Progress

CI/CD in Action

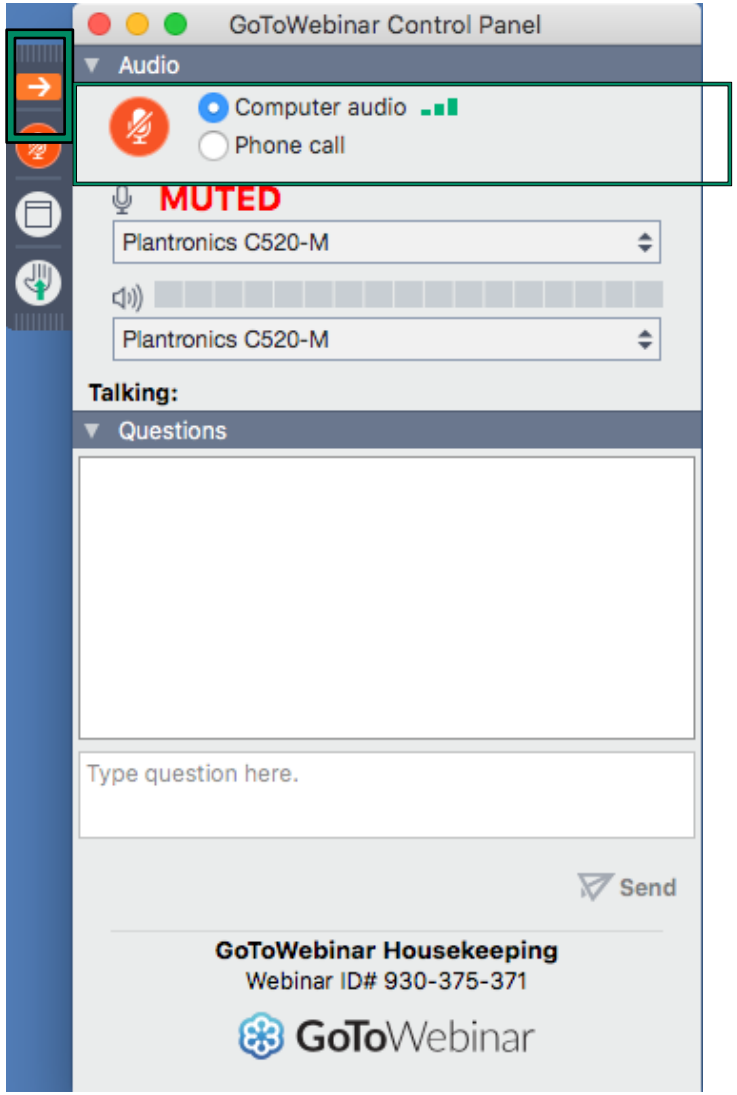
Chad Thomson

Progress

Jason Halbig

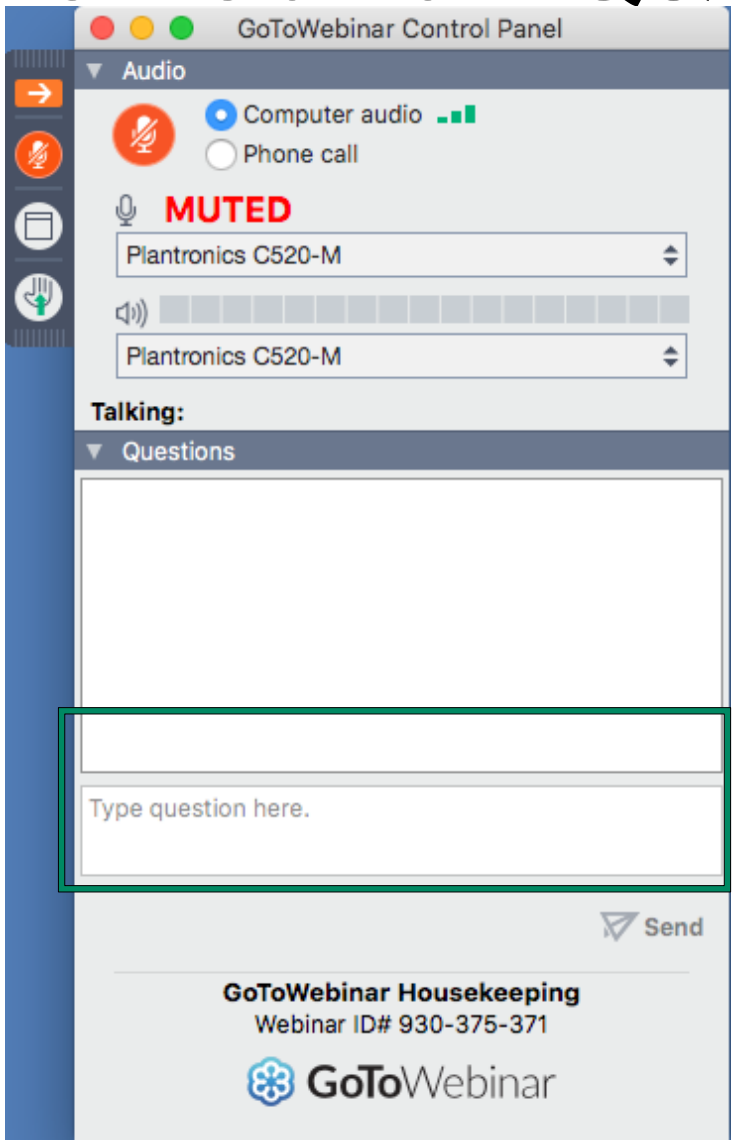
InGen Technologies

GoToWebinar: How to Participate



- Open and hide your control panel
- Join audio:
  - Choose “Computer Audio” to use VoIP
  - Choose “Phone Call” and dial using the information provided

GoToWebinar: Q&A



- Submit your questions & comments using the Questions panel at any time during the webinar

Today’s Speakers

Jason Halbig
Principal Consultant
InGen Technologies

Chad Thomson
Sr. Principal Consultant
Progress

Introduction

Observed Challenges

1. **Documentation**
    a. Permissions
    b. 3rd Party Dependencies
    c. OS Configs
    d. Integrations

2. **Developer Experience**
    a. Vi / PDSOE / VSCode
    b. Local Database or Schema Holders
    c. Are Containers or VM's an options

3. **Git fundamentals**
    a. Branching, merging, tagging

4. **IaC (Infrastructure as Code)**
    a. Configurations
    b. Environment procurement

CI/CD fundamentals

There are eight fundamental elements of CI/CD that help ensure maximum efficiency for your development lifecycle. They span development and deployment. Include these fundamentals in your pipeline to improve your DevOps workflow and software delivery:

A single source repository
Source code management (SCM) that houses all necessary files and scripts to create builds. The repository should contain everything needed for the build. This includes source code, database structure, libraries, properties files and version control. It should also contain test scripts and scripts to build applications.

