**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC CẦN THƠ**

**KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**



**LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**Đề tài**

**XÂY DỰNG ỨNG DỤNG CHẨN ĐOÁN BỆNH TÔM**

**DỰA TRÊN ONTOLOGY**

**Sinh viên: Nguyễn Hoàng Tuấn**

**Mã số: B1606858**

**Khóa: K42**

***Cần Thơ, 01/2021***

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC CẦN THƠ**

**KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**



**LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**Đề tài**

**XÂY DỰNG ỨNG DỤNG CHẨN ĐOÁN BỆNH TÔM**

**DỰA TRÊN ONTOLOGY**

**Người hướng dẫn Sinh viên thực hiện**

TS. Trần Công ÁnNguyễn Hoàng Tuấn

Mã số: B1606858

Khóa: K42

***Cần Thơ, 01/2021***

**LỜI CẢM ƠN**

Em xin gửi lời cảm ơn chân thành sâu sắc đến thầy Trần Công Án đã tận tình hướng dẫn và giúp đỡ em trong suốt quá trình hoàn hành luận văn.

Xin cảm ơn quý thầy cô khoa CNTT và TT đã truyền đạt những kiến thức giúp em có những kiến thức nền tảng để có thể hoàn thành luận văn.

Xin cảm ơn Ban Giám Hiệu, tất cả thầy cô và nhân viên thuộc trường Đại học Cần Thơ đã tạo điều kiện cho em hoàn thành chương trình đào tạo.

Cảm ơn gia đình và bạn bè đã luôn luôn bên cạnh ủng độ, động viên, giúp đỡ em trong những lúc khó khăn nhất.

Em xin chân thành cảm ơn.

Cần Thơ, ngày 01 tháng 01 năm 2021

Tác giả luận văn

Nguyễn Hoàng Tuấn

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

Cần Thơ, tháng 01 năm 2021

**(GVHD ký tên)**

**NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN**

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

Cần Thơ, tháng 01 năm 2021

**(GVPB ký tên)**

**NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN**

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

Cần Thơ, tháng 01 năm 2021

**(GVPB ký tên)**

**MỤC LỤC**

[DANH MỤC BẢNG vii](#_Toc60982662)

[DANH MỤC HÌNH viii](#_Toc60982663)

[DANH MỤC TỪ VIẾT TẮT ix](#_Toc60982664)

[TÓM TẮT x](#_Toc60982665)

[ABSTRACT xi](#_Toc60982666)

[PHẦN 1: GIỚI THIỆU 1](#_Toc60982667)

[I. Đặt vấn đề 1](#_Toc60982668)

[II. Những nghiên cứu liên quan 1](#_Toc60982669)

[III. Mục tiêu 2](#_Toc60982670)

[IV. Đối tượng và phạm vi nghiên cứu 3](#_Toc60982671)

[1. Đối tượng nghiên cứu 3](#_Toc60982672)

[2. Phạm vi nghiên cứu 3](#_Toc60982673)

[V. Phương pháp nghiên cứu 3](#_Toc60982674)

[VI. Nội dung nghiên cứu 3](#_Toc60982675)

[VII. Bố cục quyển luận văn 4](#_Toc60982676)

[PHẦN 2: NỘI DUNG 5](#_Toc60982677)

[CHƯƠNG 1: CƠ SỞ LÝ THUYẾT 5](#_Toc60982678)

[I. Đặc tả yêu cầu 5](#_Toc60982679)

[1. Yêu cầu chức năng 5](#_Toc60982680)

[2. Yêu cầu phi chức năng 5](#_Toc60982681)

[II. Giới thiệu về bệnh tôm 6](#_Toc60982682)

[III. Giới thiệu về ontology 6](#_Toc60982683)

[1. Các thành phần trong ontology 6](#_Toc60982684)

[2. Các bước xây dựng một ontology 7](#_Toc60982685)

[3. Ontology bệnh tôm 9](#_Toc60982686)

[IV. Giới thiệu về Sementic Web 15](#_Toc60982687)

[1. RDF 16](#_Toc60982688)

[2. OWL 16](#_Toc60982689)

[3. SWRL 16](#_Toc60982690)

[4. Jena Fuseki 17](#_Toc60982691)

[5. SPARQL 17](#_Toc60982692)

[V. Giới thiệu một số công nghệ thiết kế web 18](#_Toc60982693)

[1. Xampp 18](#_Toc60982694)

[2. Apache 18](#_Toc60982695)

[3. Ngôn ngữ PHP 18](#_Toc60982696)

[4. Ngôn ngữ HTML 19](#_Toc60982697)

[5. Ngôn ngữ CSS 19](#_Toc60982698)

[6. Bootstrap 20](#_Toc60982699)

[7. Ngôn ngữ JavaScript 20](#_Toc60982700)

[8. JQUERY 20](#_Toc60982701)

[CHƯƠNG 2: THIẾT KẾ VÀ CÀI ĐẶT GIẢI PHÁP 21](#_Toc60982702)

[I. Kiến trúc hệ thống 21](#_Toc60982703)

[II. Jena Fuseki 21](#_Toc60982704)

[III. Thư viện Sparqllib 23](#_Toc60982705)

[IV. Chuẩn đoán bệnh 25](#_Toc60982706)

[V. Tra cứu 27](#_Toc60982707)

[CHƯƠNG 3: ĐÁNH GIÁ VÀ KIỂM THỬ 32](#_Toc60982708)

[I. Mục tiêu kiểm thử 32](#_Toc60982709)

[II. Kịch bản kiểm thử 32](#_Toc60982710)

[III. Kết quả kiểm thử 33](#_Toc60982711)

[PHẦN 3: KẾT LUẬN 34](#_Toc60982712)

[I. Kết quả đạt được 34](#_Toc60982713)

[1. Kết quả 34](#_Toc60982714)

[2. Hạn chế 34](#_Toc60982715)

[II. Hướng phát triển 34](#_Toc60982716)

[TÀI LIỆU THAM KHẢO 35](#_Toc60982717)

**DANH MỤC BẢNG**

[Bảng 1. Danh sách các liên kết 11](#_Toc61191651)

[Bảng 2. Danh sách các thuộc tính 12](#_Toc61191652)

[Bảng 3. Thuộc tính của mối quan hệ bậc ba 13](#_Toc61191653)

[Bảng 4 . Danh sách các luật 14](#_Toc61191654)

[Bảng 5. Các phương thức 14](#_Toc61191655)

[Bảng 6. Một số thẻ trong HTML 19](#_Toc61191656)

[Bảng 7. Một số hàm / phương thức trong thư viện sparqllib 24](#_Toc61191657)

[Bảng 8. Kịch bản kiểm thử 32](#_Toc61191658)

[Bảng 9. Kết quả kiểm thử kịch bản 33](#_Toc61191659)

**DANH MỤC HÌNH**

[Hình 1. Giao diện chọn triệu chứng của Ứng dụng ESforRPD2 1](#_Toc61192678)

[Hình 2. Giao diện kết quả của Ứng dụng ESforRPD2 2](#_Toc61192679)

[Hình 3. Giao diện chi tiết loại bệnh của Ứng dụng ESforRPD2 2](#_Toc61192680)

[Hình 4. Sơ đồ chức năng 5](#_Toc61192681)

[Hình 5. Ví dụ TBox và ABox của ontology 6](#_Toc61192682)

[Hình 6. Mô hình cơ sở dữ liệu bệnh tôm 9](#_Toc61192683)

[Hình 7. Các lớp và lớp con được tạo trong protégé 9](#_Toc61192684)

[Hình 8. Các lớp của otology bệnh tôm 10](#_Toc61192685)

[Hình 9. Các quan quan hệ được tạo trong protégé 10](#_Toc61192686)

[Hình 10. Các thuộc tính được tạo trong protégé 11](#_Toc61192687)

[Hình 11. Các thể hiện được tạo trong ontology 12](#_Toc61192688)

[Hình 12. Cách lấy dữ liệu axixom 13](#_Toc61192689)

[Hình 13. Kiến trúc của một web ngữ nghĩa 15](#_Toc61192690)

[Hình 14. Mối quan hệ bộ ba trong RDF 16](#_Toc61192691)

[Hình 15. Kiến trúc hệ thống chẩn đoán bệnh 21](#_Toc61192692)

[Hình 16. Sơ đồ nguyên lý hoạt động của server Jena Fuseki 21](#_Toc61192693)

[Hình 17. Tạo dataset trên server Jena Fuseki 22](#_Toc61192694)

[Hình 18. Tải file ontology lên server 22](#_Toc61192695)

[Hình 19. Truy vấn câu lệnh sparql trên Jena Fuseki 23](#_Toc61192696)

[Hình 20. Import thư viện sparqllib 23](#_Toc61192697)

[Hình 21. Lấy kết quả từ CSDL 23](#_Toc61192698)

[Hình 22. Lưu đồ dữ liệu chẩn đoán bệnh 25](#_Toc61192699)

[Hình 23. Phương thức POST 25](#_Toc61192700)

[Hình 24. Giao diện chẩn đoán bệnh 26](#_Toc61192701)

[Hình 25. Giao diện kết quả chẩn đoán sơ bộ 26](#_Toc61192702)

[Hình 26. Giao diện kết quả chẩn đoán nâng cao 27](#_Toc61192703)

[Hình 27. Lưu đồ tra cứu bệnh 27](#_Toc61192704)

[Hình 28. Giao diện tra cứu bệnh 28](#_Toc61192705)

[Hình 29. Giao diện thông tin chi tiết bệnh 28](#_Toc61192706)

[Hình 30. Lưu đồ dữ liệu tra cứu triệu chứng 29](#_Toc61192707)

[Hình 31. Giao diện tra cứu triệu chứng 29](#_Toc61192708)

[Hình 32. Giao diện thông tin chi tiết triệu chứng 30](#_Toc61192709)

[Hình 33. Lưu đồ dữ liệu tra cứu thuốc 30](#_Toc61192710)

[Hình 34. Giao diện tra cứu thuốc 31](#_Toc61192711)

[Hình 35. Giao diện thông tin chi tiết thuốc 31](#_Toc61192712)

**DANH MỤC TỪ VIẾT TẮT**

RDF Resource Description Framework

RDFS Resource Description Framework Shema

CSDL Cơ sở dữ liệu

OWL Web Ontology Language

SWRL Semantic Web Rule Language

HTML Hypertext Markup Language

PHP Hypertext Preprocessor

CSS Cascading Style Sheets

JS JavaScript

XML Extensible Markup Language

TBox Terminological Box

ABox Assertion Box

**TÓM TẮT**

Hiện nay, Việt Nam là một trong những quốc gia xuất khẩu tôm lớn nhất trên thế giới, đem lại nguồn ngoại tệ ngày càng lớn cho đất nước. Với thời đại công nghệ số hiện nay, việc nuôi tôm cũng nên được hỗ trợ bởi công nghệ, điển hình như một phần mềm tìm ra bệnh tôm để dễ dàng tìm ra thuốc chữa trị cũng như phương pháp xử lý hiệu quả nhất. Đề tài luận văn mong muốn thiết kế một trang web dễ dàng sử dụng giúp chẩn đoán bệnh tôm dựa trên một số triệu chứng tôm mắc phải. Ứng dụng giúp người nuôi tôm đắc lực trong việc tìm ra bệnh tôm cũng như cách phòng chống, đặc biệt là những người mới bắt đầu nuôi tôm chưa có kinh nghiệm hay những triệu chứng không xảy ra phổ biến.

Với ontology, máy tính có thể tổng hợp cùng lúc nhiều nguồn dữ liệu cùng lúc cũng như có thể hiểu được trực tiếp nguồn dữ liệu do có cấu trúc ngữ nghĩa từ đó thông tin sẽ được tìm kiếm nhanh chóng và chính xác hơn. Đặc biệt mô hình phân cấp của nó rất phù hợp cho việc chẩn đoán. Ứng dụng chẩn đoán sử dụng cơ sở dữ liệu là ontology với ngôn ngữ chính là PHP chạy trên máy chủ Apache, được xây dựng trên công cụ Visual Studio Code. Các công nghệ khác được sử dụng như HTML, CSS, JavaScript, Bootstrap, SPARQL, jQuery.

