Studiomap Quick Audit

Routine audit Studiomap: 4 July 2024

Github : <https://github.com/selego/studiomap-new>

[**The backend 1**](#_ek4trf35llh0)

[1. Improve naming of file (LOW) 1](#_xug9sodxfpvg)

[**Admin 2**](#_6gpjoagxod2)

[2. Fetch functions outside useEffect (LOW) 2](#_ggz62gp27vkf)

[3. Replace Redux with zustand ( MEDIUM ) 2](#_2b2woscxvi74)

[4. Keep one api call per function ( LOW ) 2](#_wfwpa5d2kev3)

[5. Use FileInput component? If you need style you can modify that based on the project ( LOW ) 2](#_5p90pmvghbc8)

[6. Use try catch blocks ( HIGH ) 2](#_rto69c7vdk9j)

[**The web app 3**](#_1gg1r2y6du9j)

[7. Fetch functions outside useEffect: (MEDIUM) 3](#_lfquz4bi7ebz)

[8. I think using for ( const f of files){} , make the code cleaner and you do not need promise.all() (LOW) 3](#_yzi2dpdxw57p)

[9. Replace Redux with zustand ( MEDIUM ) 3](#_rqzm5xjfbw2t)

[10. Use FileInput component? If you need style you can modify that based on the project: ( LOW ) 3](#_us851t4ofcqa)

[11. Why an endpoint that needs another call before? ( HIGH ) 3](#_kpmau6cd74yd)

[12. Filter in back ( HIGH ) 4](#_o9zxmgmattec)

[13. Is this file to modify calendar style by using its classes ? (question) 4](#_5f2303ocuw79)

[**Webflow 4**](#_dw5stphgztax)

### 

# The backend

### Improve naming of file (LOW)

<https://github.com/selego/studiomap-new/blob/main/api/src/a.js>

### 

# Admin

### Fetch functions outside useEffect (LOW)

<https://github.com/selego/studiomap-new/blob/main/admin/src/App.jsx>

### Replace Redux with zustand ( MEDIUM )

### Keep one api call per function ( HIGH )

<https://github.com/selego/studiomap-new/blob/main/admin/src/scenes/studios/view/index.js>  
  


### Use FileInput component? If you need style you can modify that based on the project ( HIGH )

<https://github.com/selego/studiomap-new/blob/main/admin/src/scenes/studios/view/more_info.js>

### Use try catch blocks ( HIGH )

<https://github.com/selego/studiomap-new/blob/main/admin/src/scenes/studios/view/pricing.js>



# The web app

### Fetch functions outside useEffect: (MEDIUM)

<https://github.com/selego/studiomap-new/blob/main/app/src/App.jsx>

### I think using for ( const f of files){} , make the code cleaner and you do not need promise.all() (HIGH)

<https://github.com/selego/studiomap-new/blob/main/app/src/scenes/studio/viewTabs/general.jsx>  


### Replace Redux with zustand ( MEDIUM )

### Use FileInput component? If you need style you can modify that based on the project: ( LOW )

<https://github.com/selego/studiomap-new/blob/main/app/src/scenes/dashboard/gerant/profil.jsx>  


### Why an endpoint that needs another call before? ( HIGH )

<https://github.com/selego/studiomap-new/blob/main/app/src/scenes/dashboard/gerant/reservations.jsx>



### Filter in back ( HIGH )

<https://github.com/selego/studiomap-new/blob/main/app/src/scenes/dashboard/artist/reservations.jsx>



### Is this file to modify calendar style by using its classes ? (question)

<https://github.com/selego/studiomap-new/blob/main/app/src/scenes/booking/calendar.css>

# Webflow

Generally good, just I think would be better to use “ok” to check

| const fetch = require('node-fetch'); const { URLSearchParams } = require("url");  const { WEBFLOW\_TOKEN } = require("../config");  //https://developers.webflow.com/data/reference/rest-introduction  class Api {  constructor() {  this.token = ``;  this.siteId = ``;  }   setSiteId(siteId) {  this.siteId = siteId;  }   sendRequest = async (path, method, body) => {  try {  // add 1 second delay to avoid rate limiting  await new Promise((resolve, reject) => setTimeout(resolve, 1000))   const url = `https://api.webflow.com/v2${path}`;   const headers = { Authorization: `Bearer ${WEBFLOW\_TOKEN}`, "accept-version": "1.0.0", };   if (method === "post" || method === "put" || method === "patch") {  headers["Content-Type"] = "application/json";  }   const options = { method, headers, body: body ? JSON.stringify(body) : null, };   const response = await fetch(url, options);  const result = await response.json();  return result;  } catch (error) {  console.error("Error:", error);  return null;  }  };      async get(path, params = null) {  let fullPath = path;  if (params) fullPath += "?" + new URLSearchParams(params);  return await this.sendRequest(fullPath, "get");  }  async post(path, body = null) {  return await this.sendRequest(path, "post", body);  }   async put(path, body = null) {  return await this.sendRequest(path, "put", body);  }   async patch(path, body = null) {  return await this.sendRequest(path, "patch", body);  }   async remove(path) {  return await this.sendRequest(path, "delete");  } }  const a = new Api(); |
| --- |

| const { WEBFLOW\_TOKEN, ENVIRONMENT } = require("../config");  const sendRequest = async (path, method, data) => {  try {  const url = `https://api.webflow.com/v2${path}`;   const headers = {  Authorization: `Bearer ${WEBFLOW\_TOKEN}`,  "accept-version": "1.0.0",  };   if (method === "post" || method === "put" || method === "patch") {  headers["Content-Type"] = "application/json";  }    const response = await axios({ method, url, headers, data, validateStatus: () => true });      if (response.data && response.data.message) return { ok: false, errorData: response.data };   return { ok: true, data: response.data };  } catch (error) {  console.error("Error:", error);  return null;  } };  async function get(path, params = null) {  let fullPath = path;   if (params) {  // Remove undefined values from the params object  const cleanedParams = Object.fromEntries(Object.entries(params).filter(([\_, value]) => value !== undefined));  fullPath += "?" + new URLSearchParams(cleanedParams);  }   return await sendRequest(fullPath, "get"); }  async function post(path, body = null) {  return await sendRequest(path, "post", body); }  async function put(path, body = null) {  return await sendRequest(path, "put", body); }  async function patch(path, body = null) {  return await sendRequest(path, "patch", body); }  async function remove(path) {  return await sendRequest(path, "delete"); }  async function listSites() {  try {  const response = await get(`/sites`);   return response;  } catch (error) {  console.error("Error fetching list of sites:", error);  return { ok: false, errorData: error };  } }  async function getSite(id) {  try {  const response = await get(`/sites/${id}`);   return response;  } catch (error) {  console.error("Error fetching site:", error);  return { ok: false, errorData: error };  } } |
| --- |