openIMIS

Gumzo ya Mwezi 06/04/2020

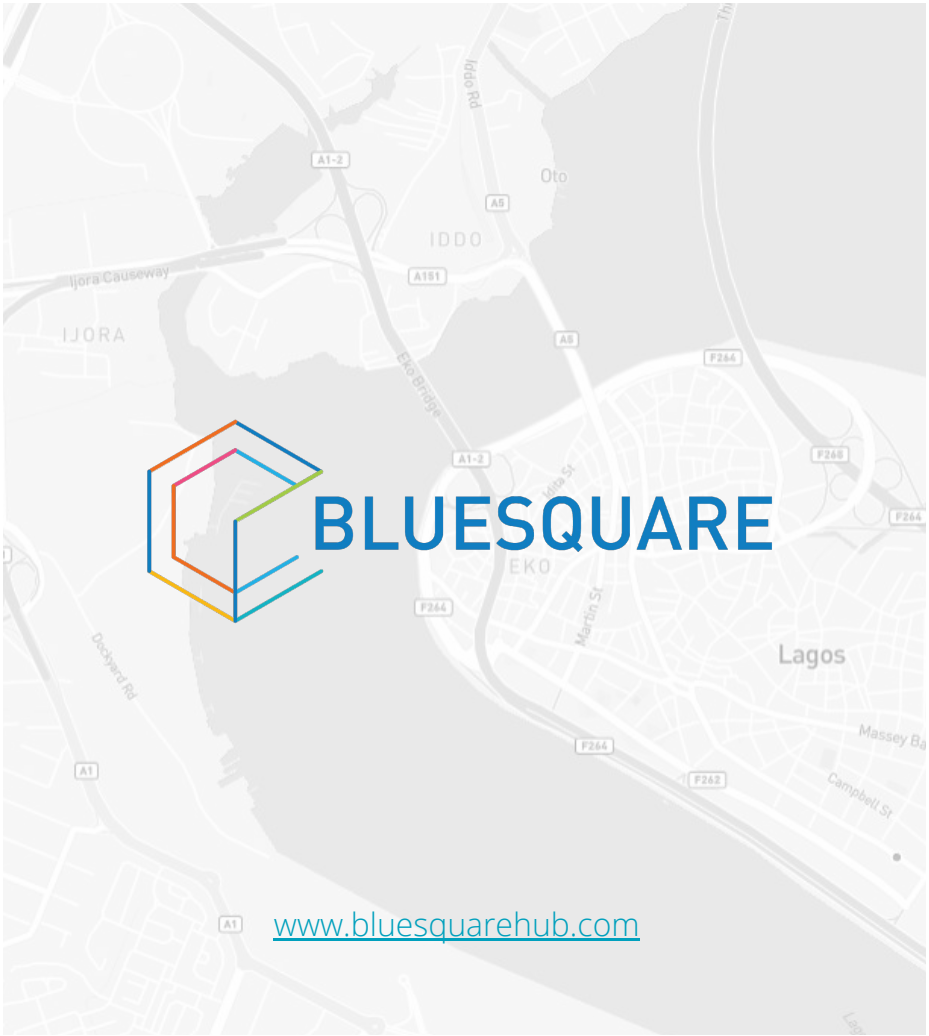
Agenda

- Bluesquare:
  - who we are
  - our engagement towards openIMIS community
  - our methodology
- Achieved

  what we delivered this month

- Roadmap

  what we will deliver and what are our dependencies



Delivering innovative technology for better lives.

www.bluesquarehub.com

Fall 2018

what we do

COUNTRY LEVEL DATA SYSTEMS 24 COUNTRIES

We build technologies that enhance governmental health data systems with a focus on three markets:

HEALTH FINANCING DATA SYSTEMS

- Data systems for purchasers, health insurance, Ministries of Health
- Example: Develop a Pay for Performance data system in Kyrgyz hospitals

GOVERNMENT HEALTH DATA WAREHOUSES

- Example: The health data warehouse in Morocco

DISEASE OR THEMATIC DATA SYSTEMS

- Diabetes
- HIV
- Tuberculosis
- Malaria
- Immunization systems
- Vector Borne eradication systems (i.e. sleeping sickness)
- Family Planning
- Emergency Obstetric Care

Bluesquare develops these data systems based on a suite of in-house software products connected to DHIS2 a popular open source data management platform used by over 40 governments.

