**A MACHINE LEARNING MODEL AND INTERFACE FOR PREDICTING COVID-19 TEST RESULTS**

**BY**

**AJIBOYE Oluwaferanmi Kelechi**

**180202003**

A PROJECT SUBMITTED TO THE DEPARTMENT OF PHYSICAL AND COMPUTER SCIENCES, COLLEGE OF NATURAL AND APPLIED SCIENCES, MCPHERSON UNIVERSITY, SERIKI SOTAYO.  
 OGUN STATE, NIGERIA.

IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF BACHELOR OF SCIENCE [B.Sc.] DEGREE IN THE DEPARTMENT OF COMPUTER SCIENCE

JULY, 2022

**DECLARATION**

I hereby declare that this project was written by me and is a correct record of my own research work. It has not been presented in any previous application for any degree of this or any other University. All citations and sources of information are clearly acknowledged by means of references

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

AJIBOYE, Oluwaferanmi Date

**CERTIFICATION**

This is to certify that this project was carried out by AJIBOYE Oluwaferanmi Kelechi in partial fulfillment of the requirement for the award of Bachelor of Science (B.Sc.) Degree in Computer Science, College of Natural and Applied Sciences, McPherson University, Seriki Sotayo, Ogun State, Nigeria.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Dr. F.E. Ayo Date

**DEDICATION**

I dedicate this work to God - the giver of every good thing, my parents for their unending sacrifice for me, and to myself because I deserve some credit

**ACKNOWLEDGMENTS**

My profound gratitude goes to God Almighty for His Grace, Mercy, Wisdom, Knowledge, Understanding and loving kindness bestowed upon me. I greatly express my gratitude to the lecturers in the Department of Computer Science, McPherson University, and my supervisors: Dr AYO Emmanuel & Mr Abiodun Mustapha for their challenging and kindly advice. I also acknowledge my course mates who remind me every day that I’m surrounded by good people. A special thanks to my parents Mr & Mrs Ajiboye for putting up with me, listening to my very many complaints and encouraging me to press on, and to my siblings and my Uncle Niyi, Uncle Bidemi and Uncle Obi, for their support. I also thank the CEO of Xclaim Labs –Mr Stephen Lawal for his patience and advice in my final year. I pray that God’s Grace will continually abound for you all.

**CHAPTER ONE**

**1.0 INTRODUCTION**

**1.1 BACKGROUND**

Artificial Intelligence (AI) is possibly the greatest innovation by man in solving many manual tasks performed by humans in various fields. John McCarthy widely regarded as the father of AI defined it as the science and engineering of making intelligent machines. AI has also been defined as intelligence exhibited by an artificial entity. Such a system is generally assumed to be a computer. AI has diverse fields including (but not limited to) Automated Reasoning, Robotics, Knowledge Representation, Perception, Natural Language Processing (NLP) (Neelam, 2021).

Machine learning is an evolving field of computer algorithms with the purpose to emulate human intelligence by learning from the environment. They are considered the flagship products of the new period of so-called big data. Procedures based on machine learning have been applied with optimum results in multiple fields ranging from pattern recognition, computer vision, spacecraft engineering, finance, entertainment, and computational biology to biomedical and medical applications. More than 50 percent of the patients diagnosed with cancer receive ionizing radiation (radiotherapy) as a major aspect of their treatment, and it is the main recovery modality at advanced stages of disease. (Issam et al, 2015)

A machine learning algorithm can be defined as a computational procedure that makes use of input data to perform a specific action or complete a given task without being explicitly programmed to exact a particular outcome. These algorithms traditionally learn from actual historic data that can include text, pictures, tables e.g. CSV files, and even voice and video data. (Quiroz-Jua´rez et al, 2021)

Coronaviruses are enveloped, non-negative single-stranded huge RNA viruses that contaminate human beings, but can also infect a wide spectrum of animals. Coronaviruses were first documented in 1966 by the scientists: Tyrell and Bynoe, who extracted and maintained the viruses from common-cold infected patients. Based on their morphology as spherical virions with a core shell and surface projections that were similar to a solar corona, they were dubbed coronaviruses. (Velavan & Meyer, 2020) In December 2019, a huge number of pneumonia cases were reported in the Wuhan province of China. Many of these cases reported visiting seafood and live animal markets. Investigations found that the disease was caused by a new strain of the coronaviruses. The disease was subsequently named COVID-19. This virus later spread to China and other parts of the world.

Coronaviruses are large groups of viruses and they consist of a core of genetic material enveloped by a liquid covered in protein spikes. This gives them an appearance of a crown. Crown in Latin is ‘Corona’. Hence the name, coronavirus. These viruses can cause respiratory disorders ranging from the common cold to more severe viruses including SARS (first discovered in China in 2003), MERS (First discovered in Saudi Arabia in 2012) and SARS-COV2 (discovered in China in 2019). Coronaviruses circulate in a range of animals including Camels, Bats and Snakes, the virus could however spillover to humans due to a range of factors including contact between humans and animals. The disease spreads between humans when an infected person comes in contact with an uninfected person. It can also spread through droplets.

COVID has different symptoms at different stages in the human body. The incubation stage of the virus (ranging from 5-6 days) include symptoms such as fever, fatigue, cough, and loss of smell or taste, sore throat. There can also be more severe symptoms like pneumonia. However some people may remain asymptomatic (show no symptoms of infection).

The most popular COVID test is Reverse Transcriptase-Polymerase Chain Reaction (PCR). This test identifies the virus via its genetic fingerprint. Treatment for COVID is majorly based on supportive care. There is currently no cure for the virus. The preventive measures for the virus include Covering the mouth and nose while sneezing, washing hands regularly with soap and water, maintaining social distance and appropriate use of masks and hand gloves. Contacting a medical personnel as soon as you experience symptoms guarantees the greatest chance of recovery from the virus (WHO, 2020)

Deploying applications and storing data on systems in data center accessible over the internet is filled with advantages. Although wherever they run, applications are deployed and maintained on a particular kind of platform. For applications on certain grounds, such as those deployed within a company’s data center, this platform typically involves an operating system, some medium to retain data, and maybe more. Applications deployed in the cloud require a related basis. The goal of Windows Azure is to provide this. Included in the much larger Azure platform, Azure is a foundation for running applications and storing data in the cloud.

(Chappell, 2010) Microsoft Azure is an umbrella brand name for Microsoft’s cloud-computing platform and functions. It entails a wide, and still evolving, spectrum of services that often form the basic components of cloud computing (Copeland et al, 2015).

One of the central themes of Azure Machine Learning is the ability to quickly create machine learning ‘experiments’, evaluate them for accuracy, and then “fail fast,” to shorten the cycles to produce a usable prediction model. In the end, the overarching goal of predictive analytics is to always be able to achieve a better chance of success than what you could achieve with a purely random guess (Microsoft, 2015)

**1.2 Motivation for the study**

The following are the motivations for the study:

1. Any and everyone can be infected with the virus. The transmission rate of coronaviruses is very high and uses complex routes. Testing should be made available to anyone around the globe (Liu et al, 2020)
2. Limiting the spread of the virus from already infected people with symptoms. Determining the infected using their symptoms will mitigate the spread to other people (Fegert et al, 2020)
3. Aid the enactment of lockdown and isolation rules. Reduce the impact of the virus in regional localities by determining the lockdown rules - if applicable - to be enacted.
4. Enable organizations to efficiently operate even with infected members of staff. Remote working has enabled many organizations to continue functioning with slight difficulty during the pandemic period (Sivasubramanian, 2020)
5. Better understanding of the spread of the virus. The knowledge of the pandemic is insubstantial. A better understanding of the nature of the virus and how it spreads from patient to patient will aid health workers in minimizing the transmission.
6. Patients who are currently in the incubation phase of the virus can receive immediate care for enhanced chances of recovery (WHO, 2021)

**1.3 Problem Statement**

A number of techniques have been put in place by respective governments, world organizations and private companies to counter the spread of the virus, however as humanity still has a lot to learn from COVID-19, many of these techniques are flawed. The problems facing the world today are:

1. According to the World Health Organization, there is no cure for the virus known to man currently (McAleer, 2020).
2. Testing kits are not readily available especially to less privileged parts of the world i.e. third-world countries. This presents a problem when diagnosing the virus in people in those areas (Giri & Rana, 2020)
3. High cost of manufacturing, distributing of testing kits, vaccines and preventive measures like hand sanitizers, face-masks etc. (Zunyou et al, 2020).
4. The risk of infection and transmission of the disease is higher when individuals go to testing centers. The risk of uninfected patients contacting the disease is increased when they visit testing centers. The risk of spread on infected people to others is also increased when they journey out to get tested. An alternative to this is home/office delivered testing kits, but this is one of the most expensive means of getting tested (Garner, 2021).

**1.4 Research Objectives**

The following are the research objectives of this study:

1. Perform a critical analysis on existing systems for detecting the coronavirus in patients using only symptoms
2. Design a cloud-based system for detecting coronavirus without performing actual coronavirus test.
3. Implement the system in Azure and validate system performance and reliance

**1.5 Organization of Work**

The remainder of the work is so: Chapter two focuses on the introduction to coronaviruses, COVID-19, the concept of cloud-based applications, Artificial Intelligence & Machine Learning, Deep Learning techniques and methodologies and related existing works.

Chapter three focuses on the methodology used in implementing the system on the cloud, architectural components of the system and algorithms. Chapter four describes the actual implementation of the system, result analysis, evaluation of the system and summary. In the Chapter five, conclusions are made from the implementation of the system with contributions to knowledge and recommendations of the research methodology.

