Une image contenant personne

Description générée automatiquement

|  |
| --- |
| RÉfÉrentiels  **Création de profils FHIR avec SUSHI**  29/12/2022 |

Sommaire

[Introduction 3](#_Toc123310169)

[Installation de Node.js 3](#_Toc123310170)

[Installation de SUSHI 3](#_Toc123310171)

[Téléchargement d’un exemple de projet FSH : squelette à suivre 4](#_Toc123310172)

[Exécution de SUSHI 4](#_Toc123310173)

[Génération de l’IG 5](#_Toc123310174)

[Définition des cardinalités dans un profil 6](#_Toc123310175)

[Création d’extension pour FishPatient 6](#_Toc123310176)

[Définition de Value Set pour les espèces poissons 7](#_Toc123310177)

[Définition d’alias 7](#_Toc123310178)

[Création d’une instance de FishPatient 8](#_Toc123310179)

[Étendre le profil du Veterinarian 8](#_Toc123310180)

[Exemple de projet FHIR créé avec SUSHI et Node.js 9](#_Toc123310181)

[ Page Home 9](#_Toc123310182)

[ Page Profiles 9](#_Toc123310183)

[ Page Artifacts 10](#_Toc123310184)

[ Page Extensions 11](#_Toc123310185)

[ Page Terminologies 11](#_Toc123310186)

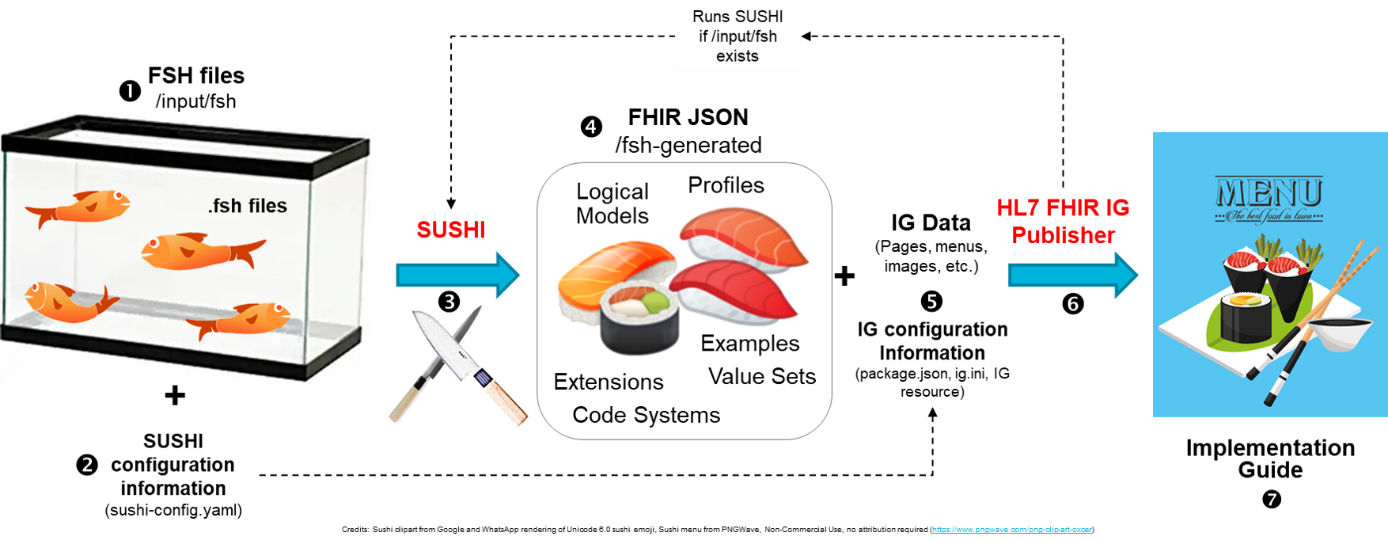
[ Page Table of Contents 12](#_Toc123310187)

[Conclusion 12](#_Toc123310188)

# Introduction

FHIR Shorthand (FSH) est un langage spécialement conçu pour définir le contenu des guides de mise en œuvre FHIR (IG). Il est simple et compact, avec des outils pour produire des profils, des extensions et des IG Fast Healthcare Interoperability Resources (FHIR). FSH est compilé à partir de fichiers texte vers des artefacts FHIR à l'aide de SUSHI .

Pour commencer à utiliser FSH, on doit installer et exécuter SUSHI en suivant plusieurs étapes qui sont expliquées au fur et à mesure au fil de ce document.



# Installation de Node.js

SUSHI nécessite Node.js. Pour installer Node.js, rendez-vous sur <https://nodejs.org/>et on sélectionne le téléchargement « LTS ». Si le téléchargement ne convient pas au système d'exploitation, on clique sur le lien "Autres téléchargements" pour obtenir une liste complète des téléchargements. Une fois le programme d'installation est téléchargé, exécutez-le en utilisant les options par défaut. Il faut s’assurer que Node.js est correctement installé en ouvrant une fenêtre de commande « cmd » et en tapant les deux commandes ci-dessous. Chaque commande doit renvoyer un numéro de version.

|  |
| --- |
| $ node --version |
| $ npm --version |

# Installation de SUSHI

Pour installer SUSHI, il faut ouvrir une invite de commande « cmd » et on tape la commande suivante:

|  |
| --- |
| $ npm install -g fsh-sushi |

Après installation de SUSHI on lance la commande ci-dessous pour vérifier son installation :

|  |
| --- |
| $ sushi -h |

# Téléchargement d’un exemple de projet FSH : squelette à suivre

Pour commencer avec un exemples de travail de fichiers FSH et un squelette de projet FSH, on accède au lien suivant « <https://fshschool.org/docs/sushi/tutorial/fsh-tutorial-master.zip> », on procède au téléchargement et on le décompresse dans un répertoire bien défini.

