CI/CD

By: Carlos Murillo, Nicholas Dobard, and Norman Cattles (Pod 6)

What is Continuous Integration?

- CI is an approach to software development that automates repetitive and error-prone tasks that also promotes consistently integrated environments

- In simpler terms, it is the automated building and testing of your application on every push to a shared repository

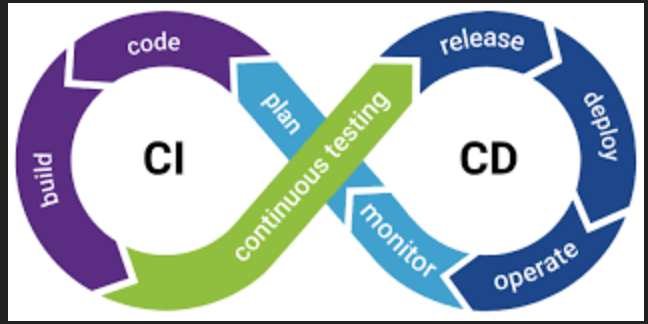
- Continuous integrations consists of 4 main steps:
  - Commit
  - Build
  - Test
  - Inform

What is Continuous Deployment?

- The automated deployment of successful code changes from a pre-production environment into production
- If all tests pass, every new commit deploys directly to production without the need for manual intervention
- Continuous Deployment consists of 3 main steps:
  - Deployment
  - Monitoring
  - Feedback

What is a CI/CD Pipeline?

- A sequence of automated steps to deliver software from development to production
- Key Stages
  - Source: Triggered by code changes (code push or pull request)
  - Build: Compiling and packaging the application
  - Test: Running unit, integration, and system tests
  - Deploy: Releasing the application to staging or production



What are the Benefits?

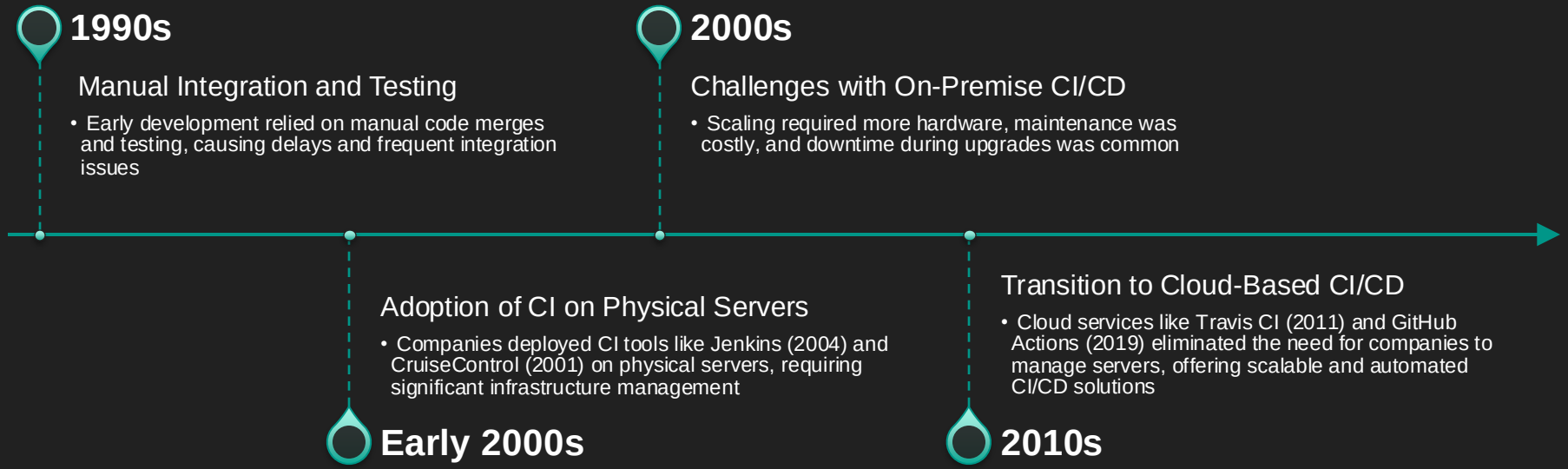
Version control

Faster release cycles

Code quality improvements

Enhanced team collaboration

History of CI/CD



GitHub Actions

- What is GitHub Actions?
  - Introduce Workflows
  - Events/Jobs/Runners
  - Integration with GitHub

