

AI Allergy Detection

Sdmay24-13

Team 13

Eric Christensen

Zoe Davis

Josh Dutchik

Blake Friemel

Jack Gray

Michael Koopmann

Jihun Yoon

Team Email: sdmay24-13@iastate.edu

Team Website: <https://sdmay24-13.sd.ece.iastate.edu/>

Table of Contents

1 Introduction	3
1.1 Problem and Project Statement	3
1.2 Intended Users and Uses	3
1.3 Greater Context.....	4
2. Revised Design	4
2.1 Requirements (Functional and non-functional)	4
2.2 Engineering Standards	6
2.3 Security Concerns and Countermeasures	6
2.4 Description of Design Evolution (Since 491)	7
3. Implementation Details	8
3.1 Detailed Design.....	8
3.2 Description of Functionality	12
3.3 Notes on Implementation	13
4. Testing.....	14
4.1 Process	14
4.2 Results	15
5. Broader Context	15
5.1 Public health, safety, and welfare	15
5.2 Global, cultural, and social	15
5.3 Environmental.....	16
5.4 Economic	16
6. Conclusion.....	16
6.1 Review Progress	16
6.2 Future Steps.....	16
6.3 Appendices	17
Appendix 1: Operation Manual	17
Appendix 2: Alternative version of Initial Design	23
Appendix 3: Other Considerations	25
Appendix 4: Code.....	25
6.4 References	25

1 Introduction

1.1 PROBLEM AND PROJECT STATEMENT

This project addresses the challenge of optimizing medical treatments for individuals with allergies. In traditional healthcare, doctors face the task of considering a patient's unique medical history when prescribing treatments. Allergic reactions to medications can be unpredictable, posing risks to patient health. This AI-driven solution aims to streamline this process by leveraging a comprehensive dataset of patient information and skin allergies. By identifying correlations between individual profiles and the chemical compositions of prescribed products, the system assists doctors in making more informed decisions, ultimately minimizing the potential for allergic reactions and enhancing the overall effectiveness of medical treatments. In essence, it seeks to provide a personalized and data-driven approach to healthcare, promoting patient well-being through a more nuanced understanding of individual health factors while remaining non-invasive in nature.

1.2 INTENDED USERS AND USES

Doctors/Medical Personnel:

In a modern healthcare setting, the integration of AI technology revolutionizes the way medical professionals can now approach treatment planning. The doctor can run a non-invasive test using patient input data, and the AI will output a list of allergens the patient is over 70% likely to have an allergic reaction to. This information is a valuable tool to the doctor's decision-making process, enabling the doctor to tailor the patient's treatment plan with heightened focus on patient safety and efficiency. These users can:

- Log in
- Search a patient by username
- View model results
- Run an analysis to receive a list of products containing allergens
- Sign out

Patients:

The patient experiences the benefits of this advanced approach to healthcare. The patient can input their allergy and medical history into the user interface from the comfort of their home. Each piece of patient data can then be meticulously considered by the trained model. The AI generates a personalized list of allergens reflecting the potential allergic reaction risk. The result not only considers the complexities of the patient's medical history, but also considers correlations identified by the AI. Once approved by the patient's doctor, the results can then be released to be viewed. As a result, the patient gains information about what additional treatment/diagnosis options are necessary while not having to see a doctor face-to-face and still maintaining confidence in their treatment.

These users can:

- Input personal data
- Submit survey

1.3 GREATER CONTEXT

Related Products

There are many products similar to ours relating machine learning to the healthcare field that are being utilized by various groups and establishments. Some of these products include clinical decision support systems, drug interaction prediction systems, and healthcare data analytics platforms. Like these other products, our solution utilizes emerging and adaptable technologies to personalize medical treatment and reduce human risk when prescribing medications. However, what sets our product apart is the specificity on the relation between skin allergies and existing conditions by integrating comprehensive patient data and allergy profiles. By collecting and utilizing such data, we enable our model to have a more accurate understanding of the correlations between various genetic and geological factors to potential allergens. Our solution empowers medical professionals by allowing more certainty in their decisions and minimizes associated risks through a personalized and tailored approach which caters to the specific needs of allergy-prone individuals.

Related Literature

Literature with principles and advancements in the emerging realm of technologically assisted diagnostics in the healthcare realm. With machine learning and artificial intelligence becoming more prevalent in various fields, literature has adapted and provided guidelines for their use cases. Furthermore, an emphasis on the precision of programs of these literatures that we derived for this product was Collins and Varmus (2015), who advocate for ensuring medical treatments are tailored for individual patients based on their genetic characteristics and specific health profile. Likewise, our approach with this project aligned closely with the field of pharmacovigilance which is covered by studies such as Bates et al. (2003), which pertained the role of artificial intelligence in assisting healthcare professionals in evidence-based decisions with prescriptions and risk-assessment. By utilizing the emphases from these studies, we were able to guide our product to align with these standards and produce an overall safer and more effective product for healthcare professionals to utilize in the diagnosis process. We also used literature to help us figure out what accuracy level our AI model needed to be at. According to the Allergy Asthma Clin Immunol medical journal entry “Diagnostic accuracy of skin-prick testing for allergic rhinitis: a systematic review and meta-analysis” by Nevis, Binkley, and Kabali, found that “a stand-alone skin-prick test for diagnosing allergic rhinitis with sensitivity ranging from 60 to 79% and specificity ranging from 68 to 69%”. This medical journal shows that skin-prick tests aren’t always as accurate as we think they are, with the allergic rhinitis skin-prick test only being around 70% accurate. With this information, we decided that we needed the accuracy/precision for the AI to be 71% or better.

2. Revised Design

2.1 REQUIREMENTS (FUNCTIONAL AND NON-FUNCTIONAL)

Functional Requirements:

Website (React application)

- a. The website must allow patients to...
 - **navigate** to the survey
 - **input** their typed data into the survey
 - **select** from our options provided by our UI
 - **submit** the survey
- b. The website must allow doctor users to...
 - **input** typed data such as username, password, and patient name
 - **login** to their account
 - **search** for a patient
 - **run** the model
 - **view** the results of the model
- c. The website itself must...
 - **display** an interactable GUI
 - **output** patient data
 - **output** predicted allergens
 - **communicate** with the backend to display relevant and accurate pages/information
 - **be hosted** on an ec2 instance

Backend (Node.js Server)

- Sends and receives HTTP requests to and from the Amazon RDS database
- Sends a JSON files to and from the model
- Communicates with the front end

Database (Amazon RDS Database)

- Stores patient, doctor, and product tables
- Sends and receives HTTP requests to and from the backend

AI Model

- Is trained using an excel file
- Inputs and outputs JSON files
- Uses rules of association to predict potential allergic reaction
- Outputs ingredients with over 70% likelihood of allergic reaction
- Outputs products that contain high-risk ingredients
- Is stored on an s3 bucket to be retrieved

Non-Functional Requirements:

Website

- Should be intuitive and easy to navigate
- The survey should reduce the amount of variability added to the data by formatting certain inputs
- Doctor log in must be accessible to only medically licensed individuals
- Must be reliable and have little downtime
- Must be accessible from anywhere in the United States
- Must show accurate and relevant information
- Should be aesthetically pleasing

Backend

- The backend should communicate promptly within a short period of time

Database

- The database fields and tables should be clear and related to their stored variables
- The database is secure and require proper authentication and authorization
- The database is scalable in both vertical and horizontal dimensions
- The database should have reasonable response time and throughput

AI Model

- Maintains a high level of prediction accuracy
- Can be retrained
- Returns results in a timely manner (10 seconds or less)

2.2 ENGINEERING STANDARDS

- ***IEEE 829: Software Test Documentation***
 - This standard was used to determine the appropriate breadth and depth of testing during development and to be maintained.
- ***IEEE 2801-2022: Recommended Practice for the Quality Management of Datasets for Medical Artificial Intelligence***
 - This standard was used to determine the quality objectives for the dataset used in our model.
- ***IEEE 2830-2021: IEEE Standard for Technical Framework and Requirements of Trusted Execution Environment based Shared Machine Learning***
 - This standard helped us determine how to use our obfuscated data and train a model with it.
- ***IEEE 7000-2021: IEEE Standard Model Process for Addressing Ethical Concerns during System Design***
 - This standard gave us ideas on how to maintain transparent communication with stakeholders, with emphasis on ethics.
- ***P7003: Algorithmic Bias Considerations***
 - This standard helped us with understanding how to utilize bias values in tuning our model.

