Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

IBM

CSU2013 - Requirements Presentation

GROUP 39

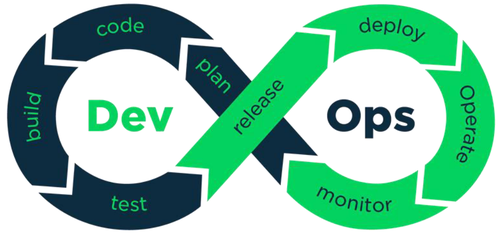
Cormac Madden
Tom Roberts
Neil Shevlin

Emer Murphy
Prathamesh Sai
Yi Xiang Tan

Our project:

8	IBM	A containerised CI/CD pipeline for Kubernetes / OpenShift applications using GitOps	Develop a persistent application (in a language of your choice) that consists of a front-end component, a back-end database and (optionally) a messaging system (Redis, MQ, Kafka, etc). Set up a CI/CD pipeline to BUILD, BAKE and DEPLOY your application to a container platform using GitOps methodologies and approaches (fully automated deployment). Include scanning of code, container images and production environments for best practices and security purposes in your automated process.

CI/CD Pipeline



Purpose of system

The purpose of building a CI/CD pipeline is to facilitate DevOps. Working together with IBM, we plan to build a DevOps pipeline to build and deploy an application on OpenShift.

Scope

In order to build this GitOps CI/CD pipeline project, there are a few things that must be developed first,

1. A web application using Node.js to test the deployment of the code
2. Automated testing and static program analysis
3. Automated deployment to Red Hat OpenShift

CI/CD Pipeline

Project objectives

➔ Familiarise ourselves with the technologies available to build a CI/CD pipeline.

➔ Decide on the suitable infrastructure to build the pipeline

➔ Develop a rudimentary application to develop our pipeline with.

➔ Create a pipeline using GitHub Actions that will:

- Build the application
- Package our application in the form of a container using podman
- Deploy the application to openshift

➔ Create clear documentation of the plan and the process.

➔ Deliver the code bundle to the client before April 23rd

Success criteria

➔ Completing the objectives outlined.

➔ Satisfying our client.

➔ Ultimately having a pipeline that will build and deploy an application on to OpenShift

Current System

Resources available

The pipeline for building a node.js application, perform tests and deployment is widely available.

CI solutions

GitHub Actions

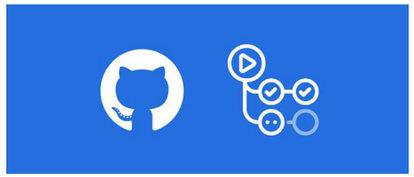
A group member worked with GHA in the past, and it's simple to set up.

Travis CI

Ideal for open source projects that require testing in multiple environments.

Jenkins

Suited for larger projects that require a high degree of customisation.



Version-Control platform

GitHub

Distributed version control system.

SVN

A centralised version control system.

Containerisation

Podman

We are mandated to use this.

Docker

Podman is built on the considerations of ease of replacing Docker.

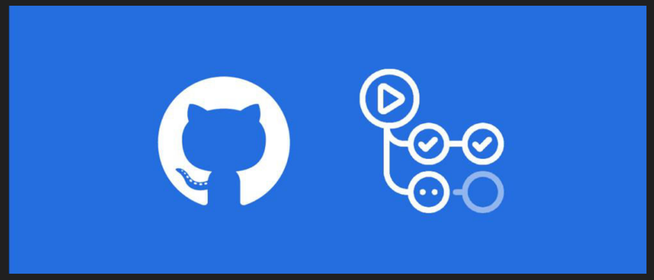
Proposed System

Overview

Our design is for a complete CI/CD pipeline and a simple application to demonstrate the use of this pipeline. Version control will be handled with GitHub, and we'll use GitHub Actions as the Continuous Integration server. There won't be a database, instead replaced by an external API for demonstration. The simple application to test the effectiveness of pipelines will be a Webapp with Javascript and Node.js.

Functional requirements

- ➔ The pipeline should take code submitted by developers on GitHub, build it, then perform static analysis such as unit testing.
- ➔ The DevOps pipeline must package the application using podman or buildah, and then deploy the containers onto a Red Hat OpenShift Container Platform (OCP)
- ➔ The containers should then run individually on Linux systems using `podman`.