Une fois le fichier décompressé, le dossier contient deux sous-répertoires :

* FishExample : Squelette
* FishExampleComplete : Exemple Complet

On accède au répertoire de travail FishExample, il existe un fichier sushi-config.yaml et deux fichiers FSH dans le sous-répertoire /input/fsh : FishPatient.fsh Veterinarian.fsh

De plus, il existe plusieurs fichiers pour la construction de l'IG.

# Exécution de SUSHI

Après l’installation de SUSHI et un squelette de projet FSH, on ouvre une fenêtre en ligne de commande « cmd » et on accède au répertoire FishExample . On Exécute après SUSHI sur ces fichiers FSH en exécutant :

|  |
| --- |
| $ sushi . |

L'exécution de SUSHI va créer le répertoire **FishExample/fsh-generated** et le remplira avec les fichiers nécessaires pour créer l'IG à l'aide de l'outil HL7 FHIR IG Publisher.

Lors de l'exécution de SUSHI, il y aura une série de messages d'information. Une fois SUSHI terminé, une sortie récapitulative semblable à celle-ci apparaîtra :

|  |
| --- |
| ╔════════════════════════ SUSHI RESULTS ══════════════════════════╗  ║ ╭───────────────┬──────────────┬──────────────┬───────────────╮ ║  ║ │ Profiles │ Extensions │ Logicals │ Resources │ ║  ║ ├───────────────┼──────────────┼──────────────┼───────────────┤ ║  ║ │ 2 │ 0 │ 0 │ 0 │ ║  ║ ╰───────────────┴──────────────┴──────────────┴───────────────╯ ║  ║ ╭────────────────────┬───────────────────┬────────────────────╮ ║  ║ │ ValueSets │ CodeSystems │ Instances │ ║  ║ ├────────────────────┼───────────────────┼────────────────────┤ ║  ║ │ 0 │ 0 │ 0 │ ║  ║ ╰────────────────────┴───────────────────┴────────────────────╯ ║  ║ ║  ╠═════════════════════════════════════════════════════════════════╣  ║ Fin-tastic job! 0 Errors 0 Warnings ║  ╚═════════════════════════════════════════════════════════════════╝ |

Les fichiers générés par SUSHI se trouvent dans le répertoire FishExample/fsh-generated . Il faut vérifier aussi que le sous-répertoire FishExample/fsh-generated/resources contient deux StructureDefinitions générées et une ImplementationGuide générée.

# Génération de l’IG

Ouvrir la ligne de commande « cmd » et taper :

|  |
| --- |
| $ \_updatePublisher |

En tapant cette commande, le téléchargement de la dernière version de l'outil HL7 FHIR IG Publisher dans ./input-cache s’effectue.

Après avoir exécuter cette commande, il faut lancer dans l’étape d’après :

|  |
| --- |
| $ \_genonce |

Cela exécutera le générateur HL7 FHIR IG, ce qui peut prendre plusieurs minutes. Une fois l'éditeur terminé, ouvrir le fichier /FishExample/output/index.html pour voir l'IG résultant.

Si on clique sur l'élément résumé des artefacts dans le menu, on verra que l'IG contient deux profils, FishPatient et Veterinarian. Si on regarde chacun d'eux, on remarquera qu'ils ont des différentiels minimes. La seule façon dont ils diffèrent de leur ressource de base est qu'ils nécessitent au moins un nom.

# Définition des cardinalités dans un profil

FHIR est conçu pour être utilisé aussi bien en médecine vétérinaire qu'humaine. Pour un patient non humain, il faut enregistrer l'espèce. Les patients de ce tutoriel seront d’espèces poissons.

Étant donné que les poissons ne se marient pas légalement (bien que certaines espèces créent des liens de couple) et qu'ils ne communiquent pas dans un langage humain, la première chose que nous ferons dans le profil FishPatient est d'éliminer ces éléments. Pour ce faire, il faut ouvrir le fichier FishPatient.fsh dans l’éditeur de texte et ajouter les règles suivantes après la dernière ligne non vide du fichier :

|  |
| --- |
| \* maritalStatus 0..0  \* communication 0..0 |

* Les règles commencent par \*. FSH exprime la cardinalité comme, {min}..{max}, identique à FHIR.

# Création d’extension pour FishPatient

Pour préciser l'espèce aquatiques de nos patients, nous avons besoin d'une **extension** .

Nous avons le choix de rendre l'extension **en** **ligne** ou **autonome** . Les extensions en ligne ne donnent pas lieu à une StructureDefinition distincte et ne peuvent pas être réutilisées dans d'autres profils. Les extensions autonomes sont à l'opposé : elles ont des StructureDefinitions indépendantes et peuvent être réutilisées. Pour cet exemple, on va créer une extension autonome. Les extensions sont créées à l'aide du mot clé « contains ». Pour ajouter une extension d'espèce autonome, ajouter la règle suivante après les règles de cardinalité :

|  |
| --- |
| \* extension contains FishSpecies named species 0..1 |

Cette règle stipule que **l’extension** tableau de la ressource Patient intégrera l' **FishSpecies** extension avec le nom local **species**.

