Allergy Prediction Using Artificial Intelligence

Client Lead: Joseph Trembley
Team Lead: Noah Ross
Minute Taker: Ella Godfrey
Research Lead: Xerxes Tarman
Quality Assurance Lead: Alex Ong

Client: Ashraf Gaffar

Advisors: Ashraf Gaffar, Ashfaq Khokhar

Introduction:

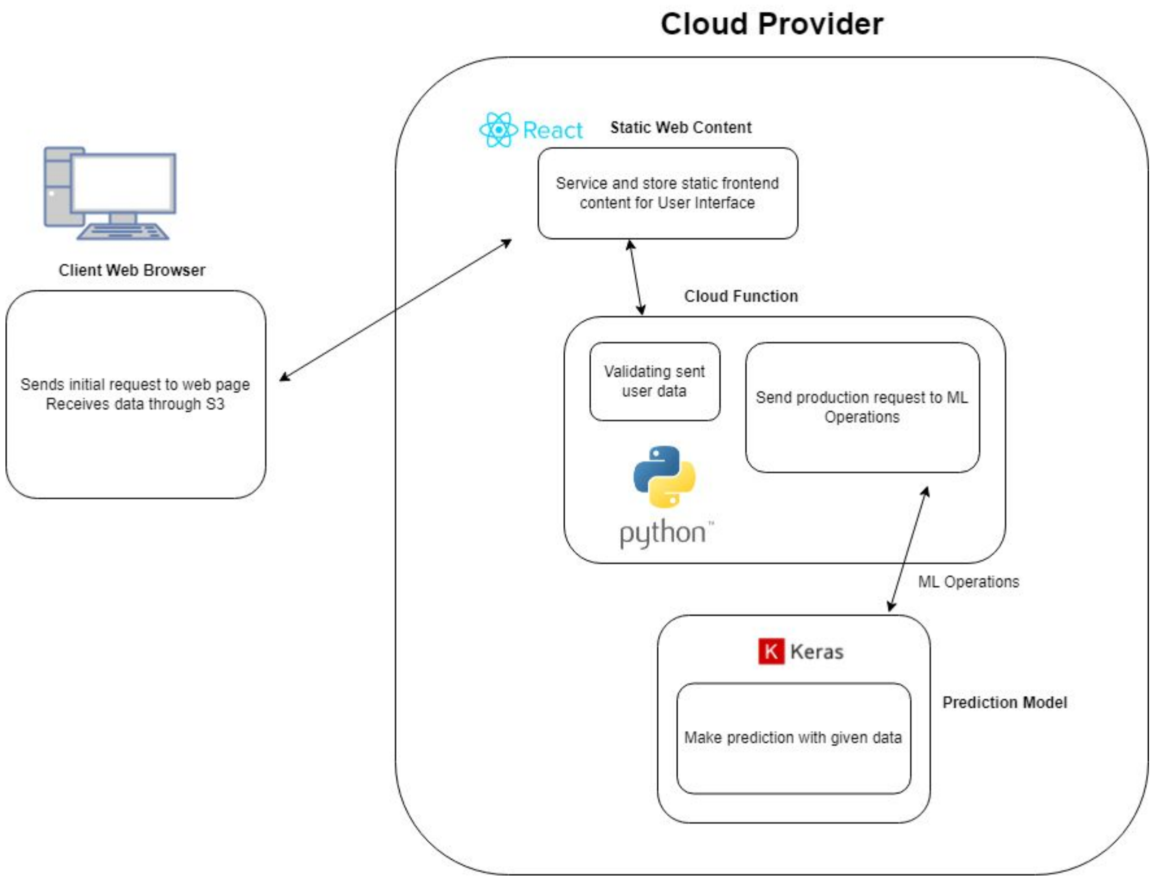
- **Goal:** predict allergic reactions to medicines using machine learning
    - Rapid response time
    - Non-invasive testing
    - Genetic/Biological factors
        - Gender
        - Age
        - Skin Conditions
        - Skin Tone

