SymCheck: Self-Diagnosing Web Application Using Image Recognition

Alan Singleton, Briana Figueroa, Jasmine Washington, Morgan Franklin, Sayema Islam, Swapnil Kadekar
Seidenberg School of Computer Science and Information Systems
Pace University, New York, NY, USA

***Abstract*— Medical diagnosis, when stripped down to its bare-bones, fundamental underlying principles, can be seen as a process grounded in pattern recognition—a careful analysis of an instance through the lens of existing trends and data, and their subsequent classifications. And in a world where medicine and technology see a greater degree of interplay each year, a number of technological applications exist for allowing people to thus diagnose themselves of having or not having certain ailments and illnesses based on comparisons of their own symptoms to those of positive cases past. However, each of these applications has its own features and methods of delivery, and as a result also show various grades of accuracy in producing a reasonable prediction.**

This paper offers a solution that strives to work off of and improve on its predecessors, offering a form of machine learning-based classification that closely parallels a physical medical diagnosis. In using an image of symptoms as input, and analyzing its features piece by piece before consolidating them to compare with known diagnoses to produce a reasonable result, this study produces a novel application that allows users to save time and money, while still receiving a considerably reliable understanding of what is going on in their bodies. And in doing so, the application produced strives to make its own contribution to an era that welcomes speed, convenience, and self-sufficiency.

***Keywords*— self-diagnosis, computer science, medical imaging, machine learning, deep learning, image classification, convolutional neural network, CNN, chickenpox, varicella, pinkeye, conjunctivitis**

I. Introduction

SymCheck is an application that arose as the product of an everchanging world—one where people seem to get busier and busier each year, and thus show a desire to take more control of the uncertainty in their lives through a degree of self-sufficiency, as well as one taken aback and driven to ever-increasing wariness by a crippling pandemic. The idea behind the self-diagnosing web application is quite simple: create a means by which people can assess their own medical conditions *without* paying a physical visit when deemed unnecessary; for why should one expend precious time and gain exposure if a situation does not demand it? And according to SolvHealth, an acclaimed online healthcare facility locator, symptoms of [itchy] rashes, as well as illnesses such as conjunctivitis or pinkeye are only two of the most common, yet generally not life-threatening, conditions treated at urgent care facilities [28]. Thus SymCheck was born, becoming one of the first image-based web applications for self-diagnosis capable of classifying illnesses and abnormalities specific to more than one part of the body.

The SymCheck application can be broken down into two major components: a trained machine learning-based model for image classification, and the frontend user interface. The learned model used for this study is one based on a convolutional neural network, or CNN, and it receives its image inputs for classification from uploads through a dynamic webpage that serves as the user interface, with the two being integrated via a classic client-server model. Both components will be discussed in more detail in the sections to follow, with II and III examining current machine learning studies and the specific algorithms used as well as existing applications for self-diagnosis respectively, IV giving a brief overview of SymCheck's base requirements, V detailing the roles of each member of the SymCheck team, VI discussing the technologies used, VII detailing the [current] results of both the classification model and the user webpage as well as an evaluation of the completed application with ideas for future improvements, and lastly VIII to provide some closing remarks.

II. Literature Review

Data mining techniques have been showing increasing prevalence in implementation in medical analyses around the world, including in more rural regions in the far East such as Tibet for everything from formulating association rules for sets of diseases to mining relationships in drug prescription for the same [27]. In fact, for many regions in China as well as in various other parts of the world, the existence of data mining techniques and the subsequent understanding of the relationships they show are very important in the sense that these less-urbanized regions oftentimes lack specialized doctors who can correctly identify symptoms as being akin to those of a particular illness/disease [14]. This issue in specialization can also then be extended to more urbanized regions as well, given that even in regions where people have more access to specialized medical care, these specialists are not available at convenient urgent care facilities. Moreover, a visit to a specialist generally costs more than a visit to a primary care

doctor. Thus image-based classification grounded in machine and deep learning is becoming increasingly popular for certain illnesses and diseases, especially those in the skin and eyes, something SymCheck attempts to cater to [14].

According to Hameed et. al.'s study on inflammatory skin lesions, "the human skin is the largest human organ, and it acts as a barrier between the human body and microbes as well as pathogens. When this barrier [is breached] and harmful environmental elements invade the human body, skin problems originate" [15]. Thus it is no surprise that various studies exist for the classification of a number of different skin illnesses and abnormalities including, but not limited to, melanoma (both malignant and benign), measles and German measles, and chicken pox [18] [22] [23]. However, only Islam et. al.'s study speaks specifically to the [successful] identification of chicken pox, or varicella, and implements an artificial neural network or ANN based on a texture-analysis-based approach for classification, achieving an overall accuracy of 80% [18]. On the other hand, while Namazov and Cho's study speaks to the classification of melanoma only, the two researchers employed the use of a CNN as opposed to an ANN (i.e. more layers for feature extraction), and were able to achieve an overall accuracy of 95-98% [22].

As for existing studies revolving around the image-based classification of pinkeye or conjunctivitis, interestingly enough, there are much fewer in the ophthalmic specialty as compared to the dermatological realm. And of those that exist, classification of the illness is achieved through more comparative algorithms (as opposed to breaking images down into finer features) such as K-nearest neighbors or KNN, support vector machine or SVM, and Histogram of Oriented Gradient or HOG [19] [31]. And while these methods of classification are computationally fast and less costly than a CNN, they work off of a similarity measure between *similar* illnesses. This notion is something that ought not be employed when considering two different illnesses of two different parts of the body, and when considering the costs of *misclassification* and comparison of two unrelated maladies [25].

Thus given the existing studies analyzed—all their outcomes and considerations—SymCheck becomes the first application of its kind to study the results of employing a CNN for the classification of two (and eventually more) popular, but dissimilar, generally non-life-threatening ailments. And in doing so, the application both exists as a manifestation of convenience and accessibility while also creating a means for users to save both time and money.

III. Current Solutions

A self-diagnosing application is an application that will help determine what illness a person may have based on their symptoms. This type of application is not one hundred percent accurate because it is a program "diagnosing" the illness, as opposed to a trained professional. Yet, self-diagnosing applications are continuously gaining popularity due to accessibility, user-friendliness, speed, and ability to save a visit to a doctor's office. Since the appearance of COVID-19 in the world in late 2019 to early 2020, people are more likely to avoid leaving home unless it is for essential needs/items. As more people are staying home due to the pandemic, they are less likely to go to the doctor's office for illnesses that aren't life-threatening. So self-diagnosing applications are a great tool for people who want a reliable diagnosis for their illness without having to see a doctor in person.

As self-diagnosing applications are widely accessible, all that is required is a user page and Wi-Fi. Depending on the user's preference he or she can choose from several different applications to determine his or her illness. Currently, the three major solutions for self-diagnosing applications are AskMD, Mayo Clinic, and Symptomate. Each application contains unique features which can impact whether users are interested in using that self-diagnosing tool or another. For reference Table 1 portrays a simple example of how these tools are similar and different.

TABLE I
AN OVERVIEW OF THREE SELF-DIAGNOSING APPLICATIONS

App Name	Platform	Upload Images	Other Functionalities
AskMD (ShareCare)	Mobile and Web Application	No	Microphone voice feature for ease of use
Mayo Clinic	Web Application	No	Sidebar when to seek medical care
Symptomate	Web Application	No	Graphical representation of affected area

AskMD, also called ShareCare, is a self-diagnosing tool that is available as a mobile and web application. The ShareCare application was launched in 2012 and has been providing self-diagnoses to many people ever since. Figure 1 contains a screenshot of the homepage of the ShareCare application.