2.3 SECURITY CONCERNS AND COUNTERMEASURES

The security of patients' data as well as system data is of the utmost importance with our product, since it handles and utilizes sensitive information. Keeping this in mind, we integrated various countermeasures to ensure the security of our patients' data. One of the largest security concerns

would be third parties being able to obtain data from our database. In order to prevent these attacks from happening, we implemented authentication and authorization into our program which limit the permissions of various users. With these permissions, we limit the individuals that can view the data to admins and doctors, which significantly increases the security of the accessible data. Additionally, we implemented an input validation and sanitation mechanism which prevents injection attacks on our database. A large concern with the database was ensuring that inputs from either the user or doctor are valid and cannot be utilized to gain information to data through unintended means. By implementing an input validation, we reduce the total input and control the data flow to secure the data stored in the database. Another concern that is apparent is that the database holds sensitive data which can be accessed by both the backend and frontend. In order to combat this, we implemented encryption techniques for both resting data and transmitted data to further protect the information of the patient. Firewalls were implemented as well, which limit the addresses that are able to access the database and increase overall security of our system.

2.4 DESCRIPTION OF DESIGN EVOLUTION (SINCE 491)

Our design has had many changes since our initial prototype in 491, and with that, we further optimized our overall user experience while using our product. One of the largest concerns that we had to keep in mind while developing was how to ensure that on a large-scale system, our product was not consuming too many resources or taking a long time to send back results. With these new considerations, there were several different aspects of our design that we had to change. The first of which was the service we host our database from and the database architecture.

Initially we had planned to have the model train after every instance of a new record in our database to increase the accuracy of the results for the next user. This, on a small-scale system, would not be an issue with resource utility, but for a large-scale system this would be very taxing and utilize a lot of bandwidth and potentially slow down the server significantly. To reduce these costs, the retraining of the model was changed to be manual and on the condition that there are several new instances of patient data in the database, otherwise the training would not be as useful. Another change to reduce costs was to have the trained AI model hosted on the server. This allows the doctor to run their patients through the model without any unnecessary training which reduces the total cost and resources utilized per iteration of our model.

Another way that we changed our design was by making the doctor input only one patient into the model and return a list of ingredients and products that the patient is allergic to. By this being done, we push the scope of our project from not only a single product to more potential products that the patient might be allergic to, thus creating a safer environment for diagnosis and treatment. Additionally, we found that it would be beneficial to get rid of the lambda function and have the model hosted completely on the server, and directly output into our database. This decision made our system more compact and increased the overall security of our system as well by reducing the number of points of vulnerability. Likewise, this change also has benefits with consolidating updates to a smaller software surface area, which reduces the cost of maintenance and resources.

The biggest change by far was the adaptation of the patient role and what a patient means to our service. Before a patient had a huge role with an account and the ability to view other patient's statistics and see doctor information. However, as our product evolved, we started asking questions on how and why exactly a patient account is needed. We wanted to make our service easy and accessible, and we felt that having a user constantly sign up, login, and update their account we risked some of the easiness our project was going for. With that we decided to add more responsibilities to the doctor's role. The doctor was now in charge of figuring out how to access their account and is responsible for getting in touch with their patients. In turn, the patients' only

responsibility was to fill out the survey after contacting their doctor. We felt this decision took some of the pressure off the patient and was truer to our original vision of our product and service.

3. Implementation Details

3.1 DETAILED DESIGN

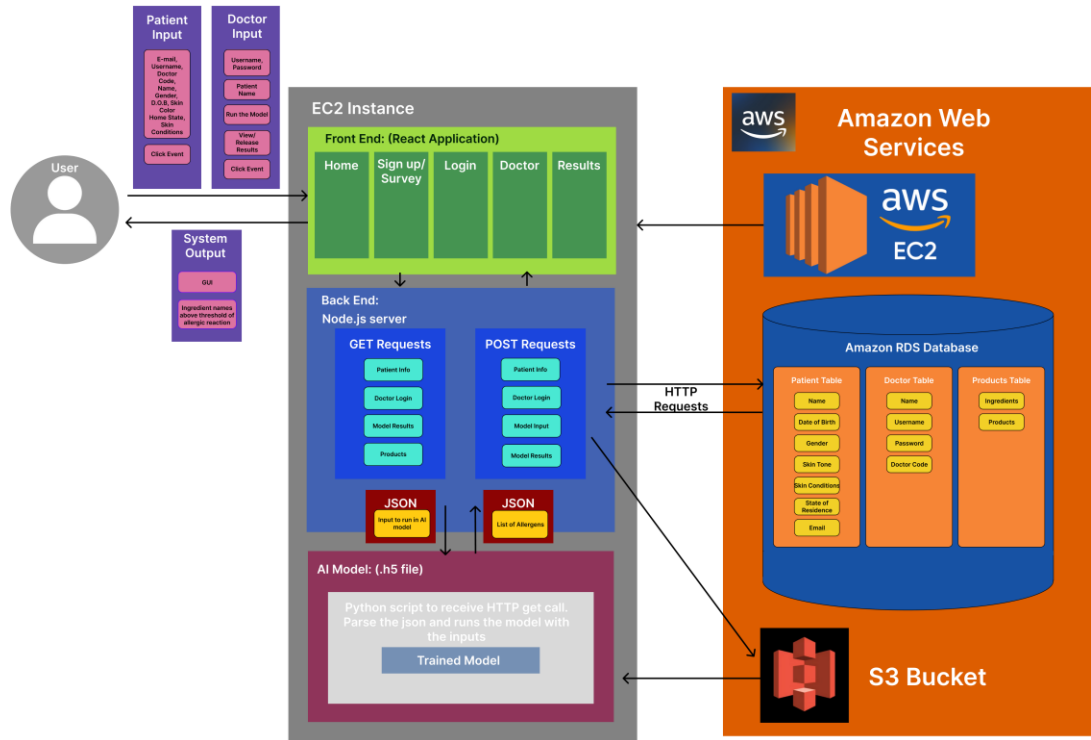


Fig 1. Project Design Overview.

All interaction with our project for users is done through our frontend website, which is hosted on an ec2 instance via AWS. The website is a React application, which allows multiple users to simultaneously utilize our website from a list of known IP addresses. There are two types of users, patients and doctors. The patients can fill out a survey with their information using our UI. Once the survey is submitted, their information is added to the patient tables in our database. The information is then run through our model, and the results are added to the patient's row. The patient has no access to this information, so their experience is finished until the doctor contacts them. The doctor user logs in via username and password on the home page. Once logged in, the doctor can search their patients via patient username, and their information will become visible. The doctor can then analyze the patient's likely allergens, and cross reference them with a list of products containing those allergens for a list of products to avoid for the patient.

