Testing Microservices: fast and with confidence

Stephan Jaensch
@s_jaensch
WELCOME TO YELP
Two years ago...
The Testing Pyramid

- E2E
- Integration
- Unit
The Testing Pyramid
Why end to end tests?

Source: https://www.slideshare.net/danveloper/microservices-the-right-way
### Job Failures (1)

Acceptance Test

### Tests Never Ran (18)

18 acceptance/tests/business/detail_acceptance_test.py::test_edit_business_attributes[changeset_data0-expected_response0]

### Test Failures (6)

- acceptance/tests/account_acceptance_test.py::test_account_info_v2[expected_response0]
- acceptance/tests/business/detail_acceptance_test.py::test_edit_address[8-submit_address1-expected_address_response1]
- acceptance/tests/business/detail_acceptance_test.py::test_edit_location
- acceptance/tests/business/info_acceptance_test.py::test_info_missing_location_info_v1
- acceptance/tests/user_interface_acceptance_test.py::test_ui_marketing_banner_v1[show_accelerator_banner_since_business_has_v1]
Improving speed and reducing flakiness
Clone → Merge branches → Test → Push changes

- Acceptance Test
  - Passed in 7m, 46s
  - Mock Test
  - Paasta iTest
  - Unit Test
Contract testing

Contract Testing is writing tests to ensure that the explicit and implicit contracts of your microservices work as advertised.

Bob Reselman
A service call at Yelp

1. uses bundled spec
2. make API call

Our code

client library

Service

API specification
Anatomy of a client library request
Inject mock in client library

Construct request

Marshal data

Validate request

Unmarshal response

Validate response

Mock

HTTP request

HTTP response

REMOTE SERVICE
Patching the client library

class TestExampleHappyhourClient():
    @pytest.fixture(autouse=True)
    def setup_teardown(self):
        with patch_clientlib() as mock_client:
            yield

    def test_get_hours_simple():
        hours = get_hours_simple()

    assert hours == {
        'SUN': {'time_start': 0, 'time_end': 0},
        'MON': {'time_start': 0, 'time_end': 0},
        ...
    }
Providing a mock response

class TestExampleHappyhourClient():
    @pytest.fixture(autouse=True)
    def setup_teardown(self):
        with patch_clientlib() as mock_client:
            mock_client.hours.getHours\
                .set_return_value_and_status_code(
                    return_value={
                        'FRI': {'time_start': 21, 'time_end': 23},
                        'SAT': {'time_start': 21, 'time_end': 23},
                        'SUN': {'time_start': 21, 'time_end': 23},
                    },
                    status_code=200,
                )
        yield
Mocks with errors

```
jsonschema.exceptions.ValidationError:
'id' is a required property
```
Conclusion

- End to end tests don't scale infinitely
- Contract tests are supposed to help
- We can leverage existing infrastructure to do consumer driven contract testing (sort of)
- We're relying on the fact that the service adheres to the specification
We're Hiring!

www.yelp.com/careers/
Questions?

sjaensch@yelp.com