Pour définir l' **FishSpecies** extension, il faut ajouter les lignes suivantes à la fin du fichier **FishPatient.fsh** :

|  |
| --- |
| Extension: FishSpecies  Id: fish-species  Title: "Fish Species"  Description: "The species of the fish." |

On exécute à nouveau SUSHI ( sushi .). Le nombre d'extensions devrait maintenant être 1.

# Définition de Value Set pour les espèces poissons

L'extension « FishSpecies » ne fait pas encore tout à fait son travail, car nous n'avons pas spécifié le type de valeurs qu'elle pourrait accepter. Pour ajouter ces informations, entrez ces lignes après la description de FishSpecies :

|  |
| --- |
| \* value[x] only CodeableConcept  \* valueCodeableConcept from FishSpeciesValueSet (extensible) |

La première règle limite la valeur [x] (un élément intégré de chaque extension FHIR) à un « CodeableConcept » utilisant le mot clé **only**. La seconde le lie à un ensemble de valeurs (encore à définir) à l'aide du mot clé **from**. La force de liaison sera extensible, ce qui signifie que les codes de « FishSpeciesValueSet » doivent être utilisés sauf lorsqu'il manque une espèce de poisson spécifique dans l'ensemble de valeurs.

Pour définir « FishSpeciesValueSet », on ajoute les lignes suivantes au même fichier :

|  |
| --- |
| ValueSet: FishSpeciesValueSet  Title: "Fish Species Value Set"  Id: fish-species-value-set  Description: "Codes describing various species of fish, taken from SNOMED-CT."  \* codes from system http://snomed.info/sct where concept is-a SCT#90580008 "Fish (organism)" |

Il faut après exécuter à nouveau SUSHI. Le nombre de ValueSets devrait maintenant être 1. Générer l'IG en exécutant **\_genonce** à nouveau. Il faut au final ouvrir le fichier **/FishExample/output/index.html** pour voir l'IG résultant.

# Définition d’alias

An **Alias** est un moyen de définir un raccourci pour une URL ou un OID. Par exemple, dans la définition de l'ensemble de valeurs ci-dessus, nous aurions pu définir un **Alias** pour faciliter le référencement de l' <http://snomed.info/sct> URL, comme indiqué ci-dessous. Les alias sont classiquement définis en haut du fichier.

On ajoute cette ligne en haut du fichier FishPatient.fsh :

|  |
| --- |
| Alias: SCT = <http://snomed.info/sct> |

puis on remplace la dernière ligne de FishSpeciesValueSet par :

|  |
| --- |
| \* codes from system SCT where concept is-a #90580008 "Fish (organism)" |

# Création d’une instance de FishPatient

Chaque IG devrait fournir des exemples de ses profils. Faisons cela en introduisant un exemple FishPatient nommé « **Shorty »**. On Crée cet exemple d'instance à l'aide du mot clé **Instance**, avec **InstanceOf** set to **FishPatient** et **Usage** set to **Example**.

Inclure après les informations suivantes sur « **Shorty »** dans l'instance :

|  |
| --- |
| * Son prénom (prénom) est "Shorty" et son nom de famille (dernier) est "Koi-Fish". * Shorty est un poisson Koi (Cyprinus rubrofuscus), représenté sous le code SNOMED-CT 47978005 "Carpiodes cyprinus (organisme)". |

* En cas de besoin d'aide pour cela, on peut se référer à la section **Définition des instances** de la spécification et on peut aussi consulter les fichiers FSH dans le répertoire **FishExampleComplete**.

Il faut exécuter à nouveau SUSHI et régénérez l'IG.