Frequent check-ins to main branch
Integrating code in your trunk, mainline or master branch — i.e. trunk-based development — early and often. Avoid sub-branches and work with the main branch only. Use small segments of code and merge them into the branch as frequently as possible. Don't merge more than one change at a time.

Automated builds
Scripts should include everything you need to build from a single command. This includes web server files, database scripts and application software. The CI processes should automatically package and compile the code into a usable application.

Self-testing builds
Testing scripts should ensure that the failure of a test results in a failed build. Use static pre-build testing scripts to check code for integrity, quality and security compliance. Only allow code that passes static tests into the build.

Frequent iterations
Multiple commits to the repository results in fewer places for conflicts to hide. Make small, frequent iterations rather than major changes. By doing this, it's possible to roll changes back easily if there's a problem or conflict.

Stable testing environments
Code should be tested in a cloned version of the production environment. You can't test new code in the live production version. Create a cloned environment that's as close as possible to the real environment. Use rigorous testing scripts to detect and identify bugs that slipped through the initial pre-build testing process.

Maximum visibility
Every developer should be able to access the latest executables and see any changes made to the repository. Information in the repository should be visible to all. Use version control to manage handoffs, so developers know which is the latest version. Maximum visibility means everyone can monitor progress and identify potential concerns.

Predictable deployments anytime
Deployments are so routine and low-risk that the team's comfortable doing them anytime. CI/CD testing and verification processes should be rigorous and reliable. This gives the team confidence to deploy updates at any time. Frequent deployments incorporating limited changes also pose lower risks and can be easily rolled back.

Case Studies

	Client S	Client E	Client B
Team Size	**13** Developers **1** DBA **2** Ops	**12** Developers **1** DBA **2** Ops	**8** Developers
Configuration	10.1c to 11.7 Character App Webspeed	11.6 to 12.2 Windows Application	10.2 to 11.7 Character App New PAS based services
Infrastructure as Code	Ansible		
Source Control	Bitbucket	Github	Bitbucket
IDE	vi / VScode	PDSOE	PDSOE / VSCode
Build	Jenkins + ANT + PCT		
LoadTesting	Jmeter		Gatling / Jmeter
Server OS	Linux	Windows	Linux / Windows