How we do
IT products and data services

We deliver technologies and services that strengthen governmental health data systems, mainly:

Hesabu (aka ORBF)

- An open sourced rule engine that allows complex calculations in DHIS2, a popular open source data management platform. This is particularly useful for health financing data systems.

Data Viz

- A public web dashboard that allows showcasing results.

Modeling and data science

- Statistical analysis, Data cleaning, Modeling & machine learning and analysis automation to help customers bring value out of their health data.

Bluesquare suite of in-house software products and services allow collecting, computing, analyzing and visualizing data in a intelligent and friendly manner.


24
COUNTRIES

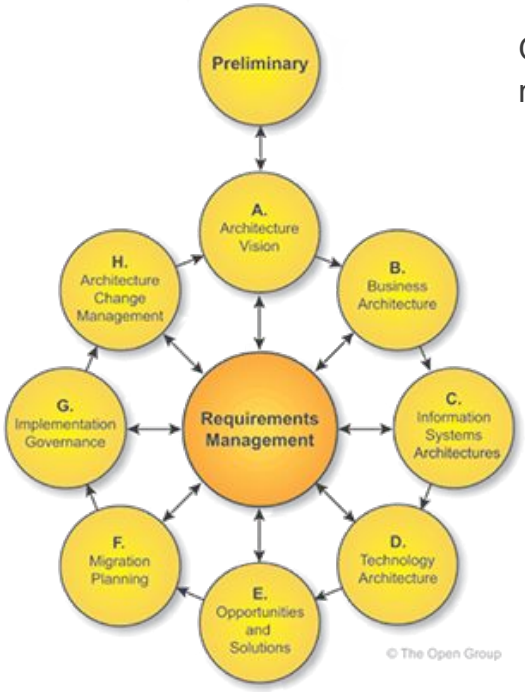
Bluesquare: our engagement towards openIMIS community

We believe that health insurance will be at the heart of the UHC agenda in many countries.

openIMIS modular transformation is an opportunity to develop code that can be used at scale to help provide health services “for the global good” (i.e. exact DNA of Bluesquare).

Creating synergies with our existing and future health-financing portfolio, promoting the tool in the countries where we operate.

Bluesquare: our methodology



Our approach to deliver the openIMIS modules borrows several concepts from TOGAF, most important one being the ADM (Architecture Development Method):

- Iterative, ensuring pragmatism and responsiveness in delivered solution
- We strive to keep things simple: we aim to use the TOGAF framework as a guide not a rule book. Where we feel it will serve this project we will make use of it. However, our proposed approach is much lighter than a traditional TOGAF implementation effort.
- It helps any community member to find/contribute to the appropriate part of the system.

Agenda

- Bluesquare:
  - who we are
  - our engagement towards openIMIS community
  - our methodology
- Achieved

