Statement of Work
Team of Two Students Version
Final Project
ITS 340

Develop an application that interviews a patient for their medical history using a binary tree to drive the interview process. The data collected from the interview must be stored in a mySQL database and a log file created to record the interview. Think of the interview as a digital physician that asks the user questions regarding their medical history. A graphical user-interface (GUI) must be developed to select a patient and perform CRUD record operations on the patient's demographic information and general medical history.

Implement the application using the Java programming language and the Java Swing library for building the graphical user interface. Use a MySQL database to store the data and a text file for recording the interview.

1. Scope of Work

   Create a graphical user-interface (GUI) that includes forms for: 1) *Patient Selection*, 2) *Patient Demographics*, 3) *General Medical History*, 4) *Allergy History, and 5) Family History* using Java Swing.

   a. GUI functionality:
      i. Each form must have individual fields to display all the data from each associated database table (patient, general history, allergy history, family history)
      ii. The General Medical History form must display one record of information for a selected patient.
      iii. The Allergy History form must show zero to multiple allergies in a list or data grid type control for a selected patient. Upon selection of an allergy from the list, all the detail for that allergy must be displayed in individual text boxes on the form where they then can be edited or new record added.
      iv. The Family History form must display all family history records in a gridview-type control associated with the selected patient. Upon selection of a family history record from the list, all the detail for that family history must be displayed in individual text boxes on the form where they then can be edited or new records added.
      v. Each form that displays data from a database table must have buttons for Edit, New, Save, and Delete.
      vi. Each form that displays data from a database table must open in a "View" mode where the retrieved data can be viewed, but not edited (the fields should be locked). The background of the text fields should be a light gray color. When you click the Edit or New buttons, then the form changes to

"Edit" mode where the fields can be editable. Change the text field background to white when in "Edit" mode.

vii. When the Save button is clicked, save changes or new record data to the database and restore the mode of the form to "View" mode.

viii. Use database stored procedure calls in the Java code to perform SQL operations with the database.

ix. Create a separate directory or source package in the project called DBUtils that holds classes with methods for interacting with each database table.

x. Create a separate directory or source package in the project called Validation that holds classes with methods to perform validation on at least two fields per form.

xi. Designate required fields with an asterisk on the GUI.

xii. When clicking the New button, the form must be set to "Edit" mode and all the fields must be cleared of any data so that a new record of data can be added.

xiii. The GUI must include the functionality to select a patient. Patient selection can be done via a separate form or be included in the patient demographics form. Upon selection of a patient, navigate to the patient demographics form which automatically displays all the selected patient's demographic information. Patients must be able to be selected from a list using a list or gridview-type control and via a search box where a last name can be entered.

xiv. There must be capability to navigate to all forms from every form.

b. Automated Interview functionality:

i. The General Medical History form must have a button to start the automated interview for specific general medical history data.

ii. The Allergy History form must have a button to start the automated interview for allergy history data (see Allergy table).

iii. The Family History form must have a button to start the automated interview for family history.

iv. The patient form must have a button that starts an interview for all medical history including general medical history, allergy history, and family history.

v. The interviews must use Swing graphical components to interact with the user.

vi. Use a binary tree data structure to implement the interviews. Each node must include a question and depending on a YES/NO or TRUE/FALSE answer navigate to either the left or right branch of the tree to ask the next question. Suggestion: You may want to create many smaller binary trees for specific interview subsets rather than one extremely large tree. For example, a tree specific to getting general medical history data or specific trees for asking questions regarding specific general medical history data such as blood type, another for specifically for questions on alcohol consumption, etc. You will have a lot of flexibility on how to implement the tree(s) but **MUST** use a binary tree data structure to drive the interview process.

vii. The interview results must be saved to the proper database tables.

c. Log File Functionality:
   i. Record the interview in a text log file that includes both the questions and answers from an interview. Make sure to also record the patient name and patient id in the log.
   ii. log file must be a text file.
   iii. Use Java File I/O.
   iv. The file name must includ the log file interview, patientID and date.

2. The tasks on this project will be divided up between two students as follows:

STUDENT 1

a. GUI for allergy history (see database fields for data to be collected).
b. Automated interview for allergy history.
c. GUI for general medical history (see database fields for data to be collected).
d. Automated interview for general medical history.
e. GUI for patient selection.
f. Database table for allergy and associated stored procedures.
g. Database table for general medical history and associated stored procedures.

STUDENT 2

h. GUI for patient demographics
i. GUI for family history.
j. Automated interview for family history.
k. Log File.
l. Database table for patient demographics and associated stored procedures.
m. Database table for family history and associated stored procedures.

Note: For a single-member team (by instructor permission only) do the tasks for STUDENT 2 above except substitute "Log File" with "GUI for patient selection".