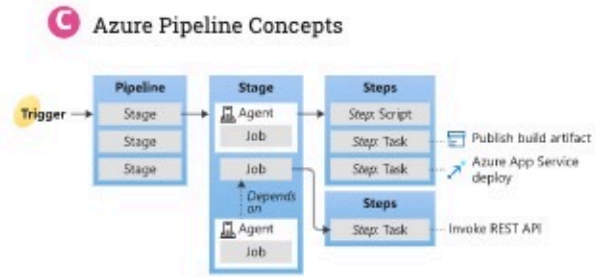
Team Modes

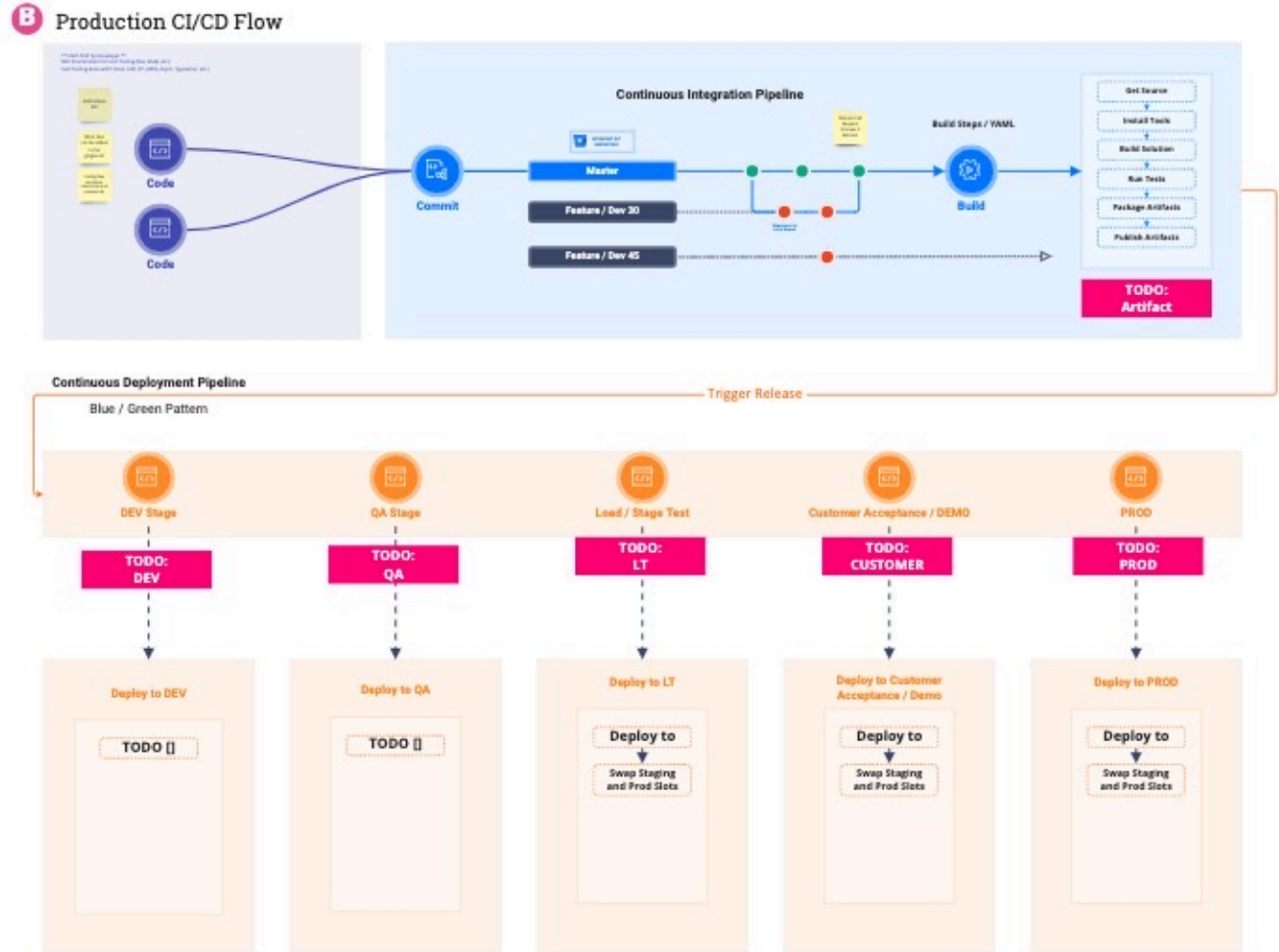
3 Typical Team Modes

1. **You build it, <u>You</u> run it**
   a. Came from Amazon CTO Werner Vogels 2006
   b. All developers can do all DevOps tasks on their own.

2. **You build it, <u>They</u> run it**
   a. Dedicated Ops engineer/team that manage DevOps tasks. Developers are