THE SDE INTERNSHIP PLAYBOOK

INDIA • 2027 RECRUITMENT INTELLIGENCE

A practical 6-month, 24-week system for computer science students targeting software engineering internships at leading product companies.

Company hiring windows • Eligibility checks • DSA mastery • Core CS • Projects • Resume • Interviews

RESEARCH SNAPSHOT: 23 NOVEMBER 2026

Designed for Indian B.E./B.Tech CSE, IT and related-degree students

An independent educational guide. Not affiliated with or endorsed by any employer.

How to use this playbook

START HERE • SCOPE • ASSUMPTIONS

This guide is built for a student preparing for SDE / Software Engineer internship selection in India. It combines a current-cycle snapshot with a repeatable preparation system. Company recruitment is not synchronized: some roles are campus-only, some are off-campus, some are global, and some appear with little notice.

The three operating rules

- Treat a live company job description as the final authority. Eligibility, graduation year, location, duration and work authorization can change by requisition.
- Prepare continuously; do not wait for a company to announce a role. Strong candidates build DSA fluency, CS fundamentals and project depth before applications open.
- Apply early and track every application. Rolling review means a role may stop accepting candidates before a public deadline.

Who this is for

Primary audience: undergraduate students in India aiming at product engineering internships, especially students in the pre-final year or approaching it. The guide assumes access to a laptop, internet, a coding platform and approximately 12–18 focused study hours per week for 24 weeks. The roadmap can be adapted around classes and exams.

What this guide does not promise

No preparation plan can guarantee an interview, offer, stipend or conversion. Hiring depends on role availability, eligibility, applicant pool, interview performance, location, college hiring channels and business priorities. Dates labelled 'pattern' are planning estimates, not confirmed announcements.

How to get value from it

1. Read the company calendar and identify roles matching your graduation year.
2. Use the 24-week roadmap as a weekly execution contract.
3. Complete the application tracker and run timed mock interviews.
4. Re-check official job pages every week; update your plan when a real posting appears.

2027 hiring calendar: the big picture

INDIA • RESEARCH SNAPSHOT

As of 23 November 2026, the 2027 internship market is already in motion. Several large-company summer recruiting windows began months earlier. For a student who missed an early campus cycle, the correct response is not to abandon 2027: pursue late/off-cycle roles, research programmes where eligible, and the next recruiting cycle while maintaining interview readiness.

Period	What to do	Planning interpretation
Nov–Dec 2026	Check open 2027 roles; finish resume; apply to active postings; watch Qualcomm close window.	Immediate action window. Some cycles close in December.
Jan–Mar 2027	Monitor off-cycle hiring, new requisitions, startup/product roles and research programmes.	Hiring is posting-specific; do not assume a single summer opening date.
Apr–Jun 2027	Target late openings and internships with flexible start dates; prepare for next cycle.	Many summer roles may already be filled.
Jul–Sep 2027	Begin monitoring the next summer recruiting wave; refresh resume and project evidence.	A number of large firms recruit early.
Oct–Dec 2027	Continue rolling applications and campus hiring; build for 2028 cycle.	Use official alerts and campus placement cell.

Verified current-cycle examples

- Google Student Researcher 2027 in India is a live research-oriented listing with a stated 4 December 2026 application deadline; it is open to enrolled bachelor's, master's and PhD students with relevant research-area experience. It is not the same as a general undergraduate SWE internship.
- Amazon lists a 2027 India SDE internship as a six-month, in-person programme, with start dates between January and June 2027. Do not assume it is a standard three-month summer placement.
- Qualcomm India publishes a July-to-December application window for summer and six-month internships; the summer programme runs May–July and requires at least one semester remaining after the internship.
- Goldman Sachs lists its 2027 India Summer Analyst programme for students graduating in 2028, with applications opening July 1. Engineering is among the business divisions, but this is a finance-firm programme, not a pure software-company internship.

These examples show why exact role pages matter more than generic 'internship season' charts.

