Samvera - As an API

Insert Self Deprecation Slide Here

Who is this clown?

Rob Kaufman
@orangewolf
rob@notch8.com
https://www.notch8.com

Founder of Notch8 - An App Development Consultancy since 2007
This Deck
http://bit.ly/n8sc2018-1

What is an API

Application Programming Interface

For accessing data outside of the existing application

Should provide some sort of improvement over connecting to the data directly

Lots of types of APIs out there, though we will be talking about lightweight RESTful resources.

Why Would We Want an API

Split off smaller applications

Allow external users to make things we haven’t thought of yet

Prevent feature creep in the main application

Provide a more controlled interface point

Designing A Good API

Mapping to the data model

Usage first

Finding a middle ground

Versioning

Why we version

URL version vs header version

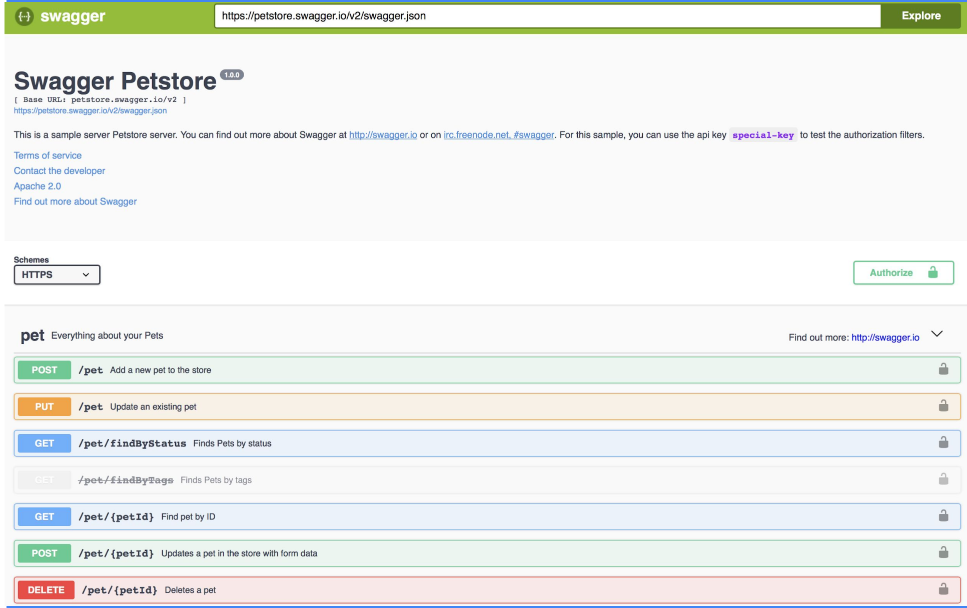
Versioning each call vs versioning whole sets

Design and Versions Quiz

http://bit.ly/sc2018-3

Documentation First

Lets get some swagger


swagger https://petstore.swagger.io/v2/swagger.json Explore

Swagger Petstore 1.0.0
[Base URL: petstore.swagger.io/v2]
https://petstore.swagger.io/v2/swagger.json

This is a sample server Petstore server. You can find out more about Swagger at http://swagger.io or on irc.freenode.net, #swagger. For this sample, you can use the api key `special-key` to test the authorization filters.

Terms of service
Contact the developer
Apache 2.0
Find out more about Swagger

Schemes: HTTPS
Authorize

pet Everything about your Pets — Find out more: http://swagger.io

POST /pet Add a new pet to the store
PUT /pet Update an existing pet
GET /pet/findByStatus Finds Pets by status
GET ~~/pet/findByTags~~ Finds Pets by tags
GET /pet/{petId} Find pet by ID
POST /pet/{petId} Updates a pet in the store with form data
DELETE /pet/{petId} Deletes a pet

Document First vs Test First

rswag

An rspec DSL that creates the Swagger json

Lets us specify both the specs for our API and our docs at the same time

More likely for docs to be accurate over time this way

https://github.com/domaindrivendev/rswag

Lets Build One

rswag exercise

1) Start with api in your docker image
`docker-compose up web`
2) In a new terminal, enter the container
`docker-compose exec web bash`
3) Take a look at the running backend by going to localhost:3000 in your browser.
4) Add an API, call it version 1 via headers and be able to perform crud options on our works. Use rswag specs to do TDD
5) Generate the completed the swagger json and take a look at the docs mounted in our Rails app at localhost:3000/api-docs

gitlab.com/notch8/samvera-connect-2018

~~rails g rswag:install~~

rails g rswag:api:install

rails g rswag:ui:install

Authentication

Let me in!

