CSCI 1320
Creating Modern Web Applications
Lecture 34: Testing
Security Challenge and Testing

• What were you doing in the security challenge
  • Seeing if you could break other’s web sites
  • Seeing if you could break your web site
  • Fixing your code and having it checked again

• Coming up with scripts that try to break it
  • Scripts that can be reused
Testing

• When looking at security and privacy
  • We keep asking “what can go wrong”
  • What happens if a user does <x> when <y>

• You want to do this in general for your application
  • To make sure it will work
  • To make sure it will keep working
You’ve Built a Web Application

• What do you know about it
  • Does it work?
  • Does it work correctly?
  • Does it work correctly under all circumstances?
  • Will users like it?
  • Did you build the right application?
  • Will it scale?

• How do you answer such questions?
  • Testing
What is Testing?

• The process of running software in order to find bugs
  • Not to show that bugs are not there
  • What is the difference?

• **A successful test case is one that finds a bug**

• Good testers (QA) are people who
  • Can sit in front of software and break it
  • Are in the frame of mind where you want to break things
  • Are TAs grading homework assignments

• Testing won’t show what’s right, just what isn’t wrong
Software Testing

• Introduced in 15/16/17/18/32
  • Agile programming: write the test cases first
  • Incremental development: continuous testing
• You’ve possibly seen tools to help with testing
  • JUnit for java testing
    • Test cases are methods annotated with @Test
    • Automatically find and run all tests for a system
• Supports repeated testing
Regression Testing

• Testing software once is not very useful
  • You might make it work for some case then
  • But what if the software changes
  • Did you test the right case?

• Regression tests
  • Tests that are run each time the system changes
  • Rerun after each change to ensure no regression

• Test cases are permanent, not throw-away
  • How to do this for web tests?
Testing Web Applications

• You should test your web applications
  • Lots of tools and techniques exist
    • Both commercial and open source

• Testing should be done at all levels

• Testing should be considered from the start
  • Plan a test database, test users, test data, ... to facilitate
  • Have a separate server running for test purposes
  • Design the application to facilitate testing

• We’ve been doing this to some extent throughout the course
Testing Your Projects

• You will be expected to test your projects
  • Have a test plan
  • Test continuously (not just once)

• We will have a testing lab next week to get you going
  • Each person will be responsible for testing some aspects of the project in class
  • With continuing responsibilities until the final hand-in

• As we cover different testing methods
  • Think about how they affect your project
Question

What is not the purview of web site testing:

A. HTML, CSS, and Script validity
B. Determining if the SQL schema is correct
C. Vulnerability to security attacks such as XSS
D. Universal usability
E. Determining how the web application handles a large user load
What Can Be Tested

• Usability
  • Universal Accessibility testing
• Front end: HTML, CSS, Links
• Back end: unit test the node.js/php/python/…
• Application testing (front + back end)
• Compatibility testing
• Performance testing
• Stress testing
• Security testing
Front End Testing

• Are the HTML and CSS correct?
  • What happens if they aren’t?
  • Can the site be crawled by search engines?

• Are the links correct (and active)

• Are the forms correct?
  • Are values validated correctly
  • Are default values correct
  • What happens to incorrect inputs

• Cookie testing
  • Does the application work without cookies
  • Are cookies encrypted correctly
  • Do sessions expire correctly
Front End Testing

• HTML validation
  • W3C HTML validation service (http://validator.w3.org)
• Link checkers
  • W3C HTML validation (http://validator.w3.org/checklink)
• CSS validation
  • W3C CSS validation (http://jigsaw.w3.org/css-validator)
User (Usability) Testing

- Test the effectiveness of the user interface
  - What is liked or disliked (subjective)
  - Speed and ease of use
  - What errors are made (and the error rate)
- How understandable is the interface
  - What instructions/help is required, what is obvious
- Is the content logical and easy to follow
  - Consistency of navigation and presentation
  - Spelling errors, colors and fonts, English
- Universal usability testing
  - Accessibility testing
  - Internationalization testing
Doing Usability Testing