# Étendre le profil du Veterinarian

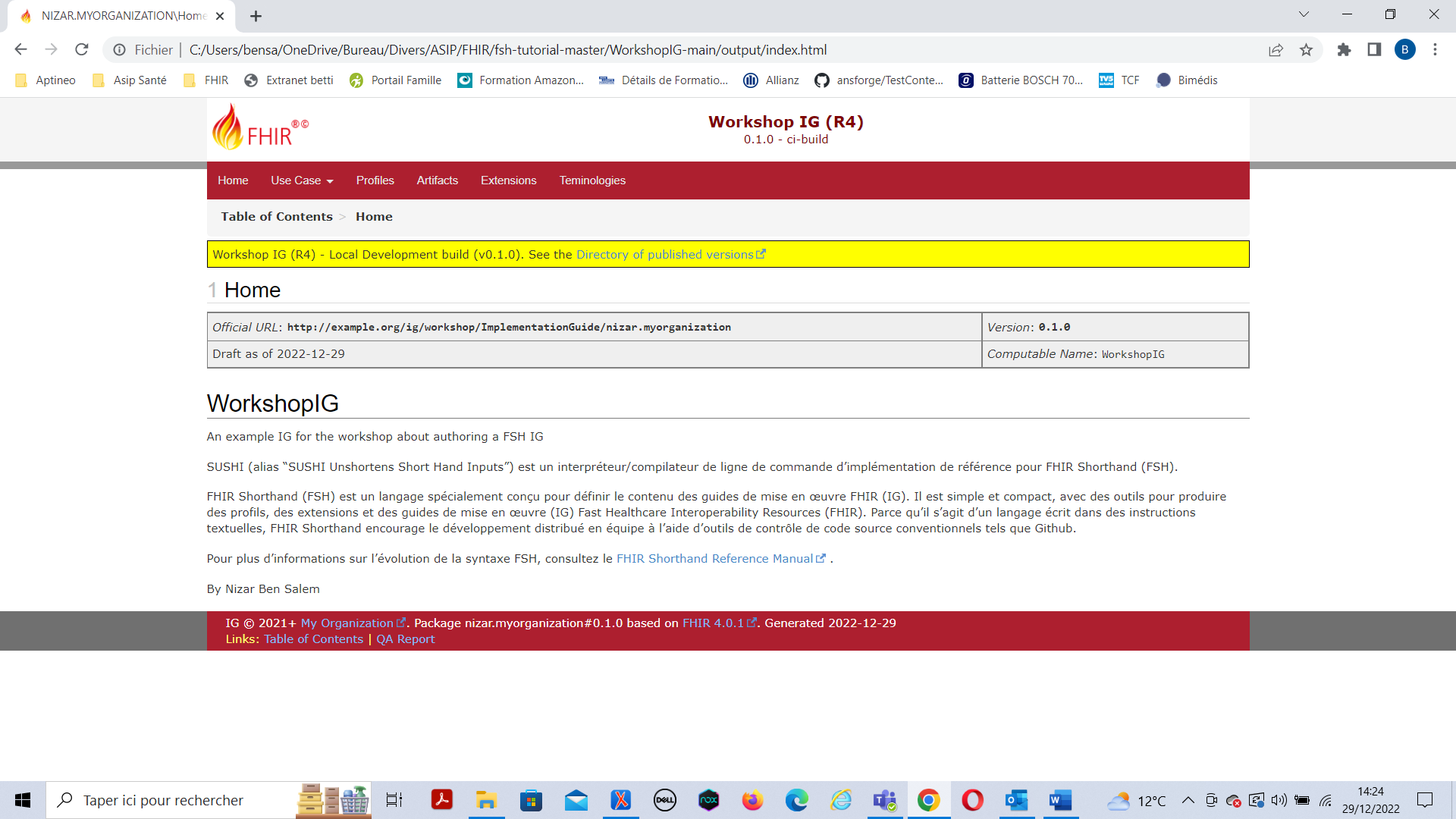
Ajouter des contraintes et/ou des extensions au profil Vétérinaire : Ajouter des qualifications compatibles avec une pratique « Veterinarian ». Les qualifications sont tirées du système de codes <http://nucc.org/provider-taxonomy>, et le code est **174M00000X**, pour " Veterinarian ".

De plus, il faut découper le **identifier** baie en tranches, ce qui rend un numéro de licence requis. Le code système est <http://terminology.hl7.org/CodeSystem/v2-0203> et le code est **LN**, pour "Numéro de licence".

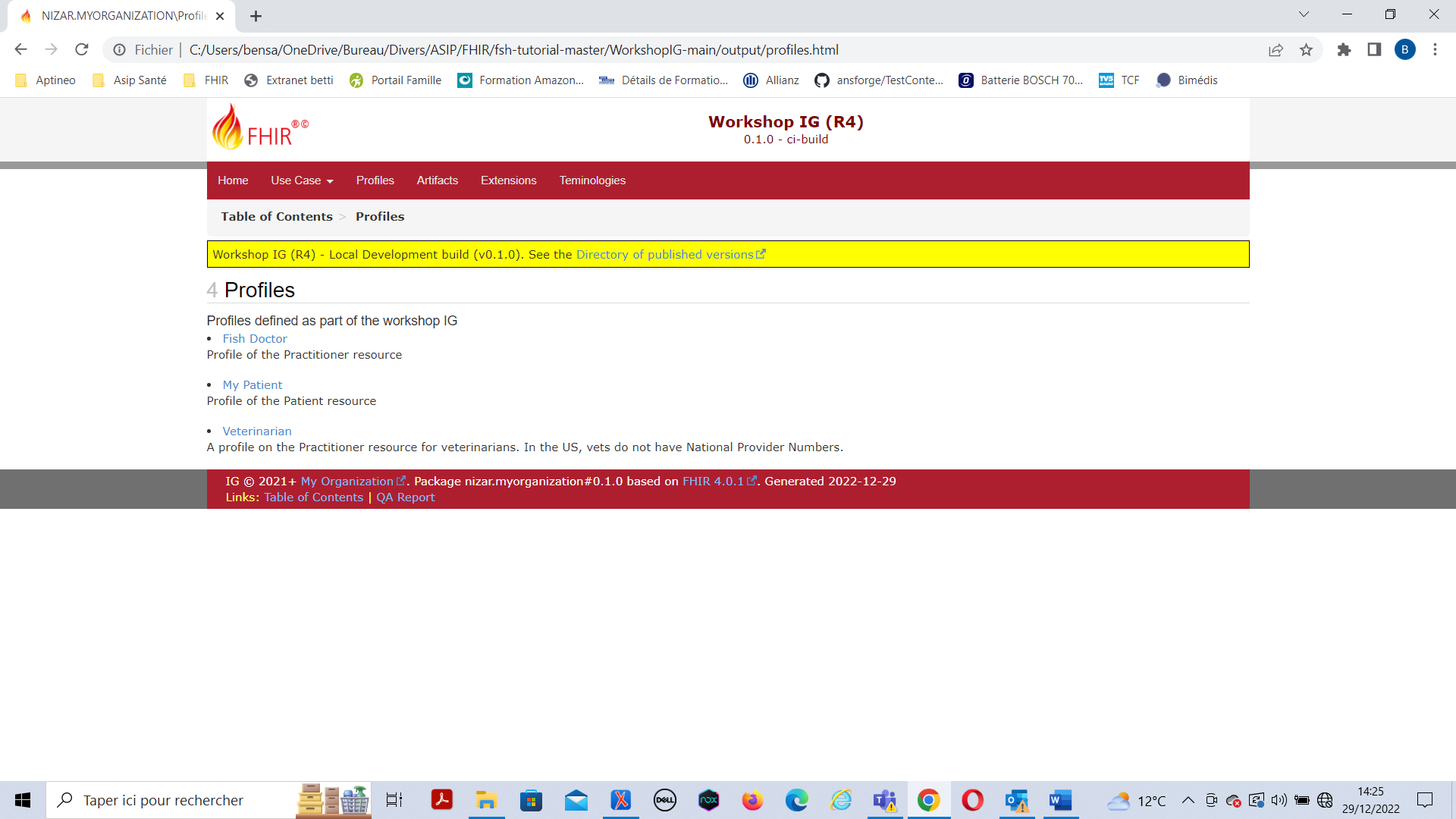
Il faut exécuter à nouveau SUSHI et régénérez l'IG.

