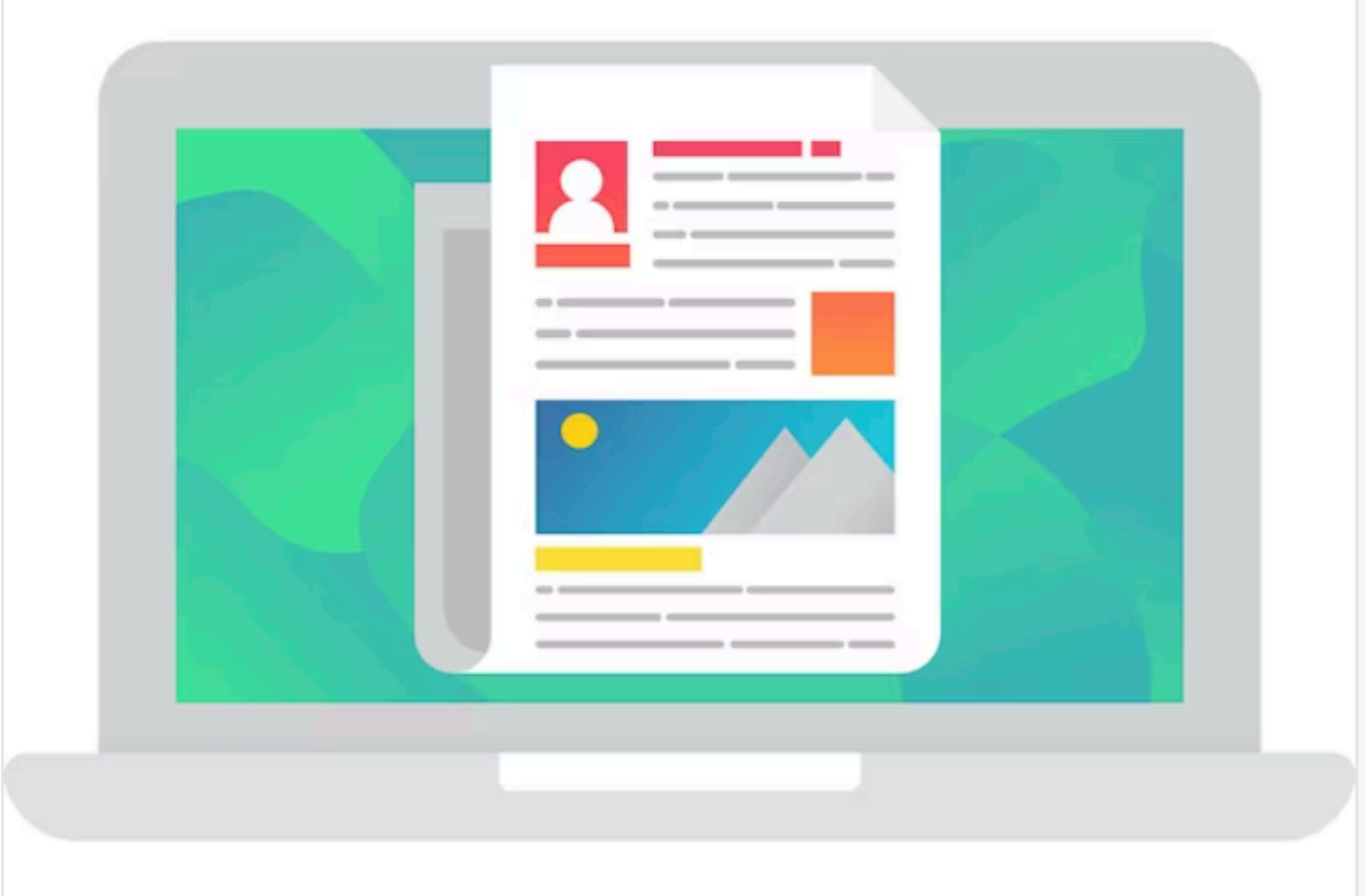
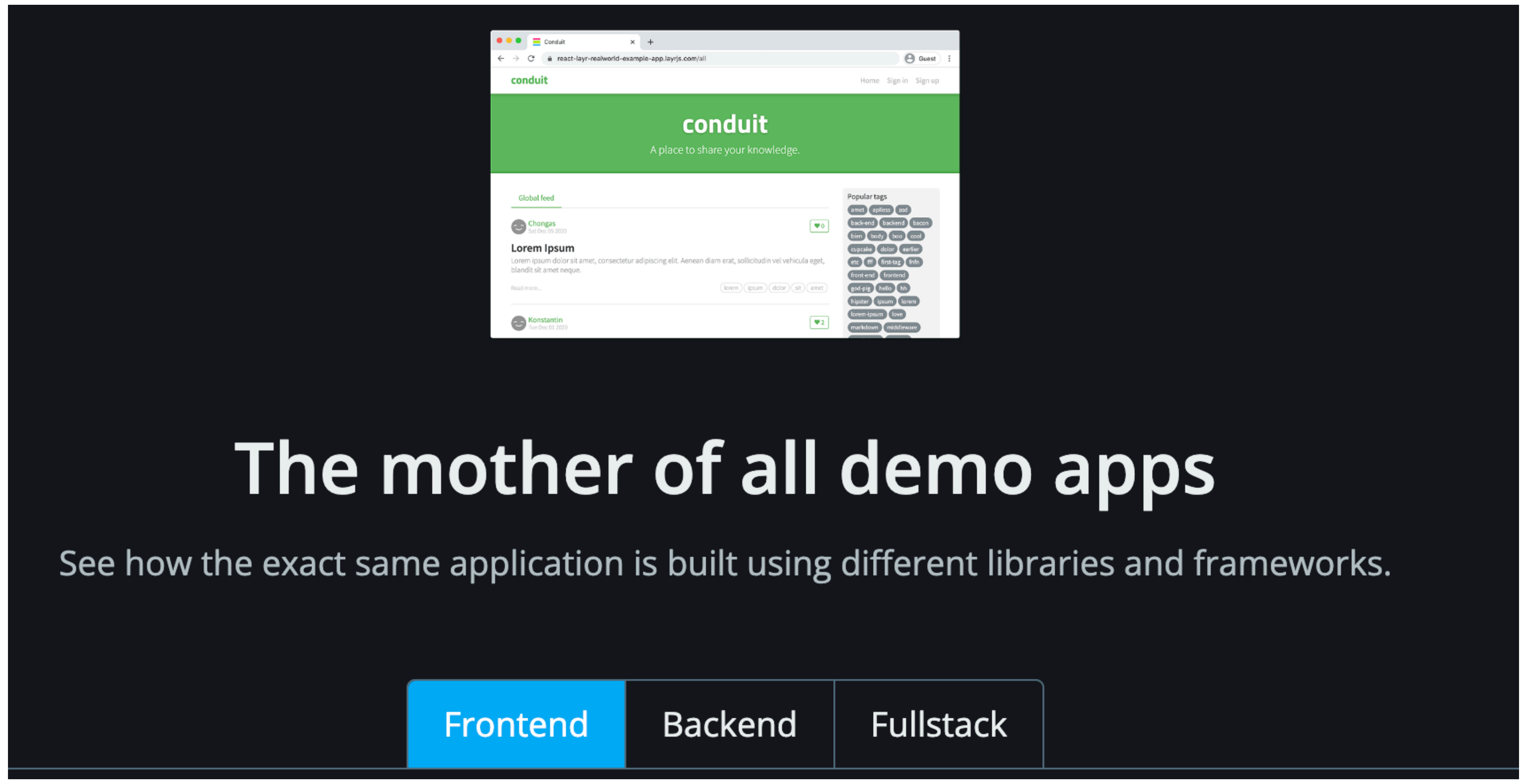
Assignment 1 Specification

Concept, detail and grading spectrum for Assignment 1

Assignment Motivation & Philosophy

- Assignments present multiple opportunities:
  - Crystallise knowledge on relevant module(s)
  - Opportunity for creativity & innovation
  - Potentially important portfolio contribution
- Its not over when submitted, presented and graded
  - Ongoing maintenance & enhancements
  - Periodically rebuild using new technology stack



