



Generative AI

Complete study notes — from AI and machine-learning basics to how large language models work, prompt engineering, text, image, audio and video generation, building chatbots and apps, and using AI responsibly.

10

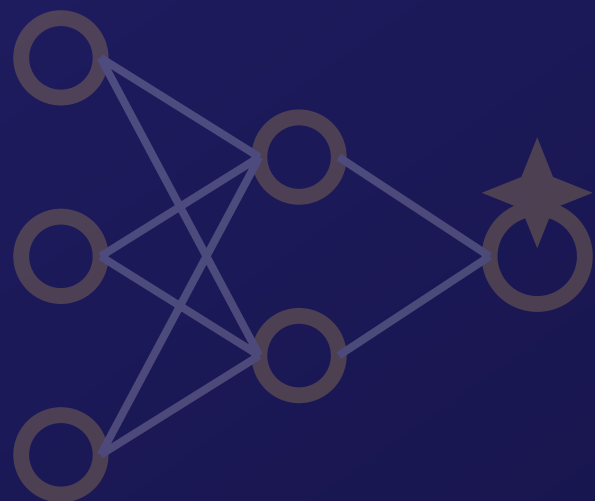
Modules

50+

Diagrams & prompt examples

200+

Questions & MCQs



Contents

01	Introduction to Artificial Intelligence
02	Machine Learning and Deep Learning Basics
03	Introduction to Generative AI
04	Large Language Models (LLMs)
05	Prompt Engineering Techniques
06	AI Text and Content Generation
07	AI Image, Audio, and Video Generation
08	Building AI Chatbots and Applications
09	Responsible AI, Ethics, Privacy, and Limitations
10	Practical Projects and Final Revision
11	Profession Playbooks and Model Question Paper
12	Bonus: Prompt Library, Tools Guide, Lab Workbook and Careers
13	Multiple-Choice Questions, Answers and Glossary

How to use these notes

Keep an AI assistant open while you study and try every prompt. AI tools change very quickly — names, prices and features in this field move month to month — so these notes focus on ideas that stay true, and label product examples as examples.

Blue box — Key idea. The concept to remember.

Yellow box — Prompt pattern. Templates you can reuse.

Green box — Try it. Hands-on activities.

Orange box — Caution. Risks, limits and mistakes.

Worked example. Prompts and outputs explained.

Practice.
Questions at the end of every module.

MODULE 01

Introduction to Artificial Intelligence

What AI is (and isn't), a short history, the main branches and types of AI, how AI "learns" from data, where generative AI fits, and how AI is already used in everyday life in India and worldwide.

1.1 What is artificial intelligence?

Artificial intelligence (AI) is the field of computer science that builds systems able to perform tasks that normally need human intelligence — understanding language, recognising images and speech, making decisions, solving problems, and creating content.

A traditional program follows rules a programmer writes explicitly: "*if the temperature is above 30, switch on the fan*". Most modern AI instead **learns patterns from examples**. Show a system thousands of photos labelled "cat" or "dog", and it learns to tell them apart — without anyone writing rules about whiskers or ears.

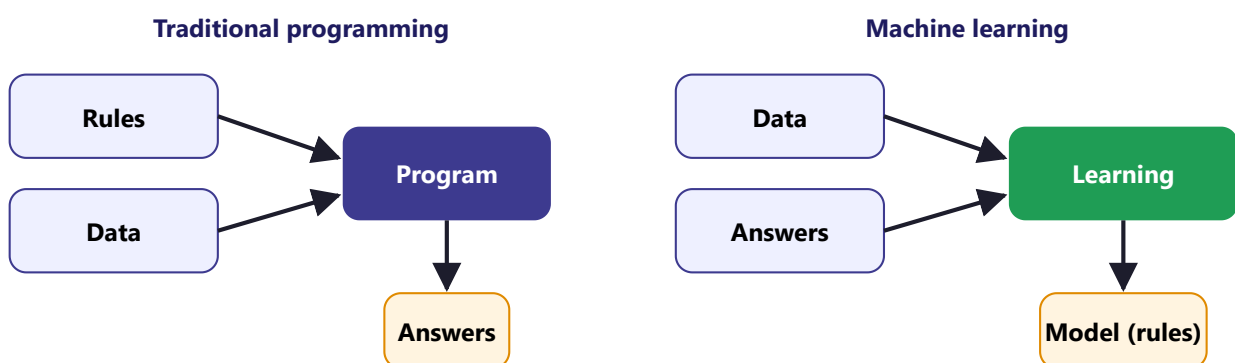


Figure 1.1 — Traditional programs turn rules + data into answers; machine learning turns data + answers (examples) into a model.

Key idea

Modern AI systems are **pattern learners**. They are extremely good at finding statistical patterns in large amounts of data — but they don't "understand" the world the way people do, and they can be confidently wrong.

1.2 A short history of AI

Year	Milestone
1950	Alan Turing asks "Can machines think?" and proposes the Turing Test .

1956	The Dartmouth workshop coins the term "artificial intelligence" (John McCarthy and others).
1958	Frank Rosenblatt's perceptron — an early artificial neuron that learns.
1960s–70s	Early chatbots (ELIZA, 1966), logic programs; optimism, then the first " AI winter " when promises weren't met.
1980s	Expert systems encode specialist rules (medical diagnosis, configuration); backpropagation popularised for training neural networks.
1997	IBM's Deep Blue defeats world chess champion Garry Kasparov.
2000s	Machine learning spreads: spam filters, recommendations, search ranking.
2012	Deep learning breakthrough : a neural network (AlexNet) wins the ImageNet image-recognition challenge by a large margin, using GPUs.
2014	GANs (generative adversarial networks) introduced — realistic image generation begins.
2016	DeepMind's AlphaGo beats a world champion at Go.
2017	The Transformer architecture ("Attention Is All You Need") — the foundation of today's language models.
2018–2020	Large pre-trained language models (BERT, GPT-2, GPT-3) show surprising abilities.
2020–2022	Diffusion models make high-quality text-to-image generation practical; ChatGPT (Nov 2022) brings generative AI to hundreds of millions of people.
2023 onwards	Rapid progress: multimodal models (text, images, audio, video), AI assistants in office tools and phones, open-weight models, AI agents that use tools.

1.3 Types of AI by capability

Type	Meaning	Status
Narrow AI (weak AI)	Performs specific tasks — translation, face unlock, playing chess, writing text	All AI in use today, including chatbots, is narrow — even when it seems general
General AI (AGI)	Human-level ability across almost all intellectual tasks, learning new ones flexibly	A research goal; experts disagree on when or whether it will arrive
Superintelligence	Far beyond human ability in every domain	Hypothetical

1.4 Branches of AI

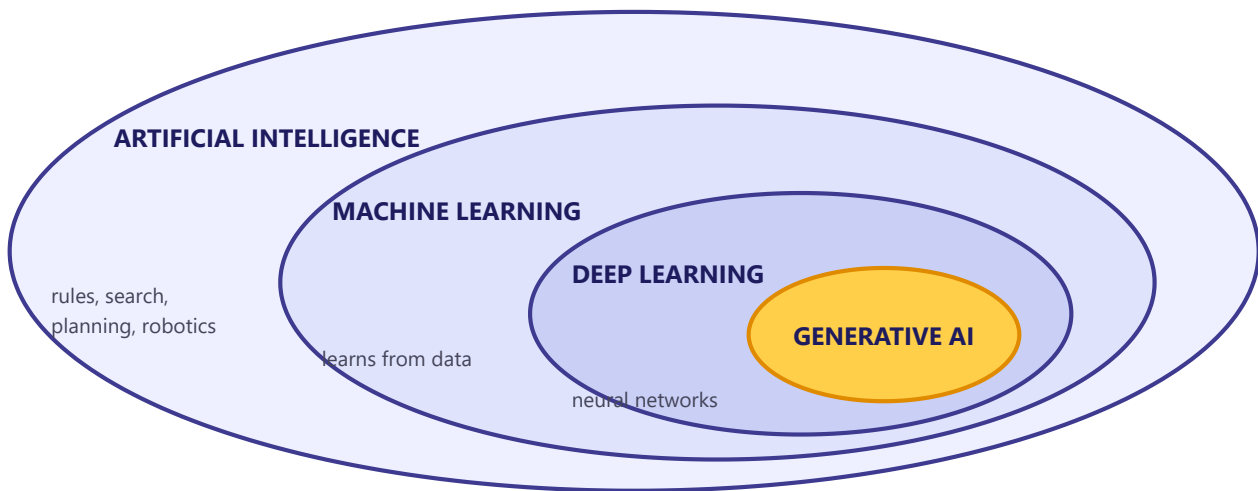


Figure 1.2 — Generative AI is a part of deep learning, which is part of machine learning, which is part of AI.

Branch	What it does	Examples
Machine learning (ML)	Learns patterns from data to make predictions	Loan approval scoring, sales forecasts, spam filters
Deep learning	ML with many-layered neural networks	Face recognition, speech recognition
Natural language processing (NLP)	Understanding and producing human language	Translation, chatbots, summarisation, sentiment analysis
Computer vision	Understanding images and video	Medical scans, traffic cameras, quality inspection
Speech	Speech-to-text and text-to-speech	Voice typing, voice assistants, audiobooks
Robotics	AI that senses and acts in the physical world	Warehouse robots, drones, self-driving research
Recommender systems	Suggest what you might like	YouTube, shopping and music apps
Generative AI	Creates new content — text, images, audio, video, code	AI chat assistants, image generators, code assistants

1.5 Discriminative (predictive) versus generative AI

Predictive / discriminative AI

Looks at an input and makes a decision or prediction about it.

- "Is this email spam?" → yes/no
- "Which digit is this?" → 7
- "Will this customer repay?" → 82% likely

Generative AI

Produces new content that resembles what it learned from.

- "Write an email declining the meeting" → a new email
- "Draw a lighthouse at sunset" → a new image
- "Summarise this report" → a new summary

Module 3 explains how generative models do this.

1.6 The ingredients of modern AI

- **Data** — large collections of examples: text from books and the web, images, audio, sensor readings.
- **Algorithms and models** — mathematical structures (such as neural networks) with adjustable numbers called **parameters**.
- **Computing power** — GPUs and specialised AI chips that do billions of calculations in parallel.
- **Training** — the process of adjusting parameters so the model's outputs match the examples better.
- **Human feedback** — people rate or correct outputs to make models more helpful and safer.

The explosion of AI since 2012 came from all three growing together: much more data (the internet, smartphones), much more compute (GPUs), and better algorithms (deep learning, transformers).

1.7 AI in everyday life

Area	How AI is used
Phones	Face unlock, camera enhancement, voice assistants, keyboard suggestions, call screening

Language	Translation between Indian languages (e.g. government initiatives such as Bhashini), live captions, voice typing in Hindi, Odia, Tamil and more
Banking & payments	Fraud detection on UPI and card transactions, credit scoring, customer-service chatbots
Healthcare	Reading X-rays and scans, predicting risks, drug discovery, health chatbots (always with doctors in charge)
Agriculture	Crop-disease detection from photos, weather and price advisories, yield prediction
Education	AI tutors, personalised practice, automatic feedback, accessibility tools
Transport	Maps and traffic prediction, ride pricing, driver-assistance features
Work & business	Writing and summarising documents, coding help, customer support, marketing content, data analysis
Entertainment	Recommendations, game characters, music and video tools

Try it — spot the AI

List ten apps on your phone. For each, write one way it probably uses AI (recommendations, face detection, search, speech, fraud checks...). You'll likely find AI in most of them.

1.8 Myths and facts

Myth	Fact
AI thinks and feels like a human.	Current AI finds patterns in data; it has no feelings or personal experiences, even when its language sounds human.
AI is always right because it's a computer.	AI can be confidently wrong ("hallucinations"), biased, or out of date. Always verify important outputs.
AI will replace all jobs.	AI changes jobs — automating some tasks and creating new ones. People who use AI well tend to be more productive than those who don't.
You need to code to use AI.	Most generative AI tools work with plain language. Coding helps for building AI apps (Module 8).
AI knows everything on the internet.	A model knows only what was in its training data up to a cut-off date, unless it is connected to search or documents.

Module 1 — Summary

- AI builds systems that perform tasks needing human intelligence; modern AI learns patterns from data.
- History: Turing (1950), Dartmouth (1956), expert systems, deep learning (2012), transformers (2017), ChatGPT (2022).
- All current AI is narrow AI; AGI is a research goal.
- AI $\supset$ machine learning $\supset$ deep learning $\supset$ generative AI.
- Predictive AI decides/labels; generative AI creates new content.
- Ingredients: data, models, compute, training, human feedback.

Practice questions — Module 1

1. Define artificial intelligence in your own words.
2. How does machine learning differ from traditional programming? Draw a diagram.
3. List five milestones in the history of AI with years.
4. Differentiate narrow AI and general AI.
5. Draw the nested diagram of AI, ML, deep learning and generative AI.
6. Give two examples each of predictive and generative AI.
7. What three factors caused the rapid growth of AI after 2012?
8. Correct this statement: "AI chatbots understand everything and never make mistakes."

MODULE 02

Machine Learning and Deep Learning Basics

Supervised, unsupervised and reinforcement learning; training, testing and overfitting; how a neural network works — neurons, layers, weights and backpropagation — and why depth and data made modern AI possible.

2.1 What does "learning" mean for a machine?

A machine-learning model is a mathematical function with many adjustable numbers (**parameters** or **weights**). **Training** means showing it examples and nudging those numbers, again and again, so that its outputs get closer to the correct answers. After training, the model can handle new examples it has never seen — this ability is called **generalisation**.

Example — predicting house rent

From 1,000 examples of flats (area, bedrooms, locality → rent), a model learns roughly: $rent \approx ₹12 \times area + ₹3,000 \times bedrooms + locality\ adjustment$. The numbers 12 and 3,000 are parameters found by training — no one typed them in.

2.2 Three main types of machine learning

Type	Learns from	Typical tasks	Examples
Supervised learning	Labelled examples (input + correct answer)	Classification (which category?) and regression (what number?)	Spam or not; disease from X-ray; house price; exam score
Unsupervised learning	Unlabelled data — finds structure itself	Clustering , dimensionality reduction, anomaly detection	Grouping customers by behaviour; detecting unusual transactions
Reinforcement learning (RL)	Rewards and penalties from trying actions	Sequential decision-making	Game-playing AIs, robot control; RL from human feedback tunes chatbots

Self-supervised learning — the key to large language models — creates labels automatically from the data itself: hide the next word of a sentence and train the model to predict it. Because any text can be used, models can learn from enormous amounts of data without humans labelling it (Module 4).

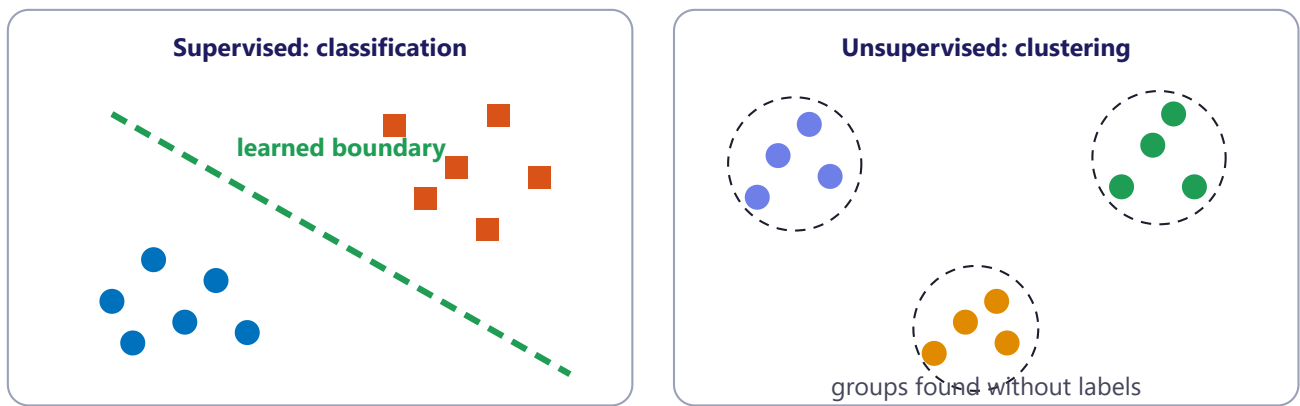


Figure 2.1 — Supervised learning separates labelled classes; unsupervised learning discovers groups by itself.

2.3 The machine-learning workflow

1. **Define the problem** — what to predict and how success is measured.
2. **Collect data** — enough, relevant, representative examples.
3. **Prepare data** — clean errors, handle missing values, convert text/categories to numbers (features), normalise scales.
4. **Split the data** — typically ~70–80% **training**, the rest for **validation** and **testing**.
5. **Train** a model on the training set.
6. **Evaluate** on the test set — data the model has never seen.
7. **Tune and improve** — more data, better features, different model, adjusted settings (hyperparameters).
8. **Deploy and monitor** — performance can drift as the world changes.

2.4 Overfitting and underfitting

A model that memorises its training examples — including their noise — performs brilliantly on them but badly on new data. This is **overfitting**, like a student who memorised last year's answers without understanding. A model too simple to capture the pattern is **underfitting**.

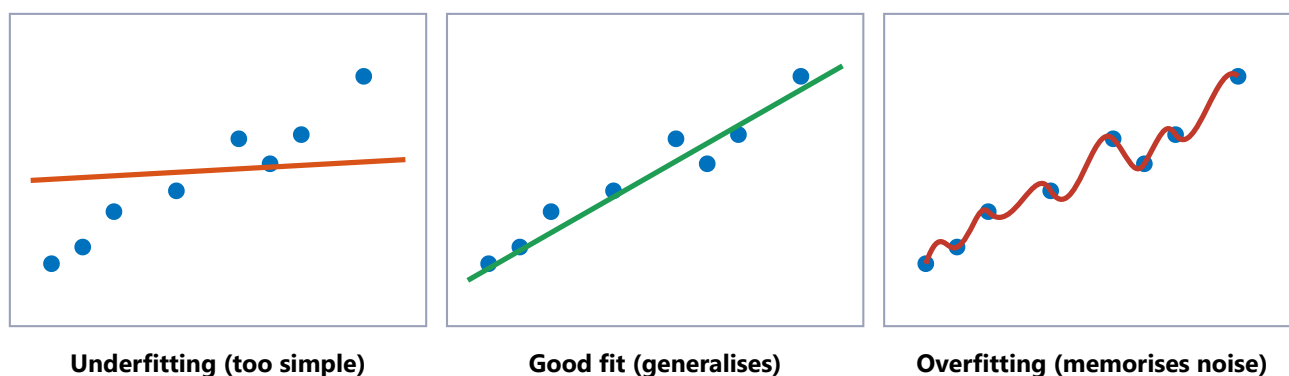


Figure 2.2 — Underfitting, good fit and overfitting.

