Journey building a Twitter bot with Serverless on AWS

about

```
def about

•Rails Software Engineer 🛤️👨‍💻
•@ghosteathuman

end
```

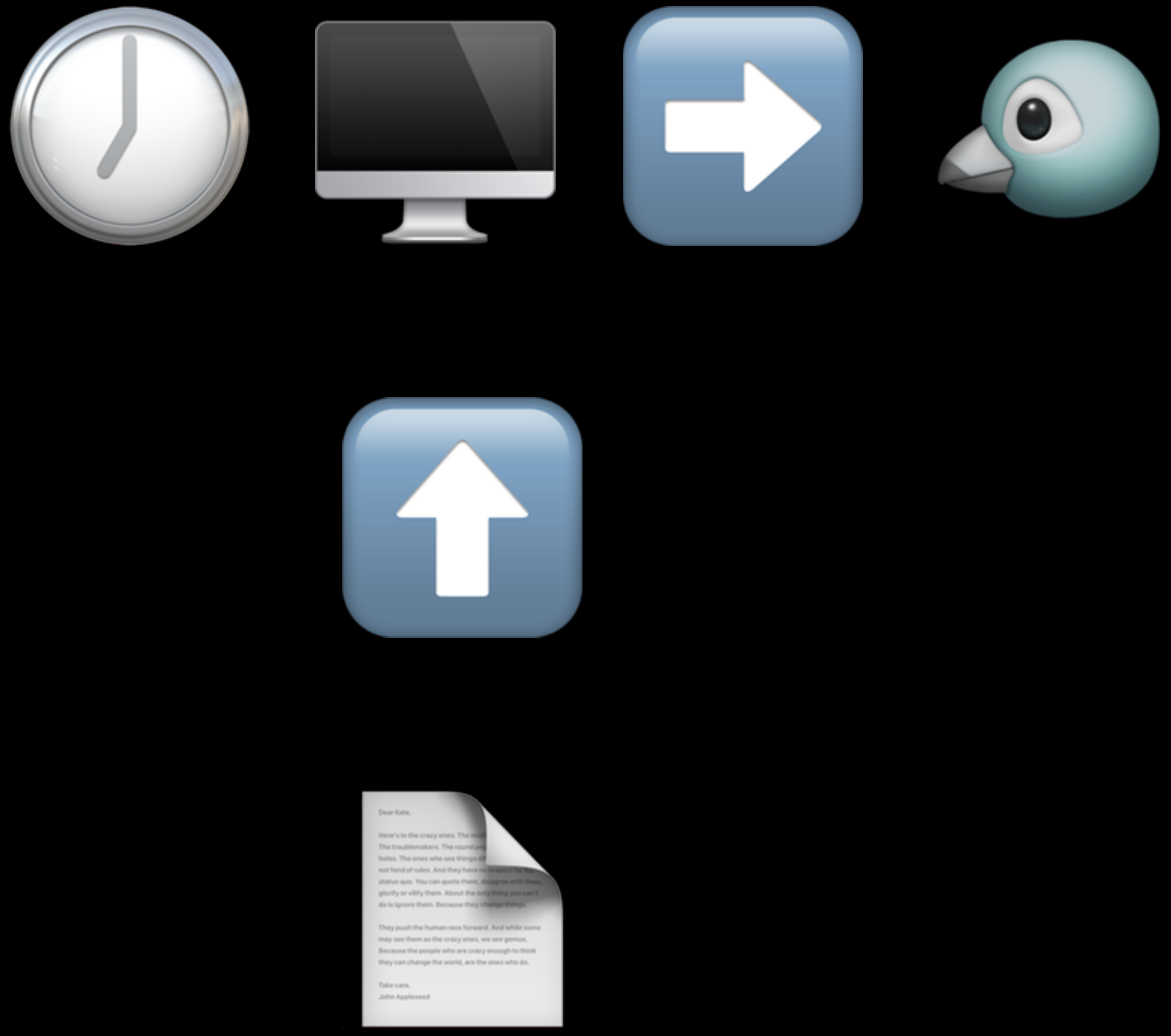
journey

- Twitter bot breakdown
- Trying Ruby On Jets
- Trying Serverless Framework
- ~~Comparison & tutorial of frameworks~~

disclaimer

Your Mileage May Vary (YMMV)