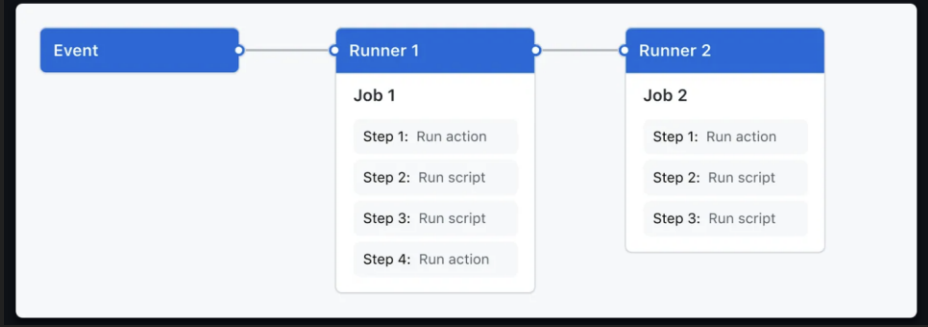
GitHub Actions

What is GitHub Actions?

GitHub's Native CI/CD Tool	Automates workflows directly within GitHub repositories
Workflow Automation	Uses **YAML** files to define automated processes like builds, tests, and deployments
Event-Driven	Triggers workflows based on GitHub events (e.g., push, pull_request, or schedule)
Highly Customizable	Supports custom actions, parallel jobs, and matrix builds for flexible automation
Seamless Integration	Built into GitHub, enabling easy integration with existing projects and third-party services

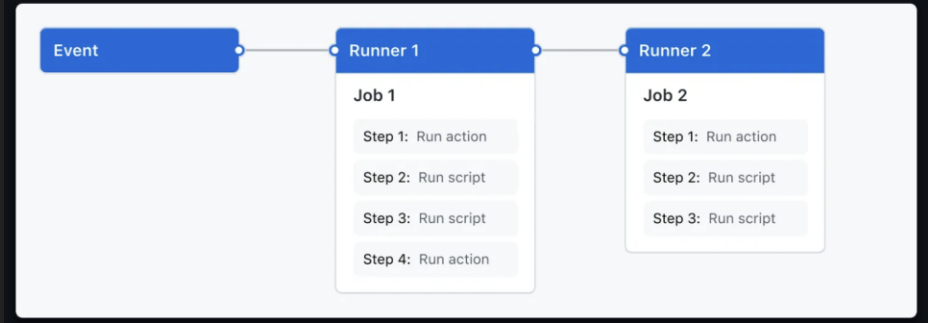
Workflows

- **Definition**: Automated processes defined in **YAML files** that run jobs in response to GitHub events (e.g., `push`, `pull_request`)
- **Location**: Workflows are stored in the `.github/workflows/` directory of your repository



Workflows

- **Components**
  - **Triggers**: Conditions that initiate a workflow in response to specific events, like a code push
  - **Jobs**: A collection of steps executed on virtual machines (runners), which can run in parallel or sequentially
  - **Steps**: Individual commands or actions that are part of a job