strictly separated and don't care much about configs and infrastructure.

3. **Shadow operations**
   a. Senior devs doing operation work for less experienced devs
   b. Some developers can do some DevOps tasks on their own and are constantly busy helping less experience developers.





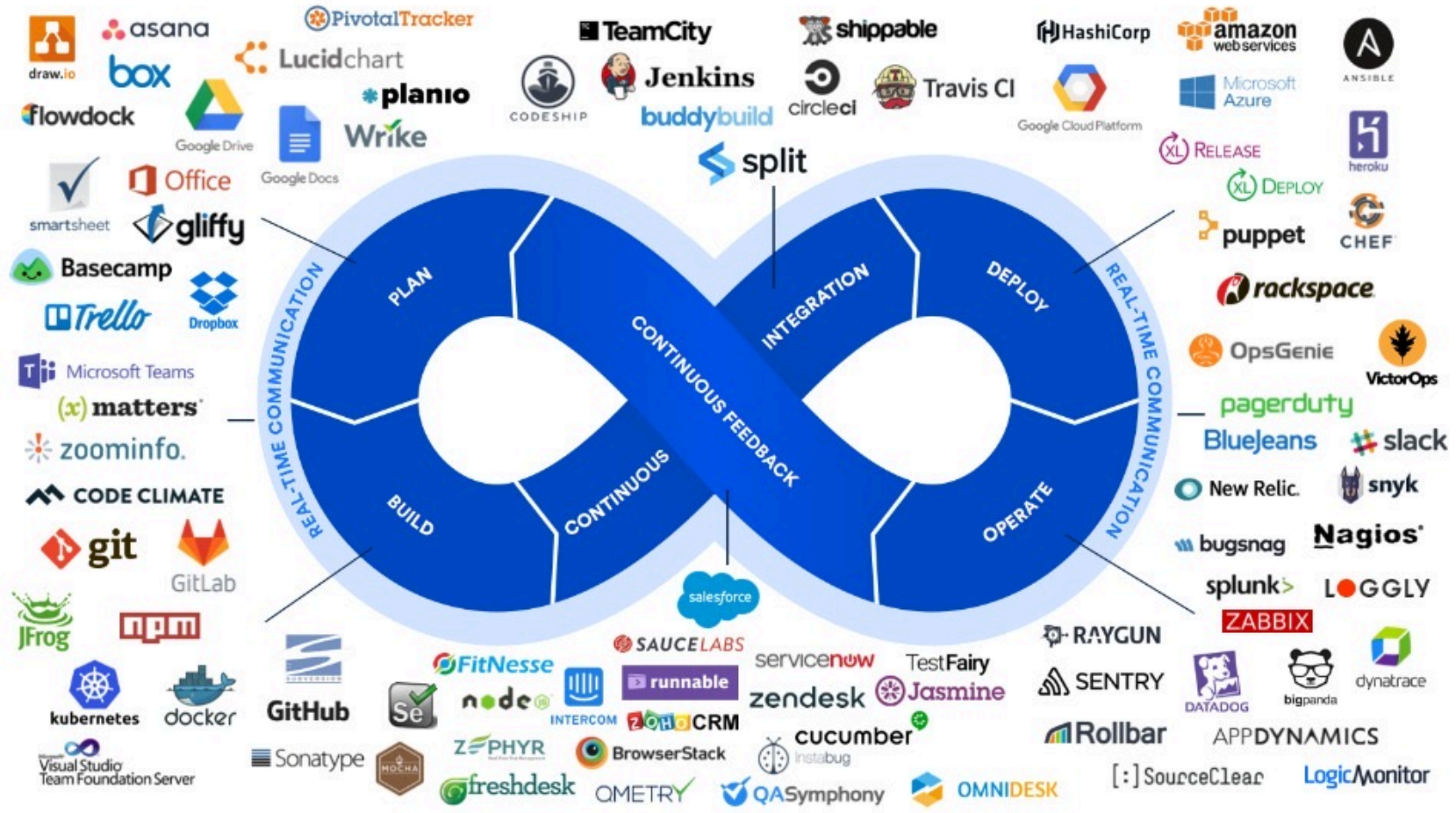
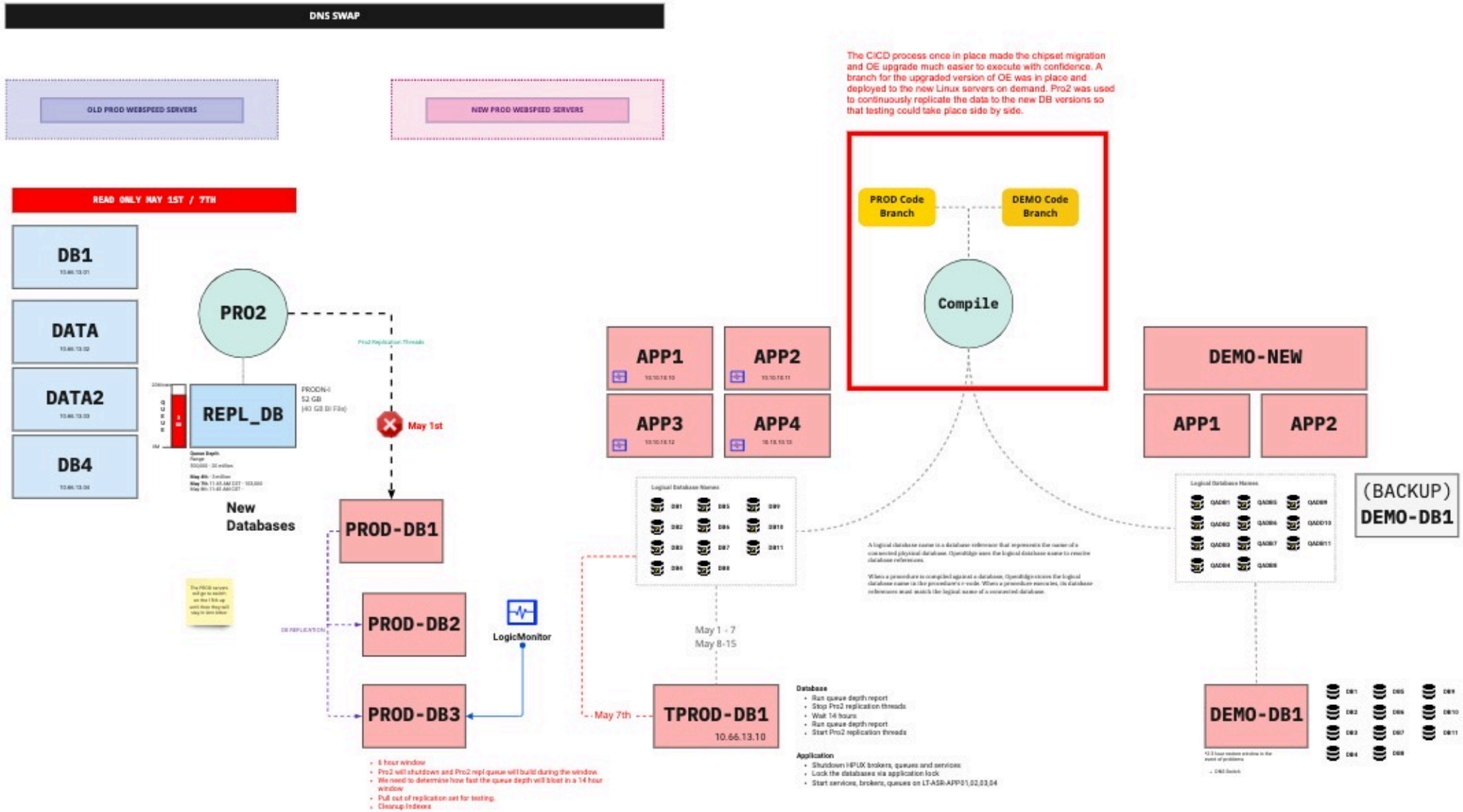


