Clean Project
Recommended Practices

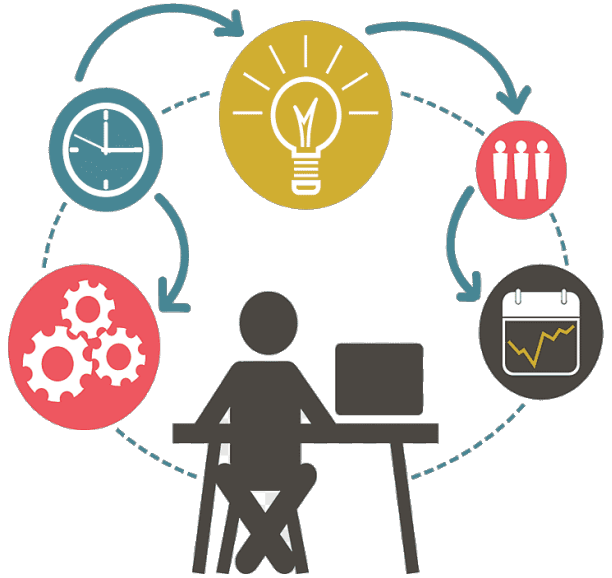
Introduction to Software Engineering

João Rafael Almeida

(joao.rafael.almeida@ua.pt)

Summary

- Active backlog management (plan & track)
- Feature-branching workflow
- Containers-based deployment



Importance of software engineering

- Example:
  - Someone hired us to create an application to compete against Facebook
- How we do that?
- Who is responsible for
  - deployments?
  - defining the application features?
  - infrastructure?
  - ensuring scalability?
  - so many other things ...

Roles

Team manager

Product owner

Architect

DevOps master

Team manager

- Moderates the team discussions
  - Promote collaboration in the team
  - Take initiative to solve problems
- Manages and assign tasks
- **Responsible for delivering project outcomes in time**

Product owner

- Represents the interests of the stakeholders
- Knows what the application should do
  - Features
  - Requirements
- Responsible for accepting the solution increments

Architect

- Responsible for the software architecture
  - Modeling the applications
  - Interactions between components
- Knows the technologies used
  - Frontend
  - Backend
  - Caching
  - Message queues and others

DevOps master

- Responsible for the infrastructure
- Ensures system portability
- Prepares
  - Deployment machine
  - Git repository
  - Cloud infrastructure
  - Databases operations
  - Other aspects

Roles

Team manager

Product owner

Architect

DevOps master

Developer

Software Planning

- Specification
  - Defining what the system should do;
- Design and implementation
  - Defining the organization of the system and implementing the system;
- Validation
  - Checking that it does what the customer wants;
- Evolution
  - Changing the system in response to changing the customer needs;

Software Planning

- **Specification**
  - **Defining what the system should do;**
- Design and implementation
  - Defining the organization of the system and implementing the system;
- Validation
  - Checking that it does what the customer wants;
- Evolution
  - Changing the system in response to changing the customer needs;

Specification

- Definition of requirements and stories
  - Already discussed in previous classes
- Use tools for managing the development
  - Prioritize, assign, and track the work

Specification

- Definition of requirements and stories
  - Already discussed in previous classes
- Use tools for managing the development
  - Prioritize, assign, and track the work
- But... How to do that?
  - Using project planning tools
    - Some with code repository incorporated

Pivotal Tracker

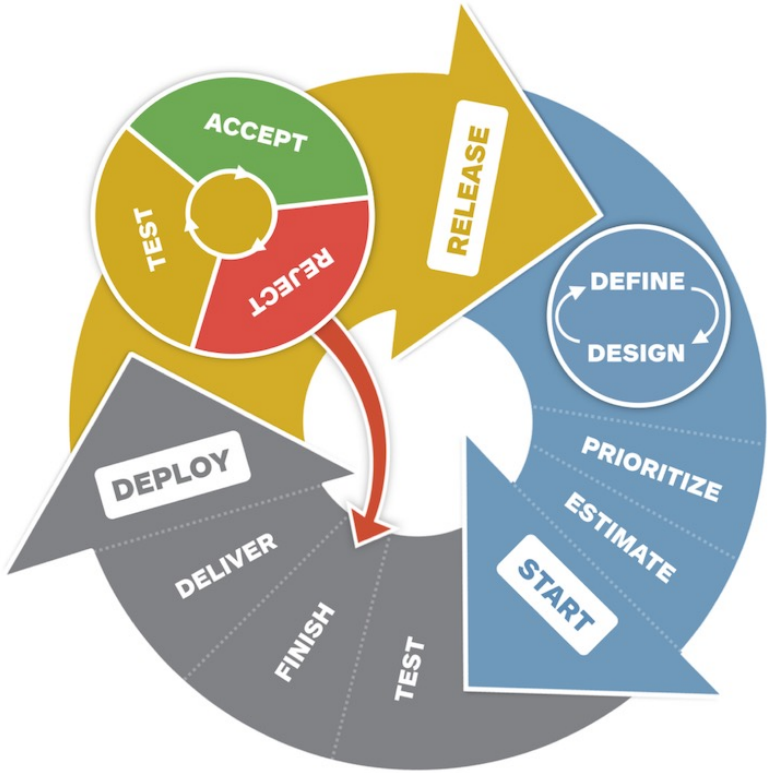
GitHub

Jira Software

GitLab

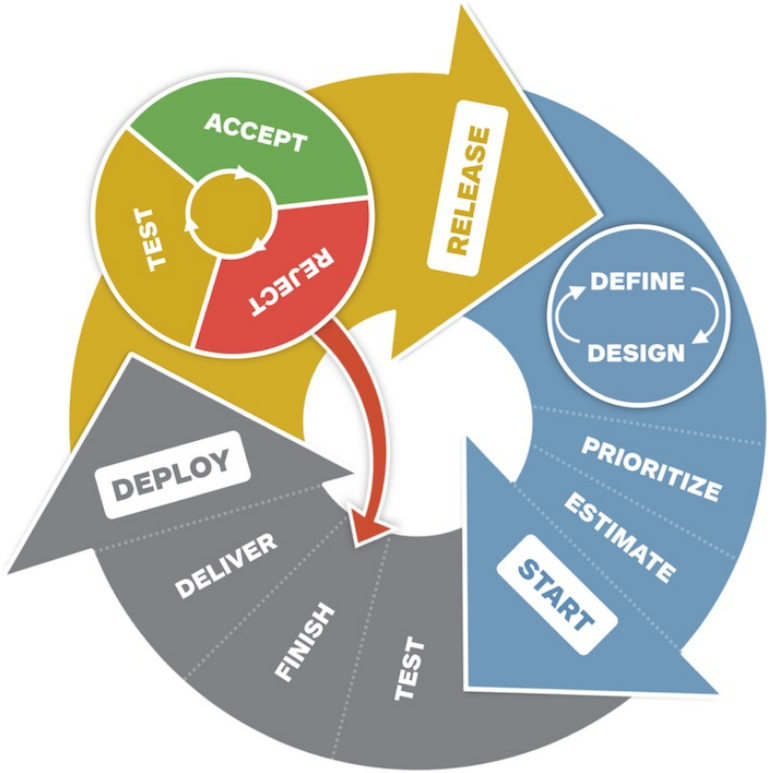
Pivotal Tracker

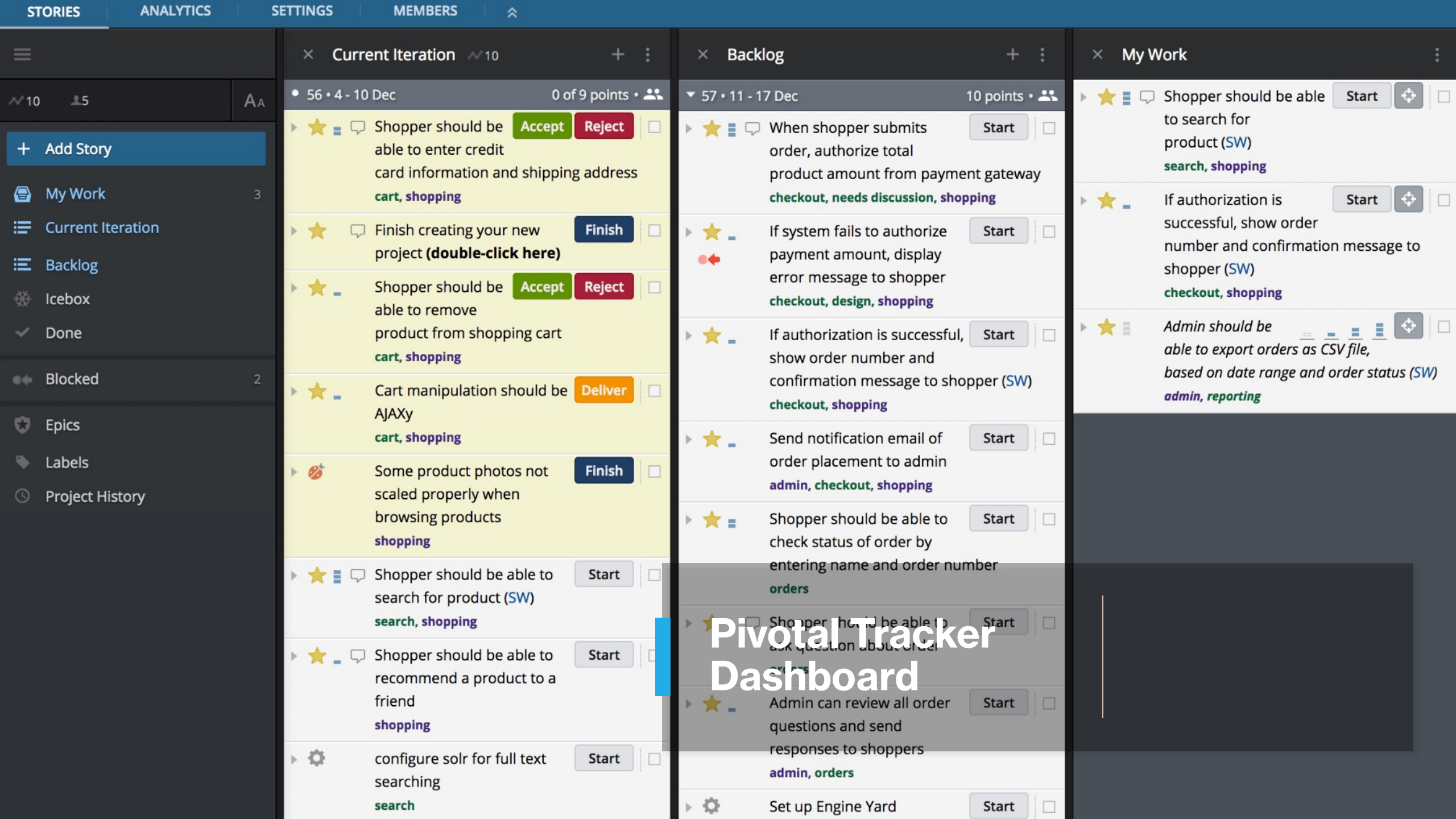
- Agile project manager tool
- Allows the easy management of stories
  - Features, bugs, chores and releases
- Estimation of effort
  - Divide into 4 levels
- Backlog divide into iterations
- Provides good documentation