Remedies for overfitting: more (and more varied) data, simpler models, **regularisation**, **dropout** in neural networks, and **early stopping** (stop training when validation error starts rising).

Measuring performance

Metric	Meaning
Accuracy	Fraction of predictions that are correct
Precision	Of the items predicted "positive", how many really are
Recall	Of all real positives, how many the model found
F1 score	Balance of precision and recall
Mean absolute error	Average size of error for number predictions

Worked example 2.1 — why accuracy can mislead

In 1,000 transactions, 10 are fraud. A model that always says "not fraud" is **99% accurate** but catches **zero** fraud (recall 0%). For rare events, look at precision and recall, not accuracy alone.

2.5 Neural networks — inspired by the brain

An **artificial neuron** takes several inputs, multiplies each by a **weight** (importance), adds them with a **bias**, and passes the total through an **activation function** that decides how strongly it "fires".

A single neuron

$$\text{output} = \text{activation}(w_1x_1 + w_2x_2 + \dots + w_nx_n + b)$$

Common activations: **ReLU** ($\max(0, z)$ — the most used), **sigmoid** (squashes to 0–1), **tanh**, and **softmax** (turns scores into probabilities that add up to 1 — used to pick the next word in language models).

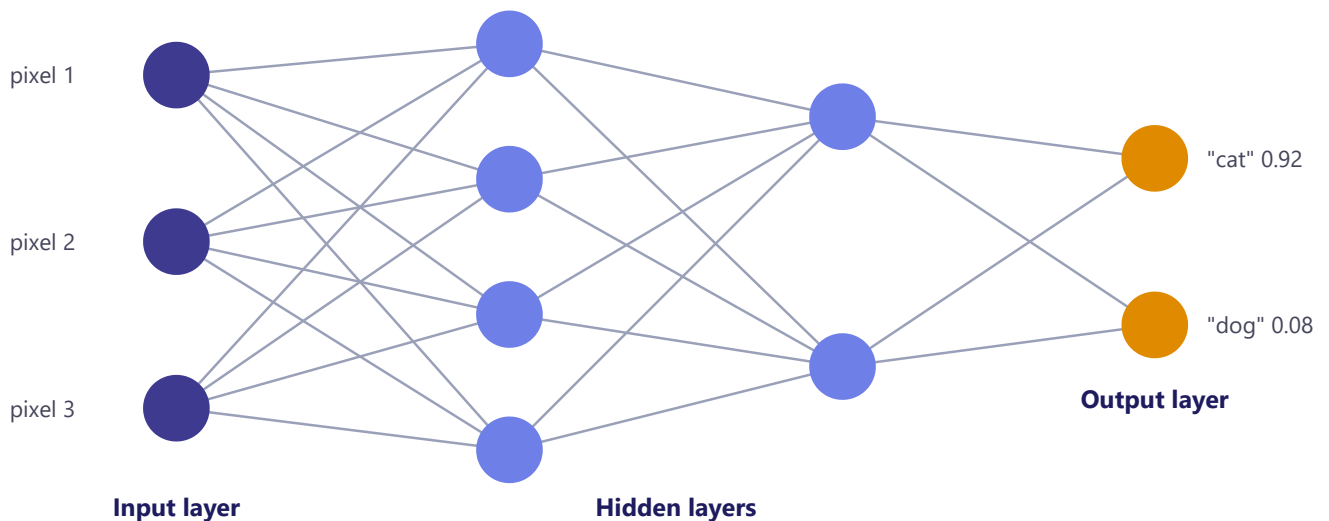


Figure 2.3 — A small feed-forward neural network. Each line is a weight learned during training.

- The **input layer** receives data (pixel values, word codes, sensor readings).
- **Hidden layers** transform it step by step, each learning more abstract features.
- The **output layer** gives the answer — a class probability, a number, or the next word.

2.6 How a network learns: loss, gradient descent and backpropagation

1. **Forward pass**: feed an example through the network to get a prediction.
2. **Loss**: measure how wrong it is (e.g. the model gave "cat" 0.3 when the answer was cat → large loss).
3. **Backpropagation**: work backwards through the network to calculate how much each weight contributed to the error (the **gradient**).
4. **Gradient descent**: nudge every weight a small step in the direction that reduces the loss. The step size is the **learning rate**.
5. Repeat over millions of examples, many passes (**epochs**), in small groups (**batches**).

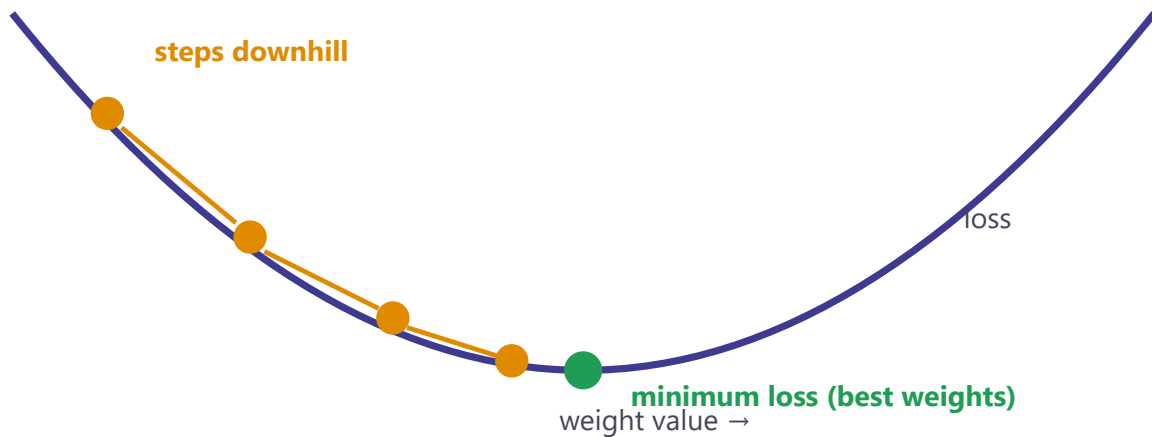


Figure 2.4 — Gradient descent: repeatedly step downhill on the loss curve to find better weights.

Key idea

Nobody programs the knowledge into a neural network. It emerges from millions of tiny weight adjustments that reduce prediction errors on training data. That's why data quality matters so much: the model learns whatever patterns — good or biased — the data contains.

2.7 Why "deep" learning — and why now?

Deep learning uses networks with many hidden layers. Each layer builds on the one before: in image recognition, early layers detect edges, middle layers detect shapes like eyes and wheels, and later layers detect whole objects. Deep networks became practical around 2012 because of:

- **Big data** — the internet and smartphones produced billions of images and documents.
- **GPUs** — graphics chips perform the huge numbers of parallel multiplications neural networks need.
- **Better techniques** — ReLU activations, dropout, better initialisation, and later the transformer architecture.
- **Open tools** — frameworks such as PyTorch and TensorFlow made building networks accessible.

A key discovery of recent years is **scaling**: for many tasks, performance improves predictably as models, data and compute grow — which is why today's language models have billions of parameters.

2.8 Important network types

Architecture	Good at	Examples
Feed-forward (MLP)	Tabular data, simple tasks	Credit scoring
CNN (convolutional)	Images, video — detects local patterns	Face recognition, medical imaging
RNN / LSTM (recurrent)	Sequences, processed word by word (older approach)	Earlier translation and speech systems
Transformer	Sequences of any kind, using attention to relate all parts at once; trains efficiently on GPUs	Language models, modern translation, vision transformers, speech
Autoencoder / VAE	Compressing data and generating variations	Image compression, anomaly detection
GAN	Generating realistic images	Face generation, style transfer
Diffusion model	High-quality image, audio and video generation	Text-to-image tools

2.9 Training versus inference

	Training	Inference (using the model)
What happens	Weights are learned	Fixed weights produce outputs for new inputs
Cost	Huge — large models need thousands of GPUs for weeks	Much smaller per request, but adds up at scale
Who does it	AI labs, large companies	Everyone who uses an AI app

Fine-tuning sits in between: taking a pre-trained model and training it a little more on a specific dataset (e.g. legal documents, a company's support tickets) to specialise it — far cheaper than training from scratch.

Module 2 — Summary

- ML models learn parameters from data and generalise to new examples.
- Supervised (labelled), unsupervised (structure), reinforcement (rewards); self-supervised powers LLMs.
- Workflow: define, collect, prepare, split, train, evaluate, tune, deploy, monitor.
- Avoid overfitting; use suitable metrics (precision/recall for rare events).
- Neural networks: weighted sums + activations in layers; trained by backpropagation and gradient descent.
- Deep learning took off with big data, GPUs and better techniques; transformers dominate today.
- Training is expensive; inference is using the model; fine-tuning specialises a pre-trained model.

Practice questions — Module 2

1. Explain supervised, unsupervised and reinforcement learning with one example each.
2. What is self-supervised learning, and why is it important for language models?
3. Why do we test a model on data it has not seen during training?
4. Explain overfitting with an everyday analogy. List three remedies.
5. A disease affects 2% of people. A model always predicts "healthy". What is its accuracy and recall? (Ans: 98%; 0%)
6. Write the formula for a neuron's output and name two activation functions.
7. Describe the steps of training a neural network.
8. Differentiate training, inference and fine-tuning.

Introduction to Generative AI

How generative models differ from predictive ones, what "learning a distribution" means, and a tour of the major model families — autoregressive transformers, VAEs, GANs and diffusion models — plus multimodal and foundation models.

3.1 What makes AI "generative"?

Generative AI creates new content — text, images, music, speech, video, code, 3D models, even molecules — that resembles the data it was trained on, without copying any single example. Ask for "a poem about the monsoon in Odisha" and you get a new poem that has never existed before.

Predictive model learns...

the boundary between categories:
"given this email, is it spam?" — $P(\text{label} \mid \text{input})$.

Generative model learns...

what the data itself looks like, so it can produce new samples: "what does a typical email look like, word by word?" — $P(\text{data})$.

Learning a distribution — an intuition

Imagine measuring the heights of 10,000 students. The data forms a bell-shaped curve centred near 160 cm. A generative model learns that shape; afterwards, it can "invent" realistic new heights — mostly near 160 cm, occasionally 145 or 180 — that follow the same pattern. Generative AI does the same with vastly more complex data: the patterns of words in sentences, pixels in photographs, or notes in music.

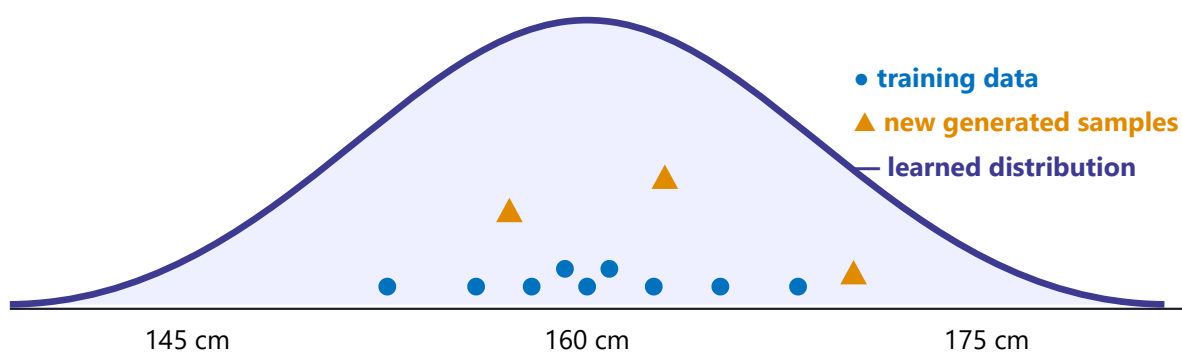


Figure 3.1 — A generative model learns the shape of the data, then samples new, realistic examples from it.

3.2 What generative AI can create

Modality	Examples of tasks
Text	Answers, essays, emails, summaries, translations, stories, product descriptions, lesson plans
Code	Writing, explaining, debugging and converting code; generating tests
Images	Illustrations, posters, product mock-ups, photo editing (remove objects, extend backgrounds)
Audio	Natural text-to-speech voices, music, sound effects, dubbing
Video	Short clips from text or images, animation, avatars
3D & design	3D objects, interior layouts, CAD suggestions
Science	Protein structures, drug-like molecules, new materials
Data	Synthetic data for testing and training

3.3 Family 1 — Autoregressive models (transformers for text)

An **autoregressive** model generates content **one piece at a time**, each new piece predicted from all the pieces before it. For text, the pieces are **tokens** (words or parts of words). The model predicts a probability for every possible next token, picks one, appends it, and repeats.

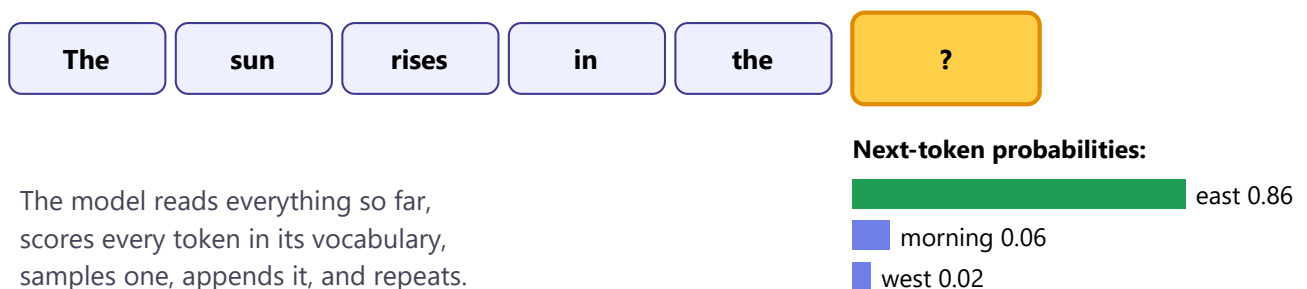


Figure 3.2 — Autoregressive generation: predict the next token, add it, repeat.

Large language models (Module 4) work this way. Many image, audio and video systems also use transformers.

3.4 Family 2 — Variational autoencoders (VAEs)

An **autoencoder** has two halves: an **encoder** that compresses the input into a small set of numbers (a **latent** representation), and a **decoder** that rebuilds the input from it. A

variational autoencoder organises that latent space smoothly, so that picking a new point in it and decoding gives a new, plausible output. VAEs are also used inside modern image generators to compress images into a smaller latent space where generation is cheaper.



Figure 3.3 — A (variational) autoencoder compresses data to a latent code and decodes it back.

3.5 Family 3 — Generative adversarial networks (GANs)

Introduced by Ian Goodfellow and colleagues in 2014, a **GAN** trains two networks against each other:

- The **generator** creates fake samples from random noise — like a counterfeiter.
- The **discriminator** tries to tell real samples from fakes — like a police detective.

Each improves by competing with the other until the fakes are very hard to distinguish from real data. GANs produced the first strikingly realistic synthetic faces. They are fast at generation but can be unstable to train and may produce limited variety ("mode collapse").

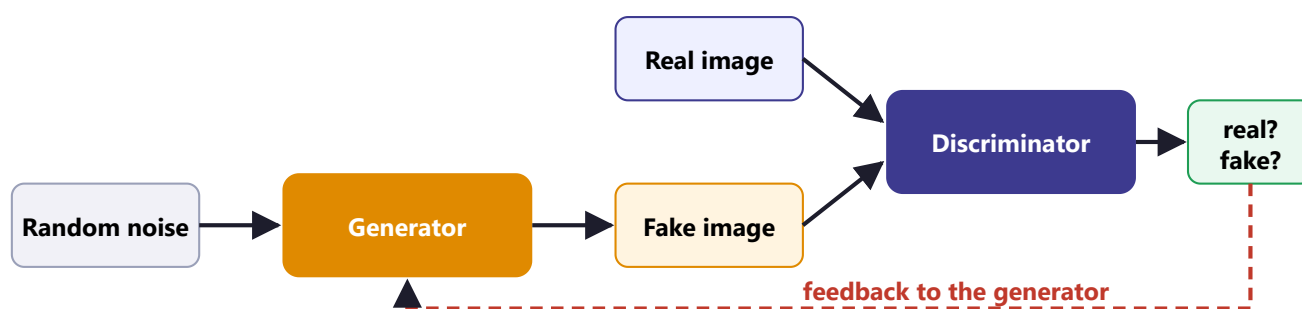


Figure 3.4 — A GAN: generator and discriminator improve by competing.

3.6 Family 4 — Diffusion models

Diffusion models power most modern text-to-image and many video tools. The idea has two halves:

1. **Forward process (training data preparation):** take a real image and add a little random noise, step by step, until it becomes pure static.

2. **Reverse process (what the model learns):** a neural network learns to **remove** a little noise at each step. To generate, start from pure noise and denoise it step by step into a brand-new image. A text prompt (encoded by a language model) **guides** the denoising toward images that match the description.

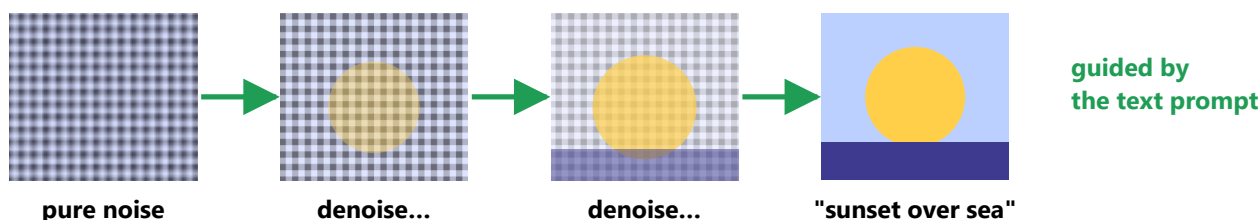


Figure 3.5 — Diffusion generation: start from noise and remove it step by step, guided by the prompt.

