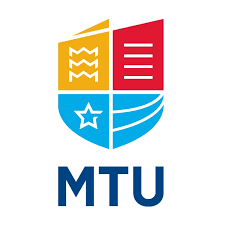
Project Report

Sean Long

Damian Gornik

Cian Ó Cathasaigh

Kian Cliffe



# Table of Contents

## **Introduction**

## **Requirements Specification**

### [Web Application](#_hsk61idyfjha)

### [Android Application](#_vt1khh5u4h3s)

### [Medi-AI](#_kssmhjh0j05o)

### [Blockchain & Utility Token](#_5zzoqrqlg279)

## **Prototypes**

### [Web Application](#_9vos3bz0mdm1)

### [Android Application](#_15dbz38n52bk)

### [Blockchain & Utility Token](#_ubl0ex5aox4e)

## **Design and Implementation**

### [Web Application](#_zd58g3c5qoqo)

### [Android Application](#_48cgayl8g0x0)

### [Medi-AI](#_5sjehh1kc82j)

### [Blockchain & Utility Token](#_id0sg3dgtdez)

## 

## **Testing**

### [Web Application](#_bqsnym327ryt)

### [Android Application](#_z8kycn553erd)

### [Medi-AI](#_o6hg3fz9lqbs)

### [Blockchain & Utility Token](#_5cipp3ljrxj8)

## **Conclusion and Future Work**

### [Web Application](#_pe7ne3qcu4sz)

### [Android Application](#_t8yupr5bcbne)

### [Medi-AI](#_71vefvhjm4b3)

### [Blockchain & Utility Token](#_jzf4dkmyv0jg)

# Introduction

The motivation behind the software was to develop a full-scale platform for medical professionals and patients to predict future health problems using AI technologies and datasets of patient information, supported by an ICO (Initial Coin Offering) to fund a start-up.

The phrase 'prevention is better than cure' is still up-to-date. Prevention costs less than curing, both health and money. Given the context, the problem we tackled was the lack of tools on the market to predict the likelihood of someone contracting the medical conditions of heart disease, diabetes and Alzheimer's disease. To make the health risk detection process simpler.

Using AI technology a doctor can take a patient's medical information and determine what , if any, health issues they are at a risk of developing. The necessary health information is collected from the patient by the way of an android application which then returns to a user a prediction statement, this application also provides the patient with the support necessary for contacting their personal healthcare professional as well as the paying of their insurance with a utility token.

For the doctors, we provide a web portal from which they can manage their patients and their respective health predictions, as well as handling datasets used by the predictive Artificial Intelligence.

Patient data is to be stored in a secure database as well as in an immutable blockchain, this data is only accessible to the medical professionals to whom the patient is assigned, these access restrictions exist to insure maximum privacy for the patient.

We hope to achieve a seamless user experience for both patients and healthcare professionals alike across the Medi-App and Medi-Web services, we aim to transform the way healthcare management is carried out and to provide help to patients before any life threatening conditions are able to manifest.

**Tools and technologies used:**

***Web Application***

1. Python
2. Django
3. Firestore Database

***Android Application***

1. Java
2. Android studio
3. Chaquopy

***Medi-AI***

1. Python
2. ScikitLearn
3. Pandas and numpy
4. Pickle

***Blockchain & Utility Token***

1. Python
2. Solidity
3. Infura
4. Web3py
5. Chaquopy
6. Ganache

# 

# Requirements Specification

### **Web Application**

**Web app should include:**

• Registration of medical professionals using email and password

• Access Restrictions: professionals are only able to see the details of their patients

• Risk Profiles Interface allows professionals to run reports establishing levels of risk for all categories

• User Profiling Interface allows professionals to view their patient risk profiles

• Aggregation of new patient data. Using new patient information, the administrator has an option to create and export new datasets by extending those used by Medi-AI

### **Android Application**

**Android app should include:**

• Registration of patients using email and password

• Patients can log in using email and password

• Patients can contact GP and insurance through forms in the app

• Patients can change their GP and insurance provider through the app

• Patients can pay insurance premium with Medi-Coin

• Patients can submit medical history for MediAI to analyse

• Patients can view Medi-AI predictions

• Patients can contact technical support through the support form

• Patients can leave a review on the app store

### **Medi-AI**

**Should include:**

• Heart Disease model

• Diabetes model

• Alzheimer’s model

• The ability to load these models and predict whether a patient is likely to develop heart disease in the future based off current health indicators

• The ability to report and display the accuracy of each model

### **Blockchain & Utility Token**

**To be included:**

• A simple blockchain and ICO

• Crypto to be built on top of the Ethereum blockchain

• Token be used to purchase insurance

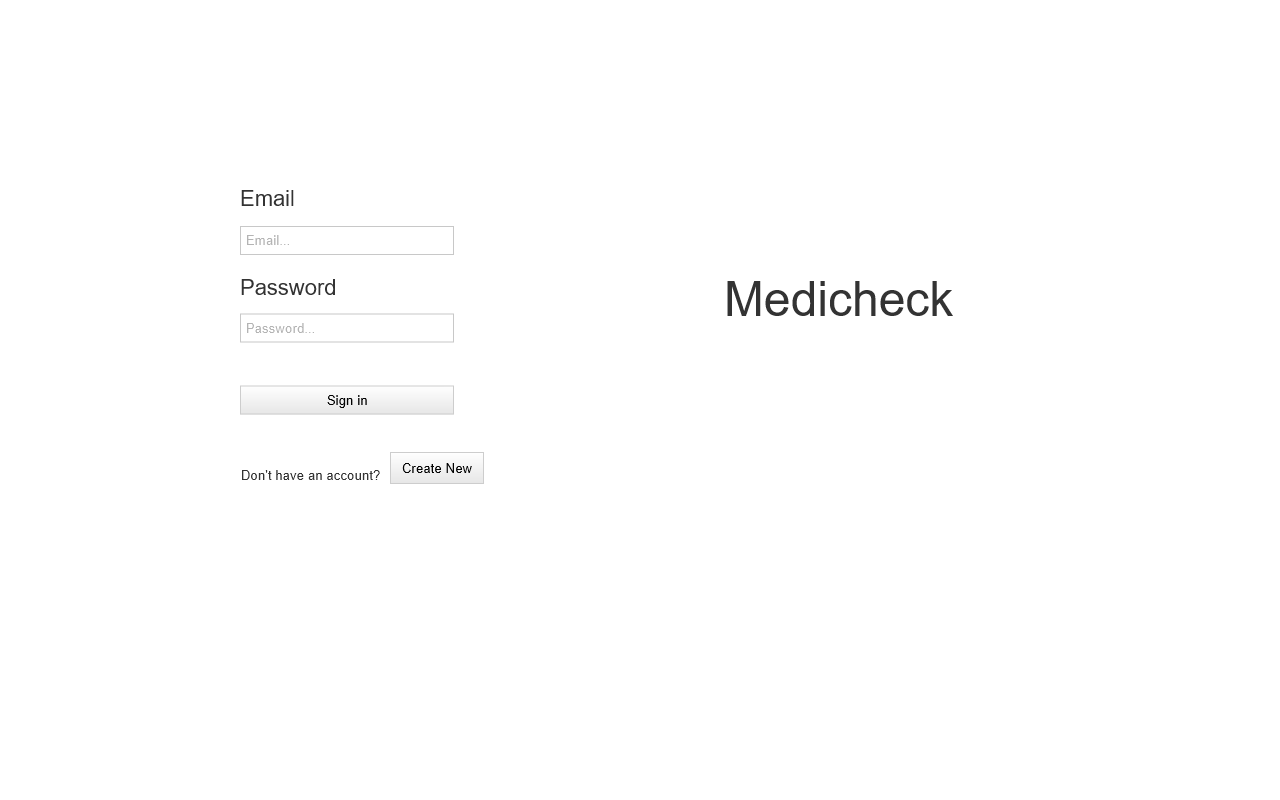
• Blockchain to be used for storing patient data

# 

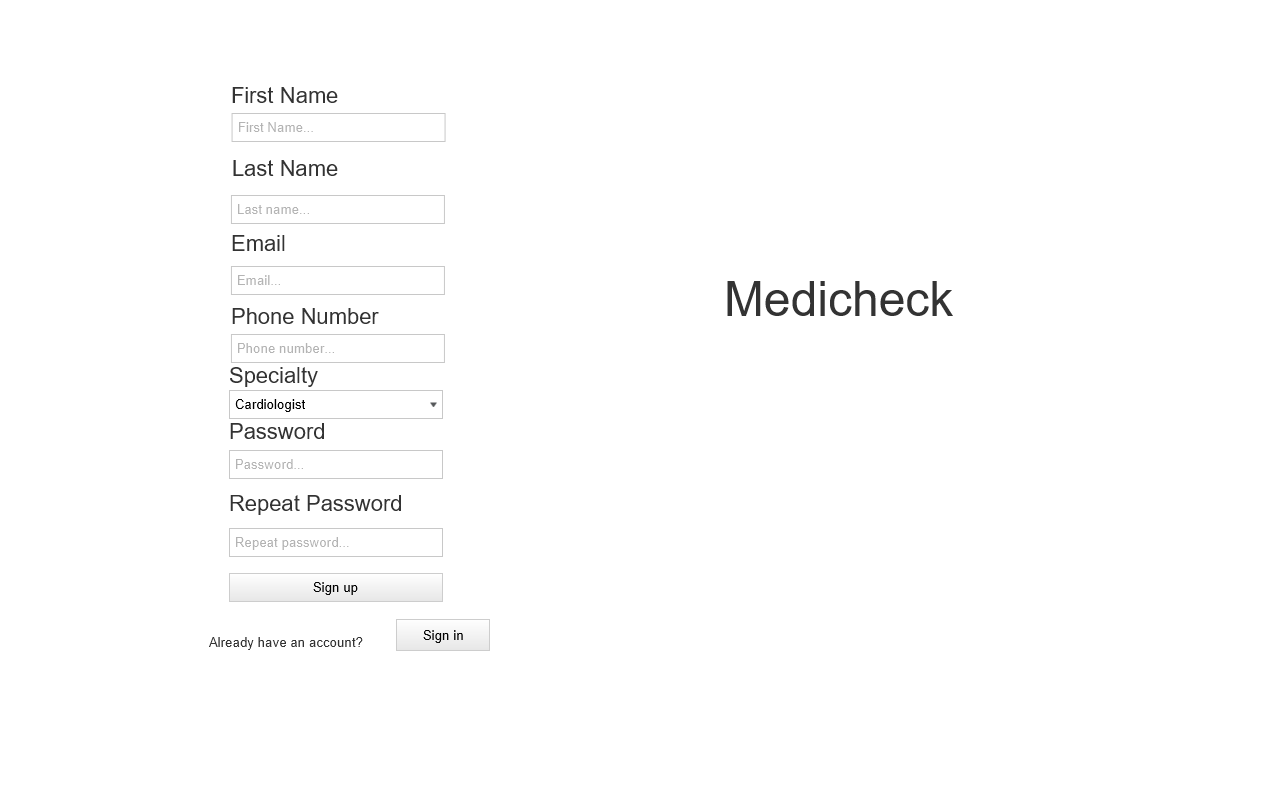
# Prototypes

### **Web Application**

Sign-in [Page] (Wireframe)



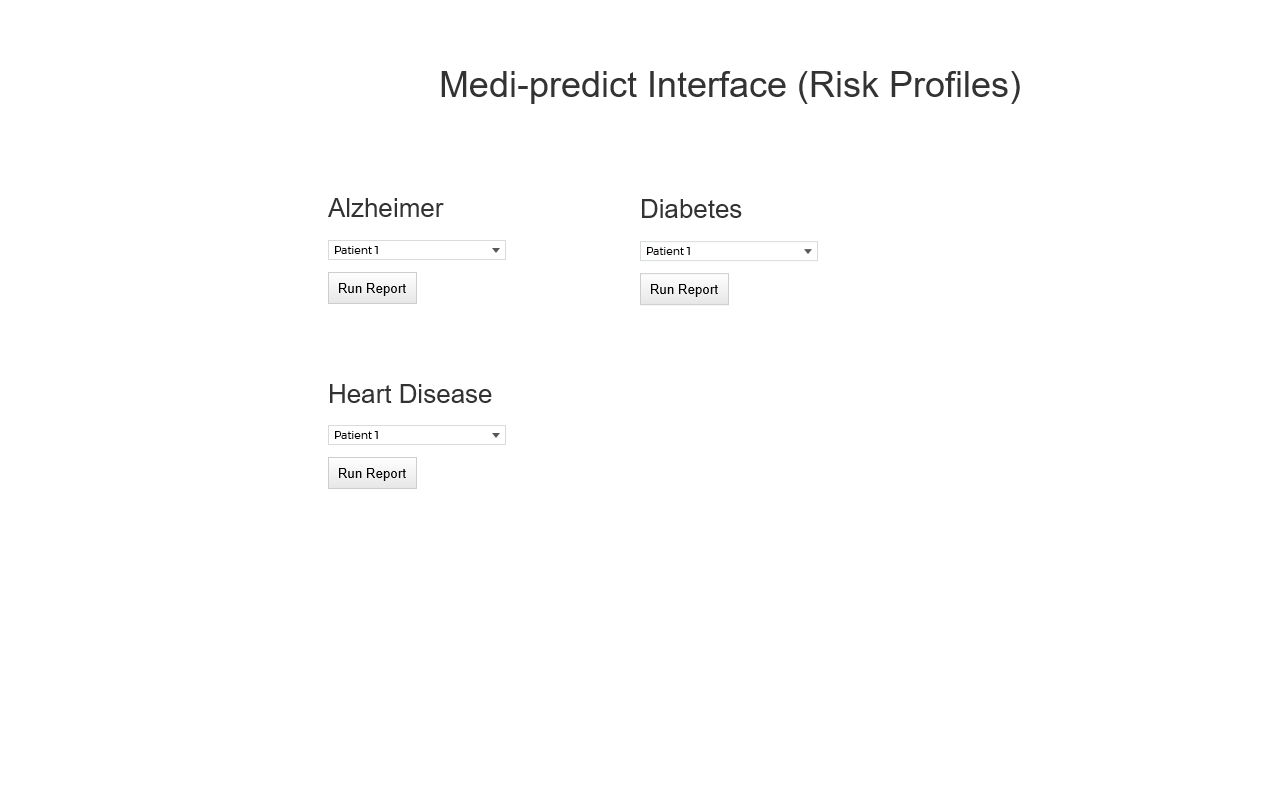
Sign-up [Page] (Wireframe)