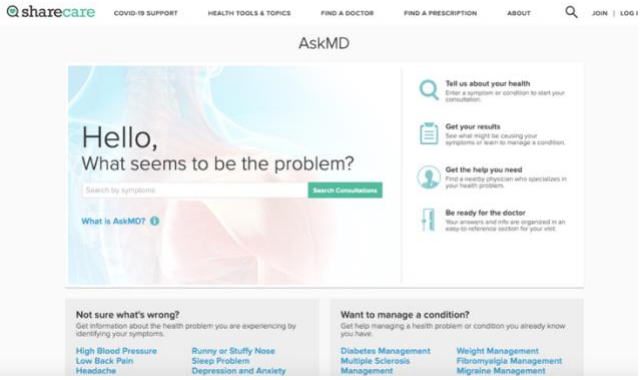


Fig. 1 Screenshot of ASKMD Symptom Checker

In the box, the user types what symptoms he or she has, then he or she clicks "Search Consultations" to continue the diagnosing process [26]. The next page lists several ways the symptom could be affecting the user. Whichever one is closest to the symptom he or she has, the user can decide to click it and then he or she is directed to a login page. This is where ShareCare becomes different from the other applications. Creating an account slows the process of diagnosis down and could likely cause a user to switch to another self-diagnosing application. On the other hand, ShareCare does provide a microphone functionality and the ability to complete a diagnosis through

their mobile application, providing a different user experience than the other applications.

Mayo Clinic provides a symptom checker on their homepage but states the symptom checker is not a diagnosing tool. Since this tool isn't a *complete* self-diagnosing tool it is faster to use as compared to AskMD and Symptomate. First, the site displays numerous Adult and Child symptoms for the user to choose from. As shown in Figure 2, the homepage of Mayo Clinic's application displays the symptoms and how many steps it takes to complete a symptom check [20].

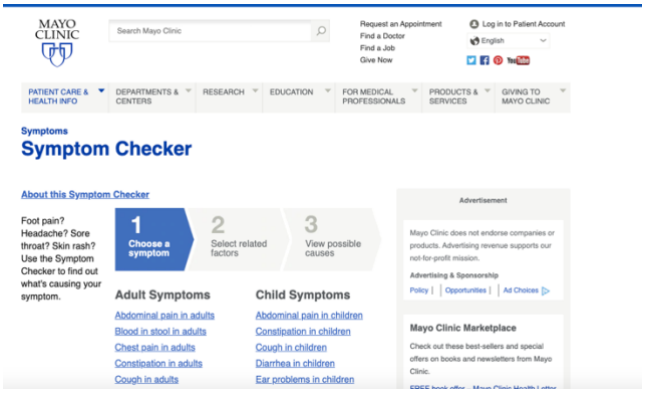


Fig. 2 Screenshot of Mayo Clinic Symptom Checker

Once a symptom is chosen, the "Select related factors" questionnaire appears which asks to select other factors that are present with the symptom. After this step is completed, selecting the "Find causes" button navigates to the next page and displays a list of possible diseases or conditions that correspond to the symptom and other factors the user selected. Under each disease or condition the factors the user selected are displayed in bold and those he or she did not select remain un-bolded. Mayo Clinic's application also provides a useful sidebar for when someone should seek medical attention based on his or her symptom.

Symptomate was also launched in 2012, around the same time as AskMD. Symptomate is an application dedicated solely to self-diagnosing the symptoms a user may have. As shown in Figure 3, Symptomate's homepage allows the user to complete a normal self-diagnosis questionnaire or complete a COVID self-diagnosing questionnaire [30].

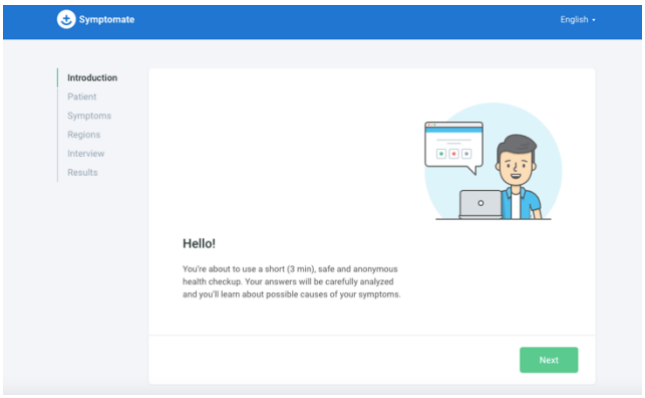


Fig. 3 Screenshot of Symptomate Symptom Checker

While completing the normal self-diagnosis questionnaire, many questions are asked about the user and his or her symptoms. An interesting addition in this section is the featuring of a graphical representation of the human body. The user can click what area on his or her body has been causing issues and then symptoms pop up to choose from based on the area selected. The questionnaire is longer than the Mayo Clinic application but provides a more thorough analysis and diagnosis. And the results from the questionnaire display a recommendation of what the user should do and provide what the application determines could be going on with the user.

As shown in Figures 1-3, AskMD, Mayo Clinic, and Symptomate all have similar web browser layouts where the user starts by inputting his or her symptoms. Then, after receiving symptom input(s), they all provide some type of possible disease or illness the user may have. Self-diagnosing applications are a new way in medicine and technology, so it is increasingly useful to come up with new features any such applications can use to improve their systems.

IV. Product Requirements

Our goal is to develop a self-diagnosing web application for users experiencing symptoms of chickenpox or pinkeye. This application differs from all self-diagnosing applications in that it allows users to upload *images* for diagnosis. SymCheck will then analyze the image uploaded and present results based on a machine learning-based algorithm. These results will then help users/patients possibly determine their illnesses, understand more about said illnesses, and provide them with information on when to seek further information or assistance from a healthcare professional.

The SymCheck application requires a number of major functionalities to become a complete self-diagnosing application:

1) *Homepage* – Users/patients should be able to first land on SymCheck's main welcome or home page, which will give them a quick, yet thorough overview of SymCheck's main functionalities, as well as how the application works (and how accurately) in attempting to better users' lives. The homepage will also feature a button allowing users to move to the image upload page in order to begin their diagnosis, as well as another button allowing them to view a short demo of how exactly, and ideally, to use and navigate the SymCheck website.
2) *Upload Image* – Users/patients will have the capability to upload an image of their chickenpox or pinkeye by clicking a button to browse one's local file-space. After the image has been successfully uploaded, a preview of said image will be displayed, and it can then be submitted for analysis by SymCheck's original machine learning algorithm to compute the user's results. If the user does not want to upload any further images, he or she can return to the homepage or find the nearest Emergency Room.
3) *Advanced Prediction Page* – Based on the illness image submitted by the user/patient, users will be directed to an "Advanced Prediction" page which, based on the classification already performed in the backend by SymCheck's machine learning algorithm, will feature

simple, illness-specific yes or no questions for the user to answer if he or she chooses to. Answering these questions allows for the user to receive a more accurate determination of the potential ailment he or she may have. The user may also choose to skip directly to his or her results without answering these extra questions, or can seek the nearest healthcare facility by clicking the "Nearest Emergency Room" button.

