4. Design

4.1. Design Content

Since this project is a software-based project, the design content will relate to the flow of data and functional components instead of physical hardware. The design will have a front-end that the user interacts with, a model which makes the prediction, and the backend to connect the two together.

4.2. Design Complexity

Our final design consists of three major components: the front-end, the backend, and the model. The front-end will be the interface seen directly by the user and requires a careful design to ensure users can navigate it properly. The backend will be built using cloud functions to call off to the model and return the result to the front-end. The model will create a prediction for whether the patient has an allergy based on the input data received. Some of the applicable principles are keeping the model and its data secure, as well as creating a transparent view of how data is used.
Developing an ML model for our project presents a significant challenge as we strive to maintain high accuracy. Since the model will be trained with a large dataset with many varying input types, there may be numerous iterations before one begins to work as intended. Additionally, it is crucial to create a robust testing suite to safeguard against overfitting the model. Since the model will be hosted on a cloud-based platform, we will need to ensure that it can be accessed from anyone without compromising the integrity of the data to follow artificial intelligence engineering principles.

4.3. Modern Engineering Tools

- Model Creation: Keras API -
  - Keras serves as a high-level API for the TensorFlow machine learning library, known for its emphasis on providing a fast and user-friendly interface for creating production-ready ML models. Therefore, our group will be using this library to create our prediction model.
- Model Hosting: Amazon Web Services -
  - Since our model needs to be hosted on a cloud service, we looked into both Google Cloud Platform and Amazon Web Services to train our model, and we settled on using Amazon Web Services as our primary provider. A cloud platform lets us make use of tools designed for this type of wide-scale machine learning development and host our components in a combined manner.
- Model Workbook: Jupyter Notebook -
  - Jupyter Notebook is a browser-based tool for developing and presenting data science projects. It is specifically designed to visualize and execute Python code, which is useful in developing machine learning models. Our group will use Jupyter Notebook as our primary development tool for our machine learning model. Like the backend, both of our cloud services have options to host Jupyter notebooks natively. Amazon Web Services has Jupyter integrated within SageMaker, so no additional configuration is required to use this.
- Backend framework: Serverless functions -
  - In order to keep our backend resilient, we will be building serverless functions utilizing the cloud service as the main component of our backend. Each of the cloud services has their own method of creating this. Amazon Web Services has Lambda, which can quickly run the functions required to transfer the data.

- Front-end framework: React.js -
    - React.js is a front-end JavaScript framework designed for building responsive single-page web applications. We will be using this library to develop the user interface for inputting data and displaying results from our ML model.

4.4. Design Context

Area	Description	Examples
Public health, safety, and welfare	Our project aims to use AI for allergy prediction. The main community area affected by our design will be the Public health, safety, and welfare area. By creating a tool that allows medical professionals to input patient data and symptoms, the goal of our design is to be able to predict specific allergies a patient may have. This can help provide faster and more accurate results, while reducing human error. This can lead to better treatment that benefits both the patient and medical professional.	Increase personal awareness of possible allergies.
Global, cultural, and social	With our project relying heavily on AI and sensitive patient information, we need to ensure that data is secure and that our design follows HIPAA regulations as this can impact the Global, Cultural, and Social areas. With our data being ported to a third party library with Keras, we need to ensure data does not get leaked onto the public web.	Keep user data secure, and follow HIPAA regulations.
Environmental	There is not much to consider outside of the energy and resources we need to consume to run our servers and AI model. Since most of our services will be through the Cloud, the main environmental considerations would be how much resources and energy is required to run our host machines.	Restrict computing power and energy consumption.
Economic	If our product is used by real medical professionals and facilities, an beneficial economic impact could be the resources, time, and overall cost saved by being able to predict patient allergies faster and more accurately. This would allow doctors to spend less time having to come up with the diagnosis and reduce the chance of inaccurate diagnoses. For our organization, some of the economic impacts to consider for us would be the third party sources we use and if we need to cover any costs for the operations if they were to exceed the free plans. Some examples are that	The model prevents the need for expensive allergy testing kits. AWS provides free tiers which should suit all of our request needs during development.

	cloud services such as AWS support free services up to a certain amount of bandwidth / requests. The same could be said for the Keras libraries and is something we may have to consider if our design scales. For now, as a small isolated school project, we believe there shouldn't be many financial impacts to consider as our project should not have to scale to a point which exceeds any of the free plan resources.	

