Advanced Modular Manikin Speech Module for Traumatic Brain Injury Applications

Authors: Johnston, T., Nguyen, H., Orji, F., Stoddart, L.

BIOEN 405: Team Design

June 10th, 2021

Trevor Johnston : Helped develop the foundational NLP platform (connection with Microsoft Azure programs, creation of the knowledge bases), short-term memory function file, neurologist & medical personnel usability study design, code commenting and streamlining, researched content for knowledge bases, designed and conducted and analyzed correct response frequency experiment, creation of user interface console when editing knowledge bases.

Hienschi Nguyen : Helped develop the foundational NLP platform (patient function, function integration/revisions), Serial Sevens function files, code commenting and streamlining, student usability study, block diagrams of NLP program, developed Getting Started guide, established project on GitHub, completed GSC integration for all function files, and student usability study design.

Favour Orji : Biogears integration, Student usability study data analysis, slurred speech function file and its code commenting.

Laila Stoddart : Helped develop the foundational NLP platform (connection with Microsoft Azure programs, creation of the knowledge bases), creation of the Advanced Cerebral Circuit Model, confidence interval variable creation and function file, mathematical function file, time/date function file, advanced the slurred speech function file, software engineer usability study, researched content for knowledge bases, designed and conducted and analyzed response time experiment.

Purpose:

To increase physician competence performing neurological examinations by establishing interactive natural language processing capabilities specific to neuropathies in a medical training manikin by UW CREST (Advanced Modular Manikin).

Abstract:

Physician's lack of education and experience surrounding neurological exams has led to an overreliance on imaging and a high rate of TBI misdiagnosis (1,2). However, medical simulations are revolutionizing medical teaching and training. These simulations offer a means for medical students and emergency responders to gain clinical skills while gaining real life experiences and reducing diagnostic errors that can occur for neuropathological diseases. The goal of this project is to develop a natural speech processing program that simulates a conversation with a neuropathy-diagnosed patient during a neurological examination. In order to achieve higher fidelity, this program will be integrated with an existing CREST Advanced Modular Manikin (AMM), a manikin that uses live physiological data from Biogears for simulation of diseases and diagnostic training. The natural speech processing program can render a physiologically-dependent vocal response to a user's audio input. It accomplishes this by utilizing Microsoft Azure's speech-to-text and text-to-speech functions, Microsoft Azure's knowledge base encoder, and unique real-time functions created by the students. The real-time functions enable this program to overcome the technological barriers surrounding the implementation of this type of simulation by allowing simulated responses to an infinite number of possible questions surrounding the same content. Live functions were made to address simple mathematical calculation questions, short-term memory assessments, serial 7 calculations, and time / date inquiries. Additionally, a slurred speech function was implemented and can be applied to all finalized outputs in order to further increase fidelity.

The program was measured in fidelity, accuracy, and usability of the program. The NLP program accuracy was high in its ability to recognize and respond to speech based on questions from the knowledge base, with usability studies further validating the clarity of program instructions and providing insight into areas of improvement. The integration of the speech processing program to the AMM manikin serves as a foundation for the development of several other pathologies. With this speech processing program, medical students training in neurology will have more opportunities to practice their knowledge and improve the accuracy of patient diagnosis.

Problem Statement and Description

Traumatic Brain Injuries (TBI's) are one of the highest causes of disability and morbidity in the U.S. with 2.87 million TBI-related hospitalizations or deaths occurring in the year 2014 alone (1). Despite the high occurrence rate, physicians report being highly uncomfortable and unconfident performing patient neurological examinations which are necessary to fully diagnose TBI's. Therefore, physicians tend to rely heavily on radiology for diagnostic information, but studies performed within the last 5 years have demonstrated that there is not a strong correlation between structural abnormalities on computed tomography (CT) or magnetic resonance imaging (MRI) scans and the severity of post-TBI symptoms (2,3,4). In fact, algorithms based on patient history and symptoms better predicted the outcome of a TBI compared to predictions that were made based solely from CT characteristics (5). This reliance on radiology alone has led to an alarmingly high rate of TBI misdiagnoses reaching reports of up to 56% misdiagnosis rate for mild TBI's (mTBI) and 51% misdiagnosis rate for severe TBI's (sTBI,6,7).

The misdiagnosis of a TBI, both mild and severe, can have serious consequences to the patient. Despite being labeled as mild, mTBI's can result in significant disabilities in cognitive, physical, psychological, or social functionality if appropriate intervention is not immediately implemented. The National Institute of Health has declared the under-diagnosis of mTBI's a major public health concern due to the risk to the patient (8). Misdiagnosing a sTBI obviously has even worse implications. Patients that were misdiagnosed for a sTBI are significantly observed to have a lower quality of life at 3 months and an increased risk of severe disability or death at 12 months (7).

From a financial aspect, the misdiagnosis of a sTBI is one of the largest sources of malpractice lawsuits and litigation claims against a hospital's emergency department (7). Failures of physicians to properly perform neurological associations have resulted in multi-million-dollar lawsuits as seen in the following few cases : Smith v. Baca (9), Chicoine v. DiBaro (10), Orr v. Bell (11). In all mentioned cases, CT's were negative for a TBI, physicians failed to perform a neurological assessment, and the lack of medical intervention resulted in severe brain damage to the patient.

Physicians often misdiagnose TBI's by failing to assess the cognitive functions of the patient. Without proper neurological assessment, the only symptom of an mTBI is a headache. Physicians will generally attribute such a common symptom to other explanations such as chronic pain, whiplash, side effect to medication, sleep deprivation, depression, post-traumatic stress disorder, or even nothing that warrants an explanation at all (12). By improving physician competence in performing these neurological assessments, TBI's would be less likely to be underdiagnosed.

Improving physician competence performing neurological assessments would also decrease the overuse of radiology. Up to 35% of CT's ordered for patients with a suspected TBI were not warranted by the results of the neurological examination according to national guidelines. These additional medical imaging scans increase the cost of healthcare to the patients and increase their risk of developing radiation-induced cancer (13).

Our project therefore aims to increase physician competence performing neurological examinations by establishing interactive natural language processing capabilities specific to neuropathies in a medical training manikin by UW CREST (Advanced Modular Manikin). Medical simulations, a technique to replace and amplify real life experiences with guided ones(14), are a novel method for teaching and training new and upcoming medical students. These simulations have been incorporated into medical school clerkship education and in emergency medicine as they provide a method to simulate complex and challenging medical conditions while providing a means to engage and resolve ethical and practical dilemmas. Manikins, full body patient simulators, safely allow for the training of clinical skills, cognitive thinking and behavioral communication in a professional healthcare setting(14,15). These manikin's prepare medical scholars by simulating situations that can arise in the field especially as it aids new physicians who struggle to apply theoretical knowledge to clinical practice; especially as WHO organizations report that diagnostic errors are one of the highest priority problems for patients safety and recommend both cognitive (second opinion) and system-based interventions(14). Medical simulations can aid in the understanding of neuropathological diseases, diseases that impact the eyes and nervous system, which can result in a reduction in physician errors while increasing confidence in the diagnosis of these diseases. Therefore, our aim is to

develop a medical simulation that uses natural language processing to mimic the interaction a physician would normally have with a patient that has that neuropathy.

Market Analysis

The end users of our medical simulation program are medical students and medical military personnel in training. The two primary entities that will purchase our program as part of the AMM will therefore be hospitals and government agencies. Other users of our program include software engineers and students who wish to add functionality to the program. However, these users fall under the umbrella of developers and will be accounted for in design considerations rather than market analysis.

Existing interactive medical simulation manikins cost upwards of $20,000 and are therefore only available to large entities (16). Given the 15.73% compound annual growth rate in Healthcare Simulation Anatomical Models (the largest share of the healthcare simulation market), the existing price range seems to be non-prohibitive (17). Currently at .5 billion USD, the market for Healthcare Simulation and Anatomical Model is expected to grow 1.5 billion USD by 2026.

Our TBI simulation program aims to fill a gap in the existing market for medical manikins. With our program, the AMM will become the first medical manikin to play pre-recorded sounds, to incorporate natural language processing and speech recognition, to dynamically generate a response, to have a TBI specific scenario or module, and to have TBI relevant speech capabilities. As seen in the table below, existing products do not contain all these features, and therefore our program in conjunction with the AMM fills a significant gap. Features of the other existing manikins and competitors will be elaborated on in the Prior Art section.

Device	Pre-recorded sound	Speech recognition/NLP	Generated response	TBI module	TBI relevant speech
US9406244B2	X				
US9280147B2	X	X			
SimMan + METIman	X				
ALEX	X	X	X		
HAL S3000	X			X	
Market Gap	X	X	X	X	X

Table 1: *Summary of the speech- and TBI-relevant features of existing manikins and products on the market. The final row demonstrates our target market gap*

Prior Art

Currently the University of Minnesota and the Center for Research in Education and Simulation Technologies (CREST) are developing a modular patient simulation which brings together various modalities (physical, VR, AR) which has the ability to render the state of the patient based on cues from the provider, sense actions taken by the provider in order to make decision and capture decisions/actions that can be used for performance evaluation(18). ALEX, a manikin by Nasco Healthcare, is the first full-body patient communication simulator that sees, listens and talks using artificial intelligence by full-body physiology with verbal functionality and analyzes various manikins(19). The manikin has built in speech and touch recognition which automatically responds in conversation style based on predefined medical interview questions and is used to provide feedback on common procedures like IV, wound care and catheterization however it has no modularity and doesn't offer any speech-based diagnosis for neuropathologies. The SimMan, a high-fidelity manikin which incorporates the latest in computer technology can display neurological symptoms as well as physiological ones (20). The manikin provides a realistic full-body patient, and offers pre-recorded sounds that simulate patient voice wirelessly; the manikin also includes a breathing feature that stimulates spontaneous breathing, bilateral and unilateral chest rise and fall, CO2 exhalation and normal and abnormal breathing sounds. Other neuropathological simulators include the Human Actor-Based Simulator to Assist, a simulation that is not restricted to manikins that

improve critical care by improving trainee knowledge and confidence. The simulator uses actors and SimMan 3G Simulation Manikin to simulate patients using three neurological scenarios: myasthenic crisis, status epilepticus and brain death.

