

CS5224 Cloud Computing

Final Report

Your Health Is In Safe Hands (HISH)

Tay Chao Jie	A0149818N
Ng Zhi Qing	A0205202A
Seah Yi Yi Chloe	A0204984X
Lim Kang Heng	A0206194H

Executive Summary	1
1 Business Case Formulation	1
1.1 Problem Statement and Project Motivation	1
1.2 Project Idea	1
1.3 Comparison with Available Services	1
1.4 Target Users	2
1.5 Cloud Utilisation	2
2 Cloud Service Overview	2
2.1 AWS Components & Architecture	2
2.1.2 Machine Learning	4
2.1.3 Database	4
2.1.4 Backend Operations	5
2.1.5 Frontend Operations	6
2.2 Value Proposition	6
2.3 Implementation Considerations	6
2.4 User Interaction	7
2.5 Technical Concerns, Limitations and Improvements	7
4 Economic Factors	9
4.1 Economic Benefits	9
4.2 Service Level Agreement	9
5 Cost Estimation and Revenue Model	9
5.1 Cost Estimation	9
5.2 On-Premise Costs	10
5.2 Revenue Model	11
6 Conclusion	12
References	12
Appendix	13
Appendix A: Screenshots of Web App	13
Appendix B: Work Distribution	17

Executive Summary

Medical diagnosis is the basis for any treatment but it is still considered a challenging task for medical professionals even with years of experience because of the complexity of diseases. A data driven diagnostic assisting tool can improve healthcare professionals' ability to deliver precise diagnosis. However, developing such tools is still a daunting task for many organisations due to the lack of training data, infrastructure to conduct intensive model training and regulations for data privacy. The Health In Safe Hands (HISH) system was developed as a novel solution to provide a feasible implementation of a machine learning aided diagnostic tool that could address the limitations mentioned earlier. The HISH system is a cloud based system that provides a Software as a Service (SaaS) with the following features: facilitates collaboration among different tenants to train a robust model, user friendly interface to conduct predictive diagnosis, data management system and a secure multi-tenant design that ensures strict isolation of individual tenant data. With the features above, HISH takes a step towards bridging the gap between healthcare entities by providing a shared diagnosis service that is trained with a collaborative corpus of data in a secure manner.

1 Business Case Formulation

1.1 Problem Statement and Project Motivation

Providing accurate medical diagnosis is still a huge challenge today even for doctors who have substantial experience in their fields of expertise. The complexity of different disease mechanisms makes it difficult to achieve early detection and treatment. Early detection enables early intervention which increases survival rate. Reports from the Canary Foundation showed the five-year survival for breast-cancer patients with early-stage disease is 98% and survival rates remain high at 10 years [1]. Furthermore, errors in medical diagnosis are still an issue today where higher costs are borne by patients. In the US alone, an estimated 5% of outpatients receive the wrong diagnosis every year. Lastly, the shortage of medical data to train a reliable decision model is still an issue today due to regulations and privacy of healthcare data. Therefore, our project seeks to alleviate these issues by using cloud services to implement a decision support system, incentivise data sharing in a regulated fashion and leverage machine learning to support medical diagnosis.

1.2 Project Idea

Machine learning assisted diagnosis promises to revolutionise healthcare by leveraging abundant patient data to provide precise and personalised diagnoses. This project aims to provide a platform to leverage data gathered from different healthcare entities and for medical professionals to efficiently use models to assist in diagnosing patients. HISH provides a user interface for healthcare workers to input, store, contribute, track patients records and provide specific diagnosis using a pre-trained model.

1.3 Comparison with Available Services

Different Software-as-a-Service (SaaS) providers have designed systems to help healthcare organisations manage electronic health records. However, each of these organisations stores their patient records on their chosen SaaS in siloes. The Ministry of Health has designed HealthHub to facilitate the sharing of patient records across different public hospitals and

polyclinics. The issue with Healthhub is its exclusive nature towards public healthcare groups only, therefore a lack of flexibility to integrate with private healthcare groups. HISH aims to bridge this gap by providing a platform for all healthcare organisations to store their patient data securely while benefiting from machine learning aided services trained with data across different entities.

1.4 Target Users

HISH is designed for healthcare organisations which operate hospitals and clinics. Each healthcare organisation will register with HISH and their medical staff will be able to utilise HISH to help them with their diagnosis.

1.5 Cloud Utilisation

A cloud-based design provides greater levels of elasticity, security and collaboration compared to the on-premise option. Table 1 below summarises the advantages of cloud computing as means for deployment.

Table 1: Summary of Cloud Attributes and Advantages

Elasticity	Predictive analytics model training using data has large processing requirements, which may be challenging for the on-premise infrastructure.
Data Storage	Ability to store large amounts of data at a lower cost than maintaining an on-premise infrastructure. Cloud also introduces redundancy and disaster recovery, where data is mirrored and can be retrieved under failure. This is crucial due to existing Ministry of Health guidelines, where patient data must be stored for years [2].
Multi-tenancy	The cloud infrastructure can support multiple instances to various healthcare groups, but operating in a shared environment using similar microservices. Each tenant's data will be isolated and will remain invisible to other tenants.
Security	User authentication ensures data accessibility to authorised personnel only. In addition, the model training ensures anonymity of patients by omitting Personal Identifiable Information (PII) variables and occlusion of variable names to protect the privacy of patients.
Collaboration	Due to the limited availability and sheer amounts of positive cases, collaboration amongst healthcare groups can improve the dataset for model training and testing.

2 Cloud Service Overview

2.1 AWS Components & Architecture

The architecture of HISH can be divided into five broad sections: web architecture, database, backend operations, frontend operations and machine learning. HISH is developed within a Virtual Private Cloud (VPC). Figure 1 illustrates the overall AWS architecture.

2.1.1 Web Architecture with AWS Services

The architecture of the web application can be further split into front-end and back-end. The front-end server is hosted in a public subnet connected to an Internet Gateway so that the web application can be accessed through the Internet. The web application is developed with React.js and deployed on AWS Elastic Beanstalk. AWS Elastic Beanstalk (EB) is chosen as it provides configuration of load balancing and scaling for the dynamic web application deployed. The front-end server is hosted on AWS Elastic Compute Cloud (EC2) instances. AWS CloudFront is used to route API requests from the frontend to the EB backend. The back-end is hosted in a private subnet and can send data to the internet through a Network Address Translation (NAT) gateway in the public subnet of the front-end. The back-end is developed using Flask and runs on EC2 instances. All EC2 instances are managed with auto-scaling and elastic load balancing to ensure availability and scalability. The number of EC2 instances required will depend on projections of user growth. Authentication of the web application is done using AWS Cognito, an AWS service for customer identity and access management. Upon registration of a new account, the information is sent to AWS Cognito using API calls. The username and password are added into AWS Cognito's User Pool, which is used for authentication during login later on.