Auth - It Means Two Things

Authentication

Authentication is the process of establishing that an entity is what/who it claims to be.

In our industry the entity is often a user. Authentication is often done by providing credentials that are not publicly available, or secret, such as a password; this process is called signing in or logging in.

Authorization

Authorization is process of giving permission to an entity to access a resource. This is often done after an entity has been authenticated.

JWT

JSON Web Token (**JWT**) is a means of representing claims to be transferred between two parties. The claims in a **JWT** are encoded as a JSON object that is digitally signed using JSON Web Signature (JWS) and/or encrypted using JSON Web Encryption (JWE).

https://jwt.io

- Header
- Payload
- Signature

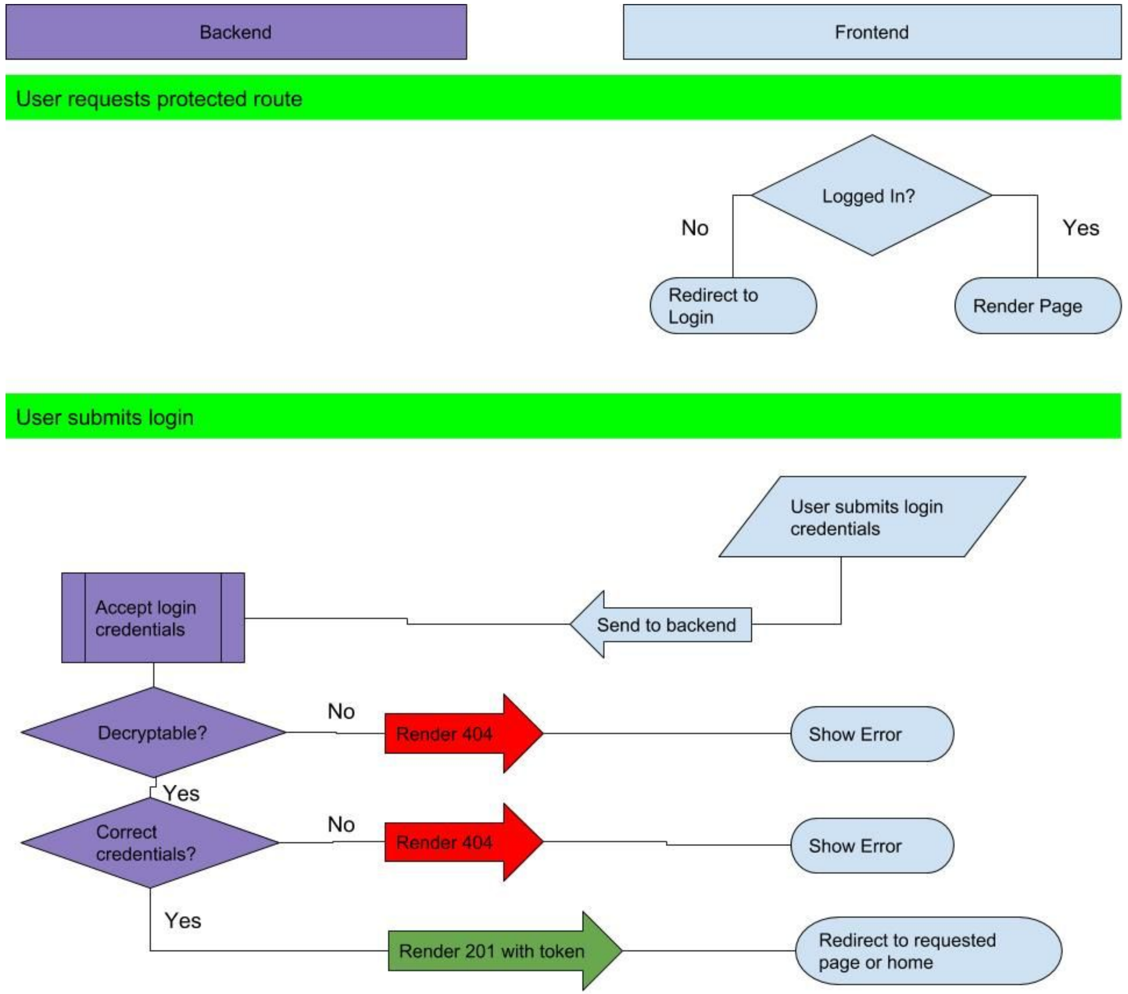
`xxxxx.yyyyy.zzzzz`

{JSON}

devise_jwt

```
Devise.setup do |config|

  # ...

  config.jwt do |jwt|

    jwt.secret = ENV['DEVISE_JWT_SECRET_KEY']

  end

end
```

```
class User < ApplicationRecord

  devise :database_authenticatable,

         :jwt_authenticatable,
jwt_revocation_strategy: Blacklist

end
```


Backend
Frontend
User requests protected route
Logged In?
No
Yes
Redirect to Login
Render Page
User submits login
User submits login credentials
Send to backend
Accept login credentials
Decryptable?
No
Render 404
Show Error
Yes
Correct credentials?
No
Render 404
Show Error
Yes
Render 201 with token
Redirect to requested page or home

CORS

Cross Origin Resource Sharing

Prevents JS front ends from interacting with our Rails / Hyrax based application

config.middleware.insert_before 0, Rack::Cors do

```
allow do
    origins 'http://your.frontend.domain.com'
    resource '/api/*',
      headers: %w(Authorization),
      methods: :any,
      expose: %w(Authorization),
      max_age: 600
  end
end
```

Revocation

Stop a token from being valid any more

Strategies

- JIT Matcher
- Blacklist
- Whitelist

Complex Authentication

Creating responsive views for site log in

Going to site to walk through login process

Returning to the app with a token as a final step

JS: `window.PostMessage` can help getting things out of HTML views

Token Auth Challenge

Token Auth Challenge

1) Start with the code from exercise 1 or from the api-swaggerd directory if you want ot skip to a done one.