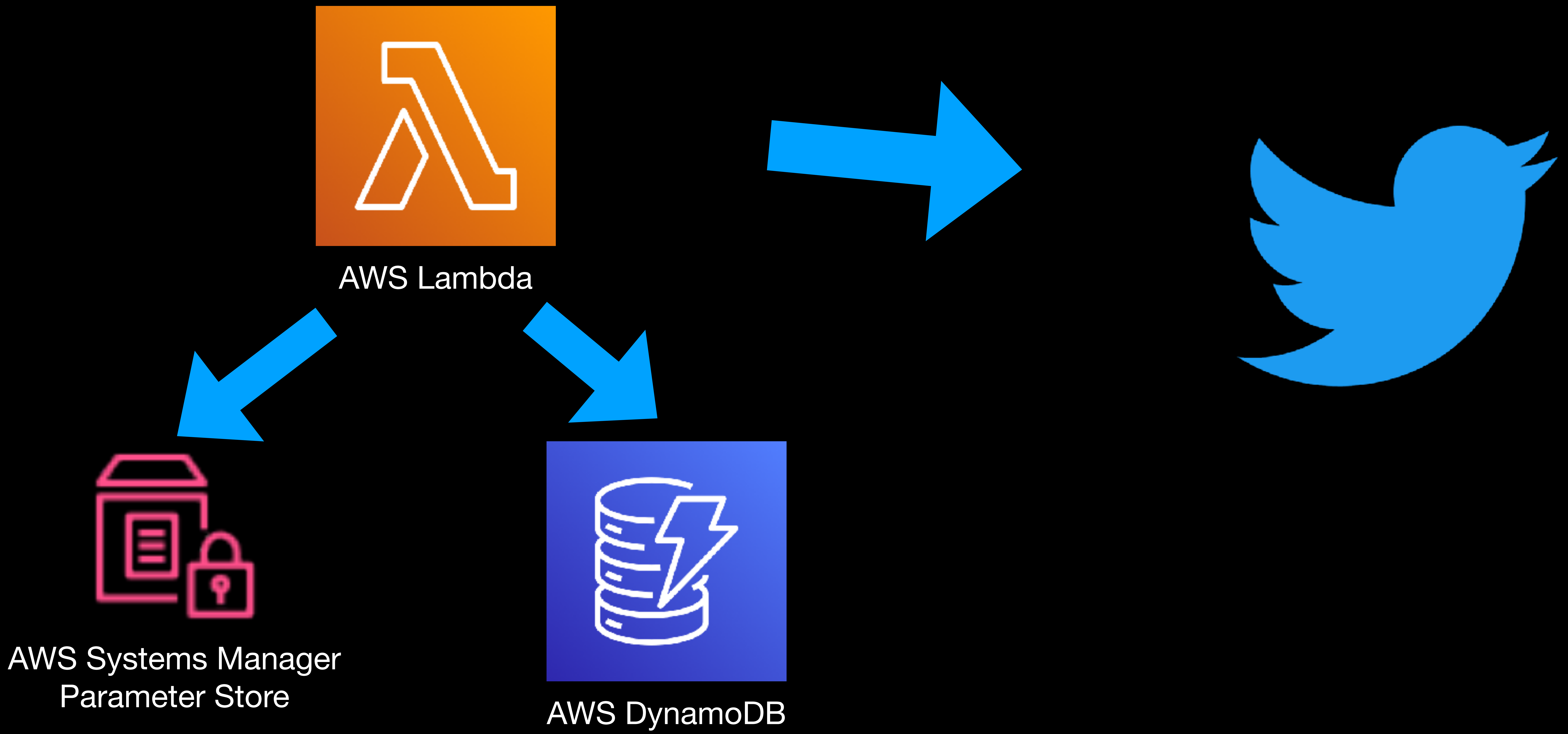
twitter bot breakdown



why serverless on aws

- Fits use case
    - Compute & Storage
- Free
    - AWS Lambda & DynamoDB has a always free tier
- Why not