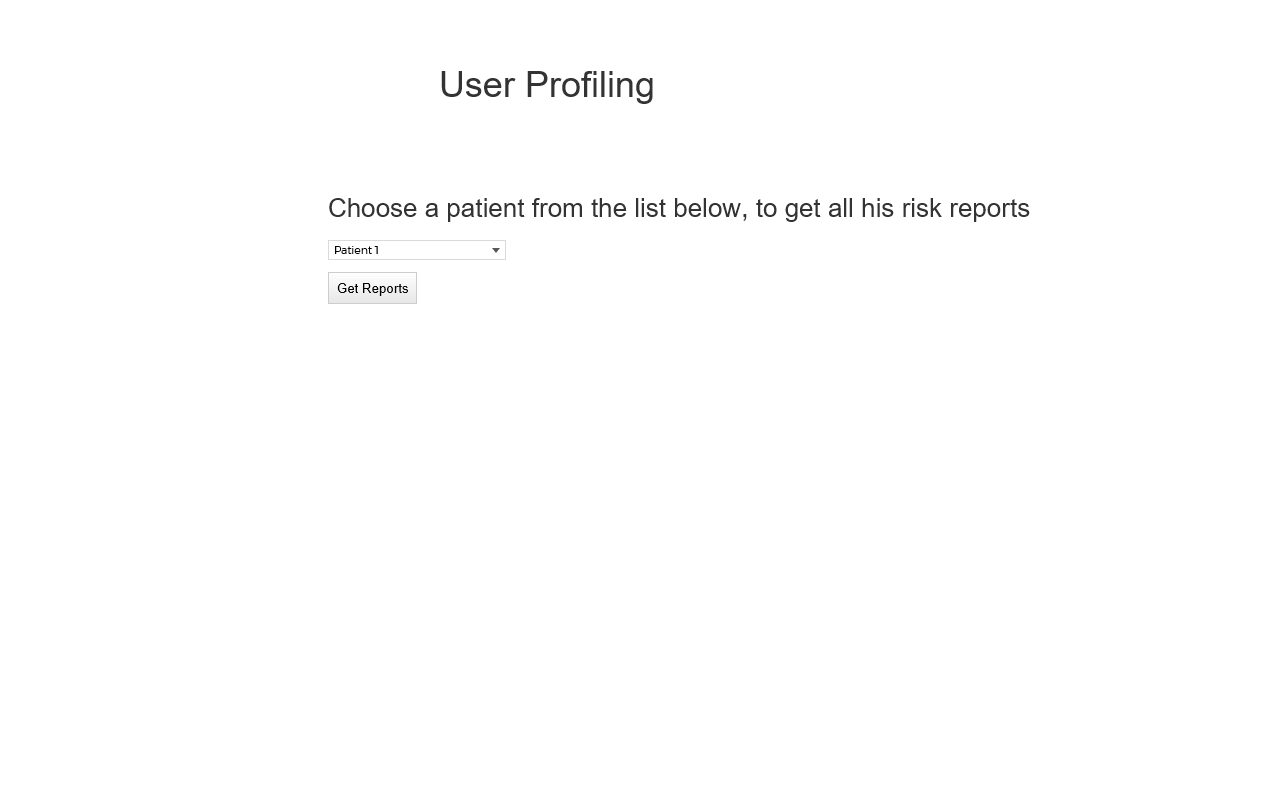
Your Patients [Page] (Wireframe)



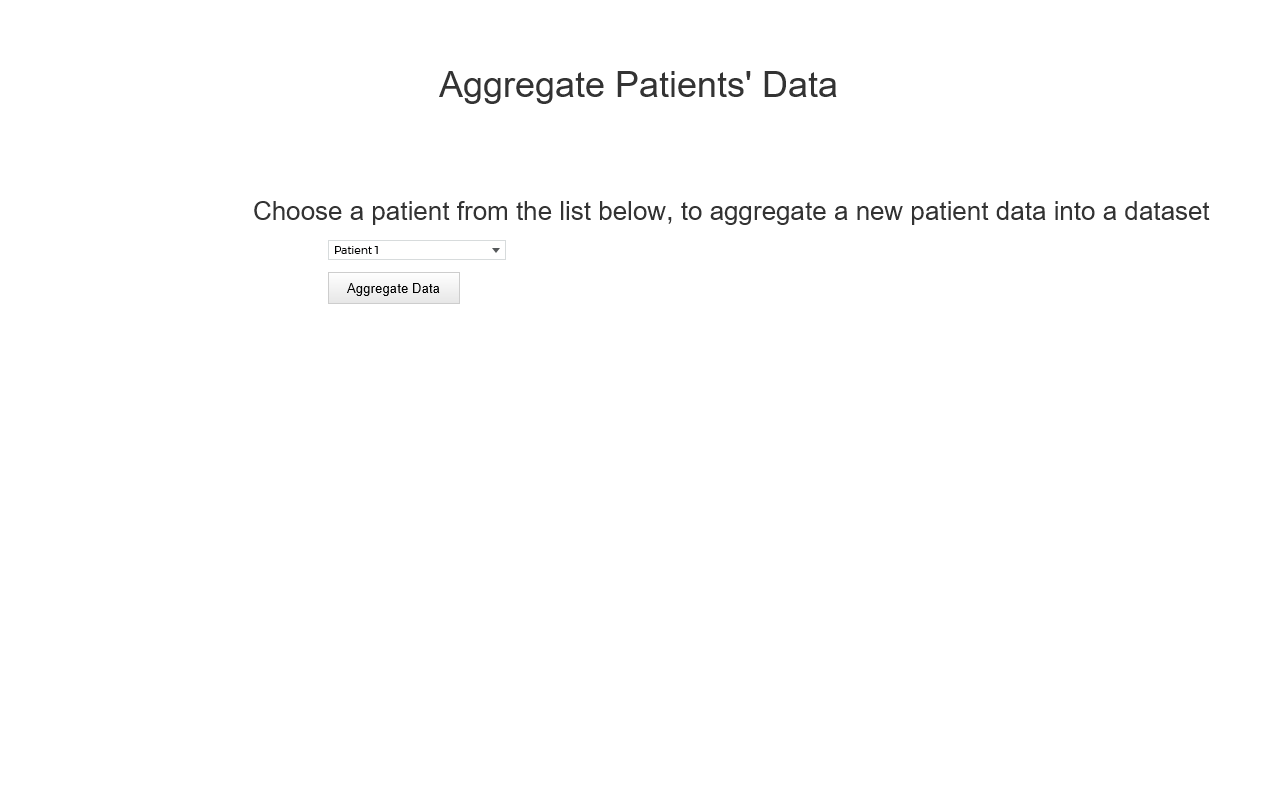
Risk Profiles [Page] (Wireframe)



User Profiling [Page] (Wireframe)



Aggregate Patient Data [Page] (Wireframe)