**CHAPTER TWO**

## 2.0 LITERATURE REVIEW

**2.1 Overview of Coronaviruses**

As stated in the previous chapter, Coronaviruses are enveloped, non-negative single-stranded huge RNA viruses. Towards the end of the year 2019, In December a rapid outbreak of an unidentified disease termed ‘Pneumonia of Unknown cause’ originated in the Wuhan Hubei Province of China. The outbreak spread substantially to infect more than 9,000 people in china with more than 200 deaths and spread to infect 106 people in 19 other countries up until January 2020. Sometime later, the source of this mysterious disease was traced as a novel strain of the coronavirus by various independent laboratories all over the world. This virus has since become known as Severe Acute Respiratory Syndrome (SARS COV-2) and the disease generated by the virus names Coronavirus disease 2019 (COVID-19) (He et al, 2022)

To get a better understanding of COVID-19 causing this pandemic, we must go back to the origin of these viruses. The coronaviruses are a family of viruses commonly found in avian and mammalian species. They are similar to each other in morphology and chemical structure, for example, the genetic structure of coronaviruses in humans and cattle have no resemblance. There is no proof that human coronaviruses could be borne and transferred by animals. In animals, several coronaviruses can penetrate several variety of tissues and lead to a variety of illnesses, but in human beings, there is only evidence that they can lead to upper respiratory diseases and sicknesses, i.e. common colds. On uncommon occasions, gastrointestinal coronavirus infection has been linked with diarrhea outbreaks in children. Coronavirus virions are spherical to pleomorphic enveloped particles. This envelope is reinforced using projecting glycoproteins, and cover a central core consisting of protein encapsulated within which is just one strand of positive-sense RNA (Mr 6 × 106) connected with nucleoprotein. The encircled glycoproteins are in charge of the attachment to the host cell and also enforce the main antigenic epitopes, most especially the epitopes detected by neutralizing antibodies. OC43 also possesses a Hemagglutinin (Tyrell & Myint, 1996).

Although, the source of the COVID-19 pandemic is zoonotic in nature, the available epidemiologic data depicts that human transmission is obviously occurring. It is proven to spread through respiratory fluids from coughing and sneezing of an infected person or persons to uninfected people who come in close reach to them. The distance for this close contact is said to be about 6 feet. Transmission can also occur to someone who comes in close contact with infected surfaces or objects and then make contact with their eyes, ears, nose or mouth. The virus spreads most from symptomatic patients who are at the peak of their symptoms. However, research is also ongoing to prove if the virus can be spread from people who are infected but have not yet become symptomatic (showing symptoms). These group of patients are said to be asymptomatic (Holstein, 2020).

The most dominant symptoms of the COVID-19 are fever, dry cough, shortness of breath. 10% of patients have presented Nausea and diarrhea 1-2 days before to the growth of fever in a study conducted in an unnamed hospital in Wuhan, China. 75% of the patients in this same study reported a condition known as bilateral pneumonia. Bilateral Pneumonia is a serious infection of pneumonia that can inflame a patient’s lungs. The disease affects the tissue around the small air sacs in the lungs (Bathoorn et al, 2009)

**2.2 Concept of Cloud Computing**

Cloud Computing is not essentially a new field of computing, but an evolving one. It is coming up fast as a powerful and mainstream medium in the way we manage data and provide Information Communication services (Prince, 2011).

According to NIST, Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction (Mell & Grance, 2011).

The service models of cloud computing include:

1. Software as a Service (SaaS): SaaS platforms enable consumers to use the provider’s applications which are running on a cloud architecture. These applications can be accessed from several end-user devices either using a thin client user interface such as a specialized operating system or a software interface. The consumer is not in charge of maintenance or controlling of the foundational cloud system but his/her configuration services
2. Platform as a Service (PaaS): The platforms enables consumers to run applications created and deployed to the cloud by the users using popular programming languages, repositories, IDEs, and other software development tools of the users choice (Rabiek, 2017).
3. Infrastructure as a Service (IaaS): The consumers are given the capability to run unpredictable and scalable software on the resources, networks, data storage, API mechanisms provided by the Cloud Provider. Here, the user also does not manage the fundamental cloud infrastructure but has total control over his/her deployed applications (Barry & Dick, 2013)

A simplified definition of cloud computing is A set of IT services that are distributed over a network by a cloud provider to a client on a leased basis with the option for the user to scale (upgrade or downgrade) all their service functionality (Nandgaonkar & Raut, 2014).

Cloud Applications provide business and individuals the following benefits: A seemingly infinitely scalable platform for their apps, less expense in deploying or maintaining software over the internet, higher resource utilization, Recovery from data loss/disaster using cloud recovery.

The greatest advantage of cloud-driven apps however is the device-location independence. This means that both devices and servers can be in various locations and still connect flawlessly over the internet. (Ijcsmc, 2014)

**2.3 Introduction to Artificial Intelligence**

Popular commercial applications of Artificial Intelligence include Language Processing, Machine Learning, and Pattern Recognition etc. (MahaLakshmi, 2021).

Products like Siri, Alexa are classified as Language Processing software capable of understanding human language including slangs, accents and filtering out background noise. These software can also be used for voice authentication to prevent unauthorized access (Brownlee, 2019).

Artificial Intelligence is also used to regulate temperature in smart cars and smart homes. AI is also employed in IoT devices & software (Internet of Things). IoT deals with smart devices able to communicate with each other and human beings without requiring direct input from humans. Smart devices like Refrigerators, Waste bins and Air Conditioners can take action based on certain conditions without receiving instruction from their users. (McClelland, 2021)

Beyond these applications, AI can also be used in the medical industry. The major application in the field of medicine has been the use of Machine Learning.

The idea of machine learning is to emulate the manner in which humans and if possible other intelligent creatures learn to process input data in order to perform a particular goal.

(Johnson, 2021) The manner in which systems learn like humans can be divided into two: Supervised and unsupervised learning.

Supervised learning is a method in which each training data factor (e.g. size, color, material, and smell) is connected with its determined grouping label. It allows the learning system to deal with similarities and contrasting features. The second type of learning pattern is termed unsupervised learning or algorithm (El Naqa & Murphy, 2015).

Semi-supervised learning is a learning methodology that deals with the detailed study of the manner computers and other entities such as humans learn with the use of both labeled and unlabeled data. Usually learning is done either supervised or unsupervised as listed above. The aim of semi-supervised learning is to decipher how combining labeled and unlabeled data may affect the learning behavior of machines (Zhu & Goldberg, 2009).

Figure 1 depicts the relationship between Supervised, Unsupervised and Semi-supervised learning:

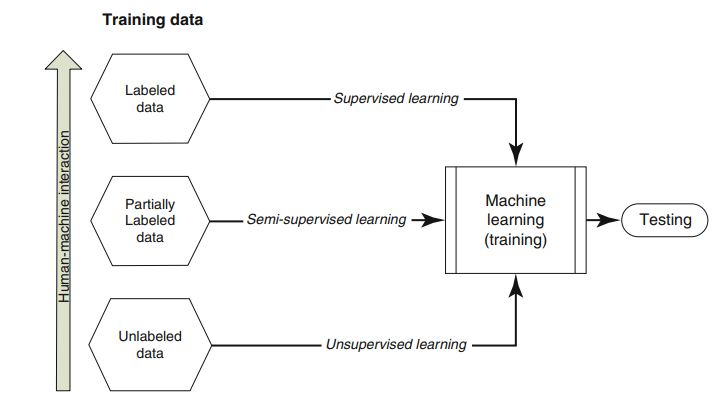


Figure 1: Aspects of Machine Learning Algorithms based on method of data training

Supervised Learning: Supervised learning allows the model to be trained using raw data containing labelled and unlabeled datum as input. Take for example, we are to build a facial recognition system, the raw dataset would include: labelled faces and unlabeled faces. The algorithm would then develop a model using the dataset giving. Supervised learning is applicable to problems where we can recognize the desired output

Unsupervised Learning: Unlike supervised learning techniques, unsupervised learning trains a model by using data that is neither classified nor labelled. This means that the model has to learn from the dataset entirely on its own. The algorithm develops what is termed “Clusters”. A cluster is created based off the visual interpretation of the model. This interpretations can include color, size, material etc.



Figure 0-1 Fields of Machine Learning

**2.4 Deep Learning Techniques**

Deep Learning is a type of machine learning methods based on learning data representations. This machine learning methodology permits models of computation that involve several layers of processing to learn depictions of data with various levels of abstraction. These methods have drastically enhanced the evolving fields of speech recognition, object recognition and detection and several other domains such as drug development (Coskun et al, 2017). Deep Learning inspects complicated structure in huge volumes of sets of data using an algorithm know as back propagation to determine how a machine should swap its intrinsic criterion that are used to calculate the depiction in each layer from the depiction in the preceding layer. (LeCun et al, 2015)

1. Convolutional Neural Networks: One of the most widely used Deep Neural Networks is the Convolutional Neural Networks (CNN). It derives its name from a mathematical linear problem between more than one matrixes called convolution. The most important feature about problems that are solved by CNNs is that the variables should be spatially independent. This means that for example, in a face detection app, critical care should not be placed to the location of the face in the image or the angle of the face, the face should be recognized notwithstanding the position (Albawi & Mohammed, 2017)
2. Recurrent Neural Networks: In the applications of sequential or time series data, typical feed forward networks are not suitable for learning and predictive algorithms. A methodology is needed that can store prior information that will be used to predict the subsequent unknown values. Recurrent Neural Networks (RNN) are a category of the traditional feed forward ANNs (Artificial Neural Networks) that can manipulate sequential data and can learn to store knowledge about previous data (Saeed, 2021)
3. Restricted Boltzmann Machines: RBMs are ‘chance’ pictorial models that can be translated as stochastic neural networks. The rise in power of computing and the creation of quicker and independent learning algorithms have made them applicable to certain problems associated with machine learning (Fischer & Igel, 2012)

**2.5 Related Works**

This section entails a review of related works in the diagnosis of diseases with the aid of Artificial Intelligence, detection of COVID-19 using symptoms and other approaches