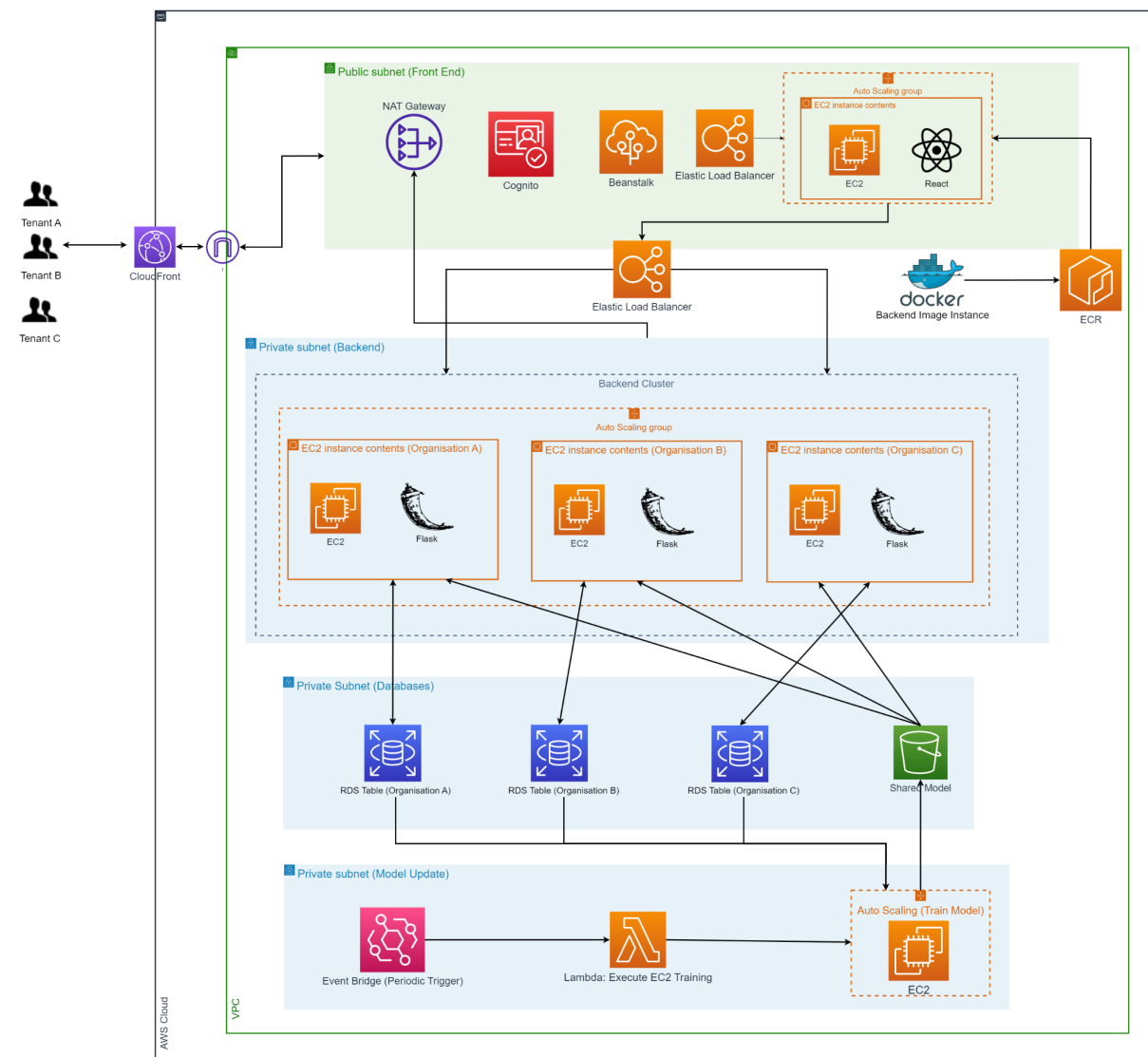


Figure 1: AWS Architecture

2.1.2 Machine Learning

The data from individual healthcare organisations is anonymised and jointly stored in a RDS table. When a doctor provides and confirms a diagnosis to a patient, the doctor can submit this diagnosis to a joint table. This joint table will then fetch the patient's biomarkers from the individual organisation's table and store them. Patient identifiers will not be stored in this joint table to ensure anonymity in the model training process.

The data from the joint table is then used to train machine learning models to identify patients who might have a particular health condition or disease based on their health data. To ensure recency in the predictive model, a periodic training routine was developed to ensure the shared model is updated with new insights from new cases. This is achieved by creating a data pipeline using AWS Lambda that is triggered by a periodic event initiated by AWS Cloud Bridge. The AWS lambda turns on an EC2 autoscaling cluster to initiate training. A new model is then uploaded to the S3 shared bucket once training is done. This model generates predictions on data from all healthcare organisations. The RDS data store is populated by the various tenants over time with new closed cases which provides opportunities to further improve the existing model.

A public dataset was chosen to simulate breast cancer records to test the workflow. The Breast Cancer Wisconsin (Diagnostic) dataset was used which contained structured features that were computed from a digitised image of a fine needle aspirate (FNA) of a breast mass [3]. The dataset emulates a binary classification problem, with 569 entries and 30 different features. To simplify the problem, a subset of 8 features were used for training and prediction. An Xgboost model was selected as the model for deployment and training because it was deemed to be sufficiently complex for this classification task. The Xgboost is a gradient boosted tree and it is capable of achieving a F1 validation score of 0.99 when trained for this binary classification task. We used this dataset as dummy medical entries to test our application.

2.1.3 Database

The database is designed according to the principles of multitenancy, data partitioning and tenant isolation. To comply with regulations such as HIPAA, a healthcare organisation can only access its own data. Data is stored in AWS Relational Database Service (RDS) PostgreSQL, a relational database service. Multitenancy can be implemented on RDS easily and cost-effectively through creating a table for each tenant under a shared database instance. A tenant, or one healthcare organisation, is assigned to a table upon signing up. There can be multiple users (doctors) under each healthcare organisation who can each sign up for an account. The doctors can log in with their accounts to upload data to the database and fetch predictions from our machine learning model.

