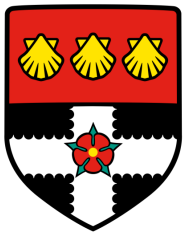


University of Reading

Department of Computer Science

An AI-assisted decision making system for thyroid nodule classification

Stefanos Stefanou

Supervisor: Huizhi Liang

A report submitted in partial fulfilment of the requirements of
the University of Reading for the degree of
Bachelor of Science in *Computer Science*

April, 2021

Declaration

I, Stefanos Stefanou, of the Department of Computer Science, University of Reading, confirm that all the sentences, figures, tables, equations, code snippets, artworks, and illustrations in this report are original and have not been taken from any other person's work, except where the works of others have been explicitly acknowledged, quoted, and referenced. I understand that if failing to do so will be considered a case of plagiarism. Plagiarism is a form of academic misconduct and will be penalised accordingly.

Stefanos Stefanou
April, 2021

Abstract

Deep learning has found numerous applications in the health care community. Recently, a massive explosion of research on the relevant field, driven by large amounts of available data, has generated important disease prevention and identification results. Fine Needle Aspiration (FNA) is the dominant procedure for thyroid nodule classification. FNA has associated risks and expenses, and in this project, we will try to reduce both using the recent advancements in Artificial Intelligence and Deep Learning. Our primary goal is to bring closer the radiologists 'on the field' with those complex algorithms and provide value to real patients by providing an interface, in the form of a web application, for probabilistically predicting the severity and the category of a given module.

Contents

Acknowledgments

Acknowledgments section is optional. You may like to acknowledge the support and help of your supervisor(s), friend(s), or any other person(s), department(s), institute(s).

Chapter 1

Introduction

Deep learning has found numerous applications in the health care community. Recently, a massive explosion of research on the relevant field, driven by large amounts of available data, has generated important disease prevention and identification results.[Eun Ju Ha (2021)]

The Dominant process for thyroid nodule identification and classification is called FNA (or Fine Needle Aspiration/Biopsy). FNA is an expensive process requiring expensive lab equipment and specialized personnel.[*Fine Needle Aspiration Biopsy of Thyroid Nodules* (2019)]

There is no way to determine the category of a thyroid nodule apart from performing FNA on a sample. Our vision is to create an application to act as a bridge between the academic community working on theoretical Deep Learning models to predict a nodule's category and the radiologists working with actual patients and accurate data. Our hope is that by establishing a common language(the application) we will improve the research process as experimental models will work on accurate nodule scans, providing instant feedback to the researchers for further analysis.

Our system needs to be as generic as possible to support any prediction model, reliable, easy to maintain, and expand. It needs to be optimized to handle the Deep Learning models and finally needs to be as secure as possible because it will eventually work with actual patients on sensitive data.

Abbreviations

FNA(Fine Needle Aspiration), AI(Artificial Intelligence), DP(Deep Learning)

Keywords

FNA, AI, DP

Chapter 2

Literature Review

2.1 Introduction

This section will note the essential sources needed to be studied and to be revised to complete this project. The sources are carefully selected to include theoretical, practical, and best practices knowledge in order to cover the wide variety of topics needed to fulfill the requirements of this project.

2.2 Brief Table of books

ISBN	Name	Type
N/A	ST1PS-18-9A: Probability and Statistics (2018/19)	Module Lectures
9780030105678	Linear Algebra and Its Applications	Book
9780131687288	Digital Image Processing	Book
9780262035613	Deep Learning	Book
9780128104088	Deep Learning for Medical Image Analysis	Book
9781491962244	Hands-on machine learning with scikit-learn and tensorflow	Book

2.3 Brief Table of papers

- Ye, H., Hang, J., Chen, X. et al. An intelligent platform for ultrasound diagnosis of thyroid nodules. Sci Rep 10, 13223 (2020). https://doi.org/10.1038/s41598-020-70159-y
- Nguyen DT, Pham TD, Batchuluun G, Yoon HS, Park KR. Artificial Intelligence-Based Thyroid Nodule Classification Using Information from Spatial and Frequency Domains. J Clin Med. 2019;8(11):1976. Published 2019 Nov 14. doi:10.3390/jcm8111976
- Manivannan T, Ayyappan N. Classification of thyroid nodules using ultrasound images. Bioinformation. 2020;16(2):145-148. Published 2020 Feb 29. doi:10.6026/97320630016145
- Nguyen DT, Kang JK, Pham TD, Batchuluun G, Park KR. Ultrasound Image-Based Diagnosis of Malignant Thyroid Nodule Using Artificial Intelligence. Sensors (Basel). 2020;20(7):1822. Published 2020 Mar 25. doi:10.3390/s20071822
- Chen J, You H, Li K. A review of thyroid gland segmentation and thyroid nodule segmentation methods for medical ultrasound images. Comput Methods Programs

Biomed. 2020 Mar;185:105329. doi: 10.1016/j.cmpb.2020.105329. Epub 2020 Jan 9. PMID: 31955006.

