

University of Leicester  
Department of Computer Science  
CO7201 Individual Project – Dissertation

**AUTHOR**

<b>Student Name</b>	<b>Email</b>
<b>ADWOA DANSOA DONYINA</b>	<a href="mailto:adwoa@mcs.le.ac.uk"><u>adwoa@mcs.le.ac.uk</u></a>

**Project title:** SAPIAN-VTHS (Virtual Teaching Hospital System)  
Electives networking project

**Project Supervisor:** Prof. Dr. Reiko Heckel

**Second Marker:** Dr. Artur Boronat

**Date of Submission:** 12 September 2008

## ABSTRACT

This report provides a detailed account of Adwoa Donyina’s development in the SAPIAN-VTHS (Virtual Teaching Hospital System) Electives networking Software development project. The main significant contribution to this software system is the re-implementation of an expert system into an artificial neural network system. Various critical design decisions were involved in the creation of the VTHS Neural Network. The main challenges presented in this project were due to the fact that Artificial Neural Networks is new and emerging technology; hence the system used the backpropagation algorithm which is one of the relatively new neural network methods.

“Backpropagation, or propagation of error, is a common method of teaching artificial neural networks how to perform a given task. It was first described by Paul Werbos in 1974, but it wasn’t until 1986, through the work of David E. Rumelhart, Geoffrey E. Hinton and Ronald J. Williams, that it gained recognition, and it led to a “renaissance” in the field of artificial neural network research.” [BP]

Various advanced concepts used in the development were taught to Adwoa in her CSC321 - Neural Networks undergraduate module at University of Toronto by Prof. Geoffery E. Hinton, who is “most noted for his work on the mathematics and applications of neural networks.”[GH] such as the backproagation algorithm, as stated in the above quote.

The aim of this project is to assist in teaching medical students the pedagogical thinking process, by modelling the business process. The objective of this project is to re-implement the diagnosis components of a working web- based diagnostic teaching system as an artificial neural network. This project is demanding because it involves re-engineering someone else’s code, and is in a non-familiar medical domain.

### DECLARATION

All sentences or passages quoted in this report, or computer code of any form whatsoever used and/or submitted at any stages, which are taken from other people’s work have been specifically acknowledged by clear citation of the source, specifying author, work, date and page(s). Any part of my own written work, or software coding, which is substantially based upon other people’s work, is duly accompanied by clear citation of the source, specifying author, work, date and page(s). I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this module and the degree examination as a whole.

Name: **ADWOA DANSOA DONYINA**

Signed:

Date:

## TABLE OF CONTENTS

<b>Abstract .....</b>	<b>2</b>
<b>1. Introduction.....</b>	<b>5</b>
1.1. Aims. ....	5
1.2. Objectives.....	6
1.3. Original results of the project.....	6
<b>2. Requirements.....</b>	<b>7</b>
2.1. Goals of the Project .....	7
2.2. Application Context .....	8
2.2.1. Description of the Problem Domain .....	8
2.2.2. Glossary .....	9
2.2.3. Relevant Facts and Assumptions .....	10
2.2.4. Model of the Problem Domain .....	10
2.2.5. Business Processes .....	11
2.3. Project Constraints.....	14
2.4. Functional Requirements .....	14
2.4.1. The Scope of the Work.....	14
2.4.2. The Scope of the Product.....	14
2.4.3. Functional and Data Requirements .....	15
2.5. Product Functions.....	21
2.5.1. Use Case Diagram.....	21
2.5.2. Description of Use Case Register Patient .....	22
2.5.3. Description of Use Cases Retrieve Patients old file.....	23
2.5.4. Description of Use Cases Medical History Management.....	24
2.6. Non-Functional Requirements .....	27
2.6.1. Look and Feel.....	27
2.6.2. Usability and Humanity.....	27
2.6.3. Performance .....	27
2.6.4. Operational.....	28
2.6.5. Maintainability and Support .....	28
2.6.6. Security.....	28
2.6.7. Cultural and Political.....	28
2.6.8. Legal.....	29
0	
<b>3. System Design .....</b>	<b>30</b>
3.1. Software Architecture .....	30
3.1.1. Neural Networks .....	31
3.1.2. Web database Application .....	35
3.2. Analysis Sequence Diagrams .....	37
3.3. Design Class Diagram .....	38
3.4. GUI Design .....	38
<b>4. Implementation.....</b>	<b>40</b>
4.1. Neural Network API.....	40

4.1.1. JOONE (Java Object Oriented Neural Engine) .....	40
4.1.2. Implementation of Java Neural Network.....	40
4.1.3. Use of NN API on current knowledge base.....	45
4.1.4. Preliminary Input Data Setup.....	46
4.2. Web Application.....	51
4.2.1. Resolved Re-engineering Issues .....	51
4.2.2. User-interface modifications.....	51
<b>5. Evaluation Of The Product .....</b>	<b>53</b>
5.1. Experiment Trial #1 .....	53
5.2. Experiment Trial #2.....	54
5.3. Experiment Trial #3.....	55
5.4. Testing on Simple Data.....	55
<b>6. Conclusion.....</b>	<b>57</b>
6.1. Summary .....	57
6.2. Related Work.....	57
6.2.1. Multi-attribute decision support systems in Healthcare .....	57
6.2.2. Artificial Intelligence in Medicine .....	58
6.2.3. Neural Networks in Medical Diagnosis .....	58
6.2.4. E-Learning .....	60
6.3. Future Work .....	60
6.4. Personal Experience.....	61
<b>7. Bibliography .....</b>	<b>62</b>
<b>Appendix A .....</b>	<b>65</b>
<b>Appendix B .....</b>	<b>66</b>

## 1. INTRODUCTION

This dissertation report will provide a detailed account of the SAPIAN-VTHS (Virtual Teaching Hospital System) Electives networking project with a thorough evaluation of the achievements, a critical comparison to the related works and a personal reflective analysis of the project as a whole. This report will also provide a clear description of the problem being solved and its usefulness.

This project began in October 2007, when consultation of design ideas took place with the primary customer John Omara, Part-Time clinical demonstrator, Leicester Medical School. On October 26, 2007 the gathering of initial requirements for the project took place, by interviewing the client Professor Stewart Petersen, Head of Medical Education at the University of Leicester, School of Medicine. Throughout the academic year of 2007-2008, these requirements were presented to the 2nd and 3rd year Computer Science project students of CO2015 (Software Engineering Project.) and CO3014 (Mathematics and Computer Science Project.) to develop some of the technical functionality for this project.

This project reengineered an existing prototype created by a 2<sup>nd</sup> year project group. On July 2<sup>nd</sup>, 2008 the prototype was successfully presented to Prof. Petersen. The medical school provided a flowchart detailing the required pedagogic teaching process for the VTHS system. The compliance of the prototype to the proposed flowchart was later achieved. Also the diagnosis component was re-implemented as an artificial neural network (ANN).

### 1.1 Project Aims

The aim is to assist teaching medical students the pedagogical thinking process, by modelling the business process of a clinical conversation. A clinical conversation is a patient centred approach, where the patient tells their story in their way and the physician listens to determine a diagnosis, while combining the knowledge of the patient's history. The teaching system will assist medical students in rehearsing the problem solving process and help determine what patient information is needed to determine a possible diagnosis. This expert system should provide guidance and direction with the evolving notion of what might support or refute the diagnosis; this pedagogical thinking process is an important skill in medicine, which is used for reasoning and decision making of diagnoses.

### 1.2. Project Objectives

The objective of this project is to re-implement the diagnosis components of a working web- based diagnostic teaching system as an artificial neural network (ANN), because the original system's knowledge base was hard to maintain. The original version of the VTHS system will continue to facilitate students in setting up a hypothesis of a clinical decision by relating probable diagnoses to the mass of science,

in order to correctly fine tune treatment for a patient; however portions of the user-interface will be re-engineered to cater to the Leicester Medical School's five year clinical course.

### **1.3. Original results of the project**

The project was successfully re-engineered creating flexibility of scripting path code to the original VTHS web system. Minor modifications to the user-interface of the original prototype were made and the primary goal of transforming an expert system into a neural system was achieved. The expert system was successfully transformed into a neural network system with the JOONE Neural Network framework the Java class was able to create, train and test Neural Networks. The Java Neural Network was linked to the MySQL database for training data. The objective was experiment if the project was feasible; unfortunately due to overfitting the neural network had a high error. However the neural network had low error rates when tested on different training sets.

## 2. REQUIREMENTS

In this section we will present the system requirements. The structure of this section is based on a combination of the Volere Requirements Specification template [RR] and the CO1006 Software Engineering & System Design- Requirements Document Template [HR].

**Project:** SAPIAN-VTHS (Virtual Teaching Hospital System) Electives networking Project

**Client:** Professor Stewart Petersen, Head of Medical Education at the University of Leicester, School of Medicine

**Contractor:** Adwoa Donyina, University of Leicester

The primary client for the product is Professor Stewart Petersen, Head of Medical Education at the University of Leicester, School of Medicine.

The secondary client for this product is Dr. Jonathan Hales, Phase I Coordinator at the University of Leicester, School of Medicine.

Eventually the clients intend to use this product to assist students in studying for clinical module exams at Leicester Medical School.

The customers for the product are lecturers and professors of Leicester Medical School represented by John Omara, Part-Time clinical demonstrator, Leicester Medical School.

Potential customers for the product include lecturers and professors of other medical schools located in United Kingdom.

Other stakeholders include sponsors, testers, system designers, domain experts and other experts.

### 2.1. Goals of the Project

#### *Purpose of the system*

**Purpose:** To provide a diagnostic teaching aid, that models a clinical conversation.

**Advantage:** It will assist in developing medical students' pedagogical thinking process.

**Measurement:** The advantage can be measured through student assessment. (If a student practices a minimum of 25 cases the measure of their testing performance should improve by 5%)

**Reasonable:** The advantage is greater than the product construction effort

The University of Leicester, School of Medicine requires an e-learning system to assist teaching medical students the pedagogical thinking process. The system should provide patient management and diagnostic aiding capabilities for students and medical school staff.

### **Motivation and goals**

In order to model the business process of a clinical conversation the following aspects are required:

- simple patient registration
- easy access to patient history
- clear display of past diagnostic cases
- good educational perspective of the diagnostic pedagogical thinking process

These goals should be achieved within the limited time frame provided.

### **Background and Experience of Users**

**User Group:** medical school students, academic staff members

**Subject matter experience:** range from novice to master

**Technical experience:** novice level

Medical school students and academic staff members are members of the university community and are assumed to have general computer skills. It cannot be assumed that the user has expert computer knowledge because they are in the medical domain. However, users can be trained on the software to perform sophisticated functions, such as the ability to modify clinical data.

## **2.2. Application Context**

This section will present the application area of the system and explain relevant technical terms and important business processes.

### ***2.2.1. Description of the Problem Domain***

The medical schools in United Kingdom teach a structured method for gathering medical history on the wards.

Medical students at the University of Leicester, School of Medicine practice gathering medical history from patients. These patients are either real or role playing actors. Each medical student in phase 2 is required to submit Summary of Portfolio Cases document as shown in the template (Appendix A).

The diagram below is an illustration of the problem domain of taking medical history.



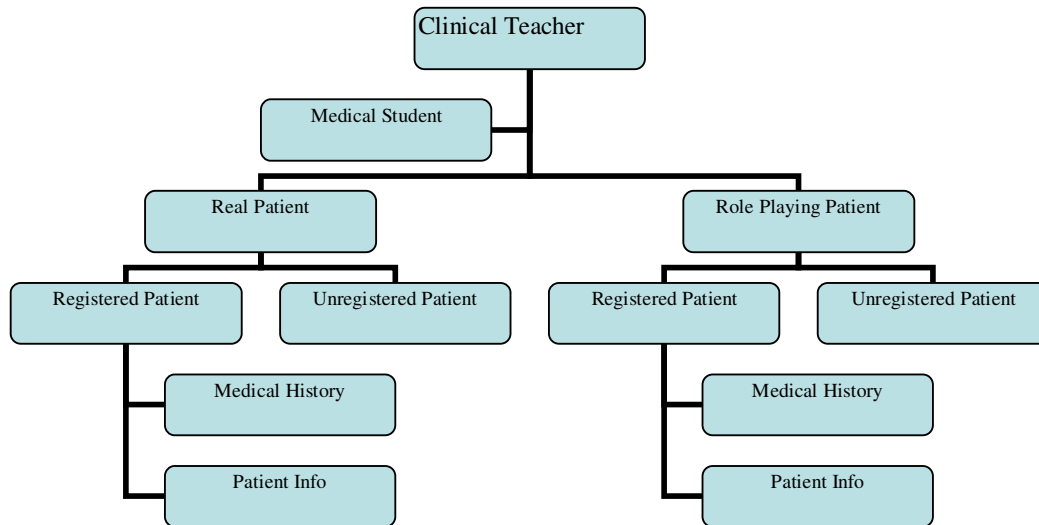


Diagram 1: Problem Domain- Taking Medical History in UK

### 2.2.2. Glossary

**Clinical conversation:** Patient centred approach, where the patient tells the story in their own way and the physician listens to determine a diagnosis, while combining the knowledge of the patient’s history.

**Medical History:** All relevant information gathered from the patient about their illness, their co-existing problems, their current drug treatment, their significant history, their social background and their family background. [UM]

**Analysis of Medical History:** Most likely single cause of the patient’s presentation and the other causes that need to be taken into account and the reasons for the choices. [UM]

**VTHS:** Virtual Teaching Hospital System

**Pedagogical Process:** “systematic transfer of knowledge and - or skills from an instructor to a learner. Depending upon the basic objectives of the process, the transfer may be limited to cognition of facts or may be extended to application and extension of facts, their inter-linkages and derivative concepts.” [YP]

**Epidemiological parameters:** represents patient parameters such as age, sex, race, and disease onset. [TN]

**Pathological parameters:** represents medical symptoms presented such as vomiting, nausea, night sweats, and laboratory data. [TN]

### 2.2.3. *Relevant Facts and Assumptions*

The current diagnosis data were manually accumulated by an expert in the medical domain. The expert generated predictive values for the relativity of a diagnosis based on epidemiological and pathological parameters and accumulated a “diagnosis formula.”

This “diagnosis formula” was used to calculate the final predictive value for each diagnosis in the VTHS expert system as shown below.

$$Dpv = f [Pe_1 \times Pe_2 \times Pe_3 \dots \times Pe_n] [Pp_1 + Pp_2 + Pp_3 \dots + Pp_n]$$

where

Dpv = Disease predictive value

Pe<sub>i</sub> = epidemiological parameter predictive values

Pp<sub>i</sub> = pathological parameters predictive values

Formula 1: Diagnosis formula [TN]

The epidemiological parameters are multiplied together and the pathological parameters are added together. The intermediate values are then multiplied together to get the final predictive value for a particular diagnosis.

### 2.2.4. *Model of the Problem Domain*

This section provides a UML class diagram of the problem domain, discussed in section 2.2.1. The diagram was broken down into four critical classes. The Student class represents a medical student, who interacts with their clinical teacher. The teacher can have zero or more students; this is represented with the asterisk symbol ‘\*’. Each medical student in phase 2 clerks a patient generating medical history, which is presented to the supervisor for marking.