Workflow Overview

1. Write stories
2. Prioritize stories
3. Estimate stories
4. Start stories
5. Finish and deliver stories
6. Test stories
7. Accept or reject stories
8. Stories move to the Done panel




STORIES
ANALYTICS
SETTINGS
MEMBERS
10
5
Add Story
My Work 3
Current Iteration
Backlog
Icebox
Done
Blocked 2
Epics
Labels
Project History
Current Iteration 10
56 • 4 - 10 Dec
0 of 9 points
Shopper should be able to enter credit card information and shipping address
cart, shopping
Accept
Reject
Finish creating your new project (double-click here)
Finish
Shopper should be able to remove product from shopping cart
cart, shopping
Accept
Reject
Cart manipulation should be AJAXy
cart, shopping
Deliver
Some product photos not scaled properly when browsing products
shopping
Finish
Shopper should be able to search for product (SW)
search, shopping
Start
Shopper should be able to recommend a product to a friend
shopping
Start
configure solr for full text searching
search
Start
Backlog
57 • 11 - 17 Dec
10 points
When shopper submits order, authorize total product amount from payment gateway
checkout, needs discussion, shopping
Start
If system fails to authorize payment amount, display error message to shopper
checkout, design, shopping
Start
If authorization is successful, show order number and confirmation message to shopper (SW)
checkout, shopping
Start
Send notification email of order placement to admin
admin, checkout, shopping
Start
Shopper should be able to check status of order by entering name and order number
orders
Start
Start
Admin can review all order questions and send responses to shoppers
admin, orders
Start
Set up Engine Yard
Start
My Work
Shopper should be able to search for product (SW)
search, shopping
Start
If authorization is successful, show order number and confirmation message to shopper (SW)
checkout, shopping
Start
Admin should be able to export orders as CSV file, based on date range and order status (SW)
admin, reporting
Pivotal Tracker Dashboard

Tracker KB Content

STORIES | ANALYTICS | SETTINGS | MEMBERS

Send us feedback!

- Project
- Iteration
- Epics
- Releases & Burndowns
- Story Activity
- Cycle Time
- Auto-Refresh

Tracker KB Content Overview

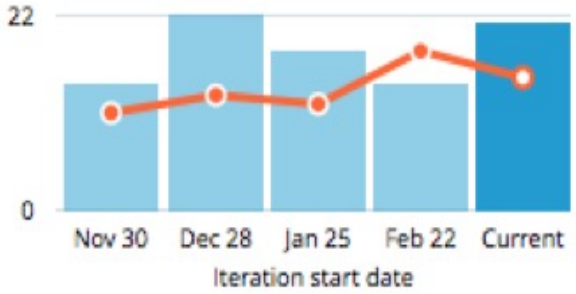
Project trends | Burnup | Cumulative flow

Velocity and points accepted

view report

21 points

15 pts velocity



Stories accepted

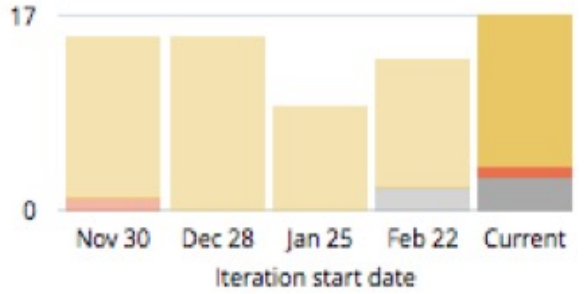
view report

17 stories

13 features

1 bug

3 chores



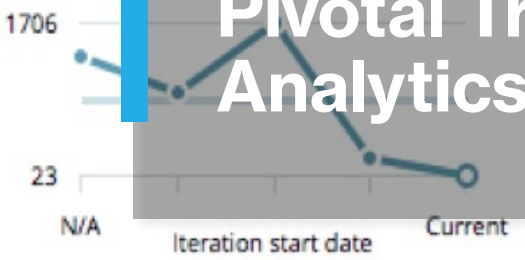
Story cycle time

view report

Typical time between Start and Accept

23 hrs per story

5-iteration avg: **844 hrs**



Rejection rate

view report

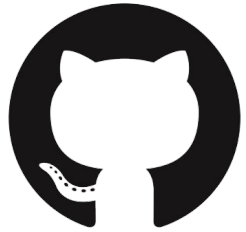
Percentage of rejections vs. acceptances and rejections

5.7%

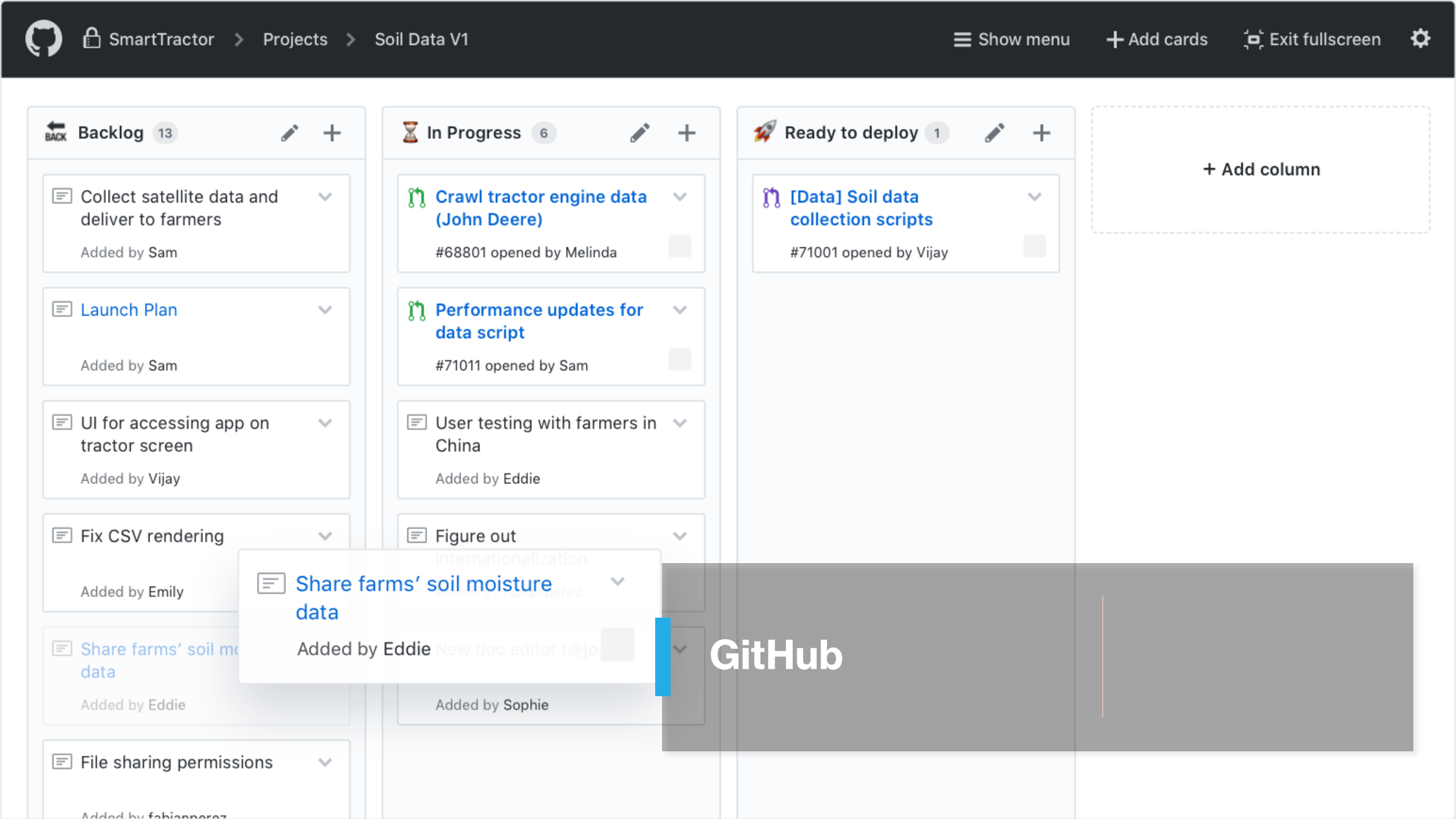
5-iteration avg: **22%**

Pivotal Tracker Analytics

GitHub

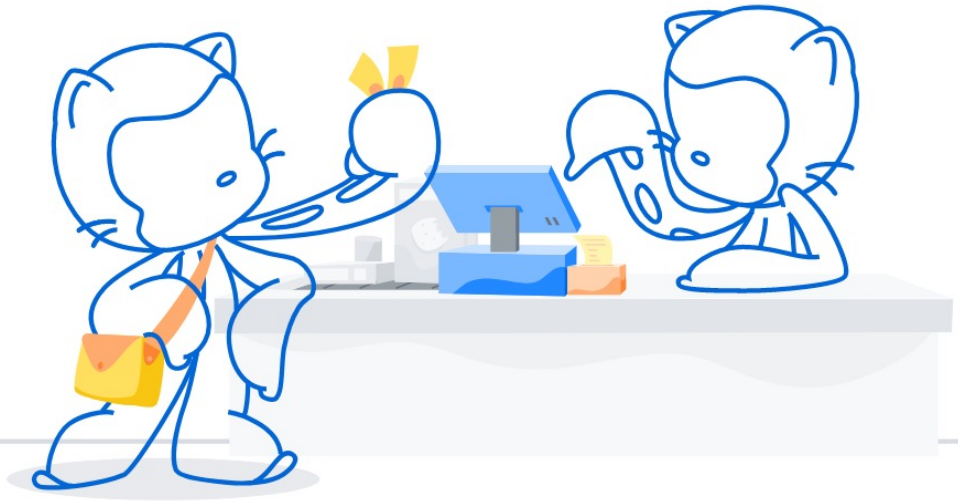


- GitHub is more than a code repository
- Project management features
  - Team management
  - Issue tracking
    - Could follow similar principles as stories
- Community continuously creating new apps
  - For personalized management
- Can GitHub replace the Pivotal Tracker?


