Chapter 7: How to Know It Actually Works

> *"You're not writing tests. You're being thorough."*

The Reality

AI built your app. It works on your computer. But how do you know it'll work for everyone else?

You don't need unit tests or test-driven development. You need to be systematic about checking if things work before real users find the problems.

This chapter teaches you: How to catch obvious issues before launch, and build confidence that your app is ready.

The Three Testing Phases

Phase 1: Before You Deploy

Test locally. Catch the obvious stuff.

Phase 2: After You Deploy (Before Telling Anyone)

Test live. Make sure deployment didn't break anything.

Phase 3: When You Add Features

Test the new thing and make sure you didn't break the old things.

Phase 1: Before You Deploy

The Happy Path Test

What it is: Use your app exactly as you expect users to use it.

For the bookmark app:

1. Sign up with email/password
2. Add a bookmark
3. See it in your list
4. Search for it
5. Delete it
6. Log out

Do this 3 times. If something breaks on attempt 2 or 3, you found a bug.

The "What If I'm Dumb?" Test

What it is: Try to break things like a confused user would.

Try these:

- Leave fields empty, hit submit
- Type weird stuff (`<script>` , emoji, super long text)

- Click submit 10 times rapidly
- Hit back button randomly
- Refresh mid-action

Look for:

- Helpful error messages (not "Error 500")
- App doesn't crash
- Can't get "stuck" in a broken state

The Device Test

What it is: Test on different devices and screen sizes.

Minimum tests:

- Your phone (small screen)
- Your computer (large screen)
- Try both portrait and landscape on phone

Check:

- Can you see all the buttons?
- Can you tap/click everything?
- Does text overflow weirdly?
- Are forms usable on mobile?

The Browser Test

What it is: Test in multiple browsers.

Test these:

- Chrome (most users)
- Safari (iPhone users)
- Firefox (privacy-conscious users)

Open incognito/private mode each time (clears cache and cookies).

The Slow Internet Test

What it is: See what happens when things take time.

How:

- Chrome DevTools → Network tab → Throttle to "Slow 3G"
- Do your happy path test again

Check:

- Do you see loading states?
- Does anything feel "broken" because it's slow?
- Are there helpful messages while waiting?

Phase 2: After You Deploy

You deployed to Vercel/Netlify. Before you tell anyone:

The Deployment Check

Visit your live URL. Test again:

- Sign up with a NEW email (test email sending)
- Do the happy path test
- Try a few "dumb user" tests

Why this matters: Environment variables might be wrong. Database might not be connected. Email might not send. Things that work locally can fail in production.

The Fresh Eyes Test

Ask 1-2 friends to try it. Don't tell them how to use it.

Watch them:

- Where do they get confused?
- What do they click that doesn't work?
- What do they expect that isn't there?

Take notes. Users will try things you never imagined.

The Real Device Test

Test on actual devices, not just Chrome's device simulator:

- Your phone
- Friend's Android phone (if you have iPhone)
- Tablet if you have one

Simulators lie. Real devices show real problems.

Phase 3: When You Add Features

Every time you add something new:

The New Feature Test

1. **Test the new feature** thoroughly
2. **Test the old features** to make sure you didn't break them

Example: You added "export bookmarks to CSV"

Test the new thing:

- Export with 1 bookmark
- Export with 100 bookmarks
- Export with 0 bookmarks
- Does the file download?

- Can you open it in Excel/Google Sheets?

Test the old things:

- Can you still add bookmarks?
- Can you still delete bookmarks?
- Does search still work?

AI makes changes across multiple files. It might break something you didn't expect.

Common Problems You'll Catch

Forms That Don't Validate

- Submit empty fields
- Paste 10,000 characters in a field
- Use special characters

Broken Navigation

- Click back button
- Type URL directly
- Try to access pages you shouldn't

State Management Issues

- Do the same action twice
- Refresh in the middle of something
- Open app in two tabs

Mobile Issues

- Buttons too small to tap
- Text overlaps
- Keyboard covers input fields
- Can't scroll properly

Loading State Problems

- No indication something is happening
- Looks broken when it's just slow
- Can click buttons while loading (creates duplicates)

Tools That Help

Chrome DevTools (Free)

Console tab: Shows errors in red. Copy them, paste to AI, say "fix this"

Network tab: Shows what requests are happening. Useful for seeing if API calls fail

Device simulation: Test different screen sizes without multiple devices

How to open: Right click anywhere → "Inspect"

Actual Phones

Nothing beats real devices. Test on actual phones. Borrow friends' phones if needed.

iOS vs Android: They behave differently. Test both if possible.

When to Stop Testing

You'll never catch everything. Real users will always find something.

Ship when:

- Happy path works reliably
- Major "dumb user" scenarios don't crash
- Works on your phone and computer
- 1-2 friends tested it successfully

Don't wait for:

- Every possible edge case
- Perfect mobile experience
- Zero bugs

Perfect is the enemy of shipped.

What Happens When Users Find Bugs

They will. Here's how to handle it:

1. Thank Them

"Thanks for reporting this! I'm looking into it."

2. Reproduce It

Ask them exactly what they did. Try to make it happen on your end.

3. Fix It

Copy the error, paste to AI, describe what happened, let AI fix it.

4. Deploy the Fix

Push to GitHub, Vercel auto-deploys. Usually live in 2-3 minutes.

5. Tell Them It's Fixed

"Just deployed a fix, should work now. Let me know if you still see issues!"

Users respect builders who fix things fast more than builders who ship perfect products late.

The Testing Mindset

You're Not Writing Tests

You're **clicking around systematically** to catch obvious problems.

You're Not Paranoid

You're **thorough**. There's a difference.

You're Not Trying to Be Perfect

You're trying to **ship with confidence**.

Checklist for Every App You Build

Print this. Keep it by your computer.

Before Deploy:

- [ ] Happy path works (3 times)
- [ ] Empty form fields handled
- [ ] Works on my phone
- [ ] Works in Chrome, Safari, Firefox
- [ ] Tested with slow internet

After Deploy:

- [ ] Works at live URL
- [ ] New signup works (email sends)
- [ ] 1-2 friends tested it
- [ ] Tested on real phone

Adding Features:

- [ ] New feature works
- [ ] Old features still work
- [ ] Tested on phone
- [ ] Deployed and checked live

The Reality Check

Your first app will have bugs. So will your tenth app. Professional developers ship bugs all the time.

The difference: You caught the obvious ones. You tested systematically. You're confident the main flows work.

That's enough to ship.

Testing isn't about perfection. It's about **catching the stuff that would embarrass you** before users see it.

Next Steps

- Build your next feature
- Test it with this checklist
- Ship it
- Fix bugs as users report them
- Learn what breaks and why

The more you build, the better you get at knowing what to test.

This is learned through doing, not through reading. Go test something.

Connect & Share

💌 **Newsletter**: Build to Launch - Weekly AI building tips, templates, and real builder stories

✍️ **Medium**: AI Builders - Read more articles and guides

💬 **Reddit**: r/VibeCodingBuilders - Join the community

🦋 **Bluesky**: @jenny-ouyang - Connect

💼 **LinkedIn**: Jenny Ouyang - Connect