6-Month Plan to Crack Product-Based Company Interviews

Executive Summary

This comprehensive 6-month roadmap is designed to help full-stack developers (especially those with 2-3 years of experience) systematically prepare for interviews at top product-based companies like Amazon, Microsoft, Google, Flipkart, Swiggy, Paytm, etc. The plan focuses on three core pillars: **Data Structures & Algorithms (DSA)**, **System Design**, and **Technical Communication**.

Month 1: Foundation Building (Weeks 1-4)

Focus: DSA Fundamentals & Problem Solving Approach

Week 1-2: Core Data Structures

Topics to Cover:

- Arrays and Strings (manipulation, sliding window, two pointers)
- Linked Lists (reversal, detection, intersection)
- Stacks and Queues (applications, monotonic stacks)
- Hash Maps and Sets (frequency counting, duplicate detection)

Daily Schedule:

- 60 min: Learn concept from GeeksforGeeks or LeetCode
- 60 min: Solve 3-4 problems (Easy → Medium)
- 30 min: Code review and optimization

Resources:

- LeetCode (Easy → Medium problems)
- InterviewBit DSA module
- Your existing knowledge in JavaScript/Node.js

Target: Complete 40-50 problems

Week 3-4: Advanced Data Structures

Topics to Cover:

- Binary Trees (traversals, LCA, BST operations)
- Graphs (DFS, BFS, shortest path)
- Heaps (priority queues, k-largest elements)
- Tries (prefix matching, word search)

Daily Schedule:

- 90 min: Concept learning + visualization
- 60 min: Solve 2-3 medium problems
- 30 min: Time complexity analysis

Target: Complete 30-40 problems

Milestone: By end of Month 1, you should comfortably solve 2-3 medium LeetCode problems in 30-45 minutes.

Month 2: Algorithmic Techniques & Pattern Recognition (Weeks 5-8)

Focus: Problem-Solving Patterns & Competitive Coding

Week 5-6: Critical Algorithms

Topics to Cover:

- Sorting and Searching (binary search, merge sort, quicksort)
- Divide and Conquer
- Greedy Algorithms
- Backtracking and Dynamic Programming basics

Daily Schedule:

- 90 min: Learn algorithm + study 2-3 implementations
- 60 min: Solve 3-4 problems (Mix of Easy-Hard)
- 30 min: Analyze edge cases and optimize

Resources:

- LeetCode's "Study Plans" (particularly DSA or Interview prep)
- AlgoExpert (if budget allows)
- YouTube: Fraz, Code Decoded

Target: Complete 50-60 problems

Week 7-8: Dynamic Programming & Optimization

Topics to Cover:

- 1D DP problems (climbing stairs, house robber, palindrome)
- 2D DP (matrix problems, edit distance)
- DP state design and transition
- Backtracking combinations and permutations

Daily Schedule:

- 90 min: Concept learning with multiple approaches
- 75 min: Solve 2-3 DP problems (Medium → Hard)
- 30 min: Space optimization discussion

Key Insight: For product companies, "good DP understanding" matters more than "every DP pattern". Focus on 15-20 core DP problems.

Target: Complete 35-40 problems

Milestone: By end of Month 2, you should have a solid mental library of ~150 problem patterns and be able to recognize and solve them efficiently.

Month 3: Specialized Topics & Interview Simulation (Weeks 9-12)

Focus: LeetCode Hard Problems & Mock Interviews

Week 9-10: Complex Problem Solving

Topics to Cover:

- Graph algorithms (Dijkstra, DFS/BFS variations)
- Advanced string algorithms (KMP, rolling hash)
- Interval problems and merge operations
- Bit manipulation for optimization

Daily Schedule:

- 90 min: Master 1 complex topic
- 90 min: Solve 2-3 hard LeetCode problems
- 30 min: Document approach and edge cases

Target: Solve 25-30 Hard problems

Week 11-12: Mock Interviews & Assessment

Activities:

- Take 2 timed LeetCode contests (weekly)
- Complete 1 full mock interview session (2 problems, 90 min total)
- Review company-specific frequent questions
- Analyze your weak areas

Mock Interview Platforms:

- Pramp or Interviewing.io (free sessions)
- LeetCode Premium (interview questions filtered by company)
- Gain confidence in live coding

Assessment Metrics:

- Solve 2-3 medium problems in 30-40 minutes
- Solve 1 hard problem in 45-60 minutes
- Explain approach clearly without stumbling

Milestone: You're now interview-ready for the DSA/Coding round.