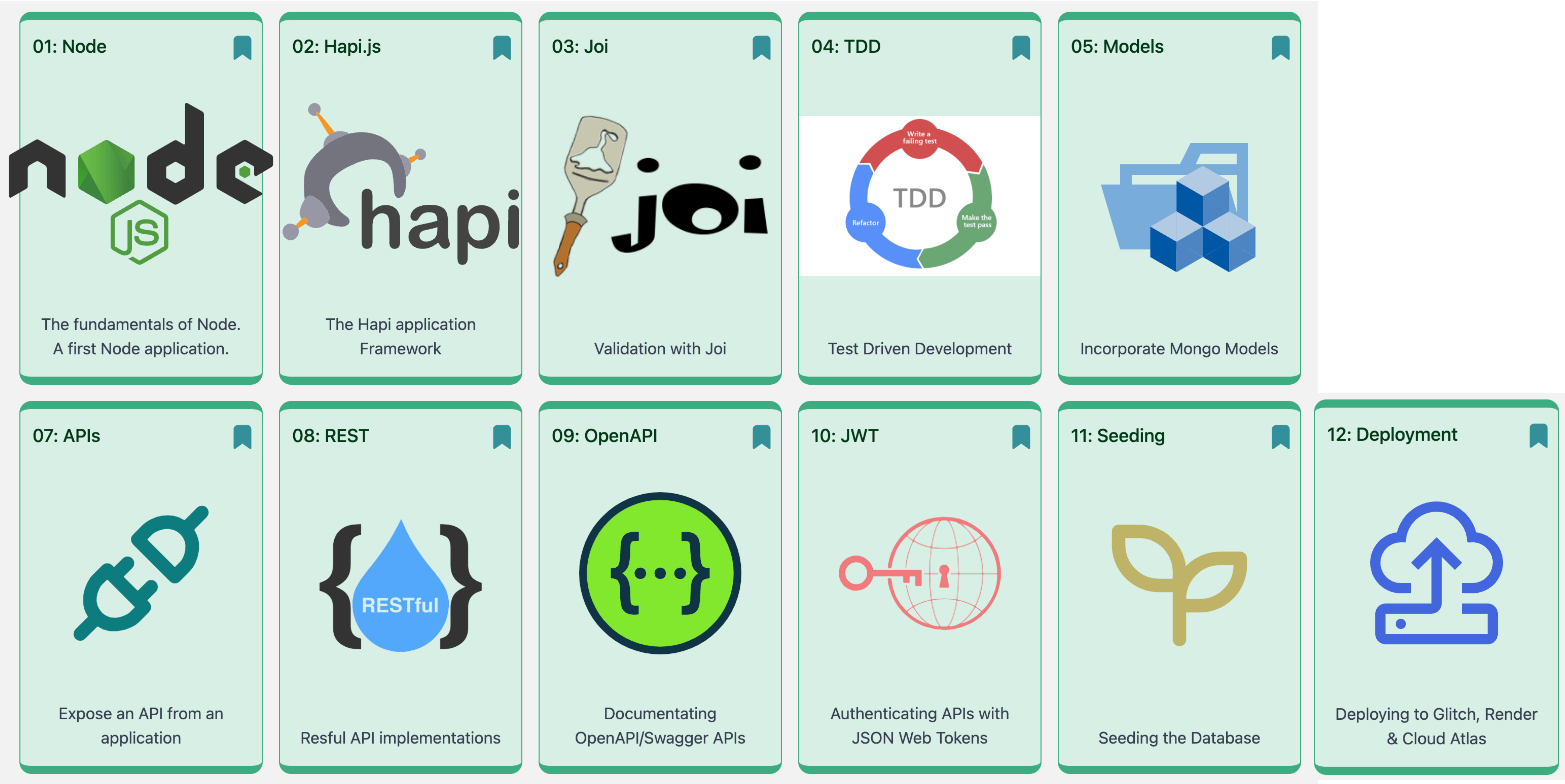


The mother of all demo apps

See how the exact same application is built using different libraries and frameworks.

https://github.com/gothinkster/realworld
https://codebase.show/projects/realworld

Assignment 1 Topics



01: Node
The fundamentals of Node. A first Node application.

02: Hapi.js
The Hapi application Framework

03: Joi
Validation with Joi

04: TDD
Test Driven Development

05: Models
Incorporate Mongo Models

07: APIs
Expose an API from an application

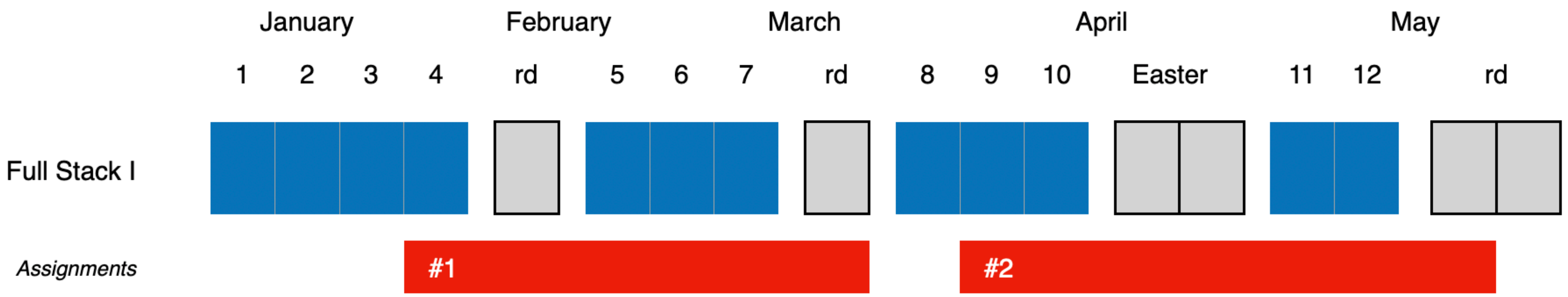
08: REST
Resful API implementations

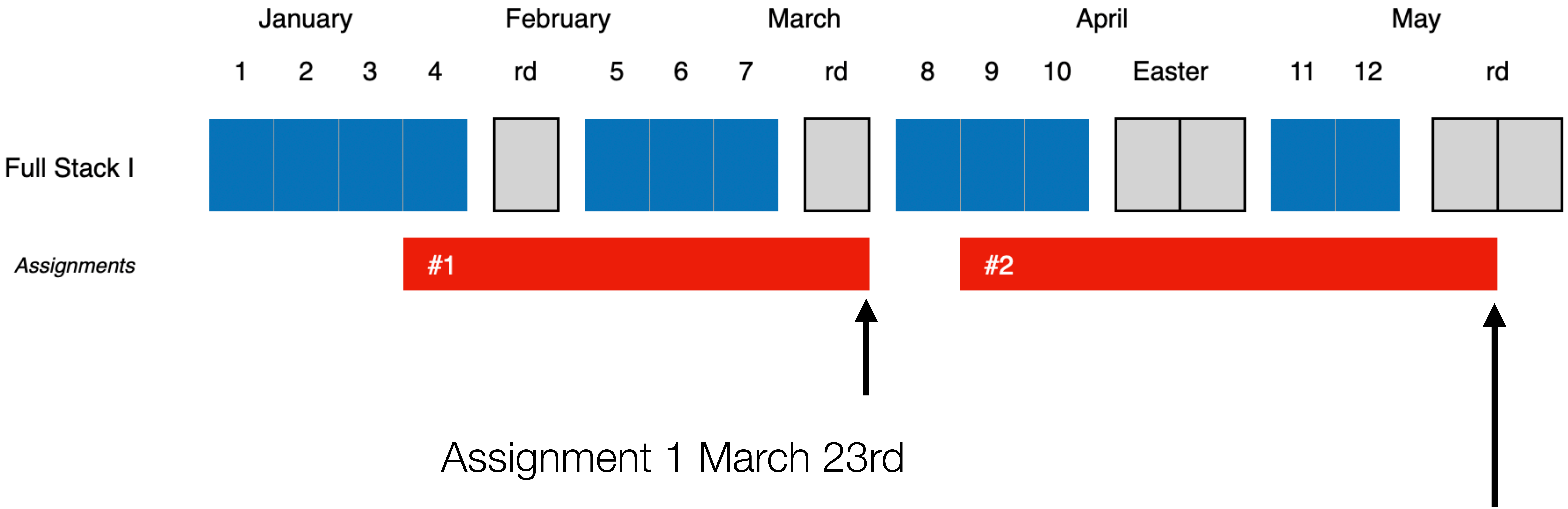
09: OpenAPI
Documentating OpenAPI/Swagger APIs