- Ha EJ, Baek JH. Applications of machine learning and deep learning to thyroid imaging: where do we stand? Ultrasonography. 2021 Jan;40(1):23-29. doi: 10.14366/usg.20068. Epub 2020 Jul 3. PMID: 32660203; PMCID: PMC7758100.

Chapter 3

Requirement Analysis

3.1 Introduction

Before we even start exploring this project and its features, it is essential to define the requirements that need to be fulfilled strictly and this project's scope. Failing to perform a requirement analysis beforehand puts additional and unnecessary risks to the project due to the project's unspecified and volatile scope and target set.

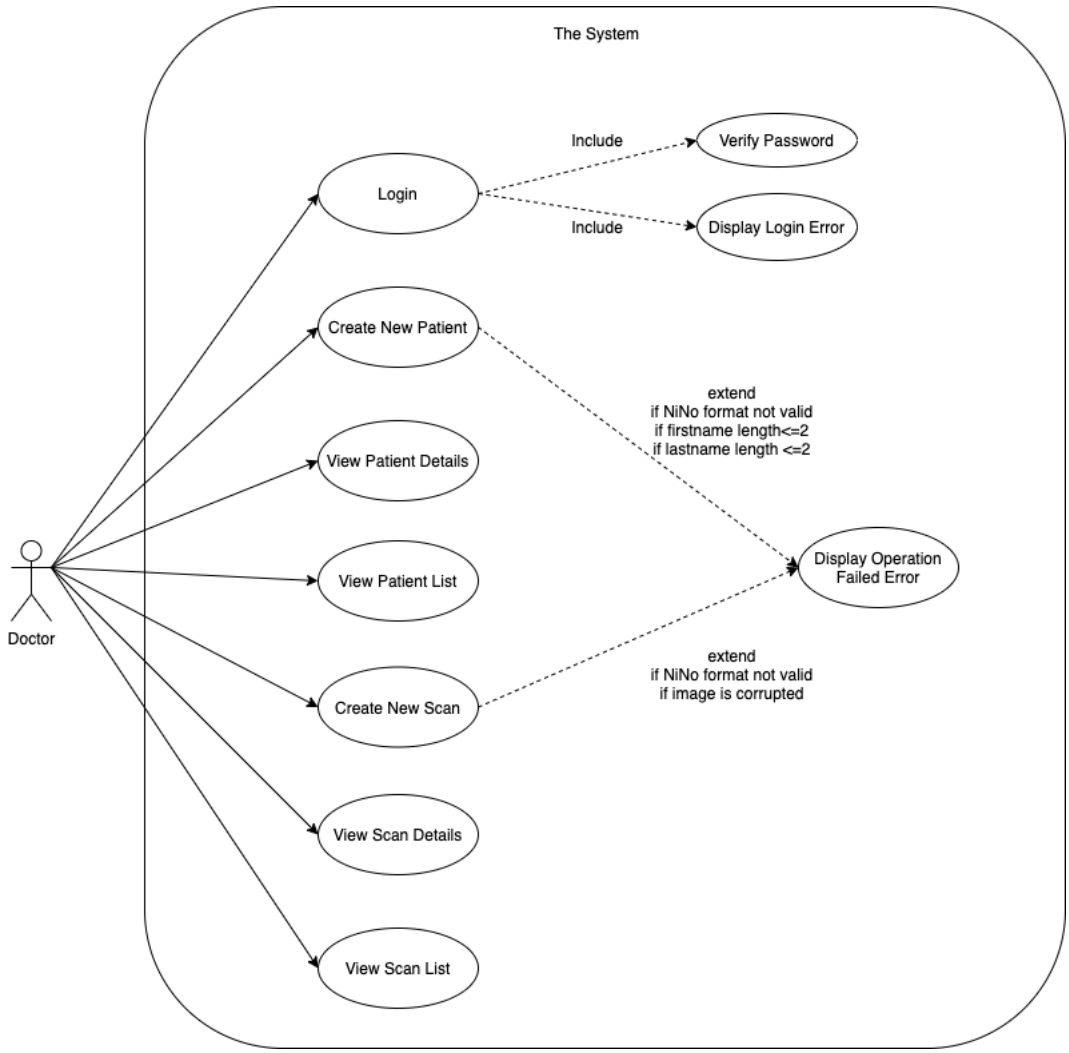
3.2 Functional Requirements

Functional requirements define the basic system behavior. We define the functional requirements as follows.

- User needs to log in with a personal password.
- User needs to be able to create a new patient record
- User needs to be able to upload a new ultrasound scan image associated with a given patient
- User needs to be able to see its associated patients
- User needs to be able to see its uploaded ultrasound images
- User needs to be able to search for a specific patient
- User needs to be able to see a list of all ultrasound images for a specific patient
- User needs to be able to see the details of a specific patient
- User needs to be able to see the details of a specific submitted scan, as well as the prediction results if available.
- User should be notified if the prediction results are ready

Those requirements can be easily visualized in a Use Case Diagram, given below.