Company tracker: Tier 1 product engineering

OFFICIAL STATUS VS PLANNING WINDOW

Company	2027 India signal / timing	Role fit & action
Google	Student Researcher 2027 India listing: deadline 4 Dec 2026. General UG SWE availability must be checked separately; 2027 SWE window may have passed.	SWE, systems, ML/research. Apply only if role's degree and research criteria fit.
Microsoft	India eligibility page describes University Interns for pre-final-year students; exact posting dates are requisition-specific.	SWE / Explore are distinct programmes. Check year-of-study and remaining-study rules.
Amazon	2027 India SDE I Intern listing: six months, in-person; start dates Jan–Jun 2027.	DSA, coding, OOP, problem solving. Check duration and availability.
Meta	No verified India 2027 undergraduate posting established in this snapshot.	Set official job alerts; apply only to India-eligible student requisitions.
Apple	No fixed India undergraduate internship calendar verified here.	Search Apple jobs by Intern and India; team-specific roles vary.
NVIDIA	Internship and university programmes vary by team and location; no fixed India 2027 date confirmed here.	C++/Python, systems, GPU/AI roles; check exact requisition.

Practical priority

Do not rank companies only by brand. Rank openings by eligibility match, technical fit, location feasibility, duration and application deadline. A role with a genuine fit and a live application is more valuable than a famous company role that does not accept your graduation year.

Official pages: Google Careers (careers.google.com); Microsoft Careers (careers.microsoft.com); Amazon Jobs (amazon.jobs); Meta Careers (metacareers.com); Apple Jobs (jobs.apple.com); NVIDIA Careers (nvidia.com/en-us/about-nvidia/careers/). Use the exact student/intern filter and India location.

Company tracker: high-value India employers

PRODUCT, CLOUD, ENTERPRISE AND FINTECH

Company	Published / observed pattern	Recommended preparation
Qualcomm	Official India page: July openings to December closure for summer & six-month programmes; summer runs May–July.	C/C++, DSA, OS, computer architecture, embedded/systems depending on team.
Atlassian	Official early-careers page: India recruitment period July–December; intern programme May–August, rolling dates.	DSA, practical coding, collaboration, product thinking; monitor rolling listings.
Salesforce	A Summer 2027 Software Engineer Intern posting was published 28 July 2026 for Class of 2028, Bengaluru/Hyderabad.	DSA, OOP, APIs, databases, debugging, teamwork; read posting for exact requirements.
Oracle	Official student page advertises Software Engineer Intern roles; dates are role-specific.	Java/C++, SQL, OOP, DBMS, cloud/infrastructure basics.
Adobe	No fixed India 2027 date verified in this research snapshot.	DSA, C++/Java, OOP, debugging and strong project explanation.
Uber	No fixed India 2027 undergraduate date verified here.	DSA, practical coding, backend fundamentals, reliability and data modelling.
LinkedIn	No fixed India 2027 date verified here.	DSA, APIs, databases, distributed systems basics.
Walmart Global Tech / Flipkart	Campus and off-campus requisitions vary by year and hiring team.	DSA, Java/Python, backend, DBMS, scalability basics.

Use the career pages as the source of truth: Qualcomm Careers India; Atlassian Early Careers; Salesforce Careers; Oracle Students & Graduates; Adobe Careers; Uber Careers; LinkedIn Careers; Walmart Careers; Flipkart Careers. Search by location and 'intern' rather than relying on reposted job-board lists.

Additional targets and smart diversification

FINANCE, CONSULTING AND OFF-CYCLE EMPLOYERS

Goldman Sachs: a relevant adjacent target

Goldman Sachs 2027 Summer Analyst India is designed for students graduating in 2028; the official programme page states applications open July 1 and an 8–10 week summer experience. Engineering is one of the listed divisions. Treat it as a finance-sector engineering opportunity: the assessment may include coding and problem solving, but the role and business context differ from a conventional product-company SDE internship.