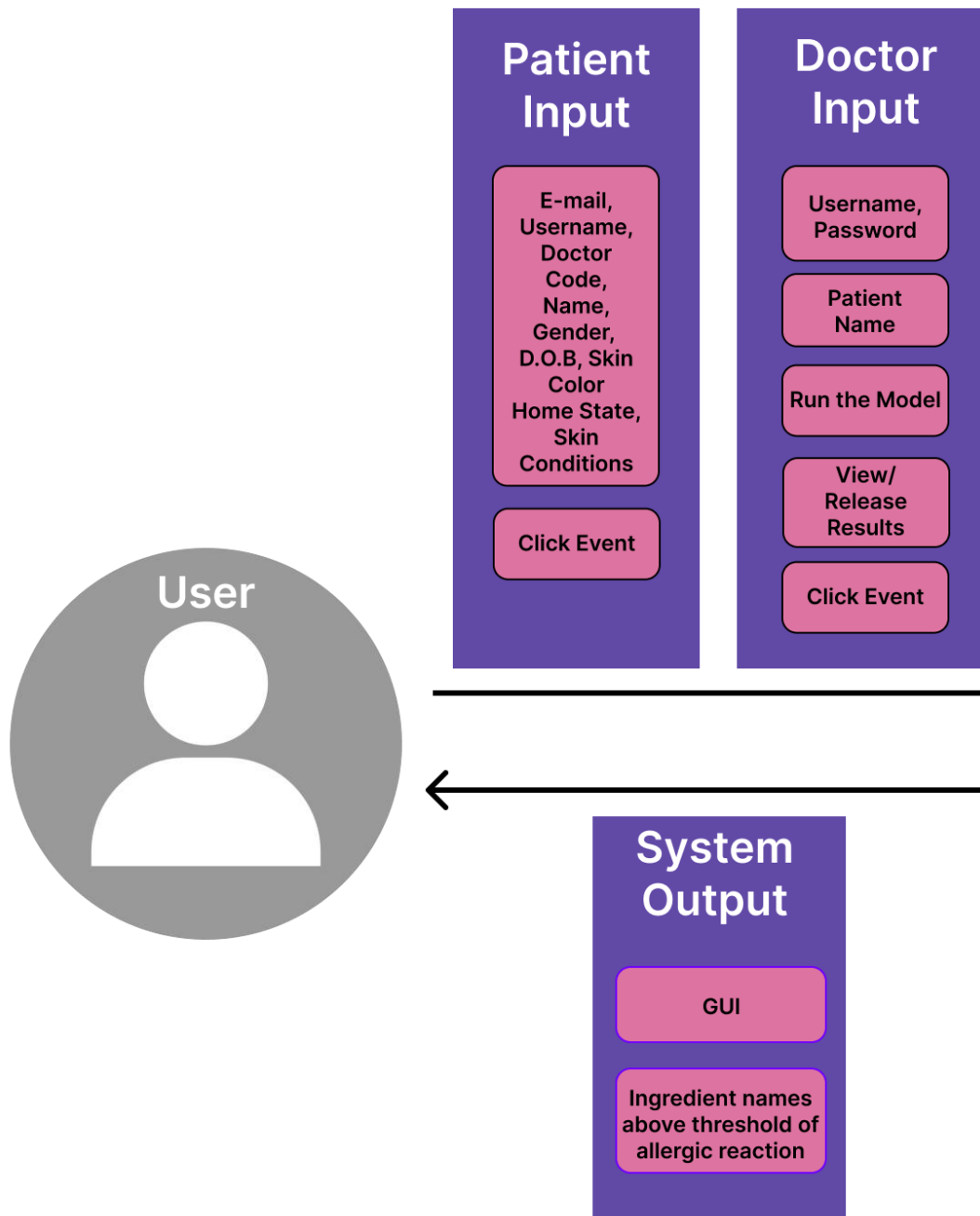


Fig 2. Close up on User Interaction.

The backend is a Node.js server, also hosted on an ec2 instance using an Amazon RDS Database. The backend is managed through MySQL and node.js. It calls the ai model using S3 Bucket and sends and receives data in the form of a JSON file. It uses HTTP GET and POST requests to receive information and update our database, where patient, doctor, and product data is stored in tables. From the frontend we use the 'await fetch' function call that allows to connect to a given URL with a request of a specified type. From there the call is made to the backend to run more code. We use

a connection to connect to our senior design SQL database and send queries over from the backend to return a JSON back to our frontend for more data handling.

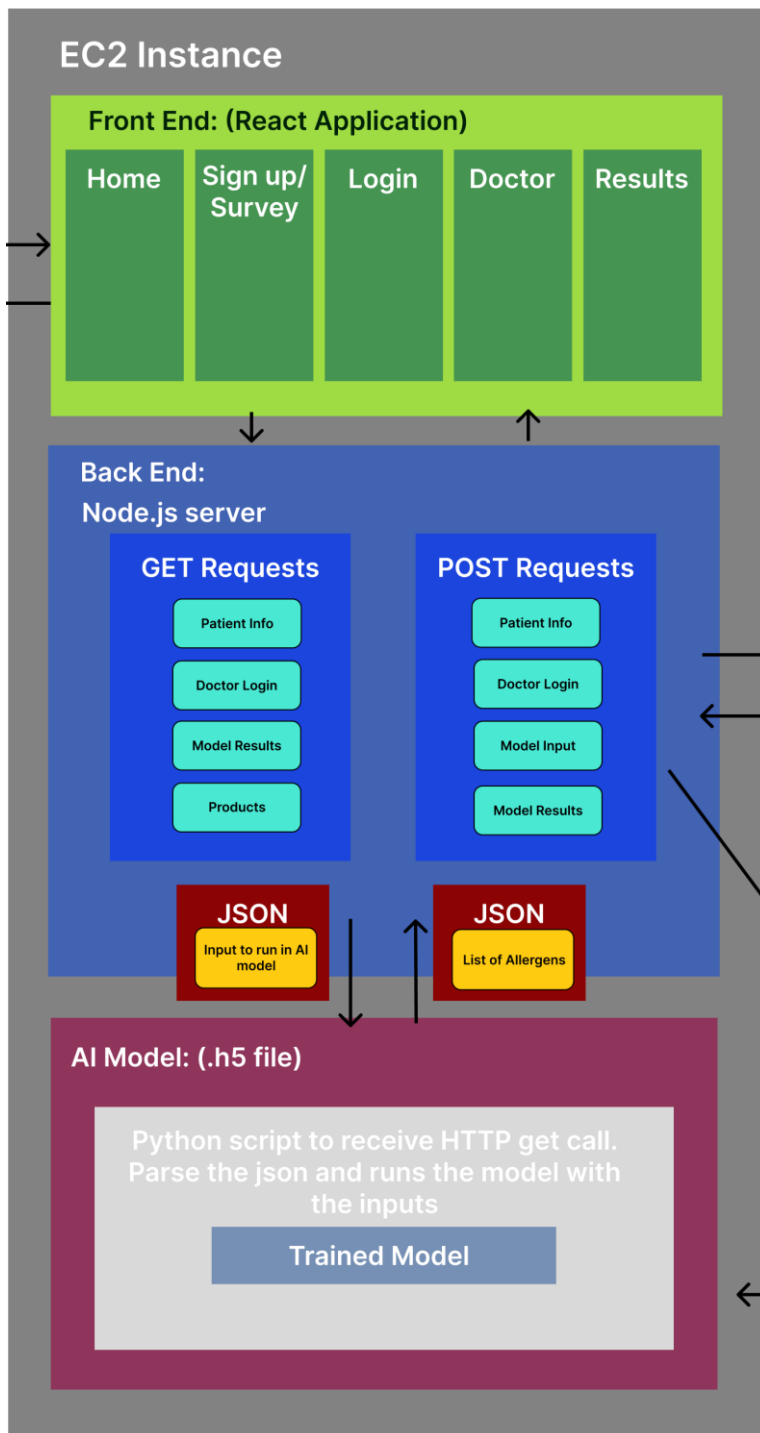


Fig 3. Close up of EC2 Instance.