**Từ khóa**: ontology, SPARQL, PHP.

**ABSTRACT**

Currently, Vietnam is one of the largest shrimp exporting countries in the world, bringing more and more foreign currency to the country. With today's digital technology, shrimp farming should also be supported by technology, typically a software to find shrimp diseases to easily find the cure and the most effective treatment method. The thesis aims to design an easy-to-use website to help diagnose shrimp diseases based on some symptoms of shrimp acquired. The application helps shrimp farmers effectively in finding out shrimp diseases as well as how to prevent them, especially those who have just started farming inexperienced shrimp or the symptoms do not occur commonly.

With ontology, the computer can synthesize many data sources at the same time as well as directly understand the data source due to the semantic structure from which information will be searched more quickly and accurately. Especially its hierarchical model is very suitable for diagnostics. The diagnostic application uses the database ontology with the main language PHP running on the Apache server, built on the Visual Studio Code engine. Other technologies used are HTML, CSS, JavaScript, Bootstrap, SPARQL, jQuery.

**Keyword**: ontology, SPARQL, PHP.

**PHẦN 1: GIỚI THIỆU**

1. Đặt vấn đề

Đường bờ biển nước ta dài 3260 km kéo dài qua 28 tỉnh thành từ Bắc vào Nam. Đây là điều kiện rất thuận lợi để phát triển ngành đánh bắt nuôi trồng thủy hải sản. Trong đó, ngành nuôi tôm đang được đầu tư và có những phát triển vượt bậc, mức tăng trưởng tăng đều qua từng năm, trở thành một trong những quốc gia xuất khẩu tôm lớn nhất thế giới, đem lại nguồn ngoại tệ khá lớn cho quốc gia. Tuy nhiên, việc chăm sóc tôm thật sự không dễ dàng, đặc biệt đối với những người mới bắt đầu chưa có nhiều kinh nghiệm, cụ thể là không biết được tôm bị bệnh gì. Ngay cả những người nuôi tôm có khi nghiệm khi gặp phải những bệnh hiếm gặp cũng sẽ khá lúng túng trong việc xử lý. Do đó, nhu cầu chẩn đoán bệnh tôm là cần thiết.

Bên cạnh đó, công nghệ thông tin dường như phủ sóng khắp mọi nơi, khắp mọi ngành nghề và nó có thể được ứng dụng vào để hỗ trợ cho việc chăm sóc tôm tốt hơn nhằm tăng năng suất, đạt hiệu quả cao hơn. Vừa hay, cơ sở dữ liệu phân cấp *ontology* rất phù hợp cho việc chẩn đoán tìm ra bệnh tôm dựa trên tập triệu chứng, cũng như tra cứu các loại bệnh và các loại thuốc đặc trị tương ứng. Do đó, ý tưởng xây dựng ứng dụng chẩn đoán bệnh dựa trên ontology đã được hình thành.

1. Những nghiên cứu liên quan

**Hệ thống chẩn đoán bệnh cho cây lúa ESforRPD2**

Hệ thống EsforRPD2 giúp chẩn đoán bệnh ở lúa dựa trên những đặc điểm hình thái, với dữ liệu đầu vào là danh sách các triệu chứng sẽ cho ra danh sách các bệnh.



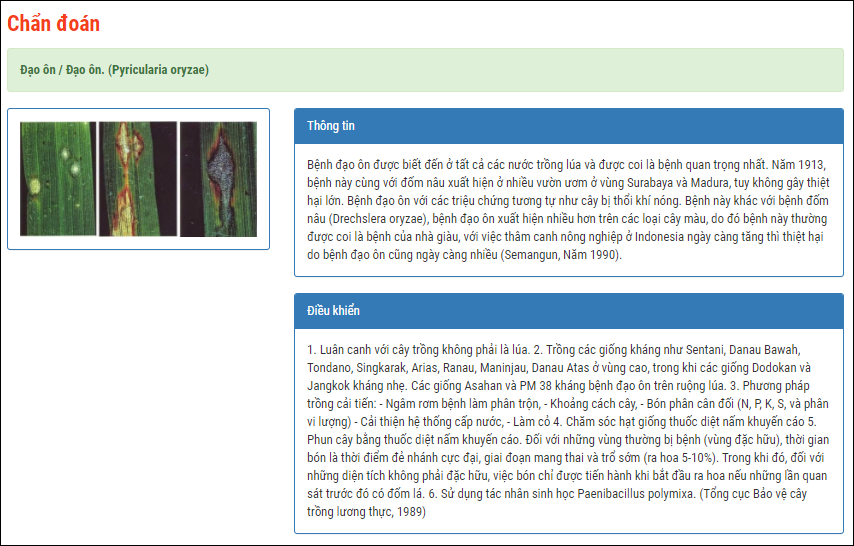
Hình 1. Giao diện chọn triệu chứng của Ứng dụng ESforRPD2

Đầu tiên, người dùng chọn những triệu chứng lúa đang mắc phải và gửi chẩn đoán.



Hình 2. Giao diện kết quả của Ứng dụng ESforRPD2

Kết quả sẽ cho ra danh sách các bệnh và tính toán mức độ tin cậy toàn thể.



Hình 3. Giao diện chi tiết loại bệnh của Ứng dụng ESforRPD2

Sau khi tìm ra loại bệnh, người dùng có thể xem chi tiết danh sách loại bệnh.

1. Mục tiêu

Đề tài mong muốn sẽ tổng hợp các loại bệnh, các triệu chứng ở tôm và hệ thống chúng đưa vào phần mềm chẩn đoán giúp tìm ra loại bệnh tôm đang mắc phải cũng như cách phòng chống dựa trên các triệu chứng mắc phải. Đề tài còn mong muốn xây dựng chức năng tra cứu giúp người dùng tìm thấy thông tin của danh sách các bệnh, các triệu chứng và thuốc.

1. **Đối tượng và phạm vi nghiên cứu**
2. **Đối tượng nghiên cứu**

Đối tượng nghiên cứu của đề tài bao gồm nguồn dữ liệu về bệnh tôm (các loại bệnh tôm, triệu chứng, thuốc) và cơ sở dữ liệu “Ontology” bao gồm các thành phần liên quan:

* Ontology: cơ sở dữ liệu phân cấp
* SPARQL: ngôn ngữ truy vấn dữ liệu trong ontology
* sparqllib: thư viện giúp kết nối dữ liệu và thực hiện các truy vấn SPARQL được viết bằng ngôn ngữ PHP

Ngoài ra, còn có các công nghệ về thiết kế web.

1. **Phạm vi nghiên cứu**

Phạm vi nghiên cứu của đề tài chủ yếu là làm việc với:

* Phần mềm Protege để thiết kế cơ sở dữ liệu
* Phần mềm Jena Fuseki để tạo endpoint giúp kết nối với trang web đồng thời chuyển mô hình dữ liệu về dạng RDF
* Phần mềm Visual Studio Code để thiết kế trang web

Ngoài ra, còn có các thư viện hỗ trợ như:

* Thư viện Bootstrap: hỗ trợ thiết kế web
* Thư viện jQuery: tạo các sự kiện ,tương tác giữa các phần tử web
* Thư viện sparqllib: kết nối và truy vấn dữ liệu RDF

1. **Phương pháp nghiên cứu**

Do nội dung đề tài khá mới mẻ nên phương pháp nghiên cứu chủ yếu dựa vào các tài liệu online trên mạng như trang chủ của các thư viện, youtube, các trang chia sẻ kiến thức. Đối với mỗi thành phần của đối tượng nghiên cứu sẽ có phương pháp:

* Ontology: tìm hiểu các nội dung liên quan đến ở trang chủ W3C, đọc tài liệu online tìm kiếm trên google
* SPARQL: đọc tài liệu trên các trang chia sẻ kiến thức
* Thư viện sparqllib: tải thư viện về và đọc, nghiên cứu các hàm, phương thức bên trong
* Dữ liệu bệnh tôm: đọc các tài liệu chẩn đoán bệnh ở tôm

1. **Nội dung nghiên cứu**

Nội dung nghiên cứu bao gồm:

* Tìm hiểu về các loại bệnh ở tôm cũng như các triệu chứng liên quan, tác nhân gây bệnh, cách phòng chống và thuốc đặc trị
* Tìm hiểu về ontology, cách sử dụng phần mềm thiết kế ontology (Protege), mô hình dữ liệu RDF và ngôn ngữ tuy vấn dữ liệu RDF (SPARQL)
* Tìm hiểu các công nghệ hỗ trợ cho việc thiết kế website như Bootstrap, jQuery.
* Tìm hiểu, tham khảo các đề tài liên quan để tìm ra giải pháp tốt nhất cho phần mềm.

1. Bố cục quyển luận văn

Bố cục luận văn bao gồm các phần và các chương sau:

Phần 1: Giới thiệu:

* Đặt vấn đề
* Những nghiên cứu liên quan
* Mục tiêu
* Đối tượng và phạm vi nghiên cứu
* Phương pháp nghiên cứu
* Nội dung nghiên cứu
* Bố cục quyển luận văn

Phần 2: Nội dung:

Chương 1: Đặc tả yêu cầu

* Yêu cẩu chức năng
* Yêu cầu phi chức năng

Chương 2: Thiết kế giải pháp

* Cơ sở lý thuyết
* Mô hình dữ liệu

Chương 3: Cài đặt giải pháp

* Server Jena Fuseki
* Thư viện Sparqllib
* Chẩn đoán bệnh
* Tra cứu

Chương 4: Đánh giá kiểm thử

* Mục tiêu kiểm thử
* Kịch bản kiểm thử
* Kết quả kiểm thử

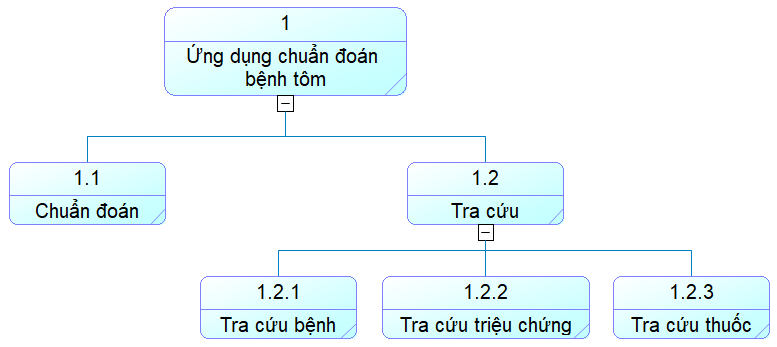
Phần 3: Kết luận:

* Kết luận
* Hướng phát triển

**PHẦN 2: NỘI DUNG**

**CHƯƠNG 1: CƠ SỞ LÝ THUYẾT**

1. **Đặc tả yêu cầu**
2. **Yêu cầu chức năng**



Hình 4. Sơ đồ chức năng

Sơ đồ chức năng Ứng dụng chẩn đoán bệnh tôm gồm có 2 chức năng chính là chẩn đoán và tra cứu. Chức năng chẩn đoán gồm có chẩn đoán sơ bộ và chẩn đoán nâng cao. Chức năng tra cứu gồm có tra cứu bệnh, tra cứu thuốc và tra cứu triệu chứng.