**2.5.1**

**An expert system to diagnose COVID-19 and predict its severity using chest CT scans**

**(Abbasi et al, 2021)**

* **Objectives**

1. To provide an inexpensive and concise diagnostic test for majorly asymptomatic patients
2. Design a system to detect the severity of the virus in the patient

* **Methodology used**

The authors used a Deep Learning model where they automatically learned an efficient component representation of an image. This model has been successfully utilized in classification of various images and investigative tasks. Using transfer learning for feature extraction from the available CT images in their data sets was performed using various easily available CNN already trained models from ImageNet. The novelty of their suggested methodology is that it makes use of a mixture of already trained CNN based models to bring out features and shallow learning procedures such as SVMs for the means of classification. The proposed methodology is someway based on the pattern of transfer learning. Their aim was to acquire knowledge on two different functions for the identification and severity estimation of COVID-19. To accomplish this task, they made use of three separate machine learning-based classification procedures: classical Support Vector Machine (SVM), Random Forest (RF), and Gradient Boosting Machine (XGBoost)

* + **Results**

1. Improved performance over already existing methods including methods proposed by Kang et al, where models were trained using handmade features and produced an accuracy of 86%
2. Proposed methodology can not only be used for the diagnosis of COVID-19 but also be used to predict the severity of the virus in patients and reduce the surge of the virus by advising timely isolation.
   * **Strengths**
3. The proposed system is made easily available to the public through an open-cloud based webserver and open source code
4. The system detects not just the presence of COVID in a person but also determines the severity to determine counter measures
   * **Weaknesses**
5. The proposed methodology using CT scans causes certain problems like inability of very symptomatic patients to hold breath
6. CT scans can also lead to bone damage around the scanned area, this limits the number of scans the average human can run

**2.5.2**

**Artificial Intelligence-enabled rapid diagnosis of patients with COVID-19**

**(Mei et al, 2021)**

* + **Objectives**

1. To provide an alternate medium to testing COVID than the standard PCR test which takes up to 48 hours to complete
2. To control the spread of the virus by implementing a system that can track potential spread of the virus
   * **Methodology used**

The authors accumulated data on initial test CT studies and other hospital-related information from 905 patients from the period of 17th of January till 3rd of March 2020 in centers in thirteen provinces in China where the ill subjects had been exposed to the virus, fever and a PCR test. The clinical information recorded included: Age, Gender, Exposure history, symptoms, white blood cell count, percentage lymphocytes etc. The categorical parameters were Sex, Exposure history and Symptoms. They made use of LabelEncoder function in the popular scikit-learn package to encrypt the defined categorical parameters into numeric parameters. Then they normalized all the features from 0 to 1 with the MinMaxScaler function in the scikit-learn module for future development of the model. The available data from PCR tests were used to train the AI models. These models were developed using CT images and other clinical data. Then they created a joint CNN model which was a combination of radiological and clinical data. The first step after the models was to pre-process the images. They selected relevant images from the many images produced by a CT scan. Image Segmentation was used to detect parenchymal tissue from this images. CT images were then segmented into two parts, body and lung. They body part was divided by obtaining the biggest attached component that had both pixels and an intensity greater than 175. The size of the lung region was increased by 10px to fully show the pleural boundary. Images were abandoned if the size of the lung was less than 20% of the body area. Two CNNs were used: Slice selection CNN and Disease diagnosis CNN. The researchers used as the primary function a binary cross-entropy. The neural network was trained using the Adam optimizer and had a learning rate 0.001. They designed an impotently supervised task to give starting weights of the CNN model.

The joint model was created by combining CT imaging and other clinical data. The authors trained a model to integrate both data from the CT and clinic. They applied the general averaging level to the final layers of the convolutional network earlier prescribed to extract a 512 measured feature vector to depict a CT output. The sum of these measurements was used as the overall objective function to train the joint model. They used an optimization strategy of model 1 to train the CNN. It was also initialized by the weakly supervised task of classifying the tiny image patches

* + **Results**

1. Comparison of the ROC curves for the joint model, the CNN model trained on the foundation of CT images, the Multilayer Perceptor model trained using clinical data and two radiologists
2. Comparison of accomplishments of diagnosing patients who are ill with the virus with normal CT scans yielded 13/25 for the CNN model, 16/25 for the MLP model and 16/25 for the Joint model.
   * **Strengths**
3. CNN models, MLP model and the Joint model are trained on datasets from wide range of sources in the Wuhan Province of China. This gives diversification of data and enhances the probability of AI generating valid test results for patients
4. Test analysis for result prediction is 68% valid and this rapidly reduced the time taken averagely to detect COVID patients without AI
   * **Weaknesses**
5. Two radiologists were required to train the basis of the MLP model unlike the other CNN model.
6. Detection of COVID in patients by the CNN model yielded only a 52% probability of success as from 25 patients, only 13 were detected

**2.5.3**

**Application of Big Data Analytics to Control COVID-19 Pandemic**

**(Alsunaidi et al, 2021)**

* + **Objectives**

1. To reduce the spread of the coronavirus throughout the world by application of Big Data Analytics techniques
2. To diagnose the virus in patients by detecting the virus in pre-symptomatic patients before they become contagious
3. Reduce the strain on medical practitioners by applying Big Data Analytics
   * **Methodology used**

The authors proposed a model that will differ COVID from other chest diseases. This model makes use of many different sensors placed on the body to gather information and keep track of the patient’s health status involving temperature, blood pressure, heartbeat, respiration, glucose level etc. The information gathered was kept on a cloud database with Expert Systems that aid diagnosis of ill patients using their symptoms and determine the appropriate response for patients suspected of having the virus.

The authors gleaned multiple papers about mathematical models to enhance the effectiveness in the process of detecting and estimating COVID positive patients.  
The authors also obtained medical data which they believe the analysis can predict future circumstances, give a better understanding of the current situation and improve decision making process. This data was obtained from many public open databases on the internet. The authors discovered various tools that can be used to analyze data obtained from prior research including the R language, Python, MATLAB, MS Excel, and GraphPad Prism among many others.

* + Results

1. They provided a taxonomy structure which grouped the possible applications of COVID-19 into four spheres: diagnosis, estimate or predict risk score, healthcare decision-making, and pharmaceutical.
2. The authors also assessed essential insights on a variety of difficulties that may hamper the application of data analytics assets for COVID-19. These disadvantages include healthcare information security and patient confidentiality matters, the arduousness of distributing data among professionals, lack of data credibility for some researches that may give rise to flawed results, and the patients’ collaboration in distributing certain aspects of their clinical data
   * **Strengths**
3. Due to the increase in the volume of data over time as the pandemic continues to effect various countries over the world, the credibility of the author’s work will continue to scale globally leading to a rise in the accuracy.
4. A number of possible actions to take in the future can be assessed from the authors research
   * **Weaknesses**
   1. Absence of data validation for certain aspects
   2. Confidentiality within patients with the virus lead to limited clinical data available depending on the category. The difficulties involved in sharing data with other researchers to enhance their work.

**2.6 Summary of Related Works**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Title** | **Objectives** | **Methodology** | **Results** | **Strengths** | **Limitations** |
| Application of Big Data Analytics to Control COVID-19 Pandemic  (Mei et al, 2021) | To reduce the spread of the coronavirus throughout the world by application of Big Data Analytics techniques | Placed sensors on certain areas of patient’s body to gather information including: temperature, blood pressure etc.  Made use of tools like MATLAB, MS Excel, Python & R to analyze the information acquired | They provided a taxonomy structure which grouped the possible applications of COVID-19 into four spheres: diagnosis, estimate or predict risk score, healthcare decision-making, and pharmaceutical. | Due to the increase in the volume of data over time as the pandemic continues to effect various countries over the world, the credibility of the author’s work will continue to scale globally leading to a rise in the accuracy. | Confidentiality within patients with the virus lead to limited clinical data available depending on the category. The difficulties involved in sharing data with other researchers to enhance their work. |
| Artificial Intelligence-enabled rapid diagnosis of patients with COVID-19 (Alsunaidi et al, 2021) | To provide an alternate medium to testing COVID than the standard PCR test which takes up to 48 hours to complete | Use of Chest CT Scan data and other clinical information from a survey to detect COVID-19 | Comparison of accomplishments of diagnosing patients who are ill with the virus with normal CT scans yielded 13/25 for the CNN model, 16/25 for the MLP model and 16/25 for the Joint model | Test analysis for result prediction is 68% valid and this rapidly reduced the time taken averagely to detect COVID patients without AI | Detection of COVID in patients by the CNN model yielded only a 52% probability of success as from 25 patients, only 13 were detected |
| An expert system to diagnose COVID-19 and predict its severity using chest CT scans  (Abbasi et al, 2021) | To provide an inexpensive and concise diagnostic test for majorly asymptomatic patients.  To Design a system that can detect the severity of the virus in the patient | Created a model based on Deep Learning Techniques and three separate machine learning-based classification procedures: classical Support Vector Machine (SVM), Random Forest (RF), and Gradient Boosting Machine (XGBoost) | Improved performance over already existing methods including methods proposed by Kang et al, where models were trained using handmade features and produced an accuracy of 86% | The proposed system is made easily available to the public through an open-cloud based webserver and open source code. The system detects not just the presence of COVID in a person but also determines the severity to determine counter measures | The proposed methodology using CT scans causes certain problems like inability of very symptomatic patients to hold breath  CT scans can also lead to bone damage around the scanned area, this limits the number of scans the average human can run |
| A model for the effective COVID-19 identification in uncertainty environment using primary symptoms and CT scans (Abdel-Basst et al, 2020) | To Create a diagnosis model for COVID-19 assumption and diagnosis of medical signs to describe proper care methods | The authors made use of the BWM (Best Worst Method) to assess a group of surrogate options with consideration to a group of decision parameters. The BWM is based off a systematic pairwise comparison of the decision criteria | The model can differentiate COVID-19 from four other viral chest diseases with 98% accuracy | Model requires primary symptoms for its training | Method of application and consensus findings are applicable to a certain geographic region, leading to possible inaccuracy of the model in other locations |
| A Headset Like Wearable Device to Track COVID-19 Symptoms. (Stojanovic et al, 2020) | To design and develop a wearable device capable of tracking key symptoms of the virus | The authors used symptoms (both primary and latter symptoms) of the virus. These symptoms were recorded using sensors placed on the patient’s body. The information was used to train a model for prediction | The methodology gave ground to good and reliable outputs and can be enhanced to absorb the use of more sensors to diagnose other COVID-19 symptoms. | The paper presents a low cost and flexible design of a medical device for purposes of detecting and tracking symptoms of COVID\_19. Requires only a simple configuration | The model requires extensive patient input to obtain data which the authors used for model training |
| COVID-19 symptoms predictive of healthcare workers' SARS-CoV-2 PCR results (Lan et al, 2020) | To provide a symptom-based screening approach among health workers that will prove crucial in eliminating the spread of the virus. | We performed a retrospective study of HCWs undergoing both COVID-19 telephonic symptom screening and nasopharyngeal SARS-CoV-2 assays during the period, March 9—April 15, 2020 | Some of the strongest symptoms of COVID discovered in their research included: Fever, anosmia, and myalgia while no symptoms were only peculiar to nasal congestion/sore throat associated with negative cases | Screening of patients was conducted with phones over the internet, this ensured that patients could isolate while performing necessary tests | Accuracy of prediction compared to other researches which made use of CT scans or bodily sensors was much lower than using phone-based symptoms tests |
|  |  |  |  |  |  |

