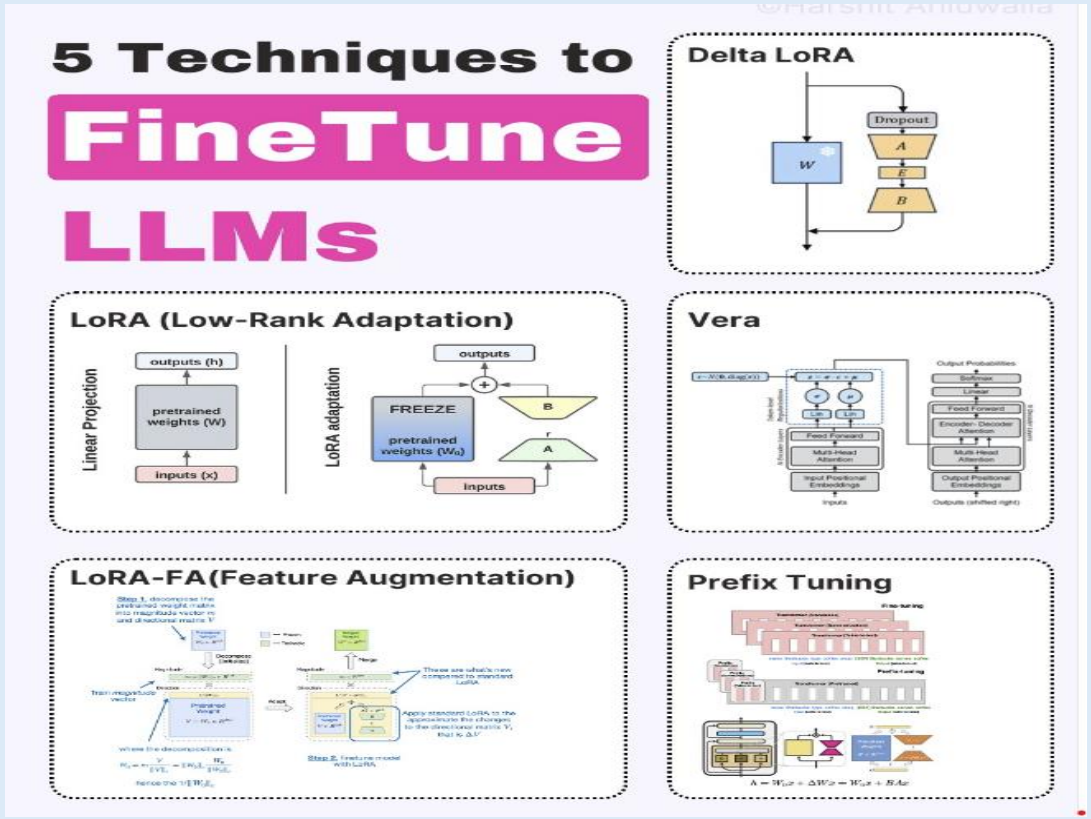


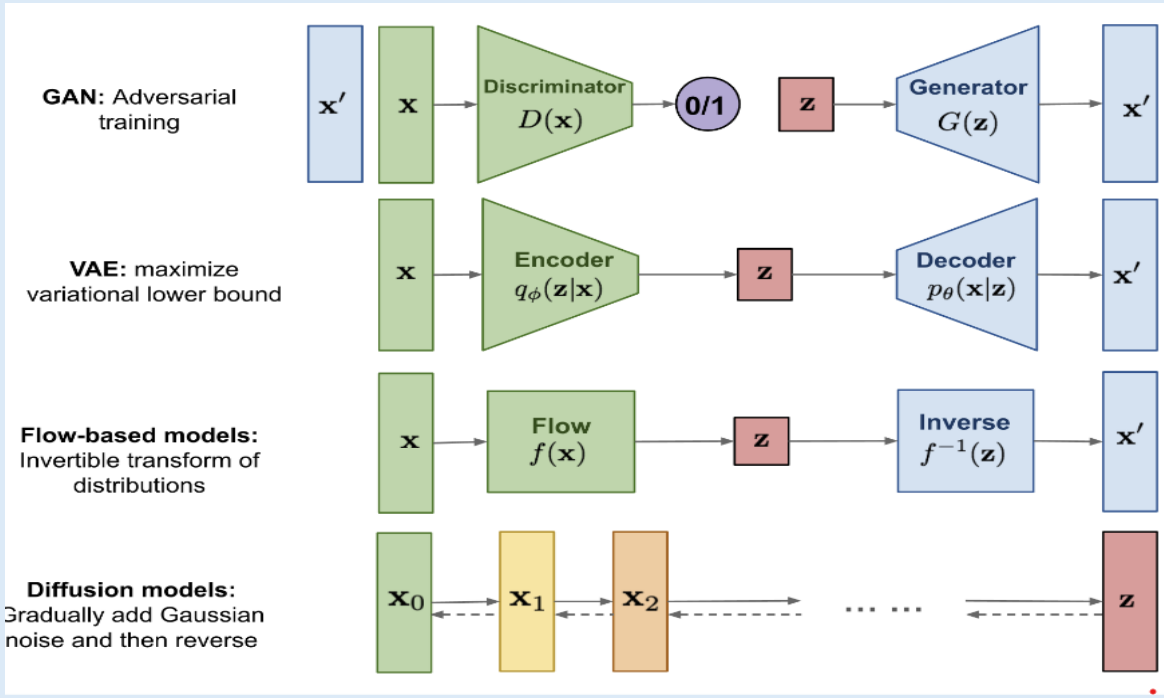


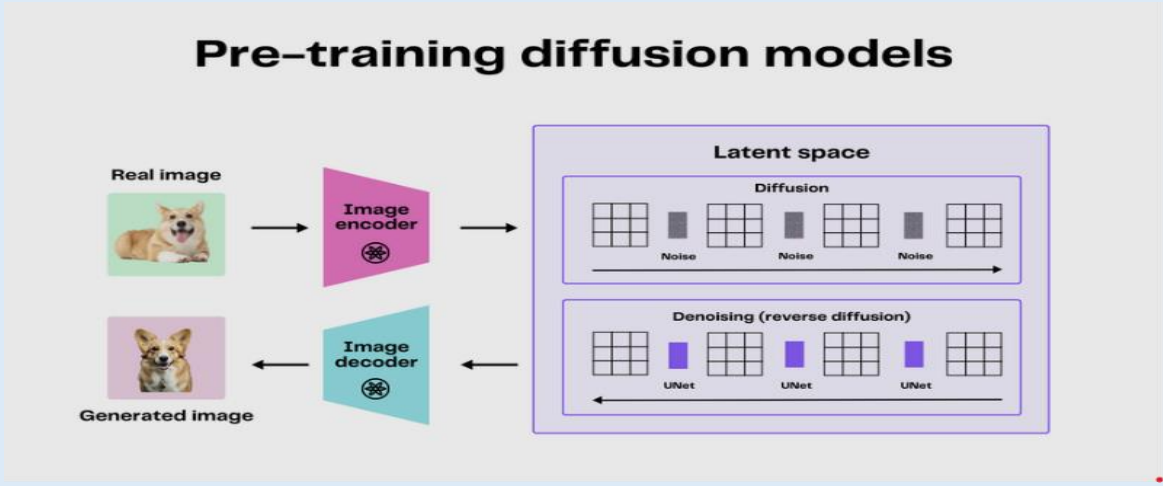


RAG, Diffusion model and Model Fine Tuning