SmartTractor
Projects
Soil Data V1
Show menu
Add cards
Exit fullscreen
Backlog 13
In Progress 6
Ready to deploy 1
+ Add column
Collect satellite data and deliver to farmers
Added by Sam
Launch Plan
Added by Sam
UI for accessing app on tractor screen
Added by Vijay
Fix CSV rendering
Added by Emily
Share farms' soil moisture data
Added by Eddie
File sharing permissions
Crawl tractor engine data (John Deere)
#68801 opened by Melinda
Performance updates for data script
#71011 opened by Sam
User testing with farmers in China
Added by Eddie
Figure out internationalization
Added by Sophie
[Data] Soil data collection scripts
#71001 opened by Vijay
GitHub

GitHub Marketplace

- Apps to integrate in GitHub projects
- Different categories
  - Code review
  - Continuous integration
  - Security
  - Testing
  - Monitoring
  - Among others



Software Planning

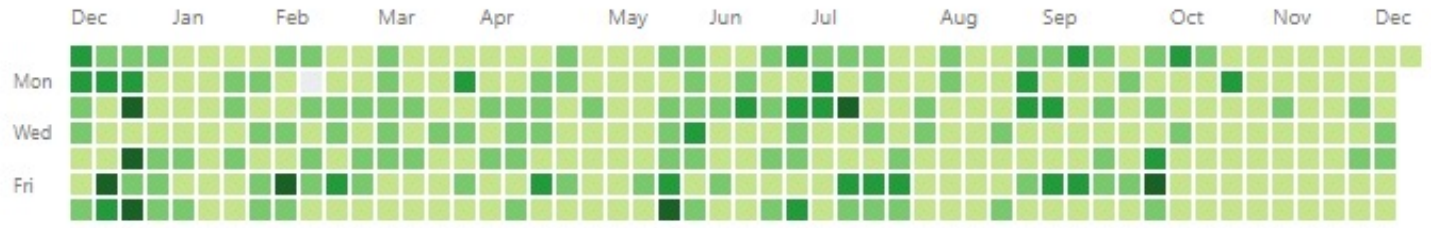
- Specification
  - Defining what the system should do;
- **Design and implementation**
  - **Defining the organization of the system and implementing the system;**
- Validation
  - Checking that it does what the customer wants;
- Evolution
  - Changing the system in response to changing the customer needs;

Feature-branching workflow

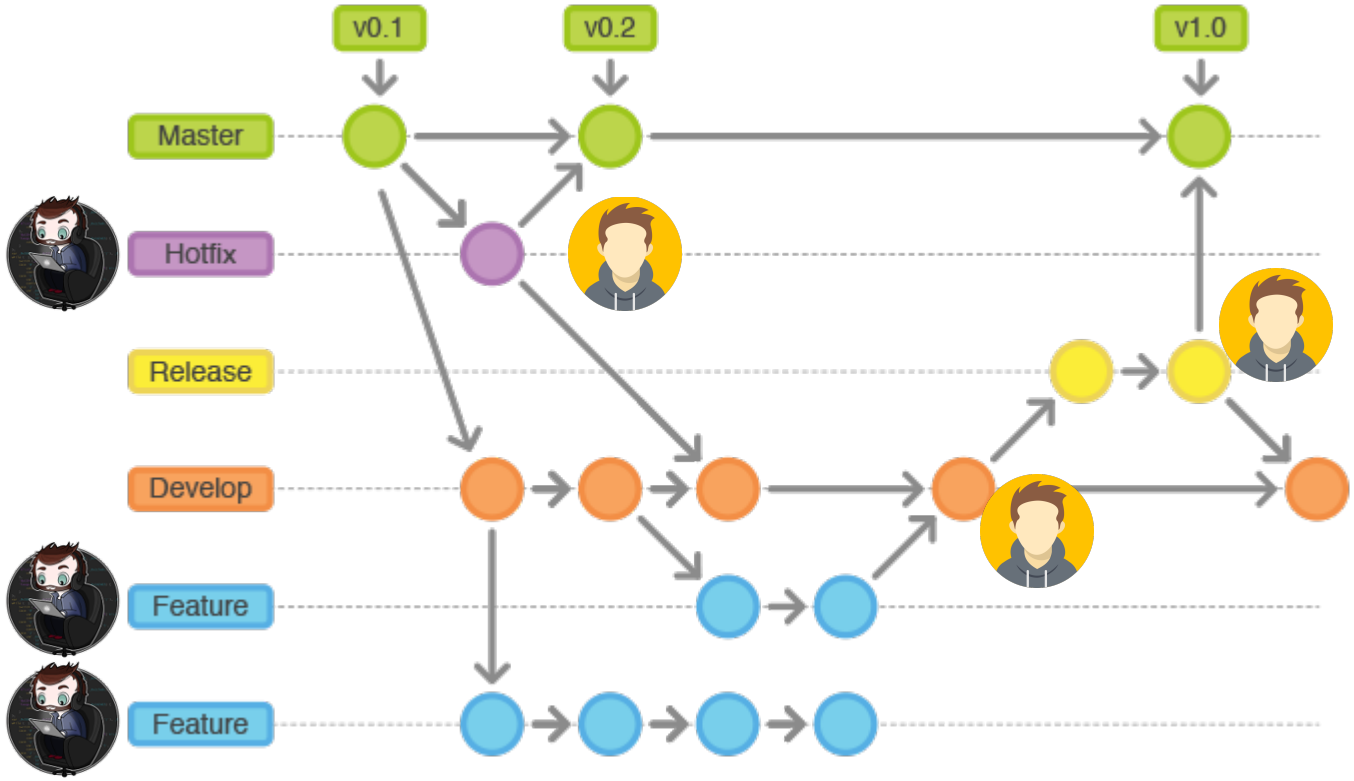
- Code repositories
  - Version control system
    - Git
- Not new for you, but...
- Let's see about good practices

Daily commits

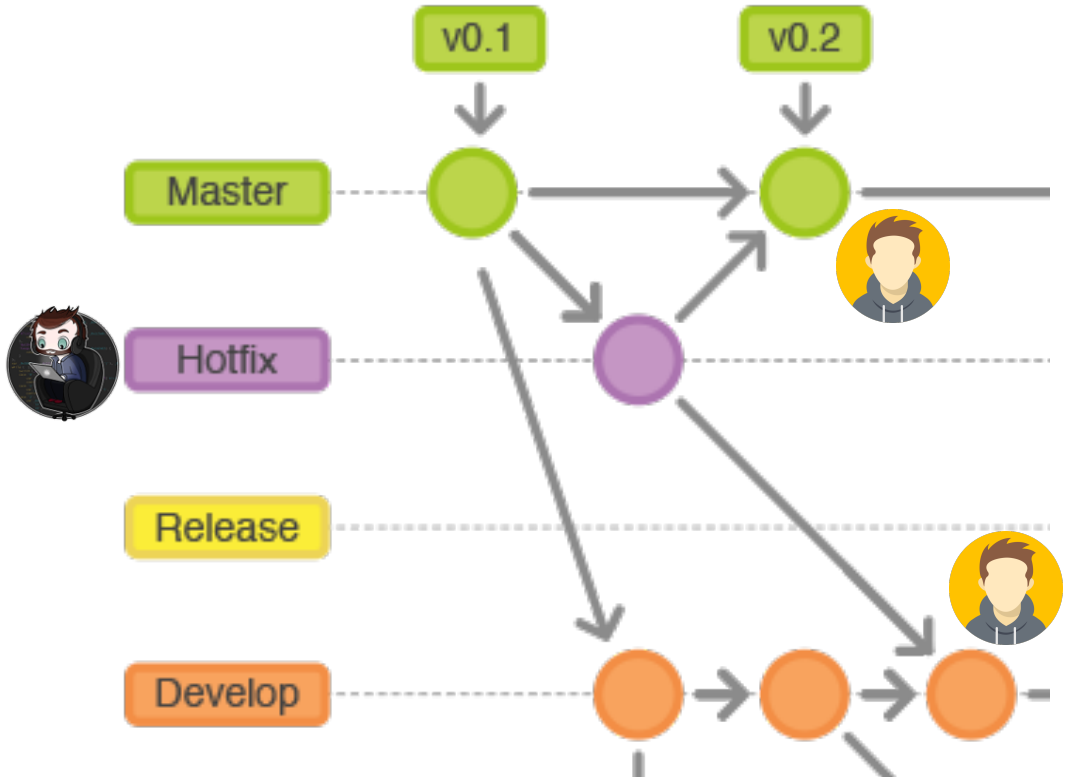
- Scenario
  - Working on a project for two weeks without doing a single commit. The disk decides to die. What should we do now?
- Never wait to finish a task to create a commit
- Every day, commit the work and push the code to the repository