**Chức năng chẩn đoán bệnh**

Chức năng chẩn đoán bệnh được chia làm 2 giai đoạn:

* Tìm những bệnh có một trong các triệu chứng đã chọn
* Từ danh sách bệnh, tìm các triệu chứng liên liên quan khác và chọn tiếp trong các triệu chứng đó để giới hạn lại danh sách bệnh tìm được. Giai đoạn 2 có thể lặp lại đến khi nào không còn triệu chứng liên quan nữa

**Chức năng tra cứu**

Chức năng tra cứu gồm có 3 phần:

* Tra cứu bệnh: từ danh sách các bệnh có thể xem chi tiết bệnh đó, các triệu chứng của bệnh, các thuốc đặc trị cũng như thông tin về tác nhân gây bệnh và cách phòng chống
* Tra cứu triệu chứng: từ danh sách triệu chứng có thể tra ra mắc triệu chứng đó sẽ có thể bị những bệnh nào
* Tra cứu thuốc: từ danh sách thuốc, chọn một loại thuốc sau đó tìm ra các bệnh mà thuốc đặc trị

1. **Yêu cầu phi chức năng**

Các yêu cầu phi chức năng gồm:

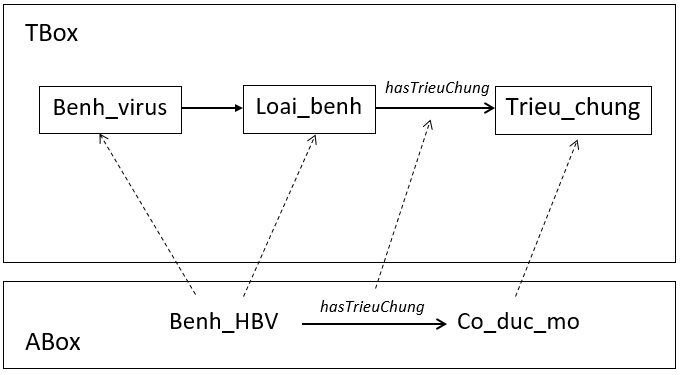
* Giao diện đơn giản, thân thiện, dễ dùng.
* Ứng dụng chỉ hỗ trợ chạy tốt nhất trên màn hình desktop.

1. **Giới thiệu về bệnh tôm**

Tôm là loài rất nhạy cảm với môi trường. Bất kỳ thay đổi nào cũng có thể gây cho chúng một số loại bệnh. Ngoài ra, sự phát triển dịch bệnh là rất nhanh, tôm có thể chết rất nhanh sau khi nhiễm bệnh gây ra thiệt hại rất lớn cho người nuôi tôm có thể lến đến hàng trăm tỷ đồng. Một trong những nguyên nhân quan trọng gây ra thiệt hại là do người nông dân chưa đủ kiến thức về kỹ thuật. Nếu muốn tăng năng suất, giảm thiệt hại do bệnh tật gây ra xuống mức thấp nhất, người nông dân cần công cụ giúp phát hiện kịp thời và gợi ý các phương pháp điều trị hiệu quả.

Triệu chứng bên ngoài của tôm là đặc điểm thông thường và dễ phát hiện bệnh trên tôm, từ đó người nuôi có thể vạch ra chiến lược phòng chống hiệu quả. Điều này có có ý nghĩa quan trọng trong việc ngăn dịch bệnh bùng phát trên quy mô lớn, giúp giảm thiệt hại đến mức thấp nhất. Tuy nhiên, có một số triệu chứng hiếm gặp hoặc một số loại bệnh có triệu chứng tương đối giống nhau gây rất nhiều khó khăn cho người nuôi đặc biệt là những người chưa có nhiều kinh nghiệm.

1. **Giới thiệu về ontology**



Hình 5. Ví dụ TBox và ABox của ontology

Ontology là một phương pháp luận mô tả miền cấu trúc tri thức của một lĩnh vực nào đó, được dùng để liên kết ngữ nghĩa các đối tượng liên quan trong lĩnh vực đó. Cơ sở tri thức của ontology bao gồm có 2 yếu tố: Tbox (gồm các khái niệm, các thuộc tính và các định nghĩa) và Abox (gồm các thể hiện và các giá trị của thuộc tính).

1. **Các thành phần trong ontology**

**Các lớp (Class)**

Các lớp là các nhóm, tập hợp các đối tượng gọi là thể hiện. Chúng có thể chứa các thể hiện, chứa cả lớp khác hay cả hai cùng lúc đều được. Một lớp có thể xếp gộp vào một lớp khác để trở thành lớp con của lớp đó. Mối quan hệ giữa các lớp xếp gộp tạo ra một cấu trúc có thứ bậc. Thường trong một ontology, tất cả các lớp được đặt trong một lớp gốc gọi là Thing. Ví dụ điển hình như lớp con người, lớp động vật, lớp thực vật, lớp gia súc, lớp gia cầm,…

**Các thể hiện (Individuals)**

Các thể hiện là các đối tượng, là thành phần cơ bản của một ontology. Các thể hiện là những đối tượng như con người, động vật, đồ vật, địa điểm,… Thậm chí là các đối tượng trừu tượng như thành viên, các từ,...Ví dụ như: gà, vịt, ngỗng là các thể hiện của lớp gia cầm.

**Các thuộc tính (Data Properties)**

Các thuộc tính mô tả các đặc trưng, đặc điểm, tính chất của các thể hiện trong ontology. Mỗi thuộc tính có tên và đều có một giá trị đối với mỗi thể hiện. Các thuộc tính dùng để lưu trữ các thông tin mà mỗi thể hiện có thể có. Ví dụ như: con người là một lớp sẽ có các thuộc tính như họ tên, ngày sinh, địa chỉ, độ tuổi,…

**Các quan hệ (Object Property)**

Các quan hệ là mối liên kết giữa các thể hiện trong ontology. Một quan hệ cũng như một thuộc tính nhưng có giá trị là một thể hiện khác trong ontology.Ví dụ: John là bố của Anna thì bố cũng có thể được xem là một quan hệ hay Mari đã kết hôn với Dan thì đã kết hôn với là một quan hệ.

1. **Các bước xây dựng một ontology**

Có nhiều phương pháp khác nhau để xây dựng một ontology, nhưng nhìn chung các phương pháp đều có 2 bước cơ bản là: xây dựng cấu trúc phân cấp và định nghĩa các thuộc tính cho lớp. Trong thực tế, việc phát triển một ontology là một việc không đơn giản, phụ thuộc rất nhiều vào công cụ sử dụng, tính chất, quy mô cũng như các quan hệ phức tạp trong nó. Ngoài ra, việc xây dựng ontology còn phải quan tâm đến khả năng mở rộng miền trong tương lai và khả năng kế thừa các hệ thống ontology sẵn có. Tóm lại, để xây dựng một ontology có thể áp dụng theo các bước sau:

* Xác định miền quan tâm và phạm vi của ontology
* Xem xét việc kế thừa ontology sẵn có
* Liệt kê các thuật ngữ quan trọng
* Xây dựng các lớp theo cấu trúc phân cấp
* Định nghĩa các thuộc tính và quan hệ cho các lớp
* Định nghĩa ràng buộc về thuộc tính và quan hệ của lớp
* Tạo các thể hiện cho lớp

Vấn đề cốt lõi của các ontology là xây dựng được các lớp và các thuộc tính. Chúng ta phải sử dụng mô hình để xây dựng ontology sao cho thỏa mãn yêu cầu để giải quyết các vấn đề của ứng dụng. Các ứng dụng trên ontology hầu hết được thể hiện ở các dạng hệ chuyên gia (Expert System), web ngữ nghĩa (Symantic Web) và khai khoáng dữ liệu (data mining). Trong đó, phát triển theo hệ chuyên gia là tối ưu nhất.

* **Xây dựng một ontology bao gồm 7 bước sau:**

**Bước 1: Xác định miền quan tâm và phạm vi của ontology**

Để xác định miền và phạm vi cho ontology trả lời các câu hỏi sau:

* Cần mô tả miền nào?
* Phục vụ cho mục đích gì?
* Phục vụ nhứng đối tượng nào?
* Ai là người xây dựng, quản trị ontology?

**Bước 2: Xem xét việc kế thừa ontology có sẵn**

Đây là một công đoạn thường hay dùng để tiết kiệm công sức xây dựng một ontology. Bằng cách kế thừa một otology có sẵn người xây dựng có thể thêm hặc bớt các lớp, thuộc tính, thể hiện,… để tùy chỉnh theo mục đích của mình.

**Bước 3: Liệt kê các thuật ngữ quan trọng**

Đây là bước rất hữu ích, làm tiền đề cho các bước tiếp theo như xây dựng cấu trúc lướp phân cấp và định nghĩa các thuộc tính cho lớp. Công việc này bao gồm liệt kê tất cả thuật ngữ xuất hiện trong miền như tên khái niệm, quan hệ, thuộc tính.

**Bước 4: Xấy dựng các lớp theo cấu trúc phân cấp**

Phân loại và tìm ra các nhóm thuật ngữ có khả năng là một lớp, thông thường là các thuật ngữ là danh từ. Từ nhóm tìm ra tiếp tục gom nhóm những thuật ngữ có cùng tính chất và tiến hàng phân cấp theo nguyên tắc subclass cũng là một class. Ví dụ, ta có các thuật ngữ như: loại bệnh, bệnh tim, bệnh dạ dày. Ta có thể xếp bệnh tim và bệnh dạ dày là lớp con của loại bệnh do bệnh tim cũng là một loại bệnh, bệnh dạ dày cũng là một loại bệnh.

**Bước 5: Xác định các thuộc tính và các quan hệ cho lớp**

Ta xem xét các thuật ngữ còn lại sau khi xác định các lớp có khả năng rất cao đó là thuộc tính (data property) và quan hệ (). Sau khi tìm được thuộc tính, cần xác định nó mô tả cho lớp nào

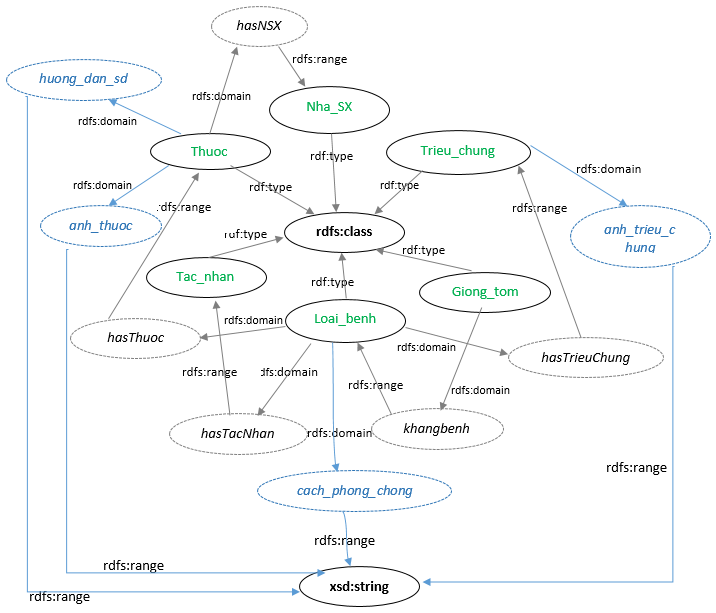
**Bước 6: Định nghĩa ràng buộc cho thuộc tính và quan hệ**

Trước tiên, cần xem thuộc tính mô tả cho lớp nào. Sau dó, dựa vào tên thuộc tính đó mà gán cho nó một kiểu giá trị xác định. Ví dụ: thuộc tính “độ tuổi” sẽ là kiểu số, “địa chỉ” sẽ là kiểu chuỗi. Đối với quan hệ ràng buộc là các lớp. Ví dụ: quan hệ “kết hôn” có ràng buộc là 2 lớp nam và nữ

**Bước 7: Tạo các thể hiện cho lớp**

Bước cuối cùng là tạo ra các thể hiện cho các lớp đã phân cấp. Đến đây, các thành phần của ontology đã hoàn thiện. Người dùng có thể thu thập thông tin của các thể hiện để nhập dữ liệu vào cho các thuộc tính.