LLM

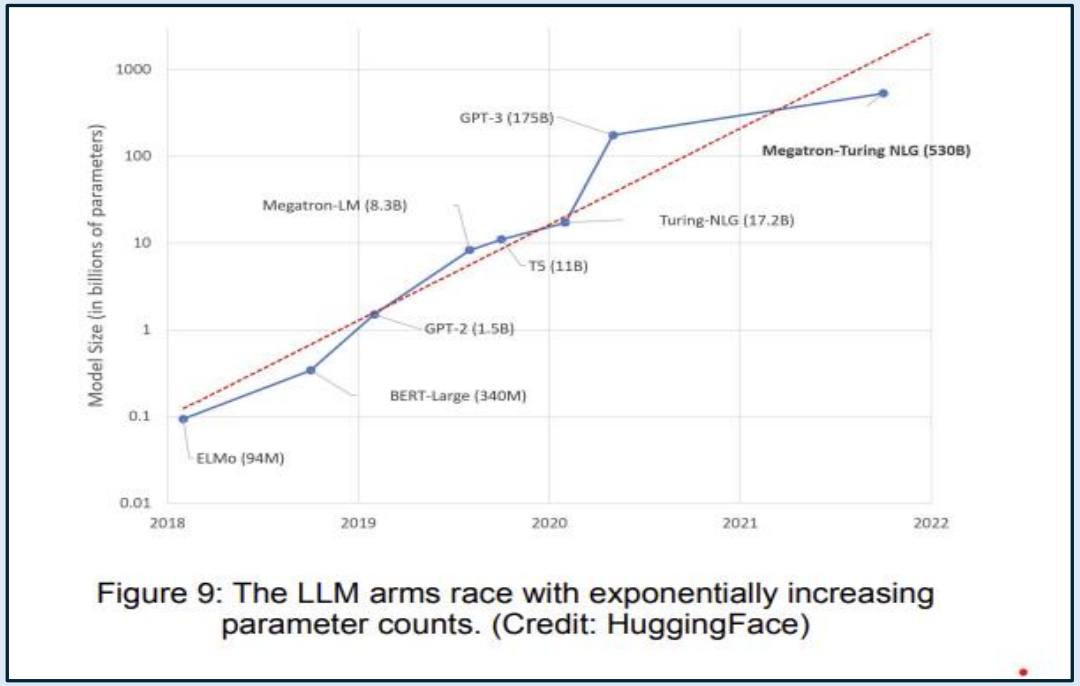
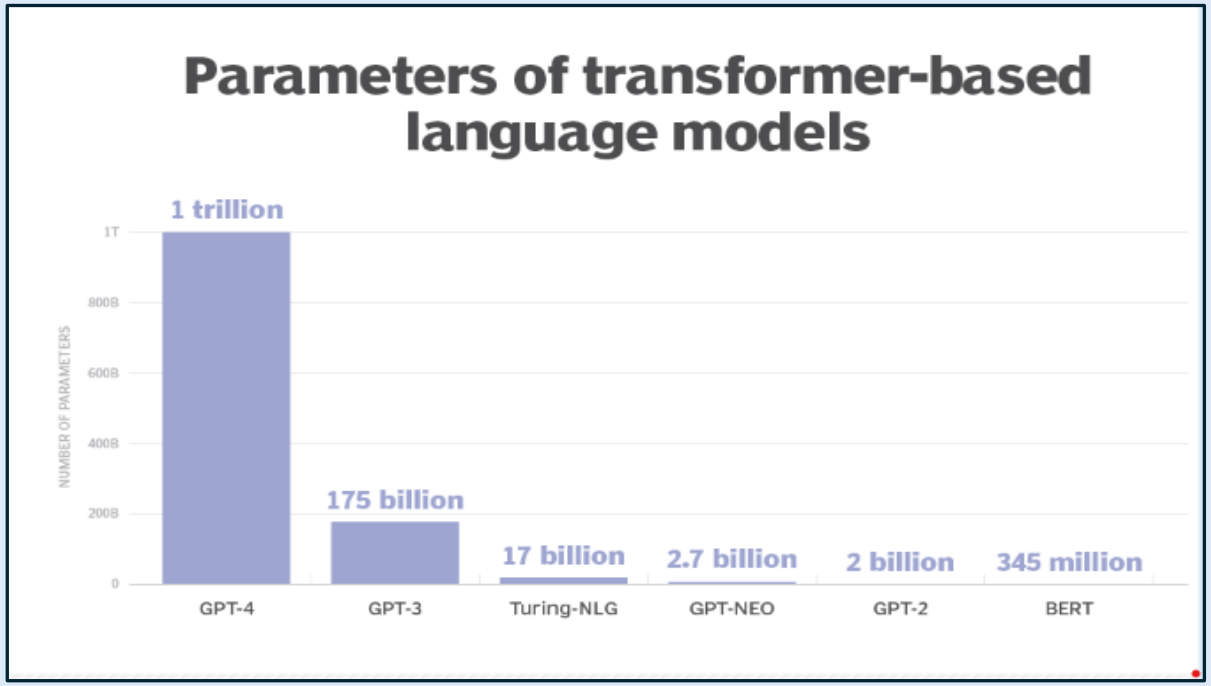
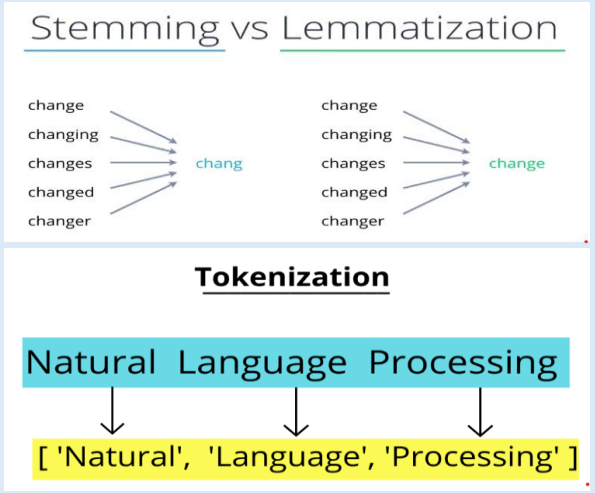
