Date:05/10/2023

Maintenance Document
Seeing Asthma

Team Information

Team Name: Healthy Coder
Team Number: TA 09

Team Member:

Role	Name	LinkedIn
Business Analysis	Kaijia Yu	Kaijia Yu
Full Stack Engineer	Ridong Jiang	Ridong Jiang
Front-end Engineer	Shiyu Wu	Shiyu Wu
Data specialists	Navrattan Singh Dhillon	Navrattan Dhillon
Cyber Security Specialist	Sonia Lakhani	Sonia Lakhani

Table of Contents

1. Product Overview

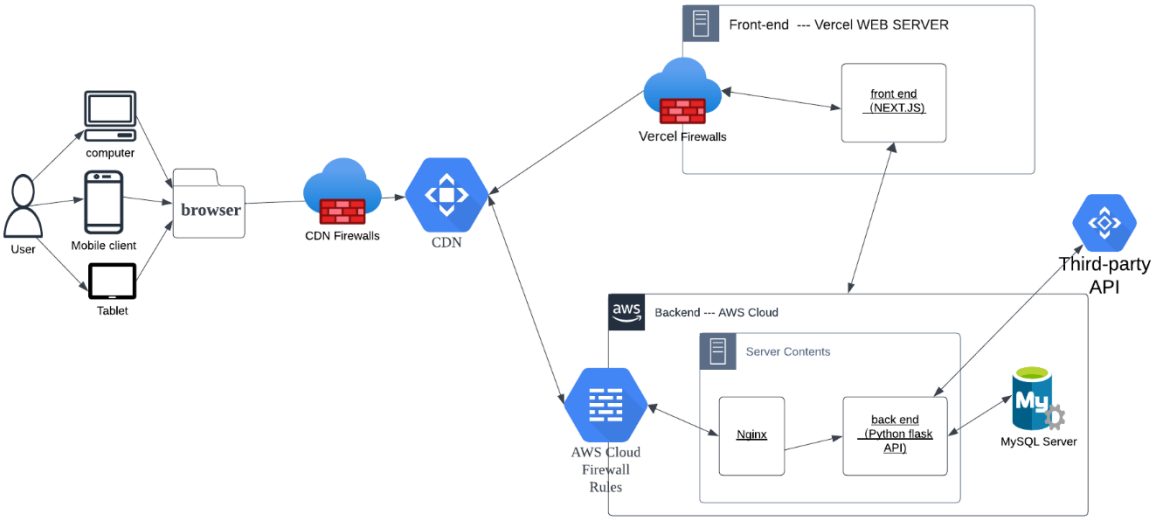
The project "Seeing Asthma" is based on research into the many allergens that are widespread in Australia and pose a serious threat to the health of children with asthma. The main objective of this project is to provide a platform for parents of asthmatic children to learn more effectively and utilize features to better manage their child's asthma through this website.

The Seeing Asthma website is an integrated platform with not only comprehensive asthma knowledge but also many new features to help you better manage your child's asthma. These include a "picture recognition" feature that helps parents identify asthma triggers regardless of location, time, or device. Recommendations for asthma-friendly activities and diets. In addition, animations and games have been developed to promote self-protection awareness, so that children can be taught how to prevent asthma.

2. Architecture

The system architecture is explained in this section. The maintenance team can understand the working of the website by reading this section.

2.1 High Level System Architecture Diagram



Front-end: Vercel WEB SERVER: A web server that hosts the entire front-end content. Front-end (NEXT.JS): Our front-end is developed using the NEXT.JS framework, providing users with an intuitive interface and a smooth user experience.

