CHATBOTS

FOR FUN AND PROFIT

```
function whoami() {
  return {
    name: "Vincent Roy",
    title: "Senior Software Engineer",
    company: "Dell EMC",
    tags: ["lazy", "code", "automation"],
    site: "https://vroy.ca/"
    github: "https://github.com/vroy",
    twitter: "@vroy"
  };
}
```

What is a Chatbot?

*A chatbot is a computer program which conducts a conversation via auditory or textual methods. Such programs are **often** designed to convincingly simulate how a human would behave as a conversational partner, thereby passing the Turing test.*

-- https://en.wikipedia.org/wiki/Chatbot

The Basics of a Chatbot

- A program - not unlike many others
- Hear and respond to messages from chat rooms
- Send rich & timely notifications

- More importantly it does work for you

STOP! DEMO TIME!

```
Botkit = require("botkit")
process = require("process")

controller = Botkit.slackbot({
  json_file_store: "/tmp/profit"
})

bot = controller.spawn({
  token: process.env.SLACK_TOKEN
})

bot.startRTM (err, bot, payload) ->
  if err
    throw new Error("Could not connect to Slack")
```

```
Botkit = require("botkit")
process = require("process")

controller = Botkit.slackbot({
  json_file_store: "/tmp/profit"
})

bot = controller.spawn({
  token: process.env.SLACK_TOKEN
})

bot.startRTM (err, bot, payload) ->
  if err
    throw new Error("Could not connect to Slack")
```

```
Botkit = require("botkit")
process = require("process")

controller = Botkit.slackbot({
  json_file_store: "/tmp/profit"
})

bot = controller.spawn({
  token: process.env.SLACK_TOKEN
})

bot.startRTM (err, bot, payload) ->
  if err
    throw new Error("Could not connect to Slack")
```

```
Botkit = require("botkit")
process = require("process")

controller = Botkit.slackbot({
  json_file_store: "/tmp/profit"
})

bot = controller.spawn({
  token: process.env.SLACK_TOKEN
})

bot.startRTM (err, bot, payload) ->
  if err
    throw new Error("Could not connect to Slack")
```

```
controller.hears [/^hello/i, /^hi/], ["direct_message", "mention"],
  (bot, message) ->
    greeting = _.capitalize(message.match[0])

    # Use the Slack API to get information about this user.
    bot.api.users.info { user: message.user }, (err, response) ->
      if err
        return bot.reply(message, "#{greeting} yourself! :wave:")

      bot.reply(message, "#{greeting} #{response.user.profile.first_name}! :wave:")
```

```
controller.hears [/^hello/i, /^hi/], ["direct_message", "mention"],
  (bot, message) ->
    greeting = _.capitalize(message.match[0])

    # Use the Slack API to get information about this user.
    bot.api.users.info { user: message.user }, (err, response) ->
      if err
        return bot.reply(message, "#{greeting} yourself! :wave:")

      bot.reply(message, "#{greeting} #{response.user.profile.first_name}! :wave:")
```

```
controller.hears [/^hello/i, /^hi/], ["direct_message", "mention"],
  (bot, message) ->
    greeting = _.capitalize(message.match[0])

    # Use the Slack API to get information about this user.
    bot.api.users.info { user: message.user }, (err, response) ->
      if err
        return bot.reply(message, "#{greeting} yourself! :wave:")

      bot.reply(message, "#{greeting} #{response.user.profile.first_name}! :wave:")
```

```
controller.hears [/^hello/i, /^hi/], ["direct_message", "mention"],
  (bot, message) ->
    greeting = _.capitalize(message.match[0])

    # Use the Slack API to get information about this user.
    bot.api.users.info { user: message.user }, (err, response) ->
      if err
        return bot.reply(message, "#{greeting} yourself! :wave:")

      bot.reply(message, "#{greeting} #{response.user.profile.first_name}! :wave:")
```

```
controller.hears [/^hello/i, /^hi/], ["direct_message", "mention"],
  (bot, message) ->
    greeting = _.capitalize(message.match[0])

    # Use the Slack API to get information about this user.
    bot.api.users.info { user: message.user }, (err, response) ->
      if err
        return bot.reply(message, "#{greeting} yourself! :wave:")

      bot.reply(message, "#{greeting} #{response.user.profile.first_name}! :wave:")
```

```
controller.hears [/^hello/i, /^hi/], ["direct_message", "mention"],
  (bot, message) ->
    greeting = _.capitalize(message.match[0])

    # Use the Slack API to get information about this user.
    bot.api.users.info { user: message.user }, (err, response) ->
      if err
        return bot.reply(message, "#{greeting} yourself! :wave:")

      bot.reply(message, "#{greeting} #{response.user.profile.first_name}! :wave:")
```

```
controller.hears /^make me a (sandwich|soup)/, ["direct_message"],
  (bot, message) ->
    food = message.match[1]

    handler = buildFoodMakerConversation(food)

    if food == "sandwich"
      bot.startConversation(message, handler)
    else
      bot.startConversationInThread(message, handler)
```

```
controller.hears /^make me a (sandwich|soup)/, ["direct_message"],
  (bot, message) ->
    food = message.match[1]

    handler = buildFoodMakerConversation(food)

    if food == "sandwich"
      bot.startConversation(message, handler)
    else
      bot.startConversationInThread(message, handler)
```

```
controller.hears /^make me a (sandwich|soup)/, ["direct_message"],
  (bot, message) ->
    food = message.match[1]

    handler = buildFoodMakerConversation(food)

    if food == "sandwich"
      bot.startConversation(message, handler)
    else
      bot.startConversationInThread(message, handler)
```

```
controller.hears /^make me a (sandwich|soup)/, ["direct_message"],
  (bot, message) ->
    food = message.match[1]

    handler = buildFoodMakerConversation(food)

    if food == "sandwich"
      bot.startConversation(message, handler)
    else
      bot.startConversationInThread(message, handler)
```