We used the Keras and TensorFlow libraries to build, compile, and save our models which was all done locally using jupyter notebooks. Once our model was saved, we then used AWS S3 buckets to

store the model. Once the model/predictions are needed, the AWS EC2 instance runs a python script containing the model.

Finally, our group collectively collaborated on this project using Git and GitHub.

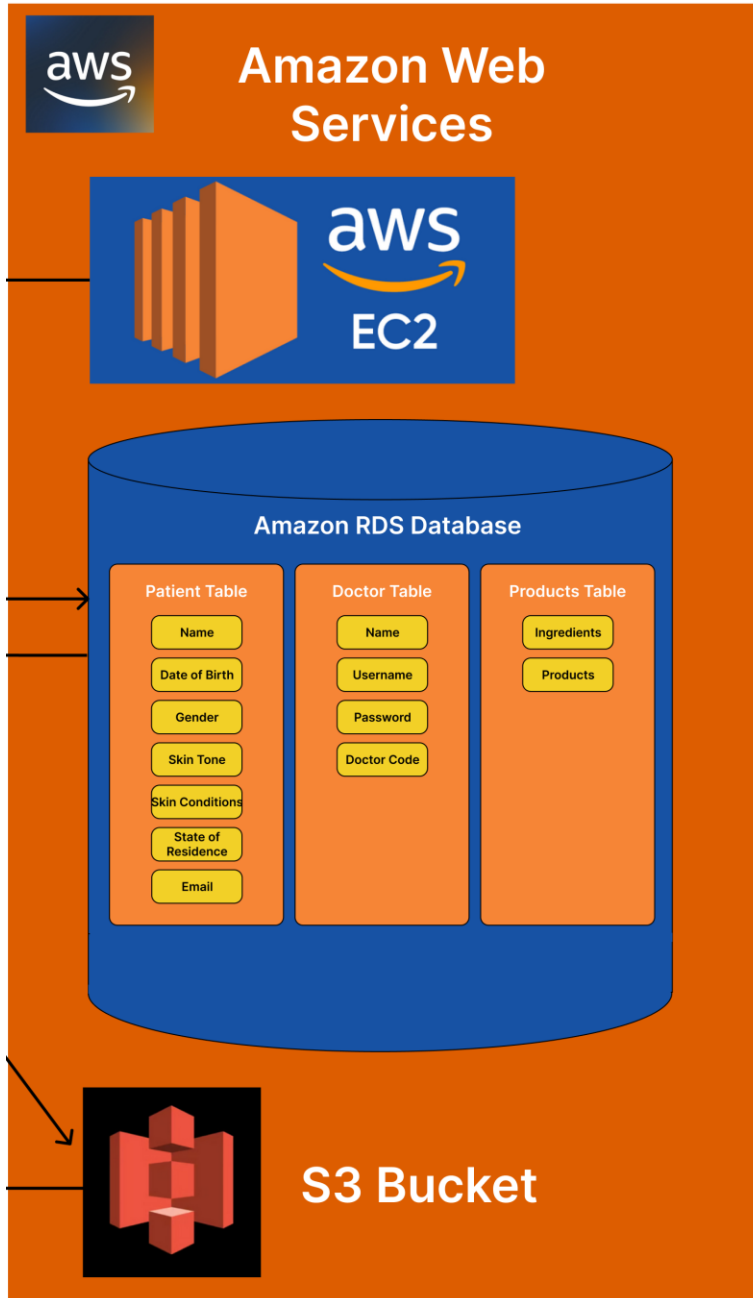


Fig 4. Close up of AWS.

Technologies used:

- Languages
 - Python

- JavaScript
- CSS
- HTML
- *Libraries*
 - React
 - Keras
 - Teras
 - Sklearn
 - Pandas
 - Matplotlib
 - TensorFlow
- *Interfaces*
 - Fetch API
- *Environments*
 - Node.js
- *Platforms*
 - Postman
 - GitHub
- *Software*
 - NumPy
 - SQL Workbench
 - Git

3.2 DESCRIPTION OF FUNCTIONALITY

The first function that users use is to fill in surveys. Only doctors can login because doctors are given a premade account verified by us to avoid misuse of information. Patients need to fill out an online survey. This survey asks them for their username, name, email, date of birth, state they live in, gender, skin tone, skin symptoms, and doctor code. The doctor code has been given to the doctor by us who in turns gives it back to the patient. Entering the correct doctor code is important because the code connects the doctor to the patient and allows permission to access the patient's data. Once the survey is done, the patient's role is now complete. Now the doctor needs to be involved as their role is the only one that's allowed to access patient data and the AI results.

On the website, doctors start by logging in. Once logged in, they are taken to their profile. Their profile includes their name and doctor code that can be shared with patients. There's also a search function at the top of the page that allows them to search for a patient so they can look at that patient's data, but they only have access to data if patients entered their doctor code in the patient survey. Once they find the patient they're looking for, that patient's profile pops up. The patient profile includes the patient's name, email, date of birth, state they live in, gender, skin tone, symptoms, and current known allergens. On this page, they can also input the patient's data into the AI and see the output the AI provides. The artificial intelligence model provides the doctor with ingredients that it predicts the patient may be likely to be allergic to. Using the ingredients given, we can parse through a list of given medical and consumer products and the ingredients they contain. We return the list of products to avoid going back to the doctor to not only keep in mind

when prescribing medications themselves but to rely on the patient so that they may be cautious in the future.

3.3 NOTES ON IMPLEMENTATION

The two main ways to deploy our project was with either AWS or Google Cloud. The reason we ultimately chose AWS over google for the final version of our project is due to familiarity and ease of use. Some group members were either already familiar with or had used AWS in the past. Not many group members had experience with Google Cloud. Comparing the free tier offerings and free credits between Google Cloud Platform (GCP) and Amazon Web Services (AWS) involves assessing various factors such as available services, resource limits, and duration of the free tier. Another good reason to use AWS is because there are more online resources and tutorials when learning how to use it.

Both GCP and AWS provide free tiers aimed at enabling users to explore their cloud platforms without incurring immediate costs. However, there are distinct differences and trade-offs between them.

AWS Free Tier:

AWS offers a free tier that includes many free services. The AWS Free Tier provides new users with 12 months of access to select AWS services and a certain amount of usage each month. The popular services included in the free tier are Amazon EC2, Amazon S3, and Amazon RDS. Which is exactly the services that we are using. The downside is that our project manager got billed after a week, but it is low cost. About 3-6 dollars per week with the services we are running. It is hard to estimate exactly however because there is no monthly bill estimate until you run a server for a month.

GCP Free Tier:

GCP offers \$300 in free credits valid for 12 months. However, notably, GCP's free tier provides more flexibility in terms of services covered and usage limits compared to AWS. However, the free credits have a fixed duration of 12 months, after which users will be charged based on their usage. Another advantage to GCP is that the billing seems more accurate and more controllable. For example, at any time at the top of the GCP console, it always says how much remaining of the 300 dollars there is left in the free credit. It also seems much more upfront with cost when creating the server and databases. Every setting that is able to change when creating a database shows an exact cost/hour. This is extremely helpful when estimating cost. However, our group noticed that when we deployed a google cloud compute server and google SQL storage it was much more costly than AWS. It was about 20 dollars per week.

Our AI model was trained on jupyter notebook outside of the AWS and GCP. After the model was trained, it was saved as a .h5 file and could manually be put into the s3 bucket or google storage bucket, so there is no comparison between AWS and google for training. Likewise, the model has the same criteria (same parameters and data type and size) since the .h5 file can be uploaded to the

s3 bucket and the google storage bucket. Our model isn't too large so there would be no issue retrieving the file on either platform.