10: JWT
Authenticating APIs with JSON Web Tokens

11: Seeding
Seeding the Database

12: Deployment
Deploying to Glitch, Render & Cloud Atlas





January February March April May

1 2 3 4 rd 5 6 7 rd 8 9 10 Easter 11 12 rd

Full Stack I

Assignments

#1

#2

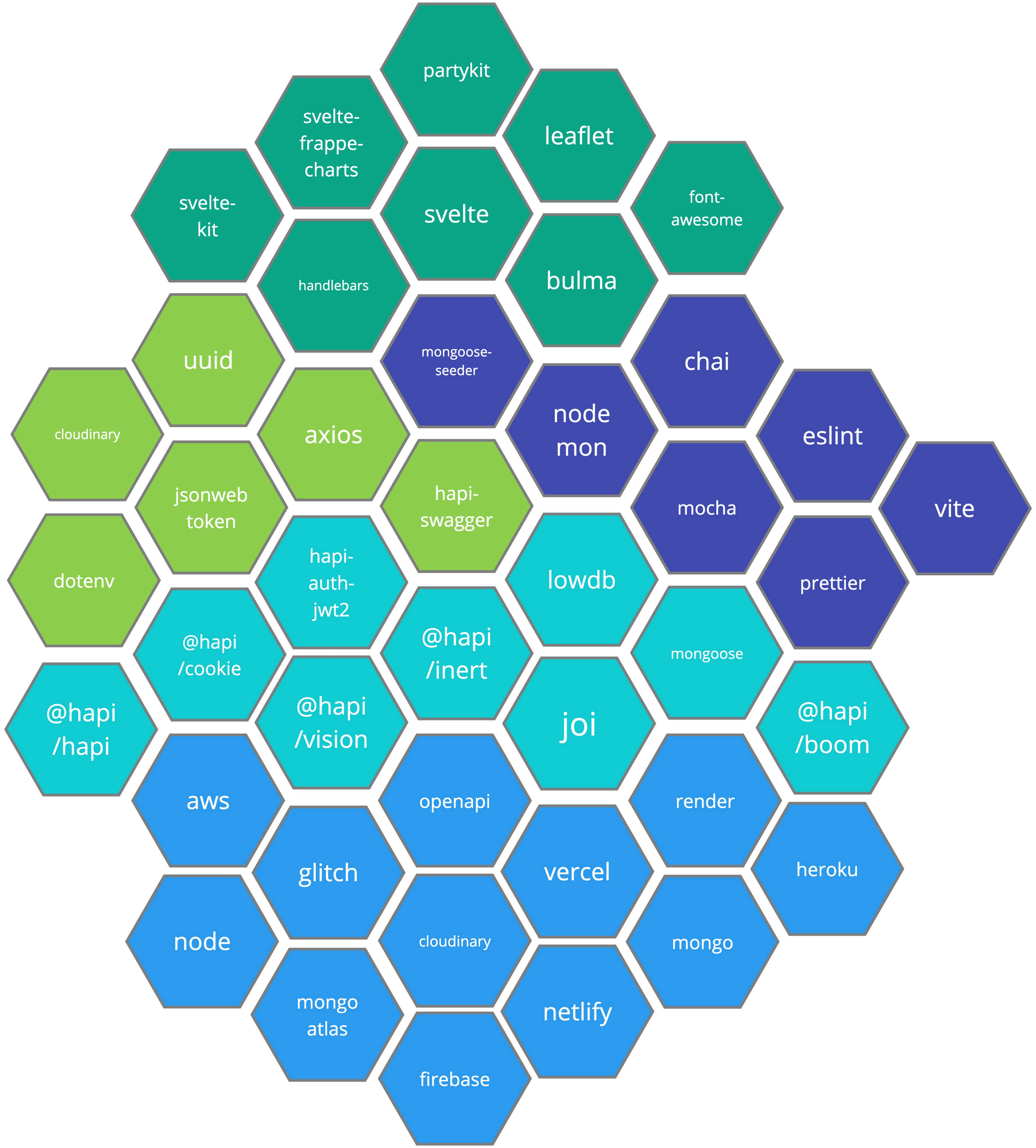
Assignment 1 March 23rd

50%

Assignment 2 May 18th

50%

Full Stack Components & Services



partykit

svelte-frappe-charts

leaflet

svelte-kit

svelte

font-awesome

handlebars

bulma

uuid

mongoose-seeder

chai

cloudinary

axios

node mon

eslint

jsonweb token

hapi-swagger

mocha

vite

dotenv

hapi-auth-jwt2

lowdb

prettier

@hapi /cookie

@hapi /inert

mongoose

@hapi /hapi

@hapi /vision

joi

@hapi /boom

aws

openapi

render

glitch

vercel

heroku

node

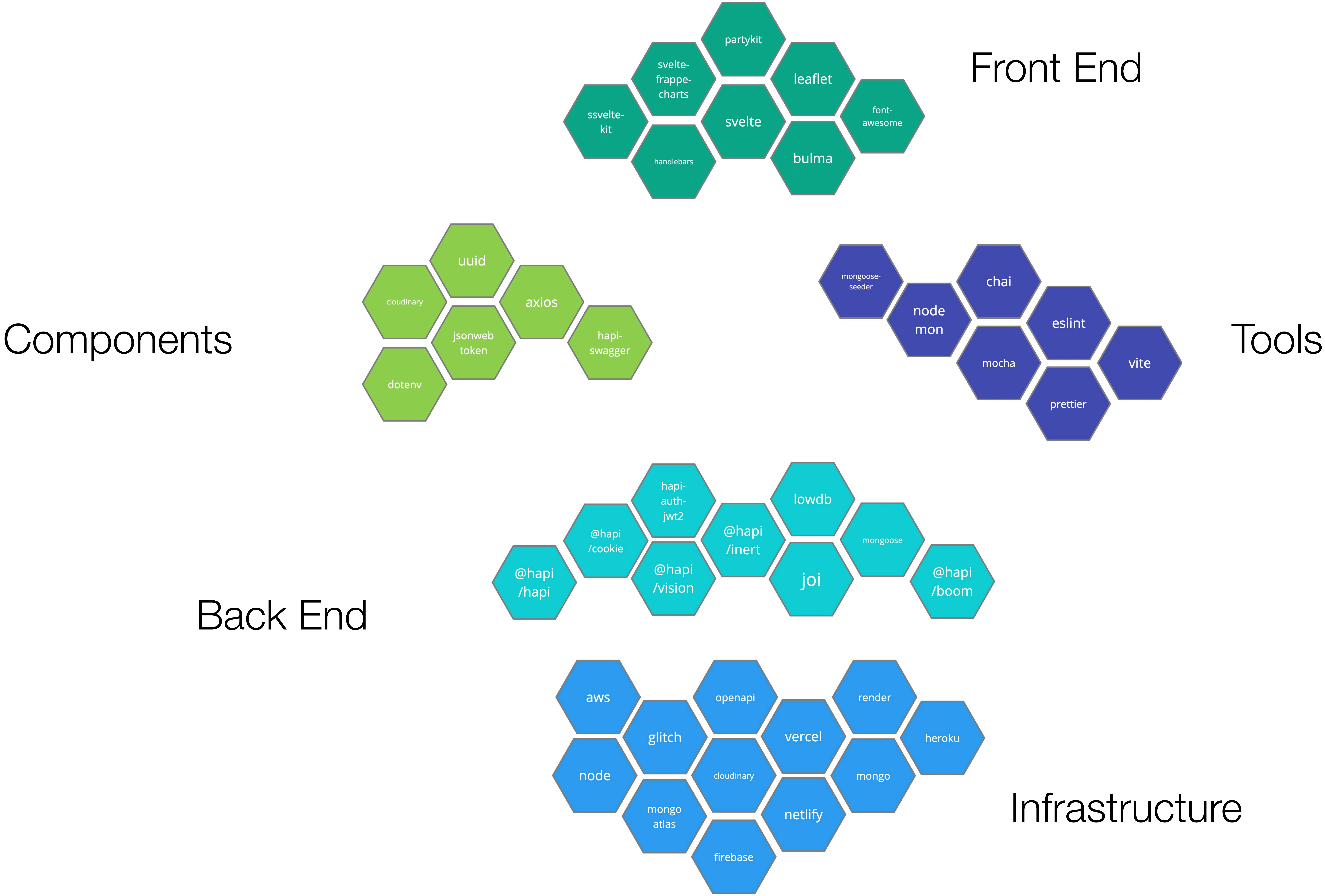
cloudinary

mongo

mongo atlas

netlify

firebase



Front End

partykit
svelte-frappe-charts
leaflet
ssvelte-kit
svelte
font-awesome
handlebars
bulma

Components

uuid
cloudinary
axios
jsonweb token
hapi-swagger
dotenv

Tools

mongoose-seeder
chai
node mon
eslint
mocha
vite
prettier

Back End

hapi-auth-jwt2
lowdb
@hapi /cookie
@hapi /inert
mongoose
@hapi /hapi
@hapi /vision
joi
@hapi /boom

Infrastructure

aws
openapi
render
glitch
vercel
heroku
node
cloudinary
mongo
mongo atlas
netlify
firebase

PlaceMark

A point of interest, or POI, is a specific point location that someone may find useful or interesting.