While current medical manikins already incorporate natural language processing, they lack neuropathology applications(21). Working with the CREST lab, we hope to develop a neuropathy specific language processing program. Ongoing design work in this field includes the METIman® human patient simulator from CAE healthcare(22). The METIman human patient simulator is a human-like clinical simulator that offers highly responsive physiological experiences (blinking, reactive pupils, bilateral chest movement,etc). This simulator is novel in that it automatically simulates physiology without trends with an updated airway that replicates challenging reality-based scenarios which can be seamlessly integrated into nursing and medical curriculum. Current patents and intellectual properties in the field of medical simulations are highlighted below.

Patent # : US9406244B2

Interactive education system for teaching patient care(23)

This patent describes the mechanical mechanisms of a physical manikin to simulate childbirth for an interactive learning experience. This patent claims the mechanical mechanisms to model cervix dilation, breech delivery, contractions, and more. The claim also includes the ability to communicate data wirelessly between an external device and the maternal simulator. The manikin contains a voice module that is able to play pre-recorded responses chosen by the user. Some responses include location (“my leg”), confused answers (“Are you a doctor?”), occurrence answers (“since last week”), or other sounds (coughs, gagging). Our intended project is to establish question and answer responses controlled by a program rather than a human user. The module described helps us build an idea of the content to include in the response library to simulate natural conversation based on the patient’s physiology. Besides pre-recorded response, the manikin is described to be able have a user speak into a mic to act as the patient and uses audio amplifiers and sound boards to mimic hoarseness or having blocked air passage. For the team, this confirms the feasibility to alter a pre-recorded patient voice for different breathing rates or hoarseness. Although the mentioned features are very similar to our project, they are not directly

patented in the claims. Since their voice module is not listed as a claim, the team is able to move forward with the project. If the team considers to patent an aspect of the project, then it can be a possibility.

Patent # : US7702508B2

System and method for natural language processing of query answers(24)

This patent describes the process of integrating a natural language processing (NLP) component on webpages in order to facilitate e-commerce. In this system, the user delivers a vocal question into their device's microphone. This audio is then sent to an online server that utilizes natural query language to perform speech-to-text translation. After converting these data into structured query language (SQL), noun phrases can then be extracted using a natural language engine. The extracted noun phrases are then run against a SQL database which contains all of the stored questions and corresponding answers specific to that website. The program attempts to locate a stored question within the SQL database that is linguistically similar to the question asked by the user. After selecting the best match to the user's original question, the response linked to that question is then converted into a text file. The text file is then uploaded to the online server's text-to-speech function. For our intended project, we plan to incorporate a natural language processing component into the AMM. Our proposed method for accomplishing question recognition is very similar to the process described above. However, generating the appropriate patient response in the AMM will be a bit more challenging. Rather than having set answers to each question type, we have to guide the patient response according to additional variables that define the diagnostic and physiological state of the simulated patient. This means that multiple SQL databases will need to be created and these patient variables will have to encode for which SQL database the original question is run against. Since we will be adding these unique components to the NLP method presented here, we can move forward with the project and potentially apply for a patent.

Patent #: US4828501A

Compact interactive training manikin system(25)

This patent describes a cardiopulmonary resuscitation training manikin that includes a speech synthesizer of pre-recorded human speech wherein a students actions

are correlated to real-time voice feedback. The patent describes the compact, inexpensive electronic controller system wherein pre-stored phrases/speech, in the form of coaching instructions, is stimulated, synthesized and reproduced. For our project, we aim to create real-time patient voice feedback wherein a provider is able to engage with the manikin in order to effectively assess baseline. As such, this patent confirms the feasibility of generating said real-time feedback system. Furthermore, the patent acts as a starting off point especially as our team expands the scope of our project to having our sound module integrated with breathing. In order to integrate breathing, the team can adopt the patent's implementation of a switching transducer. To avoid overlap with this patents' claim, the project needs to avoid a breathing feature that is activated externally by an instructor. If the project scope widens to integrate breathing, the feature proposed would be automated breathing that is reflective of the patient's conditions based on additional variables and the questions asked by the provider. This additional reflective feature will help prevent overlap with the patent's claim.

Patent # : US9280147B2

Systems and methods for robotic patient synthesis(26)

This is a patent for a robotic manikin system with four primary components: at least one sensor in an array of sensors, a control system processor, control data, and a synthetic patient robot that carries out commands. The sensor array includes but is not limited to a camera and a microphone. The control system processor includes but is not limited to a facial feature tracker and interpreter, speech recognition, and a natural language processor. Control data consists of data from a preexisting library and of data gathered from the sensors and processor. The libraries contain information about the disease state, pre-recorded sounds, and pre-recorded patient speech. The control processor integrates the preexisting libraries and the gathered data into a verbal and kinetic response. The synthetic patient robot must contain at least one actuator, a speaker, and a means of displaying facial features.Given the similarity of this patent to the AMM, and specifically the claim of a speech function, this patent has several implications for our project. First, we should avoid the exclusive use of unmodified, pre-recorded patient-actor speech given the claims laid by this patent. This means that we should not pursue any "canned" responses,

even for common questions. Text-to-speech must always be used. Second, our mentors mentioned that we should enable our speech module to be utilized with a fully virtual version of the manikin. In this fashion, we can work around this patent which claims a speech module for a physical manikin.

Design Deliverables/Specifications

Our project intended to generate two deliverables: 1) A foundational natural language processing program that allowed for future development and integration, 2) neuropathy-specific language simulation capabilities. We ultimately added a third deliverable during the course of the project: a new advanced cerebral circuit model that allowed for pathological localization.

Our initially assigned project was to advance any aspect of the current AMM model. This gave our design a lot of freedom in choosing the content area of our project. Without a specified content area, our only initial design constraints were those that applied to the AMM model as a whole. Any component of the AMM model needed to be modular, interoperable, and autonomous. On top of these AMM-imposed constraints, we also included remote application and completion as a constraint. Due to the pandemic, our access to UW CREST's labs or current AMM model would be minimal at best. Our project needed to be able to be completed remotely without requiring any in-person meetings between team members or mentors. Finally, we also had additional time and monetary constraints due to the specificities described in the capstone course. The project must be able to be completed within 3-4 months and must cost under $250. The establishment of a software program as our project enabled us to meet all of these needs.

After selecting the final project concept, however, we additionally established some performance goals for our program derived from acceptable values used in other studies (Table 2). Since our program will be used to train medical professionals, high fidelity, accuracy and intent recognition is essential to the success of our project in achieving its overall goal of increasing physician competence performing neurological assessments. However, our program is also a foundational platform on which other engineers, students,

or medical personnel might want to build off of in future works. Therefore, the usability of our program against a wide audience is similarly important to our project's success.

Need	Parameter	Acceptable Goal	Ideal Goal
High Fidelity	Response Time	<3 seconds	<2 seconds (27)
	Content Rating by Neurologist	>8/10	>9/10 (28)
High accuracy and Intent Recognition	Correct Response Percentage	>90%	>95% (29)
Practical for a Wide Array of Users	Percentage of Users Able to Use Program Effectively	>80%	>90% (30)

Table 2. *Project Design Goals*

Solution Generation and Selection

Preliminary Design Generation and Selection

Initial generation of ideas were primarily conducted through brainstorming sessions with the team and feedback from Dr.Hananel and Mr.Gong. Dr.Hananel suggested that the team research existing open-source speech APIs and to talk to Austin Baird, expert on Biogears, to receive further background on Biogears data generation. After following these suggestions, the team found 1) four open-source APIs: Alexakit, Google Cloud, Microsoft Azure, and Sirikit 2) Biogears data generation can originate from a text or Excel file. The team divided into pairs for devising ideas on NLP program behaviors to be able to query a response based on live physiology data and dependent on TBI location and severity. During this brainstorming session, the selection of the programming language and integrated development environment (IDE) was decided.

Speech Recognition Software	Requires Wifi	Costs	Device Requirements	Support
Microsoft Azure	Yes and with limited offline capabilities	Yes	Web	Tutorials + Contacts in Microsoft
Alexa Skills Kit	Yes	Yes	Alexa installation	Tutorials
Siriki	Yes	Yes	iOS/Apple device	Limited Tutorials
Google Cloud	Yes	Yes	Web	Tutorials

Table 3: *Summary of description of each Speech/Text APIs.*

Criteria	Microsoft Azure	Alexa Skills Kit	Siriki	Google Cloud
Real-time transcription	4	4	3	5
Offline Capabilities	2	0	0	0
Costs	4	2	5	3
Team Feasibility	5	3	2	4
Score	15	9	9	12

Table 4: *Summary of criteria for choosing which API service to use. The scale is 0 – 5 where 0 represents does not meet criteria and 5 exceeds standard.*

The team summarized the findings of the four APIs in Table 3. A comparison was done among these four open-source API options based on following criteria: accurate real-time transcription, offline capabilities, costs, and team feasibility. After comparing and ranking options using a Pugh chart as shown in Table 4, the team determined the highest ranked API, Microsoft Azure.

With Microsoft Azure, the team there is able to easily collaborate together without having to be constrained for a specific OS, receive additional support if needed, and potential for offline use. Furthermore, Microsoft Azure utilizes both SQL and deep learning to answer sets of questions and answers while having the capabilities to handle a multiple version of a similar question. It also keeps the information in the cloud which does not require users to update the questions and answers on each manikin's permanent storage. For these reasons, Microsoft Azure was chosen for the NLP program.

After selecting the speech recognition software, the team discussed which programming language to use. Microsoft Azure is compatible with the following languages: Java, Python, Ruby, C++, Go, REST, and CLI. All team members have experience programming in Java and one team member has experience in Python. Based on the criteria of team feasibility and usability, Java was selected as the baseline programming language. The team will be able to utilize time to iterate on the program rather than learning a new language for a few weeks. However, a trade-off is that Biogears and the software engineers utilize C++. A NLP program that is programmed in Java would need to acquire Biogears data externally from a file rather than receiving the information from the Biogears program that is programmed in C++. Java has backwards compatibility making our program more robust in future use to be translatable in another programming language. The issue with translating Java to C++ is the program will work behavioral but the written code is at risk for low readability for human readers. The effective use of time for team feasibility on the NLP project was ranked more important than usability for the UW CREST's software engineers use. As a result, the team moved on with Java.