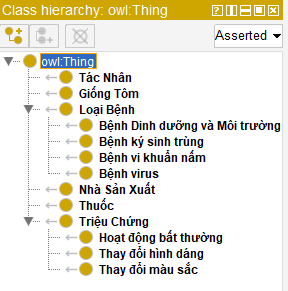
1. **Ontology bệnh tôm**



Hình 6. Mô hình cơ sở dữ liệu bệnh tôm

Mô hình cơ sở dữ liệu bệnh tôm bao gồm các lớp, các thuộc tính và các liên kết giữa các lớp. Chúng được xác định rõ miền giá trị *rdfs:domain* và phạm vi hoạt động *rdfs:range.*

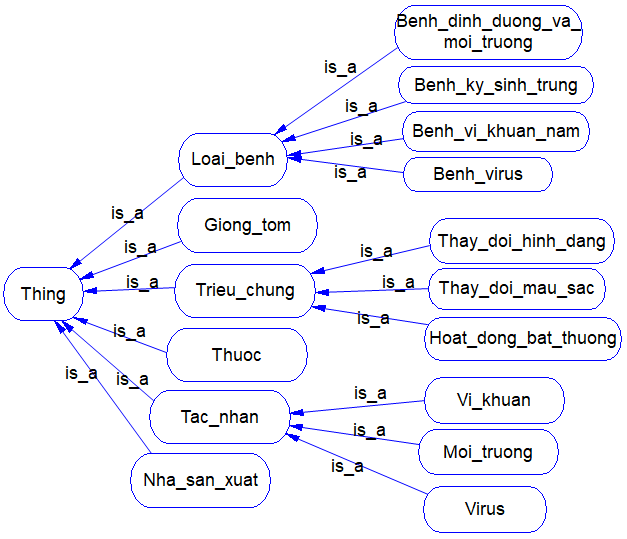
**Lớp – Class**



Hình 7. Các lớp và lớp con được tạo trong protégé

Các lớp trong ontology bệnh tôm:

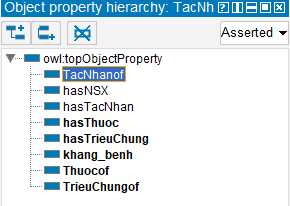
* Loai\_benh: nhóm bệnh gây ra ở tôm, gồm có 4 phân lớp là *Benh\_dinh\_duong\_va\_moi\_truong*, *Benh\_ky\_sinh\_trung*, *Benh\_vi\_khuan\_nam* và *Benh\_virus*
* Trieu\_chung: nhóm các triệu chứng tôm mắc phải, gồm 3 phân lớp là *Hoa\_dong\_bat\_thuong*, *Thay\_doi\_hinh\_dang* và *Thay\_doi\_mau\_sac*
* Thuốc: Nhóm các loại thuốc chữa trị bệnh tôm
* Nha\_san\_xuat: nhóm các nhà sản xuất phân phối thuốc
* Giong\_tom: nhóm các giống tôm
* Tac\_nhan: nhóm các tác nhân gây ra bệnh cho tôm



Hình 8. Các lớp của otology bệnh tôm

Các lớp con trong ontology được hiểu là một trong các trường hợp của của lớp cha. Ta có thể nói lớp con là môt lớp cha, ví dụ như Benh\_virus là một Loai\_benh.

**Quan hệ – Object Property**



Hình 9. Các quan quan hệ được tạo trong protégé

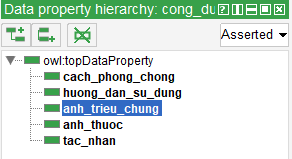
Các quan hệ trong ontology bệnh tôm:

* hasTrieuChung: liên kết giữa lớp *Loai\_benh* và lớp *Trieu\_chung*, mô tả một loại bệnh có một số triệu chứng
* TrieuChungof: liên kết giữa *Trieu\_chung* và *Loai\_benh*, mô tả một triệu chứng có khả năng mắc một số bệnh
* hasThuoc: liên kết giữa *Loai\_benh* và *Thuoc*, mô tả một loại bệnh có thể được trị bởi một số loại thuốc
* Thuocof: liên kết giữa *Thuoc* và *Loai\_benh*, mô tả một loại thuốc có thể đặc trị một số bệnh
* Khangbenh: liên kết giữa *Giong\_tom* và *Loai\_benh*, mô tả một giống tôm có thể kháng một số bệnh
* hasNSX: liên kết giữa *Thuoc* và *Nha\_san\_xuat*, mô tả một loại thuốc có thể do nhiều nhà sản xuất phân phối
* hasTacNhan: liên kết giữa *Loai\_benh* và *Tac\_nhan*, mô tả một loại bệnh có thể do một số tác nhân gây ra
* TacNhanof: liên kết giữa *Tac\_nhan* và *Loai\_benh*, mô tả một tác nhân có thể gây ra nhiều loại bệnh

Bảng 1. Danh sách các liên kết

|  |  |  |
| --- | --- | --- |
| Object | Domain | Ranges |
| hasTrieuChung | Loai\_benh | Trieu\_chung |
| TrieuChungof | Trieu\_chung | Loai\_benh |
| hasThuoc | Loai\_benh | Thuoc |
| Thuocof | Thuoc | Loai\_benh |
| Khang\_benh | Giong\_tom | Loai\_benh |
| hasNSX | Benh | Nha\_xan\_xuat |
| hasTacNhan | Loai\_benh | Tac\_nhan |
| TacNhanof | Tac\_nhan | Loai\_benh |

**Thuộc tính – Data Property**



Hình 10. Các thuộc tính được tạo trong protégé

Các thuộc tính trong ontology bệnh tôm:

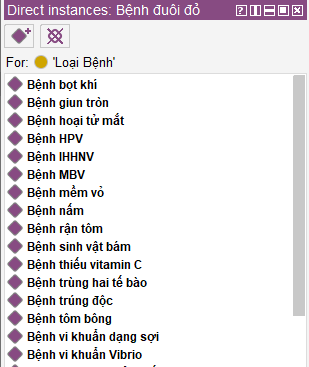
* Cach\_phong\_chong: lưu thông tin cách phòng chống của một loại bệnh
* Anh\_trieu\_chung: lưu thông tin đường dẫn ảnh của triệu chứng
* Anh\_thuoc: lưu thông tin đường dẫn ảnh của thuốc
* Huong\_dan\_su\_dung: lưu thông tin hướng dẫn sử dụng của một loại thuốc
* Tac\_nhan: lưu thông tin tác nhân gây bệnh của một loại bệnh

Bảng 2. Danh sách các thuộc tính

|  |  |  |
| --- | --- | --- |
| Property | Domain | Ranges |
| Cach\_phong\_chong | Loai\_benh | xsd:string |
| Anh\_trieu\_chung | Trieu\_chung | xsd:string |
| Anh\_thuoc | Thuoc | xsd:string |
| Huong\_dan\_su\_dung | Thuốc | xsd:string |

Thuộc tính *Cach\_phong\_chong* lưu trữ cách phòng chống cho từng loại bệnh khác nhau. Thuộc tính *Anh\_trieu\_chung* dùng lưu đường dẫn của triệu chứng. Thuộc tính *Anh\_thuoc* dùng lưu đường dẫn của thuốc. Thuộc tính *Huong\_dan\_su\_dung* lưu trữ cách sử dụng của từng loại thuốc.

**Thể hiện - Individual**



Hình 11. Các thể hiện được tạo trong protégé

Các thể hiện trong ontology bệnh tôm:

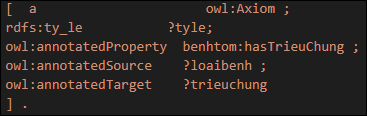
* Loai\_benh: có 22 loại bệnh. Trong đó:
* Benh\_dinh\_duong\_va\_moi\_truong: có 4 loại bệnh
* Benh\_ky\_sinh\_trung: có 5 loại bệnh
* Benh\_vi\_khuan\_nam: có 5 loại bệnh
* Benh\_virus: có 8 loại bệnh
* Trieu\_chung: có 37 triệu chứng. Trong đó:
  + Hoat\_dong\_bat\_thuong: có 7 triệu chứng
  + Thay\_doi\_hinh\_dang: có 14 triệu chứng
  + Thay\_doi\_mau\_sac: có 16 triệu chứng
* Thuoc: Gồm 46 loại thuốc

**Axiom**

Bảng 3. Thuộc tính của mối quan hệ bậc ba

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Axiom Type | Source | Object | Target | Type |
| ty\_le | Loai\_benh | hasTrieuChung | Trieu\_chung | xsd:interger |

Để giải quyết mối quan hệ bậc ba giữa Loai\_benh, Trieu\_chung và hasTrieuChung, hệ thống sử dụng giải pháp *owl:axiom*. Nhờ chức năng có thể tự định nghĩa một thuật ngữ riêng của RDFS, ta tạo một thuật ngữ làm thuộc tính riêng cho bộ ba Loai\_benh, Trieu\_chung và hasTrieuChung. Đó có nghĩa là đối với mỗi loại bệnh liên kết với một triệu chứng sẽ có thêm thông tin là tỷ lệ mắc bệnh. Tỷ lệ này có tổng sẽ là 100% đối với mỗi loại bệnh.



Hình 12. Cách lấy dữ liệu axixom

Cách lấy dữ liệu axiom khác với lấy theo bộ ba như thông thường mà sẽ được định danh theo 4 thành phần gồm: *ty\_le* là tên của thuộc tính axixom, *annotatedProperty* là liên kết giữa 2 đối tượng *annotatedSource* và *annotatedTarget*.

**Tập luật – SWRL**

Bảng 4 . Danh sách các luật

|  |  |  |
| --- | --- | --- |
| Ký hiệu | Tập luật | Giải thích |
| S1 | hasTrieuChung(?x, ?y) -> TrieuChungof(?y, ?x) | Nếu bệnh x liên kết hasTrieuChung với y thì đồng thời y sẽ liên kết TrieuChungof với x |
| S2 | hasThuoc(?x, ?y) -> Thuocof(?y, ?x) | Nếu bệnh x liên kết hasThuoc với y thì đồng thời y sẽ liên kết Thuocof với x |

Các luật sẽ ràng buộc dữ liệu giúp tạo liên kết cho các đối tượng dựa trên những liên kết đã có. Tập luật giúp đảm bảo tính đúng và đầy đủ của các liên kết.

**Các phương thức**

Bảng 5. Các phương thức

|  |  |  |
| --- | --- | --- |
| Tên phương thức | Ý nghĩa tham số | Diễn giải |
| get\_subClassOf\_trieuchung() |  | Lấy ra các lớp con của lớp trieuchung |
| get\_trieuchung($subclass) | Lớp con của lớp trieuchung | Lấy ra các triệu chứng trong lớp con $subclass |
| get\_loaibenh() |  | Lấy ra các loại bệnh từ các triệu chứng đã chọn |
| get\_label\_trieuchung() |  | Lấy ra nhãn của các triệu chứng |
| get\_trieuchung2() |  | Lấy ra các triệu chứng liên quan |
| get\_loaibenh\_result() |  | Lấy ra loại bệnh được lọc lại bởi triệu chứng liên quan |
| get\_label\_trieuchung\_result() |  | Lấy ra nhãn của tất cả các triệu chứng đã chọn |
| get\_trieuchung\_lienquan\_result() |  | Lấy ra các triệu chứng liên quan sau dựa trên các bệnh đã lọc |
| get\_detail\_benh\_trieuchung() |  | Lấy ra các triệu chứng của một loại bệnh nào đó |
| Get\_detail\_benh\_thuoc() |  | Lấy ra danh sách các loại thuốc trị loại bệnh đó |
| get\_detail\_trieuchung() |  | Lấy ra nhãn và hình ảnh của triệu chứng |
| get\_detail\_trieuchung\_benh() |  | Lấy ra các bệnh có một triệu chứng nào đó |
| get\_detail\_benh() |  | Lấy ra nhãn, tác nhân, cách phòng chống của một loại bệnh |
| get\_subClassOf\_benh() |  | Lấy ra các lớp con của lớp loaibenh |
| get\_all\_benh($name\_subclass) | Lớp con của lớp loai\_benh | Lấy ra các thể hiện của lớp $name\_subclass |
| get\_all\_trieuchung() |  | Lấy ra nhãn, hình ảnh của tất cả triệu chứng |
| get\_all\_thuoc() |  | Lấy tất cả các loại thuốc |
| Get\_detail\_thuoc() |  | Lấy ra thông tin của thuốc |
| Get\_detail\_thuoc\_benh() |  | Lấy ra danh sách các loại bệnh mà thuốc đặc trị |

Các phương thức trong bảng được dùng để truy vấn đến cơ sở dữ liệu và lấy ra các thông tin của những đối tượng như loại bệnh, triệu chứng, thuốc,... Tất cả các phương thức đều trả về dữ liệu dạng bảng giúp dễ dàng sử dụng để chèn vào nội dung của các phần tử web.