Figure 3.1: Use Case Diagram



3.3 Non Functional Requirements

Nonfunctional requirements are the properties of the system; an comprehensive list of the agreed nonfuctional requirements is given below

- The system must be secure, as it handles the personal information of the patients
- The system should be reliable, as downtimes are affecting the hospital's performance
- The system should be able to complete a prediction scan in a reasonable amount of time(1-10 mins)

Chapter 4

Entity Relation Analysis

4.1 Introduction

After the requirements have been set. We need to translate them into workable relational entities in order to be able to modeled through a classical relational database system (RDBMS).

4.2 Entities

We will start our exploration by defining our entities for this project.

4.2.1 Scan

A scan is the result of an ultrasound scan performed in a specific patient(see [4.2.2)]. A scan entity has certain attributes

Image	The image produced by the ultrasound scan. 360x560 pixels
Prediction	The result of the prediction algorithm. Acceptable Values = Maligrant, Benign
Results	The logs of the algorithm performed the prediction, Optional
Algorithm	The algorithm used to perform the prediction. Acceptable Values = SVC,RES
Token	The scan identifier across the application services. token type is UUID [Leach (2005)]

4.2.2 Patient

A patient is a physical person that is suspected to have a thuroid nodule. A person may have 0 up to n scans, where n is the theoretical maximum number of records(no limit is enforced by the database or the application). A patient has characteristics explaned below

First Name	The first name of the patient.
Last Name	The last name of the patient
NiNo	The National Insurance Number(NiNo) of the patient
Enrolled Date	The Date that the patient was registered in the system
Ascosiate Doctor	The Doctor identification number, handling the case of the patient(see 4.2.3)
Comments	The Doctors(see 4.2.3) comments for the particular patient

4.2.3 Doctor

A doctor is a physical person with access on the system. Is the end-user of the system and has rights of uploading ultrasound image scans and retrieve predictions for those scans. It can also provide feedback to the system for a given prediction to be used for further research and develepoment. A doctor has specific characteristics presented below.

Username	Plain-text username
Password	MD5 Hashed[Rivest (1992)] and salted[Manber (1996)] password
First Name	National Insurance Number[*National Insurance Manual* (2021)]
Last Name	The date that the patient was registered in the system
Title	The title of the doctor.
Enrolled Date	The date and time of the user enrolled to the system
Last Seen	The date and time of last login of the user
Online Status	The status of the user, acceptable values are Connected,Not Connected
Tasks	The number of scans uploaded by the user

4.2.4 Notification

A notification is a short message from the system to the end-user(The doctor). Its sole purpose is to inform the user about various events that may interest the end-user. An example of this may be that the scan results for a given scan task are ready to view. A notification has specific characteristics witch are displayed and explained below.

Message	The message in question
Ascosiated Doctor	The receipient doctor identification number
Created Date	The Date and Time where the event in question where happened

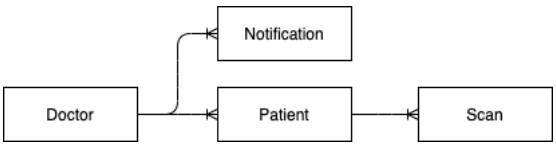
4.3 Entity Relations

The aforementioned entities have well defined relations. An exhaustive list is given below

- A doctor has many patients (1 - ∞)
- A doctor has many notifications(1 - ∞)
- A Patient has many Scans(1 - ∞)

A above relations can be summarized in the following E-R[1] Diagram

Figure 4.1: Entity-Relation Diagram



[1] Entity-Relation

Chapter 5

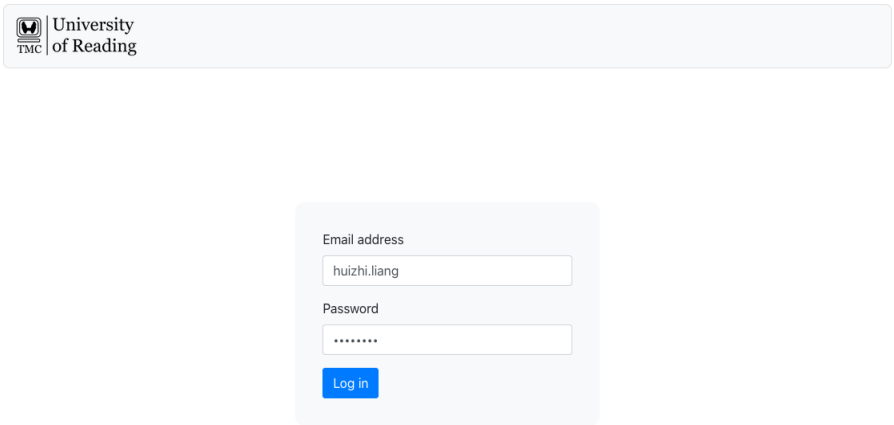
Users Perspective

5.1 Introduction

In this section, we will start our exploration of the application and its features. As the nature of the requirements of this the system is complicated. Unavoidably the system will be complex as well. Taking this into consideration, we will follow a natural top-to-bottom approach explaining its internals, starting as end-users and seeing the system as a black box. In this section, we will analyze its functionality from the user's perspective. This section may also serve as an instruction manual for the end-user as it contains everything needed for an inexperienced user to start working with the software.