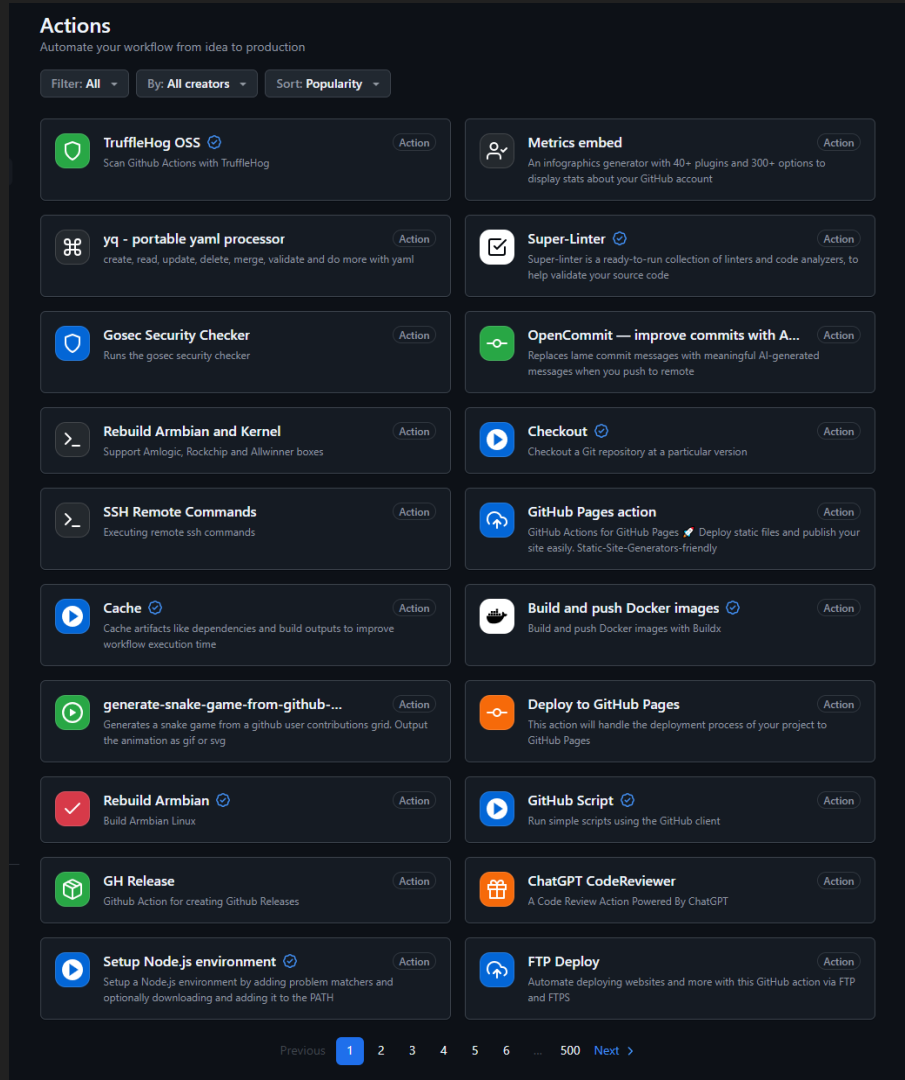
Jobs

- **Definition**
  - A set of steps that are executed sequentially within a workflow
- **Parallel Execution**
  - Multiple jobs can run simultaneously by default, improving workflow efficiency
- **Runs on Virtual Machines**
  - Each job runs in its own isolated environment (runner), such as Ubuntu, macOS, or Windows
- **Dependencies Between Jobs**
  - Jobs can be set to run sequentially or dependent on the success of other jobs

```yaml
jobs:
  my_first_job:
    name: My first job
  my_second_job:
    name: My second job
```

Actions

- **Definition**
  - Predefined, reusable tasks that perform specific functions within a job
- **Reusable Across Projects**
  - Actions can be reused in multiple workflows or repositories, saving time and effort
- **Types of Actions**
  - **Built-In Actions**: Provided by GitHub, like `actions/checkout` to clone a repository
  - **Custom Actions**: Created by developers to perform specific tasks
- **Marketplace**
  - GitHub Actions Marketplace offers thousands of ready-made actions for common tasks (e.g., building, testing, deploying)
- **Used in Steps**: Actions are invoked within steps to automate tasks



Jobs

```yaml
# Trigger the workflow on development branch

on:
    push:
        branches:
            - development

jobs:
    test:
        runs-on: ubuntu-latest
        steps:
            - uses: actions/checkout@v2
            - name: Set up Python
              uses: actions/setup-python@v2
              with:
                python-version: '3.9'
            - run: pip install -r requirements.txt
            - run: pip install pytest
            - name: Run Tests
              run: pytest

    build:
        runs-on: ubuntu-latest
        needs: test
        steps:
            - uses: actions/checkout@v2
            - name: Build Docker image
              run: docker build -t task-api .
```

