Patch Production Now!

Critical role of CI/CD, test driven development, & automated testing for managing application security over time.

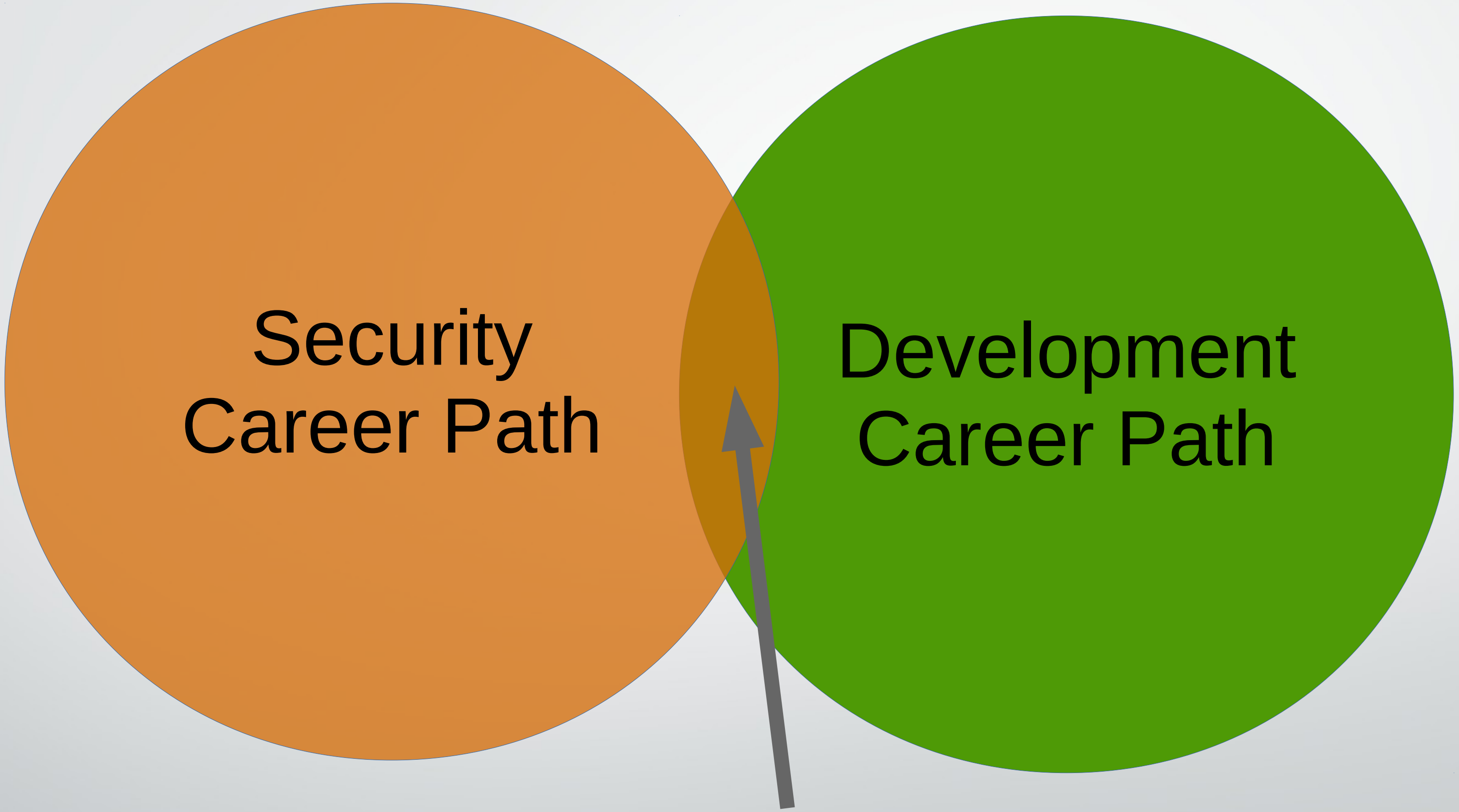
Frank S. Rietta
M.S. Information Security

CEO of Rietta.com, a
Security Focused Web Application Agency
@FrankRietta on Twitter

March 5, 2020

RIETTA.COM™
/SECURITY

A talk on the road less traveled...



We’re exploring the small overlap today!

All security depends on software security.

“Security is a non-functional requirement”

Project Management Professor

How confident are you that your production software can be updated and deployed right now?

Afraid it would break production?

You're not alone. This talk is how to have confidence that it will work!

Have you seen this meme?

I DON'T KNOW WHO YOU ARE
BUT IF YOU DEPLOY ON A FRIDAY, I WILL FIND YOU AND I WILL KILL YOU
imgflip.com

Have you seen this meme?

DEPLOYING ON A FRIDAY AT NOON?
I TOO LIKE TO LIVE DANGEROUSLY
Memekingz.net

Don’t live by fear. Friday is just as good as any other day to deploy if **you have confidence that your software works**.

This talk is about how to build confidence!

What We'll Cover Today

- Why Patching Production is So Slow
- The ideal of Test Driven Development
- Fixing Bugs in Legacy Code without Tests
- Different Types of Tests for your Test Suite
- Prerecorded Live Demo of Browser Tests with Selenium via Capybara (Ruby on Rails)
- Continuous Integration / Continuous Deployment

Why is patching slow?

How long does it take to patch production?

