Front-end recruitment challenge: Agile board

Create a simple agile board web application using the REST API supplied.

- Your submission will be judged on *code quality* rather than aesthetics.
- You should only use the REST API supplied for persistence.
- You may use any libraries, frameworks or dependencies of your choice.
- Please include instructions for how to build and run your code.
- Don't forget to include any tests in the package.
- Keep the scope of your work limited as much as possible. You shouldn't spend more than a couple of hours on the solution.

Agile Board

An example agile board is shown below:

Story	To Do	In Progress	Done
As a product owner, I can create new stories so that I can request features	Create form for submitting new story	Create JSON fixtures for stories	Create example stories
As a developer, I can create new tasks for a story so that I can split it into work items		Create JSON fixtures for tasks	Create example tasks
As a product owner, I can change story priorities so I can make sure that work is tackled in priority order			
As a developer, I can change the status of a task so that I can track my work			

Note that a story can have many tasks.

More details about agile boards can be found here

Agile board API

The agile board API uses HTTP verbs and a RESTful endpoint structure. Request and response payloads are formatted as JSON.

Building, testing and running

To build, test and run the application use node package manager.

Build `npm install`

Test `npm test`

Run `npm start`

API summary

A complete REST operation is formed by combining an HTTP method with the full URI to the resource you're addressing. For example, here is the operation to create a new story:

```
POST http://localhost:3000/v1/stories
```

To create a complete request, combine the operation with the appropriate HTTP headers and any required JSON payload.

A summary of the available API operations is listed in the table below:

Action	Verb	Location
list stories	`GET`	`/stories`
create a story	`POST`	`/stories`
read a story	`GET`	`/stories/:story-id`
update a story	`PATCH`	`/stories/:story-id`
delete a story	`DELETE`	`/stories/:story-id`
list tasks	`GET`	`/tasks`
create a task	`POST`	`/tasks`
read a task	`GET`	`/tasks/:task-id`
update a task	`PATCH`	`/tasks/:task-id`
delete a task	`DELETE`	`/tasks/:task-id`

Current version

All requests should use the `v1` version of the API. This is expressed as the first path in the URL. E.g.

```
http://localhost:3000/v1/stories
```

Headers

`Accept` should be set to `application/json` for all requests.

`Content-Type` should be set to `application/json` for all `POST` and `PATCH` requests with a non-empty payload.

Client errors

Standard HTTP semantics apply for client errors (`4xx` status codes), e.g. when requesting a resource that does not exist you can expect `404 Not Found`.

Sending invalid JSON or incorrect JSON values for a resource will result in `400 Bad Request`. E.g.

```
{
  "message": "Body must be a JSON object"
}
```

Resources

Stories

A story is a high-level definition of a user's requirement, containing just enough information for developers to produce a reasonable estimate of effort and complexity. It captures which user want to achieve what and why in everyday business language.

For the purposes of this web application every story has an associated priority, persona (user type), feature summary and justification.

Story model

A story model contains the following fields:

Field	Type
id	uuid
created	ISO formatted date
modified	ISO formatted date
priority	integer
persona	string
feature	string
justification	string

List all stories

Request `GET /stories Accept: application/json`

Response `200 OK Content-Type: application/json`

```
[
  {
    "id": "fb61237c-99b4-41fe-bf29-cb10fde1213a",
    "created": "2015-08-17T18:04:15Z",
    "modified":  "2015-08-17T18:04:15Z",
    "priority": 1,
    "persona": "product owner",
    "feature": "create new stories",
    "justification": "request features"
  }
]
```

Create a story

Request `POST /stories Accept: application/json; Content-Type: application/json`

```
{
  "priority": 1,
  "persona": "product owner",
  "feature": "create new stories in the backlog",
  "justification": "request features"
}
```

Response
`201 Created Content-Type: application/json; Location: /stories/fb61237c-99b4-41fe-bf29-cb10fde1213a`

```
{
  "id": "fb61237c-99b4-41fe-bf29-cb10fde1213a",
  "created": "2015-08-17T18:04:15Z",
  "modified":  "2015-08-17T18:04:15Z",
  "priority": 1,
  "persona": "product owner",
  "feature": "create new stories in the backlog",
  "justification": "request features"
}
```