Event/Trigger

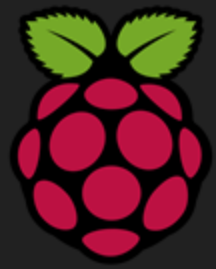
Jobs

Workflow Trivia

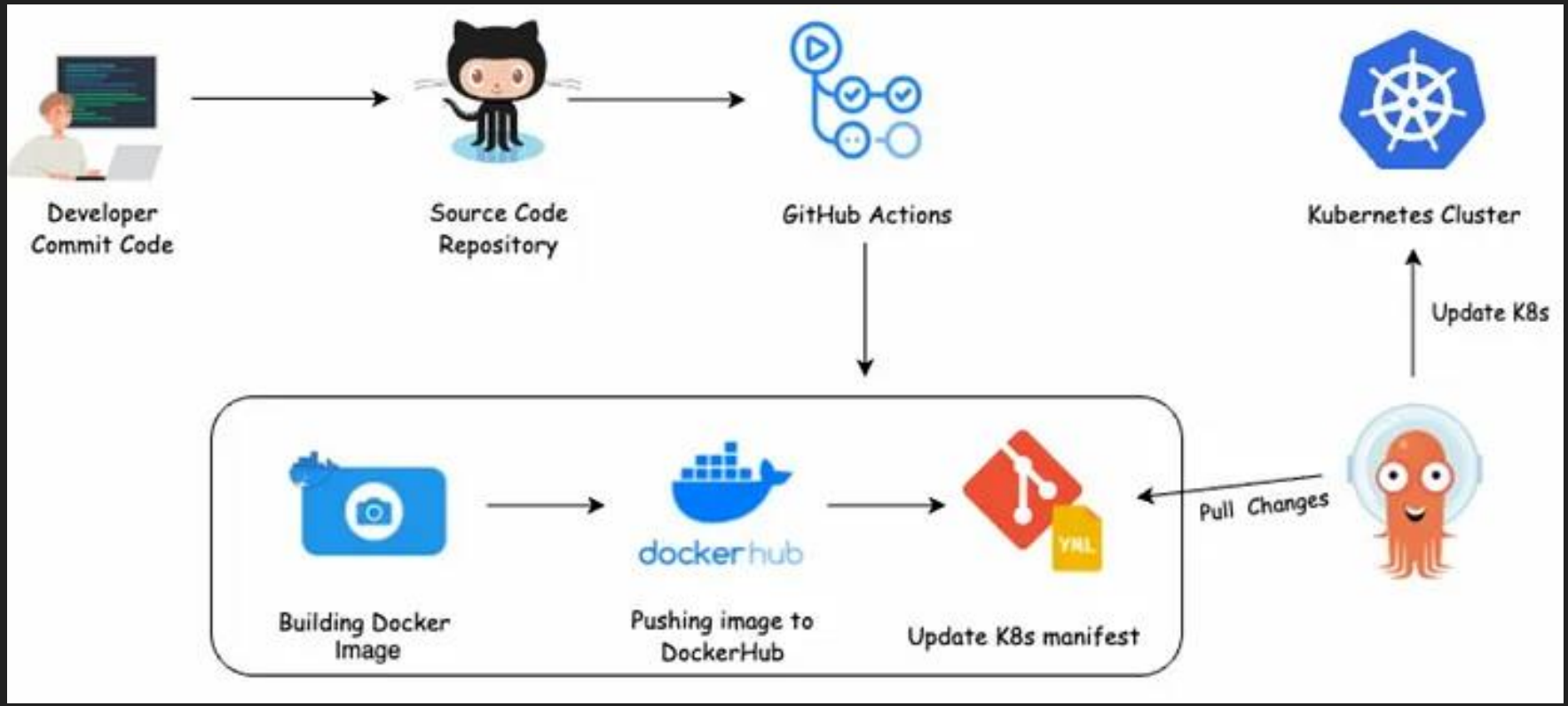
Building a CI/CD Pipeline with GitHub Actions

Touchless Kiosk Integration

- **GitHub Actions**:
  - Builds a new Docker image when code is pushed to the main branch
  - Pushes the image to a container registry
  - Updates the Kubernetes deployment config in the repo
- **Kubernetes (K3s or MicroK8s)**:
  - Runs containers on Raspberry Pis
  - Manages deployments
- **ArgoCD**:
  - Monitors the repo for config changes
  - Automatically updates Kubernetes with the new image version
  - Ensures seamless deployment across all Raspberry Pis



Touchless Kiosk Integration



Deliverable

Create a GitHub Actions Workflow and upload it to GitHub!

Questions? / Happy Halloween