Issues we met: When installing the various libraries we had to install on the ec2 instance, at one point we got errors we couldn't figure out. Eventually we realized the instance ran out of storage, so we had to upgrade our EBS storage from 8 GB to 10 GB to increase storage for libraries.

We decided to fully implement our application on AWS. Getting the full application over to GCP would not be as easy as pulling the main branch on the GCP terminal and clicking run. We would have to fork the code and change the request URLs, change the credentials variables, replace the s3 retrieval code to use google storage bucket, migrate the RDS tables to Google SQL tables, install all of the new packages and libraries, potentially have to upgrade storage, update the .env file, install pm2 again, and debug this entire process. The problem is we were unaware that we needed to even attempt to deploy on GCP as well as AWS until our very last Wednesday meeting. Therefore, our most up to date project is deployed on AWS.

4. Testing

4.1 PROCESS

From the frontend to backend side of our web service we used Postman to allow us to verify the functionality from the frontend to the backend side of our web service, we used Postman to allow us to verify the functionality and results of our API. We began our testing with our early GET requests. This mostly included the Login and making sure we could acquire the patients, the doctors, and the products table in our SQL database. With verification that our requests were working as intended we moved towards more of the heavy and complicated requests like the ones we would need for our survey and got results of products from the AI model's list of ingredients. With these POST requests, we sent JSON-style raw text in add data to our database. Postman was great for replicated our frontend and testing with a variety of inputs and user text values. By not having to use our frontend every time to make a request to our backend testing went very fast and it was extremely easy for us to continue to use and make changes to our backend. Postman was also a great resource for updating and making sure our database was working as intended. Postman allowed us to directly interact with our database through GET, POST, and DELETE requests.

For the more frontend heavy testing we explored the use of Selenium which is used for browser automation. Although the strategy and use case was there the best way to test the frontend in our case was to write a list of use cases of what we wanted the user to be able to do. With a test suite identified we were able to walk through step by step everything the user should be able to interact with and do from the frontend side.

For the backend only side our testing consisted of console logs that would indicate where problems were occurring and if anything went wrong. For each request we designed use cases for each possible outcome and like the frontend we designed test suites for each request that was being made.

When testing the different models, the main parameters we checked were testing MSE and loss using k-fold to compare multiple trained instances and datasets. Loss and accuracy/MSE were our key values as loss gives a good overview of how our model is learning and improving while accuracy gives a good overall prediction of the model's performance. Once we got more distinct models, we ran confusion matrixes to check the accuracy rates for the different accuracies. We also wanted to ensure our model kept an accuracy/precision of at least 71% and the loss value below 20%.

4.2 RESULTS

Overall, the results of our testing were more than effective and led to a much more polished and safer website for our use. There were two main direct positive effects our testing had on the finished product and design. The first effect was the confidence and relief that our requests, frontend, and other aspects of our project had the correct functionality. By covering all possible use cases and scenarios we were given confidence in our project. The second effect was the impact and overall quality of the project increased once we started to apply testing. Right from the start, testing allowed us to find new ways to solve our problems and brought attention to problems we would not have discovered without the use of testing. By adding edge cases and try catches in our code and changing certain accessibility /privileges in the frontend our project became much smoother and more reliable as we continued to move on. Testing also showed that our model has accuracy/precision of 98.77%.

```
: test_score = model.evaluate(x_test, y_test)
  train_score = model.evaluate(x_train, y_train)
  print(train_score, test_score)

62/62 [=====] - 0s 3ms/step - loss: 0.1558 - binary_accuracy: 0.9877
248/248 [=====] - 1s 2ms/step - loss: 0.1556 - binary_accuracy: 0.9877
[0.15564894676208496, 0.9876905679702759] [0.1558469831943512, 0.9876942038536072]
```

Fig 5. Model Accuracy Test Results

5. Broader Context

5.1 PUBLIC HEALTH, SAFETY, AND WELFARE

This project aims to improve public health by making allergy diagnosis more accessible. With people being able to input their symptoms in real time and not having to wait to book an appointment with the doctor, the doctor is able to see they've added a new symptom and can run it through the AI. With how accessible our project will be to the public; we also hope it will encourage people who usually try to avoid the doctor to at least try our website and see if this way of diagnosis works better for them.

5.2 GLOBAL, CULTURAL, AND SOCIAL

As of right now, only patients in the USA can use our website. This is because the test data we were given was from an American medical center and takes into consideration how different areas in the USA can affect allergies. The ability to use the website for patients and doctors worldwide is in our project's future, it would require data from different regions worldwide. Data from around the world would allow us and the AI to understand which allergies are more prevalent in which countries.

5.3 ENVIRONMENTAL

Since our project can be used remotely by patients and doctors, this could become a replacement for allergy appointments, which would mean less emissions from cars going to and from medical facilities for appointments. Furthermore, patients can have confidence in knowing that they can still have diagnosis options despite physically residing far from doctor access.

5.4 ECONOMIC

As of right now, we are hoping that this project remains free to use, but since the use of it is up to the doctor, we aren't sure if medical facilities would want to charge patients for use of our project. This project was made with it being free in mind, so if there's a way for us to ensure that this resource remains free for all patients, then we will do everything we can to ensure it.

6. Conclusion

6.1 REVIEW PROGRESS

Looking back on this project, our main constraint was lack of knowledge about AI. AI is such a new technology that Iowa State University hasn't fully implemented into classes. For most of the group members, this was their first time working with AI as well as delving into the training of a variety of different kinds of models. Although there was a learning curve in the beginning, we are thankful that we were able to explore the world of AI before we entered the workforce. This project has given all of us skills that we can take into the workforce, and skills we can use in our everyday lives. This project required technical skills, time management, communication, the ability to work with others, and many other skills necessary for any student or person entering the workforce. Our group feels confident that this project has provided us with knowledge and skills that will help us for years to come.

6.2 FUTURE STEPS

For this project's future, we would like to be able to provide it worldwide so anyone can have access to non-invasive diagnosis of what they may be allergic to, as well as a safer and more reliable prescription process. We hope to have this product utilized by any practice that prescribes medications to patients and minimize the likelihood that patients will encounter unpredictable allergic reactions based on their own personal characteristics. We would also like to fully implement the service into other software, such as Epic Systems that McFarland clinic uses. By fully integrating it, we could ensure appropriate doctors are assigned, and a different treatment option is available to our local healthcare system. Taking it worldwide, we would also want to maintain its no-cost use, we want it to be free for everyone to use. As we continue our product journey we would love to hear feedback from patients, doctors, and hospitals that are interested in our service. It is particularly important to use that we continue to improve and sharpen our artificial intelligence model service, and to do that we need to have constant feedback from the users of the service itself. Another future step we were like to take is to have more data and more input to train our model with. For AI, data is what is most important for it, and we need to make sure that we are constantly updating and improving the model and its training inputs and parameters. The more data we possess, the more accurate our model will be, and the more people we can positively affect.

