Learnbay

Data Structure Algorithm & System Design

This program includes

- ✓ GenAI
- ✓ DevOps
- ✓ Data engineering
- ✓ Product management

For **Software developers** & IT professionals

Inclusive of **GenAI**

www.learnbay.co

Table Of Content

This program is for software developers and IT professionals looking to master **DSA**, **system design**, and problem-solving skills. With practical projects and advanced topics, it helps you prepare for top MNC interviews. The curriculum includes electives like GenAI, DevOps, and Product Management for broader skills.

Practice 200+ DSA Problems
Work on 200+ DSA Problems asked in top tech interviews & improve problem solving

Referrals in Top Tech MNCs
Secure referrals in top product-based companies with our Program

System Design Projects
Work on end to end system design projects & Gain real experience

Multiple Electives
Choose electives like GenAI, Devops, Product management & Data engineering

Program **Summary**

1 **Training Mode**
100% Live Online Session with Instructors from top companies

2 **Program Faculty**
Renowned **industry experts** from top global MNCs

3 **Program Duration**
9-10 Months Program
Weekday and Weekend Batch

4 **200+ Practice Problems**
Practice **interview problems** of top tech companies.

Crack the Interview @ Top Tech MNC

Learn DSA + System Design (HLD +LLD) + GenAI

How many electives are offered in this program?
In addition to DSA and system design, this program includes GenAI, DevOps, Data engineering, and product management (tech) for software engineers. These courses help prepare you for various roles in product-based companies.

What is the number of projects in this program?
You'll work on design projects, using **high- and low-level design** to address real challenges. With mentor guidance, these projects offer hands-on experience similar to working in leading tech companies

Why is GenAI important for software developers?
GenAI helps software developers create smarter apps, automate tasks, and boost productivity, making it essential for fast innovation in top tech MNCs.

About **Course**

This program is for software developers targeting interviews at **top tech MNCs.** Using a project-based approach with high- and low-level design, it mirrors real work at leading tech companies, guided by industry experts and dedicated mentors.

The course also covers **Data Structures and Algorithms (DSA), System Design, and Generative AI (GenAI) for software developers**. By the end, you'll be ready to integrate AI & ML in software development and prepare for software engineering interviews.

Our Commitment

“We are dedicated to delivering accessible and industry-relevant education that empowers India's workforce to grow and succeed.”

We offer flexible learning options, allowing you to choose 100% online learning with personalized one-on-one doubt sessions.

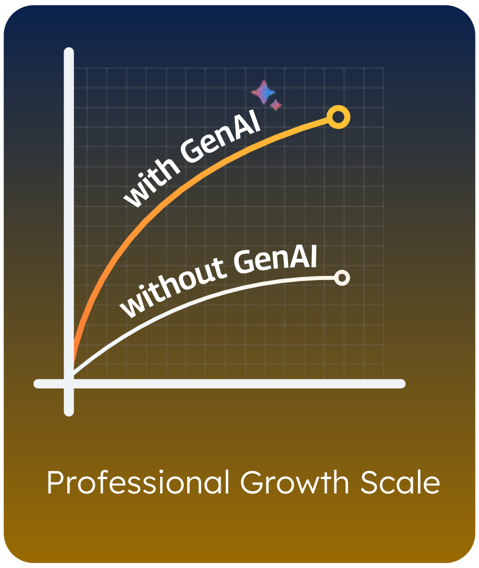
Furthermore, our comprehensive career support services include interview preparation, resume building, and job placement assistance, all designed to help you smoothly transition into SDE role at top product based companies.

100%
Assured Interview

350+
Hiring Partners

7+
Centers Across India

40k+
Professionals Upskilled



70%

of companies are likely to adopt GenAI in next 5 years, to

- enhance efficiency,
- automating tasks,
- improving decision-making for better project outcomes.

***By integrating GenAI into our program**, we ensure that our learners are well-prepared to lead and innovate in their respective fields.

Program **Eligibility**

Software Developer with min **1+ Year of Experience**

A minimum of **1 year of experience** for programmers or developers with a basic **understanding of programming, preferably in Java.**

- **Software Engineer**
- **Executives (IT domain)**

Important Note: This program is not for freshers, fresh grads, students.

Program Outcome: What's in it for you?

Gain Hands-On Experience with Real Projects

Leverage your skills through real-world design projects, to enhance expertise in areas like **high-level and low-level design**. With GenAI, You will be able to integrate LLMS in your application.

DSA + System Design + GenAI for Software Developers

Learn AI/ML along with DSA and System Design to solve real-world problems. Build skills to add AI to software and improve job readiness. Get ready to crack **AI engineering interviews** with GenAI in top MNCs.