**CHAPTER THREE**

## 3.0 METHODOLOGY

**3.1 Architecture of the System**

The following components & tools will be required for the system:

1. Datasets from the Israeli Ministry of Health containing data that will help identify whether a person is positive with the virus or not based on some predefined standard symptoms set by the WHO.
2. ML5.JS: ML5 is a free to use machine learning package that was built on Tensorflow. ML5 runs in the browser on the frontend and can be used for Neural Networks for classification or regression tasks
3. Python library: Since Python is the primary programming language for data science activities, it will be used to process our dataset obtained. Python contains several libraries that allow developers to work with large sets of data. Pandas is the most popular library for handling data. Pandas will be used to work with the data obtained, convert it to a JSON (Javascript Object Notation) format before it will be used in ML5
4. JavaScript: The development of both the Frontend and Backend of the application will be done using JavaScript frameworks: React and NodeJS. React will be used to write JSX that will compile down to the standard HTML, CSS and Vanilla JS for frontend UI development. NodeJS which is simply JavaScript on a server (backend) will handle our API creation and will become the server which we will make requests to from the frontend
5. MongoDB: MongoDB is the most popular database for NodeJS applications. Unlike SQL, MongoDB does not store data in tables and rows and columns. Instead, data is stored in JSON format which is a better structure for data that may contain undefined values. MongoDB provides a free hosting tier which will be used in the initial stages of the applications
6. Github: All our frontend and backend code will be stored on Github. Our workflows for CI/CID (Continuous Integration/Continuous Deployment) will be instantiated from Github when we integrate with Azure Static hosting
7. Microsoft Visual Studio Code: VS Code is an Integrated Development Environment that will be used to write our locally sourced code during our staging phase. However, during production, updates to codebase will be performed using Azure DevOps to circumvent the issue of writing locally after deployment.

Figure 2 describes what our typical architecture is. The datasets contains cleaned data on patients who recorded their symptoms Using the Python library in Jupyter Notebooks, Pandas, a Notebook will be created so we can convert out dataset files from CSV to JSON. We can then proceed with creating and training the model with ML5 on the browser. When the model has been built, NodeJS will be used to handle our API calls as seen in Figure 2. All API requests handled by NodeJS will enable the storage of data in the popular database MongoDB. When API requests are made by the Node server, responses can be sent to the React frontend which will serve as the User Interface on the Client Side.

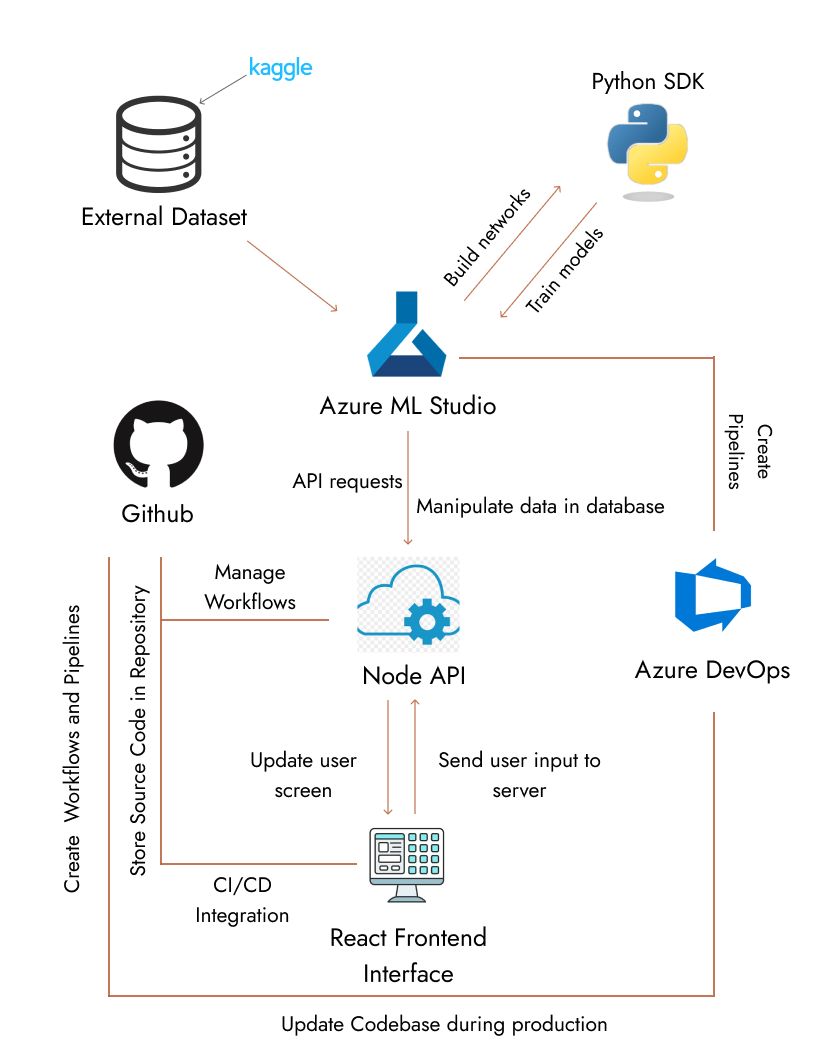


Figure 2: Architecture of the System

**3.2 Design Consideration**

GPU and CPU: Due to the high requirements for graphics and computing power by Deep Learning algorithms, Machine Learning approaches will suit the implementation of the system better. An average of 8 gigabytes of Virtual Memory is required for Mild – Severe solutions using Deep Learning. Serious calculations require even more memory in some cases up to 11 gigabytes. Due to this limitation in graphics and processor capabilities, Reinforcement Learning, Unsupervised Learning and Supervised Learning techniques will be used.

**3.3 System Specifications**

The system will be required to perform the following:

1. Display COVID-19 guidelines by the WHO: The WHO has laid out certain guidelines for the world concerning the pandemic. Some of these rules include: Keeping physical distance of at least 1m from others, Avoiding crowds, Wearing a properly fitted mask, Disinfecting hands and surfaces in your house and workplace with an alcohol-based sanitizer. The system must display these guidelines to users when the open the application even before they run a test.
2. Determine the probability that a user is positive with COVID-19: This will be done by collecting data from the user including: Symptoms, Travel history, Medical history, Isolation & Social distancing patterns, Exercise and work routines, possibility that user has come in contact with an ill patient, geographic location, number of ill patients in geographic location, pandemic severity in region, age group.
3. Suggest responsive measures for users: The system should be able to suggest measures that users can take including: Finding available medical centers based on the users location, isolation and social distancing guidelines, alerting medical authorities depending on the users results from the model.

**3.4 Development of the Model**

The following steps will be taken in developing the prediction model

1. Data Collection: This is the first process in developing the model. The primary dataset used in the training of this Model was obtained from a publicly available dataset by the Ministry of Health in Israel. The data contains the records of more than 1 million patients who were tested for COVID-19. Many of the patients were symptomatic. While some of the patients, after testing were found to not havve the virus, their records were kept thus increasing the accuracy of the data for data scientists. The symptoms on which this data was based can be found on the official WHO website. The data consists of the cough, Age, Current Symptoms, gender, and result outcome.
2. Preparation of Data/ Data Wrangling: The data which we obtain from Kaggle is currently raw and must be cleaned.
3. Choose the Model
4. Train the Model: The model will be trained with reinforcement, supervised and semi-supervised learning using the dataset from Kaggle. It will recognize and deduce if a user could be positive with the virus using the variables listed in **Figure X**
5. Evaluation of the model & Tune Parameters
6. Prediction and Inference

The following inputs are required for the model to predict test results. Figure x displays a list of inputs from the user required by the model. This inputs will be fed into the algorithm.

|  |  |
| --- | --- |
| VARIABLES | DESCRIPTION |
| Gender | User gender: Male or female |
| Location | Geographic location |
| Symptoms | Does user have COVID-19 related symptoms |
| Age | Users age in years |

Figure X: Variables required and description

Based on the received input for some of the variables, further inputs could be collected from the user including: Severity of symptoms experienced and current medical condition