2) Add devise-jwt's config to your application and set up token auth
3) Add cors config to your application
4) Restart after setting up these 2 configs
5) In a new terminal, log in to the running container with `docker-compose exec web bash`
6) Start the react app by going to api-auth directory and running yarn start
7) Check that you can log in via localhost:3001 in your browser

Hint: gems should be in place

Hint: run rails db:seed to get a user with email samvera@example.com / testing123

Hint: by default Devise will not respond to json. Add these two lines in ApplicationController

```
 protect_from_forgery unless: -> { request.format.json? }

  respond_to :json
```

Let’s Put it All Together

Frontends

ReactJS / ViewJS: fetch or axios

Angular: $http

Ember: ember-fetch

jQuery: $.ajax / $.get / $.post

JS Widget Challenge

JS Widget Challenge

1) Create a JS widget, in whatever JS framework you want that connects to an unauthenticated API and displays a work
2) Create a JS widget that displays a cover flow of work images
3) Create authentication in your JS framework of choice and see if you can hook up JWT tokens

https://www.robinwieruch.de/react-fetching-data/

Questions?

Rob Kaufman
@orangewolf
rob@notch8.com
https://www.notch8.com

This Deck
http://bit.ly/n8sc2018-1

4) Samvera as an API

Description: This workshop will discuss tools and capacities for building external interfaces (React apps, mobile apps) that use Samvera as an API and data source. We recommend some familiarity with developing Samvera or Rails applications as a pre-req to this workshop. We'll go over API design, look at authentication gotchas and walk away with a solid understanding of what it would take to build external tools that connect to your Samvera instance.

Presenter: Rob Kaufman & George Wheeler

Audience: Developers

Equipment: Personal Laptop (development environment instructions)

Room: **1009**

Quiz - design and versions

Rswag challenge

Token Auth Challenge

JS Widget Challenge

What is an api

What can we do with apis
- Discussion

Designing a good api
- Usage First
- Data Model First

Versioning API calls
- Why Version
- In URL
- In Header

Documenting APIs
- Swagger
- Rspec
- rswag
- Exercise swagger based API

Authentication
- What is JWT
- devise_jwt
- Dealing with complex auth
- Exercise

Connecting React to our API
- What is React
- Exercise

~~Connecting React Native to our API~~