MI4WEB

Transition phase

Projeto em Informática - Group 7
LEI / Universidade de Aveiro
2021/2022

Context and Problem

- Zero-footprint, DICOM compliant, fully web-based visualization is making its way and gaining wider acceptance within the medical imaging community.

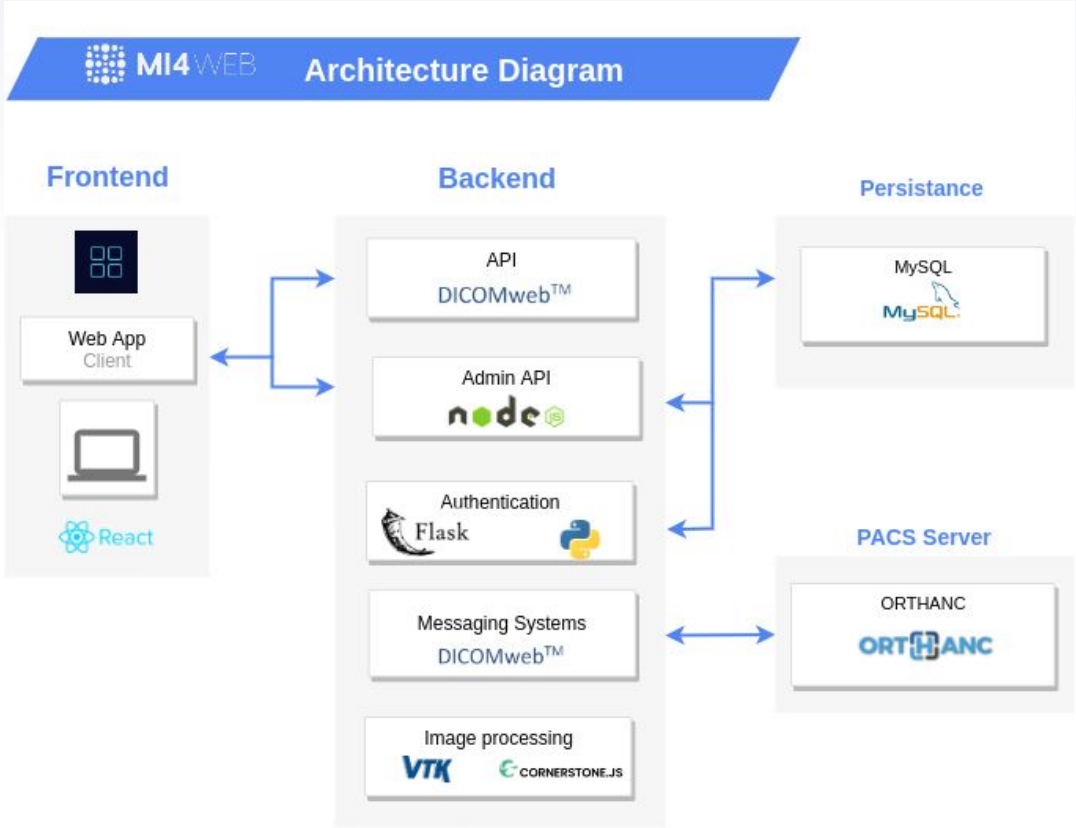
- Since the level of acceptance is growing up, the necessity of upgrading and building new functionalities and features is mandatory, so that it can continue to gain recognition and grow.

Context

- One of these applications is OHIF, a zero-footprint, open source and web-based medical imaging viewer, that gives us the ability to add and configure extensions, which makes it very expandable. Therefore, this platform was the starting point of our project.

Open Health
Imaging Foundation

System Architecture



OHIF Extension

- An extension is a group of resources that provides functionalities, interface of components, and new features.
- Advantages:
  - Isolate distinct resources
  - Allows to extend entirely different implementations
  - Flexible

- To add an extension to the viewer there is a specific structure that must be followed, in this structure, there are modules like the *Toolbar* and the *DataSource*. To activate an extension it must be registered in a specific file.