Three different schemas are used in the database as shown in Figure 2 below. *userDetails* associates each user to an organisation which is used to identify the corresponding table accessible by a user. *tenant table* shows the attributes of the table created for each organisation and the name of the table will be of the form '*{organisationName}_Data*'. The organisation's table stores the personal information and medical records of their patients. *joint_table* stores the details of each medical record without a patient's personal identifiers and will be used to train machine learning models to support the doctor's medical diagnosis.

Each user will be granted permissions to insert and update *joint_table* and view, insert, update and delete entries of their organisation's table in the database instance. The user will not be able to view the data stored in *joint_table* and *userDetails*.

userDetails	tenant table	joint_table
userName	patientid	entryID
password	firstName	textureMean
organisationName	lastName	symmetryMean
	dateOfBirth	areaWorst
	dateOfService	areaSE
	areaCode	areaMean
	phoneNumber	concavityWorst
	remarks	concavitySE
	concavityMean	concavityMean
	concavitySE	diagnosis
	concavityWorst	dateofclosure
	areaMean	
	areaSE	
	areaWorst	
	symmetryMean	
	textureMean	
	prediction	
	diagnosis	
	dateOfClosure	

Figure 2: Database Schema

Storage auto scaling is enabled to automatically scale up when storage demand increases to provide scalability and elasticity to the database. In this project, the Learner's Lab account is used to set up the RDS instances and thus only the Free Tier option is available. One RDS instance is created with no standby instance. The RDS saves snapshots of the database every day and stores them in Amazon S3. It also copies transaction logs to S3 up to every 5 minutes. The transaction logs can be used for point-in-time recovery of the RDS instance. However, outside of Learner's Lab, Amazon RDS can be configured with a standby RDS instance in a different availability zone to provide higher availability.

2.1.4 Backend Operations

The backend enables all operations for the designed features to work. Some operations include interaction with databases, prediction of input data and getting frontend inputs. As the number of users for the application increases, it is key that our backend instances are properly scaled and load balanced to achieve efficient resource utilisation with high availability. Hence, our EC2 instances are set with auto scale and load balancing to achieve the above mentioned properties. These capabilities were configured directly through the AWS Elastic Beanstalk interface.

The backend development is dockerized and stored in the AWS Elastic Container Registry (ECR). Although the AWS Elastic Beanstalk provides support for Flask (python) applications, using Docker allows for greater flexibility to switch to another cloud provider service in the event that there are any external requirements or legislative mandates. As such, all required dependencies, environment configurations and directories have been pre-defined for improved portability.

2.1.5 Frontend Operations

The build folder of the React development is stored in an S3 bucket. The bucket is linked to the AWS CloudFront Content Delivery Network (CDN) distribution to improve performance of frontend loading times, enable HTTPS and route requests to the Elastic Beanstalk backend.

2.2 Value Proposition

Training a decision making support system requires a huge amount of data which is usually an issue to attain by a single healthcare entity. HISH enables different healthcare institutions to manage their own data securely but benefit from machine learning models trained from data gathered across all healthcare organisations. This is particularly crucial in the healthcare setting, as the number of positive cases is often disproportionate to the number of negative cases. By integrating the dataset, it allows for higher recall in the model predictions.

2.3 Implementation Considerations

The user input form was designed to contain three portions: patient details, medical information and medical diagnosis, such that it is reflective of the process during a doctor's consultation. This creates ease of user experience on the doctor's end, as well as ease of integration with the backend.

For the database multi-tenancy design, a shared database and separate table approach was used. This was deemed to be the best option compared to the other two possible designs due to the sensitive nature of the data being stored and the cost required for implementation. There are 3 possible designs for database-level multitenancy. Shared database and shared table is the easiest to set up but provides the lowest security guarantee. Separate database provides the highest level of security but is too costly to implement as a new database instance has to be created for each new tenant. In the shared database separate table design, tenants share a database instance but they are still isolated to their respective tables. This provides a good trade-off between database security and cost. To achieve isolation with a large number of users from multiple tenants (organisation), the RDS assigns roles and privilege rights to only access the tables that belong to their respective organisations while only providing write privileges to the joint_table.

For the model training, we avoided using AWS Lambda only to conduct the training of the shared ML model due to the possible sheer size of data, long runtime and also the computational resources required. Using EC2 also opens up more cheaper pricing options such as spot or reserved instances since the training is triggered on a periodic schedule and could take a long time to train. The downside of using EC2 alone is the redundant cost incurred when the instances running are idle. Hence, our design draws on the benefits and

strengths of both platforms to architect a training line that uses Lambda events to trigger EC2 instances to start, train and stop.

2.4 User Interaction

Figure 3 illustrates the user interaction with the HISH system (see Figure A.1). The diagram depicts several actions that the user can execute, and what happens in the backend correspondingly. When the user signs up for an account according to their tenant group, the user details are added into AWS Cognito for authentication later on (see Figure A.3). When the user logs in with their registered details (see Figure A.2), these details are passed to AWS Cognito for authentication before being allowed to execute the next action - input patient information (see Figure A.4). When the user inputs patient information, the patient's medical information is used as inputs for prediction with the classification model (see Figure A.5 and A.6). The information is also updated in the respective isolated table that the hospital's organisation is in. When the case is updated to be closed, the data is then anonymised and pushed into the joint table for training of the model (see Figure A.7 and A.8).

