RELEVANT
SOFTWARE

Tips and Tricks **For Testing On Your Own.**

It’s absolutely okay if during development some bugs and errors appear in the product. And sometimes it happens that you do not have QA team or QA engineer on your project to detect them for you before release. And unfortunately, users see them and write about it somewhere public or use for their own benefit, like the billing system flaws.

Even if you don’t have professional testers on board,
there is still a need to verify if your product works as planned and if it’s good enough for your users. Product owner, project manager, developer, whoever is going to make testing should consider at least basic principles of verifications.

Our company cares about the quality of the products we work on. That’s why we gathered few tips on how you can test the product on your own.

Tip #1. Understand what exactly you need to test

Make sure you have answered the essential questions before starting your verifications. Here are some of them:

- Do I know what browsers/OS/platforms/resolutions I have to test on?

- Do I have requirements for performance? (For example: will my product be ok if 1000 users will use it at the same time?)

- Do I have requirements for security?

- What are the severity and priority of the project? (this should help you to prioritize your testing scope)

- Do I know if the product works as expected? (If a tester is not the product owner then you need to make sure this person has a clear understanding/requirements of how it should work/ look)

Feel free to extend the list with more questions that are specific to your project.

Tip #2.
Cover common weak spots

Use boundary value technique for testing:
bugs love to hide on the boundaries. Every partition has its maximum and minimum values. You should cross them and see what happens. (e.g. the form accepts age 18-56, submit 17 and 57 to check if the form validates data right)

Try different screen sizes/devices/platforms:
make sure your interface looks great not only on your phone and laptop. You should ensure that your product works well on every device, platform and in at least most popular browsers.

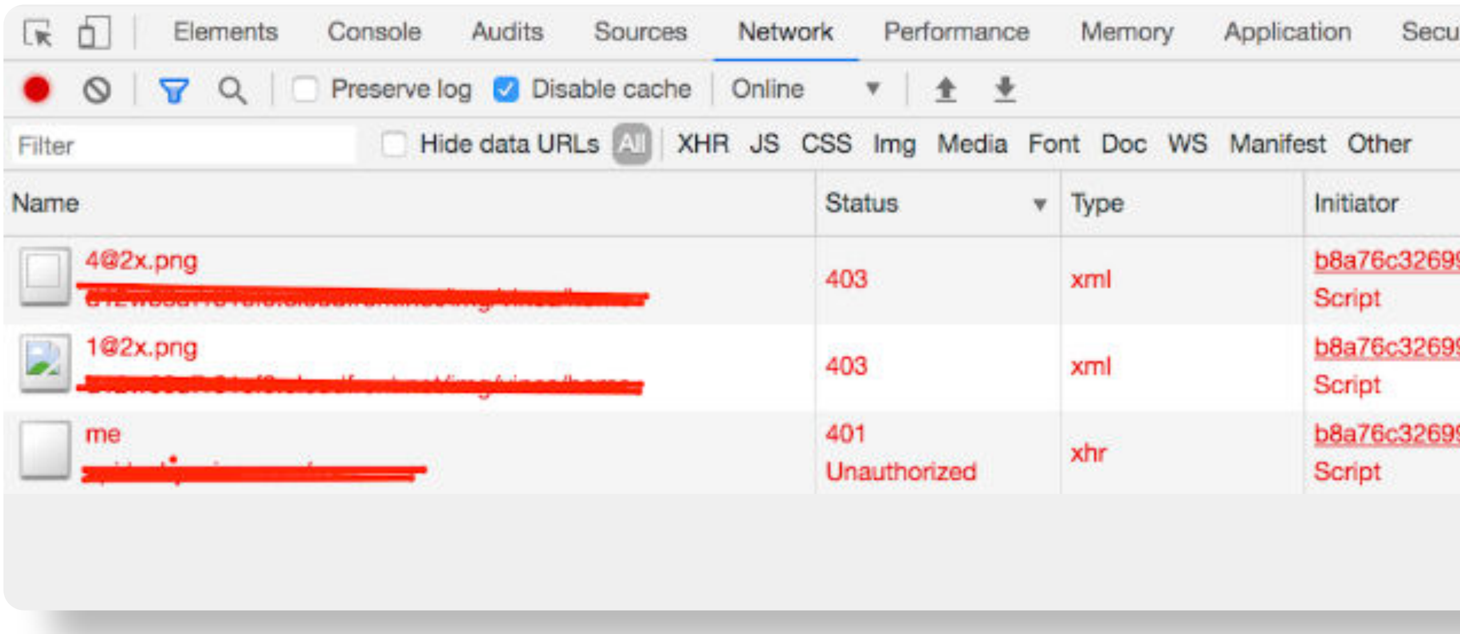
Use simulators if you don't have real devices:
if you have a MacBook, then you can use a native XCode **simulator** to test your application/website on a bunch of apple devices. For the Android platform, we suggest using **Android Studio Native Emulator**