Git workflow

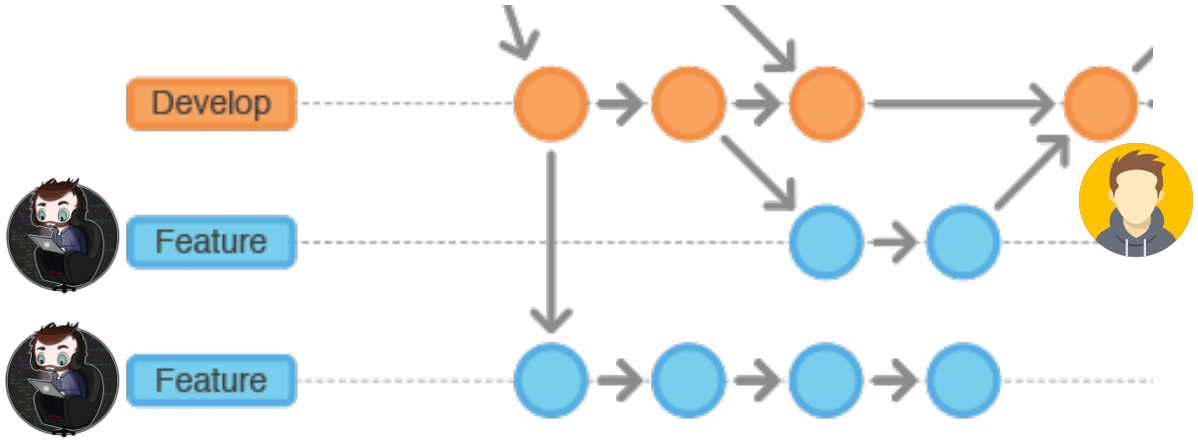


Hotfix



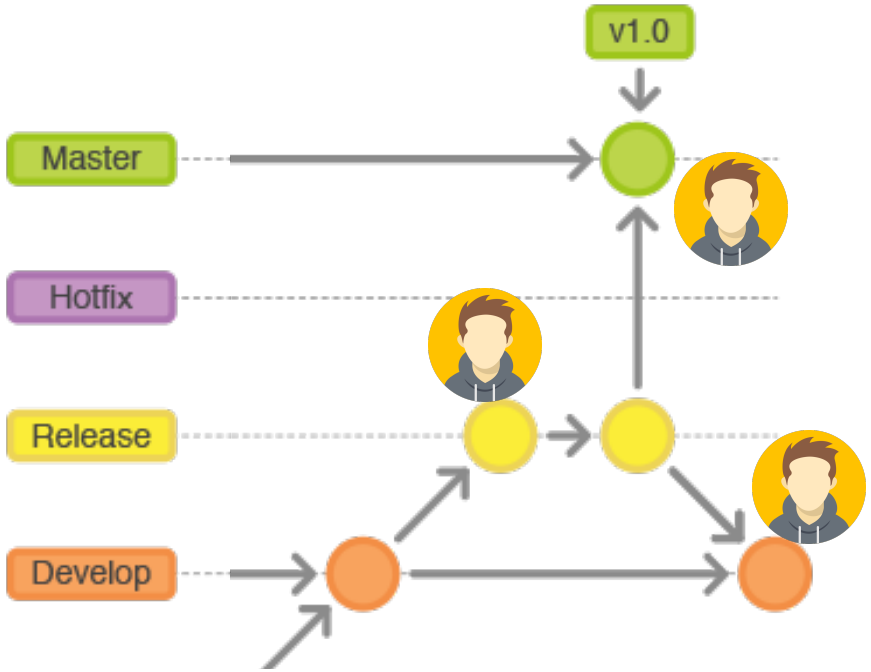
- Catastrophic bug was found
- Procedure
  1. Checkout master
  2. Fix bug
  3. Merge into master and dev
- Does not require a new branch for a release

New Feature



- **New branch** for each feature
- Checkout from dev
- When feature is complete
  - Merge dev into feature branch
  - Merge branch into dev
- Why these two merges?

New Release



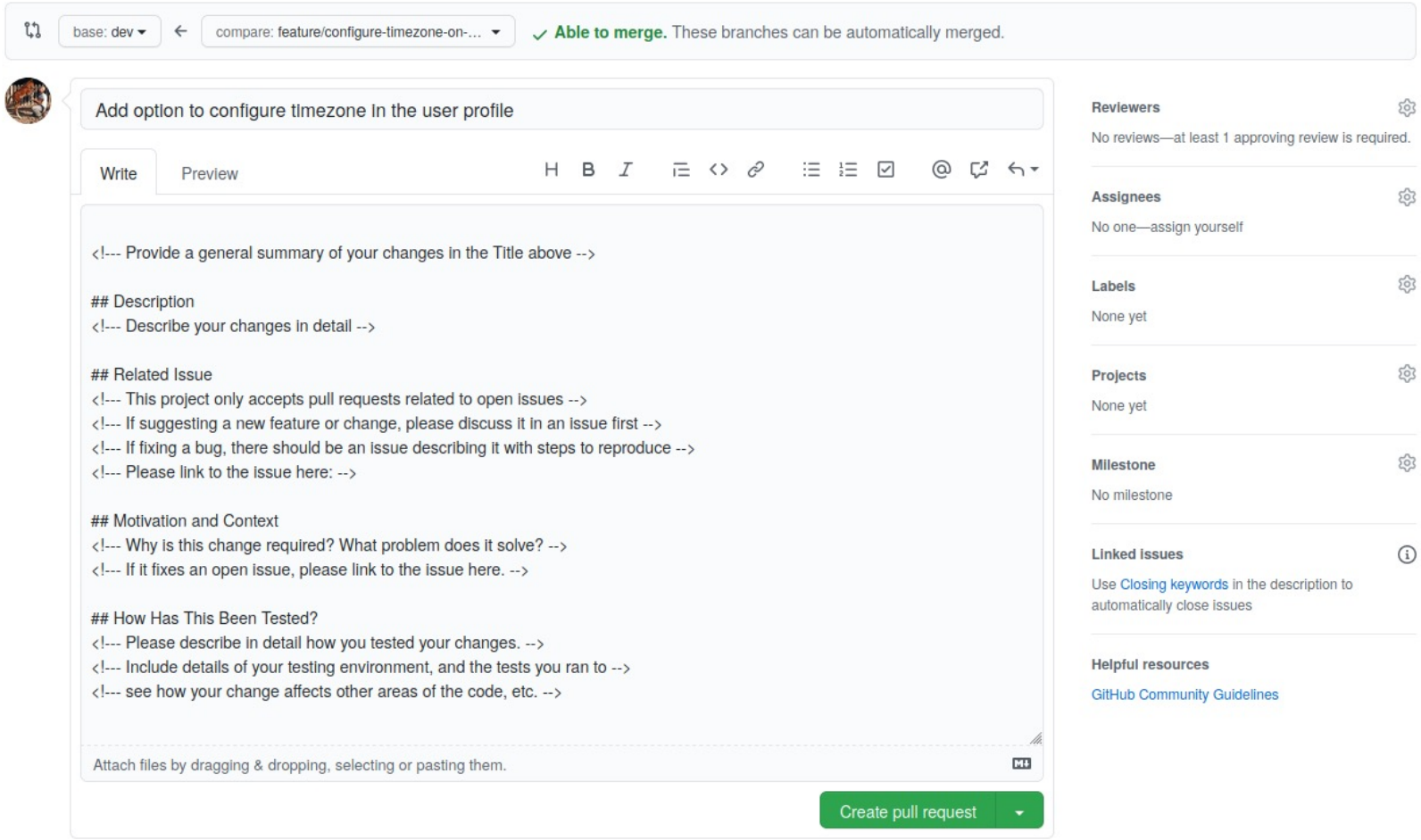
- Preparing the product to show the client
  - Closing one development cycle
- Checkout from dev
- When release is ready
  - Merge release into master
  - Merge branch into dev

- Why these two merges?