5.2 Login and Authentication

Figure 5.1: Login Screen

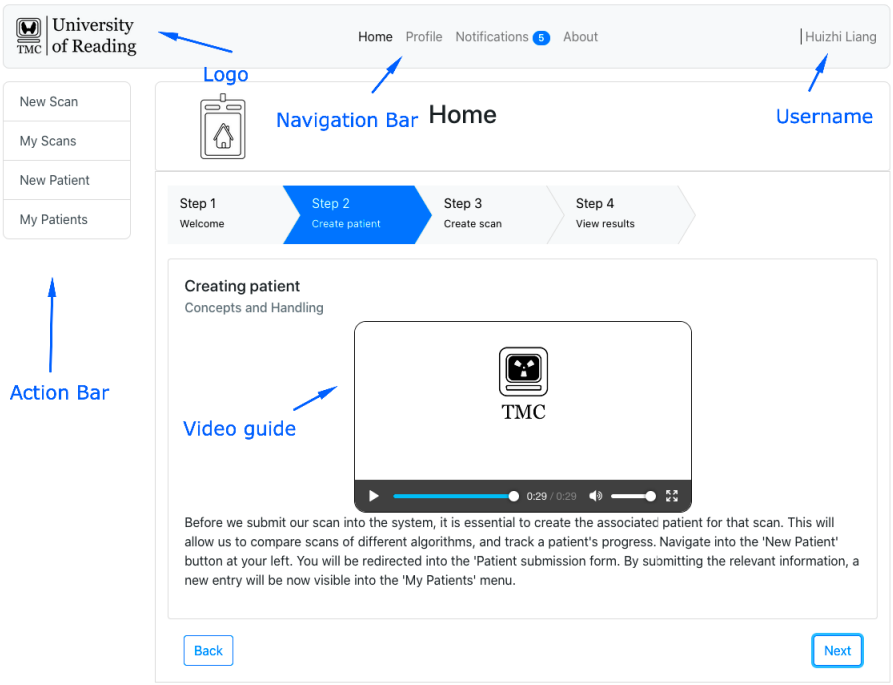


The login screen is the first screen that our end-users will encounter. Here a username and a password is required to be given by the user to log in. The Credentials of the user remain encrypted during the process of login, as the system utilizes an HTTPS[Rescorla (2000)] protocol for its connection, this is essential for the first non-functional requirement about security (see 3.3). The username and the password may be requested by the system administrator or the NOC[1] of the hospital.

[1]Network Operations Center

5.3 Home

Figure 5.2: Home Screen



After the login process is completed. The user encounters the 'home screen. From here, it is possible to navigate to the features of the software as well as learn about how the software can be utilized through detailed guides and videos. The UI/UX[2] has been designed to be as user-friendly as possible. Some areas of interest are given below.

Action bar	The Actions that can be performed using the software can be accesed from here.
Navigation bar	General information and notification bar.

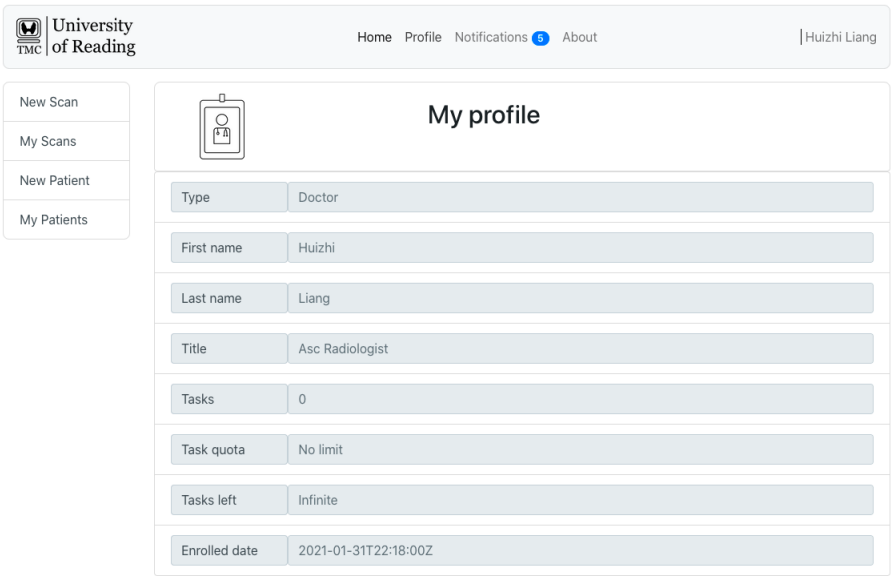
5.4 Navigation bar

In this section, we will briefly look at the options under the Navigation bar.