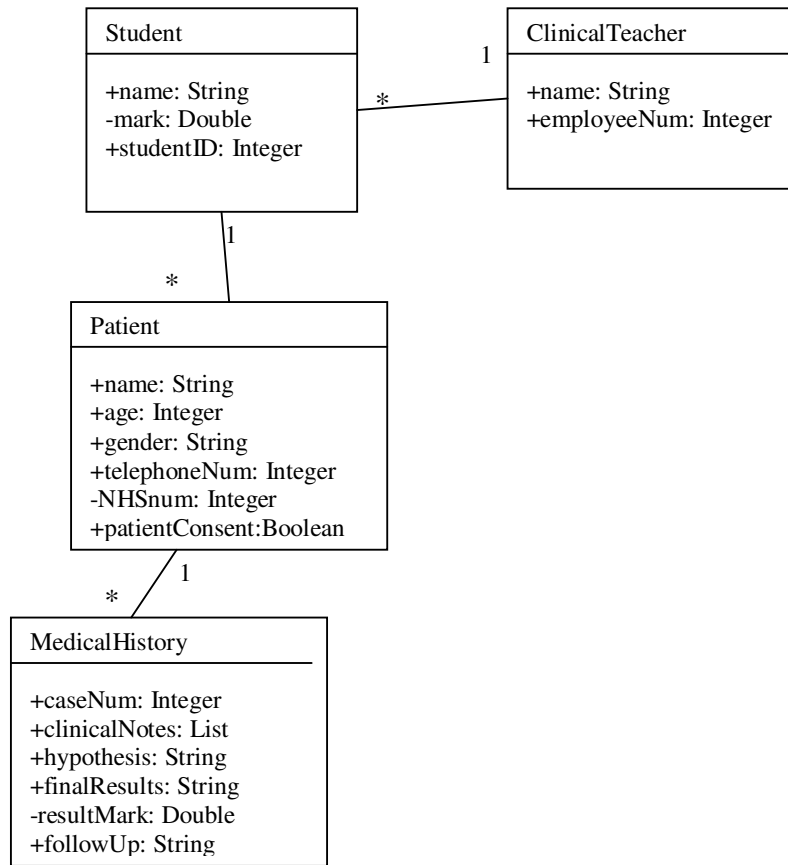


Diagram 2: Class Diagram of Problem Domain

### 2.2.5. Business Processes

Relevant processes include:

- Generate medical history on patient’s case
- Register patient

#### **Process Medical History: Generate medical history on patient’s case**

Process Details for Medical History

<b>Trigger event:</b>	The student requests a new patient from his/her clinical teacher.
<b>Result:</b>	Medical student has new medical history case to add to their Summary of Portfolio Cases document.
<b>Participants:</b>	Medical student, patient, clinical teacher

## Process Description Medical History

### **Main scenario**

#### **Time-tabling**

Students and teachers may want to develop their own methods, but one that seems to work well is as follows:-

1. The teacher selects a suitable patient.
2. The student clerks the patient making clinical notes as usual.
3. The student presents the findings to the teacher either at the same session or within 1 or 2 days.
4. The teacher reviews the quality of the history and examination at the bedside with the student. This should involve direct observation of the student repeating at least some of the clerking under supervision.
5. Discussion then takes place around the case including hypothesis generation, the appropriate use of investigations and issues of management.
6. The student then writes up the case on their own in the agreed format, incorporating the issues discussed with the teacher and the results of their own reading.
7. On a separate occasion the result is either marked by the teacher or presented verbally by the student for critical comment. (The model here is the Oxbridge essay).
8. The student amends the case document in the light of these comments and retains it for private study. There is no further teacher involvement unless the student specifically requests it.

Outline 1: Business Process description of main scenario[UM]

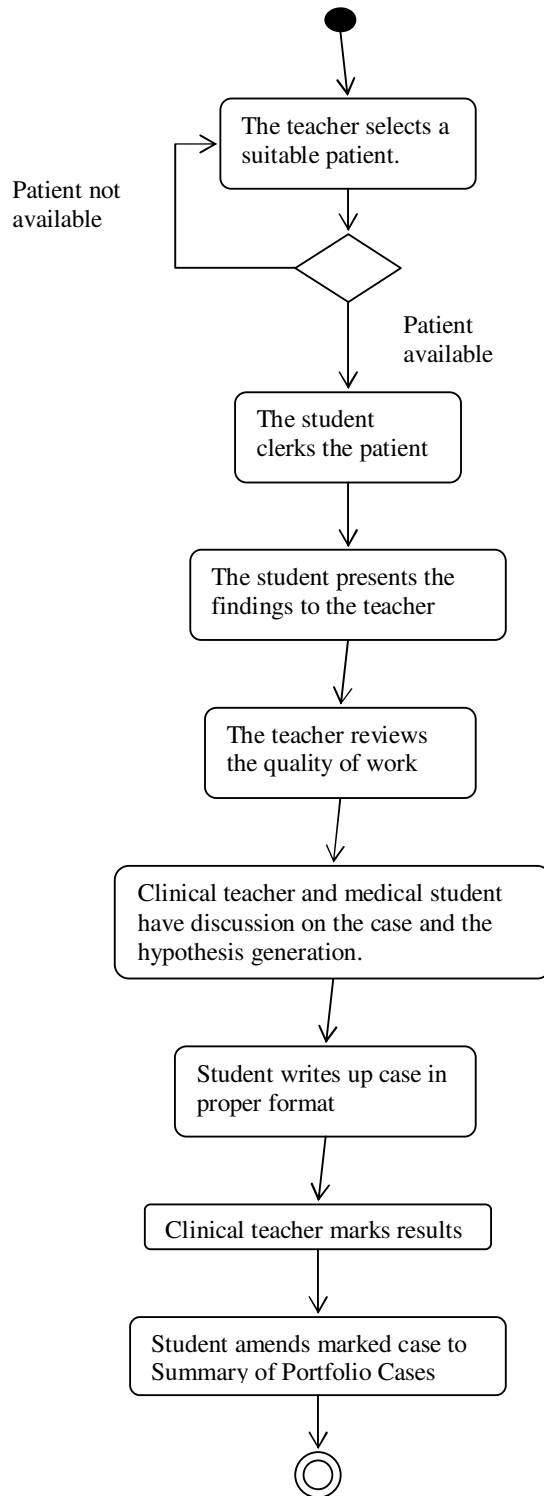


Diagram 3: Activity Diagram for Medical History

## 2.3. Project Constraints

In this section restrictions on product will be presented.

### Time Constraints

This project has a time constraint of only 2 months of development time; however it is not required by the medical school until July 2009.

### Environmental Constraints

This is a client-server application, such that patient and user information is held on a database server. The product must be web-based, because it provides easy access and fast upgrading; hence the system will be hosted on the internet.

### Hardware Environment

Clients' computers must be able to run graphical user interface.

### Software Environment

The product shall build-on the VTHS system developed by Bapodra M., Chong J., Kakayev K., Papaya D., Pickering J for their Computer Science 2<sup>nd</sup> year project. (Spring 2008) [BCKPP1]. It will re-engineer an existing version, which was developed using PHP, MySQL, Ajax, Javascript, and HTML. The server will run a web and data base server and a supporting operating system.[HR]

## 2.4. Functional Requirements

In this section we will present fundamental subject matters of the product.

### 2.4.1. *The Scope of the Work*

**The current situation:** Medical students practice clinical conversation in determining a diagnosis during sessions involving real or simulated patients.

**The context of the work:** Medical domain of the diagnosis field. Hence, we require real clinical cases for testing and training Neural Networks.

### 2.4.2 *The Scope of the Product*

**Product Boundary:** The following Use Case Diagram represents the current VTHS system structure.

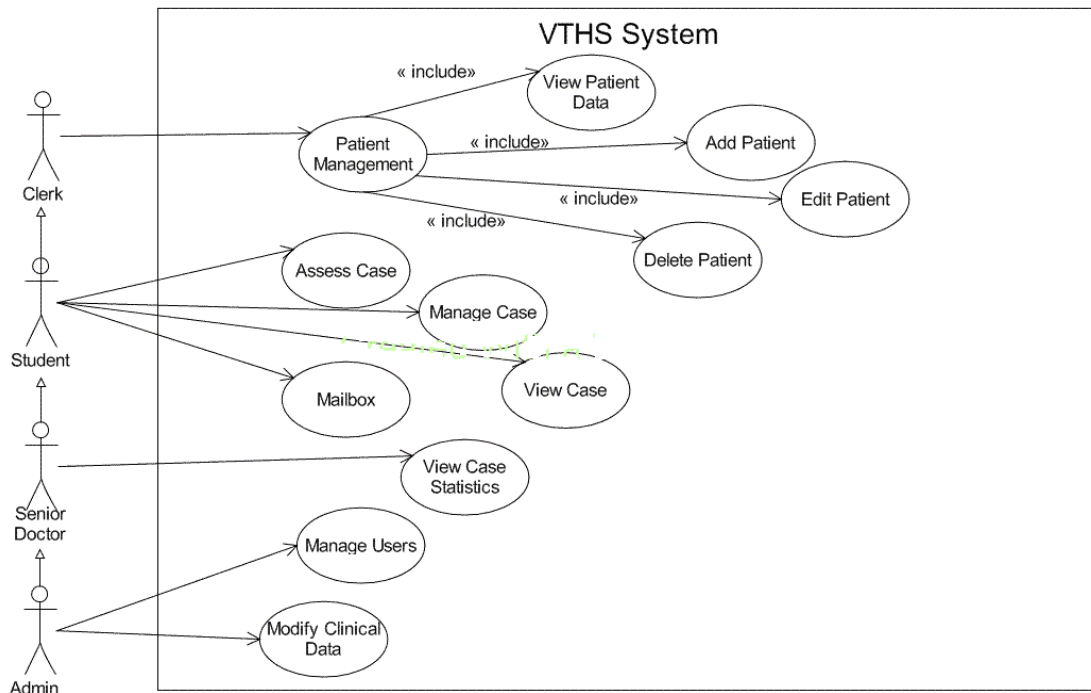


Diagram 4: Use Case Diagram of system at start of project (version: June 2008)

### 2.4.3 Functional and Data Requirements

**Functional Requirement:** Build upon existing system requirements

**Rationale:** The Virtual Teaching Hospital System should assist with teaching students to recognise key symptoms or red flags for particular diagnosis. The user interface of the existing system is shown in diagram 2. The system only allows students to enter one parameter of epidemiological or pathological patient data at a time. The student is required to enter their reasoning prior to proceeding in the diagnosis process. This supports the teaching of the pedagogical thinking process. The display of resulting diagnosis outcomes are lightened or darkened in colour based on calculated Disease predictive value as described in section 2.2.3.

**Fit criterion:** Maintain the initial requirements of the current working diagnosis teaching system.

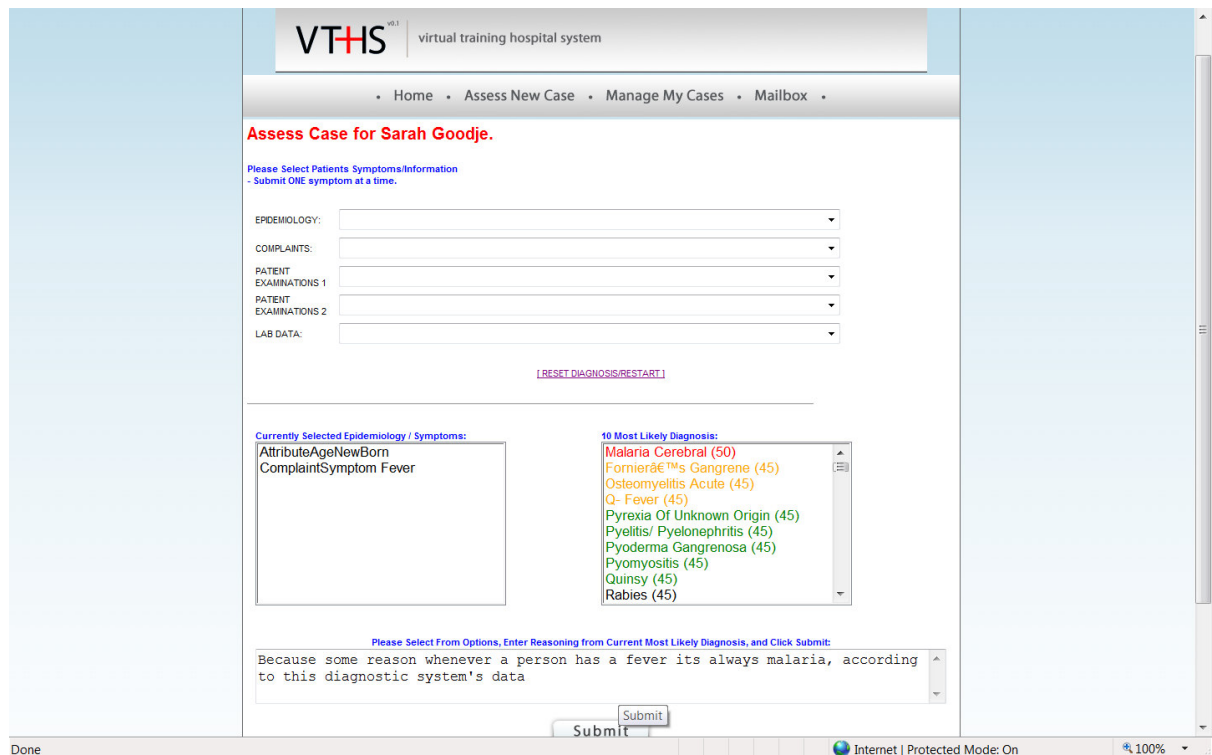


Diagram 5: Existing System [BCKPP1]

The first aim is to improve the pedagogical thinking process elements of the system. The client presented a detailed flowchart outlining requirements of the ‘VTHS Pedagogic Teaching Process’, as shown in its entirety in Appendix B. In order to achieve this first aim of improving the elements for the pedagogical thinking process, re-engineering of the existing the system is required in order to ensure compliance with the provided flowchart. The compliance with the flowchart will be discussed in three separate parts: Start (Portion Prior to Pedagogical Rectangle), Middle (Pedagogical Rectangle) and End (After Pedagogical Rectangle). Also note, that this teaching process is a cyclical process; hence multiple patient cases can be presented one at a time.

The current VTHS system is in conformity with the flowchart in most elements; however a few modifications are necessary. This flowchart corresponds to a student level access to the system. The system has the capabilities of searching an existing patient or registering a new patient, as described in diagram 6.