Other employers to monitor

- JPMorgan Chase: technology analyst and software engineering internships; role names, graduation windows and locations vary.
- Morgan Stanley: technology summer analyst roles; track India requisitions and campus channels.
- SAP Labs: developer and technology internships; Java, databases, cloud and enterprise software can be relevant.
- Cisco: software, networking, security and systems internships; C/C++, networking and OS knowledge may matter.
- Intuit, ServiceNow, PayPal, Stripe and Expedia Group: monitor India student roles when listed; do not assume annual openings.

Off-cycle is a real strategy

Off-cycle internships appear throughout the year when teams need capacity. Build alerts for 'Software Engineer Intern', 'Software Development Intern', 'Backend Intern', 'SDE Intern', 'University Intern' and 'Developer Intern'. Search Bengaluru, Hyderabad, Chennai, Pune, Gurugram, Noida and remote India where permitted.

A third-party off-cycle listing aggregator reported dozens of India software engineering internship openings during the 2026 season. Use aggregators for discovery only; verify every role on the employer's own application page before sharing personal information.

Eligibility: avoid wasting applications

GRADUATION YEAR, DURATION AND WORK RULES

The eligibility checklist

- Graduation year: confirm whether the role accepts your expected degree completion year. 'Class of 2028' is not interchangeable with 'graduating in 2028' unless the employer defines it that way.
- Study status: some programmes require pre-final-year status; others accept any currently enrolled student or research candidates.
- Remaining semesters: verify whether you must return to university after the internship and how much study must remain.
- Degree and branch: check CSE/IT/related discipline requirements; some hardware roles accept ECE/EE, while software roles may be broader.
- Duration: confirm whether the internship is 8 weeks, 10–12 weeks, 3 months, 6 months or 11 months.
- Location and attendance: in-person, hybrid and remote roles have different travel and availability requirements.
- Work authorization: international postings may require local work rights; an India-based student should not assume visa sponsorship.
- Academic and administrative conditions: CGPA cut-offs, backlog restrictions, NOC, attendance and college approval can be decisive.

Important programme distinctions

Microsoft's India eligibility page describes University Interns and Explore Interns with pre-final-year conditions; the Explore programme page describes an 8-week India programme for early-college students. The programme pages should be read together with the live posting because programme-specific year rules can differ.

Google Student Researcher is research-oriented and may accept undergraduate students, but it expects relevant experience in a research area. It should not be presented as a substitute for a standard undergraduate SWE internship.

What SDE intern interviews actually test

A COMPETENCY MAP

Competency	What interviewers look for	Evidence you should build
DSA / algorithms	Correctness, complexity, edge cases, problem decomposition.	Solved patterns, timed practice, clear verbal explanation.
Programming fluency	Readable, testable code in one strong language.	One primary language; no syntax hesitation.
OOP / design basics	Encapsulation, interfaces, composition, clean responsibilities.	Small maintainable project; explain design choices.
Core CS	OS, DBMS, networking and concurrency fundamentals.	Short notes plus scenario-based answers.
Practical engineering	Debugging, APIs, tests, Git, code review, deployment.	A working project with tests, README and live demo.
Communication	Clarifying questions, trade-offs, concise explanation.	Mock interviews and project walkthroughs.
Ownership	Ability to take a vague task to a reliable result.	Specific project contribution and measurable outcomes.

A balanced preparation split

For a typical generalist SDE internship, allocate approximately 45% of technical preparation to DSA, 20% to core CS, 20% to project engineering and 15% to interview communication and applications. Adjust after reviewing actual job descriptions. A systems, embedded or ML role needs a different weighting.

One language, not five

Choose one primary interview language and become fluent. C++ is strong for DSA because of its standard library and performance; Java and Python are also widely used. Do not switch languages mid-preparation unless a role requires it. Learn enough of a second language to read code, but build interview speed in one.

