Wilson McGill Spring 2023 Ears Project

This is a document for describing the codes function in detail in all parts.

It is for the future groups who work on the project to utilize.

The general design overview and some technical aspects were gone over in the General documentation file, this is specifically for the technical aspects of the project.

https://github.com/Jamtam2/EarsLobo <- link to the Github with all code

General Information

The RubyonRails folder contains the actual working Rails environment for the software.

This project is created entirely using the Ruby on Rails framework and built in database. There was some JavaScript injected into various pages for dynamic functions especially the test page.

Design process (basic)

The general design process for the skeleton of this project centered around the examples and information provided in the Demos and Deets for Ruby on Rails.

The relations and database items follow the same format.

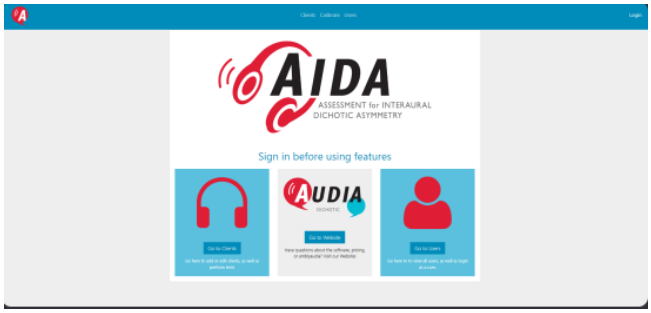
https://human-se.github.io/rails-demos-n-deets-2021 <- link to setting up Ruby on Rails Environment

Front End development

The front end of the project primarily was designed to be simple and in your face, this will be used in place of older more dated software that users are used to and in places where changes to UX were necessary the goal was that the new way of doing things be so obvious that anyone could unlearn and relearn how to perform tasks.

Where possible we avoided small text dropdowns and tab systems used commonly in the old app, here is an example from the old and new app versions.

Old vs. new



Most features of the front end html is non dynamic basic buttons, due to ease of use and simplicity of implementation, use of more dynamic systems caused problems between developers environments.

All styling is using Bootstrap 4.

New, Index, Edit Pages

In Ruby on Rails web applications, three standard pages are typically associated with each model: New, Index, and Edit. Since the underlying code for these pages is quite similar across models (with only model and attribute references changing), it is unnecessary to document them individually. For assistance in understanding the structure and formatting of these pages, refer to the documentation provided for the Client's Pages (found in app/views/clients/).

Here's a brief summary of the purpose of each page for those who may be unfamiliar:

New: This page allows you to create a new instance of a model, such as adding a new client.

Index: This page displays a list of all instances of a model, such as showing all created clients.

Edit: This page enables you to modify the attributes of a model instance.

If you are having any trouble formatting these pages, you can use https://human-se.github.io/rails-demos-n-deets-2021 as a reference

TABS

Tabs were implemented to enhance the organization and flow of pages within the web application. However, they currently require a page refresh before any content is displayed. If you're interested in incorporating tabs into the web app, you can find the relevant code in the **BrokenCode** folder on GitHub, specifically it will be an edit.html.erb file for the Client model.

While no known solutions exist for eliminating the need to refresh the page, this file will provide a starting point for continuing the troubleshooting process from where it was previously left off.

Back end development

The back end development focused mainly on successful client creation and saving, as well as test saving and creation.

These were again based on the old site and the demos and deets examples.

There were instances of Ruby logic being used to calculate some values for use in the test page, for example: This calculates a clients age in years based off their date of birth and provides the correct age to the test page JavaScript.

```ruby
#age in years method that calculates a clients age based on DOB then passes it to the script test page
    def age_in_years
      now = Time.now.utc.to_date
      dob = date_of_birth
      age = now.year - dob.year
      age -= 1 if now < dob + age.years # for days before birthday
      age
    end
end
```

The largest and most complex javascript that will be necessary to use and understand when creating new and different tests on the app is the

EarsLobo / RubyonRails / app / views / tests / **new.html.erb** page.

This contains all the logic for calculating a child's diagnosis in real time based on the buttons checked and age of the client.

This is an example of of the logic for diagnosis based on score values provided by Deborah for ages 13-15.

```
if (age >= 13 && age <= 15){
  if (direction == "Neutral"){
    if (percent1 <= 76){
      interpretation = "Dichotic Dysaudia";
    }
    else{
      interpretation = "Within Normal Limits";
    }
  }
  else{
    if (((dom <= 92) || (nondom <= 76)) && ((advantage <= -16) || (advantage >= 16))){
      interpretation = "Amblyaudia + Dichotic Dysaudia";
    }
    else if ((dom <= 92) || (nondom <= 76)){
      interpretation = "Dichotic Dysaudia";
    }
    else if (((dom > 92) && (nondom > 76)) && (advantage <= -16)){
      interpretation = "Ambylaudia Right Ear Dominant";
    }
    else if (((dom > 92) && (nondom > 76)) && (advantage >= 16)){
      interpretation = "Ambylaudia Left Ear Dominant";
    }
    else{
      interpretation = "Within Normal Limits";
    }
  }
}
```

A test plays audio in the left and right ear channels to the patient and says different words at the same time. The child then confirms what they heard and buttons are checked if they got it right. The difference between the left and right scores and the nondominant and dominant ears are used to calculate the Diagnosis the child receives.

This is one of the most important things to understand so confirm with Deborah talking in real time that the diagnosis system is correct.

Conclusion

The code is commented at all relevant features and examples, this should give insight into how development was performed and should continue to be performed. Future groups have discretion to change update and manipulate the code at Deborah's approval and instructions. Good luck!