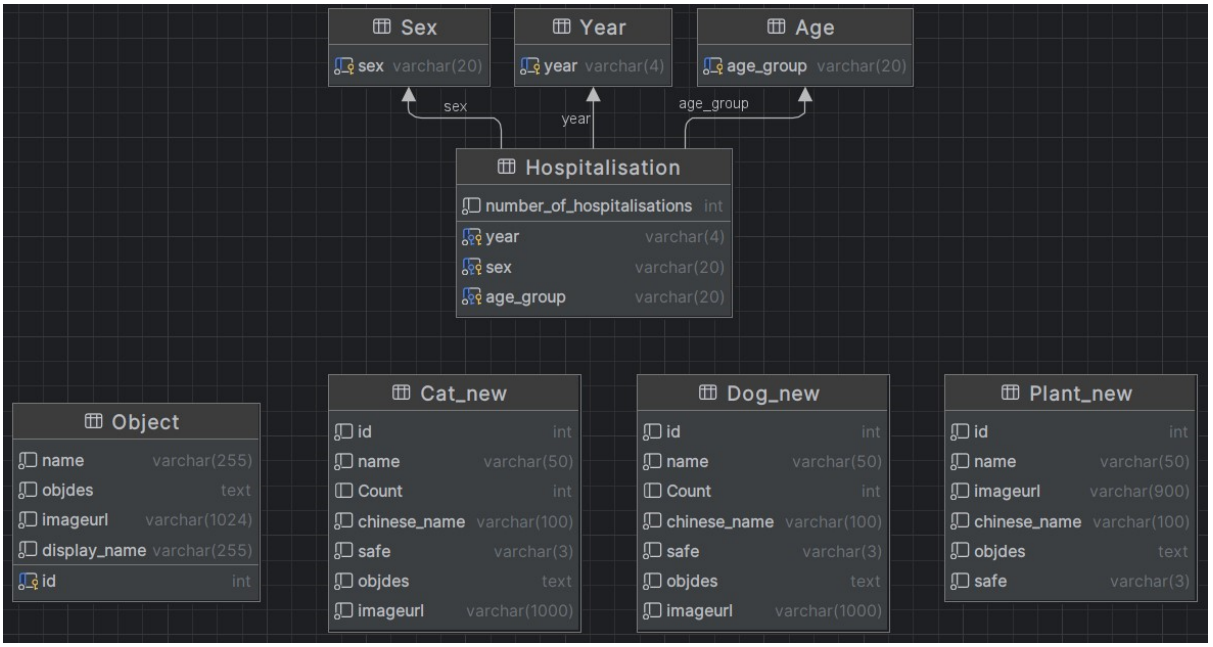
Backend: Backend Cloud Service (AWS Cloud): All backend resources are hosted in the AWS cloud service, protected by the AWS firewall, ensuring data security. Backend (Python Flask API): A backend API developed using Python Flask, handling business logic and data interactions. MySQL Server: A database server used to store application data.

Third-party APIs: YouTube API, Baidu vision API, Google vision API: Our application integrates three crucial external APIs, interacting with YouTube, Baidu, and Google's vision APIs, thereby extending its range of functions and services.

This system architecture aims to ensure high-speed content transmission, secure data storage, and efficient interaction with various third-party APIs. Each component has been meticulously designed to meet specific business needs and performance standards.

2.2 Database Entity Relation (ER)

There are 8 tables in the database that are being used in this project. Some of these tables have connections and dependencies while a few are stand alone tables. The ER diagram helps explain the connections between the diagrams. These connections would be used when the maintenance needs to make any changes to the design of the tables.

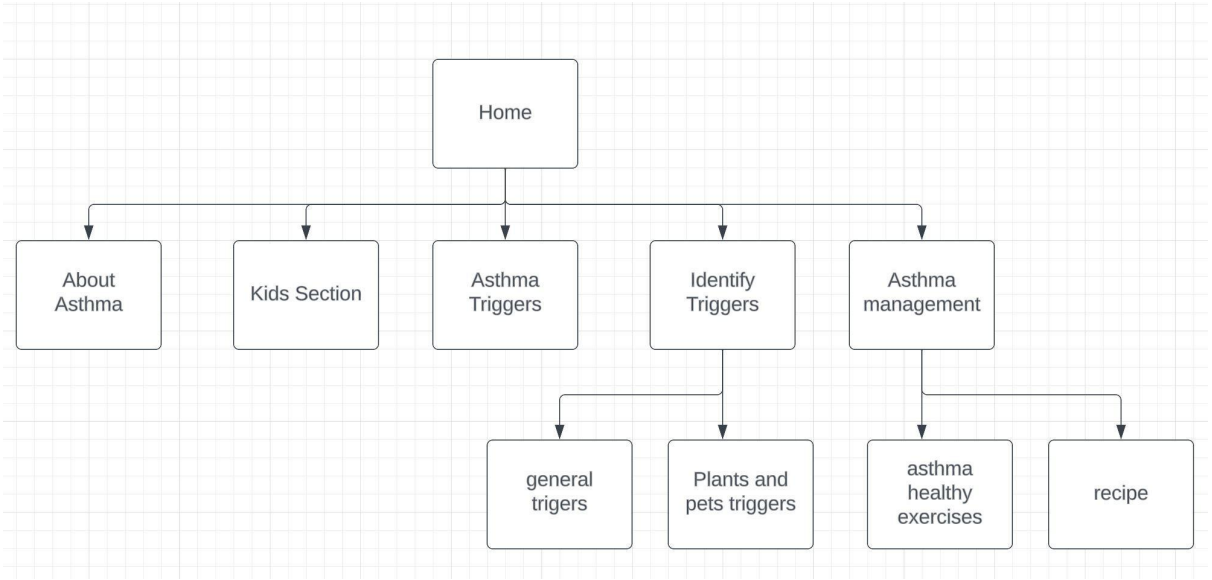


The schema of the data tables is stored in ‘seeing_asthma.sql’ file which is stored in the Database folder in PG folder google drive

3. Website Map

The entire website design and functionality is explained in this section. The maintenance team can understand the flow of the website using this map.