### **Android Application**

**Log in**

### 

### 

### 

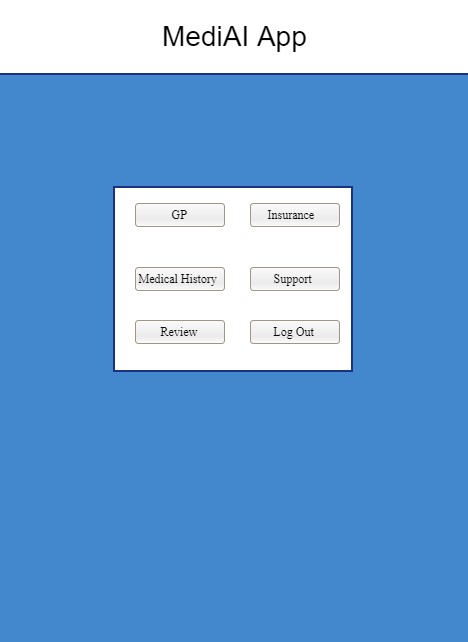
### 

### 

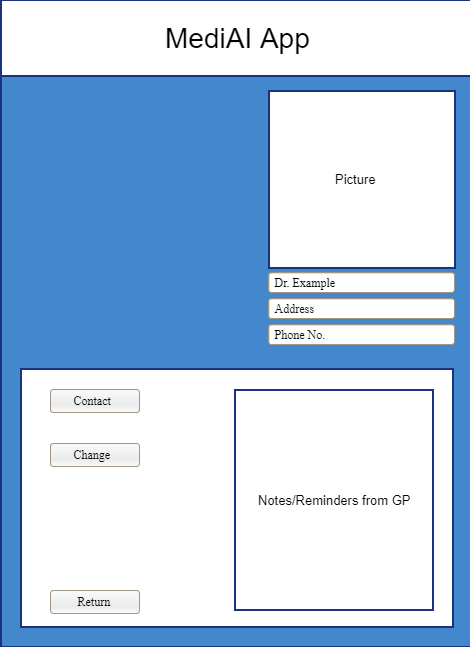
Register

### 

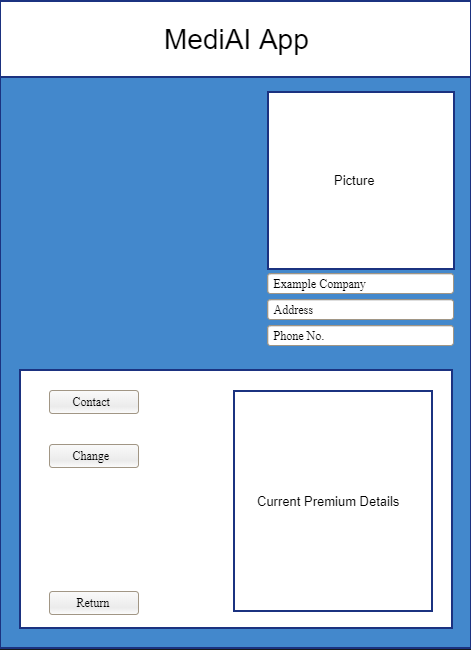
Home



GP



Insurance



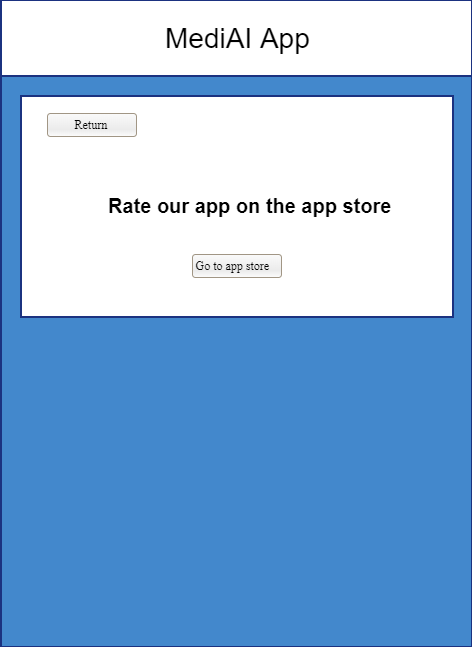
Medical History/Medi AI

### 

Support Form

### 

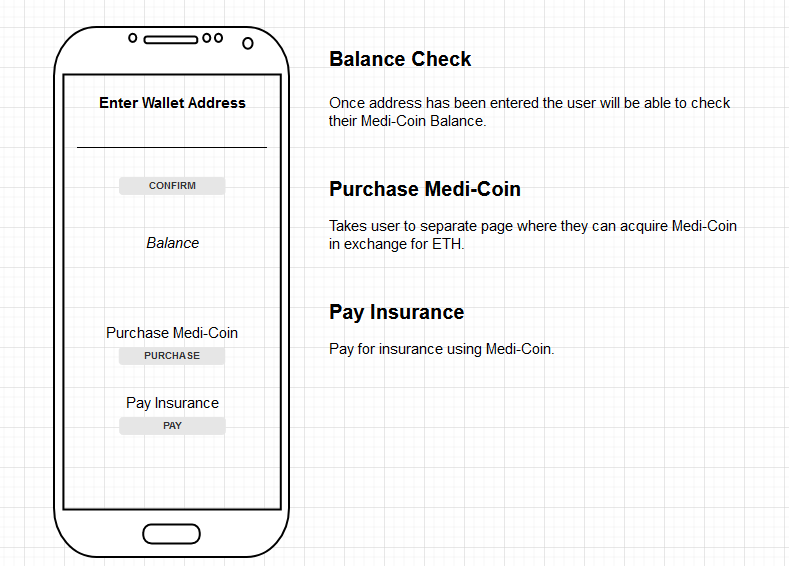
Review



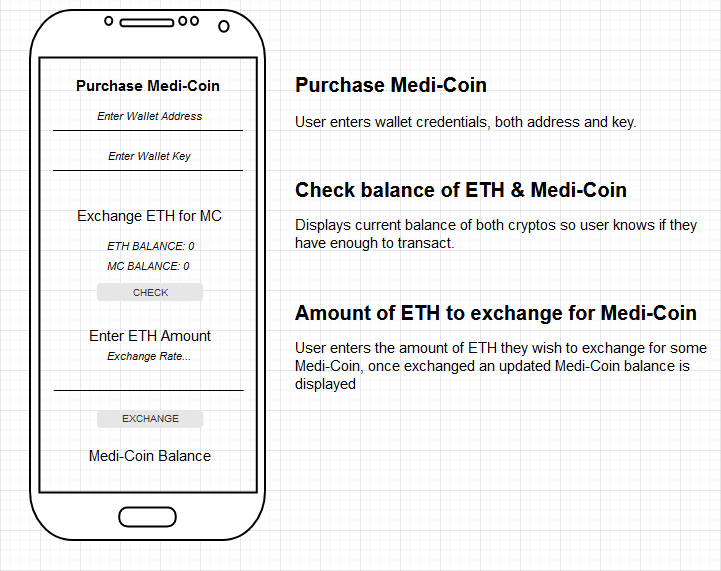
### 

### **Blockchain & Utility Token**

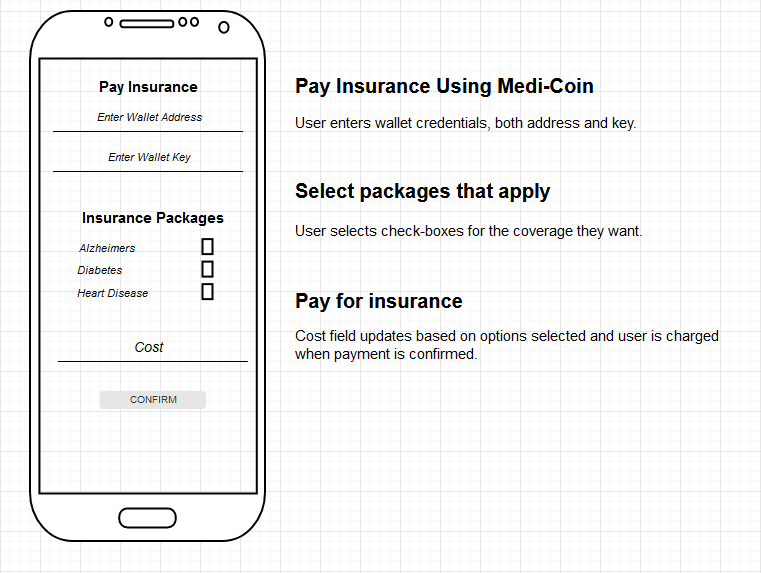
Utility Token Home Page



Purchase Medi-Coin



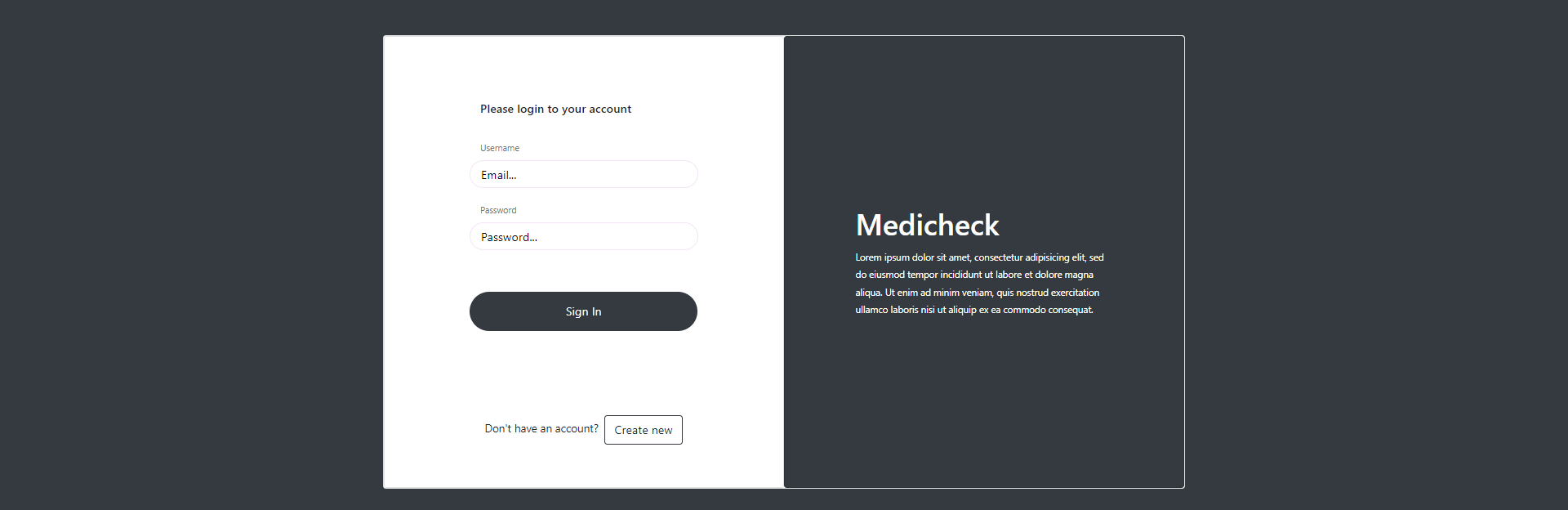
Pay Insurance



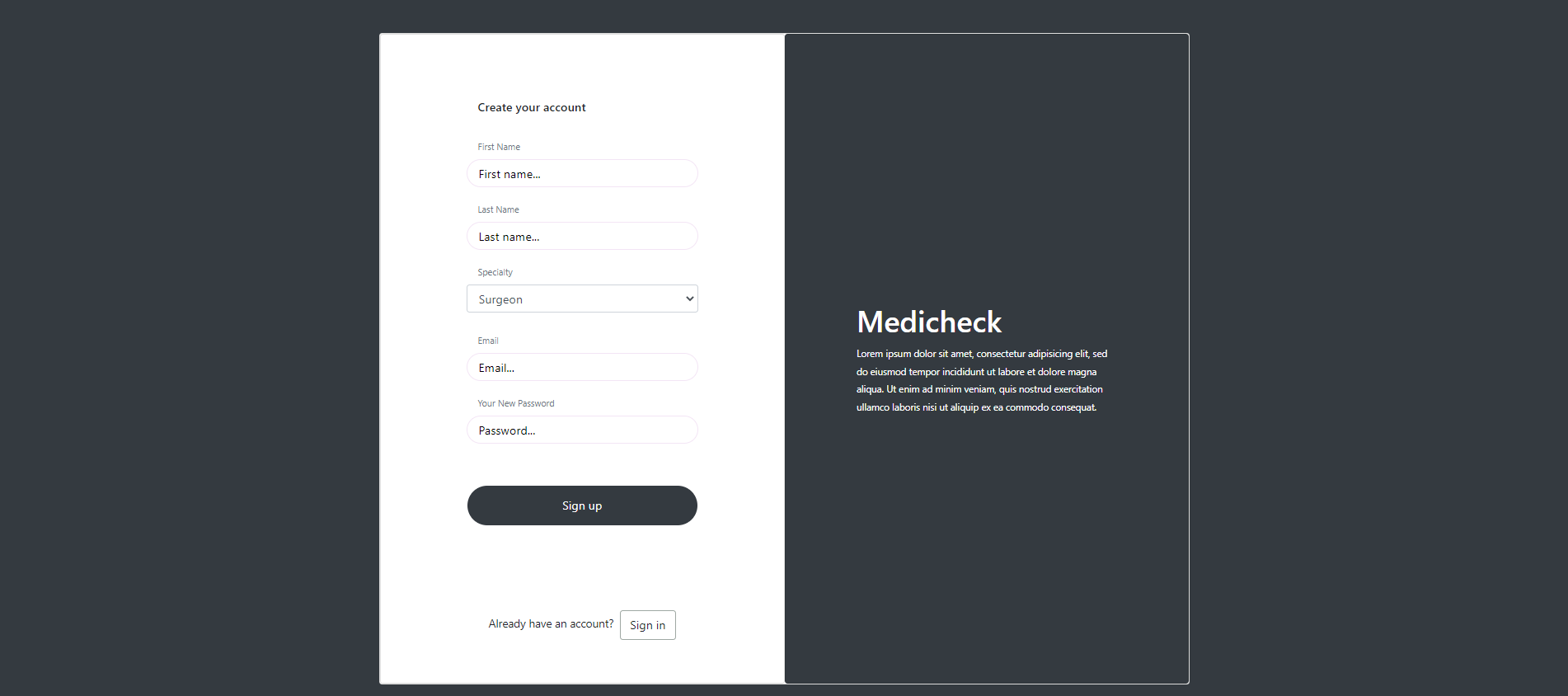
# Design and Implementation

### **Web Application**

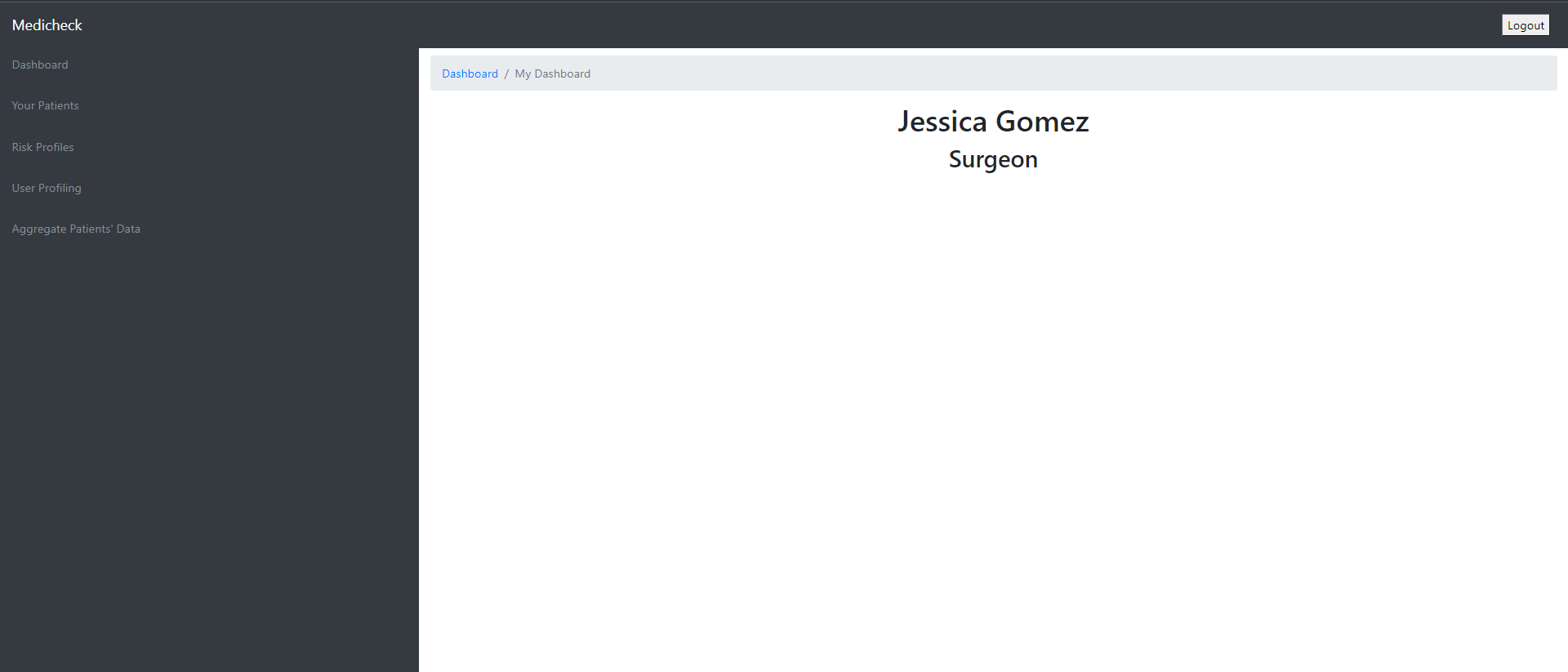
Sign-in [Page]



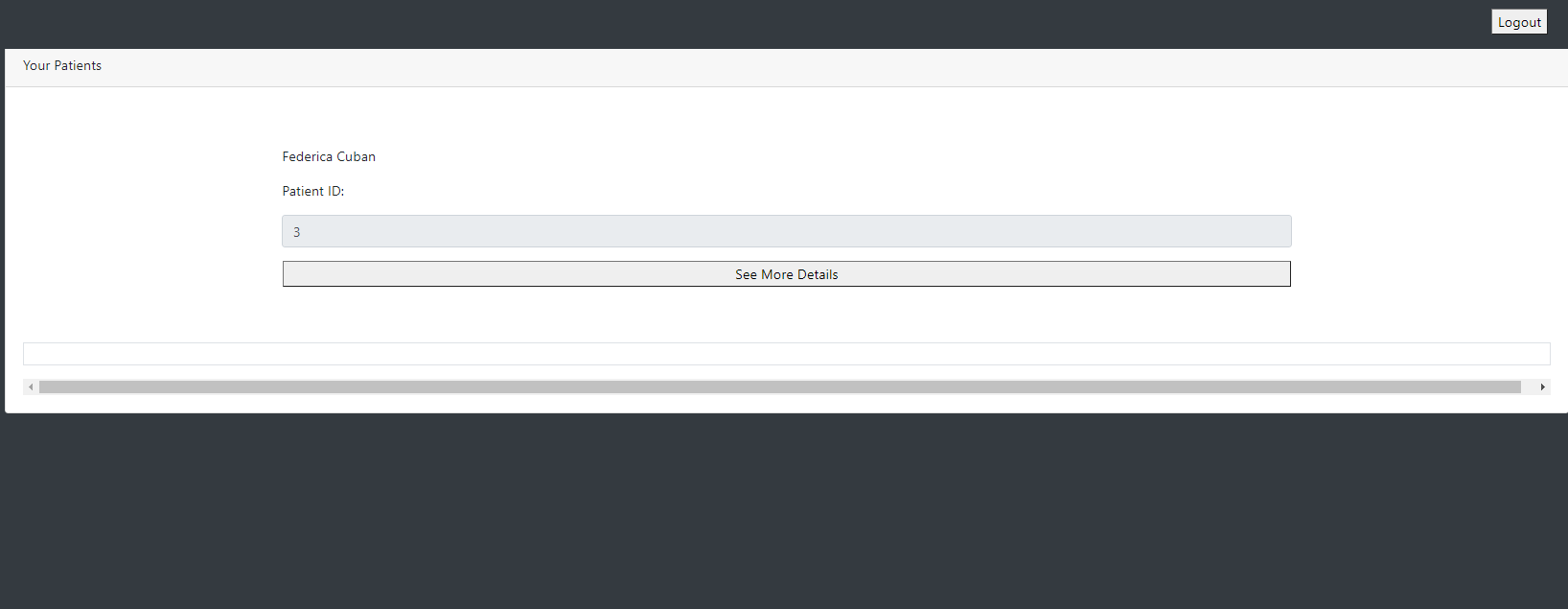
Sign-up [Page]