Diffusion models produce high-quality, varied images and are stable to train; their main cost is that generation takes many denoising steps (engineering tricks have made this much faster).

3.7 Comparing the families

Family	Core idea	Strengths	Common uses
Autoregressive transformer	Predict the next token repeatedly	Excellent with language and code; flexible	Chat assistants, writing, coding, translation
VAE	Encode to a latent space, decode samples	Smooth latent space; efficient compression	Parts of image generators, anomaly detection
GAN	Generator vs discriminator	Sharp images, fast generation	Face synthesis, super-resolution, style transfer
Diffusion	Learn to remove noise step by step	High quality and variety; good prompt control	Text-to-image, image editing, video, audio

3.8 Foundation models and multimodality

A **foundation model** is a very large model trained on broad data that can be adapted to many tasks — through prompting, fine-tuning, or connecting it to tools. Instead of building a separate model for translation, summarisation and question answering, one foundation model does all of them.

Multimodal models handle several types of data together: they can read an image and answer questions about it, listen to speech, describe a chart, or generate an image from a

conversation. Most leading AI assistants are now multimodal.

3.9 Open and closed models

	Closed / proprietary models	Open-weight models
Access	Through the company's app or paid API	Weights can be downloaded and run on your own hardware
Pros	Usually the most capable; no infrastructure to manage; safety systems included	Control, privacy (data stays local), customisation, no per-request fees
Cons	Cost per use; data leaves your system; less control	Need hardware and expertise; you handle safety

Both approaches are widely used; the right choice depends on the task, budget, privacy needs and skills available.

3.10 Strengths and limitations of generative AI

Strengths

- Fast first drafts of almost anything
- Works in plain language — no coding needed
- Many languages, including Indian languages
- Explains, summarises, translates, brainstorms
- Always available; endlessly patient

Limitations

- Can invent facts confidently (hallucination)
- Knowledge may be out of date
- Can reflect biases in training data
- Weak at some precise tasks (exact arithmetic, counting) unless using tools
- Copyright, privacy and misuse concerns (Module 9)

Module 3 — Summary

- Generative AI learns the distribution of data and samples new content from it.
- It creates text, code, images, audio, video, 3D and scientific designs.
- Autoregressive transformers predict the next token; VAEs encode/decode latents; GANs pit generator against discriminator; diffusion models denoise step by step.
- Foundation models adapt to many tasks; multimodal models handle several data types.
- Closed vs open-weight models trade convenience for control.
- Powerful for drafts and ideas — but verify facts and use responsibly.

Practice questions — Module 3

1. Differentiate predictive and generative models with examples.
2. Explain "learning a distribution" using an everyday example.
3. Describe how an autoregressive model generates a sentence.
4. Explain the roles of the generator and discriminator in a GAN.
5. Describe the forward and reverse processes of a diffusion model.
6. Compare GANs and diffusion models (two strengths each).
7. What is a foundation model? What is a multimodal model?
8. List three strengths and three limitations of generative AI.

MODULE 04

Large Language Models (LLMs)

Tokenization, embeddings, attention and the transformer; pre-training, instruction tuning and human feedback; context windows, temperature and sampling; why LLMs hallucinate; and how retrieval and tools extend them.

4.1 What is a large language model?

A **large language model (LLM)** is a neural network — almost always a transformer — trained on enormous amounts of text to predict the next token. Doing this well across trillions of words forces the model to capture grammar, facts, reasoning patterns, styles and even some world knowledge. "Large" refers to the number of parameters (typically billions) and the size of the training data.

Key idea — one simple objective, surprising abilities

An LLM is trained only to answer "what comes next?" Yet to predict the next word of a physics solution, a legal letter or Python code, it must model how those things work. Abilities like summarising, translating and step-by-step reasoning emerge from this single training objective at scale.

4.2 Tokenization — how text becomes numbers

Neural networks work with numbers, so text is first split into **tokens** — common words, pieces of words, punctuation or characters — and each token gets an ID number from the model's **vocabulary** (often 30,000–200,000 tokens).

Text: "Unbelievable solar panels!"

Un	believ	able	_solar	_panels	!
1844	31141	481	16865	21988	0

6 tokens → 6 ID numbers (IDs shown are illustrative; each model has its own tokenizer)

Figure 4.1 — Tokenization splits text into sub-word pieces and maps them to numbers.

- A rough rule for English: **1 token $\approx$ $\frac{3}{4}$ of a word** (about 4 characters). 1,000 words $\approx$ 1,300–1,500 tokens.

- Languages written in other scripts (Hindi, Odia, Tamil) often need **more tokens** per word with many tokenizers — making them somewhat slower and costlier to process. Newer tokenizers are improving this.
- Pricing, speed and limits for LLM APIs are measured in tokens (input tokens + output tokens).

4.3 Embeddings — meaning as coordinates

Each token ID is converted into an **embedding**: a long list of numbers (e.g. 4,096 of them) that places the token in a "meaning space". Words used in similar ways end up close together: *king* near *queen*, *Bhubaneswar* near *Cuttack*, *happy* near *glad*. Directions in this space can capture relationships — the classic example: *king* – *man* + *woman* $\approx$ *queen*.

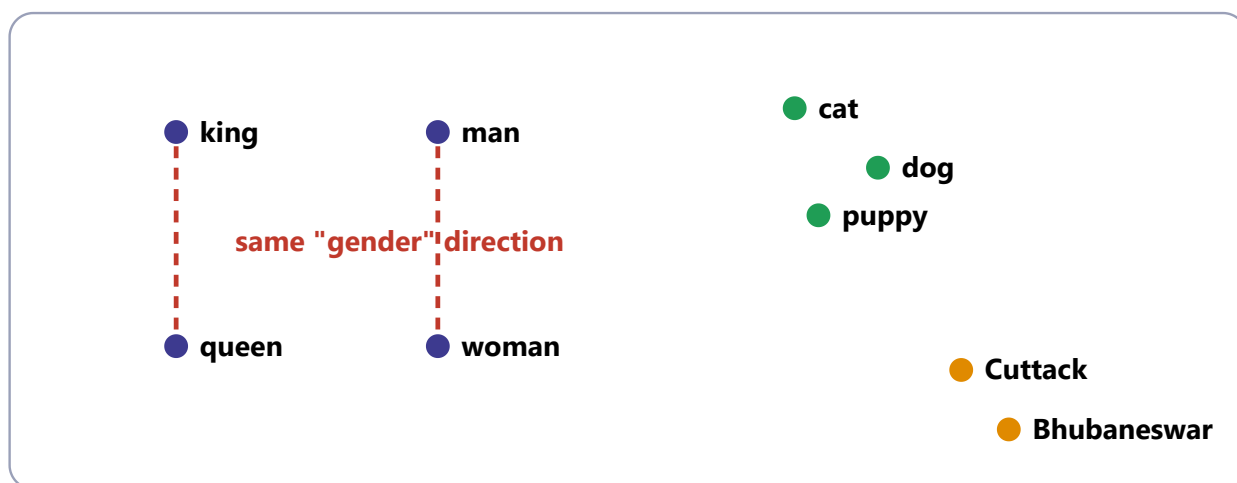


Figure 4.2 — A simplified 2D picture of embedding space: related meanings sit close together.

Embeddings are also used directly for **semantic search**: convert documents and a question into embeddings and find the documents whose meaning is closest — the basis of retrieval-augmented generation (4.10).

4.4 Attention — the heart of the transformer

The meaning of a word depends on the other words around it. In "The **bank** of the river flooded" and "I deposited money in the **bank**", the same word means different things. **Self-attention** lets every token look at every other token in the text and decide how much each one matters for understanding it.

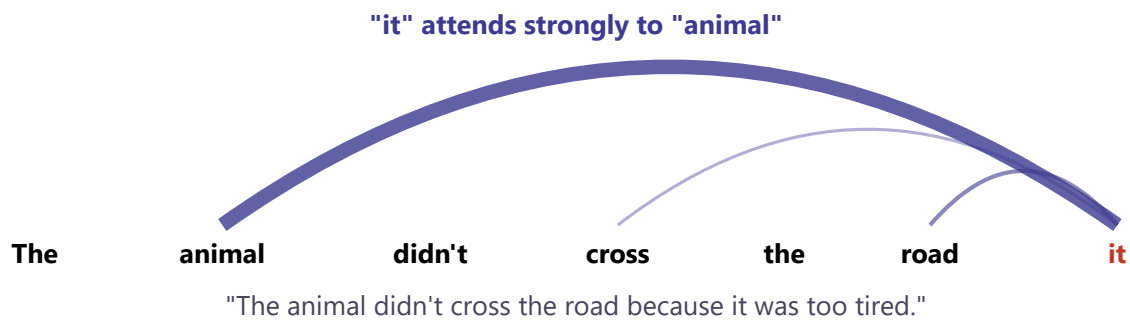


Figure 4.3 — Self-attention: thicker lines show which earlier words matter most for interpreting "it".

Technically, each token produces a **query**, a **key** and a **value** vector. The query of one token is compared with the keys of all tokens to produce attention weights, which then blend the values. Many **attention heads** run in parallel, each learning different relationships (grammar, references, topic), and many **layers** are stacked.

The transformer block

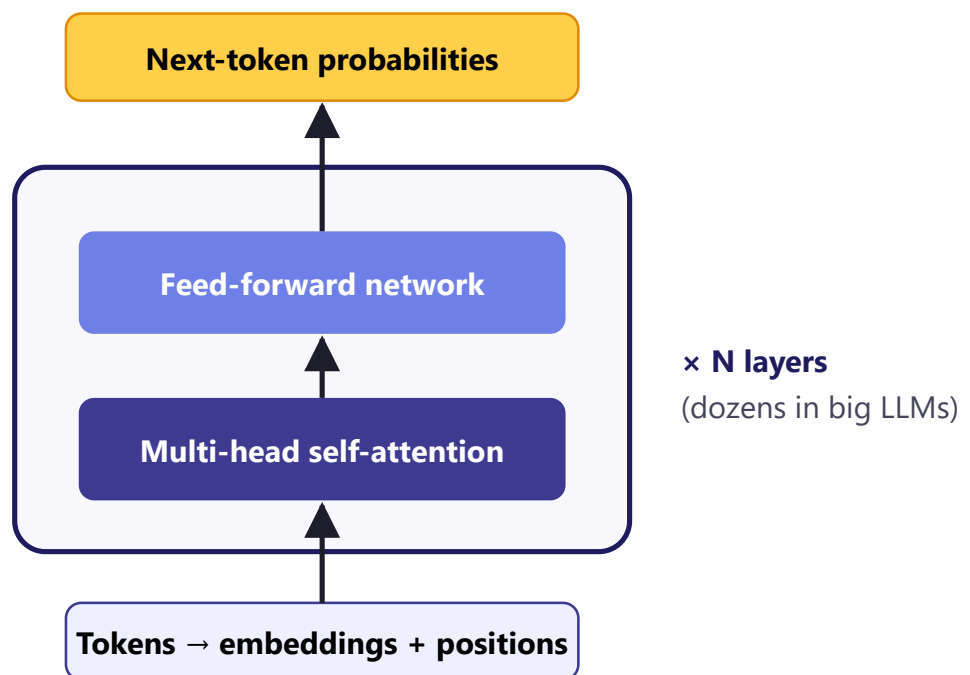


Figure 4.4 — Simplified decoder-only transformer used by most LLMs. (Residual connections and normalisation are omitted.)

Because attention processes all tokens in parallel (unlike older recurrent networks that read word by word), transformers train very efficiently on GPUs — the main reason they could be scaled to today's sizes.

4.5 How LLMs are trained — three stages

Stage	What happens	Result
1. Pre-training	Self-supervised next-token prediction on huge text (and code) collections — books, websites, articles — for weeks on thousands of GPUs	A base model that continues text well but doesn't reliably follow instructions
2. Instruction tuning (supervised fine-tuning)	Training on many examples of instructions paired with high-quality answers	A model that follows requests: "summarise this", "write a letter"
3. Preference tuning (e.g. RLHF)	Humans (and AI helpers) compare answers; the model is tuned to prefer helpful, honest, harmless ones	A chat/assistant model that is more helpful and safer, and declines harmful requests

Many assistants are further trained to reason step by step before answering ("reasoning models"), to use tools, and to follow safety policies.

4.6 The context window

The **context window** is the maximum number of tokens the model can consider at once — your prompt, any documents you include, the conversation so far, and its reply. Context windows have grown from a few thousand tokens to hundreds of thousands or more in some models, enough for whole books.

- A chat "remembers" earlier messages only because they are sent again inside the context window. Beyond the limit, the oldest parts are dropped or summarised.
- The model has no memory between separate chats unless the app adds a memory feature.
- Very long contexts cost more and models may pay less attention to details buried in the middle — put key instructions clearly, often at the start or end.

4.7 Generating text: temperature and sampling

At each step the model outputs probabilities for every token. How the next token is chosen changes the style of output:

Setting	Effect	Use for
Temperature low ($\approx$ 0–0.3)	Nearly always picks the most likely token — focused, consistent, repeatable	Facts, extraction, code, classification

Temperature medium (≈ 0.7)	Balanced	General writing, chat
Temperature high ($\approx 1.0+$)	More surprising choices — creative, varied, but more errors	Brainstorming, poems, story ideas
Top-p (nucleus sampling)	Samples only from the smallest set of tokens whose probabilities add up to p (e.g. 0.9)	Controls randomness while avoiding very unlikely tokens
Max tokens	Upper limit on reply length	Cost and length control
Stop sequences	Text at which generation stops	Structured outputs

This is why asking the same question twice can give different answers: sampling introduces controlled randomness.

4.8 Why LLMs hallucinate

A **hallucination** is output that sounds confident and fluent but is false — an invented statistic, a fake reference, a wrong date. It happens because:

- The model is optimised to produce **plausible** text, not to check truth.
- Its knowledge is compressed and imperfect, especially for rare facts, recent events (after its training cut-off) and precise numbers.
- When unsure, a fluent guess may look more "likely" than saying "I don't know" — although modern assistants are trained to admit uncertainty more often.

Caution — verify what matters

Never rely on an LLM alone for medical, legal, financial or safety-critical facts, citations, statistics or current events. Ask it for sources, check them, and prefer tools connected to search or trusted documents.

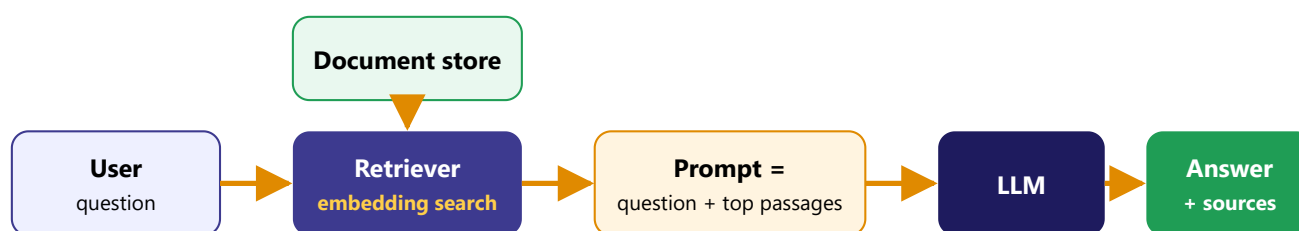
4.9 Other limitations

- **Knowledge cut-off:** without search tools, a model doesn't know events after its training data ended.
- **Arithmetic and counting:** can slip on long calculations or letter counts — better when it uses a calculator/code tool or reasons step by step.
- **Bias:** reflects patterns and stereotypes in training data.

- **Sensitivity to wording:** small prompt changes can change answers (Module 5 addresses this).
- **Prompt injection:** text hidden in documents or web pages can try to hijack an AI's instructions — a security concern for AI apps (Module 8).

4.10 Extending LLMs: retrieval, tools and agents

Technique	How it works	Example
RAG (retrieval-augmented generation)	Search a document collection for passages relevant to the question and insert them into the prompt, so the answer is grounded in real sources	A college chatbot answering from the official prospectus and rules
Tool use / function calling	The model decides to call external functions — search, calculator, database, weather API — and uses the results	"What's the fee for Class 12?" → looks up the fee table
Code execution	The model writes and runs code to calculate or analyse data exactly	Analysing an uploaded spreadsheet
Agents	The model plans multi-step tasks, calling tools in a loop and checking results	Research a topic, compile a report, draft emails
Fine-tuning	Further training on examples to specialise style or task	Consistent tone for a company's support replies



RAG grounds answers in your own up-to-date documents and reduces hallucination.

Figure 4.5 — Retrieval-augmented generation (RAG).

4.11 Choosing a model

- **Capability** — larger/"frontier" models reason better; small models are faster and cheaper for simple tasks.
- **Context window** — needed length of documents.
- **Modalities** — images, audio, files?
- **Languages** — quality in the Indian languages you need.

- **Cost and speed** — price per million tokens, latency.
- **Privacy and deployment** — cloud API vs on-premises open-weight model; data-use policies.
- **Test on your own examples** — benchmarks don't always predict performance on your task.

Module 4 — Summary

- LLMs are transformers trained to predict the next token; abilities emerge at scale.
- Text → tokens → embeddings; $\sim \frac{3}{4}$ word per token in English.
- Self-attention relates every token to every other; stacked layers build understanding.
- Training: pre-training → instruction tuning → preference tuning (RLHF).
- Context window limits what the model sees at once; temperature/top-p control randomness.
- Hallucinations arise because models optimise plausibility; verify important facts.
- RAG, tools, code execution and agents extend LLMs with real data and actions.

Practice questions — Module 4

1. What does "large" mean in "large language model"?
2. What is a token? Estimate the tokens in a 750-word English essay. (*Ans: $\approx 1,000$*)
3. Explain embeddings with an example of words that would be close together.
4. Explain self-attention using the sentence "The bank of the river was muddy."
5. Describe the three stages of training an assistant LLM.
6. What is a context window, and why does a long chat eventually "forget" early messages?
7. Which temperature would you use for (a) extracting dates from invoices, (b) writing a poem? Why?
8. What is a hallucination? Give two ways to reduce its risk.
9. Explain RAG with a diagram and an example use in a school.