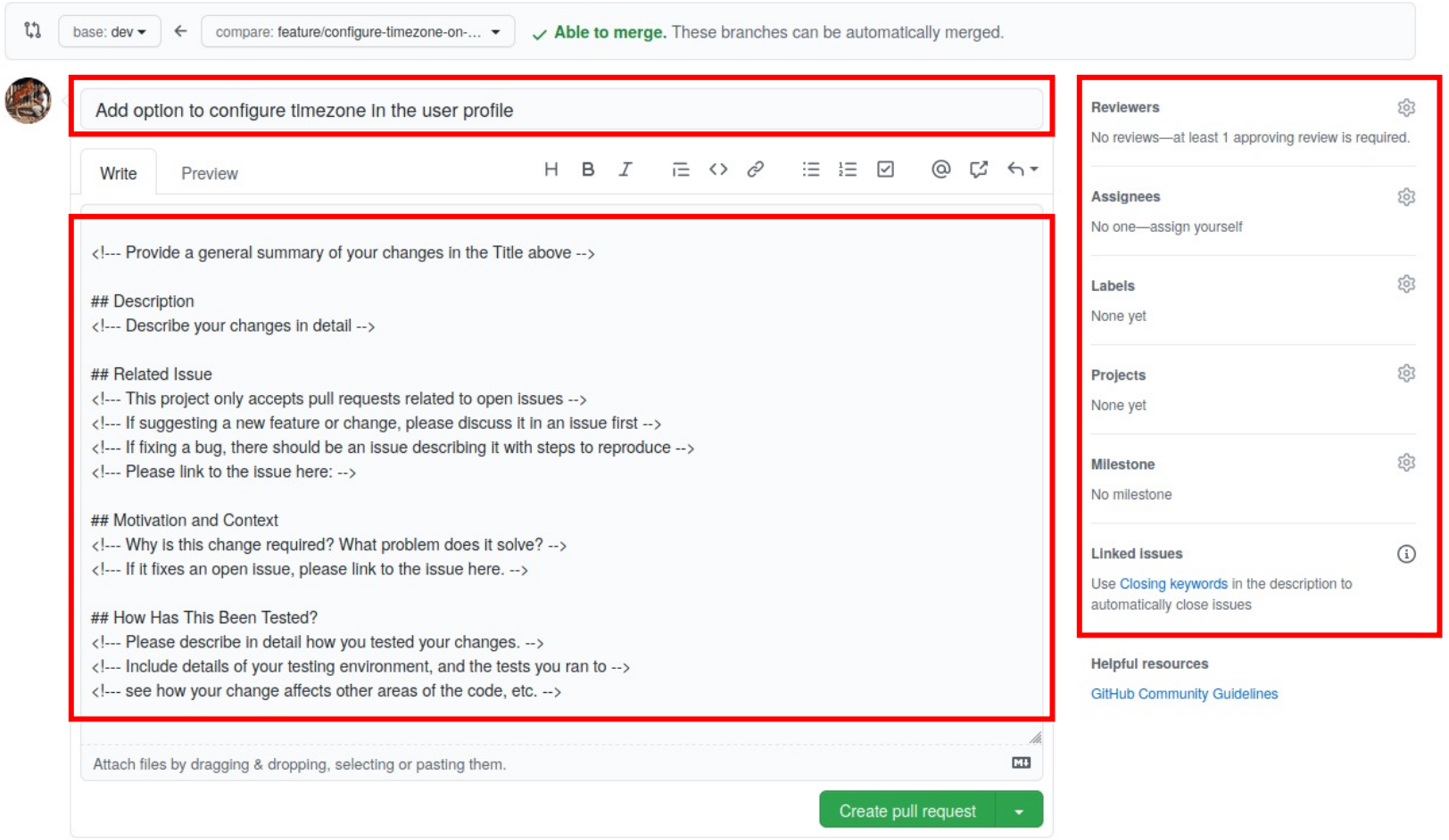
Pull/Merge Requests

- Merging branches needs a request
  - Usually to protected branches (master and dev)
- Pull request needs approval
  - From git manager (DevOps master)
- Sometimes the implementation needs improvements
  - Feature is incomplete
  - Complex conflicts during merging

Pull/Merge Requests



Pull/Merge Requests



Branching Names

- Each programmer likes his own convention
  - Which is bad in team projects
- These conventions are not standards
  - But choosing one helps in the repository management
- Branch names are important
  - Like good names when coding variables

Branching Names

CATEGORY	DESCRIPTION
bug	Bug fixing
imp	Improvement on already existing features
new	New features being added
wip	Works in progress - Big features that take long to implement and will probably hang there
junk	Throwaway branch created to experimentation
release	New release before merging with master

Examples Branching Names

- URL redirects to the wrong page #123
  - bug/fixURLRedirect (good)
  - bug/fix_url_redirect (also good)
  - bug/fix_url_redirect_123 (better)

- Accounts: URL redirects to the wrong page #123
  - bug/accounts/fix_url_redirect_123 (much better)

Troubleshooting

- Merging conflicts that are too complex
  - Request the developer to update branch
  - Merging current dev into branch
- Committed sensitive data
  - It is possible to revert
  - But it could be a dangerous procedure
- Dependencies
  - "It works in my machine"

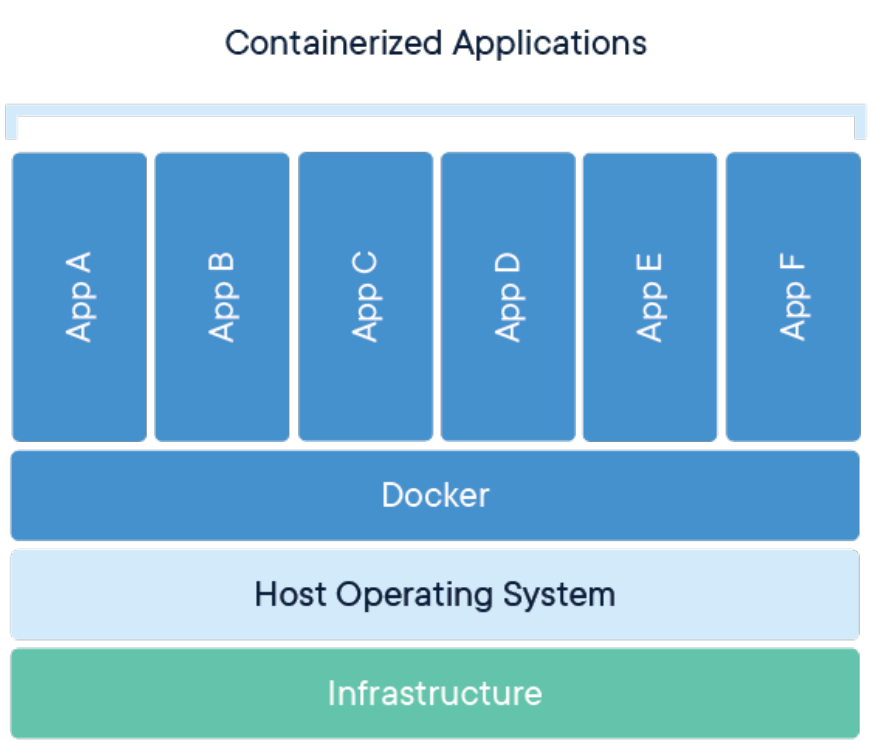
Troubleshooting

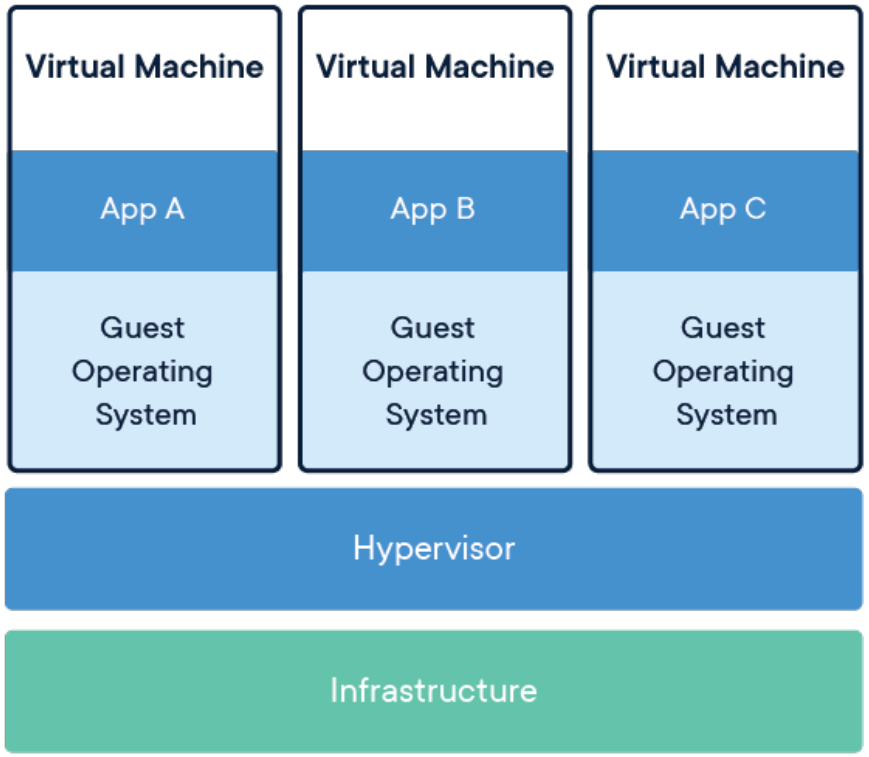
- Merging conflicts that are too complex
  - Request the developer to update branch
  - Merging current dev into branch
- Committed sensitive data
  - It is possible to revert
  - But it could be a dangerous procedure
- Dependencies
  - "It works in my machine"

Containers-based deployment

- A good solution for dependency problems
- Everyone is using the same environment
- Production and development environments are very similar
- Simplifies the integration of different services
- Easy to deploy

Virtualization & Containerization





Virtualization & Containerization

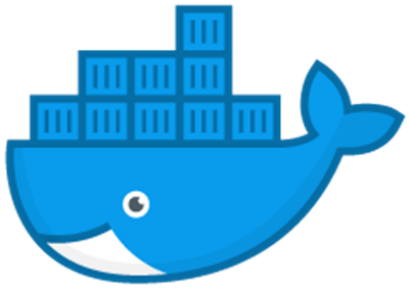
Virtualization	Containerization
More secure and fully isolated	Less secure and isolated at the process level
Heavyweight, high resource usage	Lightweight, less resource usage
Hardware-level virtualization	Operating system virtualization
Each virtual machine runs in its own operating system	All containers share the host operating system
Startup time in minutes and slow provisioning	Startup time in milliseconds and quicker provisioning

Virtualization & Containerization

- Both have positive aspects
- Containerization is better
  - For the class project
- Real world uses hybrid approaches

Docker

- Already studied in practical classes
  - But let's review a few concepts
- Production and development images are the same
  - But with different configurations
- In production
  - **Always** use volumes for the sensitive data
  - Containers die, volumes not (usually)

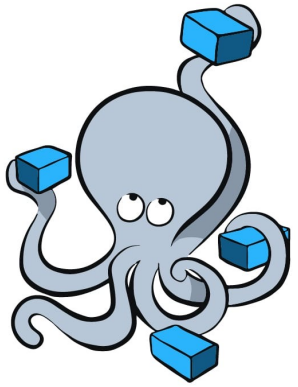


Images

- Images are the bases of containers
- One Image can serve multiple containers
  - But one container can only have one image
- Allows inheritance
  - FROM ubuntu:20.04
  - FROM myImage:base
- Should I use an official image or create mine?