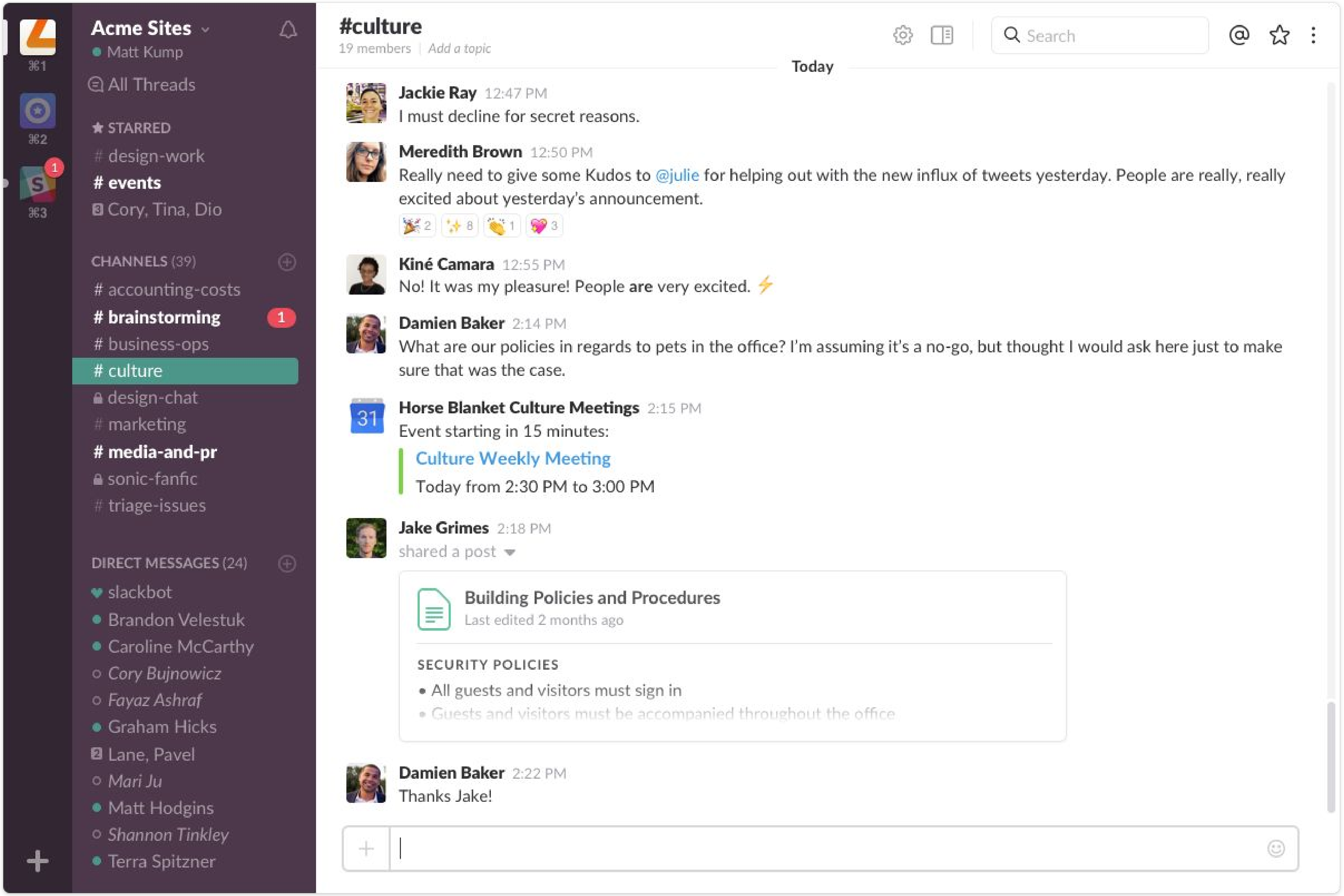
```
controller.hears /^make me a (sandwich|soup)/, ["direct_message"],
  (bot, message) ->
    food = message.match[1]

    handler = buildFoodMakerConversation(food)

    if food == "sandwich"
      bot.startConversation(message, handler)
    else
      bot.startConversationInThread(message, handler)
```

```
buildFoodMakerConversation = (food) ->
  return (err, conversation) ->
    conversation.addQuestion "What? Make your #{food} yourself.", [
      {
        pattern: "sudo make me a (sandwich|soup)",
        callback: (response, convo) ->
          convo.say("Okay. :#{food}:")
          convo.next()
      },
      {
        default: true,
        callback: (response, convo) ->
          convo.next("stop")
      }
    ]
```

```coffeescript
buildFoodMakerConversation = (food) ->
  return (err, conversation) ->
    conversation.addQuestion "What? Make your #{food} yourself.", [
      {
        pattern: "sudo make me a (sandwich|soup)",
        callback: (response, convo) ->
          convo.say("Okay. :#{food}:")
          convo.next()
      },
      {
        default: true,
        callback: (response, convo) ->
          convo.next("stop")
      }
    ]
```

```
buildFoodMakerConversation = (food) ->
  return (err, conversation) ->
    conversation.addQuestion "What? Make your #{food} yourself.", [
      {
        pattern: "sudo make me a (sandwich|soup)",
        callback: (response, convo) ->
          convo.say("Okay. :#{food}:")
          convo.next()
      },
      {
        default: true,
        callback: (response, convo) ->
          convo.next("stop")
      }
    ]
```

VXRACK
DELL EMC



Jackie Ray 12:47 PM
I must decline for secret reasons.

Meredith Brown 12:50 PM
Really need to give some Kudos to @julie for helping out with the new influx of tweets yesterday. People are really, really excited about yesterday's announcement.
🎉 2 ✨ 8 👏 1 💖 3

Kiné Camara 12:55 PM
No! It was my pleasure! People **are** very excited. ⚡

Damien Baker 2:14 PM
What are our policies in regards to pets in the office? I'm assuming it's a no-go, but thought I would ask here just to make sure that was the case.

Horse Blanket Culture Meetings 2:15 PM
Event starting in 15 minutes:
Culture Weekly Meeting
Today from 2:30 PM to 3:00 PM

Jake Grimes 2:18 PM
shared a post
Building Policies and Procedures
Last edited 2 months ago
SECURITY POLICIES
- All guests and visitors must sign in
- Guests and visitors must be accompanied throughout the office

Damien Baker 2:22 PM
Thanks Jake!

```
reserve <lab name>:<node>|all:<expected duration in hours>|?>:[jira number]
```

```
release <lab name>:<node>|all
```