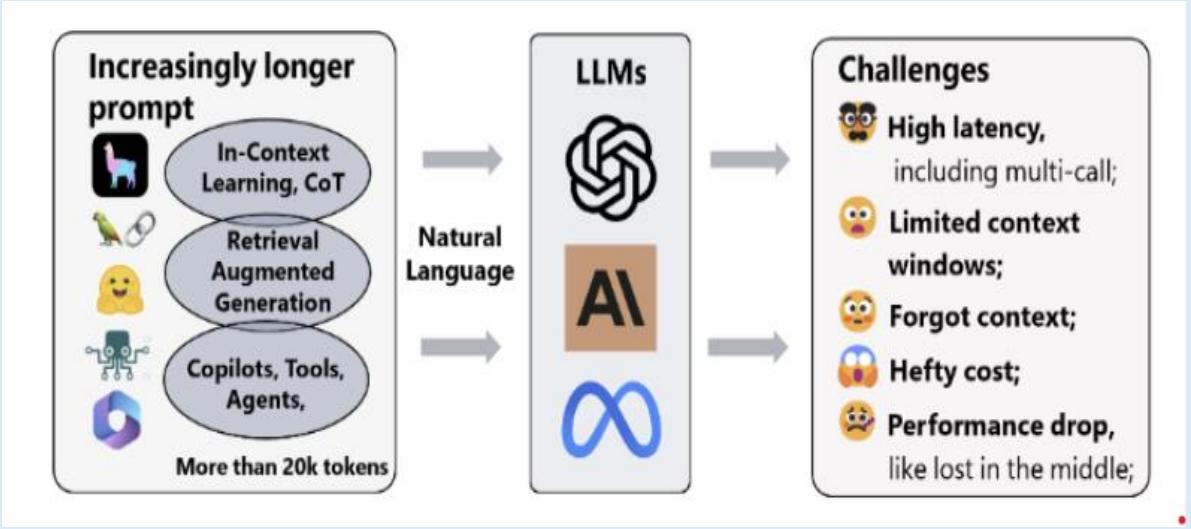


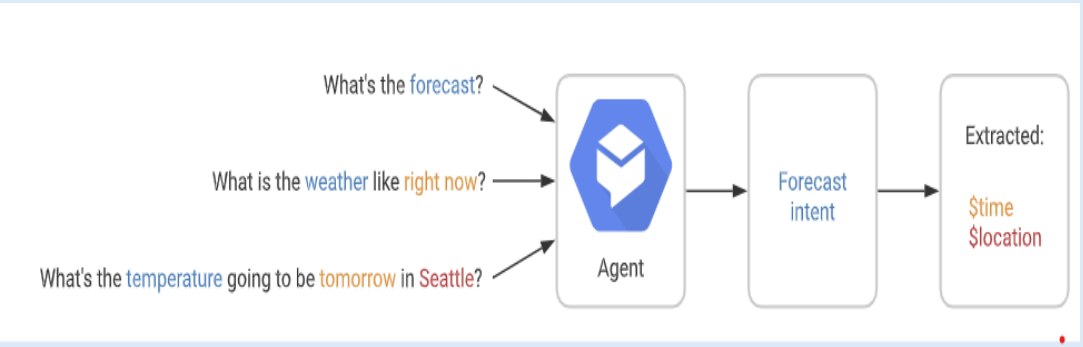
Figure 9: The LLM arms race with exponentially increasing parameter counts. (Credit: HuggingFace)