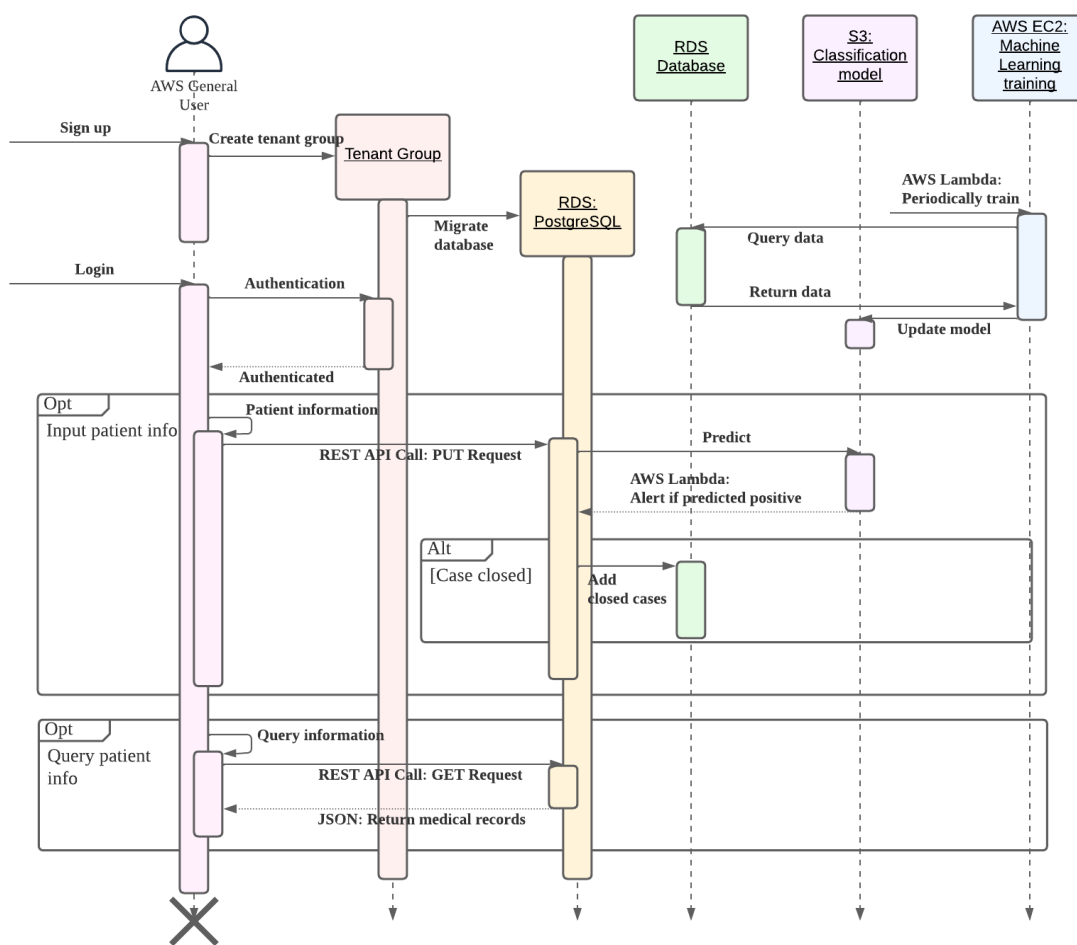


Figure 3: User Interaction Diagram

2.5 Technical Concerns, Limitations and Improvements

A preferred approach to store the training dataset would be in an AWS Healthlake environment, which was part of our original design but not implemented due to constraints in the Learner Lab environment. The AWS Healthlake was specifically designed to handle healthcare related datasets and complies with the HIPAA (The Health Insurance Portability

and Accountability Act). Healthlake also supports a wide array of unstructured data types such as Fast Healthcare Interoperability Resources (FHIR) files which is definitely more flexible compared to our current design which uses a relational database. Furthermore, Healthlake opens up more room for model improvement through feature transformations and extractions using specialised ML models that have been trained to understand and extract meaningful information from unstructured healthcare data. For example, a useful extension would be to store mammogram images of the breast in an object-oriented database, and to build a multi-modal classifier based on the doctor's relational input and the medical images. Therefore, integrating healthlake into the design would potentially make the product more attractive and relevant to healthcare groups.

Another aspect of the current design that could have been improved was the model training pipeline. The current design runs on an autoscaled EC2 instance which requires a lot of manual configurations, setup and custom coding to establish a training pipeline. Using AWS Sagemaker makes ML model training far more simplistic due to the presence of pretrained models and proprietary hyperparameter optimization functions. AWS Sagemaker is also cheaper, and the EC2 instances Sagemaker is built on top of are optimised for ML in general. Hence, Sagemaker could easily spin hundreds of training jobs with just a few lines of code, which would take months to implement on bare EC2s. Sagemaker would henceforth significantly reduce the cost of development. However, Sagemaker cannot be used in Learner's Lab due to IAM restrictions.

Figure 4 below shows a potential AWS design with the additional improvements mentioned.

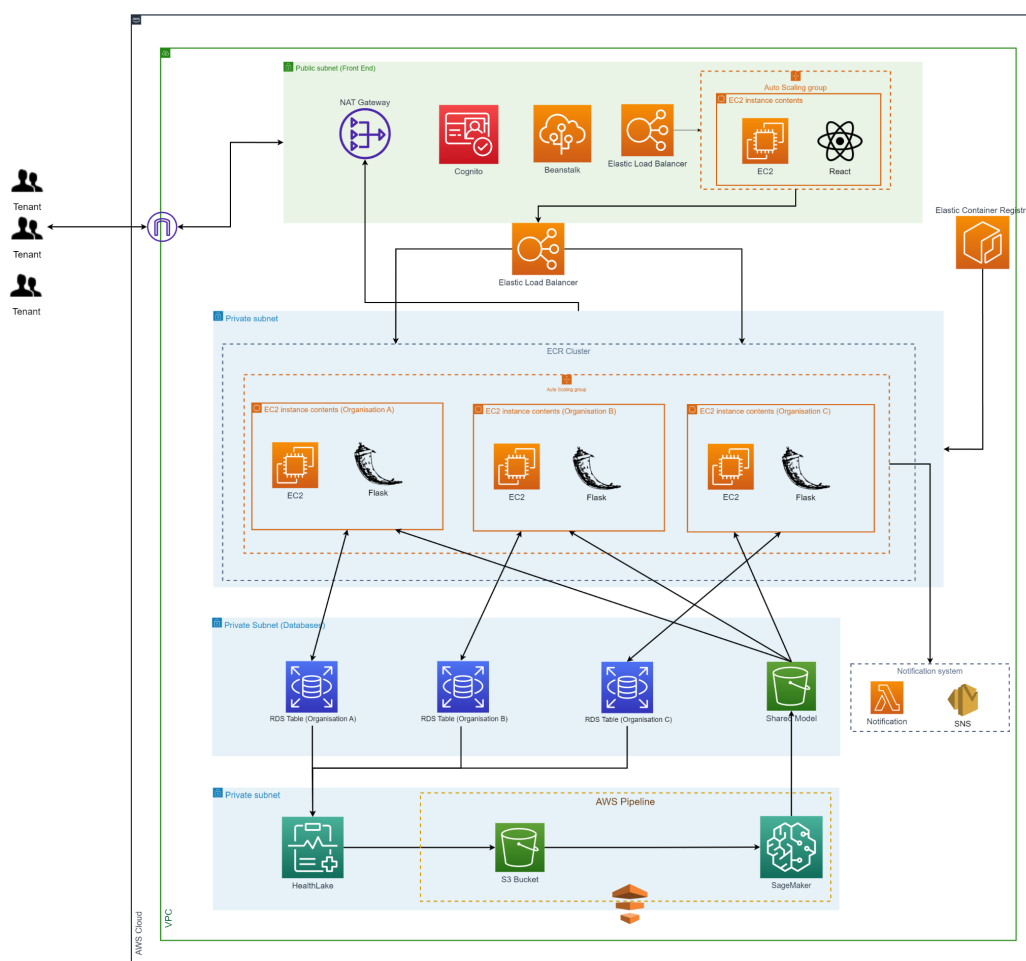


Figure 4: Improved AWS Architecture

4 Economic Factors

4.1 Economic Benefits

This product contributes positively to public health because it aids healthcare workers to make better decisions during diagnosis. Apart from that, this SaaS service promotes collaboration among healthcare groups in a safe manner and could facilitate future medical machine learning research using the collective data that adheres to data protection requirements.