> *"A "point of interest" (POI) is a location for which information is available. A POI can be as simple as a set of coordinates, a name, and a unique identifier, or more complex such as a three-dimensional model of a building with names in multiple languages, information about opening and closing hours, and a civic address. POI data has many applications, including augmented reality browsers, location-based social networking games, geocaching, mapping and navigation systems."*
>
> http://www.opengeospatial.org/projects/groups/poiswg

https://en.wikipedia.org/wiki/Point_of_interest

https://en.wikipedia.org/wiki/Point_of_interest

PlaceMark Examples

- Landscape feature
- National monument
- Walking Trail
- Bridge
- Tree
- Venue
- Ringfort
- Dolmen
- River
- Bog
- Island
- Forest
- Beach
- Museum
- Climb
- Cycleway
- Birdwatching Locations
- Diving

Product Feature Sets

POI Sharing /
Community Features

Community

- You are free to innovate an the type, range and variety of features

Richer Media
Experience + User
Management

Enhanced

- These categories are to stimulate your brainstorming

Essential Data
Structure & Support
Service

Core

- Some ideas on the next slides…

Information Model

- Placemark
  - Name
  - Category
  - Description
  - Analytics
- User
  - Basic user details

Features

- Sign up / log in to service
- Create, Read, Update, Delete
- Query
- Group into Categories

Core

Information Model

- Placemark
  - Images
  - Location
  - Weather
- User
  - Types

Features

- Placemark images
- Placemark Map Location
- Admin
  - Admin user
  - Admin dashboard

Enhanced

Information Model

- Placemark
  - Creator / Editor
  - Rating
  - Reviews
  - Discussion
- User
  - Interaction information

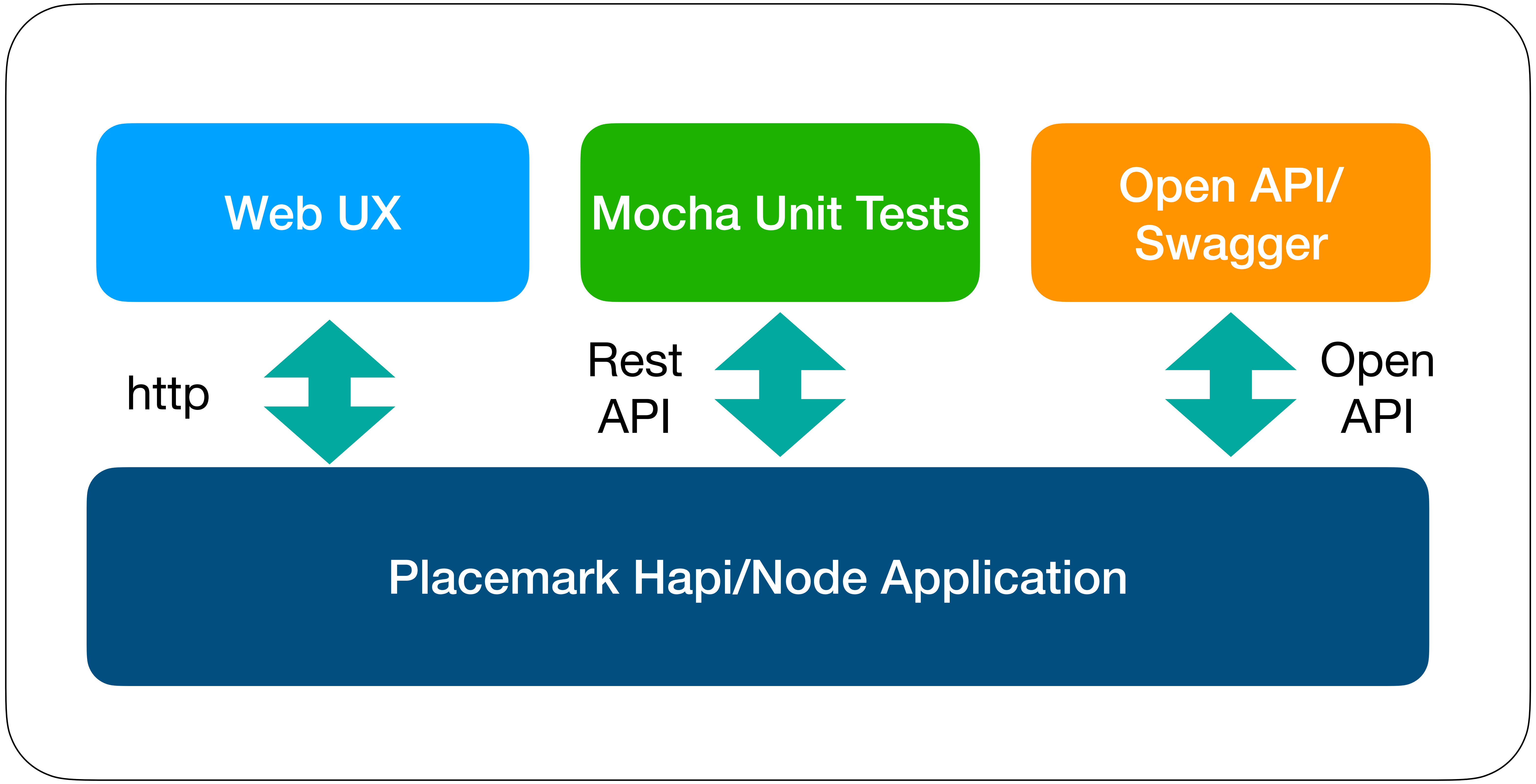
Features

- Share
- Review & Rate
- Notice Board
- Live Updates
- Analytics

Community

PlaceMark 1.0 Architecture

(Assignment 1)



Web UX

Mocha Unit Tests

Open API/ Swagger

http

Rest API

Open API

Placemark Hapi/Node Application

Placemark 1.0 Feature Set

Enhanced

Core

Implement POI Core + selected features of POI Enhanced

Support the following User Management Features:

- Sign up / Login in / Delete Account
- User Admin Dashboard with simple analytics

Support the following POI Characteristics:

- Name, Description, Category, Location, Image

Support the following app features:

- Create, Read, Update & Delete POIs
- Organise PlaceMarks into categories

API

- Endpoints + tests
- Swagger Documentation
- JWT Security

	Accounts	*Placemark Features*	*API/Tests*	*Models*	*Deployment*	*Git*
Level 1	Signup / Login	Name	Basic API	Mem	Locahost	git + commit History
Level 2	Cookie Authentication/ User Settings	Description, Location (lat,lng)	Core unit Tests	JSON	Glitch	Structured readme
Level 3	Basic Admin Account List/Remove Users	Categories	Open API (Swagger) + tests	Mongo	Cloud Atlas	Tagged releases
Level 4	Admin Dashboard + analytics	Images	JWT + tests	Firebase	Heroku / Render	Development / Feature / Master branch

Getting Started

- Identify a set of suitable Point of Interest concept for your app. Consider how they would be created, grouped, shared, Consider the value of the service you are inventing. To whom would it be of interest? What would they do with it?

