SAUCECON 19

WELCOME TO THE WORKSHOP

Optimizing the Testing Experience with Analytics and the Sauce Labs REST API

SAUCELABS

SAUCE LABS

GOALS

After completing this workshop, you will be able to:

- Learn how to access and use common Sauce Labs REST API endpoints
- Work through common use cases for the Sauce API with exercises and demos
- Learn how to use the Analytics API endpoints to gather and utilize test trends and test metrics

Agenda

- **First Half: API Methods**
  - Introduction
  - Account
  - Job
  - Activity and Usage
  - Tunnel
- **Break**
- **Second Half: Analytics**
  - Test Analytics UI
  - Build Efficiency
  - Tools / Integrations
  - Q & A

SAUCELABS

Your Trainer

James Tacker

Technical Content Producer at Sauce Labs

@spider-sauce

@SauceSpider

/in/jamestacker

SAUCELABS

Prerequisites

- REST/HTTP basic knowledge
- CLI or REST Client
  - Postman
  - `curl`
- For Hands On Portion:
  - `git`
  - `node/npm`

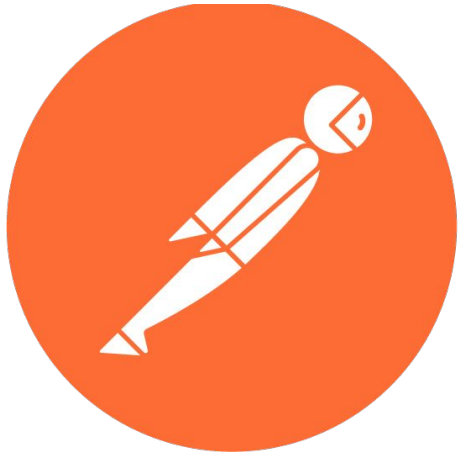
If you need help fulfilling these requirements please visit the open lab across the hall

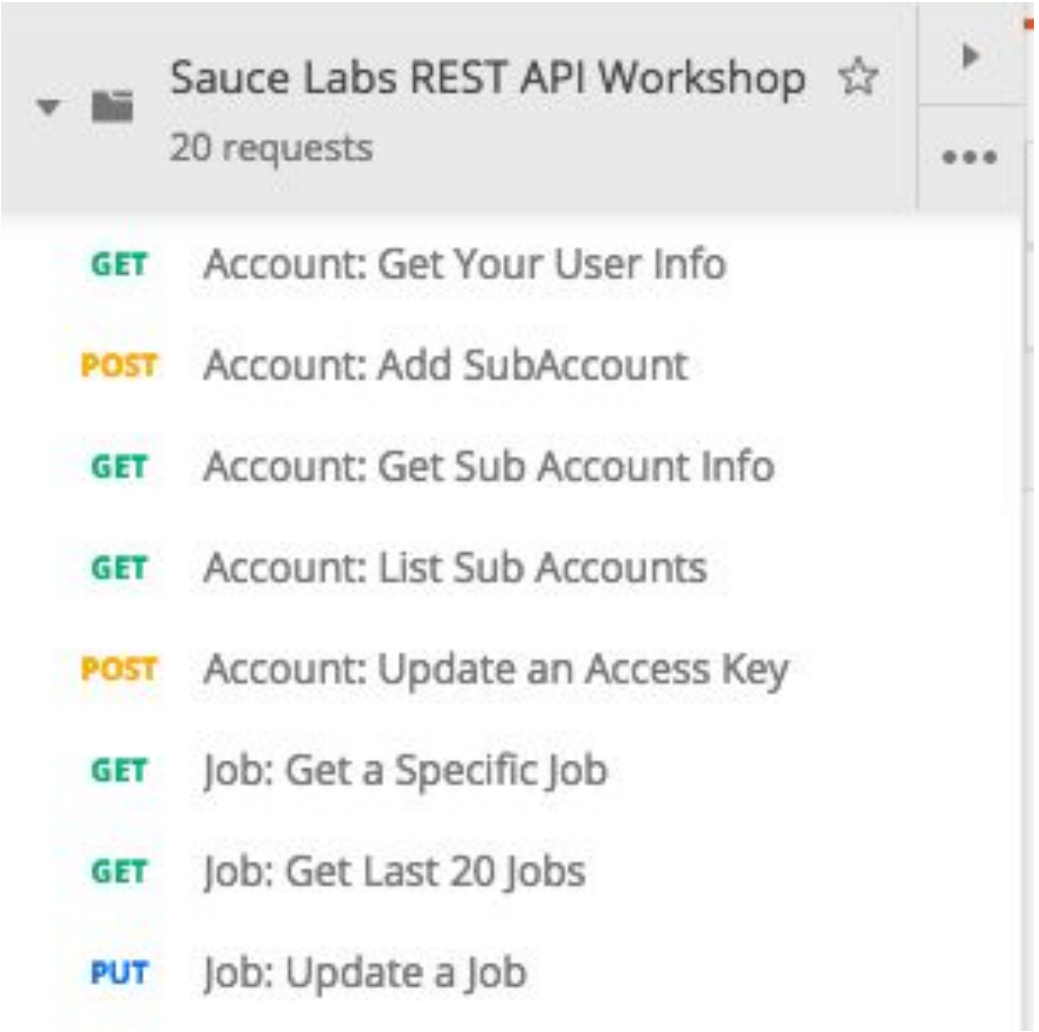
Refer to the README in the project or view it here.

Setup

- **Download / Install Prerequisites**
  - If you're missing software, or need help configuring your environment, please go to the lab across the hall
- **Open Project / Grab from GitHub:**
  - Download the Latest Release
  - Open Postman
  - Import Postman Collection

SAUCELABS





Testing the API with Postman Collections

SAUCE LABS

Introduction

SAUCE LABS

Objectives

After completing this section, you will be able to:

- Access the Sauce Labs API
- Identify API security best practices
- Authenticate your account against the Sauce Labs Hub

SAUCELABS

Accessing the API

- Choosing a Data Center
- JSON Encoding
- Authentication:
  - Request URL
  - Authorization Header

SAUCELABS

REST API Endpoints

Data Center	REST API Endpoint	Whitelisted IPs
US	`https://saucelabs.com/rest`	`162.222.72.0/21, 66.85.48.0/21`
EU	`https://eu-central-1.saucelabs.com/rest`	`185.94.24.0/22`

- Accessed over HTTPS
- Uses basic authentication methods
- Request and response data uses JSON encoding
- Visit the following link: https://wiki.saucelabs.com/display/DOCS/Accessing+the+API for other Sauce Labs service endpoints

SAUCELABS

Client Libraries

- Managed by third parties:
  - NodeJS
  - Ruby
  - Python
  - PHP
  - Java

SAUCELABS

Authentication

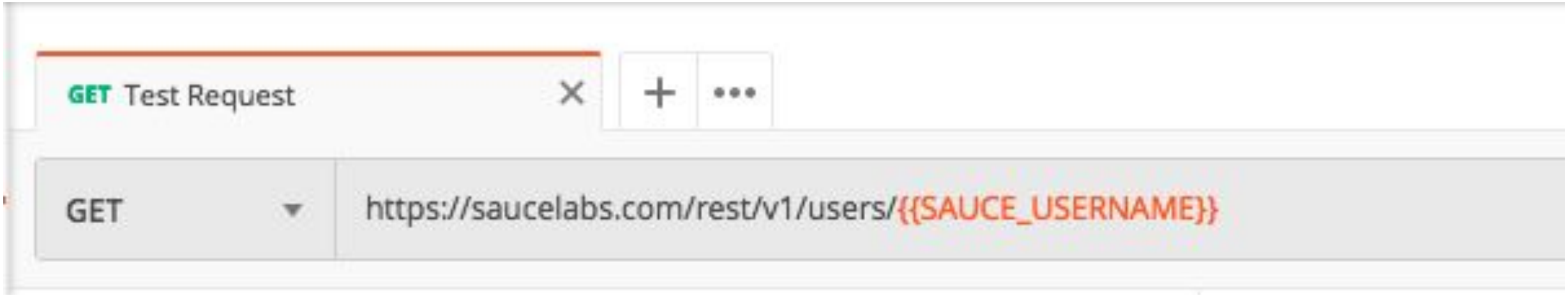
Request URL	`https://SAUCE_USERNAME:ACCESS_KEY@saucelabs.com/rest/v1/users/SAUCE_USERNAME`
Auth Header	`curl -u SAUCE_USERNAME:ACCESS_KEY https://saucelabs.com/rest/v1/users/SAUCE_USERNAME`

- Sauce Labs REST API uses HTTP Basic Authentication
- If you're using the Authorization header method:
  - you must use a command line tool like `curl` or a REST client such as Postman
  - all Sauce API methods default to `GET`, so `PUT` or `POST` must have `content-type` set to `application/json`

SAUCELABS

Security Best Practices

- Tips to avoid API Access Key farming:
  - Do not expose API Access Key in public places e.g. GitHub
  - Use environment variables and/or JSON web tokens (JWT)
    For example: Travis CI + Sauce Connect + JWT Addon
- Review OWASP SaaS API security standards
- Rotate keys frequently (either manually or programmatically)