• **User studies**
  - Watching users use the site (video taping for analysis)
  - Surveys or polls after use
  - Determining what information is needed

• **Log studies**
  - What are the navigation paths? What are the common operations? How are key pages reached?
  - Detecting errors from the logs
  - Timings
  - Using Google Analytics and similar tools

• **Tools and External Sources**
  - [http://www.youtube.com/watch?v=uLyWxXNDNbl](http://www.youtube.com/watch?v=uLyWxXNDNbl)
  - [http://www.youtube.com/watch?v=xLIBe6VWmrY](http://www.youtube.com/watch?v=xLIBe6VWmrY)
Usability Testing Tools

• UserTesting
  • http://info.usertesting.com/EduDemo.html

• Usage
  • Develop a well-thought out test first
    • What you want the user to do
    • What questions you want to ask
    • What questions you want answered
  • Sign up: https://www.usertesting.com/users/sign_up?client=true
  • Choose ORDER a TEST
  • Select no more than 3 participants
  • Use code U-BU6 in lieu of payment
A/B User Testing

- Once you have a system working
  - Want to test possible modifications
- Randomly choose a subset of your users
  - Give them the new interface
  - Give different subsets different new interfaces
  - Be consistent
- Measure effectiveness, usability, etc.
Compatibility Testing

• Browser compatibility
  • IE, Firefox, Mozilla, Safari, Opera, Chrome, ...
  • Different versions of each
  • Testing: browsershots.org
  • Testing: on-site testing

• OS Compatibility
  • What might be OS-dependent

• Mobile Compatibility
  • iPhone, Android, Blackberry, other phones
  • Different versions of each
Printing Testing

• Do the pages print correctly
  • Fonts, alignment
  • Size, layout
  • What prints, what doesn’t (frames)
• Printing from different browsers
• Printing to different types of printers
• International printing
Front End Functional Testing

• Test the JavaScript in the front end
  • Unit testing (test functions individually)

• Tools
  • Qunit, Jasmine
  • Introduction and examples: http://qunitjs.com/intro/
  • Testing tools for jQuery
  • Several other JavaScript testing frameworks exist
Back End Functional Testing

- Are requests handled correctly?
  - Are the proper pages generated
  - Are the proper actions taken
- Depends on technology used in the back end
  - Simulate front end calls through function calls
  - Tools depend on language
- Closest to standard software testing
- Tools:
  - PHP: SimpleTest
  - Python/Django: PyUnit
  - Node (Jasmine, node-unit, Expresso, mocha + chai, nemo, ...)
    - See: [http://jasmine.github.io/2.4/introduction.html](http://jasmine.github.io/2.4/introduction.html)
    - See: [http://developer.android.com/training/testing/ui-testing/espresso-testing.html](http://developer.android.com/training/testing/ui-testing/espresso-testing.html)
Interface Testing

• What are the interfaces
  • Web page <-> web server
  • Web server <-> database
  • Web server <-> user server

• Check the interactions between these servers
  • Do they do the right thing
  • Are inputs validated properly
  • Are errors handled properly
  • Is validation and security correct
  • What happens if the user interrupts a transaction
  • What happens if the web connection is reset
  • What happens if the user clicks twice
Interface Testing Tools

- **httpunit**
  - Create test cases for calls to the server
    - Providing input, checking expected output
    - These are using a Java framework
  - Generating test cases automatically
    - By analyzing on the JavaScript code

- **Sikuli**
  - Test cases with visual input and output
    - Why is this difficult?
  - Examples [https://www.youtube.com/watch?v=pWLa1kxakOs](https://www.youtube.com/watch?v=pWLa1kxakOs)
  - Overview: [https://www.youtube.com/watch?v=01jF18KrEMY](https://www.youtube.com/watch?v=01jF18KrEMY)

- **Selenium**
  - Next time
Next Time

• Testing web sites
• Homework:
  • Make sure your web site is ready for testing
    • What should this entail
    • What do you need to do
    • Prelab for Wednesday