Fig. 1: Atlassian Marketplace..

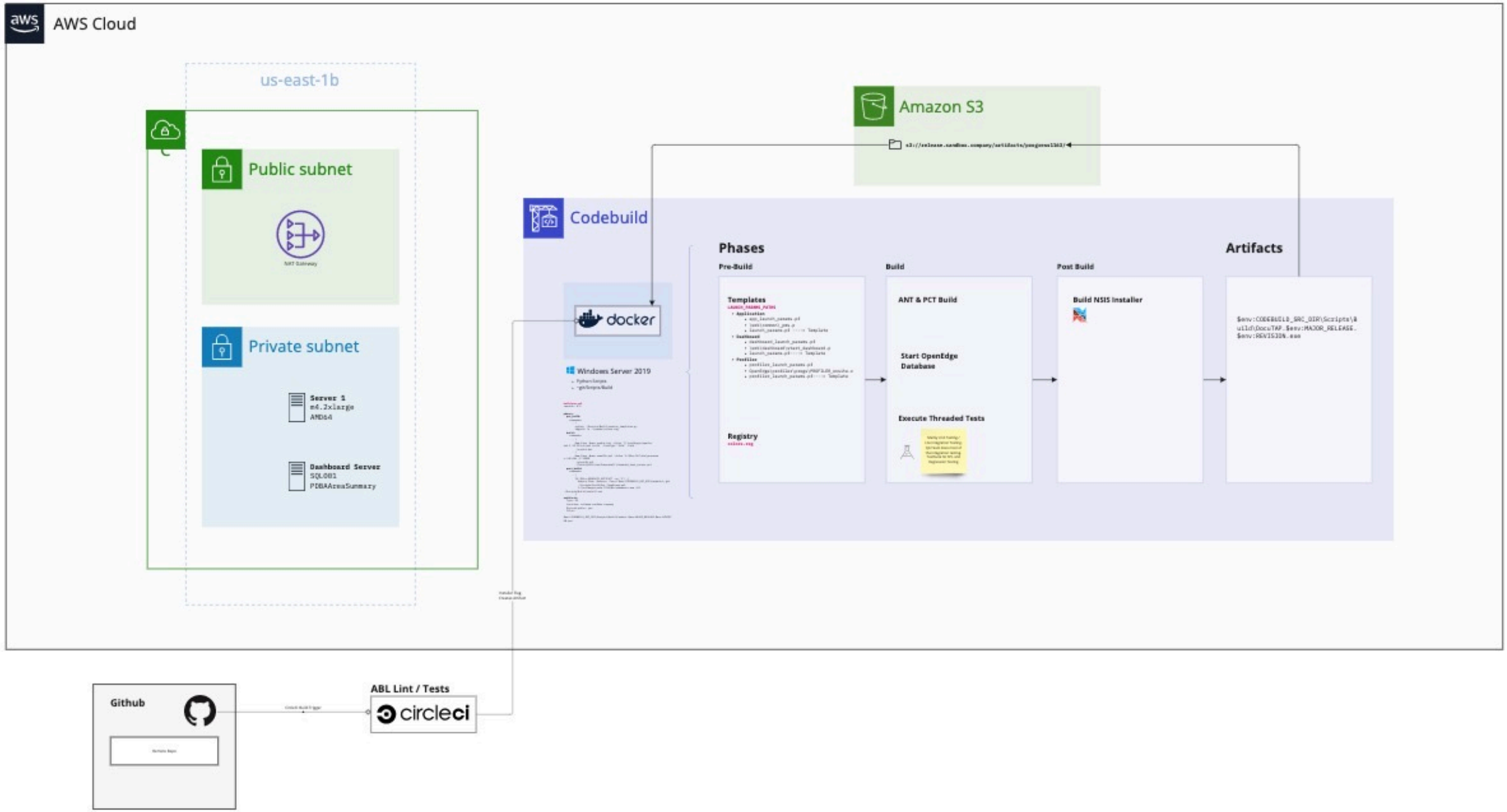
Client S

	Client S	Client E	Client B
Team Size	**13** Developers **1** DBA **2** Ops	**12** Developers **1-2** DBA **2** Ops	**8** Developers
Configuration	10.1c to 11.7 Character App Webspeed	11.6 to 12.2 Windows Application	10.2 to 11.7 Character App Added new PAS based services
Infrastructure as Code	Ansible		
Source Control	Bitbucket	Github	Bitbucket
IDE	vi / VScode	PDSOE	PDSOE / VSCode
Build		Jenkins + ANT + PCT	
LoadTesting	Jmeter		Gatling / Jmeter
Server OS	Linux	Windows	Linux / Windows



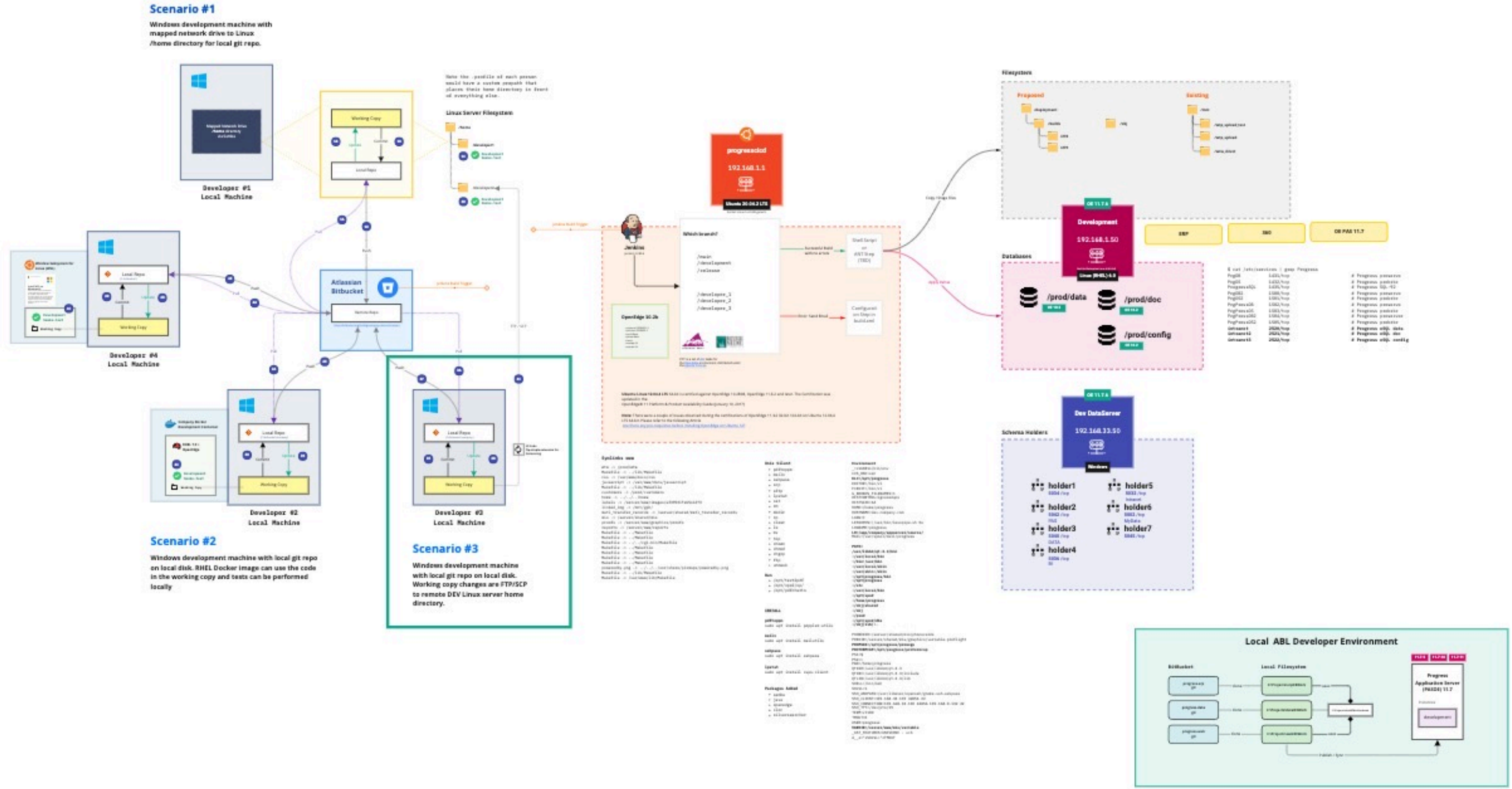
Client E

	Client S	Client E	Client B
Team Size	**13** Developers **1** DBA **2** Ops	**12** Developers **1-2** DBA **2** Ops	**8** Developers
Configuration	10.1c to 11.7 Character App Webspeed	11.6 to 12.2 Windows Application	10.2 to 11.7 Character App Added new PAS based services
Infrastructure as Code	Ansible		
Source Control	Bitbucket	Github	Bitbucket
IDE	vi / VScode	PDSOE	PDSOE / VSCode
Build		Jenkins + ANT + PCT	
LoadTesting	Jmeter		Gatling / Jmeter
Server OS	Linux	Windows	Linux / Windows