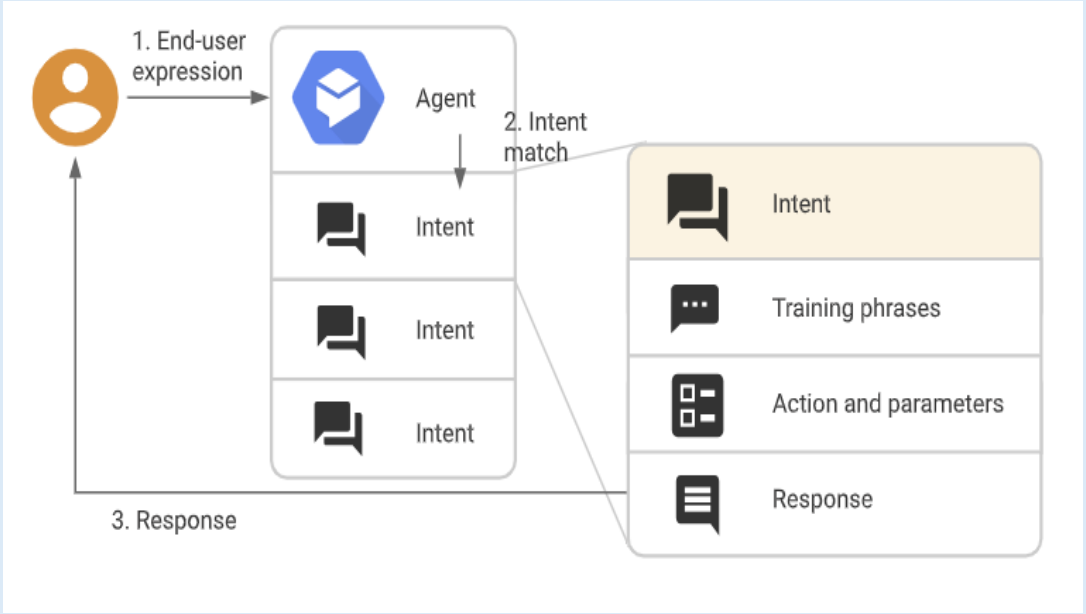




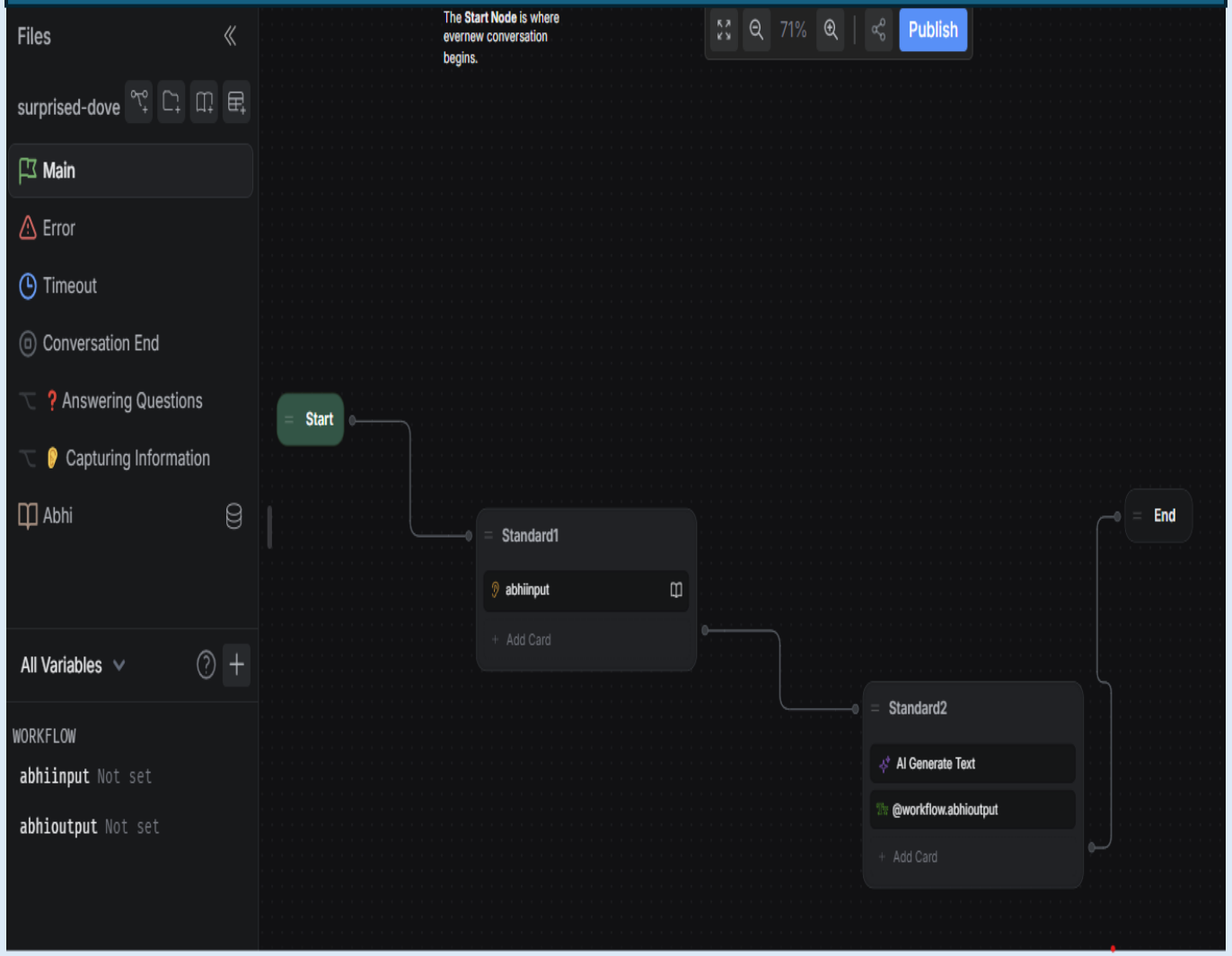


DIALOGFLOW/ BOTPRESS





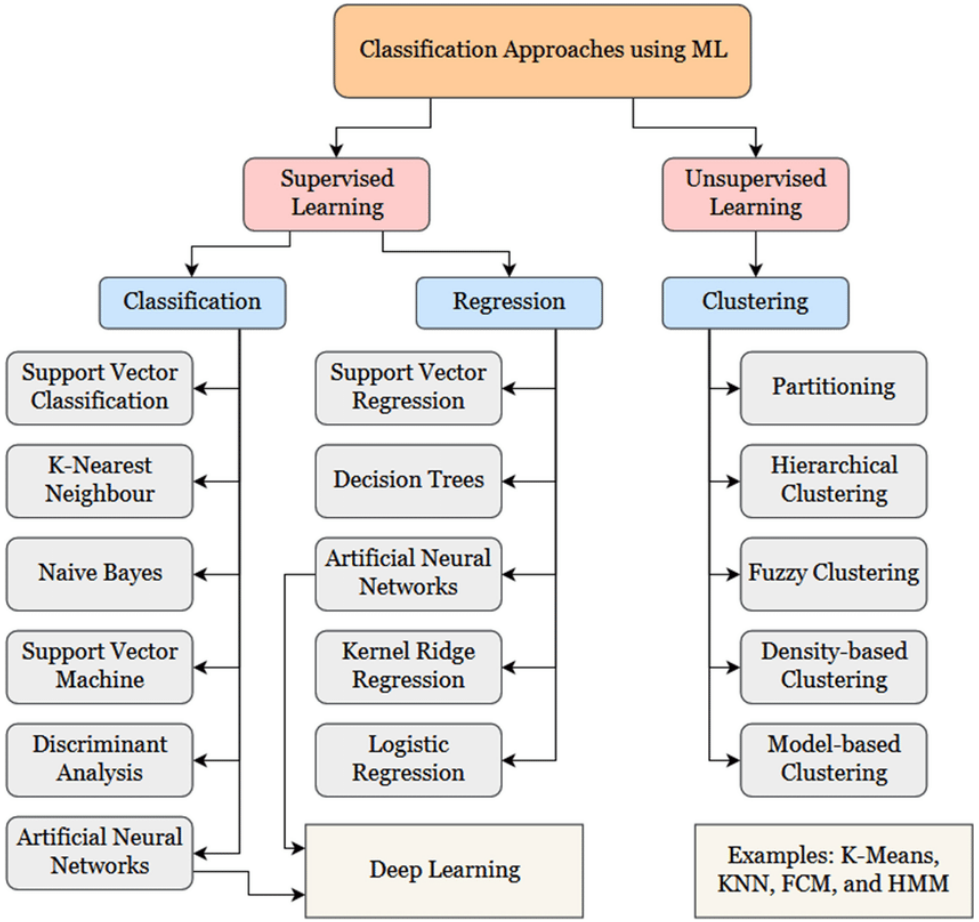
MY BOTPRESS WORK

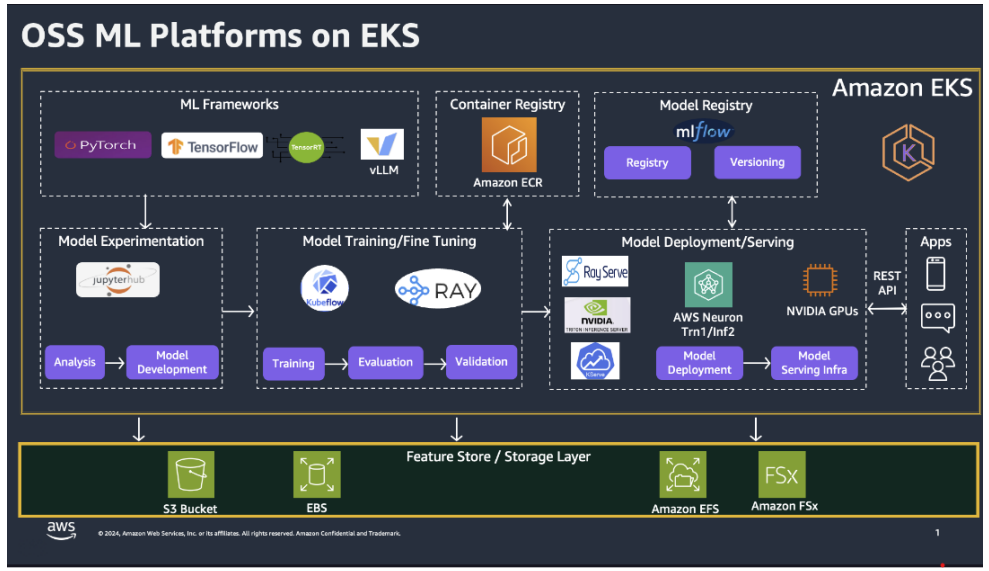


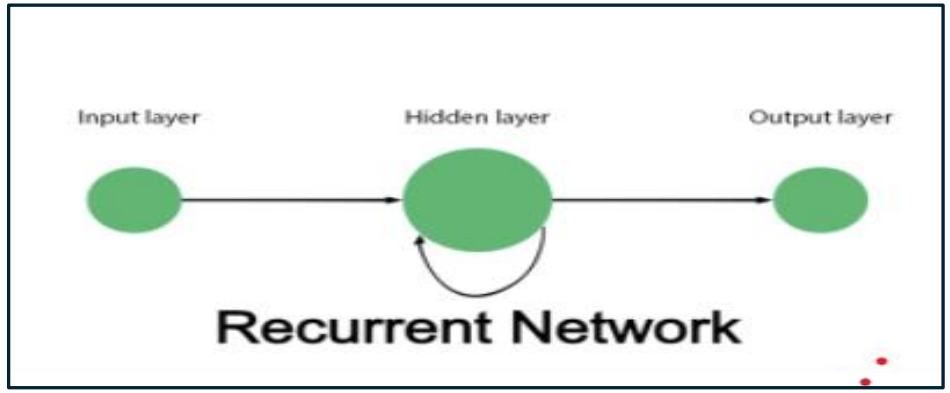
THANKYOU

Github Repository link --https://github.com/Abhishekrai129