serverless breakdown



AWS Lambda

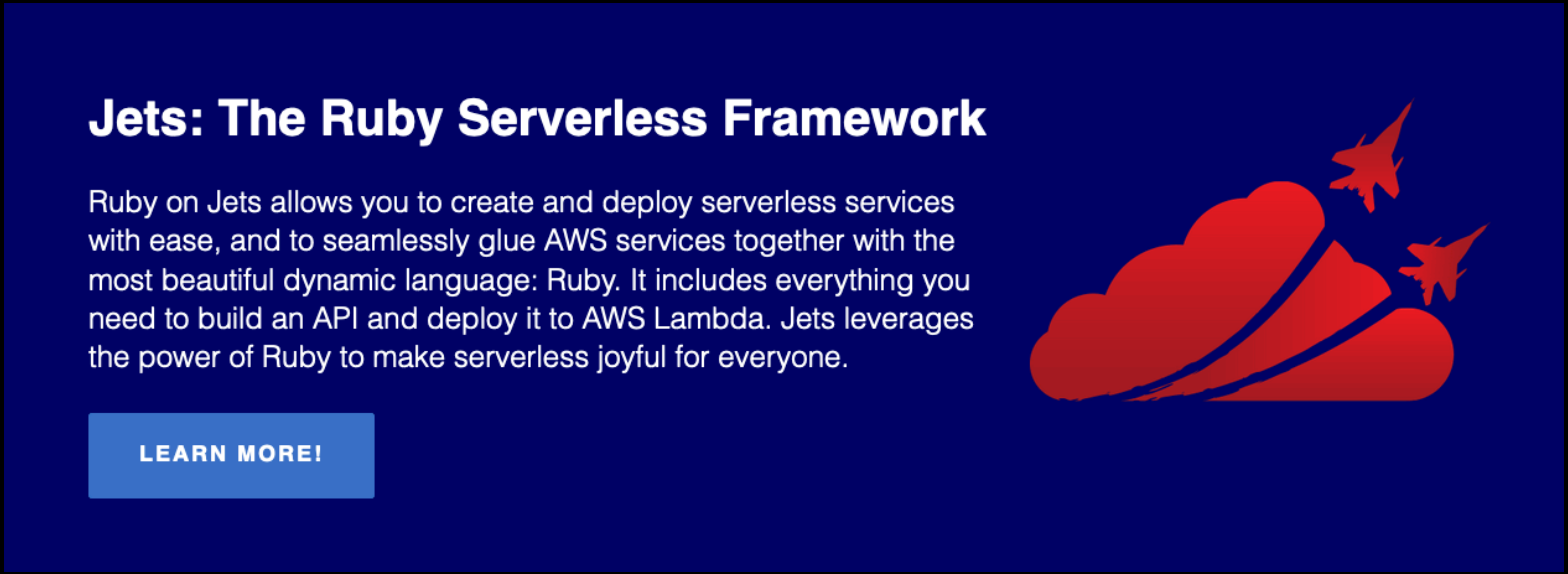
AWS Systems Manager
Parameter Store

AWS DynamoDB

lambda layers

- a .zip file archive that can contain additional code or data
- A layer can contain libraries, a custom runtime, data, or configuration files.