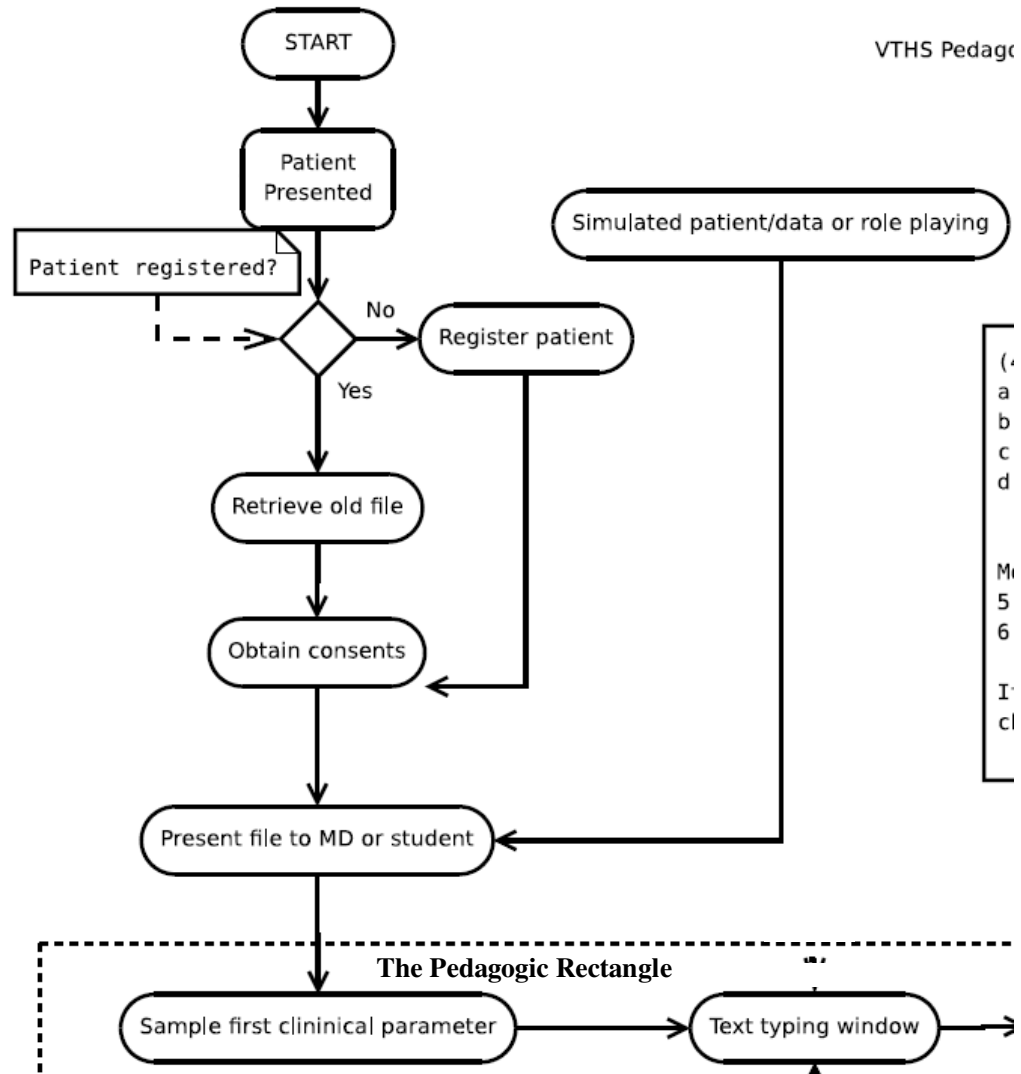


Diagram 6: Start of ‘VTHS Pedagogic Teaching Process’ [PS]

If an existing patient is selected the personal details of that patient are displayed. The system needs to be re-engineered to obtain consents from the patient, prior to displaying the medical history.

**Functional Requirement:** Add a consents page like a disclaimer page, prior to proceeding with diagnosis patient.

**Rationale:** The consent required is permission for the student to use their medical case.

**Fit criterion:** The terms should be presented with a checkbox stating that the student identified themselves.

The system then proceeds into the diagnostic section. All aspects of the Pedagogic Rectangle in diagram 7 correspond exactly with the existing system, except for the parameter deselect option.

**Functional requirement:** Add the ability to deselect Epidemiological or Pathological parameters.

**Rationale:** Provides the user with the ability to change their mind.

**Fit criterion:** To be in compliance with the provided VTHS Pedagogic Teaching Process Flowchart.

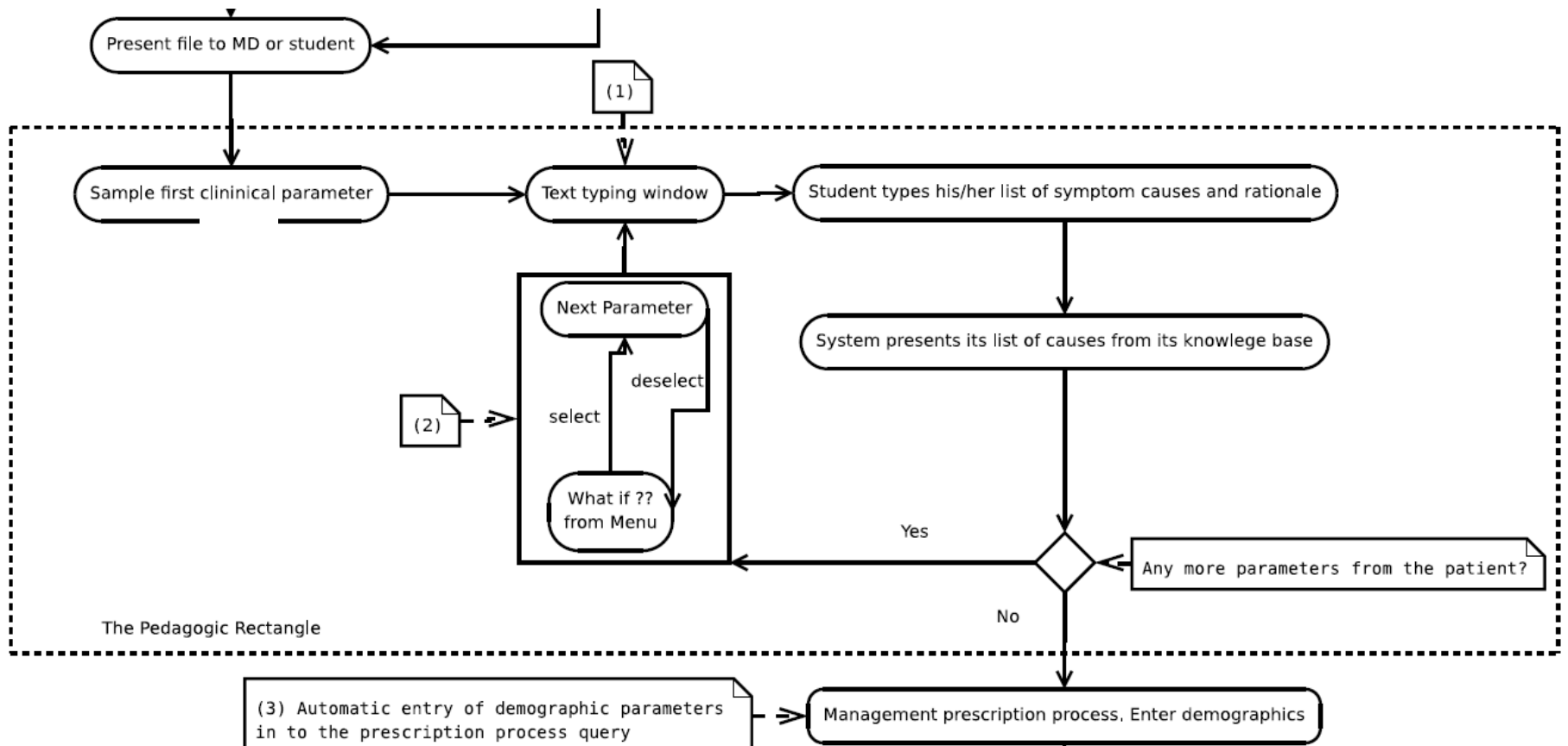


Diagram 7: Middle of 'VTHS Pedagogic Teaching Process' [PS]

The prescription management process, which is outlined in diagram 8, is partly implemented in the current system; however the prescription and treatment data needs to be updated.

**Functional Requirement:** Update prescription and treatment data.

**Rationale:** In order to provide the user with up to date data.

**Fit criterion:** In order to conform to the given 'VTHS Pedagogic Teaching Process' Flowchart.

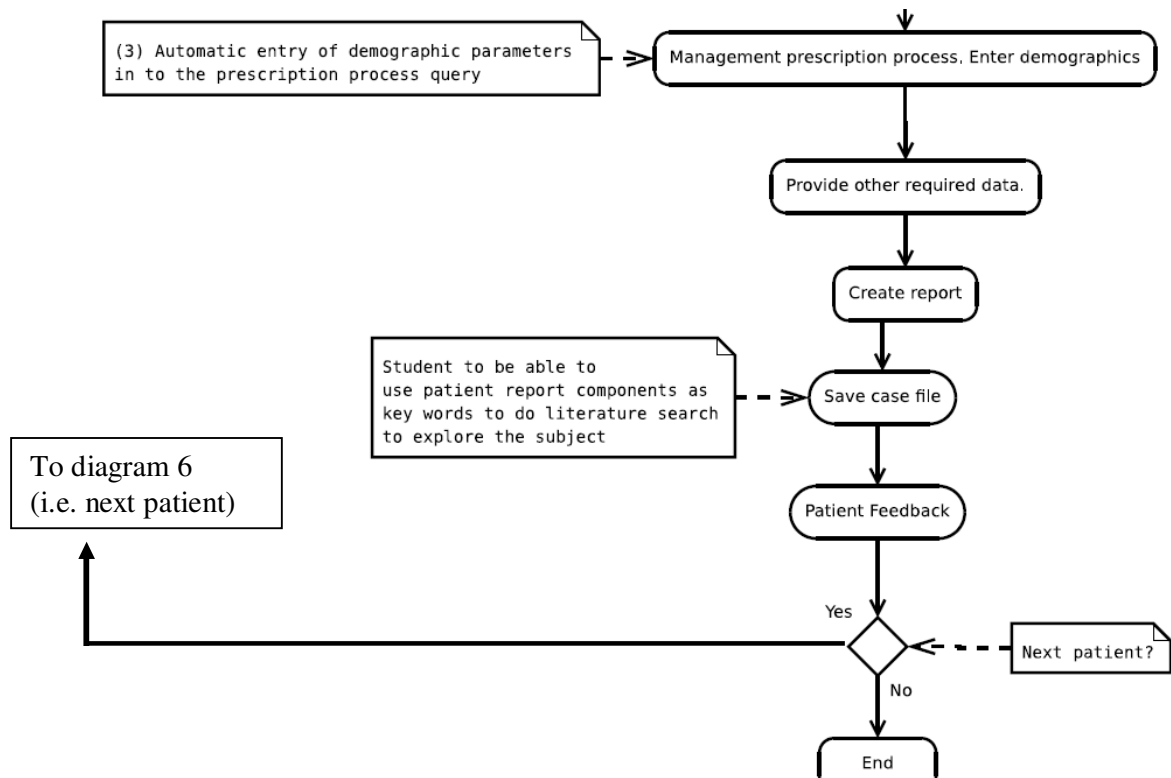


Diagram 8: End of 'VTHS Pedagogic Teaching Process' [PS]

**Additional Requirements**

**Functional Requirement:** Link the epidemiology patient details automatically to the diagnosis

**Rationale:** Provides more accurate diagnosis

**Fit criterion:** Reduce user from inputting duplicate patient data

**Functional Requirement:** enhancing the quality of diagnosis data.

**Rationale:** To insure this system is helpful in supporting medical student studying.

**Fit criterion:** Insure that all the key clinical presentations from Phase I and II are present in the system.

**Data Requirement:** Gather all key clinical presentations from Phase I and II, MBChB, Leicester Medical School

**Gathered books/ Documents:**

- Phase 2 Course Document. [UM1]
- Phase 1 Course Document, Five Year Course, MBChB. [UM2]
- Phase 1 Integrated Medical Sciences Written Assessment, MB ChB Phase 1. [UM3]
- Integrative Module Workbook, MB ChB Phase I. [UM4]

## 2.5. Product Functions

This section is a detailed account of the high-level system functionality.

### 2.5.1. Use Case Diagram

This diagram includes only the most obvious functions and those mentioned explicitly in the VTHS Pedagogic Teaching Process Flowchart (Appendix B).

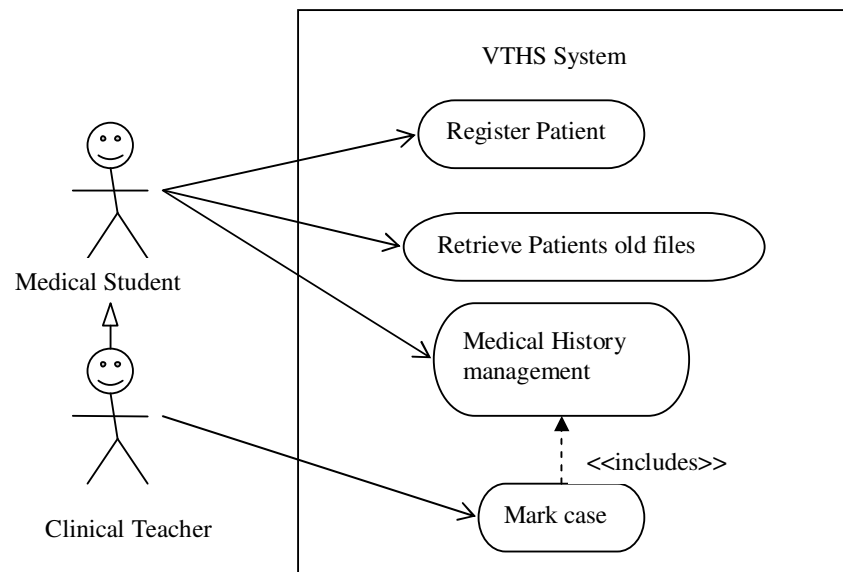


Diagram 9: Subsection of Use Case

The Clinical Teacher sub actor has all capabilities of the medical student actor, plus the ability to mark medical history cases.

## 2.5.2. Description of Use Case Register Patient

### Details of Use Case Register Patient

<b>Name:</b>	Register Patient
<b>Primary Actor:</b>	Medical Student
<b>Goal (of the User):</b>	Add patient to VTHS system database
<b>Precondition:</b>	Patient is new to the database
<b>Postcondition (successful outcome):</b>	Patient is successfully added to the database
<b>System (implementing it):</b>	VTHS system
<b>Participating Actor:</b>	None

### Main Scenario for Use Case Register Patient

Step	Actor/System	Description of Activity
1	Medical Student	Verifies patient is new
2	VTHS	Validates if patient exists in database
3	Medical Student	Enters patient’s details into VTHS
4	VTHS	Confirms successful add of patient

**Add New Patient**  
**Step 1 of 3**  
 Complete the fields below and then click 'Next' to move on to step 2

First Name:

Surname:

Middle Name(s):

Date of Birth: 1 / 1 / 2008

Gender:  Male  Female

Address:

Country:

Continent:

Region of Continent:

Diagram 10: GUI Sketch of Register Patient Use Case

## Alternative Scenarios for Register Patient

<b>Step</b>	<b>Condition for Alternative</b>	<b>Description of Activity</b>
2	Patient already exists	Cancel use case

**2.5.3. Description of Use Case Retrieve Patient's old files**

## Details of Use Case Retrieve Patient's old files

<b>Name:</b>	Retrieve Patient's old files
<b>Primary Actor:</b>	Medical Student
<b>Goal (of the User):</b>	Patient data is displayed to user.
<b>Precondition:</b>	Patient is registered
<b>Postcondition (successful outcome):</b>	The correct patient details are displayed
<b>System (implementing it):</b>	VTHS system
<b>Participating Actor:</b>	None

## Main Scenario for Use Case Retrieve Patient's old files

<b>Step</b>	<b>Actor/System</b>	<b>Description of Activity</b>
1	Medical Student	Retrieve NHS number from patient
2	VTHS	Search for patient details by unique identifier
3	VTHS	Display patient info



Diagram 11: GUI Sketch of Retrieve Patient old files Use Case

### Alternative Scenarios for Retrieve Patient’s old files

Step	Condition for Alternative	Description of Activity
3	Patient does not exist	Cancel use case.