[2]User Interface-User Expieriance

5.4.1 Profile

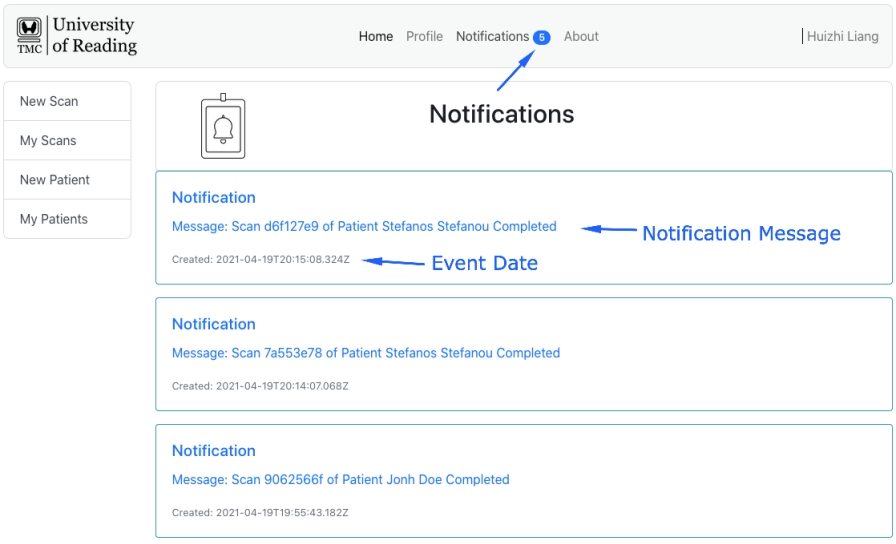
Figure 5.3: Profile



In the profile section, the user can see its associated information, saved on the registration date. The information for security reasons cannot be altered by the user itself, but only after a request to the system administrator or NOC[3].

5.4.2 Notifications

Figure 5.4: Notifications

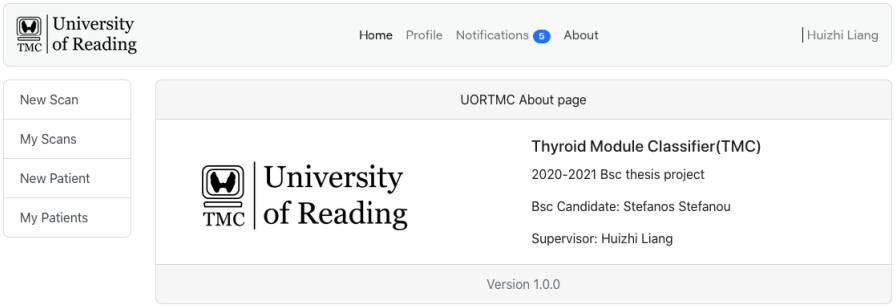


In the notification section, helpful information about events that may interest the end-user can be found, such as the fact that uploaded scan results are ready to view.

[3]Network Operations Center

5.4.3 About

Figure 5.5: About



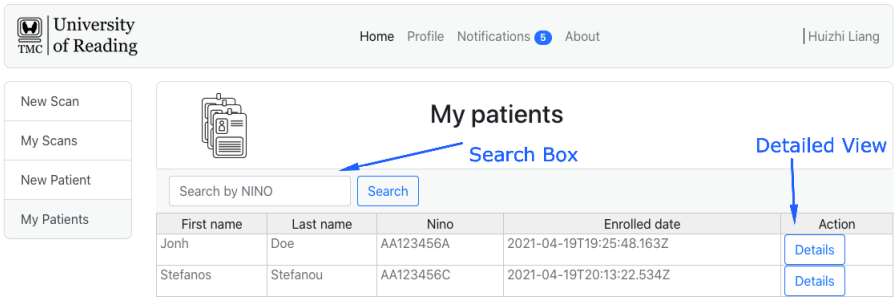
From here, a user may found helpful information about the software, such as the current version.

5.5 Action Bar

In this section, we will briefly look at the options under the Action Bar.