Severity Level	Average Days to Patch
All (Regardless of Severity)	38 days
High	34 days
Medium	39 days
Low	54 days
Oldest unpatched CVE	340 days

Source: TCell 2018

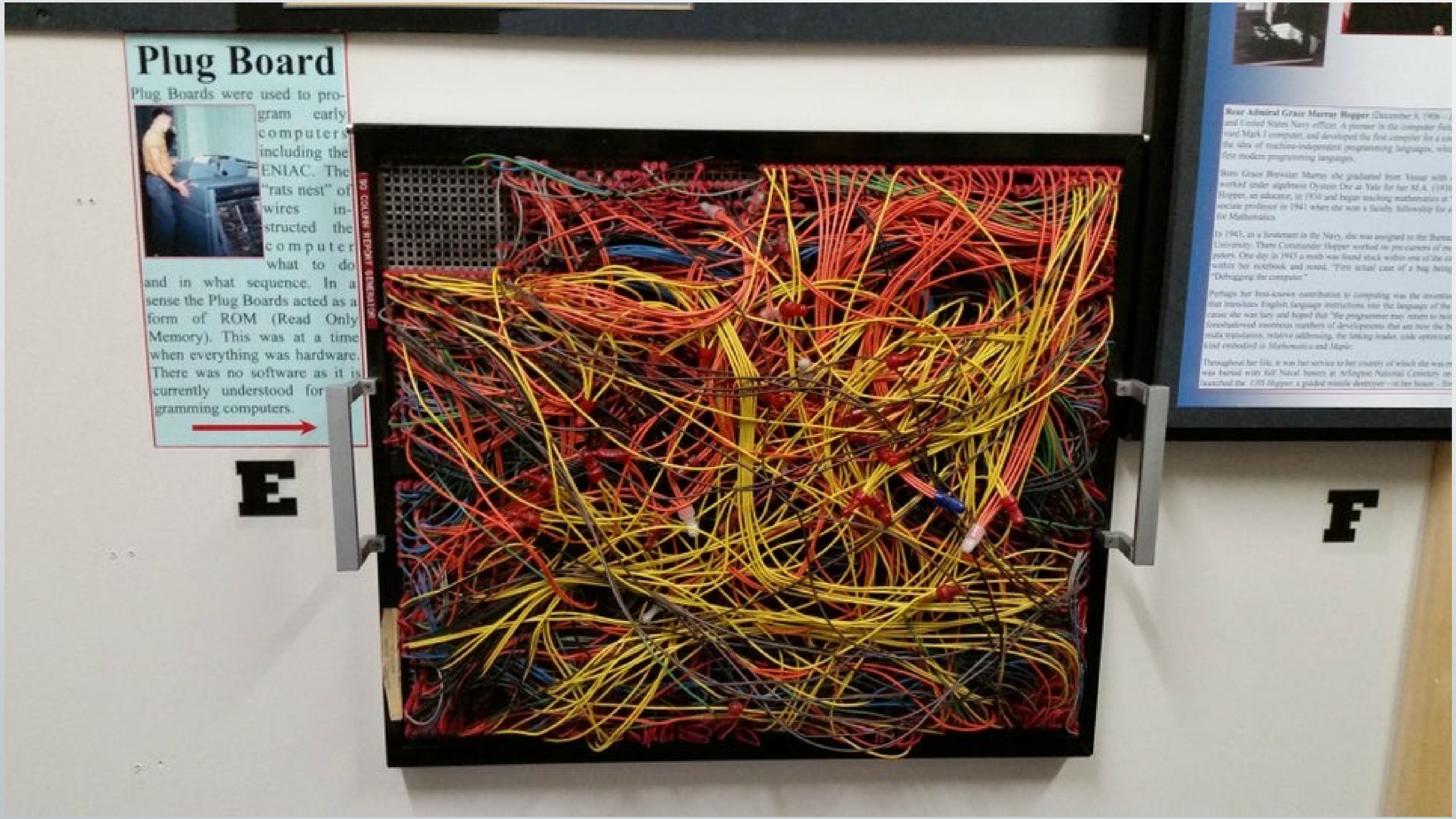
No one is around to do any work...

Change is hard...

Change breaks things..

Don’t known all the dependencies!

Spaghetti code is real!


Plug Board
Plug Boards were used to program early computers including the ENIAC. The "rats nest" of wires instructed the computer what to do and in what sequence. In a sense the Plug Boards acted as a form of ROM (Read Only Memory). This was at a time when everything was hardware. There was no software as it is currently understood for programming computers.
E
F

Source:https://www.reddit.com/r/ProgrammerHumor/comments/3esa42/spaghetti_code_gets_real/

Lack of confidence

It’s a manual process

Plenty of time to get hacked before the fix is deployed!

Key Observation for the Future

A computer should **deploy to production without human intervention** as soon as *all the automated tests pass*.

Who Should Write Tests?

Developers Write Tests

“It is irresponsible for a developer to ship a line of code that [he or she] has not executed any unit test for, and one of the best ways to make sure you have not shipped a line of code without testing is to practice TDD”