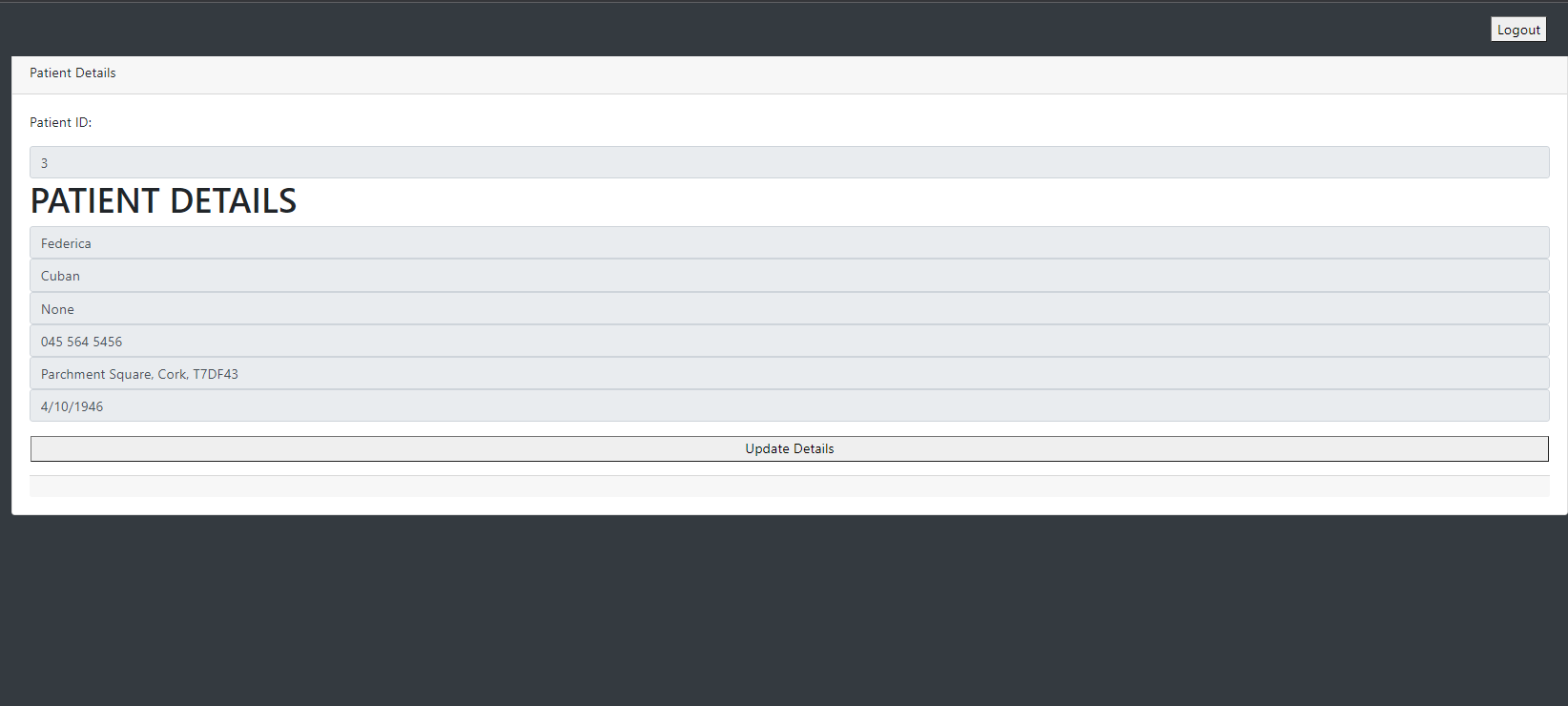
Dashboard [Page]



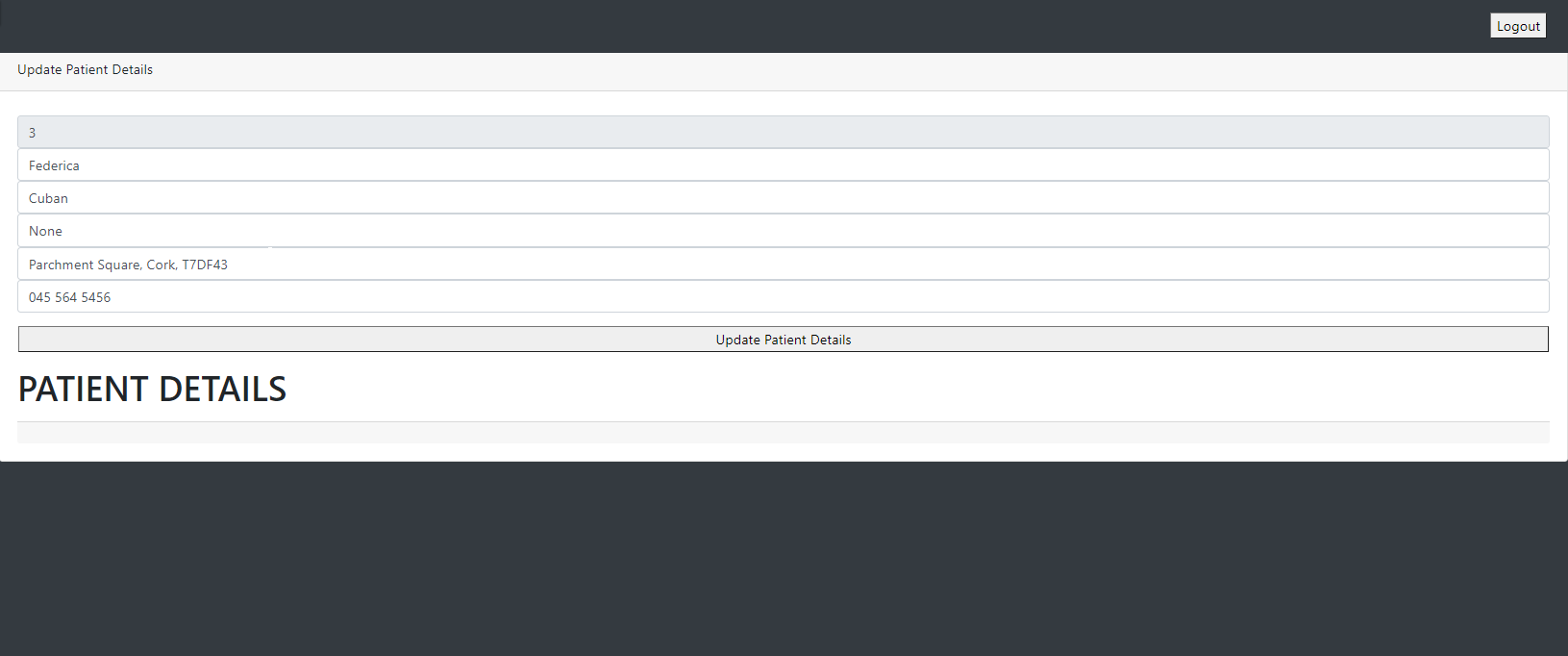
Your Patients [Page]



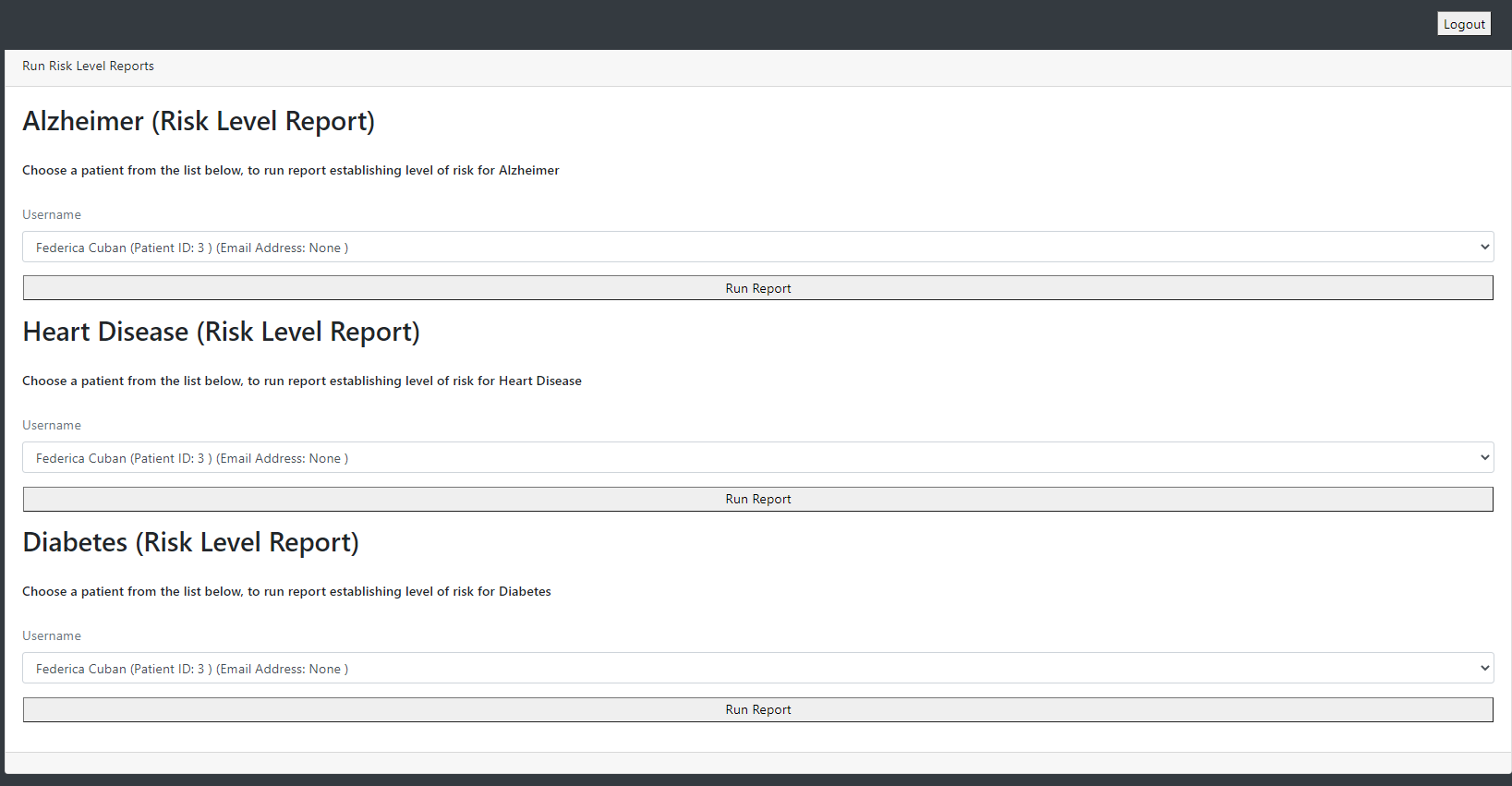
See Patient Details [Page]



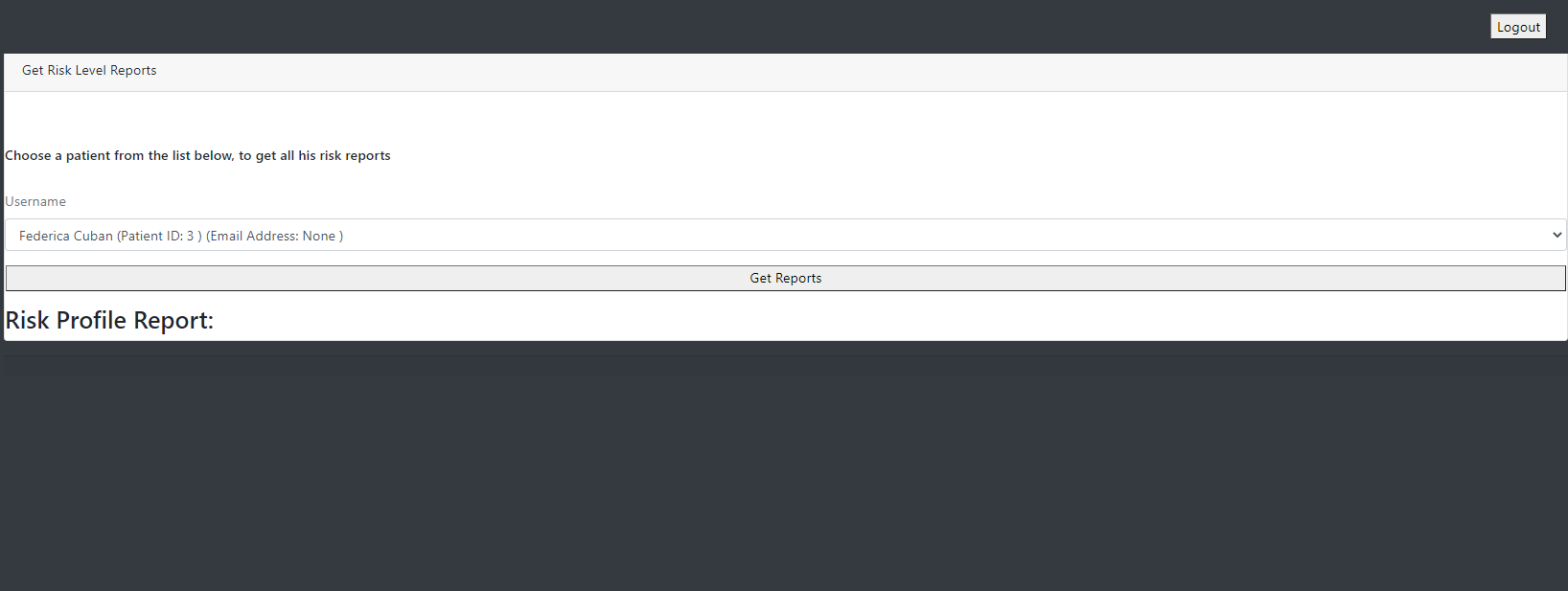
Update Patient Details [Page]