DSA mastery map: the required concepts

FROM FUNDAMENTALS TO INTERVIEW PATTERNS

Order	Topic	Must be able to do
1	Complexity analysis	Big-O time/space; compare solutions; amortized intuition.
2	Arrays & strings	Two pointers, sliding window, prefix sums, hashing, intervals.
3	Sorting & searching	Binary search variants, lower/upper bound, custom sorting.
4	Linked lists	Reverse, cycle detection, merge, middle, pointer manipulation.
5	Stacks & queues	Monotonic stack, next greater element, deque, queue simulation.
6	Trees & BST	DFS/BFS, recursion, height, diameter, LCA, validation.
7	Heaps	Top-K, k-way merge, scheduling, streaming median.
8	Graphs	BFS/DFS, components, cycle detection, topological sort, shortest paths.
9	Backtracking	Subsets, permutations, combinations, pruning, constraint search.
10	Dynamic programming	1D/2D DP, subsequences, knapsack patterns, grid DP, state design.
11	Greedy & intervals	Sorting-based choices, proofs/intuitions, merge/meeting rooms.
12	Tries / bit manipulation	Prefix search, XOR, masks, bit operations.

Depth standard

For each pattern: understand the brute-force approach, identify the bottleneck, derive the optimized method, state complexity, code it without hints, test boundary cases and explain why it works. Memorizing a final code snippet is not mastery.

Problem selection

Use curated topic lists and company-tagged practice as a guide, not as a guarantee of exact interview questions. Prioritize representative problems across patterns. Re-solve missed problems after 1 day, 1 week and 3 weeks. Keep a mistake log: wrong pattern, missed edge case, complexity error, implementation bug or communication gap.

DSA practice system and quality bar

HOW TO TURN SOLVING INTO SKILL

The 4-pass method

5. Attempt independently for 25–35 minutes. Write down the pattern you suspect and the simplest correct approach.
6. If stuck, take a small hint rather than reading the full solution. Close the explanation and reconstruct the solution.
7. Implement cleanly, test normal and adversarial inputs, then state time and space complexity.
8. Revisit the problem without notes after 24 hours and again after 7–14 days.

Weekly output target

A realistic target for a student with classes is 12–18 high-quality problems per week, with revision. A student who already has strong foundations may do more; a beginner should do fewer and spend longer understanding each pattern. The metric is independent re-solve rate, not raw submission count.

A good solution explanation

- Clarify input constraints, duplicates, empty inputs, sorting assumptions and expected output.
- Describe brute force briefly and explain why it is too slow.
- State the invariant or key observation behind the optimized algorithm.
- Walk through one example and one edge case.
- Give time and space complexity, including auxiliary space.

Mistake log template

Date / problem	Failure category	What I missed	Revisit date
Example: sliding window	Wrong invariant	Window must maintain at most K distinct values	+1 day, +7 days
Example: graph BFS	Edge case	Disconnected components not visited	+1 day, +7 days
Example: DP	State definition	State did not include previous choice	+1 day, +7 days

Core CS concepts to master

OS • DBMS • NETWORKS • OOP

Operating systems

- Process vs thread; context switching; user vs kernel mode; system calls.
- Concurrency, race conditions, critical sections, mutexes, semaphores and deadlocks.
- Virtual memory, paging, page faults, stack vs heap, scheduling and basic file systems.

Database management systems

- Relational model, primary/foreign keys, constraints, normalization and denormalization.
- SQL joins, grouping, subqueries, indexes, query plans and transactions.
- ACID, isolation levels, dirty/non-repeatable/phantom reads and basic locking.

Computer networks

- TCP vs UDP, HTTP/HTTPS, DNS, TLS, sockets and common status codes.
- Request-response lifecycle, cookies, sessions, JWT trade-offs and CORS.
- Latency, throughput, timeouts, retries, idempotency and load balancing basics.

Object-oriented programming

- Encapsulation, abstraction, inheritance and polymorphism with practical examples.
- Composition vs inheritance, interfaces, constructors, access control and SOLID basics.
- Explain when a simple function or data structure is better than an elaborate class hierarchy.

Interview standard: explain concepts in your own words, give an example, identify a trade-off and connect the concept to a real system. Avoid reciting textbook definitions without application.