Month 4: System Design Fundamentals (Weeks 13-16)

Focus: Building Architectural Thinking

Week 13: System Design Basics

Core Concepts to Master:

- Scalability (horizontal vs vertical)
- Load balancing (Round-robin, consistent hashing)
- Caching strategies (LRU, Write-through, Write-back)
- Database design (SQL vs NoSQL trade-offs)
- CAP theorem and consistency models

Daily Schedule:

- 60 min: Study one concept with real-world examples
- 60 min: Read case studies (Instagram, YouTube, Uber)
- 30 min: Draw architecture diagrams

Resources:

- "Designing Data-Intensive Applications" by Martin Kleppmann
- SystemsExpert (Alex Xu's course)
- YouTube: Tech Dummies, DesignGurus
- Your AWS knowledge from work

Target: Understand all 8-10 fundamental building blocks

Week 14-15: Real-World System Design Problems

Problems to Solve:

- URL Shortener (like bit.ly, tinyurl)
- Designing Uber/Ola
- Instagram or Twitter feed design
- E-commerce platform (like Flipkart/Amazon)
- Web crawler and search index
- Distributed cache (like Redis)

Approach for Each:

1. Clarify requirements and constraints
2. Draw high-level architecture
3. Deep-dive into critical components
4. Discuss trade-offs and scaling challenges
5. Handle failures and edge cases

Daily Schedule:

- 90 min: Study problem walkthrough (1 problem per day)
- 60 min: Independently design different system
- 30 min: Compare approaches and improvements

Target: Comfortably design 10-12 systems

Week 16: Database Design & Query Optimization

Topics to Cover:

- Normalization vs denormalization
- Indexing strategies (B-tree, hash indexes)
- Query optimization
- Sharding strategies (by user ID, geographic, etc.)
- Replication and consistency

Daily Schedule:

- 75 min: Study one database topic
- 60 min: Design database schema for assigned problem
- 30 min: Write optimized SQL queries

Your Advantage: Use your MySQL and MongoDB experience! Practice schema design for real projects.

Milestone: You understand how to design databases that scale to millions of users.

Month 5: Advanced System Design & Behavioral Prep (Weeks 17-20)

Focus: Depth, Trade-offs, and Communication

Week 17-18: Advanced Scenarios

Complex System Design Problems:

- Video streaming platform (Netflix, YouTube)
- Real-time messaging (WhatsApp, Telegram)
- Distributed job scheduler
- Payment system (handling transactions, failures)
- Analytics platform (BigData)

Deep-Dive Elements:

- Microservices vs monolithic trade-offs
- Event-driven architecture
- Message queues (Kafka, RabbitMQ)
- Distributed transactions and SAGA pattern
- Rate limiting and throttling
- Monitoring, logging, and observability

Daily Schedule:

- 120 min: Study complex system with multiple trade-offs
- 90 min: Design system with emphasis on scalability
- 30 min: Discuss monitoring and debugging strategies

Target: Master 8-10 advanced problems

Week 19-20: Behavioral Interview Preparation

STAR Method Practice:

- **S**ituation: Set the context
- **T**ask: Describe the challenge
- **A**ction: Explain what you did
- **R**esult: Share the outcome with metrics

Stories to Prepare (Using Your Experience at Dotsquares):

1. Most challenging technical problem and how you solved it
2. Time you had to learn new technology quickly
3. Conflict with team member and resolution
4. Project where you showed leadership
5. Failure and learnings from it
6. Time you optimized code/query for performance
7. Multi-team collaboration and coordination

Preparation:

- Write 8-10 stories in STAR format (1-2 min each)
- Practice aloud (reduces nervousness)
- Get feedback from friends or mentors
- Connect stories to FAANG leadership principles

Additional Preparation:

- Research company culture and values
- Prepare 3-5 thoughtful questions to ask interviewers
- Mock behavioral interviews on Pramp
- Study your resume thoroughly (be ready to explain every project)

Milestone: You have polished, concise STAR stories ready for any question.