  what we delivered this month

- Roadmap

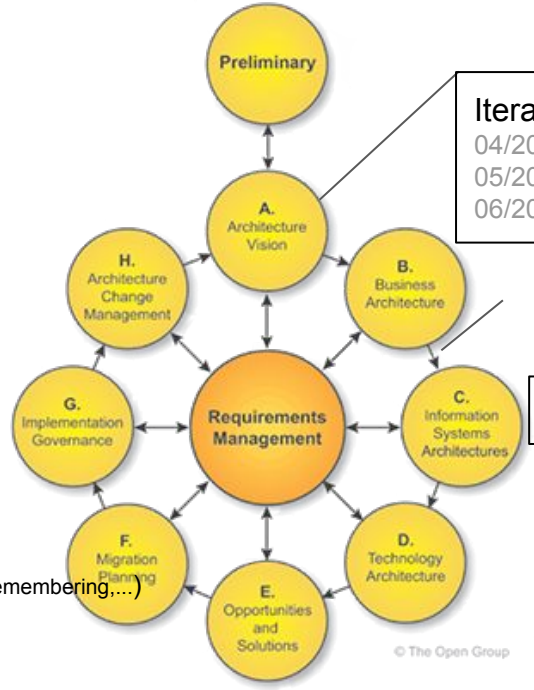
  what we will deliver and what are our dependencies

Iteration 3:

144

90 (initial estimate)

Achieved (Iteration 3): Claim Module



Iteration 3:

04/2019 : Claim module scope document (draft)	2 m/d
05/2019 : Mapping attempt to JLN business processes	0.5 m/d
06/2019 : Claim main screen mockups	4,5 m/d

06/2019: Django Dynamic Rest > (GraphQL) 4 m/d

07/2019: 12 m/d (scope, design & Enquiry)
08/2019: 34 m/d (Claim implementation)
09/2019: 29 m/d (Claim refinement)
10/2019: 22,5 m/d (Nepali cust., Install & debug)
11/2019: 12 m/d (Addis Abeba, bug fixes, deploy)
12/2019: -
01/2020: 1 m/d (restarted batch processing to python)
02/2020: 10,5 m/d
03/2020: 12 m/d
- bug fixing (mainly claim processing logic)
- changes (mainly UI: open claim in new tabs, filter criteria remembering,...)

Iteration was closed 01/03
… but was **re-opened** to depict the actual work being done (more than 'just follow-up')

Iteration is now definitively closed, with 54 m/d over budget

Opening claims in new tabs 

Reported ‘independently’ by Purushottam (OMT-149) and Patrick (OMT-166)

(… so probably I “should have known” that users need to open several claims at once)

Modern app are not supposed to spread over multiple tabs

- UX designers recommend ‘immersive’ applications (users stick into it)

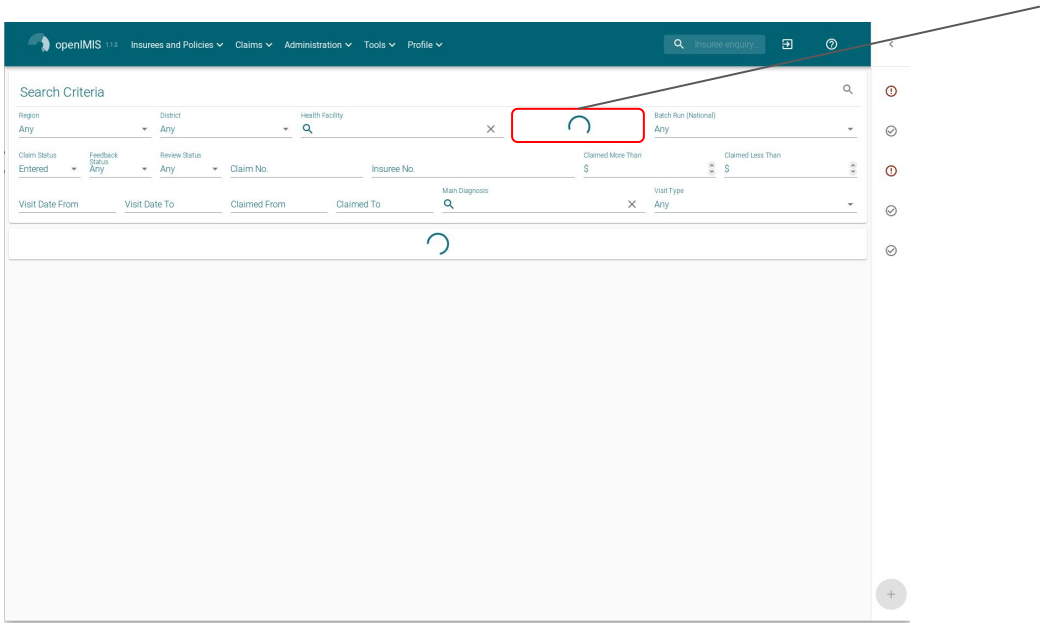
  so we should implement the tabs ‘within’ openIMIS screen