### 2.5.4. Description of Use Case Medical History Management

#### Details of Use Case Medical History Management

<b>Name:</b>	Medical History Management
<b>Primary Actor:</b>	Medical Student
<b>Goal (of the User):</b>	Generate Medical History for a Patient’s case
<b>Precondition:</b>	Patient has a Medical case
<b>Postcondition (successful outcome):</b>	Medical History is generated for the patient’s case
<b>System (implementing it):</b>	VTHS system
<b>Participating Actor:</b>	Clinical Teacher



### Main Scenario for Use Case Medical History Management

Step	Actor/System	Description of Activity
1	Clinical Teacher	Selects a patient
2	Medical Student	Enters initial clinical parameter into the VTHS system
3	Medical Student	Types his/her rationale in a textbox in the VTHS system
3	VTHS	System outputs list of possible causes
4	Medical Student	Enters clinical parameter into the VTHS system
5	Medical Student	Types his/her rationale in a textbox in the VTHS system
6	VTHS	System outputs list of possible causes
Steps 4 -6 are repeated until no more relevant parameters need to be entered		
6	VTHS	Generates a case report
7	Clinical Teacher	Marks case report

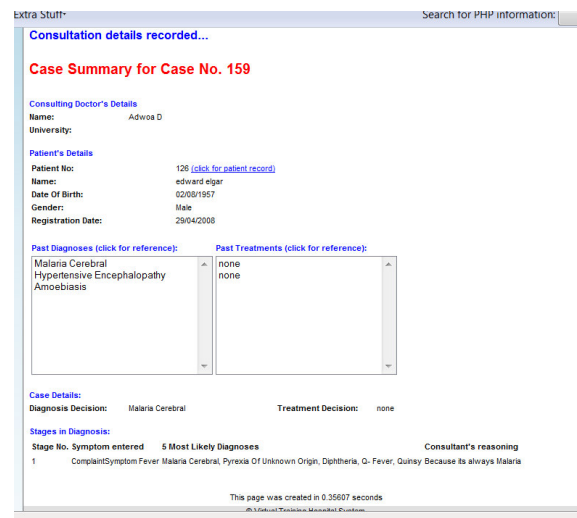


Diagram 12: GUI Sketch of Medical History Management Use Case

### Alternative Scenarios for Medical History Management

<b>Step</b>	<b>Condition for Alternative</b>	<b>Description of Activity</b>
1	No Patients available	Cancel use case.

## 2.6. Non-Functional Requirements

In this section we will discuss properties of the functions discussed in section 2.4.

### 2.6.1. Look and Feel

There are various non-functional requirements, which go beyond the scope of this project and therefore the current system's overall look and feel will not be dramatically changed. The client requested a more graphical user-interface, such that the user will select a patient to examine from a virtual waiting room. This will provide the student with visualization of patient demographics. This requirement will have to be modified in future developments of this system due to time constraints.

### 2.6.2. Usability and Humanity

**Usability Requirement:** The product shall be easy for medical students to use.

**Fit criteria:** A test panel of medical school students shall be able successfully to complete a list of tasks within a specified time

**Usability Requirement:** The product shall be easy for medical school staff to use

**Fit criteria:** A test panel of medical school staff shall be able to successfully complete a list of tasks within a specified time

**Usability Requirement:** The product should make users enjoy using it.

**Fit criteria:** An anonymous survey shall show that 75% of the intended users (medical students and medical school staff) are regularly using the product after a two-week familiarization period.

**Learning Requirement:** The product shall be used by medical students who will receive no training before using it.

**Fit criteria:** 85% of a test panel of medical students shall successfully complete the diagnosis process within 30 minutes.

**Accessibility Requirements:** The product shall conform to the British Disability Discrimination Act (DDA); hence ensure that the website is accessible to disabled users. [DDA]

### 2.6.3. Performance

**Speed requirement:** The response time in accumulated likely diagnoses shall be fast enough to avoid interrupting the user's flow of thought.

**Fit criteria:** No response shall take longer than 2.5 seconds.

**Precision or Accuracy Requirement:** All diagnoses shall be 99% accurate.

**Fit criteria:** 100 real medical cases should result in the correct diagnosis.

**Availability Requirement:** The product shall be available 24 hours a day, 7 days a week.

**Robustness/Fault Tolerance Requirement:** The product shall be able to continue to operate in local mode whenever it loses connection via the internet.

**Capacity Requirement:** The product shall be able to cater to all Leicester Medical School students simultaneously

**Fit Criteria:** Test this by quantifying the number of students and getting volunteers to simultaneously run the site at the same time.

#### ***2.6.4. Operational***

**Requirement for Expected Physical Environment:** The product shall be used by a medical student in training at a hospital or clinic.

**Requirement for Expected Physical Environment:** The product shall be used by a medical student studying at home, library or campus.

**Requirement for interfacing with adjacent systems:** The product shall work on the last four releases of the five most popular browsers.

#### ***2.6.5. Maintainability and Support***

**Maintenance Requirement:** Database data should easily be updated by the administrator without affecting site availability.

#### ***2.6.6. Security***

**Access Requirement:** The website shall be password protected.

**Fit Criteria:** User roles (Actors) shall have access to allowed sections as described in section 2.4.2.

**Privacy Requirements:** The product shall inform the user to gather patients' consents to use their medical case on the system

#### ***2.6.7. Cultural and Political***

**Cultural Requirement:** The product shall not be offensive to religious and ethnic groups.

**Political Requirement:** The product shall make all functionality available to Head of Medical Education at the University of Leicester

### ***2.6.8. Legal***

**Compliance Requirement:** The implementation for the data entry of patient information shall comply with the Data Protection Act.

***Fit Criteria:*** Receive a Lawyer’s opinion that the site does not break any laws.

### 3. SYSTEM DESIGN

In this section the system’s design will be analysed.

#### 3.1. Software Architecture

This section justifies architectural style choices and expresses the logical architecture with the use of a component diagram.

Due to the fact that this system is a project that is re-engineered the decisions of technologies used for the web based implementation were decided by the previous developers. Hence, the system is based on PHP and MySQL technologies. The present version has the addition of Java.

The architecture of the system has expanded to include a Neural Network, as shown in Diagram 1. The Neural Network will initially use the database for training and the PHP business logic will use the trained network to diagnose. A deployment diagram of the system is shown in diagram 2.

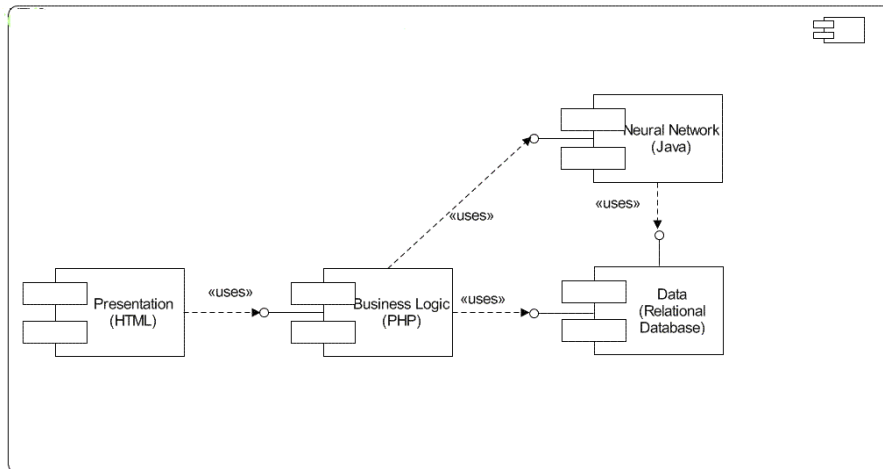


Diagram 13: Component Diagram of New Architecture of VTHS system

The multi-architecture of the VTHS system is similar to the J2EE multi-architecture shown in the below diagram. In the client tier the VTHS website is represented as a web application and the neural network is represented as a standalone Java client. The web-tier runs a web server to handle client requests and invokes JavaServer Pages (JSPs). [BL] The business tier provides a bridge to the database. The JDBC driver is used by the Java Neural Network to connect to the relational database management system (RDBMS) in the Enterprise information systems (EIS) tier.

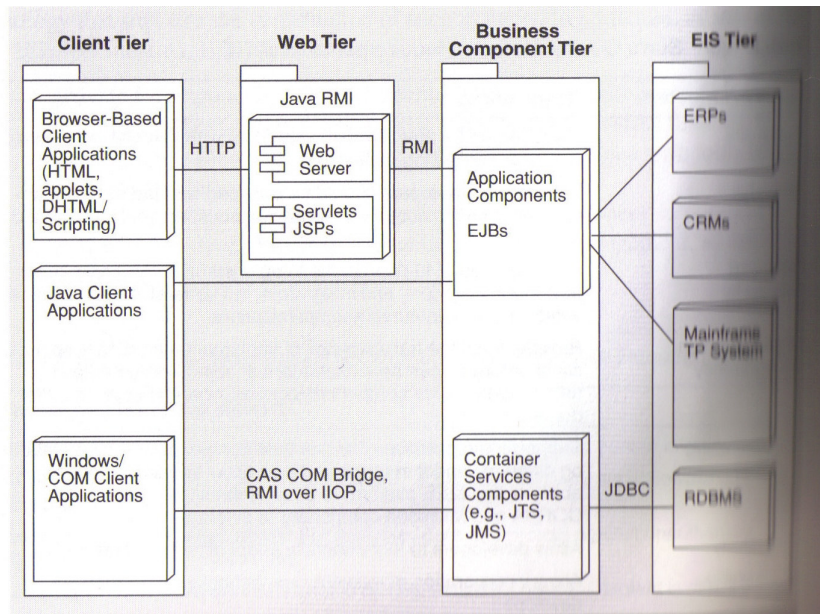


Diagram 14: Deployment view of the J2EE multi-architecture [BL]

### 3.1.1 Neural Networks

#### General Architecture

The architecture for artificial neural networks consists of multiple neurons organized in multiple layers. Each neuron in a preceding layer is connected to all neurons on the next layer.[NR] The neurons of the neural network in the VTHS system are organised into an input layer, a hidden layer and an output layer, as shown in the diagram below. The neurons located on the input layer are based on absence (0) or presence (1) of a particular symptom, given a particular diagnosis case. The number of input neurons will be the total number of symptoms found in the database tables. The diagnosis index corresponds to a particular diagnosis in the database tables. The expansion of the database in terms of symptoms (columns) or diagnosis (rows) will require minor changes in the setting up of the neural network and retraining of the neural network.

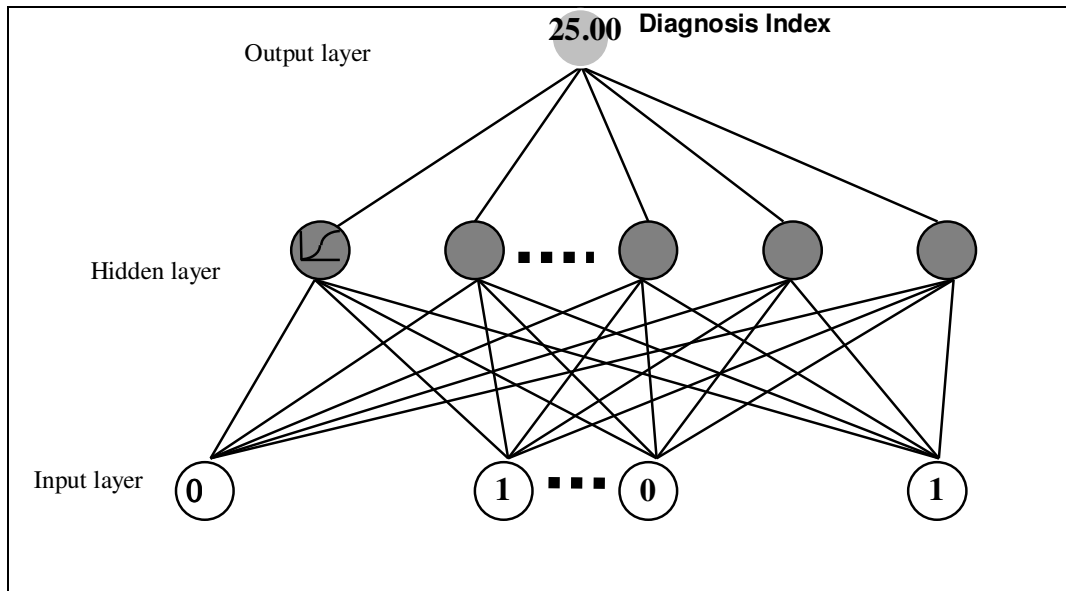


Diagram 15: Architecture for artificial neural networks

### Learning in multi-layer networks

The connectivity of the perceptron's are connected such that the input values are recoded producing non-adaptive 'hand-coded' features called hidden neurons. [HG] Whereas the output unit is a binary threshold neuron, which is independently learnt.

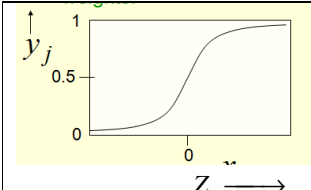
There are various neural network algorithms, which can be used for learning in multi-layer networks. The basic online - backpropagation algorithm was chosen to be used in the VTHS ANN because it is an efficient way of adapting the weights of the network.

The main idea behind the backpropagation algorithm is to compute how fast the error changes as we change activity, by using error derivatives with respect to hidden activities. Below is a detailed structure of the design set up for VTHS neural network and its corresponding backpropagation algorithm components.



**Legend**  
 $x$  = activities of input units  
 $y$  = activities of output units  
 $z$  = the summed input to an output unit  
 $w$  = weight  
 $b$  = bias  
 $E$  = error

We will use the logistic function, as shown below to generate smooth derivatives with the non-linear neurons.



$$z_j = b_j + \sum_i x_i w_{ij}$$

$$y_j = \frac{1}{1 + e^{-z_j}}$$

$\therefore$

$$\frac{\partial z_j}{\partial w_{ij}} = x_i \quad \frac{\partial z_j}{\partial y_i} = w_{ij}$$

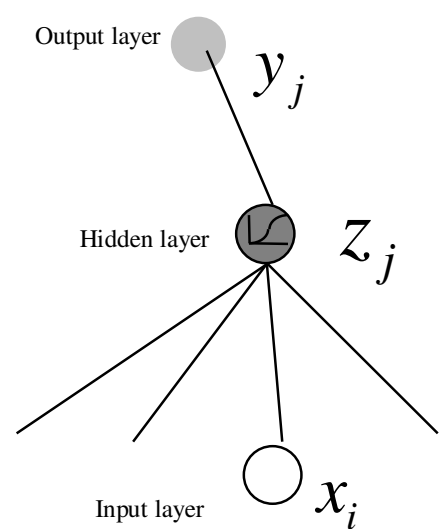
$$\frac{dy_j}{dz_j} = y_j (1 - y_j)$$

Hence the Error derivatives are as follows:

$$\frac{\partial E}{\partial z_j} = \frac{dy_j}{dz_j} \frac{\partial E}{\partial y_j} = y_j (1 - y_j) \frac{\partial E}{\partial y_j}$$

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial z_j}{\partial w_{ij}} \frac{\partial E}{\partial z_j} = x_i \frac{\partial E}{\partial z_j}$$

$$\frac{\partial E}{\partial x_i} = \sum_j \frac{dz_j}{dx_i} \frac{\partial E}{\partial z_j} = \sum_j w_{ij} \frac{\partial E}{\partial z_j}$$



Formula 2: Use of Backpropagation in VTHS Neural Network [HG]

### Connection of Neural Network to Database

The following diagram represents the moderate-level fidelity simulation with static input and output data files. The static input and output files are equivalent to the database tables used in the VTHS System.