Month 6: Final Polish & Interview Success (Weeks 21-24)

Focus: Integration, Confidence, and Closing Gaps

Week 21: Topic Mastery & Weak Area Correction

Activities:

- Identify remaining weak areas from mocks
- Re-solve 20-30 problems in weak categories
- Study 2-3 new system design patterns you missed
- Review recent company-specific interview questions

Platform: LeetCode company-filtered problems

Daily Schedule:

- 120 min: Targeted problem-solving in weak areas
- 60 min: Review editorial solutions and learn optimization
- 30 min: Maintain strong areas with 1-2 problems

Week 22-23: Full Mock Interview Rounds

Comprehensive Testing:

- 4-5 complete mock interviews (code + system design)
- Platforms: Pramp, Interviewing.io, or arrange with friends
- Record and review your performance
- Feedback analysis and improvement

Mock Interview Schedule:

- Monday: Full coding round (2 problems, 90 min)
- Wednesday: Full system design round (45-60 min)
- Friday: Behavioral + technical round (60 min)

Evaluation Criteria:

- Problem-solving approach and communication
- Code quality and edge case handling
- System design clarity and trade-off discussions
- Behavioral story delivery and authenticity
- Time management

Target: Score 8-9/10 on most mocks

Week 24: Final Week - Confidence & Execution

Pre-Interview Checklist:

- [ ] Review 20 most commonly asked problems
- [ ] Understand your top 5 system design patterns
- [ ] Refine all behavioral stories
- [ ] Practice coding setup (IDE, debugging)
- [ ] Test internet, camera, microphone (for remote interviews)
- [ ] Get 7-8 hours sleep for 2-3 days before interview
- [ ] Light exercise or yoga before interview

- [ ] Have water bottle ready

Day Before Interview:

- Light revision only (no heavy problem-solving)
- Prepare 2-3 questions for interviewers
- Gather documents needed (resume, project details, references)
- Relax and build confidence

Interview Execution:

- Show genuine enthusiasm for the company
- Ask clarifying questions
- Think aloud (communicate your thought process)
- Acknowledge mistakes and pivot
- Ask for hints if stuck (it's encouraged!)
- Answer behavioral questions with confidence

Daily & Weekly Structure

Recommended Daily Routine

5-6 Hours Daily (Optimal for employed professionals):

Morning (before work):

- 30 min: Warm-up with 1-2 easy problems
- 30 min: Learn new concept

Evening (post-work):

- 90 min: Solve 2-3 problems with focus
- 60 min: System design or review
- 30 min: Optional - read article or watch video

Weekends:

- 2-3 hours: Deep learning session
- 1-2 hours: Mock interview or timed contest
- Relaxation and recovery

Weekly Goals

Week	DSA Focus	System Design	Behavioral	Target Problems
1-4	Fundamentals	-	-	120-150
5-8	Patterns & Algorithms	-	-	80-100
9-12	Hard Problems & Mocks	Basics	-	60-80
13-16	Maintenance	Core Concepts	-	40-50
17-20	Maintenance	Advanced	Stories	30-40
21-24	Weak Areas	Polish	Refinement	20-30

Key Resources by Category

DSA & Coding

Platform	Cost	Best For
LeetCode	Free / Premium	Practice & company-specific questions
InterviewBit	Paid	Structured learning path
GeeksforGeeks	Free	Concept explanations
Pramp	Free	Mock interviews
HackerRank	Free	Algorithms track

System Design

Resource	Cost	Type
SystemsExpert	Paid	Comprehensive video course
DesignGurus	Paid	Course + practice problems
"Designing Data-Intensive Applications"	$30	Book (excellent reference)
YouTube: Tech Dummies	Free	Video explanations
Your AWS Experience	Free	Leverage your real-world knowledge

Mock Interviews

- **Pramp**: Free peer-to-peer mock interviews
- **Interviewing.io**: Anonymous coding interviews with real engineers
- **LeetCode Premium**: Company-specific questions

Tips for Success

1. Consistency Over Intensity

- Regular 5-6 hours daily beats sporadic 12-hour sprints
- Build sustainable habits for 26 weeks
- Take 1-2 rest days per month (crucial for retention)