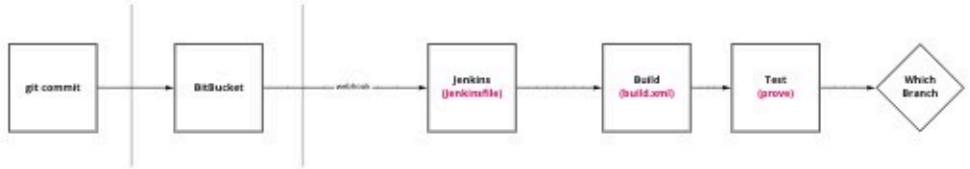


Client B

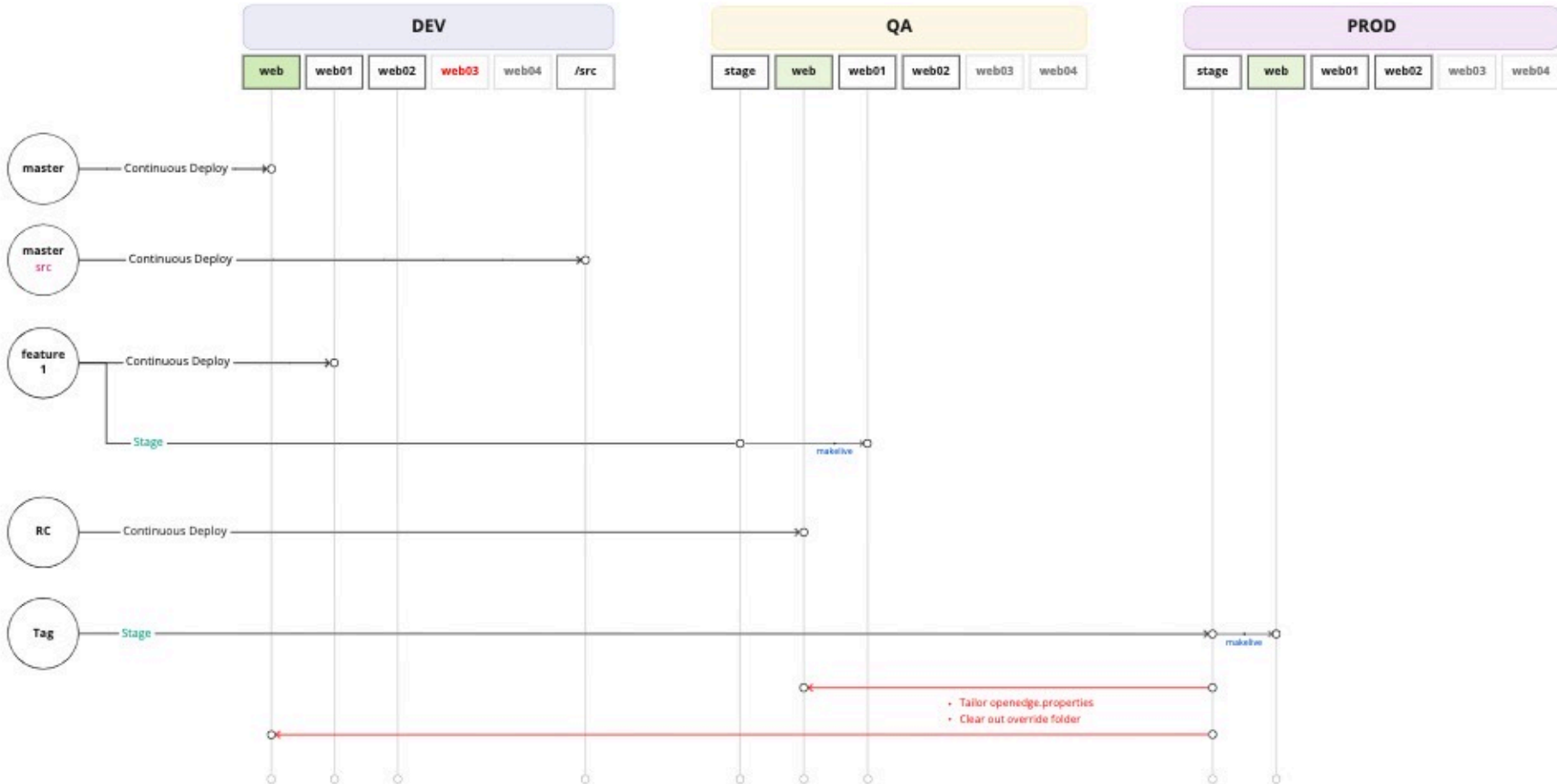
	Client S	Client E	Client B
Team Size	**13** Developers **1** DBA **2** Ops	**12** Developers **2** Ops **1** DBA	**8** Developers
Configuration	10.1c to 11.7 Character App Webspeed	11.6 to 12.2 Windows Application	10.2 to 11.7 Character App Added new PAS based services
Infrastructure as Code	Ansible		
Source Control	Bitbucket	Github	Bitbucket
IDE	vi / VScode	PDSOE	PDSOE / VSCode
Build		Jenkins + ANT + PCT	
LoadTesting	LoadRunner /Jmeter		Gatling / Jmeter
Server OS	Linux	Windows	Linux / Windows