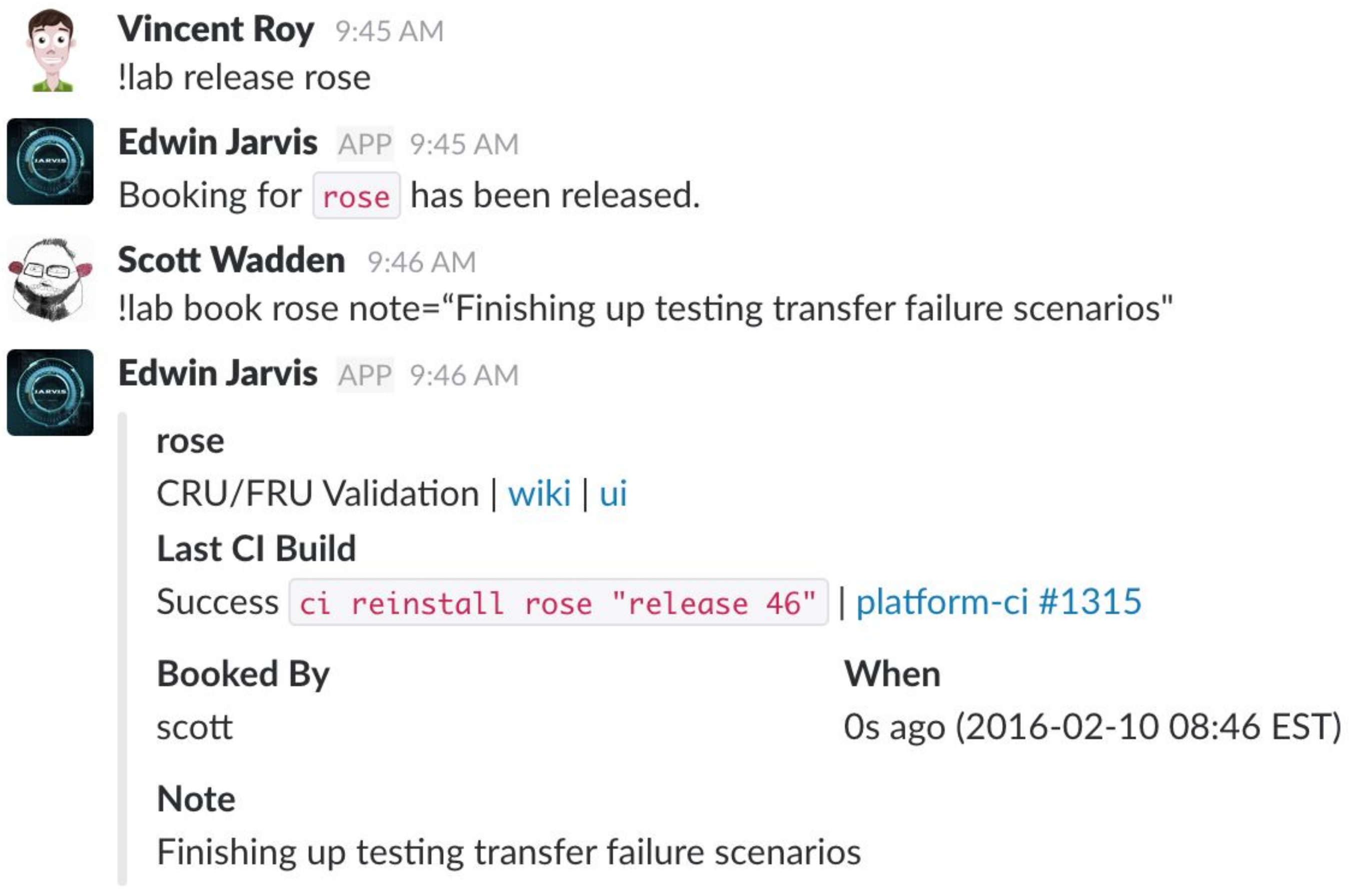
```
reserve <lab name>:<node>|all:<expected duration in hours>|?>:[jira number]
reserve rose:all:?:CP-123
```

```
release <lab name>:<node>|all
release rose:all
```

```
reserve <lab name>:<node>|all:<expected duration in hours>|?>:[jira number]
reserve rose:sandy,provo:3:NPS-124
```

```
release <lab name>:<node>|all
release rose:sandy,provo
```

J.A.R.V.I.S.



Vincent Roy 9:45 AM
!lab release rose

Edwin Jarvis APP 9:45 AM
Booking for `rose` has been released.

Scott Wadden 9:46 AM
!lab book rose note=“Finishing up testing transfer failure scenarios"

Edwin Jarvis APP 9:46 AM

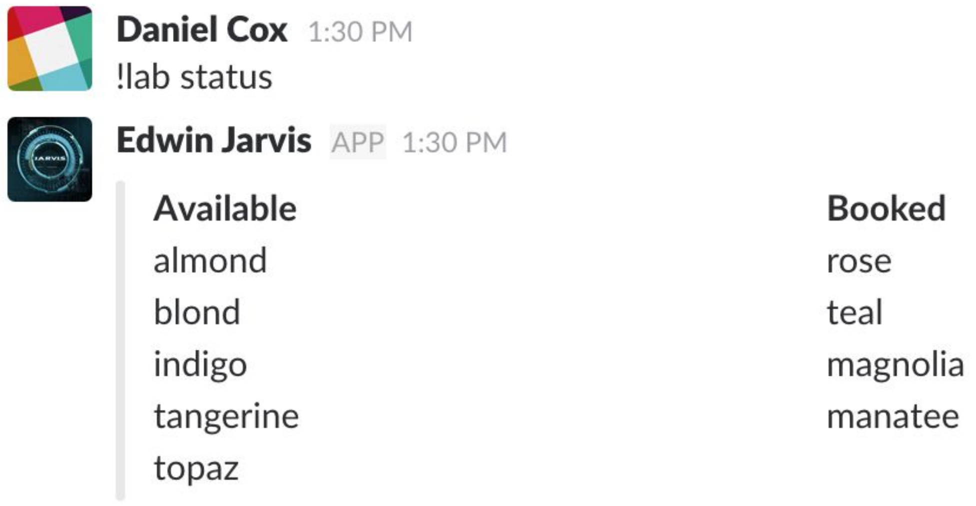
rose
CRU/FRU Validation | wiki | ui

Last CI Build
Success `ci reinstall rose "release 46"` | platform-ci #1315

Booked By
scott

When
0s ago (2016-02-10 08:46 EST)