Get Job Referrals in Product based companies

Access referrals to **product-based companies** with our support, including mock interviews, resume building, LinkedIn optimisation, cover letter tips, and company-specific mentorship for interview success

Learnbay

Why choose Learnbay?

A unique program for
Tech Professionals!

www.learnbay.co

77956 87988

1. **Project Based** Learning

Our program gives software developers real-world experience in **system design** and full-stack projects, simulating challenges at top tech companies. By tackling **end-to-end design and integrating GenAI**, participants build hands-on skills for today's software development demands.

Live Projects from industry

Dedicated Project Mentor

2. Electives & **Specialization**

In addition to DSA and system design, this program includes **GenAI, DevOps, Data engineering, and product management** (tech) for software engineers. These electives help prepare you for various roles in product-based companies.

GenAI

Product management

Data engineering

DevOps

3. Placement Assistance

Resume Optimization
Expert assistance to enhance your professional resume

Interview Preparation
Mock interviews to improve your performance

Interview Opportunities
Scheduled interviews with potential employers

Career Counseling
Professional guidance for your career advancement

4. Mentors from **Top Tech MNCs**

Guidance from experienced mentors from **top tech MNCs.** They bring hands-on industry expertise to help you navigate **challenging projects**, enhance your skills, and gain the practical knowledge .

5. **Companywise** Interview Prep

Prepare with problems from top tech companies like **Google, Amazon, and Microsoft**, covering coding, system design, and machine coding rounds. The machine coding round hones skills in building structured, scalable code modules, essential for **top MNC interviews.**

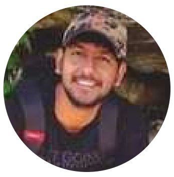
Others Vs **Learnbay**

	Learnbay	OTHERS
Training Mode	✓ 100% Online & Hybrid (Online + Classroom)	✗ Only recorded class & few live online
Support	✓ 24/7 Student Support	✗ Limited Support Hours
Placement	✓ 100% Placement Assistance	✗ Limited Placement Support
Curriculum	✓ Included in Latest Curriculum	✗ Often Not Included
Faculty	✓ Experienced Industry Professionals	✗ Academics and Trainers
Real-Time Projects	✓ Practice with Live Projects and Team Management	✗ Simulated Projects

Alumni Spotlight

Shishir Kamal

I'm currently enrolled in the Full Stack course and have completed the DSA and System Design modules. The trainers' teaching method has been enlightening for me as a beginner. Highly recommend Learnbay for upskilling.



Arpit Agarwal

This online software development course was the best I've taken. Great instructor, easy-to-understand explanations, well-structured and effective hands-on exercises. Highly recommended!

Ritam Mukherjee

Great course for software development, with real-world examples and practical exercises. I learned a lot and could apply it in my career. Highly recommend for learners.

Amrita Panjwani

Highly recommend course for software development. Well-organized material, practical exercises to apply and build skills. Suitable for anyone interested in the field.

Alumni **Spotlight**

Jatin Kumar Khilrani

The software development course was a game-changer. Extensive content, highly skilled trainer, exceptional job support laid the foundation for my career in software development.

Tushar Singhal

Learnbay's software course surpassed my expectations, with clear instruction and helpful instructors. Grateful for the gained knowledge and accessible support.

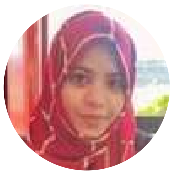
Gowthaman Jeganathan

I'm currently enrolled in the Full Stack course and have completed the DSA and System Design modules. The trainers' teaching method has been enlightening for me as a beginner. Highly recommend Learnbay for upskilling.

Abhishek Pakhide

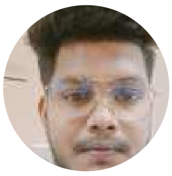
I recently finished Learnbay's Basic DSA Course online. The course is well-structured, taught by knowledgeable trainers with a focus on interview preparation and data structure concepts. It benefits both novices and experts.

Alumni Spotlight



Qudsia Tahniyath

Learnbay has helped me a lot to learn data science applications in the e-commerce industry. The live class concept was really helpful in receiving proper DS training. Thanks to all my mentors and the placement team.



Binit Kumar Swalsingh

I've been with this org since Jan 4, 2023, studying Full Stack Software Development. If you have ambition and need a mentor to show you the right path to success with the right knowledge, I highly recommend this org.