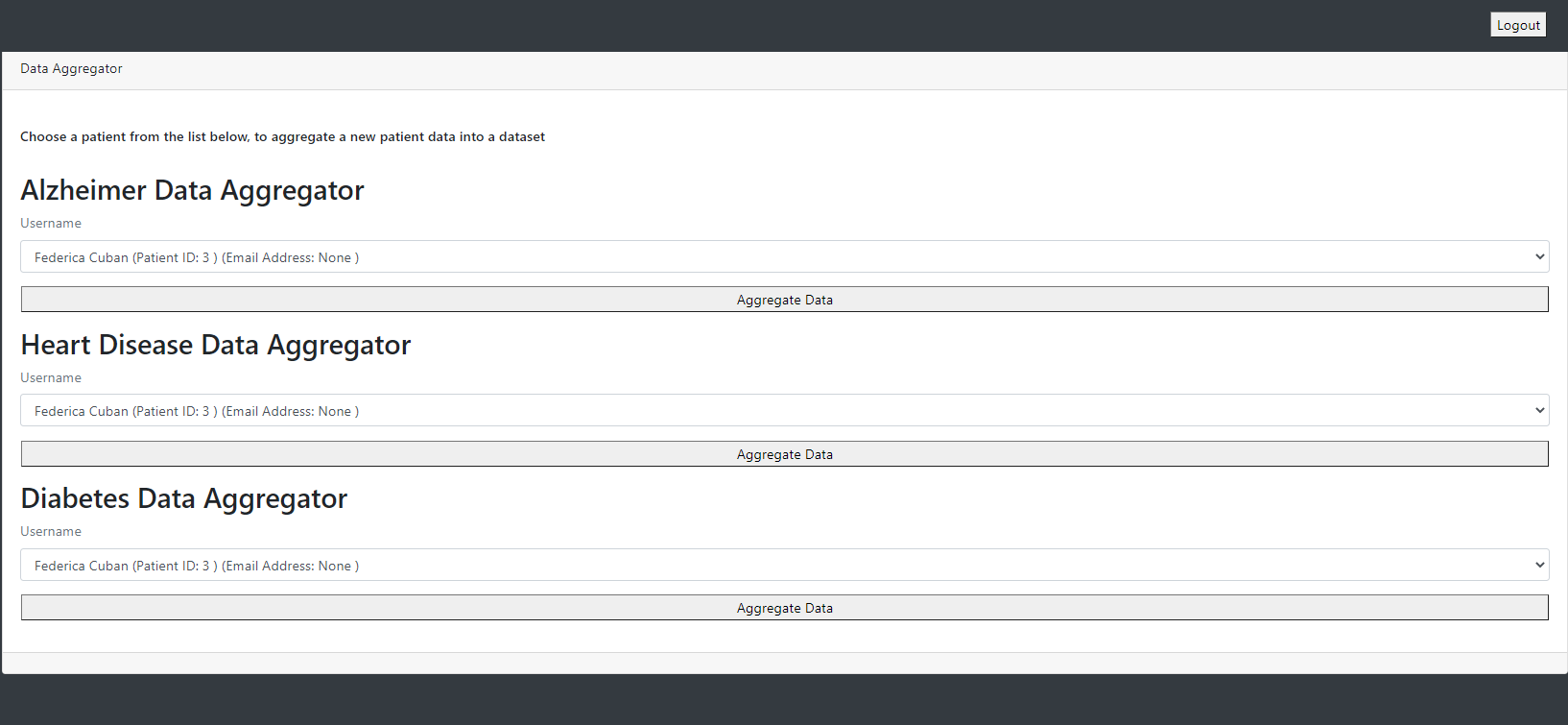
Risk Profiles [Page]



User Profiling [Page]



Aggregate Patient Data [Page]



# 

### **Android Application**

Log in

### 

### 

Register

### 

### 

### 

### 

### 

Home

### 

GP

### 

Insurance

### 

Medical History/MediAI

### 

### 

Support & Review

### 

### **Medi-AI**

#### Algorithms Used

Each Algorithm is a Supervised Learning Algorithm, meaning it uses labelled training data to learn.

**Random Forest Classifier**

For the Diabetes model I used RandomForestClassifier() 'RF' as the algorithm.

Steps to understand this algorithm can be broken into 4 steps.

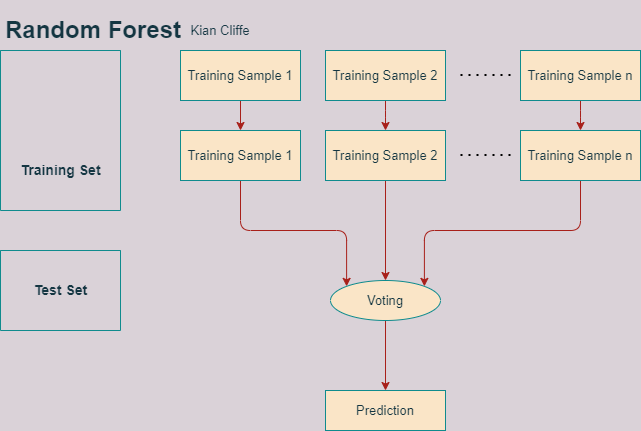
1. The idea comes from decision tree

2. Create a number of bootstraps i.e., a subset of data set

3. For each bootstrap create multiple decision trees by randomly selecting a number of attributes

4. Apply the new entry on all the trees and see which target value is returned more.

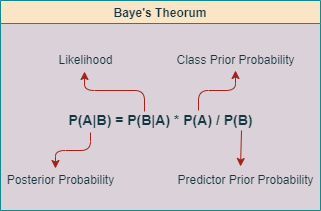
It uses a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

Random forest implementation in Sklearn has an argument called n\_estimators. This is the number of trees in the forest and the default value is 100. Higher n\_estimators potentially results in higher accuracy for the test set but has a higher running time. 

**GaussianNB**

For the Heart Disease model I used GaussianNB() 'NB' as the algorithm.

Sklearn implements several naïve bayes algorithms. Gaussian naïve bayes is one of them that can be used as a classification algorithm.



**Gradient Boosting Classifier**

Gradient boosting is a type of machine learning boosting. It relies on the intuition that the best possible next model, when combined with previous models, minimizes the overall prediction error. The key idea is to set the target outcomes for this next model in order to minimize the error. How are the targets calculated? The target outcome for each case in the data depends on how much changing that case's prediction impacts the overall prediction error:

If a small change in the prediction for a case causes a large drop in error, then the next target outcome of the case is a high value. Predictions from the new model that are close to its targets will reduce the error.

If a small change in the prediction for a case causes no change in error, then the next target outcome of the case is zero. Changing this prediction does not decrease the error.

The name gradient boosting arises because target outcomes for each case are set based on the gradient of the error with respect to the prediction. Each new model takes a step in the direction that minimizes prediction error.

#### 

#### Methodology

This section will be split into 3 parts, one part for each model (Heart Disease, Diabetes and Alzheimer’s)

Each model’s data is organized into an excel spreadsheet with patient specific information. The datasets are composed of both numerical and categorical data, both of which require treatment before being used in our models.

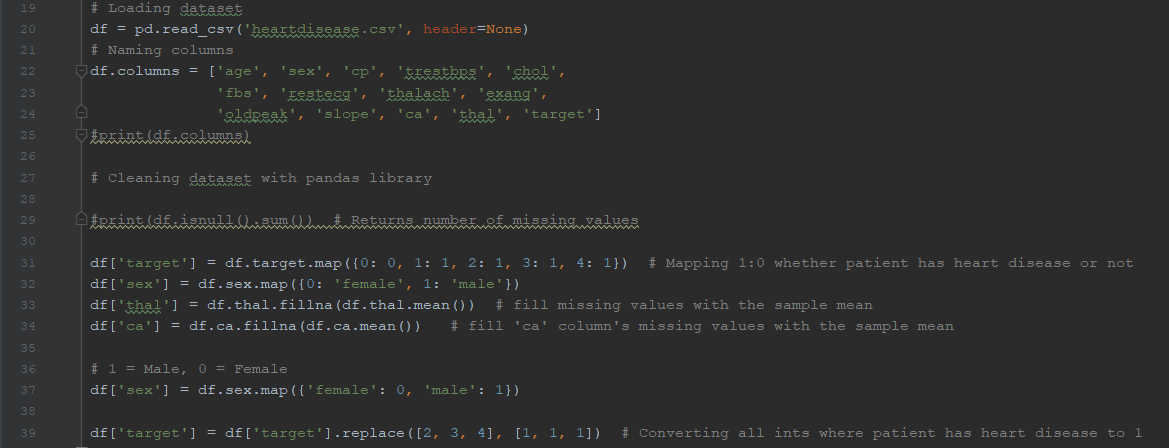
##### 1. **Heart Disease Model**

*Feature Engineering*

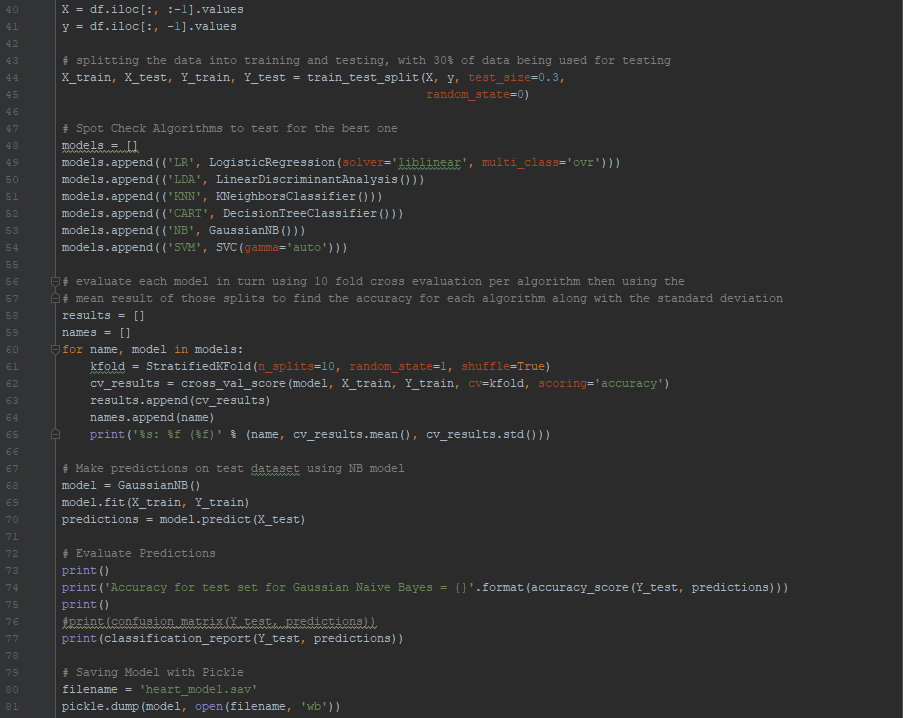
*Numerical Features*: Both the “ca” column and the “thal” column have empty columns so we fill these with the sample mean for each individual column.

The column we want for our ‘y’ variable in the train/test split has 5 values from 0 - 4 to show varying degrees of heart disease 0 meaning the patient doesn't have it while 4 being the most severe case. We don’t want this many values so I mapped the 5 individual values to 2. 0:1, ‘0’ stating the patient doesn’t have heart disease and ‘1’ stating they do.

*Categorical Features*: Both ‘male’ and ‘female’ are columns with text based values in this dataset which we dealt with by mapping 1:0 to ‘male’ and ‘female’ respectively.



Machine learning portion of Heart Disease Model

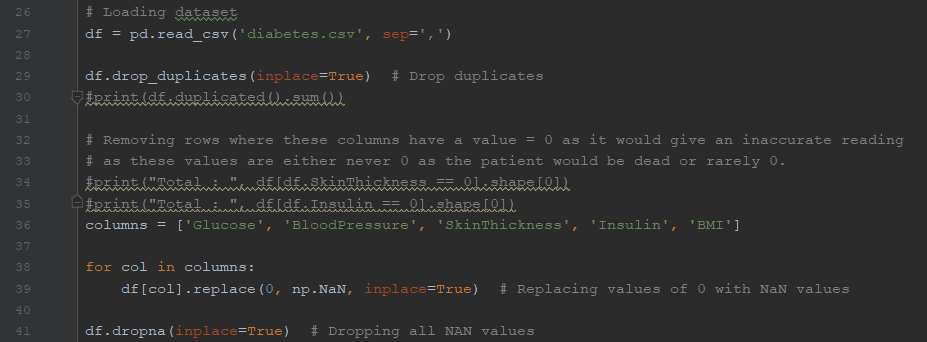


##### 2. **Diabetes Model**

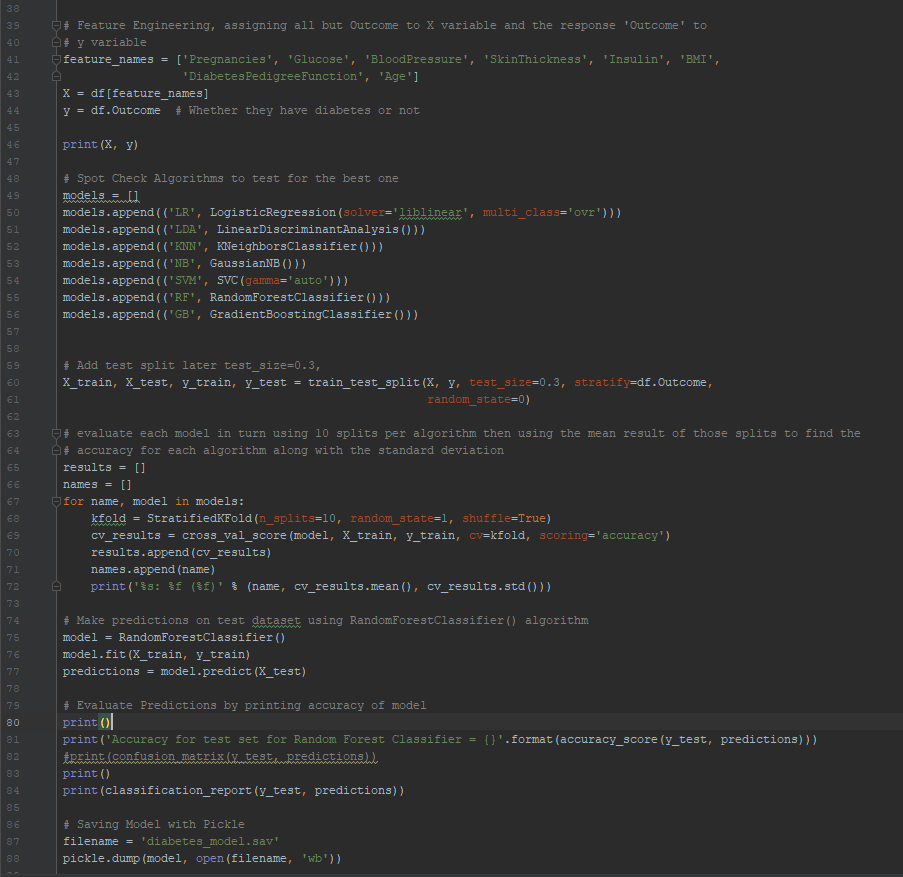
*Feature Engineering*

*Numerical Features*: We remove all rows where value is equal to ‘0’ in these columns 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin' and 'BMI' as it would give an inaccurate reading

as these values are either never 0 as the patient would either be dead, or these values would rarely be 0 due to human error filling out the spreadsheet.