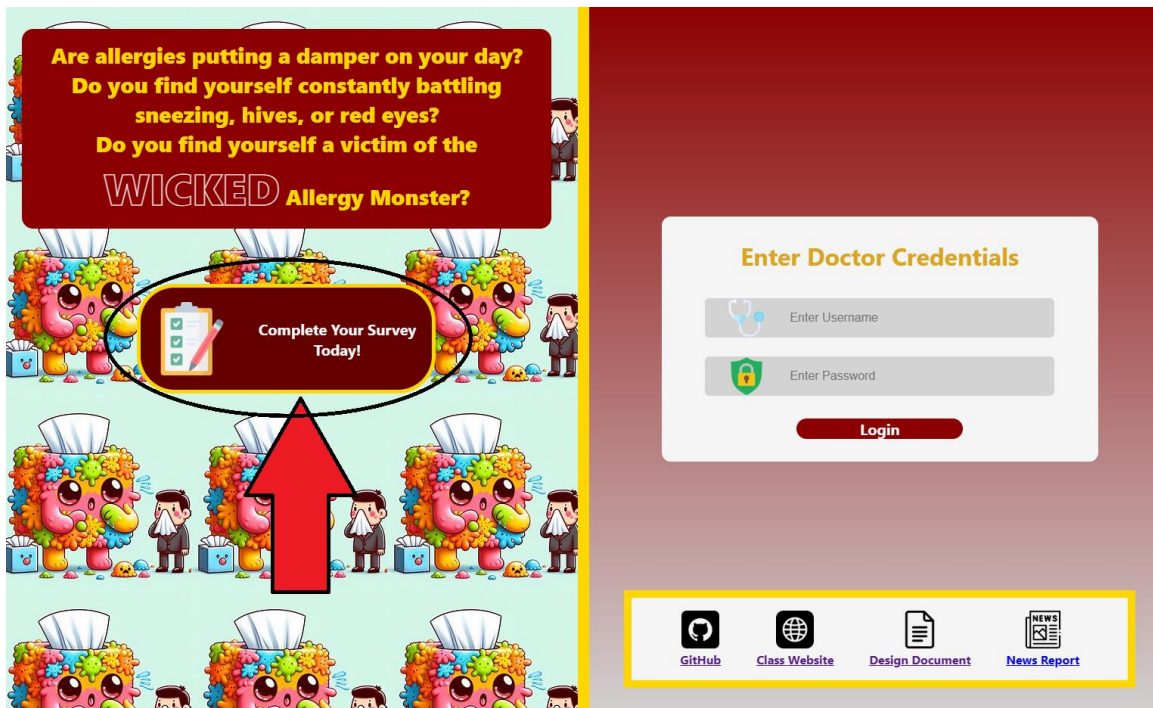
By staying connected with our past, present, and future clients we will ensure a bright future for the service we are providing to the public.

6.3 APPENDICES

APPENDIX 1: OPERATION MANUAL

Patient Users:

1. Click on the “Complete Your Survey Today!” button to navigate to the survey.



2. The button will take you to the survey page. Fill out the survey in 3 easy steps.

Welcome! Let's get started 📧

To ensure accurate determination of allergies, it is crucial for users to input correct and comprehensive information. Please provide precise details about the patient's name, age, gender, skin tone, geographical information, and any symptoms you are experiencing. Accurate data enables our Artificial Intelligence model to come to a more informed conclusion on the patient's potential allergies. Additionally, if you have a change in any of the listed info in the future, please consider updating this information to help deliver personalized and reliable results. Your cooperation in providing accurate information plays a pivotal role in optimizing the effectiveness of the allergy determination process.

Email
example@gmail.com

Username
Enter Username

Doctor Code
Enter Your Doctor's Code

Full Patient Name
Enter Full Patient Name

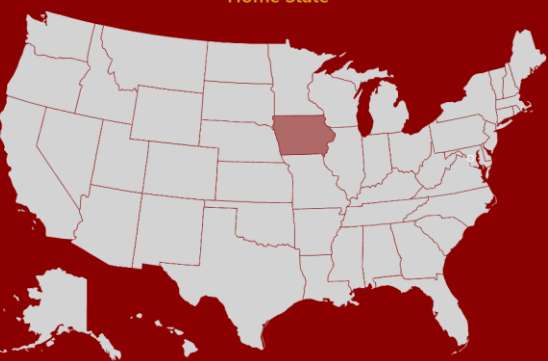
Gender
Select Gender

Date of Birth
2024 | April

Su	Mo	Tu	We	Th	Fr	Sa
31	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	1	2	3	4

- a. Type your email, username, doctor code, full name, and select your gender and date of birth from the UI. The doctor code correlates to the doctor who will receive the information. The patient needs to contact their doctor for the doctor code. Our preexisting doctor is Dr. Dingle who has the doctor code: 202020.

Home State



Skin Tone

Dark Brown Olive Medium Fair Light

- b. Next, hover over and click on the state you reside in. Once selected, the state will become blue. Then, choose the from among the listed skin tones the one closest to your own skin tone.

Skin Symptoms

- Sensitive skin allergist diagnosed
- Sensitive skin self diagnosed
- Allergic contact dermatitis
- Eczema atopic skin
- Dry chapped skin
- Acne pimples
- Skin allergies
- Rosacea
- Discoloration hyperpigmentation
- Fine lines wrinkles
- Psoriasis
- I have celiacs disease and need my products to be gluten free
- Blackheads whiteheads
- Coconut
- Textile Dye Mix
- Gluten
- Respiratory
- Patchy rash
- Tbd
- Congestion
- Contact dermatitis
- Ffa
- Dermatologist
- Rash
- Balsam Peru
- I
- Itching
- Rashes
- Mystery body rash
- PPD
- My gf gets rashes so I need to change my products
- Alopecia
- AP93
- Dry skin
- Cocamidopropyl Betaine
- Metal
- Ethylenediamine Dihydrochloride, Potassium Dichromate
- Nickel
- Low level allergy to Balsam of Peru
- Allergies
- Glutaral & iodopropynyl butyl carbamate
- Balsam of Peru allergic
- Licus planus
- Lip inflammation
- Hives
- Lichen Sclerosis
- Dry Lips
- Occasional rash outbreaks
- Propolis
- Allergic to Cocamidopropyl Betaine
- Surgical site healing issues
- Urticaria
- Polysorbate 80
- Fragrance
- Contact Dermatitis
- Red, peeling, itchy, burning lips
- Celiac
- Perioral dermatitis
- Scalp
- Scalp issues
- Hives on legs
- Itchy scalp with hair loss
- Allergic reaction
- Exccema
- Dermatitis Herpetiformis (dermatologist diagnosed)
- Allergic to polyethylene glycol (peg)
- Oral lichen Plans

Submit

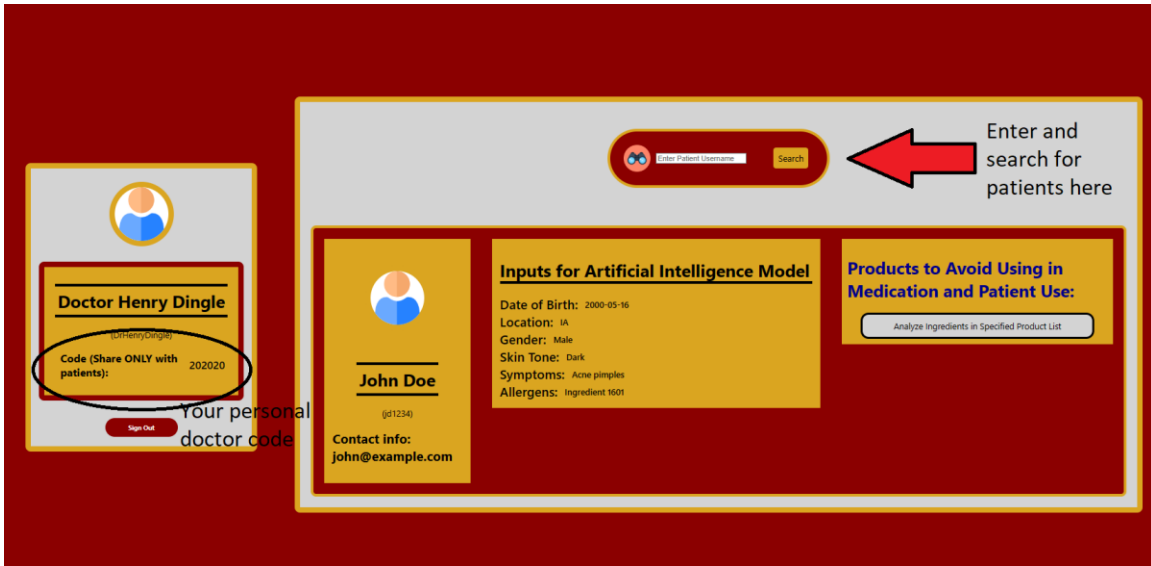
- c. Now, click the boxes to select all the skin symptoms you are currently experiencing, or have experienced in the past. When finished, press submit.
3. Now that the survey has been submitted, you wait for your doctor to view the results and contact you about your AI predicted allergen results.