Robert Martin in his 2012 debate on TDD with Jim Coplien, video at

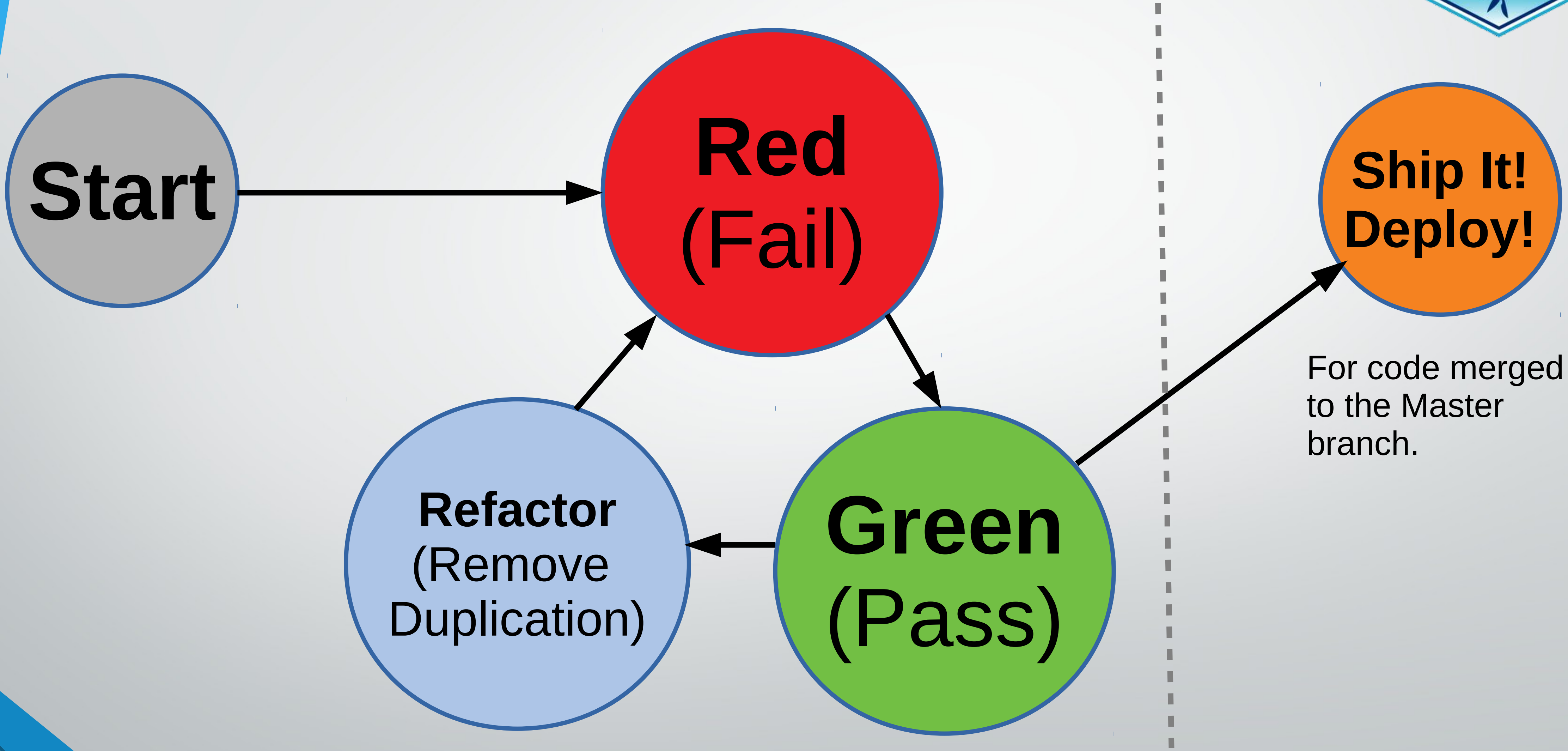
https://www.youtube.com/watch?v=KtHQGs3zFAM&t=14%3A42

The Three Rules of TDD

1) Don’t write a line of *production* code without having a corresponding failing test
2) Don’t write too many failing tests without writing *production* code
3) Don’t write more *production* code than is sufficient to pass the currently failing test

Source: Robert Martin
http://butunclebob.com/ArticleS.UncleBob.TheThreeRulesOfTdd (2005)

The TDD Cycle + CD/CD



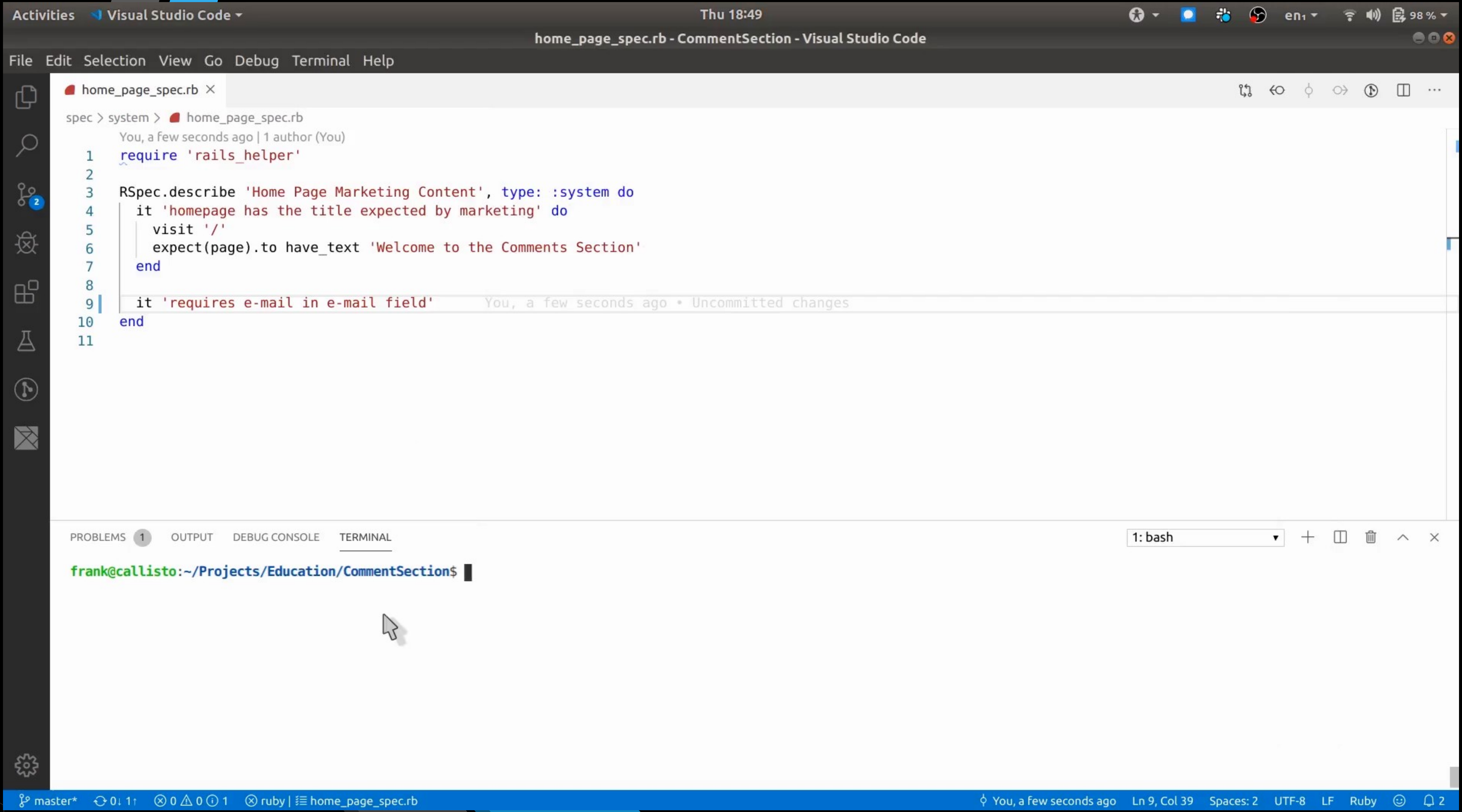
For code merged to the Master branch.