*Categorical Features*: No categorical features.

Machine Learning portion of Diabetes model



##### 

##### 

##### 

##### 

##### 

##### 3. **Alzheimer’s Model**

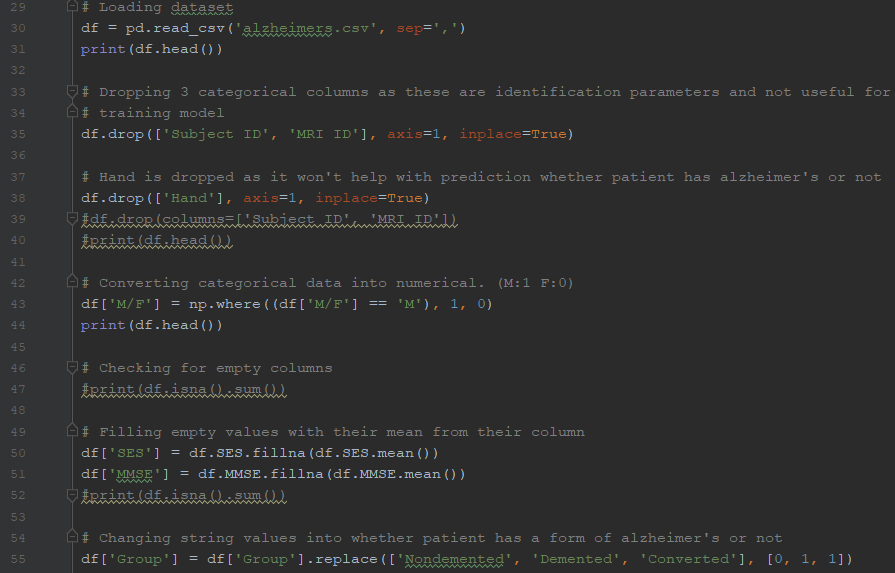
*Feature Engineering*

*Numerical Features*: We fill empty values with their mean from their column in both ‘SES’ and ‘MMSE’ columns.

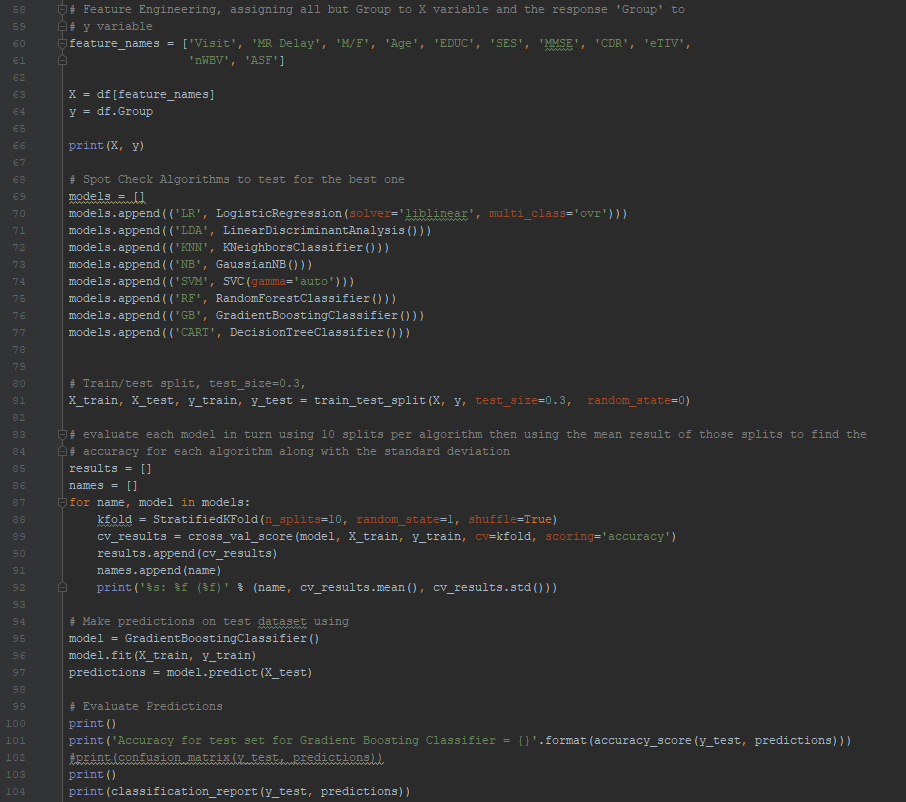
*Categorical Features*: We drop 3 categorical columns as these are identification parameters and not useful for training our model 'Subject ID', 'MRI ID' and ‘Hand’

We have one column that we want to use but is categorical ‘M/F’ so we convert this data into numerical. (M:1 F:0)

We use ‘Group’ column for our ‘y’ variable in our train/test split so we converted the 3 string values 'Nondemented', 'Demented', 'Converted' to [0, 1, 1] respectively.



Machine Learning code for Alzheimer’s dataset



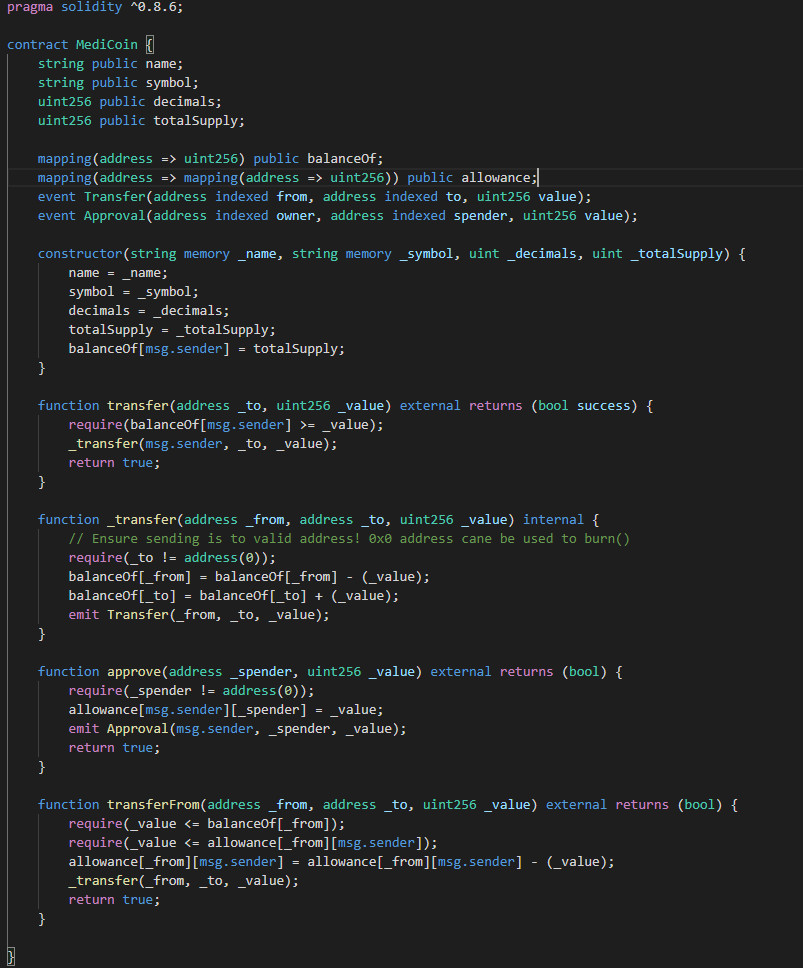
Each model was saved and loaded using the pickle library. Also use the model to predict result inputted as shown below.

### 

### 

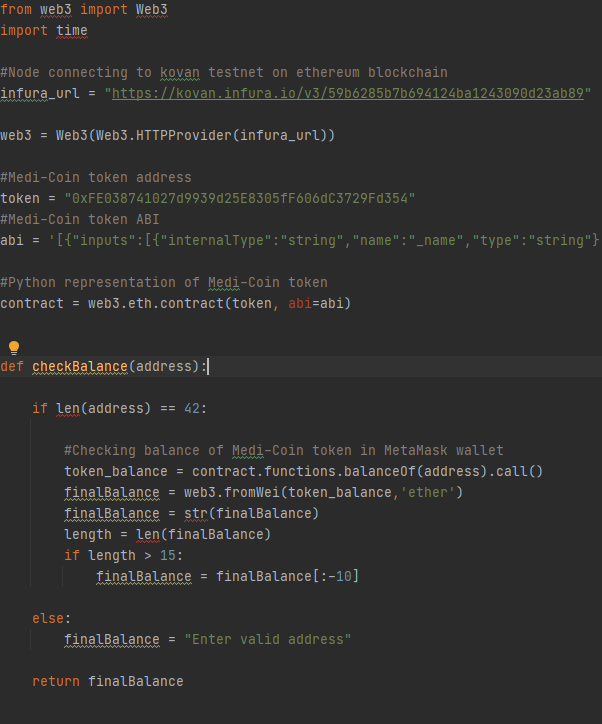
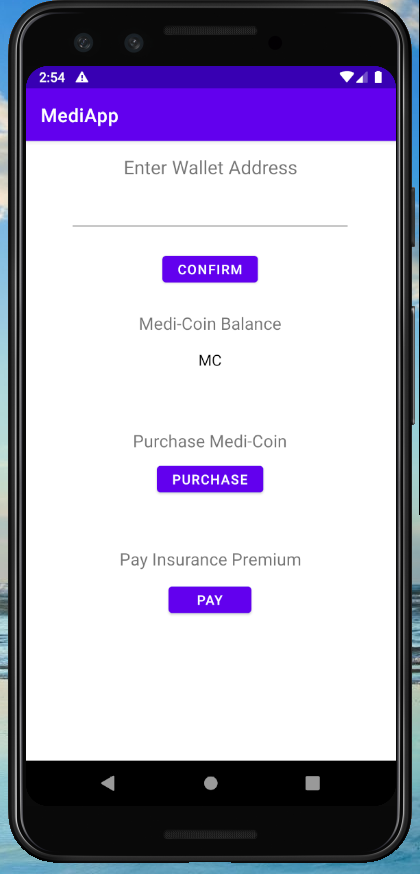
### **Blockchain & Utility Token**

Medi-Coin ERC20 contract https://kovan.etherscan.io/address/0xFE038741027d9939d25E8305fF606dC3729Fd354