4.2 Service Level Agreement

Some important service level agreements (SLAs) to consider include: uptime, response time, accuracy, data security, privacy and support. Table 2 below summarises details of each SLA factor. For actual deployment, a quantified value should be given to tenants for a more objective understanding of the SLA (e.g., 99.995% uptime).

Table 2: Summary of SLA Factors

Uptime	The availability of the application by use of the doctors. As hospitals run 24/7 operationally, HISH has to have a high uptime SLA to ensure that doctors can access the application at any time when required.
Response Time	The time taken for the application to respond to user requests. In the event of medical diagnosis application, it is crucial to have low response time SLA to ensure that diagnoses can be provided promptly, and data can be stored quickly under a low latency environment.
Data Security	High security and privacy SLA is required to ensure that user data is protected at all times. In this application, data has to comply with legal and regulatory compliances such as HIPAA.
Support	Necessary support has to be provided to the users of the application. This includes necessary documentation, training and access help when needed.

5 Cost Estimation and Revenue Model

5.1 Cost Estimation

The cost estimation of the SaaS implementation was done assuming the host region is US East (N Virginia/us-east-1a). Additional assumptions were made to facilitate the monthly estimates: every tenant contributes up to 100 uploads per day (ie. constant workload daily), model training is only done weekly, the training EC2 instances are spot instances while the server hosting EC2 instances are either on demand or reserved. This ensures that our SaaS service runs on a reliable source of instances and the reserved scheme helps to save some level of cost. Spot instances are great for the training instances because it is not particularly time critical and significant updates of the model are accumulated over a long period of time.

For AWS Lambda, it is used for triggering tasks which do not require a significant amount of resources or uptime. Hence it is assumed that the total amount of memory usage is only 1024MB and the total uptime per month is 12 hours maximum. Therefore, a total of 44236 GB-seconds.

For AWS Cognito, we assumed it to be free since it offers free services for the first 50 thousand users. Table 3 below shows the monthly average cost of this design. Note that one month is 750 hours approximately.

For AWS RDS, the t4g.medium instance is chosen which provides 4GB of memory. Due to the limitations of Learner's Lab, single AZ deployment is adopted. Since the RDS instance is required to be always available, a reserved instance would provide maximum cost savings. 1 year Reserved Instance with no upfront payment is chosen to provide more flexibility.

Table 3: Cost of AWS per month

Components	Cost	Type	Remarks
EC2 (Front end & Backend)	Free	t2.micro	
EC2 (Training)	$0.0336 \times 750 \times 0.5 = 12.6$	t4g.medium	Spot instances: assume 50% discount
S3	$0.023 \times 5 + 0.005 \times 2 + 0.004 \times 20 = 0.133$		5 GB standard storage, 20 000 GET, 2000 PUT
Lambda	$44236 \times 0.0000166667 = 0.74$		
Elastic Load Balancer	$0.025 \times 750 \times 2 = 37.5$		Application Load Balancer
Cognito	Free		First 50k user
RDS	$0.047 \times 750 = 35.25$	Postgres, db.t4g.medium	Reserved instance, 1 year, no upfront
Total Price (\$USD)		\$86.23	

While the above capitalises fully on the free tier privileges provided by AWS, the cost will definitely be higher when the application demands higher volumes of computational resources as the service increases in demand.

The monthly cost of using the AWS services amounts to USD\$86.23 and this is excluding development or platform maintenance cost. Assuming that the cost of development and maintenance cost per month is USD\$500. The overall monthly cost is approximately USD\$586.23.

5.2 On-Premise Costs

On-premise costs can be approximated with the total cost of ownership (TCO) of a datacenter. TCO accounts for the depreciation of the datacenter and server and the operating expenditure of running the datacenter and servers, such as electricity, maintenance cost and salaries. We approximate the depreciation of the datacenter and the servers with the cost of a

server. A Dell PowerEdge R450 Rack Server costs around USD\$5663.37, with 8GB of memory, 600GB of storage, power rating of 105W and a 3 year warranty [5]. Assume that this server has a lifespan of 3 years. We expect higher maintenance costs for on-premise resources and approximate it to be USD\$600. The average monthly cost will be about USD\$841.55. Hence, by running HISH on cloud, we enjoyed cost savings of around 31.1%.

Table 4: On-Premise Cost per month

Components	Cost	Remarks
Server	157.32	Total cost of 5663.37, assuming a lifespan of 3 years
Maintenance	600	Higher maintenance cost than AWS
Electricity	$30 \times 24 \times 0.105 \times 0.23 = 17.39$	Server has power rating 105W, electricity tariff about USD0.23 per kWh [6]
Network	66.84	Cost of business networking plan [7]
Total Price (\$USD)	841.55	

5.2 Revenue Model

The pricing model of the service would be based on a subscription plan basis. Having different types of subscription provides more flexibility and caters to the needs of different organisations and their varying levels of commitment to this product. Using subscription plans also allows for unlimited use of the service for healthcare professionals, yet still maintaining a stream of income on the business' end to generate revenue for further improvements and maintenance of the service. Moreover, a subscription plan incentivises maximum utilisation from the consumer, which is also beneficial for HISH as more data can be collected to train the machine learning model. Potential issues could, however, arise from such a pricing model as for most subscription based products is the startup phase. As a new service, there are no statistics on its impact on healthcare efficiency. Thus, some considerations to waive fees for new subscribers for a period of time could help healthcare institutions test the program and bring them on board the service. Another potential issue that this pricing model can face is the retention of customers and cancellation rates. This usually occurs when customers feel that the product is lacking in value progressively as time goes and eventually deems it irrelevant if there are no updates. This service could therefore work on being more flexible and innovative by providing a diversity of disease diagnosis or new features like collaborative disease diagnosis to solve more complicated problems.

Applying industry standards of a reasonable 20% profit margin, the subscription fee for this software should be charged at \$703.48 per month. The subscription plan is shown below in Table 5, there is the 1 year plan (Standard) and monthly plan (Try it out). To incentivise consumers to choose the standard package, the monthly plan will be slightly more expensive and also limits the variety of diagnosis models available.