Types of Tests

- Experimenting in the interactive debugger
- Two general categories of automated tests
  - Integration Testing
    - User Interface / Browser Tests for Web UI testing as a human sees it
    - Request Tests (Interaction with Full Web Server) for API
    - Controller Tests
  - Unit Testing
    - Automated code analysis (SAST) for security and code style
    - Unit / Model
    - View tests
- Feedback from production via Exception monitoring

Source: https://twitter.com/d3veducation/status/813056723271450624

Demo Testing Web UI with Selenium/Capybara



```ruby
require 'rails_helper'

RSpec.describe 'Home Page Marketing Content', type: :system do
  it 'homepage has the title expected by marketing' do
    visit '/'
    expect(page).to have_text 'Welcome to the Comments Section'
  end

  it 'requires e-mail in e-mail field'      You, a few seconds ago • Uncommitted changes
end
```

```
frank@callisto:~/Projects/Education/CommentSection$
```

Must change code to fix vulnerabilities!

Changing Software to Fix Defects

- When there is a security defect, how do the developers change the code without breaking the software?
- Two industry approaches:
  - Edit and Pray
  - Cover and Modify

Legacy Code is Hard to Change

“In the industry, legacy code is often used as a slang term for difficult-to-change code that we don’t understand. But over years of working with teams, helping them get past serious code problems, I’ve arrived at a different definition. **To me, legacy code is simply code without tests.**"

Feathers, Michael. Working Effectively with Legacy Code (Robert C. Martin Series) (Kindle Locations 226-229). Pearson Education. Kindle Edition.

Feather's Legacy Code Change Process

When you have to make a change in a legacy code base:

1) Identify change points.
2) Find test points.
3) Break dependencies.
4) Write tests.
5) Make changes and refactor

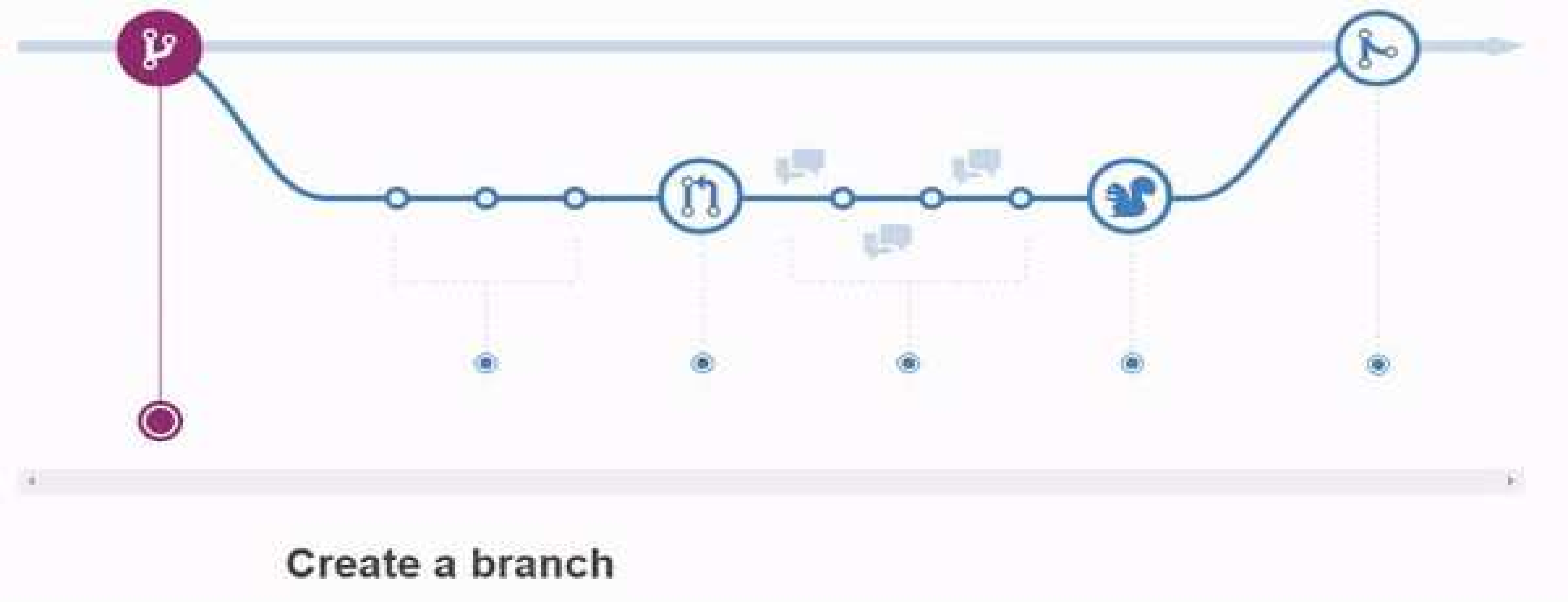
Source: Feathers, Michael. Working Effectively with Legacy Code (Robert C. Martin Series) (Kindle Locations 624-628). Pearson Education. Kindle Edition.

