UnCodemy

B.Tech Skills by Semester Checklist PDF

Spread your skill building across all eight semesters: learn one programming language and Git in the first year, data structures and core subjects in the second, a career track with real projects and an internship in the third, and placement preparation with a strong final project in the fourth. Add communication, LinkedIn and GitHub along the way. Steady work each semester beats a panic in final year.

Key takeaways

- First year is for one language, logic building and good study habits, not ten courses at once.
- Data structures and algorithms plus core CS subjects (DBMS, OS, networks, OOP) form the base for most tech interviews.
- By the end of third year, aim for one career direction, two or three solid projects and at least one internship or equivalent.
- Communication, a clean resume, LinkedIn and GitHub are skills too; build them early.
- Non-CS branches can follow the same plan, swapping in core tools and projects from their field.

How to use this checklist

Each semester has a short list of skills and outcomes. Tick them off as you go. If you are behind, do not panic. Start from the semester you are in and cover the earlier essentials during breaks.

The plan assumes about one to two hours a day on skills outside your college syllabus during term, and more during breaks. Adjust it to your own pace.

Semester 1: Foundations

Goal: build logic and study habits.

- Learn one programming language well: C, C++, Python or Java. Use the one your college teaches unless you have a strong reason.
- Solve 50 to 80 easy problems on loops, conditions, functions, arrays and strings.
- Learn to use a code editor, the terminal and basic Linux commands.
- Create GitHub and LinkedIn accounts.
- Build a daily revision habit for college subjects.

Outcome: you can write small programs without copying.

Semester 2: Tools and first project

Goal: start building.

- Learn Git: commit, branch, merge, push, pull requests.
- Learn basic HTML, CSS and JavaScript, or Python scripting, depending on interest.
- Build your first small project, for example a calculator app, a quiz app or a personal website.
- Put it on GitHub with a README.
- Join one college club or community.

Outcome: one project online, comfortable with Git.

Semester 3: Data structures begin

Goal: core problem-solving.

- Start data structures and algorithms (DSA): arrays, strings, hashing, linked lists, stacks, queues, recursion, sorting and searching.
- Solve problems regularly, about 4 to 5 a week.
- Learn object-oriented programming concepts in your language.
- Start a second project that uses a database (SQL basics).
- Write one technical blog post or LinkedIn post about something you learned.

Outcome: steady DSA practice and a second project.

Semester 4: Core subjects and a direction

Goal: understand the base and explore tracks.

- Continue DSA: trees, binary search, two pointers, sliding window, basic graphs.
- Study core subjects deeply: DBMS, operating systems, computer networks, OOP. These are asked in interviews.
- Try a beginner lesson or mini project in two tracks: web development, data analytics or data science, AI and machine learning, cloud and DevOps, software testing, cybersecurity.
- Take part in one hackathon or coding contest.

Outcome: you have a shortlist of one or two career directions.

Semester 5: Specialise and build

Goal: go deep in one track.

- Pick one track and follow a structured roadmap.
- Build a substantial project in that track: for web, a full-stack app with login, database and deployment; for data, an end-to-end analysis with a dashboard; for AI, a model or LLM-based app solving a real problem.
- Continue DSA at a medium level.
- Make your first one-page resume.
- Apply for summer internships, research internships or open source programs.

Outcome: one strong project and active internship applications.

Semester 6: Internship and second big project

Goal: real-world experience.

- Do an internship in the summer, or a research project, open source contribution or freelance project if you cannot get one.
- Build a second strong project, ideally with one AI feature, testing or cloud deployment.
- Practise aptitude for placement tests: quantitative, logical, verbal.
- Start mock interviews with friends or seniors.
- Update your resume, LinkedIn and GitHub.

Outcome: proof of real work and a placement-ready resume.

Semester 7: Placement season

Goal: convert preparation into offers.