- Technically speaking, multi tabs is a bad idea
  - can’t share (redux) cache between tabs
  - because of the React ‘client router’ mechanism, “permalinks” had to be quickly added (i.e. I hacked it)

Opening claims in new tabs: the cache ‘problem’

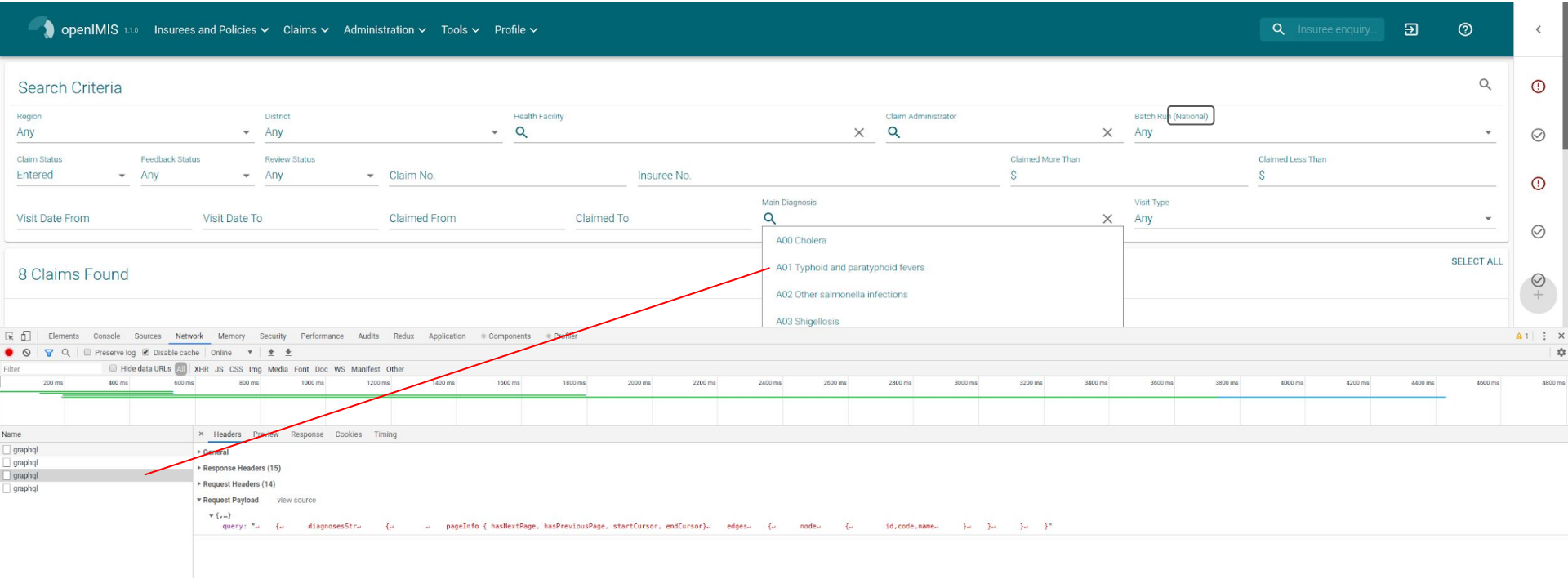
When one <<screen>> is opened, several cache entries are populated

Claim Administrators, being loaded in cache



1 <<screen>> (searching claim in HF), 4 (graphql) queries towards server (...the first time):

