Automated Testing

By Jack Slinger

epiGenesys
A University of Sheffield Company

Who am I?

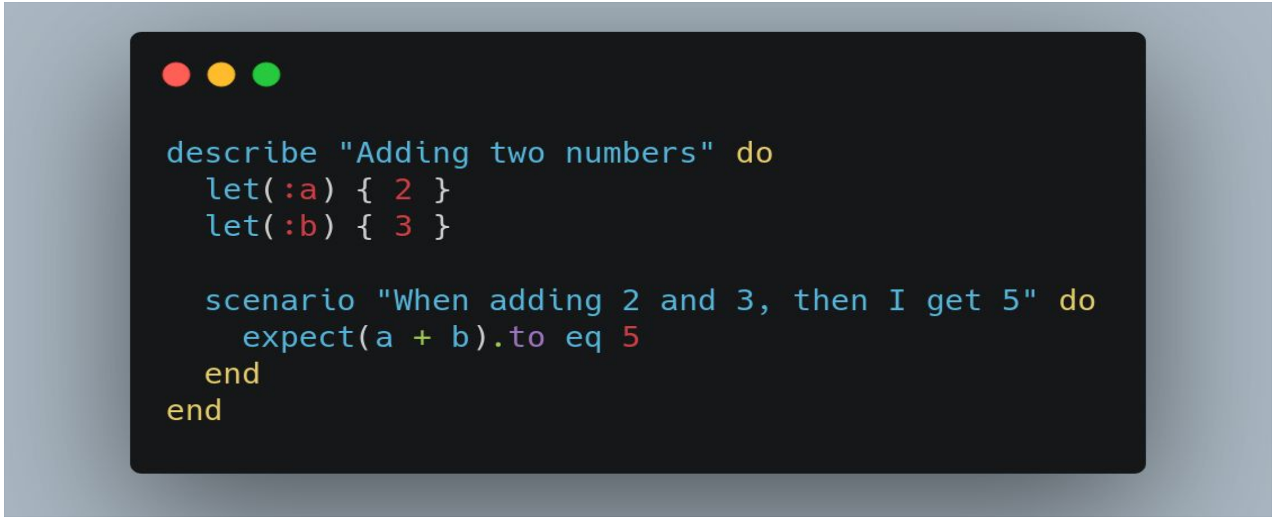
- Lead Software Engineer at epiGenesys.
- A bespoke software development company with ~15 employees.
- My team work with Sheffield CTRU to develop and maintain their clinical trials database called Prospect.

What is Automated Testing?

The components of an automated test

1. Setup the system in a particular state.
2. Run through a set of actions.
3. Check that the system behaves correctly / produces the expected output.

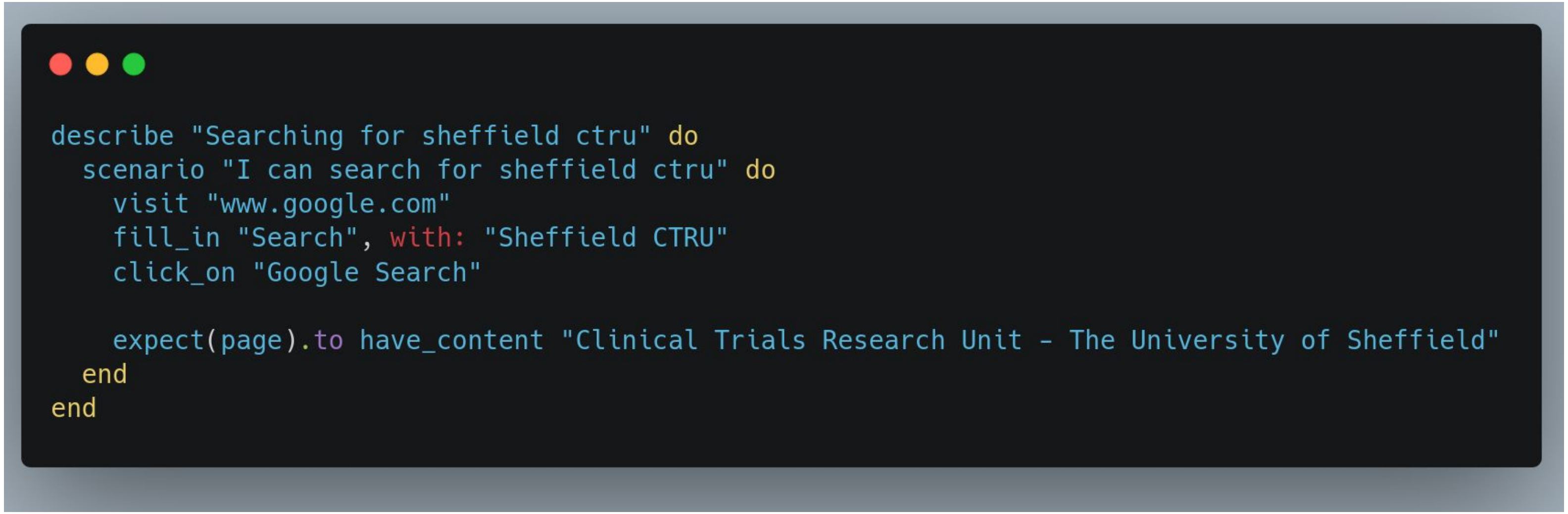
A simple addition example



```ruby
describe "Adding two numbers" do
  let(:a) { 2 }
  let(:b) { 3 }

  scenario "When adding 2 and 3, then I get 5" do
    expect(a + b).to eq 5
  end
end
```