Main features

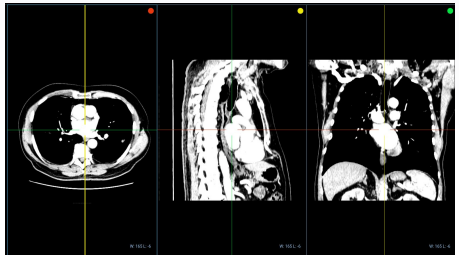
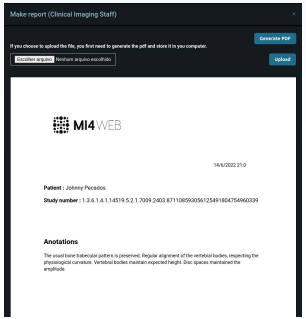


Image Reformating

Multiplaner image display functionalities

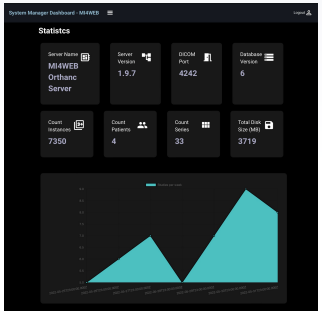
3D Views

3D Display tools as VTK



Reports

Creation of Medical Reports



Admin dashboard

Admin Interface to manage the application software and the user permissions.

Image Reformatting
Implementation

- Made using the VTK extension
- Load Data from PACS server
- Create different views
  - Axial
  - Sagittal
  - Coronal

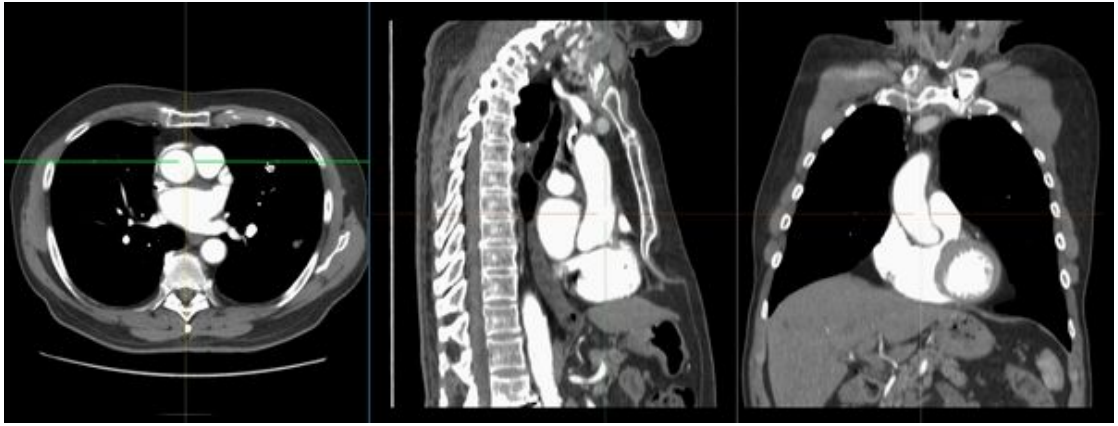
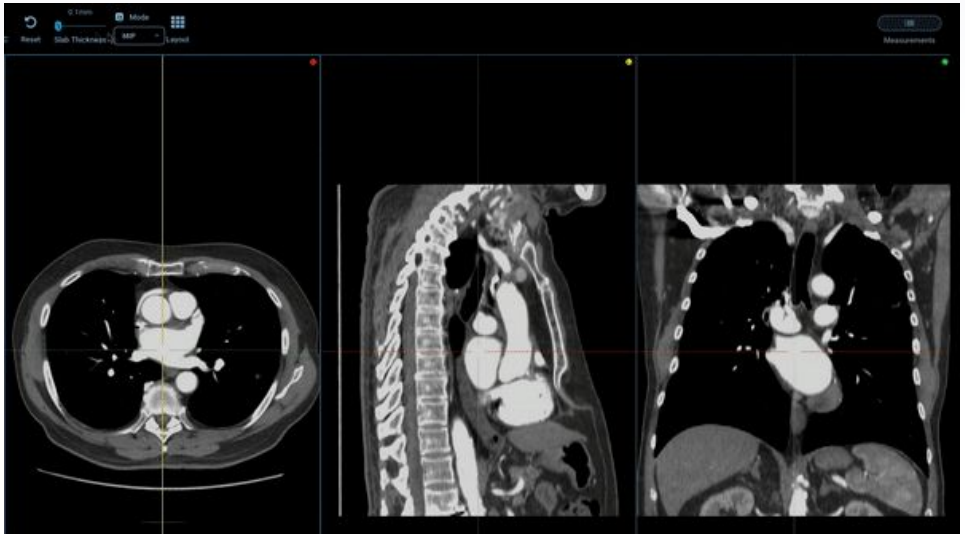


Image Reformatting
Implementation

- MIP - (Maximum Intensity Projection)
- MinIP - (Minimum Intensity Projection)

- This features were made by rendering each pixel with the maximum value in the line of sight.