Note
Finishing up testing transfer failure scenarios

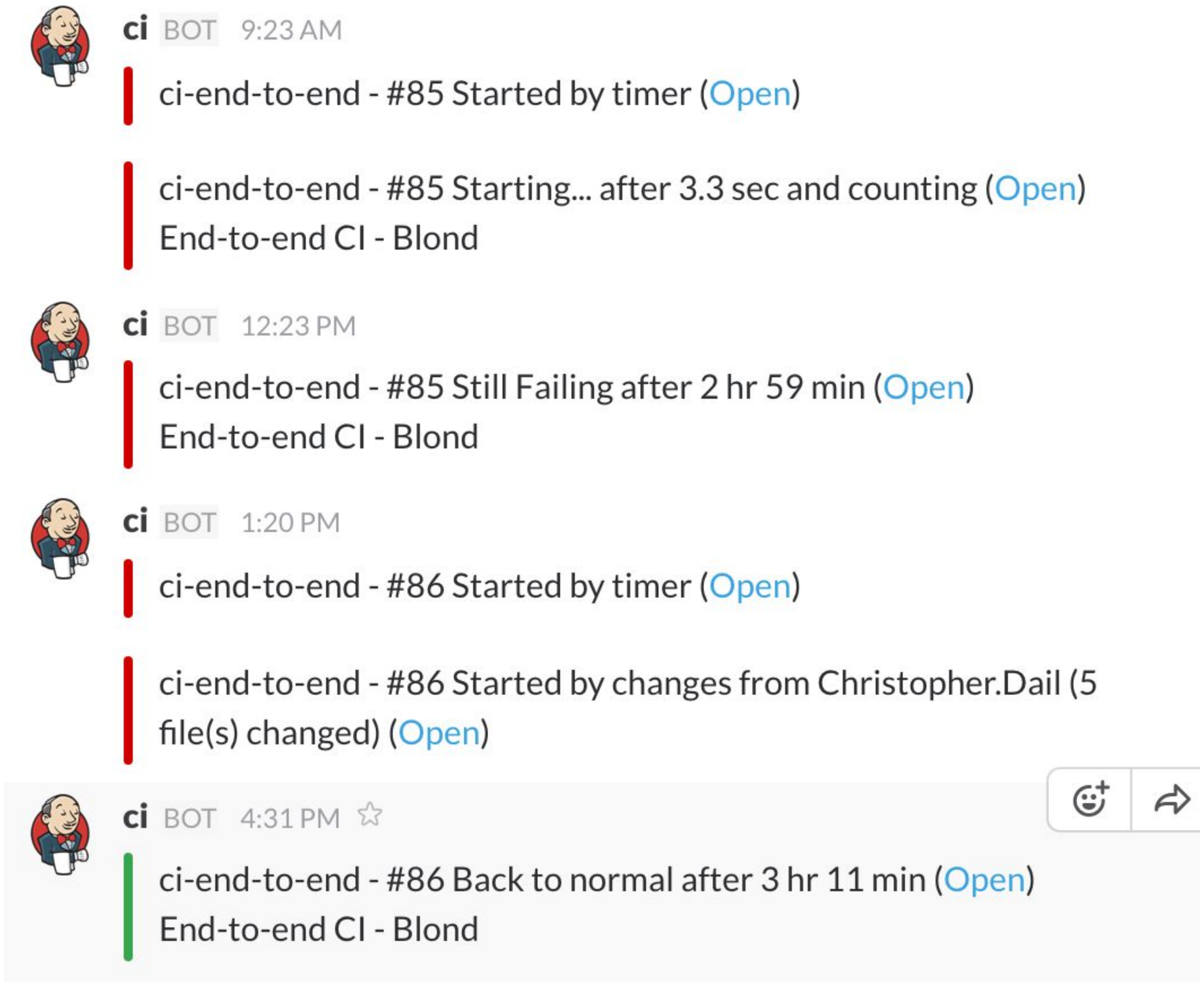


Daniel Cox 1:30 PM
!lab status

Edwin Jarvis APP 1:30 PM

Available	Booked
almond	rose
blond	teal
indigo	magnolia
tangerine	manatee
topaz	

SYNCHRONOUS



ci BOT 9:23 AM

ci-end-to-end - #85 Started by timer (Open)

ci-end-to-end - #85 Starting... after 3.3 sec and counting (Open)
End-to-end CI - Blond