Continuous Integration / Continuous Deployment

Team Requirements for CI/CD

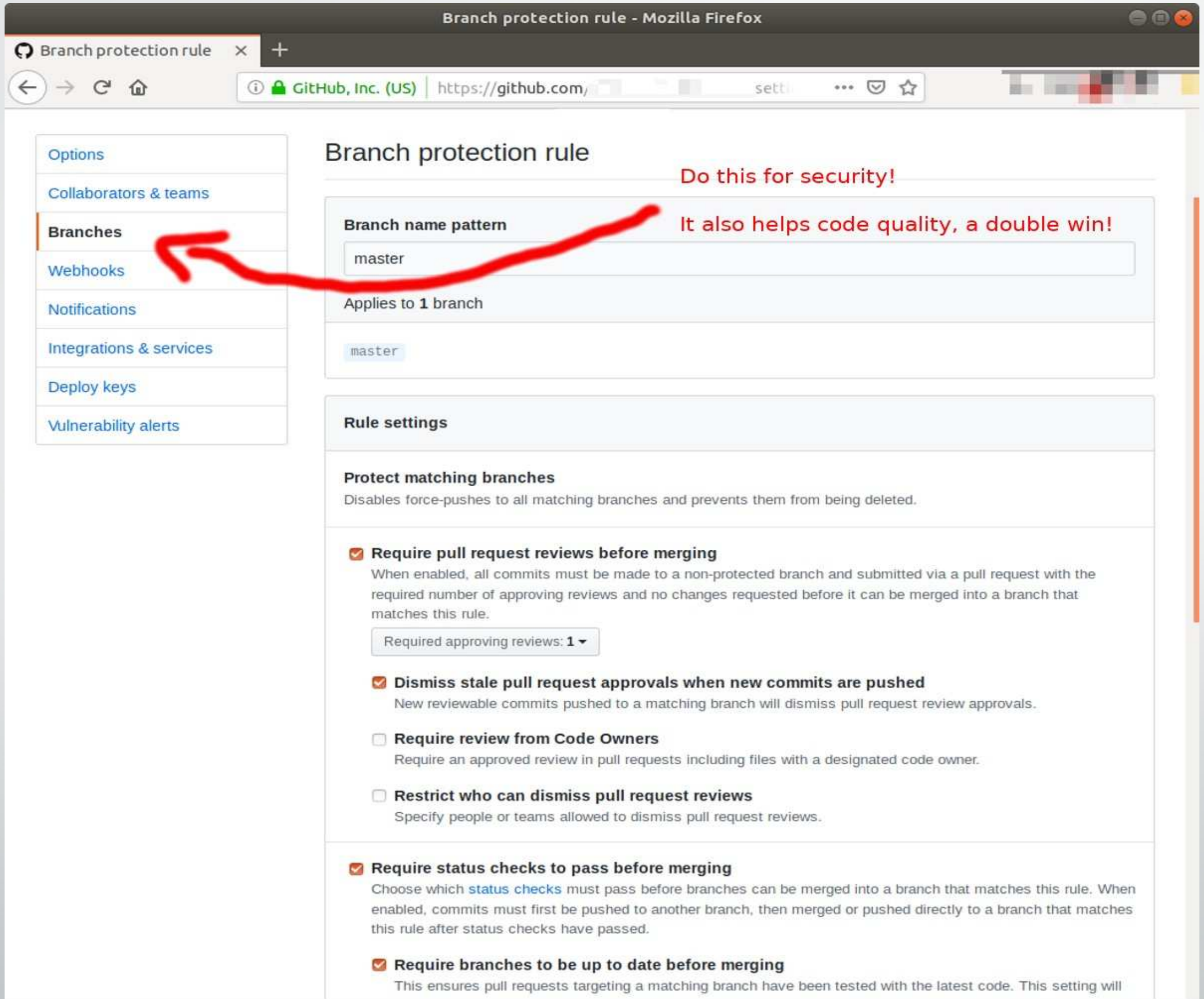
- Developers write detailed automated tests
- Containerized deployment runs tests on demand
- Configure tests to run for each pull request
- Make Master branch protected so that tests MUST PASS before code can be merged into it
- Another dev reviews code and asserts that the tests seem on-the-face to cover intended functionality

Pull Request Driven Git Flow



Source: https://medium.com/bdo-digital/git-workflows-99f7174fe461

Protect the Master Branch



Branch protection rule

Do this for security!

It also helps code quality, a double win!

Branch name pattern

master

Applies to **1** branch

`master`

Rule settings

Protect matching branches
Disables force-pushes to all matching branches and prevents them from being deleted.

☑ **Require pull request reviews before merging**
When enabled, all commits must be made to a non-protected branch and submitted via a pull request with the required number of approving reviews and no changes requested before it can be merged into a branch that matches this rule.