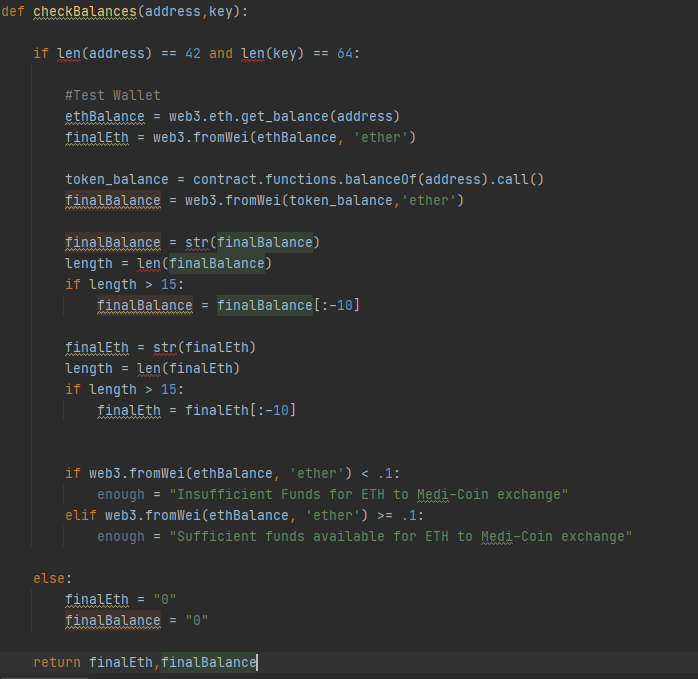
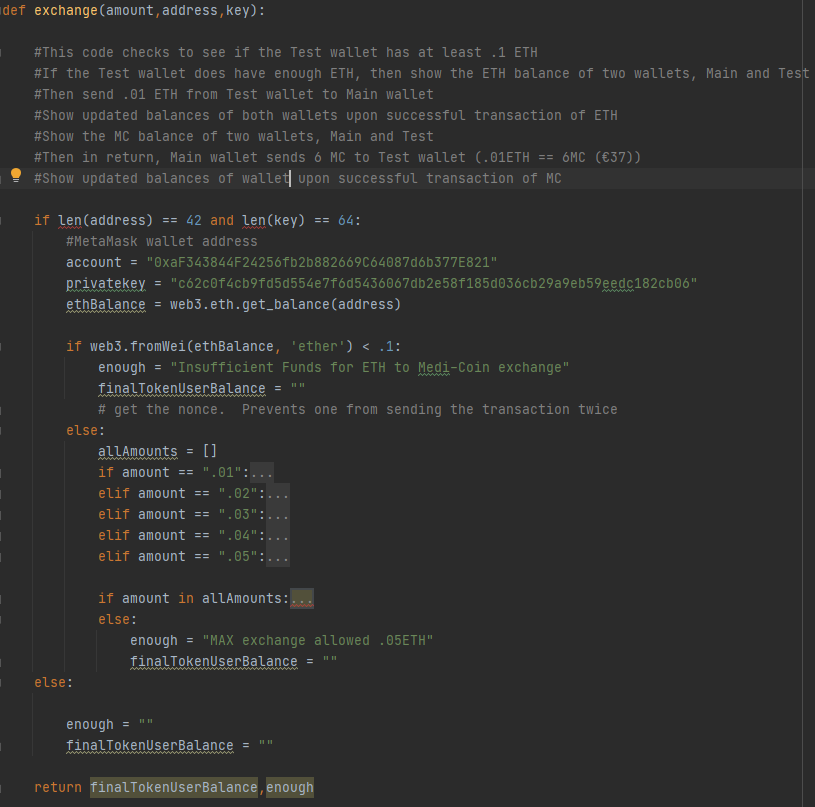
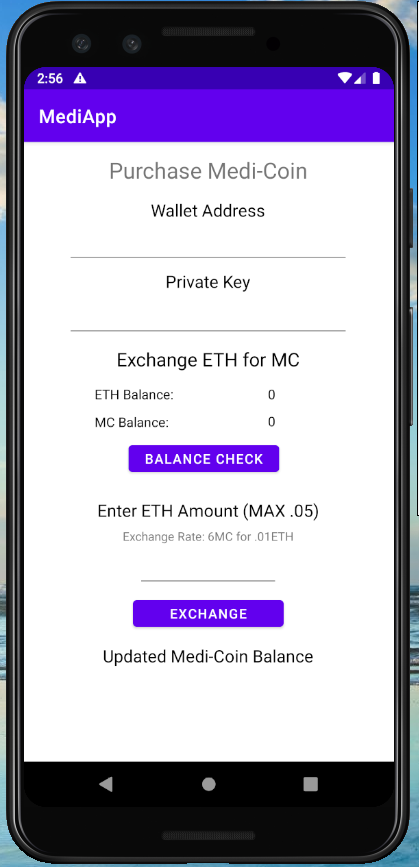


Pay Insurance with utility token home page

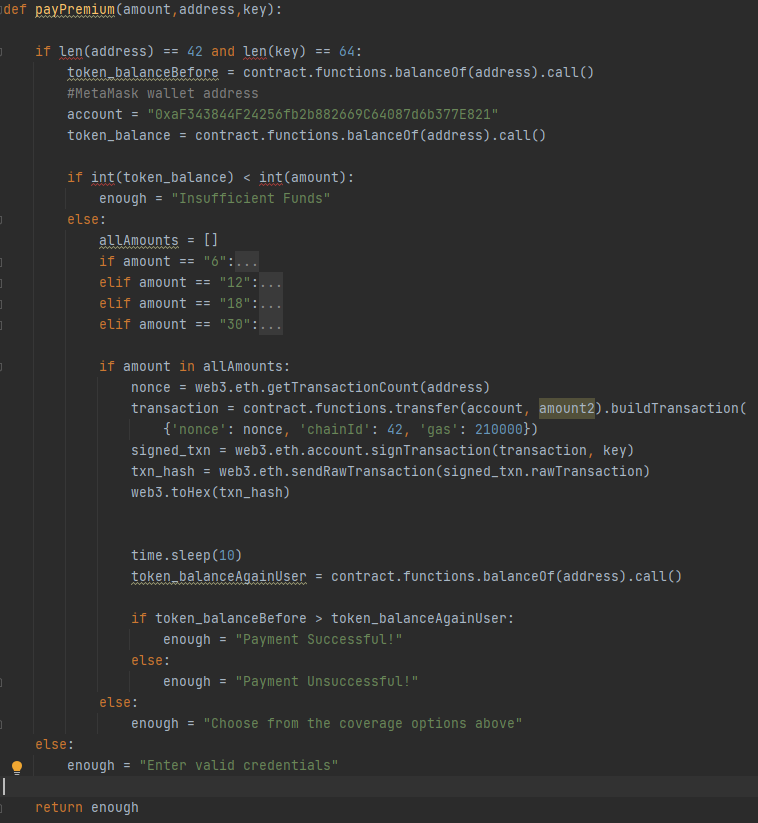
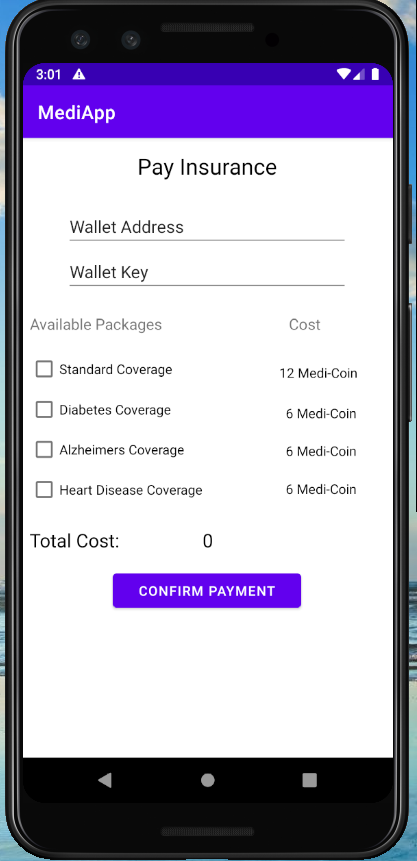