ci BOT 12:23 PM

ci-end-to-end - #85 Still Failing after 2 hr 59 min (Open)
End-to-end CI - Blond

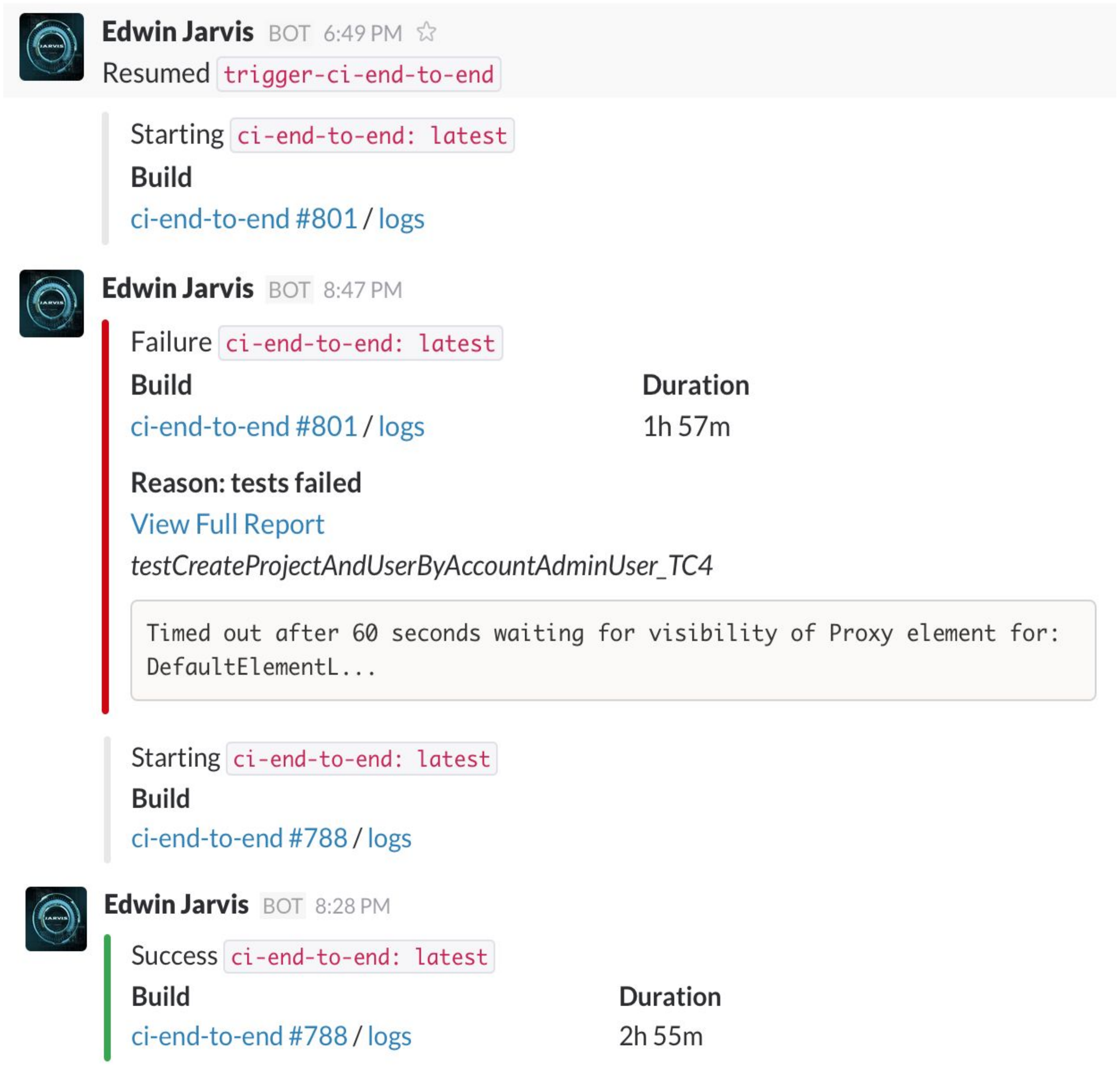
ci BOT 1:20 PM

ci-end-to-end - #86 Started by timer (Open)

ci-end-to-end - #86 Started by changes from Christopher.Dail (5 file(s) changed) (Open)

ci BOT 4:31 PM

ci-end-to-end - #86 Back to normal after 3 hr 11 min (Open)
End-to-end CI - Blond



Edwin Jarvis BOT 6:49 PM

Resumed `trigger-ci-end-to-end`

Starting `ci-end-to-end: latest`

Build

ci-end-to-end #801 / logs

Edwin Jarvis BOT 8:47 PM

Failure `ci-end-to-end: latest`

Build	**Duration**
ci-end-to-end #801 / logs	1h 57m

Reason: tests failed

View Full Report

testCreateProjectAndUserByAccountAdminUser_TC4

```
Timed out after 60 seconds waiting for visibility of Proxy element for:
DefaultElementL...
```

Starting `ci-end-to-end: latest`

Build

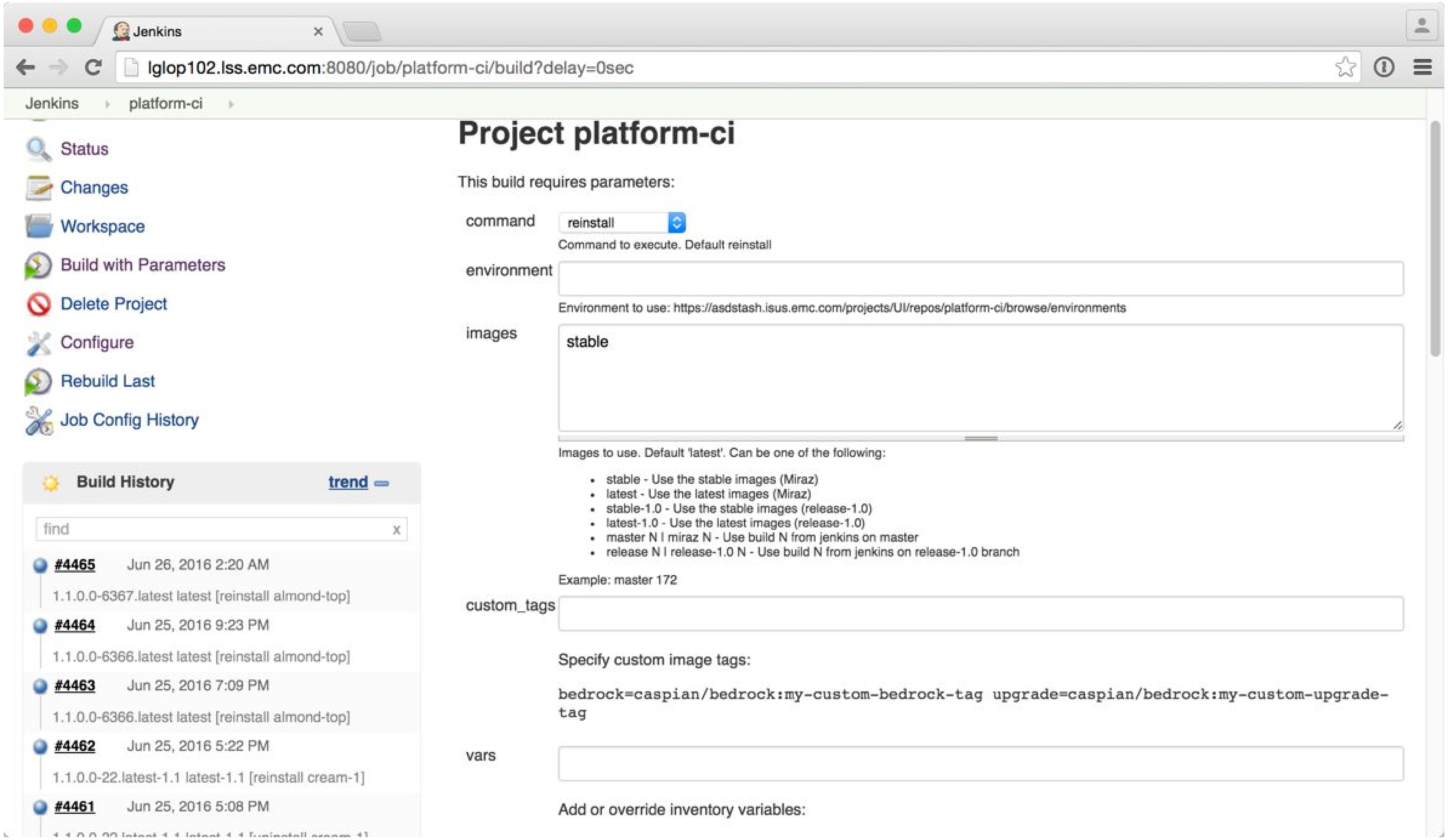
ci-end-to-end #788 / logs

Edwin Jarvis BOT 8:28 PM

Success `ci-end-to-end: latest`

Build	**Duration**
ci-end-to-end #788 / logs	2h 55m

VISIBILITY



Jenkins

lglop102.lss.emc.com:8080/job/platform-ci/build?delay=0sec

Jenkins › platform-ci ›

- Status
- Changes
- Workspace
- Build with Parameters
- Delete Project
- Configure
- Rebuild Last
- Job Config History