Docker Compose

- Simplifies the integration between containers
- Allows container orchestration
  - Based on a certain order
- Do not change docker-compose.yml file
  - Instead, define variables (.env)
  - Create a .env-example file with the default variables
- After configured, the startup is trivial
  - docker-compose up -d



Putting all together

Example – client's needs

- Client wants a web application to generate random number
- Procedures:
  - User sets a seed
  - Clicks generate a random number
  - Random number is generated
- Let's plan this project

Product owner

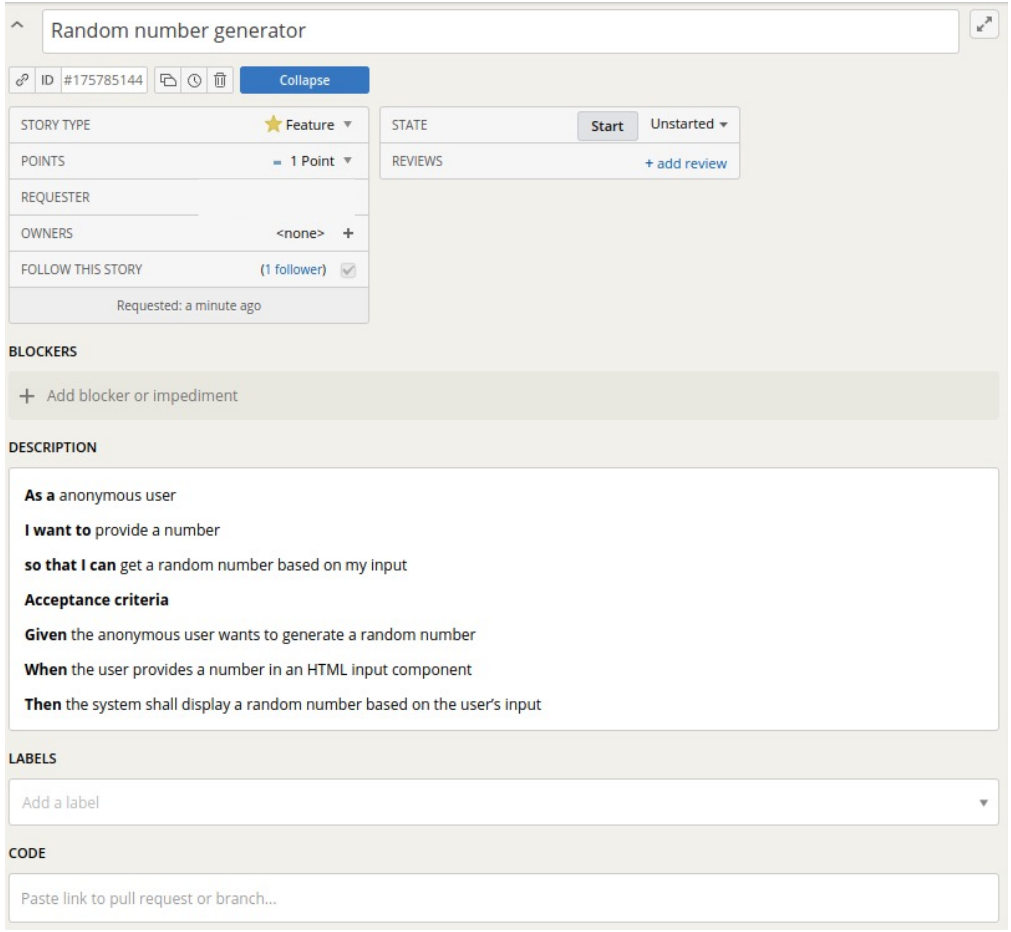
Story

Title: Random number generator **Priority:** 1 **Estimate:** 1

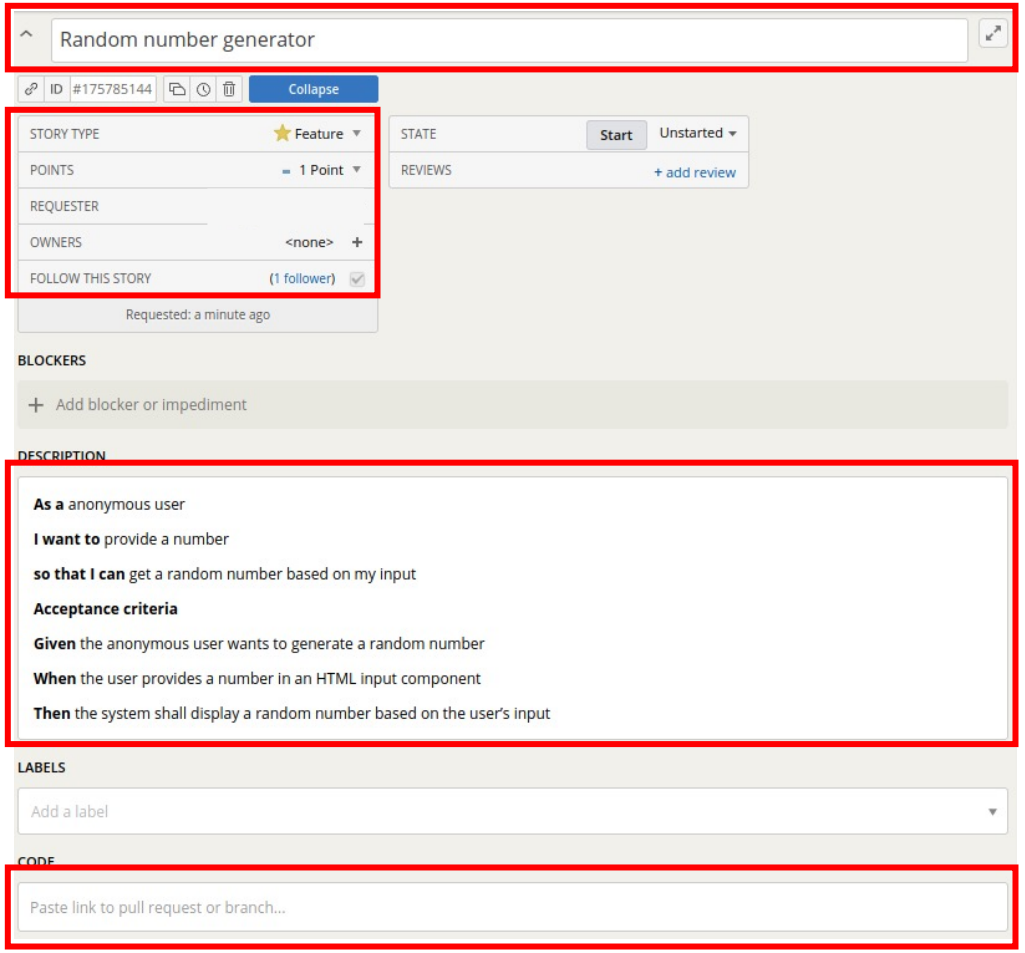
As an anonymous user
I want to provide a number
so that I can get a random number based on my input

Acceptance criteria
Given the anonymous user wants to generate a random number
When the user provides a number in an HTML input component
Then the system shall display a random number based on the user's input