1.



- **Homepage**
  Path in development project: app/page.tsx
  This feature includes the introduction of asthma and our website.

If you would like to make changes to the homepage, all the elements are in the /Components folder. There is a global variable, top navigation bar in /Components/Navbar.tsx. You can manage tabs and related paths from there.

- **About Asthma**
  Path: /AsthmaLibrary/
  This feature includes authoritative professional introduction on childhood asthma. If you would like to make changes to this feature, code for sidebar is in /AsthmaLibrary/Sidebar.tsx and all related articles and corresponding tag is introduced in line 45-51 in /AsthmaLibrary/Sidebar.tsx. You can feel free to add, edit or delete articles and sidebar tags.

- **Thunderstorm asthma game**
  Path: /AsthmaGame
  This feature included a game and an animation video for kids to interact with and learn about thunderstorm asthma. The video was uploaded to Youtube and the game was developed with p5.js.

- **Asthma trigger library**
  Path: /TriggerLibrary
  This feature introduced possible asthma triggers in daily life especially plants, pets in Australia for parents to protect their kids from asthma.

- **Asthma trigger recognition**
  Path: /Trigger
  Path for subpages:
  - ➢ General trigger recognition: /Trigger/ImageRecognition.jsx
  - ➢ Plants and pets trigger recognition: /Trigger/PlantPet
    1. Plants:/Trigger/PlantPet/PlantImage.jsx
    2. Cats:/Trigger/PlantPet/CatImage.jsx
    3. Dogs:/Trigger/PlantPet/DogImage.jsx

  These features are used for users to recognise asthma triggers inside or outside their house. Users can take a picture and upload to the website to get recognition results. If users would like to make a further recognition of plants or pets, they can jump to the plants and pets function page then choose related tabs. If you would like to make changes to the feature, please follow the above paths and edit the code.

- **Asthma diet management**
  Path: /Diet
  This feature is developed to provide healthy recipes for users to add food that benefits recovering from asthma to their diet. Please feel free to edit files in /Diet/Cooking path to edit the feature.

4. Software & Hardware Details

4.1 Source Code Access

The details on how to access the source code for the project is given below:

- **Frontend:** Download the zip from Github, choose **Shiyu** branch
- **Backend:** Download the zip from Github, choose **python-api** branch

4.2 Frontend Software

This project is builded in Next.js framework: https://nextjs.org/. Next.js is a React framework that gives developers a block to build the web application. In order to use Next.js, please install Node.js 18.18.0 LTS version or later https://nodejs.org/en.

If you would like to make any change of the frontend, you have to:

- Know about HTML, CSS, JavaScript
- Know about Next.js which is a React framework
- We also used Chakra-ui in our development including some TypeScript coding. Please refer the following links to check for the template: https://chakra-ui.com/docs/components, https://chakra-templates.dev/

4.3 Backend Software

The backend was developed using Python. The code was stored in Github, choose **python-api** branch to download the zip. Please make sure you run the python code using Python 3.9.7 version. Then based on requirements.txt to install all packages.

The contents of requirements.txt are as follows:

```
mysql-connector == 2.2.9
pillow == 10.0.0
flask == 2.2.5
flask_restx == 1.1.0
pydantic == 2.1.1
flask_sqlalchemy == 3.0.5
Flask_Cors == 4.0.0
matplotlib == 3.7.2
boto3 == 1.28.23
requests~=2.31.0
```

For security purposes, the service runs on port 5000 of the local host by default, and then uses Nginx to reverse proxy the domain name of the backend api with port 5000 of the local host.

The backend service runs on AWS instances.

4.4 Database

The database is a MySQL Database which is hosted on Amazon Web Services (AWS) servers. The details to access the database are given in the support document.

4.5 Server/Hosting Requirements

Hardware Requirements:

CPU: 2.0 GHz or faster processor

RAM: Minimum 1 GB

Hard Drive: Minimum 15 GB

5. Database Changes

5.1 Database Migration

If the maintenance team wants to migrate the database into a separate server for security purposes, then all they need to do is enter the details of the new database server into the DataGrip tool before loading the data in section 3.3 of Support Document. Afterwards, DataGrip would be linked to the new server and all the information would be stored in that server.

If the maintenance team has migrated the database server, then they need to edit the details of the database in the config.py file so that the data query is passed to the new database server.