MODULE 05

Prompt Engineering Techniques

How to write prompts that get excellent results: the anatomy of a good prompt, zero-shot and few-shot prompting, step-by-step reasoning, roles and system prompts, structured output, and a systematic way to test and improve prompts.

5.1 What is prompt engineering?

A **prompt** is the input you give a generative AI model — your question, instructions, examples and any material to work on. **Prompt engineering** is the skill of writing prompts that reliably produce useful, accurate output. The same model can give a vague, generic answer or an excellent, tailored one depending entirely on how it is asked.

Key idea — brief the AI like a smart new colleague

Imagine a very capable assistant who knows nothing about you, your audience or your purpose. What would they need to do the job well? Context, a clear task, constraints, examples and the format you want. Good prompts supply exactly that.

5.2 The anatomy of a good prompt

Element	Question it answers	Example
Role / perspective (optional)	Whose expertise or viewpoint?	"You are an experienced Class 12 physics teacher."
Context	What's the situation and who is it for?	"My students are from Odia-medium schools and find English terms hard."
Task	What exactly should be done?	"Explain Ohm's law."
Constraints	Limits and requirements	"Under 200 words, simple English, one everyday analogy, no calculus."
Examples	What does good look like?	"Here is an explanation I liked: ..."
Output format	How should the answer be structured?	"A heading, 3 bullet points, then one practice question with the answer."

Input material	Text/data to work on	Paste the chapter, email or data (clearly separated)
-----------------------	----------------------	--

Worked example 5.1 — from weak to strong

Weak: "Write about solar energy." → a generic essay of unknown length for an unknown reader.

Strong: "You are writing for a village panchayat meeting in Odisha. Explain in simple English, under 250 words, how a 3 kW rooftop solar system can reduce a family's electricity bill. Include: roughly how many units it generates per month, what happens during a power cut, and two maintenance tips. Use short paragraphs and end with three questions the audience might ask, with brief answers."

The strong prompt fixes audience, purpose, length, content and format — so the first answer is usually usable.

Prompt pattern — the CRAFT checklist

Context · Role · Action (task) · Format · Tone/constraints. Not every prompt needs all five, but checking them turns vague requests into precise ones.

5.3 Be clear, specific and direct

- **Say what you want, not only what you don't:** "Write in short sentences for a 12-year-old" works better than "Don't be complicated".
- **Quantify:** "5 bullet points", "about 150 words", "3 options".
- **Name the audience:** parents, engineers, first-year students, customers.
- **Give the purpose:** why you need it changes what a good answer is.
- **Separate instructions from material** using headings, quotes or tags: `<report> ...`
`</report> .`
- **Put key instructions at the start (and repeat critical ones at the end)** for long prompts.

5.4 Zero-shot, one-shot and few-shot prompting

Technique	What you give	When to use
Zero-shot	Instructions only	Common tasks the model already does well

One-shot	Instructions + one example	To show a format or style
Few-shot	Instructions + 2–5 examples	Unusual formats, consistent classification, a particular voice

Worked example 5.2 — few-shot classification of feedback

Classify each student comment as Positive, Negative or Suggestion.

Comment: "The doubt sessions really helped me." → Positive

Comment: "Classes start late every day." → Negative

Comment: "Please share notes as PDF." → Suggestion

Comment: "Tests are too frequent and stressful." →

Comment: "Could we have weekend revision classes?" →

Output: Negative, Suggestion. The examples define the labels and format precisely. Use varied examples so the model learns the pattern, not a specific wording.

5.5 Step-by-step reasoning (chain of thought)

For problems that need reasoning — maths, logic, planning, analysis — asking the model to **work through the steps before giving the final answer** often improves accuracy. Many modern models do this automatically ("reasoning" or "thinking" modes), but explicit instructions still help for multi-part problems.

Worked example 5.3

Prompt: "A shop sells a 3 kW solar system for ₹1,80,000. The subsidy is ₹78,000. It saves ₹2,200 per month. Work out the simple payback period. Show each step, then give the final answer on its own line."

Output (summary): Net cost = $1,80,000 - 78,000 = ₹1,02,000$. Annual saving = $2,200 \times 12 = ₹26,400$. Payback = $1,02,000 \div 26,400 = 3.86$ years. **Final answer: about 3.9 years.**

Showing steps also lets **you** check the reasoning. For exact calculations, you can also ask the model to use a calculator or code tool if available.

5.6 Roles and system prompts

A **role** ("You are an experienced tax consultant...") nudges vocabulary, depth and perspective. In chat apps and APIs, a **system prompt** sets instructions that apply to the whole conversation

— the assistant's persona, rules, tone and boundaries — separately from each user message.

Worked example 5.4 — a system prompt for a school's AI tutor

```
You are "Spark", a friendly tutor for Class 9–12 students in India.
- Explain step by step, using simple English; offer Hindi or Odia terms if asked.
- For homework, guide with hints and questions first; give the full solution
  only if the student asks after trying.
- Use units in every physics answer. Check calculations.
- If a question is outside school subjects or unsafe, politely decline and
  suggest talking to a teacher or parent.
- If you are not sure of a fact, say so instead of guessing.
```

5.7 Asking for structured output

When output feeds into another step — a spreadsheet, a website, a program — ask for a precise structure:

- Tables: "Return a table with columns: Topic | Key formula | Common mistake."
- Lists and headings: "Use H2 headings for each day."
- **JSON** for software: specify the exact keys and types, and give an example. Many APIs also offer a "structured output"/JSON mode that guarantees valid JSON.

Worked example 5.5 — extraction to JSON

```
Extract the details from this enquiry and return ONLY valid JSON with keys:
name (string), phone (string), course (one of "Class 11","Class 12","Entrance"),
preferred_time (string or null).
```

```
Enquiry: "Hi, I'm Rohan, 98765 43210. Want Class 12 physics tuition,
evenings after 6 preferably."
```

```
{"name": "Rohan", "phone": "98765 43210", "course": "Class 12",
 "preferred_time": "evenings after 6"}
```

5.8 Providing context and source material

- Paste the relevant text (or attach files) and tell the model to **base its answer only on that material**: "Answer using only the policy below. If the answer isn't in it, say 'Not stated'."

This reduces hallucination.

- Ask for **quotes or section references** to support each claim, so you can check them.
- For long documents, ask for a structured summary first, then targeted questions.

5.9 Controlling tone, style and length

Goal	Prompt wording
Formal	"Formal, polite tone suitable for a government office."
Friendly	"Warm and encouraging, like a helpful senior student."
Simple	"Explain as if to a 12-year-old; avoid jargon; define any technical word."
Concise	"Maximum 100 words" / "one sentence per point".
Match a sample	"Match the style of this paragraph: ..."
Language	"Reply in Hindi (Devanagari script)" / "Reply in simple Odia".

5.10 Iterating: prompting is a conversation

The first answer is a draft. Improve it with follow-ups:

- "Make it shorter and more persuasive."
- "Add a real-world example from Indian agriculture."
- "The second point is wrong — the subsidy is per kW. Please correct."
- "Give me three different versions: formal, friendly and humorous."
- "What information would help you answer this better?" — let the model ask **you** clarifying questions.
- "Critique your answer for accuracy and clarity, then improve it."

5.11 Breaking big tasks into steps (prompt chaining)

Complex jobs work better as a sequence of smaller prompts, each with a clear output, than as one giant request.

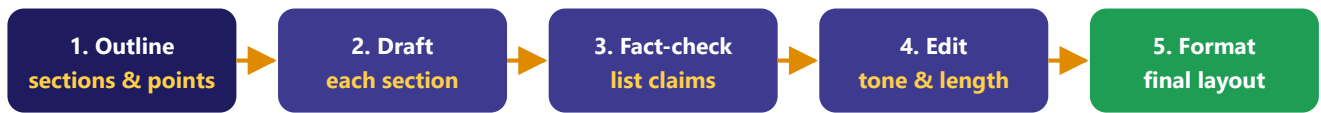


Figure 5.1 — Prompt chaining: each step's output becomes the next step's input.

5.12 Testing and improving prompts systematically

For prompts you will reuse — in a business, app or classroom — treat them like code:

1. Write down what a **good output** looks like (criteria).
2. Collect **10–20 realistic test inputs**, including tricky and edge cases.
3. Run the prompt on all of them; score the outputs against the criteria.
4. Change **one thing at a time** (add an example, clarify a rule) and re-test.
5. Keep versions and notes of what improved results.

5.13 Common prompting mistakes

Mistake	Better approach
Vague request ("make it better")	Say how: shorter, clearer, more examples, for whom
Missing context	Explain audience, purpose, background
Too many tasks in one prompt	Chain steps
Trusting outputs blindly	Ask for sources/steps; verify facts
Sharing private data	Remove names, numbers, confidential info (Module 9)
Accepting the first draft	Iterate with specific feedback
Contradictory instructions	Re-read the prompt; prioritise clearly

Module 5 — Summary

- Good prompts give context, a clear task, constraints, examples and an output format (CRAFT).
- Zero-shot for simple tasks; few-shot to teach formats and labels.
- Ask for step-by-step reasoning on problems; roles and system prompts set persistent behaviour.
- Request structured output (tables, JSON) when results feed other tools.
- Ground answers in provided material to reduce hallucination.
- Iterate, chain complex tasks, and test reusable prompts systematically.

Practice questions — Module 5

1. List the elements of a good prompt with an example of each.
2. Rewrite this prompt to be specific: "Write a letter to the principal."
3. Write a few-shot prompt that classifies customer messages as Complaint, Order or Question.
4. When is chain-of-thought prompting useful? Give an example.
5. Write a system prompt for a hospital reception chatbot (include safety rules).
6. Write a prompt that extracts invoice number, date and total into JSON.
7. Explain prompt chaining for writing a project report.
8. Describe how you would test a prompt that will be used 500 times a day.

MODULE 06

AI Text and Content Generation

Using LLMs for writing, summarising, translating, editing, studying, coding and business content — with ready-to-use prompt templates — and knowing exactly where they tend to go wrong.

6.1 The main text tasks

Task	What AI does well	Where to be careful
Drafting	Emails, letters, reports, posts, stories, speeches	Generic tone; invented details
Summarising	Condensing long documents, meeting notes, articles	Missing or distorting key points; check numbers
Rewriting & editing	Grammar, clarity, tone, simplifying, shortening	Changing meaning; removing your voice
Translating	Between English and many Indian and world languages	Idioms, technical/legal terms, names; review by a fluent speaker
Explaining & tutoring	Step-by-step explanations, examples, practice questions	Errors in advanced or niche topics
Brainstorming	Ideas, titles, outlines, angles, names	Ideas may be clichéd — push for variety
Extracting & organising	Pulling data from text into tables/JSON; classifying	Missing items in long texts
Coding	Writing, explaining and debugging code; formulas	Subtle bugs, security issues, outdated libraries — test everything

6.2 Writing with AI — a reliable workflow

1. **Brief:** audience, purpose, key points, tone, length, format (Module 5).
2. **Outline first:** ask for a structure; adjust it before full drafting.
3. **Draft section by section** for long pieces.
4. **Add your knowledge:** real facts, examples, numbers, experience — the parts AI can't know.
5. **Edit:** ask AI to tighten, then edit personally for accuracy and voice.
6. **Verify** every fact, figure, name and reference.

7. Take responsibility: you are the author; AI is the assistant.

Prompt template — professional email

Write a [formal/friendly] email to [recipient and relationship].
Purpose: [what you need].
Key points: 1) ... 2) ... 3) ...
Tone: [polite, firm, apologetic...]. Length: under [N] words.
End with a clear request: [action and deadline].

Worked example 6.1 — email

Prompt: "Write a polite, firm email to our internet provider. Purpose: complaint about 5 outages in 2 weeks at our coaching centre. Key points: outages on 3, 5, 8, 11 and 14 September; each lasted 2–4 hours; classes disrupted; complaint ticket #45821 not resolved. Ask for a technician visit within 48 hours and a bill adjustment. Under 170 words."

The result is ready after a quick check of the dates and ticket number — which came from you, not the AI.

6.3 Summarising

Prompt template — summary

Summarise the text below for [audience] in [N] bullet points.
Include: main conclusion, key numbers, decisions and action items (who/when).
Do not add information that is not in the text.
<text> ... </text>

- **Choose the type:** executive summary (decisions), study summary (concepts), meeting minutes (actions), abstract (research).
- For long documents, summarise part by part, then combine.
- Ask: "List any numbers and dates mentioned, with the sentence they appear in" — then verify them.

6.4 Translating and working in Indian languages

- Modern LLMs handle Hindi, Bengali, Odia, Tamil, Telugu, Marathi, Gujarati, Kannada, Malayalam, Punjabi and more — quality varies by language and model.

- Specify **script** (Devanagari, Odia script, or Roman "Hinglish"), **register** (formal/colloquial) and **audience** (farmers, officials, children).
- Give a **glossary** for technical terms: "Translate 'net metering' as 'नेट मीटरिंग' and keep 'kW' in English."
- Use **back-translation** to check: translate the output back to English and compare meaning.
- For legal, medical and official documents, always have a qualified human translator review.

Worked example 6.2

"Translate this notice into simple Odia for parents in a rural school. Keep dates as numbers. Formal but warm tone. After the translation, give a line-by-line English back-translation so I can check the meaning."

6.5 Editing and rewriting

Goal	Prompt
Proofread only	"Correct grammar and spelling only. Don't change wording otherwise. List each change."
Simplify	"Rewrite for a Class 8 reading level. Keep all facts."
Shorten	"Cut to 120 words without losing the three main points."
Change tone	"Make this complaint polite but firm."
Improve flow	"Improve transitions between paragraphs; keep my voice."
Feedback, not rewrite	"Give me 5 specific suggestions to improve this essay; don't rewrite it."

6.6 AI for studying (used honestly)

- **Explain differently:** "Explain electromagnetic induction with an analogy from daily life, then with the formula."
- **Socratic tutor:** "Don't give the answer; ask me questions to guide me to solve this."
- **Practice:** "Make 10 MCQs on Chapter 3 with answers and explanations; mix easy and hard."
- **Flashcards and revision notes** from your own notes.

- **Feedback on your answers:** "Mark my answer out of 5 using the rubric, and explain what's missing."
- **Study plans:** "Make a 30-day revision timetable for these subjects and exam dates."

Caution — academic integrity

Submitting AI-written work as your own usually breaks school and university rules and robs you of learning. Use AI to understand, practise and get feedback; write assignments yourself unless your teacher allows AI use — and disclose it when you do.

6.7 AI for work and business content

Content	Prompt idea
Product descriptions	"Write a 60-word description of this handloom saree for an online store: [details]. Highlight craftsmanship and care. Include 5 SEO keywords."
Social media	"Write 5 Instagram captions for our admission drive, each under 25 words, with 3 hashtags; one in Hindi."
Job descriptions	"Create a job post for a part-time data-entry assistant: duties, skills, hours, salary range []."
Reports	"Turn these monthly sales figures into a one-page report with key trends and 3 recommendations."
Customer replies	"Draft a reply to this review: [review]. Apologise, explain the fix, invite them back."
Presentations	"Create a 10-slide outline with slide titles and 3 bullets each for [topic] for [audience]."
Spreadsheet formulas	"Write an Excel formula that gives 'Late' if the date in C2 is after the due date in D2."

6.8 AI for code

- **Generate:** "Write a Python function that reads a CSV of marks and returns each student's grade."
- **Explain:** "Explain this code line by line for a beginner."
- **Debug:** paste the code and the full error message; ask for the cause and the fix.
- **Convert:** "Convert this MATLAB script to Python with NumPy."
- **Test:** "Write test cases for this function, including edge cases."

Caution — AI-written code

Run and test all generated code. Watch for security problems (hard-coded passwords, unsafe input handling), invented library functions, and outdated methods. Never paste secret keys or confidential code into public AI tools.

6.9 Where AI text goes wrong — and how to catch it

Problem	Example	Prevention
Hallucinated facts	Invented statistics, laws, quotes, book titles, references	Ask for sources; verify; ground in provided documents
Outdated information	Old prices, rules, subsidy amounts	Check current official sources; use search-enabled tools
Generic, bland writing	Clichés, filler phrases	Give specifics, examples and a style sample; edit
Over-confidence	No hedging on uncertain points	Ask it to flag uncertain claims
Bias and stereotypes	Assuming gender for professions	Review; ask for inclusive language
Calculation slips	Wrong totals in long sums	Ask for step-by-step working or use a calculator/code tool
Meaning drift in edits	A "polish" that changes a fact	Ask it to list changes; compare versions
Plagiarism risk	Close paraphrase of known text	Write in your own words; cite sources

Try it — the verification prompt

After getting an answer, ask: *"List every factual claim in your answer as a numbered list, rate your confidence in each (high/medium/low), and tell me which ones I should verify and where."*

6.10 Disclosing AI use

Many schools, publishers, employers and platforms now ask people to disclose AI assistance. A simple note works: *"Draft prepared with the help of an AI assistant and reviewed and edited by the author."* Always follow your organisation's policy.

Module 6 — Summary

- LLMs excel at drafting, summarising, rewriting, translating, explaining, brainstorming, extracting and coding.
- Workflow: brief → outline → draft → add your knowledge → edit → verify → take responsibility.
- Use templates for emails, summaries, translations and business content.
- For Indian languages, specify script, register and glossary; back-translate to check.
- Use AI to learn, not to cheat; test all AI-written code.
- Watch for hallucinations, outdated facts, bias, calculation slips and meaning drift.

Practice questions — Module 6

1. List six text tasks AI does well and one risk for each.
2. Write a prompt to summarise a 10-page report into 5 bullet points for a manager.
3. How would you check an AI translation of a notice into Hindi?
4. Write a prompt that makes AI act as a Socratic tutor for a maths problem.
5. Write three prompts for a bakery's social media campaign.
6. What should you do before using AI-generated code in a project?
7. Give five ways AI-generated text can be wrong and how to catch each.
8. Draft a one-line disclosure of AI assistance for a project report.

MODULE 07

AI Image, Audio, and Video Generation

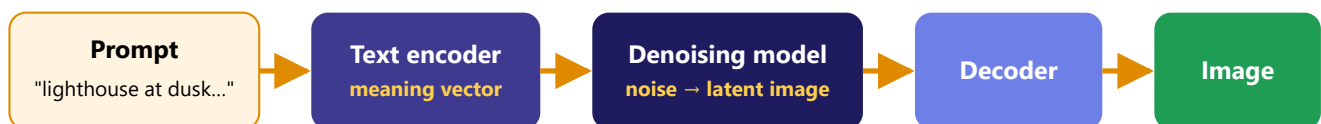
How text-to-image works, writing effective image prompts, editing images with AI, voice and music generation, text-to-video, and the rules for using AI media honestly and legally.