Required approving reviews: **1** ▾

☑ **Dismiss stale pull request approvals when new commits are pushed**
New reviewable commits pushed to a matching branch will dismiss pull request review approvals.

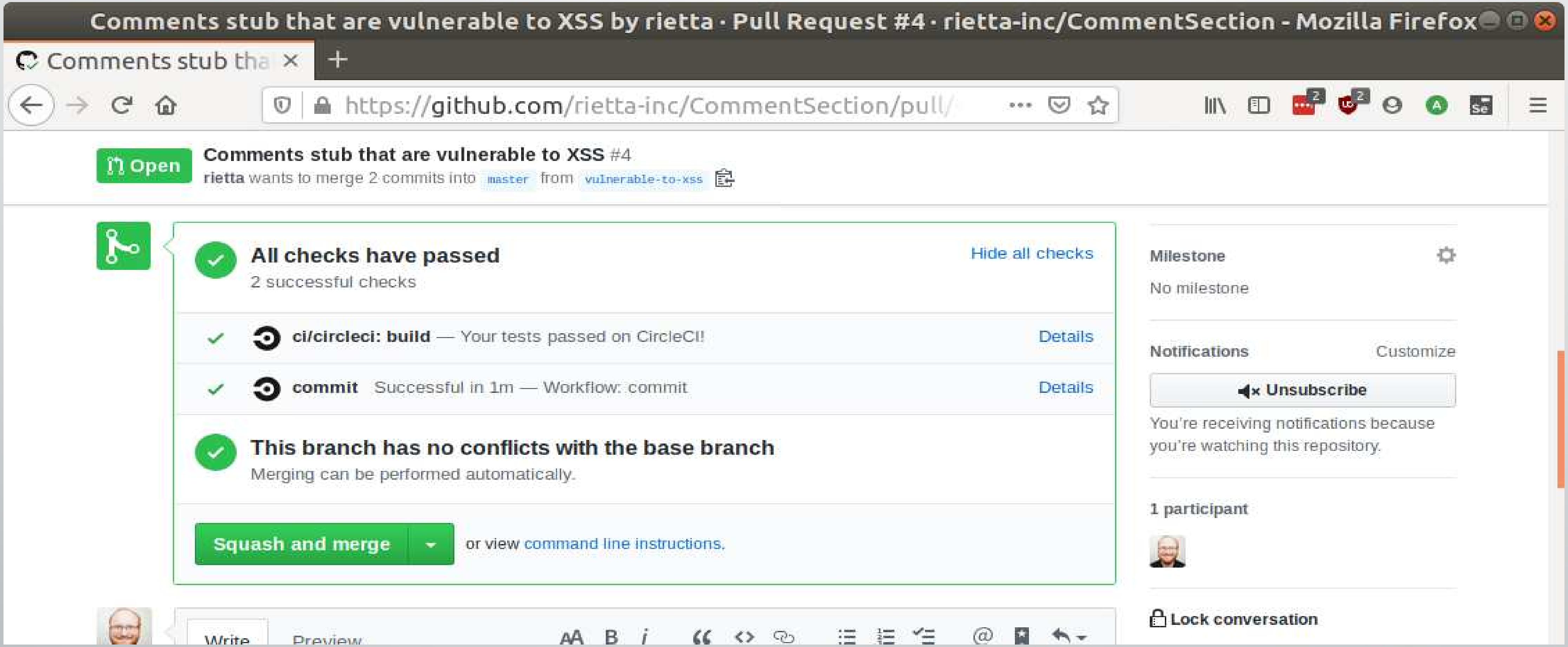
☐ **Require review from Code Owners**
Require an approved review in pull requests including files with a designated code owner.

☐ **Restrict who can dismiss pull request reviews**
Specify people or teams allowed to dismiss pull request reviews.

☑ **Require status checks to pass before merging**
Choose which status checks must pass before branches can be merged into a branch that matches this rule. When enabled, commits must first be pushed to another branch, then merged or pushed directly to a branch that matches this rule after status checks have passed.

☑ **Require branches to be up to date before merging**
This ensures pull requests targeting a matching branch have been tested with the latest code. This setting will

CI is Part of the Code Review Process



Comments stub that are vulnerable to XSS by rietta · Pull Request #4 · rietta-inc/CommentSection - Mozilla Firefox

https://github.com/rietta-inc/CommentSection/pull/

Open **Comments stub that are vulnerable to XSS** #4

rietta wants to merge 2 commits into `master` from `vulnerable-to-xss`

All checks have passed
2 successful checks

✓ **ci/circleci: build** — Your tests passed on CircleCI! Details

✓ **commit** Successful in 1m — Workflow: commit Details

This branch has no conflicts with the base branch
Merging can be performed automatically.

or view command line instructions.