Table 5: Subscription Plan

Package	Duration	Cost
Standard	1 year	\$8490/yearly
Try it Out	1 month	\$750/month

6 Conclusion

A collaborative data driven medical diagnosis tool is promising but challenging to implement due to the sensitive nature of the data, privacy and profit driven motive by individual healthcare groups. HISH is a proof of concept of a cloud based architecture that capitalises on a multi-tenant design to enable secure collaboration of healthcare tenants and mutually benefit through our SaaS. Our work contribution includes: AWS architecture to implement a practical cloud system to solve the problem statement, multitenant database design to ensure isolation of tenants, automatic machine learning pipeline that trains a model with incoming data, user interface to facilitate prediction using a shared model and using cloud services to achieve a highly available, cost efficient and scalable SaaS.

References

- [1] C. Y. Yeh, R. Adusumilli, M. Kullolli, P. Mallick, E. M. John, and S. J. Pitteri, "Assessing biological and technological variability in protein levels measured in pre-diagnostic plasma samples of women with breast cancer," *Biomarker Research*, vol. 5, no. 1, 2017.
- [2] *Licensing terms and conditions on enhanced nursing home standards*. (2015, April 1). Retrieved February 22, 2023, from [https://www.moh.gov.sg/docs/librariesprovider5/licensing-terms-and-conditions/licensing-terms-and-conditions-on-nursing-homes-\(circular-dated-1-apr-2015\).pdf](https://www.moh.gov.sg/docs/librariesprovider5/licensing-terms-and-conditions/licensing-terms-and-conditions-on-nursing-homes-(circular-dated-1-apr-2015).pdf)
- [3] P.M.Murphy and D.W.Aha. UCI repository of machine learning databases, 1992. www.ics.uci.edu/ml/MLRepository.html.
- [4] AWS pricing calculator: <https://calculator.aws/>
- [5] Poweredge R450 Rack Server: Dell Singapore. Dell. (n.d.). Retrieved April 9, 2023, from <https://www.dell.com/en-sg/shop/servers-storage-and-networking/poweredge-r450-rack-server/spd/poweredge-r450/per450tm01sgoo>
- [6] *Regulated tariff*. EMA. (2023, April 1). Retrieved April 9, 2023, from https://www.ema.gov.sg/Residential_Electricity_Tariffs.aspx
- [7] Best Business Broadband Promotions & Deals Singapore. Singtel. (n.d.). Retrieved April 9, 2023, from <https://www.singtel.com/business/promotions/business-fibre-broadband-offers>