Features	Eclipse	BlueJ	jGRASP
GitHub Access	5	5	0
Debugging Interface	4	3	5
Source Code Integration	5	0	0
XML/Maven Integration	5	4	1
Plug-in Availability	5	4	1
Team Feasibility	3	3	5
Score	27	19	12

Table 5: *Summary of criteria for choosing which IDE to use. The scale is 0 – 5 where 0 represents does not meet criteria and 5 exceeds standard.*

We then selected Eclipse as our integrated development environment (IDE). Due to the team's budget constraints, the team decided to research free IDE's. The three IDE considered were Eclipse, BlueJ, and jGRASP. While the three were relatively similar, Eclipse was selected due to a few unique features that would make the program more available for large-scale usage: built in testing, built in debugging, source code generation, and large library of plug-ins.

Finally, the team in pairs brainstormed two ideas to meet the project's goal of providing a response to questions based on live physiology. The first idea was to use a patient constructor and two QnA Bots. The patient constructor queries and converts Biogears' physiology data to a health status (Healthy, Poor, Adequate, Unhealthy) and symptoms to be used in the first QnA Bot. The first QnA bot directs the question to the matching answers that contain a generic answer skeleton and instruction to query Biogears of the relevant variables such as heart rate from patient constructor. Then, the second QnA bot takes in the earlier question skeleton with the value of the queried relevant patient data and returns to the client the appropriate response.

The second idea was to make a knowledge base for each disease and have the program use one QnA Bot and direct it to the correct knowledge base depending on the patient scenario. The knowledge base will have the same questions, but the answers will reflect the current patient scenario. If the answer is dependent on severity level, then the answer will redirect to code that retrieves the information for the finished response.

	Idea 1	**Idea 2**
Efficiency	Low - $O(n^2)$	High - O(n)
Usability	Low - Requires vague question skeleton and answers; complex quantification and conversion of patient state to levels (Poor, Healthy, Unhealthy)	High - Requires a straightforward set of questions and answers based on patient scenario
Integratible (with Biogears)	Accounts for multiple scenarios	Accounts for a single scenario

Table 6: *Summary of criteria for choosing which programming behavior to implement.*

The specification of the NLP program requires the program to create a verbal response in under 3 seconds to be efficient and be able to easily add response content depending on the simulation scenario or physiology of the patient. The benefit of Idea 1 is that it is able to create a response based on any simulation scenario using two QnA Bots while Idea 2 cannot. However, that high flexibility comes with a cost of efficiency since the time it takes to query a question for an answer using two QnA Bots from the Microsoft Cognitive Service cloud server doubles. Idea 2 considers the use of one QnA bot to combat efficiency issues. Another efficiency concern is raised when scaling the NLP program to contain more than thousands of responses to cover all scenarios and physiology states. By storing all possible questions and response pairs in a single knowledge base for a single QnA bot with responses that may not even be relevant to the patient's health condition, creates an efficiency issue when the program scans for a matching question and answer. On the other hand, Idea 2 helps reduce the number of possible responses the program needs to

scan through by creating a knowledge base for each scenario subsets and utilizing one QnA bot. To further quantize this, a computer science standard of comparing efficiency called the O(n) time complexity was estimated. Idea 1 would have an $O(n^2)$ while Idea 2 would have an O(n) making the latter more efficient. In addition, question-and-answer content becomes more intuitive for the programmer in Idea 2 because they will only need to create another knowledge base with all the questions and answers for the new scenario. Meanwhile, Idea 1 will require the programmer to write Mad Libs type of questions and complex quantification and conversion of patient state to levels. Under the criteria of program efficiency, usability, and integrability, Idea 2 was chosen.

Since the focus of the NLP program is on neuropathology, patients often experience slurred speech. The team decided to add a filter or word manipulator that alters the wav file or any other method that can alter the speech. Furthermore, the team decided to focus on traumatic brain injuries (TBIs) at three locations: cerebellum, parietal lobe, and occipital lobe. By focusing on these three neuropathologies, the feasibility of a quality project completion within the timeline increases. With that, the preliminary program has been decided and mapped out on the high-level block diagram as shown in Figure 1.

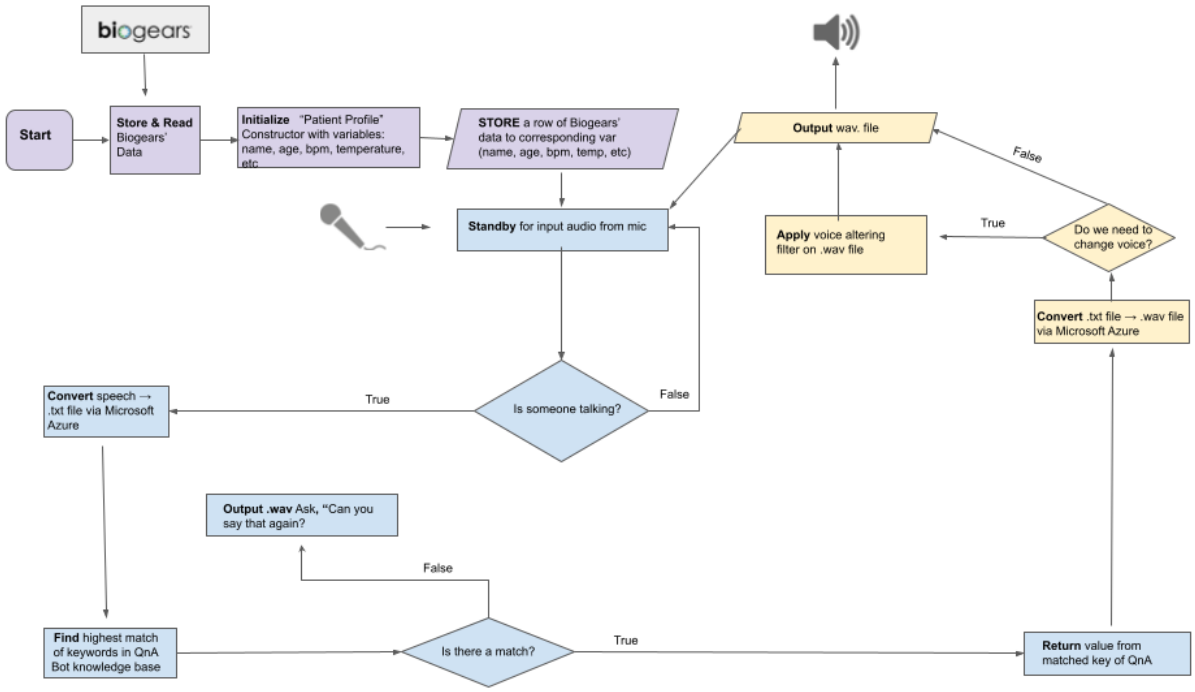

Figure 1: *Block diagram on the initial natural language program.*

Revised Design 1: Dynamic/Live Questions

Along with the NLP program, the content of questions and answers are just as important. The team investigated through reputable sources for questions asked in neurology examinations and symptoms for the three TBIs. Through research, the team learned that neurology examination consists of three main components: mental status, motor function/balance, and sensory status (31).

The types of assessment posed an issue with the preliminary program design. The first issue is that mental and sensory assessment questions are not static. For instance, asking someone to remember a list of items, asking someone to do basic math, or whether they can see something. The second issue is that the motor assessment involves the patient to move specific body parts. The current manikin does not have any mobile features. As a result, the team brainstormed some changes to the preliminary program that can address these issues.

The solution for the first issue is to add onto the program to check whether the question involves a live/dynamic task before going into the QnA Bot. If the question does involve a dynamic task, then it will skip retrieving a response from the QnA Bot and checks which appropriate function to create a response based on the question containing the dynamic task. For instance, a live/dynamic task can be counted down by sevens from number X. The number X can be any number which makes it dynamic. The program will check if the question contains words that match the words in the list of dynamic tasks. If there is a much, it will go into the function that counts down from sevens with the given starting number X.

Since the team won't be able to make all possible live/dynamic tasks yet, setting up code for future coders to easily implement their own function for these dynamic tasks becomes important. The team decided to code a function checker with an if-else structure to allow coders to add their function as an if-else statement. The drawback of this convenient structure is it will involve numerous if checks and may slow down the program in the case of high volumes of functions implementation. It also risks the potential of coders misunderstanding the code and which will have to be evaluated by user studies. Clear documentation and in-code examples were used to mitigate any misunderstandings.

Each team then researched the most frequent tasks asked during a neurology exam to be made into functions for the program. The first function is the calculate function, which can calculate any basic arithmetic and return a response. The second function is the memory function, which answers any question that asks about remembering a list of items. The third function is serial sevens, which can return a String of numbers counting down by sevens at any starting number. All these functions are able to provide responses under different TBI severity levels. At the time, the team used arbitrary 1-10 severity levels to be based on cerebral pressure values for the initial program.

The solution for the second issue is to add onto the program a second speaker acting as a nurse in the room and explaining any observations happening to the patient or any patient information. For example, if the user asks the patient to move their arm, then the program will respond in a non-patient voice, “The patient is able to move their arm.” Furthermore, having a nurse as second speaker helps with the accuracy or more specifically the authenticity of the NLP program since often doctors may ask the nurse about information on the patient. The drawback is that for any sensory and/or motor function-related question, then the patient will not be the one responding which reduces the realism of the simulation. The benefits outweigh the drawback since information on the ability to motor and sensory tasks is more than half the information needed for the neurological exam and for assessing the GCS. If the patient was the one reciting their actions, this would sound unnatural.

After confirming the solutions with Dr.Hananel and Mr.Gong, the team added new live/dynamic function questions to each of the knowledge base, motor and sensory related questions and answers, and an additional nurse speaker. A high level of the revised program behavior is shown in Figure 2.