Use developer tools:
It will take a few seconds for you to open a dev tool
(for example in Google Chrome: shortcut Ctrl+Shift+C
(on Windows) or Command+Option+J (on Mac)).

Let’s take the Network tab for example to view unexpected responses:

- It will help you to:
- find bad responses.
- get into details of existing visual bugs
- see performance
- detect 404 errors
- and much more.

Chrome DevTools is the best assistant for your manual testing.
You can learn more about its functionality in **the official documentation.**



[ ] Preserve log [✓] Disable cache Online ▼

[ ] Hide data URLs

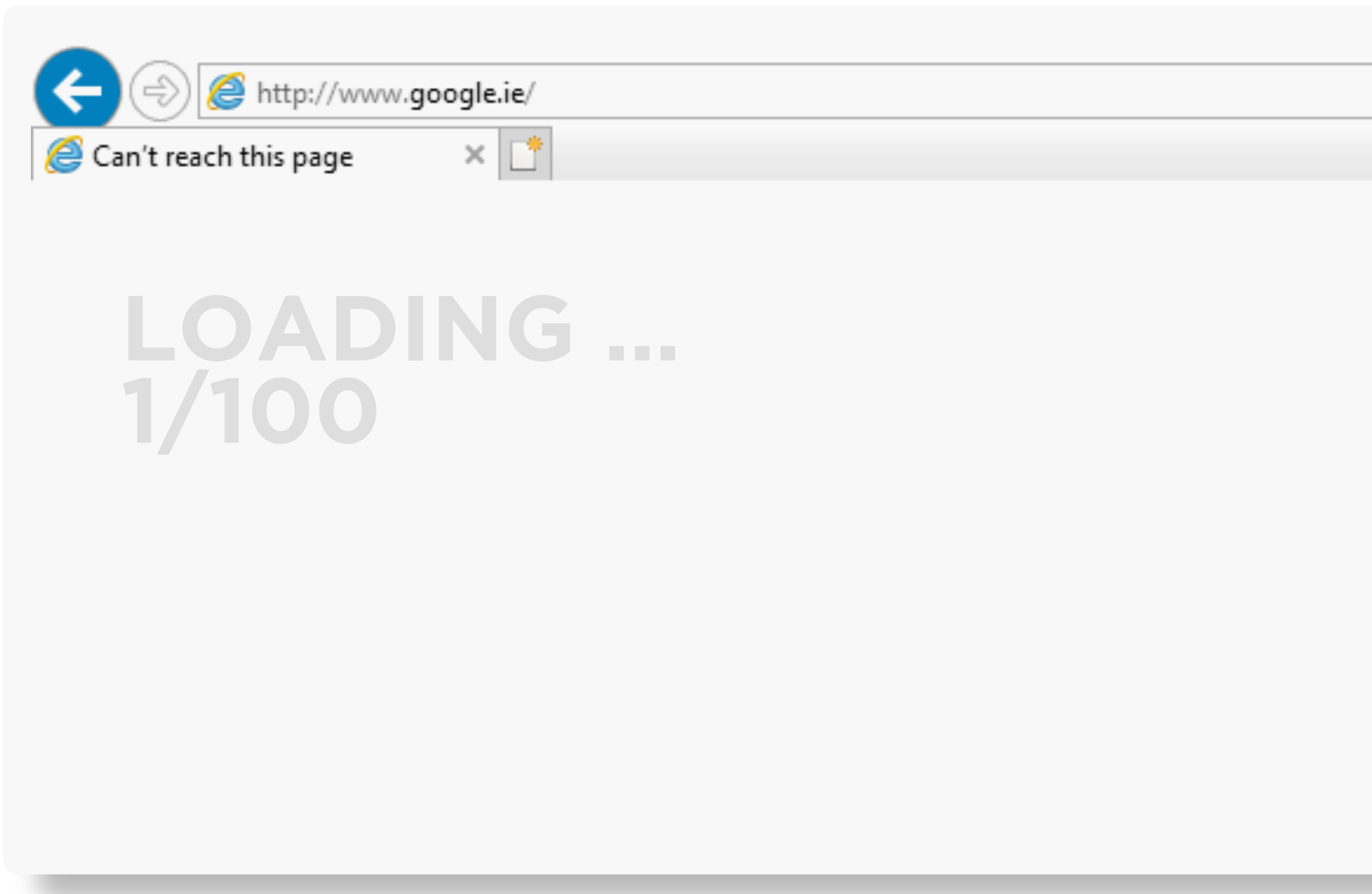
Name	Status	Type	Initiator
4@2x.png [redacted]	403	xml	b8a76c3269 Script
1@2x.png [redacted]	403	xml	b8a76c3269 Script
me [redacted]	401 Unauthorized	xhr	b8a76c3269 Script