Backend and practical engineering

BUILD EVIDENCE BEYOND CODING PLATFORMS

Backend foundations

- HTTP methods and semantics; REST resource design; request validation and consistent error responses.
- Authentication vs authorization; password hashing; sessions/tokens; role-based access control.
- Database schema design, migrations, transactions, indexes and ORM query behaviour.
- Pagination, filtering, sorting, rate limiting, caching and background jobs.
- Logging, structured errors, configuration management and secrets handling.

Testing and quality

- Unit tests for business logic; integration tests for database/API behaviour; API contract tests.
- Test success, invalid input, missing records, duplicate requests, permission failures and boundary values.
- Use linting/formatting, meaningful commits, pull requests and a concise code review checklist.

Deployment basics

Be able to deploy a small application, configure environment variables, connect a database, inspect logs and diagnose a failed request. Know the difference between local development and production configuration. You do not need to become a cloud architect during this six-month preparation period; you do need to understand how your project runs outside your laptop.

AI coding tools

AI assistants can accelerate debugging and scaffolding, but they cannot replace your understanding. Be prepared to explain every important line, test generated code, identify security risks and describe what you changed. Never claim ownership of code you cannot defend in an interview.

Build one interview-grade project

DEPTH BEATS A PILE OF CLONES

Choose one project that demonstrates engineering decisions, not just a tutorial interface. Examples: a job-application tracker with reminders; a collaborative task/workflow API; a small analytics dashboard with ingestion and filtering; or a reliable service with authentication, search and background processing.

Minimum quality checklist

- Clear problem statement and target user; explain why the project exists.
- Backend API with meaningful data model, validation and authorization.
- Database schema that supports real queries; include indexes where justified.
- At least 10–15 useful automated tests across success and failure paths.
- Pagination, filtering, sorting and consistent API errors where relevant.
- Deployment link or reproducible local setup; no exposed secrets.
- README with architecture sketch, setup, environment variables, API examples and trade-offs.
- Git history that shows iterative work; issue list or roadmap for future improvements.
- Short demo video or screenshots; explain your own contribution precisely.

How to make it interview-ready

Prepare a 90-second overview, a 5-minute technical walkthrough and a 10-minute deep dive. Explain one difficult bug, one design trade-off, one performance improvement and one feature you would redesign. If you used an existing codebase, clearly distinguish your work from inherited code.

Avoid these weak signals

- A generic CRUD app with no tests, no deployment and no meaningful edge cases.
- A project description full of buzzwords that you cannot explain.
- Claiming millions of users, production scale or performance gains without evidence.
- Three unfinished projects instead of one polished and defensible project.

Resume, GitHub and application assets

MAKE THE FIRST 20 SECONDS COUNT

One-page internship resume

- Header: professional email, phone, city, LinkedIn, GitHub and portfolio.
- Education: degree, institution, expected graduation year and CGPA if competitive or required.
- Skills: group languages, frameworks, databases, tools and core concepts; list only defensible skills.
- Projects: 2–3 strong entries with action, technical method and verifiable outcome.
- Experience: explain scope, ownership, collaboration and impact without inflating title or duration.

Bullet formula

Action + technical implementation + context + evidence. Example: 'Implemented role-based API permissions and validated multi-step workflow submissions in Django; added automated tests for invalid transitions and duplicate submissions.' Replace the example with facts from your own work.

GitHub hygiene

- Pin the 3–4 repositories that best demonstrate current ability.
- Remove secrets, credentials, personal data and accidental environment files.
- Ensure each public repository has a useful README and a working setup.
- Private repositories are fine; provide a demo, screenshots or a sanitized technical case study when appropriate.

Application packet

Keep a PDF resume, editable master resume, transcript if required, portfolio link, project demo, short cover note and a clean application tracker ready. Tailor the top third of the resume to the role without changing facts.