4) *Results Page* – After users/patients have successfully uploaded an image of their chickenpox or pinkeye, and the algorithm has completed analyzing the image with or without the additional, illness-specific questions being answered, the results page will appear. On the results page, the first item to be shown is the result of the image analysis containing what illness the user may have and the percentage probability of having that illness. This headline will also be accompanied by a dynamic *probability percentage bar*, a feature described in detail in #9 following. Next, there will be more information about the illness displayed such as statistics, treatments, and/or severity. And lastly, the results page will contain another display about different conditions that indicate when it is best to see a doctor. As with all the other main pages, the user will also have the option of seeking the nearest healthcare facility if he or she chooses based on his or her results.
5) *FAQ Page* – If users/patients have any questions regarding the SymCheck application (i.e. regarding more information on any of the featured diseases or general questions about the SymCheck application or website), they will be able to navigate to a "Frequently Asked Questions" page that can be accessed from any page on the SymCheck site through the top navigation bar. Here the user can peruse a list of the questions most commonly asked by patients using SymCheck, and read the full answers to these quick questions without having to contact the team, or interact with SymCheck's built-in chatbot (see #6 following). Users will also be able to find the nearest healthcare facility via a "Nearest Emergency Room" button on this page.
6) *About Us Page* – Users/patients will have the capability to access a page on which they can learn more about the team behind the SymCheck application. This page features quick summaries of each team member's role in delivering the final application.
7) *Navigation Bar* – Users/patients will be able to navigate to any page of the SymCheck application (Home, Upload, FAQ, About Us) through the use of the top navigation bar. This feature was added for user convenience, so that users could access all components of SymCheck from any page (as opposed to being restricted to a specific website flow as was the case with the original SymCheck delivered in semester 1).
8) *Find Nearest Emergency Room* – A "Nearest Emergency Room" button will allow users/patients to open an embedded window featuring a Google Maps API within the SymCheck application that will allow nearby hospitals and other healthcare facilities to be displayed. Users will have access to this functionality on the Upload page, the Advanced page, the Results page, and the FAQ page.
9) *Probability Percentage Bar* – On the final results page of SymCheck application, users will also be able to view a visual representation of their results in addition to all the textual descriptions displayed in order to allow for even more convenience for a wider range of users/patients. The probability percentage bar appears above the final determination that a user has a certain illness/ailment based on their submitted image (and any additional information from optional advanced prediction questions), and it is filled in based on the likelihood of a user having a certain illness, as well as color-coded based on how threatening said results are (in correspondence to the percentage)
10) *Chatbot* – If users/patients have any quick questions they wish to ask while using and navigating our website, a Chatbot will be available on <u>every</u> page of the website. The user need only open the chatbot and enter whatever question he or she has, to which the chatbot will respond with pre-programmed responses. Currently, the chatbot's main functionality is to provide users with guidance on the illnesses featured on the site, as well as how best to get in contact with the SymCheck team.
11) *Page Footer* – <u>Every</u> page in the SymCheck application will also feature the site's footer, containing links to all component pages of the application (much like the top navigation bar), as well as links to various resources used to build the SymCheck application. This footer will also contain links to all of SymCheck's *social media accounts*, as well as a link to contact SymCheck via *email* if a user/patient wishes to provide any feedback or related any additional questions/concerns.

Figure 4 shows a high-level process diagram of a user's experience when using the SymCheck application. Each bullet point signifies the features accessible within that page. The upper right hand corner of each page signifies the presence site-wide features.

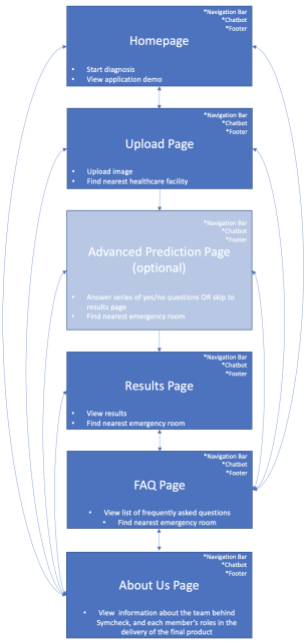


Fig. 4 Process flow diagram of SymCheck

V. Team Roles

Behind every great product or application is a strong, hardworking team. In order to deliver the requirements outlined in the previous section, roles and tasks were carefully divided up as *specialties* to each member of our six-person team, in a way that played to each person's strengths. And while the team's roles did see some interplay at times in order to deliver the best possible results, above all else, the team members showed incredible chemistry in working together, communicating with each other at all times, embracing and accommodating one another's idea, and helping one another whenever needed. But the main team roles were as follows:

- *Scrum Master/Project Manager* – Jasmine Washington
  - Led team meetings and assigned all team tasks
  - Assisted with frontend and backend development
  - Co-authored technical paper and maintains project's GitHub repository and Wiki
- *Product Owner/Lead Developer* – Sayema Islam
  - Originator of Symcheck
  - Reviewed code for accuracy and assisted with debugging
  - Authored technical documentation and maintains project's GitHub repository and Wiki
- *Frontend Developer/UI Designer* – Briana Figueroa
  - Coded several pages of the application
  - Designed page layouts and various graphics
  - Made our web application mobile-friendly
- *Machine Learning Engineer* – Swapnil Kadekar
  - Developed original machine learning algorithm
  - Designed backend architecture
  - Researched and formulated datasets for training, and performed statistical analysis
- *Full-Stack Developer* - Alan Singleton
  - Incorporated APIs such as Google Maps and Dialogflow
  - Optimized application code for modularity
  - Developed logic for advance prediction pages
- *Cloud Engineer* – Morgan Franklin
  - AWS Administrator
  - Provisioned code collaboration and cloud services
  - Documented test cases and co-authored technical paper

VI. Methodology

The SymCheck application has been implemented in a classic client-server model, using various programming languages, technological modules, APIs, and Agile methodology. The details of the technologies used are described as follows:

1) *Client* – The frontend of SymCheck utilizes HTML, CSS, and Javascript. Thus allows users to view and interact with SymCheck within all supported browsers.
2) *Server* - The backend of the SymCheck is implemented using Django as its web framework. And the machine learning algorithm developed has been implemented using the Keras library with TensorFlow as its backend engine. The motivations behind using this server architecture, as well as all other supporting technologies used to add additional features to SymCheck are discussed in depth in the following subsections.

A. Django

Django is a server-side web application framework that is written in Python. It encourages rapid development and allows for simple design of web applications [9].

Django's architecture implements a framework called the Model-Template-View (MTV). The model layer consists of the tools used to manipulate data and databases. The view layer is responsible for taking the data provided by the model layer and transforming it into an HTTP response object to be sent to the client. And the template layer provides the user interface which generates the HTTP request and displays the HTTP response in the client's web browser [32]. Figure 5 shows the Django framework MTV architecture.

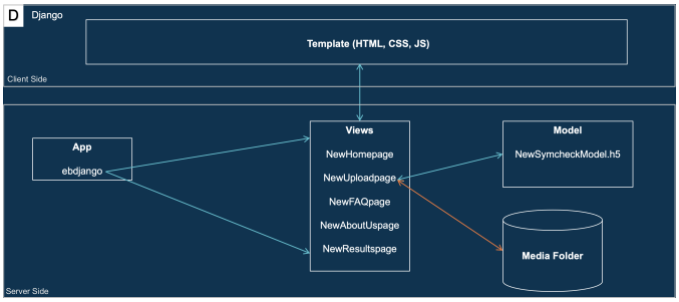


Fig. 5 Django's Client/Server architecture diagram as used for SymCheck

B. Why Django?

When initially searching for a web application framework, it was important to find solutions that supported machine learning algorithms. Thus, two frameworks came into consideration: Flask and Django. Table 2 shows a comparison between Flask and Django on various attributes [8]. But it is important to note that many popular websites, web applications, and software tools have been programmed with the Django framework such as Instagram and Spotify [8]. Thus, it was due to its wider community presence and faster development speed that we chose to use Django as our web framework.

TABLE II
A comparison between Flask and Django [17]

Term	Flask	Django
Simplicity and Flexibility	Simpler	Hard
Development speed	Slow	Fast
Size of project	For bigger ML project	For small ML project
Availability of Add-ons	Less	More
Community	Small	Wider

C. TensorFlow and Keras

TensorFlow is an open source library for various machine learning models. It uses Python for its frontend API and executes applications in C++ [35]. The library can train and run neural networks for image recognition, natural language processing (NLP), and more.

Keras is a high-level API that is embedded in TensorFlow. Like TensorFlow, Keras is also written in Python and it supports multiple neural network computation engines [12].

Keras is user friendly and it provides, "out-of-the-box implementations of common network structures" [12].