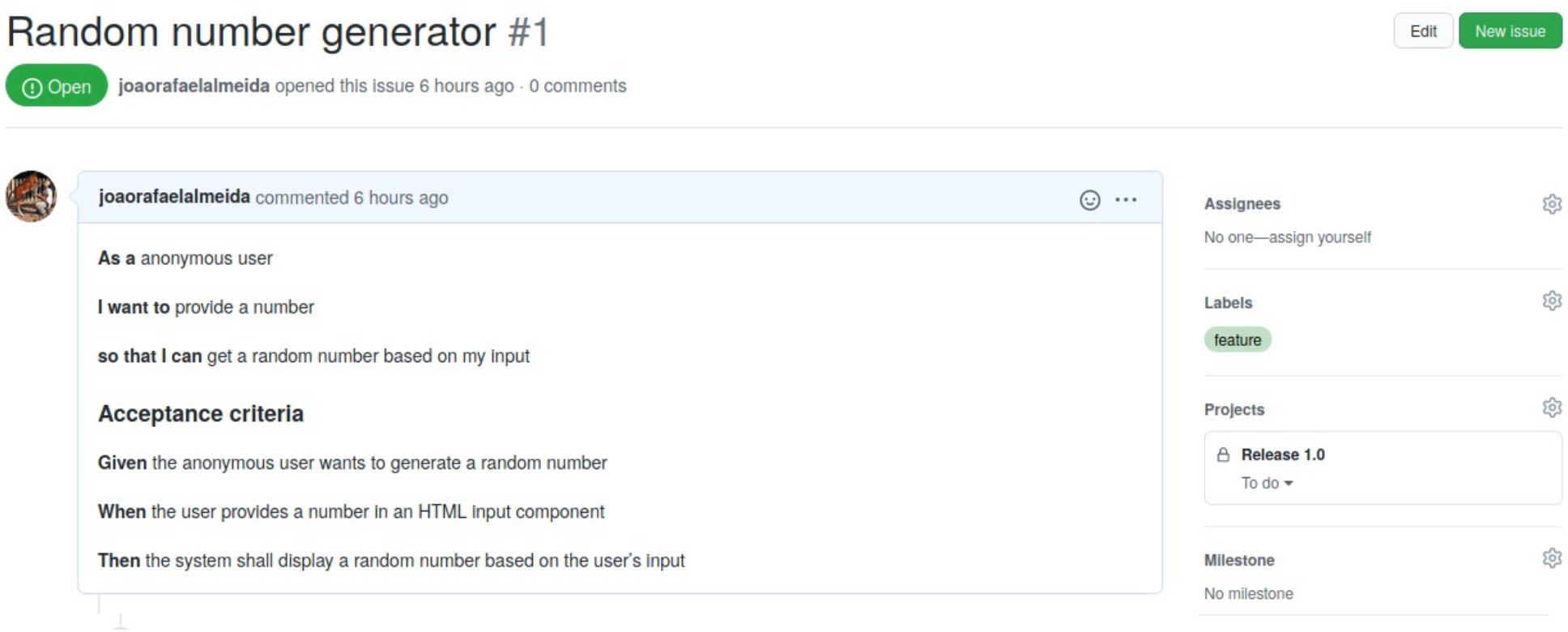
Pivotal tracker



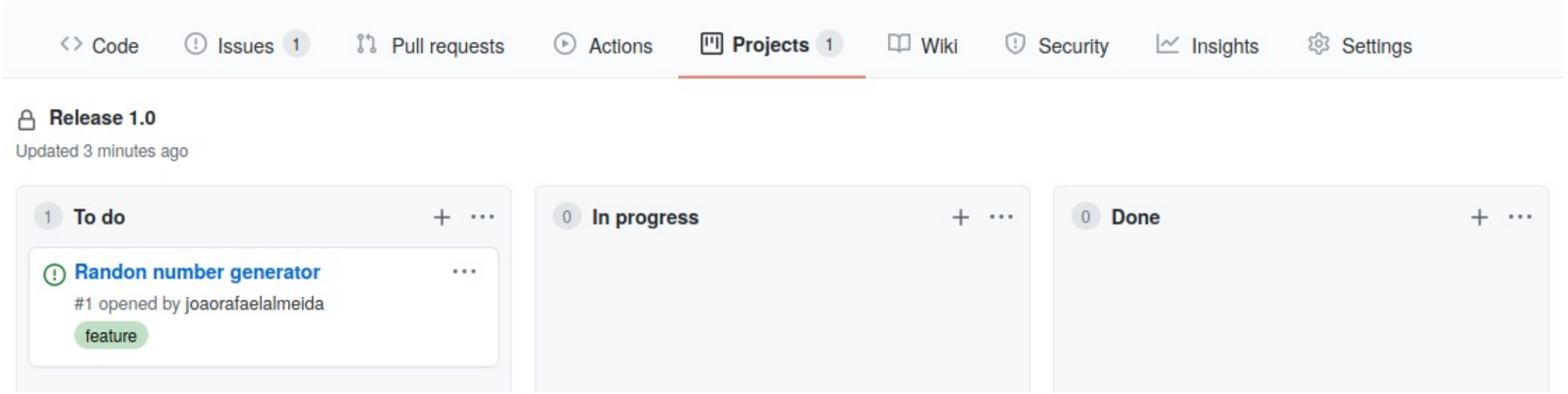
Pivotal tracker (story)



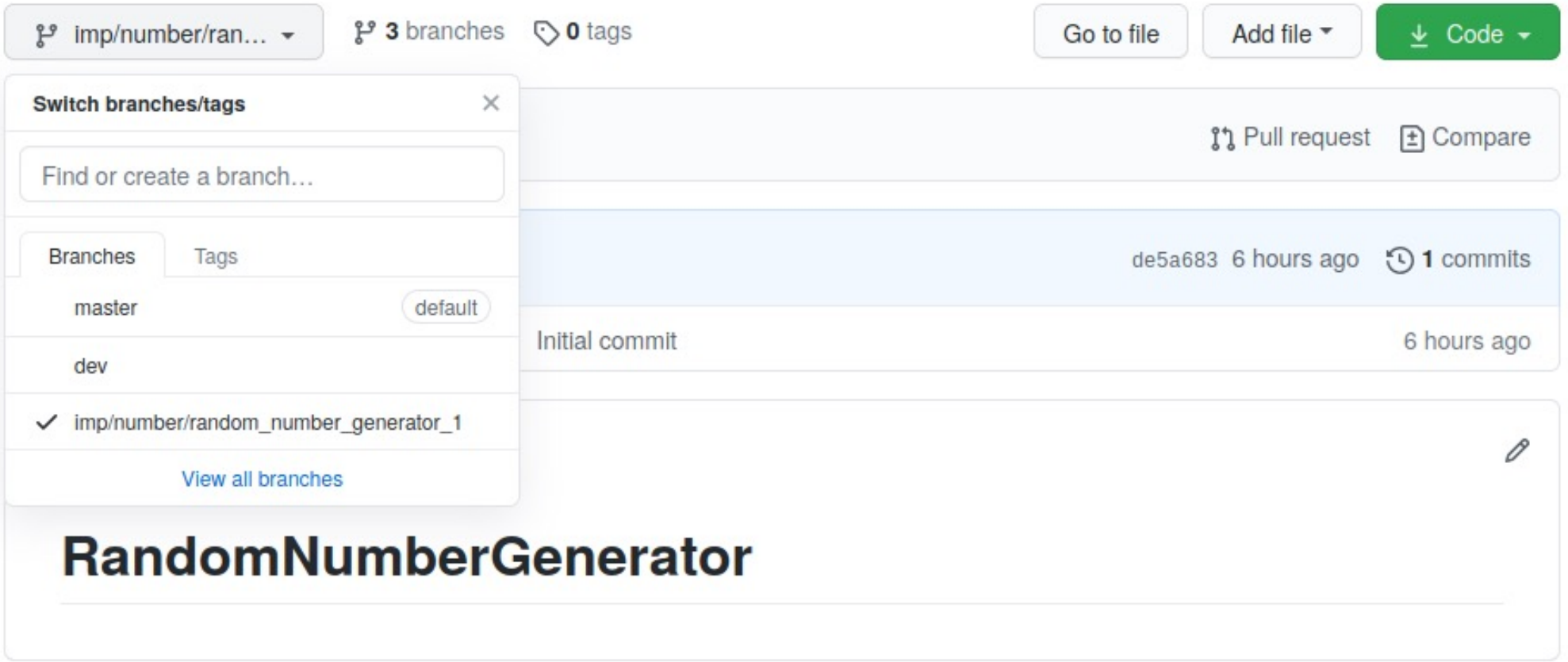
GitHub Issues (story)