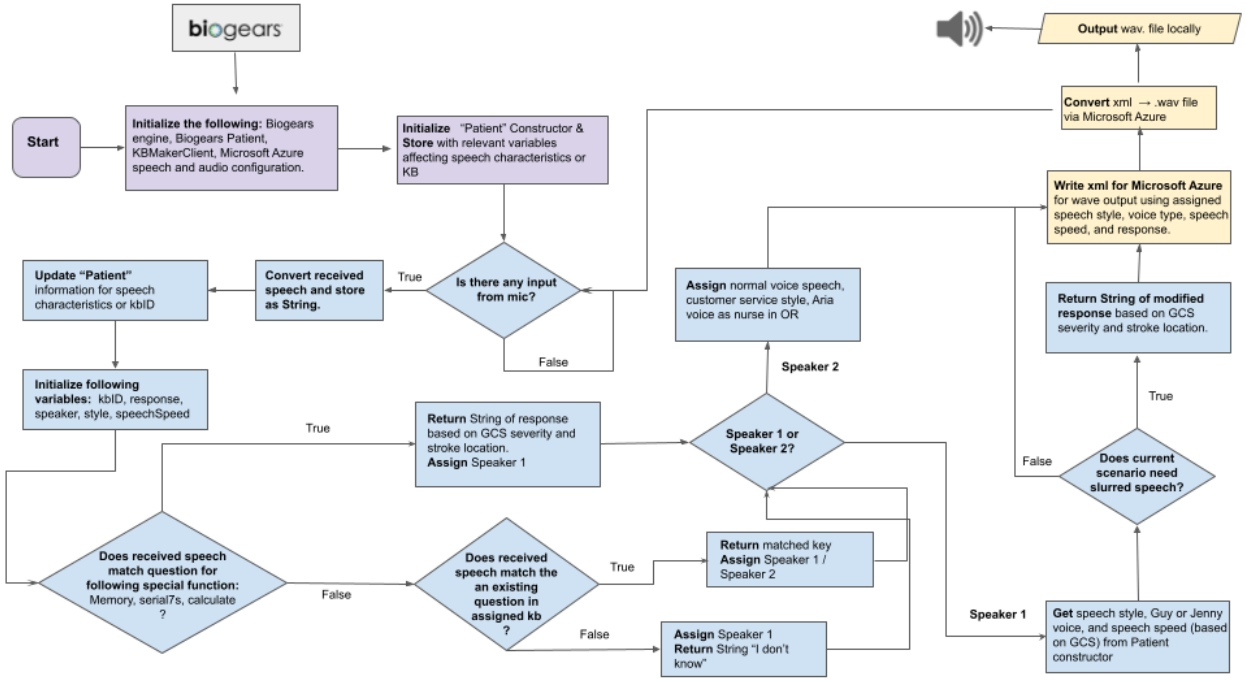

Figure 2: *Block diagram of revised natural language processing program.*

Revised Design 2: Usability Program and Content

The usability of a program becomes an important aspect for our broad target audience of novice programmers to software engineers. Usability of the program will encompass the principle for a design that is easy to understand, regardless of the user's experience and knowledge. Since the target audience includes college student users who are at least novice programmers and the team matches the background of the target audience, the team decided to record any difficulties the team experienced regarding the program and brainstormed ideas for other helpful features near the final stages of our program. The difficulties the team experienced and new program features are recorded in Table 7. Based on the team's feedback and suggestions, the team created three additional materials and program changes that can increase usability: detailed block diagrams, getting started guide, user-interactive Microsoft Azure knowledge base maker, and user-interactive main program instructions.

Difficulty	Solution
Describing code/function behavior through showing code in a java file to mentors and team members was an ineffective method to communicate.	A detailed block diagram, commonly used in programming, of all the functions and main program added an optional download.
Team members experienced difficulty in using Microsoft Azure's code to make a new knowledge base, changing content in an existing knowledge base, examining what knowledge bases existed, and other knowledge base functionalities. It required utilizing multiple methods to complete a desired task such as adding questions and answers to the knowledge base required the user to manually type each question themselves within the method.	An additional program will be made with a console user interface to alter the knowledge base without having to edit the code. The user will only need to follow instructions on screen to add, delete, and change a knowledge base as well as downloading content within the knowledge base.
Team members experienced some difficulty with acquiring Microsoft Azure subscription. The team also wanted to ensure understanding for users to add new dynamic functions and for completely novice users to be able to run the NLP program on Eclipse.	A "Getting Started" document has guided steps with screenshots on the following subjects: Running the NLP, alter Microsoft Azure subscription, knowledge base alteration, and adding a new dynamic function.
Team members experienced some inconvenience having to find the lines in the java file to change the file path.	For any java files requiring file path specification, the program will now have user-interactive console instructions for entering the file path.

Table 7: *Summary of difficulties the team experienced with the NLP program and their solutions.*

While designing these solutions, the team devised some risk management strategies. For creating the user-interactive knowledge base maker, the team considered the risk of the user accidentally deleting a knowledge base. The team implemented a default setting where when the user deleted a knowledge base, then it will also download the knowledge base onto the computer before it is deleted. Another risk is the user might enter the incorrect format of the file path and result in the automatic halt of the program. To address this risk, the team implemented code to check the user-input is valid and if it is not valid, then the user will need to try again until it is correct. Some other risk management is ensuring all users from different levels of programming backgrounds can use the program and preventing confusion between the instructions and code references for the Getting Started Guide. The risk was addressed by assuming the user has minimal coding experience and using a different font and highlight color for the code references. A possible drawback of curating the Getting Started Guide for novice programmers is it will include information the intermediate programmers will not need. To address this, the instructions will be sectioned with header informing whether the instruction is more pertinent to the novice programmer.

Revised Design 3: Neurologist Feedback + Live/Dynamic Function Changes

The team evaluated the accuracy of the program and its contents with neurologists. The first session was to receive feedback on the knowledge base content and functions. After this session with the neurologist Dr. Srinivasan, the team concluded the program needed to be changed as summarized in Table 8.

Previous Design	New Design
Project focuses on TBI with a specific location (parietal, cerebellum, occipital).	Project focuses on both TBI without a specific location and stroke affecting the brain's parietal lobe, cerebellum, or occipital lobe.
Use of an arbitrary scale of 1-10 that is based on cerebral pressure values for evaluating TBI severity for the NLP program.	Use of GCS for evaluating TBI and stroke severity for the NLP program. TBI or stroke patients with GCS of eight or seven may have a response with indistinguishable sounds. A GCS of six or below will produce no response.
Serial Sevens function returns a response with higher frequency in error and pause phrases, a higher deviation in correct response, and less counted numbers for higher severity in TBI.	Serial Sevens function returns the same as previous design behavior but also includes a higher frequency of randomized words for high severity in TBI/stroke.
Stutter function returns a response with random	Stutter function returns the same as previous design behavior but also includes a higher frequency of groan- and groggy-like sounds.

Table 8: *Summary of neurologist feedback and alterations to program.*

The team learned that the exact location of a TBI is not significant during a diagnosis but the location of brain lesions during a stroke is significant. As a result, the project shifts the previous TBI question-and-answer contents affecting the parietal lobe, cerebellum, and occipital lobe to the stroke question-and-answer contents affecting the same brain region. This helps maintain accuracy of medical content in the program and broadens the program content to both stroke and TBI. The team learned that both neuropathy use Glasgow Coma Scale (GCS), which is a reliable clinical scale to measure a person's consciousness after a brain injury or acute medical trauma, to determine the severity of the neuropathy. The GCS evaluates a person's motor, verbal, and eye response.

The live/dynamic function (Serial Sevens, Calculate, Memory) will need to substitute the previous 1-10 scale with the GCS (3-15). Furthermore, the neurologist explained that

his experience with patients below or at a GCS of 8 means that the patient is less likely to respond, and a GCS score of 6 or lower should elicit no response. The previous code did not account for this behavior and was added onto the revised design. Finally, the neurologist provided insight that often patients with lower GCS may respond with an answer unrelated to the questions or random words. The neurologist also emphasized that patients with lower GCS have a response of slowed speech and indistinguishable sounds. This prompted the team to alter the live/dynamic functions and the stuttering function to add random words, additional groans, and indistinguishable sounds as GCS score lowers.

Final Design: Student Usability

The team evaluated the usability of the program by conducting a student usability test. After conducting the student usability test, the team added the following items needed to be changed: explicit examples for any requests for user input in both the Getting Started Guide and in the console, ensure the figures in the Getting Started Guide circles/highlights where the user should look at and the figures are on the same page of the sentence the figure is referred, and increased clarity and behavior information in all Java files. The rationale for these changes are as follows in the respective order: common confusion whether to add "C://" in file path or whether the naming a file is requesting the specific file path or the file name; some confusion on where to look look in the screenshot of the Eclipse screen; and some confusion with the inconsistent naming of the same variables and program function.

Design Description

The natural language processing software was built according to the simplistic block diagram depicted in Figure 3. First, diagnostic information is extracted from the AMM's physiology-simulating program. This information includes patient information, the pathology, the time-changing severity, and the corresponding physiological parameters.

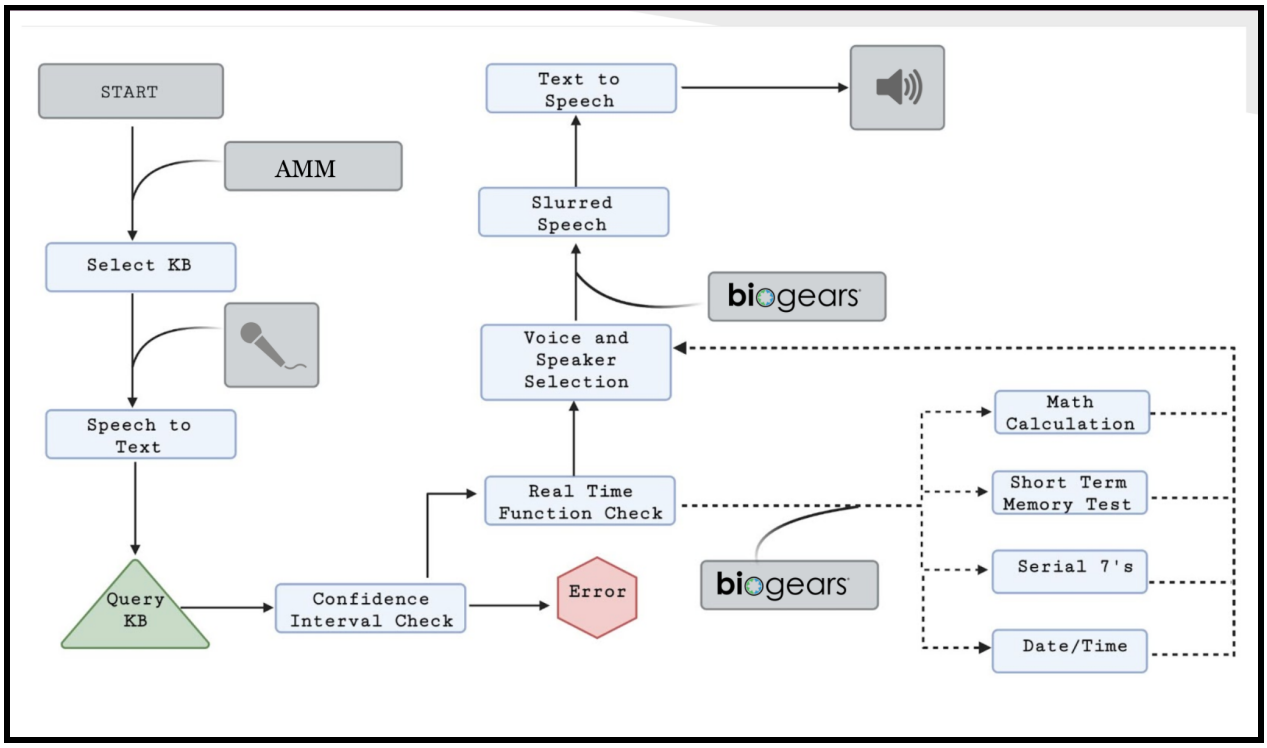

Figure 3: *Block diagram of final natural language processing program.*

However, the neuropathy scenario encoded by the AMM did not provide any information regarding the relative location of the simulated neuropathy within the brain. Since the exhibited symptoms of a patient with a neuropathy can be highly dependent on the location of the injury, an effective rendering of some neuropathy simulations will require location information as an input. Although we later noted that TBI was not one of these location-dependent pathologies, we did complete the foundational coding for an advanced cerebral circuit model depicted in Figure 4. This advanced cerebral circuit model divides the brain into sections following the Circle of Willis vasculature. Injuries can now be localized to the anterior, medial, or posterior vascular territories on either the left or right side of the brain.