D. *SymCheck's Image Classification Training Model*

Using the Keras API, a Convolutional Neural Network (CNN) using the Sequential model was built for the purposes of "diagnosis" on the SymCheck application. Figure 6 shows the process of image classification training followed by said CNN.

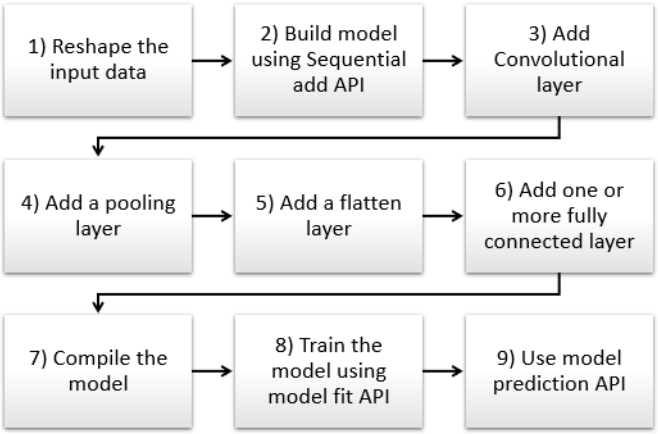


Fig. 6 Typical Process flow diagram for building a CNN architecture [21]

1) Reshape the input data

The input data, or image in our SymCheck application, has to be resized into a suitable format for proper processing. Trying to analyze multiple images with varying sizes could cause the algorithm to inaccurately diagnose illnesses.

2) Build model using Sequential add API

Within the Keras library, we can start building our Sequential model using the add function. Steps three through six will be parameters to add functionality for feature extraction.

3) Add Convolutional layer

A Convolutional layer scans an image with a predetermined filter to extract features for classification. In our application, features would include pinkeye or chickenpox symptoms.

4) Add a pooling layer

The pooling layer helps to further feature extraction by reducing its dimensionality (i.e. number of features) [21].

5) Add a flatten layer

"Flattening is a technique that transforms a two-dimensional matrix into a vector that can be fed into a fully connected neural network classifier" [21]. In other words, the data is "flattened" from two dimensions to one (i.e. one long vector of features), that can then be fed into the neural network classifier.

6) Add one or more fully connected layer

This step is performed to classify the images.

7) Compile the model

After adding all the previous methods, we compile our model with some parameters such as an optimizer, which is a process that evaluates the input weights by comparing the prediction and the loss function, and metrics, which are used to score the performance of the CNN [33].

8) Train the model using model fit API

In this step, our model will train to differentiate between normal or illness-free images vs. apparent illness images.

9) Use model prediction API

Finally, the CNN will make a prediction of whether the user has an illness (or not) and provide a percentage of probability/accuracy.

E. *Google Maps API – SymCheck's Live Location Checker*

The Google Maps Platform API was utilized in the SymCheck application for the purposes of providing users real-time and real-location depictions of the nearest healthcare facilities around them. Google Maps itself is a very popular and well-established map platform, with the added capability of remaining extremely up-to-date compared to its competitors. Additionally, the API works across multiple platforms (i.e. web, mobile [Apple, Android], etc.), integrating location, routes, points of interest, etc. And its integration into applications is particularly straightforward, and easy to work through and troubleshoot issues due to the presence of a large community that utilizes it [13]. For these reasons, as well as due to cost considerations, Google Maps was the team's top choice for an integrated live map API.

F. *Dialogflow – SymCheck's Live Chat*

Dialogflow, created by Google, is a natural language AI platform that makes it easy to design and integrate a conversational user interface into web applications, mobile apps, and more [10]. Additionally, it is able to analyze various types of input including text and audio. The bot, in turn, can then respond to users through text or even synthetic speech. Figure 7 showcases a quick graphic describing how Dialogflow works [7].

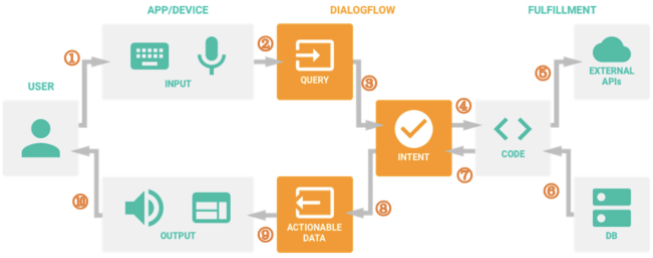


Fig. 7 Dialogflow Steps [7]

1) *A User sends a text/voice message to a Device or an App*
2) *The App/Device transfers the message to Dialogflow*
3) *The message is categorized and matched to a corresponding Intent (Intents are defined manually by developers implementing Dialogflow, i.e. through the use of keywords)*
4) *Following actions are predefined by developers for each intent in the fulfillment phase (Webhook).*

5) *When a certain intent is determined by Dialogflow, the webhook uses External APIs to find a response from external Databases*
6) *The external Databases send back all required information to the webhook*
7) *The webhook sends a formatted response [back] to the Intent*
8) *The Intent generates Actionable Data to different, required channels*
9) *The Actionable Data go to their output Apps/Devices.*
10) *The user receives his/her respective text/image/voice response*

G. Why DialogFlow?

We chose Dialogflow for the sake of building the SymCheck Chatbot for several reasons. Firstly, its Standard Edition is completely free, and accomplishes everything required for the execution of the SymCheck application's MVP—i.e. basic request/response of information about SymCheck. Secondly, the integration of Dialogflow into the Django framework is relatively straightforward, requiring only a few lines of HTML and JavaScript code. And lastly, the user interface of Dialogflow is simple and easy to navigate. We were able to create various intents as needed without changing any code on SymCheck's front-end, leaving room for potential expansion in the future as well.

H. Repl.it – Hosting SymCheck on the Cloud

Repl.it is an interactive, collaborative programming environment hosted on the web. You can create a workspace in a number of languages such as Python, Ruby, or C, and can initialize the use of templates such as Django or React.js as well. The workspace essentially serves to provide a container on a virtual machine where your code can run [17]. And in any given repl instance, there are two main parts, namely, the editor and the console. The editor uses the same technology that powers Visual Studio Code, and the console is where the standard output appears [17].

I. Why Repl.it?

One of the tools required for the development of SymCheck was a collaborative coding environment that supported the Django framework. This need was met with Repl.it, as it was the *only* open-source tool we were able to find with extensive research to meet that specific need. In addition to this feature, Repl.it also provided web hosting capabilities, which is something we had already been considering in terms of being able to host the SymCheck application on the cloud for users' convenience. Thus repl.it became a natural choice for SymCheck as we learned more and more about how to use the software. Each repl instance is actually hosted using the following URL pattern: `https://REPL-NAME--USERNAME.repl.co`. And any changes made to the repl instance are automatically updated, as long as it is consistently up and running [24].

All things considered however, the biggest advantage of using Repl.it for our SymCheck application over other Cloud Service Providers or CSPs is that Repl.it has an editor along with a console that is *collaborative* in nature. This feature is not found in other major cloud solutions such as AWS (Amazon Web Services) or the Google Cloud Platform. For example, in AWS, users have to edit their code via the command line of their instance rather than from within a convenient editor with their colleagues in real time.

However, as our team's project continued to grow, halfway through the semester, we encountered storage issues as Repl.it's servers could no longer handle running SymCheck due to the size of the TensorFlow library. Thus, in the meantime, we turned to Amazon Web Services (AWS) as a fallback, and continued developing SymCheck using AWS Cloud 9 for collaboration (described in section N following). But then, close to the end of the semester, perhaps as a direct result of our team's concerns and appeals during the process, the Repl.it team upgraded their servers, thus allowing our then completed application to run on their servers again! As a result, in the end, our team decided to allow for users to access SymCheck through the repl.it instance for the foreseeable future due to Repl.it's lower costs, ease of installation and access, and fewer infrastructure issues (as compared to AWS).