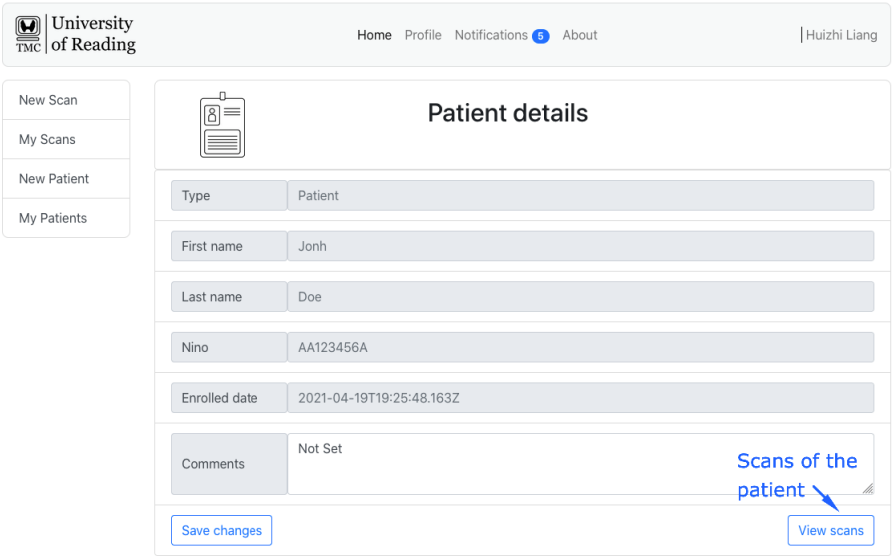
5.5.1 My Patients

Figure 5.6: Patients List



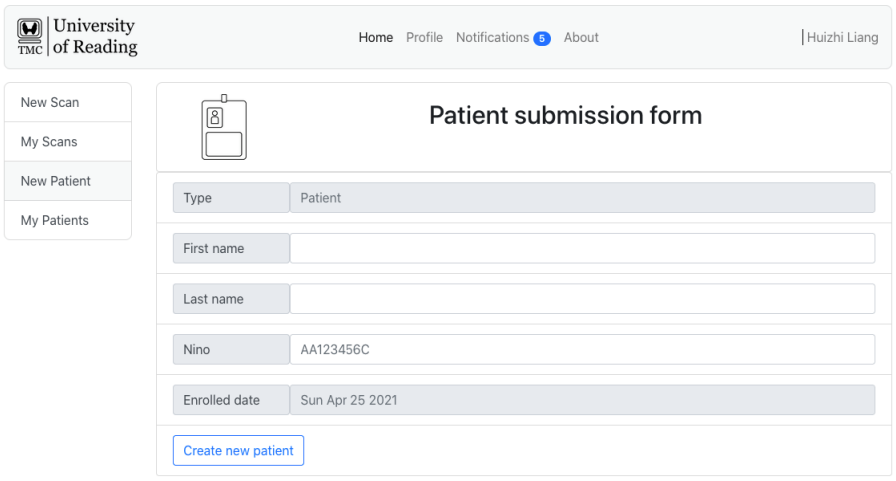
This page will show us a list of the currently registered patients. Each end-user(doctor) may only see its patients and not others. The end-user can search the list based on NiNo[*National Insurance Manual* (2021)] of the given patient for convenience. The end-user can also view the details of a given patient and record various notes/comments for that patient by clicking the 'Details' button on his selected patient, as seen below. Finally, clicking the button 'View Scans' can see the specific patient history of uploaded scans.

Figure 5.7: Patient Details



5.5.2 New Patient

Figure 5.8: New Patient

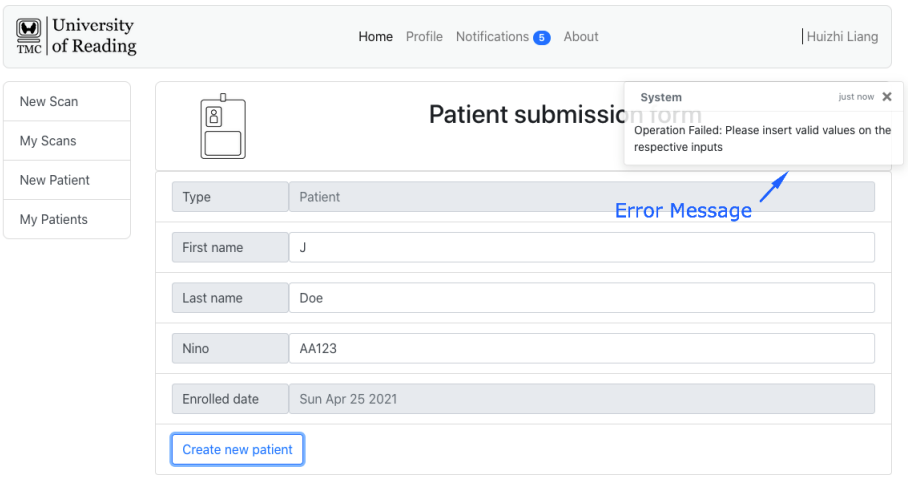


By clicking the 'New Patient' action on Action Bar, the user can register a new patient on the system. The following conditions need to be met for the operation to be successful.

- First name length should be more than 2 characters, encoded as UTF-8[Yergeau (2003)]
- Last name length should be more than 2 characters, encoded as UTF-8[Yergeau (2003)]
- NiNo should be at standard format [*National Insurance Manual* (2021)], encoded as UTF-8[Yergeau (2003)]

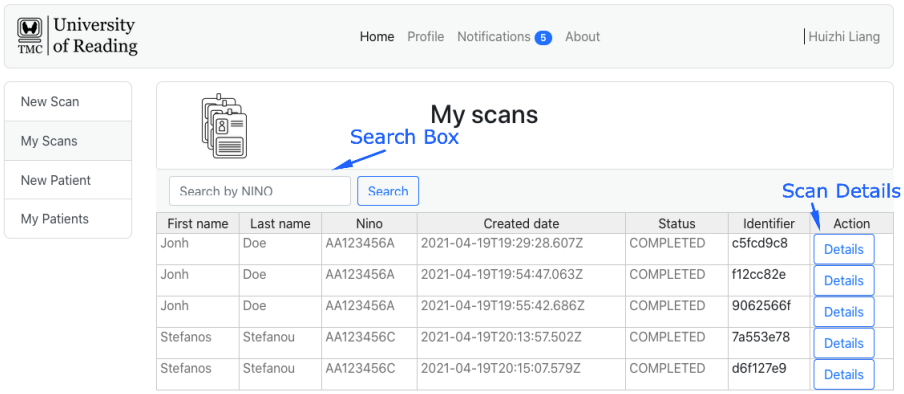
Failing to fulfill these constraints should lead to an error, as shown below.