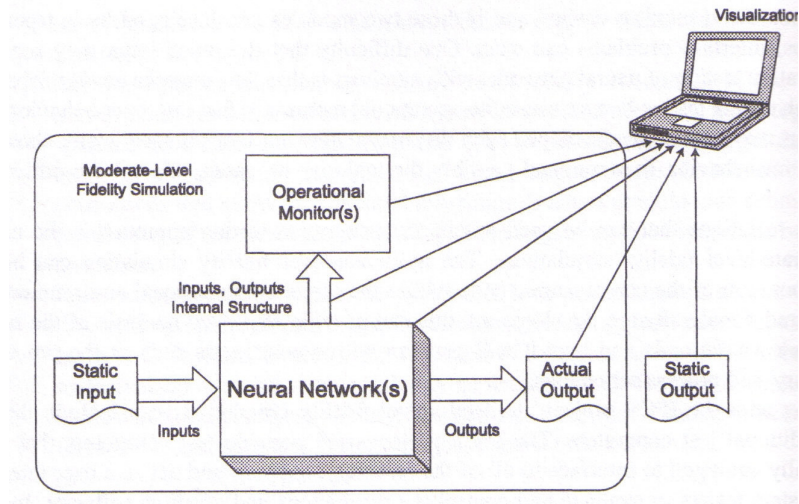


Diagram 16: Neural Network connection to data files [PL]

### 3.1.2 Web database Application

#### Three-Tier Architecture

The VTHS web page is based on a three-tier architecture, which is composed of the following tiers: Database tier, Middle tier and Client tier, as shown in the diagram below.

The database tier consists of the MySQL database management system (DBMS). The middle tier contains the application logic and communicates data between the other tiers. [WL] The top tier is the client tier, which contains web browser technologies. The web technologies' section below will go into more detail.

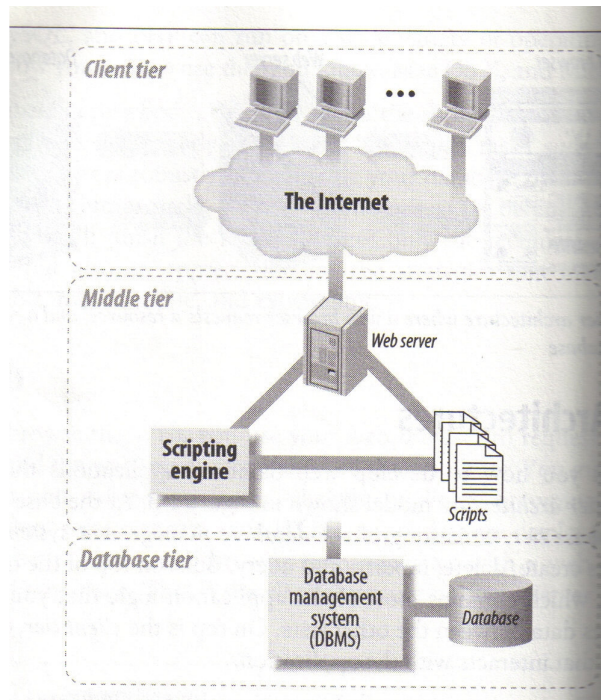


Diagram 17: Three –tier architecture model of the web database application [WL]

#### Web Technologies

The web technologies that were used in the VTHS system are as follows:

- PHP (PHP: Hypertext Pre-processor)
- HTML (Hyper Text Markup Language)
- MySQL
- SQL (Structured Query Language)
- JavaScript
- Ajax (Asynchronous JavaScript And XML)

The website is based on the general-purpose scripting language of PHP and runs on a web server that takes PHP code as input and outputs HTML web pages. The website connected a MySQL relational database management system (RDBMS) and used SQL to define, manipulate and query the database. Additional scripting features of the site were done in Javascript, which is located on the client-side of the client-server architecture. Ajax is used for creating interactive web applications, which can retrieve data from a server asynchronously without interfering with the display or behaviour of the existing web page. [AP]

The following deployment diagram depicts the run-time static view of the Client-Server web architecture for the VTHS system.

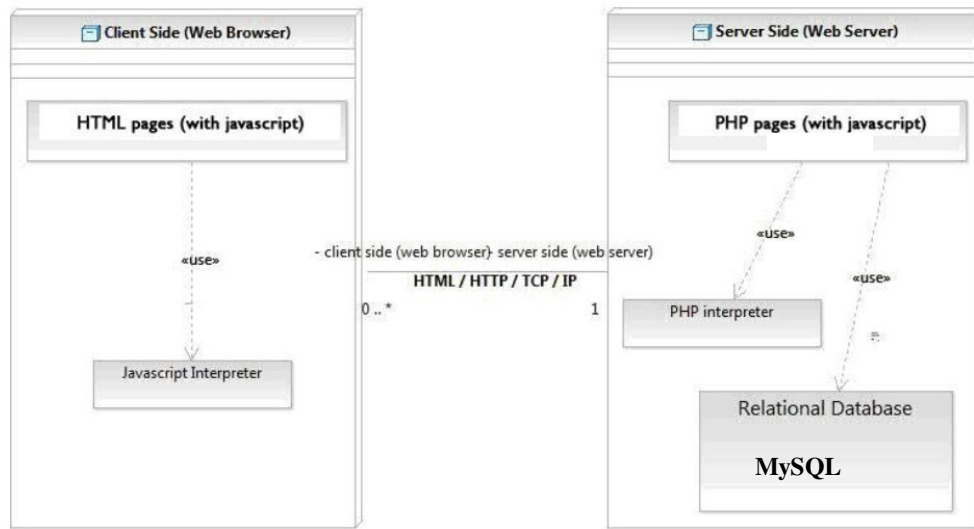


Diagram 18: Deployment Diagram of Web portion of VTHS System [BCKPP2]

### 3.2. Analysis Sequence Diagrams

This section discusses the interaction between the components shown in the deployment and component diagrams of section 3.1. The architecture is composed of the following four components:

- HTML
- PHP
- MySQL Database
- Java Neural Network

The following sequence diagram is used to emphasize the interaction of the system’s components for a particular scenario. The use case scenario displays the sequence of events required for a user to retrain the VTHS neural network

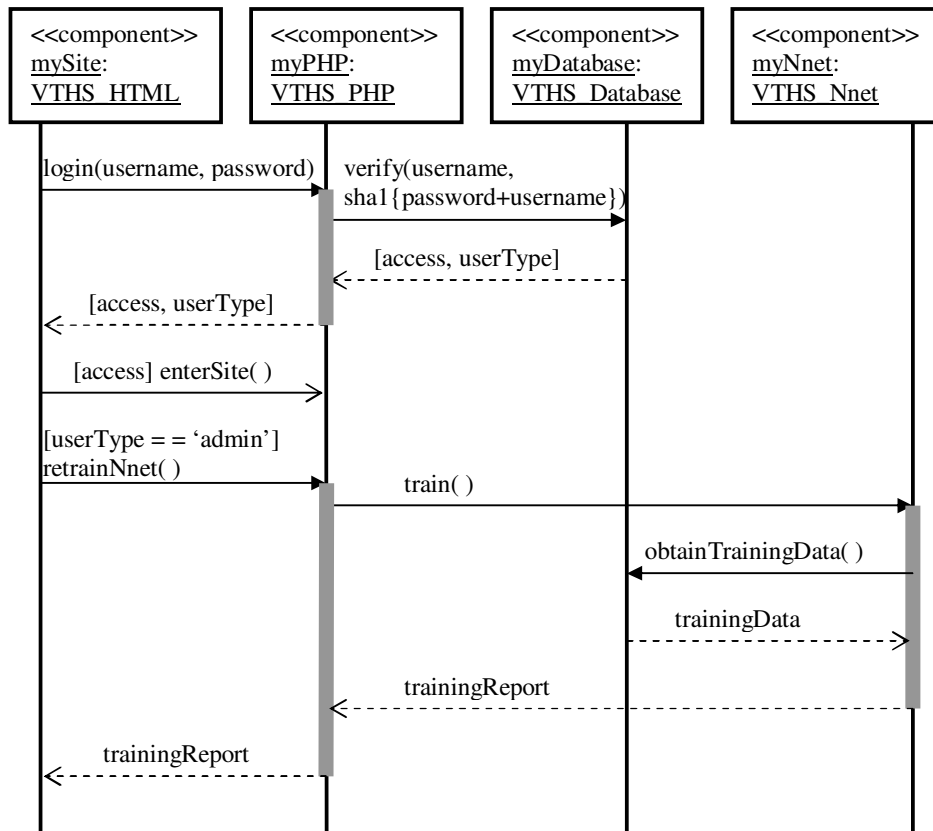


Diagram 19: Sequence Diagram on Component Instances

Note: sha1 {password+username} means encryption with the password concatenated with the username. This Secure Hash Algorithm (SHA1), produces a 160bit digest message.

### 3.3. Design Class Diagram

Many of the design decisions were influenced by Jeff Heaton’s “Using JOONE for Artificial Intelligence Programming” tutorial. [HJ]

Below is a class diagram of the neural networks portion of the system. It provides a static view of the implementation. This diagram conforms to the software architecture, sequence diagram and problem domain, as discussed in prior sections.

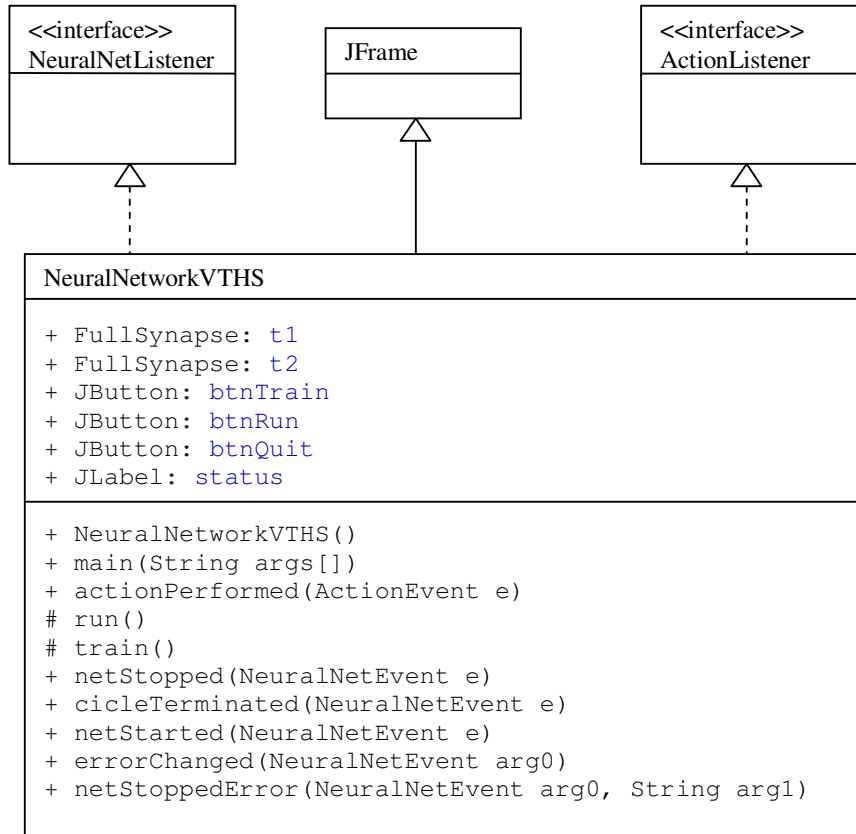


Diagram 20: Class Diagram of NeuralNetworkVTHS

### 3.4 GUI Design

The following diagram is the GUI Design for the NeuralNetworkVTHS class presented in the above section. There are three buttons used for running, training and quitting operations. The status label will output information proclaiming to the current state of the system.

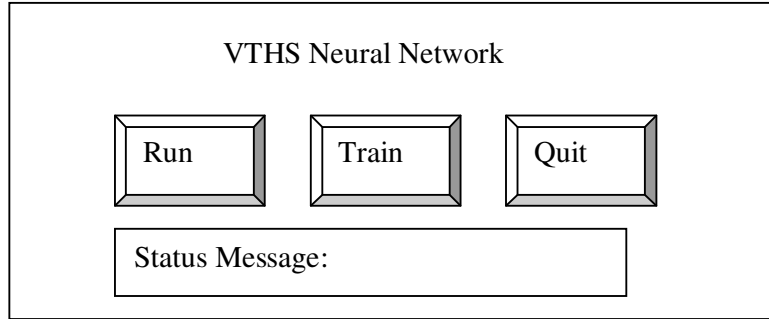


Diagram 21: GUI Design of Neural Network

## 4. IMPLEMENTATION

In this sub-section we discuss the two main components of technologies used in the VTHS project.

### 4.1. Neural Network API

In this section we will discuss the JOONE API and the implementation in VTHS system.

#### 4.1.1. JOONE (Java Object Oriented Neural Engine)

In order to incorporate Artificial Neural Networks into the system the JOONE Neural Network framework was selected. This engines API provides the Java classes to create, train and test Neural Networks.

The Core Engine API is composed of various features. The key features used are described below:

- **Neural net architecture creation components:** Used to create a Feed Forward neural network (FFNN)
- **Supervised training algorithms:** The engine provides three options for back propagation of either On-line, Batch or Resilient.
- **Input/Output (I/O) Components:** This provides the option to read input data from sources such as databases or text files.

The JOONE API is composed of various packages. The most important packages are described below:

- **org.JOONE.engine:** The core engine is used to build the neural nets.
- **org.JOONE.engine.learning:** This package is used for training the neural net
- **org.JOONE.io:** The I/O package is used for read and writing patterns from/to external sources.
- **org.JOONE.net:** The net package is used for managing a neural net.

#### 4.1.2. Implementation of Java Neural Network

In this section we will discuss the implementation of the Java Neural Network class, which corresponds to the class diagram in section 3.3.

#### Basic Structure

The following code outlines the fact that NeuralNetworkVTHS is in generalization with JFrame and realization relationship with the interfaces of



NeuralNetListener and ActionListener. It also presents the various java libraries used in this system.

```

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

import org.joone.engine.*;
import org.joone.engine.learning.*;
import org.joone.io.*;

/**
 * VTHS with JOONE
 * @author Adwoa Donyina
 * @version 1.0
 */

/**
 * Neural Networks GUI
 */
public class NeuralNetworkVTHS extends JFrame implements
ActionListener,NeuralNetListener {

```

## Neural Networks GUI

In order to create the GUI described in section 3.4, the following code is necessary.

```

RJava XOR_using_NeuralNet. NeuralNetworkVTHS.java mySample.java JDBCInputSyr
public NeuralNetworkVTHS()
{
    setTitle("VTHS Solution");

    Container content = getContentPane();

    GridBagLayout gridbag = new GridBagLayout();
    GridBagConstraints c = new GridBagConstraints();

    content.setLayout(gridbag);

    c.fill = GridBagConstraints.NONE;
    c.weightx = 1.0;

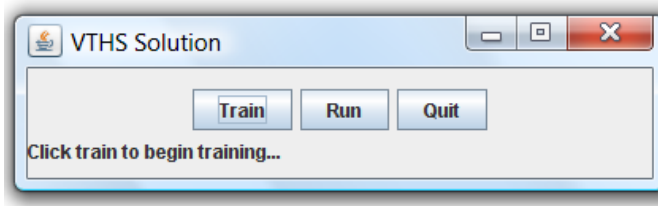
    // the button panel
    JPanel buttonPanel = new JPanel(new FlowLayout());
    buttonPanel.add(btnTrain = new JButton("Train"));
    buttonPanel.add(btnRun = new JButton("Run"));
    buttonPanel.add(btnQuit = new JButton("Quit"));
    btnTrain.addActionListener(this);
    btnRun.addActionListener(this);
    btnQuit.addActionListener(this);

    // Add the button panel
    c.gridwidth = GridBagConstraints.REMAINDER; //end row
    c.anchor = GridBagConstraints.CENTER;
    content.add(buttonPanel,c);

    // Training input label
    c.gridwidth = GridBagConstraints.REMAINDER; //end row
    c.anchor = GridBagConstraints.NORTHWEST;
    content.add(
        status = new JLabel("Click train to begin training..."),c);

```

The above code produces the following output at runtime:



### ‘Train’ operation

When a user selects the ‘Train’ button on the user interface, the `train()` method is called. When `train()` is invoked the Neural Network is set up.