Take a look at performance:

After 3 seconds, 40% of new visitors will abandon your site. Use “Performance” tab in the Chrome Dev tool to see how long your pages take to load.

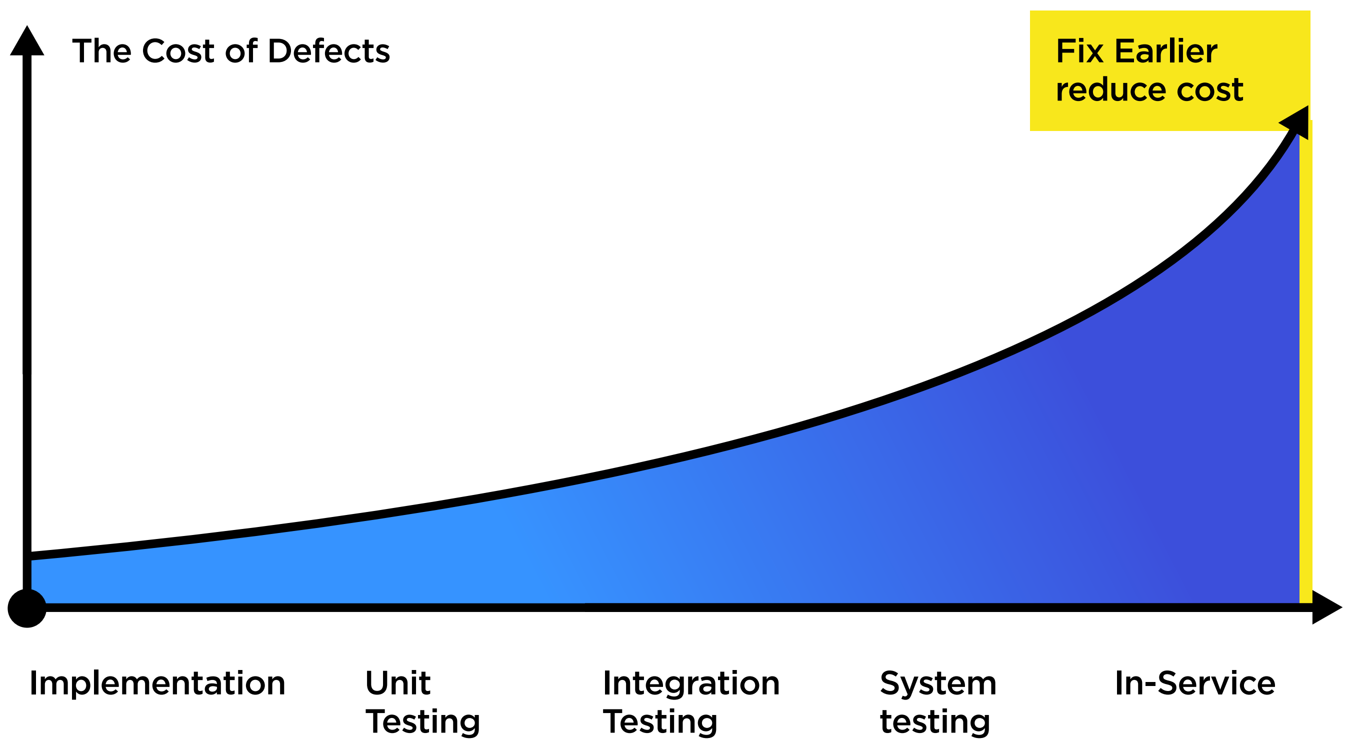
Don’t forget about Internet Explorer:

are you sure you don't have users who are still using IE? IE is different from other browsers and a lot of bugs appear there. Developers hate to support it, but if this browser is still important for your business, make sure your product works in it.



Tip #3.
The earlier testing starts, the lower development costs are

The price of a mistake is decidedly higher if detected at the end of development.



Here is a simple real-life example:

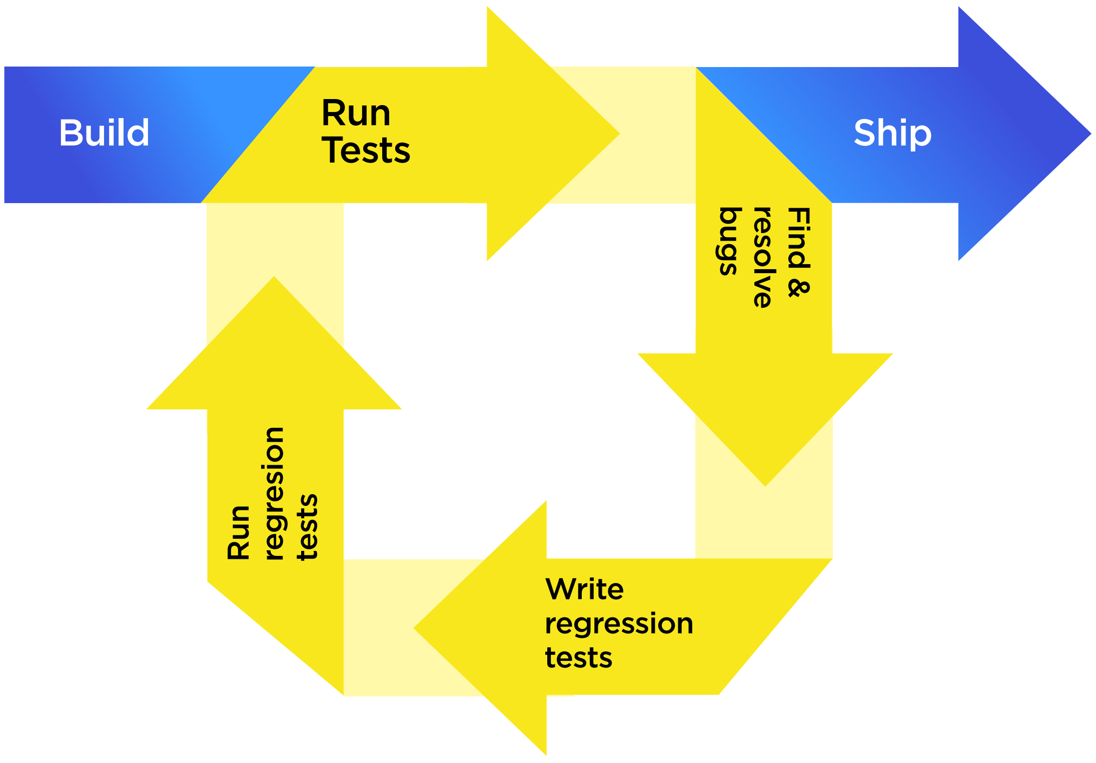
Ambiguous scenario is found on the stage of specifications testing. It is much faster and cheaper to fix this in documentation instead of fixing it when it has passed the implementation phase.