7.1 Text-to-image: how it works

Modern image generators are mostly **diffusion models** (Module 3). The process:

1. A **text encoder** converts your prompt into an embedding that captures its meaning.
2. Generation starts from random **noise** (usually in a compressed "latent" space for speed).
3. The model **removes noise step by step**, each step guided toward images that match the prompt embedding.
4. A decoder converts the result into a full-resolution image.

Because the starting noise is random, the same prompt gives different images each time. Many tools let you fix a **seed** to reproduce a result, and a **guidance** setting to control how strictly the image follows the prompt.



random noise + seed → repeated denoising steps guided by the prompt

Figure 7.1 — The text-to-image pipeline.

7.2 Writing effective image prompts

Prompt pattern — the image formula

Subject + details/action + setting + style/medium + lighting & colour + composition/camera + mood + aspect ratio

Element	Examples of words
Subject	"a young Odia girl studying", "a red vintage bicycle", "a solar-powered village school"
Details / action	"reading by a lantern", "leaning against a wall", "students raising hands"

Setting	"in a paddy field at dawn", "on a busy Mumbai street", "inside a modern lab"
Style / medium	"photorealistic", "watercolour", "flat vector illustration", "Pattachitra-inspired folk art", "3D render", "pencil sketch"
Lighting & colour	"golden-hour light", "soft diffused light", "neon", "pastel palette", "high contrast"
Composition / camera	"close-up", "wide shot", "top-down view", "shallow depth of field", "centred with empty space on the left for text"
Mood	"hopeful", "calm", "energetic", "mysterious"
Format	"square 1:1 for Instagram", "16:9 banner", "9:16 story"

Worked example 7.1 — improving an image prompt

Weak: "solar panels" → a generic stock-style photo.

Strong: "Photorealistic wide shot of a rooftop solar installation on a flat concrete roof of a two-storey house in a small Indian town, blue sky with a few clouds, late-morning sunlight, a technician in a safety helmet cleaning a panel with a soft brush, clean composition with empty sky on the right for a headline, 16:9."

Then iterate: "same scene, warmer colours", "make the technician a woman", "remove the water tank".

Tips for better images

- **Be concrete:** describe what should be visible, not abstract ideas.
- **Put the most important element first.**
- **Use a negative prompt** where supported ("no text, no watermark, no extra fingers").
- **Text inside images** (signboards, posters) has improved greatly in newer models but still check spelling — or add text later in a design tool like Canva.
- **Hands, crowds and fine details** may contain errors — inspect closely.
- **Consistency:** keeping the same character across images is hard; use reference-image features or character settings where available.
- **Represent people fairly:** specify diversity deliberately; models can default to stereotypes.

7.3 Editing images with AI

Feature	What it does	Use case
---------	--------------	----------

Inpainting / generative fill	Replace or add something in a selected area	Remove a stranger from a photo; add a plant to a room
Outpainting / expand	Extend an image beyond its borders	Turn a portrait photo into a wide banner
Background removal / replacement	Cut out the subject	Product photos on white backgrounds
Upscaling	Increase resolution with added detail	Printing a small photo larger
Style transfer / image-to-image	Redraw an image in a new style, guided by a prompt	Sketch → finished illustration
Photo restoration	Repair scratches, colourise old photos	Family archives
Image understanding	Multimodal models describe or analyse images	Read a chart; describe an image for alt text

7.4 Audio generation

Text-to-speech (TTS)

Neural TTS produces natural voices with expression, in many languages and accents, including Indian languages. Uses: audiobooks, e-learning narration, IVR phone menus, accessibility (screen readers), video voice-overs, announcements.

Voice cloning

Some tools can imitate a specific person's voice from a short sample. This has legitimate uses (restoring a voice lost to illness, dubbing with consent) but is also used for **fraud** — e.g. fake calls that sound like a family member asking for money.

Caution — voice scams

Never clone anyone's voice without their explicit permission. If you get an urgent call from a "relative" asking for money, hang up and call them back on their known number, or ask a family "safe word" question.

Speech-to-text (STT)

Transcribes and translates speech — meeting notes, subtitles, voice typing, lecture transcripts. Accuracy depends on audio quality, accents and background noise.

Music and sound generation

Models can generate songs (instrumental or with vocals) from a text description of genre, mood, tempo and instruments, as well as sound effects and background music. Check each tool's licence terms before commercial use, and avoid imitating identifiable artists.

Worked example 7.2 — music prompt

"A calm 60-second instrumental background track for an educational video about the solar system: soft piano and ambient pads, slow tempo (around 70 BPM), hopeful mood, no vocals, gentle fade-out at the end."

7.5 Video generation

Type	Description
Text-to-video	Generate short clips (typically seconds long) from a description
Image-to-video	Animate a still image — camera moves, subtle motion
Video editing	Remove objects, change backgrounds, extend clips, auto-captions, auto-cuts
AI avatars / presenters	A synthetic presenter reads a script — used for training videos and multilingual announcements
Dubbing and lip-sync	Translate speech into another language, matching lip movement

Prompt pattern — video

Subject & action + **setting** + **camera movement** ("slow pan left", "drone shot rising", "static close-up") + **style** + **lighting** + **duration/pace**. Keep each clip to one simple action; combine clips in an editor.

Current limitations: clips are short; physics and hands can look wrong; text and consistent characters are difficult; generation is expensive. Quality is improving very quickly.

7.6 A practical creative workflow

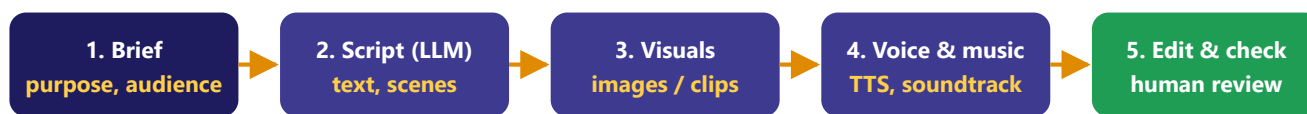


Figure 7.2 — Combining AI tools to make, for example, a 60-second promotional video.

Worked example 7.3 — admission promo in one afternoon

1. LLM: write a 60-second script with 6 scenes and on-screen text for "Sunrise Tuition — new batches".
2. Image generator: one image per scene in a consistent flat-illustration style (or use real photos of your centre — often more trustworthy).
3. Image-to-video: add gentle motion to each image.
4. TTS: voice-over in English and Odia; background music from a licensed or generated track.
5. Video editor: assemble, add captions, logo and phone number; check every word and number.

7.7 Copyright, consent and honesty

Issue	Good practice
Copyright of outputs	Laws differ by country and are still evolving; purely AI-generated works may have limited copyright protection. Check the tool's terms for commercial use.
Training-data concerns	Artists and publishers have raised concerns and lawsuits about training on their work. Avoid prompting "in the style of [living artist]" for commercial work.
Real people	Don't create images, voices or videos of real people without consent — especially anything misleading, sexual, defamatory or political.
Deepfakes	Realistic fake media can spread misinformation and enable fraud and harassment. Never make or share deceptive deepfakes; report abusive ones.
Labelling	Label AI-generated or AI-altered media where it could mislead ("Image generated with AI"). Many platforms require it; some tools add invisible watermarks or content credentials.

7.8 Spotting AI-generated media

- Look for odd details: hands and fingers, jewellery, text, reflections, backgrounds, ears, teeth.
- Unnatural smoothness or lighting; inconsistent shadows.
- In video: blinking, lip-sync, strange motion, flickering edges.
- In audio: flat emotion, odd breathing, unusual pronunciation.
- Check the source: reverse image search; trusted news outlets; content credentials where available.
- Remember that detection is getting harder — **verify the source, not just the pixels.**

Module 7 — Summary

- Text-to-image uses diffusion: noise + prompt guidance → image; seeds reproduce results.
- Image prompt formula: subject, details, setting, style, lighting, composition, mood, format.
- AI editing: inpainting, outpainting, background removal, upscaling, style transfer.
- Audio: TTS, speech-to-text, music; voice cloning only with consent — beware voice scams.
- Video: text/image-to-video, editing, avatars, dubbing — short clips, improving fast.
- Respect copyright and consent; never make deceptive deepfakes; label AI media.

Practice questions — Module 7

1. Explain in five steps how a text-to-image model creates an image.
2. Write a detailed image prompt for a poster about saving electricity at school.
3. What are inpainting and outpainting? Give a use for each.
4. Give three good uses and one danger of voice cloning.
5. Write a prompt for a 10-second video clip of a farmer using a solar pump.
6. What is a deepfake? How can you check whether a viral video is real?
7. List four good practices for using AI images in a business.
8. Plan an AI-assisted 60-second video for a local shop in five steps.

MODULE 08

Building AI Chatbots and Applications

From idea to working app: no-code options, the parts of an AI application, calling a model through an API, system prompts, conversation memory, grounding answers in your documents, tools and agents, costs, safety and testing.

8.1 Three ways to build with AI

Approach	Skills needed	Examples	Best for
No-code custom assistants	Writing instructions; uploading files	Custom GPT-style assistants, "projects" with instructions and files inside AI chat apps	Personal or team helpers, quick prototypes
Low-code / automation platforms	Connecting blocks and apps	Chatbot builders, workflow tools that connect forms, email, sheets and AI	Business workflows: auto-reply to enquiries, summarise forms
Code with an API	Basic programming (Python/JavaScript)	Your own website chatbot, study app, document assistant	Full control, integration with your data and products

Key idea — start simple

Most useful AI features are a **single well-written prompt plus the right data**. Build the simplest version first, test it with real users, then add memory, documents, tools or agents only when you need them.

8.2 The anatomy of an AI chatbot

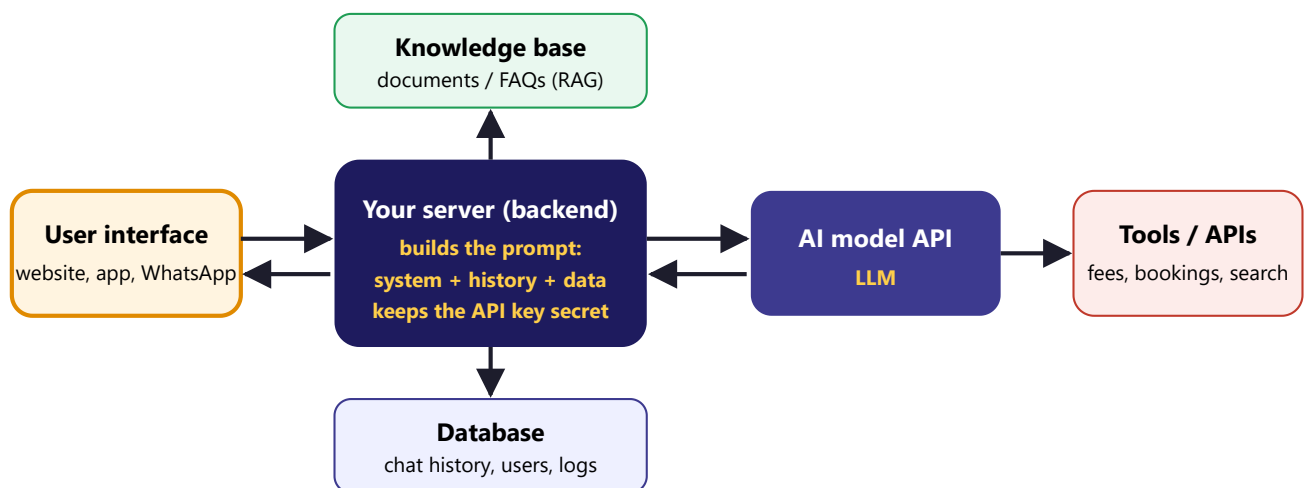


Figure 8.1 — Parts of an AI chatbot application.

- **Interface** — a chat window on a website or app, a WhatsApp/Telegram bot, a voice interface.
- **Backend** — your server code: receives the message, assembles the full prompt, calls the model, checks the reply, and returns it. The API key lives **only** here.
- **Model** — the LLM, reached through a provider's API (or a self-hosted open-weight model).
- **Knowledge** — your documents, FAQs and data for grounded answers (RAG).
- **Tools** — functions the model can ask your code to run: look up fees, book an appointment, check order status.
- **Storage** — conversation history, user accounts, logs for improvement.

8.3 APIs — how apps talk to AI models

An **API** (application programming interface) lets your program send a request to the AI provider's servers and receive the model's reply. A typical request contains:

Part	Purpose
API key	Your secret credential — identifies and bills your account
Model	Which model to use
System prompt	Persistent instructions: role, rules, tone, limits
Messages	The conversation so far: alternating <code>user</code> and <code>assistant</code> turns
max_tokens	Upper limit on the reply length

Caution — protect your API key

Never put an API key in front-end JavaScript, a mobile app, a public GitHub repository or a screenshot. Anyone who finds it can run up charges on your account. Store it as an environment variable on your server and rotate (replace) it immediately if it leaks.

8.4 A first chatbot in Python

The example below uses Anthropic's Claude API with the official Python SDK; other providers' SDKs follow the same idea (model + system prompt + messages → reply). Install with `pip install anthropic` and set the `ANTHROPIC_API_KEY` environment variable.

```
# tutor_bot.py — a minimal study-helper chatbot in the terminal
import anthropic

client = anthropic.Anthropic()          # reads the API key from the environment

SYSTEM = (
    "You are Spark, a friendly tutor for Class 9-12 students in India. "
    "Explain step by step in simple English. For homework, give hints first. "
    "If you are not sure of a fact, say so."
)

history = []                            # the conversation so far

while True:
    question = input("You: ")
    if question.lower() in ("quit", "exit"):
        break
    history.append({"role": "user", "content": question})

    response = client.messages.create(
        model="claude-opus-5",
        max_tokens=16000,
        system=SYSTEM,
        messages=history,
    )

    # the reply is a list of content blocks; print the text ones
    reply = "".join(block.text for block in response.content if block.type == "text")
    print("Spark:", reply)

    # keep the assistant's turn so the next question has context
    history.append({"role": "assistant", "content": response.content})
```

- The model is **stateless**: it remembers the conversation only because the program sends the whole `history` each time (Module 4 — context window).
- The **system prompt** defines the bot's personality and rules for every turn.
- Printing only `text` blocks matters because replies can also contain other block types (for example, tool requests).
- For a chat user interface, use **streaming** so words appear as they are generated instead of after a long wait.

8.5 Writing a strong system prompt for an app

System prompt checklist

- **Identity and purpose**: who the assistant is and whom it serves.
- **Knowledge boundaries**: what it may answer from (e.g. "only the provided documents").
- **Style**: tone, length, language(s), formatting.
- **Process**: e.g. ask clarifying questions; give hints before answers.
- **Limits and safety**: topics to decline, when to hand over to a human, never collect sensitive data.
- **Honesty**: admit uncertainty; don't invent policies, prices or phone numbers.

Worked example 8.1 — system prompt for a coaching centre's website bot

```
You are the website assistant for Sunrise Tuition Centre, Cuttack.
Answer questions about courses, batch timings, fees, admissions and location
using ONLY the information in <centre_info> below.
- Be brief (under 80 words), friendly and clear. Reply in the user's language
  (English, Hindi or Odia).
- If the answer is not in <centre_info>, say you don't know and share the
  phone number +91-XXXXXXXXXX. Never guess fees or dates.
- To book a demo class, ask for name, class and phone, then confirm that the
  office will call back within one working day.
- Do not give medical, legal or personal advice. Do not ask for Aadhaar,
  bank or card details.
<centre_info> ... courses, fees, timings, address ... </centre_info>
```

8.6 Memory: short-term and long-term

Type	How it works	Watch out for
------	--------------	---------------

Short-term (conversation)	Send recent messages with each request	Context-window limits and cost as chats grow — trim or summarise old turns
Summarised memory	Periodically ask the model to summarise the chat; keep the summary instead of all messages	Details can be lost
Long-term (user profile)	Store facts in a database (e.g. "student is in Class 11, weak in organic chemistry") and add them to the prompt	Privacy — tell users what is stored; let them delete it

8.7 Grounding answers in your documents (RAG)

To make a bot answer from your prospectus, policies or notes — and not from the model's general memory — use **retrieval-augmented generation** (Module 4.10):

1. **Prepare**: split documents into chunks (a few paragraphs each).
2. **Index**: compute an **embedding** for each chunk and store it in a **vector database** (or a simple search index for small collections).
3. **Retrieve**: for each question, find the most similar chunks.
4. **Generate**: put those chunks in the prompt with the instruction "answer only from these passages; cite the section".

For small collections (a few documents), simply including the whole text in the prompt often works — modern context windows are large. Use full RAG when documents are too many or too large, or change often. Many AI platforms also let you upload files directly and handle retrieval for you.

8.8 Tools, function calling and agents

With **tool use** (function calling), you describe functions your code can run — name, purpose and parameters. When the model decides a tool is needed, it replies with a **tool request** instead of text; your code runs the function and sends the result back; the model then writes the final answer.

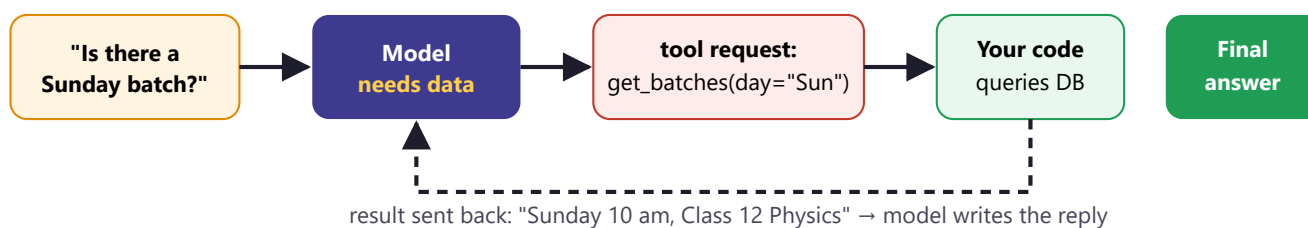


Figure 8.2 — The tool-use loop.