This is shown in the following code:

```

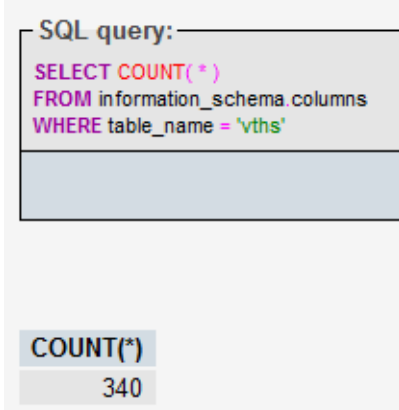
OR.java  XOR_using_NeuralNet.  NeuralNetworkVTHS.java
protected void train()
{
    /*
     * Setting up the neural network
     * Create 3 layers: input, hidden, and output
     * and Set corresponding layer names
     */
    input = new SigmoidLayer();
    hidden = new SigmoidLayer();
    output = new SigmoidLayer();
    input.setLayerName("input");
    hidden.setLayerName("hidden");
    output.setLayerName("output");
}

```

The above code shows how each layer was created as a `SigmoidLayer` object and named input, hidden and output accordingly.

The input layer retrieves data directly from the `nnet1_vths` database. The total number of nodes on the input layer is number of columns from the 5 tables, which will be described in more detail in section discussed in section 4.1.3. For simplicity, these 5 tables were merged into 1 table called ‘vths’, using the sql ‘join’ command. The calculation of total columns was accumulated by query the database table and subtracting the unused index and name columns.

An example query is as follows:



The above query result states that the vths table contains 340 columns.  
 $\therefore$  input nodes =  $(numberColsInTable) - (nonInputCols) = 340 - 2 = 338$   
Hence we require 339 hidden nodes because the chosen neural network structure requires hidden nodes =  $numInputNodes + 1$  . The number of output nodes required is 1 according to the structure of the discussed Neural Network is shown in section 3.1.1 (subsection- General structure); this is represented as the index column in the vths database table.

The following code sets up the networks dimensions:

```

/*
 * input dimension corresponds to columns in input
 *
 * Current dimensions
 *      *
 *      * *
 *      *
 *      *
 */
// sets layer dimensions
input.setRows(338);
hidden.setRows(339);
output.setRows(1);

```

Each layer on the network is connected with a synapse. The following shows the creation 'FullSynapse' object and the construction of the appropriate connections. The synapse from input layer to hidden layer is abbreviated by 'IH', similarly the synapse from hidden layer to output layer is referred by 'HO'. As shown in the following code:

```

// Now create the two Synapses
// input -> hidden conn.
FullSynapse synapse_IH = new FullSynapse();
// hidden -> output conn.
FullSynapse synapse_HO = new FullSynapse();

synapse_IH.setName("IH");
synapse_HO.setName("HO");
t1=synapse_IH;
t2=synapse_HO;

// Connect the input layer with the hidden layer
input.addOutputSynapse(synapse_IH);
hidden.addInputSynapse(synapse_IH);

// Connect the hidden layer with the output layer
hidden.addOutputSynapse(synapse_HO);
output.addInputSynapse(synapse_HO);

```

The chosen algorithm for teaching the neural network is by supervised learning, specifically based on the basic On-Line backpropagation learning method.

Information about this design choice is found in section 3.1.1 (subsection- Learning in multi-layer networks). Whereas below shows how to implement it in Java.

The On-Line backpropagation learning method is the default algorithm set by the JOONE library. It requires two parameters of learning rate and momentum to be appropriately set. This is done in java by creating a ‘Monitor’ object to regulate the neural network, then set the parameters on the instance variable of the ‘Monitor’ and assign the monitor to the corresponding neural network layers. [HJ] This process is shown in the following code snippet.

```

    * Based on Backpropagation Equation, can change parameters
    */
    // Create the Monitor object and set the learning parameters
    monitor = new Monitor();
    monitor.setLearningRate(0.8);
    monitor.setMomentum(0.3);

    // Pass the Monitor to all components
    input.setMonitor(monitor);
    hidden.setMonitor(monitor);
    output.setMonitor(monitor);

```

At this point the ‘JDBCInputSynapse’ object is set to the data in the VTHS database, this will be described in greater detail in section 4.1.3. The ‘JDBCInputSynapse’ object is added to the input ‘SigmoidLayer’, as follows:

```

// This is the database that contains the input data
input.addInputSynapse(inputStream1);

```

Next a ‘TeachingSynapse’ object is created and is used to train the neural network, while the ‘Monitor’ object runs the neural network through various training iterations. [HJ], as follows:

```

trainer = new TeachingSynapse();
trainer.setMonitor(monitor);

```

The Neural network requires desired output to be used to compare to the actual results during the training process, because the required error calculation is the difference between the actual output of the neural network and the desired output.[HJ] Hence the desired responses are queried from the database table and added to the ‘TeachingSynapse’ object, and then the ‘TeachingSynapse’ object is connected to the output layer as follows:

```

// Setting of the database containing the desired responses,
// provided by a JDBCInputSynapse
samples1 = new JDBCInputSynapse();
samples1.setdriverName("com.mysql.jdbc.Driver");
samples1.setdbURL("jdbc:mysql://localhost:3306/nnet1_vths?user=root");
samples1.setSQLQuery("select index from vths;");

// The output values are on the 1st column of this table
samples1.setFirstCol(1);
samples1.setLastCol(1);
trainer.setDesired(samples1);

// Connects the Teacher to the last layer of the net
output.addOutputSynapse(trainer);

```

In order to start the training process, the ‘start()’ method needs to be called on all layers and the ‘Monitor’ object training parameters must be set, such as the training patterns which equals to the number of training cases, this is calculated as follows  
`trainingPatterns= numOfVthsTableRows` and shown below:

```
// All the layers must be activated invoking their method
// start; the layers are implemented as Runnable objects, then
// they are allocated on separated threads. The threads will
// stop after training and will need to be restarted later.
input.start();
hidden.start();
output.start();

// number of rows (patterns) contained in the input database
monitor.setTrainingPatterns(1410);
// How many times the net must be trained on the input
// patterns
monitor.setTotCicles(20000);
// The net must be trained
monitor.setLearning(true);
// The net starts the training job
monitor.Go();
}
```

The training process at run-time appears as follows:

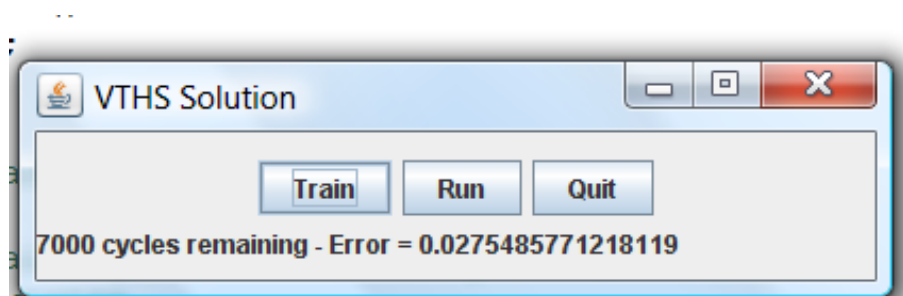


Diagram 22: Resulting Prototype at run-time

### ‘Run’ Operation

Similar to the ‘Train’ operation discussed above except it uses the trained neural network; hence no learning is involved. Once the run is complete the output is printed in the database.

#### 4.1.3. Use of NN API on current knowledge base

The current knowledge base is composed of one epidemiology and four pathological diagnosis database tables. Diagram 23 shows a snippet of one of the five database tables used in the current system.

		DiseaseDiagnosisNameCompl	ComplaintSymptom Fever	ComplaintSymptom NightSweats
<input type="checkbox"/>		Abdominal Distension	0	0
<input type="checkbox"/>		Abdominal Noise	0	0
<input type="checkbox"/>		Abdominal Pain	0	0
<input type="checkbox"/>		Abdominal Swelling	0	0
<input type="checkbox"/>		ABO Incompatibility	0	0
<input type="checkbox"/>		Abortion	0	0
<input type="checkbox"/>		Abortion Complete	0	0
<input type="checkbox"/>		Abortion Criminal	45	0
<input type="checkbox"/>		Abortion Habitual	0	0
<input type="checkbox"/>		Abortion Incomplete	0	0
<input type="checkbox"/>		Abortion Inevitable	0	0
<input type="checkbox"/>		Abortion Missed	0	0
<input type="checkbox"/>		Abortion Septic	45	0
<input type="checkbox"/>		Abortion Spontaneous	0	0

Diagram 23: Snippet of 1 of 5 database tables used in the original system

In order to satisfy the input requirements of JOONE's input mechanism, a format converter and selection mechanism is required. The only permitted input data types are integer, double, and float. Hence, column 1 in the database table type should be indexed by corresponding numbers instead. This can be done by inserting an addition column in the table that corresponded with the name, then using the provided 'AdvancedColumnSelector' class to ensure not to use diagnosis name column.

The JOONE API provides a JDBCInputSynapse class, which provides a connection to the MySQL database. The required Driver protocol is as follows [MP]:

```
{jdbc:mysql://[hostname][, failoverhost...][:port]/[dbname][?param1=value1][& param2=value2]...} MySQL Protocol
```

```
Example {jdbc:mysql://localhost/test?user=blah&password=blah}
```

This is coded in the Java class as follows:

```
InputStream1 = new JDBCInputSynapse();
InputStream1.setDriverName("com.mysql.jdbc.Driver");
InputStream1.setDBURL("jdbc:mysql://localhost:3306/nnet1_vths?user=root");
InputStream1.setSQLQuery("select * from vths;");
```

#### 4.1.4. Preliminary Input Data Setup

The database tables were configured to conform to the input requirement set by the JOONE API. The manipulation of the tables occurred in a new database called nnet1\_vths, by exporting five tables from the original knowledge base, as shown in the below diagram.

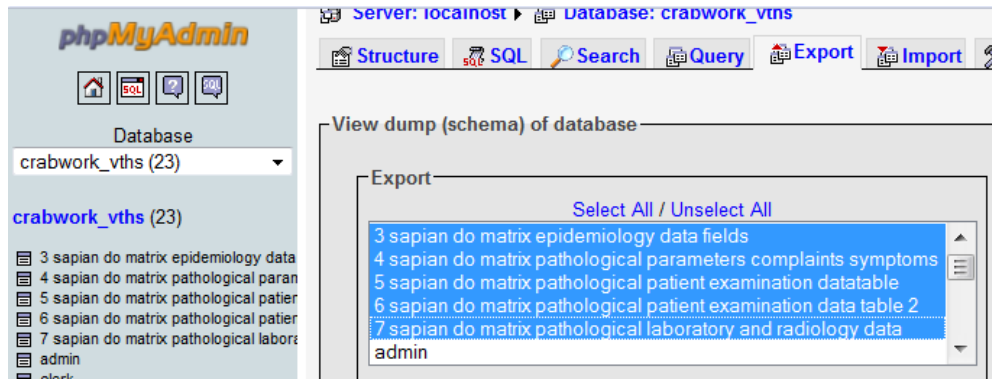


Diagram 24: Exporting database tables from old database

The exported data was saved in a text file in order to manually make various query changes, such as shortening removing spaces from the table names as follows:

- '3 sapian do matrix epidemiology data fields' to epi
- `4 sapian do matrix pathological parameters complaints symptoms` to path1
- `5 sapian do matrix pathological patient examination datatable` to path2
- `6 sapian do matrix pathological patient examination data table 2` to path3
- `7 sapian do matrix pathological laboratory and radiology data` to path4

Secondly, a majority of the datatypes and corresponding default values of the table columns were changed from varchar(50) to double in order to conform to the number input requirement of the JOONE neural network library.

Old Table Structure
<pre>CREATE TABLE `3 sapian do matrix epidemiology data fields` (   `DiseaseDiagnosisNameEpidem` varchar(50) NOT NULL default '1',   `AttributeAgeBirthDay` varchar(50) default NULL,   `AttributeAgePremature` varchar(50) default NULL,   ...</pre>
New Table Structure
<pre>CREATE TABLE epi (   `DiseaseDiagnosisNameEpidem` varchar(50) NOT NULL default '1',   `AttributeAgeBirthDay` double default 0,   `AttributeAgePremature` double default 0,   ...</pre>

Once the manual changes were completed the tables and corresponding data were imported into the new database. Unfortunately, this process ran into issues due the enormous size of data. Importation was halted because the query reached the maximum execution time of 300 seconds. This issue was resolved by modifying the maximum execution time in the WampServer configuration file, as follows:

\$cfg['ExecTimeLimit'] = 0; , which results in no limit. The importation of the whole database whole took one hour and twenty minutes which was extremely time-consuming. The resulting database is shown below.

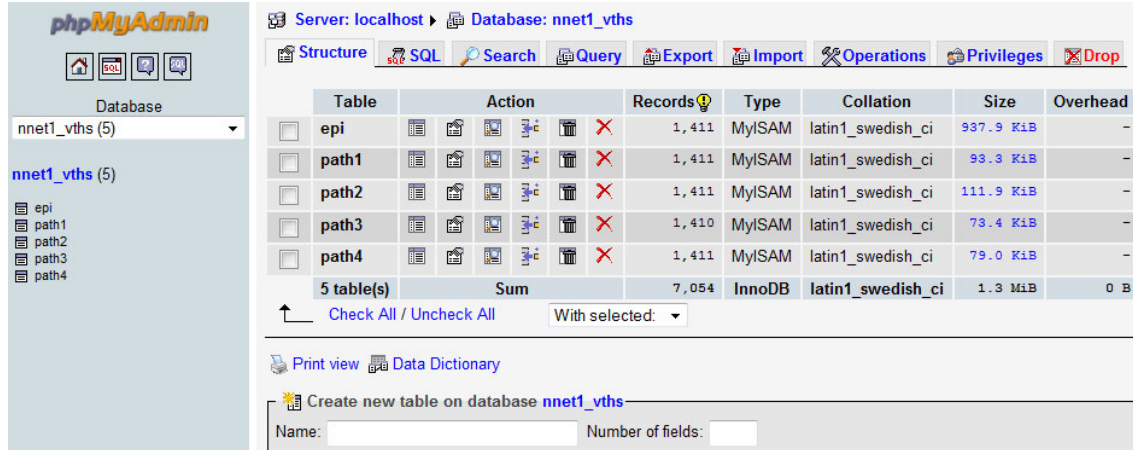
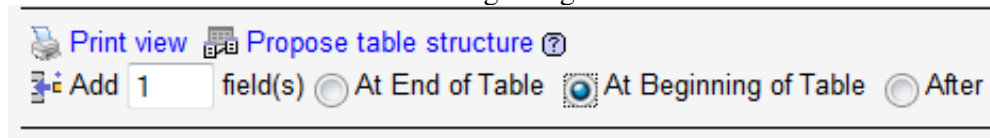


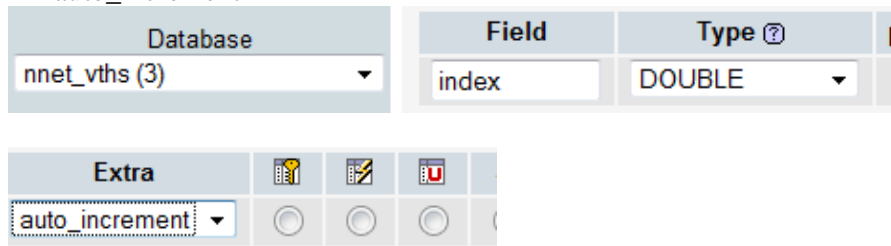
Diagram 25: New nnet1\_vths database

As discussed earlier in the above ‘Use of NN API on current knowledge base’ section indexes were generated to match corresponding diagnosis names. This was automatically generated using the phpMyAdmin in the WampServer, as shown below.