Source Codes: <https://github.com/chloesyy/cloud-computing/tree/main>

Website: <https://d18mq98qvad2m0.cloudfront.net/>

Appendix

Appendix A: Screenshots of Web App

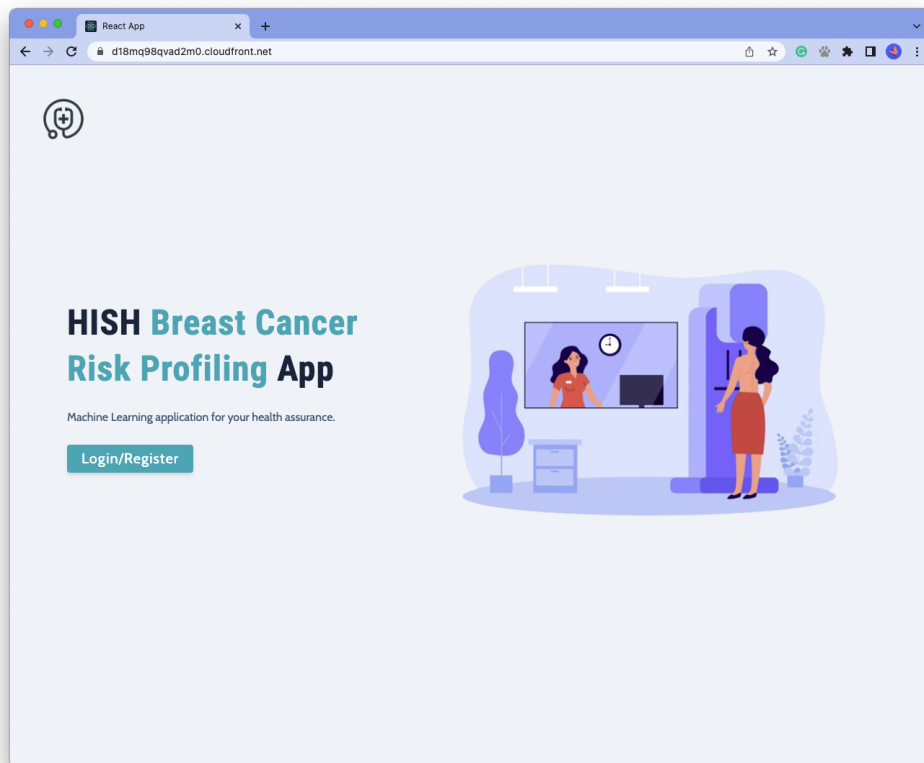


Figure A.1 Landing Page

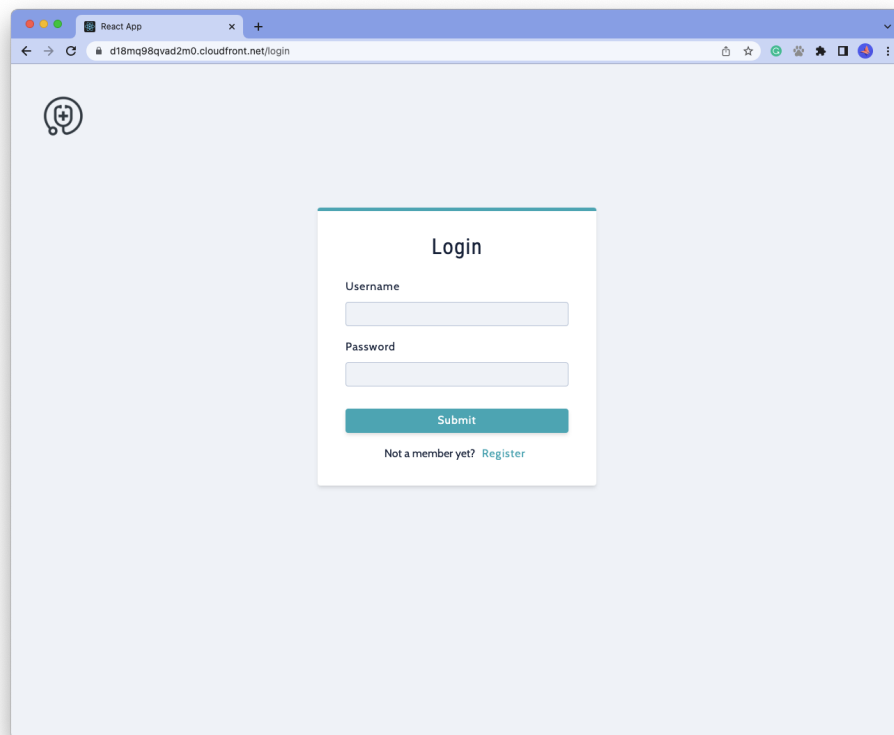
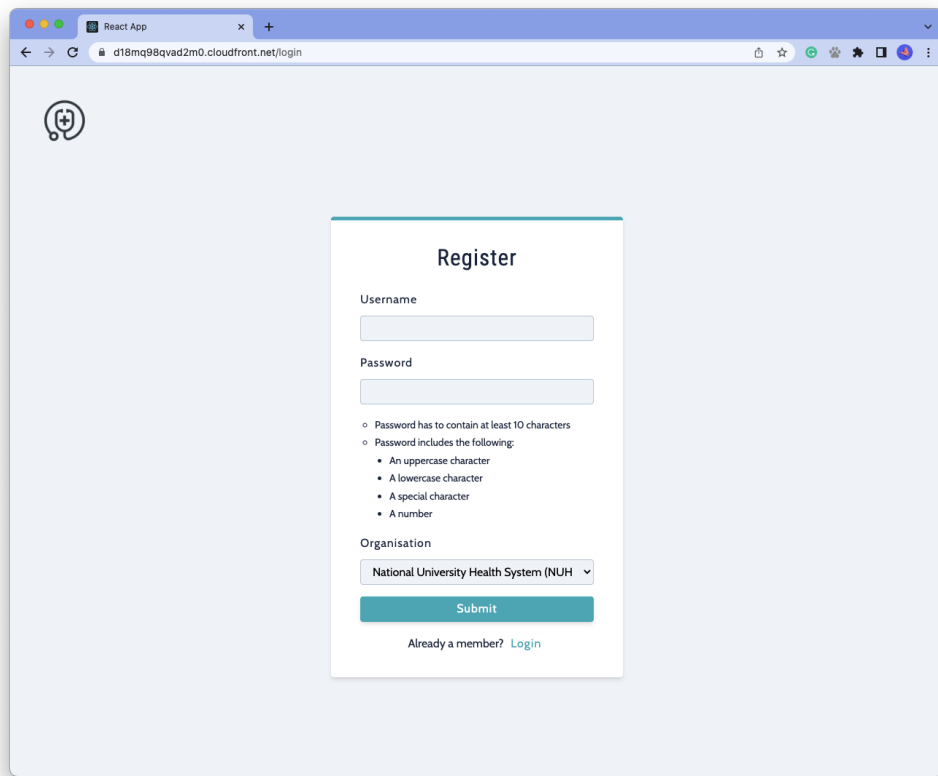


Figure A.2: Login Page



A screenshot of a web browser displaying a registration form. The browser's address bar shows the URL "d18mq98qvad2m0.cloudfront.net/login". The page has a light blue background with a small circular logo in the top left corner. The registration form is centered and titled "Register". It includes input fields for "Username" and "Password". Below the password field, there are two lines of text: "Password has to contain at least 10 characters" and "Password includes the following:", followed by a bulleted list: "An uppercase character", "A lowercase character", "A special character", and "A number". There is a dropdown menu for "Organisation" with "National University Health System (NUH)" selected. A teal "Submit" button is at the bottom of the form. Below the button, it says "Already a member? [Login](#)".

React App

d18mq98qvad2m0.cloudfront.net/login

Register

Username

Password

Password has to contain at least 10 characters

Password includes the following:

- An uppercase character
- A lowercase character
- A special character
- A number

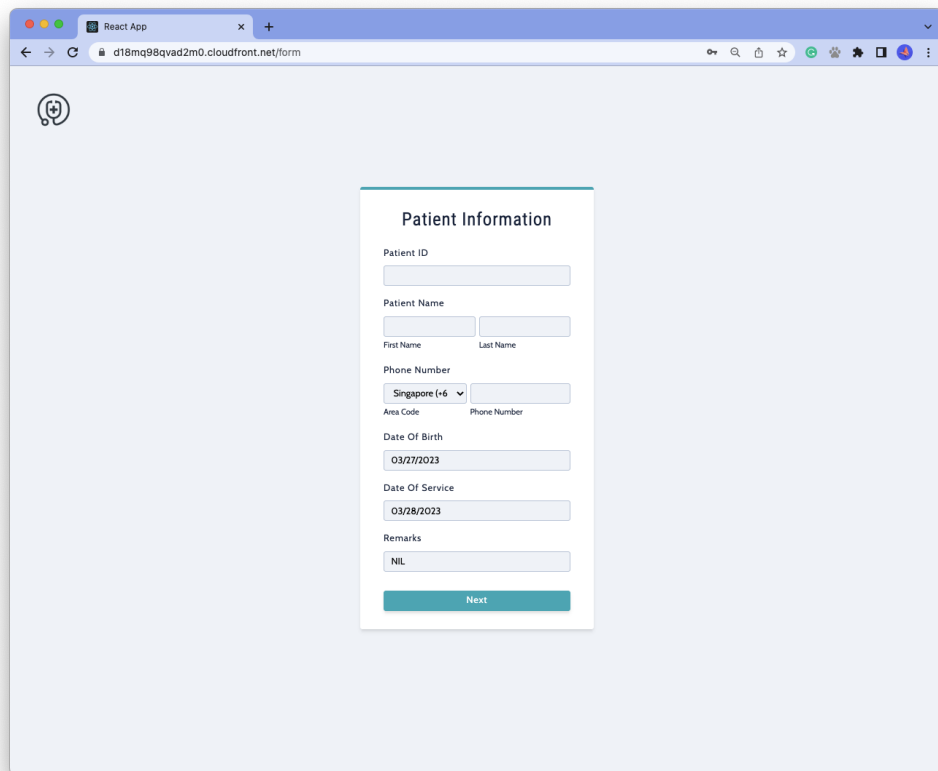
Organisation

National University Health System (NUH)

Submit

Already a member? [Login](#)

Figure A.3: Registration Page



A screenshot of a web browser displaying a patient information form. The browser's address bar shows the URL "d18mq98qvad2m0.cloudfront.net/form". The page has a light blue background with a small circular logo in the top left corner. The form is titled "Patient Information" and includes input fields for "Patient ID", "Patient Name" (split into "First Name" and "Last Name"), "Phone Number" (with a dropdown for "Area Code" showing "Singapore (+6)" and a "Phone Number" field), "Date Of Birth" (with a date picker showing "03/27/2023"), "Date Of Service" (with a date picker showing "03/28/2023"), and "Remarks" (with a text area containing "NIL"). A teal "Next" button is at the bottom of the form.