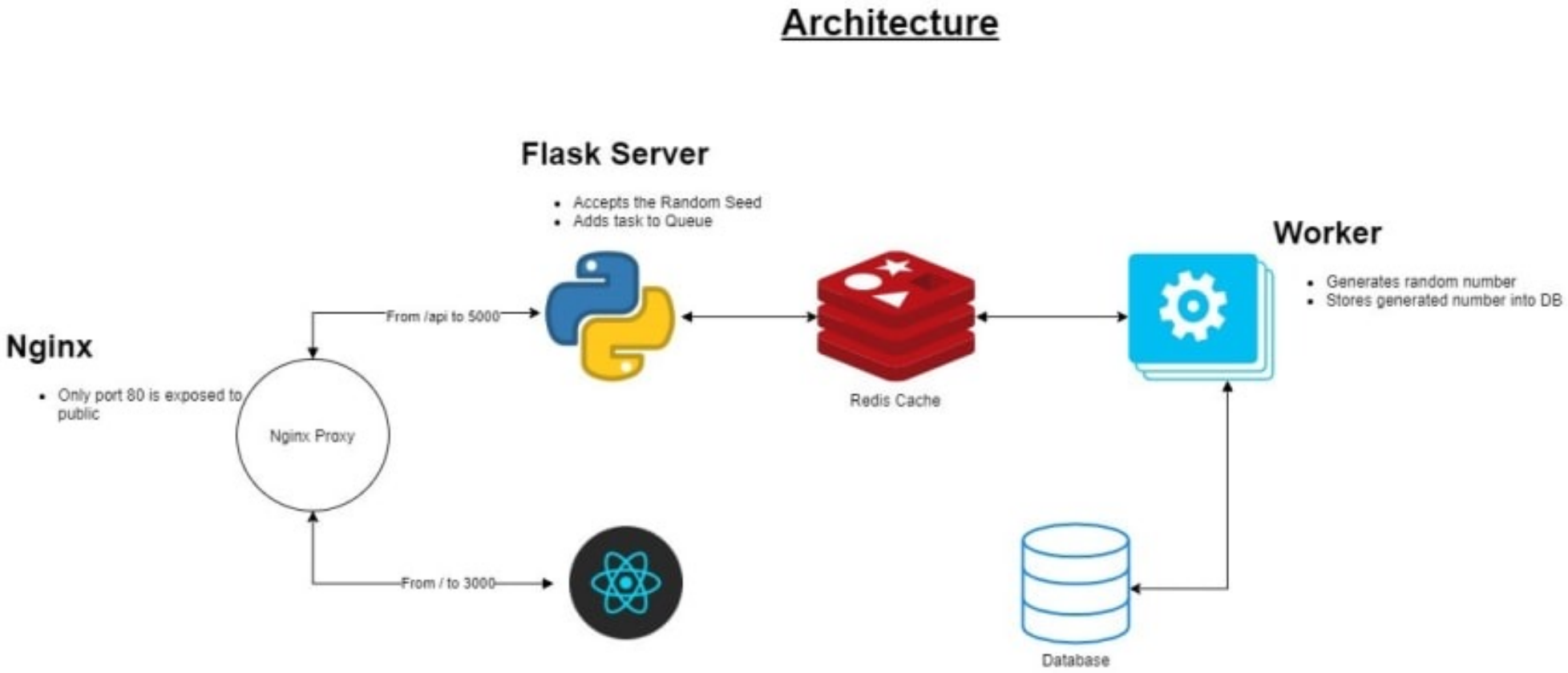
GitHub Dashboard



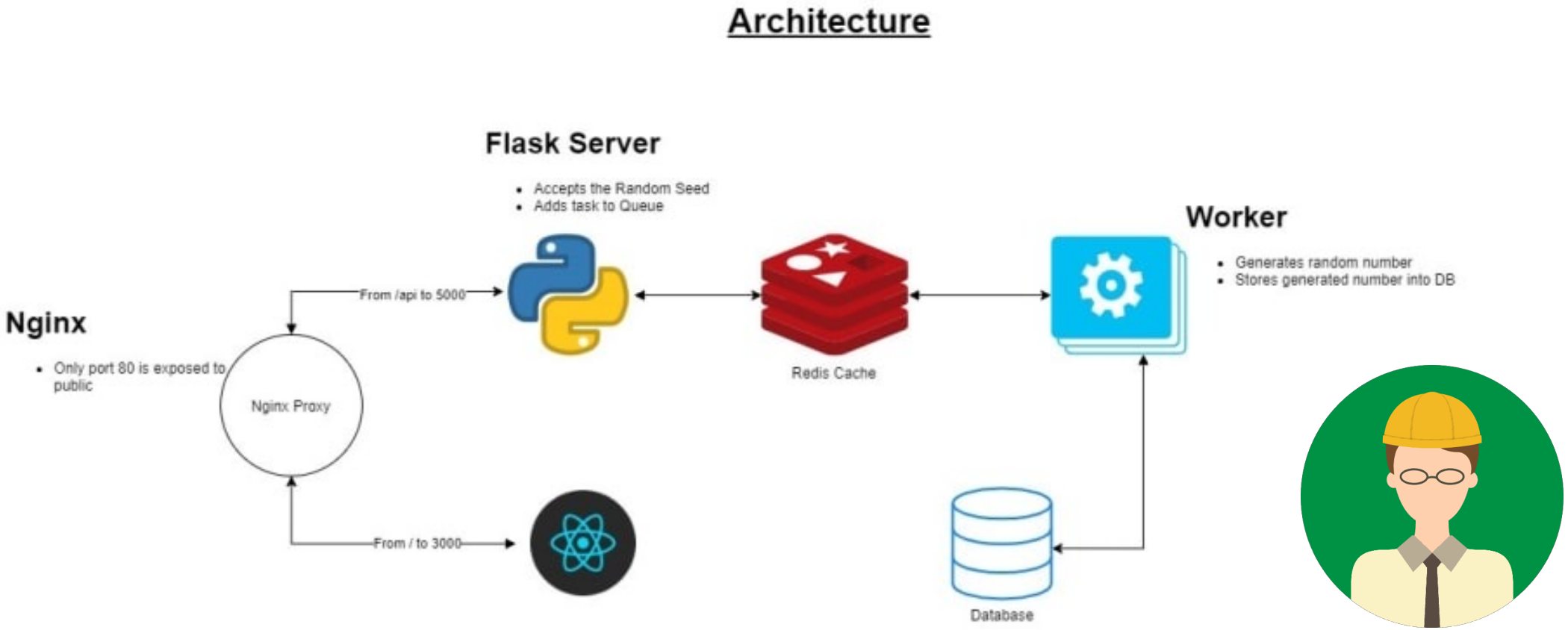
GitHub Code Repository



System architecture

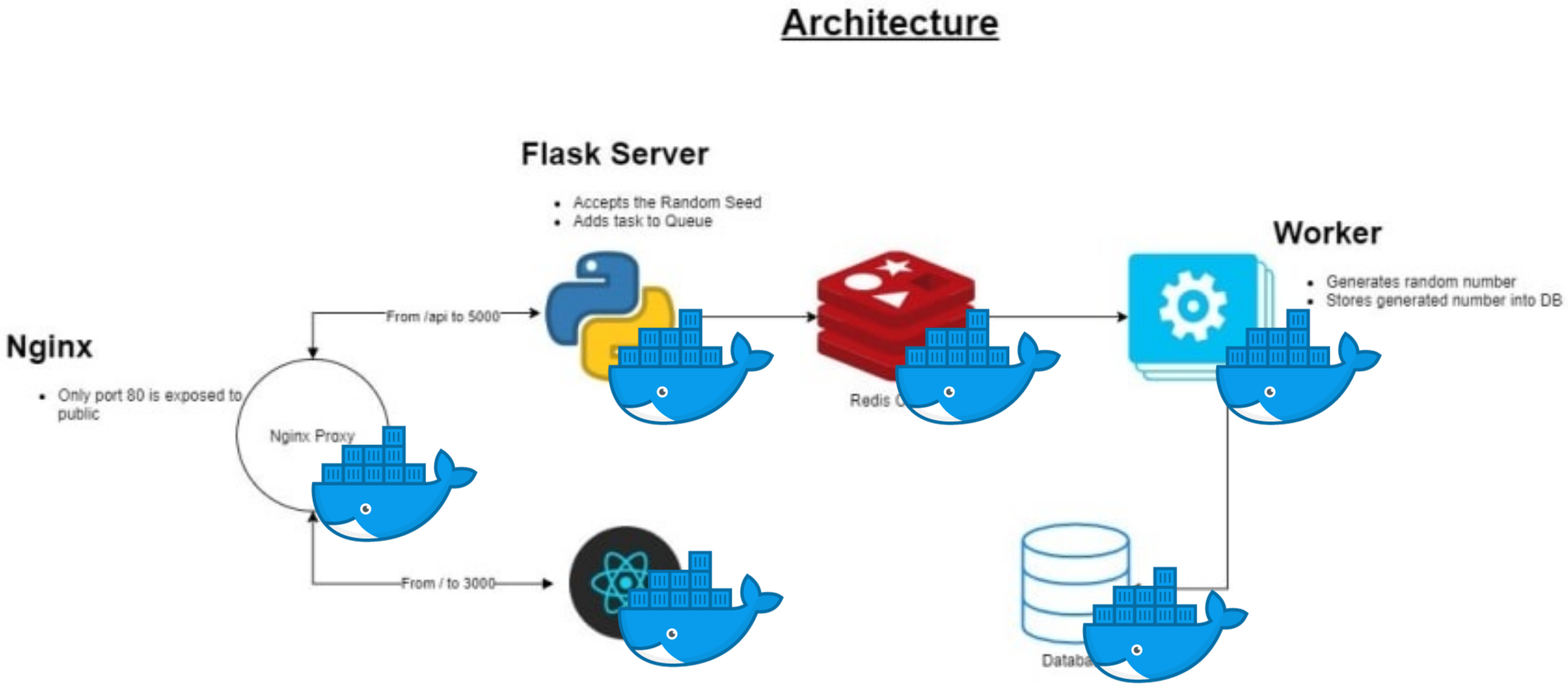


System architecture

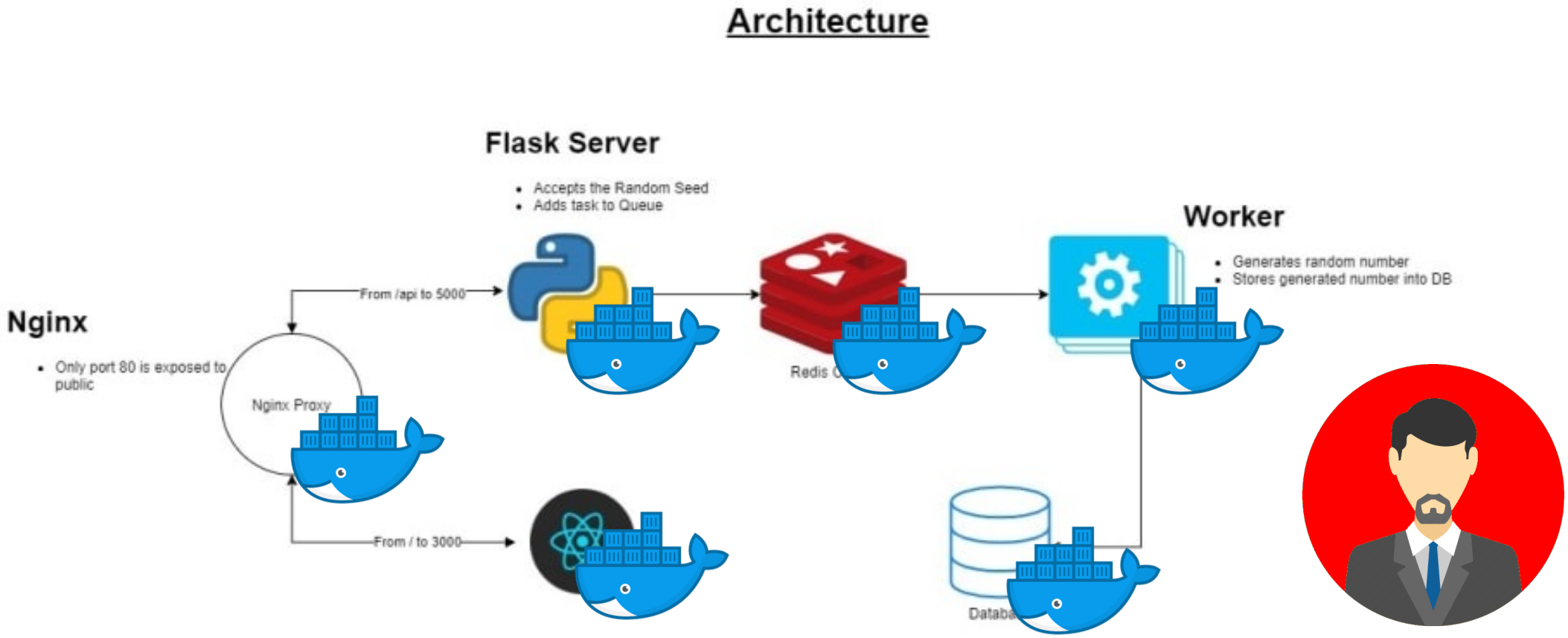


Architect

System architecture



System architecture



DevOps master

Docker Compose

```
services:
  proxy:
    container_name: proxy
    build:
      ...
    ports:
      - 80:80

  database:
    container_name: database
    build: database/.
    volumes:
      - ./database/db_data:/var/lib/postgresql
    ...

  client:
    container_name: client
    build:
      ...
    environment:
      ...
```

```
  api:
    container_name: api
    build:
    ...
    volumes:
      - ./api:/app

  worker:
    container_name: worker
    build:
    ...

  redis:
    container_name: redis
    build:
    ...
```

Docker Compose Best Practices

- Use a file for variables (**.env**)
  - Exposed port numbers
  - Volumes' paths
- Use docker health checks
  - To coordinate the right timings to run each container automatically
- Create a custom network
  - Avoid problems when deployed in different hosts
- Do not expose unnecessary ports

Who to blame if the project fails?

Who to blame if the project fails?

Developer