- Once you have determined the Poi concept, compose a small set of example data you will use for testing

- Sketch out a web interface - using pencil & paper of a perhaps even a simple wireframe too like Excallidraw

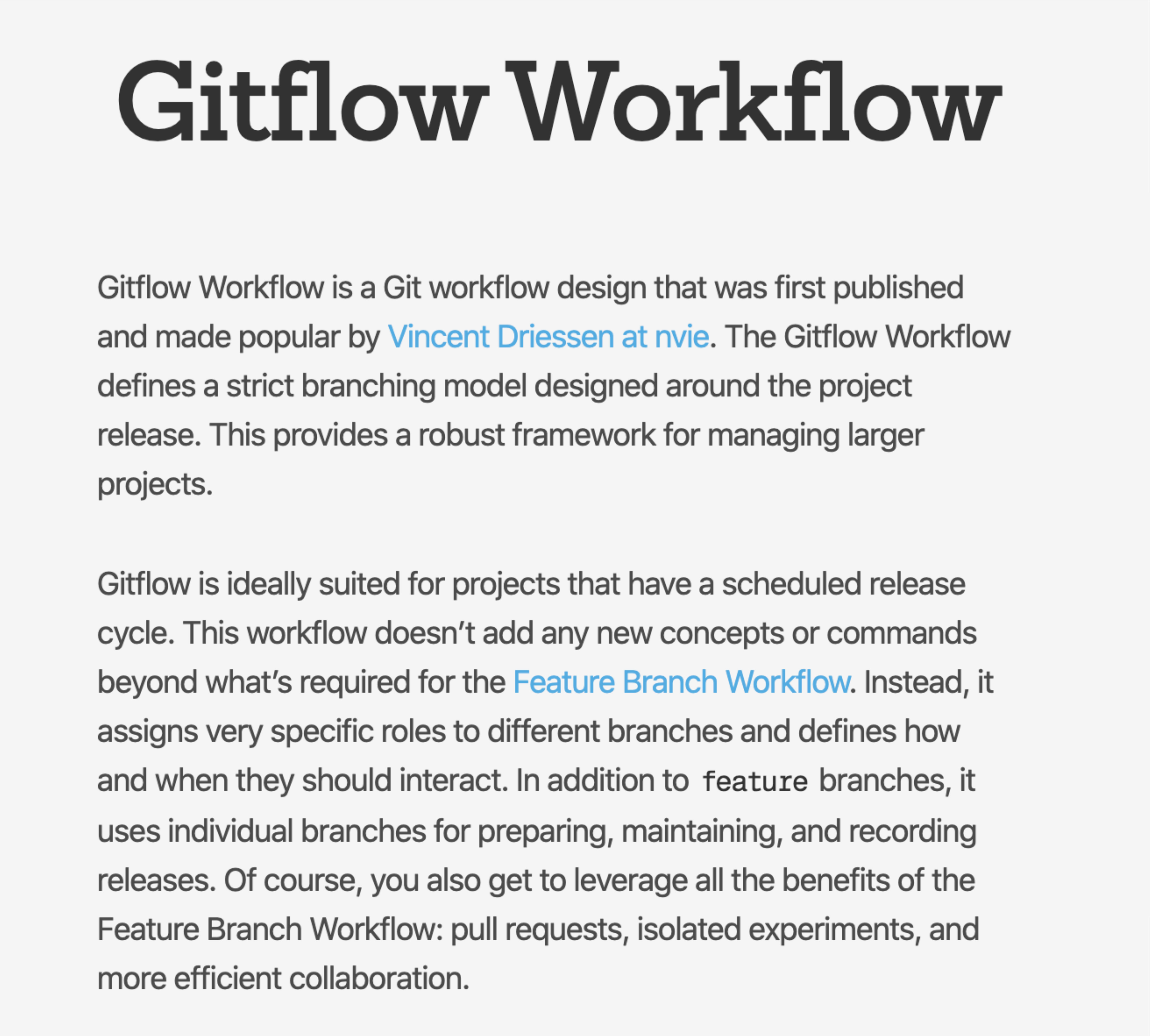
- Revisit the Playtime labs - perhaps consider using the latest completed lab as a starter project.

- The User model can probably be adopted as is, however you will need to implement a new model to match your PoI concept.

- Get the user system working + some UX for creating and viewing your PoI.

- Grouping / Categorising PoIs: What would a group represent? What would the UX for this look like? Prototype an approach.

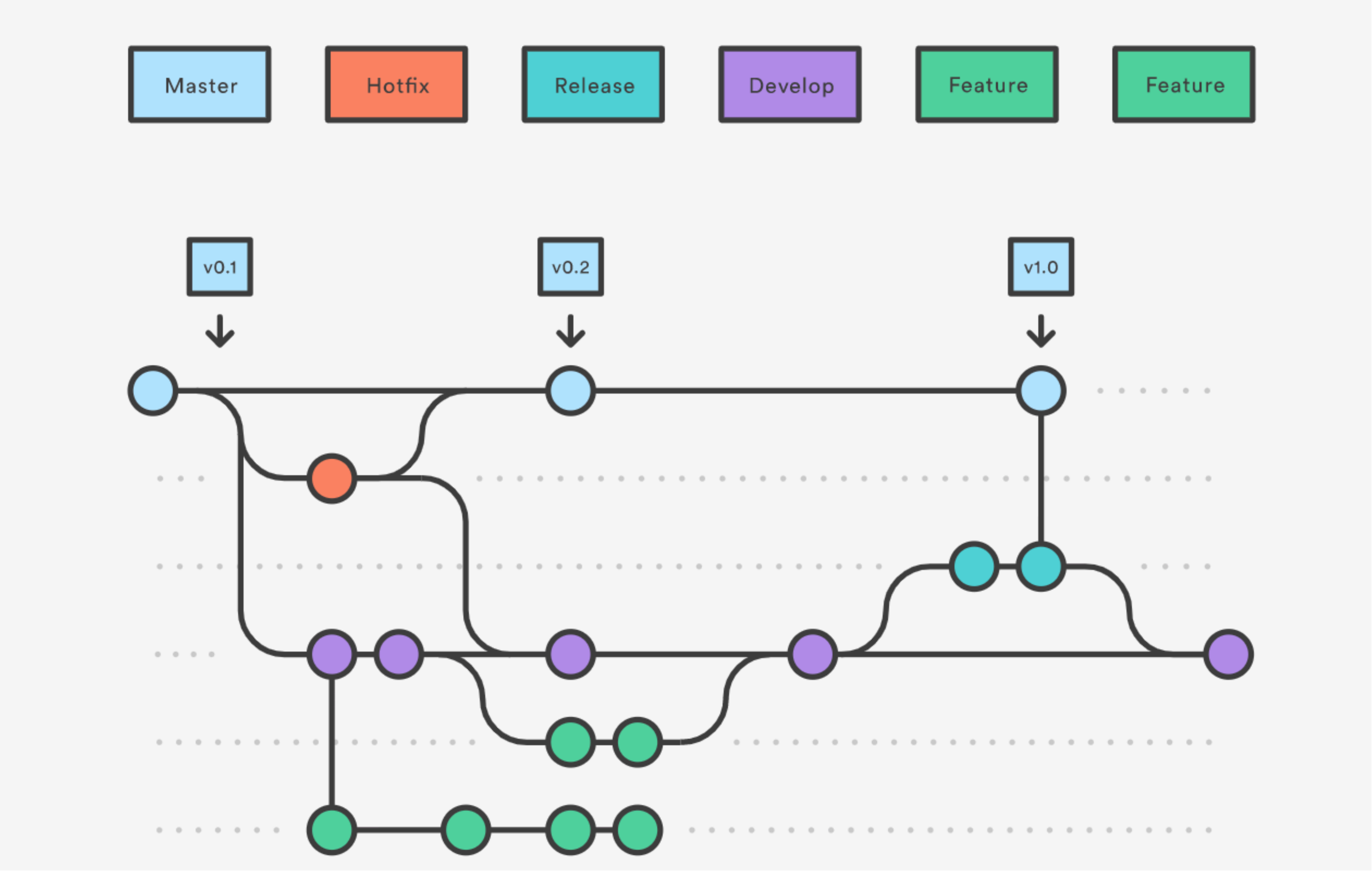
- At this stage, you might have a sufficient set of ideas/concepts/tasks in play to consider organising your work in some way. This could be a simple whiteboard - or you might like something more structured. Github Projects is worth evaluating. It can be used without necessarily using advanced git features