1. Insert the new column at the beginning of the table:



2. Name the column index, set the type to double and select in extra auto\_increment



3. Change table values to either 0 or 1 to represent absence or presence of a particular symptom. If the data value was between 0-0.49, it was replaced with a 0 otherwise it was replaced with a 1.

Found by querying the table as follows:

```
SELECT *
FROM `path1`
WHERE `ComplaintSymptom Fever` >1 OR `ComplaintSymptom
NightSweats` >1 OR `ComplaintSymptomOverweightWeightGain` >1 OR
`ComplaintNervousSystemHeadache` >1 OR
`ComplaintNervousSystemPhotophobia` >1 OR [...]
```



**Before**

Field	Type	Function	Null	Value
index	double			67
DiseaseDiagnosisNameEpidem	varchar(50)			Alzheimers Disease
AttributeAgeBirthDay	double		<input type="checkbox"/>	0
AttributeAgePremature	double		<input type="checkbox"/>	0
AttributeAgeNewBorn	double		<input type="checkbox"/>	0
AttributeAgeNeonate	double		<input type="checkbox"/>	0
AttributeAgeInfant	double		<input type="checkbox"/>	0
AttributeAge1-5YearChild	double		<input type="checkbox"/>	0
AttributeAge6-13YearChild	double		<input type="checkbox"/>	0
AttributeAgeChildBearingAge	double		<input type="checkbox"/>	0.2
AttributeAgeMiddleAge	double		<input type="checkbox"/>	1
AttributeAgeElderly	double		<input type="checkbox"/>	2

**After**

Field	Type	Function	Null	Value
index	double			67
DiseaseDiagnosisNameEpidem	varchar(50)			Alzheimers Disease
AttributeAgeBirthDay	double		<input type="checkbox"/>	0
AttributeAgePremature	double		<input type="checkbox"/>	0
AttributeAgeNewBorn	double		<input type="checkbox"/>	0
AttributeAgeNeonate	double		<input type="checkbox"/>	0
AttributeAgeInfant	double		<input type="checkbox"/>	0
AttributeAge1-5YearChild	double		<input type="checkbox"/>	0
AttributeAge6-13YearChild	double		<input type="checkbox"/>	0
AttributeAgeChildBearingAge	double		<input type="checkbox"/>	0
AttributeAgeMiddleAge	double		<input type="checkbox"/>	1
AttributeAgeElderly	double		<input type="checkbox"/>	1

After the five database tables were individually modified, they were merged into one table, which contained all input and expected output data. The new 'vths' table was created by executing a 'join' query and directly inserting the results into the table. The following diagrams show the steps taken:

Table	Action	Records	Type	Collation	Size	Overhea
epi		1,411	MyISAM	latin1_swedish_ci	937.9 KiB	
path1		1,411	MyISAM	latin1_swedish_ci	93.3 KiB	
path2		1,411	MyISAM	latin1_swedish_ci	111.9 KiB	
path3		1,410	MyISAM	latin1_swedish_ci	73.4 KiB	
path4		1,411	MyISAM	latin1_swedish_ci	79.0 KiB	
5 table(s)	Sum	7,054	InnoDB	latin1_swedish_ci	1.3 MiB	0

Diagram 26: initial nnet1\_vths database structure

```

SQL query:
CREATE TABLE vths AS (
SELECT *
FROM epi, path1, path2, path3, path4
WHERE epi.index = path1.index1
AND path1.index1 = path2.index2
AND path2.index2 = path3.index3
AND path3.index3 = path4.index4
)
    
```

Diagram 27: SQL Query- creates new vths table

Resulting table is 4.5 MiB size and a subsection is shown below.

index	DiseaseDiagnosisNameEpidem	AttributeAgeBirthDay	AttributeAgePremature	AttributeAgeNewBorn	AttributeAgeNeonate
1	Abdominal Distension	1	1	1	1
2	Abdominal Noise	1	1	1	1
3	Abdominal Pain	1	1	1	1
4	Abdominal Swelling	1	1	1	1
5	ABO Incompatibility	1	1	1	1
6	Abortion	0	0	0	0
7	Abortion Complete	0	0	0	0
8	Abortion Criminal	0	0	0	0
9	Abortion Habitual	0	0	0	0

Diagram 28: Subsection of resulting vths table

Due to strict format restrictions of `org.joone.io.JDBCInputSynapse` the non-numeric column of ``DiseaseDiagnosisNameEpidem`` was removed from the `vths` training table and then inserted into a new diagnosis table. As shown below,

```

SQL query:
CREATE TABLE diagnosis AS (
SELECT 'index', 'DiseaseDiagnosisNameEpidem'
FROM `vths`
)
    
```

Diagram 29 : SQL query- creation of diagnosis name table

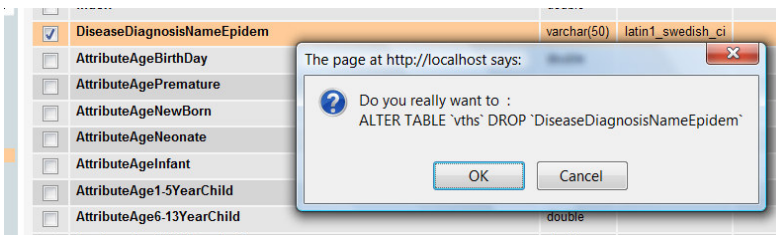


Diagram 30: Process of removing `DiseaseDiagnosisNameEpidem` column from vths table

At this point the input data was ready for training.

## 4.2. Web Application

In the section we will discuss implementation issues related to the web components of the VTHS system.

### 4.2.1. Resolved Re-engineering Issues

Due to the fact that the web components were re-engineered from a previous project version various issues were encountered.

At the point of initial setup of the site (<http://www.virtualths.org>) on to localhost, it was realized that the database, all image files and few source files were missing. This issue was quickly resolved after contacting the previous developers.

Once all the source files were placed on the localhost, various broken links were found. This issue was resolved by converting all absolute paths to relative paths.

Some of the php scripting had unrecognized tags on my Wampserver due to the use of shorthand. Scripting block with `<?>` and end with `?>`, only work on servers with shorthand support enabled. W3School PHP tutorial recommends “for maximum compatibility use the standard form (`<?php>`) rather than the shorthand form.” [PHPS] Hence to insure the site receive maximum compatibility, all `<?= ?>` were replaced with `<?php echo`.

The last issue involved understanding the correct hash to user password creation, because unfortunately it was not clearly documented. This was resolved by studying the code to create correct hash combination.

### 4.2.2. User-interface modifications

Modifications to the user-interface of the original prototype were made. The system now has the capability to automatically link the patients’ epidemiological data to the diagnosis section; hence it eliminates entering duplicate data, as shown with the gender attribute in the following php code:

```

<code>
</code>

```

All the attribute display names on the assess page were truncated, by using code like the following:

```
.str_replace( array("Attribute", "Attribut"), "", $row[0] ) .
```

The above code truncated the long display name of AttributeAgeBirthDay to AgeBirthDay; hence other changes removed the follow beginnings from display names: Complaint, Examination, Exam, LabsRadiology, and LadsRadiology. These beginnings were removed because they were not unique to the table they were extracted from; therefore created unnecessary long names.

## 5. EVALUATION OF THE PRODUCT

This product has an execution time of 1 min/cycle hence the preferred training of 20000 cycles was not executed due to time constraints. Ideally the Error should be below 0, unfortunately this was not the case in each of the trail run the error resulted in 813.6308540533921. This is due to overfitting of the input data. Overfitting is a disaster caused by modelling sampling error or noise instead of the correct regularities in a training set. The Neural Network can only be as good as the data used to train it, because if the training data contains unreliable data a sampling error will occur.

In all of the experiments the neural network incorrectly learned that all diagnosis cases matched diagnosis index 1. This can be partially due to the fact a majority of the input data is not extremely unique as the diagram shows below:

index	AttributeAgeBirthDay	AttributeAgePremature	AttributeAgeNewBorn	AttributeAgeNeonate	AttributeAgeInfant	AttributeAge1.5YearChild	A
1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1

Diagram 30: Subsection of vths Training data

### 5.1 Experiment Trial #1

#### Trained with the following parameters

- 20 cycles (20 mins)
- Learning Rate = 0.8 ,
- Momentum =0.3

The execution trace is as follows:

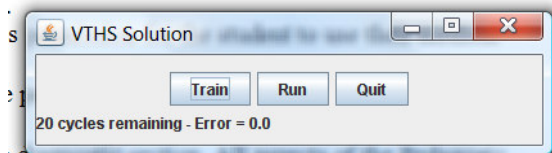


Diagram 31: Initial start point

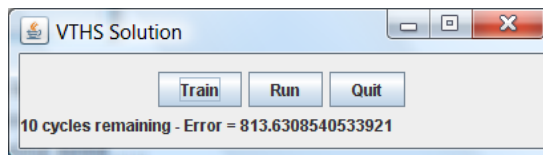


Diagram 32: The constant Error of 813.631 when cycle > 1

	expected	actual
1	0.99999999658985	0.99999999658985
2	0.99999999658985	0.99999999658985
3	0.99999999658985	0.99999999658985
4	0.99999999658985	0.99999999658985
5	0.99999999658985	0.99999999658985
6	0.99999999688364	0.99999999688364
7	0.99999999692808	0.99999999692808
8	0.99999999700746	0.99999999700746
9	0.99999999694288	0.99999999694288
10	0.99999999704666	0.99999999704666
11	0.99999999697026	0.99999999697026

Diagram 33: Subsection of results table

Notice that in every case the neural net diagnosis the input to the diagnosis index #1. This is an example of overfitting.

## 5.2 Experiment Trial #2

### Trained with the following parameters

- 300 cycles (5 hrs)
- Learning Rate = 0.8 ,
- Momentum =0.3

The execution trace is as follows:

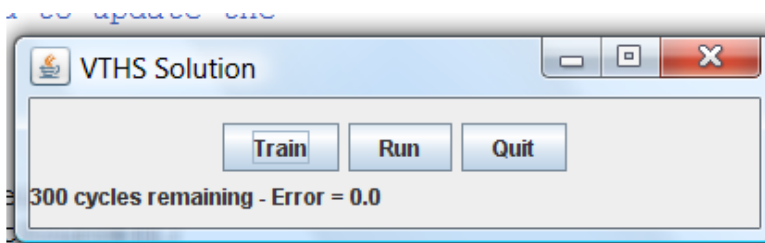


Diagram 34: Initial start point

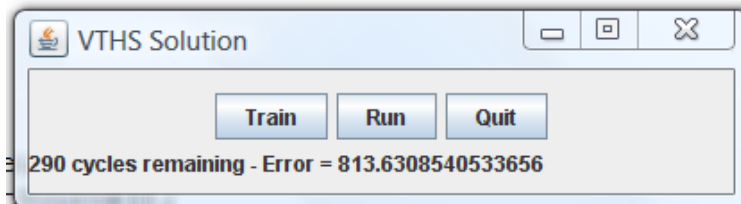


Diagram 35: The constant Error of 813.631 when cycle > 1

After 300 cycles Error equated to the exact same result as the previous trial run of 813.63 hence number of cycles had no effect on this Neural Network.

### 5.3 Experiment Trial #3

In this trial run the momentum value was increased from 0.3 to 0.9. This produced a slight change in the results table, where the actual values all equalled to exactly 1 opposed to a decimal value close to 1; however the Error value equalled the same value as the previous two trial experiments.

**Trained with the following parameters**

- 20 cycles (20 min)
- Learning Rate = 0.8
- Momentum = 0.9

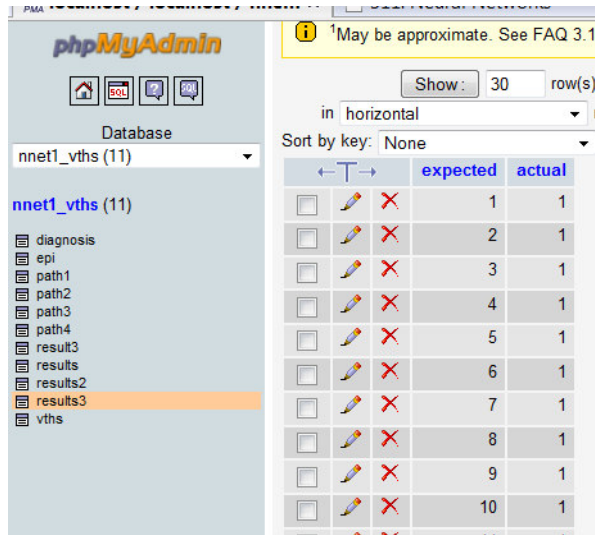


Diagram 36: Subsection of results table

### 5.4 Testing on simple data

In order to prove that the VTHS neural network program works on simple database table this section will show that the neural network can learn the XOR truth table.

The training data is as follows:

input1	input2	output
0	0	0
0	1	1
1	0	1
1	1	0

Diagram 37: XOR Truth Table

**Trained with the following parameters**

- 20 000 cycles
- Learning Rate = 0.8
- Momentum = 0.3

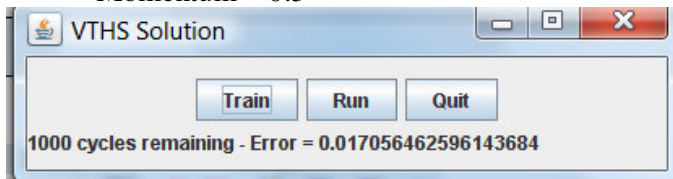


Diagram 38: Run-Time with 1000 cycles remaining

There is a high level of accuracy, due to the fact that after 19000 cycles the error is the extremely low value of 0.017056462596143684.

The results table is as follows:

expected	output
0	0.0123968490576025
1	0.984735581262885
1	0.984533606355905
0	0.0191707351409607

Diagram 39: Results table

When the output column values are rounded they equal to corresponding expected value; hence the VTHS neural network can successfully learn from simple distinct datasets.



## 6. Conclusion

In this section we will present a critical appraisal of the project's results by providing a summary of the resulting works, by comparing it to related works, by providing suggestions for future development and by discussing my overall personal project experience.

### 6.1. Summary

The VTHS Neural Network Java class was able to successfully train and run, as described in section 4.1.2; however the saving neural network feature was not accomplished due to time constraints. Implementation of this feature required excess studying of the JOONE API and the construction of an additional Java class. Due to the scope of this project various user interface modification were not implemented due to the order of prioritizes; however major background changes were implemented to increase flexibility of the system as described in section 4.2.1.

### 6.2. Related Work

This section provides a bibliographic account of related work to the SAPIAN-VTHS (Virtual Teaching Hospital System) Electives networking project.

#### 6.2.1. Multi-attribute decision support systems in Healthcare

Hierarchical multi-attribute decision models are based on the decomposition of complex decision problems into smaller and less complex sub-problems. [BM] DEX is an example of an expert system shell for multi-attribute decision support that is based on qualitative attributes and decision rules. DEX has various applications in health care such as breast cancer risk assessment.

In the past hierarchical decision models were developed “manually”, with the collaboration experts and decision analyst; hence expert systems such as DEX, were only used as a ‘computer-based editor’ [BM]. Currently, the discovery of hierarchical decision models from data has become a popular methodology. The data being used in the medical applications comes from patient data, due to the fact that patient data is systematically collected and are available for analysis; hence a variety of data mining and intelligent data analysis methods are being used. One of the new data mining tools which were introduced had the capability to automatically construct hierarchical models from data; it was called HINT. Therefore, when performing a task such as reconstructing the assessment model of breast cancer risk it required 20 000 randomly selected examples as opposed to manually entered decision rules, which were defined by experts.