Doctor Users:

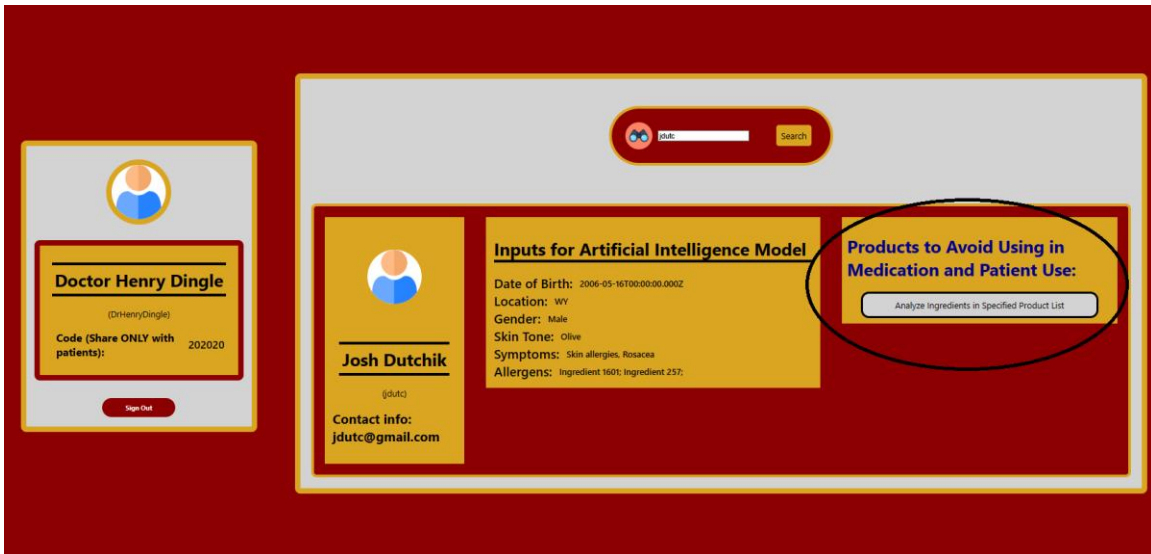
1. Doctor's will already have a premade account, so enter the credentials and click login. Our current existing doctor credentials are:
 - a. username: **"DrHenryDingle"** with the password: **"Password123"**

The image is a composite of two parts. On the left is a colorful advertisement for a survey. It features a dark red box with yellow text asking: "Are allergies putting a damper on your day? Do you find yourself constantly battling sneezing, hives, or red eyes? Do you find yourself a victim of the WICKED Allergy Monster?". Below this is a yellow box with a clipboard icon and the text "Complete Your Survey Today!". The background is filled with cartoonish, colorful monsters and small human figures. On the right is a dark red background with a white login form titled "Enter Doctor Credentials". The form has two input fields: "Enter Username" (with a stethoscope icon) and "Enter Password" (with a shield icon). Below the fields is a red "Login" button. A large red arrow points upwards from the bottom of the page towards the login button. At the bottom of the right side, there is a yellow-bordered box containing four icons and links: GitHub, Class Website, Design Document, and News Report.

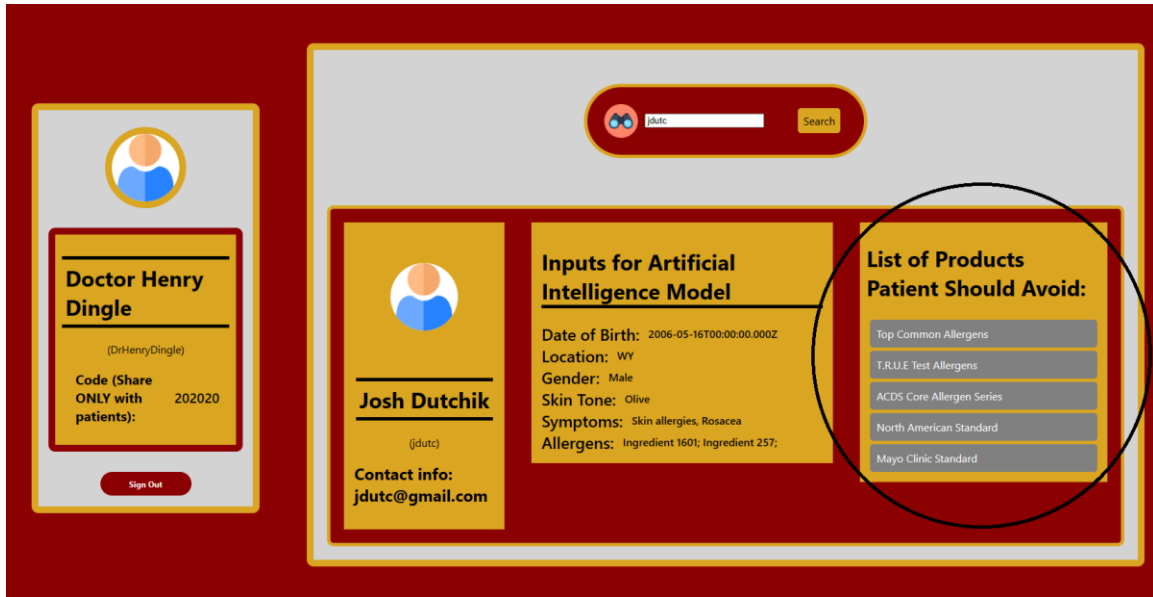
- Once the doctor is logged in, they are then able to: view their reference doctor code on the left side, and search for patients on the top of the screen.



- Entering and searching for a patient username pulls up their information. Once a patient's information is brought up, the doctor can click the "Analyze Ingredients in Specified Product List" button to see which products contain the ingredients they are likely to be allergic to.
 - A preexisting patient you can search for is: "jdtuc"