Algorithm for Diagnostic Process

1. RECEIVE USER INPUTS AS LISTED ABOVE
2. TRANSMIT USER DATA INTO MODEL
3. PROCESS INPUTS & DIAGNOSE TEST RESULTS
4. RETURN MODEL OUTPUT RESULTS
5. RECOMMEND POSSIBLE COUNTER MEASURES DEPENDING ON RESULTS

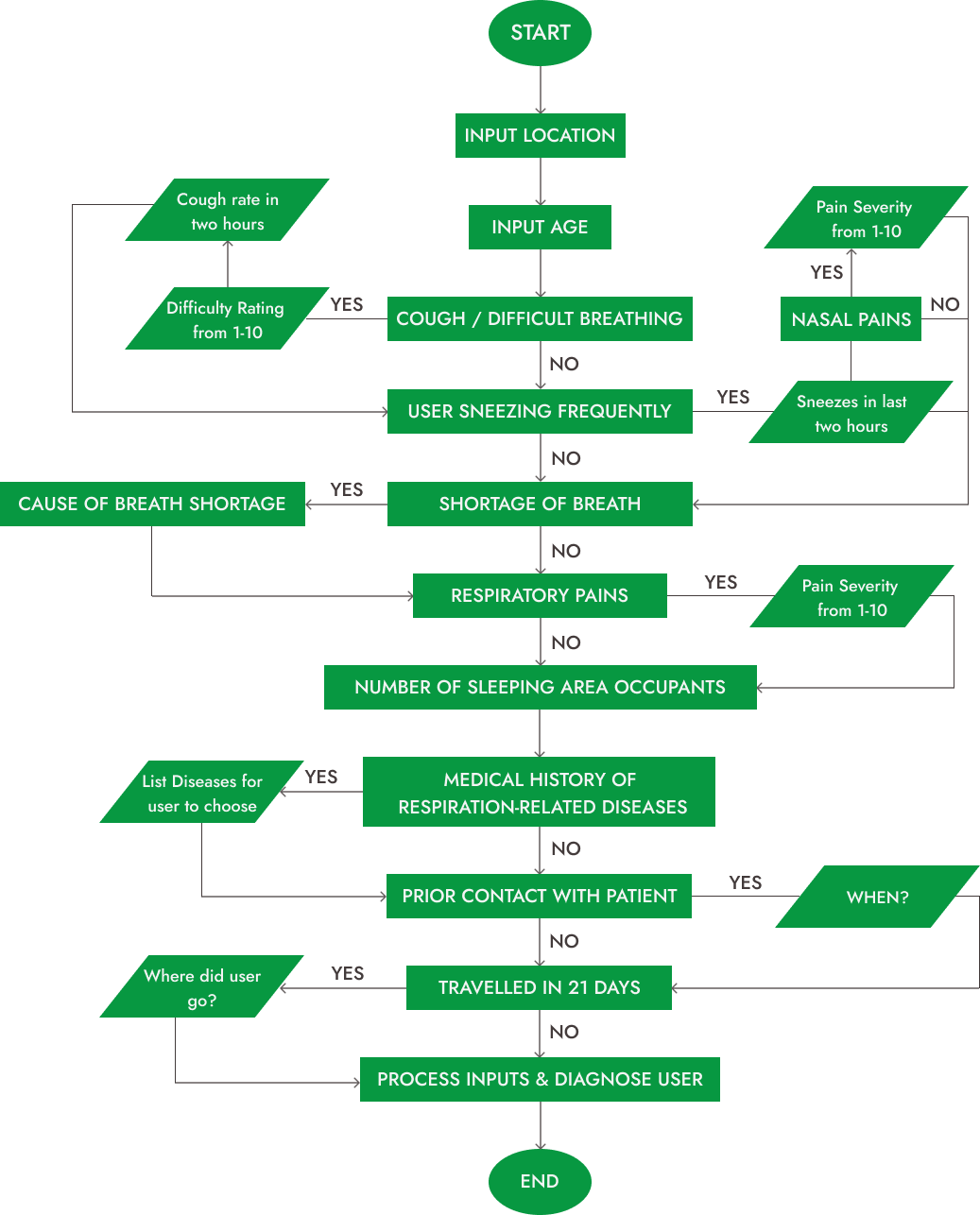
MAJOR FACTORS AFFECTING A POSITIVE TEST

1. If the user is in a vicinity with a high spread of the virus
2. If the user has a high scale of difficulty in breathing
3. User has shortage of breath after doing minor or no strenuous activities
4. Experiences unusual respiratory pains in regions around the chest and back
5. Has recently travelled to a region that is highly affected by the spread of the virus

MINOR FACTORS AFFECTING A POSITIVE TEST

1. If the user has frequent sneezing without cough or respiratory pains
2. If the number of occupants in the sleeping area is high (poor isolation guidelines)
3. If the user came across a positive patient but has no symptoms (This is not a perfect ideology since many patients – particularly those infected with the Omicron variant – are asymptomatic)
4. If the user has a medical history of respiratory related diseases and has symptoms related to both his medical conditions and COVID-19. This may affect the result and the user will be advised to take an actual PCR test in a qualified medical center

If a user who has poor isolation at work or home generates a positive result prediction, then other members would be advised to take a test as well. Since the tests are not performed taking actual fluid from the users, a more accurate diagnostic would require the user taking at least three tests and finding the average result based on the tests

Figure X: Flowchart of the Diagnostic Process

**3.5 Build the Frontend Interface**

The frontend development process is divided 4 into major modules:

1. The UI/UX design
2. The Implementation of the Design
3. Mobile-responsive view
4. Testing the developed frontend

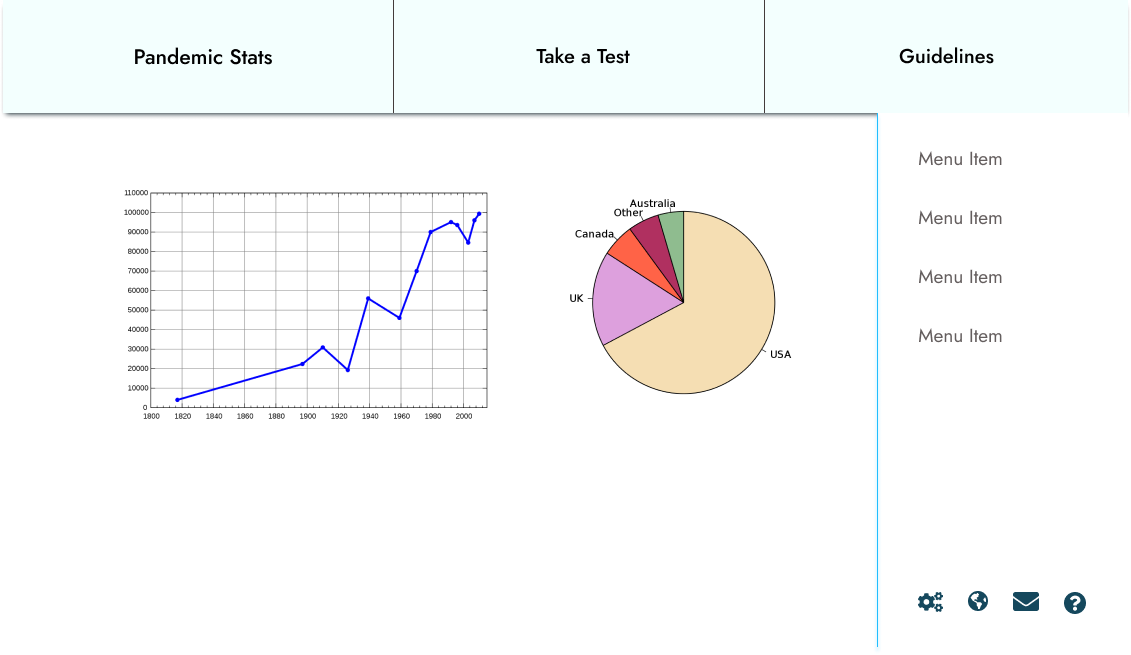


Figure X: Sample prototype of the User Interface

The UI/UX design is typically performed after requirement gathering and analyzing. The requirement gathering stage is complete upon development of the model, so we proceed to the design of the interface. In this case, we will use Figma although there are numerous alternatives available including Illustrator, Framer etc.

Next, we will implement the design in the popular frontend framework ReactJS. As is seen in the Architecture Diagram in **Figure 2**, React will serve as the interface our users will interact with.

After our React implementation with the aid of certain modules including: MUI and js-ccokie, we will design a mobile-first version of the frontend. As at November 2021, more than 46 percent of the total activities done on the web are done with mobile devices, compared to approximately 50.4 percent coming from desktops and PCs (Oberlo, 2021)

The final stage of the frontend development is the Evaluation and Testing. Testing will ensure that there are no glitches or bugs hidden in our interface and we need to the design to ensure that the constraints from the UI are applicable in our code. The frontend of our application should be complete.

**CHAPTER FOUR**

## 4.0 IMPLEMENTATION

**4.1 COVID DETECTION MODEL USING ML5 NEURAL NETWORK**

**4.1.1 TRAINING THE NEURAL NETWORK MODEL**

An Artificial Neural Network (ANN) is based on a collection of connected units or nodes called artificial neurons which loosely model the neurons in a biological brain.

An ANN consists of neurons. A neuron in an ANN is simply a building block similar to the ones in the human brain. Neurons make up the hidden layers in an ANN that process input layers to produce output. There is a concept of activation functions in neurons. These activation functions take inputs and give outputs. When a neuron receives a value, it passes the value through the activation function.