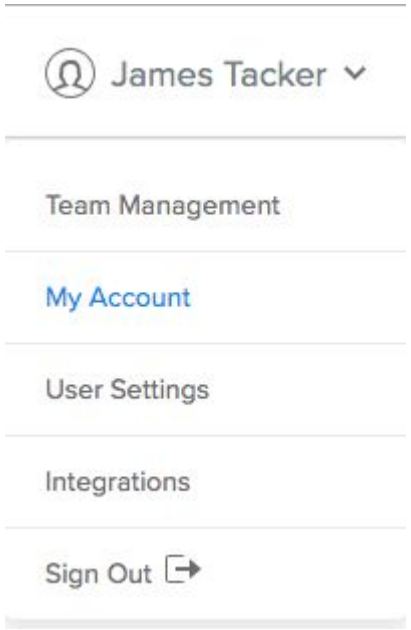


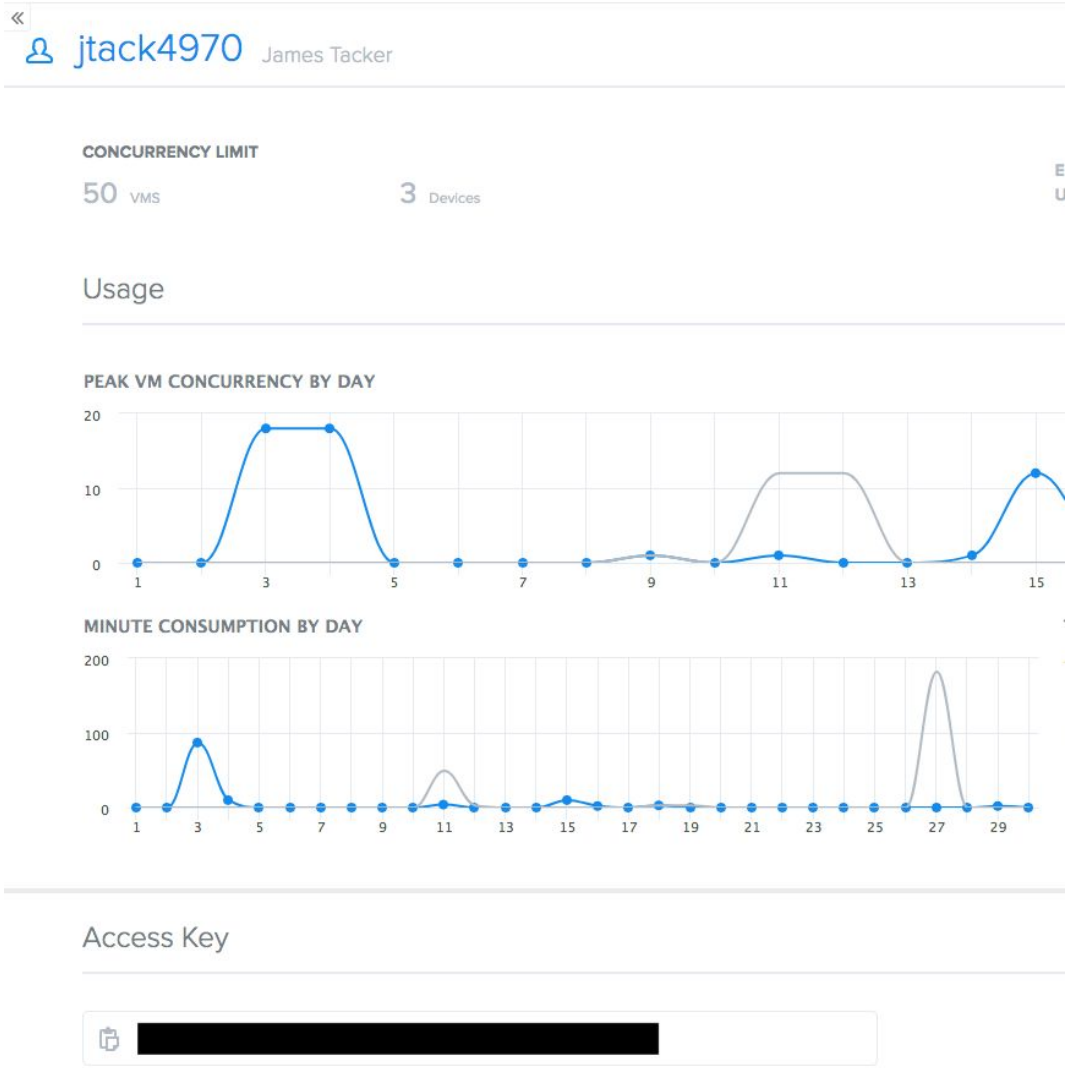
Example GET Request with Postman

Exercise #1

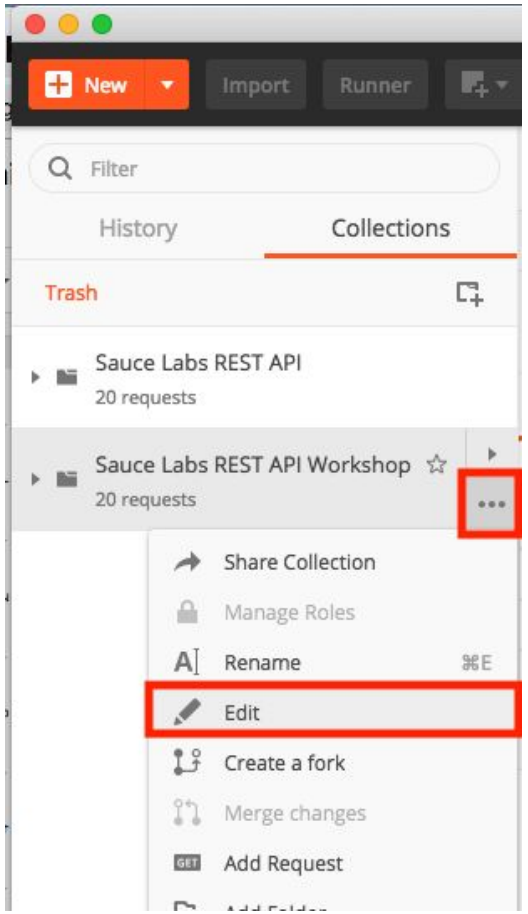
- Grab Credentials:
  - Go to SauceLabs.com
  - Retrieve your Username and Access Key
- Open Postman and update the collection variables
- If Testing APIs with `curl`:
  - open up a terminal (OSX) or command prompt (Win)
  - create two environment variables:
    - `SAUCE_USERNAME`
    - `SAUCE_ACCESS_KEY`

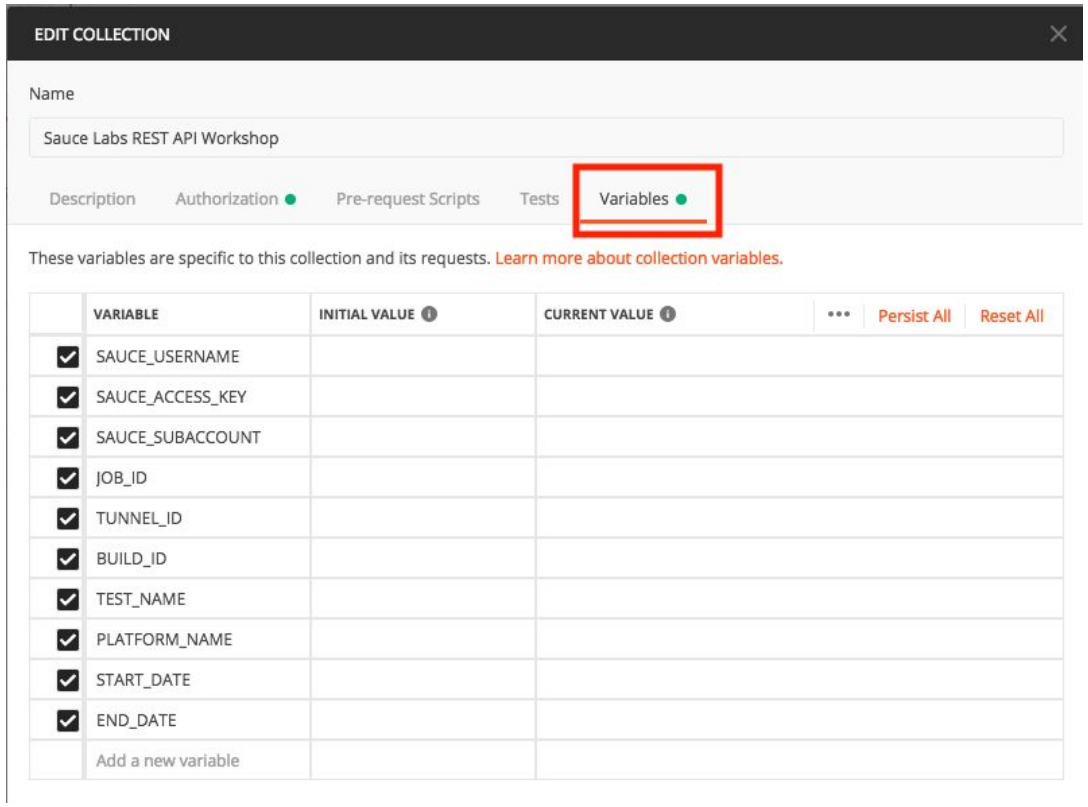
SAUCELABS





Grabbing the Username and Access Key





Read about Collection Variables here:
https://learning.getpostman.com/docs/postman/environments_and_globals/variables/

Update Postman Collection Variables

```
$ export SAUCE_USERNAME="your saucelabs username"
$ export SAUCE_ACCESS_KEY="your saucelabs API access Key"
```

```
curl -u $SAUCE_USERNAME:$SAUCE_ACCESS_KEY \
https://saucelabs.com/rest/v1/users/$SAUCE_USERNAME
```


```
{
   "username":"xxxxxx",
   "vm_lockdown":false,
   "new_email":null,
   "last_name":"xxxxxx",
   "tunnels_lockdown":false,
   "parent":null,
   "subaccount_limit":3,
   "team_management":false,
   "creation_time":1543341104,
   "user_type":"invoiced",
   "monthly_minutes":{
      "manual":"infinite",
      "automated":"infinite"
   },
...
```