Build History trend

#4465	Jun 26, 2016 2:20 AM	1.1.0.0-6367.latest latest [reinstall almond-top]
#4464	Jun 25, 2016 9:23 PM	1.1.0.0-6366.latest latest [reinstall almond-top]
#4463	Jun 25, 2016 7:09 PM	1.1.0.0-6366.latest latest [reinstall almond-top]
#4462	Jun 25, 2016 5:22 PM	1.1.0.0-22.latest-1.1 latest-1.1 [reinstall cream-1]
#4461	Jun 25, 2016 5:08 PM	

Project platform-ci

This build requires parameters:

command: reinstall

Command to execute. Default reinstall

environment:

Environment to use: https://asdstash.isus.emc.com/projects/UI/repos/platform-ci/browse/environments

images: stable

Images to use. Default 'latest'. Can be one of the following:

- stable - Use the stable images (Miraz)
- latest - Use the latest images (Miraz)
- stable-1.0 - Use the stable images (release-1.0)
- latest-1.0 - Use the latest images (release-1.0)
- master N I miraz N - Use build N from jenkins on master
- release N I release-1.0 N - Use build N from jenkins on release-1.0 branch

Example: master 172

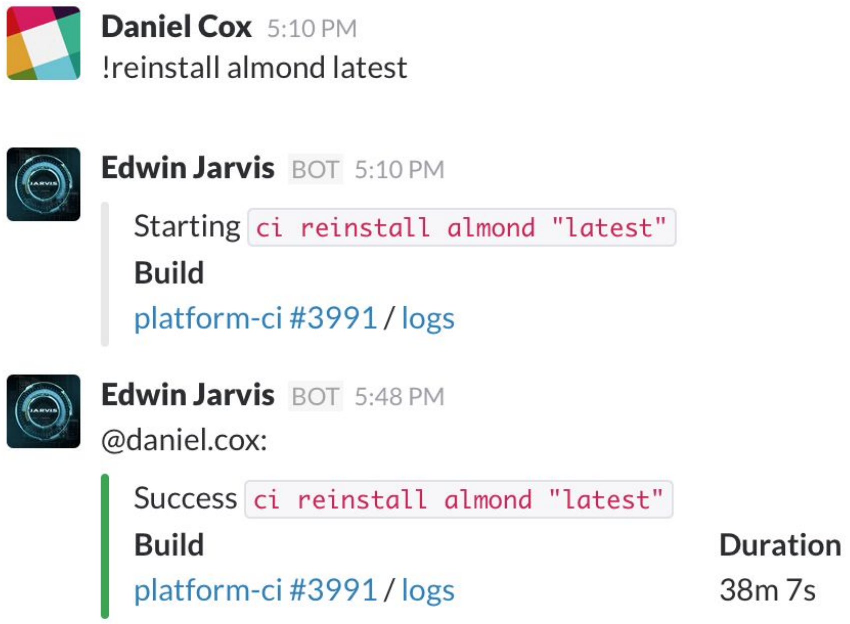
custom_tags:

Specify custom image tags:

```
bedrock=caspian/bedrock:my-custom-bedrock-tag upgrade=caspian/bedrock:my-custom-upgrade-tag
```

vars:

Add or override inventory variables:



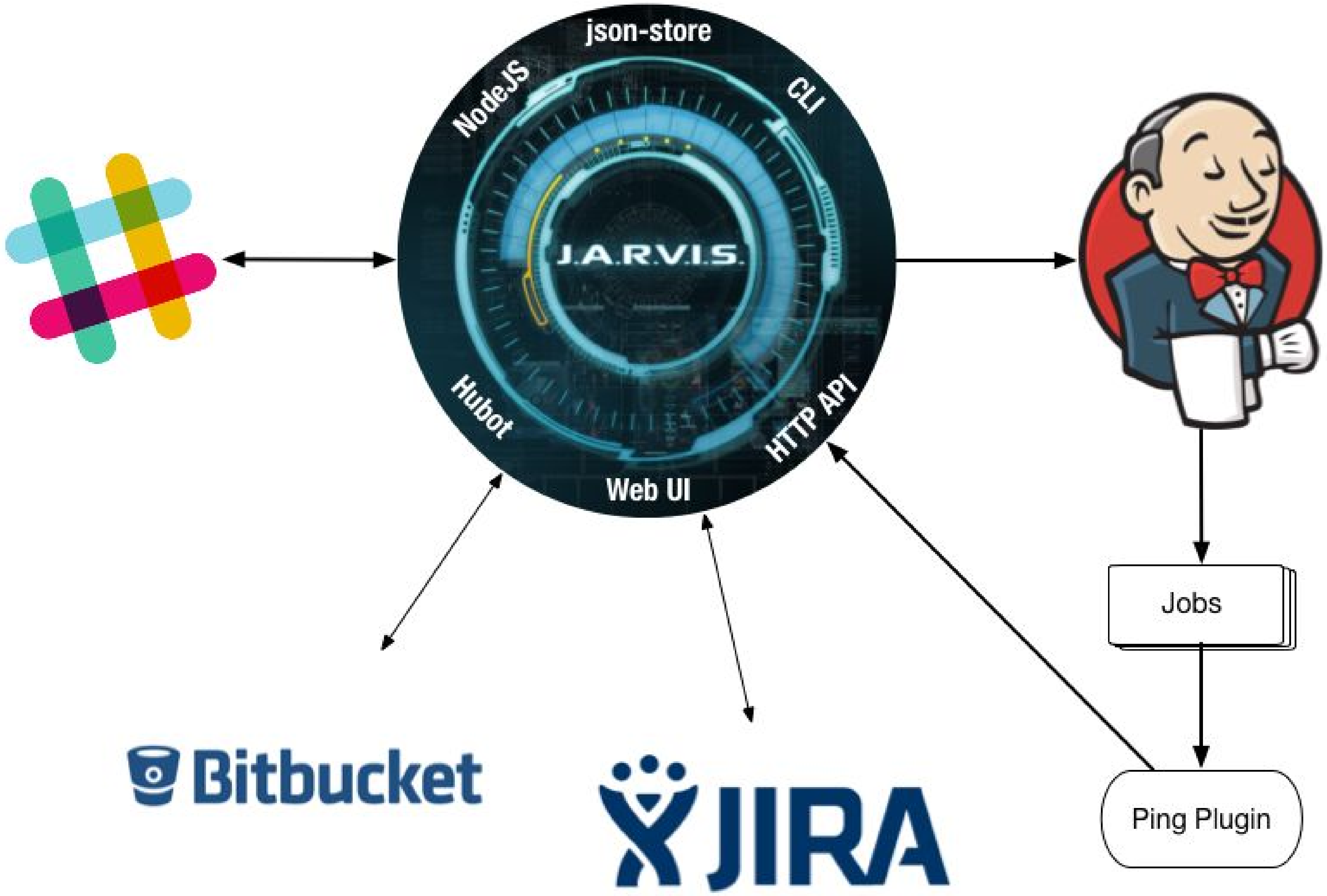
Daniel Cox 5:10 PM
!reinstall almond latest