3D Views
Implementation

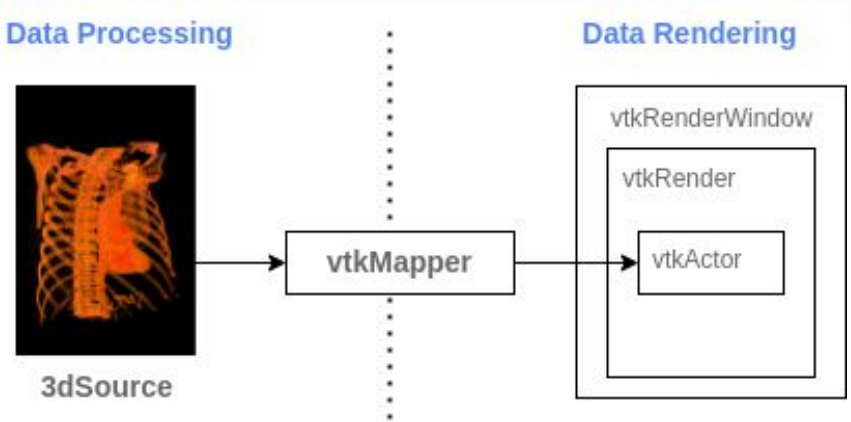
- 3D views performed on the client side using the vtk.js library

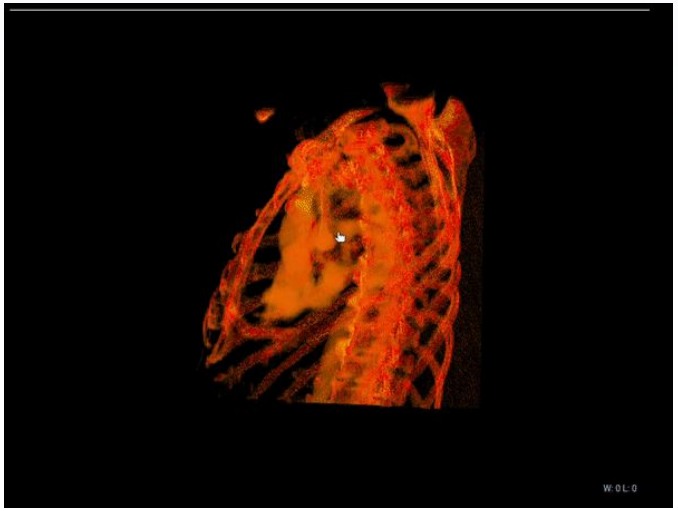
We used vtk.js library, that is a standard visualization library

- Data received from an Orthanc server, by a DICOMWeb API.