Table 2: Table describing the impact our project may have within various areas of concern.

4.5. Prior Work/Solutions

Allergen testing is already done, but it is mostly done on test animals and in some cases humans. Monitoring skin pricks or digestive reactions to new products in animals like guinea pigs and humans gives us most of our current insight into allergic reactions. Not much exists in predicting allergic reactions like we are trying to do.

It is only in recent years where the idea of using artificial intelligence to predict any sort of medical event has become popular. While some of these models exist, none seem to have gained a large-scale adoption in the medical field. An article [4] published in August 2023 states that even though artificial intelligence is not at the point where it can currently be used in the industry, it will approach that point and is an overall benefit to healthcare.

We are not knowingly following in the footsteps of any other projects or programs as Machine Learning is pretty new, and has yet to be widely adopted in the medical field.

The article [4] states that the biggest hurdles to overcome with creating an AI model that could be used in the healthcare industry are the wide scale data requirements, lack of ability to see how the model processes data, and difficulty in validating the model. In order for a machine learning model to begin predicting data correctly, it will need a significant amount of data.

4.6. Design Decisions

One of the major design decisions we needed to make in the project was choosing whether to use Amazon Web Services or Google Cloud to use computing power and host our machine learning model. To make this decision, we had to carefully compare the features offered by both such as pricing, ease of use, and functional capabilities. After comparing them both by seeing available resources and using our own experiences with them, we elected to choose Amazon Web Services as our primary cloud platform for the project.

Another major decision we made was choosing which language to build our front-end with. We chose to use React, since many of us have experience working with the language, and its flexibility will be a great benefit to us as we implement our design.

The last major decision for the project was choosing how to create our backend. After some thoughtful discussion, we decided that we wanted to use a cloud function which would call our model and return the data back to the front-end. This method requires less coding overall, and it allows us to build the backend using the same cloud platform as the machine learning model uses.