Swarup Halder

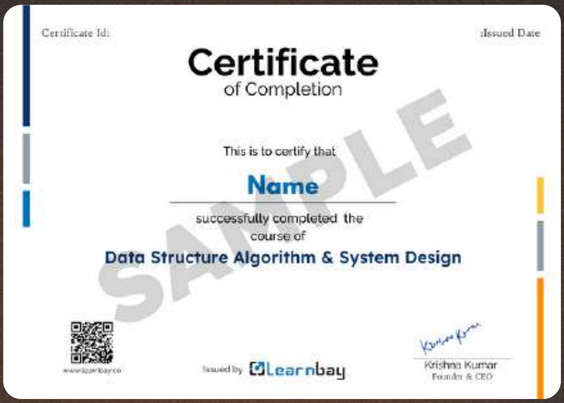
I have enrolled in Software Development course. I had basic knowledge of DSA and since then it has been a wonderful learning experience. The teacher does an excellent job of explaining the concepts in a way that is easy to understand.

Karanveer Bansal

I enrolled in Learnbay's full stack program. I already completed basic DSA and the live sessions were clear, with good teaching pace. They offer an easy and simple EMI option for course fees.

Get certified
and accelerate your career growth

Learnbay Course Completion Certificate



Certificate Id:

:Issued Date

Certificate
of Completion

This is to certify that

Name

successfully completed the course of

Data Structure Algorithm & System Design

SAMPLE

www.learnbay.co

Issued by Learnbay

Krishna Kumar
Founder & CEO

Practical Validation:

The certification reflects not only theoretical knowledge but also the **practical implementation of concepts** through live projects and case studies.

Program Fee

Live online classes

Benefits:

- All classes are covered in live interactive mode
- Schedule doubt clearing session with industry expert
- Online capstone project session
- 1 on 1 Job assistance with online resume build up and mock interview sessions

Program Fee

₹ 75,000 + 18% GST

Pay in easy EMIs starting as low as

₹ 4,916/ month

Learnbay

Program Curriculum

A unique program for Tech Professionals!

www.learnbay.co

77956 87988

Journey Summary

Module 0 8 hrs

Programming Fundamentals

Tools: Java

Term 1 3.5 Months

Data Structures & Algorithms

Tools: NumPy pandas Matplotlib

Term 2 1 Month

Computer Fundamentals

Tools: Machine Language GenAI

Term 3 2.5 Months

System Design : LLD + HLD

Tools: Power BI SQL Tableau Big data

2 months

Elective 1 **GenAI for Software Developers**

Elective 2 **Product Management for Tech Professionals**

Elective 3 **Data Engineering Essentials for Software Engineers**

Elective 4 **DevOps Essentials for Software Engineers**

Module 0 Programming Fundamentals

Duration: 2-3 weeks

Outcome of this term: This module helps engineers brush up on fundamental programming skills, ensuring a strong foundation before diving into advanced topics. If you're already comfortable with the basics and actively coding, **you can skip this** and jump straight into the **DSA & Problem Solving.**

Topic 1: Introduction to Programming

- Basics of programming and Java installation (**JDK, IDE like IntelliJ, Eclipse**)
- Writing and running your first Java program (e.g., Hello World)

Topic 2: Variables, Data Types, and Operators

- **Variables**: Declaration, Initialization, Scope in Java
- **Data types**: int, float, double, char, boolean
- **Operators**: Arithmetic, Relational, Logical, Increment/Decrement
- Input and output in Java: **Scanner class and System.out.println**

Topic 3: Control Flow

- Conditional statements: if, else if, else
- **Switch-case** in Java
- **Loops**: for, while, do-while
- Break and continue

Topic 4: Functions (Methods)

- Defining and calling methods in Java
- Method parameters, return types, and scope
- Method overloading
- **Recursion in Java**

Topic 5:
Arrays and Strings

- **Arrays**: Declaration, initialization, and traversal
- **Multidimensional arrays**
- **Strings**: String class methods, concatenation, comparison, and manipulation

Topic 6:
Object-Oriented Programming (OOP)

- Classes and Objects
- **Constructors** and destructors
- Encapsulation, **Inheritance**, Polymorphism, Abstraction
- Access modifiers: private, public, protected
- Static and non-static members
- **OOP practice problems (e.g., real-world object modeling)**

Note:
"Mastering the fundamentals is key to understanding complex concepts later. Treat this as the bedrock of your coding journey."

To Do:
Create and debug a simple program that performs basic arithmetic operations using user input.

