# OFM Frontend Scaffolding: status and next steps

## Background

This document presents information on the objectives and results achieved towards the creation of a 1st draft vue3 frontend for the OFM project. Additionally discussed, are the current frontend status and suggested next steps taken towards a 2nd iteration.

## Objectives: 1st iteration of OFM frontend

* Create a vue 3 project from scratch using Vite.
* Review the STUDENT-ADMIN frontend and move any common code (i.e. UI, authorization, backend integration) into the OFM frontend.
* 1st draft OFM frontend will not be connected to authorization or backend services, however common authorization/backend code has been copied over from STUDENT-ADMIN and commented out until auth/backend integration is ready.
* Bring the frontend to a state where the application will load to a homepage with common UI components (header/footer, menu) styled from STUDENT-ADMIN.
* Address and reduce errors as much as possible. i.e. commenting out authorization/backend code temporarily.

## Status: 1st iteration of OFM frontend

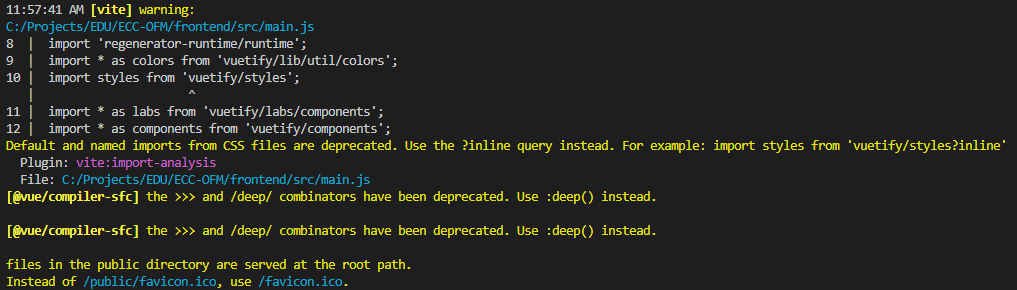
* A 1st draft of the OFM frontend has been completed while mostly meeting the stated objectives above… some styling warnings on the node.js side exists which can be further flushed out in the 2nd iteration.
* Throughout the current code base are a number of “TODO” comments added. These comments indicate where further consideration and action is required in the 2nd iteration of the frontend. NOTE: the following are examples of the ‘//TODO’ occurrences, not the full list.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

## Next Steps: 2st iteration of OFM frontend

The 2nd iteration of the OFM frontend is suggested to include the following summarized steps to further refine and build out a solid starting point for OFM development. NOTE: it is expected addition steps other than the ones stated below will be realized, and required as we work through establishing a base frontend framework for OFM:

1. Consider adding Typescript support. NOTE: STUDENT-ADMIN nor MyCCS used typescript. However vue 3 does claim it has a more seamless integration with TypeScript than vue 2.
2. Consider replacing Vuetify component library with PrimeVue. NOTE: STUDENT-ADMIN and MyCCS both use the Vuetify component library.
3. Need to determine if we’ll use Options API or Composition API. The composition API in vue 3 has many benefits over the Options API (i.e. easier to reuse code, managing complex logic\behavior, typescript support, flexibility in design of reactive state). NOTE: the current iteration of the OFM frontend (based on STUDENT-ADMIN) uses Options API.
4. The libraries included in the frontend 1st iteration are minimalist. When copying over common code from STUDENT-ADMIN, the only libs added to package.json were very obvious ones (i.e. vue 3, vite, pinia, vuetify, eslint to name a few) or ones from which runtime errors were raised as missing. A more thorough exercise should be conducted to review the OFM package.json against the STUDENT-ADMIN package.json to determine further desired libs and configurations.
5. Review code in OFM frontend marked as “unsure if common”. The following artifacts were included from STUDENT-ADMIN, however there was some uncertainty if they’re 100% common or business specific to STUDENT-ADMIN. Remove if any are determined STUDENT-ADMIN business specific. Ideally this would be advised by a developer familiar with the STUDENT-ADMIN application. Artifacts in question include:
   * frontend/src/components/util/SetNavigation.vue
   * frontend/src/components/util/SnackBar.vue
   * frontend/src/store/modules/notifications.js
   * frontend/src/store/modules/setNavigation.js
6. When authorization and backend services are ready for integration with the frontend, as part of that integration task, uncomment related code that was copied over from STUDENT-ADMIN. The code can be quickly identified by searching for instances of “TODO: uncomment during integration”.
7. Code related to user role/request based access determination needs to be updated to match OFM requirements. The following is a list of the artifacts that will need to be updated:
   * src\store\modules\auth.js, temp OFM roles added for the sake of example and allowing the application to load with a NavBar.
   * src/utils/constants/Roles.js, replace STUDENT-ADMIN with OFM specific roles.
   * src/common/Role-based-access.js, replace STUDENT-ADMIN request type to role mappings and methods with OFM equivalent.
8. In the constants.js there is a lot of STUDENT-ADMIN specific code that can be removed. An exercise to review and remove such code should be planned.
9. Some styling errors on the node.js side exist. These will need to be addressed. i.e.



1. In the copied over common STUDENT-ADMIN code, there were some eslint rules failing which have been addressed by adding rules to files with these failures. It is likely this has to do with further required esilint configuration. To find occurrences of these rules in the code, search for any of the rules below. An effort to determine if these can be resolved with configuration should be conducted. It’s worth noting the copied across common STUDENT-ADMIN code does include instances of these added rules in addition to other ones, thus not sure if this is an issue or not.

* // eslint-disable-next-line vue/no-reserved-component-names
* // eslint-disable-next-line vue/multi-word-component-names
* // eslint-disable-next-line vue/multi-word-component-names
* // eslint-disable-next-line vue/no-reserved-component-names

1. Address any remaining “TODO” comments.