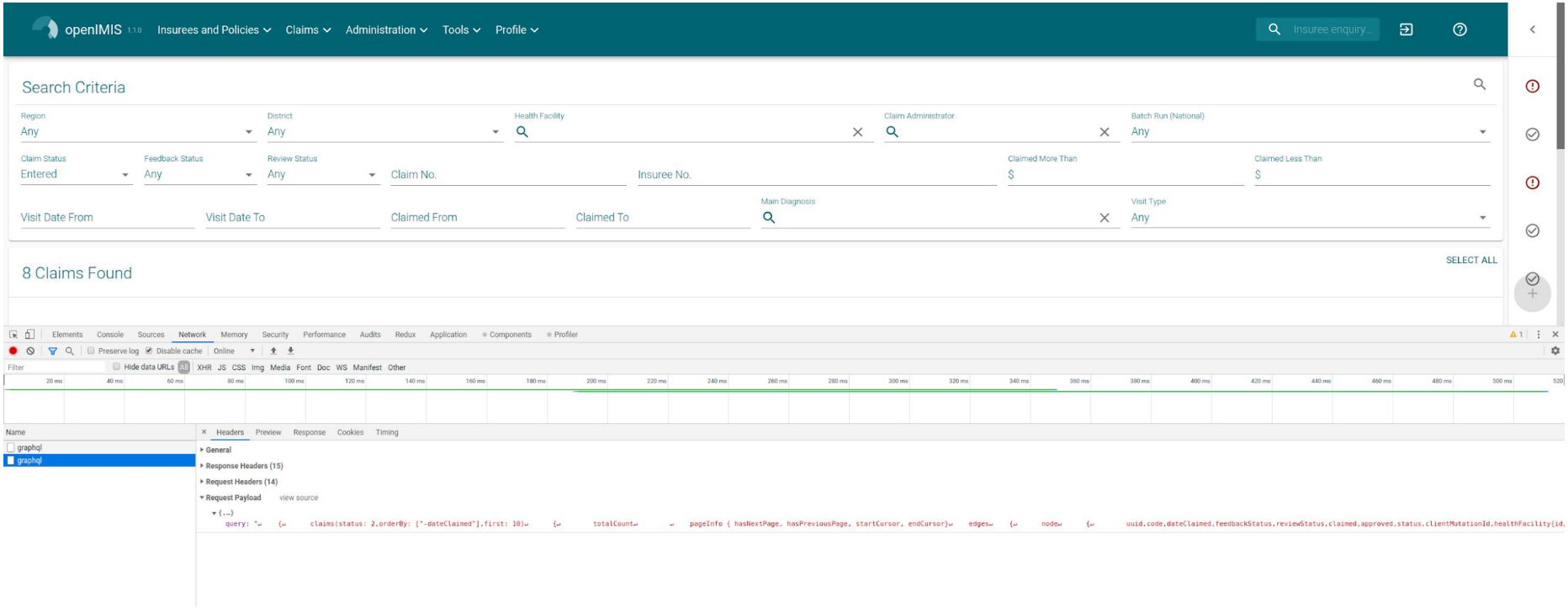
- batch run list (not cached although maybe we could/should - warning depends on selected Region/District)
- claims list (not cached, because you want fresh data on this)
- diagnosis cache for the filter
- claim administrators



When ‘coming back’ to that screen (without closing the tab),
Only 2 graphql are issued:

- batch run list (not cached)
- claims list (not cached)

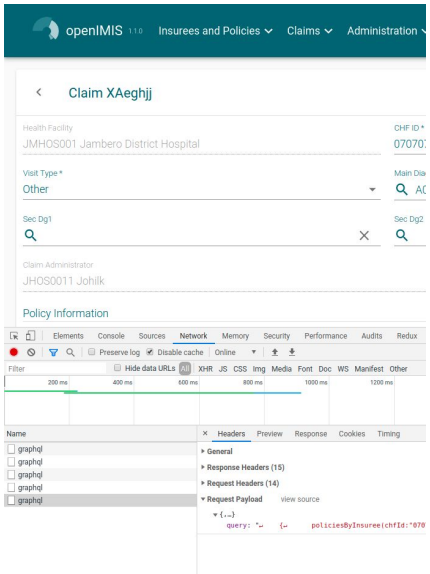
diagnosis and claim administrators are taken from the cache (saving load on server, bandwidth to client,...)