Term 1 Data Structures & Algorithms

Duration: 3.5 Months

Outcome of this term: This module helps you master **DSA and problem solving skills** by solving real problems like those in tech interviews. You'll learn to work with *arrays, linked lists, trees, graphs,* and more, focusing on building efficient solutions. Gain confidence to tackle challenging problems and **succeed in technical interviews.**

Section 1: Foundations of Data Structures and Algorithms

Time and Space Complexity

- Understanding Performance: Importance of complexity analysis in interviews.
- **Big O Notation**: Common complexities and how to analyze them.
- **Space Complexity**: Optimizing memory usage in data structure choices.

Arrays

- **Array Basics**: Handling and manipulating arrays.
- Operations: Insertion, deletion, searching, and updating,
- Two-Dimensional Arrays: Matrix manipulation.
- Maximum element in an array, **Reverse an array**, Minimum element in an array.
- Sorted array check, Count occurrences.
- Rotate an array, Merge sorted arrays, Intersection of arrays, Move zeroes, Pairs with given sum
- **Longest increasing subsequence**, Maximum subarray sum.
- Longest common subsequence, Minimum path sum, Maximum product subarray

Linked Lists & Interview Problems

- **Singly Linked List**: Creation, traversal, insertion, and deletion.
- **Doubly and Circular Linked Lists**: Differences and use cases; related problems.

- **Sample Practice Problems:**
- Middle element of a linked list, **Detect a loop in a linked list**, Length of a linked list, Check if a linked list is a palindrome, Merge two linked lists
- Reverse a linked list, Add two numbers represented by linked lists, Rotate a linked list, Find the start of the loop, Partition a linked list around a value.
- **Remove the N-th node from the end**, Flatten a multilevel doubly linked list, Reorder a linked list,Clone a linked list with next and random pointers, **Split a linked list into two halves**

Stacks and Queues

- Understanding Concepts: **LIFO** and **FIFO** principles —common in problem statements.
- Operations: Implementing stacks and queues efficiently.
- **Priority Queues and Dequeues**: Applications in interview questions.

- **Sample Practice Problems:**
- Check for balanced parentheses, Implement a stack using linked list, Implement a stack using an array,
- Implement a queue using stacks, **Evaluate a postfix expression**, Implement a double-ended queue (Deque), Generate binary numbers from 1 to N, Reverse a queue.
- Implement a queue using a circular array, Design a stack that supports duplicate elements, **Evaluate infix expressions.**

Section 2: Advanced Data Structures and Algorithms

Recursion and Backtracking

- Mastering Recursion: Key concepts and base cases crucial for problem-solving
- **Divide and Conquer**: Understanding its application in interviews.
- **Backtracking**: N-Queens, Sudoku.

- **Practice Problems:**
- Calculate the **factorial** of a number, Fibonacci sequence using recursion, Tower of Hanoi.
- Generate all subsets of a set, Solve the **N-Queens** problem, Rat in a maze.
- Solve the **Sudoku puzzle**, Permutations of a string, Word search, Generate all valid parentheses, Subset sum problem.

Searching Algorithms

- **Linear vs Binary Search:** Time complexities.
- Advanced Searches: Ternary search and applications.

- **Practice Problems:**
- Basic: Linear search in sorted array, Find mini/max in array, **Reverse an array**, Find unique element in array
- Intermediate: Binary search in sorted array, Search in **rotated sorted array**, k-th smallest element in array
- Advanced: First and last position of element in sorted array, **Square root using binary search**, Median of two sorted arrays, Find smallest pair sum

Assignments:
Solve at least 5 problems from LeetCode or HackerRank focusing on arrays and strings.

Sorting Algorithms

- Simple Sorting Algorithms: **Bubble**, selection, insertion—understanding their performance.
- **Advanced Sorting Algorithms**: Merge sort, quick sort—key algorithms in interviews.

- **Practice Problems:**
- Bubble sort, Selection sort, Insertion sort, Sort array of **0s, 1s, and 2s**, Merge two sorted arrays
- Merge sort, Quick sort, Heap sort, Counting sort
- Sort nearly sorted array, **K-th largest element in unsorted array**, Find minimum diff pair in array

Hashing

- **Hash Tables and Hash Functions**: Importance in optimizing search operations.
- Collision Handling Techniques: Techniques to handle collisions effectively.
- Hash Table Implementations

- **Practice Problems:**
- Count frequencies of elements in an array.
- **Two-sum problem**, Find the first non-repeating character in a string.
- **Longest consecutive sequence**, Group anagrams from a list of strings.

Mini Challenge:

Reverse a linked list using recursion and iteratively—compare the two approaches.

Mastering Trees for Problem Solving

- **Binary Trees**: Understanding traversal techniques (inorder, preorder, postorder).
- **Binary Search Trees** (BSTs)
- Balanced Trees (AVL, Red-Black): Importance of balancing in interviews.