Proposed System

Non-functional requirements

➔ Agile

- Client meetup to demonstrate progress on a biweekly basis

➔ Reliability

- Performs checks and code analysis for builds

➔ Serviceability

- Clearly documented code
- Documentation of key architectural decisions

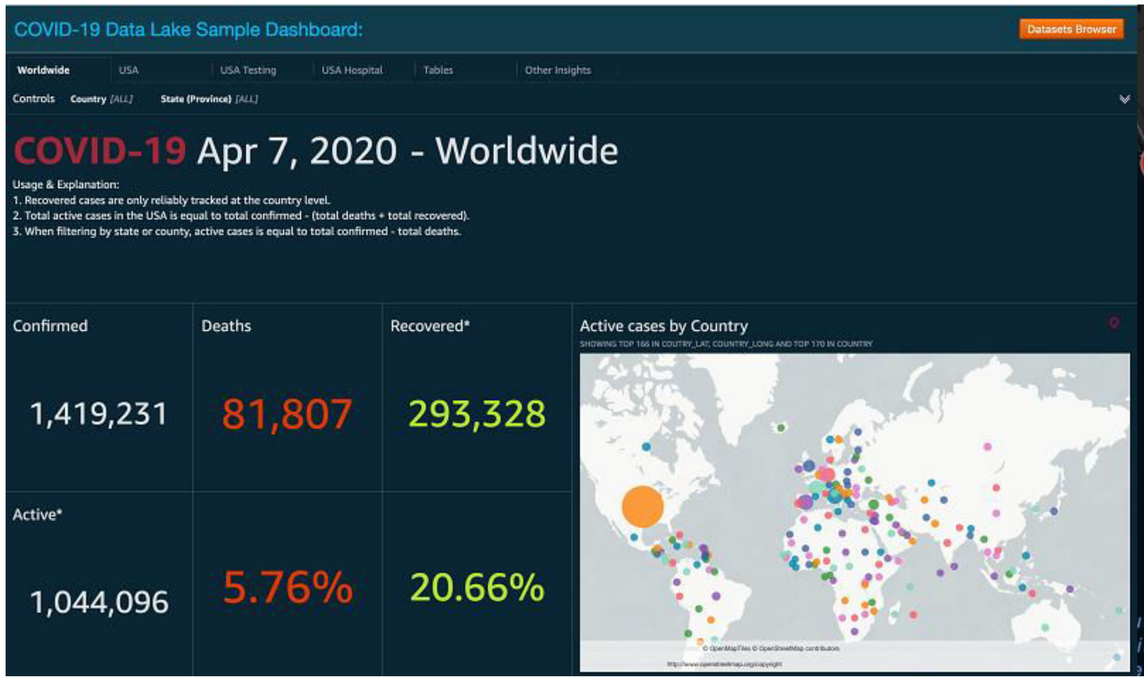
System Prototype (models)

User interface mockups

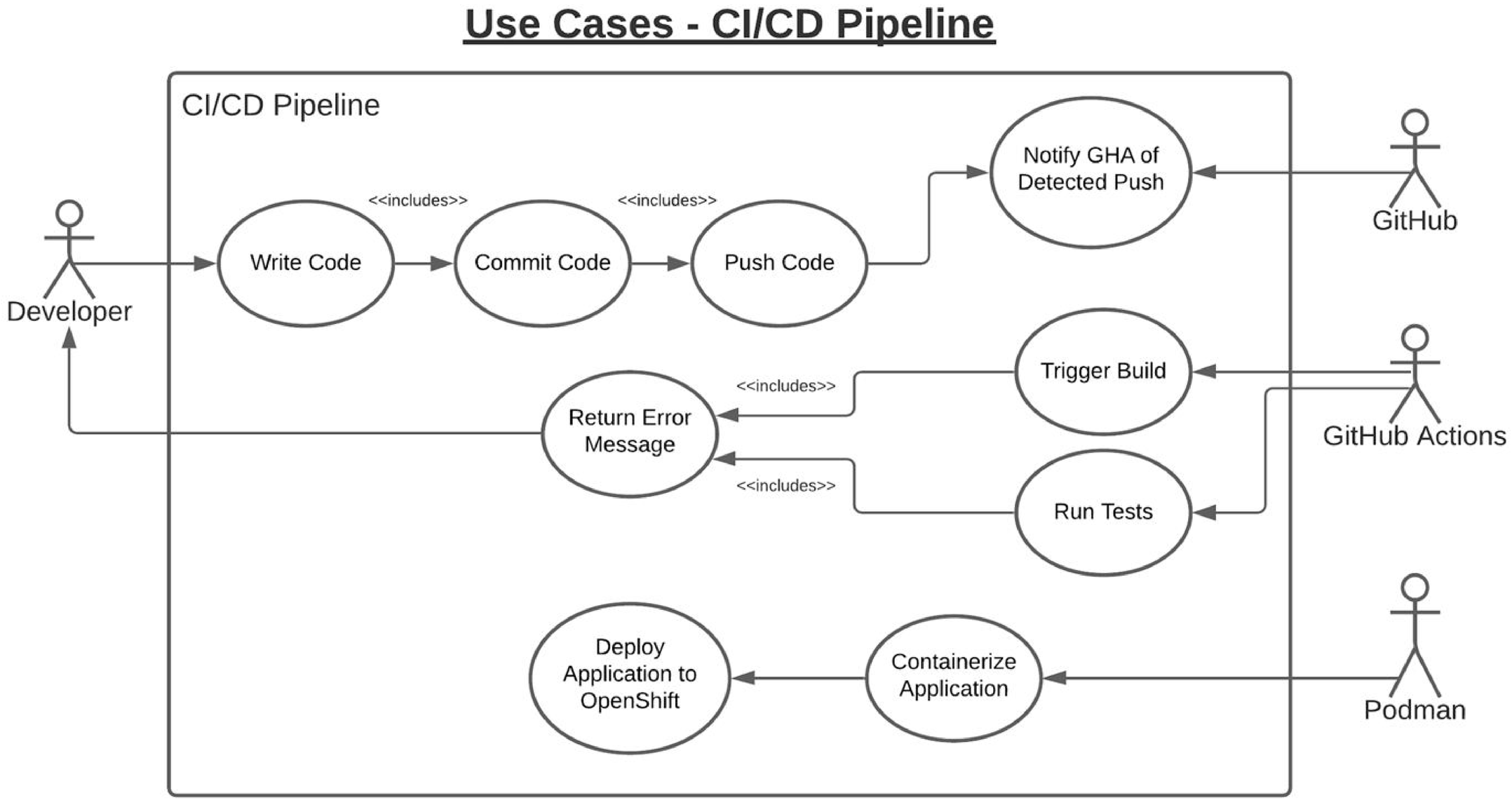
➔ Simple application to show information about Covid-19.

