AUT UNIVERSITY
NEW ZEALAND

COMP704 Research and Development Project

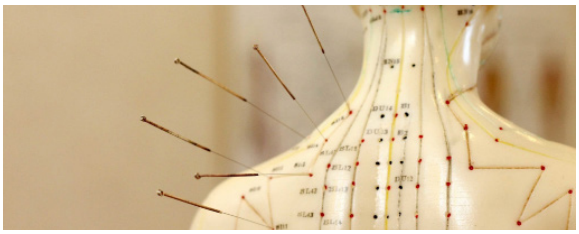
VN01 3D acupuncture healthcare data management and treatment system

Tool Feasibility Research Report

Supervisor: Dr Nhan Le Thi

Team Members:

21142643	Chuong Pham Dinh
21142377	Nhan Nguyen Cao
21142355	Tan Le Tran Ba
21142358	Trang Ho Ngoc Thao



Version: 2.0

Date: 25th February 2023

DOCUMENT VERSION CONTROL

1. DOCUMENT INFORMATION

Document code	**TFR**
Document title	**Tool Feasibility Research Report**
Version	**2.0**
Authors	**Nhan Nguyen Cao**
Distributed by	**Project VN01 team**
File name	**TFR_Tool Feasibility Research Report_2.0.pdf**
Release definition	**Only released as a finished document**

2. DOCUMENT SIGN-OFF

ID	Member	Role	Signature	Timestamp
21142355	Tan Le Tran Ba	Project Manager		26 Feb 2022 15:25

3. DOCUMENT VERSIONS

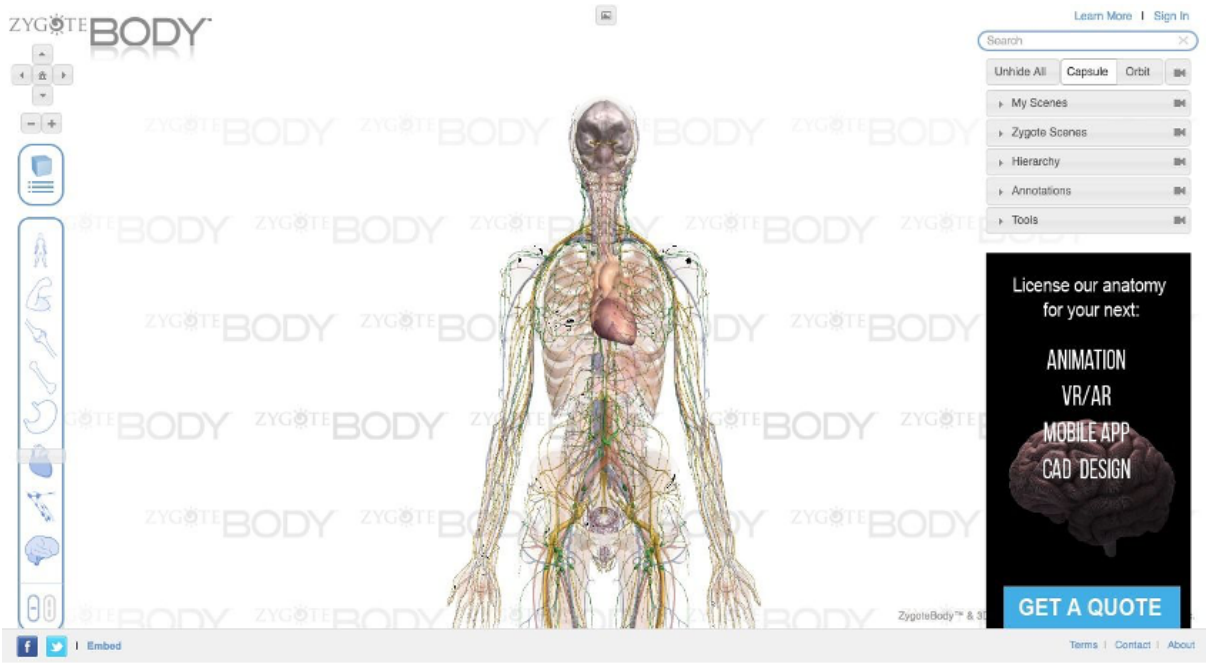
Version	Timestamp	Description	Responsible members
1.0	13 Nov 2022 22:38	First version of the Tool Feasibility Research Report document, focus on rating the feasibility of the Tools involved in the first step of the project	**Nhan Nguyen Cao** *(21142377)* **Trang Ho Ngoc Thao** *(21142358)*
2.0	25 Feb 2023 20:45	Upgraded version of the Tool Feasibility Research Report document, following the change in Hosting Service to use after Nest.js failed to be deployed into Vercel	**Nhan Nguyen Cao** *(21142377)* **Trang Ho Ngoc Thao** *(21142358)*