Interview rounds and how to prepare

FROM OA TO FINAL CONVERSATION

Round	Typical evaluation	Preparation approach
Resume screen	Eligibility, relevance, evidence, graduation year.	Match every requirement; make project bullets concrete.
Online assessment	DSA, coding speed, accuracy; sometimes aptitude.	Timed practice; read constraints; test edge cases.
Technical interview 1	One or two coding problems; reasoning and communication.	Clarify, derive, code, test, analyze complexity.
Technical interview 2	Harder DSA, practical coding, project or CS fundamentals.	Practice follow-ups and alternative approaches.
Project / hiring manager	Ownership, decisions, teamwork, learning ability.	Explain architecture, bugs, trade-offs and your contribution.
HR / recruiter	Availability, location, duration, communication.	Be accurate about dates, work mode and academic commitments.

Coding interview communication loop

9. Restate the problem and ask focused clarifying questions.
10. Discuss brute force, then optimize using constraints.
11. Explain the invariant / algorithm before coding.
12. Write readable code and narrate important decisions.
13. Dry-run sample and edge cases; fix issues openly.
14. State complexity and mention possible improvements.

Behavioural stories

Prepare concise STAR stories (Situation, Task, Action, Result) for a difficult bug, a disagreement, a missed estimate, learning a new tool, taking ownership and receiving feedback. Use genuine examples. A student project or freelance task is acceptable if your role is described accurately.

The 24-week roadmap

A 24-WEEK EXECUTION PLAN

Week	DSA focus	Engineering / career output
1	Complexity analysis; arrays and strings foundations	Baseline DSA assessment; choose one interview language; audit graduation-year eligibility, CGPA and resume.
2	Hashing, frequency counting and prefix sums	Choose one project problem; define target user, requirements, scope and success criteria.
3	Two pointers and sorting fundamentals	Design database entities and API outline; set up Git repository, issues and README.
4	Sliding window and subarray patterns	Build project skeleton, development environment and first API endpoints.
5	Binary search, lower/upper bounds and answer-space search	Implement request validation, consistent errors and basic logging.
6	Intervals, custom sorting and searching revision	Add authentication and authorization; revise OOP fundamentals and write initial unit tests.
7	Linked lists: reversal, middle, cycle and merge	Study DBMS keys, normalization, joins and SQL practice; add project data models.
8	Stacks, queues, monotonic stack and deque	Add integration tests; practise explaining API and database design.
9	Trees, recursion, traversals and height/diameter	Study OS processes, threads, context switching and system calls; improve project reliability.
10	BST, validation, LCA and tree problem patterns	Study concurrency, mutexes, semaphores and deadlocks; document one difficult bug and its fix.
11	Heaps, priority queues, top-K and k-way merge	Study networking: TCP/UDP, HTTP/HTTPS, DNS and request-response lifecycle.
12	Greedy patterns, scheduling and interval problems	Deploy the first complete project version; configure environment variables and inspect logs.
13	Graph representation, BFS/DFS and connected components	Study SQL indexes, transactions, ACID and isolation levels; add database-focused tests.
14	Cycle detection, topological sorting and shortest paths	Study HTTP APIs, status codes, sessions/JWT, CORS, retries and idempotency.
15	Backtracking: subsets, permutations and pruning	Add pagination, filtering, sorting or background processing where relevant; perform security review.
16	Dynamic programming foundations: state, transitions and memoization	Study virtual memory, paging, scheduling and file-system basics; prepare concise OS notes.
17	1D DP: climbing stairs, house robber and subsequence patterns	Improve project test coverage; target 10–15 meaningful automated tests and update README.
18	2D/grid DP, knapsack and sequence DP patterns	Review networking and OOP; prepare architecture sketch, setup guide and API examples.
19	Tries, bit manipulation and XOR patterns	Polish resume bullets with evidence; pin best GitHub repositories and prepare a project demo.
20	Mixed DSA revision: arrays through graphs	Complete two timed online assessments; analyse errors and update the mistake log.
21	Mixed DSA revision: backtracking, DP, greedy and advanced patterns	Company-specific preparation; practise project deep dive and behavioural STAR stories.
22	Timed medium sets; weak-topic repair and independent re-solves	Run two mock technical interviews; revise OS, DBMS, networks and OOP using scenario questions.