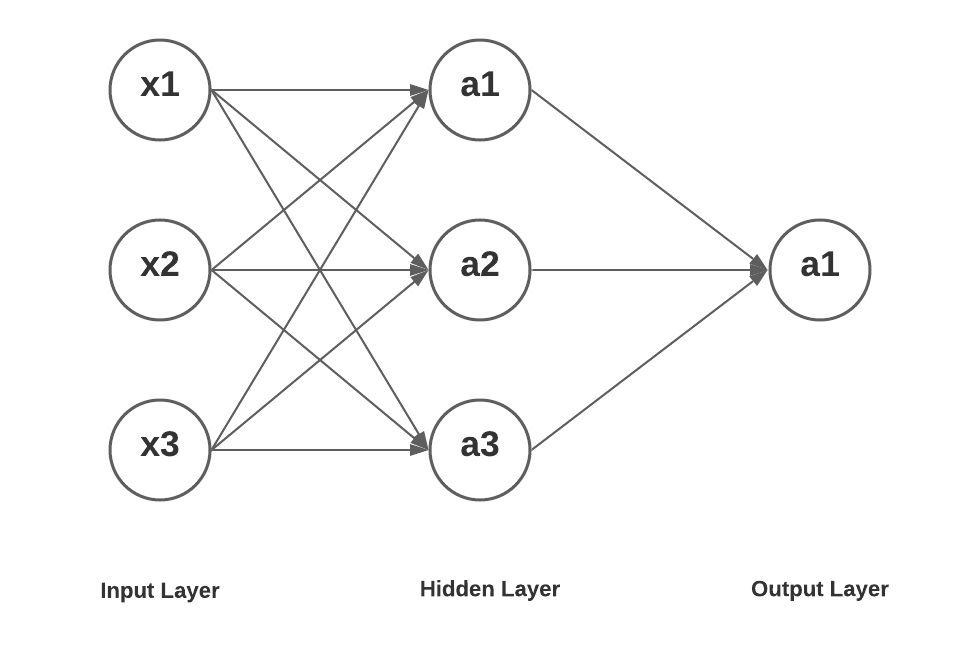


Figure X: Model of a simple artificial neural network

The figure above represents the simple model of an Artificial Neural Network containing the input layer, hidden layer and finally the output layer.

Below are the first 10 records of the dataset in csv format:

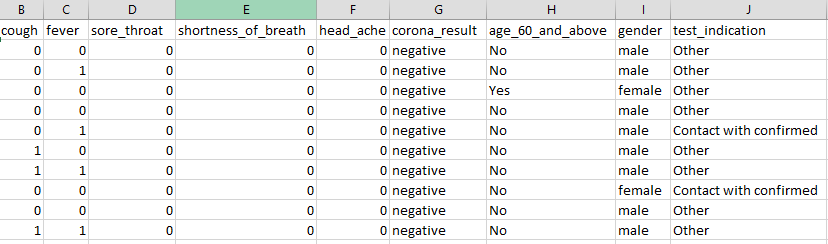


Figure X: Dataset showing first 10 records in CSV (Comma Seperated Values) format.

To make use of the dataset in ML5, it will be easier to make use of it in .JSON format. JSON (Javascript Object Notation) is a popular form of exchanging and storing data particularly on web based applications. To convert our dataset from CSV to JSON, pandas library from python and Jupyter Notebooks will be used:

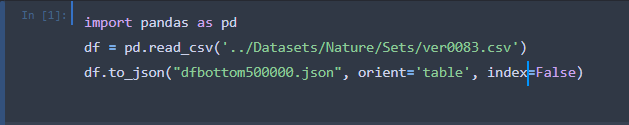


Figure X: Converting dataset from CSV to JSON using pandas

A new Jupyter terminal is created and then the pandas library is imported. The dataset is opened in a pandas dataframe and then .to\_json() method is called to convert the CSV dataframe to JSON format.

Now that the dataset is in JSON format, the process for training the model with the JSON file can commence

The inputs for our Neural Network from the records in the dataset are as follows:

|  |  |  |
| --- | --- | --- |
| INPUT | VALUE | REMARK |
| Cough | 1 or 0 | Does user have cough: true or false |
| Fever | 1 or 0 | Does the user have a fever: true or false |
| Sore\_Throat | 1 or 0 | Does the user have a sore throat: true or false |
| Head\_Ache | 1 or 0 | Does the user have a head ache: true or false |
| Shortness\_of\_Breath | 1 or 0 | Does the user experience difficulty in breathing: true or false |
| Gender | 1 or 0 | What is the user’s gender: Male or Female |
| age\_60\_and\_above | 1 or 0 | Is the user above the age of 60 or not |

The gender input accepts either “Male” or “Female” and transforms it into binary inputs: 0 or 1. If user is male, gender = 0, else if user is female, gender = 1.

All inputs have to be converted to numerical values between -1 to 1. So values can range between -1, 0 and 1. 0 represents false, 1 represents true and -1 can be “mid” or “other”.

The model generates a single array of possible output for each classification: “negative”, “positive”, and “other”. The array contains objects for each possible output listed above. The objects will also contain a “confidence” value. The total of all confidence values will equal to 1. This means that the output with the highest classification will be the classification decision.

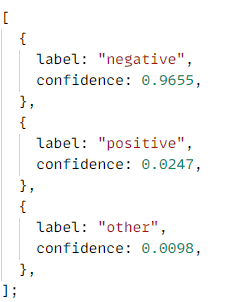


Figure X: Sample response returned by a classification

Consider the above figure containing a sample response returned by running a classification on a set of user inputs. The response is an array of three objects. Each object contains a ‘label’ and a ‘confidence’ property. The label is of type string while the confidence is a float variable. The confidence values all add up to 1 showing that the result is accurate. The output object with the highest confidence value is the model’s prediction of the most likely outcome of the test result.

ML5 Neural networks can perform either Classification and Regression tasks. We shall be creating a classification model. The model predicts a value for each classification after training on values of other data in our dataset. This model will make use of Classification to predict either positive or negative result for each test run. According to ML5, these are the following steps to create and train a model to perform classifications:

Step 1: Load data from your dataset or create some raw data in realtime

Step 2: Set your Neural Network options

Step 3: Initialize the neural network

Step 4: Add data to the Neural Network

Step 5: Normalize your data

Step 6: Train your neural network

Step 7: Use the trained model to make a classification

Step 8: Obtain results on your classification

Using the above steps, the model will be implemented in code:

1. Step 1: Load Data

The first step is to import the dataset to be used. We will fetch the dataset using Axios library JSON file from our NodeJS server currently running locally at: http://127.0.0.1:8080/.

We will train initially using the top half of the dataset which we have in a JSON file on the route /datasets/new/tophalf/

const baseURL = "http://127.0.0.1:8080/";

axios

  .get(`${baseURL}datasets/new/tophalf`)

  .then((trainResultData) => {

    const trainResult = trainResultData.data.file;

  })

Our dataset is now stored in trainResult. Using this data we will train the model

1. Step 2: Set Neural Network Options

The options for training the neural network to build the model need to be predefined. These options variables include:

task: "classification" | "regression"

debug: true | false

Since we are creating a classification model we will set our task to "classification". We will also set debug to true, this will enable the training visualization so we can see the model progress as it is being trained.

Our options now look like this:

const options = {

  task: "classification",

  debug: true

};

1. Step 3: Initialize the Neural Network

The next step is to initialize the neural network using the options we have defined above. This is done by calling the method neuralNetwork() in the ml5 object and assigning it to the neural network name:

const nn = ml5.neuralNetwork(options);

1. Step 4: Add data to the neural network

In step 1, we imported the data to be used in the model training. Now we must add the data one after the other into the model. This is done using nn.addData().

Firstly, we use the javascript forEach function to iterate over every element in the dataset array and get their inputs and single output. Then we insert both the inputs and output into the training set for the model and move on to the next. This will run for every data record we have:

trainResult.forEach((item) => {

      const inputs = {

        cough: item.cough,

        fever: item.fever,

        sore\_throat: item.sore\_throat,

        head\_ache: item.head\_ache,

        shortness\_of\_breath: item.shortness\_of\_breath,

        age\_60\_and\_above: item.age\_60\_and\_above === "Yes" ? 1 : 0,

        gender: item.gender === "male" ? 0 : 1,

      };

      const output = {

        corona\_result: item.corona\_result,

      };

      nn.addData(inputs, output);

    });

1. Step 5: Data Normalization

In Machine Learning tasks, data normalization is done to ensure that the different features in the dataset have similar ranges for their values. We use the normalizeData() method in the instance of the neural network to perform data normalization

nn.normalizeData();

1. Step 6: Training the Model

After performing the various tasks listed above up to data normalization, it is time to train the model.

The first step to training the model is to determing the number of epochs and the size of the batch to use for the model training.

An epoch refers to a traversal of the entire dataset by the model during training.