Tool Feasibility Research Report

Tool	Feasibility Evaluation
Three.js	Three.js supports the rendering of 3D model, as well as points and lines into the view space. Researched on some demo projects of Three.js, there was a famous project of Zygote Body (developed from the original Google Body) that serves quite similar features as the expectation for this project. Therefore, it can be concluded that this tool is feasible to build the basic features required for the project: Rendering the Human Body 3D model and mark some basic items Noted: It depends on whether there exists a 3D model of human body as expected. When the asset is available, rendering it out and setting the camera, handle the interactions similar to Zygote Body is in scope of Three.js. Feasibility: **Yes**
Jest	There are multiple methods available for unit testing with Jest. After research, we decided to follow this tutorial for mocking and covering the unit tests: https://betterprogramming.pub/testing-controllers-in-nestjs-and-mongo-with-jest-63e1b208503c Basically, we would write unit tests to cover all endpoints defined in Controllers and Services. While the connection to MongoDB database would be mocked and simulated. The mock data would be defined by the developer, and based on the type of HTTP request within the endpoint, the corresponding action would be examined and asserted. Feasibility: **Yes**

Tool	Feasibility Evaluation
Puppeteer	Puppeteer supports running the application on headless Chrome browser. However, it could be a problem if we tried to use Puppeteer for running end-to-end tests on pages that required loading. After research, we decided to change a little bit our approaches for applying Puppeteer: - Only limited interactions that directly triggering the HTML elements (all Three.js rendered elements are privately managed in a canvas loaded from third party, so those are not accessible). So automation tests would be on those elements only. For example, triggering the change in manual control panel and assert the changes in the viewspace. - Since some interactions cannot be asserted by code, snapshot can be used. Screenshot of the headless browser at that stage would be stored as an artifact of the test. Manually evaluated the artifacts later to consider whether the test passed or failed. - API cannot be mocked in Automation test, unlike Unit test. Calling raw API returned false due to missing authorization and authentication. Instead, all pages with API would just be examined taking into account the state of not having API. The remaining flows would be covered by unit testing and integration testing. Feasibility: **Partial, have to limit the expectations for performing automation testing**
CI/CD	CD flow for Firebase Hosting can be implemented based on the following documentation: https://circleci.com/docs/deploy-to-firebase/. However, there was some problem with the tutorial that we failed to follow. Further research suggested the use of an old method (using Firebase Token). We tested and it worked as expected. Note: This was announced to later be deprecated, so just used as a temporary solution. For CD to Heroku, we followed the tutorial https://circleci.com/docs/deploy-to-heroku/ and it worked as expected. For deploying CD from different branches to different server, we examined the use of —P flag while calling firebase deploy command in CI/CD configuration. It worked as expected. Feasibility: **Partial, have to use a temporary solution for Front-end, which will later be deprecated.**
PyMongo	What we need for the Data Integration: Receive data from Spreadsheet (directly from workspace or download and reupload) and add them into MongoDB. Followed the documentation by PyMongo and it worked as expected https://pymongo.readthedocs.io/en/stable/. Since only basic queries are needed (mostly Insert or Delete, in case of wrong insertion), and all of them worked as expected. **Feasibility: Yes**

Tool	Feasibility Evaluation
Vercel	During Research, the site deployed to staging ran successfully. Except for the failed to load Swagger page, with error message relating to the unability to navigate to Swagger theme library. Followed the guide: https://dev.to/leduc1901/how-to-deploy-your-nestjs-apps-on-vercel-3nh9 **Feasibility: Yes** However, later on, after one PR, the site just failed to deploy. We did a quick check all past class projects involving Nest.js hosted on Vercel and all of them failed. We researched but unabled to find a working solution. **Feasibility: No**
Heroku	We tried to apply Github Student account and was granted. Then applied for a free Student tier on Heroku. After waiting for 24 hours, it was approved and we had one instance for hosting. We followed the guide: https://circleci.com/blog/continuous-deployment-nestjs/, and it worked as expected. Luckily, the Swagger page is also visible. **Feasibility: Partial, had to merge staging and live server into one since only one free plan is available.**



Zygote Body