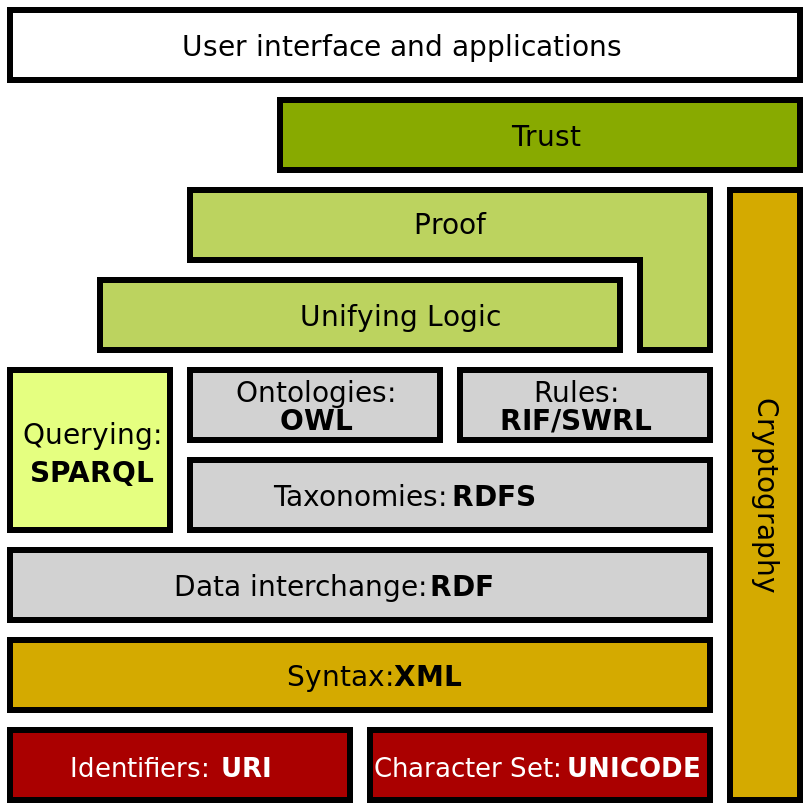
Các phương thức đã sử dụng đa số có 2 thành phần chính:

* Điểm endpoint để website kết nối với cơ sở dữ liệu
* Câu lệnh SPARQL giúp truy vấn dữ liệu

Ngoài ra, một số phương thức còn dùng các biến được truyền qua phương thức POST hoặc phương thức GET để hỗ trợ truy vấn lọc dữ liệu.

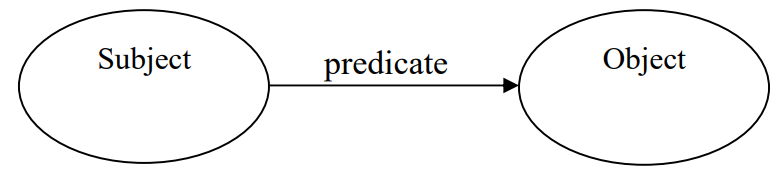
1. **Giới thiệu về Sementic Web**

Web ngữ nghĩa (web dữ liệu) là ý tưởng được Tim Berners-Lee hình thành vào năm 2001. Mục đích không phải là thay thế world wide web (web tài liệu) hiện có mà là mở rộng nó bằng cách thêm ngữ nghĩ vào nội dung web để phần mềm và mọi người có thể hiểu được nhằm suy luận các tài nguyên web và cùng nhau làm việc. Để làm được một bộ công nghệ và tiêu chuẩn được sử dung cho phép trình bày và xuất bản các tài nguyên trên web ở dạng máy tính có thể đọc được trực tiếp và xử lý được.



Hình 13. Kiến trúc của một web ngữ nghĩa

1. **RDF**



Hình 14. Mối quan hệ bộ ba trong RDF

RDF là viết tắt của Resource Description Framework, là một khung mô tả tài nguyên , là một mô hình dữ liệu. Khối xây cơ bản của RDF là bộ ba đối tượng-thuộc tính-giá trị. RDF được cung cấp một cú pháp trong XML và kế thừa những lợi ích liên quan đến XML. Tuy nhiên, các biểu diễn cú pháp quan trọng khác không dựa trên XML nên cú pháp dưa trên XML không phải là thành phần nhất thiết phải có trong mô hình RDF.

RDF độc lập với miền, người dùng có thể tùy ý xác định một thuật ngữ riêng của mình trong ngôn ngữ lược đồ được gọi là RDF Schema (RDFS). Trong RDFS, chúng ta có thể tự xác định từ vựng, chỉ định thuộc tính nào áp dụng cho loại đối tượng nào và quyết định chúng có thể nhận những giá trị nào.

1. **OWL**

OWL (The Web Ontology Language) là một ngôn ngữ gần như XML, là một ngôn ngữ đánh dấu ngữ nghĩa được thiết kế cho một web ngữ nghĩa đang phát triển ứng dụng. Nó cho phép người dùng tạo / chỉnh sửa các bản thể luận mà hoạt động giống như các tài liệu web khác nhưng với ngữ nghĩa được thêm vào chúng để máy có thể hiểu. Với OWL, người dùng có thể đại diện cho một miền kiến thức như tập hợp các khái niệm (lớp), các thể hiện, thuộc tính và xác định mối quan hệ giữa các khái niệm theo cách mà máy tính có thể hiểu được.

Viết bản thể luận ở định dạng XML rất khó và tốn thời gian vì cú pháp ngôn ngữ và độ phức tạp. Vì vậy, chúng ta cần một phần mềm hỗ trợ để tạo và chỉnh sửa OWL một cách trực quan. Một trong những công cụ điển hình đó là protégé. Hiện nay, OWL có 3 phiên bản: OWL Lite, OWL DL, và OWL Full. Trong đó, đơn giản nhất là OWL Lite và phức tạp nhất là OWL Full.

1. **SWRL**

SWRL là viết tắt của Semantic Web Rule Language. Đó là ngôn ngữ quy tắc được đề xuất bởi World Wide Web Consortium vào năm 2004 và được thiết kế để sử dụng trong web ngữ nghĩa. Nó đã được đề nghị để khắc phục hạn chế về khả năng diễn đạt của OWL, vì ngôn ngữ OWL không thể biểu diễn mối quan hệ phức tạp (ví dụ: chuỗi thuộc tính). SWRL cho phép người dùng viết các quy tắc có thể được thể hiện dưới dạng các khái niệm OWL và điều đó có thể lý giải về các cá nhân OWL. Một trong những các tính năng mạnh mẽ nhất của SWRL là khả năng hỗ trợ cài sẵn tùy chỉnh, do người dùng xác định, để mở rộng.

Ngoài ra, SWRL cung cấp nhiều hàm tích hợp cho phép hoạt động toán học và chuỗi và cho phép người dùng để xác định cài sẵn của riêng họ trong khi phát triển một ứng dụng.

1. **Jena Fuseki**

Jena Fuseki là dùng để xây dựng các ứng dụng web ngữ nghĩa, bao gồm API để đọc, xử lý và ghi dữ liệu RDF. Nó đóng vai trò là máy chủ giúp tạo endpoint từ file ontology đã tạo bằng phần mềm Protége để kết nối với ứng dụng. Jena còn cung cấp cho các nhà phát triển một tập thư viện để xử lý RDF, RDFS, OWL và SPARQL.

Với jena, người dùng có thể tạo, đọc, viết và thao tác mô hình RDF và lưu nó vào bộ nhớ hoặc lưu trữ liên tục (tức là cơ sở dữ liệu). Jena có thể đọc / ghi tệp ở một định dạng khác như RDF / XML, N3 và N-triple.

1. **SPARQL**

SPARQL là một ngôn ngữ truy vấn RDF, gồm 4 kiểu truy vấn khác nhau:

* Select: trích xuất các giá trị thô từ SPARQL endpoint, các kết quả được trả về trong một định dạng bảng
* Construct: trích xuất thông tin từ SPARQL endpoint và chuyển kết quả thàng dạng RDF hợp lệ
* Ask: cung cấp các kết quả dạng True/False đơn giản cho các truy vấn trên SPARQL endpoint
* Describe: trích xuất một đồ thị RDF từ SPARQL endpoint, các nội dung đó được đưa tới endpoint để quyết định dựa trên những thông tin có ích

Trong câu truy vấn SPARQL, biến được chỉ định bởi tiền tố “?” hoặc “$”. Câu truy vấn SPARQL cũng dựa trên bộ ba như trong RDF và được đặt trong mệnh đề WHERE, các thành phần cần truy xuất được đặt trong mệnh đề SELECT và được thay thế bằng biến.

Một câu truy vấn SPARQL hoàn chỉnh gồm 2 mệnh đề chính, mệnh đề SELECT và mệnh đề WHERE cùng các thành phần phụ như: PREFIX, FROM, FILTER, OPTIONAL, ORDER BY, GROUP BY, DISTINCT,… Cú pháp tổng quát của SPARQL được liệt kê ví dụ như sau:

PREFIX name: <namespaceURI>

SELECT ?a ?b

[FROM <dataURI>]

[FROM NAMED <dataURI>]

WHERE {

?a name: ?b .

[FILTER] [OPTIONAL]

}

[ORDER BY]

[GROUP BY]

[OFFSET/LIMIT]

[DISTINCT]

1. **Giới thiệu một số công nghệ thiết kế web**
2. **Xampp**

XAMPP là viết tắt của Cross-Platform (đa nền tảng-X), Apache (A), MariaDB (M), PHP (P) và Perl (P), là phần mềm mã nguồn mở rất phổ biến dùng để xây dựng các dự án website theo ngôn ngữ PHP được chạy trên localhost của máy.

XAMPP là bộ phần mềm tuyệt vời, có giao diện quản lý khá dễ sử dụng, tích hợp nhiều chức năng vô cùng tiện ích. Chỉ cần cài XAMPP, người dùng có thể sử dụng bộ ba tiện ích thiết kế web quyền lực:

* + Apache: server dùng để xử lý các file php
  + PHP: ngôn ngữ viết cho server
  + MySql: cơ sở dữ liệu thông dụng dùng trong thiết kế web

1. **Apache**

Apache là phần mềm web server miễn phí mã nguồn mở giúp đưa nội dụng lên web. Apache như một người giao hàng, khi có người gửi yêu cầu đến server Apache sẽ gửi file tương ứng với yêu cầu xuống client.

Apache là một phần mềm đa nền tảng, xử lý tốt các file dưới dạng ngôn ngữ PHP, Python, Java,… Apache biến những ngôn ngữ thành một phần của HTML và có thể hiển thị trên trình duyệt cho người dùng thấy được.

1. **Ngôn ngữ PHP**

PHP là viết tắt của Hypertext Preprocessor, là một ngôn ngữ lập trình dùng để phát triển các ứng dụng viết cho máy chủ. Nó thường được dùng để làm web do có thể dễ dàng nhúng và trang HTML, tốc độ nhanh, nhỏ gọn, dễ lập trình, cú pháp khá dễ dùng và đạt hiệu quả cao. Ngoài ra, PHP còn có rất nhiều các thư viện hỗ trợ đắc lực cho việc lập trình web.

**Ưu điểm của PHP**

* Cú pháp dễ học, dễ tiếp cận, có nhiều nguồn tài liệu tham khảo
* Tương thích với hầu hết mọi trình duyệt
* Có thể viết code trên bất kỳ trình soạn thảo văn bản nào
* Có tốc độ xử lý rất nhanh
* Tính cộng đồng cao, nhiều thư viện hỗ trợ

1. **Ngôn ngữ HTML**

HTML là viết tắt của Hypertext Markup Language, không phải là ngôn ngữ lập trình, chỉ bao gồm các thẻ giúp trình bày các nội dung văn bản.