J. Amazon Web Services (AWS) EC2

As described in the previous section, when our team encountered cloud-deployment issues due to server speed and storage on Repl.it, we turned to Amazon Web Services (AWS)'s EC2. EC2 is essentially a Cloud service run by Amazon that provides resizable compute capacities, easily controlled/manipulated and billed only as they are used, with the added convenience of being available in every Amazon region. Using different types of instances, users are able to store, develop, and run various applications, such as SymCheck [5].

For the purposes of compiling, deploying, and storing SymCheck, a m5.large EC2 instance with 2 virtual CPUS and 8 GiB of memory was used—this large instance was required due to the same issue of having to install the large TensorFlow library which initially broke our Repl.it instance as spoken to in the previous section. And AWS Elastic Beanstalk was utilized behind the scenes of the instance in order to store and run a load-balanced EC2 for deploying the SymCheck site.

K. AWS Elastic Beanstalk

AWS Elastic Beanstalk is a service provided by Amazon Web Services for the purposes of deploying, running, and scaling web applications, and as mentioned in the previous

section, it works behind the scenes of our main EC2 instance for running SymCheck [3]. In fact, in working to load balance our EC2, Elastic Beanstalk allows for SymCheck to run on auto-scaling so that when there is more traffic on the site, it initializes more EC2 instances to run the site.

In order to run SymCheck, which uses Django on Elastic Beanstalk, we had to create a virtual environment within the command line and then download/save all Django and TensorFlow files within it. Only then were we able to upload the actual SymCheck application in order to ensure that the application was running properly. With the application running, thus it was made so that when the user clicks the website URL, the instance connects the Load Balancer back to Elastic Beanstalk.

L. AWS Certificate Manager (ACM)

AWS Certificate Manager is another resource made available by Amazon Web Services for the purposes of providing free public certificates to integrate Secure Sockets Layer (SSL)/Transport Layer Security (TSL) certificates into AWS-connected resources; SSL/TLS certificates also create secure connections between browsers and applications on user applications and establish their identity—both very important consideration with regards to a medical application such as SymCheck. And once the certificate is created, ACM takes care of the rest and deploys the certificate to the website [2].

On top of security concerns, our SymCheck application required the use of ACM in order to give it a more professional, registered domain name using AWS Route 53 (described in the next section) led with https. And ACM specifically was chosen for this task simply considering its ease of use and integration with an AWS-deployed instance.

M. Amazon Route 53

Amazon Route 53 is a reliable and cost-efficient Cloud Domain Name System or (DNS) that allows for Domain Name Registration with the added capability of *automatically* configuring the DNS for the domain (as opposed to requiring an IP address). This is done in three steps:

1. *Register Domain Name* – we registered our application as symcheckapp.com
2. *Route Internet Traffic* to resources for your website – when a user enters symcheckapp.com into his or her address bar, Route 53 helps connect his or her browser to our SymCheck application site.
3. *Check resource health* – requests are sent over the internet to our server in order to verify that the site is reachable, available, and functional at any given point in time.

So Route 53 essentially works by first connecting to the domain name, and subsequently sending the user to the correct Application Load Balancer, which, as mentioned earlier, works to automatically distribute application traffic across multiple targets using automatic scaling via connection to Elastic Beanstalk Container [1].

N. Amazon Web Services (AWS) via Cloud9 – Code Collaboration Tool

AWS Cloud9 is a cloud-based code collaboration tool hosted by Amazon Web Services which enables users to, essentially, code with numerous team members in real time in a variety of languages including JavaScript, Python, PHP, etc. All programmers require in order to do this, are an internet connection and a browser of choice. Cloud9 also comes with a terminal that works with Amazon EC2 or AWS' Elastic Compute Cloud, which, at its barest function is essentially a resizable (hence elastic) amount of secure space within the cloud in which computing can be carried out. Given this, Cloud9 can also directly access the numerous features and services offered by AWS, many of which were utilized for the storage, compilation, and running of SymCheck as described in earlier sections [5] [6].

O. Why AWS & Cloud 9?

As mentioned earlier, SymCheck was originally being developed using an interactive coding collaboration environment known as Repl.it which essentially allows users to create a workspace in which a number of languages such as Python, Ruby, or C can be used, and templates such as Django or React.js can be initialized as well. However, the team hit a blip when the instance itself began throwing disk space errors due to the size of TensorFlow's instantiation (an integral part of running the SymCheck application). And after an appeal to the Repl.it team on resolving this issue showed that a resolution was uncertain, it was decided that another platform should be sought out as the developers over at Repl.it continue working on ameliorating said issue.

Thus our team then turned to AWS' Cloud9. Given the support Cloud9 shows for various applications in various languages, as well as the fact that the coding collaboration tool allows access to a Linux terminal (thereby allowing for the execution of applications using Django), Cloud9 stood as a great alternative to Repl.it, also allowing the team to overcome the issue of exceeding disk space in the initialization of TensorFlow via access to AWS features as well as EC2 as mentioned in a previous sub-section [5] [6]. Thus Cloud 9, as well as all the other technologies at our disposal upon introducing AWS to the mix, allowed for the team to finally successfully deploy the complete SymCheck application on the Cloud for convenient user use.

However, as mentioned in an earlier section, near the end of the semester, the original issues that had forced our team to AWS as a fallback from Repl.it, were resolved when the Repl.it team upgraded their servers. And thus our team is now

left with *two* potential Cloud hostings of SymCheck, with plans to shut down the AWS instance in the near future and only continue running the Repl.it one due to the vast difference in costs between the two. Repl.it also stands as the more convenient option between the two due to its simpler, and thus less conflicting infrastructure when consideration integration for Django-deployed applications.

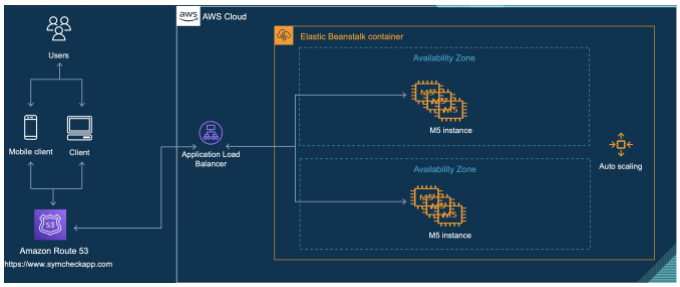


Fig. 8 Diagram of SymCheck AWS Architecture (using all aforementioned components)

P. Agile Methodologies – SCRUM

Last but never the least, throughout the course of this entire project, our team learned about and increasingly employed methods of Agile development, and techniques of SCRUM. Agile development essentially involves understanding requirements and developing solutions through high levels of collaboration between development teams and product consumers. And it focuses on *adaptive* planning, and thus delivery of small consumable chunks in short, productive increments of time (known as Sprints). Agile is built on 12 fundamental principles that make up the Agile Manifesto, and focuses on 4 values that stress *software* over documentation, collaboration and interaction, and flexibility to change [11].

Moreover, SCRUM is a subset of Agile, and is essentially a framework for development that utilizes the aforementioned principles of development in an agile mindset. It helps teams work together by outlining sets of meetings, tools, and roles that work within a self-organizing team in order to deliver high quality, scalable products, with increased efficiency. [4][34]

VII. Product Results And Evaluation

At this current point in time, SymCheck's machine-learning algorithm for image classication has been trained to validate chickenpox and pinkeye with an accuracy of about 88%. Originally, the machine learning model was trained on grayscale images instead of color ones, but this created an error where chickenpox was at times being classified as normal skin (with a high percentage of certainty), and thus the algorithm was updated to use color images instead for a potentially better measure of accuracy. This error continued to persist at times throughout the second semester, however, as more and mire images were trained to the model, the accuracy of the model's classification for both chickenpox and pinkeye increased definitively.

As for the user interface to date, Figures 8 through 20 show the developed designs of SymCheck's HTML pages.