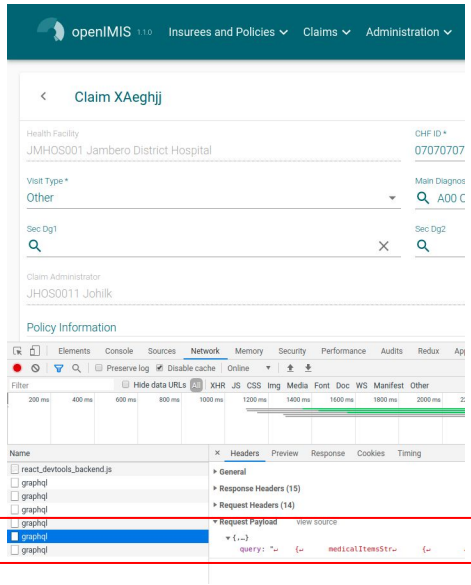
For obvious security reason, browser's tabs are **sandboxed** (don't share cache)

… so when opening a claim in a new tab, all the cache related to the claim is loaded (again) in tab's cache… **invalidating the very objective of a cache**

Opening the claim 'in app tab'
(with benefit of cache)



Opening the claim 'in new tab'
(items/services loaded in tab's cache)



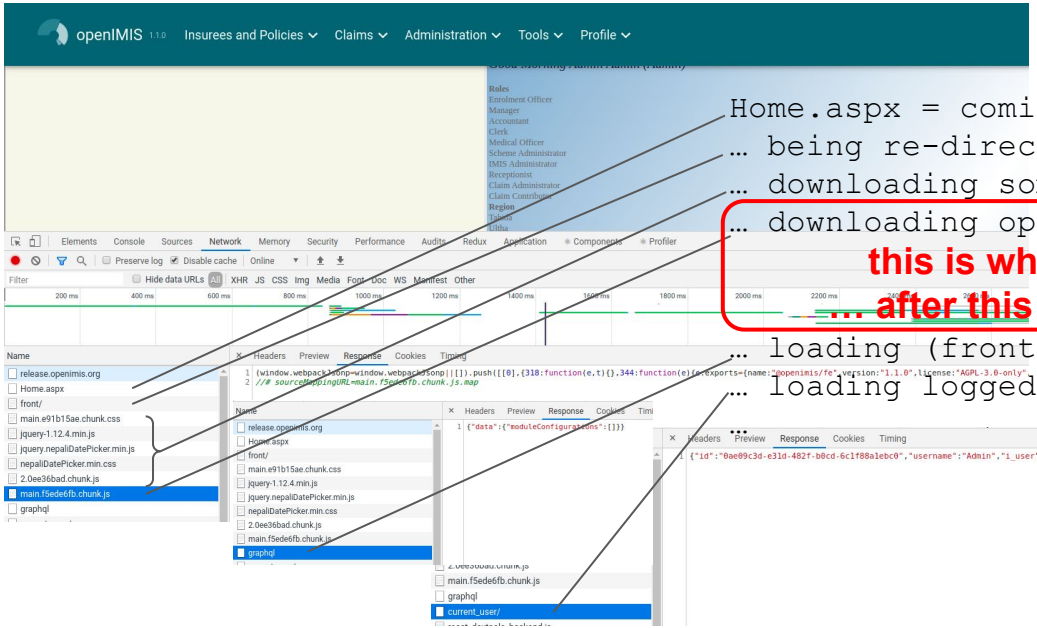
The list of items and service, pricelists,... **are loaded in every browser's tab**

Opening claims in new tabs, the ‘permalink’ problem

As just seen, one “screen” is in fact “created” from several server calls.