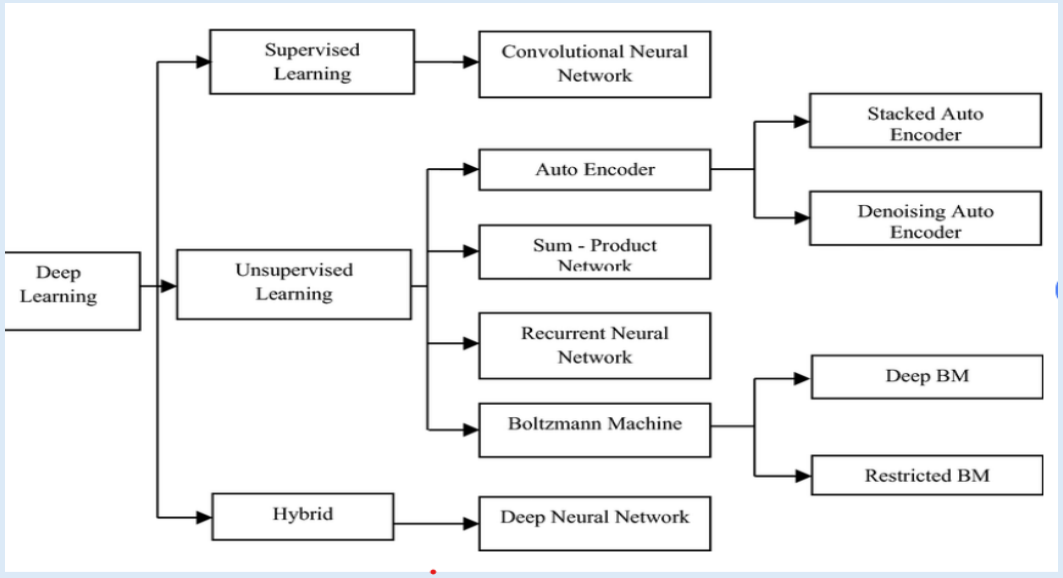
APPENDIX

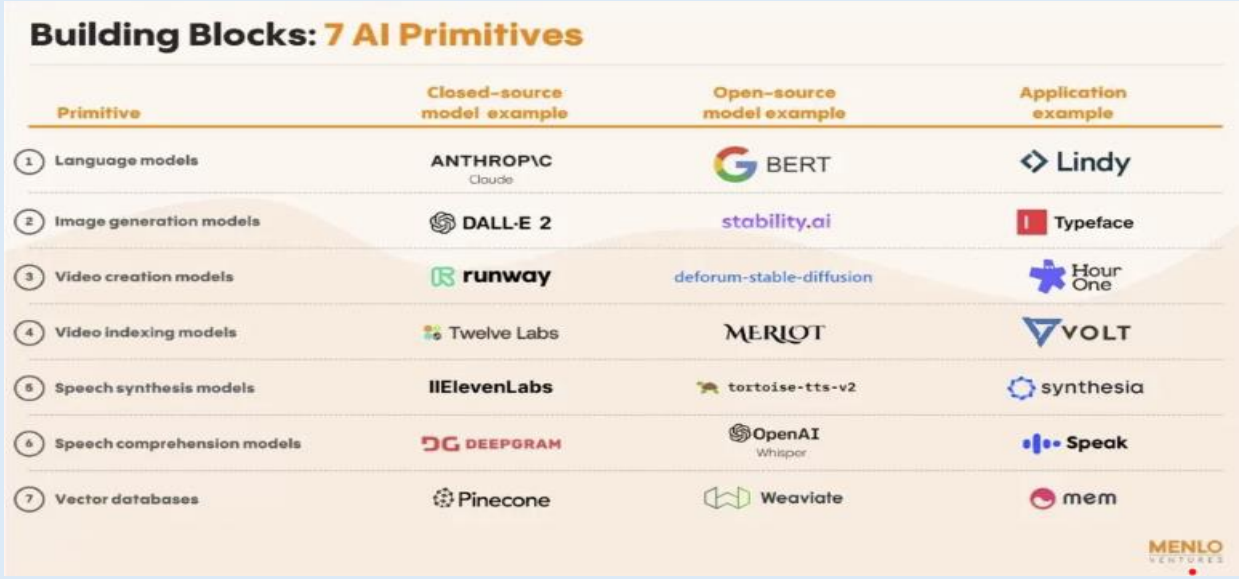

Classification Approaches using ML
Supervised Learning
Unsupervised Learning
Classification
Regression
Clustering
Support Vector Classification
K-Nearest Neighbour
Naive Bayes
Support Vector Machine
Discriminant Analysis
Artificial Neural Networks
Support Vector Regression
Decision Trees
Artificial Neural Networks
Kernel Ridge Regression
Logistic Regression
Deep Learning
Partitioning
Hierarchical Clustering
Fuzzy Clustering
Density-based Clustering
Model-based Clustering
Examples: K-Means, KNN, FCM, and HMM

