8

HubSpot

UCD
DUBLIN

INTERCOM

Peter
GET SHITE DONE
Brian
Eoin
Adam

Main Points

1. Specifications
2. Minimum Viable Products
3. Testing and Continuous integration
4. Development process
5. Maintenance, logging and monitoring
6. Communication

Specifications

- Bring idea from someone else’s brain into the real world
- State a problem clearly and concisely and write down the steps to required to create a solution

- Client/manager tells you the idea on high level
- Break the idea down into small chunks
- Think about all of the implementation details they’re not thinking about
- Create a time estimate
- Is proof of what you agreed to build

Spec example

- Client/Manager/Stakeholder asks for an application where a user can share photos with other users in their network and photos are viewable in an Instagram style feed
- Agree on which features are critical for MVP
- Divide their idea into smaller problems
- Figure out your timeline

MVPs

- Choose languages
- Choose service provider (AWS, Google Cloud, Azure, Heroku, Smaller provider)
- Execute on the specification
- Use SASS products to your advantage
- Your goal is to prove the product is a good idea

Downside of MVPs

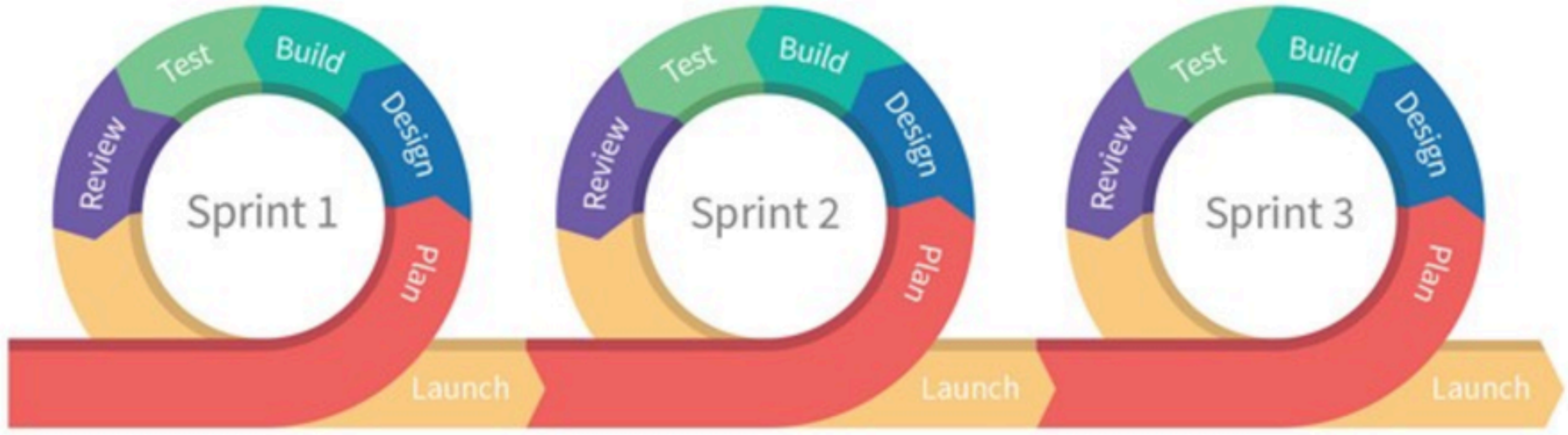
- It needs to be made clear that this is just an MVP
- They tend not to scale well
- They can be the source of technical debt if they’re used for too long
- Feature Creep
- Very little automated testing

Testing

- Makes your life easier in the long run
- Allows you to refactor and add new features with confidence
- Understand the different between unit test, integration tests and E2E tests
- If the team you join doesn’t write tests, be the change they need and start writing them

Development Process

- New feature is created based on spec along with tests
- Pull request made into your release branch
- CI is triggered to run tests
- Code review from colleague
- If the tests pass the changes are deployed



Maintenance and Monitoring

- The most important part of running a live system
- Detailed logging to know when users are getting errors
- Refactor your code to make it clearer and to prevent code rot
- Keeps servers and dependencies updated to avoid security issues and take advantage of improvements
- Fault tolerant systems

Communication

- Being in the same office is best but not a requirement
- If working remote you need clear communication channels
- Have a source of truth (project tracking software) for everyone to refer to and keep updated from
- You need buy in from your team on whatever communication system you use

Questions

peter@8bytes.ie