Introduction: (Continued)

- **Implementation:**
  - Components
    - React Front end
    - Python Back end to handle requests
    - Locally trained Keras model
  - Comparison between cloud providers
    - AWS
    - Google Cloud

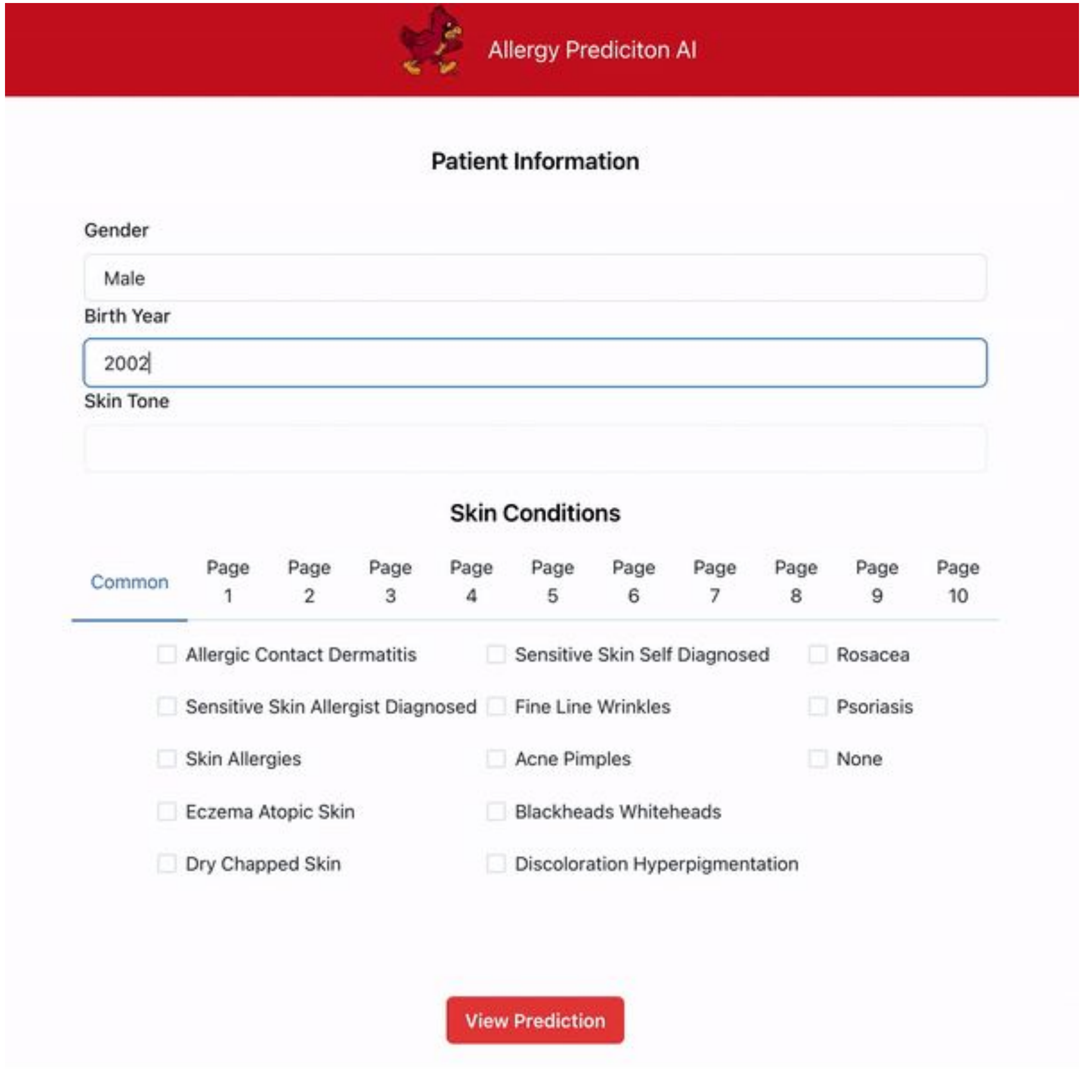
Design:

- Server-Side: Data processing, validation, and running the ML model.
- Client-Side: Entering user data, calling REST API, displaying prediction results.
- Changes: Database has been removed from design.