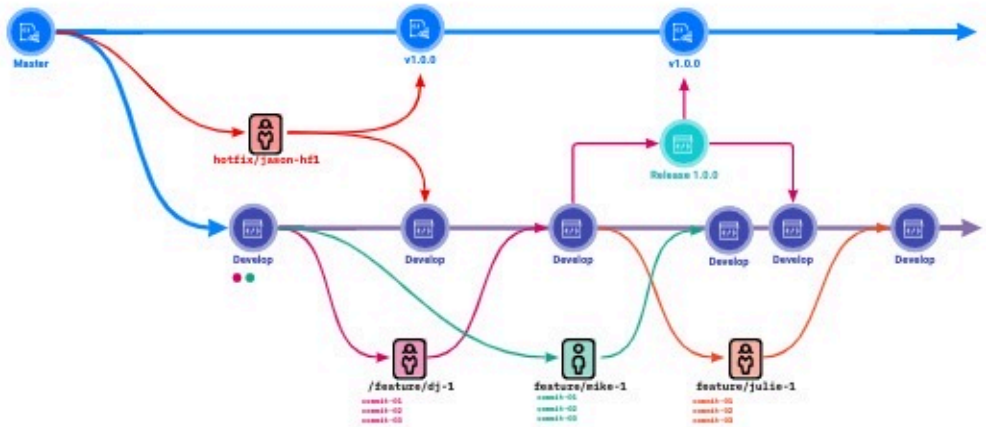
CICD

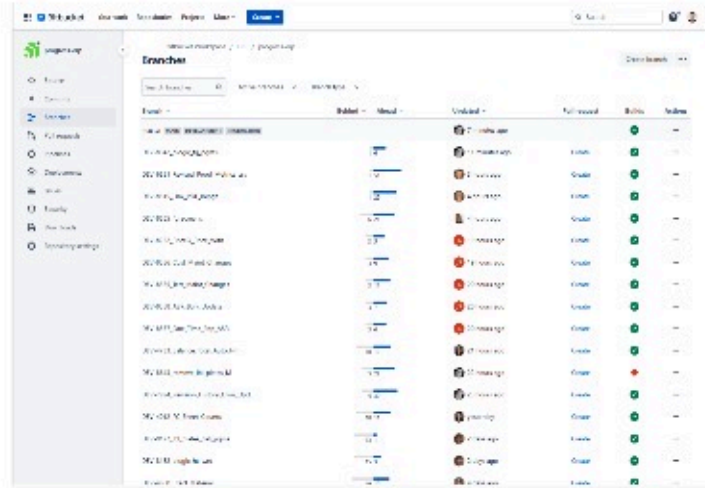


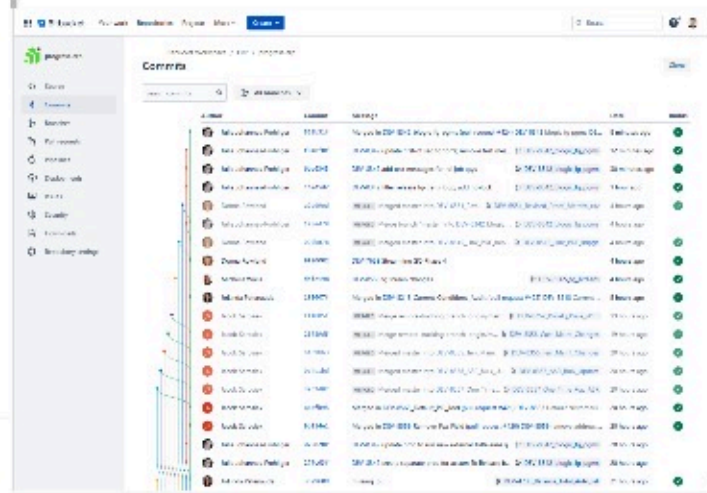
PAS Deployment / Staging



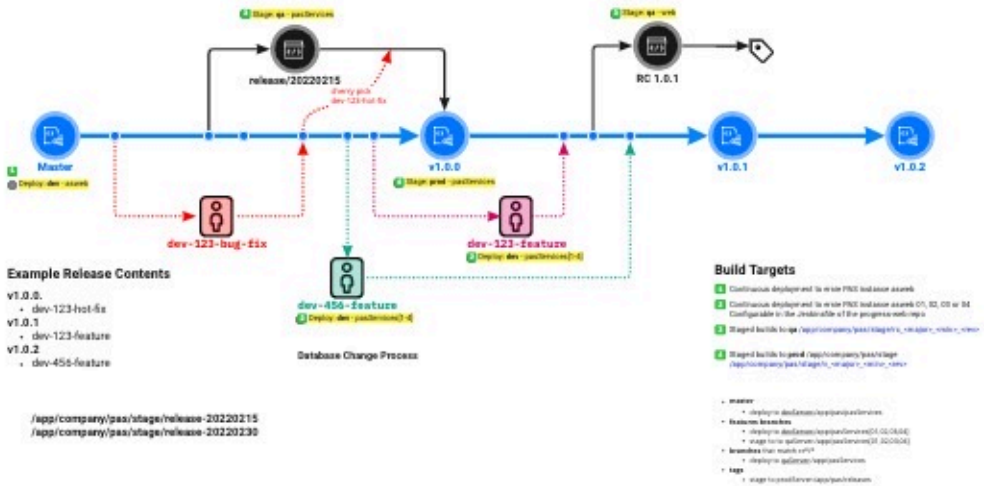
Gitflow

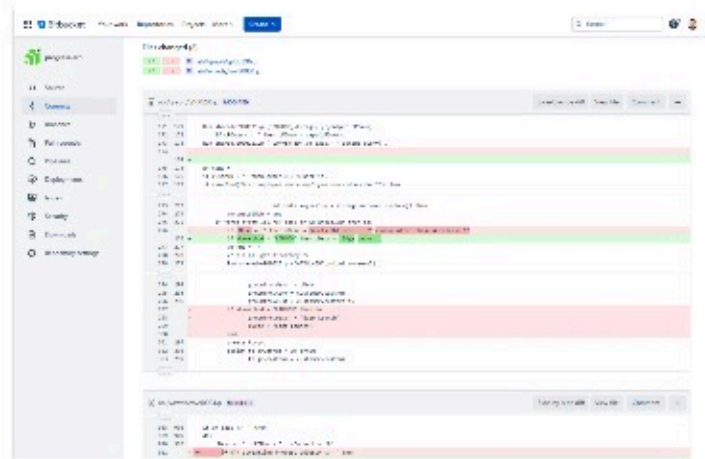