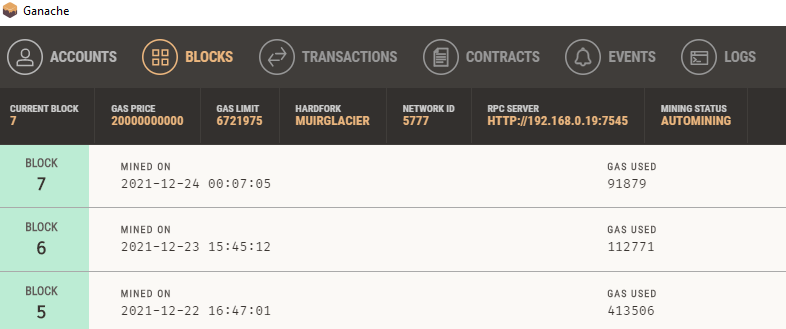
Crypto Exchange page



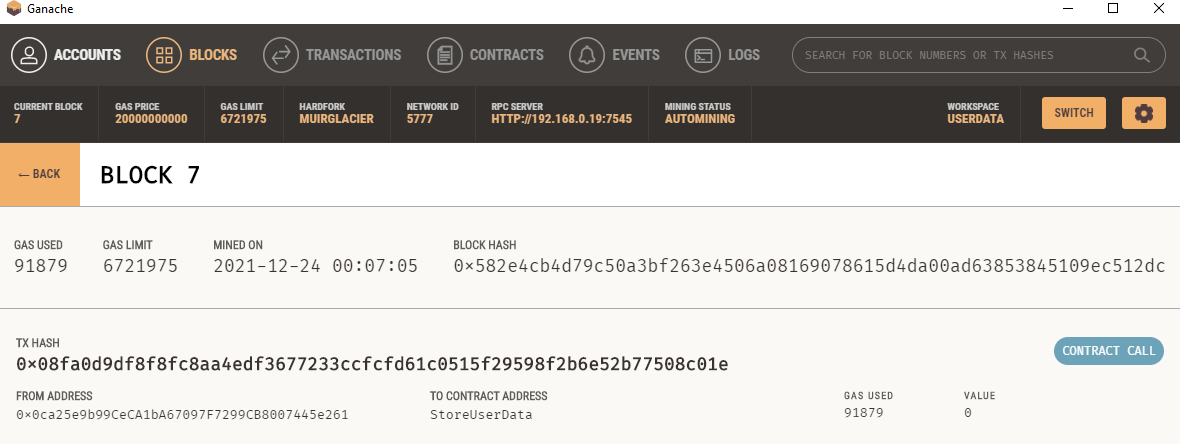
Pay Insurance page



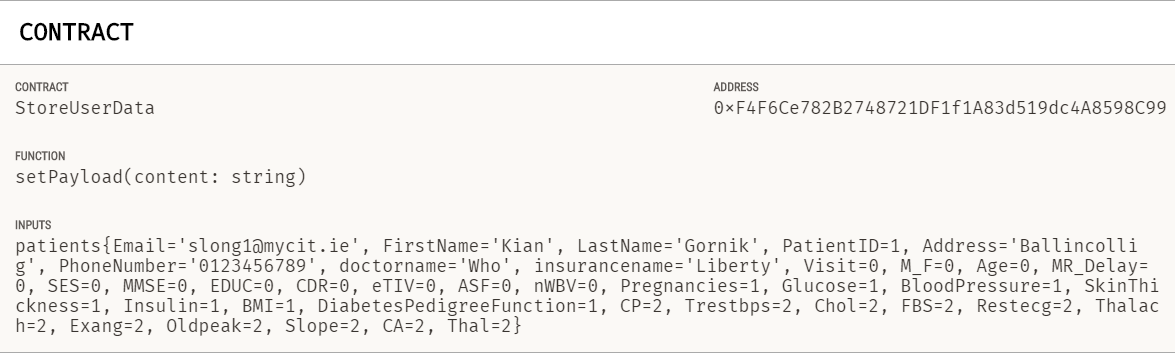
Ganache Blockchain



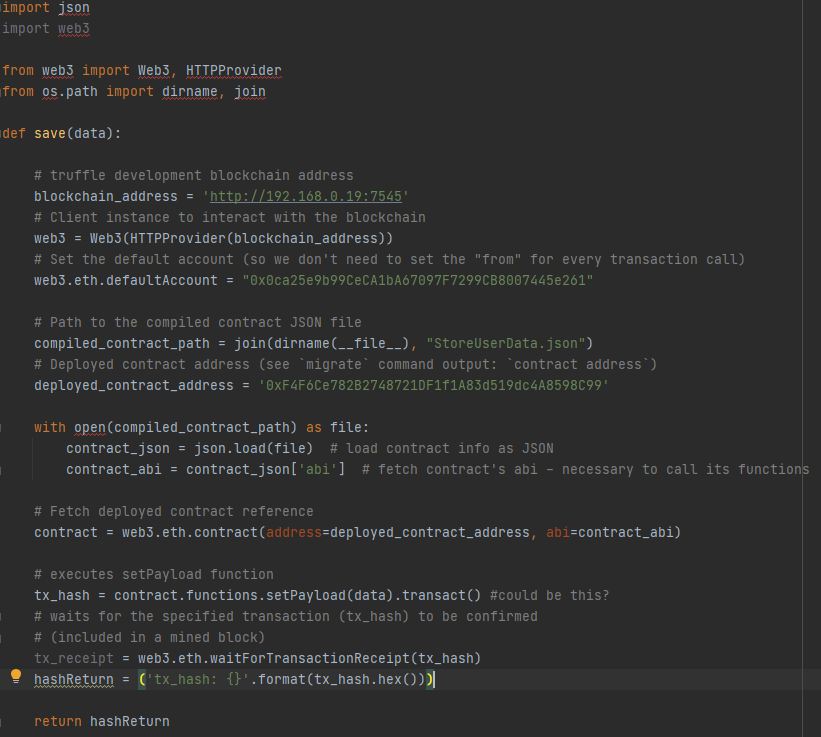
Block



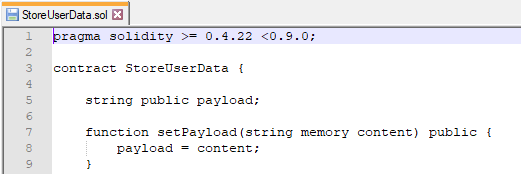
User data stored



Saving user data to block in blockchain, python

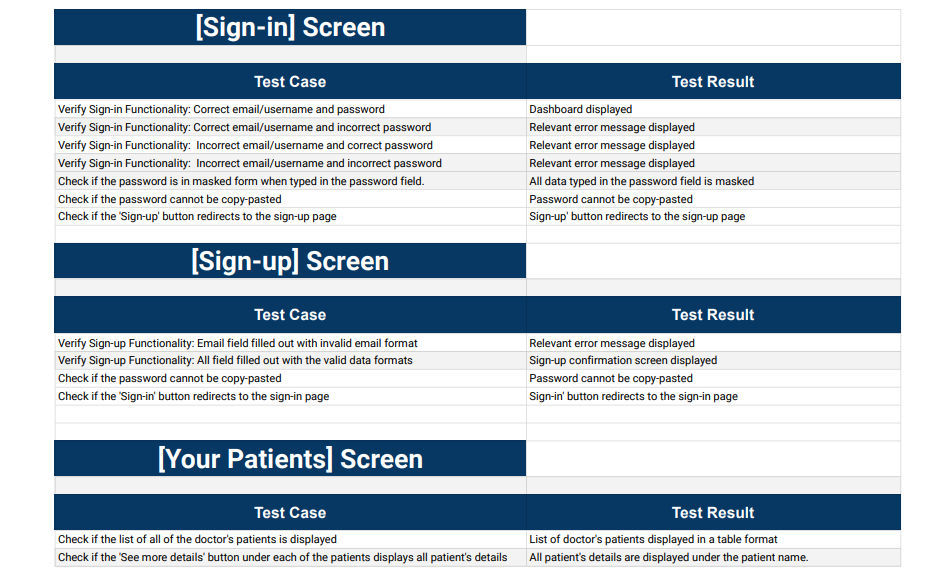


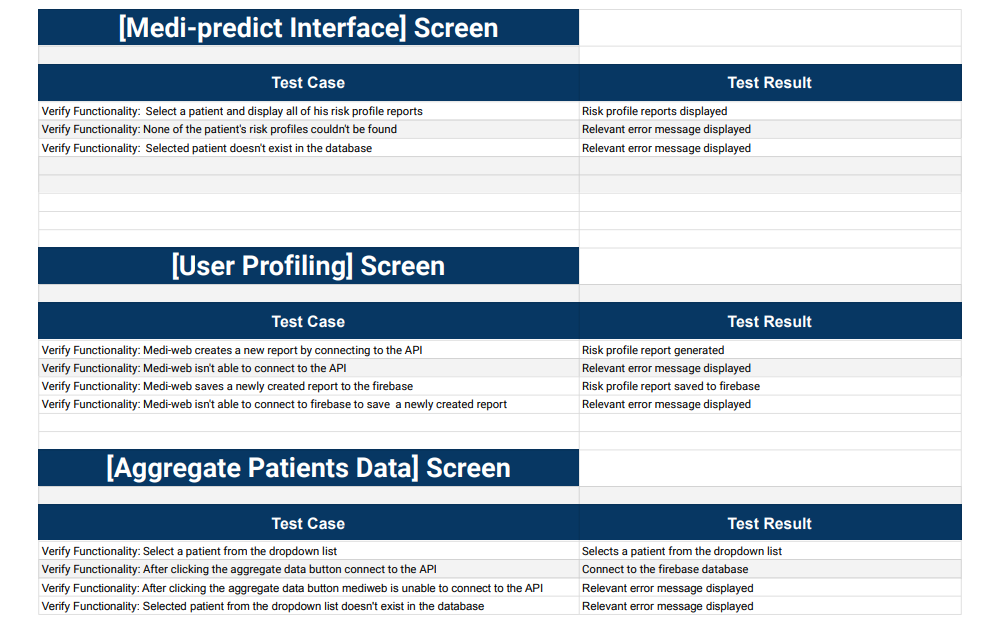
Solidity Contract



# Testing

### **Web Application**





# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

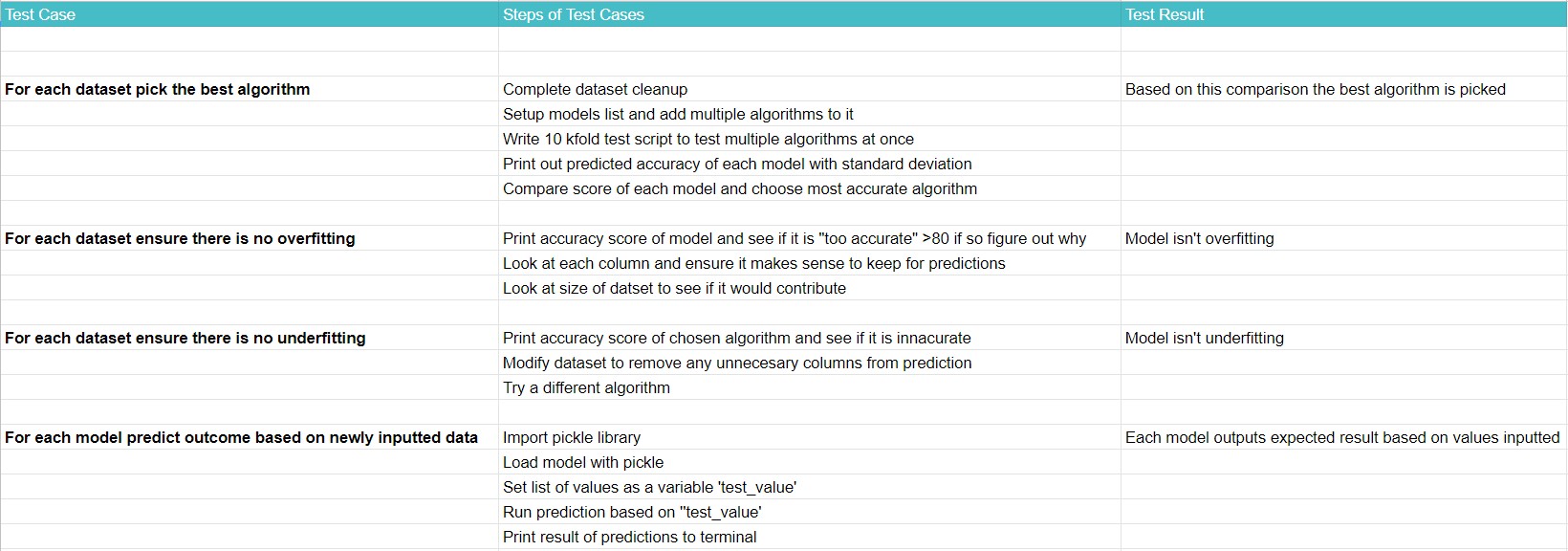
# 

# 

### **Android Application**

### 

### **Medi-AI**



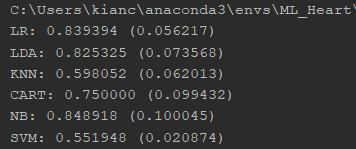
**Testing & Results**

For testing the accuracy we chose multiple models then evaluated each model in turn using 10 splits of the StratifiedKFold per algorithm then using the mean result of those cross validation splits, find the accuracy for each algorithm along with the standard deviation.

We then fit the most accurate model and evaluated the predictions by printing accuracy.

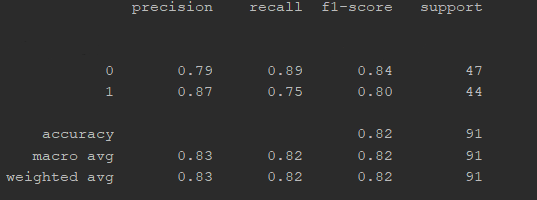
1. **Heart Disease Model**

Accuracy results for Models using heart disease dataset with (standard deviation)



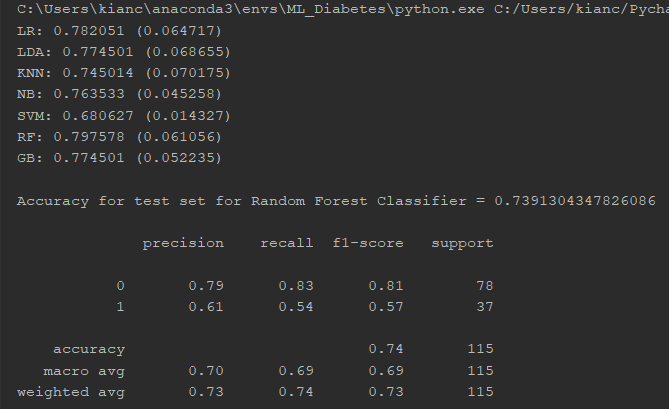
Fitting accuracy for Gaussin Naive Bayes (NB)





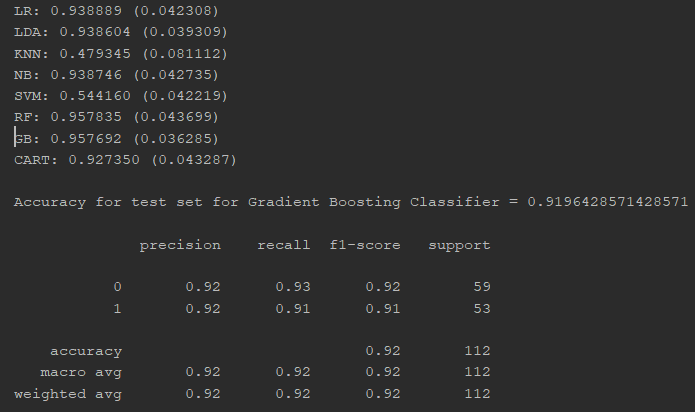
2. **Diabetes Model**

Accuracy results for Models using Diabetes dataset



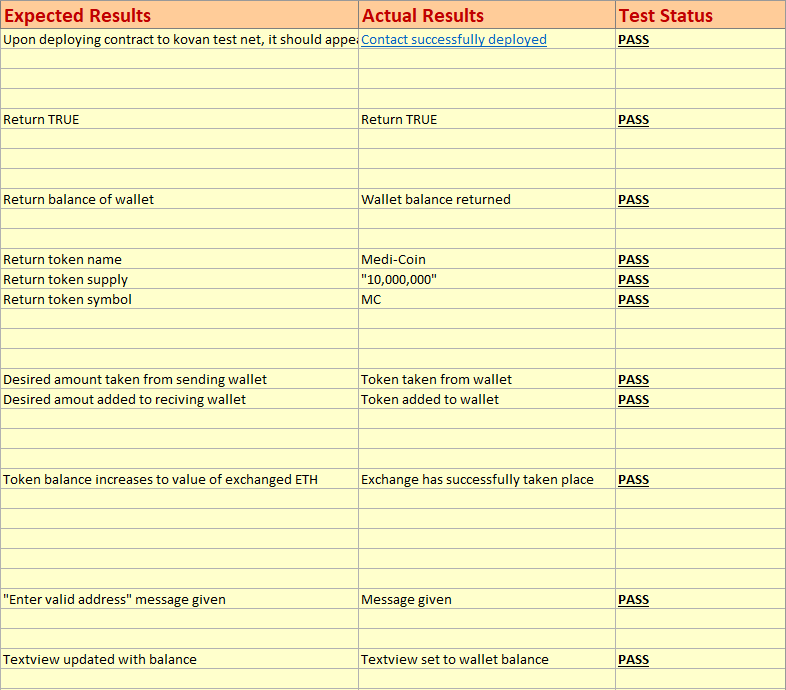
3. **Alzheimer’s Model**

Accuracy results for Models using Diabetes dataset

****

### **Blockchain & Utility Token**





# 

# 

# 

# 

# 

# 

# Conclusion and Future Work

### **Web Application**

Front-end and how the software looks affects the user experience. Better front-end increases conversion rates among the potential clients. That is why the future work plan includes improving the front-end design of the Mediweb and the android mobile app field. To improve the user experience and provide a visually appealing product.

Another aspect is machine learning. Part of the future work plan is to make Medi-AI available through an API instead of inserting the whole code package into the application. In this way, a machine learning team will be working on the ML part of the software independently, leading to future changes and implementations without integrating into the web app source code.

Lastly, the growing number of patients and medical professionals registered in our software will need more storage capabilities. To store all that new data and user details. In this context, we plan to switch from the Spark Plan to the Blaze Plan in the Firebase. To be able to store all the new data.

### **Android Application**

The ability for the patient to submit data to the FireBase FireStore collection is key to the functioning of the software. Without that data, the MediAI can only have sample data to work with, the patient cannot view their Medi-AI predictions or change their insurance or GP details and the medical professional on the website will have no data from the patient’s end on the app to work with.

Another important issue, but less detrimental to the functioning of the software as a whole, is that the functions to contact insurance, support, and GP cannot work without a functioning email client, and the email client in the emulator would not recognise the email it was given. It is possible that the functionality would work on a real android phone but we did not have time to test it.

In future, we will need to figure out what issue is causing the inability of the app to submit data to FireBase and run the app on a real phone instead of an emulator, as the contact forms will likely work on a real phone with a functioning email client.

### 

### **Medi-AI**

Future Work: The accuracy of each model can be improved upon in the future by more thorough testing with different algorithms and different amounts of epochs. We can also improve the accuracy of these models by changing certain parameters within each algorithmic function to see what effect that would have on the results.

More tests can be carried out to find out more about the underfitting and overfitting of these current models and whether the data or algorithm has to be altered in any way to accommodate for this.

### 

### **Blockchain & Utility Token**

Having successfully implemented both Blockchain and Utility Token functionality into the Medi project I can now reflect on what went well and what could be improved in the future.

It had been a relatively easy process writing the ERC-20 contract that represents the Medi-Coin token as well as deploying it to a test net, there was an abundance of tutorials that cover these steps. Where I ran into difficulty was with interacting with the smart contract and making use of the functions I had written. I learned of the Web3j library which would facilitate the calling and transacting with the ERC-20 contract. I made some progress using the Web3j library, through the help of online tutorials I was able to generate the necessary Java wrapper for the contract, but I was never able to call the necessary functions and get the desired results. This was quite a significant setback and the work I had done up to this point had taken up a decent amount of my overall project time.

It was at this point that I decided to approach the task using a different programming language, Python, I had the Web3py library at my disposal and a more easily digestible documentation to go along with it. I had a great deal more success interacting with the ERC-20 contract using Python than I had with Java and was able to make use of all the functions I had written for the contract, transferring Medi-Coin, checking wallet balance etc.

Having implemented the Utility Token functionality into the android application using the Chaquopy library, I moved onto the blockchain. My initial interpretation of the blockchain was a simple Block class that contained the users information which would be appended to an ArrayList representing the Chain, as to have the blocks persist I would save them to a database and would populate the Chain with any Blocks that existed in this database. I realised that this may have been too simple of a blockchain and decided to approach the task differently. I set up a local blockchain using ganache and wrote a solidity contract which would take a string containing the user information and save it as a block on the blockchain. I think had I opted to use a local instance of a blockchain from the beginning, I would have had more time to familiarise myself with the technology and I could have handled the user data better than just saving it as a string.

Looking back on my work for the project I can say I wasted a lot of time that could have been better spent elsewhere on the project. Had it not taken me so long to implement the Utility token feature into the Medi-App I could have spent more time fleshing out the blockchain as well as assisting other team members who were having difficulty. Having said all this I am happy with my work, I believe I have achieved everything that I had been tasked with and learned a lot about the technologies I used throughout.

## 

## 