- Revise DSA patterns and solve company-style problems under time limits.
- Revise DBMS, OS, networks, OOP and your projects in depth.
- Prepare HR answers: introduce yourself, strengths, weaknesses, why this company.
- Do aptitude practice tests every week.
- Apply off campus as well as on campus.
- Start your final-year project with a problem you care about.

Outcome: active interviews and a final project underway.

Semester 8: Finish strong

Goal: graduate with skills and a plan.

- Complete the final project with a clear report and demo.
- Keep applying and interviewing if you do not have an offer.
- If you have an offer, learn the tools your company uses.
- If you plan higher studies, finish exams, applications and documents.
- Collect recommendation letters from faculty and internship mentors.

Outcome: degree, project, job or admission, and a learning plan for the next year.

Skills that run through every semester

Skill	How to build it
Communication	Present in class, join a club, explain projects to friends
Writing	READMEs, LinkedIn posts, project reports
English and interviews	Speak in mock interviews, record yourself
Time management	Weekly plan, fixed study slots

Skill	How to build it
Teamwork	Group projects, hackathons, club events
Learning with AI tools	Use AI assistants to explain and review, but write and understand the code yourself

Notes for non-CS branches

Mechanical, civil, electrical, electronics and other branches can follow the same structure:

- Replace web or data projects with branch tools: CAD and simulation software, MATLAB, circuit simulators, PLC programming, embedded systems.
- Keep programming basics and one data skill (Excel, Python or SQL); they help in almost every engineering job.
- Look at GATE, PSUs and core company internships from semester 5.
- If you want to switch to IT, follow the CS plan above from semester 3 with extra time.

If you are behind

- **In semester 5 with no projects:** pick one track, build one project in six to eight weeks, and practise DSA alongside.
- **In semester 7 with weak DSA:** focus on the most common patterns and core subjects first, and practise every day.
- **No internship:** do an open source contribution, a freelance project or a strong self-driven project with real users.

A note on certifications

Short online certificates are not a substitute for these skills. Use courses to learn, then prove the learning with projects, contest results or graded exams such as NPTEL. Two or three relevant, graded certifications add value; twenty short course badges do not.

Printable checklist summary

- One language mastered by end of semester 2
- Git and GitHub by semester 2
- Two projects by semester 4
- DSA basics to medium by semester 5
- One career track chosen by semester 5
- One internship or equivalent by semester 6
- Resume, LinkedIn and GitHub updated every semester
- Placement-ready by semester 7

Review every semester

At the end of each semester, spend one hour reviewing this checklist. Mark what you finished, note what slipped, and move unfinished items into the next semester's plan. Update your resume, LinkedIn and GitHub in the same sitting, so your profile always matches your real progress.

Frequently asked questions

What should I learn in the first year of B.Tech?

Learn one programming language well, basic problem solving, Git, and good study habits. Build one small project and start your GitHub and LinkedIn profiles.

When should I start DSA in B.Tech?

Most students start in the third semester after learning one language. Practise regularly through semesters 4 to 7.

How many projects should I have before placements?

Two or three solid projects in your chosen track, at least one deployed or easy to demo, are more useful than many small tutorial projects.

Is this checklist useful for non-CS branches?

Yes. Keep the structure and swap in tools and projects from your branch, plus basic programming and one data skill.

What if I am already in third year and have done nothing?

Pick one track, build one strong project in six to eight weeks, practise DSA and core subjects daily, and apply for internships. It is late but very doable.

Is this PDF free?

Yes. Fill in the short form on this page and the PDF downloads right away.

Sources and further reading

- AICTE Model Curriculum for Undergraduate Degree Courses in Engineering
- GitHub Skills: learn Git and GitHub
- roadmap.sh: developer roadmaps
- AICTE Internship Portal

About this PDF

Prepared by the Uncodemy career team from the public sources listed. We update it every January. Corrections: info@uncodemy.com.

More free resources and courses: uncodemy.com | info@uncodemy.com | +91 98183 66550