Set Credentials and Test the API with CLI Tools

SAUCE LABS

Account API Methods

SAUCE LABS

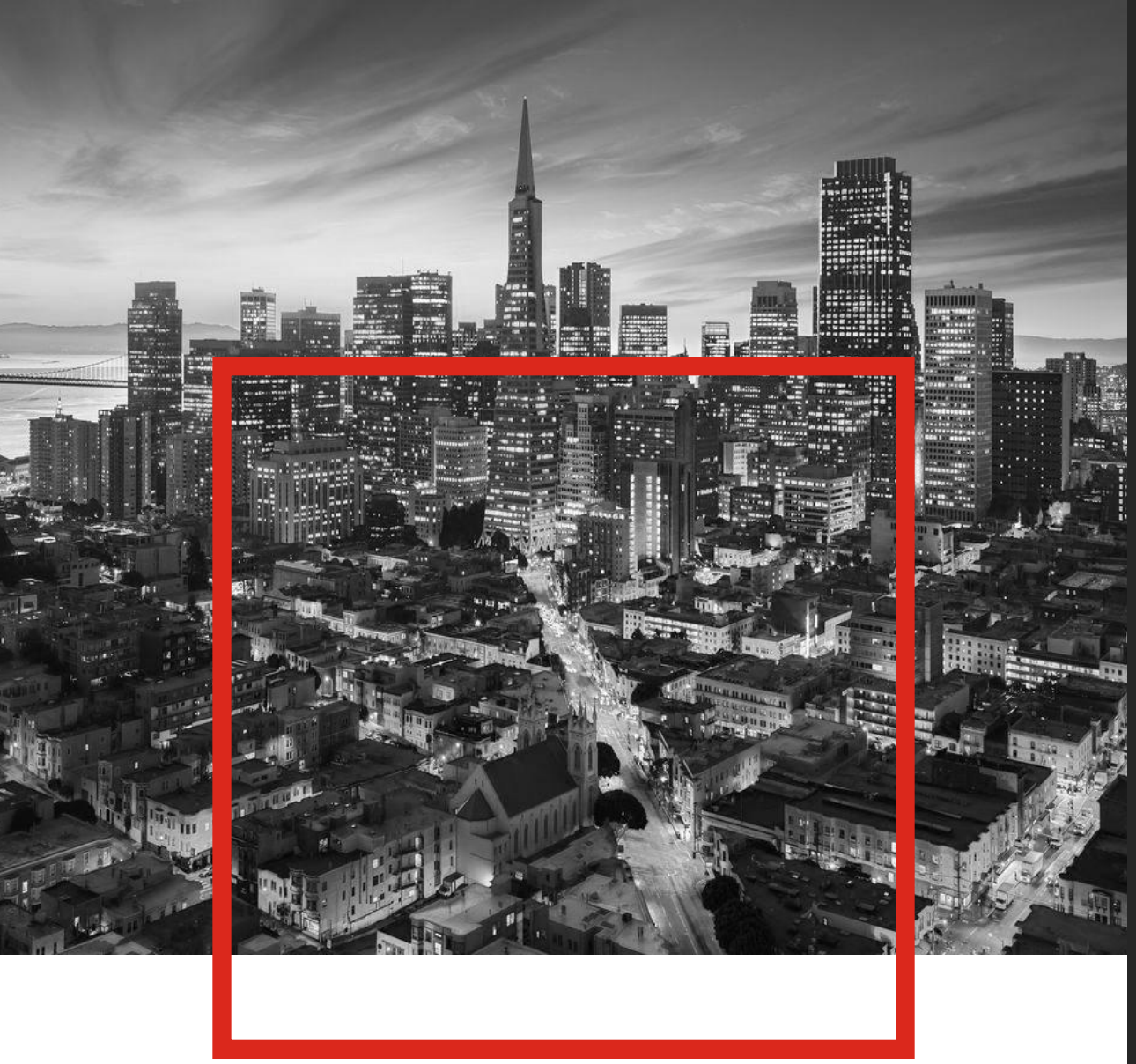
Objectives

After completing this section, you will be able to:

- Gather user and account information
- Create, manage, and monitor different account types
- Update API access key

Team Management

- Org Account
  - Team Accounts
  - Sub Accounts
    - User
    - Sibling



Real World Scenarios

- Addition of new team
- Security breach
- Lost or stolen hardware

SAUCELABS

SAUCELABS

Account Methods Part 1

Use Case	HTTP Method	REST API Endpoint	Required Parameters
Grab basic account info	`GET`	**`/rest/v1/users/SAUCE_USERNAME`**	**`N/A`**
Create a Sub account User	`POST`	**`/rest/v1/users/SAUCE_USERNAME`**	• `username` • `password` • `email` • `name`

SAUCELABS

Account Methods Part 2

Use Case	HTTP Method	REST API Endpoint	Required Parameters
List Sub account users	GET	**/rest/v1/users/SAUCE_USERNAME/subaccounts**	**N/A**
List Sibling account users	GET	**/rest/v1.1/users/SAUCE_USERNAME/siblings**	**N/A**
Update Access Key	POST	**/rest/v1/users/SAUCE_USERNAME/accesskey/change**	**N/A**

POST Add SubAccount

Add SubAccount

POST https://saucelabs.com/rest/v1/users/{{SAUCE_USERNAME}}

Params Authorization Headers (2) Body Pre-request Script Tests

none form-data x-www-form-urlencoded raw binary JSON (application/json)

```
{
    "username":"fake-user",
    "password":"fake-password",
    "name":"fake-user",
    "email":"fakeEmail1234@hotmail.com"
}
```

Add a Sub Account

POST Update an Access Key

Update an Access Key

POST https://saucelabs.com/rest/v1/users/{{SAUCE_SUBACCOUNT}}/accesskey/change

Params Authorization Headers (1) Body Pre-request Script Tests

none form-data x-www-form-urlencoded raw binary

Body Cookies Headers (12) Test Results

Pretty Raw Preview JSON

```
1 {
2     "accessKey": "[redacted]"
3 }
```

Update an Access Key

Exercise #2

- Scenario:
  - A new offshore team is being onboarded to the Sauce Labs platform and you need to **create new sub accounts**.
  - After creating the accounts one of the employees mistakenly uploaded their access key to a public repository. Now you must **update the user's access key**.
- Objectives
  - Use Postman to **create a new sub account** in your org
  - Use Postman to **rotate the user's access key**

SAUCELABS

SAUCE LABS

Job API Methods

SAUCE LABS

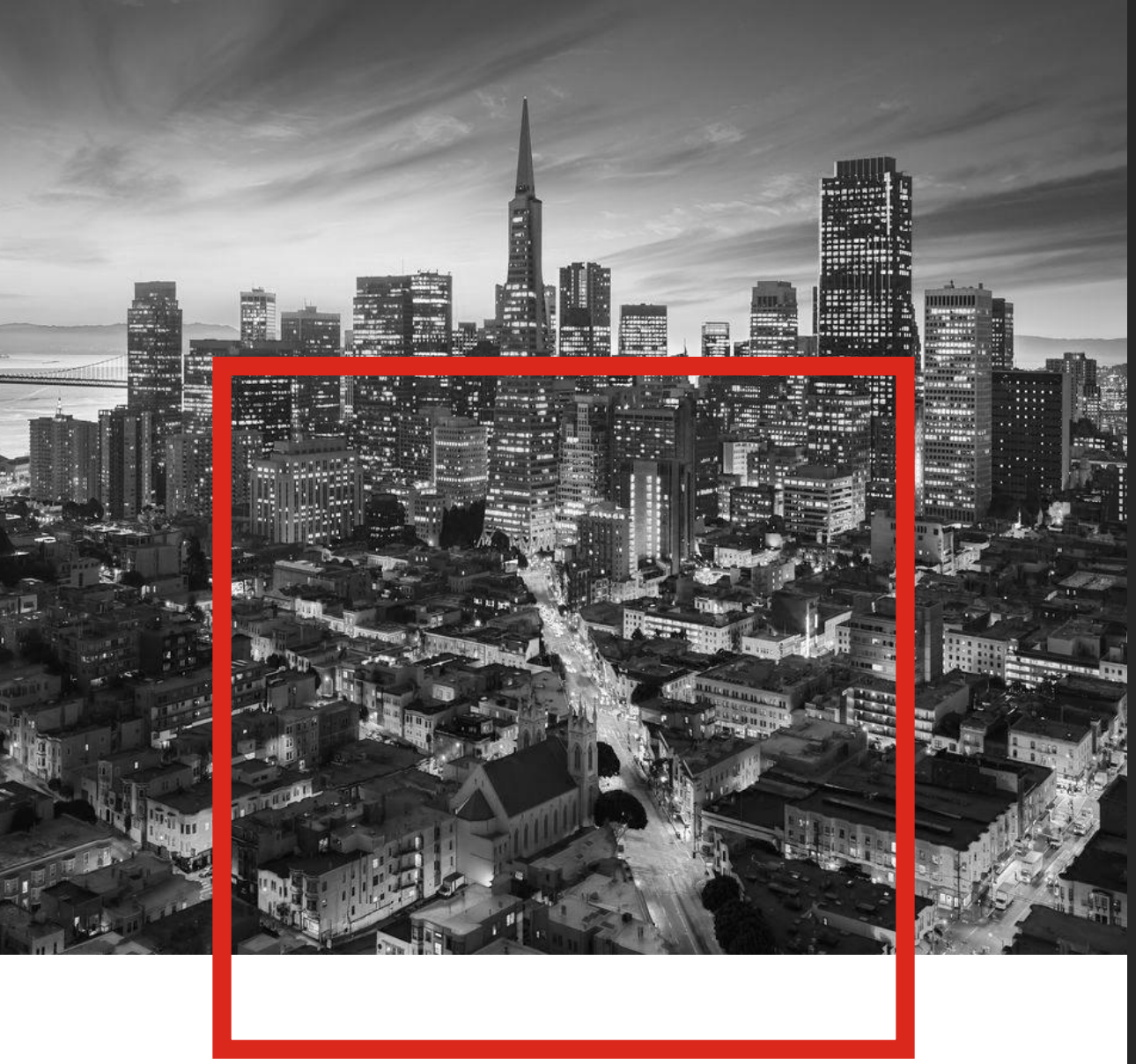
Objectives

After completing this section, you will be able to:

- Examine or modify job information
- Manage job asset files

Job Management

- Get, Update, Delete, Stop
- Grab and Delete Asset Files



Real World Scenarios

- Pull job info for cloud BI tool
- Change specific job visibility
- Update pass/fail job status
- Add metadata to jobs
- Manage job assets
- Stop jobs that are timing out

SAUCELABS

SAUCELABS

Job Methods Part 1

Use Case	HTTP Method	REST API Endpoint	Required Parameters
Grab jobs based on username	GET	**/rest/v1/SAUCE_USERNAME/jobs**	**N/A**
Update jobs based on ID	PUT	**/rest/v1/SAUCE_USERNAME/jobs/JOB_ID**	**N/A**
Delete jobs based on ID	DELETE	**/rest/v1/SAUCE_USERNAME/jobs/JOB_ID**	**N/A**
Stop Running Job	PUT	**/rest/v1/SAUCE_USERNAME/jobs/JOB_ID/stop**	**N/A**

SAUCELABS

Job Methods Part 2

Use Case	HTTP Method	REST API Endpoint	Required Parameters
Get job assets	GET	**/rest/v1/SAUCE_USERNAME/jobs/JOB_ID/assets/FILE_NAME**	• **file_name** Accepted Values: • **selenium-server.log** • **video.mp4** • **{number}screenshot.png** • **final_screenshot.png**
Delete job assets	DELETE	**/rest/v1/SAUCE_USERNAME/jobs/JOB_ID/assets**	**N/A**

- {number} refers to the index of the screenshot, basically any number between 0000-9999

SAUCELABS

Request Examples:

Use Case	API Request
Get last 10 jobs from a specific user, skip 2 most recent jobs	`curl -x GET -u $SAUCE_USERNAME:$SAUCE_ACCESS_KEY https://saucelabs.com/rest/v1/$SAUCE_USERNAME/jobs?limit=10&skip=2`
Update job information metadata	`curl -x PUT -u $SAUCE_USERNAME:$SAUCE_ACCESS_KEY \ -H "Content-Type: application/json" \ -d ' { "tags": "testing-job-api", "name": "job API Test" }' \ https://saucelabs.com/rest/v1/$SAUCE_USERNAME/jobs/$JOB_ID`
Delete a user's job based on an ID	`curl -x DELETE -u $SAUCE_USERNAME:$SAUCE_ACCESS_KEY https://saucelabs.com/rest/v1/$SAUCE_USERNAME/jobs/$JOB_ID`
Stop Running Job	`curl -x PUT -u $SAUCE_USERNAME:$SAUCE_ACCESS_KEY https://saucelabs.com/rest/v1/$SAUCE_USERNAME/jobs/$JOB_ID/stop`

GET https://saucelabs.com/rest/v1/{{SAUCE_USERNAME}}/jobs?limit=20&skip=5

Params Authorization Headers (1) Body Pre-request Script Tests

Query Params

	KEY	VALUE
☑	limit	20
☑	skip	5
	Key	Value

Body Cookies Headers (12) Test Results

Pretty Raw Preview JSON


```
[
    {
        "id": "b7d04c521226437bb472c350124dfcd8"
    },
    {
        "id": "809baa8d71f640e193578a6a668a1945"
    },
    {
        "id": "7514a145ba75441fb9052108f99cc492"
    },
    {
```


Get Job Request (grabs last 20, skips first 5)

PUT Update Job

Update Job

PUT https://saucelabs.com/rest/v1/{{SAUCE_USERNAME}}/jobs/{{JOB_ID}}

Params | Authorization | Headers (3) | Body | Pre-request Script | Tests

none | form-data | x-www-form-urlencoded | raw | binary | Text

```
1 {
2     "tags":[
3         "testing-rest-api",
4         "update-job-test"
5     ]
6 }
```

+ Update Job Request (adds `tags` to metadata)

```
if (result.isSuccess()) {
    sauce.jobPassed(((RemoteWebDriver) getWebDriver()).getSessionId().toString());
} else {
    sauce.jobFailed(((RemoteWebDriver) getWebDriver()).getSessionId().toString());
}
```

Documentation for: [Java API Client Library]

Updating Jobs with the API Client Libraries

```
    @AfterMethod
    public void teardown(ITestResult result) {
        ((JavascriptExecutor) driver)
                .executeScript( s: "sauce:job-result=" +
                        (result.isSuccess() ? "passed" : "failed"));
        driver.quit();
    }
}
```

Examples of: [Using JavaScriptExecutor with Sauce Labs]

Documentation for: [JavaScriptExecutor]

Updating Jobs with the `JavaScriptExecutor`

Exercise #3

- Scenario:
  - The executives scoped out a new cross-functional project that involves a separate team.
  - You as the admin must loop in the new team. They require access to a schedule jobs that was restricted to your team only. Use the **update job API** to set **visibility** to **shared**
- Objectives
  - Use the **get job API** to **list the last 10** but **skip first 5** jobs
  - Use the **update job API** to **set a job visibility status** to **shared** across your “team” (sub-accounts)
  - Attempt to **add custom data** using the **update job API** (use Postman example as baseline)

SAUCELABS

SAUCE LABS

Test Activity and Usage API Methods

SAUCE LABS

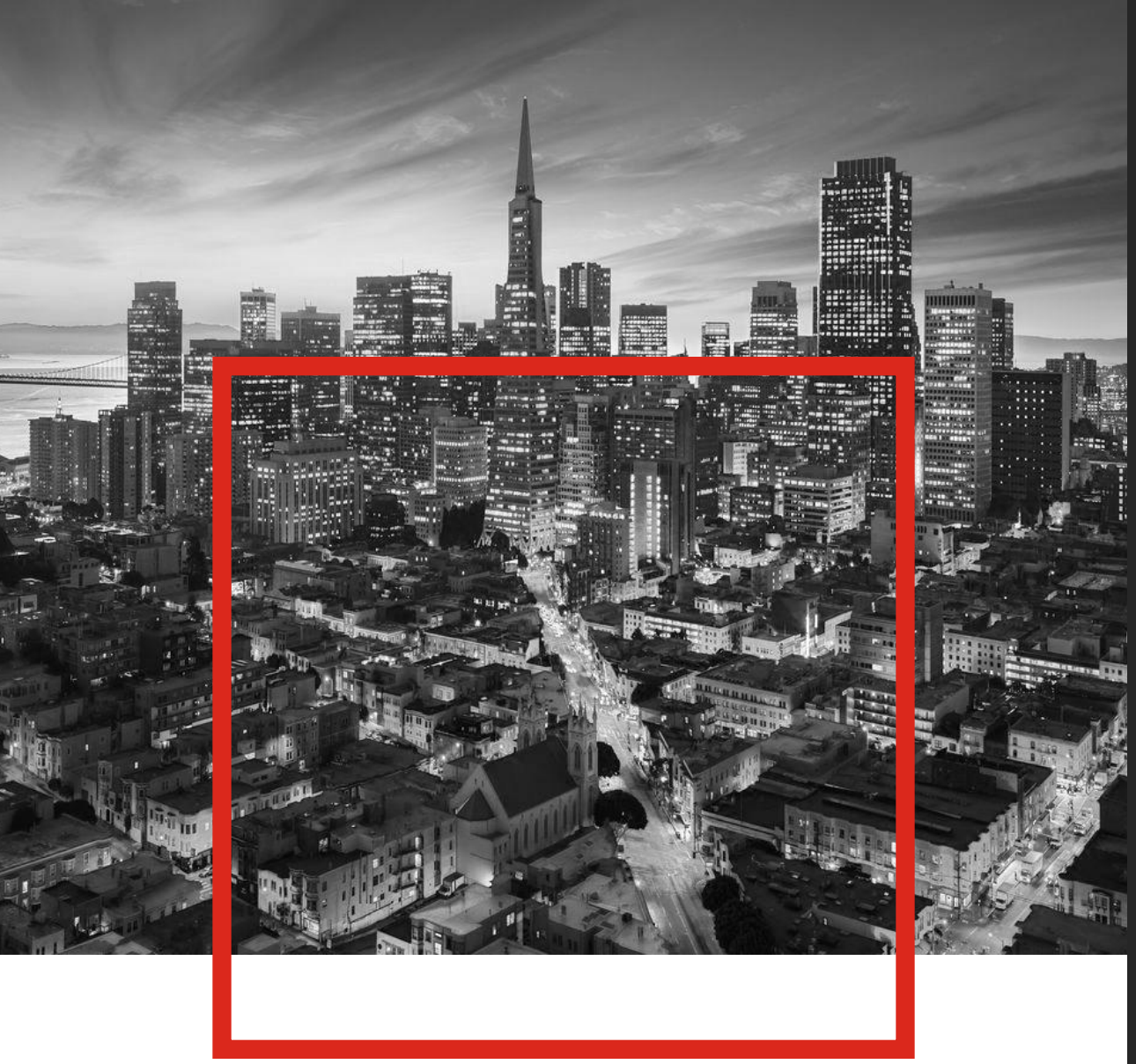
Objectives

After completing this section, you will be able to:

- Monitor and set user concurrency limits
- Monitor Account Usage

Test Activity

- Monitor and set account concurrency
- Monitor account jobs:
  - Total amount
  - Job status
  - Children job details



Real World Scenarios

- Monitor user VM usage
- Determine employee VM allowance based on activity
- Filter usage based on time range in order to investigate pipeline or release bottlenecks

SAUCELABS

SAUCELABS

Test Activity Methods

Use Case	HTTP Method	REST API Endpoint
Get account concurrency limits	`GET`	**`/rest/v1.1/users/SAUCE_USERNAME/concurrency`**
Get current account activity	`GET`	**`/rest/v1/SAUCE_USERNAME/activity`**
Get current account usage	`GET`	**`/rest/v1/users/SAUCE_USERNAME/usage`**

SAUCELABS

Concurrency Example:

Use Case	API Request
Grab historical data from a specific timeframe	`curl -u $SAUCE_USERNAME:$SAUCE_ACCESS_KEY` `https://saucelabs.com/rest/v1.1/users/$SAUCE_USERNAME/concurrency`

- JSON response returns the amount of VMs allowed per user including the parent and children accounts. For example a VM readout may look like this:

```
"concurrency" : {
      "ancestor" : {
         "allowed" : {
            "mac" : 100,
            "real_device" : 30,
            "manual" : 100,
            "overall" : 100
         },
         "username" : "ANCESTOR_USERNAME",
         ...
```

SAUCELABS

Activity Example:

Use Case	API Request	Example JSON Response
Grab a specified account's real time activity	`curl -u $SAUCE_USERNAME:$SAUCE_ACCESS_KEY https://saucelabs.com/rest/v1/$SAUCE_USERNAME/activity`	`{ "totals" : { "in progress" : 30, "all" : 45, "queued" : 15 }, "subaccounts" : { "DEVOPS_TEAM" : { "in progress" : 20, "all" : 30, "queued" : 10 }, "DEVOPS_CHILD" : { "in progress" : 10, "all" : 15, "queued" : 15 } } }`

SAUCELABS

Usage Example:

Use Case	API Request	Example JSON Response
Grab historical data from a specific timeframe	`curl -u $SAUCE_USERNAME:$SAUCE_ACCESS_KEY https://saucelabs.com/rest/v1/users/$SAUCE_USERNAME/usage&start=2018-03-01&end=2018-03-20`	`{ "usage" : [ ["2018-3-20", [5, 467]], ["2018-3-1", [7, 114]] ], "username" : "$SAUCE_USERNAME" }`

- Possible Query Parameters
  - “start”
  - “end”
- Format = YYYY-MM-DD
- Response = the total job number and VM time grouped by day

GET Get User Activity

Get User Activity

GET https://saucelabs.com/rest/v1/{{SAUCE_USERNAME}}/activity

Params | Authorization | Headers (1) | Body | Pre-request Script | Tests

KEY	VALUE
Authorization	Basic anRhY2s0OTcw
Key	Value

Body | Cookies | Headers (13) | Test Results

Pretty | Raw | Preview | JSON


```
{
    "subaccounts": {
        "jtack4970": {
            "in progress": 0,
            "all": 0,
            "queued": 0
        },
        "fake-user2": {
            "in progress": 0,
            "all": 0,
            "queued": 0
        }
    },
    "totals": {
        "in progress": 0,
        "all": 0,
        "queued": 0
    }
}
```


Grab User Account Activity

GET Activity: Grab Usage History

Activity: Grab Usage History

GET https://saucelabs.com/rest/v1/users/{{SAUCE_USERNAME}}/usage?start=2019-02-20&end=2019-03-20

Params Authorization Headers (1) Body Pre-request Script Tests

Query Params

	KEY	VALUE	DESCRIPTION
☑	start	2019-02-20	
☑	end	2019-03-20	
	Key	Value	Description

Body Cookies Headers (13) Test Results Status: 200 OK Time: 85

Pretty Raw Preview JSON


```
{
    "usage": [
        [
            "2019-2-20",
            [
                1,
                24
            ]
        ],
        [
            "2019-2-22",
            [
                347,
                11274
            ]
        ],
        [
            "2019-2-28",
            [
                229,
                5268
```


Grab Usage History From Specific Date Range

Exercise #4

- Scenario:
  - Your organization's Sauce Labs account has a finite amount of VMs, it's up to you as the release manager and sauce labs admin to **gather user concurrency** and calculate the VMs to each individual account.
  - However the number of VMs and team members is dynamic, therefore the way you determine the VM value must be programmatic and instant
- Objectives
  - Gather **account user concurrency**
  - Experiment with the **test-activity** and **usage** query parameters (use examples in Postman as a baseline)

SAUCELABS

SAUCE LABS

Tunnel Methods

SAUCE LABS

Objectives

After completing this section, you will be able to:

- Install and configure Sauce Connect
- Setup tunnel pools
- Monitor tunnels using the REST API

What is a Tunnel?

- Secure connection to run tests of site or app sitting behind a firewall
- Requires additional client-side configuration

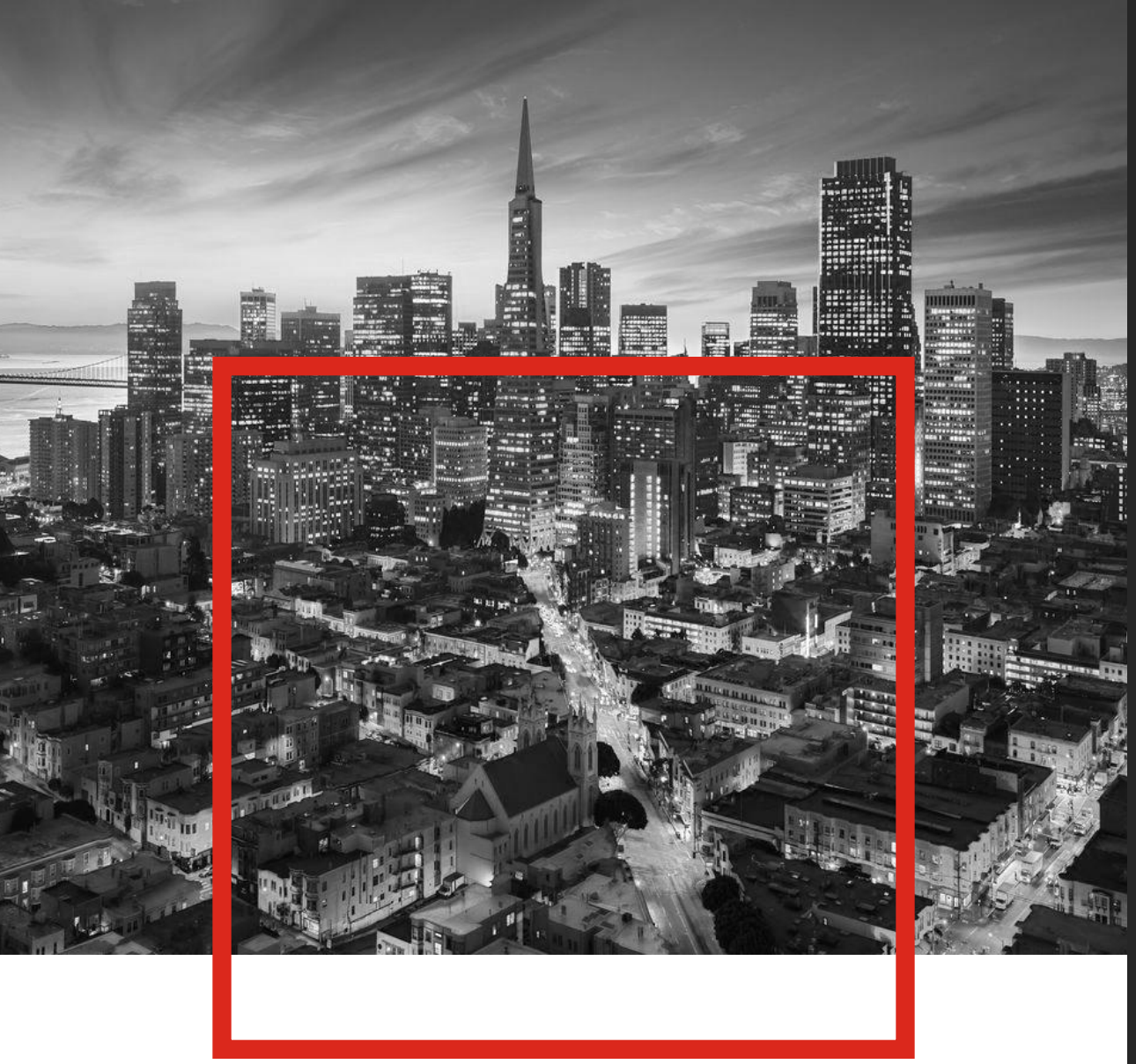
SAUCELABS

Tunnel Basics

- Loop in net security team
- Download and install the necessary client
  - IPSec VPN
  - Sauce Connect
- Configure the proxy
- Set any ENV variables
- Test the network config
- Run a test using a tunnel

SAUCELABS

Use Case for Tunnel REST API - Tunnel Traffic Control



Real World Scenarios

- Query details about specific tunnels to investigate network timeouts or bad gateway errors
- Compare job distribution across tunnels to troubleshoot network load balancer issues