Similarly, initially the VTHS system decisions were based on manually entered decision rules. These decision rules were defined as ‘diagnosis formula’ as described in section 2.3.2. Like the patient data used above, the VTHS system used clinical

cases to reconstruct hierarchical decision models. Hence DEX is similar to the initial state of the system whereas HINT is similar to the presently evolved state of the VTHS system.

### 6.2.2. Artificial Intelligence in Medicine

Artificial Intelligence (AI) is a form of machine intelligence that has the ability to deal with incomplete and imprecise information and to accumulate knowledge. [PF] There are three aspects of information pre-processing technology which are: neural networks, expert systems and software languages. Neural networks are intended to emulate the capabilities of the human brain; hence they are taught rather than programmed, whereas expert systems are programmed.

In the VTHS project we transformed the initial expert system into a neural network, by revising the diagnosis component. The updated version of the VTHS system is designed to contain a Neural Network diagnosis component based on structure, memory, learning, action mode and faults.

AI has been increasingly used throughout clinical laboratory practice both for clinical decision-making and knowledge based systems. Neural Networks are used in Medical Diagnosis in the following particular domains: Myocardial infarction, ECG, Neonatal chest radiographs, Dementia and IR spectroscopy. [PF] Contrary to the VTHS system, each example focuses on a particular diagnosis domain, whereas the VTHS system diagnoses a vast range of diseases.

### 6.2.3. Neural Networks in Medical Diagnosis

A Neural Network (NN) is an aspect of Machine Learning, which bases decisions on learnt experiences. In technical terms a NN is based on a collection of many examples with specified correct output for a given input. A machine learning algorithm uses these examples to produce a program that solves problems. This type of learning is called supervised learning, which is the ability to learn to predict output given input vector.[HG]

In supervised learning each training case consists of an input vector  $x$  and a desired output  $y$ . Learning means adjusting the parameters to reduce discrepancy between desired output and actual output produced by the model. It is important to learn with hidden units because without an extra layer the system will become too simple and produce rigid outputs. A NN is successful when it shows balance between the error that exhibits with its training and set its power to generalize based on its training set. [PF]

Similarly, when a neural network implements a medical diagnosis system supervised learning can be used, such that each training case is represented by the input vector of symptoms and desired output diagnosis.

The following MI diagnosis function corresponds to the diagram below. It is a non-linear regression function, which is used for training purposes i.e. setting the values of weights' thresholds. The 5 on the outer summation formula corresponds to

the 5 hidden weights, whereas the 6 on the inner summation formula corresponds to the 6 inputs.

$$F_{MI}(X) = \sigma \left( \sum_{i=1}^5 w_{ij}^o \sigma \left( \sum_{k=1}^6 w_{jk}^i x_k \right) \right) \text{ [DW]}$$

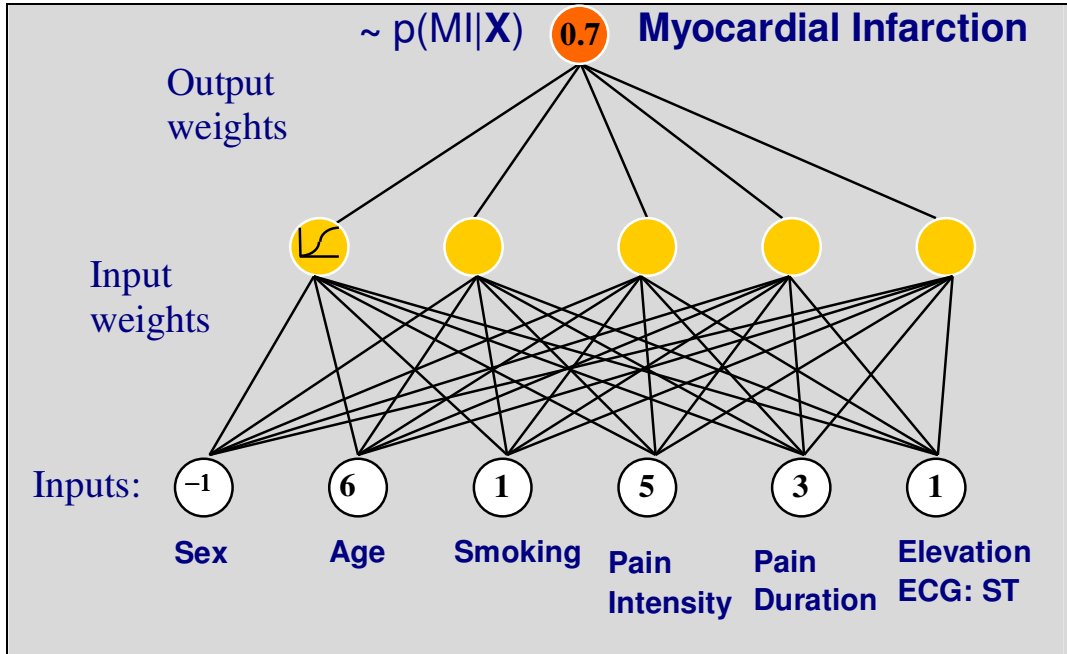


Diagram 40: Neural Network of Myocardial Infarction [DW]

In contrast, the training cases in the VTHS were based on presence of symptom, not the degree of presence. The input values were in Boolean format, therefore set to 0 or 1 only, due to lack of clinical data.

For instance the following table shows the knowledge representation for another system.

Table 1: Knowledge Representation used in MYCIN Expert System [PF]

Represents	Input value
“definitely”	-1.0
“almost certainly not”	-0.8
“probably not”	-0.6
“no evidence”	-0.2 to 0.2
“slight evidence”	0.3
“probably”	0.6
“almost certain”	0.8
“definite”	1.0

#### **6.2.4. E-Learning**

E-learning includes applications of computer-based learning, web-based learning, virtual classrooms and digital collaboration. [PW] Web-based learning is a form of e-learning which is delivered via the internet. Similarly, the web-based learning methodologies were employed in the VTHS system. The system provided online capabilities for students to learn and teachers to teach.

Computer-Aided Education (CAE) systems provide a problem-centred approach for cooperative learning and constructive problem-solving skills. The problem centred approach is based on the pedagogical development. The student is in charge of the learning process, and the system only provides intelligent clues without providing a blueprint solution. [PJ] Similarly, the VTHS diagnostic teaching system emphasized the pedagogic development process by conforming to the VTHS Pedagogic Teaching Process Flowchart (Appendix B).

## **6.2. Future Work**

### **User-Interface**

There are various non-functional requirements based on the overall look and feel. The client requested a more graphical user-interface, such that the user will select a patient to examine from a virtual waiting room. This will provide the student with visualization of patient demographics. The client suggested a video game perspective for the graphics, where the user clicks on a patient who then walks into the clinical examination room. Additional functional requirements includes the ability to deselect Epidemiological or Pathological parameters, and also the addition of a consents page.

### **Simulated Dialogue**

Since real medical history is obtained with a patient telling their story in their own way and patients are not always available, the system should provide a simulated dialogue on various cases in non-medical language. This can be accomplished by mapping symptom names to simple non-medical phrases. This addition would make the system more user friendly.

### **Neural Network**

I recommend future development in the Java Neural Network to add capabilities to save the trained Neural Network to a 'NeuralNet' object. Then export it to a file in order reuse neural network a multiple number of times.

## 6.3. Personal Experience

This project development successfully overcame three main challenges. Each of these challenges created various levels of difficulties throughout the development stages of the project. The first challenge to be overcome was due to that fact that the project is based on re-engineering another developer's code; which unfortunately led to missing source code which slowed down the initial development. The second challenge is making the system self-learning in order to accomplish high-quality of data; this unfortunately was not succeeded due to overfitting. The final challenge was being able to produce a medical system for the non-familiar medical domain; hence it is difficult to test for accuracy when the medical terms were unfamiliar.

Overall this project has provided me with a beneficial learning experience. It has provided me the chance to apply my Neural Network knowledge to the field of software engineering. My prior experience with Neural Networks was only based on coding the Neural Network on Matlab and using text based training data. For this VTHS I have gone beyond my basic neural network knowledge and have achieved results further than I could have ever imagined. My greatest achievement in this project was the creation of the VTHS Neural Network and its ability to link to the MySQL database to read and write data. I learnt a lot of valuable skills from this projects which will have huge impact my future developments. This project has broadened my arisen in the Neural Network field.

## 7. BIBLIOGRAPHY

- [AP] “Ajax (programming).” Last modified on 28 August 2008. Wikipedia. Accessed on 29 August 2008. [http://en.wikipedia.org/wiki/Ajax\\_\(programming\)](http://en.wikipedia.org/wiki/Ajax_(programming)).
- [BP] “Backproagation.” Last modified on 22 August 2008. Wikipedia. <http://en.wikipedia.org/wiki/Backpropagation>. > Accessed on 6 September 2008.
- [BCKPP1] Bapodra M., Chong J., Kakayev K., Papaya D., and Pickering J. Virtual Teaching Hospital System. Last modified 2008. *University of Leicester*. Accessed on 26 August 2008. <http://virtualths.org/>.
- [BCKPP2] Bapodra M., Chong J., Kakayev K., Papaya D., and Pickering J. Virtual Teaching Hospital System Implementation Documentation. CO2015 Course Work. Leicester: *University of Leicester*, 2008.
- [BL] Bass, L. et al. Software Architecture in Practice. 2<sup>nd</sup> edition. London: Addison-Wesley, 2003.
- [BM] Bohanec, M., et al. “Applications of Qualitative Multi-Attribute Decision Models in Health Care.” International Journal of Medical Informatics. December 1999.
- [BA] Boronat, A. “Analysis, Design, Implementation and Test.” CO7207 Generative Development. University of Leicester, 2008.
- [DDA] “Disability Discrimination Act (DDA) & web accessibility.” Last modified November 2005. User experience services. *Webcredible*. Accessed on 26 August 2008. <http://www.webcredible.co.uk/user-friendly-resources/web-accessibility/uk-website-legal-requirements.shtml> > .
- [DW] Duch, Włodzisław. “Introduction to Neural Networks in Medical Diagnosis.” Neural networks as an aid in medical diagnostics. *Powerpoint slides on Dept. of Informatics, Nicholas Copernicus University, Toruń, Poland*. Accessed on 26 August 2008. Presented at Conference Bayer Diagnostic Lab Symposium, Berlin, April 8-10, 2002. <http://www.is.umk.pl/~duch/ref/02/02-Bayer/Terrytown.ppt>.
- [GH] “Geoffery Hinton.” Last modified on 15 May 2008. Wikipedia. Accessed on 6 September 2008. [http://en.wikipedia.org/wiki/Geoffrey\\_E.\\_Hinton](http://en.wikipedia.org/wiki/Geoffrey_E._Hinton).
- [HJ] Heaton, Jeff. “Using JOONE for Artificial Intelligence Programming.” Last modified on 21 November 2002. Developer. Accessed on 6 August 2008. [www.developer.com/java/other/article.php/1546201](http://www.developer.com/java/other/article.php/1546201).
- [HR] Heckel, Reiko. “Requirements Document Template”. Last modified 2008. CO1006 Software Engineering and System Design.

- <<https://campus.cs.le.ac.uk/teaching/resources/CO1006/worksheets/CO1006TemplateReq.doc>>. Accessed on 26 August 2008. University of Leicester.
- [HG] Hinton, G. “CSC321 Winter 2008 – Lectures.” Last Modified April 2008. CSC321 Winter 2008 Introduction to Neural Networks and Machine Learning. Accessed on 26 August 2008. <<http://www.cs.toronto.edu/~hinton/csc321/lectures.html>>.
- [MP] Marrone, P. JOONE: The Complete Guide Last modified January 2007. Accessed on 26 July 2008. <http://www.JOONEworld.org>
- [NR] Naguib, R. et al. “Artificial Neural Networks in Cancer Diagnosis, Prognosis and Patient Management.” CRC Press London, 2001.
- [PS] Petersen, S . et al. “VTHS Pedagogic Teaching Process Flowchart.” Leicester: *Leicester Medical School*, 2008.
- [PHPS] PHP Syntax. [http://www.w3schools.com/PHP/php\\_syntax.asp](http://www.w3schools.com/PHP/php_syntax.asp) W3Schools 2008. Accessed on 26 August 2008.
- [PF] Place, F. et al. “Use of Artificial Intelligence in Analytical Systems for Clinical Laboratory.” Clinical Biochemistry 28 (1995) 373-389.
- [PJ] Plessis, J. et al. “A model for intelligent computer-aided education systems.” Computers & Education. 24 (1995) 89-106.
- [PW] Pollard E., and Wilison R., “Beyond the Screen: Supporting E-learning.” IES Report. 425, 2005.
- [PL] Pullum, L. L. et al. “Guidance for the verification and validation of Neural Networks.” John Wiley & Sons, Inc. (2007).
- [RR] Robertson, James and Robertson, Suzanne. “Volere Requirements Specification Template.” 2006. The Atlantic Systems Guild. 11<sup>th</sup> ed. Technical Report on The Atlantic Systems Guild Limited. Accessed on 26 August 2008. <<http://systemsguild.com/GuildSite/Robs/Template.html>>.
- [TN] Talwar, N. CO3015 Computer Science Project SAPIAN Apprenticeship Patient Diagnosis System. Final Report for CO3015 Course Work. Leicester: *University of Leicester*, 2006.
- [UM1] University of Leicester’s Faculty of Medicine. Phase 2 Course Doc. Leicester: *Leicester Medical School*, 2004.
- [UM2] University of Leicester’s Faculty of Medicine. Phase 1 Course Document, Five Year Course, MBChB. Leicester: *Leicester Medical School*, 2007.
- [UM3] University of Leicester’s Faculty of Medicine. Phase 1 Integrated Medical Sciences Written Assessment, MB ChB Phase 1. Leicester: *Leicester Medical School*, 2007.

[UM4] University of Leicester's Faculty of Medicine. Integrative Module Workbook, MB ChB Phase I. Leicester: *Leicester Medical School*, 2007.

[WL] Williams, H. and Lane, David. Web Database Applications with PHP and MySQL. Cambridge: O'Reilly, 2005.

[YP] Yadav, Prashant "Pedagogical Effectiveness of Online 1-1 Classes." 18 Jun. 2007. Pedagogical Effectiveness Of Online 1-1 Classes. Technical Report on EzineArticles.com. Accessed on 26 August 2008.  
<<http://ezinearticles.com/?Pedagogical-Effectiveness-Of-Online-1-1-Classes&id=611513>>.



## 5. APPENDIX A: PHASE 2- PORTFOLIO CASE SUMMARY FORM [UM]

PORTFOLIO CASE			No:
<i>Patient Initials</i>	<i>Hospital Number</i>	<i>Age</i>	<i>Gender</i>
<i>Portfolio Case presentation number:</i>			
Phase 2 course document objective(s):			
Has the patient' permission, including for follow up contact by telephone, been sought and recorded in the case notes?			
<b>Referral information</b>			
Source of referral and a summary of key information			
<b>History</b>			
All <b>relevant</b> information gathered from the patient about the presenting illness, co-existing problems, current treatment, <b>significant</b> past medical history and the social and family background. The patient's view of the nature of the problem and their expectations for treatment.			
<b>Analysis of history</b>			
The most likely single cause of the presentation, other possible causes and reasons for these choices. The findings to be looked for on physical examination to help decide the cause.			
<b>Physical examination</b>			
Highlight the findings most relevant to your clinical problem solving by underlining them			
<b>Analysis of history and examination</b>			
Reasons for your choice of the cause of the patient's problem(s) and any other cause that still needs to be considered at this stage			
<b>Formulation of the patient's problem(s)</b>			
Encapsulate this in physical, psychological and social terms (the triple diagnosis)			

## Appendix B: VTHS Pedagogic Teaching Process Flowchart [PS]