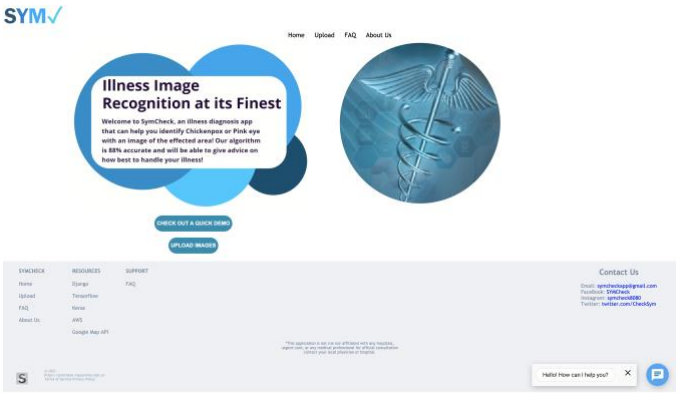


Fig. 9 Screenshot of the homepage of SymCheck

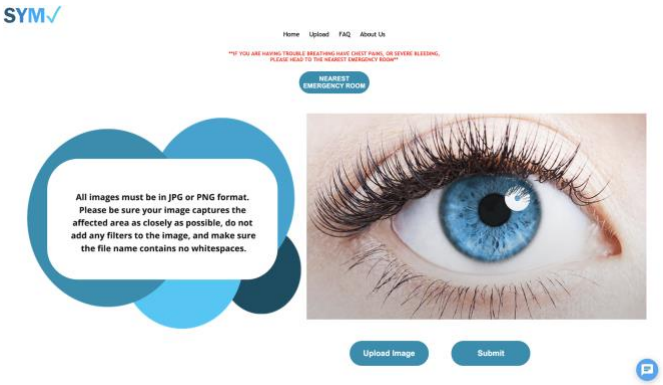


Fig. 10 Screenshot of Image Upload Page with uploaded image preview

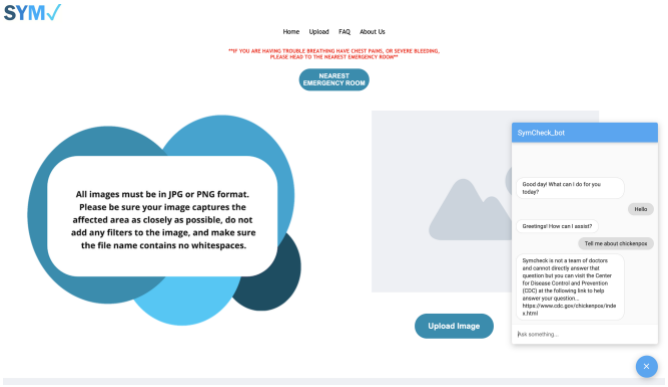


Fig. 11 Screenshot of Image Upload Page with Chatbot open (chatbot available on all pages of the application)

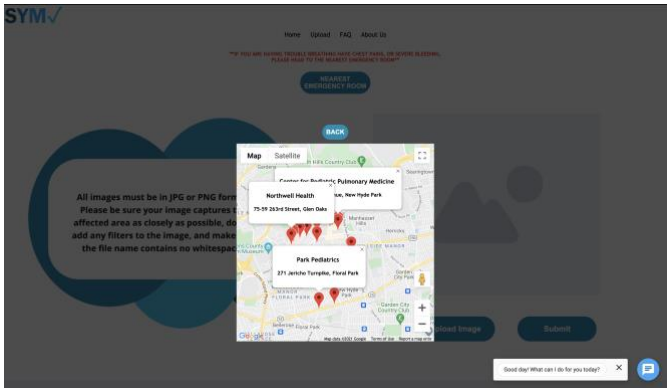


Fig. 12 Screenshot of Image Upload page with map open showing nearest healthcare facilities

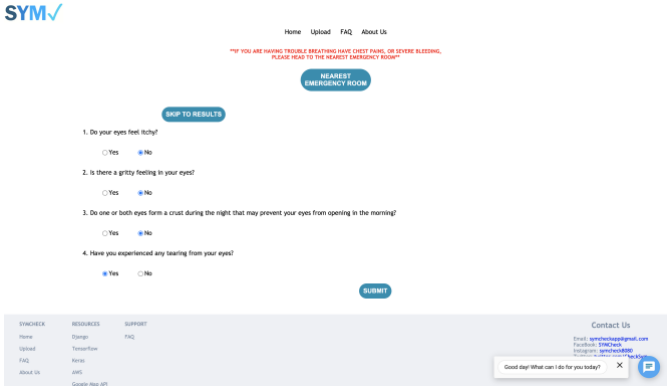


Fig. 13 Screenshot of Advanced Prediction Page

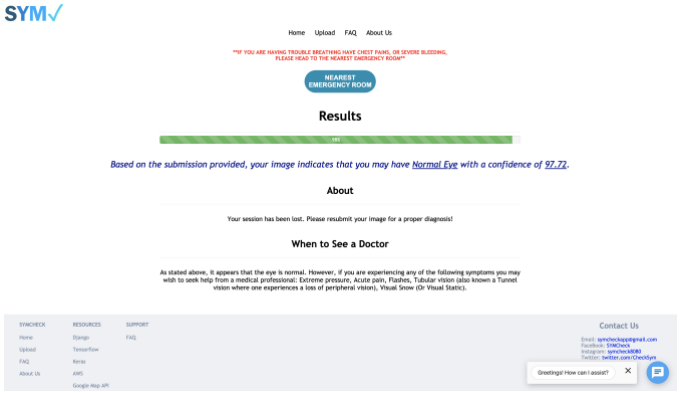


Fig. 14 Screenshot of Results page

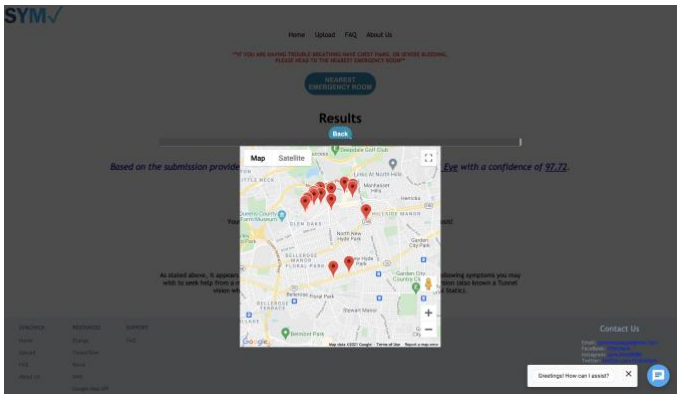


Fig. 15 Screenshot of Results page with map open showing nearest healthcare facilities