Progress: Front End

- Second iteration complete
- Sends/receives requests
- Takes user data and displays information from back end
- Selectable options for Skin Conditions, simplifies values to feed into the model



Progress: Cloud Function

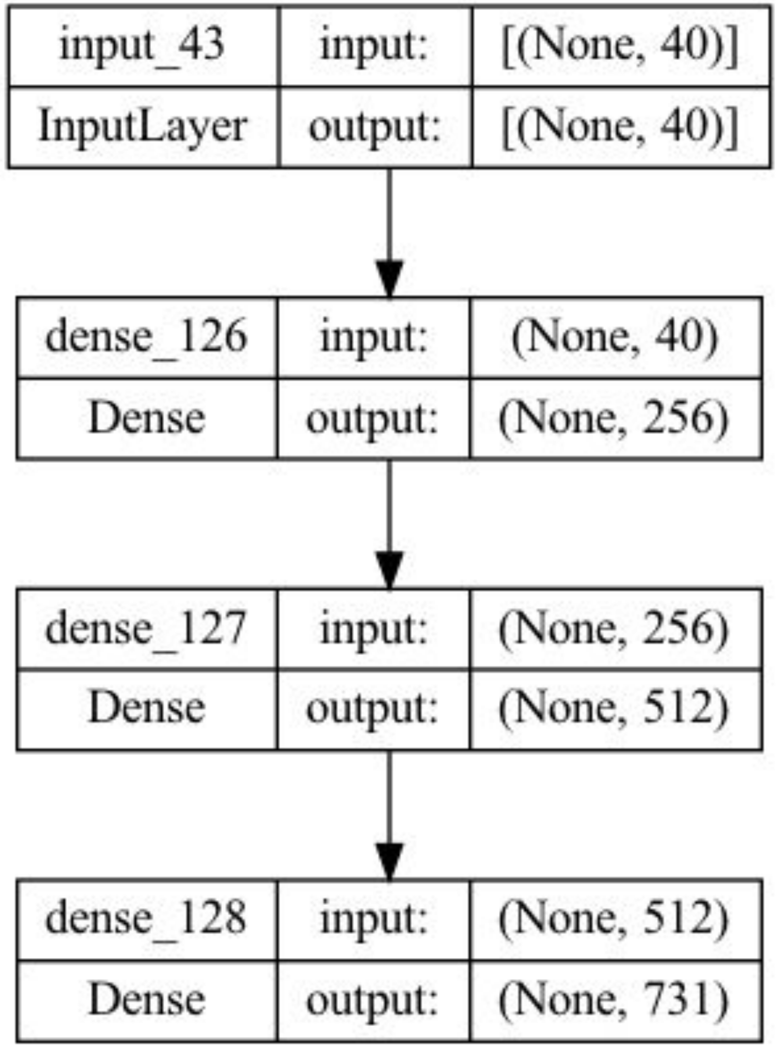
- Converted cloud function to Python
  - Receives request from front end
  - Prepares request for model
  - Returns response to user
- Not deployed until model is finished

backend.py 900 B

```python
import os
import io
import boto3
import json
import csv

# grab environment variables
ENDPOINT_NAME = os.environ['ENDPOINT_NAME']
runtime= boto3.client('runtime.sagemaker')

def lambda_handler(event, context):
    print("Received event: " + json.dumps(event, indent=2))

    data = json.loads(json.dumps(event))
    payload = data['data']

    #Import textvectorization from model and convert SkinConditions

    response = runtime.invoke_endpoint(EndpointName=ENDPOINT_NAME,
                                       ContentType='text/csv',
                                       Body=payload)
    result = json.loads(response['Body'].read().decode())

    predicted_labels = result['predictions']

    #Remove the 0 prediction from the returned elements
    for x in predicted_labels:
        if(predicted_labels[x] == '0'):
            predicted_labels.pop(x)

    return predicted_labels
```