Figure 5.9: New Patient Error



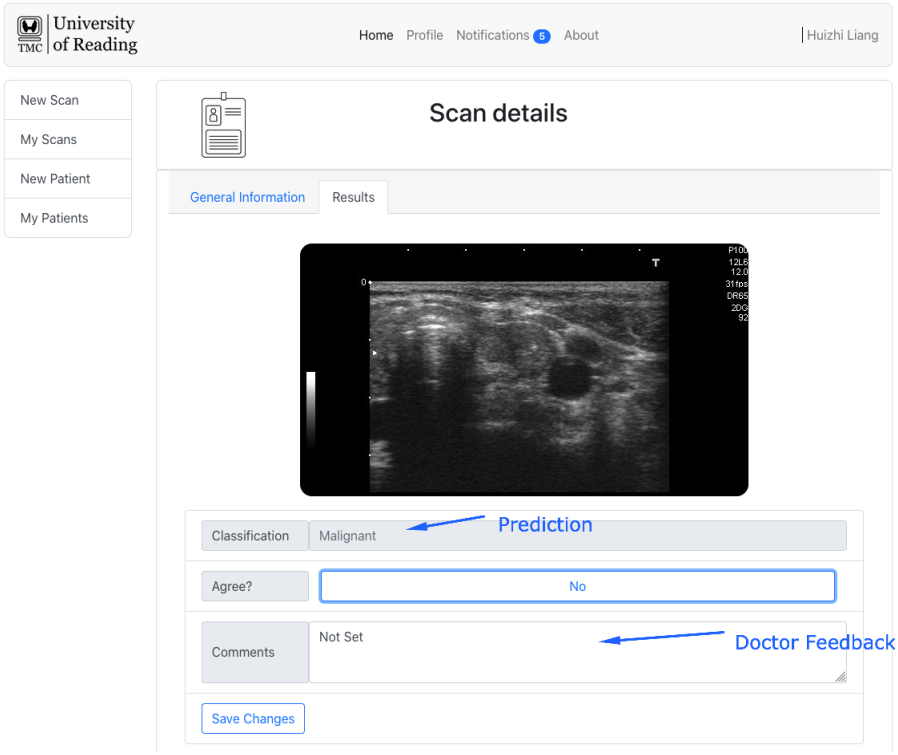
5.5.3 My Scans

Figure 5.10: My Scans



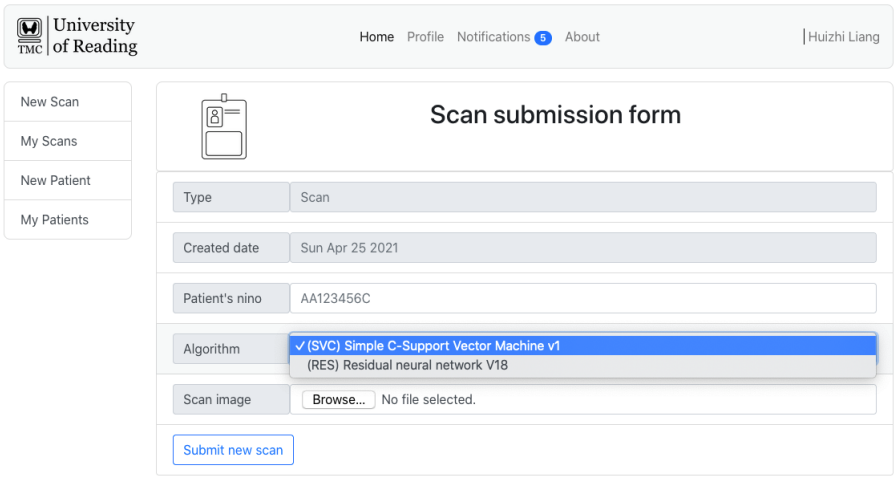
'MyScans' are a complete list with all submitted scans for a given end-user. The user can search for the scans of a specific patient by using the search box and viewing the scan results (if a given scan is complete) by clicking the 'Details' button of the scan in question.