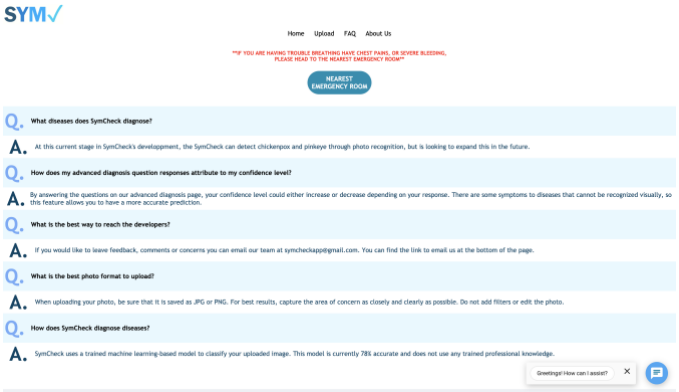


Fig. 16 Screenshot of SymCheck's FAQ page

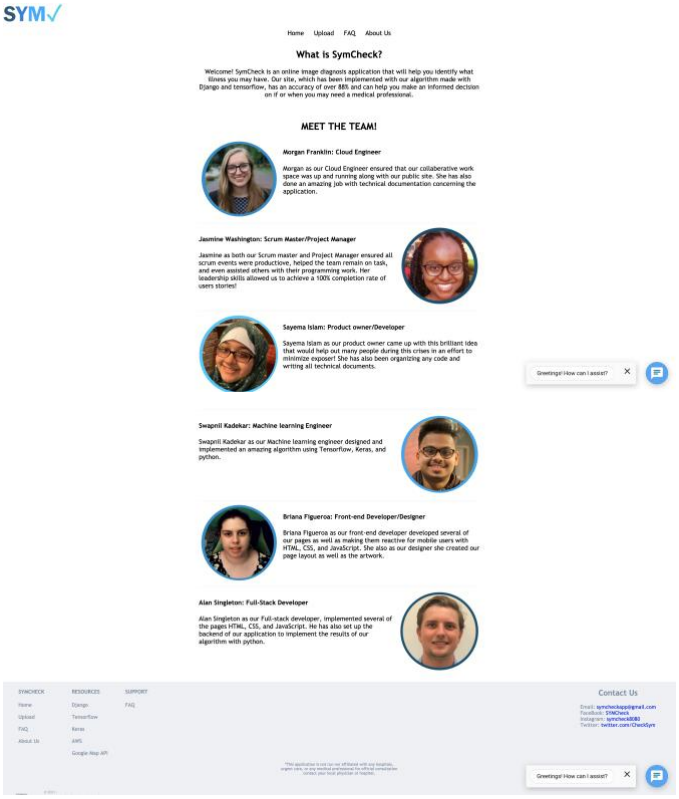
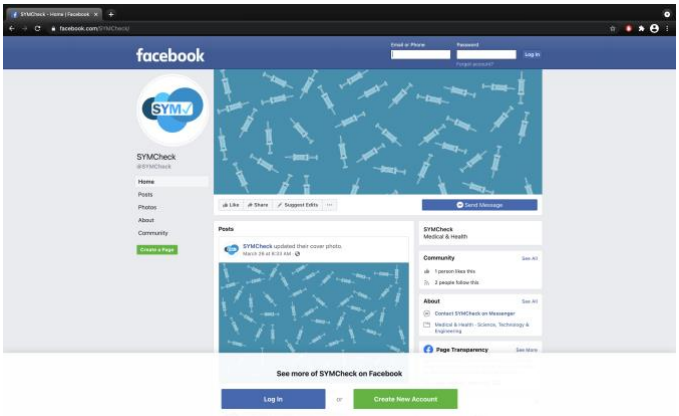


Fig. 17-18 Screenshots of About Us page



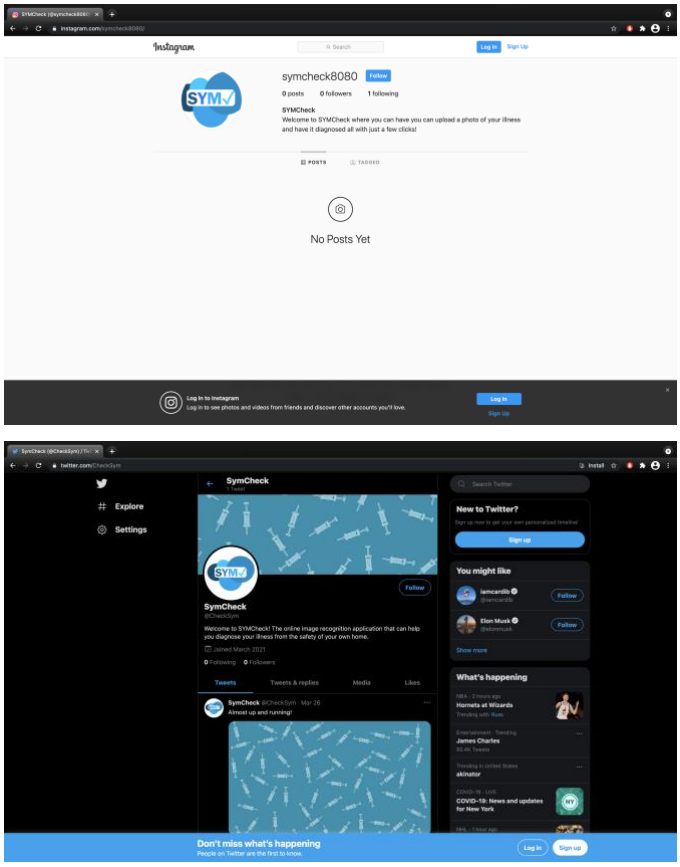


Fig. 19-21 Screenshots of SymCheck's Social media pages (Facebook, Instagram, Twitter)

In the end, after two semesters and eight Sprints of work on the application, SymCheck stands as a novel application with true potential in making a difference in society. The application capitalizes on its unique functionality of allowing users to receive cursory diagnoses through simple *image* inputs, without the lengthy or bothersome creation of an account or answering long lists of mandatory, tedious questions. Instead, SymCheck saves its users time and money by allowing patients to simply upload an image to their browser or mobile device, with an added *option* to opt in or out of answering a short list of yes or no questions (simply for a *more* accurate final determination). The application's strengths lie in its speed of delivery with simple input, its easy-to-use site with fairly straightforward functionalities, its ability to decrease physical traffic in healthcare facilities (an important consideration in the time of a pandemic requiring social distancing), as well as its ability to give more power to its users in making more *informed* decisions about their own care.

However, the final SymCheck application to date does have a lot of potential for improvement as well. For one thing, the application could definitely be more secure when considering the nature of the application as one that provides *medical* diagnoses. This is something the team is considering in the future by perhaps implementing a secure database (as opposed to a static media folder) to store patient images, as well as or solely through the use of a script to delete any stored images after a set amount of time. Additionally, the number of illnesses/ailments checked on SymCheck could definitely be extended moving forward in order to make the application more useful and versatile to a wider range of users. In the future, the team is considering other non-life-threatening illnesses/ailments such as ringworm, eczema, athlete's foot, etc. Moreover, currently, the application only accepts .png and .jpg/.jpeg image files; this can be expanded as well in the future to include more image formats (i.e. .heic for Apple mobile device users), with an added functionality to properly validate all inputs for image upload.

SymCheck remains a complete working application, but with an open mind to grow and improve in order to provide a better quality of life to all its users.

VIII. Conclusion

SymCheck is a web application that currently serves as a tool to provide users with a means of self-diagnosis for common illnesses such as chickenpox and pinkeye. This application strives to fulfill this aim by relying on an original machine learning-based algorithm for image classification, that is able to feed its results directly to a user. And an option for completing an advanced diagnosis with the addition of textual-based inputs on top of solely, image-based classifications adds another layer of accuracy for those users who might desire a "better" prediction. In implementing all these features and functionalities, SymCheck has, and continues to strive to become a fast and convenient method of self-diagnosis that allows its users to take more control of their health, and understand when physical visits to healthcare facilities should be prioritized, or can be avoided altogether—an especially important consideration in the scope of the worldwide COVID-19 pandemic.

Future scopes of this project are already well underway, including the addition of other non-life-threatening illnesses such as ringworm, as well as perfecting the responsive web pages that allow for SymCheck to be hosted as a mobile web application as well. And the SymCheck team continues to seek to improve the accuracy of the machine-learning algorithm for image classification by continuing to add even more image data for training the model.

References

[1] Amazon Web Services, Inc., "Amazon Route 53," *AWS*. [Online]. Available: https://aws.amazon.com/route53/. [Accessed: 23-Apr-2021].

[2] Amazon Web Services, Inc., "AWS Certificate Manager," *AWS*. [Online]. Available: https://aws.amazon.com/certificate-manager/. [Accessed: 23-Apr-2021].

[3] Amazon Web Services, Inc., "AWS Elastic Beanstalk," *AWS*. [Online]. Available: https://aws.amazon.com/elasticbeanstalk/. [Accessed: 23-Apr-2021].