Bảng 6. Một số thẻ trong HTML

|  |  |
| --- | --- |
| Tên thẻ | Mô tả |
| <html> | Bao bọc toàn bộ code của trang web |
| <head> | Chứa các thẻ định dạng tiêu đề, icon, thẻ nhúng các thư viện |
| <body> | Chứa nội dung trang web |
| <meta> | Định dạng bảng mã sử dụng |
| <title> | Gán tiêu đề cho trang web |
| <link> | Thẻ giúp chèn vào các file javascript hay css |
| <div> | Dùng nhóm các phần tử lại với nhau |
| <style> | Chứa các định dạng css |
| <script> | Chứa nội dụng javascript |
| <a> | Định nghĩa là một đường dẫn |
| <img> | Hiển thị hình ảnh dựa vào thuộc tính src |
| <p> | Chứa văn bản |
| <ol> | Danh sách theo thứ tự |
| <ul> | Danh sách không theo thứ tự |
| <li> | Phần tử trong danh sách |
| <table> | Định dạng bảng |
| <input> | Biểu diễn trường có thể nhập liệu vào, có nhiều kiểu để lựa chọn như: text, checkbox, radio, password,… |
| <button> | Tạo nút có thể click đặt trong form |

Tất cả nội dung trang web thường nằm trong thẻ <html></html>. Cấu trúc tổng thể của một trang web gồm thẻ <html></html> chứa 2 thẻ chính bên trong gồm thẻ <head></head> và thẻ <body></body>.

1. **Ngôn ngữ CSS**

CSS là viết tắt của Cascading Style Sheets, là ngôn ngữ giúp thay đổi định dạng nội dung của các phần tử HTML như là font chữ, màu sắc, vị trí, cách hiển thị,... Trang web có đẹp mắt, thân thiện người dùng hay không phần lớn là nhờ vào CSS.

CSS có thể được viết trực tiếp trong HTML bằng cách đặt trong các thẻ của HTML với thuộc tính style hoăc đặt trong thẻ <style>. Ngoài ra, CSS cũng có thể viết thành file riêng biệt và nhúng vào HTML bằng thẻ <link>. CSS có thể sử dụng một cách dễ dàng, có thể được viết trên bất kỳ trình soạn thảo văn bản nào, có cú pháp rất dễ tiếp cận và dễ học.

1. **Bootstrap**

Bootstrap là một framework bao gồm các HTML, CSS, JavaScript template dùng để thiết kế giao diện web nhanh hơn, chuyên nghiệp hơn. Bootstrap được dùng chủ yếu dựa vào định danh của các thẻ qua *class* hoặc *id*. Bootstrap được sử dụng rất dễ dàng với tính tùy chỉnh cao, người lập trình có thể tùy ý chỉnh sửa các định dạng sẵn có của bootstrap một cách nhanh chóng và đơn giản.

Những lợi ích của bootstrap:

* Phát triển website trở nên nhanh chóng
* Sự tương thích với trình duyệt và thiết bị
* Tương tác tốt với smartphone
* Dễ dàng tuỳ biến

Tuy nhiên, cũng có một số hạn chế:

* Có nhiều code thừa
* Làm giảm tính sáng tạo

1. **Ngôn ngữ JavaScript**

JavaScript được viết tắt là JS, là ngôn ngữ kịch bản, được thông dịch và thực thi trực tiếp bởi trình duyệt ở client, có thể hoạt động trên nhiều trình duyệt, nền tảng khác nhau và có tốc độ xử lý rất nhanh. JS là ngôn ngữ lập trình rất phổ biến, được sử dụng một cách rất linh hoạt, hiệu quả giúp tăng tính tương tác trên website.

JS có thể được nhúng trực tiếp và trang web qua thẻ <script>hoặc được tham chiếu qua một file .js riêng. JS có thể dễ dàng thay đổi nội dung của các thẻ HTML dựa vào id hay class và hơn nữa là có thể thay đổi các định dạng CSS của thẻ. JS là ngôn ngữ phía client nên làm giảm bớt lượng truy cập đến server, vừa tăng tốc độ xử lý vừa làm giảm gánh nặng cho server.

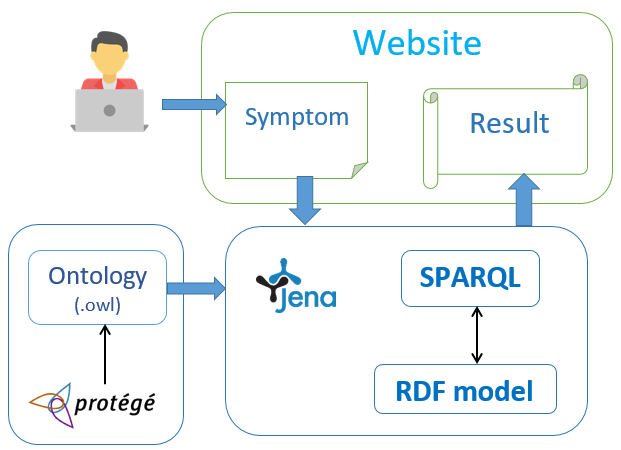
1. **JQUERY**

jQuery là một thư viện được viết bằng ngôn ngữ JavaScript, giúp xử lý các thuộc tính của đối tượng HTML, tạo các hiệu ứng hay xử lý sự kiện. Mục đích của jQuery là làm cho việc sử dụng JavaScript trở nên dễ dàng hơn, giúp website có tính tương tác và hấp dẫn hơn.

Với jQuery, chúng ta có thể làm được nhiều việc hơn mà ít tốn công sức và thời gian hơn. Ngoài ra, jQuery còn đơn giản hóa việc sử dụng AJAX và DOM. Một số ưu điểm khác của jQuery như là nhanh, nhỏ, nhẹ, tốn ít công sức mà hiệu quả lại khá cao, tương thích với nhiều loại trình duyệt và có rất nhiều tính năng rất hữu ích.

**CHƯƠNG 2: THIẾT KẾ VÀ CÀI ĐẶT GIẢI PHÁP**

1. **Kiến trúc hệ thống**

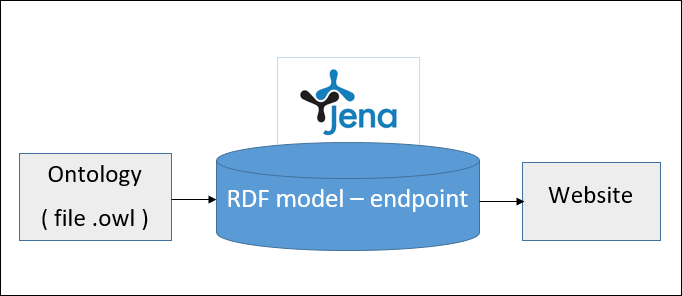


Hình 15. Kiến trúc hệ thống chẩn đoán bệnh

Người dùng truy cập vào hệ thống chẩn đoán bệnh của website, tiến hành thao tác chức năng gửi yêu cầu chẩn đoán. Website sẽ kết nối với server Jena thực hiện câu truy vấn SPARQL và nhận về kết quả chẩn đoán.

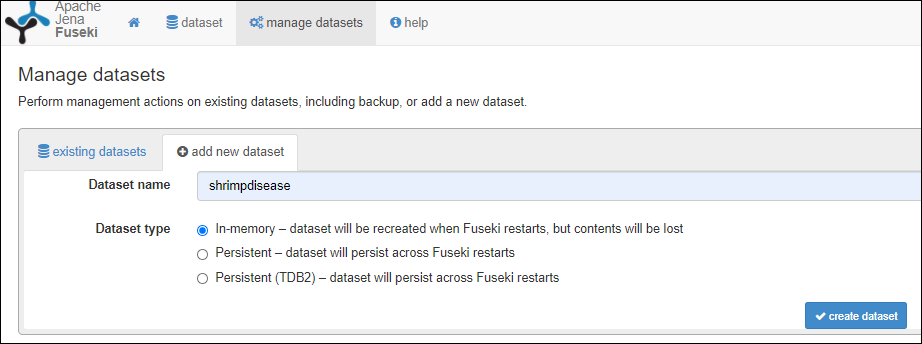
Về nguồn dữ liệu, sau khi thiết kế thành công một ontology dưới sự hỗ trợ của phần mềm protégé được kết quả là một file .owl có cấu trúc dạng XML. Upload file này lên server Jena để vừa tạo endpoint để kết nối được với website vừa chuyển cấu trúc về dạng RDF để có thể truy vấn bằng ngôn ngữ SPARQL. Đến lúc này, Jena đóng vai trò là nơi lưu trữ dữ liệu ontology vừa đóng vai trò là môi trường để website kết nối và truy vấn dữ liệu.

1. **Jena Fuseki**



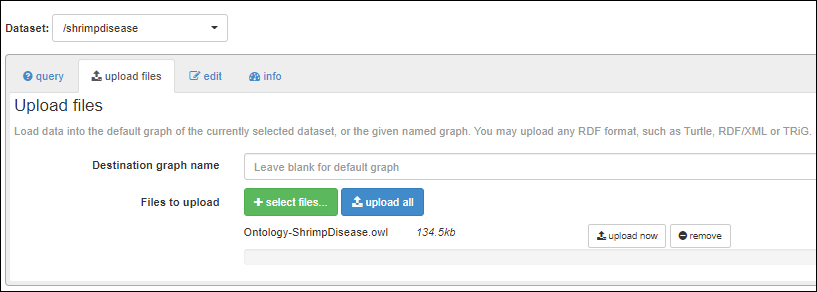
Hình 16. Sơ đồ nguyên lý hoạt động của server Jena Fuseki

Server Jena Fuseki nhận vào một ontology và chuyển cấu trúc của dữ liệu về dạng mô hình RDF gồm các bộ ba và tạo endpoint (điểm truy cập) để website có thể kết nối vào truy vấn dữ liệu. Jena vừa là môi trường để liên kết với website vừa là phần mềm quản trị mô hình dữ liệu RDF.



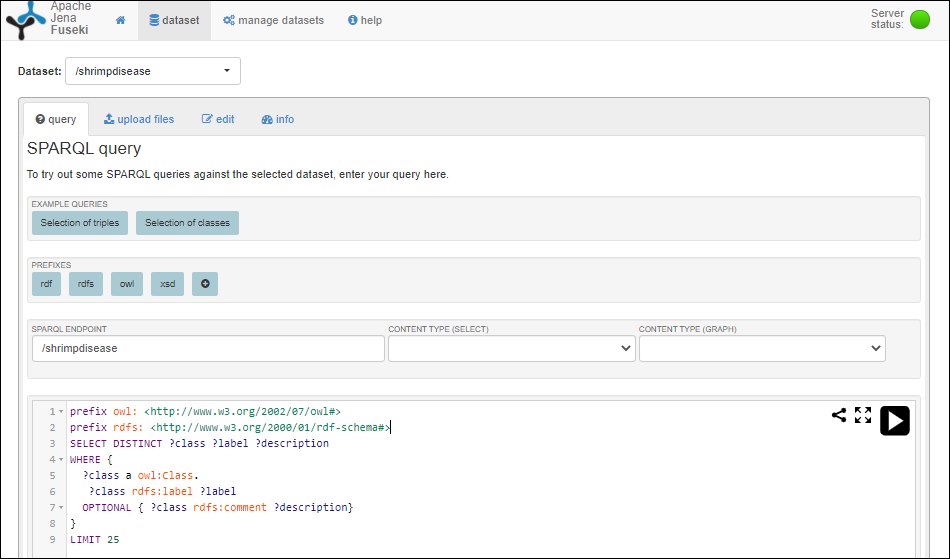
Hình 17. Tạo dataset trên server Jena Fuseki

Trước tiên, ta tạo một dataset (*endpoint*), là tên giúp website định danh để truy cập đến. Để khởi tạo dataset, cần nhập tên dataset và lựa chọn loại lưu trữ cho dataset là sẽ tạo lại dữ liệu hay giữ nguyên qua mỗi lần khởi động lại server.



Hình 18. Tải file ontology lên server

Từ dataset đã tạo, ta tải file ontology lên để điền dữ liệu vào dataset, đồng thời dữ liệu sẽ chuyển đổi sang mô hình RDF để có thể truy cập từ website thông qua các ngôn ngữ truy vấn RDF.