- **Practice Problems:**
- Implement binary tree traversals, Count the number of nodes in a binary tree.
- Lowest common ancestor in a binary tree, Serialize and deserialize a binary tree.

Understanding and Implementing Tries

- **Understanding Tries**: Structure and its applications in string problems.
- Insertion and Search Operations

- **Practice Problems:**
- Insert and search in a trie.
- Find all words with a given prefix, **Count distinct substrings.**
- Implement autocomplete system, Implement a phone directory.

Heap Structures and Their Applications

- **Max-Heaps and Min-Heap**s: Key properties & operations.
- Heap Applications

- **Practice Problems:**
- Build a max heap from an array, Find the maximum element in a heap.
- Implement heap sort, **K-th largest element** in a stream using a min heap.
- **Merge k sorted linked lists**, Top k frequent elements in an array.

Greedy Algorithms

- **Understanding Greedy Strategy**: When to apply greedy methods in problem-solving.

- Practice Problems:
- Coin change problem (greedy version), Activity selection problem.
- Job sequencing problem, **Fractional knapsack problem**.
- **Huffman coding**, Minimum spanning tree using Prim's or **Kruskal's** algorithm.

Dynamic Programming (DP)

- **Introduction to DP**: Recognizing overlapping subproblems and optimal substructure.
- Top-down vs. Bottom-up:
- **Common DP Problems**: Longest common subsequence, 0/1 knapsack

- Practice Problems:
- **Calculate the nth Fibonacci number using DP.**
- Longest increasing subsequence, Coin change problem.
- 0/1 knapsack problem, Edit distance between two strings.

Graphs

- **Graph Representations**: Adjacency matrix vs. adjacency list—pros and cons.
- Graph Traversal Algorithms: BFS and DFS—understanding their applications in problem-solving.
- **Shortest Path Algorithms**: Dijkstra's, Bellman-Ford; how to apply them in interviews.

- Practice Problems:
- Implement **BFS** and **DFS** for a graph.
- Detect cycles in an undirected graph, Find connected components.
- **Dijkstra's algorithm** for shortest path, Kruskal's or Prim's algorithm for minimum spanning tree.

Term 2 Computer Fundamentals

Duration: 1 Months

Important Note: We will explore foundational concepts like **computer architecture, operating systems, data representation, and networking**.

Dive into programming principles, version control, and cybersecurity basics, while practicing these topics through real-world projects and case studies.

Topic 1: Introduction to Computer Systems

- Overview of Computer Architecture
- Core Components: CPU, Memory, Storage Devices
- **Memory Types**: RAM, ROM, Cache
- Basics of Program Execution

Topic 2: Operating Systems

- **OS Functions** and **Resource Management**
- Processes and **Threads**, Scheduling Basics
- **Memory Management:** Paging, Virtual Memory
- Concurrency Basics: Locks, Semaphores

Topic 3: Data Representation and Number Systems

- Number Systems: **Binary, Decimal, Hexadecimal**
- Conversions between Number Systems
- **Encoding Basics**: ASCII, Unicode

Topic 4: Networking Fundamentals

- Networking Basics: **LAN, WAN, Internet**
- **OSI** and **TCP/IP** Models, Key Protocols
- IP Addressing Basics, Routing Concepts
- **Network Security:** Firewalls, NAT

Topic 5:
Version Control Systems

- **Git and GitHub Essentials**
- Basic Commands: Clone, Commit, Push
- **Branching**, Merging, **Collaboration Practices**

Topic 6:
Computer Security Basics

- **Cybersecurity** Fundamentals
- Encryption: Symmetric, Asymmetric
- **Authentication and Common Security Threats**

Quick Tip:
"Understand the OSI model thoroughly—it's foundational for network design interviews."

To Do:
Use an online tool like Packet Tracer to simulate a small network with switches and routers.

Term 3 System Design : LLD + HLD

Duration: (2.5 Months)

Outcome of this term: This System Design course covers **High-Level** (HLD) and **Low-Level Design** (LLD) with hands-on projects, similar to work at top tech companies.

You'll learn key concepts like **scalability**, **reliability**, and **security**, work with components like APIs and databases, and integrate AI, including GenAI.

This prepares you to design complex systems and tackle machine coding in **real-world applications** and **technical interviews.**

Overview of System Design

- **Introduction to System Design:** Explanation of key concepts in system design, including scalability, reliability, performance, and security.
- **Components of System Design**: Overview of high-level vs. low-level design, explaining how each contributes to a well-architected system.
- **Building Blocks of System Design**: Introduction to core components like databases, servers, clients, APIs, and message queues.