➔ Show information about Covid for each country.

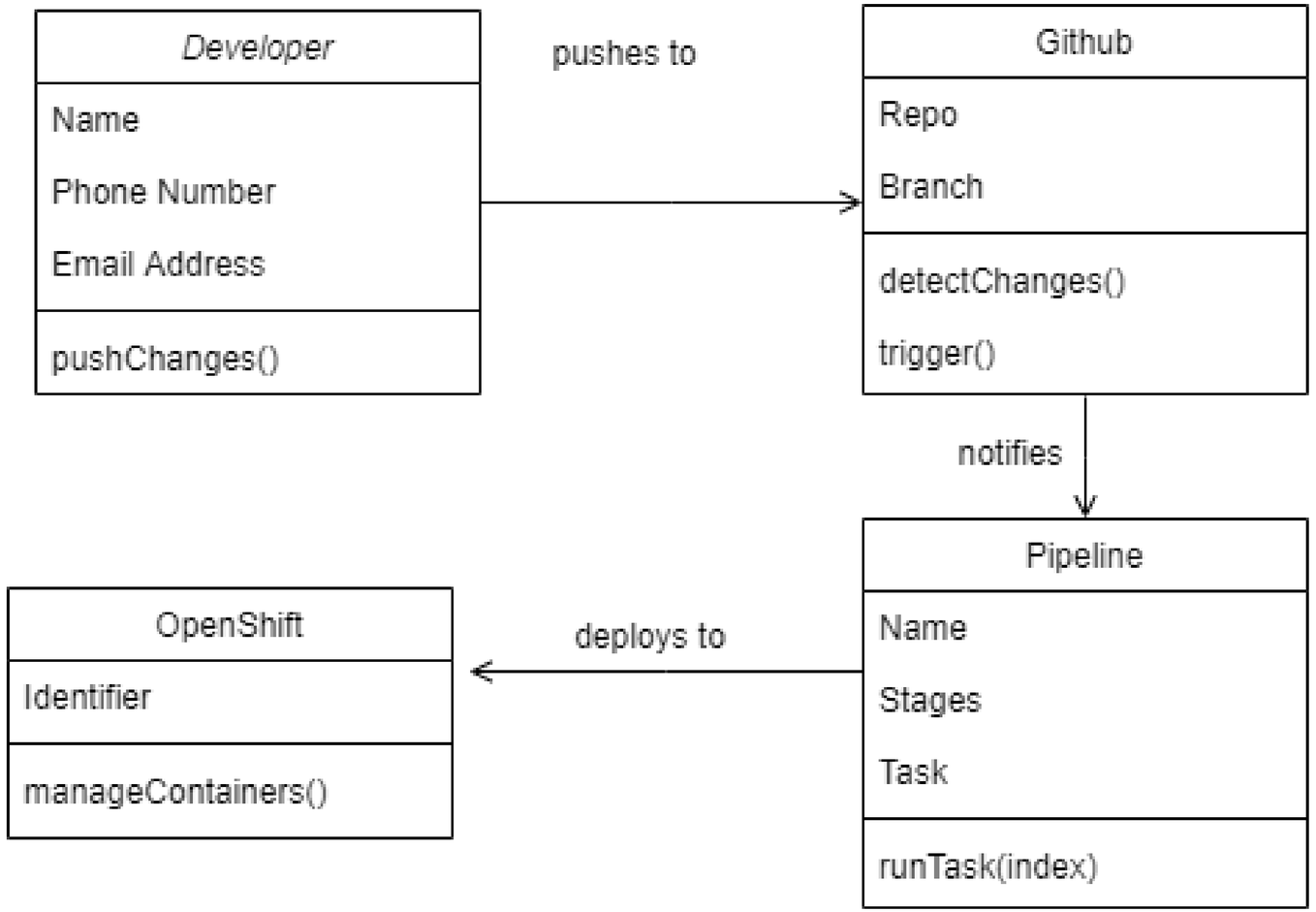
➔ We will use an API for all the information that we'll show.



