

Sitemate backend challenge

[Introduction](#_92km1nie59hw)

[REST API Server](#_5cu7auzemls1)

[REST API Client](#_khu0b65y9eed)

[Requirements](#_bq0ntoez8mos)

[Bonus point](#_tmv8q6xgw16l)

## Introduction

Your challenge is to create a simple **REST API** **Server + Client** for **Issues** (think GitHub or Jira Issues) within 2 hours.

Issues can be hard-coded JSON objects with just 3 attributes: id, title + description.

The client + server should accept or send these hardcoded JSON objects according to each API call: Create, Read, Update & Delete.

## REST API Server

The REST API server can be anything that can return static JSON - local on your machine or in the cloud. For example: local Node.js, Python or any server running on your machine, a serverless function in the cloud, a container, whatever you’re most comfortable with.

The server should support these 4 standard operations:

* Create: accepts a JSON object & prints/logs the object
* Read: returns a static JSON object
* Update: accepts a JSON object & prints/logs the object
* Delete: prints/logs out the object or id to delete

## 

## REST API Client

The REST API Client can be a command line interface (CLI) or a web app that connects to each of the server endpoints and prints out the response. Ideally, this is a Node.js or a React client, but could be anything that connects to your server.

The client should support the same 4 standard operations:

* Create: sends a JSON object to the server
* Read: requests a JSON object & prints it out
* Update: sends a JSON object to the server
* Delete: requests the server delete an issue

## Requirements

1. Your client and server should communicate together.

2. The API should be designed to be extensible.

3. Create a git repo in either Github, Gitlab, Bitbucket etc. Commit & push as you would for normally, we expect to see at least a few separate commits. Please share the url in the submission page

4. Please use Loom to record a short, 2 minutes high-level walkthrough for your solution covering the following points:

* Your technology choices
* Your API design
* Demonstrate each operation
* Any improvements you would like to make

## Bonus point

If you finish within the time limit, feel free to address one or more of these items:

* Send different issues objects depending on each API read request.
* Add unit tests
* Introduce a data store
* Create an interactive client
* Use docker to wrap the application