Section 1: Low-Level Design (LLD)

Chapter 1: Fundamentals of Object-Oriented Programming (OOP)

- **Core principles:** Abstraction, Encapsulation, Inheritance, and Polymorphism.
- **Practical applications with real-world examples to solidify understanding.**
- **Modular component design** for ease of code maintenance.

Chapter 2: SOLID Principles for Effective Design

- **Detailed coverage of each SOLID principle**: Single Responsibility, Open-Closed, Liskov Substitution, Interface Segregation, Dependency Inversion.
- **Exercises for applying these principles to various scenarios to achieve adaptable and clean code.**

Chapter 3: Essential Design Patterns

- **Overview of major design patterns**: Creational (Factory, Singleton), Structural (Adapter, Composite), Behavioral (Observer, Strategy).
- **Identifying the right pattern** based on requirements and scenarios.
- **Practical examples to solve common challenges with design patterns.**

Note:
"LLMs like GPT-3 and GPT-4 work on a transformer architecture, where attention mechanisms allow them to focus on relevant parts of the input data, making them incredibly effective for natural language processing tasks."

MINI PROJECT **DURATION** 2 HRS

Spotify-inspired Playlist Management Build-up

Description: Design a user-friendly system for creating, modifying, and organizing playlists, with features like shuffle, repeat, and sharing options.

Tools covered: OOP principles, Factory, Singleton, Observer patterns.

Chapter 4: UML Diagrams for Visual Representation

- Introduction to essential **UML** diagrams, including **Class, Sequence, and Component diagrams**.
- Exercises in creating visual maps for clear communication of design.

- **Best practices for organized, understandable diagramming.**

Chapter 5: Designing Efficient Database Schemas

- Basics of **schema design** for efficient data management and retrieval.
- Creation of **Entity-Relationship Diagrams** (ERDs) and normalization techniques.
- **Optimization methods** for scalable schemas in high-volume databases.

Chapter 6: API Design and Implementation Basics

- **RESTful API** design principles: Structuring endpoints, managing errors, and versioning.
- Exercises for creating well-structured, scalable APIs with error handling.

- **Best practices in API development for smooth integrations.**

MINI PROJECT **DURATION** 2 HRS

End-to-end Design of Flipkart-like Shopping Cart

Description: High-quality user interaction, rich shopping cart system creation from scratch. Includes common features like add, replace, and delete items. The design focus remains on user experience and scalability.

Concepts Used Normalization of database schema, Sequence diagrams, UML diagrams, and class.

Chapter 7: Multithreading and Concurrency Management

- Introduction to **multithreading** and concurrency control basics.
- Managing issues like race conditions and deadlocks effectively.
- **Practical examples to design systems handling concurrent tasks efficiently.**

Chapter 8: Basic Integration AI/ML in Software Design

- **Basics of embedding ML models** and **GenAI** in systems for functionality enhancement.
- **Setting up data flows for AI/ML** and GenAI: Data preprocessing, model execution, and output integration.
- **Model lifecycle management**, including monitoring and updates for performance consistency.

Chapter 9: Testing and Code Quality Best Practices

- Overview of **unit** and **integration** testing to maintain design stability.
- Techniques for documenting and structuring code for clarity and maintenance.
- **Continuous monitoring** and refinement for long-term code and design quality.

Section 2: High-Level Design (HLD)

Chapter 1: Introduction to System Architecture and Design

- **System design fundamentals**: Overview of monolithic, client-server, and microservices architectures.
- **Principles of system design**: Scalability, reliability, maintainability, and fault tolerance.
- **Evaluating architecture choices:** Choosing the right design patterns for different application needs.

Chapter 2: Distributed Systems Overview

- Key concepts: **Data replication**, data consistency, and distributed communication.
- **Design challenges in distributed systems**: Fault tolerance, data integrity, and managing distributed transactions.
- **Use cases: Examining examples of distributed systems and architectures in action.**

Chapter 3: Network and Communication Protocols

- Network fundamentals: How **TCP/IP**, **DNS**, and **HTTP/HTTPS** protocols function within HLD.
- **Load balancing techniques:** Understanding load balancers, DNS-based load balancing, and API gateways.
- **Case studies: Implementing load balancing in distributed environments.**

Chapter 4: ZooKeeper and Kafka (Distributed Messaging Queue)

- **ZooKeeper**: Overview of distributed coordination, leader election, and configuration management.
- **Apache Kafka**: Introduction to message queuing, real-time data streaming, and event-driven architecture.
- **Implementing a distributed messaging system: Practical applications and setup for reliable messaging.**

Chapter 5: Designing with Microservices and CAP Theorem

- Microservices architecture: Benefits and challenges of service decomposition and design.
- CAP Theorem: Trade-offs between Consistency, Availability, and Partition tolerance.
- Data partitioning: Techniques like sharding, horizontal scaling, and load distribution.