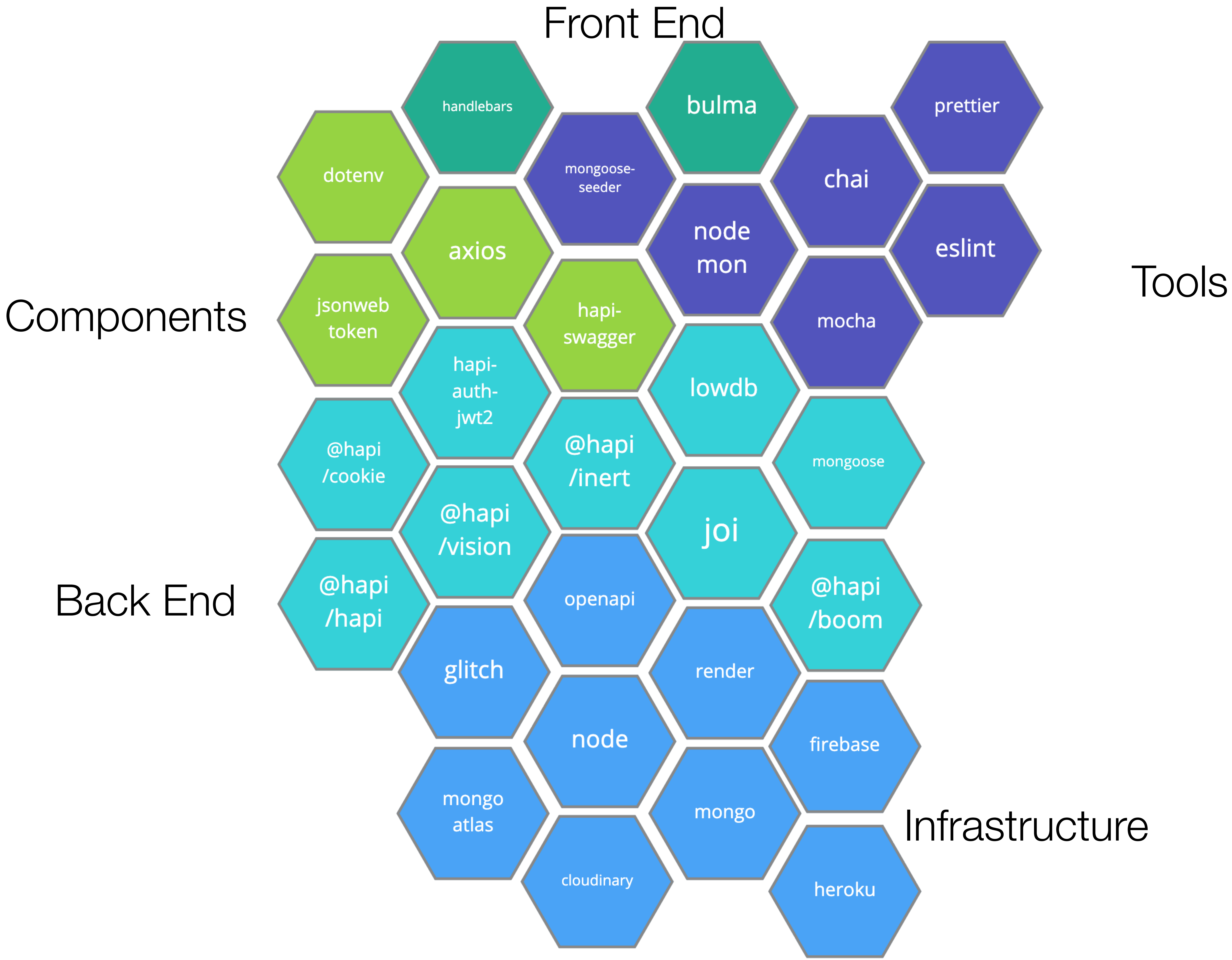
Gitflow Workflow

Gitflow Workflow is a Git workflow design that was first published and made popular by Vincent Driessen at nvie. The Gitflow Workflow defines a strict branching model designed around the project release. This provides a robust framework for managing larger projects.

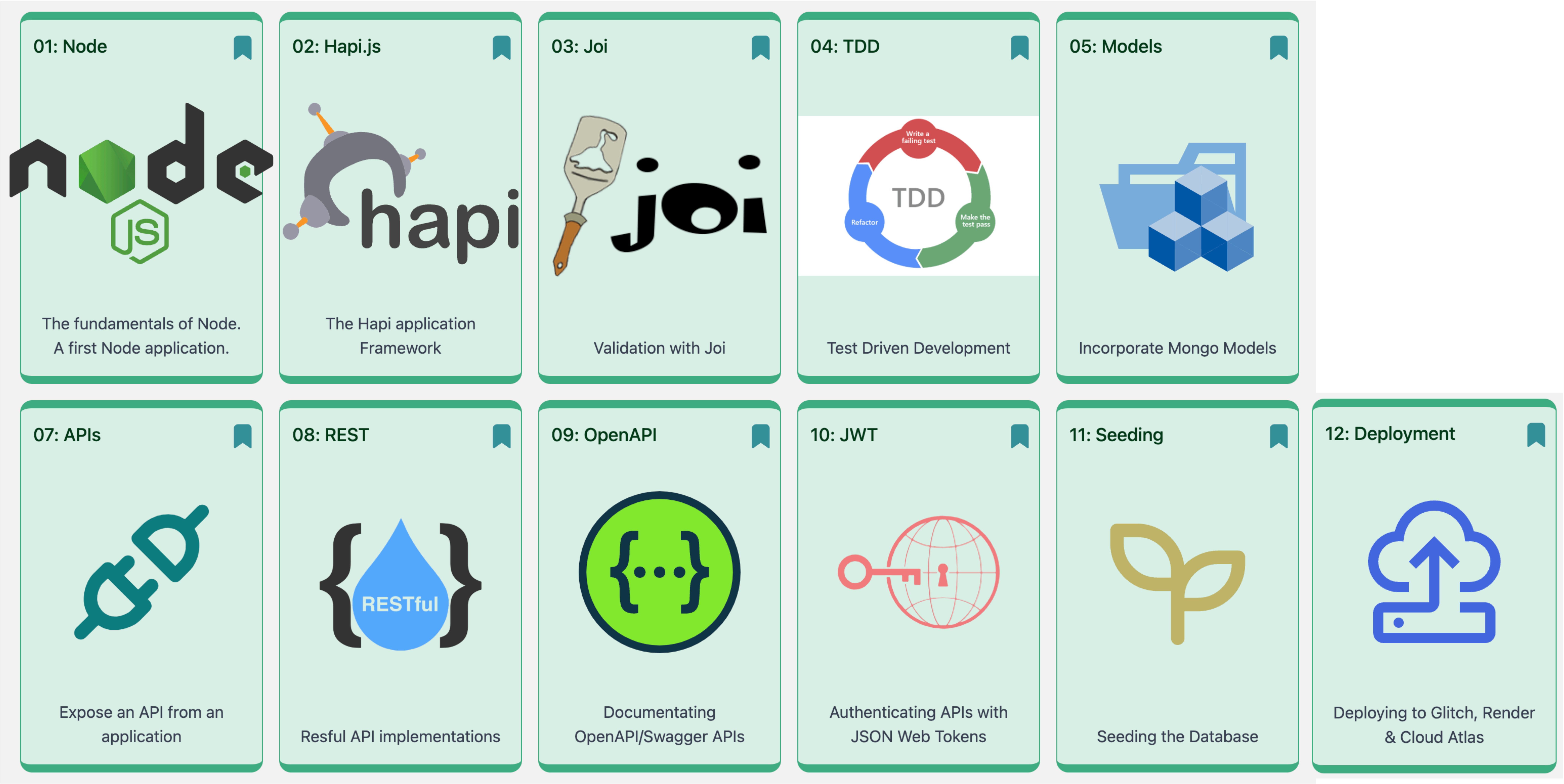
Gitflow is ideally suited for projects that have a scheduled release cycle. This workflow doesn't add any new concepts or commands beyond what's required for the Feature Branch Workflow. Instead, it assigns very specific roles to different branches and defines how and when they should interact. In addition to `feature` branches, it uses individual branches for preparing, maintaining, and recording releases. Of course, you also get to leverage all the benefits of the Feature Branch Workflow: pull requests, isolated experiments, and more efficient collaboration.



https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow


Front End
Components
Tools
Back End
Infrastructure
handlebars
bulma
prettier
dotenv
mongoose-seeder
chai
axios
node mon
eslint
jsonweb token
hapi-swagger
mocha
hapi-auth-jwt2
lowdb
@hapi /cookie
@hapi /inert
mongoose
@hapi /vision
joi
@hapi /hapi
openapi
@hapi /boom
glitch
render
node
firebase
mongo atlas
mongo
cloudinary
heroku

Assignment 1 Topics



01: Node

The fundamentals of Node. A first Node application.

