Interview Preparation Guide for Final Year Students

Kamrul Hasan

Topics Overview

- Core Technical Skills
- Project Expectations
- AI & Open Source
- Communication Hacks
- Problem Solving
- Books & Resources

Core Technical Skills

✅ OOP (Object Oriented Programming)

✅ Design Patterns (Singleton, Factory, Observer...)

✅ SOLID Principles (Clean code structure)

✅ DB (SQL/NoSQL) (Understand CRUD, joins, indexes)

✅ Docker (Containers, Images, Basics)

✅ Operating System (Processes, Threads, Scheduling)

✅ Networking (TCP/IP) (Basics of protocols, HTTP/HTTPS)

✅ REST API (How to build and consume APIs)

Projects: What Companies Expect

- Real-world, working projects
- Complete projects (frontend + backend + deployment)
- Code Quality matters (readable, maintainable)
- Unit Tests show you care about quality
- TDD (Test-Driven Development) is a bonus!
- Documentation

AI Integrations

AI features are attractive in projects

Example:

- Chatbots using OpenAI API
- Smart recommendations
- MCP (Model, Code, Prompt) — Important framework for AI integration

AI Agents: https://platform.openai.com/docs/guides/text?api-mode=responses

Open Source Contribution

- Shows real-world collaboration
- Start small: bug fixes, documentation updates
- GitHub Profile = Your New CV

Communication Skill Development Hacks

- Practice "thinking aloud" when solving problems
- Structure your answers (STAR format: Situation, Task, Action, Result)
- Join coding discussion groups
- Record yourself explaining concepts

Basic Problem Solving

- Master basic Data Structures
- Algorithms: Sorting, Searching
- Graphs: DFS and BFS
- Practice daily on platforms like LeetCode, Codeforces

Gen AI Knowledge

- Understand what Gen AI is (e.g., ChatGPT, DALL·E)
- Know basic use cases: text generation, summarization, Q&A bots
- Explore tools: HuggingFace, OpenAI APIs

Important Books to Read

- Cracking the Coding Interview – Gayle Laakmann McDowell
- System Design Interview – Alex Xu
- Designing Data-Intensive Applications – Martin Kleppmann
- Refactoring.Guru – (Online Resource for Design Patterns)

System Design Interview Prep

- Practice explaining:
  - How to design a URL shortener
  - How to scale an API
- Learn basics of load balancers, caching, sharding

Object Oriented Programming in C++

- Review Classes, Objects, Inheritance, Polymorphism, Friend Functions
- Hands-on practice:
    - Design a Bank Management System
    - Build a Library Management System

Refactor Guru

- Best free resource to understand:
  - Design patterns visually
  - Code smells and how to fix them
- Must study 5-7 common patterns deeply

Designing Data-Intensive Applications

- Deep dive into:
  - Storage Engines
  - Distributed Systems
  - Data Replication and Partitioning
- Critical for backend and system design interviews