Codecov

Testing Milestones: A Checklist for Every Stage

Testing Milestones: A Checklist for Every Stage

A lot of development teams start off in scrappy startup mode, shipping code fast and fixing bugs as they come. At this stage, shipping perfectly tested, super-stable code may not be as important as getting to MVP as quickly as possible.

But eventually, teams usually come to a point when they need to start testing. It could be because of industry regulations or because time spent fixing bugs has become excessive and it's time to get more efficient. Either way, when you're starting with a large, untested codebase, getting it all covered can look like a daunting process, so we made a checklist to help.

See what stage you're at by looking at the starting points and main challenges. Then, use the checklist to move through your testing journey.

> *"40% of our engineering capacity was going to fixing things that were broken. Codecov has been a critical tool in reducing that trend to around 10%."*
>
> **– Ian Arko, Head of Engineering, Axle Health**

Start here — choose the path that best fits your team

	Build a foundation	***Scale & add rigor***	***Streamline & optimize***
Starting point	"We're barely testing, and we're certainly not tracking it."	"We test some of our code, but we need to do more."	"Testing is a way of life, now it's time to perfect our process."
Main challenges	• Constantly de-bugging prod • Devs don't want to write tests	• Prod still buggy • Testing process is too manual	• Slow or flaky tests impacting efficiency • Unit tests missing more complex issues
Main focus	• Build testing habits • Measure coverage	• Integrate testing into workflows • Improve code quality	• Maximize patch and project coverage • Improve test suite performance

Build a foundation

If your organization is starting from scratch or has minimal testing in place, start here. Set up basic tools and build habits.

- [ ] **Set Basic Goals**
  - [ ] Identify critical code areas to test first (e.g., login functionality, APIs).
  - [ ] Set a realistic coverage baseline, such as 30% on new code.
- [ ] **Implement Core Tooling**
  - [ ] Choose a testing framework based on your stack (e.g., Pytest, JUnit, Mocha).
  - [ ] Install Codecov to measure and track code coverage over time.
  - [ ] Integrate automated testing into your CI/CD pipeline (e.g., GitHub Actions, CircleCI).
- [ ] **Encourage Developer Buy-In**
  - [ ] Add "Write unit tests" to your team's definition of done for every task. Make sure their tests are isolated to a single piece of code without external dependencies.
  - [ ] *Celebrate small wins*: Publicly acknowledge developers who write tests that improve coverage or catch bugs early.
- [ ] **Automate Feedback Loops**
  - [ ] Enable Codecov's Pull Request Comments to help devs get used to tracking coverage.
  - [ ] Enable Codecov's Informational Status Checks to surface test coverage changes during reviews.

Scale testing and add rigor

Now that you have some testing in place, you can improve quality, reduce risks, and integrate testing into workflows.

- [ ] **Expand Test Coverage**
  - [ ] Target 50-70% coverage on new code.
  - [ ] Add integration tests for workflows that span multiple components (e.g., submitting forms, API/database interactions).
  - [ ] Use Codecov's Flags or Components to focus on critical modules or uploads (e.g., checkout, auth).
- [ ] **Address Test Flakiness**
  - [ ] Use Codecov's Test Analytics to identify flaky tests and prioritize fixes.
  - [ ] Quarantine or rework flaky tests to avoid CI disruptions.
- [ ] **Set and Enforce Standards**
  - [ ] Configure Codecov's Status Checks to enforce a minimum threshold (e.g., 50%).
  - [ ] Notify developers when coverage drops using integrations such as Slack.
- [ ] **Leverage AI Tools**
  - [ ] Use Codecov's AI PR Review to get through the review cycle faster.
  - [ ] Use Codecov's AI test generation to automate unit test coverage.

Become a certified testing boss

Congratulations! Testing is now a way of life, and you can work toward comprehensive coverage while honing scalability and efficiency.

- [ ] **Optimize Test Coverage**
  - [ ] Target 80–90% coverage for critical paths (e.g., checkout workflows, admin tools).
  - [ ] Use Codecov's historical data to identify recently updated areas with low coverage.
- [ ] **Implement Advanced Test Types**
  - [ ] Add end-to-end tests for key user workflows using tools like Cypress or Playwright. (e.g., a full e-commerce user flow, from shopping through payment confirmation)
  - [ ] Include performance and security testing as part of CI/CD pipelines.
  - [ ] Run mutation testing tools to identify gaps in your test suite.
- [ ] **Improve Test Suite Performance**
  - [ ] Regularly review test performance metrics in the Codecov Test Analytics *dashboard*.
  - [ ] Update or replace tests with high flake rates or excessive run times.
- [ ] **Promote Cross-Team Collaboration**
  - [ ] Host quarterly retrospectives with QA, DevOps, and product teams to align on quality goals.
  - [ ] Use Codecov's Dashboards to report progress and celebrate achievements across teams.

Codecov
BY SENTRY