# Exemple de projet FHIR créé avec SUSHI et Node.js

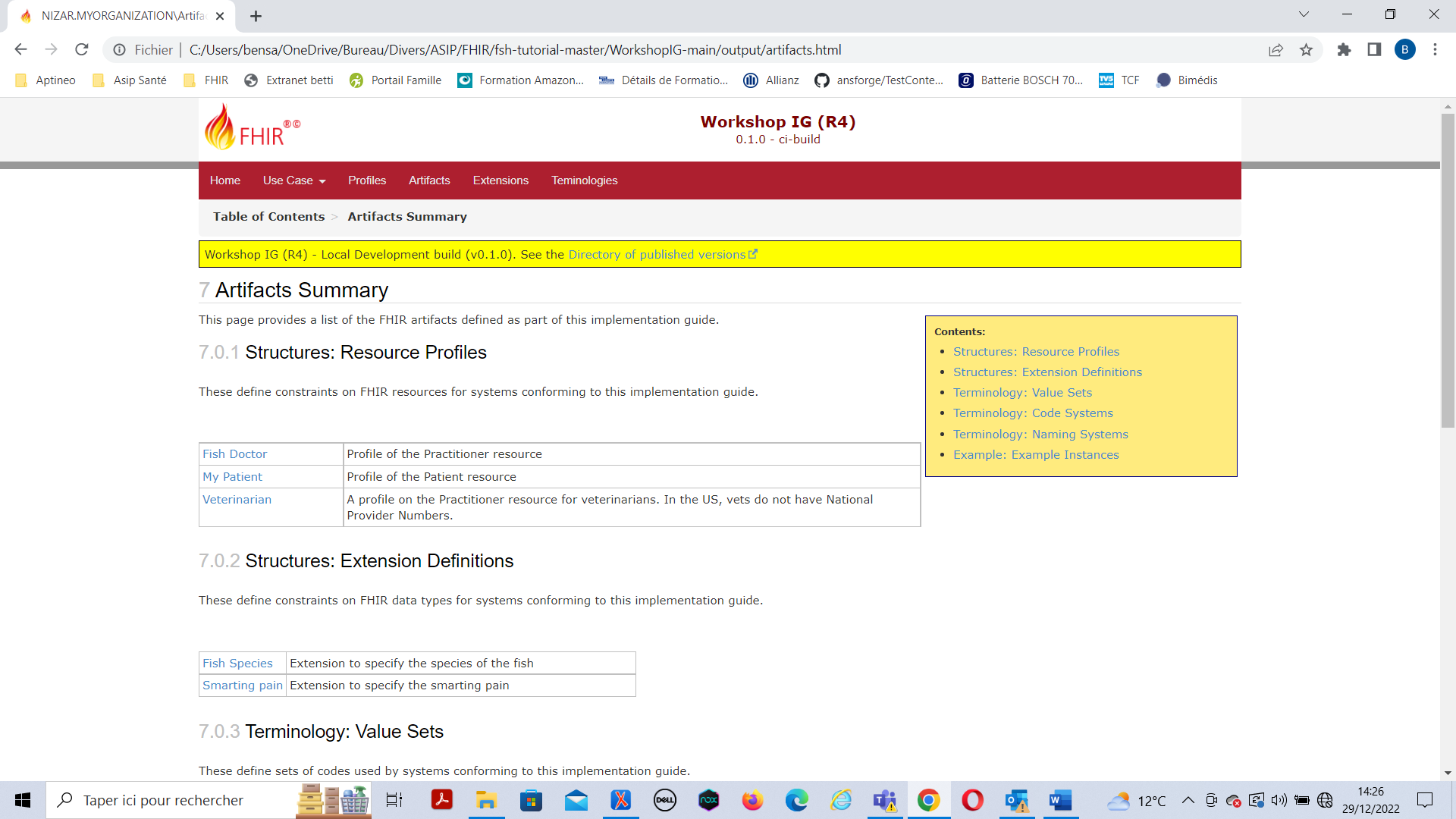
## Page Home

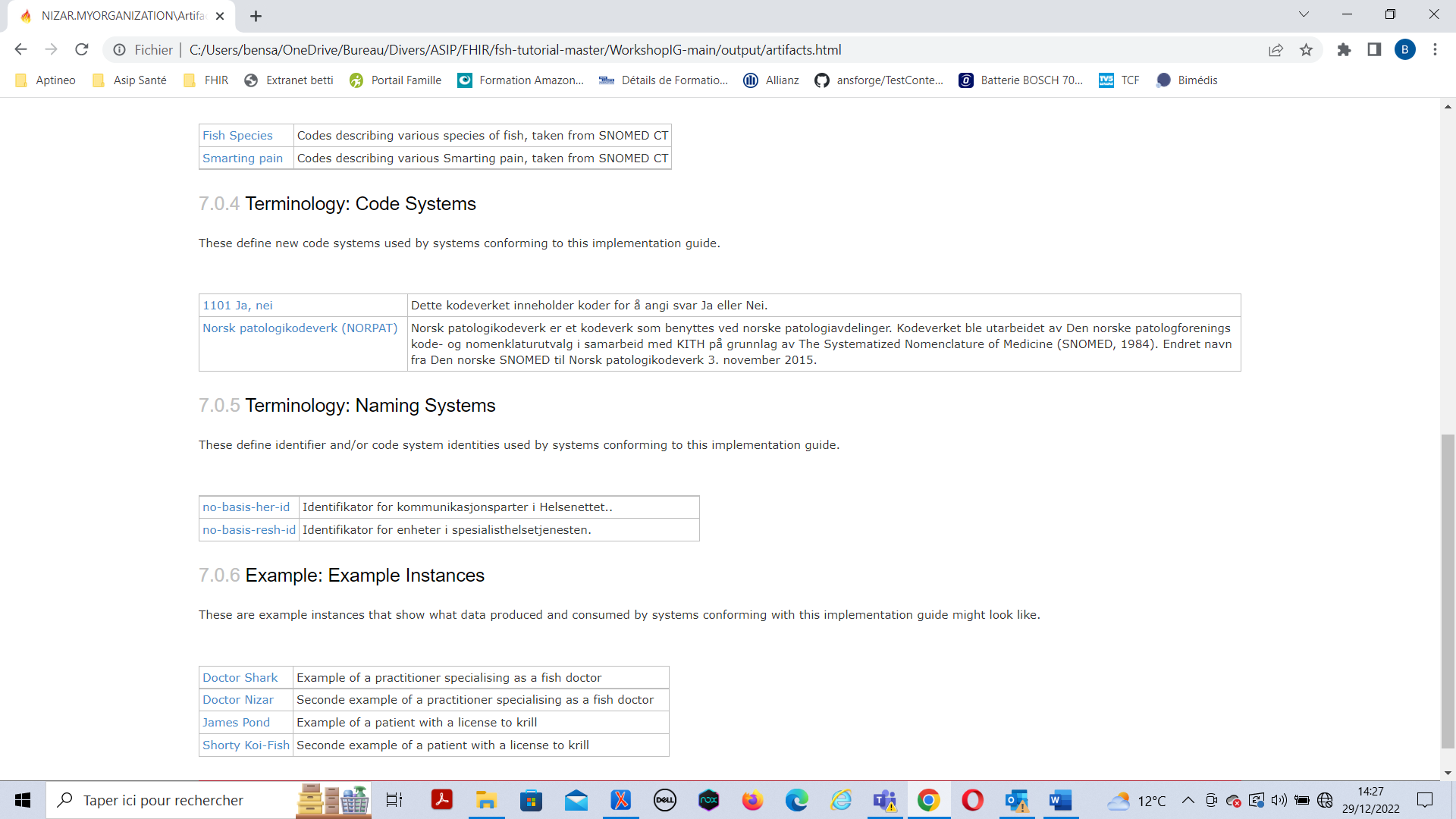


## Page Profiles

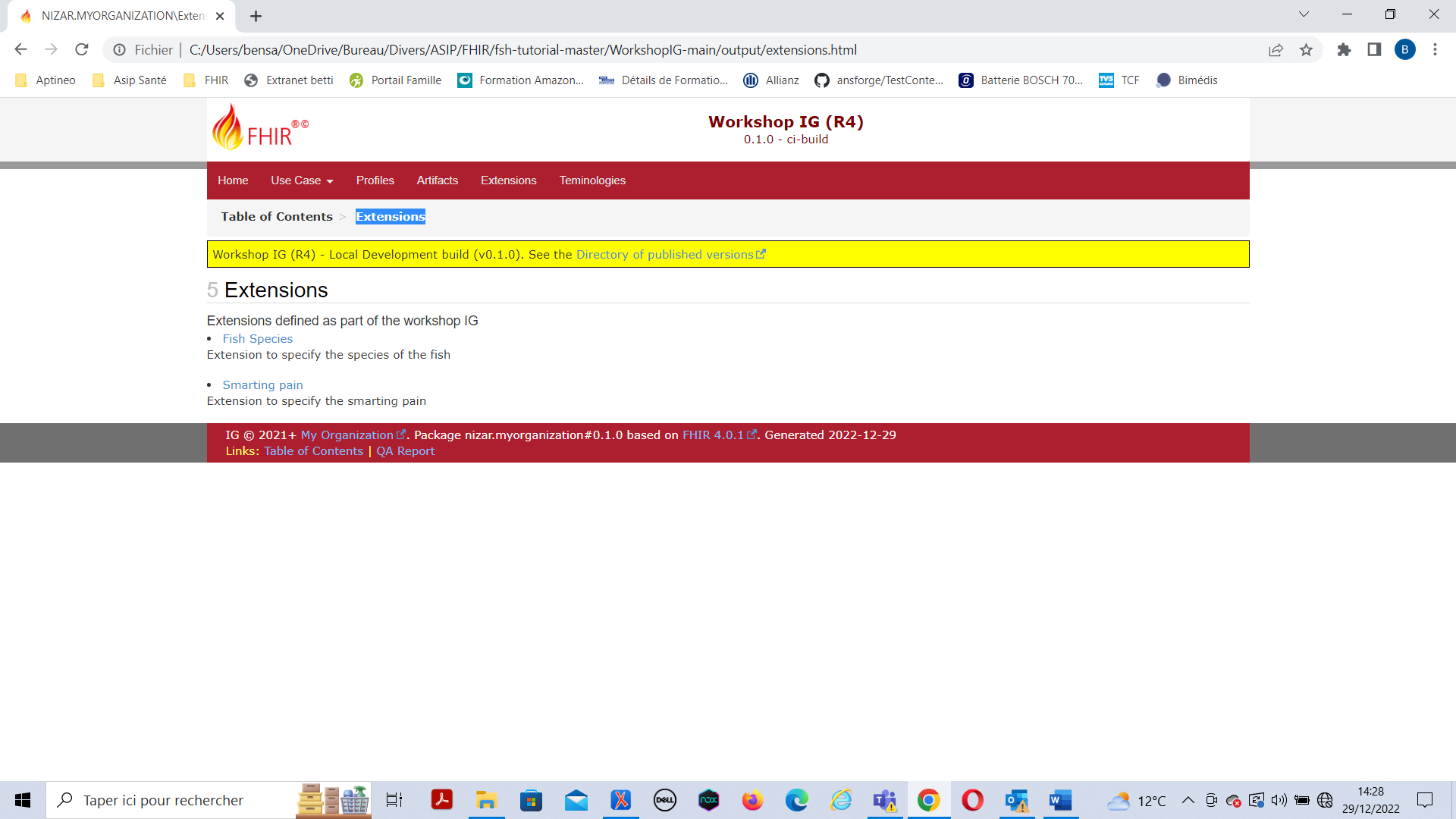


