# *Easy Diagnosis: README*

**COSC 310 Software Engineering**

# 1. Installation:

## 1.1. Hardware and Software requirements

- Windows Operating System (XP or higher)

## 1.2. Installation instructions

- Download and install the jre7 from http://www.oracle.com/technetwork/java/javase/downloads/jre7u9-downloads-1859586.html
- Download and install winRar from http://www.rarlab.com/download.htm
- Download the latest production release of MongoDB from http://www.mongodb.org/downloads
    - Extract the.zip file contents to "C:\"
- In "C:\" create a folder labeled "data" and another folder within it labeled "db"
- Right-click on "EasyDiagnosis.rar" and select "Extract files…"
    - Choose a destination and select "OK"

# 2. Execution:

## 2.1. Hardware and Software requirements

- Windows Operating System (XP or higher)
- Java Run-time Environment 7

## 2.2. Execution instructions

- Find and double-click the file labeled "EasyDiagnosis.exe" in the previously chosen location

# 3. Overview

## 3.1. System summary

The Easy Diagnosis system is an interactive chat agent program design for use by a psychiatrist and patient in order to detect symptoms from the patient and help guide the psychiatrist to a diagnosis by providing symptoms, disorders, and conversation guidance.

## 3.2. Feature list

**Interactive GUI:** The program uses a GUI for easy interaction between the user and the system. This includes a text field for user input, a text area for displaying conversation history, and a tabbed panel for additional features.

- The text input field allows the user to enter a sentence (or multiple sentences) and then submit the entry by either pressing the "Enter" key, or clicking "Send".
- The text area displays the entire conversation, prefixing each sentence with the identity of who sent the message (either the user or the system). When a conversation reaches a length exceeding the size of the window, a scroll bar is activated to allow easy browsing. When a conversation ends, the symptom scores and diagnosis scores for the patient are also displayed here along with a message from the system stating its confidence in the diagnosis.
- In the tabbed panel, 4 additional features are available including "Symptoms", "Diagnosis", "Conversation Tree", and "Statistics".

- The symptoms tab lists all the symptoms that the user (patient) has exhibited throughout the conversation and is updated with each input from the user.
- The diagnosis tab provides the current most plausible diagnosis and is updated with each input from the user (keeping the old diagnosis to see how it has changed through the conversation).
- The conversation tree tab provides an extensive listing of all possible responses as well as the chosen response by the system; after each input from the user, the system decides on its next response and displays it along with what responses could be used next.
- Finally, the statistics tab provides varying statistical data (including but not limited to: total symptom scores shown across all patients and total diagnosis scores across all patients) for analytical purposes.

**Database:** In order to collect information on a ranging history of patients, the system stores selected data from each completed conversation to a MongoDB for future use. Within the database is a single collection (table) containing one document (tuple) per conversation. This document stores the patients name, symptom scores, diagnosis scores, trigger words used, and the date/time of the conversation.

### 3.3. Test cases

In order to display the systems capabilities of diagnosing a wide range of patients exhibiting a multitude of symptom combinations, each disorder the system is able to diagnose was researched to compile enough grounds for interacting with the system acting as a patient with that particular disorder. A passing test is considered one which received the correct diagnosis with a confidence level %50 or higher. Each conversation in the file "transcript_works.pdf" is a record of these tests.

## 4. Files and directories

### 4.1. Directories and Source files

- *src\CVcontroller.java* – The conversation controller contains methods for filtering user input, deciding on responses, and controlling conversational flow.
- *src\DBcontroller.java* – The Database controller contains instance variables for MongoDB objects such as documents (tuples) and collections (tables), and methods for interacting with these objects.
- *src\HTcontroller.java* – The Hash table controller contains all of the hardcoded data used by the system (ie lists of trigger words and associated values for symptom and disorder points, and all possible agent responses), and methods for accessing this data.
- *src\STcontroller.java* – The statistics controller contains stats on the current conversation (such as symptom and disorder points) as well as methods for analyzing user input to detect symptoms and infer diagnoses.
- *src\MainController.java* – The main controller manages each other controller and acts as a middle man between the GUI and backend code.
- *src\GUI.java* – Contains all code pertaining to the graphical user interface as well as event handlers for specific user actions.

### 4.2. External libraries

- *Forms-1.3.0.jar* – java library used for creation of GUI from https://developers.google.com/java-dev-tools/download-wbpro
    - Used by GUI.java
- *src\mongo-2.9.2.jar* – MongoDB driver for java from https://github.com/mongodb/mongo-java-driver/downloads
    - Used by DBcontroller.java

## 5. Bugs and Issues

### 5.1. Known bugs and triggers

- Occasionally an error code is thrown at the end of the conversation in an unknown state of the conversation tree and when the user input contains "yes" or some variation of "no". The direct cause of this bug is unknown and it is difficult to replicate.

### 5.2. Issues considered for future improvement

Although the system is functional and proven to be accurate in performing its purpose, areas still exist where improvements could be made. The conversational aspect of the system parallels that of a questionnaire, but to feel more fluent and human some comment-type responses or statements should be built into the system. Another area which could be improved is the statistics tab of the interface. The statistics provided are quite basic and to provide a more detailed understand of the history of patients, symptoms, and diagnoses, more correlations need to be made. A final area that could be improved upon is the GUI itself; the interface has a raw feel to it and could be better styled to feel more inviting and interesting.