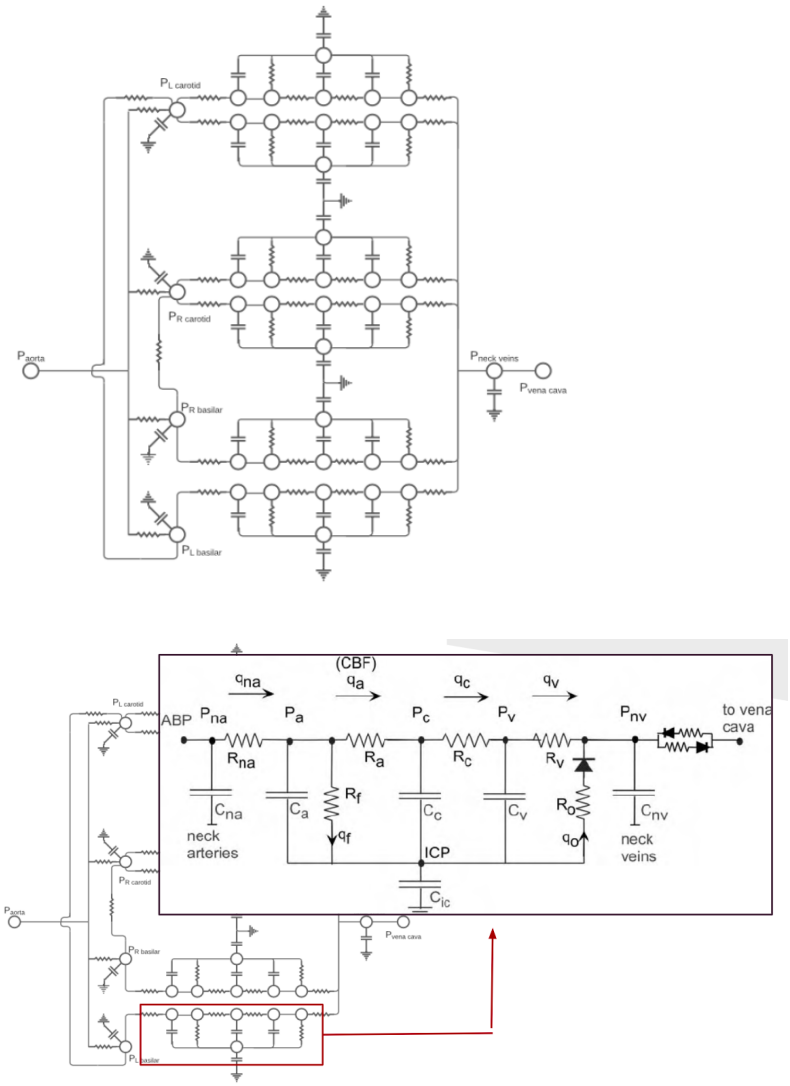

Figure 4. *Advanced cerebral circuit design. ICP = intracranial pressure; CBF = cerebral blood flow; ABP = arterial blood pressure; R = resistance; C= compliance; P=pressure; a= arterioles; c = capillaries; v=veins; nv = neck veins; na = neck arteries; q = blood flow.*

After our developed software stores all of the diagnostic information from AMM, it selects the knowledge base that corresponds to that particular pathology. The knowledge base is a set of previously encoded questions that clinicians might ask and the corresponding simulated answers. Since the simulated answers of patients will be so diverse depending on the pathology, a KB will have to be created for each pathology.

Once the knowledge base has been selected, auditory input from the environment can be received by the program. Auditory input is delivered to the DDS by the sound module, a modality within the AMM that controls both the microphone and the speaker. Our natural language processing software will continuously extract all incoming audio from the DDS backbone.

Our program then delivers the audio input to Microsoft Azure's open source speech to text program in order to convert the audio file into text (Appendix 1.1). Microsoft Azure is an open-source cloud computing service that provides a lot of the functional groundwork our software uses to generate natural language processing in the AMM.

The text file returned from the speech to text program will then be run against the selected knowledge base to look for a match. The search algorithm incorporates all alterations in syntax and synonyms before selecting the highest match. This intent recognition allows our software to encompass some of the enormous variability in natural speech. However, to ensure a true match was found, the search algorithm also returns a confidence score variable whose value ranges from 0-100 that specifies how confident the program is that it has identified a match. If the confidence score is below 50, our program will return an error and ask the user to try a different question. This function ensures that our program will not return an answer to a question it was not designed to answer, improving the overall medical accuracy of our program.

If the match is greater than 50, the program then checks to see if any real-time functions are needed to render the simulated response. These real-time functions are a large reason why neuropathy simulations are uncommon in other NLP programs. The number of possible ways these questions can be asked is infinite rendering the previously-stored information model useless. However, we advanced this model in such a way that allows our program to address these infinite possibility questions. The questions and answers are still previously stored in the knowledge base. However, if the answer to an encoded question requires the use of a real-time function, the text output was encoded as a highly specific coding key that specifies which function file is needed to render the simulated answer. Our program checks to see if this key is present in the text output. If the real-time coding key is present within the text variable, the program will deliver the text as an input to the specified function function. Using the intent recognition capabilities within

the knowledge base search algorithm, our program can now answer an unlimited number of similar questions using just that one knowledge base question and the associated function files. We encoded 4 real-time functions.

First, we created a mathematical calculation function. It is common in neurological assessments to ask the patient to perform simple calculations. This function is able to return the correct answer for any non-algebraic mathematical question asked of our program by interfacing with Google search. The correctness of the response is dependent on the severity of the pathology as measured by GCS. As the severity of the pathology increases, the patient's response further deviates from the correct answer. To accomplish this, the correct answer was multiplied by a specific decimal value ranging from 0.2 to 0.5 depending on the severity. Lower severities were multiplied by 0.2 while the highest severities were multiplied by 0.5. These values were then added and subtracted from the correct answer to generate a wide range of acceptable values. This range was then used to create a Gaussian bell-curve from which values would be randomly extracted. This randomization greatly increased the fidelity of the program.

Next, we created a short term memory function file. This function allows physicians to test the short term memory of patients by asking them to remember and repeat a string of words. The function stores strings of text when prompted. When asked to deliver these stored variables, additional randomization and severity-dependent processes are then applied before delivering the final output. As the severity of GCS increases, the patient responds with more 'uh' and 'umm'. Additionally, if a patient had a GCS below 14, the patient will respond with one or two fewer remembered words, depending on a randomly generated number. If the GCS is below a 12, the patient will respond either with "I don't know" or with a list of random words, depending on a randomly generated number.

Next, we created a serial severn function. This function accounts for a common question of physicians that tests a patient's mathematical reasoning. Patients are asked to count down from a number by sevens. The serial sevens function gave an appropriate response based on the GCS score and the number which the patient was asked to start at. At a GCS lower than 14 and greater than 9, the function will cause a random error from an increasingly large range, beginning at a range of 20 for GCS 10 and a range of 20 for GCS 9.

And finally, we created several real time functions that returned the patient's perceived data, time, and day of week depending on GCS severity. Healthy patients return the accurate time and date. Similarly to the mathematical calculation function, the range from which a random date or time is chosen increases with increasing GCS severity.

After the textual content of the response has been generated utilizing the knowledge base and real time functions, slurring is applied depending on the severity of the GCS score. With increasingly severe GCS score, a random number from a wider range of pauses and "uhs" will be applied in spaces. Additionally, a random number of consonants from a wider range will be replaced with "r" or an "s".

The text file is then converted into speech using Microsoft Azure's text-to-speech function and played out on a speaker thus enabling successful speech capabilities within the manikin (Appendix 1.2).

Design considerations to increase the usability of our program for both the medical personel end users and the programmers/developers were also included in the final design.

For medical personnel users not familiar with coding, we created a basic console user interface that allows users to view, edit, add, or delete knowledge bases without knowledge of the code or of Microsoft Azure. Additionally, we created a basic console interface for interaction with the program that updates users on the processes being performed, such as speech recognition, knowledgebase querying, speech generation, and audiofile saving and distribution. We also included extensive documentation to visually walk these users through the downloading and editing process including altering microsoft azure credentials.

For the users that are interested in further developing our program or incorporating it into their own projects, we thoroughly commented all of our code. We also included a 'ReadMe' function file that gets downloaded with the program. This file is completely commented and provides further explanations on the embedded functions within our program and on possible integration and development applications. A PDF file of both a detailed and simplified version of a block diagram has also been uploaded to GitHub for anyone's desired use. Additionally, the code is designed so that any probable change that a programmer will make, such as Microsoft Azure credentials, can be implemented by changing one or two lines of code.

Test data/ Engineering Analysis

The developed program was assessed in three areas: fidelity, accuracy, and usability.