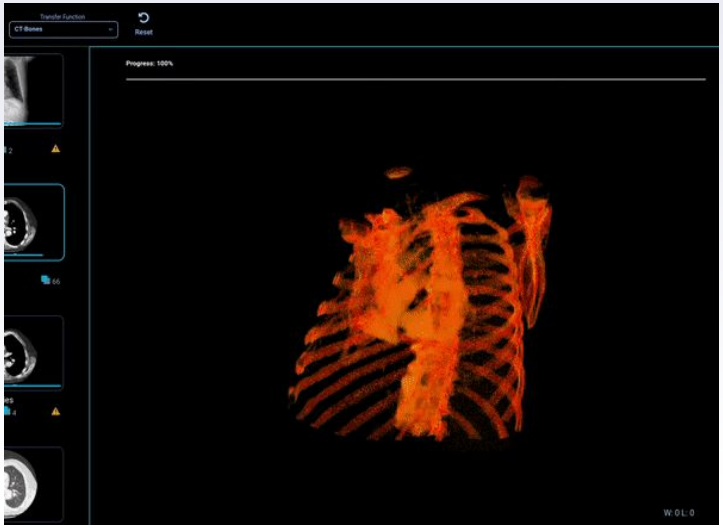
DICOMweb™ VTK ORTHANC

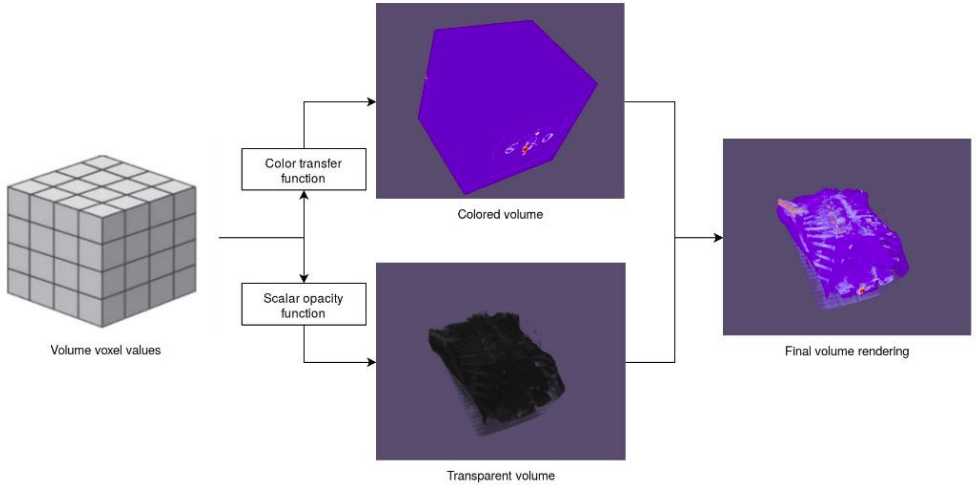
3D Views
Implementation





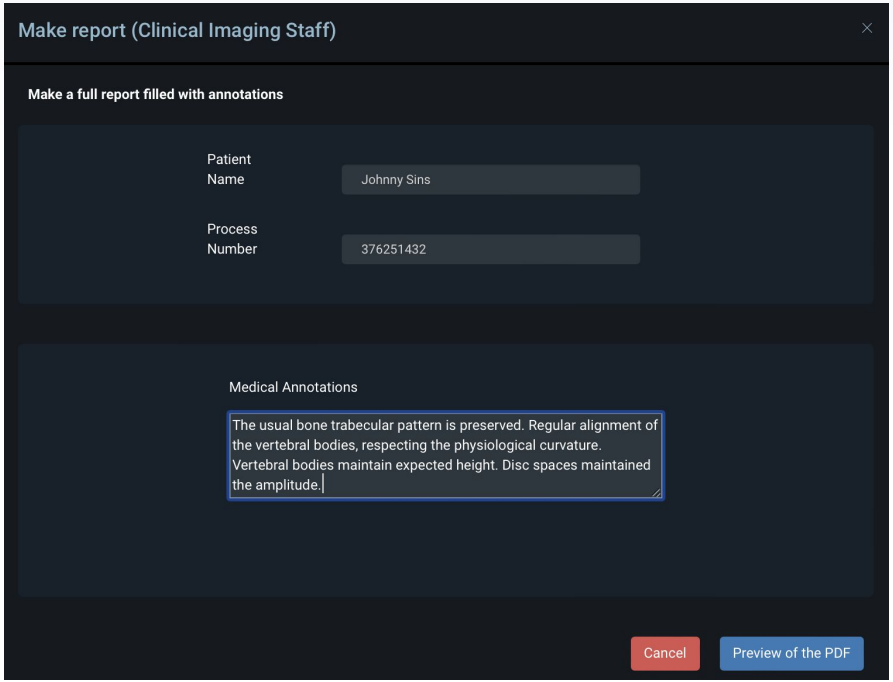
3D Views
Implementation

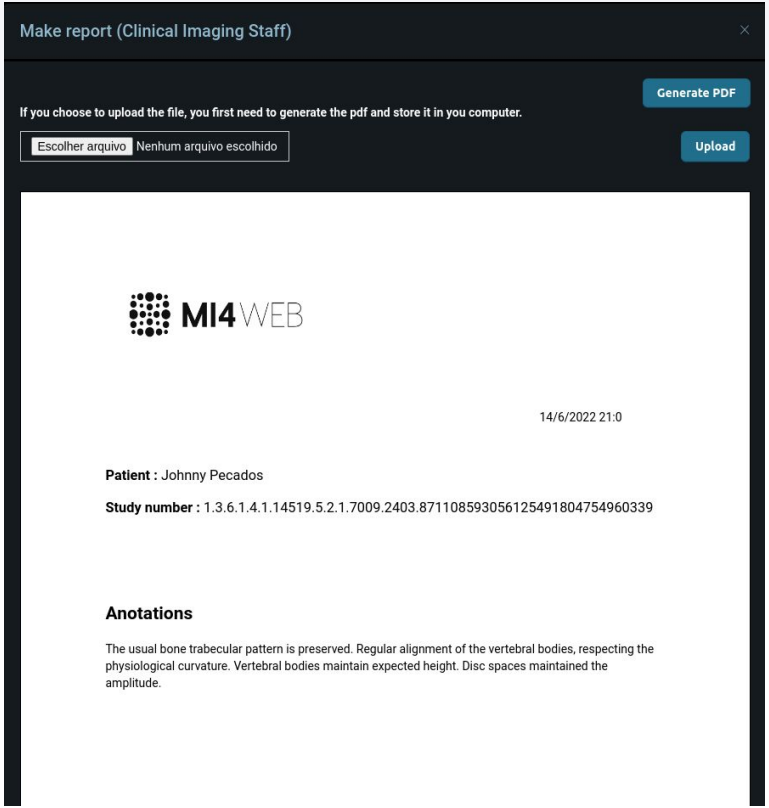




Report

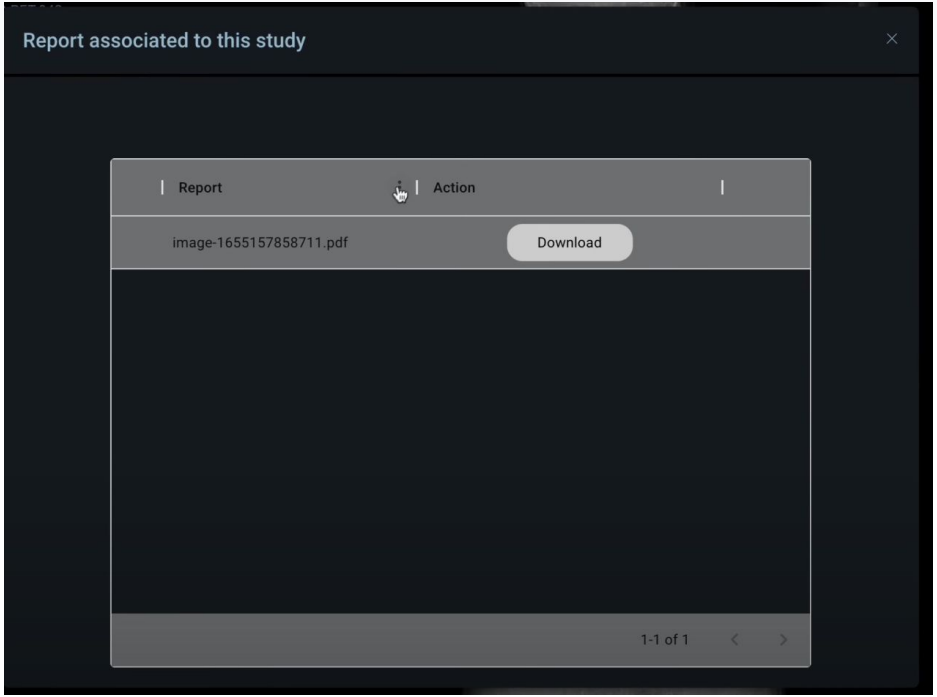
Clinical Imaging Staff





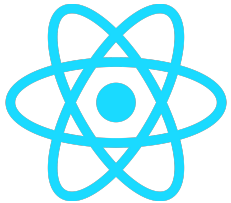
Report

Referring Imaging Staff



Report
Implementation and Decisions