React App

d18mq98qvad2m0.cloudfront.net/form

Patient Information

Patient ID

Patient Name

First Name Last Name

Phone Number

Singapore (+6)

Area Code Phone Number

Date Of Birth

03/27/2023

Date Of Service

03/28/2023

Remarks

NIL

Next

Figure A.4: Patient Information Page

A screenshot of a web browser displaying a "Medical Information" form. The form is centered on a light blue background. It contains ten input fields for the following attributes: Concavity Mean, Concavity SE, Concavity Worst, Area Mean, Area SE, Area Worst, Symmetry Mean, and Texture Mean. At the bottom of the form are three buttons: a red "Predict" button and two teal buttons labeled "Back" and "Next". The browser's address bar shows the URL "d18mq98qvad2m0.cloudfront.net/form".

Figure A.5: Medical Information Page

A screenshot of the same "Medical Information" form, but with a dark gray background. A white modal window titled "Prediction Results" is overlaid on the form. The modal contains the text "Negative" and "Positive Confidence Score: 0.04". The form behind the modal is dimmed, and the "Predict" button is now grayed out. The browser's address bar and logo remain the same.

Figure A.6: Prediction Message

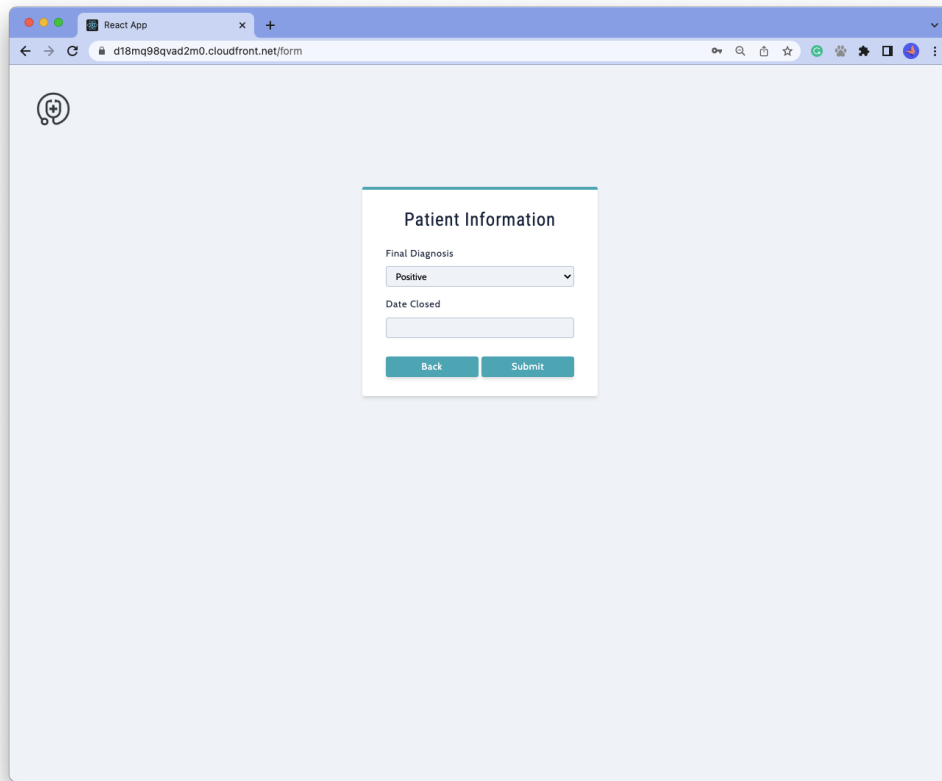


Figure A.7: Diagnosis Page

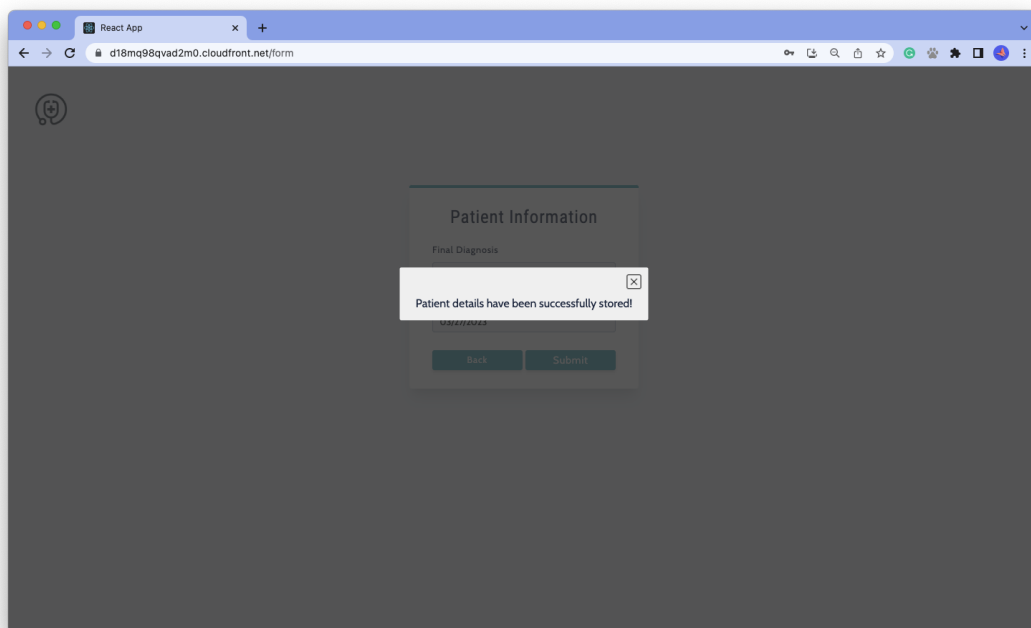


Figure A.8: Data Storage Success Message

Appendix B: Work Distribution

Table B.1: Work Distribution

Task	Done By
Front-End	Kang Heng
Back-End	Chloe
Database	Zhi Qing
Machine Learning	Chao Jie
Report	All
Poster	All