02: Hapi.js

The Hapi application Framework

03: Joi

Validation with Joi

04: TDD

Test Driven Development

05: Models

Incorporate Mongo Models

07: APIs

Expose an API from an application

08: REST

Resful API implementations

09: OpenAPI

Documentating OpenAPI/Swagger APIs

10: JWT

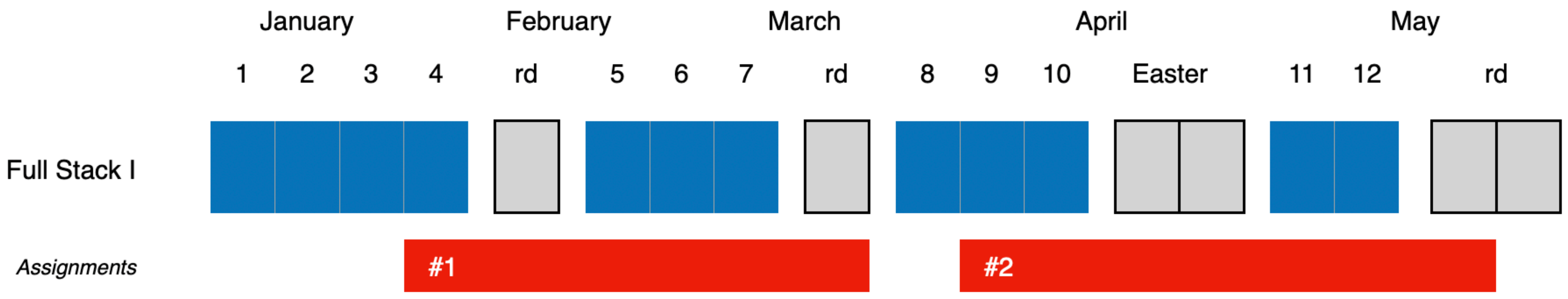
Authenticating APIs with JSON Web Tokens

11: Seeding

Seeding the Database

12: Deployment

Deploying to Glitch, Render & Cloud Atlas



January February March April May

1 2 3 4 rd 5 6 7 rd 8 9 10 Easter 11 12 rd

Full Stack I

Assignments

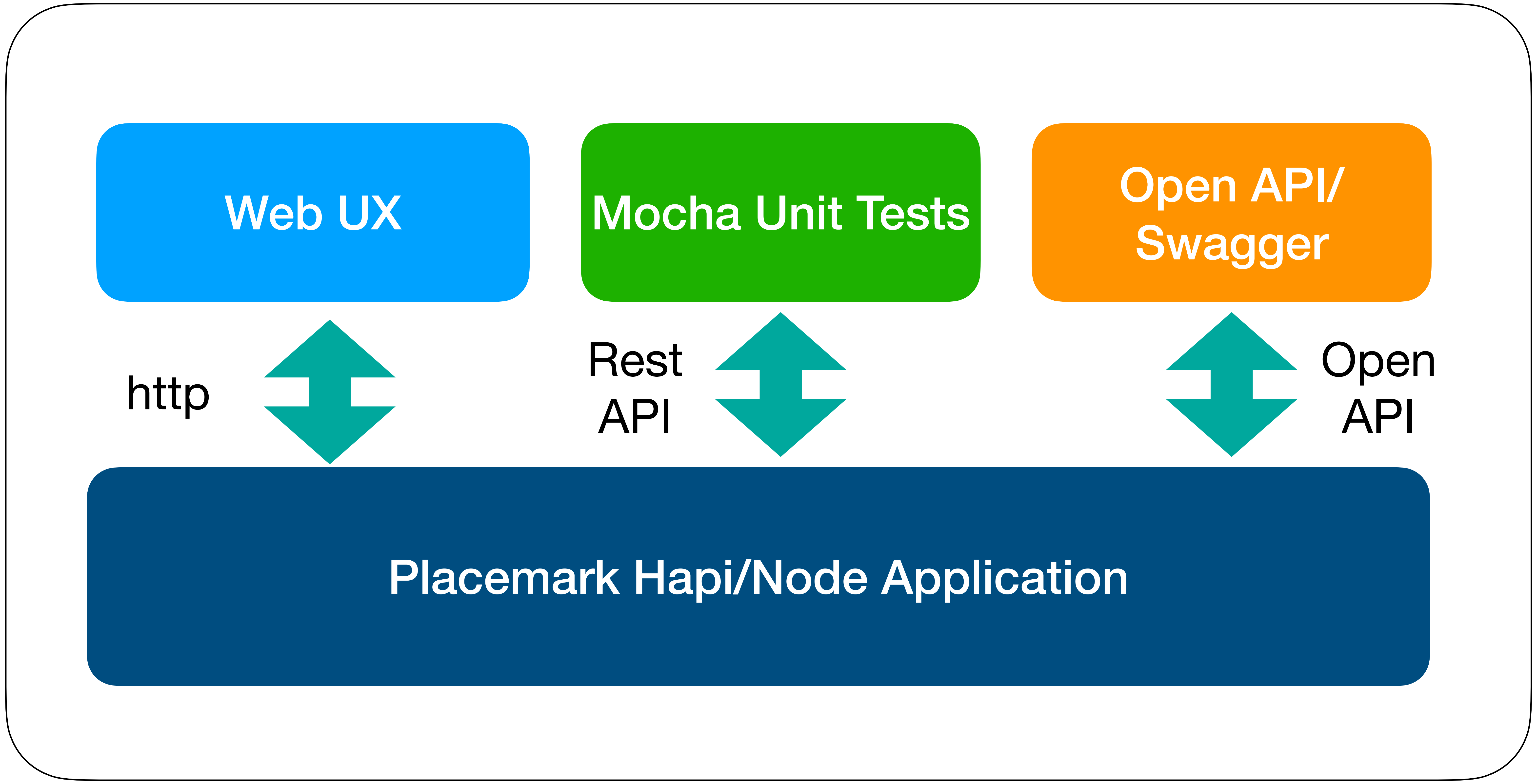
#1

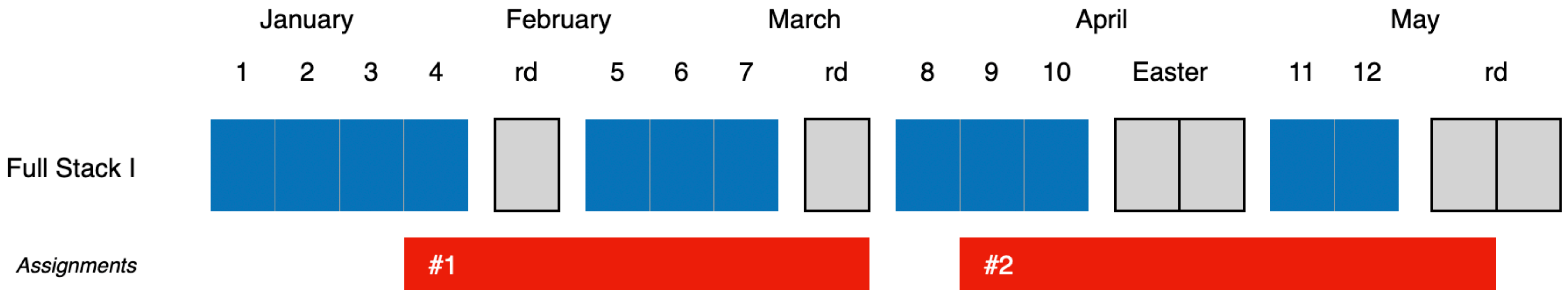
#2

PlaceMark 1.0 Architecture

(Assignment 1)