… but looking a little bit deeper, theses call are **just json data**!

In fact all **‘screen shells’** are loaded once, when loading the react (javascript) app



Home.aspx = coming from “legacy” login

… being re-directed to “/front” (aka ‘new openIMIS’ app)

… downloading some ‘technical dependencies’

… downloading openimis (1.1.0) frontend application

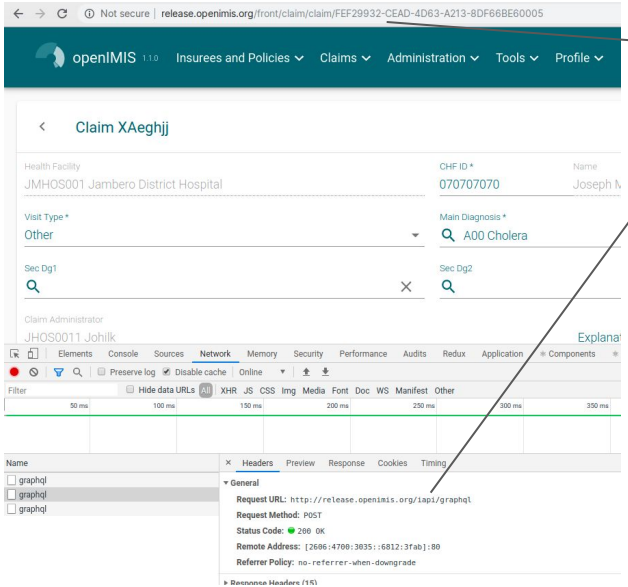
this is where all ‘screens’ (‘empty shells’) are loaded!!

… after this point, the app only loads/send (json) data

… loading (frontend) modules configuration from server

… loading logged user’s profile (rights, registered HF,...)

… **consequence:** there is **no "direct link"** between:



the "url" displayed by the browser's address bar

the requests sent to the server to actually 'feed' (display) the screen

>> This is known as React "client side" router

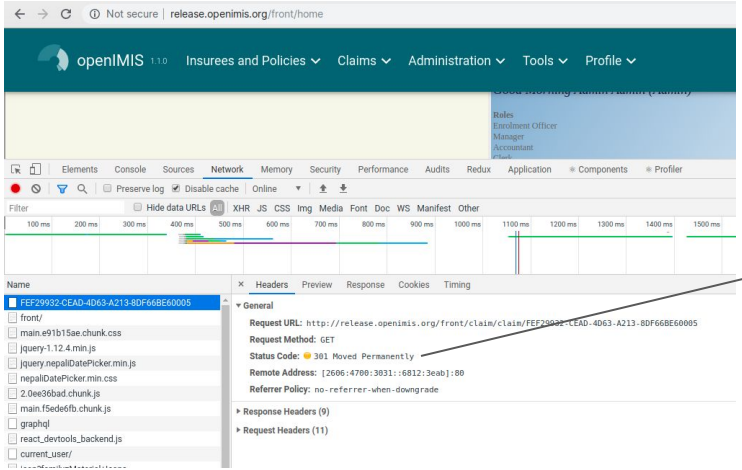
… so what happens if the user copy/paste the url into another tab?

… there is no way the new tab is already under react app's (client router) control

(it is brand new and "empty")

… so the (http) request (provided url) is **really** sent to the server

… but the server has **no clue** of what it has to do (it is used to serve json data)



… so the best it can do is **redirect to home**

… so how do we do this?

A. Find someone who has already had the problem
   … and reuse his solution
   (provided he published it open source)
   … and contribute to it as needed

B. Hack it!

C. (Really) Solve the problem
   … and package/publish it open source (for itself)

The (only) good answer

… and spend a few hours this way but:

- welcome to the (versions alignment,...) jungle
- which is the good horse to beg on (is React ‘core’ going to do something
  what about xxxx maintainability,...)