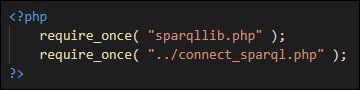


Hình 19. Truy vấn câu lệnh sparql trên Jena Fuseki

Trên server Jena Fuseki, người dùng có thể nhập câu lệnh SPARQL và thực thi thử nghiệm các câu truy vấn trước khi sử dụng truy vấn dữ liệu vào website. Server còn có công cụ ví dụ các câu truy vấn đơn giản như lấy ra các lớp, lớp con và các thông số như nhãn, ghi chú.

1. **Thư viện Sparqllib**

**Import thư viện sparqllib**



Hình 20. Import thư viện sparqllib

Dùng hàm *require\_once(<ten\_thu\_vien>)* để nhúng thư viện sparqllib và các thư viện php cần thiết khác vào. Các hàm và phương thức được viết trong thư viện *connect\_sparql.php*.

**Kết nối vào cơ sở dữ liệu**



Hình 21. Lấy kết quả từ CSDL

Hàm *sparql\_get*giúp kết nối vào endpoint tại địa chỉ *$endpoint* và thực hiện câu truy vấn sparql là *$sparql*sau đó trả về dữ liệu dạng bảng*.*

**Một số hàm / phương thức trong thư viện**

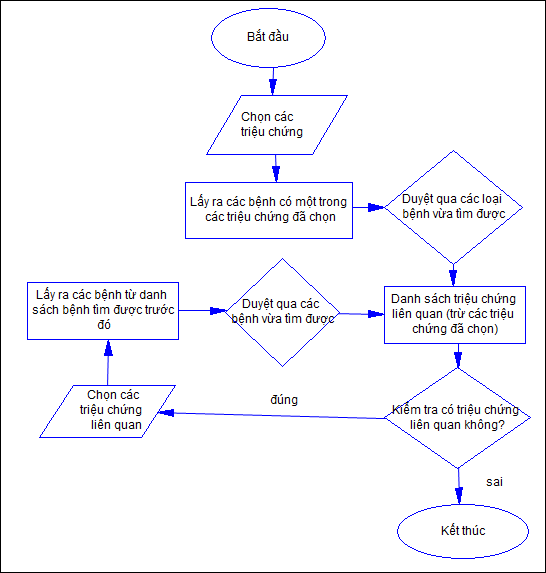
Bảng 7. Một số hàm / phương thức trong thư viện sparqllib

|  |  |
| --- | --- |
| Tên hàm / phương thức | Diễn giải |
| sparql\_connect( $ep ) | Kết nối đến cơ sở dữ liệu có url là $ep |
| sparql\_errno( $db) | Trả về lỗi khi kết nối vào endpoint $db |
| sparql\_get( $ep, $sparql ) | Kết nối đến endpoint $ep và thực hiện câu truy vấn $sparql và trả về kết quả dạng bảng |
| sparql\_query($sparql) | Giúp hực hiện câu truy vấn dữ liệu sparql và trả về dữ liệu dạng bảng |
| sparql\_fetch\_all( $result ) | Lấy ra tất cả các dòng từ kết quả dạng bảng |
| sparql\_fetch\_array( $result ) | Lấy ra từng dòng dữ liệu trong bảng |
| sparql\_num\_rows( $result ) | Đếm số dòng trong bảng |
| sparql\_field\_array( $result ) | Lấy ra tất cả các cột |
| sparql\_field\_name($result, $i) | Lấy ra dữ liệu ở cột $i trong kết quả dạng bảng $result |

Các hàm và phương thức trên được dùng để truy xuất dữ liệu RDF trong cơ sở dữ liệu. Người dùng có thể chọn một trong các cách khác nhau để kết nối vào cơ sở dữ liệu. Hầu hết tất cả các kết quả trả về đều ở dạng bảng, người lập trình có thể tùy theo mục đích sử dụng dữ liệu mà lại có các cách lấy dữ liệu theo từng dòng hay từng cột tùy ý.

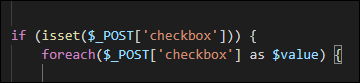
1. **Chẩn đoán bệnh**

* **Lưu đồ dữ liệu**



Hình 22. Lưu đồ dữ liệu chẩn đoán bệnh

Đầu tiên, ứng dụng chẩn đoán bệnh nhận vào danh sách các triệu chứng (tc), từ đó tìm ra được danh sách các bệnh (lb). Sau đó, từ danh sách bệnh, tìm ra được các triệu chứng liên quan (tc2). Tiếp theo, ta chọn từ các triệu chứng liên quan và tiếp tục chẩn đoán. Hệ thống sẽ giới hạn lại các loại bệnh (dựa vào lb và tc2) và danh sách lb sẽ được cập nhật kèm theo tc2 cũng được cập nhật. Đến khi nào tc2 rỗng, hệ thống sẽ dừng lại.



Hình 23. Phương thức POST

Hệ thống chủ yếu sử dụng phương thức POST để truyền dữ liệu qua các trang.

* **Thiết kế giao diện**

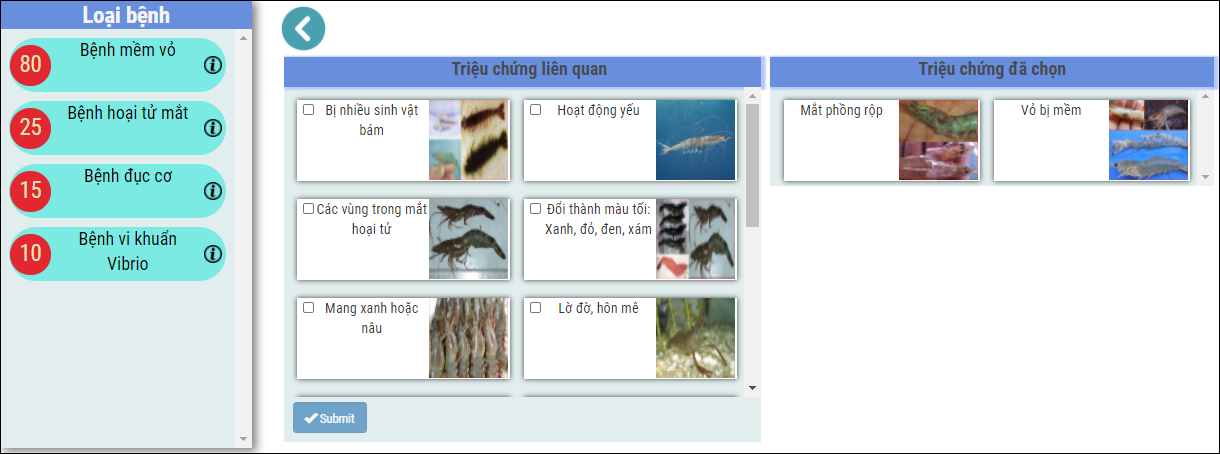
**Giao diện chẩn đoán bệnh**



Hình 24. Giao diện chẩn đoán bệnh

Khi vào giao diện chẩn đoán bệnh, hệ thống sẽ hiện ra các nhóm loại bệnh để người dùng có thể lựa chọn. Sau khi chọn nhóm bệnh, tất cả các triệu chứng của nhóm bệnh đó sẽ hiển thị cho người dùng lựa chọn.

**Giao diện kết quả chẩn đoán sơ bộ**



Hình 25. Giao diện kết quả chẩn đoán sơ bộ

Giao diện chẩn đoán sơ bộ hiển thị kết quả là các loại bệnh kèm theo tỷ lệ mắc bệnh. Từ các bệnh đó, tìm ra được các triệu chứng liên quan đến các bệnh và hiển thị cho người dùng chọn và tiến hành chẩn đoán nâng cao. Nếu không có triệu chứng liên quan, hệ thống sẽ dừng lại.

**Giao diện kết quả chẩn đoán nâng cao**



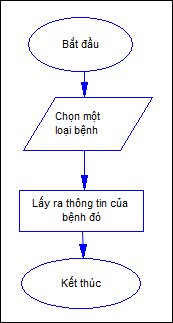
Hình 26. Giao diện kết quả chẩn đoán nâng cao

Giao diện chẩn đoán nâng cao sẽ hiển thị các loại bệnh đã được giới hạn lại và làm mới các triệu chứng liên quan để người dùng tiếp tục chọn. Ở bước này, hệ thống có thể thự hiện lặp lại cho đến khi nào không còn triệu chứng liên quan nữa. Đến lúc đấy, loại bệnh sẽ chỉ còn lại một bệnh duy nhất và tỷ lệ sẽ 100%.

1. **Tra cứu**

* **Tra cứu bệnh**

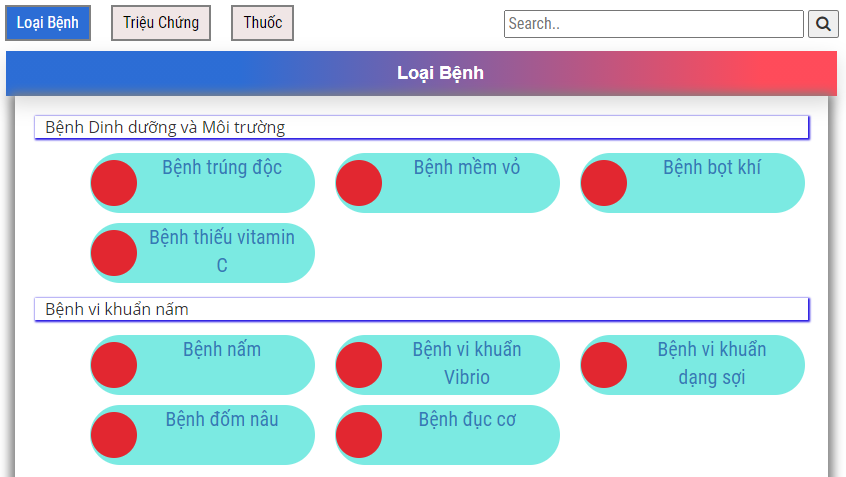
**Lưu đồ tra cứu bệnh**



Hình 27. Lưu đồ tra cứu bệnh

Đầu tiên chọn một bệnh trong danh sách các bệnh. Sau đó, hệ thống lấy ra các thông tin của bệnh đó. Chức năng tra cứu bệnh sử dụng phương thức GET để truyền dữ liệu.

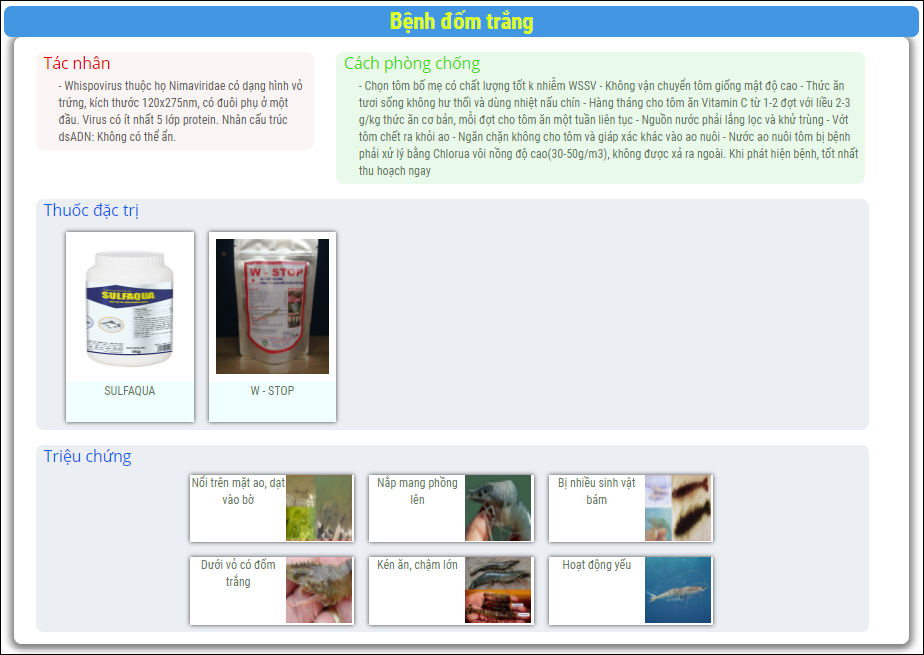
**Giao diện tra cứu bệnh**



Hình 28. Giao diện tra cứu bệnh

Giao diện tra cứu bệnh hiển thị tất cả các loại bệnh để người dùng chọn. Sau khi chọn sẽ xem được các thông tin về bệnh đó. Ngoài ra, người dùng có thể tìm kiếm theo tên của loại bệnh.