Milestone
No milestone

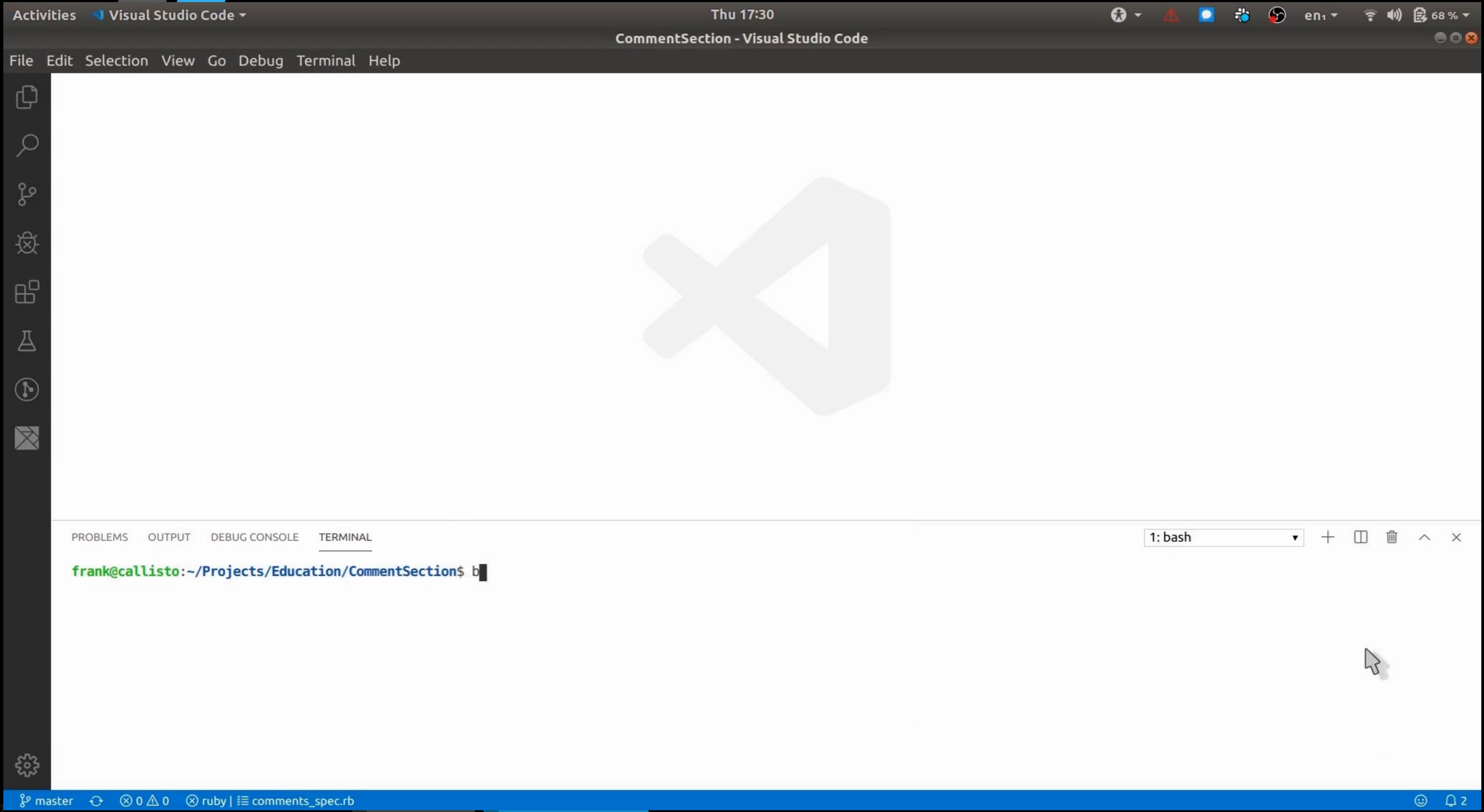
Notifications Customize

You're receiving notifications because you're watching this repository.

1 participant

@FrankRietta @RiettaInc #SnowFROC

Demo Fixing XSS Vulnerability with Test and CI/CD in Github



```
frank@callisto:~/Projects/Education/CommentSection$ b
```

Automated Security Tests ARE Acceptance Criteria

When automated CVE checking or SAST tests fail, then the **pull request fails code review** and **cannot be merged into the Master branch.**

This simple merge gate is the foundation for much more effective application security.

The Dependency Avalanche

Many Dependencies

- Before a dev does any work at all, there are:
  - 64 Java run time libraries in a new Grails 4 project
  - 75 Gems in a brand new Rails 6 project
  - 996 node modules in a brand new React project

- These libraries are produced by volunteer open source maintainers, many of whom are not paid anything to work on open source

DIY SIEM – Nightly CVE Scan (CircleCI YAML Config)

```
security:
    docker:
      - image: circleci/ruby
        environment:
    working_directory: ~/repo

    steps:
      - checkout
      - run:
          name: Install bundler-audit
          command: gem install bundler-audit
      - run:
          name: Run bundle-audit
          command: bundle audit -u
```