What we did

… and are **not** proud of
(I spare you the details right now, hint: openimis-fe-core_js)

What we could be doing

… but from a hack to a solution, there is work...

Opening claims in new tabs… some conclusions

No one to blame

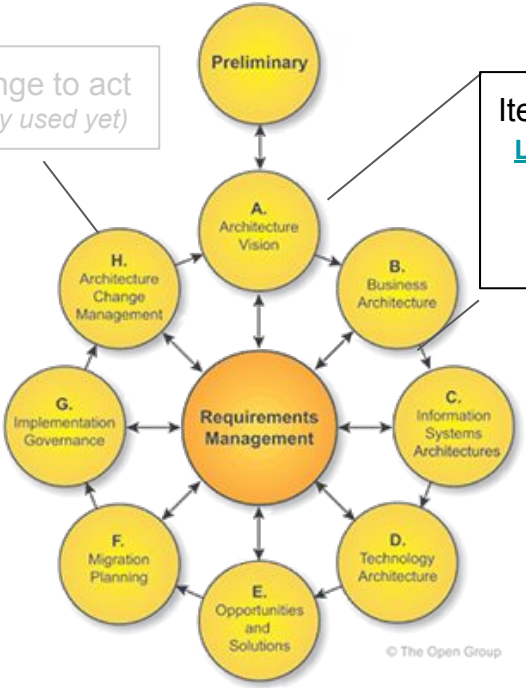
Late changes lead to hack solutions
(proper solution would have been to refactor screen and have tabs ‘inside’ was clearly out of scope)

- that **cost** a lot to put in place (all in all, spent about 1 m/d on this)**,**
- may have **performance impact** on the global solution (cache problem)
- ... and are probably **not the easiest to maintain** (permalink problem)

… but ok, that’s life (and I’ve seen worst ;-))

Iteration 4: Locations & Health Facilities Module

No architecture change to act
(but deliverable not really used yet)



Iteration 4 scope:
Locations & HF Administration

10/2019: 1,5 m/d (scope & 'analysis')
11/2019: 5 m/d Initial version Locations
12/2019: 9 m/d Initial version HFs management
01/2020: -
02/2020: -
03/2020: - 1 m/d
- 8 Tickets, nearly only UI bugs

Iteration was closed 01/03
… but was re-opened to reflect the actual work being done (// with the claim module)

Iteration is now definitively closed, with 16,5 m/d budget

Project extensions and closing activities

Allocated budget:

~~I1: **56.5** m/d + I2: **18.5** m/d + I3: **132** m/d + I4: **15,5** m/d = **222,5** m/d (on **240** m/d)~~
~~>> Closing activities have a budget of **18,5** m/d~~

reviewed

I1: **56.5** m/d + I2: **18.5** m/d + I3: **144** m/d + I4: **16,5** m/d = **235,5** m/d (on **240** m/d)
>> Closing activities have a budget of **4,5** m/d

01/2020: 3,5 m/d (Release, support to other teams, project management,…)
02/2020: 3 m/d (Nepali team support, FHIR fine grained security - started, Project management) >> 2 m/d over budget
03/2020: 5 m/d (over budget)

- FHIR fine grained security
- Nepali team support
- Increasing amount of meetings (mobile, FHIR module decomposition,...)
- Project management

TODO Bluesquare

- FHIR module fine-grained security project scope extension (we did 4 of 6-9 m/d initially estimated and we estimate the remaining to 2-3 m/d)
  With Bluesquare management agreement, we will deliver the modules by 08/04
  … but budget will be recorded on 'team support' for contract to be signed
- One (re-opened) bug to close (OMT-170 check on ceilings not fully operational) + 5 to follow-up (currently in review by Swiss TPH team)
- Project is for us closed, with **247 m/d** (3% over budget) and **respected deadlines**.
- Closing reports will be sent in the coming days.