It is a common mistake for product owners to start testing on the late stages before the release. Testing on the early stages prevents the issues that can potentially delay the delivery and mess with the deadlines. That’s why you should start testing as soon as possible.

Tip #4.
Make verifications part of your process

If you are using the Agile approach, try to make testing part of each iteration. Each time a new build is delivered to your local/testing/staging/production/whatever environment do two things:

- Perform a "smoke" test to make sure nothing was broken after adding new features to prevent regression.
- Test all new features.



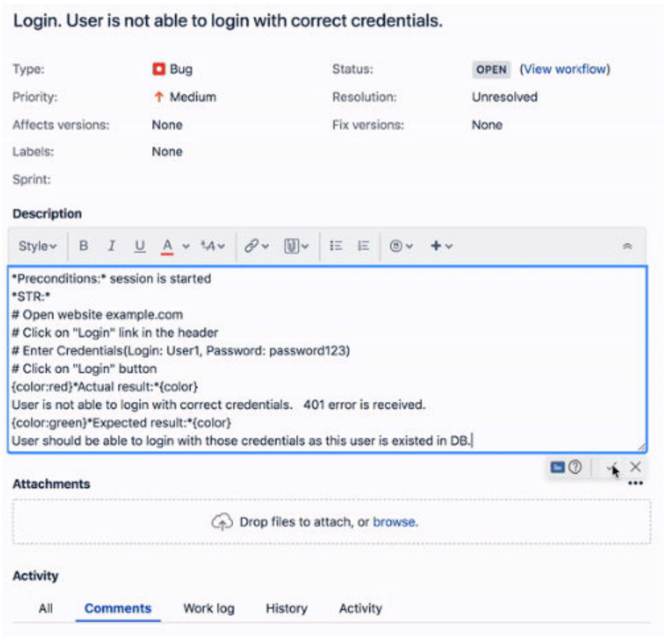
Tip #5.
Make bugs reporting clear

To avoid wasting the time of your developers on clarifications, make sure you report your bugs in a clear and simple way. Templates for bug reporting will unify your bugs and make it super clear for developers to understand. Here is an example of the simple template for JIRA:

```
*Preconditions:*

*SepsToReproduce:*

#

#

{color:red}*Actual result:*{color}

{color:green}*Expected result:*{color}
```

Jira has its own markup language which will turn this template into the simple bug report.

Here how it looks like:



Login. User is not able to login with correct credentials.

Type:	Bug	Status:	OPEN (View workflow)
Priority:	Medium	Resolution:	Unresolved
Affects versions:	None	Fix versions:	None
Labels:	None		
Sprint:			

