



Front-end Developer Challenge

The aim of this task is not to 100% complete the assignment (of course this would be great), but rather to find out more about your approach, the **quality, cleanliness and structure** of your code, and get to know your personal **prioritization**. This means that it's better to send an unfinished app with code you can be proud of, rather than a complete app with lots of messy code. Mockups in this task are just a design suggestion - you are free to implement any other design as long as it looks OK and meets the requirements.

Notes

- The backend can be accessed via `http://localhost:3001`
- It runs with a package called **json-server**, and the data comes from the **db.json** file
- The JSON structure of the **db.json** file is twisted on purpose. Please do not change this original structure. The contents of the file are going to change once the application is running and you start adding, editing or deleting records, but the structure has to remain, so you'll have to find a way to edit the records without losing this structure

Steps for the challenge

- Clone this repository (Please, do not fork it!): <https://github.com/movingimage-evp/mi-afec> and complete the tasks below
- Upload your code to a repository of your choosing (GitHub, BitBucket, etc.) and send us the link

Steps to run the project

- Install dependencies with:

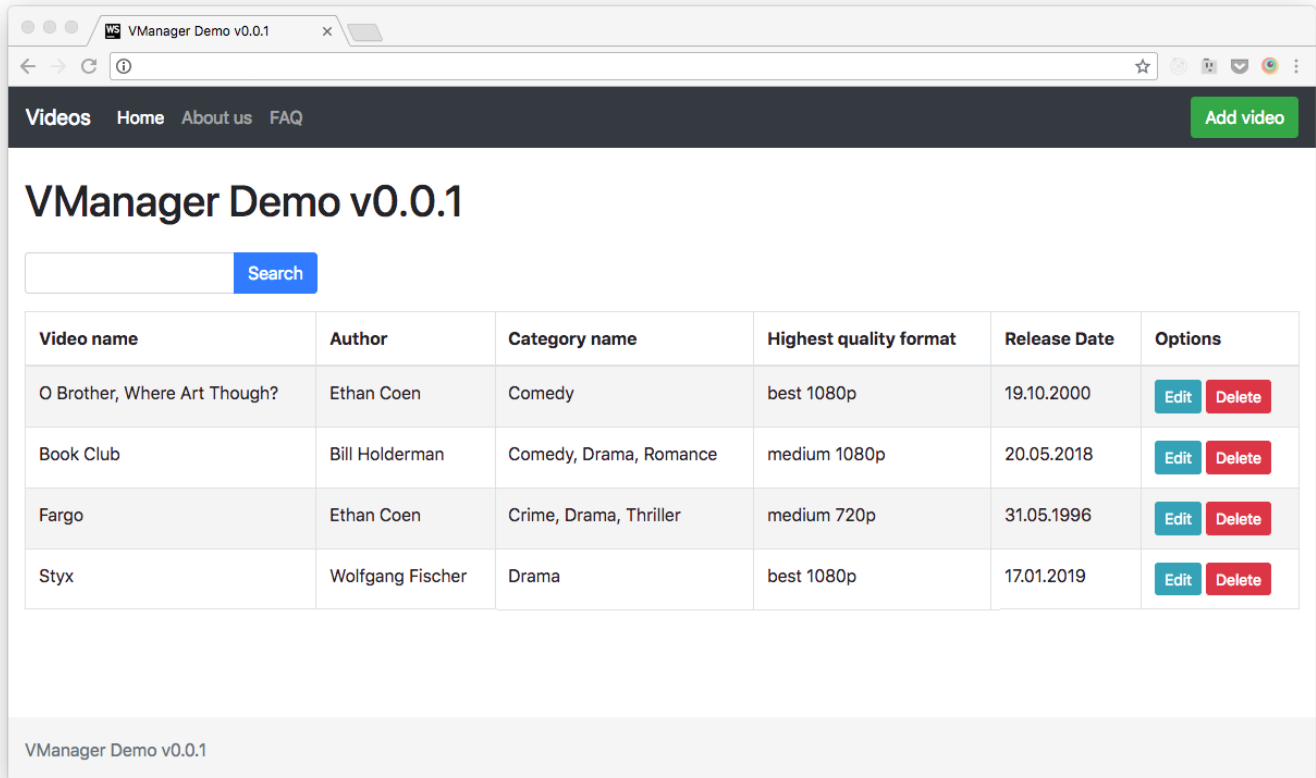
yarn

- Run both the frontend and backend with:

yarn start

List View

UI Suggestion



Requirements

- The landing page displays a list of videos with the following columns:
 - Video name
 - Author
 - Categories
 - Highest quality format
 - Release Date (Can be random data)
 - Options (Buttons: [Edit](#) [Delete](#))
- The table/list can be searched
- **Optional:** The list can be sorted