Read a story

Request `GET /stories/fb61237c-99b4-41fe-bf29-cb10fde1213a Accept: application/json`

Response `200 OK Content-Type: application/json`

```
{
  "id": "fb61237c-99b4-41fe-bf29-cb10fde1213a",
  "created": "2015-08-17T18:04:15Z",
  "modified":  "2015-08-17T18:04:15Z",
  "priority": 1,
  "persona": "product owner",
  "feature": "create new stories in the backlog",
  "justification": "request features"
}
```

Update a story

Request
`PATCH /stories/fb61237c-99b4-41fe-bf29-cb10fde1213a Accept: application/json; Content-Type: applicatic`

```
{
  "priority": 2
}
```

Response `200 OK Content-Type: application/json`

```
{
  "id": "fb61237c-99b4-41fe-bf29-cb10fde1213a",
  "created": "2015-08-17T18:04:15Z",
  "modified":  "2015-08-18T10:37:51Z",
  "priority": 2,
  "persona": "product owner",
  "feature": "create new stories in the backlog",
  "justification": "request features"
}
```

Delete a story

This only deletes the story resource. Linked tasks are not modified or deleted.

Request `DELETE /stories/fb61237c-99b4-41fe-bf29-cb10fde1213a Accept: application/json`

Response `204 No Content`

Tasks

A task is an item of work within a story that can be worked on by a single developer.

For the purposes of this web-application a task belongs to a single story and every task has an associated priority, description and status.

Task model

Field	Type
id	uuid
created	ISO formatted date
modified	ISO formatted date
story_id	uuid
priority	integer
description	string
status	string

List all tasks

Request `GET /tasks Accept: application/json`

Response `200 OK Content-Type: application/json`

```
[
  {
    "id": "993efc4e-0c85-4d1f-babc-8d7493f022ac",
    "created": "2015-08-17T18:05:17Z",
    "modified":  "2015-08-18T10:39:30Z",
    "story_id": "fb61237c-99b4-41fe-bf29-cb10fde1213a",
    "priority": 1,
    "description": "create JSON fixtures for stories",
    "status": "in progress"
  }
]
```

Create a task

Request `POST /tasks Accept: application/json; Content-Type: application/json`

```
{
  "story_id": "fb61237c-99b4-41fe-bf29-cb10fde1213a",
  "priority": 1,
  "description": "create JSON fixtures for stories",
  "status": "to do"
}
```

Response
`201 Created Content-Type: application/json; Location: /tasks/993efc4e-0c85-4d1f-babc-8d7493f022ac`

```
{
  "id": "993efc4e-0c85-4d1f-babc-8d7493f022ac",
  "created": "2015-08-17T18:05:17Z",
  "modified":  "2015-08-17T18:05:17Z",
  "story_id": "fb61237c-99b4-41fe-bf29-cb10fde1213a",
  "priority": 1,
  "description": "create JSON fixtures for stories",
  "status": "to do"
}
```

Read a task

Request `GET /tasks/993efc4e-0c85-4d1f-babc-8d7493f022ac Accept: application/json`

Response `200 OK Content-Type: application/json`

```
{
  "id": "993efc4e-0c85-4d1f-babc-8d7493f022ac",
  "created": "2015-08-17T18:05:17Z",
  "modified":  "2015-08-17T18:05:17Z",
  "story_id": "fb61237c-99b4-41fe-bf29-cb10fde1213a",
  "priority": 1,
  "description": "create JSON fixtures for stories",
  "status": "to do"
}
```

Update a task

Request
`PATCH /tasks/993efc4e-0c85-4d1f-babc-8d7493f022ac Accept: application/json; Content-Type: application/`

```
{
  "status": "in progress"
}
```

Response `200 OK Content-Type: application/json`

```
{
  "id": "993efc4e-0c85-4d1f-babc-8d7493f022ac",
  "created": "2015-08-17T18:05:17Z",
  "modified":  "2015-08-18T10:39:30Z",
  "story_id": "fb61237c-99b4-41fe-bf29-cb10fde1213a",
  "priority": 1,
  "description": "create JSON fixtures for stories",
  "status": "in progress"
}
```

Delete a task

Request `DELETE /tasks/993efc4e-0c85-4d1f-babc-8d7493f022ac Accept: application/json`

Response `204 No Content`