3. EXTRA CREDIT (20 points per student):

a. STUDENT 1
   i. Medications GUI and Interview

b. STUDENT 2
   i. Immunization History GUI and Interview

4. Database Tables:

Patient Table (patient demographics)

```
CREATE TABLE `patienttable` (
 `PatientID` int(11) NOT NULL AUTO_INCREMENT,
 `PtLastName` varchar(128) DEFAULT NULL,
 `PtPreviousLastName` varchar(128) DEFAULT NULL,
 `PtFirstName` varchar(128) DEFAULT NULL,
```

```
  `HomeAddress1` varchar(128) DEFAULT NULL,
  `HomeCity` varchar(128) DEFAULT NULL,
  `HomeState/Province/Region` varchar(50) DEFAULT NULL,
  `HomeZip` varchar(15) DEFAULT NULL,
  `Country` varchar(75) DEFAULT NULL,
  `Citizenship` varchar(75) DEFAULT NULL,
  'PtMobilePhone' varchar(14) DEFAULT NULL,
  `EmergencyPhoneNumber` varchar(14) DEFAULT NULL,
  `EmailAddress` varchar(128) DEFAULT NULL,
  `PtSS#` varchar(12) DEFAULT NULL,
  `DOB` datetime DEFAULT NULL,
  `Gender` varchar(50) DEFAULT NULL
  `EthnicAssociation` varchar(75) DEFAULT NULL,
  `MaritalStatus` varchar(25) DEFAULT NULL,
  `CurrentPrimaryHCP` varchar(128) DEFAULT NULL,
  `Comments` varchar(254) DEFAULT NULL,
  `NextOfKin` varchar(128)  DEFAULT NULL,
  `NextOfKinRelationshipToPatient` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`PatientID`),
  KEY `I_PtLastFirstName` (`PtLastName`,`PtFirstName`),
  KEY `I_HomePhone` (`PtHomePhone`),
  KEY `I_SSN` (`PtSS#`)
) ;
```

Allergy History

```
CREATE TABLE `allergyhistorytable` (
  `AllergyID` int(11) NOT NULL AUTO_INCREMENT,
  `PatientID` int(11) DEFAULT NULL,
  `Allergen` varchar(254) DEFAULT NULL,
  `AllergyStartDate` varchar(25) DEFAULT NULL,
  `AllergyEndDate` varchar(25) DEFAULT NULL,
  `AllergyDescription` varchar(254) DEFAULT NULL,
  `deleted` tinyint(1) DEFAULT '0',
  PRIMARY KEY (`AllergyID`)
) ;
```

General Medical History

```
CREATE TABLE `generalmedicalhistorytable` (
  `GeneralMedicalHistoryID` int(11) NOT NULL AUTO_INCREMENT,
  `PatientID` int(11) DEFAULT NULL,
  `Tobacco` varchar(50) DEFAULT NULL,
  `TobaccoQuantity` varchar(75) DEFAULT NULL,
  `Tobaccoduraton` varchar(75) DEFAULT NULL,
  `Alcohol` varchar(50) DEFAULT NULL,
  `AlcoholQuantity` varchar(75) DEFAULT NULL,
  `Alcoholduration` varchar(75) DEFAULT NULL,
```

```
  `Drug` varchar(25) DEFAULT NULL,
  `DrugType` varchar(254) DEFAULT NULL,
  `Drugduration` varchar(75) DEFAULT NULL,
  `BloodType` varchar(10) DEFAULT NULL,
  `Rh` varchar(10) DEFAULT NULL,
  `deleted` tinyint(1) DEFAULT '0',
  PRIMARY KEY (`GeneralMedicalHistoryID`),
  KEY `GeneralMedHxPatientIDIndex` (`PatientID`)
) ;
```

Family History

```
CREATE TABLE `familyhistorytable` (
  `FamilyID` int(11) NOT NULL AUTO_INCREMENT,
  `PatientID` int(11) DEFAULT NULL,
  `Name` varchar(50) DEFAULT NULL,
  `Relation` varchar(50) DEFAULT NULL,
  `Alive` tinyint(1) DEFAULT '0',
  `Lives with patient` tinyint(1) DEFAULT '0',
  `MajorDisorder` varchar(254) DEFAULT NULL,
  `SpecificTypeDisorder` varchar(254) DEFAULT NULL,
  `DisorderHRF` tinyint(1) DEFAULT '0',
  `deleted` tinyint(1) DEFAULT '0',
  PRIMARY KEY (`FamilyID`),
  KEY `I_PatientID` (`PatientID`)
)
```

Major Disorder Example: Cancer, Heart Disease, etc.

Specific Type Disorder:

Cancer: Lung Cancer, Breast Cancer, etc.

Heart Disease: Congestive Heart Failure, Heart Arrythmia, High Blood Pressure, etc.

Project Extra Credit Tables: Immunization and Medication Tables

Immunization History

```
CREATE TABLE `immunizationshistorytable` (
  `ImmunizationsID` int(11) NOT NULL AUTO_INCREMENT,
  `PatientID` int(11) DEFAULT NULL,
  `Vaccine` varchar(128) DEFAULT NULL,
  `ImmunizationDate` date DEFAULT NULL,
  `ExperationDate` date DEFAULT NULL,
  `Delivery` varchar(128) DEFAULT NULL,
  `Comments` varchar(254) DEFAULT NULL,
  `HCPId` varchar(75) DEFAULT NULL,
  `deleted` tinyint(1) DEFAULT '0',
```

```
  PRIMARY KEY (`ImmunizationsID`),
  KEY `I_PatientID` (`PatientID`)
) ;
```

(Includes form to display all prescriptions and enter/edit/discontinue medications. Include buttons to show ALL medications both current and past and button (the default view) to show only current medications. Check the medication end date to see if a medication has been discontinued.)

```
CREATE TABLE `medication` (
  `MedicationID` int(11) NOT NULL AUTO_INCREMENT,
  `CurrentMedicationID` int(11) DEFAULT NULL,
  `PatientID` int(11) DEFAULT NULL,
  `Medication` varchar(254) DEFAULT NULL,
  `Quantity` varchar(50) DEFAULT NULL,
  `QuantityUnits` varchar(50) DEFAULT NULL,
  `MedicationOrderHCP` nvarchar(75) DEFAULT NULL,
  `MedicationOrderDate` datetime DEFAULT NULL,
  `Instructions` varchar(1024) DEFAULT NULL,
  'MedicationStartDate' date DEFAULT NULL,
  'MedicationEndDate'  date DEFAULT NULL,
  `deleted` tinyint(1) DEFAULT '0',
  PRIMARY KEY (`MedicationID`),
  KEY `I_PatientID` (`PatientID`)
) ;
```