[4] Atlassian, "Scrum - what it is, how it works, and why it's awesome," *Atlassian | Agile Coach*. [Online]. Available: https://www.atlassian.com/agile/scrum. [Accessed: 25-Apr-2021].

[5] "Amazon EC2," *AWS*, 2021. [Online]. Available: https://aws.amazon.com/ec2/?ec2-whats-new.sort-by=item.additionalFields.postDateTime&ec2-whats-new.sort-order=desc. [Accessed: 13-Mar-2021].

[6] "AWS Cloud9," AWS, 2021. [Online]. Available: https://aws.amazon.com/cloud9/#:~:text=AWS%20Cloud9%20is%20a%20cloud,code%20with%20just%20a%20browser.&text=With%20Cloud9%2C%20you%20can%20quickly,other's%20input

s%20in%20real%20time. [Accessed: 13-Mar-2021].
[7] H. Chen Data Scientist and H. C. D. S. C. D. S. D. M. LearningNLP, “A brief introduction to Chatbots with Dialogflow,” *Margo*, 05-Jul-2018. [Online]. Available: https://www.margo-group.com/en/news/a-brief-introduction-to-chatbots-with-dialogflow/. [Accessed: 22-Feb-2021].
[8] “Django Tutorial for Beginners - Learn the Core Aspects of Django Framework,” *DataFlair*, 14-Jan-2020. [Online]. Available: https://data-flair.training/blogs/django-tutorial/. [Accessed: 26-Oct-2020].
[9] *Django*. [Online]. Available: https://www.djangoproject.com/. [Accessed: 26-Oct-2020].
[10] “Dialogflow ES documentation | Google Cloud,” *Google*. [Online]. Available: https://cloud.google.com/dialogflow/es/docs. [Accessed: 22-Feb-2021].
[11] K. Eby, “Comprehensive Guide to the Agile Manifesto,” *Smartsheet*. [Online]. Available: https://www.smartsheet.com/comprehensive-guide-values-principles-agile-manifesto. [Accessed: 25-Apr-2021].
[12] “5 Genius Python Deep Learning Libraries,” *EliteDataScience*, 09-Jun-2020. [Online]. Available: https://elitedatascience.com/python-deep-learning-libraries. [Accessed: 26-Oct-2020].
[13] “Google Maps Platform,” *Google Maps Platform | Google Developers*. [Online]. Available: https://developers.google.com/maps. [Accessed: 25-Apr-2021].
[14] T. Goswami, V. K. Dabhi, and H. B. Prajapati, “Skin Disease Classification from Image - A Survey,” *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 2020, pp. 599–605.
[15] N. Hameed, A. Shabut, F. Hameed, S. Cirestea, and A. Hossain, “An Intelligent Inflammatory Skin Lesions Classification Scheme for Mobile Devices,” *2019 International Conference on Computing, Electronics & Communications Engineering (iCCECE)*, 2019, pp. 83–88.
[16] M. Heller, “What is Keras? The deep neural network API explained,” *InfoWorld*, 28-Jan-2019. [Online]. Available: https://www.infoworld.com/article/3336192/what-is-keras-the-deep-neural-network-api-explained.html. [Accessed: 26-Oct-2020].
[17] “Introduction to Repls,” *Repl.it*. [Online]. Available: https://docs.repl.it/repls/intro. [Accessed: 22-Feb-2021].
[18] M. N. Islam, J. Gallardo-Alvarado, M. Abu, N. A. Salman, S. P. Rengan, and S. Said, “Skin Disease Recognition Using Texture Analysis,” *2017 IEEE 8th Control and System Graduate Research Colloquium (ICSGRC)*, August 2017, pp. 144–148.
[19] M. D. Manchalwar and K. K. Warhade, “Histogram of Oriented Gradient based Automatic Detection of Eye Diseases,” *2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA)*, 2017.
[20] “Symptom Checker,” Mayo Clinic. [Online]. Available: https://www.mayoclinic.org/symptom-checker/select-symptom/itt-20009075. [Accessed: 26-Oct-2020].
[21] “Using the Keras Flatten Operation in CNN Models with Code Examples,” *MissingLink.ai*. [Online]. Available: https://missinglink.ai/guides/keras/using-keras-flatten-operation-cnn-models-code-examples/. [Accessed: 26-Oct-2020].
[22] A. Namozov and Y. I. Cho, “Convolutional Neural Network Algorithm with Parameterized Activation Function for Melanoma Classification,” *2018 International Conference on Information and Communication Technology Convergence (ICTC)*, 2018, pp. 417–419.
[23] A. Rajesh, “Classification of malignant melanoma and Benign Skin Lesion by using back propagation neural network and ABCD rule,” *2017 IEEE International Conference on Electrical, Instrumentation and Communication Engineering (ICEICE)*, 2017.
[24] “Web Hosting,” *Repl.it*. [Online]. Available: https://docs.repl.it/repls/web-hosting. [Accessed: 22-Feb-2021].
[25] G. Schaefer, T. Nakashima, Y. Yokota, and H. Ishibuchi, “Cost-Sensitive Fuzzy Classification for Medical Diagnosis,” *2007 IEEE Symposium on Computational Intelligence and Bioinformatics and Computational Biology*, 2007, pp. 312–316.
[26] “AskMD - Get Answers - Manage Conditions and Symptoms,” Sharecare. [Online]. Available: https://www.sharecare.com/askmd/get-started. [Accessed: 26-Oct-2020].
[27] W. Shiying, Z. Lei, W. Lu, C. Nanjia, Z. Xiaolan, L. Hong, and W. Xiaoying, “Research on Syndrome Classification Prediction Model of Tibetan Medicine Diagnosis and Treatment Based on Data Mining,” *2016 12th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, 2016, pp. 497–502.
[28] SolvHealth, “What are the Most Common Urgent Care Conditions?,” *Solvhealth.com*. [Online]. Available: https://www.solvhealth.com/faq/what-are-the-most-common-conditions-treated-at-urgent-care. [Accessed: 23-Oct-2020].
[29] Shubham, “Flask vs Django: Which Python Framework is best for Machine Learning app?,” *SpiderPosts*, 10-Sep-2019. [Online]. Available: https://www.spiderposts.com/2019/06/07/flask-vs-django-which-python-framework-is-best-for-machine-learning-app/. [Accessed: 26-Oct-2020].
[30] “Symptomate – Check your symptoms online,” Symptomate â Check your symptoms online. [Online]. Available: https://symptomate.com/. [Accessed: 26-Oct-2020
[31] J. Tamuli, A. Jain, A. V. Dhan, A. Bhan, and M. K. Dutta, “An Image Processing Based Method to Identify and Grade Conjunctivitis Infected Eye According to its Types and Intensity,” *2015 Eighth International Conference on Contemporary Computing (IC3)*, 2015.
[32] “Django's Structure – A Heretic's Eye View - Python Django,” *The Django Book*, 21-Jun-2020. [Online]. Available: https://djangobook.com/mdj2-django-structure/. [Accessed: 26-Oct-2020].
[33] “Keras - Model Compilation,” *Tutorialspoint*. [Online]. Available: https://www.tutorialspoint.com/keras/keras_model_compilation.htm. [Accessed: 26-Oct-2020].
[34] Various, “Scrum (software development),” *Wikipedia*, 23-Apr-2021. [Online]. Available: https://en.wikipedia.org/wiki/Scrum_(software_development). [Accessed: 25-Apr-2021].
[35] S. Yegulalp, “What is TensorFlow? The machine learning library explained,” *Gale General Onefile*, 18-Jun-2019. [Online]. Available: https://search-ebscohost-com.rlib.pace.edu/login.aspx?direct=true&db=edsggo&AN=edsgcl.589534891&site=eds-live&scope=site. [Accessed: 26-Oct-2020].