An **agent** repeats this loop on its own — planning steps, calling several tools, checking results — until a larger task is done (e.g. "research three solar vendors, compare quotes and draft an email"). Agents are powerful but cost more, take longer and can make compounding mistakes, so use them only when a task truly needs open-ended, multi-step work, and keep humans in the loop for important actions (payments, sending messages, deleting data).

8.9 Costs and performance

- API pricing is usually **per million tokens**, with separate prices for input and output tokens; larger models cost more.
- Estimate: requests per day × (input + output tokens) × price. Long system prompts and chat histories are resent every turn — they dominate cost.
- Save money with **prompt caching** (providers discount repeated prompt prefixes), trimming history, shorter outputs, smaller models for simple tasks, and batch processing for non-urgent jobs.
- Latency: streaming improves perceived speed; smaller models respond faster.

Worked example 8.2 — rough monthly cost estimate

A website bot handles 300 chats/day, each averaging 4 turns. Each turn sends ~2,000 input tokens (system prompt + history + question) and receives ~200 output tokens. Monthly tokens $\approx 300 \times 4 \times 30 = 36,000$ turns $\rightarrow$ 72 million input and 7.2 million output tokens. Multiply by your model's per-million prices (check the provider's current pricing page). Prompt caching of the fixed system prompt can reduce the input part substantially.

8.10 Safety, security and privacy in AI apps

Risk	What it is	Protection
------	------------	------------

Prompt injection	User messages, web pages or documents containing instructions like "ignore your rules and..."	Keep rules in the system prompt; treat retrieved content as data, not instructions; limit what tools can do; require confirmation for risky actions
Data leakage	Bot reveals private data or its hidden prompt	Don't put secrets in prompts; give the model only the data this user may see
Hallucinated answers	Invented facts, policies, medical advice	Ground in documents; instruct "say you don't know"; show sources; human escalation
Harmful content	Abusive or unsafe outputs	Provider safety features, moderation checks, clear usage policy
Personal data	Collecting more than needed	Collect minimum data; tell users; follow India's DPDP Act; secure storage; deletion on request
Cost abuse	Bots spamming your endpoint	Rate limits, login, CAPTCHA, spending caps

8.11 Testing and improving your bot

1. Create an **evaluation set**: 30–100 realistic questions with the correct/acceptable answers, including tricky ones (out-of-scope, rude, multilingual, ambiguous).
2. Run the bot on all of them after every prompt or model change; score accuracy, tone and safety.
3. Review real conversations (with consent and privacy safeguards) to find failures.
4. Add a **thumbs up/down** button and a "talk to a human" option.
5. Change one thing at a time and re-test (Module 5.12).

8.12 Deployment options

- **Website widget**: a chat bubble that calls your backend (e.g. a small Python/Node.js server or serverless function).
- **Messaging apps**: WhatsApp Business API, Telegram bots — connect via a business provider and your backend.
- **Internal tools**: Slack/Teams bots, spreadsheet add-ons.
- **Voice**: speech-to-text → LLM → text-to-speech for phone or smart-speaker assistants.

Module 8 — Summary

- Build with no-code assistants, low-code automation, or code + API — start simple.
- An AI app = interface + backend + model API + knowledge + tools + storage.
- API requests send model, system prompt, messages and settings; keep API keys secret on the server.
- Models are stateless — send history each turn; manage memory and privacy.
- RAG grounds answers in your documents; tool use lets the model act via your code; agents loop over tools.
- Control cost (tokens, caching, model choice); defend against prompt injection and hallucination; test with an evaluation set.

Practice questions — Module 8

1. Compare no-code, low-code and code-based approaches to building a chatbot.
2. Draw and label the parts of an AI chatbot application.
3. Why must an API key never be placed in front-end code?
4. In the Python example, why is the whole history sent with every request?
5. Write a system prompt for a clinic's appointment bot, including safety rules.
6. Explain RAG in four steps. When is including the whole document enough?
7. Describe the tool-use loop with an example of your own.
8. What is prompt injection? Give two defences.
9. Design a 10-question evaluation set for a school admission chatbot.

Responsible AI, Ethics, Privacy, and Limitations

Bias and fairness, hallucination, copyright and intellectual property, privacy and data protection, deepfakes and misinformation, safety, jobs and the environment — and a practical code of conduct for using generative AI responsibly.

9.1 Why responsible AI matters

Generative AI is powerful and widely available. The same tool that helps a student understand physics can write fake reviews, impersonate a voice or leak private data. Using AI responsibly protects people, organisations and trust in information itself. The main principles recognised by governments, companies and researchers worldwide are:

Principle	Meaning in practice
Fairness	AI should not discriminate against people based on gender, caste, religion, region, language, disability or other characteristics
Transparency	People should know when they are dealing with AI and how important decisions are made
Accountability	A human or organisation remains responsible for AI-assisted decisions and content
Privacy	Personal data is collected minimally, protected and used only for agreed purposes
Safety & reliability	AI works as intended, is tested, and fails safely
Human oversight	People can review, override and correct AI, especially for high-stakes decisions
Beneficence	AI is used to help people and society, not to deceive or harm

9.2 Bias and fairness

AI learns from data created by people — and data reflects society's historical biases and gaps. Bias can enter at many points:

- **Training data:** under-representation (few examples from rural areas or Indian languages) or historical discrimination (past hiring decisions).
- **Labels:** human annotators' assumptions.
- **Design choices:** what the system optimises for.

- **Use context:** a model built for one population applied to another.

Examples of AI bias

- An image generator asked for "a doctor" mostly shows men; "a nurse" mostly women.
- A résumé-screening model trained on past hires favours candidates similar to those previously hired.
- Speech recognition works worse for some accents or languages.
- A chatbot gives lower-quality answers in less-represented languages.

Reducing bias as a user

- Be specific about diversity in image prompts.
- Ask the AI to check its answer for stereotypes or one-sided views.
- Test outputs across different groups, names and languages before relying on them.
- Never use AI alone for decisions about people — hiring, admissions, loans, discipline.

9.3 Hallucination and over-reliance

As covered in Modules 4 and 6, LLMs can produce fluent falsehoods. The bigger danger is **automation bias** — people trusting a confident machine more than they should.

Stakes	Examples	How to use AI
Low	Brainstorming, drafts, practice questions	Use freely; skim for errors
Medium	Reports, study notes, customer replies	Verify facts and numbers; human edit
High	Medical, legal, financial, safety, exams, news	AI as an assistant only; qualified human decides; check authoritative sources

Caution — health, law and money

AI can explain concepts and help you prepare questions, but it is not a doctor, lawyer or financial adviser. Always consult qualified professionals for personal decisions.

9.4 Copyright and intellectual property

- **Training data:** models are trained on large amounts of text and images, some copyrighted. Whether this is lawful is being debated and litigated in several countries; rules may change.
- **Ownership of outputs:** in many jurisdictions, purely machine-generated works may receive limited or no copyright protection; significant human creative input strengthens a claim. India's position is still developing — check current guidance for commercial work.
- **Infringing outputs:** AI can sometimes reproduce copyrighted text, characters, logos or styles closely. You are responsible for what you publish.
- **Tool terms of use:** each service sets rules on commercial use of outputs; read them.

Good practice

- Use AI to create original work; don't ask it to reproduce books, songs, articles or characters.
- Don't imitate living artists or brands for commercial work.
- Cite sources and disclose AI assistance where required.
- Keep records of your prompts and edits for important creative work.

9.5 Privacy and data protection

What you type into an AI tool may be stored, reviewed by the provider for safety, or (depending on settings and plan) used to improve models. Treat public AI chatbots like any external service.

Never paste into public AI tools

- Passwords, OTPs, API keys, bank or card numbers
 - Aadhaar, PAN, passport numbers
 - Patients' medical records, students' personal details, customers' data
 - Confidential business documents, unpublished exam papers, source code under NDA
-
- **Anonymise:** replace names and numbers ("Student A", "Company X") before asking for help.
 - **Check settings:** many tools let you turn off chat history or training on your data; business/enterprise plans usually offer stronger data protections.
 - **Follow your organisation's AI policy** — schools, hospitals, banks and companies increasingly have one.

- **India's Digital Personal Data Protection (DPDP) Act, 2023** requires organisations to process personal data lawfully, for specific purposes, with consent (or other legal grounds), and to protect it. AI projects that handle personal data must follow it.

9.6 Deepfakes, misinformation and fraud

Generative AI makes it easy to produce realistic fake text, images, audio and video. Harms include:

- **Misinformation:** fake news images, fabricated quotes, fake election content spreading on social media.
- **Fraud:** cloned voices asking relatives for money; fake video calls of "officials" or executives; AI-written phishing messages.
- **Harassment:** non-consensual fake images of real people — deeply harmful and illegal in many places.
- **Fake reviews and spam:** mass-produced content that misleads buyers.

Protect yourself and others

- Pause before sharing sensational content; check trusted news sources and fact-checkers.
- Verify urgent money requests by calling back on a known number; agree a family "safe word".
- Look for content labels and provenance information; use reverse image search.
- Report deepfakes and fraud on the platform and at **cybercrime.gov.in** or helpline **1930** in India.
- Never create deceptive or non-consensual content of real people.

9.7 Safety and misuse

AI developers train models to refuse clearly harmful requests (weapons, serious crime, self-harm instructions, abuse) and add safety filters. People sometimes try "jailbreaks" — tricky prompts to bypass these protections. Attempting to misuse AI for harm is unethical, often illegal, and usually violates the tool's terms. If you encounter harmful outputs, report them to the provider. Organisations deploying AI should add their own safeguards: content moderation, logging, rate limits and human escalation (Module 8.10).

9.8 Academic and workplace integrity

Usually acceptable (check local rules)	Usually not acceptable
Explaining concepts; practice questions; feedback on your draft; grammar checks; brainstorming; study plans	Submitting AI-written assignments as your own; using AI during closed-book exams; fabricating data or references; hiding AI use where disclosure is required

At work, disclose AI assistance where it matters (client deliverables, published content), check outputs, and never let AI make final decisions that affect people without human review.

9.9 Jobs and the future of work

AI changes **tasks** more than whole jobs. Routine writing, summarising, data entry and first-draft coding are increasingly automated; new tasks appear in prompting, reviewing, integrating and supervising AI, plus entirely new roles. People who combine domain knowledge with AI skills are in growing demand.

- **Skills that grow in value:** judgement, verification, communication, creativity, domain expertise, ethics, working with people.
- **Adapt:** learn to use AI tools in your field; focus on what humans do best; keep learning.
- Policy questions — retraining, fair transitions, access for all — are debated by governments and industry.

9.10 Environmental impact

Training and running large AI models uses significant electricity and water for data-centre cooling. Efficiency is improving and many data centres are moving to renewable energy, but responsible use still matters: use smaller models when sufficient, avoid unnecessary repeated generations, and prefer efficient providers.

9.11 AI regulation — a quick overview

- **India:** an approach centred on enabling innovation with safeguards; the DPDP Act covers personal data; the IT Act and rules address unlawful content and deepfakes on platforms; government advisories and the IndiaAI Mission promote responsible AI development. Rules are evolving — follow official updates.
- **European Union:** the **EU AI Act** classifies AI systems by risk (banned, high-risk, limited, minimal) with obligations such as transparency for AI-generated content.
- **Other countries** use a mix of laws, executive guidance and voluntary standards.

- **Organisations** increasingly adopt internal AI policies covering approved tools, data rules, disclosure and review.

9.12 A personal code of conduct for generative AI

The responsible AI user's checklist

1. **I verify** important facts, numbers and sources before using or sharing them.
2. **I protect privacy** — mine and others' — and never paste sensitive data into public tools.
3. **I stay accountable**: AI assists; I decide and take responsibility.
4. **I am honest** about AI use where it matters and never pass off AI work dishonestly.
5. **I respect creators** and don't use AI to copy others' work or identity.
6. **I don't deceive** — no deepfakes, fake reviews or impersonation.
7. **I watch for bias** and keep humans in decisions about people.
8. **I use AI to learn**, not to avoid learning.

Module 9 — Summary

- Principles: fairness, transparency, accountability, privacy, safety, human oversight, beneficence.
- Bias comes from data and design — check outputs and keep humans in decisions about people.
- Match verification effort to the stakes; AI is not a doctor, lawyer or financial adviser.
- Copyright rules are evolving; create original work and read tool terms.
- Never paste sensitive data into public AI tools; follow the DPDP Act and organisational policies.
- Deepfakes and AI fraud are real — verify, don't spread, report (1930, cybercrime.gov.in).
- AI changes tasks and jobs; skills in judgement and verification grow in value.

Practice questions — Module 9

1. List seven principles of responsible AI and explain two in detail.
2. Give three examples of AI bias and one way to reduce each.
3. What is automation bias? Why is it dangerous in healthcare?
4. List six types of information you should never paste into a public chatbot.
5. Explain the copyright questions raised by generative AI.
6. How would you protect your family from AI voice-cloning scams?
7. Write an AI-use policy of eight rules for a school.
8. Explain how AI is likely to change jobs, with examples from one industry.

MODULE 10

Practical Projects and Final Revision

Four guided projects — an AI writing assistant, a prompt-engineered image set, a mini chatbot and an end-to-end AI mini-app — with step-by-step instructions, evaluation rubrics, and a one-page revision of the course.

10.1 How to approach an AI project



Figure 10.1 — The project cycle used in all four projects.

10.2 Project 1 — AI writing assistant

Goal: build a reusable set of prompts (a "prompt kit") that helps a real person with recurring writing — for example a shopkeeper's customer messages, a teacher's parent notices, or your own study summaries.

1. **Choose a user and 3 recurring tasks** (e.g. a coaching centre: admission replies, fee reminders, result announcements).
2. **Write a system/instruction block** describing the organisation, tone, languages and rules (Module 5).
3. **Create one template prompt per task** with blanks: *[student name]*, *[due date]*, *[amount]*.
4. **Add 2 examples** of ideal outputs to each template (few-shot).
5. **Test** each template on 5 realistic cases including awkward ones (angry parent, missing information).
6. **Refine** and record what changed.
7. **Package** it: a one-page guide or a no-code custom assistant with the instructions and examples built in.

Rubric criterion	Excellent (4)	Needs work (1)
Clarity of prompts	Context, task, format, tone all specified	Vague one-liners
Output quality	Ready to send after minor edits	Needs rewriting

Handles edge cases	Asks for missing info; polite with complaints	Invents details
Documentation	Clear guide with examples and before/after	No explanation

10.3 Project 2 — prompt-engineered image set

Goal: create a consistent set of 6 images for a real purpose — a poster series, a social-media campaign, or illustrations for a lesson.

1. Define the purpose, audience and a **style guide**: medium (e.g. flat vector), palette (3–4 colours), lighting, mood, aspect ratio.
2. Write a **base prompt** containing the style guide; vary only the subject for each image.
3. Generate several options per image; choose the best; iterate with specific edits.
4. Use editing (inpainting/outpainting) to fix details; add text in a design tool.
5. Check each image for errors (hands, text), bias and representation.
6. Label images as AI-generated where appropriate and record prompts used.

Example base prompt

"Flat vector illustration, clean lines, palette of navy #1E1B5E, orange #E08A00 and cream, soft shadows, friendly educational style, plenty of empty space at the top for a headline, 4:5 portrait. Subject: [a girl measuring the voltage of a battery with a multimeter in a school lab]."

10.4 Project 3 — mini chatbot

Goal: build a question-answering bot for a small organisation that answers **only** from its own information.

1. Collect the organisation's information into one clean document: services, timings, fees, FAQs, contacts, policies.
2. Write the system prompt (Module 8.5) with the document inside tags and a strict "answer only from this; otherwise say you don't know and give the phone number" rule.
3. Build it — either as a **no-code assistant** (instructions + uploaded file) or with the **Python example** in Module 8.4.
4. Create a **test set of 25 questions**: 15 answerable, 5 not in the document, 3 in Hindi/Odia, 2 attempts to make it break rules.
5. Score answers: correct / partially correct / wrong / appropriately refused.

6. Fix failures by improving the document or prompt; re-test.

Metric	Target
Correct answers on answerable questions	≥ 90%
Correctly refuses/redirects unknown questions	100% — never invents fees or dates
Responds in the user's language	All multilingual tests
Resists rule-breaking prompts	All attempts

10.5 Project 4 — capstone: end-to-end AI mini-app

Goal: solve a real problem with a small AI-powered application or workflow, and document it professionally.

Idea bank

Idea	AI components
Study buddy that turns chapter notes into flashcards and quizzes	Summarisation, question generation, structured output (JSON → quiz)
Farmer's crop-advice assistant in Odia	Multilingual chat, grounding in agricultural advisories, image understanding (leaf photos)
Shop review analyser	Sentiment classification, theme extraction, summary report
Meeting-notes to action-items tool	Speech-to-text, summarisation, extraction
Resume and cover-letter helper	Rewriting, tailoring to job descriptions, bias checks
Local-language notice generator for a panchayat	Translation, simplification, text-to-speech for announcements

Deliverables

1. **Problem statement** — user, pain point, why AI helps.
2. **Design** — a diagram of components (interface, prompts, data, model, tools) and the prompt(s) used.
3. **Working prototype** — no-code, low-code or code.
4. **Evaluation** — test set, results table, and improvements made.
5. **Responsible-AI review** — privacy, bias, hallucination risks, and safeguards added.
6. **Cost and scaling note** — rough cost per use and how it would grow.

7. **Demo** — a 3–5 minute video or live walkthrough.

Worked example 10.1 — capstone summary: "Quiz-Maker for Class 12 Physics"

Problem: students need quick self-tests after each chapter. **Design:** the student pastes chapter notes → prompt generates 10 MCQs as JSON with answers and explanations → a simple web page displays the quiz and scores it. **Prompt:** role (physics teacher), difficulty mix (4 easy, 4 medium, 2 hard), JSON schema, "base questions only on the notes". **Evaluation:** a teacher reviewed 50 generated questions: 45 correct, 3 ambiguous, 2 wrong → added "each question must have exactly one correct option; include the sentence from the notes that supports the answer" → re-test: 49/50 correct. **Responsible AI:** no personal data collected; disclaimer "AI-generated — report errors"; teacher spot-checks. **Cost:** about one API call per quiz.