I.a Fidelity - Content

Our program is designed to educate and train medical students or personnel on how to recognize and diagnose specific pathologies. If our program generates a simulated answer, that response must be consistent with the current medical/scientific knowledge surrounding that pathology in order to avoid misinforming large groups of medical staff and risking the safety of patients cared for by the improperly trained staff. Fidelity was therefore a critical aspect of our experimental designs. Fidelity was defined as the scientific correctness of our simulated responses according to a specialized neurologist, Dr. Vasisht Srinivasan from the UW medical center. The task assignment used can be found in Appendix 2.1. He is still in the process of completing this study, but the results of this study will be uploaded to the final GitHub destination along with the program and all of the documentation. His age and number of years of clinical experience in neurology will be recorded.

Dr. Srinivasan also helped guide the initial creation of all QnA content. Upon his suggestion, we incorporated randomization into all responses that have not been previously encoded and are expected to be generated live. The personalities and capabilities of each individual will vary drastically across the population. The randomization of the outputted simulation responses should increase the fidelity rating of our program by encompassing some of these natural population variations.

After giving informed consent, Dr. Srinivasan was asked to interact with our program during three simulations: traumatic brain injury (severity of 11), left parietal stroke, and left occipital stroke. He was asked to score the fidelity of our program on a scale of 1-10 according to his experience as a neurologist diagnosing these pathologies with 1 being the worst and 10 being the best. He was also asked to provide additional comments explaining his given score so that future developers can make the necessary alterations. A given score above 9 will be viewed as successful. This value was selected based on a study reviewing

the realism of a previously established medical manikin compared to standardized patients. Realism ratings of the manikin were between 1.5 to 2.5 out of 5, and the ratings of the standardized patients was approximately 4.7 (28). Extrapolating this to a scoring system out of 10, a realism score of 9 would indicate that our manikin rivals standardized patients and could thus be considered successful.

Our proposed study to assess program fidelity is limited by Dr. Srinivasan's time allotted to participate. While Dr. Srinivasan's input is extremely valuable, the fidelity of each simulated neuropathy should be evaluated by more neurologists prior to public release. A group of 80 neurologists should be asked to individually interact with the program for each simulated neuropathy (30). The individual portion avoids the possibility of groupthink or social loafing that can be present in group settings. After completing the individual assessment, the neurologists should then be gathered together to discuss the feedback from the individual surveys. Each point made by a neurologist should be discussed as a group and voted on prior to implementation into the program. The characteristics of the group of neurologists used for evaluation should be documented and include the mean, standard deviation, and range of their age and years of clinical experience in neurology.

However, the extent of resources and networking opportunities that this experiment would require was outside of the limitations of our project. It should be noted that only specialized neurologists and not general physicians should be used for this experimental study. The goal of this program is to reduce the misdiagnosis of these neuropathies due to the failure of general physicians to properly perform a neurological assessment of this degree. General physicians should therefore not be utilized to determine program fidelity.

I.b Fidelity- Response Time

Outside of the simulated voice characteristics, one of the largest impacts on the 'realistic' feeling of an NLP simulator is the time it takes the patient to respond. The response time was defined as the time it takes the program to begin playing the generated audio file on the speaker after the last word in the verbal prompt was completed. A timer was created within the program temporarily for more accurate measurements. This experiment was repeated three times using 20 verbatim questions. The average response time was calculated according to the following equation :

$$t = \sum T / N$$

t = Average time per response
T = time for response
N = 20 questions

The results of this study are depicted in Table 9.

	Trial 1	Trial 2	Trial 3	Overall
Average Response Time (s)	2.732	2.854	2.814	2.800

Table 9*. Average response time of program during verbal interactions. CRP = correct response percentage; KB = knowledge base.*

A response time below 2 seconds was considered to be successful. The response time of the program is dependent on various factors outside of the program's coding efficiency like the type of the computer processor or the internet speed since our program does require internet access to function. An analysis of various NLP programs under different access networks showed that even on the slowest network, the NLP response time of all tested programs was just above 2 seconds (27). To accommodate the various situations this program will be run in, we set the goal for success at this lower response time. The average response time of 2.8 seconds does not fit into the range which we considered successful, but it met the acceptable response time of 3 seconds.

II. Accuracy

The fidelity of our program depends on the program's accuracy. The accuracy of our program was defined as our program's ability to return the correct expected response. Encoding scientifically correct information is not beneficial if the program is not able to output that information when prompted.

For this study, the program was prompted with 3 different types of questions: verbatim, synonymous, and random. Verbatim questions were delivered verbally exactly as they are encoded within the KB. This group serves as a control group. By asking the program verbatim questions, it assesses our program's baseline functionality: its ability to take auditory stimulus as an input and return the expected output according to the KB. If questions from this category do not return an accurate answer, there is a problem with how the program is processing information in general.

Synonymous questions were based upon a question within the KB, but were verbally asked with differing syntax or synonyms. This group assesses the program's ability to recognize the intent of the question being asked. Natural language processing requires strong intent recognition capabilities since phrasing will vary significantly across users.

Random questions are completely random questions not contained within the KB at all. The program should return an error to this question and ask the user to try again. If the program provides answers that were not specifically designed for the question being asked, this might confuse users and ultimately train them to perform ineffective neurological examinations. A type II error is therefore much more critical for this function than a type I error.

The program was prompted with a total of 20 questions from each category by each team member. The correct response percentage will be calculated according to the following equation :

$$\text{CRP \%} = C/N \times 100\%$$

C = Number of Correct Response
N = 20 questions

Each team member performed this testing experiment as seen in Table 10. By having each team member repeat this study, we are additionally testing the program's ability to understand various voice types, inflections, pitches, etc. ANOVA statistical analysis was used to compare the results of each team member and ensure that the program does not possess any internal bias towards a particular group (significance value of 0.05 used). While further statistics were not gathered on this topic specifically, the program's ability to

recognize other's voices was closely monitored during the live zoom session test for usability as described later.

	Verbatim KB	**Synonymous KB Questions**	**Non-KB Questions**	**Overall**
Average CRP *Team Member 1*	95%	90%	85%	90%
Average CRP *Team Member 2*	100%	90%	95%	95%
Average CRP *Team Member 3*	95%	95%	100%	97%
Average CRP *Team Member 4*	100%	85%	95%	93%
Anova Test *F-value*	0.6666667	0.35681	1.35206	

Table 10*. Average correct response percentage of program during verbal interactions. CRP = correct response percentage; KB = knowledge base.*

A cut-off CRP% value of 95% was used to determine if the program was accurate. This value was taken from a study that used natural language processing algorithms to classify PubMed abstracts. The algorithm was found to be overall 95% accurate (29). Therefore, we should strive for the same level of accuracy within our own program as those standards already established within the medical community. As observed in Table 10, our program met the ideal accuracy design goals for all categories and did not exhibit any bias toward an individual's voice.

III.a. Usability - Student Usability Study

The goal of evaluating the usability is to 1) verify whether the NLP programs can be understood by future students and 2) to receive feedback for improvements. The student usability study was conducted over Zoom. Participants were given a Google Drive with a zip file containing the NLP program, participants instruction document, and Getting Started

Guide. Before the study, all participants will be expected to have a laptop or computer with Eclipse and given consent for recording the study over Zoom. In this usability study, participants will be given four tasks to complete with minimal or no assistance from the team. The participants are allowed to ask for help on using Eclipse since the goal is not to assess their skills on using Eclipse. During the usability study, the participants will be expected to verbalize their thoughts as they proceed through the tasks. After the participants finish the tasks, then the participants will fill in a feedback and ratings form and will be surveyed by the moderator. Questions are summarized in Table 11. The moderators will ask open-ended questions to further understand the rationale of the participant's action. In addition, the moderator will ask about their background in programming in Java and using Eclipse to see if it impacted the participant's ability to complete the tasks.

Category	**Questions**
Task Questions	1. Did you complete the task? (Y/N) 2. Rate from a scale of 1-10, how helpful were the documentation to [INSERT TASK]? 3. [SELECTION] Which documentation was the most helpful to you (if any)? 4. If you chose any, can you explain why? 5. [SELECTION] Which documentation was the least helpful to you (if any)? 6. If you chose any, can you explain why? 7. Are there any improvements or anything that would help you complete the task?
Post-Task	1. Was your ability to complete tasks affected by unfamiliarity with Eclipse? (Y/N) 2. Overall, what were some issues/problems you faced as you completed the tasks (if any) 3. Overall, what about the program/documentation helped with the task (if any).

Table 11: *General feedback questions to be asked to participants.*

The usability study will be done as described in this transcript draft below. The participants will be acting as a new intern in the software engineer department of the UW CREST. The team lead has given the participant a few tasks for his/her/their first week. The tasks are described in Table 12. These are tasks future students or engineers will need to further refine the program and build for other scenarios. The control will be two team

members doing the same problem. The approach of the participants and team members will be compared.

Task Type	**Task**
Set-Up	1. Run the program without issues/errors.. 2. Ask “What is the date?” and receive an output file on the computer with an audio file saying, “May 14” in a male voice.
Knowledge Base - Content Addition	1. Add a new knowledge base called “Sepsis Patient” using the provided excel file called “Sepsis_Patient.xlsx” containing questions and answers 2. Run the program without issues/errors.. 3. Ask “Does the patient have a fever?” and receive an output file on the computer with an audio file saying “Yes the patient does” in a female voice as a nurse.
Function Integration + Biogears Query	1. Add a new function called SerialSevens.java into the program. 2. Run the program 3. Ask “Can you count down from 30 by sevens?” and receive an output file on the computer with an audio file saying “23. 16. 9.” in a male voice.

Table 12: *A summary of participant tasks and purpose.*

All students were successfully able to complete all three tasks within the required timeframe. All four students had prior CSE background with a mixture of Java and programming proficiency; however all participants also noted that they had no prior exposure to Eclipse. We anticipated that the lack of Eclipse exposure would have impacted

how fast students were able to complete the study, however we were confident that our instructions were clear enough to overcome this.

Task	Task 1: Running NLP	Task 2: Add Knowledge Base	Task 3: Add New Function
Average Score	8.5	8.75	9.25

Table 13: Average of helpfulness score for the student usability study

Following the conclusion of the first task (running NLP program), students were asked to rate which document was the most useful, students rated the helpfulness of our getting started document an average 8.5 on a ten point scale. For the second task (adding a new knowledge base), students found that our getting started document was the most helpful, rating it on average 8.75 on a ten point scale. For the third task (adding a new function), students found that getting started was the most helpful, rating it an average 9.25 on a ten point scale.