A basic example using a web browser



```
describe "Searching for sheffield ctru" do
  scenario "I can search for sheffield ctru" do
    visit "www.google.com"
    fill_in "Search", with: "Sheffield CTRU"
    click_on "Google Search"

    expect(page).to have_content "Clinical Trials Research Unit - The University of Sheffield"
  end
end
```

The problems with a purely manual testing approach

Why didn’t Prospect have automated tests?

- Prospect was first created in 2009, and has been continually developed since then.
- The tools and frameworks for automated testing were not as good back then, so we chose to rely on manual testing.
- When I joined the company ~5-6 years ago we already had a large application with very limited automated testing.

Problems with a purely manual approach

Prospect had limited automatic tests, and it was hard to add more. This required us to rely completely on our manual testing approach, which caused the following issues:

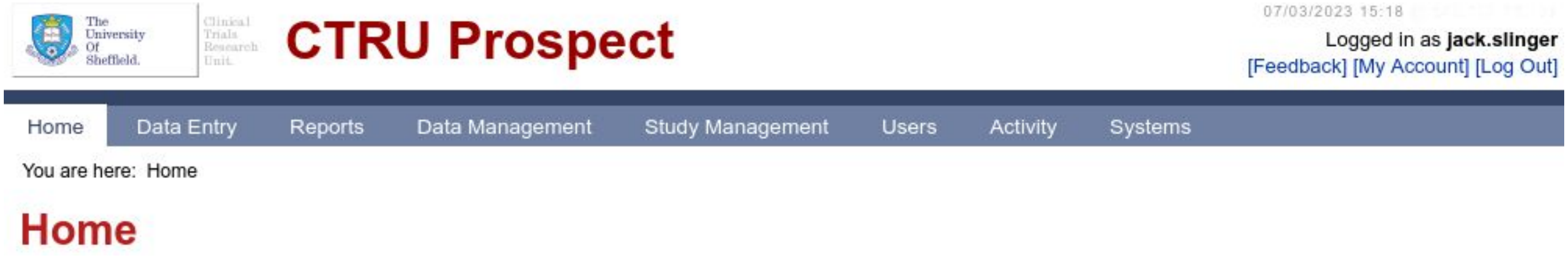
- More expensive to manually re-test existing functionality.
- Harder to move quickly and make changes with confidence.
- Fixing bugs required a lot of caution to avoid re-introducing previous issues.

How we Implemented Automated Testing

Wrapping a new system around the old one

- We are currently in the process of rebuilding Prospect, to improve security, performance, and functionality.
- To achieve this we have built a new application (New Prospect) that uses Old Prospect to display pages that haven't been moved across yet.

Old Prospect



CTRU Prospect

07/03/2023 15:18

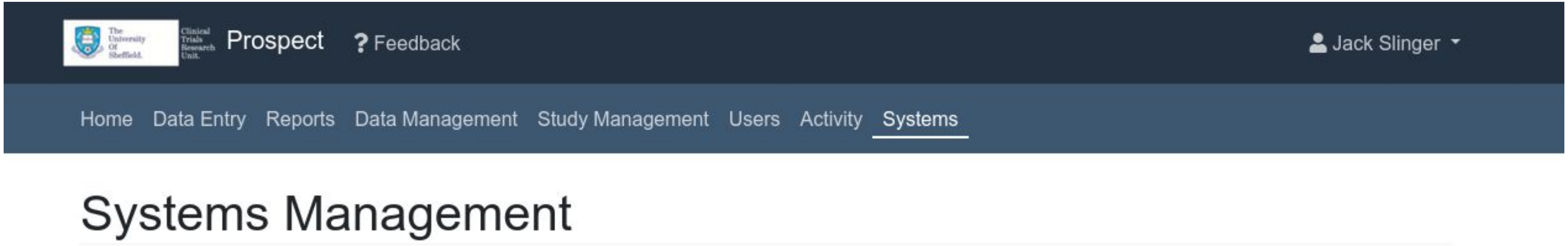
Logged in as **jack.slinger**

[Feedback] [My Account] [Log Out]