MINI PROJECT **DURATION** 2 HRS

Netflix Content Streaming Service

Objective: Design a microservices-based architecture for Netflix's streaming service, focusing on service decomposition.

Focus: Use CAP principles to manage data consistency, availability, and partition tolerance for real-time streaming with resilience.

Chapter 6: Database Selection and Data Storage Models

- Database types: SQL, NoSQL, and NewSQL, understanding their scalability and data integrity trade-offs.
- Choosing a database: ACID vs. BASE properties, and database selection based on application needs.
- Distributed databases: How they manage scalability, availability, and latency.

Chapter 7: Caching and Content Delivery Networks (CDNs)

- **Caching strategies**: Implementing in-memory caching (Redis, Memcached) and cache expiration techniques.
- **Content Delivery Networks** (CDNs): Using CDNs for load reduction and latency improvements.

- **Optimizing data access:** Practical steps to build efficient caching strategies.

- **Project: Instagram like Image and Video Feed System**
- Objective: Design Instagram's feed system to handle high-frequency data retrieval for photos and videos.
- Focus: Integrate caching and CDN strategies to reduce latency and improve user experience on a large scale.

Chapter 8: Scalability, Security, and Reliability

- **Security best practices: Authentication, data encryption, and secure API design.**
- **Reliability patterns**.
- **Scaling infrastructure**: Vertical and horizontal scaling, auto-scaling, and designing for peak load handling.

Chapter 9: AI/ML Systems in High-Level Design

- **Integrating AI/ML**: High-level considerations for data pipelines, model serving, and scalability for ML.
- **Data infrastructure for ML**: Handling large datasets, real-time processing, and model versioning.
- **Real-world case studies: Examples of high-level AI/ML systems in production environments.**

PROJECT 1 **DURATION** 1.5 HRS

Developing Property Booking System

Objective: Create a booking management system for hotels that supports user reservations and room management.

Focus: Implement a normalized database schema for hotels, rooms, bookings, and user accounts, ensuring the system can handle concurrent bookings and cancellations efficiently.

PROJECT 2 **DURATION** 1.5 HRS

Social Media Feed System

Objective: Efficient real-time notification feed management system for Twitter.

Key Focus: Lightning speed time update and access during high volume data management per second. Effective data Sharding and caching technique implementation.

PROJECT 3 **DURATION** 1.5 HRS

Designing of Booking System for Hotel Rooms (e.g. Oyo)

Objective: User reservation and room management focused booking management systems for business.

Focus: Infusion of a normalized database schema for tracking room bookings, customer accounts, concurrent bookings, and cancellations.

PROJECT 4 **DURATION** 1.5 HRS

Food Ordering System (like Zomato and Swiggy)

Objective: Build a platform connecting users to local restaurants for ordering and delivery.

Focus: Design a microservices architecture for managing restaurant listings, orders, and payments, with real-time tracking and user reviews.

Concepts Used: Microservices, Scalability, Distributed Systems, HLD, LLD.

Select any 2 electives - 2 Months

Elective 1 GenAI for Software Developers

Objective:
Empower developers to leverage **Generative AI** for application development and workflow automation, emphasizing **OpenAI APIs and LangChain integration.**

Recommended Experience:
2-4 years in software development, familiar with Python, APIs, and basic machine learning concepts. Ideal for those integrating AI into applications.

Career Fit:
Perfect for developers interested in AI-driven development and enhancing user experiences or pursuing roles in **AI application design**

Topic 1: GenAI for Software Developers

Fundamentals of Generative AI	Overview of **generative models** and their real-world applications, highlighting differences from traditional AI.
Integration of OpenAI APIs	Techniques for effectively integrating OpenAI APIs, including authentication and **best practices for data handling.**
Building Applications with LangChain	Utilizing **LangChain** to create **intelligent** applications that leverage large language models.

Utilizing Hugging Face

Accessing and deploying **Hugging Face** pre-trained models, with a focus on fine-tuning for specific uses.

Prompt Engineering

Strategies for crafting effective **prompts** to optimize AI responses and improve user interaction.

Ethical Considerations in AI

Discussion of **ethical issues in AI**, including bias, privacy, and responsible usage in development.

Projects using GenAI

Chatbot Development Using OpenAI API

Objective: Create a chatbot for user queries and conversations.

Focus: Use the OpenAI API for natural language understanding.