23	Full-length OA simulations; revisit problems after 1, 7 and 21 days	Apply to eligible live roles, follow up professionally and tailor resume to each genuine fit.
24	Final consolidation: mixed problems, edge cases and explanation practice	Final readiness review; refine interview stories, review application funnel and plan the next recruiting cycle.

The plan assumes roughly 20–25 hours per week. If you have less time, preserve the order and reduce the number of problems rather than skipping revision and project quality.

Weekly timetable and daily operating system

MAKE PREPARATION SUSTAINABLE

Day	Core session (90–120 min)	Secondary session (45–60 min)
Monday	New DSA pattern + 2 problems	Core CS notes / recall
Tuesday	DSA practice + re-solve	Project implementation
Wednesday	Timed DSA set	DBMS / OS practice questions
Thursday	New pattern + 2 problems	Project tests / debugging
Friday	Mixed revision	Networking / OOP
Saturday	Mock OA or interview	Project deep work + resume
Sunday	Review mistakes and weak areas	Applications, alerts and weekly planning

Daily 3-block structure

- Block A — learn and solve: one concept, one worked example, then independent implementation.
- Block B — build: make one small project improvement, test it and commit it.
- Block C — recall and communicate: explain one CS concept aloud and review yesterday's mistakes.

Weekly scorecard

Metric	Target / evidence
DSA	12–18 quality problems; 70%+ independent re-solve target over time.
Revision	All missed problems scheduled and revisited.
Project	At least one meaningful merged/committed improvement.
Core CS	One topic summary + 10 oral questions.
Interview	One mock or recorded explanation.
Applications	Check official alerts; apply to every genuine fit.

Do not chase a perfect daily streak. A missed day is a scheduling problem, not a reason to abandon the week. Protect sleep and classes; consistency over 24 weeks beats unsustainable bursts.

Application strategy and tracking

FIND ROLES EARLY • APPLY ACCURATELY

Where to search

- Official company career portals: primary source for role status and eligibility.
- College placement / internship cell: campus-only drives may not be publicly visible.
- LinkedIn Jobs and company talent communities: discovery and alerts.
- Wellfound, Unstop and Internshala: supplementary discovery; verify employer identity and final application URL.
- Research and open-source programmes: separate category; check whether they are employment, fellowship or contribution programmes.

Search strings

"Software Engineer Intern" India; "SDE Intern" 2027; "Backend Engineer Intern" Bengaluru; "University Intern" India; "Software Development Intern" Hyderabad; "Class of 2028" internship; "Summer 2027" software engineering.

Tracker columns

Company / role	Posted / deadline	Eligibility	Status / next action
Company A — SWE Intern	Date from live posting	Graduation year, degree, duration	Saved / applied / OA / interview / rejected
Company B — Backend Intern	Rolling; verify weekly	Location, work mode, semester left	Follow-up date and contact source

Safety and professionalism

- Never pay an application, interview, offer-letter or background-verification fee to a supposed employer.
- Verify the domain and job ID; be cautious with WhatsApp-only recruiters and personal email addresses.
- Never share OTPs, banking credentials or unnecessary identity documents.
- Do not spam employees. Send one concise, personalized referral request with role ID and resume.

A referral can help a profile get noticed; it does not replace eligibility or interview performance.

Readiness assessment and final checklist

KNOW WHEN YOU ARE READY TO INTERVIEW

Self-assessment rubric

Area	Not ready	Interview-ready evidence
DSA	Needs hints for common patterns.	Solves unseen medium problems independently and explains complexity.
Language	Syntax and STL/library use slow.	Writes clean code without searching basic syntax.
Core CS	Definitions only; cannot apply.	Explains scenarios, trade-offs and examples.
Project	Only demo works on author's machine.	Deployed/reproducible, tested, documented and defensible.
Resume	Vague claims and unclear role.	Every bullet is specific, truthful and verifiable.
Communication	Jumps into code without clarifying.	Structured reasoning and calm correction of mistakes.