You are here: Home

Home

New Prospect



Systems Management

How does this help testing

Using the more recent tools and framework of New Prospect, we are able to write automated tests both for the New Prospect pages and the Old Prospect ones.

Testing new components

- New functionality added to New Prospect has automated tests created during the development process.
- Before new components can be signed off, they require sufficient automated test coverage.

Testing existing components

Existing components have automated tests added to them in one of two ways:

1. When moving existing functionality to New Prospect.
2. When a component requires changes in Old Prospect and those changes carry some risk.

How does this integrate with validation?

Mapping tests to the Function Specification (FS)

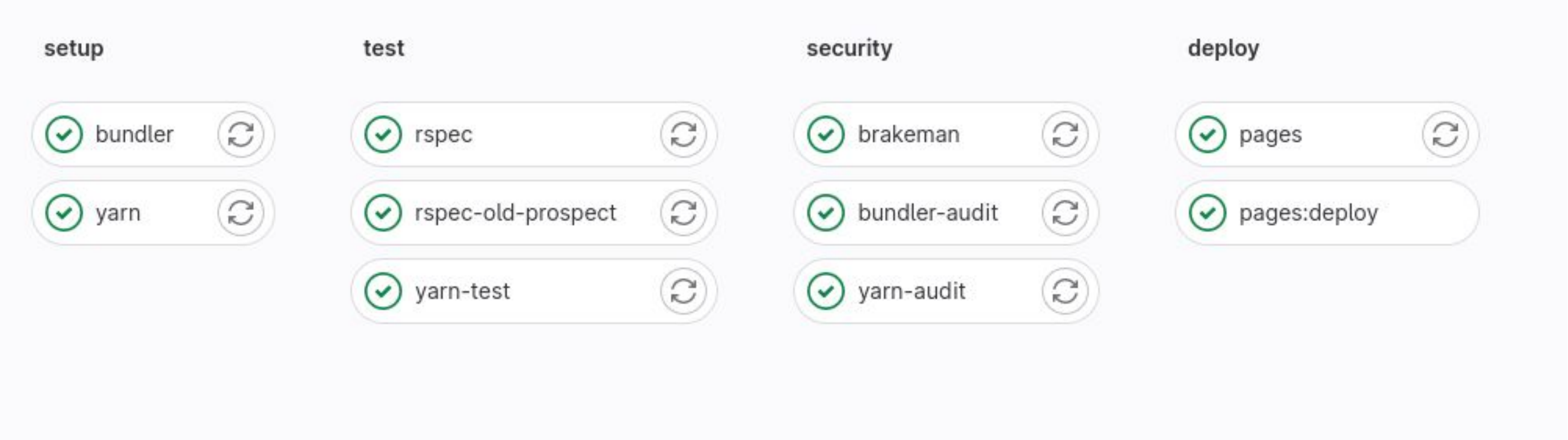
- We break the validation of Prospect into components, with each component having a User Requirements Specification (URS) and an FS.
- The FS details what a component should do, along with the manual testing steps and the results of the manual testing.
- epiGenesys then add a reference to the automated tests that cover that feature.
- Once epiGenesys and Sheffield CTRU are happy with the test coverage the component can be signed off.

An example FS

Feature	Manual Test Steps	Tested by	Automatic tests
Admin users should be able to create users	[as a logged-in admin user] - Navigate to "New user" page - Enter valid username, name, email and email confirmation - Press "Create" button - Confirm redirect to "Users list" page - Confirm display of notice that the user has been created - Confirm that the created user is present in the users list	Joe Blogs	features/admin/user_management_spec.rb

Ensuring the tests are run

Gitlab CI



setup	test	security	deploy
✓ bundler	✓ rspec	✓ brakeman	✓ pages
✓ yarn	✓ rspec-old-prospect	✓ bundler-audit	✓ pages:deploy
	✓ yarn-test	✓ yarn-audit	

Any questions?