2. Track Your Progress

- Maintain a spreadsheet of problems solved
- Track accuracy, time, and approach
- Identify pattern weak areas early
- Monitor your mock interview scores

3. Leverage Your Experience

- You have 2-3 years of production experience—use it!
- Relate system design concepts to real projects you've built
- Use your AWS, Docker, and full-stack knowledge
- Share real-world scenarios in behavioral interviews

4. Join Communities

- r/developersIndia on Reddit
- LeetCode discussion forums
- CodeChef/HackerRank communities
- LinkedIn groups for interview preparation

5. Avoid Common Pitfalls

- ✕ Don't memorize solutions—understand approaches
- ✕ Don't skip edge cases
- ✕ Don't neglect behavioral preparation (15-20% of interviews)
- ✕ Don't skip system design (if applying for senior roles)
- ✕ Don't practice only in Python/Java if you code in JavaScript

6. Communication Skills

- Practice explaining your solution as you code
- Draw diagrams for system design
- Ask clarifying questions before diving
- Discuss trade-offs openly

Target Companies & Expected Focus

Tier-1 Product Companies

Amazon, Microsoft, Google, Meta, Apple

- Coding: Hard DSA problems + edge cases
- System Design: Complex, large-scale systems
- Behavioral: Intense culture fit assessment

Tier-2 Product Companies

Flipkart, Swiggy, Paytm, Adobe, Zomato

- Coding: Medium-Hard DSA problems
- System Design: Domain-specific scenarios
- Behavioral: Standard interview process

Tier-3 Growing Companies

Unacademy, FinTech startups, etc.

- Coding: Medium DSA problems
- System Design: Practical, not enterprise-scale
- Behavioral: More relaxed, culture-focused

Assessment Milestones

Month-End Evaluations

End of Month 1:

- [ ] Solved 120-150 DSA problems
- [ ] Comfortable with all basic data structures
- [ ] Can solve 2-3 easy problems quickly

End of Month 2:

- [ ] Solved 200-250 DSA problems total
- [ ] Recognize common problem patterns
- [ ] Can solve medium problems in 30-40 minutes

End of Month 3:

- [ ] Solved 260-310 DSA problems total
- [ ] Comfortable with hard problems
- [ ] Mock interview score: 7/10 or better

End of Month 4:

- [ ] Understand 10+ core system design concepts
- [ ] Can design 10-12 systems end-to-end

- [ ] Strong database design knowledge

End of Month 5:

- [ ] Designed 20+ system design problems
- [ ] 8-9/10 on behavioral mock interviews
- [ ] Polished STAR stories for 8+ scenarios

End of Month 6:

- [ ] Ready for interviews
- [ ] Consistent 8-9/10 on all mock rounds
- [ ] Filled all knowledge gaps
- [ ] Interview-ready confidence level

Frequently Asked Questions

Q: I don't have 5-6 hours daily. Can I still prepare?
A: Yes, but timeline extends. With 3-4 hours daily, plan for 8-9 months instead. Quality matters more than quantity.

Q: Do I need to solve all 300+ problems?
A: No. 150-200 well-understood problems, covering all patterns, is sufficient. Quality over quantity.

Q: Is JavaScript/Node.js good enough or should I use Java/C++?
A: JavaScript is fine. Product companies care about problem-solving, not language. Stick with what you know best.

Q: How much weightage to system design?

- Junior roles (0-2 yrs): 20% system design, 80% DSA
- Mid-level (2-5 yrs): 50% system design, 50% DSA
- Senior roles (5+ yrs): 70% system design, 30% DSA

Q: When should I apply?
A: Apply after Month 4. This gives you 2 months to practice the specific company's patterns.

Q: Any shortcuts?
A: No magic shortcuts. Consistency, focused practice, and genuine

learning are essential. But a structured plan (like this one) accelerates your journey significantly.

Final Motivation

You already have significant advantages:

- ✅ 2-3 years of production experience
- ✅ Full-stack expertise (React, Node.js, MongoDB, MySQL, AWS)
- ✅ Real-world system design exposure
- ✅ Team collaboration and communication skills

This 6-month plan is designed to formalize and sharpen what you already know, while filling gaps in algorithmic thinking and system design depth. Product companies value real experience—use it!

Your goal: Crack top-tier product company interviews by Month 7. Let's go!