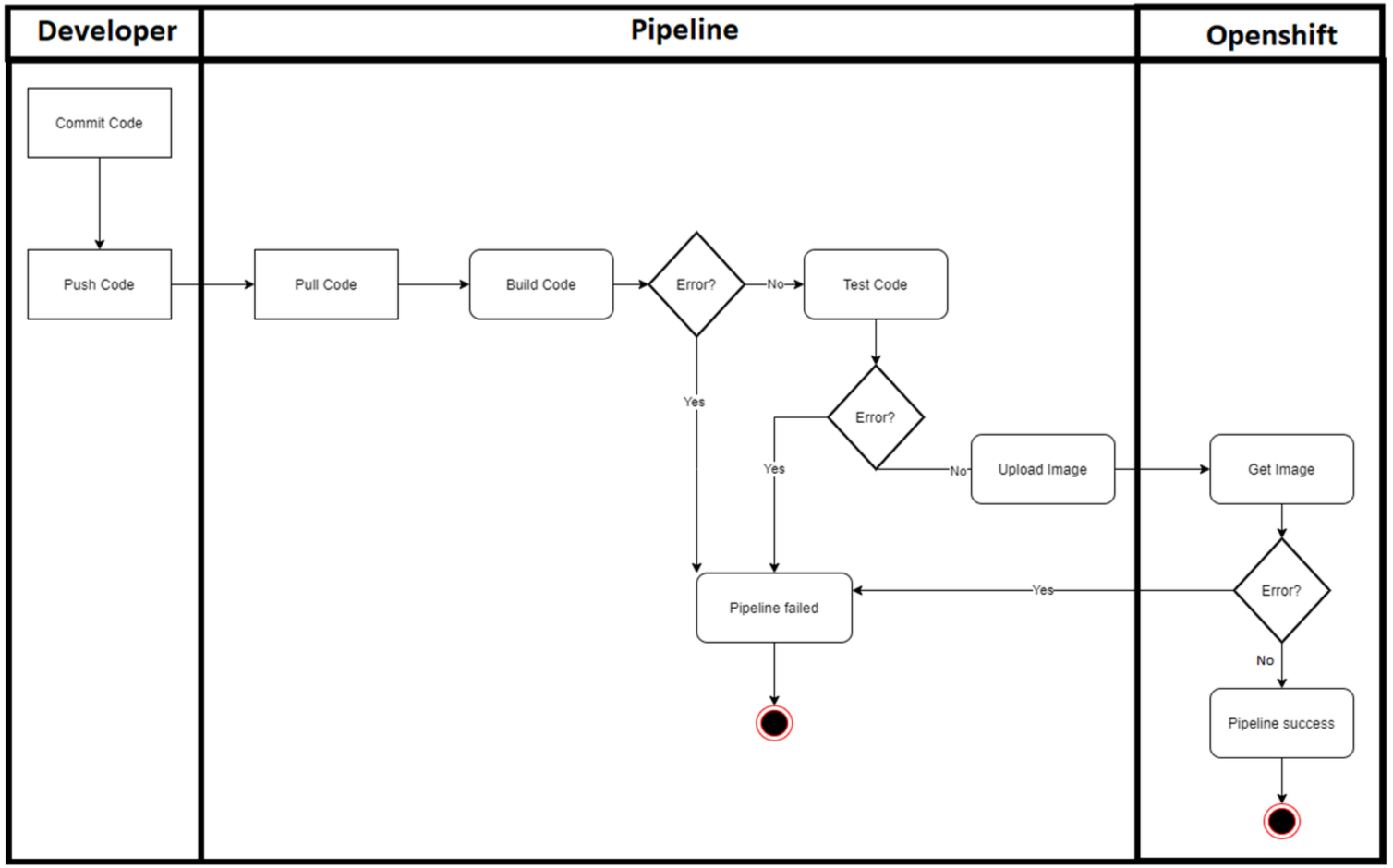
Use case diagram



Object model



Swimlane diagram



Thanks for listening!

Questions?