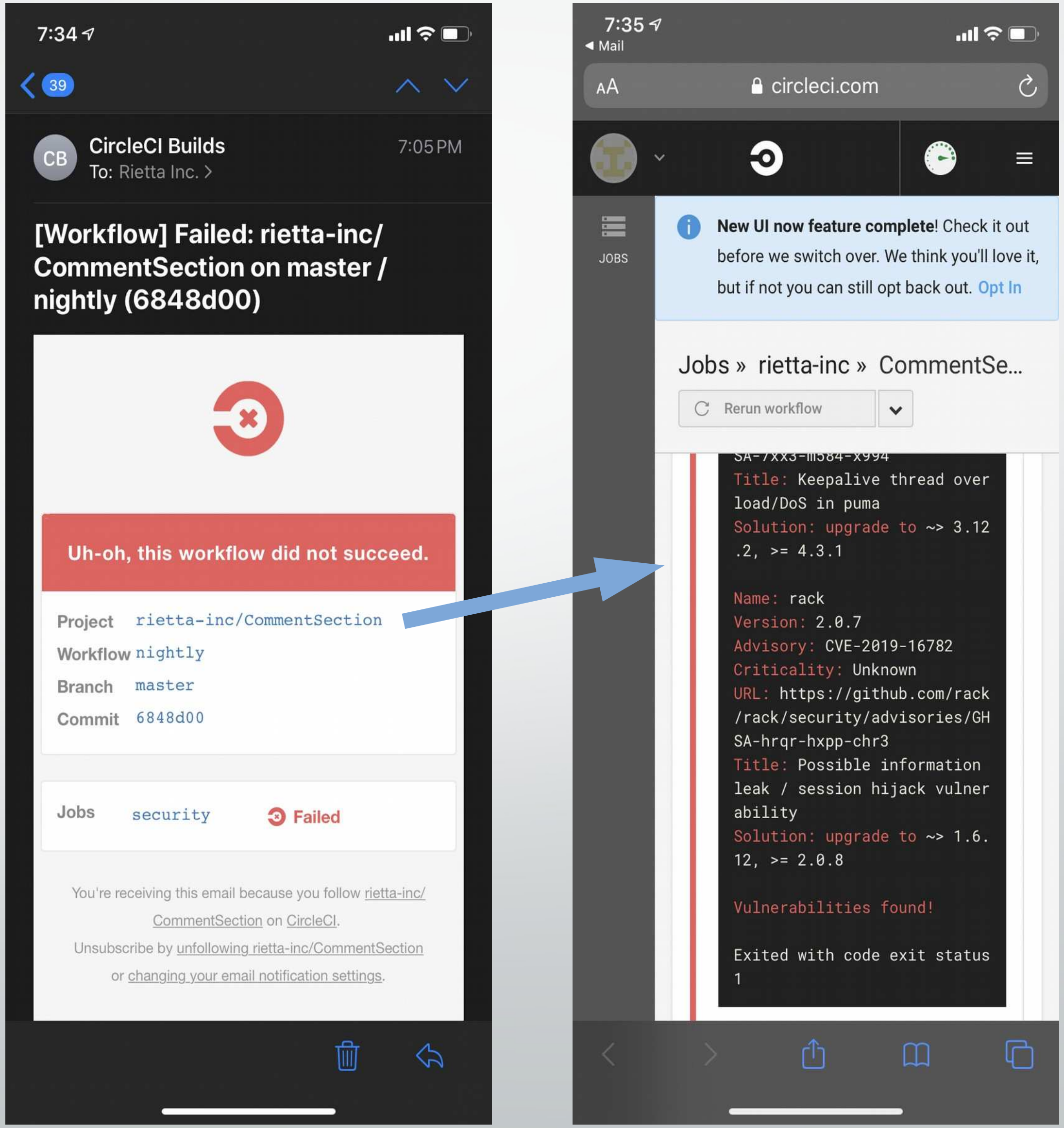
```
workflows:
  version: 2
  commit:
    jobs:
      - build
  nightly:
    triggers:
      - schedule:
          cron: "0 0 * * *"
          filters:
            branches:
              only:
                - master
    jobs:
      - security
```

DIY SIEM – It works!

When your CI runs nightly security checks it can notify you when a new security vulnerability is detected.

A practically free SIEM!

This is a real notification Saturday night on my iPhone about this demo app!



[Workflow] Failed: rietta-inc/CommentSection on master / nightly (6848d00)

Uh-oh, this workflow did not succeed.

Project rietta-inc/CommentSection
Workflow nightly
Branch master
Commit 6848d00

Jobs security Failed

You're receiving this email because you follow rietta-inc/CommentSection on CircleCI.
Unsubscribe by unfollowing rietta-inc/CommentSection or changing your email notification settings.

New UI now feature complete! Check it out before we switch over. We think you'll love it, but if not you can still opt back out. Opt In

Jobs » rietta-inc » CommentSe...

```
SA-7xx3-m584-x994
Title: Keepalive thread over
load/DoS in puma
Solution: upgrade to ~> 3.12
.2, >= 4.3.1

Name: rack
Version: 2.0.7
Advisory: CVE-2019-16782
Criticality: Unknown
URL: https://github.com/rack
/rack/security/advisories/GH
SA-hrqr-hxpp-chr3
Title: Possible information
leak / session hijack vulner
ability
Solution: upgrade to ~> 1.6.
12, >= 2.0.8

Vulnerabilities found!

Exited with code exit status
1
```

Questions?

Thank you!

- Find Us Online / Social Media
  - https://rietta.com
  - @RiettaInc on Twitter
  - @rietta-inc on LinkedIn
- Free resource on how to build a robust application security program at your company @ https://rietta.com/security.
- Subscribe to our Blog and Newsletter
  - The Rietta Blog (since 2005) @ https://rietta.com/blog
  - Rietta on Security (monthly newsletter) at https://rietta.com/on-security
- See me after for my business card or to get a cool “Code Review before Pen Test” sticker!!!

Code Review before Pen Test
I am a #securecoder
I practice #testdrivendevelopment
RIETTA.COM /SECURECODER

Tools Demonstrated Today

- Ruby on Rails application with:
  - RSpec behavior driven development framework - https://rspec.info
    - provides testing for all parts of the MVC application, including unit and integration testing
- Selenium with Chrome Webdriver - https://www.seleniumhq.org
  - Great for driving web UI tests using a real Chrome browser instance
- CircleCI with Dockerized Linux that runs the entire test suite, including the automated browser tests
- Integration of CI with GitHub so that it becomes a key part of the development process to protect the master branch.

Other Tools and Resources

- JavaScript is huge! And there is testing support for it too:
  - End-to-end / Integration testing: https://www.cypress.io
  - Unit testing: https://jestjs.io
- My friend Josh Justice has a tutorial to Learn TDD in React https://learntdd.in/react.
- And for Ruby & Ruby on Rails, I recommend Drifting Ruby at https://www.driftingruby.com for thousands of screencasts by Dave Kimura on how to dev in Rails. **Use discount code: rietta to get 25% off your monthly subscription**.

This talk is available as an article on the Rietta Blog

https://rietta.com/blog/patch-production-faster-with-agile-development