Progress: Model

- Evaluation Criteria - higher values better
  - AUC - Area Under Curve
    - broad measure of ability to predict
    - degree of separability between positive and negative
    - Current AUC is ~0.35
  - Precision
    - Ratio of true positives to predicted positives
    - Current precision is ~0.05
    - Too many false positives
  - Recall
    - Ratio of identified positives to false negatives
    - The recall is ~ 0.90
    - Sufficient false negatives
  - F1 Score
    - Comparison between precision and recall
    - F1 Score is ~0.25
    - Imbalance between precision and recall



Progress: Different Model Tested Overview

SVC(rbf Kernel)

	precision	recall	f1-score	support
0	0.00	0.00	0.00	664
1	0.00	0.00	0.00	456
2	0.00	0.00	0.00	426
3	0.00	0.00	0.00	328
4	0.00	0.00	0.00	249
5	0.00	0.00	0.00	276
6	0.00	0.00	0.00	225
7	0.00	0.00	0.00	235
8	0.00	0.00	0.00	218
micro avg	0.00	0.00	0.00	3077
macro avg	0.00	0.00	0.00	3077
weighted avg	0.00	0.00	0.00	3077
samples avg	0.00	0.00	0.00	3077

complex1NNmodel

0	0.34	1.00	0.50	664
1	0.23	1.00	0.37	456
2	0.22	1.00	0.35	426
3	0.00	0.00	0.00	328
4	0.00	0.00	0.00	249
5	0.00	0.00	0.00	276
6	0.00	0.00	0.00	225
7	0.00	0.00	0.00	235
8	0.00	0.00	0.00	218
micro avg	0.26	0.50	0.34	3077
macro avg	0.09	0.33	0.14	3077
weighted avg	0.14	0.50	0.21	3077
samples avg	0.26	0.40	0.29	3077

Random Forest

	precision	recall	f1-score	support
0	0.37	0.26	0.30	664
1	0.29	0.16	0.21	456
2	0.24	0.14	0.18	426
3	0.19	0.09	0.12	328
4	0.17	0.06	0.09	249
5	0.17	0.07	0.09	276
6	0.13	0.05	0.07	225
7	0.18	0.07	0.10	235
8	0.08	0.02	0.04	218
micro avg	0.25	0.13	0.17	3077
macro avg	0.20	0.10	0.13	3077
weighted avg	0.23	0.13	0.16	3077
samples avg	0.13	0.10	0.10	3077

Remaining Work

- Model Refinement
  - Continuously refine the machine learning model
  - Explore advanced algorithms and techniques to optimize performance
  - Balance precision and recall to acceptable parameters
- Host on Cloud
  - Utilize cloud to host, manage, and scale the system components
  - Use cloud for storage and networking to optimize performance

Remaining Work

- System Integration
  - Integrate all system components to ensure smooth communication and operation
  - Test integration points to identify and resolve any potential issues
- Evaluate Performance
  - Measure the efficiency and accuracy of the model
  - Test the scalability by increasing the workload
  - Measure the response time of the system to user requests

Challenges/Solutions:

- Testing cost on cloud platforms
  - Testing will be costly as Cloud Platforms are pay as you go, making it ideal to wait for model completion to deploy our applications
- Producing a model with high recall and precision is difficult
  - Explore various algorithms to identify the most suitable approach
  - Enhance the parameters to optimize performance metrics

Conclusion:

- Continuously improving model
- Completed front end and back end
- Host components on cloud
- Evaluate performance and costs