Submitted as a single repo



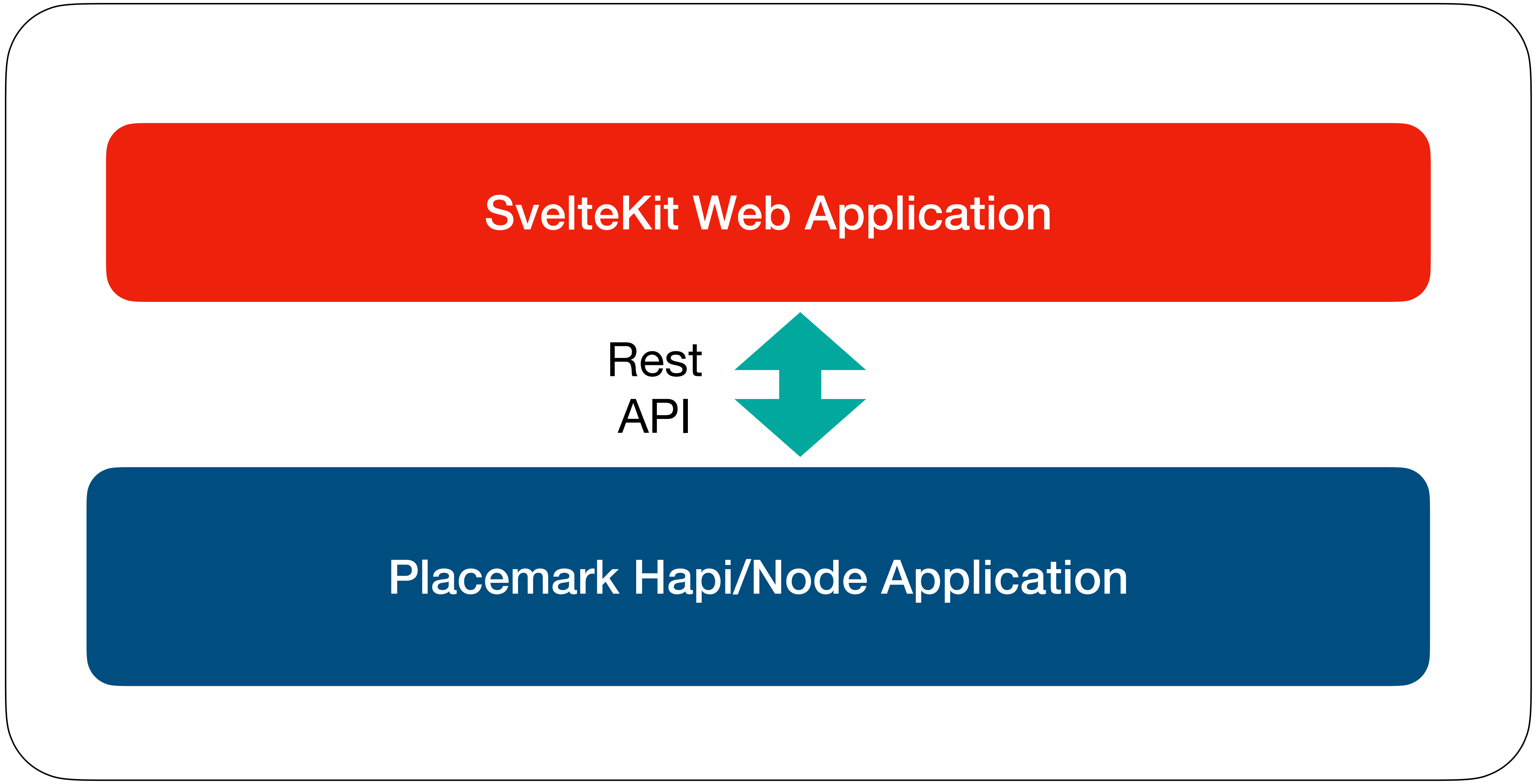


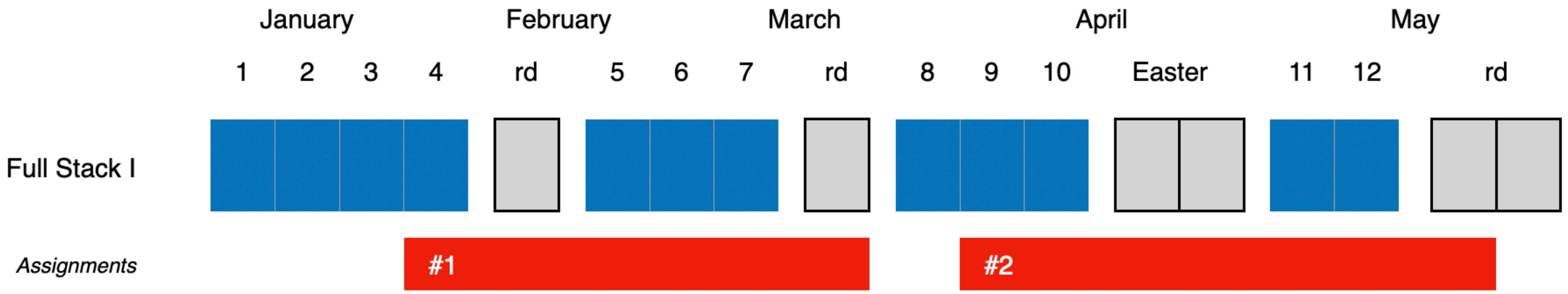
PlaceMark 2.0 Architecture

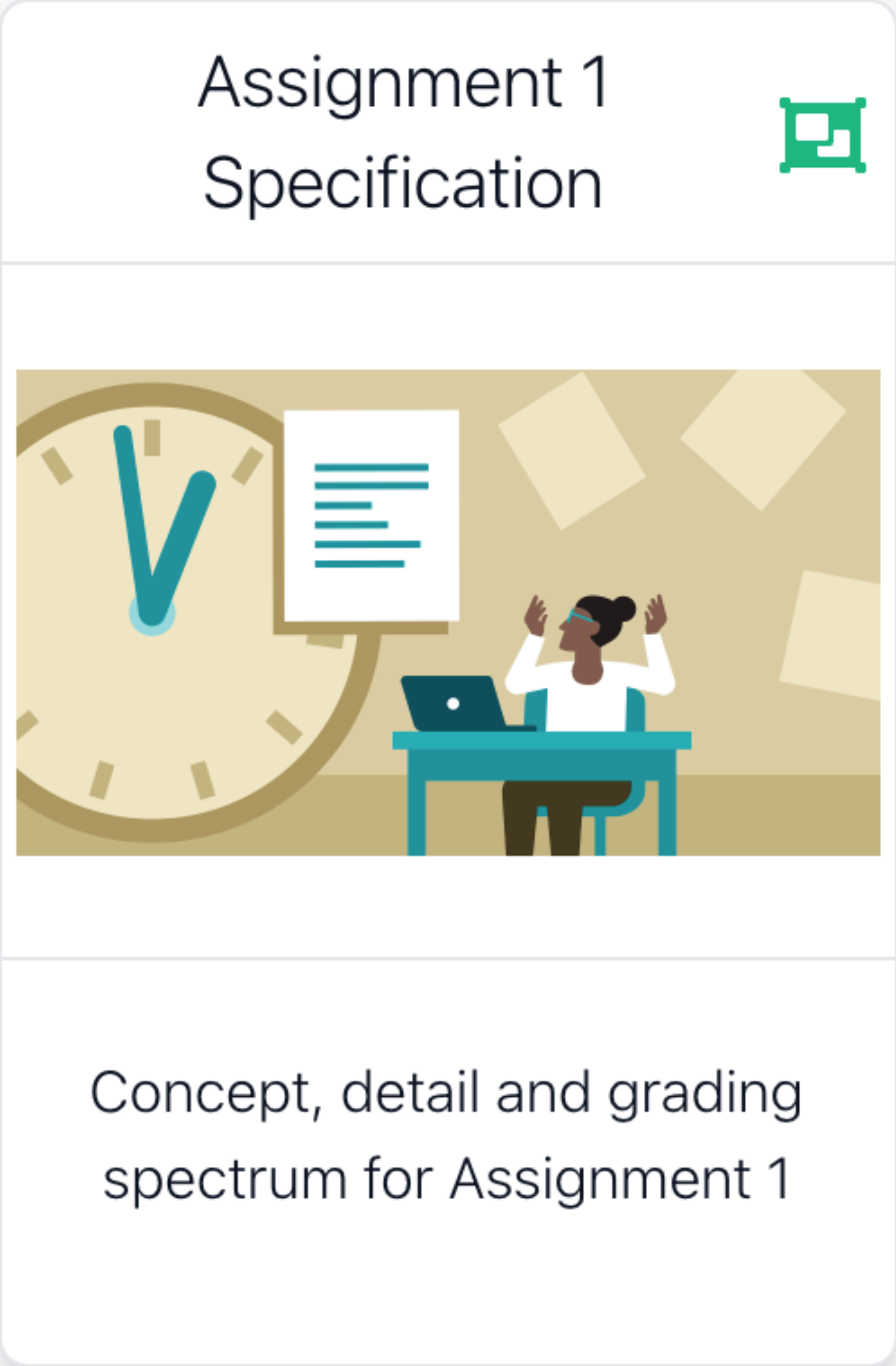
(Assignment 2)

SvelteKit Repo

Hapi/Node Repo







Assignment 1 Specification

Concept, detail and grading spectrum for Assignment 1