The "Highest quality format" is a made-up label for the format that has the biggest "size" and the highest "res". Take a look at the following database entry:

```
"formats": {
  "one": { "res": "1080p", "size": 1000 },
  "two": { "res": "720p", "size": 2000 },
  "three": { "res": "720p", "size": 900 }
},
```

So, to choose the biggest format here, we take the one that has the biggest size (in this case "two"). In cases where there are multiple formats with the same "size", we use the "res" value to decide which one to get (Note: "1080p" is bigger than "720p", "720p" is bigger than "480p" and so on).

With the biggest format selected we then form the label as "format_name res". So for this particular example, the generated label should be "two 720p" (The format name is "two", and its res is "720p").

Add a Video

UI Suggestion

Video name

Video author

Video category

Comedy
Criminal
Drama
Thriller

Submit Cancel

Requirements

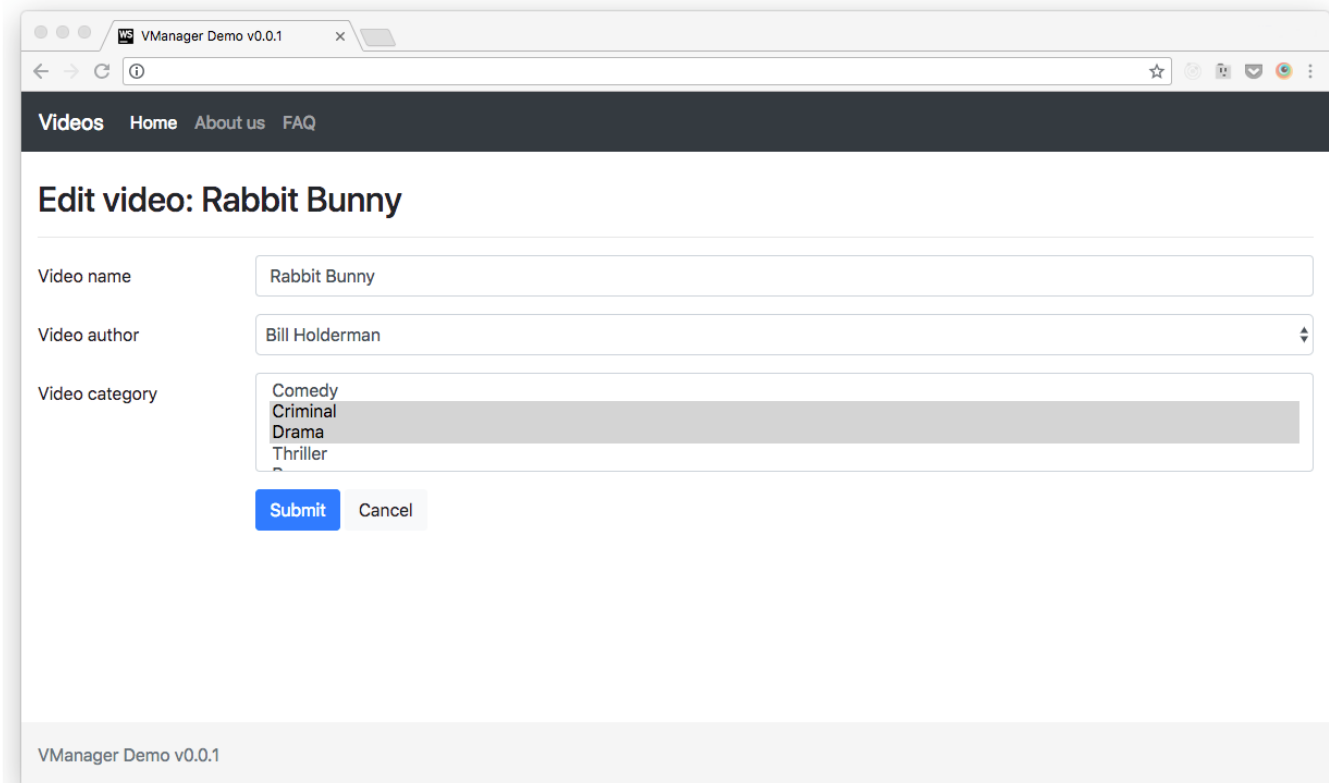
- Clicking on the "Add video" button will take you to a form, which contains the following fields:
 - Video name
 - Video author (`<select>`)
 - Video category (`<select multiple>`)
- A new video can be saved and the user will be returned to the list view
- The new video object should automatically get the following property/value:

```
formats: {  
  one: { res: "1080p", size: 1000 }  
}
```