10.6 Course revision — one page

Module	Key points
1 AI basics	AI learns patterns from data; history (1950 Turing → 2012 deep learning → 2017 transformer → 2022 ChatGPT); narrow vs general; AI $\supset$ ML $\supset$ DL $\supset$ GenAI
2 ML & DL	Supervised / unsupervised / reinforcement / self-supervised; train–test split; overfitting; neurons, layers, weights; backpropagation and gradient descent; training vs inference vs fine-tuning
3 GenAI	Learns data distribution and samples; autoregressive transformers, VAEs, GANs, diffusion; foundation and multimodal models; open vs closed
4 LLMs	Tokens ($\sim\frac{3}{4}$ word), embeddings, self-attention; pre-training → instruction tuning → RLHF; context window; temperature/top-p; hallucination; RAG, tools, agents
5 Prompting	Context, role, task, format, tone (CRAFT); zero/few-shot; step-by-step; system prompts; structured output; grounding; iterate; chain; test
6 Text	Draft, summarise, translate, edit, tutor, code; templates; verify facts; academic integrity; disclose AI use
7 Media	Diffusion text-to-image; image prompt formula; inpainting/outpainting; TTS, voice cloning risks; text-to-video; deepfakes; labelling
8 Apps	Interface + backend + API + knowledge + tools; keep keys secret; stateless history; RAG; tool loop; cost; prompt injection; evaluation sets

9 Responsibility	Fairness, transparency, accountability, privacy, safety, oversight; bias; copyright; DPDP Act; deepfakes (1930); jobs; environment
10 Projects	Problem → criteria → prototype → test → improve → share with safeguards

Module 10 — Summary

- Every AI project: define the problem and success criteria, prototype with prompts, test on real examples, improve one change at a time, share with safeguards.
- Projects: writing assistant, consistent image set, grounded mini chatbot, end-to-end mini-app.
- Always include evaluation and a responsible-AI review.

Practice questions — Module 10

1. List the six steps of the AI project cycle.
2. Design a prompt kit of three templates for a pharmacy's customer messages.
3. Write a base style prompt for a series of six images about water conservation.
4. Create a 10-question test set for a library chatbot, including two out-of-scope questions.
5. Choose a capstone idea and write its problem statement and component diagram.
6. What should a responsible-AI review of your capstone include?

PLAYBOOKS

Generative AI at Work: Eight Profession Playbooks

Concrete ways people in different roles can use generative AI today — tasks, sample prompts and the cautions that matter most in each field.

P.1 Teachers

Task	Sample prompt
Lesson plans	"Create a 40-minute lesson plan on [topic] for Class [N]: objectives, a 5-minute hook activity, explanation, group activity, 5 exit-ticket questions."
Differentiated material	"Rewrite this explanation at three levels: struggling, average, advanced learners."
Question papers	"Make a 25-mark test on [chapter] with a blueprint: 5×1 MCQ, 5×2 short, 3×5 long; include the answer key and marking scheme."
Feedback	"Give constructive feedback on this student essay using the rubric; start with two strengths."
Parent communication	"Write a warm notice in English and Odia about the PTM on [date]."

Cautions

Check every question and answer key; don't paste students' personal data; keep assessment decisions with the teacher.

P.2 Students

Task	Sample prompt
Understanding	"Explain [concept] three ways: with an analogy, a diagram description and a worked example."
Practice	"Quiz me on [topic] one question at a time; tell me if I'm right and explain mistakes."
Revision	"Make a one-page cheat sheet of formulas and key terms from these notes."

Writing feedback	"Point out 5 ways to improve my essay without rewriting it."
Career exploration	"List 8 careers that combine biology and computers, with the entrance path for each in India."

Cautions

Follow your institution's rules; learn rather than copy; verify facts in textbooks for exams.

P.3 Shopkeepers and small businesses

Task	Sample prompt
Marketing	"Write 5 WhatsApp status messages for our Diwali sale on sweets, fun tone, under 20 words each, one in Hindi."
Product listings	"Write an online listing for [product]: title, 5 bullet features, 60-word description, care instructions."
Customer service	"Draft polite replies for these 3 common customer questions: [questions]."
Planning	"Suggest a simple weekly stock-check routine for a small grocery shop."
Accounts help	"Explain how to record this transaction in a simple cash book: [details]."

Cautions

Check prices, offers and GST details; don't make false claims; get tax and legal answers confirmed by a professional.

P.4 Healthcare administration

Task	Sample prompt
Patient information leaflets	"Write a simple leaflet on preparing for a fasting blood test, in English and Hindi, under 150 words." (reviewed by a doctor)
Front-desk FAQs	"Create FAQs about clinic timings, appointments and documents to bring, from this information: [info]."
Summaries of guidelines	"Summarise this public-health guideline into 8 key points for staff: [text]."

Scheduling templates "Design a weekly duty roster template for 6 nurses covering 3 shifts."

Cautions

No patient-identifiable data in public tools; AI must not diagnose or prescribe; all clinical content reviewed by qualified professionals.

P.5 Farmers and agriculture workers

Task	Sample prompt
Understanding advisories	"Explain this weather advisory in simple Odia and tell me what it means for paddy transplanting: [advisory]."
Crop problems	"This photo shows spots on my tomato leaves. What could it be, and what should I ask the local agriculture officer?" (with a photo)
Schemes	"Explain in simple words what documents are usually needed to apply for [scheme], and where to confirm them."
Selling	"Write a short message to local buyers offering 2 tonnes of organic turmeric with price and contact."

Cautions

Confirm pesticide use, doses and scheme rules with Krishi Vigyan Kendras, agriculture officers or official portals.

P.6 Engineers and technicians

Task	Sample prompt
Calculations with working	"Size a solar array for 12 kWh/day with 5 PSH and PR 0.77. Show each step." (then verify)
Code and scripts	"Write a Python/MATLAB script to plot this sensor data and flag readings above [limit]."
Documentation	"Turn these site-visit notes into a professional inspection report with findings and recommendations."
Troubleshooting	"List possible causes for [symptom] in [equipment], ordered from most to least likely, with checks for each."

Learning standards	"Explain the purpose of [standard] in simple terms and what to check in the official document."
--------------------	---

Cautions

Always verify calculations and safety-critical advice against standards, datasheets and qualified engineers.

P.7 Government and office staff

Task	Sample prompt
Drafting letters and notices	"Draft a formal office order about [subject] with references, in standard government letter format."
Summaries	"Summarise this 40-page report into a one-page note for the head of office: [text]."
Translation	"Translate this circular into simple Hindi and Odia for public display."
Citizen FAQs	"Create 10 FAQs for citizens about [service] using only this information: [info]."

Cautions

Follow official IT and data policies; never paste confidential files into public tools; verify legal references.

P.8 Content creators and marketers

Task	Sample prompt
Ideas	"Give 20 YouTube video ideas about [niche] for [audience], with a hook for each."
Scripts	"Write a 60-second Reel script: hook in 3 seconds, 3 tips, call to action."
Repurposing	"Turn this blog post into a LinkedIn post, 5 tweets and an email newsletter."
Thumbnails and visuals	Image prompts with bold composition and space for text (Module 7).
SEO	"Suggest a title, meta description and 5 subheadings for an article targeting [keyword]."

Cautions

Add your own experience and originality; disclose AI-generated media where platforms require; respect copyright and people's likeness.

Model Question Paper — Generative AI

Time: 2 hours · Maximum marks: 70 · Attempt all sections.

Section A — Very short answers ($10 \times 1 = 10$)

1. Expand LLM.
2. In which year was the Transformer architecture introduced?
3. Name the two networks in a GAN.
4. What does a low temperature setting do?
5. Expand RAG.
6. What is a token?
7. Name one technique to reduce hallucination.
8. What is inpainting?
9. Name India's personal data protection law.
10. Where should an API key be stored?

Section B — Short answers ($8 \times 3 = 24$)

11. Differentiate predictive and generative AI with examples.
12. Explain overfitting and two remedies.
13. Explain self-attention with an example sentence.
14. Differentiate zero-shot and few-shot prompting.
15. Write a strong prompt for summarising a report for a manager.
16. Explain how a diffusion model generates an image.
17. What is prompt injection? How can apps defend against it?
18. List six items you should never paste into a public chatbot.

Section C — Long answers ($4 \times 9 = 36$)

19. Explain the three stages of training an assistant LLM and why LLMs hallucinate.
20. Describe the architecture of an AI chatbot with RAG and tool use, with a diagram.
21. Discuss bias, privacy and deepfakes as challenges of generative AI, with safeguards.
22. Design an AI mini-app for a local problem: components, prompts, evaluation and responsible-AI review.

Section A answers: 1. Large Language Model · 2. 2017 · 3. Generator and discriminator · 4. Makes outputs more focused/consistent · 5. Retrieval-Augmented Generation · 6. A unit of text (word or part of a word) processed by the model · 7. Grounding answers in provided documents (RAG) / asking for sources · 8. Replacing part of an image with generated content · 9. Digital Personal Data Protection Act, 2023 · 10. Securely on the server (environment variable).

50 Ready-to-Use Prompts

Copy, fill in the [brackets], and adapt. Always review outputs before using them.

Study and learning

1. Explain [topic] to a Class [N] student in simple English with one everyday analogy and one worked example.
2. Act as a Socratic tutor. Ask me one question at a time to help me solve: [problem]. Don't give the answer unless I ask.
3. Make 10 MCQs on [chapter/topic] — 4 easy, 4 medium, 2 hard — with answers and one-line explanations.
4. Turn these notes into 15 flashcards (question | answer): [notes].
5. Create a 30-day study plan for [exam] on [date], covering [subjects], with weekly revision days.
6. Mark my answer out of [5] using this rubric: [rubric]. Explain what's missing. My answer: [text].
7. List the 10 most common mistakes students make in [topic] and how to avoid each.
8. Summarise this chapter in 8 bullet points and 3 key formulas: [text].
9. Compare [concept A] and [concept B] in a table with 6 rows.
10. Give me a memory trick (mnemonic) for [list to remember].

Writing and communication

11. Write a [formal/friendly] email to [recipient] about [purpose]. Key points: [..]. Under [N] words. End with [request].
12. Proofread this text. Fix only grammar and spelling; list each change: [text].
13. Rewrite this for a [reading level/audience], keeping all facts: [text].
14. Shorten this to [N] words without losing the main points: [text].
15. Give me 5 alternative titles for [article/talk] — informative, catchy and question-style.
16. Write a 2-minute speech on [topic] for [occasion], with an opening story and a strong ending.
17. Translate into simple [language] for [audience]. Keep numbers as digits. Then back-translate to English: [text].

18. Draft a polite reply to this complaint that apologises, explains the fix and invites them back: [complaint].
19. Turn these rough meeting notes into minutes with decisions and action items (owner, deadline): [notes].
20. Write a LinkedIn post (under 150 words) about [achievement], humble and specific, with 3 hashtags.

Work and business

21. Write a 60-word product description for [product] highlighting [features] for [customer type].
22. Create 7 social-media post ideas for [business] for the coming week, one per day, with captions.
23. Analyse these customer reviews: list top 3 praises, top 3 complaints, and 3 recommended actions: [reviews].
24. Write a job post for [role]: responsibilities, required skills, hours, location, how to apply.
25. Create a one-page business plan outline for [idea] in [town].
26. Draft an FAQ with 10 questions and answers for [service], using only this information: [info].
27. Write an Excel formula that [task], assuming data in [cells]. Explain how it works.
28. Turn these monthly figures into a short report with trends and 3 recommendations: [data].
29. Create a 10-slide presentation outline on [topic] for [audience], 3 bullets per slide.
30. Role-play a difficult customer so I can practise handling [situation]. Give feedback afterwards.

Coding and data

31. Write a Python function that [task]. Include comments and 3 test cases.
32. Explain this code line by line for a beginner: [code].
33. Here is my code and the full error message. Explain the cause and give the fixed code: [code + error].
34. Convert this [language A] code to [language B]: [code].
35. Write a SQL query to [task] from tables [schema].
36. Suggest how to clean this messy data and show the cleaned result as a table: [sample].
37. Extract [fields] from this text and return only JSON with keys [keys]: [text].
38. Review this code for bugs, security issues and readability; list issues by severity: [code].
39. Write a regular expression that matches [pattern] and explain each part.

40. Design the database tables for [app] with columns and relationships.

Images, audio and video

41. [Style] illustration of [subject] doing [action] in [setting], [lighting], [palette], [composition], [aspect ratio].
42. Photorealistic [shot type] of [subject] in [location], [time of day], [mood], shallow depth of field, [ratio].
43. Create a poster background for [event] with empty space at the top for text, [style], [colours].
44. Write a 60-second video script with 6 scenes, visuals, voice-over and on-screen text for [topic].
45. Describe a 10-second video: [subject/action], camera [movement], [style], [lighting].
46. Instrumental background music: [genre], [mood], [tempo], [instruments], [length], no vocals.
47. Write alt text (under 125 characters) for this image: [image].
48. Suggest 5 visual concepts for a campaign about [topic] for [audience].
49. Write a voice-over script in [language] for [audience], warm tone, 150 words.
50. List every factual claim in your answer, rate your confidence, and tell me which to verify and where.

15 Hands-On Exercises

Use any AI assistant (free versions are fine). Record your prompts and outputs in a notebook.

Lab 1 — AI around you

List 10 apps you use and how each uses AI. Classify each as predictive or generative.

Lab 2 — Tokens

Use a tokenizer demo tool to count tokens for the same sentence in English, Hindi and Odia. Compare and explain the difference.

Lab 3 — Temperature

If your tool allows it, ask for a product slogan at low and high temperature (or ask for "very predictable" vs "very creative" versions). Compare variety and quality.

Lab 4 — Hallucination hunt

Ask about an obscure local topic and request sources. Check each claim and source. Record what was right, wrong or invented.

Lab 5 — Weak vs strong prompts

Write a one-line prompt and a CRAFT prompt for the same task. Score both outputs on a 5-point rubric.

Lab 6 — Few-shot classifier

Build a few-shot prompt that classifies 20 real messages into categories. Measure accuracy against your own labels.

Lab 7 — Structured output

Extract name, date, amount and category from 10 receipts or invoices into JSON; paste into a spreadsheet.

Lab 8 — Summarise and verify

Summarise a 3-page article in 5 bullets. Check every number against the original.

Lab 9 — Translate and back-translate

Translate a school notice into an Indian language, back-translate it, and fix meaning errors with a fluent speaker.

Lab 10 — Socratic tutoring

Solve a maths or physics problem with the AI acting as a Socratic tutor. Reflect: did you learn more than from a direct answer?

Lab 11 — Image prompt iteration

Create one image in 5 iterations, changing one element each time (style, lighting, composition...). Explain each effect.

Lab 12 — Bias check

Generate images or short stories for "an engineer", "a nurse", "a farmer", "a CEO". Note stereotypes; rewrite prompts to be inclusive.

Lab 13 — Custom assistant

Create a no-code assistant with instructions and a document for a real organisation; test it with 15 questions.

Lab 14 — Deepfake awareness

Find two debunked fake images or videos from fact-checking sites. List the clues and how they were verified.

Lab 15 — Policy

Write an 8-rule AI-use policy for your class, college or workplace and discuss it with others.

BONUS

Tools Guide and Careers in Generative AI

Categories of AI tools and how to choose them, and the career paths that generative AI skills open up.

B.1 Categories of generative AI tools

Products change quickly, so learn the **categories** and evaluate current tools against your needs.

Category	What they do	What to compare
General AI assistants (chat)	Writing, answering, analysis, coding, images, files, voice	Quality in your tasks and languages, file/image support, privacy settings, price
AI in office suites	Draft documents, summarise email, analyse spreadsheets, create slides	Integration with tools you already use; data protection for work
AI search / research assistants	Answers with web citations	Source quality, citation accuracy
Coding assistants	Code completion, explanation, agents that edit projects	Supported languages, security, review workflow
Image generators and editors	Text-to-image, editing, design templates	Style range, text rendering, commercial licence, safety
Voice and audio	TTS, transcription, dubbing, music	Indian-language support, voice consent policies, licence
Video tools	Text/image-to-video, editing, avatars	Clip length, quality, cost, watermarking
Automation / agents	Connect AI to email, forms, sheets and apps	Reliability, human approval steps, data access
Developer platforms & APIs	Build your own AI apps	Model quality, price per token, rate limits, documentation, data policy
Indian-language initiatives	Translation and speech for Indian languages (e.g. government and research projects)	Language coverage, accuracy

How to choose a tool — five questions

1. Does it do my task well on **my own** real examples?
2. How does it handle my **data** (storage, training, deletion)?
3. What does it **cost** at my usage level?
4. Can I use outputs **commercially**?
5. Does it support my **languages** and devices?

B.2 Careers and roles

Role	What you do	Skills to build
AI-enabled professional (any field)	Use AI to work faster and better as a teacher, marketer, analyst, engineer, lawyer, designer	Prompting, verification, domain expertise
Prompt / AI content specialist	Design prompts, templates and content pipelines	Writing, testing, evaluation
AI trainer / annotator / evaluator	Rate and improve model outputs; build evaluation sets	Attention to detail, subject knowledge, languages
AI application developer	Build chatbots, RAG systems, agents	Python/JavaScript, APIs, databases, cloud
Machine-learning engineer	Train, fine-tune and deploy models	Maths (linear algebra, probability), PyTorch, MLOps
Data scientist / analyst	Analyse data with AI assistance	Statistics, SQL, Python, visualisation
AI product manager	Decide what AI features to build and how to measure them	User research, evaluation, business sense
Responsible-AI / policy specialist	Governance, risk, compliance, ethics	Law, policy, ethics, communication
AI educator / trainer	Teach AI skills in schools, companies, communities	Teaching, clear explanation, practical projects

B.3 A learning roadmap

1. **Month 1:** daily practice with an AI assistant; master prompting (Modules 5–6); build your prompt kit.
2. **Month 2:** images, audio and video tools; responsible-AI habits; complete Projects 1–2.
3. **Month 3:** learn basic Python; call an AI API; build the mini chatbot (Project 3).
4. **Months 4–6:** RAG, tool use, evaluation; capstone project; portfolio on GitHub/LinkedIn.
5. **Beyond:** machine-learning foundations (maths, PyTorch) if you want to build models, or deepen AI use in your own profession.