The batch size refers to the number of training examples utilized in one iteration.

const trainingOptions = {

      epochs: 32,

      batchSize: 12,

   };

We call the train() method using trainingOptions as a parameter to perform a train on a neural network and then the save() method in the callback function to save the model so we can make classifications without having to retrain the model

nn.train(trainingOptions, () => {

      nn.save();

});

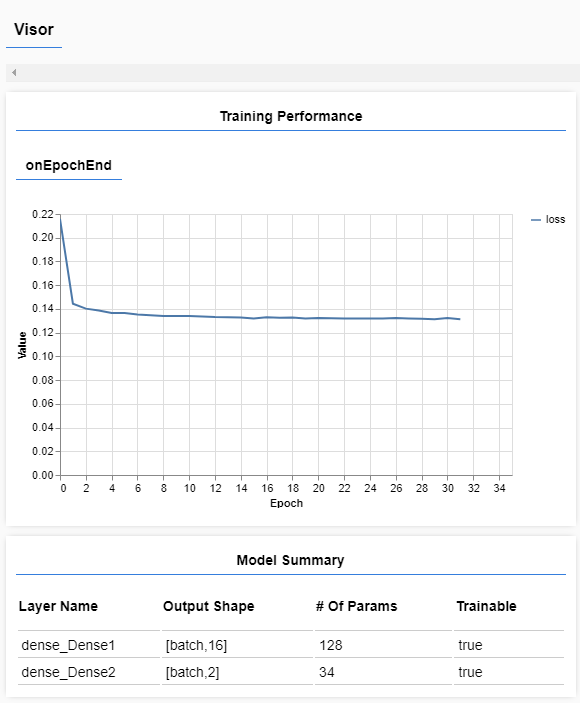


Figure X: Visual feedback for model training using ML5.js

After training, the model will be exported so that we can test it’s validation accuracy. This will determine if the model can be used or not. The model’s exports will include 3 files:

* model\_meta.json
* model.json
* model.weights.bin

Our models’s data is stored across these three files and will be explained later in detail

We will repeat the model creation process training using:

1. Bottom Half of dataset to train, Top Half of dataset to test
2. Top 75% of dataset to train, Bottom 25% of dataset to test
3. 100% of dataset to train

We will determine the validation accuracy for all models. But for our train using the Top Half of dataset to train, our full training script now looks like this:

const baseURL = "http://127.0.0.1:8080/";

//Fetch top half of data data to train

axios

  .get(`${baseURL}datasets/new/tophalf`)

  .then((trainResultData) => {

    const trainResult = trainResultData.data.file;

    console.log(trainResult);

    const options = {

      task: "classification",

      debug: true,

    };

    const nn = ml5.neuralNetwork(options);

    trainResult.splice(0, 500).forEach((item) => {

      const inputs = {

        cough: item.cough,

        fever: item.fever,

        sore\_throat: item.sore\_throat,

        head\_ache: item.head\_ache,

        shortness\_of\_breath: item.shortness\_of\_breath,

        age\_60\_and\_above: item.age\_60\_and\_above === "Yes" ? 1 : 0,

        gender: item.gender === "male" ? 0 : 1,

      };

      const output = {

        corona\_result: item.corona\_result,

      };

      nn.addData(inputs, output);

    });

    nn.normalizeData();

    const trainingOptions = {

      epochs: 32,

      batchSize: 12,

    };

    nn.train(trainingOptions, () => {

      nn.save();

    });

  })

  .catch((err) => {

    console.error(err);

  });

**4.1.2 TESTING THE NEURAL NETWORK MODEL**

Now that our model has been trained and exported, we can move on to testing the model. As mentioned earlier, we will train and test with three other methods apart from the one performed in 4.1.1:

1. Bottom Half of dataset to train, Top Half of dataset to test
2. Top 75% of dataset to train, Bottom 25% of dataset to test
3. 100% of dataset to train

The model trained with the top half of the data done in 4.1.1 will be tested with the bottom half of the data. The percentage success of the testing data will determine the validation accuracy of the model and whether we can move it to production or not.

We will create another script file for testing the model after it has been trained. We will call this script test.js

The model options for testing still remain the same as training

const options = {

  task: "classification",

  debug: true,

};

We will create separate array variables for percentageSuccess, correct, and wrong:

const percentageSuccesses = [];

var correct = [];

var wrong = [];

We will then initialize the neural network using our options, load the model.json file and finally in the callback we will begin to test the model using our testing set.

Using Axios, we will fetch the testing set from our backend server and store the set array in testSegment:

axios.get(`${baseURL}datasets/new/bottomhalf`).then((testingResult) => {

const testSegment = testingResult.data.file

})

We will iterate over each record and then get inputs and the required output from each record using the .map() javascript built-in array function

testSegment.map((testRecord) => {

      const input = {

        cough: testRecord.cough,

        fever: testRecord.fever,

        sore\_throat: testRecord.sore\_throat,

        head\_ache: testRecord.head\_ache,

        shortness\_of\_breath: testRecord.shortness\_of\_breath,

        age\_60\_and\_above: testRecord.age\_60\_and\_above === "Yes" ? 1 : 0,

        gender: testRecord.gender === "male" ? 0 : 1,

      };

      const output = testRecord.corona\_result;

Now we will define a classification function called classify() that will make use of the .classify() ML5 function. This function takes in a parameter and then a callback function. The single parameter is the input to be classified. Since we are making a classification for all members of the testSegment array, we will call the function for each record in the array. The callback function has two parameters:

1. Error: This will be returned if an error occurs while making a classification. We can then log this error to the console using console.error() to view the error.
2. testResults: If no error is found, the results of the classification can be obtained from this parameter.

For each classification we make in the callback function, it can either predict successfully or fail. If the result of the test is ‘negative’ and the model predicts ‘negative’, then it is correct and the model accuracy is said to improve. However if the result of the test is ‘negative’ and the model predicts ‘positive’ or vice versa then the model prediction has failed and the accuracy is lowered.

Finally, our percentage success which will also serve as our validation accuracy will be the (number of correct predictions) / total \* 100.

Our testing script now looks like this:

function classify() {

        nn.classify(input, (error, testResults) => {

          if (error) {

            console.error("Error: ", error);

          } else {

            const mapp = testResults.map((result, index) => {

              console.log(result);

              if (result.label === output) {

                if (result.confidence > 0.5) {

                  //Prediction is correct

                  const obj = {

                    prediction: true,

                    pos: testResults.indexOf(result),

                  };

                  correct.push(obj);

                } else {

                  //Prediction is inaccurate

                  const obj = {

                    prediction: false,

                    pos: testResults.indexOf(result),

                  };

                  wrong.push(obj);

                }

              }

              return {

                length: testResults.length,

                index,

              };

            });

            return Promise.all(mapp).then((e) => {

              const wrongNum = wrong.length;

              const correctNum = correct.length;

              const total = wrongNum + correctNum;

              const percentageSuccess = Math.floor((correctNum / total) \* 100);

              percentageSuccesses.push(percentageSuccess);

            });

          }

        });

      }

classify();

Upon testing the model trained with the top half of the dataset with the bottom half, we yielded a **94.5%** model accuracy. The model

**4.2 CREATING THE FRONTEND INTERFACE**

The frontend for the application would be solely built using ReactJS. I began by using Figma the popular design tool to create the User Interface design (UI Design). The designed would be themed with only a few colors:

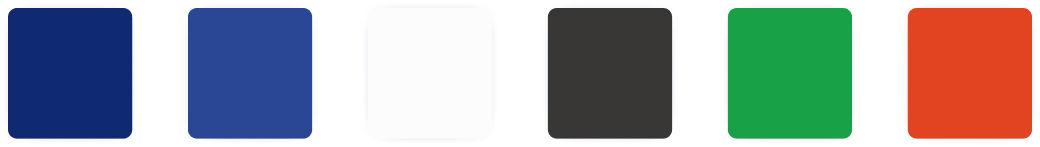


Figure X: Color palette for frontend interface

The pages available on the web site would be:

1. Home page
2. Authentication page – User login or register
3. Dashboard
4. New test page
5. View previous test page
6. User account page

Below are some the designed pages on Figma from the preliminary design:

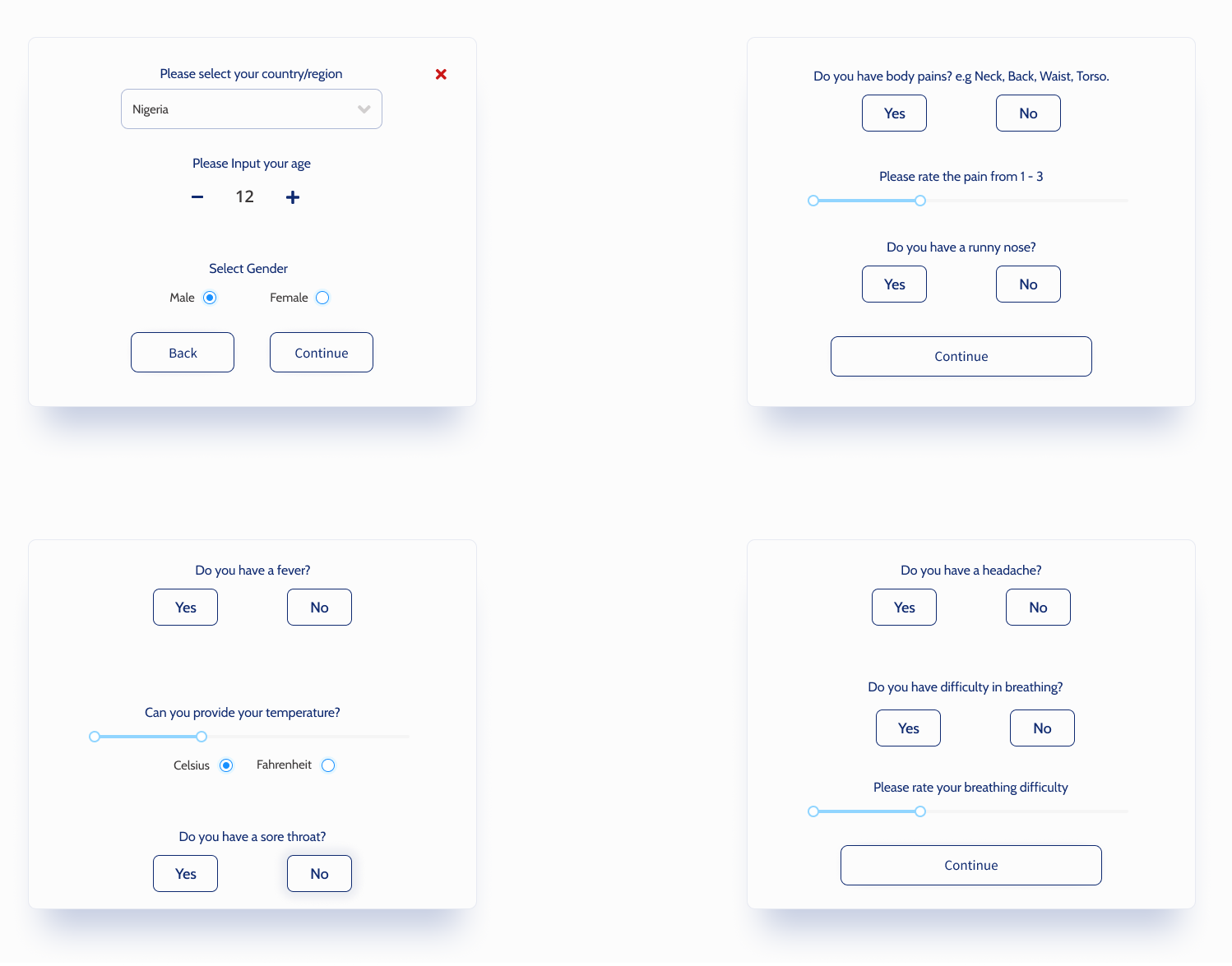


Figure X:

The screen above represents the different segments of the form the user will fill that will give input for the model to make a classification when we move from development into production



Figure X:

The screen above shows the authentication page with the register component. The user will fill this form properly to create an account to use the web application.