Before every interview

- Re-read the job description and verify role, location, duration and eligibility.
- Test the coding environment and confirm your chosen language.
- Review 5–8 recent mistakes and key patterns, not hundreds of random notes.
- Prepare a 90-second introduction and two project stories.
- Have questions ready about team scope, mentorship, project expectations and evaluation.
- After the interview, record topics and gaps while the discussion is fresh.

If no offer arrives by the end of 24 weeks

Diagnose the funnel rather than restarting from zero. No callbacks usually points to eligibility, resume targeting, timing or application volume. OA failures point to speed, accuracy and pattern coverage. Interview failures point to reasoning, communication, CS depth or coding quality. Fix the bottleneck, keep applying and continue revising. A 24-week plan builds readiness; hiring cycles may take longer.

Research sources, links and editorial notes

PRIMARY SOURCES FIRST • LAST CHECKED 23 NOV 2026

The company calendar in this guide prioritizes official employer sources. Third-party internship calendars are useful for discovering dates but can contain stale, region-mismatched or unverified listings. Always click through to the employer's current requisition.

Primary employer sources

- **Google Careers — Student Researcher 2027 India —** https://www.google.com/about/careers/applications/jobs/results/125423735539671750-student-researcher-2027?hl=en&location=India&q=Internship
- **Microsoft — Internship eligibility —** https://careers.microsoft.com/v2/global/en/internship_eligibility
- **Microsoft — Explore Microsoft —** https://careers.microsoft.com/v2/global/en/exploremicrosoft
- **Amazon Jobs — SDE I Intern, 2027 India —** https://www.amazon.jobs/pt/jobs/10541345/sde-i-intern-amazon-university-talent-acquisition
- **Qualcomm — India internships and dates —** https://www.qualcomm.com/company/careers/internships-and-early-career-opportunities/india
- **Atlassian — Early careers —** https://www.atlassian.com/company/careers/earlycareers
- **Salesforce — Summer 2027 Software Engineer Intern —** https://careers.salesforce.com/en/jobs/jr337715/summer-2027-intern-software-engineer/
- **Oracle — Development & Engineering internships —** https://www.oracle.com/careers/students-grads/development-engineering/internships/
- **Goldman Sachs — 2027 Summer Analyst India —** https://www.goldmansachs.com/careers/students/programs-and-internships/india/summer-analyst-program/
- **Meta Careers —** https://www.metacareers.com/
- **Apple Jobs —** https://jobs.apple.com/
- **NVIDIA Careers —** https://www.nvidia.com/en-us/about-nvidia/careers/
- **Adobe Careers —** https://careers.adobe.com/
- **Uber Careers —** https://www.uber.com/us/en/careers/
- **LinkedIn Careers —** https://careers.linkedin.com/
- **Walmart Careers —** https://careers.walmart.com/
- **Flipkart Careers —** https://www.flipkartcareers.com/

Research and editorial limitations

Snapshot date: 23 November 2026. The Amazon, Google, Microsoft, Qualcomm, Atlassian, Salesforce, Oracle and Goldman Sachs programme details above were checked against publicly accessible pages in this research pass. For companies where a specific 2027 India undergraduate role was not verified, the guide explicitly says so rather than inventing a date. Recruitment calendars change; a page may be updated or removed after publication.

This document is an independent educational resource, not legal, immigration or career-placement advice. Company names and trademarks belong to their respective owners. No employer has sponsored or approved this guide. Readers should confirm all requirements directly with the employer.

Suggested update policy for a sellable edition

Label this edition 'Research snapshot: 23 November 2026'. Before reselling an updated edition, re-check every live job ID, application deadline, graduation-year rule, programme duration and official URL. Replace expired postings with current ones and preserve a dated change log. Never market historical windows as guaranteed future dates.