SAUCELABS

SAUCELABS

Tunnel Methods

Use Case	HTTP Method	REST API Endpoint
Get all tunnels for a specific user	GET	**/rest/v1/SAUCE_USERNAME/tunnels**
Get tunnel information based on ID	GET	**/rest/v1/SAUCE_USERNAME/tunnels/TUNNEL_ID**
Get number of jobs for the past 60 seconds	GET	**/rest/v1/SAUCE_USERNAME/tunnels/TUNNEL_ID/num_jobs**
Delete a tunnel	DELETE	**/rest/v1/SAUCE_USERNAME/tunnels/TUNNEL_ID**

GET Get Tunnels

Get Tunnels

GET https://saucelabs.com/rest/v1/{{SAUCE_USERNAME}}/tunnels

Params | Authorization | Headers (1) | Body | Pre-request Script | Tests

KEY	VALUE
Authorization	Basic anRhY2s0OTcwOjMwNj
Key	Value

Body | Cookies | Headers (9) | Test Results

Pretty | Raw | Preview | JSON

```
1 [
2     "03a49e74ecc64bb49379d02f3d445047"
3 ]
```

Get All Running Tunnel IDs

Tunnel: Delete a Tunnel

DELETE https://saucelabs.com/rest/v1/{{SAUCE_USERNAME}}/tunnels/{{TUNNEL_ID}}

Params | Authorization | Headers (1) | Body | Pre-request Script | Tests

Query Params

KEY	VALUE
Key	Value

Body | Cookies | Headers (9) | Test Results

Pretty | Raw | Preview | JSON


```
1 {
2     "jobs_running": 0,
3     "result": true,
4     "id": "dae5cf536eac4e85b63d655011f3b618"
5 }
```


Delete a Tunnel (based on its id)

Break

Enjoy the 15 minute break!

- Grab some coffee
- Talk amongst each other
- Ask questions
- Continue working through exercises

SAUCE LABS

Test Analytics Interface

SAUCE LABS

Objectives

After completing this section, you will be able to:

- Use filters to identify desired test information
- Update test trends to visualize data
- Use insights to improve test efficiency

Filtering Test History

- Filter test results based on:
  - Test Name
  - Owner
  - Build
  - OS
  - Browser
  - Tags
  - Timeframe

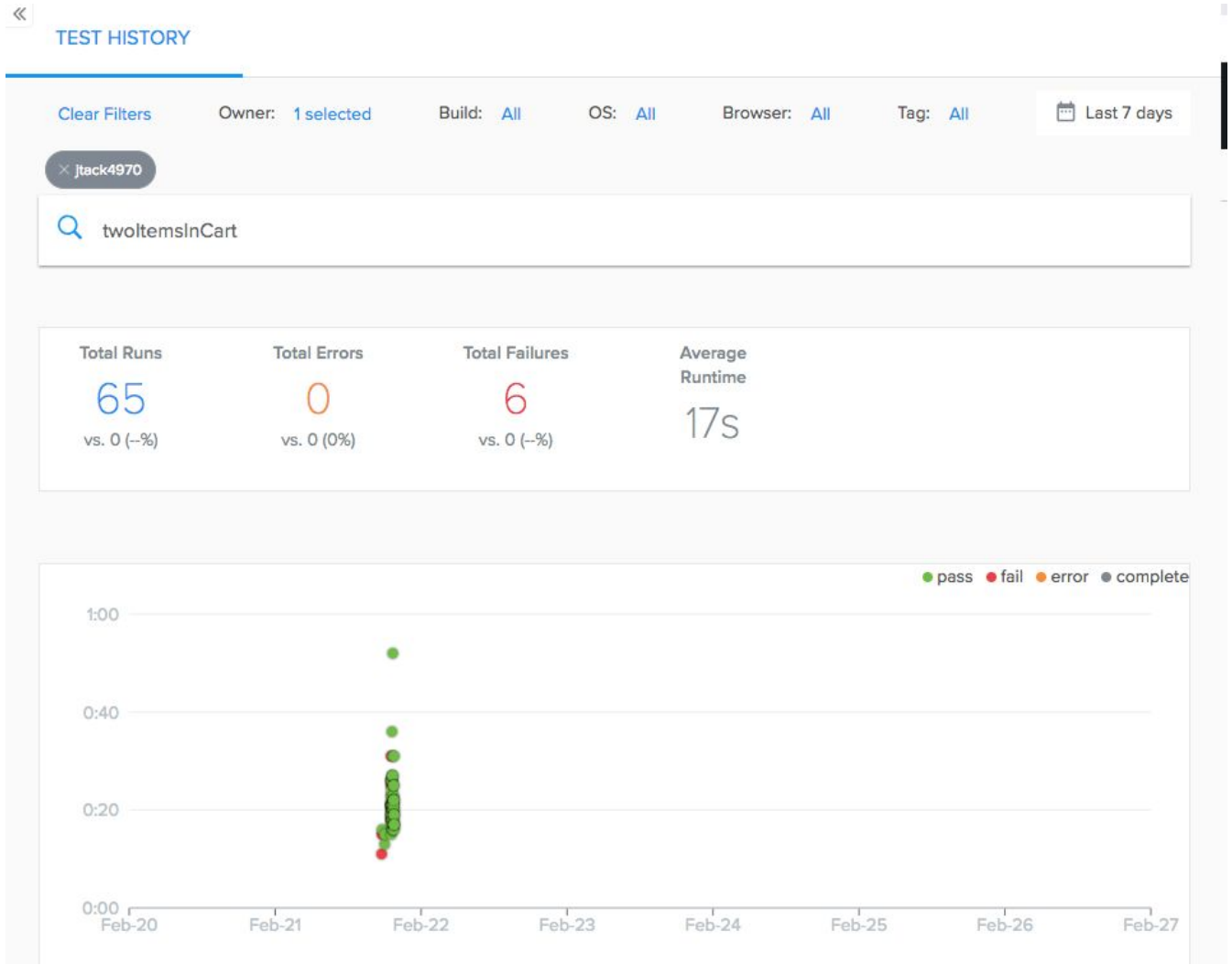
SAUCELABS

Filtering Basics

- Primary method of specifying visualization constraints
- Use the drop down options for each filtering option
- Changing filters update data visualizations
- Some options can take multiple inputs e.g. builds, owners etc.

Test History Tab

- This example filters based on:
  - Test Name: “twoItemsInCart”
  - Build Name: All
  - Owner: jtack4970
  - Build, OS, Browser, Tag: All
  - Timeframe: Last 7 days



SAUCELABS

Test Trends Basics

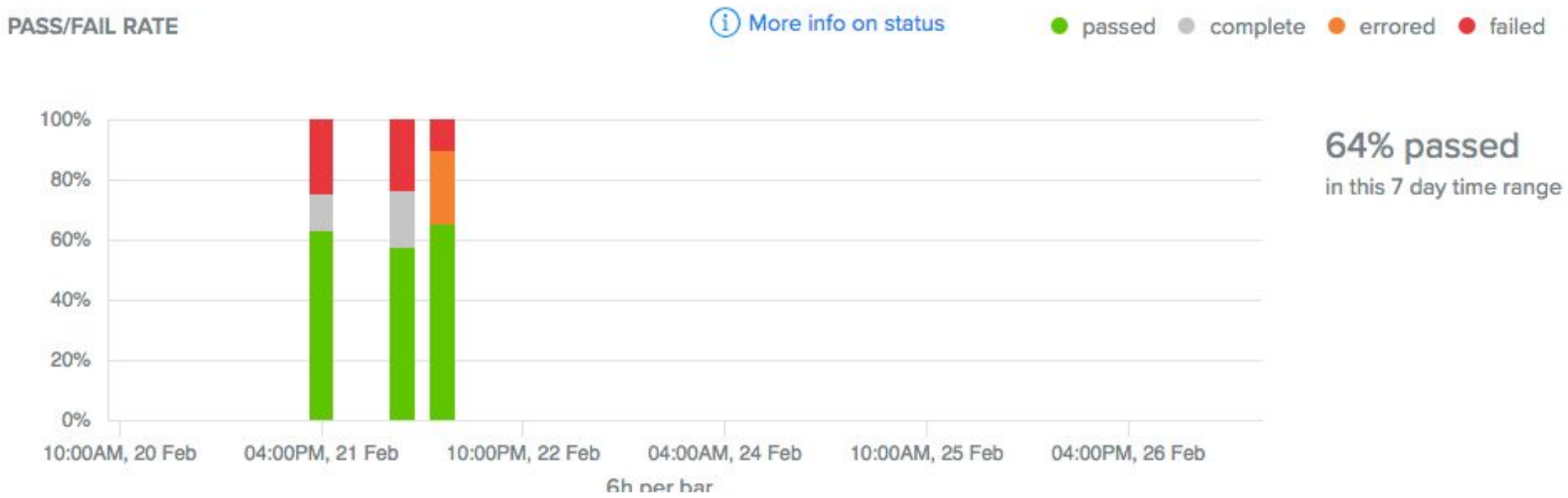
- Data visualizations in the "Trends" tab are interactive
- Filters are also available
- Visualization Graphs:
  - Number of Tests
  - Pass/Fail Rate
  - Number of Errors
  - Build and Test Statistics