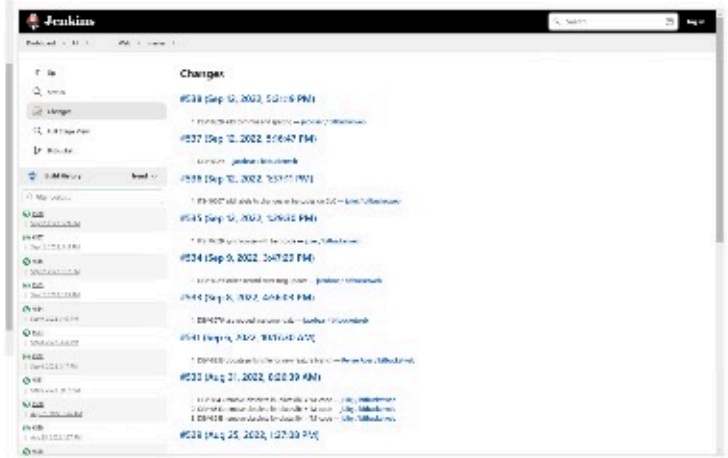


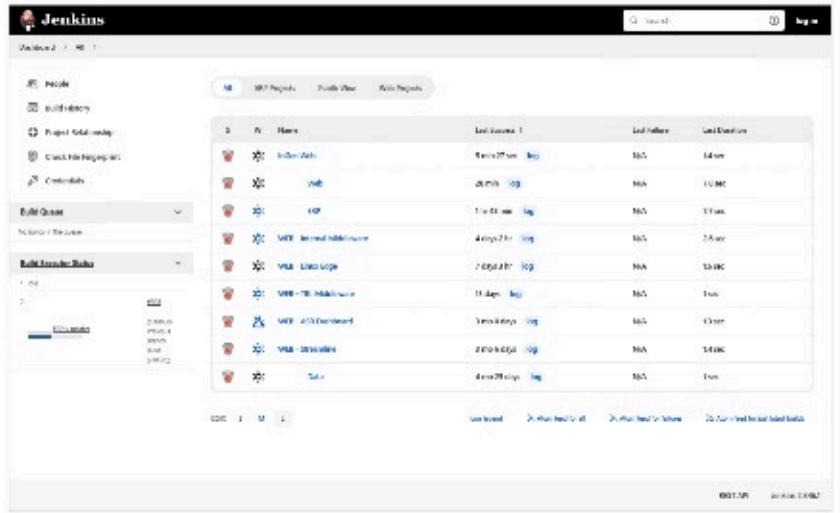


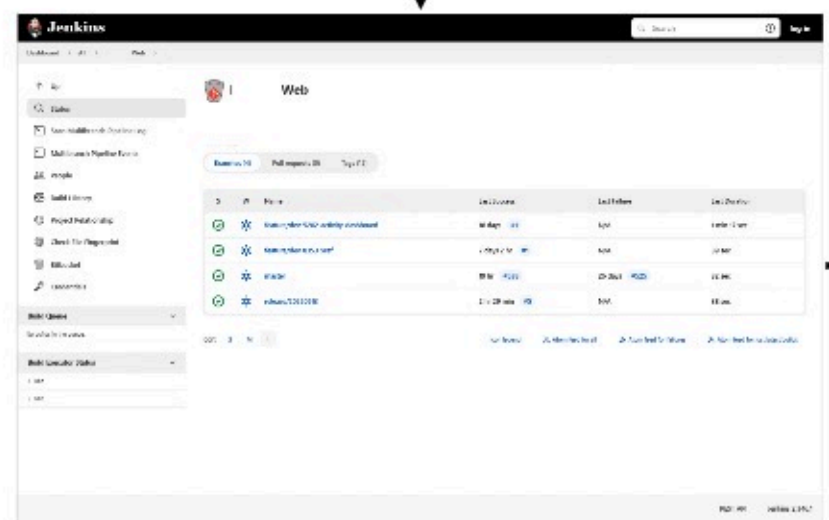
Bespoke Flow

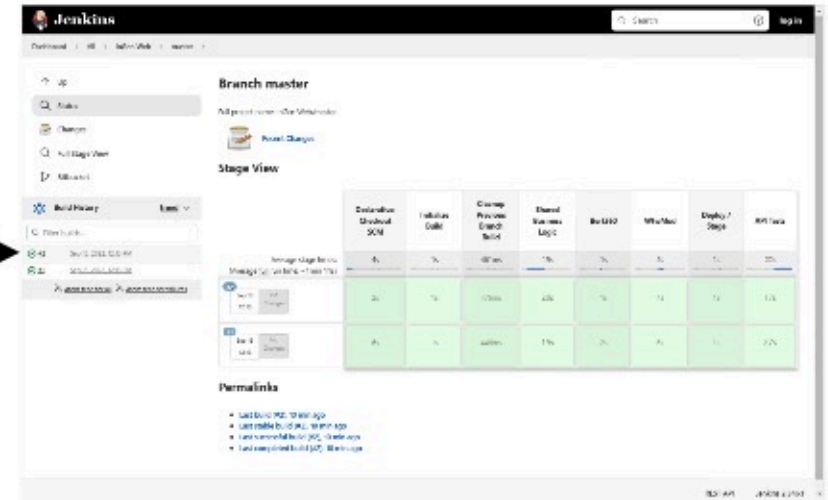


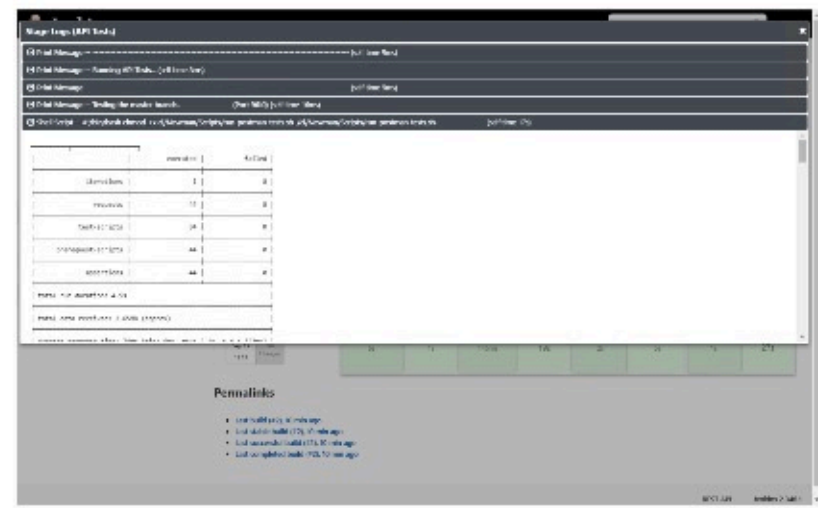













Progress®