Edwin Jarvis BOT 5:10 PM
Starting `ci reinstall almond "latest"`
Build
platform-ci #3991 / logs

Edwin Jarvis BOT 5:48 PM
@daniel.cox:
Success `ci reinstall almond "latest"`

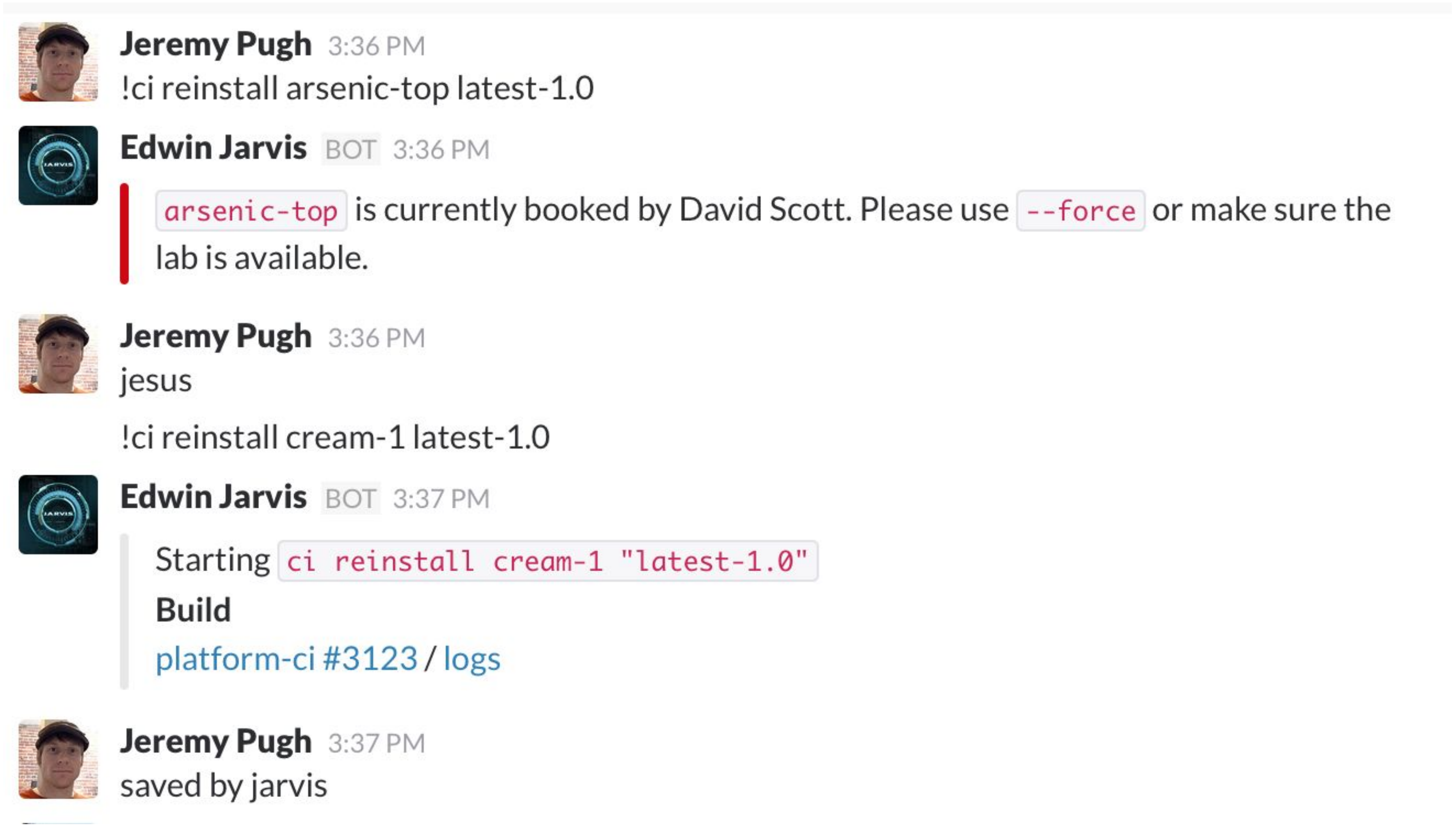
Build	**Duration**
platform-ci #3991 / logs	38m 7s

ENABLEMENT


json-store
NodeJS
CLI
J.A.R.V.I.S.
Hubot
HTTP API
Web UI
Jobs
Ping Plugin
Bitbucket
JIRA

Not Unlike Web Apps

- Web UI
- API
- Persistent database
- Always up to date
- CLI
- ACLs
  - Per username
  - Per ID
  - Per room


Jeremy Pugh 3:36 PM
!ci reinstall arsenic-top latest-1.0

Edwin Jarvis BOT 3:36 PM
`arsenic-top` is currently booked by David Scott. Please use `--force` or make sure the lab is available.