SAUCELABS

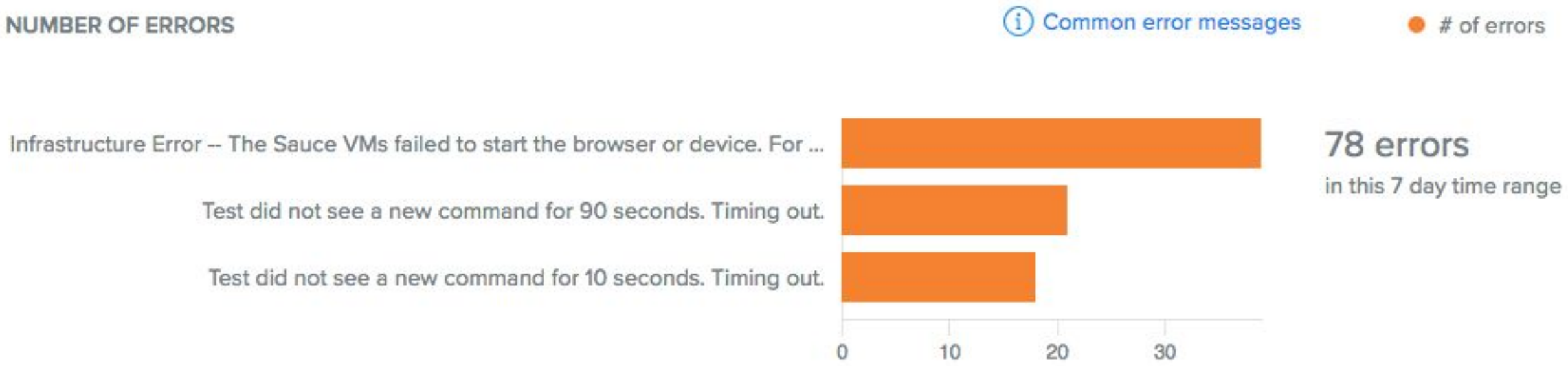
Number of Tests

SAUCELABS

Pass/Fail Rate



Number of Errors



SAUCELABS

Build and Test Statistics

BUILD AND TEST STATISTICS

Builds | Tests Without Build

FAILED TESTS ONLY | ERRORED TESTS ONLY

NAME	START TIME	DURATION	EFFICIENCY	OWNER	STATUS	ERROR
saucecon19-best-practices.02	2/22/2019, 11:08:18 ...	1m 4s	semi-parallel (90...	jtack4970	failed	(n/a for build)
saucecon19-best-practices.01	2/22/2019, 10:40:49 ...	25m 8s	semi-parallel (48...	jtack4970	failed	(n/a for build)
saucecon19-best-practices	2/21/2019, 4:27:53 PM	18h 11m 36s	sequential (0%)	jtack4970	errored	(n/a for build)

Analytics: Trends In a Date Range

GET https://saucelabs.com/rest/v1/analytics/trends/tests?interval=1h&start=2019-03-10T12:00:00Z&end=2019-03-20T12:00:00Z

Params Authorization Headers (1) Body Pre-request Script Tests

Query Params

KEY	VALUE
time_range	1h
interval	1h
start	2019-03-10T12:00:00Z
end	2019-03-20T12:00:00Z
Key	Value

Body Cookies Headers (13) Test Results

Pretty Raw Preview JSON


```
{
    "meta": {
        "status": 200
    },
    "buckets": [
        {
            "timestamp": 1552334400000,
            "datetime": "2019-03-11T20:00:00.000Z",
            "count": 2,
            "aggs": {
                "browser": [
                    {
                        "name": "Chrome 71.0",
                        "count": 2
                    }
                ],
                "browserError": [],
                "browserFail": [],
```


Query Data from a Date Range

Analytics: Aggregate Errors

GET https://saucelabs.com/rest/v1/analytics/trends/errors?start=2019-03-10T12:00:00Z&end=2019-03-20T12:00:00Z&scope=organization

Params Authorization Headers (1) Body Pre-request Script Tests

Query Params

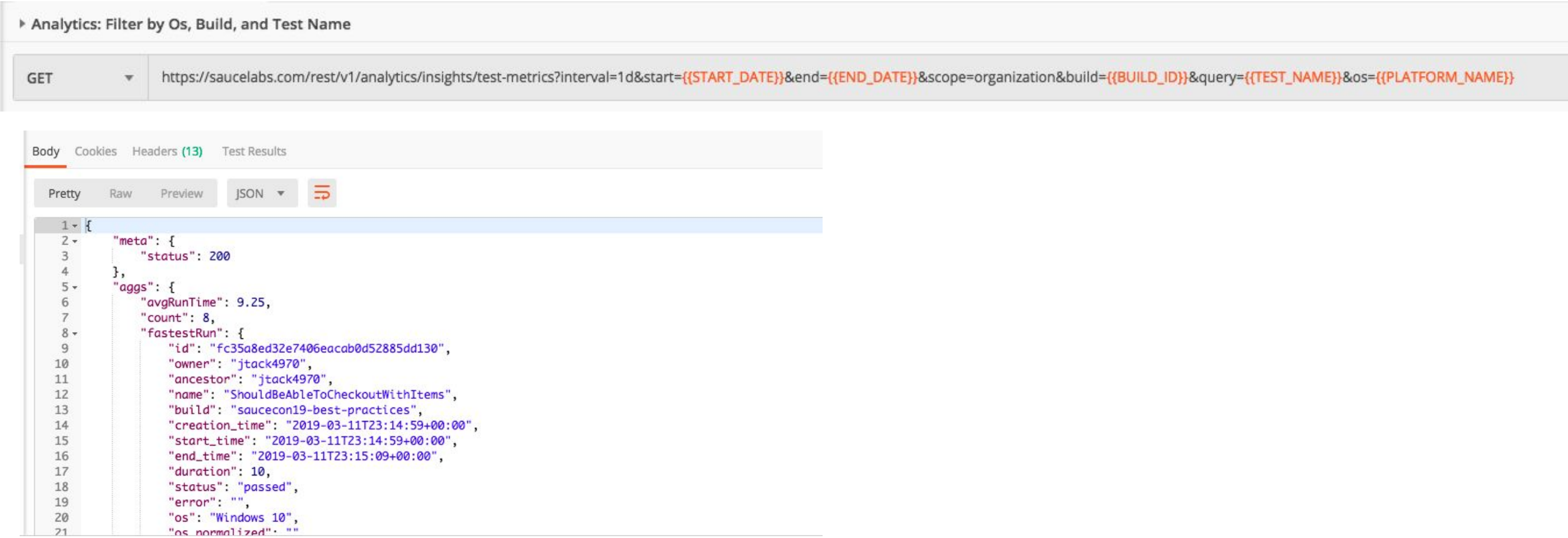
	KEY	VALUE
	time_range	1h
	interval	1h
☑	start	2019-03-10T12:00:00Z
☑	end	2019-03-20T12:00:00Z
☑	scope	organization
	Key	Value

Body Cookies Headers (13) Test Results

Pretty Raw Preview JSON


```
{
    "meta": {
        "status": 200
    },
    "buckets": [
        {
            "name": "Test did not see a new command for 100 seconds. Timing out.",
            "count": 65,
            "items": [
                {
                    "id": "8a2287232f5a4852acceeed46cd55892",
                    "owner": "jtack4970",
                    "ancestor": "jtack4970",
                    "name": "logInSuccessfully",
                    "build": "saucecon19-best-practices-workshop",
                    "creation_time": "2019-03-11T22:43:01+00:00",
```


Aggregate Test Errors in a Time Frame



Filter by OS, Build, and Test Name

SAUCELABS

LIVE DEMO: Analytics Tab

SAUCE LABS

Improve Build Efficiency

SAUCE LABS

Objectives

After completing this section, you will be able to:

- Analyze build efficiency based on parallel test runs
- Establish benchmark test durations
- Identify test inefficiencies based on test run best practices

Build Efficiency

- Indicated by the level of test parallelization based on build
- Expressed as a percentage and in the following degrees:
  - Sequential
  - Semi-parallel
  - Parallel

SAUCELABS

Sequential

- Indicates that all build time = sum total of test run time
- Indicates an absence of test framework or threads

saucecon19-best-practices	2/21/2019, 4:27:53 PM	18h 11m 36s	sequential (0%)

SAUCELABS

Semi-parallel

- build time <= sum total of test run time
- Indicates presence of test framework and/or threads, but some tests are misconfigured

saucecon19-best-practices.01	2/22/2019, 10:40:49 AM	25m 8s	semi-parallel (48%)

SAUCELABS

Parallel

- build time = longest test run time
- Indicates presence of properly configured test framework and threads

saucecon19-best-practices.03	2/28/2019, 12:22:59 PM	51s	parallelized (92%)	jtack4970

SAUCELABS

Best Practices

- Small Tests
- Atomic Tests
- Autonomous Tests

SAUCE LABS

Tool Integrations

SAUCE LABS

Objectives

After completing this section, you will be able to:

- Strategize how to integrate the Sauce REST API with various third party tools
- Deploy a POC chart website to visualize Sauce Analytics data

SAUCELABS

REST API Integrations

Checklist

- Is the API method synchronous or asynchronous?
- Data first, Sexy second...
- ChartJS or CanvasJS, good tools for rapid data visualization
- Contact customer success for assistance and recommendations

SAUCELABS

Example Tool Integration: Splunk

- Polls data from endpoint
- Uses REST Modular Input:
  - Defines input fields
  - Sets auth header
  - Sets URL args and response handlers

SAUCELABS

Example Tool Integration: Salesforce

- APEX REST Callout
- Create data structure
- Deserialize JSON
- Parse and Display
- Run report or inject into custom object