Ethical Considerations

Although manikins have been developed for trauma head injuries, these manikins are limited in their ability to simulate internal cerebral hemorrhage, and currently no manikin has been developed to simulate neuropathological diseases - a gap our project intends to fill. For our project, in particular, we are developing a natural speech processing program to simulate a conversation with neurology patients through a multi-layered integration with the CREST Advanced Modular Manikin (AMM). Language barriers in healthcare lead to miscommunication between medical professionals and patients while reducing customer satisfaction(32). As such we need to be aware of language barriers associated with trainees who are either immigrants or from different cultural backgrounds in order to make sure that our manikin integration is able to understanding the input from the user regardless of one's enunciations or pronunciation. Given that our project aims to decrease the number of misdiagnoses by serving as a training tool for future medical professionals, an ethical concern is who is responsible for the harm to a patient from a misdiagnosis. If a medical professional receives half of their clinical hours from simulation,

a concern is if the medical professional misdiagnosed as a result of learning in the simulation program. In other words, the program was not able to retrieve the correct response for a specific simulation. This means when the speech program outputs incorrect information for a disease and the medical professional learns from this. It is important when releasing the product to meticulously verify the program and have a secondary opinion such as an instructor during these simulations.

Global and Societal Impact

Manikins are drastically revolutionizing medical training, while reducing the adverse medical errors associated with medical students. Currently, changes in work practices, reduction in hours, growing concern for public safety and health scrutiny, in addition to the rise in technically demanding procedures have increased the challenges to both acquire and maintain skills(33). Over the years, there has been a growing number of medical malpractices claims and errors within the United States and around the world. A 2007 malpractice case by Neurosurgeons at Rhode Island Hospital resulted in three intra-cranial neurosurgical mistakes; while two of the mistakes were caught early enough to correct the surgical mistake, the third case resulted in the death of an 86-year-old patient(34). This malpractice case resulted in the hospital being fined $50,000, reprimanded by the Department of Health, and the surgeons license suspended for only two months. While these errors could be attributed to any number of factors, one of the most obvious is the lack of adequate physician training with medical students expressing a need for more training and education (35). The development of advanced modular manikins will change the way that medical care workers are trained and taught. Being able to use simulations drastically lowers the cost of medical care workers' education as it can complement, and maybe one day replace the use of cadavers, can better prepare professionals for when they see live patients. A modular manikin will provide medical professionals with realistic and high quality training, raising the performance standards of these professionals which can, in turn, lower the likelihood of malpractice. Reducing malpractice incidents will both lower the risk of medical error to patients while also decreasing the costs to hospitals associated with malpractice.

Regulatory Issues

A natural language processing program aimed at improving physician competence in performing neurological examinations does not have applicable FDA standards that the program must adhere to. However, our program does adhere to some suggested guidelines for software development and clinical simulations.

First, it is a standard in the software development industry to follow coding standards. Our program is code in Java and will follow the coding conventions provided by Oracle. The team plans to assess and maintain the usability of the program. A relevant standard is ISO/IEC TR 25060:2010 which provides an International Standard called the Common Industry Formats (CIF). The CIF documents the specification and evaluation of the usability of interactive systems (36).

Second, the International Nursing Association for Clinical Simulation and Learning (INACSL) has developed suggested guidelines and standards of best practice for the development of medical training simulation technologies. Our project has adhered as closely to these guidelines as possible while remaining within the limitations of our funding and restricted social interactions. While the criteria listed within these standards are quite extensive, some of the most relevant requirements are summarized below :

1. Perform a needs assessment and provide sufficient proof that the simulation is needed by medical educators (37). Our research demonstrating the critical need for our simulation program has included scholarly papers, medical lawsuit analyses, comparison to existing methods, and communication with neurologists.

2. Construct an approach that enables users to achieve specified outcomes with the target users anticipated baseline knowledge and skills (37). For our simulated program, we have two types of users : software developers who will continue to build off of our foundational natural language processing platform and medical educators and students. We have made great efforts to ensure that the program is clear and easily alterable by software engineers including commenting

all code, enabling new functions to be fully integrated by adding only a single line of code, condensing all functions into specified areas for easy access and organization. For the medical educators and students who might not have an extensive programming background, we have created a user interface that allows users to print content of QnA databases, alter or make new databases, and provide questions and feedback without ever looking at the inner workings of the program itself.

3. Provide debriefing or preparation materials (37). Upon opening our program, it will briefly describe the purpose of the program and direct you to a README file for more information if so desired. This README file contains summaries of the program mechanism, program dependencies, required program inputs, and 'how-to' files for making common program alterations. The user is also free to continue to use the program without reading the README file.

4. Pilot-test simulation before public release (37). We plan to perform extensive testing on the reliability and accuracy of the program ourselves. However, we also plan to perform usability studies on three groups of users : advanced UW CREST software engineers familiar with AMM and BioGears who plan to use our program to build their future projects, medical personnel who are interested in using our program to train physicians to perform neurological assessments, and UW engineering students who are completely new to the project, AMM, and BioGears. Since this program will be open-source and available to anyone, this last group will assess the ability of our program to be understood by new users.

5. Assess the current knowledge, capabilities, and skill of participants used for program evaluation (38). During our usability studies, we will be asking all participants to rate their comfortability with software engineering, programming, and medical diagnostics.

6. All participant evaluations must be kept confidential (39).

We also adhered to a few FDA standards for other medical training devices although these standards are not required for completion of our project. FDA standards for a cardiopulmonary resuscitation (CPR) aid with audio feedback under section 870.5210 mandates that the simulation evaluation methods must specify/describe the population of participants used to demonstrate simulation functionality (40). Although this standard doesn't specifically apply to our program, we feel that it is crucial to be transparent about the program evaluation process and promote diversity and inclusivity within our project. We will therefore include in our report and README file the disclaimer that this program was only tested on fluent English speakers with minimal accents. We will encourage future developers with more resources to evaluate the program across a broader population.

FDA standards for Electronic Records under section 11.10 for controls for closed systems requires that systems be validated to ensure accuracy, reliability and adequate controls over access to, and use of documentation for system (41). While our project does not require the use of electronic signature, we will be creating records, in the form of patient files, which our code will be modifying and transmitting electronically. As such we will work, in collaboration with the UW CREST team, to ensure system integrity is maintained especially as users/stakeholders interact with our program and their data is logged in the patient log.

Conclusions

Although the team did not have enough time to directly use Biogears data via the DDS, the latest version of the NLP program supports the feasibility of its implementation. The NLP program is able to respond to questions from the user live and its response is based on the given gender, the scenario name, who is speaking (patient or nurse), and if applicable the GCS and stroke location. In short, the program meets a satisfactory level as it is able to direct itself to the correct knowledge base, the live dynamic function is able to adjust the response to differing levels of stroke locations and GCS severity, and college-level students were able to easily get started with the NLP program without additional help.

Unfortunately, the team did not have enough time to evaluate the revised program on whether software engineers in UW CREST can easily build upon the program and

whether the revised content is accurate. A completed software engineer usability study would have provided insight into whether the intermediate programmers are able to easily understand the program's behavior or make significant changes to the program's behavior such as adding a new scenario to the program. In addition, the team was not able to schedule the second session of the content accuracy evaluation study with the neurologist Dr. Srinivasan before early June. Thus, the team's current revised stutter function and question-and-answer content is not confirmed for authenticity. Altogether, the program falls short on minimal verification on whether or not NLP program code structure can be easily iterated on and minimal verification on content accuracy on the team's current revised stutter function and question-and-answer content.

Overall, the team has greatly increased their programming experience in both Java and through the command prompt. Aside from hard skills, the team gained additional experience in communication of abstract ideas such as coding behavior and presentation skills from weekly mentor meetings, class presentations, and the bioengineering symposium. These soft skills will be valuable for explaining scientific and medical findings to future group projects or presentations to audiences with a broad range of backgrounds. Through conversation with Dr. Hananel, the team learned the values of building a network and connections to be able to conduct these interdisciplinary projects.

Future Works

The current NLP program is at the starting foundation of an open source program capable of simulating patient conversations and interactions based on live physiology data from Biogears. Up to this date, the NLP program has undergone initial content validation and student usability testing. In the future, future students can resume the final content validation with the neurologist and the usability study with the UW CREST software engineers. This final content validation will involve other medical personnel to listen to different simulations of at least one occurrence of the existing scenarios (TBI, stroke at the occipital lobe/cerebellum/parietal lobe, Appendix 2.2). The usability study with the software engineers at UW CREST is similar to the student usability study but includes creative tests that require in-depth understanding of the NLP program (Appendix 2.3). Both

protocols have already been written and ready to be conducted and located in the GitHub repository (Appendix 2.4) and the Appendix.

Some immediate modifications that will need to be made for the NLP program is the direct integration of Biogears variable instances rather than hard coded variables and the transfer of the audio file from the NLP program to the AMM manikin via DDS to be played onto the speakers of the AMM manikin. After these two modifications, testing whether the NLP program provides the expected output with the new changes using the same testing protocol for calculating the correct response frequency and whether AMM manikin is able to play the audio file from the NLP program.

In addition to the incomplete studies, the NLP program has many avenues for improvements and optimization. Some of the next steps to iterate on this project are to expand the current knowledge base of the existing scenarios with more questions-and-answer pairs, expand the knowledge base with new question-and-answer content of new scenarios, or add new dynamic/live functions. To gain and validate new content, future teams will need to conduct their own initial online research for common questions for the disease/scenario of interest and seek the appropriate medical professional to ask how they diagnose the disease/scenario of interest. To take the project to the next level, a student team could devise a program that can provide a patient response based on two or more scenarios simultaneously rather than a single scenario.

In order to create new question-and-answer content and validate it, future student teams or UW CREST will need to contact Dr. Srinivasan for any TBI related content or establish contact with the appropriate medical professional. If possible UW CREST can provide a list of contacts ready for students to contact or the students will need to establish at least three appropriate medical professionals as soon as they choose their project. If students are reaching out to specialty doctors, then students will need to make it a priority since their schedules are often booked to three to four weeks out. Finally, UW CREST will need to subscribe to a paid Microsoft Azure Cognitive services subscription to increase the available knowledge base from three to 10 or more. Funding can be either acquired from UW CREST or the team can apply grants such as Patient-Centered Outcomes Research Institute Grant.

References

(1) Centers for Disease Control and Prevention (2019). Surveillance Report of Traumatic Brain Injury-related Emergency Department Visits, Hospitalizations, and Deaths—United States, 2014. Centers for Disease Control and Prevention, U.S. Department of Health and Human Services.

