Project Description

Project Summary

It is often tedious for assessors to mark verbose solutions to extended response questions. Many of these solutions may be identical but only differs in the phrases that are selected by the student. By streamlining the question down to simple statements and connecting the statements in a tree-like structure, it can make marking easier and facilitate students' learning. BioLogic presents an interactable Web App that is capable of running on a browser for students to build answer trees using *statements* and *connectors*. This project will focus on the front-end component, using Vue.js as the framework.

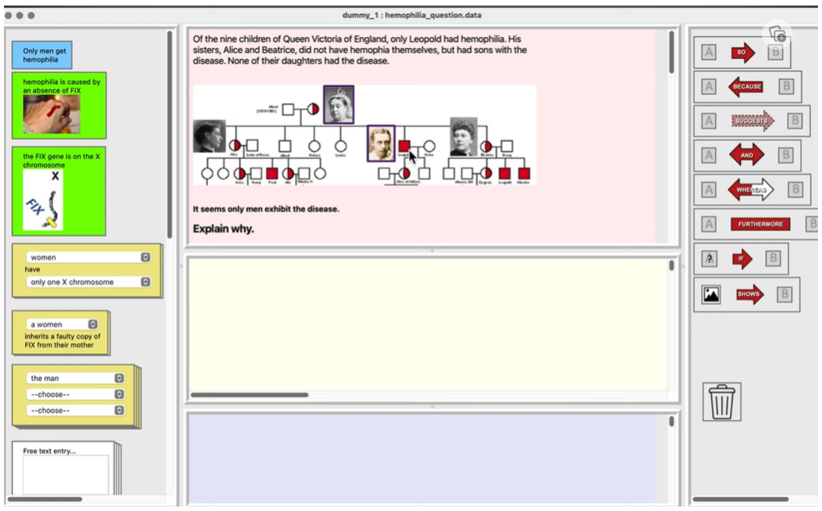
Need and Purpose

Extended response questions often require a presentation of multiple statements. These statements can be connected with various logical connectors. These connectors can be presented in many forms, e.g. [A] so [B] is identical to [A] therefore [B]. This introduces great variability in the responses, often with minimal difference between the actual intended message. This adds unnecessary work to the assessor, as they now have to spend time interpreting seeming different solutions while they convey the same solution.

To simplify this process, we can break the aforementioned statement [A] so [B] into three components - statements [A], [B] and the logical connector "so". BioLogic aims to give assessors the tool to harness this structure for their questions. Assessors create their question and define their *statements* (i.e. [A] and [B]), and the student can connect these statements using the logical operators (i.e. "so") that is available to them.

Current Solution

A trial of this idea has been built in the form of a local application by a different team from the client. The current solution is an application built in Python. This application realise the above technique through a drag-and-drop interface.



dummy_1 : hemophilia_question.data

Of the nine children of Queen Victoria of England, only Leopold had hemophilia. His sisters, Alice and Beatrice, did not have hemophia themselves, but had sons with the disease. None of their daughters had the disease.

It seems only men exhibit the disease.

Explain why.

Only men get hemophilia

hemophilia is caused by an absence of FIX

the FIX gene is on the X chromosome

women have only one X chromosome

a women inherits a faulty copy of FIX from their mother

the man --choose-- --choose--

Free text entry...

The application is able to parse local files into the question's *query* and *statements*. In addition, the application supplies its users with *connectors*, which are phrases that links two statements together (e.g., because). The students can then start to construct their answer by dragging *statements* and *connectors* onto the workspace in the middle. The students can verify their solution with the worded output located at the bottom of the screen, which is their constructed answer in words.

Current Problems

While the current program is able to help students construct the answer, it is only a stand-alone application and hence is not able to communicate with a server. It also requires all the questions downloaded onto the device beforehand in very specific locations. Therefore, to use the current program, the student will need to download the entire application and files, construct their solutions and submit the constructed solution through some alternative means. This is of great inconvenience to a lot of students as they have to download an application, install Python and learn how to run the application using Python. Ideally, the client would like the application to be accessible via a website, ultimately integrating it into the current University of Melbourne Learning Management System (LMS).

Proposed Solution

As the client has cleared stated their desire for the application to be used through a browser, we have decided to build a web application, duplicating as many features as possible from the current Python iteration, in addition to being able to communicate with a backend server. The client has a separate team working on the backend server, hence only the frontend of the application is required. By building a web application, it should be able to run in a web browser, which means that no download is necessary. Furthermore, communicating with the backend server means that questions no longer need to be downloaded: the application can retrieve the related information from the backend server. Lastly, students can submit their work directly in the application.

Scope

The scope of the project will include the preparation, planning, development and deployment of the front end of the BioLogic Editor, which can:

- run in a web browser
- interpret a file and display the question *query* and its corresponding *statements*
- allow the user to construct an answer using the *connectors* and *statements*
- display the sentences constructed using the *connectors* and *statements*
- submit the solution to a built backend server in a form that is consistent with the protocol defined by the server.

Out of Scope

While the application facilitates a student in constructing an answer, it will *not* contain the following features:

- let a user build a question
- allow a marker to view a submitted answer
- automatically grade a submitted answer
- be integrated onto the LMS
- be hosted on a website (i.e., it will only be able to run on a local server).

Project Team

Name	Role	Description
Anwen Xu	Product Owner	Anwen will ensure that the product satisfies the needs of the client, fulfilling its objective and business values.
Nan Du	Scrum Master	Nan will ensure that the team works in an efficient agile fashion, holding daily stand-up meetings and making sure tasks are completed on time.
Zhihong Sui	Developer	Zhihong, Xuemin and Yilin are responsible for the coding, programming and testing of the product to ensure it works as intended.
Xuemin Liu	Developer	
Yilin Li	Developer	