**Giao diện thông tin chi tiết bệnh**

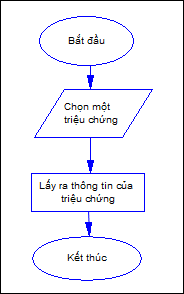


Hình 29. Giao diện thông tin chi tiết bệnh

Giao diện thông tin chi tiết bệnh hiển thị thông tin về bệnh như tác nhân gây bệnh, cách phòng trị. Ngoài ra, còn lọc ra danh sách các thuốc đặc trị và các triệu chứng của bệnh đó.

* **Tra cứu triệu chứng**

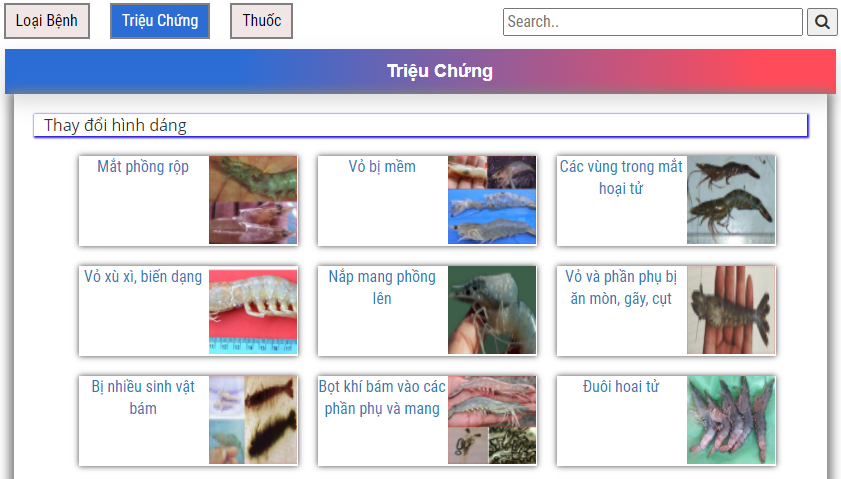
**Lưu đồ dữ liệu**



Hình 30. Lưu đồ dữ liệu tra cứu triệu chứng

Hệ thống nhận vào một triệu chứng được chọn, sau đó lấy ra các thông tin của triệu chứng kèm theo danh sách các bệnh tương ứng. Chức năng tra cứu triệu chứng sử dụng phương thức GET để truyền dữ liệu.

**Giao diện tra cứu triệu chứng**



Hình 31. Giao diện tra cứu triệu chứng

Giao diện tra cứu triệu chứng hiển thị tất cả các triệu chứng được phân loại theo nhóm để người dùng có thể chọn.

**Giao diện thông tin chi tiết triệu chứng**

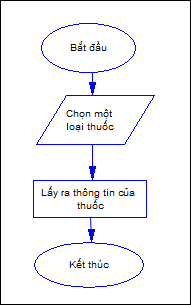


Hình 32. Giao diện thông tin chi tiết triệu chứng

Giao diện thông tin chi tiết triệu chứng hiển thị hình ảnh triệu chứng và tìm ra danh sách các bệnh có thể mắc phải.

* **Tra cứu thuốc**

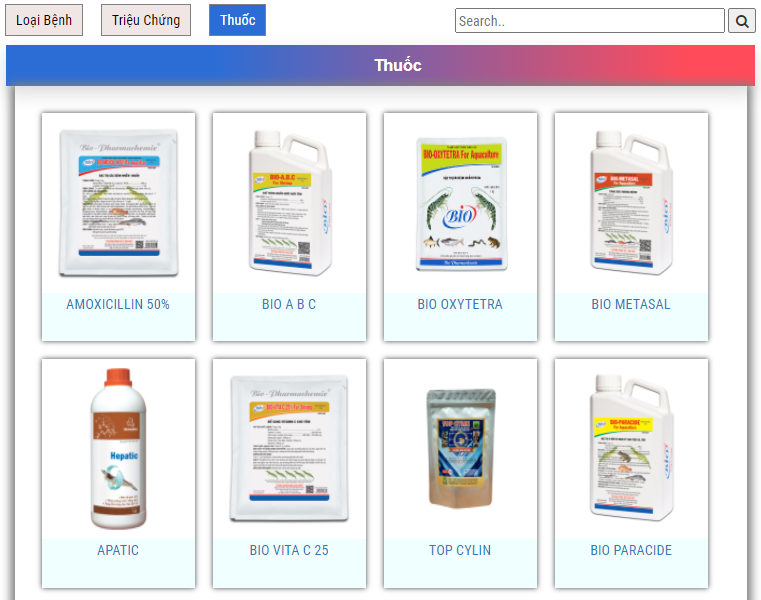
**Lưu đồ dữ liệu**



Hình 33. Lưu đồ dữ liệu tra cứu thuốc

Hệ thống nhận vào một loại thuốc, sau đó tìm và lấy ra các thông tin của thuốc kèm danh sách các bệnh mà thuốc có thể đặc trị. Chức năng tra cứu thuốc sử dụng phương thức GET để truyền dữ liệu qua các trang.

**Giao diện tra cứu thuốc**



Hình 34. Giao diện tra cứu thuốc

Giao diện tra cứu thuốc sẽ hiển thị tất cả các loại thuốc để người dùng xem và tiến hành chọn. Ngoài ra, người dùng có thể nhập tên thuốc ở ô phía trên góc phải để tìm kiếm một số loại thuốc.

**Giao diện thông tin chi tiết thuốc**



Hình 35. Giao diện thông tin chi tiết thuốc

Giao diện thông tin chi tiết thuốc hiển thị hình ành thuốc, hướng dẫn sử dụng và danh sách các bệnh mà thuốc đặc trị.

**CHƯƠNG 3: ĐÁNH GIÁ VÀ KIỂM THỬ**

1. **Mục tiêu kiểm thử**

Mục tiêu chính của kiểm thử chủ yếu là chức năng chẩn đoán bệnh tôm bao gồm các tiêu chuẩn sau:

* Kiểm tra xem ứng dụng có hoạt động tốt không
* Có phát sinh lỗi gì không
* Kết quả trả về có đúng hay không

Nhìn chung, mục tiêu cuối cùng vẫn là tìm ra các lỗi để khắc phục, sửa chữa trong tương lai.

1. **Kịch bản kiểm thử**

Bảng 8. Kịch bản kiểm thử

|  |  |
| --- | --- |
| Ký hiệu kịch bản | Kịch bản kiểm thử |
| KB001 | Truy cập vào chức năng kiểm thử, kiểm tra xem có đúng danh sách triệu chứng, có xuất hiện lỗi không? |
| KB002 | Tiến hành chẩn đoán sơ bộ và kiểm tra kết quả gồm danh sách bệnh có đúng không, có đúng tỷ lệ không, các triệu chứng liên quan có đúng không? |
| KB003 | Tiến hành chẩn đoán nâng cao, kiểm tra loại bệnh có ít lại không, tỷ lệ bệnh có đúng không, các triệu chứng liên quan còn lại có đúng không? |
| KB004 | Truy cập vào chức năng tra cứu bệnh, sau đó chọn loại bệnh và kiểm tra kết quả xem có đúng thông tin của bệnh không, các triệu chứng và loại thuôc có đúng không? |
| KB005 | Truy cập vào chức năng tra cứu triệu chứng, sau đó chọn triệu chứng và kiểm tra kết quả xem có đúng thông tin triệu chứng chưa, những loại bệnh có thể mắc phải có đúng không? |
| KB006 | Truy cập vào chức năng tra cứu thuốc, sau đó chọn loại thuốc và kiểm tra kết quả xem có đúng thông tin thuốc không, danh sách loại bệnh có đúng không? |

Tiến hành kiểm thử các kịch bản ở bảng bên trên.

Môi trường kiểm thử

Thiết bị kiểm thử: Laptop Dell 3443 Inspiron

Cấu hình thiết bị:

* Chip Intel(R) Core(TM) i7-5500U
* RAM 4 GB
* Hệ điều hành Windown 10
* Trình duyệt Google Chrome

1. **Kết quả kiểm thử**

Bảng 9. Kết quả kiểm thử kịch bản

|  |  |
| --- | --- |
| Kịch bản | Mức độ đạt yêu cầu |
| KB001 | Đạt |
| KB002 | Đạt |
| KB003 | Đạt |
| KB004 | Đạt |
| KB005 | Đạt |
| KB006 | Đạt |

Nhìn chung, các kết quả kiểm thử đều đạt yêu cầu.

.

**PHẦN 3: KẾT LUẬN**

1. **Kết quả đạt được**
2. **Kết quả**

Hoàn thành trọng tâm nghiên cứu là cơ sở dữ liệu phân cấp ontology bao gồm cách thiết kế, cách sử dụng và ý nghĩa của các thành phần bên trong. Hiểu được cách tổ chức dữ liệu của mô hình RDF và ngôn ngữ truy vấn dữ liệu SPARQL cũng như cách làm thế nào để kết nối từ trang web đến cơ sở dữ liệu.

Xây dựng được ứng dụng web chẩn đoán bệnh tôm với các chức năng như:

* Chẩn đoán bệnh
* Tra cứu:
* Tra cứu bệnh
* Tra cứu triệu chứng
* Tra cứu thuốc

1. **Hạn chế**

Ứng dụng có một số hạn chế:

* Cơ sở dữ liệu hiện là cơ sở dữ liệu tĩnh, không quản trị trực tiếp trên website được mà phải qua phần mềm thiết kế ontology
* Về phần giao diện, vẫn còn chưa được bắt mắt và chuyên nghiệp

1. **Hướng phát triển**

Trong tương lai, ứng dụng có thể phát triển thêm về:

* Ứng dụng có thể phát triển thêm chức năng quản trị, giúp cập nhật các danh sách loại bệnh, triệu chứng và thuốc cùng những thông tin chi tiết liên quan một cách nhanh chóng và chính xác hơn
* Có thể ứng dụng các công nghệ hỗ trợ thiết kế để giao diện tối ưu hơn

**TÀI LIỆU THAM KHẢO**

[1]. Ontology là gì. [Online].Có sẵn tại: [http://thuvien.tcdktcnsl.edu.vn/files/products/khai\_niem\_ve\_ontology\_L63.pdf.](file:///C:\Users\tuan\Downloads\%20https:\techmaster.vn\posts\33428\nodejs-la-gi-va-tai-sao-toi-nen-hoc-lap-trinh-nodejs.%20) [Truy cập tháng 10/2020].

[2] Wikipedia, “Ontology”. [Online].Có sẵn tại: https://vi.wikipedia.org/wiki/ Bản\_thể\_học\_(khoa\_học\_thông\_tin). Truy cập tháng 10/2020.

[3] Wikipedia, “RDF”. [Online].Có sẵn tại: https://vi.wikipedia.org/wiki/RDF. [Truy cập tháng 10/2020].

[4] W3C, “SPARQL Query Language for RDF”. [Online].Có sẵn tại: https://www.w3.org/TR/rdf-sparql-query/. [Truy cập tháng 11/2020].

[5] SPARQL RDF Library for PHP. [Online]. Có sẵn tại: http://graphite.ecs.soton.ac.uk/sparqllib/. [Truy cập tháng 11/10/2020].

[6] W3schools, “CSS”. [Online].Có sẵn tại: https://www.w3schools.com/w3css/default.asp. [Truy cập tháng 12/2020].

[7] W3schools, “JS”. [Online].Có sẵn tại: https://www.w3schools.com/js/DEFAULT.asp. [Truy cập tháng 12/2020].

[8] W3schools, “HTML”. [Online].Có sẵn tại: https://www.w3schools.com/html/. [Truy cập tháng 12/2020].

[9] TS. Bùi Quang Tề, “Bệnh của tôm nuôi và biện pháp phòng trị”, nhà xuất bản Nông nghiệp, Hà Nội, 2003.