Exercise #5

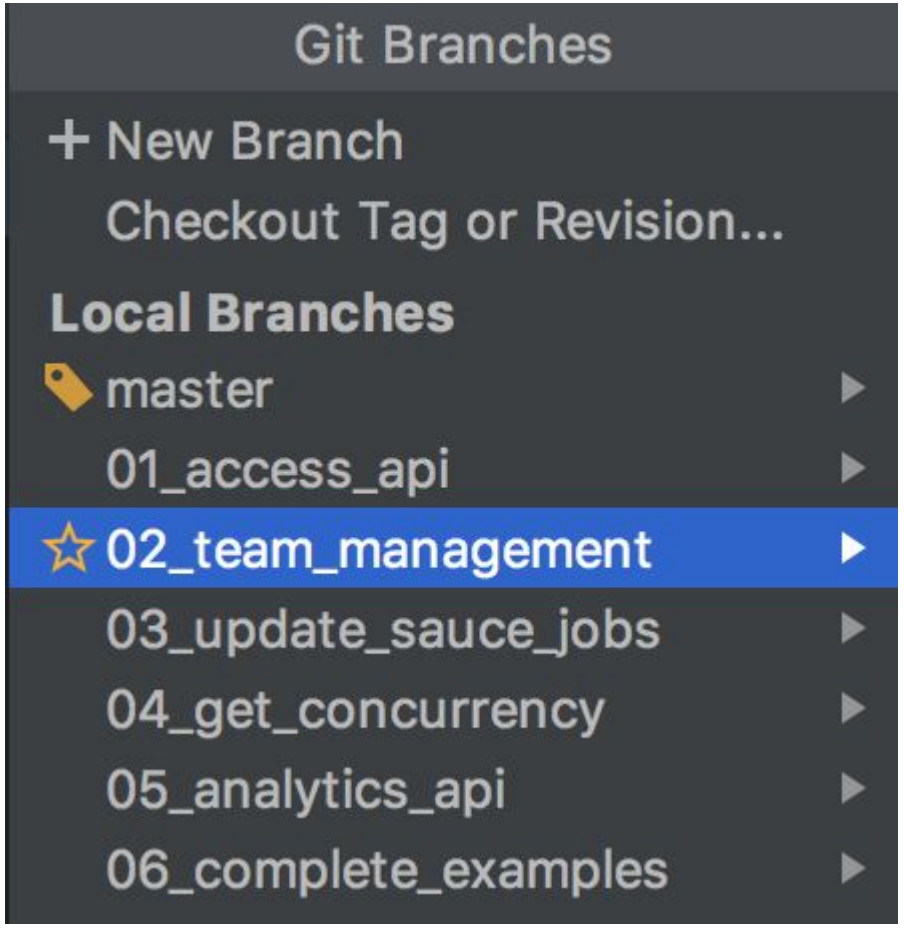
- Scenario:
  - Your company just bought a Sauce Labs.com enterprise account license.
  - The executives and QA stakeholders want to view test trends within the week. You want to gather the metrics without getting bogged down in configuring the Sauce REST API commands with your existing APM tool.
- Objectives
  - Query test-trends API and parse JSON payload
  - Visualize payload using Chart.js

SAUCELABS

SAUCELABS

Exercise Tips

- Download the release archive
- Use the local branches to `checkout` answers (see picture to the right)
- Step-by-step instructions are located in the /exercise-guides directory, here is the link to the "Getting Started" page.
- All JS examples are in the /js-examples directory



SAUCELABS

Chart.JS

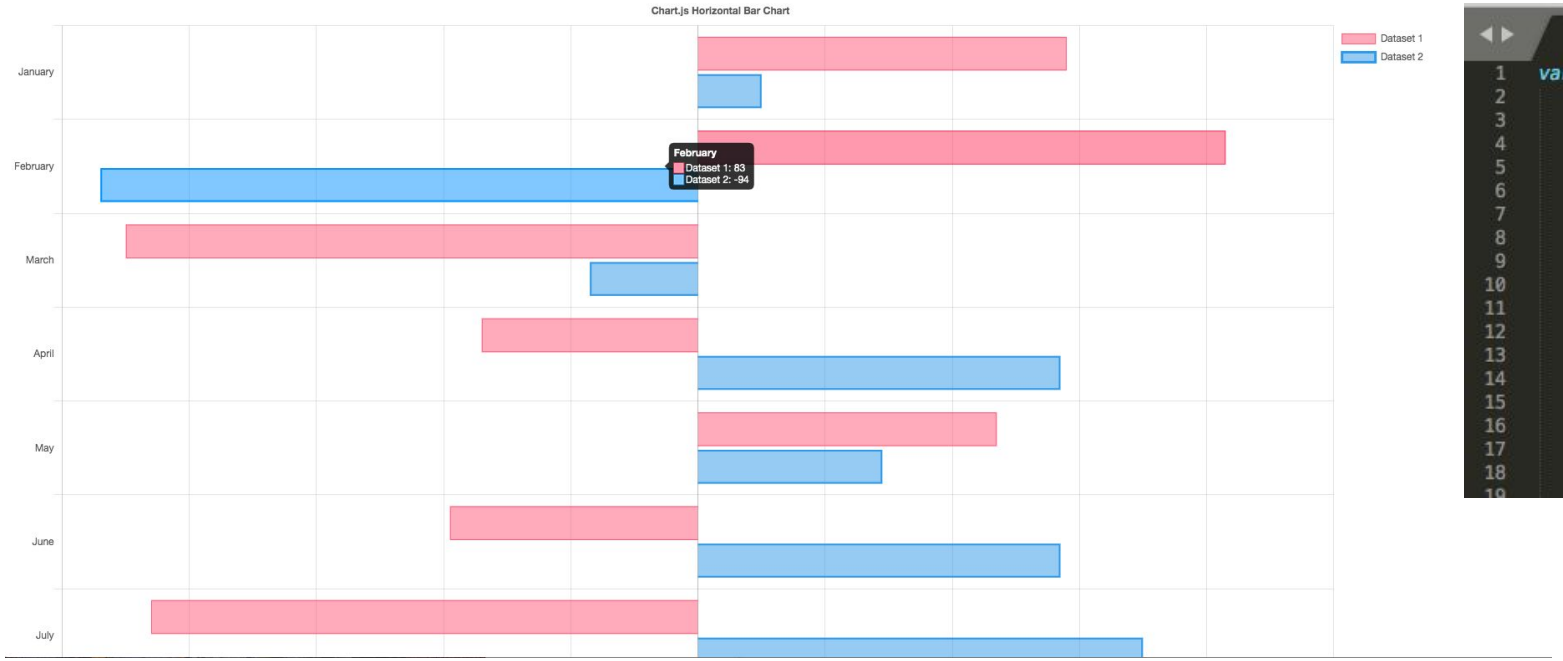
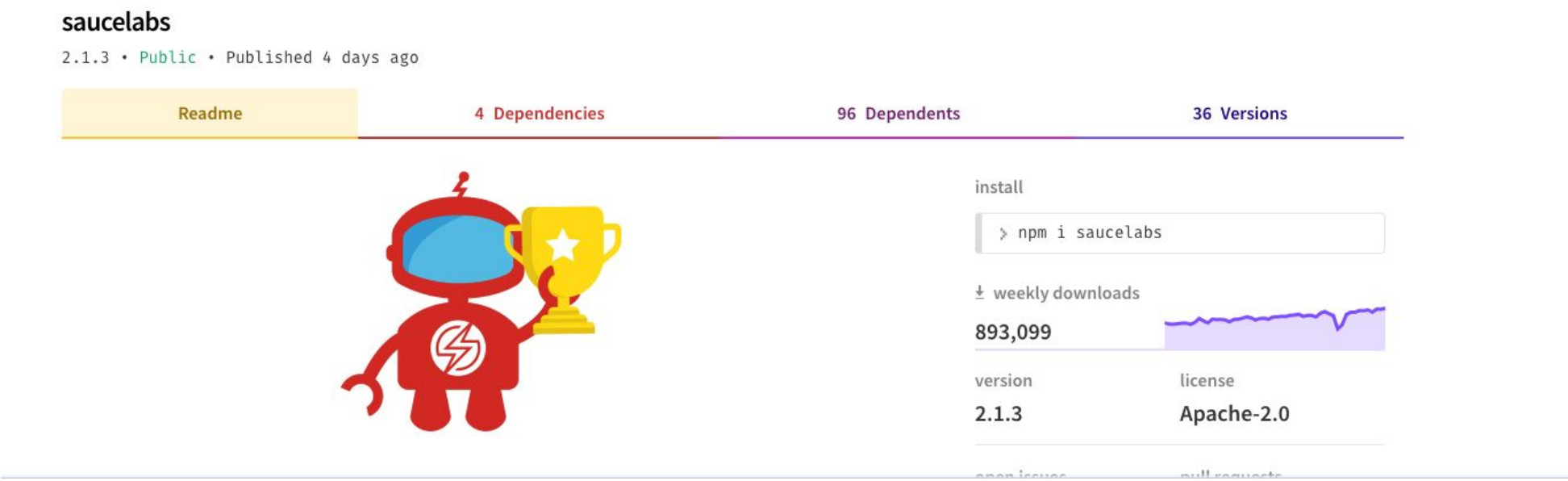


chart-psuedo-code.js

```
var MONTHS = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August',
        var color = Chart.helpers.color;
        var horizontalBarChartData = {
            labels: ['January', 'February', 'March', 'April', 'May', 'June', 'July'],
            datasets: [{
                label: 'Dataset 1',
                backgroundColor: color(window.chartColors.red).alpha(0.5).rgbString(),
                borderColor: window.chartColors.red,
                borderWidth: 1,
                data: [
                    randomScalingFactor(),
                    randomScalingFactor(),
                    randomScalingFactor(),
                    randomScalingFactor(),
                    randomScalingFactor(),
                    randomScalingFactor(),
                    randomScalingFactor()
                ]
            }, {
```

Download here: https://www.chartjs.org/

Samples here: https://www.chartjs.org/samples/latest/



NPM package located here: https://www.npmjs.com/package/saucelabs

+ saucelabs Wrapper

Hands On Exercises / Q&A

- Spend the rest of the workshop attempting the hands-on examples
- Please take our survey and enjoy the rest of the conference
- Any further questions, please email me at: james.tacker@saucelabs.com

Thank You!

SAUCELABS