4. After analyzing the ingredients, a list of products the patient should avoid will appear.



AWS link: <http://ec2-54-174-30-87.compute-1.amazonaws.com:3000>

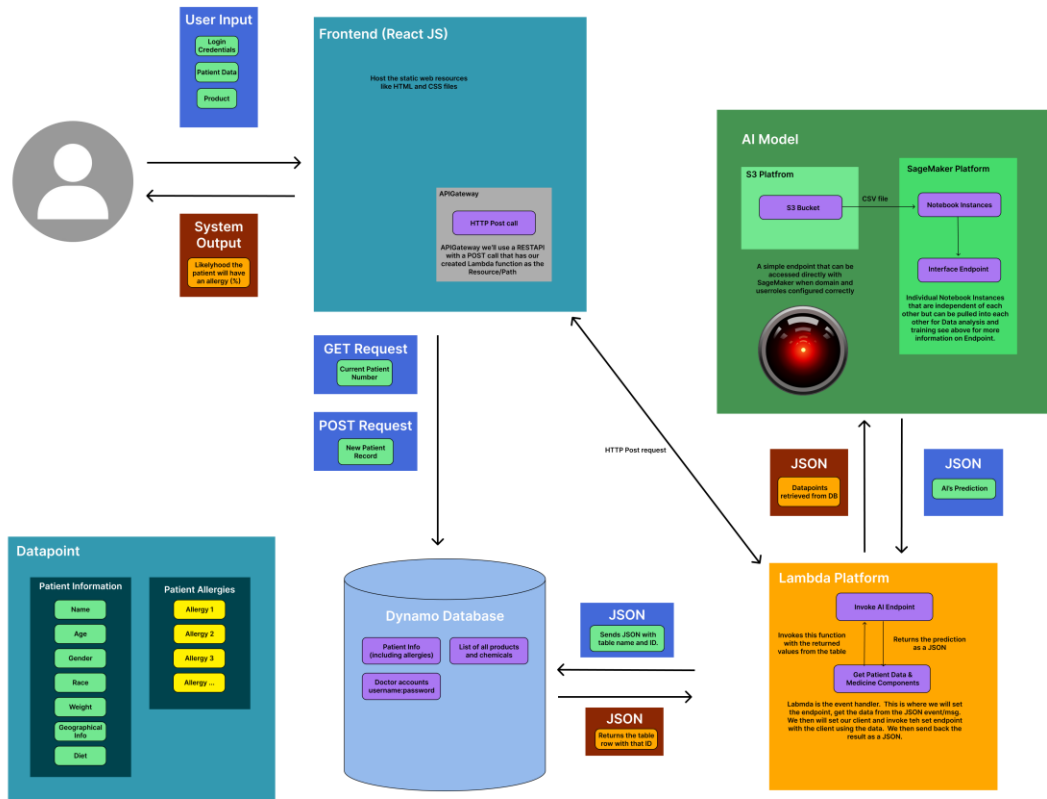
Updated AWS link: <http://ec2-52-23-238-114.compute-1.amazonaws.com:3000/>

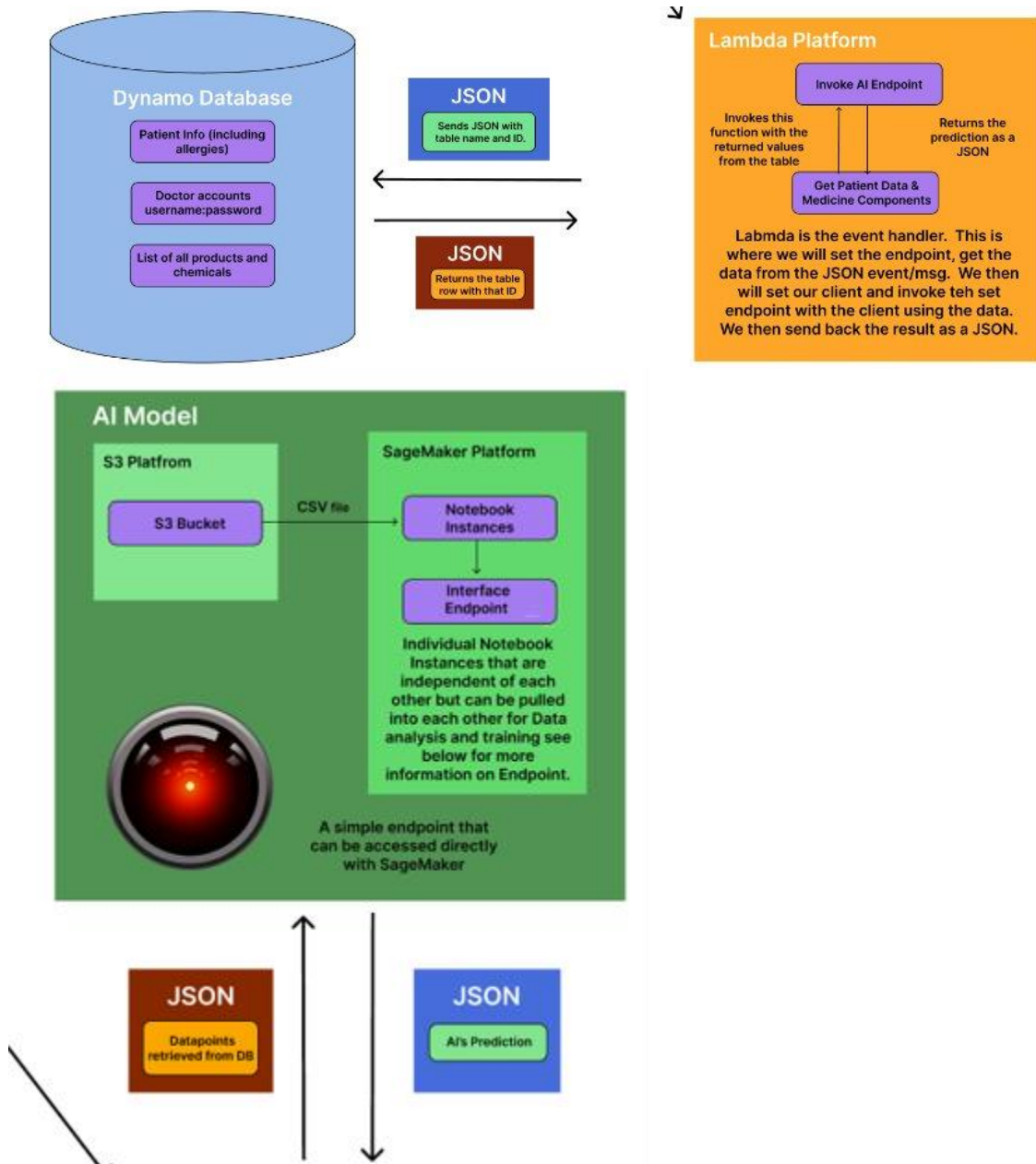
Or link on our website if the URL continues to change

GCP link: <http://34.16.194.253:3000/>

APPENDIX 2: ALTERNATIVE VERSION OF INITIAL DESIGN

This is the old block diagram made for this project in 491:





We kept some of the implementation that we planned in this diagram, such as using react for frontend and S3 for the AI. The main reason we deviated from this plan is because we found less complicated ways to implement the project that made more sense than what we planned in 491. A few team members also had more experience using certain libraries and platforms. We also learned more from classes we were in this semester that helped us figure out easier ways to implement this project.

APPENDIX 3: OTHER CONSIDERATIONS

As previously stated, our group chose to deploy our website using AWS over google cloud, but the website is able run on either, so it's mainly up to preference. Multiple programs are also required to run and deploy this project, such as Python, TensorFlow, jupyter notebooks, etc.

APPENDIX 4: CODE

All code is available through our project GitHub located here:

<https://github.com/jdutchik/SE492Group13>

6.4 REFERENCES

- Bates E, Wilson SM, Saygin AP, Dick F, Sereno MI, Knight RT, Dronkers NF. Voxel-based lesion-symptom mapping. *Nat Neurosci*. 2003 May;6(5):448-50. doi: 10.1038/nn1050. PMID: 12704393.
- Collins FS, Varmus H. A new initiative on precision medicine. *N Engl J Med*. 2015 Feb 26;372(9):793-5. doi: 10.1056/NEJMp1500523. Epub 2015 Jan 30. PMID: 25635347; PMCID: PMC5101938.
- Nevis, I.F., Binkley, K. and Kabali, C. (2016) *Diagnostic accuracy of skin-prick testing for allergic rhinitis: A systematic review and meta-analysis*, *Allergy, asthma, and clinical immunology : official journal of the Canadian Society of Allergy and Clinical Immunology*. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4848807/> (Accessed: 27 April 2024).