trying ruby on jets

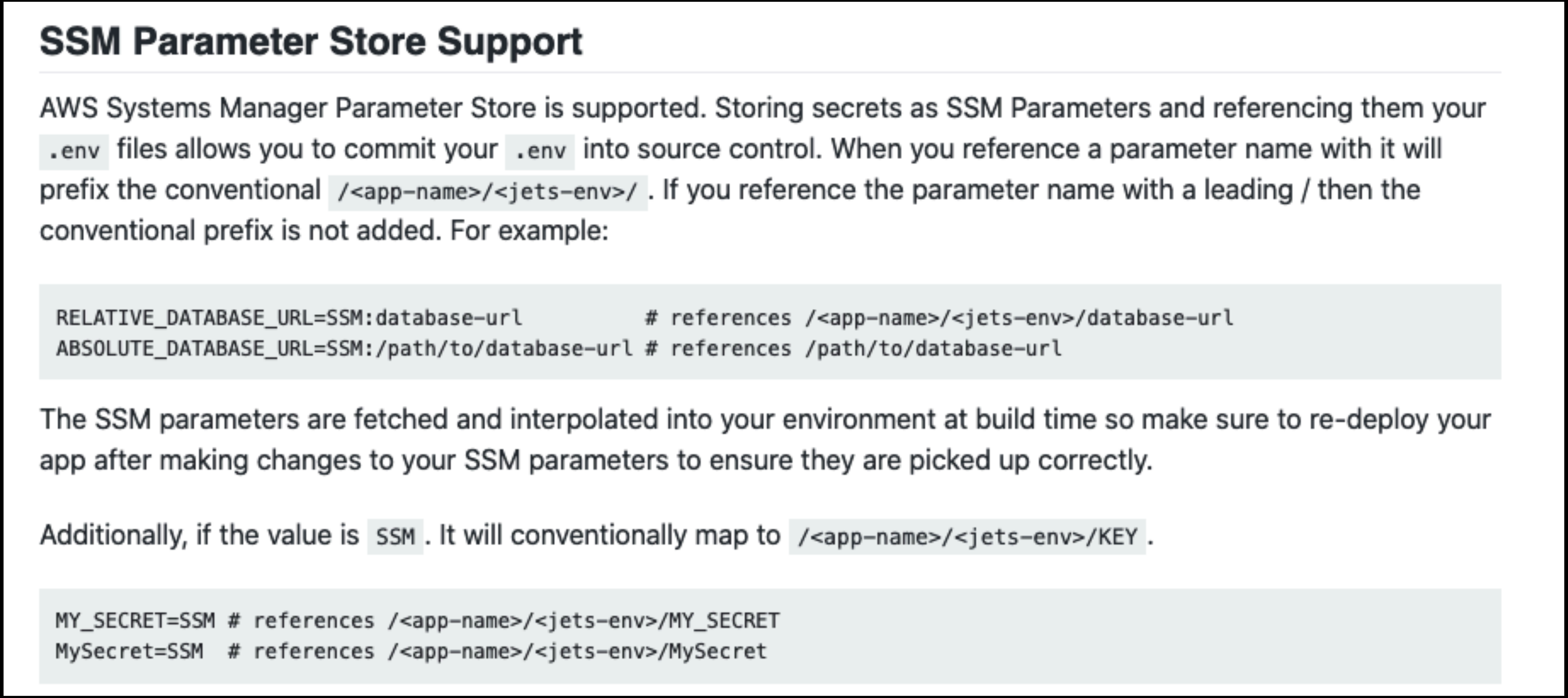


Jets: The Ruby Serverless Framework

Ruby on Jets allows you to create and deploy serverless services with ease, and to seamlessly glue AWS services together with the most beautiful dynamic language: Ruby. It includes everything you need to build an API and deploy it to AWS Lambda. Jets leverages the power of Ruby to make serverless joyful for everyone.

https://rubyonjets.com/

ruby on jets



SSM Parameter Store Support

AWS Systems Manager Parameter Store is supported. Storing secrets as SSM Parameters and referencing them your `.env` files allows you to commit your `.env` into source control. When you reference a parameter name with it will prefix the conventional `/<app-name>/<jets-env>/`. If you reference the parameter name with a leading / then the conventional prefix is not added. For example:

```
RELATIVE_DATABASE_URL=SSM:database-url          # references /<app-name>/<jets-env>/database-url
ABSOLUTE_DATABASE_URL=SSM:/path/to/database-url # references /path/to/database-url
```

The SSM parameters are fetched and interpolated into your environment at build time so make sure to re-deploy your app after making changes to your SSM parameters to ensure they are picked up correctly.

Additionally, if the value is `SSM`. It will conventionally map to `/<app-name>/<jets-env>/KEY`.