5.2 Data Updation

The maintenance team might need to update, insert or delete data to/into the database. This can be done by following the steps given here.

6. Test standard/ Test flow

Testing a system allows us to check how well our system performs as compared to the user requirements and does the application meet the acceptance criteria. In agile systems a feature is done when it meets a potential list of requirements including the user requirements and acceptance criteria as well as usability testing and security testing to be a shippable product. Testing of the system was conducted in the below flow.

Unit Testing:

We tested the individual components of the system all throughout the development process to maintain quality and reliability of the system.

Testers: Riding Jiang, Shiyu Wu, Sonia Lakhani

Test Process: Individual Component Test based on Acceptance Criteria in Leankit based on each user story.

Test Outcome: Continuous code improvements were done to ensure all the requirements in the user story and acceptance criteria were met.

Usability Testing:

To ensure that established user journey was fulfilled and user experience was improved upon to the best of our ability usability testing was conducted every iteration. Usability testing was performed by a cohort of our peers and a cohort of miscellaneous users to make improvements continuously.

Testers: Peers from IE studio, miscellaneous users.

Test Process: Recorded video of user testing the website and providing feedback.

Link to Peer usability testing:

Iteration1:
https://drive.google.com/drive/folders/1rPHBRmwjmm6Xp4vng33YmeXjoEcmUuUN

Iteration2:
https://drive.google.com/drive/folders/13B7DKSiAPfBfGHUJWxbCaoBWtg-8MQo0

Miscellaneous User Test:

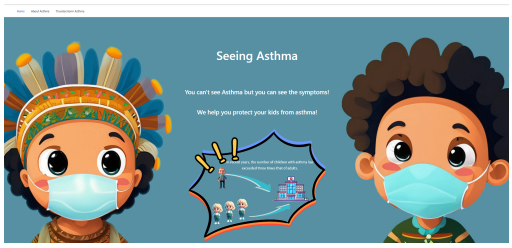
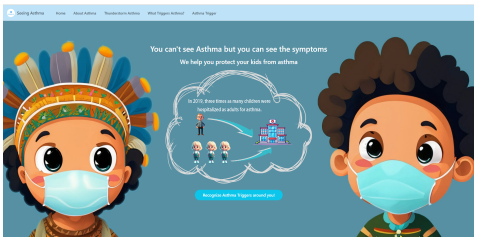
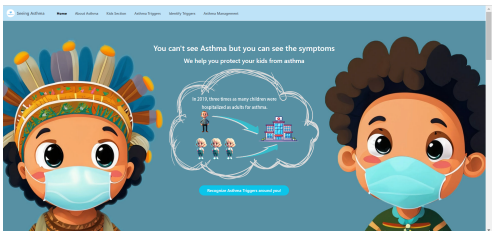
https://drive.google.com/drive/folders/1--jlm2tNLPQ6QAo3ZRva9MV806HbgNO9

Test Outcome:

Every Iteration, based on the usability test feedback changes were made to enhance the user experience. A log of changes was maintained for documentation linked below.

Changes:
https://docs.google.com/document/d/19fHj-BFzWuzLTsvgOi_uNwtvK4GEO7Df/edit

Homepage Example:

Iteration 1	Iteration 2	Iteration 3
		

Integration Testing:

End to end tests to check accuracy, quality, and functionality of the application were performed and suggestions and improvements were made to the final product.

Tester: Alex Shao

Test Process: End to end testing.

Test Outcome: Feedback given by the mentor was documented herein. Suggestions for UI changes and enhancing mobile friendliness etc. were made.

https://monashie.leankit.com/card/2018016392

Post Test Improvements: Changes made based on Test outcomes are documented and examples provided below.

1. https://monashie.leankit.com/card/2020033345
2. https://monashie.leankit.com/card/2020034046
3. https://monashie.leankit.com/card/2020034071

Security Testing:

Security is of paramount importance in Seeing Asthma and ensuring security was one of the main goals.

Tester: Sonia Lakhani

Test Process: To ensure the system was not vulnerable to attacks and was in fact secure a series of security tests and penetration tests were performed. These included tests like open ports check, tests for frontend vulnerabilities etc.

Test Link: The series of tests performed were documented in a report linked herein

https://docs.google.com/document/d/1w4-pBchh4HvUbcXgArXHqlNpM6v2z0YQ/edit

Test Results:

https://docs.google.com/document/d/1mgEXTPgMCVmWnCYDQqd7kuNiKk7Phi1N/edit

Test Outcome: A formalized report highlighting the potential vulnerabilities and their exploits, the consequences of these vulnerabilities and recommendations to mitigate them is linked herein.

https://drive.google.com/file/d/1hlR9rkq5J_nHxcN83MuQeWQDIFkPXoq8/view?usp=share_link