openIMIS frontend migration II

Proposed solution for runtime upgrade

Agenda

1. Short summary of previous presentation
2. The problem with “Create React App”
3. Possible tools/solutions
4. Prefered solution
5. Testing libraries
6. Vitest
7. Conclusions

Refresher

- Currently openIMIS uses client-side rendering with the create-react app (CRA) as a serving tool.
- Is in a limbo, but can be considered deprecated in practice
- ~~No new commits since September 2022~~ 2 new commits (theory)!
- We have to pick a new solution and we can use that opportunity to change rendering style (Client- or server-side rendering)
- CSR approach is still recommended
- Slides available here.

Prefered solution - Vite

Advantages:

- CSR approach - we would avoid major rewrites of the code
- Ongoing support
- Faster than CRA (mostly for developers - build times etc.)
- Partial support for SSR (with plugins) - it would be possible to test this approach in the future if needed (admittedly - next.js supports CSR)
- It would invalidate the need for some of the upgrades mentioned in the Part I of the frontend update.
- Time estimates - would need more research after approach is chosen.

Vite - complexity of migration

One of the resources mentioned following steps:

1. Add/remove dependencies
2. Adjust package.json and other configuration files
3. Rename all react files from *.js to *.jsx
4. Move some of the files (index.html for example)

That’s shortened version, and third point seems to be the most time consuming.

Things could get more complicated because of other dependencies, but they have to be changed anyway.

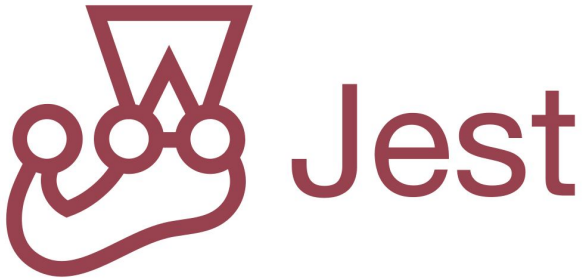
Testing libraries - badly needed

- Currently there is no testing coverage for the frontend code.
- On the backend it would have to be extended, but at the moment some tests do exist.
- We cannot add testing before choosing framework, there are different tools (CRA-Jest, Vite-Vitest, Next - Jest or React Testing library)
- It would allow better cooperation between different teams and implementations
- A lot of bugs could be avoided while developing new features.

Possible solutions

Few key points about Vitest:

- Minimal configuration
- Designed for Vite
- Some other solutions do exist - eg. Jest (actually the most popular and still maintained)
- They may be more compatible with other tools, but could be hard to set up with Vite
- Would require more research/analysis after migrating to Vite



General timeline/steps

1. Migrate from CRA to Vite (and preferably update node version).
2. Add Vitest.
3. Add basic test coverage for the general openIMIS functionalities (such as logging, running a module etc.).
4. Add a coverage to a single module (for example claims - adding a claim, editing, searching etc.) as a point of reference for other developers.
5. Encourage community members to add tests while working on different functionalities.
6. At some point (X modules covered, one covered extensively or right after step 4) migrate other dependencies (for example MUI), some are getting deprecated soon (Redux - createStore).

Conclusions

1. Important decision - has major impacts on future development of the app and adding new features - CSR and Vite preferred.
2. On the other hand - SSR would have more support and could become standard approach in the future - could be assumed by new developers.
3. Apart from keeping openIMIS up-to-date it would enable us to add tests in the openIMIS