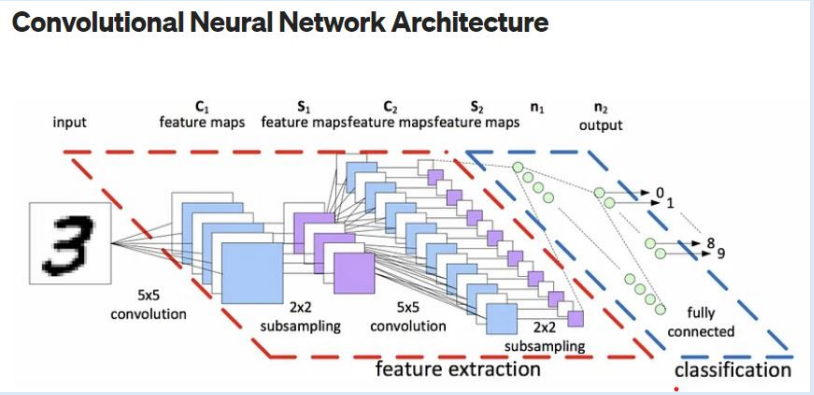

OSS ML Platforms on EKS
Amazon EKS
ML Frameworks
PyTorch
TensorFlow
vLLM
Container Registry
Amazon ECR
Model Registry
mlflow
Registry
Versioning
Model Experimentation
jupyterhub
Analysis
Model Development
Model Training/Fine Tuning
RAY
Training
Evaluation
Validation
Model Deployment/Serving
Ray Serve
NVIDIA
AWS Neuron Trn1/Inf2
NVIDIA GPUs
Model Deployment
Model Serving Infra
REST API
Apps
Feature Store / Storage Layer
S3 Bucket
EBS
Amazon EFS
FSx
Amazon FSx
aws

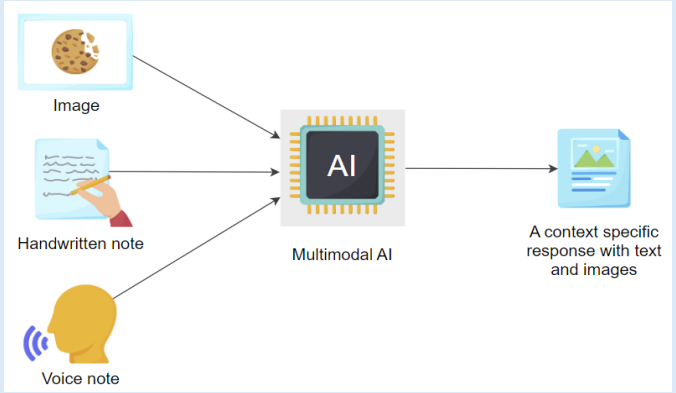

Input layer
Hidden layer
Output layer
Recurrent Network

APPENDIX



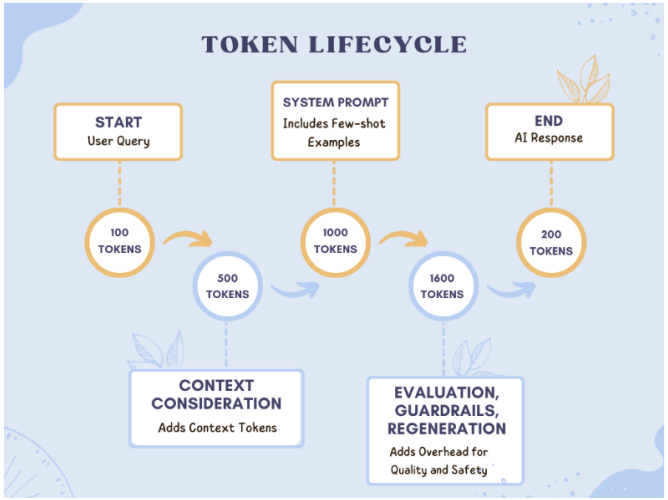






Feature	ANN	RNN	CNN
Structure	Feedforward	Recurrent (loops)	Convolutional layers
Input Type	Structured data	Sequential data	Grid-like data (images)
Memory	Stateless	Maintains state (memory)	Stateless
Use Cases	General tasks	NLP, time-series	Image processing
Training Complexity	Moderate	Higher (vanishing gradients)	Moderate

GEN AI MODELS



Category	Top 5 Models	Frameworks
1. NLP & LLM (Natural Language Processing)	1. GPT-3 / GPT-4 (OpenAI) 2. BERT (Google) 3. LLaMA (Meta) 4. T5 (Google) 5. RoBERTa (Facebook)	1. Transformers (Hugging Face) 2. TensorFlow / Keras 3. PyTorch 4. Fairseq 5. AllenNLP
2. Computer Vision	1. YOLO 2. ResNet 3. EfficientNet 4. Mask R-CNN 5. Vision Transformer (ViT)	1. OpenCV 2. TensorFlow / Keras 3. PyTorch 4. Fastai 5. Detectron2
3. Audio and Speech Processing	1. DeepSpeech 2. Tacotron 2 3. WaveNet 4. VITS 5. HuBERT	1. Kaldi 2. TensorFlow / Keras 3. PyTorch 4. ESPnet 5. Librosa
4. Generative Models (GANs, VAEs, etc.)	1. StyleGAN 2. CycleGAN 3. VAE 4. DALL·E 5. Stable Diffusion	1. TensorFlow / Keras 2. PyTorch 3. Fastai 4. NVIDIA StyleGAN2 5. CLIP
5. Reinforcement Learning & Robotics	1. DQN 2. PPO 3. A3C 4. TD3 5. OpenAI Five	1. OpenAI Gym 2. Ray RLLib 3. Stable Baselines3 4. TF-Agents 5. PyTorch Lightning

Model	Type	Use Cases
BERT	Free	Question Answering, Sentiment Analysis, NER
RoBERTa		Text Classification, Sentiment Analysis
DistilBERT		Sentiment Analysis, Text Classification
ALBERT		Sentence Classification, Sentiment Analysis
mBERT (Multilingual)		Multilingual Sentiment Analysis, Text Classification