4.7. Proposed Design

4.7.1. Design 0 (Initial Design)

Design Visual and Description

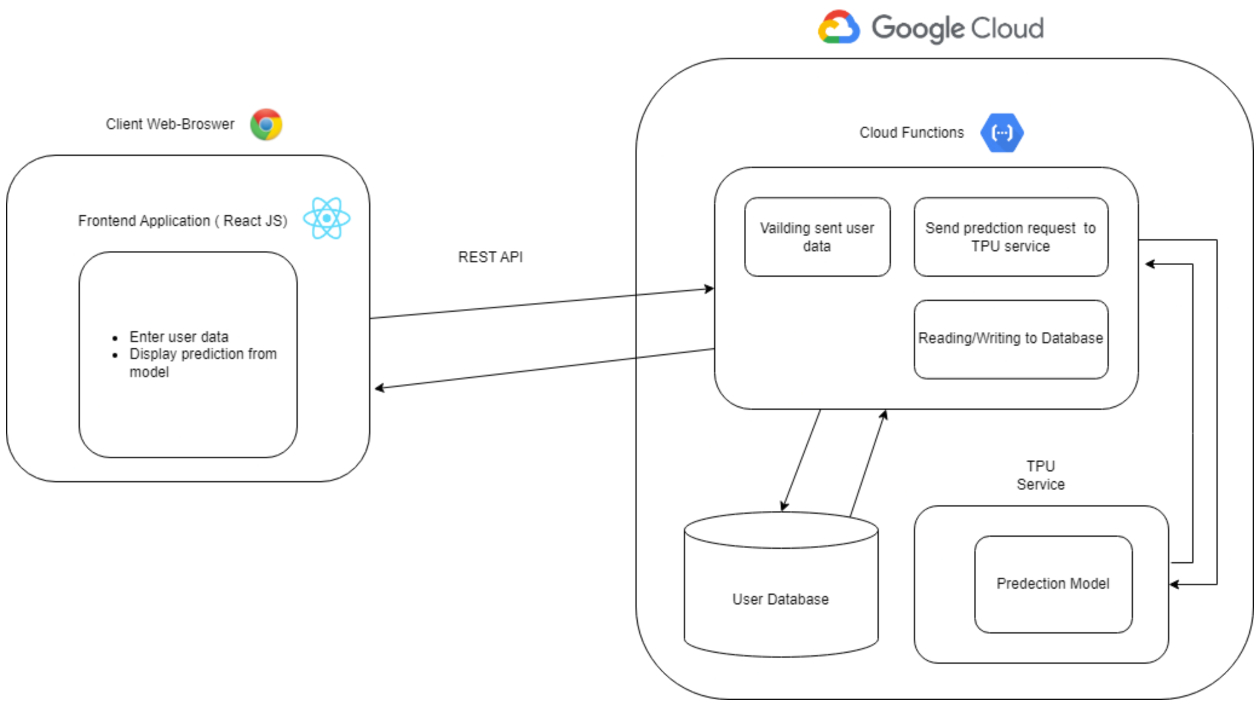

Figure 2: Initial design of the project with the React front-end, a backend utilizing cloud functions, and the prediction model.

The first portion of the design is the React front-end. It allows the user to enter the data and display the prediction. This component does not need much itself but needs to connect to the backend, which is possible with the given design. Since both the cloud function and model will be using the same cloud platform for their operations, they are placed under the same category. We used Google Cloud as a cloud provider option, but have since chosen AWS. Within the cloud platform, we have the cloud functions, the database, and the TPU service, a circuit designed for neural networks.

Functionality

In the current design, the user enters data through the front-end to get a response. The data is passed into the cloud, entering the cloud function, where the data is added to the database and sent to the model. The model, which at this point has already been trained, looks at the data and makes a prediction. The model sends its prediction to the cloud function. The function sends the data back to the front-end and adds the prediction to the database where the data is stored. The front-end displays the prediction to the user.