(2) Goldberg, Charlie. (2018). Practical Guide to Clinical Medicine: A comprehensive physical examination and clinical education site for medical students and other health care professionals. UCSD School of Medicine.

(3) Dikmen, S., Machamer, J., & Temkin, N. (2017). Mild Traumatic Brain Injury: Longitudinal Study of Cognition, Functional Status, and Post-Traumatic Symptoms. Journal of neurotrauma, 34(8), 1524–1530. https://doi.org/10.1089/neu.2016.4618

(4) Wäljas, M., Iverson, G. L., Lange, R. T., Hakulinen, U., Dastidar, P., Huhtala, H., Liimatainen, S., Hartikainen, K., & Öhman, J. (2015). A prospective biopsychosocial study of the persistent postconcussion symptoms following mild traumatic brain injury. Journal of neurotrauma, 32(8), 534–547. https://doi.org/10.1089/neu.2014.3339

(5) Jacobs, B., Beems, T., Stulemeijer, M., van Vugt, A. B., van der Vliet, T. M., Borm, G. F., & Vos, P. E. (2010). Outcome prediction in mild traumatic brain injury: age and clinical variables are stronger predictors than CT abnormalities. Journal of neurotrauma, 27(4), 655–668.

(6) Powell JM, Ferraro JV, Dikmen SS, Temkin NR, Bell KR. Accuracy of mild traumatic brain injury diagnosis. Arch Phys Med Rehabil. 2008 Aug;89(8):1550-5. Doi: 10.1016/j.apmr.2007.12.035. Epub 2008 Jul 2. PMID: 18597735.

(7) Kowalski RG, Claassen J, Kreiter KT, et al. Initial Misdiagnosis and Outcome After Subarachnoid Hemorrhage. JAMA. 2004;291(7):866–869. doi:10.1001/jama.291.7.866

(8) Wojcik S. M. (2014). Predicting mild traumatic brain injury patients at risk of persistent symptoms in the Emergency Department. Brain injury, 28(4), 422–430. https://doi.org/10.3109/02699052.2014.884241

(9) Smith v. Baca, No. CL17-894 (Va. Cir. Ct. Augusta County).

(10) Chicoine v. DiBaro, No. 591/12 (N.Y. Sup. Ct. Nassau County, Feb. 6, 2019).

(11) Orr v. Bell, No. 58820 (N.Y. Sup. Ct. Warren County, Feb. 8, 2016).

(12) Kaufman, N.K., Bush, S.S. & Aguilar, M.R. What Attorneys and Factfinders Need to Know About Mild Traumatic Brain Injuries. Psychol. Inj. and Law 12, 91–112 (2019). https://doi.org/10.1007/s12207-019-09355-9

(13) Melnick ER, Szlezak CM, Bentley SK, Dziura JD, Kotlyar S, Post LA. CT overuse for mild traumatic brain injury. Jt Comm J Qual Patient Saf. 2012 Nov;38(11):483-9. Doi: 10.1016/s1553-7250(12)38064-1. PMID: 23173394.

(14) Health Simulations. (2020). Manikin. Retrieved from https://www.healthysimulation.com/manikin/

(15) Lateef F. (2010). Simulation-based learning: Just like the real thing. Journal of emergencies, trauma, and shock, 3(4), 348–352. https://doi.org/10.4103/0974-2700.70743

(16) Dewan MC, Rattani A, Gupta S, Baticulon RE, Hung YC, Punchak M, Agrawal A, Adeleye AO, Shrime MG, Rubiano AM, Rosenfeld JV, Park KB. Estimating the global incidence of traumatic brain injury. J Neurosurg. 2018 Apr 1:1-18. doi: 10.3171/2017.10.JNS17352. Epub ahead of print. PMID: 29701556.

(17) Verified Market Research (2018). GLOBAL HEALTHCARE/ MEDICAL SIMULATION: MARKET SIZE, STATUS AND FORECAST TO 2026.

(18) Advanced Modular Manikin. (2016). About AMM. retrieved from https://www.advancedmodularmanikin.com/about1.html

(19) Nasco Healthcare. (2021). ALEX: The First Patient Communication Simulator (PCS). Retrieved from https://ss-usa.s3.amazonaws.com/c/308477710/media/35905fa1b6ed8ffce75294703998562/nasco_sell_sheet_alex_Aug23_2020_trim.pdf

(20) Laerdal Medical. (2021). SimMan 3G PLUS

(21) Immersive training for emergency care procedures. Retrieved from https://laerdal.com/us/products/simulation-training/emergency-care-trauma/simman-3g

(22) Riek, L. D. (2016). U.S. Patent No. 9,280,147. Washington, DC: U.S. Patent and Trademark Office.

(23) Braksick SA, Kashani K, Hocker S. Neurology Education for Critical Care Fellows Using High-Fidelity Simulation. Neurocrit Care. 2017 Feb;26(1):96-102. doi: 10.1007/s12028-016-0293-3. PMID: 27389006.

(24) Eggert, J. S., Eggert, M. S., & Vallejo, P. (2003). U.S. Patent No. 6,503,087. Washington, DC: U.S. Patent and Trademark Office.

(25) Bennett, I. M. (2010). U.S. Patent No. 7,702,508. Washington, DC: U.S. Patent and Trademark Office.

(26) Ingenito, M., Ingenito, E. J., & Ingenito, M. P. (1990). U.S. Patent No. 4,932,879. Washington, DC: U.S. Patent and Trademark Office.

(27) Alexakis, George & Panagiotakis, Spyros & Fragkakis, Alexander & Markakis, Evangelos & Kostas, Vassilakis. (2019). Control of Smart Home Operations Using Natural Language Processing, Voice Recognition and IoT Technologies in a Multi-Tier Architecture. Designs. 3. 32. 10.3390/designs3030032.

(28) Alsaad, A. A., Davuluri, S., Bhide, V. Y., Lannen, A. M., & Maniaci, M. J. (2017). Assessing the performance and satisfaction of medical residents utilizing standardized patient versus mannequin-simulated training. Advances in medical education and practice, 8, 481–486. https://doi.org/10.2147/AMEP.S134235

(29) Krohn, J., Martin, A., Martin, C. (2016). Accuracy Of Natural Language Processing-Based Classifiers For Automated Identification Of Abstracts Of Studies On Humanistic And Economic Burden Of Disease. ISPOR, 19 (7), 355. https://doi.org/10.1016/j.jval.2016.09.051

(30) Schmutz, A., Bohn, E., Spaeth, J., & Heinrich, S. (2019). Comprehensive evaluation of manikin-based airway training with second generation supraglottic airway devices. Therapeutics and clinical risk management, 15, 367– 376. https://doi.org/10.2147/TCRM.S194728

(31) Neurological Exam [Internet]. Hopkinsmedicine.org. [cited 2021 May 23]. Available from: https://www.hopkinsmedicine.org/health/conditions-and-diseases/neurological-exam

(32) Al Shamsi, H., Almutairi, A. G., Al Mashrafi, S., & Al Kalbani, T. (2020). Implications of Language Barriers for Healthcare: A Systematic Review. Oman medical journal,

35(2), e122. https://doi.org/10.5001/omj.2020.40

(33) Shop Anatomical. (n.d). Full Body CPR/Trauma Manikin. Retrieved from https://www.shopanatomical.com/Full_Body_CPR_Trauma_Manikin_Simulaids_p/su-2700.html

(34) ABC News. (2007). Hospital Repeats Wrong-Sided Brain Surgery. Retrieved from https://abcnews.go.com/Health/story?id=3925810&page=1

(35) Likic, R., & Maxwell, S. R. (2009). Prevention of medication errors: teaching and training. British journal of clinical pharmacology, 67(6), 656–661. https://doi.org/10.1111/j.1365-2125.2009.03423.x

(36) Oracle (1999, April). Code Conventions for the JAVA Programming Language. https://www.oracle.com/java/technologies/javase/codeconventions-contents.html

(37) INACSL Standards Committee (2016, December). INACSL Standards of Best Practice: Simulation℠: Simulation Design. Clinical Simulation in Nursing, Volume 12, S5-S12. https://doi.org/10.1016/j.ecns.2016.09.005

(38) INACSL Standards Committee (2016, December). INACSL Standards of Best Practice: Simulation℠: Participant Evaluation. Clinical Simulation in Nursing, Volume 12, S26-S29. http://dx.doi.org/10.1016/%20.%20j.ecns.2016.09.009

(39) INACSL Standards Committee (2016, December). INACSL Standards of Best Practice: Simulation℠: Professional Integrity. Clinical Simulation in Nursing, Volume 12, S30-S33. http://dx.doi.org/10.1016/%20.%20j.ecns.2016.09.010

(40) Code of Federal Regulations - Cardiovascular Devices. 21 CFR § 870.5210 (2020). https://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfcfr/CFRSearch.cfm?fr=870.5210

(41) "Guidance for Industry on Computerized Systems Used in Clinical Investigations; Availability." The Federal Register / FIND, vol. 72, no. 90, 2007, p. 26638.

Appendix 1

Resources - Microsoft Tutorials

1. Microsoft Azure Speech to Text Tutorial
    a. https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/get-started-speech-to-text?tabs=script%2Cbrowser%2Cwindowsinstall&pivots=programming-language-java
2. Microsoft Azure Text to Speech Tutorial
    a. https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/get-started-text-to-speech?tabs=script%2Cwindowsinstall&pivots=programming-language-csharp
3. Microsoft's QnA Bot Tutorials
    a. https://docs.microsoft.com/en-us/azure/cognitive-services/qnamaker/quickstarts/create-publish-knowledge-base?tabs=v1
    b. https://azure.microsoft.com/en-us/services/cognitive-services/qna-maker/#overview

Appendix 2

Attachments

1. Neurologist Fidelity Study
    a. https://docs.google.com/document/d/1Xl_PVlJvLgumM3io1bjhfa2C_WaK47tw7PPH3gKxmeE/edit?usp=sharing
2. Medical Personnel Usability Study
    a. https://docs.google.com/document/d/1FrUtwoduCV6I84yosM8FAhGGg9QcpF-3YXpS7SVTey0/edit
3. Software Engineer Usability Study
    a. https://docs.google.com/document/d/1mw76Gdr_20fUEGPajO_aQX5zge3i5j8aMva5dMa5FHg/edit
4. GitHub Repository
    a. https://github.com/victoria20001/NLP-Student-Project