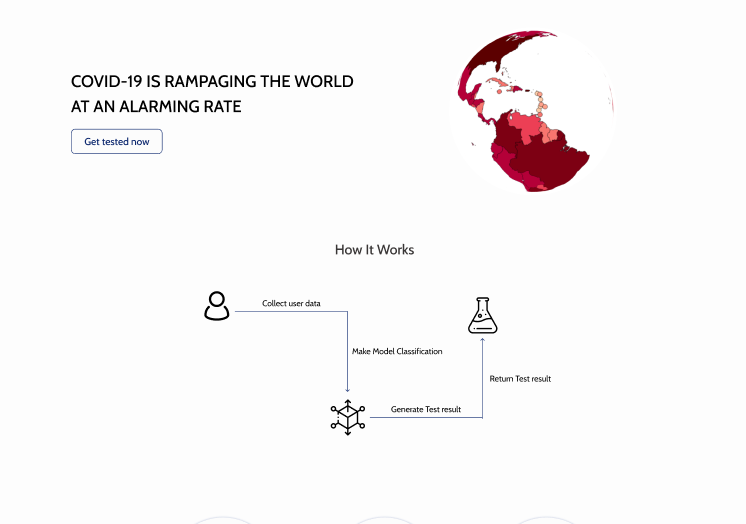


Figure X

Figure shows a section of the home page from Figma design. The home page (also sometimes called the index page) will be the default page the user of the site will visit when the launch the web app. It will contain important details about the model including: the dataset used, the model accuracy after testing and testimonials from users who have tested out the model.

After the preliminary design on Figma, the actual User Interface will be converted into code using React as mentioned. React makes it easy to write HTML in Javascript known as JSX. Our JSX code can be compiled down to a single index.html file which will be used for our virtual DOM (Document Object Module). Our frontend will be hosted on Github and will deploy on Netlify using CI/CD (Continuous Integration / Continuous Deployment) technique. The following packages will be used in the frontend to make our application more intuitive.

1. Framer motion: This is an animation package by Framer. Framer motion is a React animation library for adding transition effects and other animations. It is free to use and will serve as our animation package.
2. Ant Design: Ant design is a design toolkit that comprises of several components. Components we will make use of include:

* Select
* Radio
* Switch
* Slider
* Modal
* Notification

1. Recharts: Recharts is a Javascript package that is used to plot charts, graphs, and other data visuals. Recharts will be used to show the user the percentage probability of being either positive or negative with the virus.
2. Three.js: Three is one of the most popular 3D libraries available for Javascript. It is used to render 3D shapes, planes and other objects right in our browser. Three will be used on our homepage to show a globe affected by the influence of COVID-19
3. Keen-slider: Keen slider is one of the many available carousel packages for React. It will be used for our slideshow in our homepage for user testimonials
4. Axios: Axios is an alternative to the built in Javascript fetch() API. Axios is used for making API requests in our applications. Axios will be used to exchange data with our NodeJS server running on the backend as well as fetch Covid statistics from RapidAPI
5. Js-cookie: This is a a frontend library for storing and removing cookies in the browser. We will use cookies to store user sessions while they are logged in.
6. Material UI: MUI (formerly known as Material UI) is a React design library that is open-source. We will make use of several components from MUI including: Grid and Container

All packages used will be referenced in a package.json file for building on Netlify servers. A list of the packages in our dependencies from our package.json file:

{

    "@emotion/react": "^11.9.0",

    "@emotion/styled": "^11.8.1",

    "@mui/material": "^5.8.0",

    "@testing-library/jest-dom": "^5.16.4",

    "@testing-library/react": "^13.2.0",

    "@testing-library/user-event": "^13.5.0",

    "antd": "^4.20.5",

    "axios": "^0.27.2",

    "bizcharts": "^4.1.16",

    "framer-motion": "^6.3.3",

    "js-cookie": "^3.0.1",

    "keen-slider": "^6.6.14",

    "ml5": "^0.12.2",

    "react": "^18.1.0",

    "react-dom": "^18.1.0",

    "react-router-dom": "^6.3.0",

    "react-scripts": "5.0.1",

    "react-toast-notifications": "^2.5.1",

    "recharts": "^2.1.9",

    "three": "^0.140.2",

    "web-vitals": "^2.1.4"

  }

**4.3 CREATING THE BACKEND API**

**4.3.1 SETTING UP OUR NODE ENVIRONMENT**

The backend of our application will be created using NodeJS and several NodeJS libraries. The backend will perform the following functions:

1. Handle User authentication (Login and Signup)
2. Store new test result in database
3. Retrieve user test results from database
4. Delete test results from database

In development, the backend server will also host dataset files used for training the model. The backend will handle our connection with our MongoDB database running on MongoDB Atlas. MongoDB Atlas is a cloud-based database service that is popular with NodeJS applications but can be used with other languages including: Java, Python, dotNET etc.

I initialize a new NodeJS application by opening up an empty folder and running the node command: npm init. This initializes a new project. I then install the required project dependencies using npm install <dependency> --save. I will list the following dependencies and their functions.

Express: Express is the most popular NodeJS framework. It simplifies the process of defining routes and handling API calls from our frontend. Express replaces the standard HTTP or HTTPS server available on NodeJS with a simple express()method.

Bcrypt: Bcrypt is an encryption and decryption library that is popular for user security. This library will be used to encrypt user passwords before storage in the database.

Body Parser: Body parser is used to obtain form data sent from the frontend to the backend. It enables us to receive data sent from the frontend of our application for proccessing before sending to the database

CORS: Cross-Origin Resource Sharing is a mechanism that prevents unauthorized domains from accessing our backend. We specify a base URL which will have access to our backend. If any other access point attempts to call a route on our backend, CORS will prevent the call from going through.

Mongoose: Mongoose is an ORM (Object Relational Mapping) package available for MongoDB. It allows creating and using models for our database much simpler.

JsonWebToken: JsonWebToken (JWT) is a method of authentication users using tokens that are stored in the users browser and referenced whenever authentication is needed. JWT will be used to keep user logged in for every session.

Our dependencies object in our package.json file for our backend server now looks like this:

{

    "bcrypt": "^5.0.1",

    "body-parser": "^1.20.0",

    "cors": "^2.8.5",

    "dotenv": "^16.0.0",

    "express": "^4.17.3",

    "jsonwebtoken": "^8.5.1",

    "mongodb": "^4.5.0",

    "mongoose": "^6.3.1",

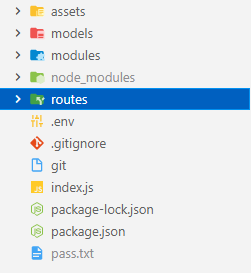
    "nodemon": "^2.0.16",

    "randomstring": "^1.2.2"

  }

After installing all the packages required, there is a need to arrange a proper folder structure:

The root directory of the project looks like this:



1. Assets: Assets folder will contain the datasets for training the model. This datasets will be sent down via an API call
2. Models: The Models folder will contain all the models and their respective schema that we will use to create data records in the database
3. Modules: The modules folder will contain a single file: main.js It will contain all methods and properties that will be imported by other files across our program structure
4. Node\_modules: Node\_modules is the directory that contains all the libraries and packages installation files used in our application
5. Routes: The routes folder will contain different files that correspond to the respective routes they will handle
6. .env: The .env file will contain environmental variables for our program. This includes port number and MongoDB private url

**4.3.2 AUTHENTICATING USERS INTO THE APPLICATION**

For users to take tests on the website, they must first be logged in to the site. This will enable their test results to be safely stored on the database. It will also allow us to use test results gathered from users for training the model in future. There are two major routes for authentication: "/user/register", "/user/login". These routes will receive form data from the frontend and process it for authentication.

Registering new users

1. Receive form data from the frontend: Name, Email, Password

const { name, email, password } = req.body;

1. Check if user already exists with the email

User.findOne(

    {

      email: email,

    },

1. If user exists, return error:

res.json({

          userExists: true,

          success: false,

        });

1. If user does not exist, use bcrypt to encrypt the password and create a hash that will be stored in the database:

bcrypt.hash(password, salt, (err, hash) => {}

1. Create a new user object using the User schema method
2. const user = new User({

           userID: userID,

           name: name,

          email: email,

              password: hash,

      });

1. Save the new user into the database

user.save();

1. User has been successfully created
2. Use JWT to create a token with the secret key and the user ID (now created in the database)

const token = signJWT(userID);

1. Return the token and a success response to the frontend.

res.json({

           success: true,

           token: token,

      });

Login existing users

1. Get user email and password from form in frontend

const { email, password } = req.body;

1. Check if account with email exists in database

User.findOne({ email: email }, (err, response) => {})

1. User Bcrypt to check is password provided matches password hash stored in database

bcrypt.compare(password, passwordHash, (err, doesPasswordMatch) => {})

1. If password matches create a new token and send positive authentication response with token to frontend.

const token = signJWT(userID);

res.json({

auth: true,

token: token,

});

1. If password does not match send error response to frontend

res.json({

auth: false,

});