- Implementation:
  - Reacts JS.
  - Database and a server to make the storage of the report.
  - API calls.
- Decisions:
  - Use DB to store the reports instead of DICOM image.



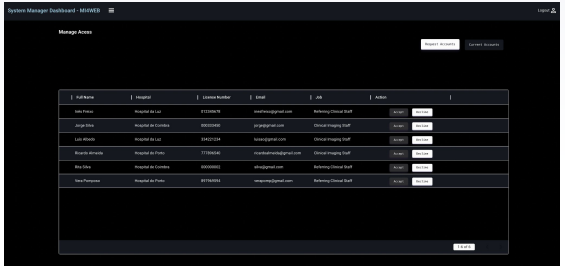
Admin
Implementation

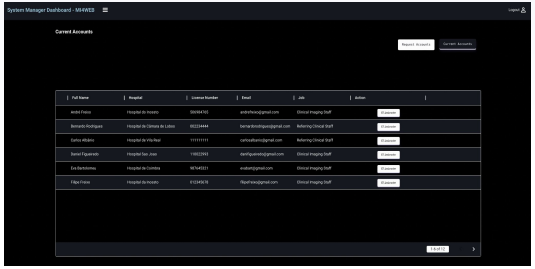
- Admin pages to manage access and collect informations about the application performance and structure using react.js.