Content Generation Tool with OpenAI API

Objective: Develop a tool to generate articles or marketing content.

Focus: Utilize the OpenAI API for customizable content creation.

Personalized Recommendation System Using LangChain

Objective: Build a recommendation engine for products or content.

Focus: Employ LangChain for data integration and flow.

Sentiment Analysis Tool Using Hugging Face

Objective: Analyze user content for sentiment.

Focus: Use Hugging Face's models to classify text sentiment

Inspiring Note:
"Generative AI is reshaping industries—from content creation to advanced decision-making."

Discussion Prompt:
"What ethical considerations should developers prioritize when integrating GenAI into applications?"

Elective 2 Product Management for Tech Professionals

Objective:
Equip developers with a **product management** mindset to align technical and business goals in tech products.

Recommended Experience:
3-5 years in software development, ideally in **cross-functional settings with product managers**. Suitable for senior developers or leads transitioning to product roles.

Career Fit:
Engineers aiming for product management or tech lead roles to navigate both **technical and business** needs effectively.

Topics covered

Product Life Cycle & Development	Understanding phases from concept to **launch** and **maintenance**.
Market Research for Tech Products	**Conducting research** to align product with user needs and market demands.
Feature Prioritization Techniques	**Balancing** business and technical requirements to drive value.

Technical Roadmapping	Creating **roadmaps** for technical execution aligned with business goals.
Stakeholder Communication	Translating complex tech concepts for business **stakeholders**.
Data-Driven Decision Making	Leveraging **metrics** and **KPIs** to guide product strategy.
Product Design Basics for Engineers	**Fundamentals of UI/UX** that impact product usability and satisfaction

Case Study Highlight:
Discuss how Slack revolutionized workplace communication with an iterative product development approach.

Discussion Prompt:
"How can tech professionals balance technical feasibility with user needs during product planning?"

Elective 3 Data Engineering Essentials for Software Engineers

Objective:
Equip software engineers with essential data engineering skills to design, manage, and optimize data pipelines and infrastructures for data-driven applications.

Recommended Experience:
2-4 years in software development, familiar with databases and cloud computing. Ideal for engineers interested in data-rich systems or supporting data science teams.

Career Fit:
Suitable for engineers moving into data engineering roles or those enhancing their understanding of data infrastructure in analytics-heavy or data-centric product teams.

Topics covered

Data Pipelines & Workflow Orchestration	Designing efficient data flow for **analytics and operations.**
Database Management & Optimization	Basics of **SQL, NoSQL**, and performance tuning for scalable solutions.
Data Warehousing Concepts	Building and maintaining **data warehouses** for business intelligence.

Big Data Processing	Using frameworks like **Hadoop, Spark for large-scale data operations.**
ETL/ELT	**Data extraction**, **transformation**, and **load processes**, with real-world application.
Data Quality & Monitoring	Ensuring accuracy and **reliability** in data-driven applications.
Cloud Data Solutions	Overview of **cloud-based tools** (AWS, GCP, Azure) for data engineering

Quick Tip:
"When building ETL workflows, always account for data quality checks to ensure downstream processes don’t fail."

Tech Debate:
"Which is more critical in data engineering: scalability or fault tolerance? Why?"

Elective 4 DevOps Essentials for Software Engineers

Objective:
Equip developers with DevOps skills for efficient, scalable code deployment by enhancing CI/CD, infrastructure, and automation workflows.

Recommended Experience:
2-4 years in software engineering with basic knowledge of scripting and cloud services; ideal for developers moving toward DevOps.

Career Fit:
Suited for developers interested in DevOps, cloud management, and automated deployment.

Topics covered

Programming & Scripting for Automation	Python, **Bash**, and scripting for DevOps automation
Linux & System Administration	Essential **Linux commands**, permissions, and troubleshooting
Version Control with Git	Git workflows for collaborative **DevOps** and **code management**

CI/CD Integration	**Jenkins**, GitHub Actions for automated deployment and testing
Containerization & Orchestration	Using **Docker** and **Kubernetes** for scalable deployment environments
Infrastructure as Code (IaC)	**Terraform** and **Ansible** for automated infrastructure setup
Monitoring & Logging	**Prometheus**, **Grafana**, and **ELK** stack for real-time monitoring and troubleshooting

Quick Quiz:

- **What’s the primary difference between containers (like Docker) and virtual machines?**

Real-World Application Insight:

"Netflix uses Spinnaker, an open-source multi-cloud continuous delivery platform, to deploy hundreds of microservices daily without downtime."

Learnbay

Thank **you!**

For more queries and information please reach out to us at:

+91 77956 87988

Visit us at

www.learnbay.co