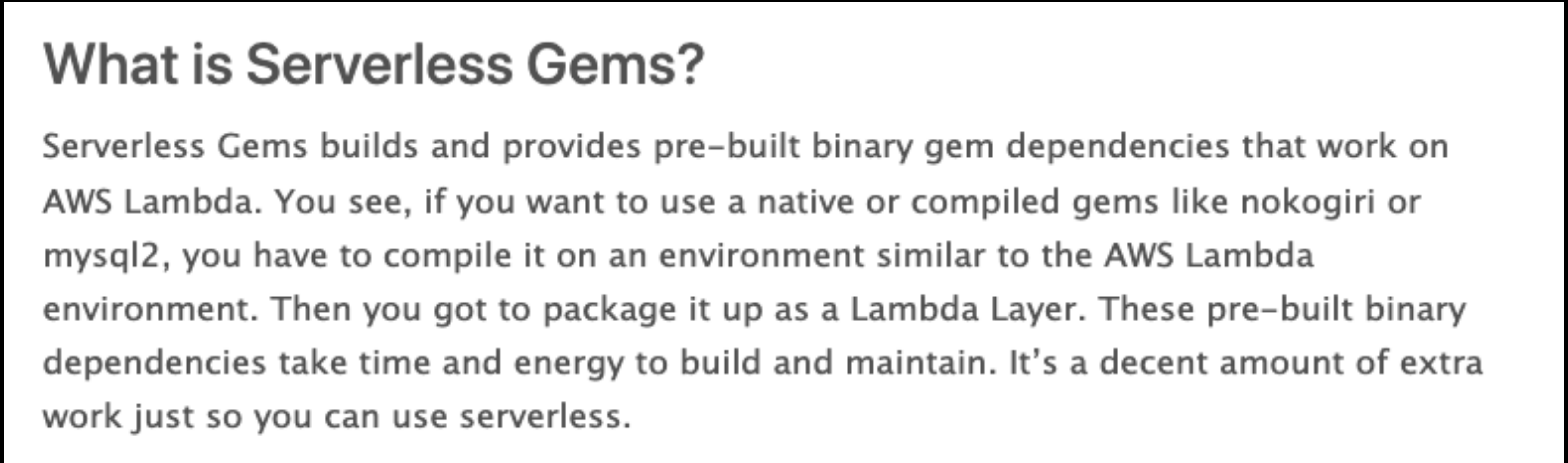
```
MY_SECRET=SSM # references /<app-name>/<jets-env>/MY_SECRET
MySecret=SSM  # references /<app-name>/<jets-env>/MySecret
```

ruby on jets

- Uses ‘dynomite’ gem by default for DynamoDB
- Kept getting ‘Aws::DynamoDB::Errors::ResourceNotFoundException: Requested resource not found’ error
- For testing purposes, switch to ‘Dynamoid’ gem and worked after additional configuration.

ruby on jets

- ‘ERROR: Limit Reached You have reached your daily anonymous download limit. You can increase the limit by registering: https://www.serverlessgems.com/rate-limits'

- 

What is Serverless Gems?

Serverless Gems builds and provides pre-built binary gem dependencies that work on AWS Lambda. You see, if you want to use a native or compiled gems like nokogiri or mysql2, you have to compile it on an environment similar to the AWS Lambda environment. Then you got to package it up as a Lambda Layer. These pre-built binary dependencies take time and energy to build and maintain. It’s a decent amount of extra work just so you can use serverless.