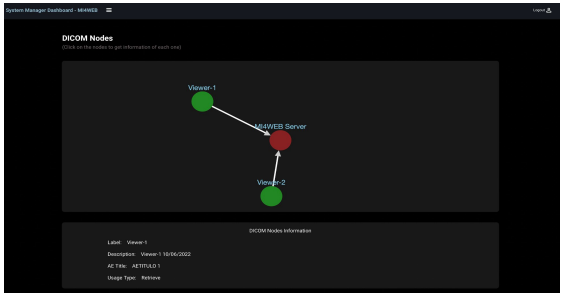
We used react.js a de facto standard

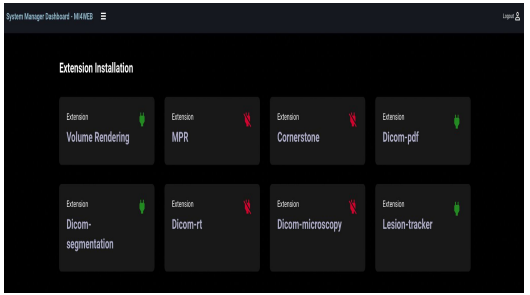
Admin
Implementation

- Dashboard
- Manage Access
- Dicom Nodes
- Extensions Installation









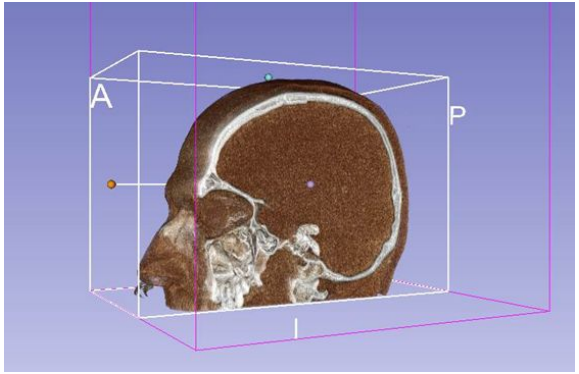
Usability Test

5 participants

Task	Difficulty (1-5)
Use mpr tool to obtain canonical and oblique views.	2.8
Use mip to see hyperdense structures.	4.2
View and manipulate 3D images.	2
Upload a report.	2.8
Find information about the server analytics.	2.8

Future Work

- Implement transfer functions on 2D views.
- Interactive 3D Clipping.
- Embed report viewing in the application.
- Increase report customization.
- Escalate the ORTHANC server.

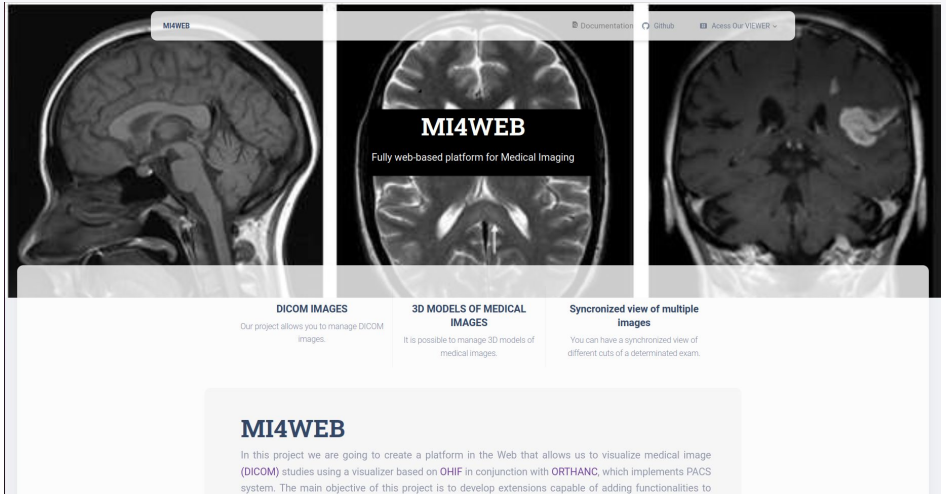


Conclusion

- Main goals were achieved and all the main features are working.
- Despite all the goals were achieved, our major difficulty was to implement the 3D functionality.
- Gained a significant amount of new knowledge (reactjs, docker, Vtk, Cornerstone, DICOM, Medical Image Concepts).

Website

https://thescorpoi.github.io/



THANKS!

Any questions?