4.7.2. Design 1 (Design Iteration)

Design Visual and Description

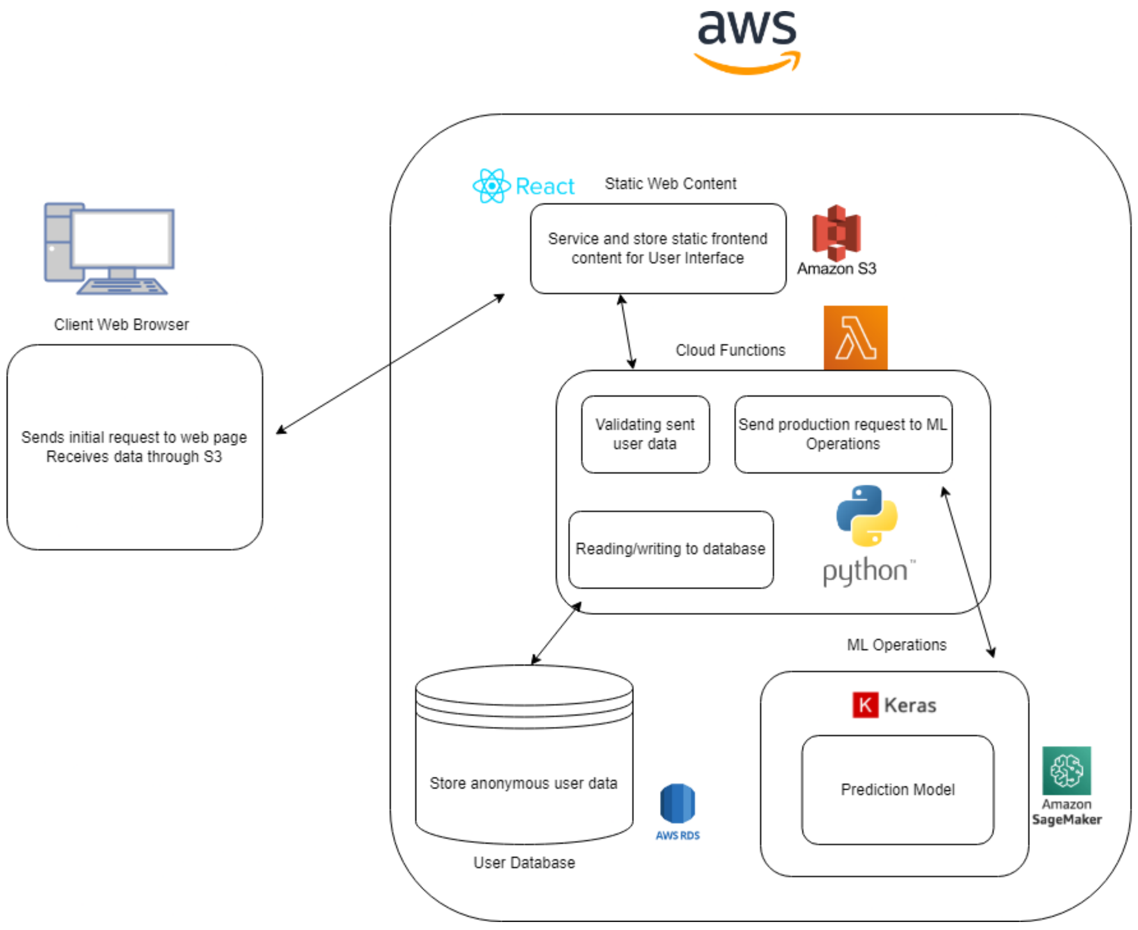
aws
Client Web Browser
Sends initial request to web page
Receives data through S3
React
Static Web Content
Service and store static frontend content for User Interface
Amazon S3
Cloud Functions
Validating sent user data
Send production request to ML Operations
Reading/writing to database
python
ML Operations
Keras
Prediction Model
Amazon SageMaker
Store anonymous user data
AWS RDS
User Database

Figure 3: Revised design of the project, including specific services within AWS to be used in the project.

Front-End (User Interface)

The user interface of our design will be a desktop web application. The user interface will be responsible for presenting a user-friendly interface for entering patient information along with relevant predictors. Additionally, it will facilitate the input and listing of chemical components to be sent to the allergy prediction model. Additionally, the front-end will handle the display of the prediction results generated by the model.

Backend

The back end will play a crucial role in managing requests from the user-facing web application. This involves updating a database to store patients' information and making prediction requests. Our backend architecture will comprise multiple AWS services. The front-end will be stored and delivered through an S3 bucket. The API, managed by AWS Cloud Functions, will be responsible for forwarding protection requests to our ML model, authenticating users, validating user input, and handling database queries for storing user and patient information. Additionally, we plan to host our prediction model using AWS SageMaker, ensuring accessibility through our Cloud Function.

4.8. Technology Considerations

By automating the technical aspects typically handled by a medical professional, our project has the potential to significantly decrease costs associated with drug prescription and development. However, a major challenge we face is our heavy reliance on obtaining high-quality data for our predictions.

One drawback associated with employing a machine learning model for allergic prediction is its reliance on the quality of the training data. The model may encounter challenges with chemical components that are not well-represented in our training dataset.

4.9. Design Analysis

As our understanding of what we needed for the project changed, we had to make some revisions to our design. Once we elected to choose AWS as our primary cloud service, we needed to change the design to reflect this decision and to illustrate which services would fulfill which purpose within the context of the project as a whole.

We have tested hosting each component individually on AWS. The front-end can be reached using a web browser and can make a call to the backend, receiving a response. In addition, the model can be stored on AWS.

Furthermore, during the round-trip testing phase, we initially decided to utilize Java for the backend functionality on AWS. However, after developing a function, we observed that Java development was challenging to integrate seamlessly with AWS. Subsequently, we discovered that developing with JavaScript proved to be a much smoother and more efficient process.