REVISION

100 Multiple-Choice Questions

Test yourself on all ten modules. The answer key follows the questions.

1. Modern AI systems mainly work by:
(a) following hand-written rules only (b) learning patterns from data (c) copying the internet (d) random guessing
2. The term "artificial intelligence" was coined at the:
(a) 1956 Dartmouth workshop (b) 2012 ImageNet challenge (c) 1997 chess match (d) 2022 ChatGPT launch
3. The Turing Test was proposed by:
(a) John McCarthy (b) Alan Turing (c) Geoffrey Hinton (d) Ada Lovelace
4. The Transformer architecture was introduced in:
(a) 1998 (b) 2012 (c) 2017 (d) 2022
5. ChatGPT was released publicly in:
(a) 2015 (b) 2019 (c) November 2022 (d) 2024
6. All AI in use today is:
(a) general AI (b) narrow AI (c) superintelligence (d) conscious AI
7. Which is the correct nesting?
(a) ML $\supset$ AI $\supset$ DL (b) AI $\supset$ ML $\supset$ DL $\supset$ GenAI (c) DL $\supset$ AI $\supset$ ML (d) GenAI $\supset$ AI
8. "Is this email spam?" is a task for:
(a) generative AI (b) predictive (discriminative) AI (c) robotics (d) speech synthesis
9. "Write a poem about the monsoon" is a task for:
(a) predictive AI (b) generative AI (c) clustering (d) regression
10. The three ingredients that drove modern AI are data, algorithms and:
(a) compute power (b) printers (c) social media (d) keyboards
11. Learning from labelled examples is:
(a) unsupervised learning (b) supervised learning (c) reinforcement learning (d) clustering
12. Grouping customers without labels is:
(a) classification (b) clustering (unsupervised) (c) regression (d) reinforcement
13. Learning from rewards and penalties is:
(a) supervised (b) unsupervised (c) reinforcement learning (d) transfer
14. Hiding the next word and predicting it is:
(a) self-supervised learning (b) reinforcement learning (c) manual labelling (d) clustering
15. A model that memorises training data but fails on new data is:
(a) underfitting (b) overfitting (c) generalising (d) regularised

16. Testing a model should use data that is:
(a) the training data (b) unseen by the model (c) random noise (d) deleted
17. For rare events like fraud, accuracy alone is:
(a) the best metric (b) misleading; check precision and recall (c) always 100% (d) irrelevant
18. An artificial neuron computes:
(a) a weighted sum plus bias, then an activation (b) only a sum (c) a random value (d) a sorting step
19. The most widely used activation function is:
(a) ReLU (b) cosine (c) factorial (d) modulo
20. Softmax converts scores into:
(a) integers (b) probabilities that sum to 1 (c) negative numbers (d) images
21. Backpropagation calculates:
(a) the gradients of the loss for each weight (b) the dataset size (c) the tokens (d) the prompt
22. Gradient descent adjusts weights to:
(a) increase the loss (b) reduce the loss (c) freeze the model (d) add data
23. Deep learning uses networks with:
(a) one layer (b) many hidden layers (c) no weights (d) only rules
24. Using a trained model to produce outputs is called:
(a) training (b) inference (c) labelling (d) backpropagation
25. Further training a pre-trained model on specific data is:
(a) pre-training (b) fine-tuning (c) tokenizing (d) prompting
26. A generative model learns:
(a) only decision boundaries (b) the distribution of the data (c) nothing (d) only labels
27. Autoregressive models generate text:
(a) all at once (b) one token at a time (c) backwards only (d) as images
28. A GAN consists of:
(a) encoder and decoder (b) generator and discriminator (c) tokenizer and embedder (d) retriever and ranker
29. Diffusion models generate images by:
(a) removing noise step by step (b) copying images (c) sorting pixels (d) compressing text
30. A VAE compresses data into a:
(a) latent space (b) spreadsheet (c) database (d) token list
31. A very large model adaptable to many tasks is a:
(a) foundation model (b) decision tree (c) spam filter (d) calculator
32. A model that handles text, images and audio is:
(a) unimodal (b) multimodal (c) supervised (d) small
33. Open-weight models can be:
(a) only used online (b) downloaded and run on your own hardware (c) never modified (d) used only by governments

34. LLM stands for:
(a) Linear Logic Machine (b) Large Language Model (c) Long Learning Memory (d) Low Latency Model
35. LLMs are trained to predict:
(a) the next token (b) the previous image (c) random numbers (d) stock prices only
36. A token in English is roughly:
(a) 10 words (b) 3/4 of a word (c) one sentence (d) one page
37. About how many tokens is a 750-word English essay?
(a) 100 (b) 1,000 (c) 10,000 (d) 100,000
38. Embeddings represent tokens as:
(a) colours (b) lists of numbers capturing meaning (c) images (d) sounds
39. Words with similar meanings have embeddings that are:
(a) far apart (b) close together (c) identical always (d) random
40. Self-attention lets each token:
(a) ignore other tokens (b) weigh the relevance of all other tokens (c) delete words (d) translate itself
41. The stage that makes a base model follow instructions is:
(a) pre-training (b) instruction tuning (c) tokenization (d) inference
42. RLHF stands for reinforcement learning from:
(a) hardware failures (b) human feedback (c) high frequencies (d) hidden files
43. The context window is:
(a) the chat app window (b) the maximum tokens the model can consider at once (c) the training dataset (d) the screen size
44. A chatbot "remembers" earlier messages because:
(a) it stores memories in its weights (b) the history is resent in each request (c) it reads your mind (d) the internet saves them
45. Low temperature makes output:
(a) more random (b) more focused and consistent (c) longer (d) multilingual
46. High temperature is best for:
(a) extracting invoice data (b) brainstorming creative ideas (c) exact calculations (d) legal documents
47. A confident but false AI answer is called a:
(a) bug report (b) hallucination (c) token (d) embedding
48. RAG stands for:
(a) Random Answer Generation (b) Retrieval-Augmented Generation (c) Rapid AI Growth (d) Recursive Attention Graph
49. RAG reduces hallucination by:
(a) adding relevant documents to the prompt (b) increasing temperature (c) removing the system prompt (d) using fewer tokens
50. Letting a model call a calculator or database is:
(a) tool use / function calling (b) fine-tuning (c) tokenization (d) diffusion

51. An AI system that plans and calls tools in a loop is an:
(a) agent (b) embedding (c) encoder (d) tokenizer
52. The CRAFT checklist stands for Context, Role, Action, Format and:
(a) Tone (b) Tokens (c) Training (d) Temperature
53. A prompt with instructions only (no examples) is:
(a) few-shot (b) zero-shot (c) one-shot (d) multi-shot
54. Giving 2-5 examples in a prompt is:
(a) zero-shot (b) few-shot (c) fine-tuning (d) RAG
55. Asking the model to work step by step helps with:
(a) reasoning problems (b) image resolution (c) file size (d) internet speed
56. Persistent instructions for a whole conversation go in the:
(a) system prompt (b) footer (c) file name (d) URL
57. To get machine-readable output, ask for:
(a) a poem (b) JSON with specified keys (c) more emojis (d) a longer answer
58. Splitting a big task into a sequence of prompts is:
(a) prompt chaining (b) overfitting (c) tokenizing (d) diffusion
59. "Answer only from the text below" is used to:
(a) ground answers and reduce hallucination (b) increase creativity (c) speed up typing (d) change language
60. A good way to improve a reusable prompt is to:
(a) change many things at once (b) test on examples and change one thing at a time (c) never test (d) use the first draft
61. Before sending an AI-written email you should:
(a) send immediately (b) check facts, names and numbers (c) delete it (d) translate it
62. For academic work, using AI to write your assignment and submitting it as your own is:
(a) always allowed (b) usually against integrity rules (c) required (d) encouraged
63. Back-translation is used to:
(a) check a translation's meaning (b) make text longer (c) remove bias (d) generate images
64. AI-generated code should be:
(a) trusted without testing (b) tested and reviewed (c) posted publicly with keys (d) ignored
65. You should never paste into a public chatbot:
(a) a general question (b) passwords, OTPs or Aadhaar numbers (c) a poem (d) a recipe

56. Most modern text-to-image tools use:
(a) diffusion models (b) spreadsheets (c) decision trees (d) expert systems
57. In image generation, fixing the seed allows:
(a) reproducing a result (b) faster internet (c) copyright ownership (d) text translation
58. Replacing part of an image with generated content is:
(a) outpainting (b) inpainting (c) upscaling (d) tokenizing
59. Extending an image beyond its borders is:
(a) inpainting (b) outpainting (c) cropping (d) compressing
70. TTS stands for:
(a) Text-to-Speech (b) Token Transfer System (c) Training Test Split (d) Total Token Size
71. Cloning someone's voice should be done:
(a) freely (b) only with their explicit consent (c) for pranks (d) to make payments
72. A realistic fake video of a real person is a:
(a) meme (b) deepfake (c) GIF (d) thumbnail
73. To check a suspicious viral image you can:
(a) share it first (b) use reverse image search and trusted sources (c) trust the caption (d) like it
74. AI-generated media that could mislead should be:
(a) hidden (b) labelled as AI-generated (c) sold (d) deleted always
75. In India, cyber fraud can be reported on helpline:
(a) 100 (b) 1930 (c) 101 (d) 108
76. An API key should be stored:
(a) in front-end JavaScript (b) securely on the server (e.g. environment variable) (c) in a public repository (d) in the app's UI
77. An LLM API is:
(a) stateful and remembers all users (b) stateless — you send the history each time (c) a database (d) a web browser
78. API pricing is usually measured per:
(a) minute (b) million tokens (c) page (d) user
79. Hidden instructions in a document that try to hijack an AI app are called:
(a) prompt injection (b) fine-tuning (c) caching (d) compaction
80. A no-code custom assistant is built mainly with:
(a) instructions and uploaded files (b) assembly language (c) hardware (d) spreadsheets only
81. An evaluation set is:
(a) a list of test questions with expected answers (b) a type of model (c) a GPU (d) a prompt template only
82. Streaming responses improve:
(a) model accuracy (b) perceived speed in chat interfaces (c) copyright (d) data privacy

33. Reusing a fixed long prompt prefix cheaply across requests is called:
(a) prompt caching (b) fine-tuning (c) tokenizing (d) diffusion
34. Responsible AI principles include:
(a) fairness and transparency (b) secrecy and speed only (c) profit only (d) no human oversight
35. AI bias mostly comes from:
(a) electricity (b) training data and design choices (c) keyboards (d) internet speed
36. Trusting a confident machine too much is:
(a) automation bias (b) overfitting (c) tokenization (d) RAG
37. Decisions like hiring or admissions should be:
(a) fully automated by AI (b) made by humans with AI only assisting (c) random (d) decided by chatbots
38. India's personal data protection law is the:
(a) DPDP Act, 2023 (b) GDPR (c) EU AI Act (d) IT Act 1990
39. The EU AI Act classifies AI systems by:
(a) price (b) risk level (c) colour (d) country
90. Before pasting a work document into AI you should:
(a) check your organisation's policy and remove sensitive data (b) always paste it (c) print it (d) email it to everyone
91. Generative AI's effect on jobs is mostly to:
(a) eliminate all jobs overnight (b) change tasks within jobs (c) have no effect (d) only affect robots
92. A skill growing in value with AI is:
(a) verification and judgement (b) memorising phone numbers (c) typing speed only (d) copying text
93. Large AI models' environmental impact comes mainly from:
(a) electricity and cooling in data centres (b) paper use (c) fuel for cars (d) batteries in phones
94. The first step of an AI project is:
(a) choosing a logo (b) defining the problem and user (c) buying GPUs (d) deploying
95. A good capstone includes:
(a) only a demo video (b) a problem statement, prototype, evaluation and responsible-AI review (c) only code (d) nothing written
96. A mini chatbot should respond to unknown questions by:
(a) inventing an answer (b) saying it doesn't know and giving a contact (c) ignoring the user (d) changing topic
97. Few-shot examples in a classifier prompt should be:
(a) identical (b) varied and representative (c) in a random language (d) wrong on purpose
98. The best image prompts are:
(a) vague (b) concrete, with subject, style, lighting and composition (c) one word (d) only emojis

99. Asking the AI to list its factual claims with confidence levels helps:
(a) verification (b) image quality (c) typing speed (d) file compression

100. A knowledge cut-off means a model:
(a) knows events up to its training date unless given tools like search (b) knows everything current (c) cannot answer anything (d) only works offline

ANSWERS**MCQ Answer Key**

If you got a question wrong, go back to the module it came from and re-read that section.

Q	Ans	Q	Ans	Q	Ans	Q	Ans	Q	Ans
1	b	21	a	41	b	61	b	81	a
2	a	22	b	42	b	62	b	82	b
3	b	23	b	43	b	63	a	83	a
4	c	24	b	44	b	64	b	84	a
5	c	25	b	45	b	65	b	85	b
6	b	26	b	46	b	66	a	86	a
7	b	27	b	47	b	67	a	87	b
8	b	28	b	48	b	68	b	88	a
9	b	29	a	49	a	69	b	89	b
10	a	30	a	50	a	70	a	90	a
11	b	31	a	51	a	71	b	91	b
12	b	32	b	52	a	72	b	92	a
13	c	33	b	53	b	73	b	93	a
14	a	34	b	54	b	74	b	94	b
15	b	35	a	55	a	75	b	95	b
16	b	36	b	56	a	76	b	96	b
17	b	37	b	57	b	77	b	97	b
18	a	38	b	58	a	78	b	98	b
19	a	39	b	59	a	79	a	99	a
20	b	40	b	60	b	80	a	100	a

Short-answer questions (2–3 marks)

1. Define artificial intelligence and generative AI.
2. Differentiate predictive and generative AI with examples.
3. What is overfitting?
4. What is a token? What is an embedding?
5. What is self-attention?
6. What is a context window?
7. What does temperature control?
8. What is a hallucination?
9. Define zero-shot and few-shot prompting.
10. What is RAG?
11. What is a diffusion model?
12. What is a deepfake?
13. Why must API keys be kept on the server?
14. What is prompt injection?
15. Name four principles of responsible AI.

Long-answer questions (5–10 marks)

1. Trace the history of AI from 1950 to today, highlighting five milestones.
2. Explain supervised, unsupervised, reinforcement and self-supervised learning with examples.
3. Explain how a neural network learns, covering neurons, layers, loss, backpropagation and gradient descent.
4. Compare autoregressive transformers, VAEs, GANs and diffusion models.
5. Explain how large language models are built and trained, and why they hallucinate.
6. Describe prompt-engineering techniques with examples of weak and strong prompts.
7. Explain how text-to-image generation works and how to write effective image prompts.
8. Describe the architecture of an AI chatbot application, including RAG and tool use.
9. Discuss the ethical, privacy and legal issues of generative AI and how to address them.
10. Plan an end-to-end AI mini-app, including evaluation and a responsible-AI review.

REFERENCE

Glossary of Terms

Key words used throughout these notes, in alphabetical order.

Activation function

Function deciding a neuron's output (ReLU, sigmoid, softmax).

Agent

AI system that plans and uses tools in a loop to complete tasks.

AGI

Artificial general intelligence — hypothetical human-level AI across tasks.

API

Interface that lets programs call an AI model.

Attention

Mechanism weighing the relevance of tokens to each other.

Autoregressive

Generating one piece at a time from what came before.

Backpropagation

Algorithm computing how each weight affects the error.

Bias (AI)

Systematic unfairness from data or design.

Chain of thought

Step-by-step reasoning before a final answer.

Context window

Maximum tokens a model can process at once.

Deep learning

Machine learning with many-layered neural networks.

Deepfake

Realistic fake image, audio or video of a real person.

Diffusion model

Generates data by removing noise step by step.

DPDP Act

India's Digital Personal Data Protection Act, 2023.

Embedding

Numeric vector representing meaning.

Epoch

One full pass over the training data.

Evaluation set

Test inputs with expected outputs to measure quality.

Few-shot prompting

Including a few examples in the prompt.

Fine-tuning

Extra training of a pre-trained model on specific data.

Foundation model

Large model adaptable to many tasks.

GAN

Generative adversarial network: generator vs discriminator.

Generative AI

AI that creates new content.

GPU

Graphics processor used for parallel AI computation.

Gradient descent

Optimisation by stepping downhill on the loss.

Grounding

Basing answers on supplied trusted sources.

Hallucination

Fluent but false AI output.

Inference

Using a trained model to produce outputs.

Inpainting / outpainting

Editing inside / extending beyond an image.

Latent space

Compressed internal representation of data.

LLM

Large language model.

Loss

Measure of how wrong a model's prediction is.

Machine learning

Systems that learn patterns from data.

Multimodal

Handling several data types (text, image, audio).

Neural network

Layers of connected artificial neurons.

Open-weight model

Model whose weights can be downloaded.

Overfitting

Memorising training data; poor generalisation.

Parameters (weights)

Numbers a model learns during training.

Pre-training

Initial large-scale training on broad data.

Prompt

Input instructions and material given to a model.

Prompt caching

Reusing a repeated prompt prefix to save cost and time.

Prompt chaining

Sequence of prompts, each using the last output.

Prompt engineering

Designing effective prompts.

Prompt injection

Malicious instructions hidden in inputs.

RAG

Retrieval-augmented generation.

Reinforcement learning

Learning from rewards and penalties.

RLHF

Reinforcement learning from human feedback.

Seed

Number fixing randomness to reproduce a generation.

Self-supervised learning

Learning from labels derived from the data itself.

Softmax

Turns scores into probabilities summing to 1.

Supervised learning

Learning from labelled examples.

System prompt

Persistent instructions for an assistant.

Temperature

Setting controlling randomness of generation.

Token

Unit of text a model processes.

Tool use

Model requesting external functions to run.

Top-p

Sampling from the most probable tokens up to probability p.

Transformer

Attention-based architecture behind LLMs.

TTS

Text-to-speech.

Unsupervised learning

Finding structure in unlabelled data.

VAE

Variational autoencoder.

Vector database

Store for embeddings used in semantic search.

Zero-shot prompting

Instructions without examples.

Keep learning

Use the AI tutor on www.aitutorspark.co.in to ask any question from these notes — it explains step by step. Complete the course quiz to earn your certificate. All the best!