- The process can be canceled and by doing so, the user is redirected to the list view
- Optional:** There are basic validations available (e.g.: The "Save" button is only active if all the content is valid)

Edit a video

UI Suggestion



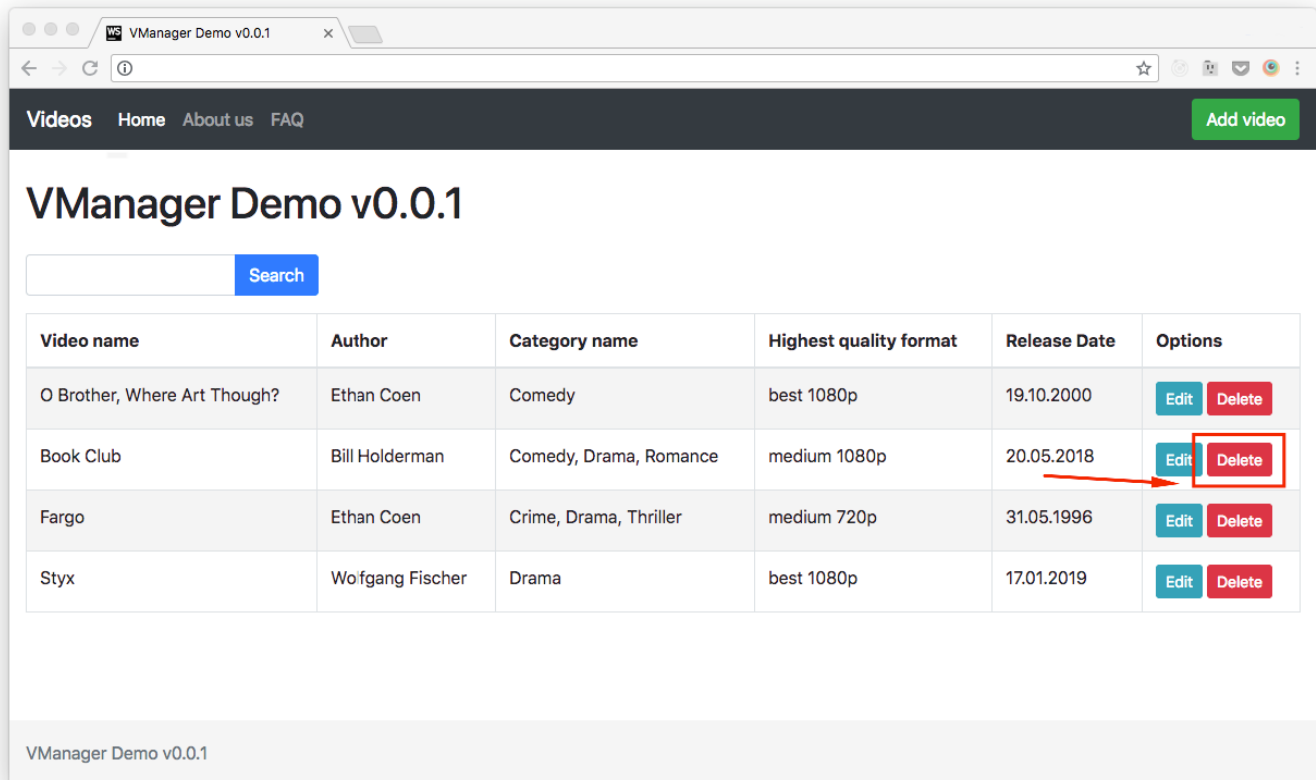
The screenshot shows a web browser window with the title 'VManager Demo v0.0.1'. The browser's address bar is empty. The website has a dark navigation bar with links: 'Videos', 'Home', 'About us', and 'FAQ'. The main content area is titled 'Edit video: Rabbit Bunny'. It contains three form fields: 'Video name' with the value 'Rabbit Bunny', 'Video author' with the value 'Bill Holderman', and 'Video category' with a dropdown menu showing 'Comedy', 'Criminal', 'Drama', and 'Thriller'. Below the form fields are two buttons: 'Submit' (blue) and 'Cancel' (gray). The footer of the page says 'VManager Demo v0.0.1'.

Requirements

- There is an interaction element to edit existing video information
- Clicking on this element will take you to a form view
- All changes can be saved (e.g.: If canceled, the user is returned to the list view)
- **Optional:** There are basic validations available

Delete a video

UI Suggestion



Requirements

- There is an interaction element to delete an existing video
- **Optional:** A dialog will appear to confirm the deletion