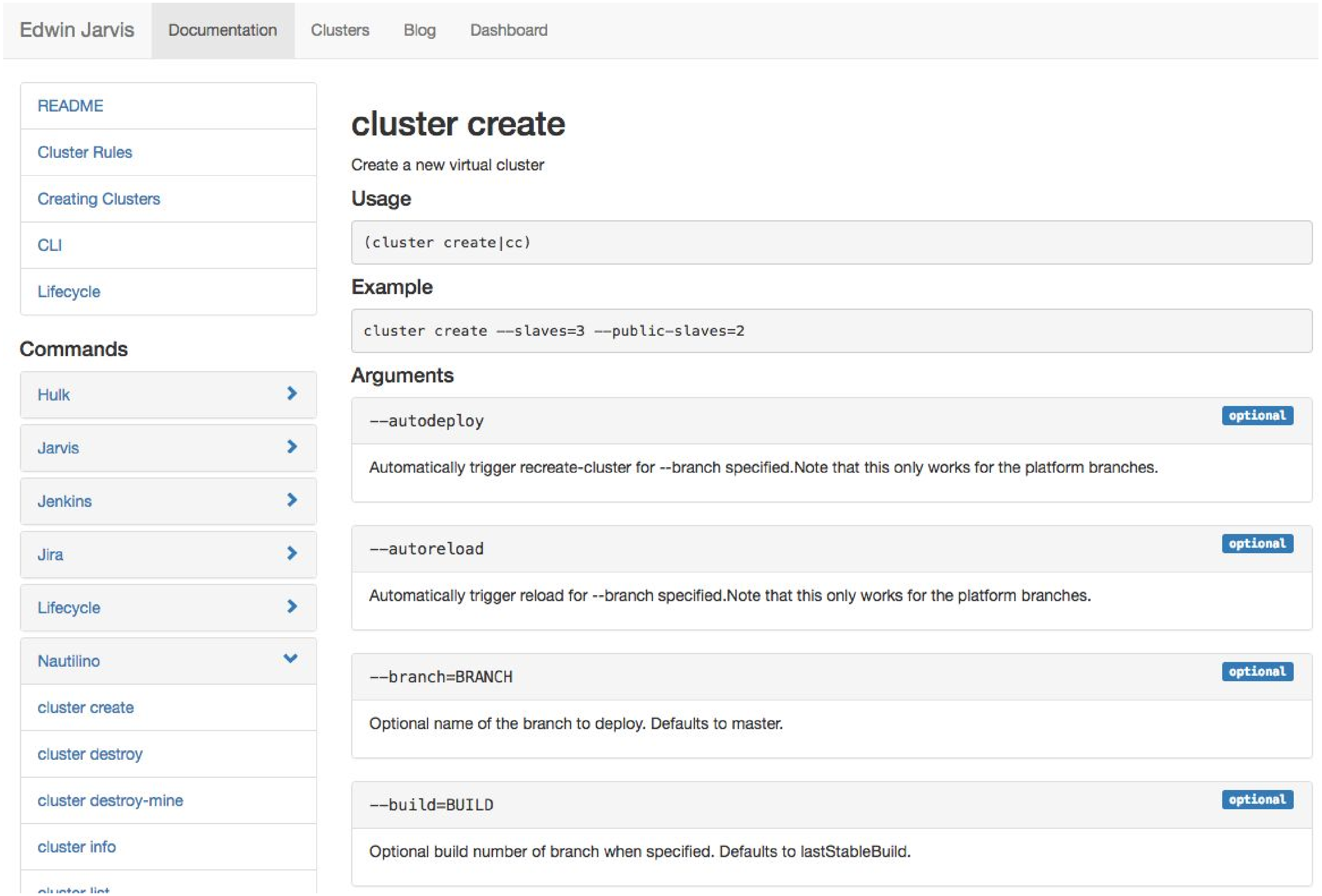
Jeremy Pugh 3:36 PM
jesus

!ci reinstall cream-1 latest-1.0

Edwin Jarvis BOT 3:37 PM
Starting `ci reinstall cream-1 "latest-1.0"`
Build
platform-ci #3123 / logs

Jeremy Pugh 3:37 PM
saved by jarvis

Discoverability



cluster create

Create a new virtual cluster

Usage

```
(cluster create|cc)
```

Example

```
cluster create --slaves=3 --public-slaves=2
```

Arguments

`--autodeploy` optional

Automatically trigger recreate-cluster for --branch specified.Note that this only works for the platform branches.

`--autoreload` optional

Automatically trigger reload for --branch specified.Note that this only works for the platform branches.

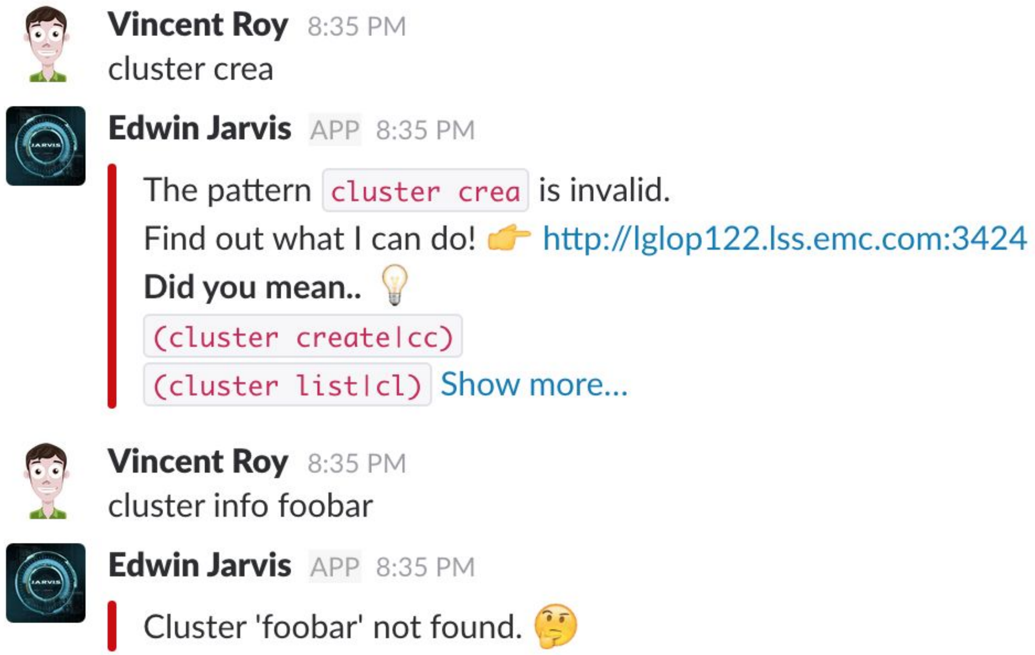
`--branch=BRANCH` optional

Optional name of the branch to deploy. Defaults to master.

`--build=BUILD` optional

Optional build number of branch when specified. Defaults to lastStableBuild.

Useful Error Messages



Vincent Roy 8:35 PM
cluster crea

Edwin Jarvis APP 8:35 PM
The pattern `cluster crea` is invalid.
Find out what I can do! 👉 http://lglop122.lss.emc.com:3424
Did you mean.. 💡
`(cluster create|cc)`
`(cluster list|cl)` Show more...

Vincent Roy 8:35 PM
cluster info foobar

Edwin Jarvis APP 8:35 PM
Cluster 'foobar' not found. 🤔

Automatic Build & Deploy

- Easy to run locally.
    - docker-compose up --build
- Easy to deploy.
    - git push origin master
- Monorepo
    - Chatbot and automation code lives in main repo

Challenges

- Testing
  - Integration
  - Personas
- Corporate Network

IT’S JUST CODE

KAIZEN

KAIZEN

KAIZEN

QUESTIONS?