Model	Cost	Use Cases
GPT-4 (OpenAI)	Paid API (Subscription or per-token)	Text Generation, Chatbots, Summarization, QA
Claude 2 (Anthropic)		Text Generation, Summarization, Conversations
LLaMA 2 (Meta)		Language Modeling, Text Understanding, QA
BLOOM		Multilingual Text Generation, Summarization
T5 (Text-to-Text Transfer)		Summarization, Translation, Sentiment Analysis

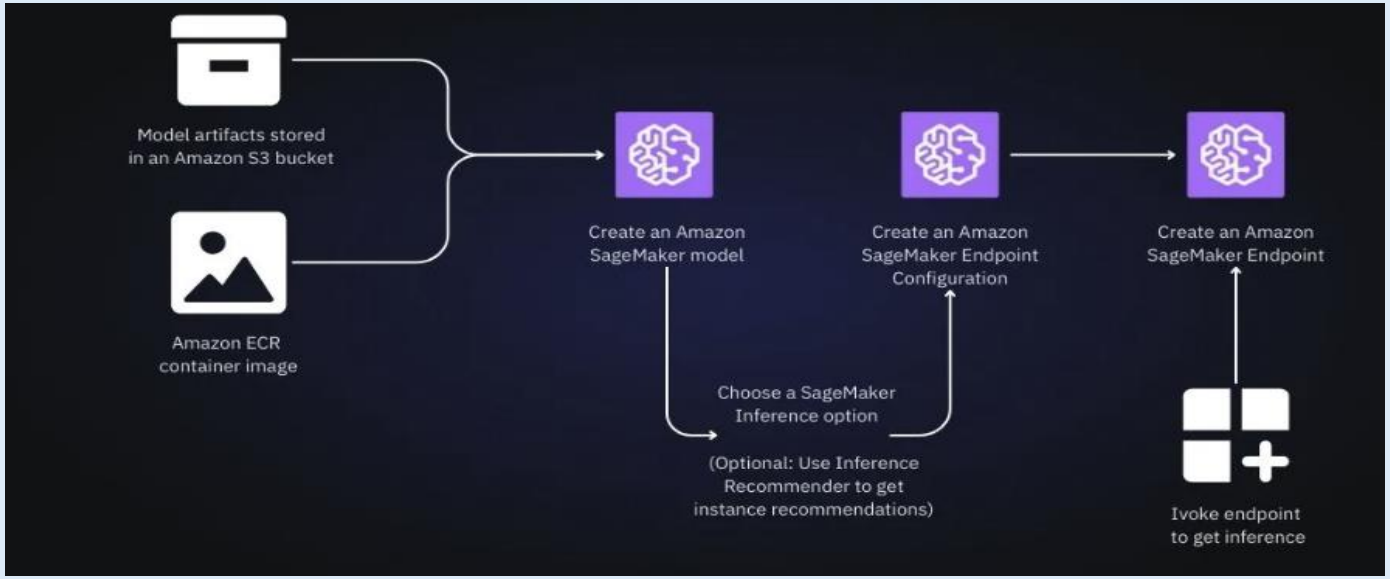
APPENDIX

Company	Model	Type	Price
OpenAI	GPT 4.0 8k context	Input	$0.03/1k tokens
		Output	$0.06/1k tokens
	GPT 4.0 32k context	Input	$0.06/1K tokens
		Output	$0.12/1K tokens
	GPT-3.5 Turbo 4k context	Input	$0.0015/1K tokens
		Output	$0.002/1K tokens
	GPT-3.5 Turbo 16k context	Input	$0.003/1K tokens
		Output	$0.004/1K tokens
Google	PaLM 2 for Text	Input	$0.0005/1K characters
		Output	$0.0005/1K characters
	PaLM 2 for Chat	Input	$0.0005/1K characters
		Output	$0.0005/1K characters

The top datasets are all based on images with a few text-based datasets such as Wikipedia and PubMed.

Table 1 Top 20 datasets mentioned in the open access subset of the GenAI corpus

	Dataset name	Citing documents	Total mentions	Main modality
1	ImageNet	2,741	6,823	image
2	MNIST	2,533	9,292	image
3	CIFAR-10	2,160	7,744	image
4	CelebA	1,705	5,713	image
5	COCO	1,141	3,390	image
6	Wikipedia	662	2,599	text
7	FFHQ dataset	596	1,983	image
8	FASHIONMNIST	520	1,375	image
9	CelebAHQ	474	1,081	image
10	SVHN	398	1,414	image



Strategies for Cost Management in GEN AI Implementation

TOKEN MANAGEMENT STRATEGIES

- **Manage token usage strategically**-Monitor usage patterns to identify high-consumption tasks'
- **Minimize prompt length**--Use concise prompts to reduce token usage.
- **Focus on narrow use cases**--Implement AI for specific tasks (e.g., customer support FAQs).

MODEL SELECTION MATTERS

- **Evaluate ROI of AI models**--Compare costs of GPT-3.5 vs. GPT-4 based on application.
- **Cost-effective models can yield similar performance**--Test GPT-3.5 for tasks like text summarization versus GPT-4 for complex dialogue.

LATENT COSTS AND INFRASTRUCTURE NEEDS

- **Latency costs can escalate**--Assess response times for large document processing.
- **Consider lightweight models or caching techniques**--Use caching for frequently accessed data to reduce load.

AVOIDING AI SPRAWL

- **Implement governance structure**--Centralize contracts and tools to avoid redundancy.
- **Prioritize clear objectives**--Define specific use cases to guide implementation (e.g., marketing automation vs. operational efficiency).

PHASED ROLLOUTS FOR COST MANAGEMENT

- **Small-scale rollouts to gather usage data**--Conduct pilot projects to refine features and gather feedback.
- **Refine models before wider deployment**-Adjust based on user behavior and performance metrics.