serverless framework

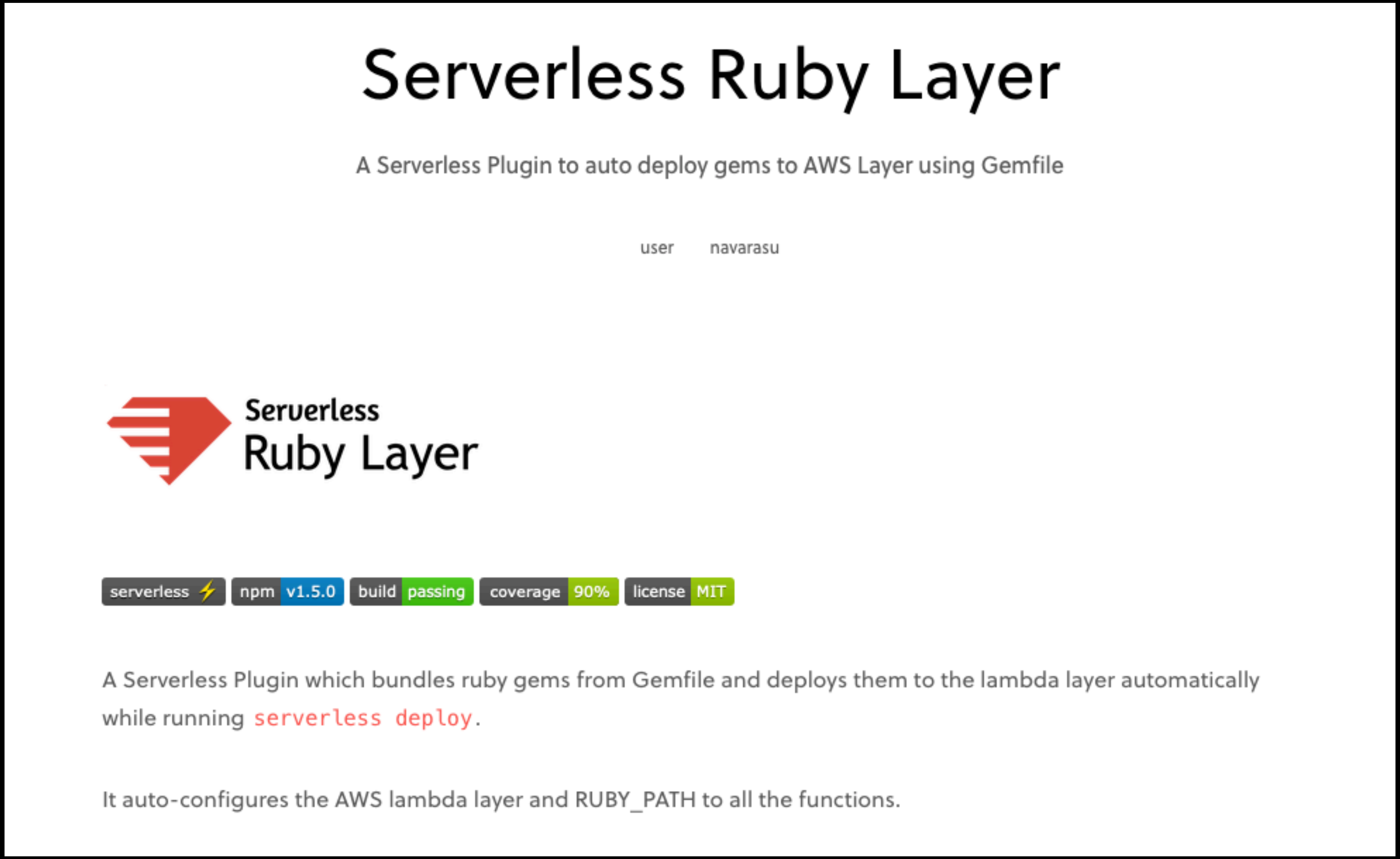
serverless framework

zero-friction serverless development

easily build apps that auto-scale on low cost, next-gen cloud infrastructure.

https://www.serverless.com/

serverless framework



Serverless Ruby Layer

A Serverless Plugin to auto deploy gems to AWS Layer using Gemfile

user navarasu

Serverless
Ruby Layer

serverless ⚡ | npm v1.5.0 | build passing | coverage 90% | license MIT

A Serverless Plugin which bundles ruby gems from Gemfile and deploys them to the lambda layer automatically while running `serverless deploy`.

It auto-configures the AWS lambda layer and RUBY_PATH to all the functions.

references

- Ruby on Jets: https://rubyonjets.com/
- Introducing Serverless Gems: https://blog.boltops.com/2021/01/07/introducing-serverless-gems
- Serverless framework: https://www.serverless.com/
- Serverless Ruby Layer: https://www.serverless.com/plugins/serverless-ruby-layer

thanks!