## Page Artifacts

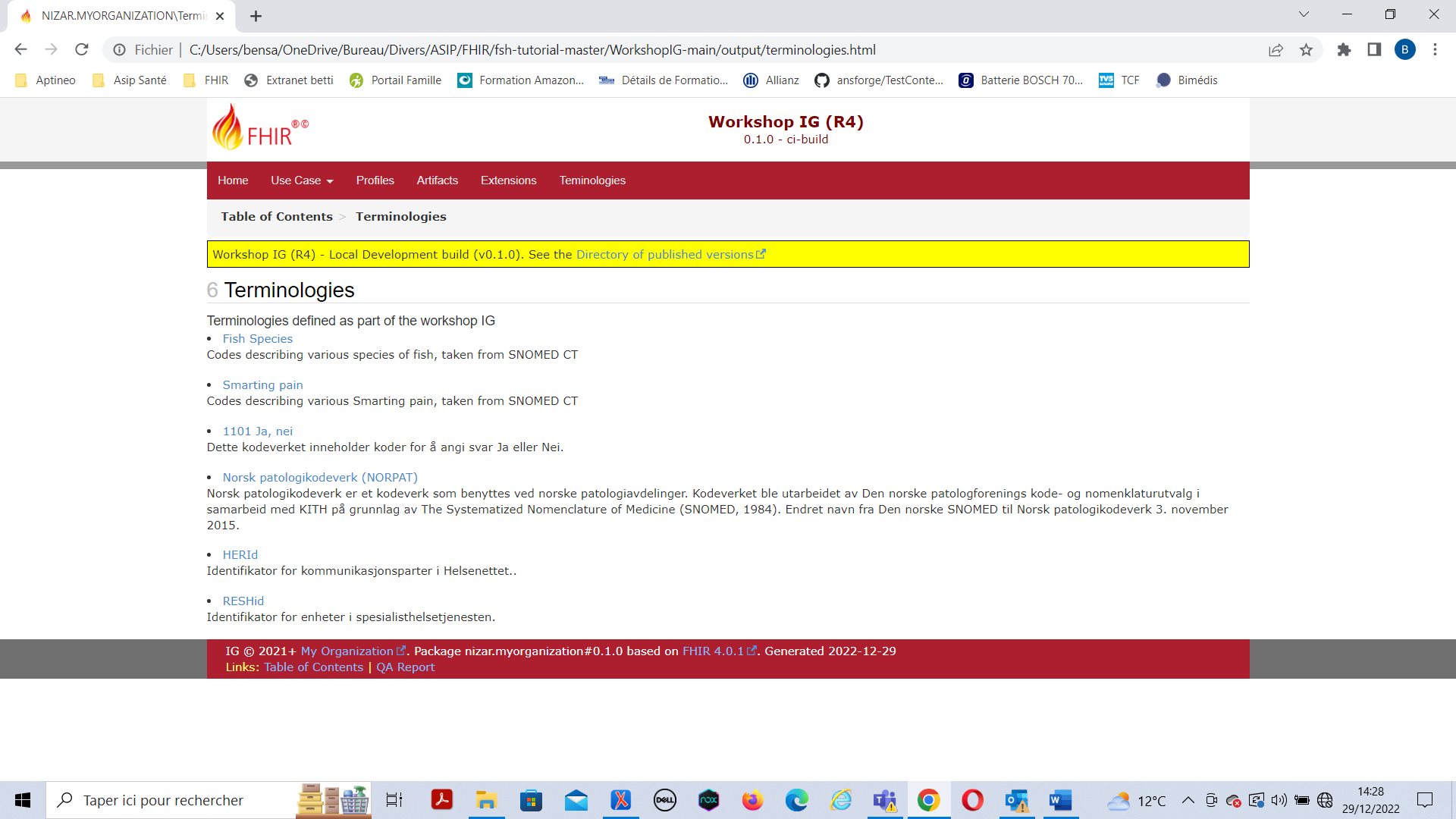




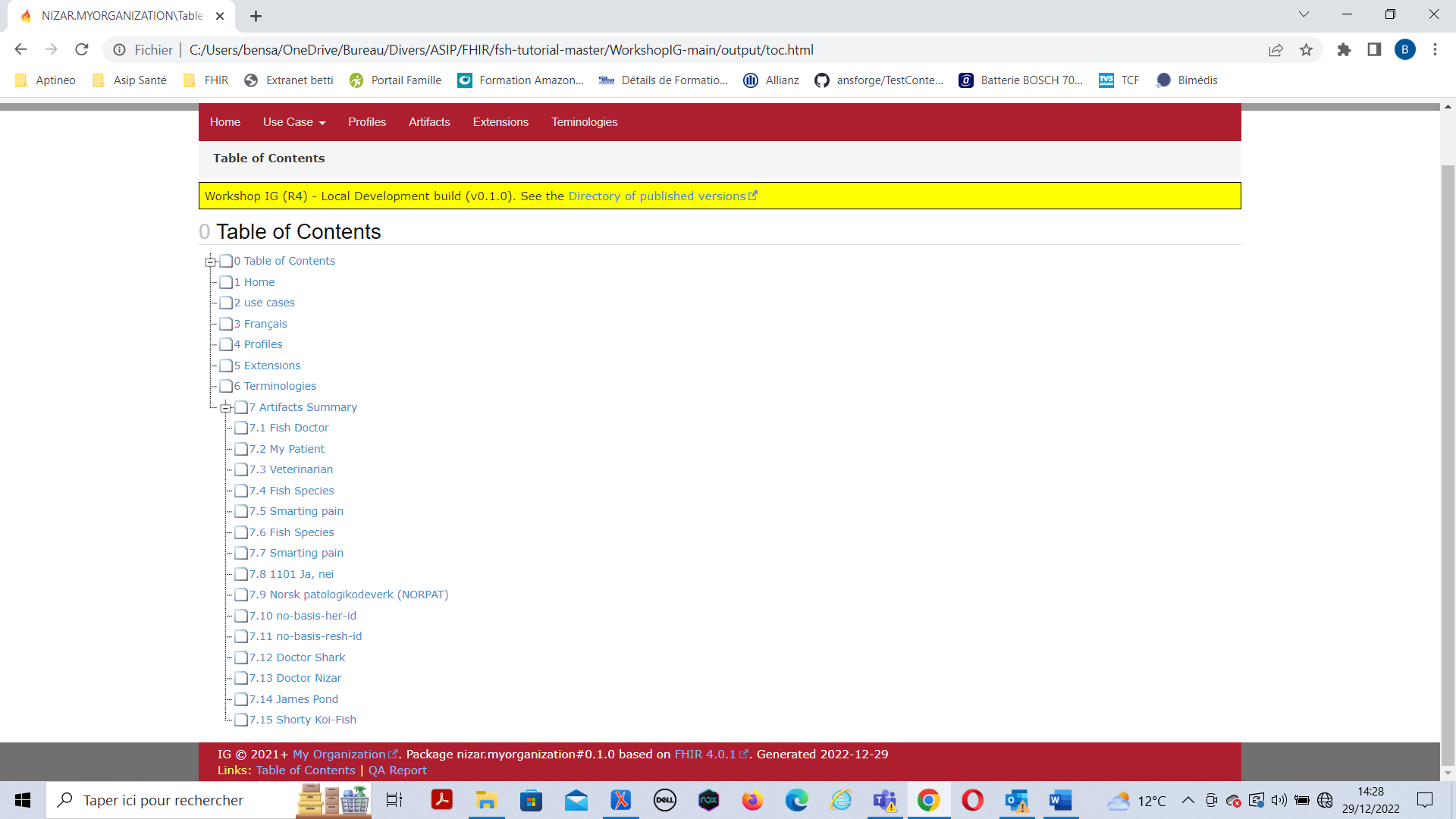
## Page Extensions



## Page Terminologies



## Page Table of Contents



# Conclusion

Le projet SUSHI-FHIR que j’ai créé est comité sur GitHub à l’adresse suivante : <https://github.com/nizarbs2508/SUSHI-FHIR>. Clonez l’exemple (https://github.com/nizarbs2508/SUSHI-FHIR.git), exécutez le et amusez vous à faire des modifications et voir les résultats en exécutant la page web index.html. N’oubliez pas d’exécuter le script \_genonce.sh après chaque modification pour qu’elle soit prise en charge lors de l’exécution de la page web.