Description

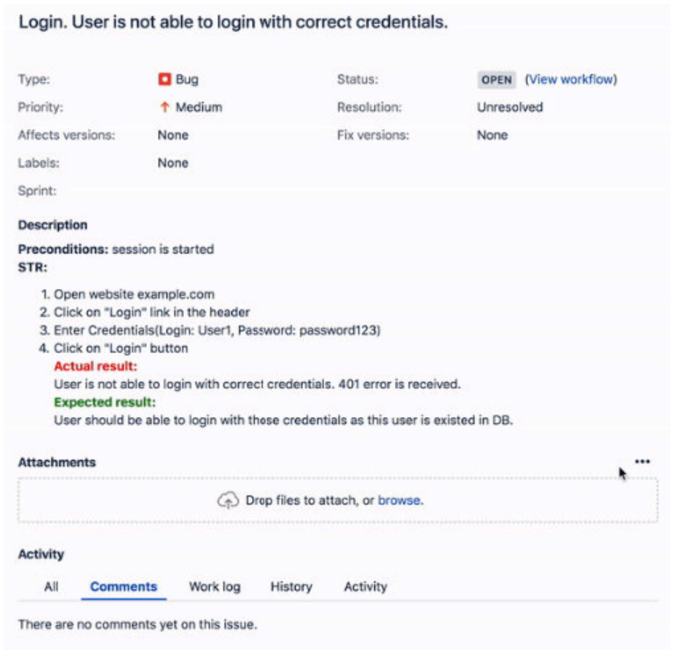
```
*Preconditions:* session is started
*STR:*
# Open website example.com
# Click on "Login" link in the header
# Enter Credentials(Login: User1, Password: password123)
# Click on "Login" button
{color:red}*Actual result:*{color}
User is not able to login with correct credentials.   401 error is received.
{color:green}*Expected result:*{color}
User should be able to login with those credentials as this user is existed in DB.
```

Attachments

Drop files to attach, or browse.

Activity

All Comments Work log History Activity



Login. User is not able to login with correct credentials.

Type:	Bug	Status:	OPEN (View workflow)
Priority:	Medium	Resolution:	Unresolved
Affects versions:	None	Fix versions:	None
Labels:	None		
Sprint:			

Description

Preconditions: session is started
STR:

1. Open website example.com
2. Click on "Login" link in the header
3. Enter Credentials(Login: User1, Password: password123)
4. Click on "Login" button
   Actual result:
   User is not able to login with correct credentials. 401 error is received.
   Expected result:
   User should be able to login with these credentials as this user is existed in DB.

Attachments

Drop files to attach, or browse.

Activity

All Comments Work log History Activity

There are no comments yet on this issue.

Tip #6.
Document test cases

Documenting test cases at least on a high level will help you to:

- Not miss anything on smoke/regression testing
- Make super fast onboarding of outside specialist
- Accelerate test automation with checklist or test cases
- See coverage clearer
- Easier plan your testing if the product is scaling

Tip #7.
Get a QA specialist in your team if the quality is your priority

It is a “gold standard” for leading software development companies to involve QA specialists in all stages of the development, starting from the stage of Requirements.

Here is how professional QA process looks like and how a QA specialist might be helpful at each stage:

Requirements

Requirment specification validation
Project timeline review

Design

Review of design documents
Create the testing project plan
Create high level scenarios

Development

Detailed test plan / test cases
Test Plan sign-off
Unit testing

Testing

Test execution (manual and automated scripts)
Bug reporting / restesting
Management reports creation

Deployment

Upgrade or migration
Acceptance testing
Final testing report
Track user feedback

Maintenance

Automation test script updates
Reproducing consumer’s issuses and indentifing causes

It is a proven fact that QA engineers are able to find “bugs” in the requirements. Moreover, they can help you by just asking the right questions of how the product should work in edge cases, as there is always a chance that some scenarios are not covered in your specifications.

Tip #8.
Get a QA specialist in your team if budget is important

We recommend involving QA engineers for many reasons including saving the budget.

Let’s do simple math:

Usually, testing employs around 25-40% of the development time. Let's suppose a developer is going to spend their time testing cases. Instead of working on the new features of your product, they are preoccupied with tests. Taking into account the fact that the developer's rate is usually 30-40% higher than a QA engineer rate, hiring a QA specialist you are saving 30-40% of the budget.

RELEVANT
SOFTWARE

Take care of your product!

www.relevant.software