Figure 5.11: Scan results



5.5.4 New Scan

Figure 5.12: New Scan



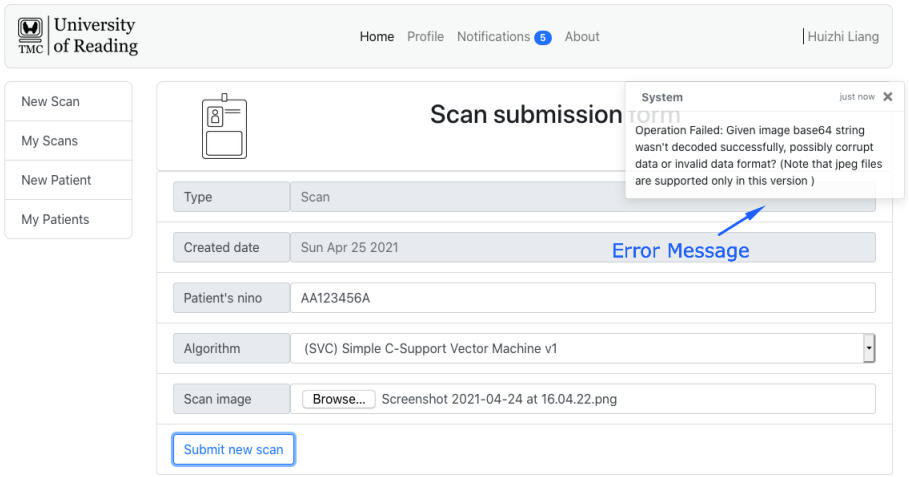
By clicking the 'New Scan' action, the user is redirected into the scan submission form. Here it is possible to submit a new ultrasound image for a given patient.The user can also select the algorithm for performing the classification (see 12). The operation to be completed should meet the following criteria.

- Patients Nino should be in standard format[*National Insurance Manual* (2021)] and encoded as UTF-8[Yergeau (2003)]

- Scan Image should be a ISO/IEC 10918-1/JPEG[International Organization for Standardization (1994)] format with 360x560 resolution. The image name should be encoded as UTF-8[Yergeau (2003)]

Failing to fulfill these constraints should lead to an error, as shown below.

Figure 5.13: Invalig image format (PNG) error



Chapter 6

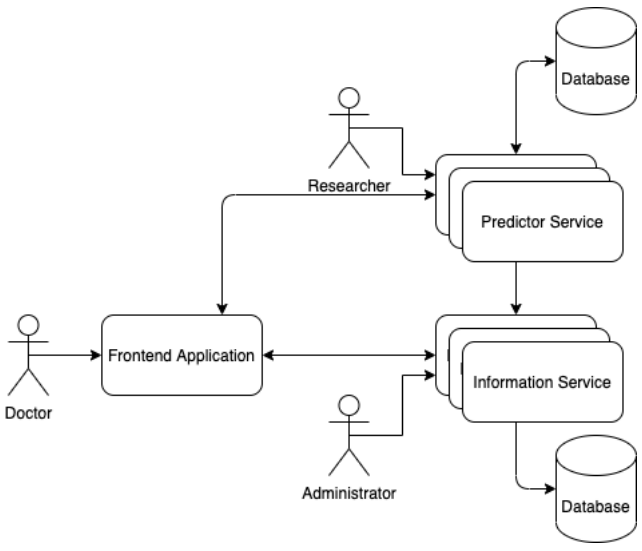
System Architecture

In this chapter, we will introduce the architecture of our system, explaining the essential elements that it is composed of and their interactions.

6.1 Overview

In the section 3.3 we discussed the non-functional requirements of this application. Two of the most important ones were security and performance. These requirements heavily influenced the design decisions of this project, leading to the microservice-inspired architecture [Newman (2020)]shown below.

Figure 6.1: Simplified Architectural Diagram



Microservice pattern[Newman (2020)] tries to decrease complexity and increase safety by splitting the internal logic of a system into several components called 'Microservices'. Each microservice is essentially a server that handles a small portion of the systems logic. As opposed to the monolithic services, microservices have a number of advantages that made them ideal for the requirements of this project such as.

- Highly maintainable and testable code
- Loosely coupled logic

- Independently deployable services
- No single point of failure
- Flexible scaleability factor
- Enchanced security

6.1.1 Maintainability

By implementing our system using microservices[Newman (2020)], we effectively separate the complex logic of our system into different services. This separation of complexity leads to several elementary and easily testable and maintainable entities. This brings down the maintenance costs.

6.1.2 Loosely coupled logic

Microservices ideally are loosely coupled. That means that changes tend to remain local to one microservice and do not span multiple ones. In case that the requirements change and new features are needed, the features will not affect a significant part of the code. This translates more negligible probability of occurring bugs and errors.

6.1.3 Indepedently deployable services

Using microservices gives us the advantage of deploying changes independently on the system, only on services that we need. This translates to fewer downtimes due to maintenance. An example of this will be a potential deployment of a new algorithm on the Prediction service. During the deployment of the new version of the Prediction service the system will be unable to perform predictions, but the rest of the functionality will be unaffected as it lies under a different microservice. With a monolith approach(all functionality in a single service), this would not be possible.

6.1.4 No single point of failure

Using microservices, we ensure that it will not propagate to the whole system when a failure occurs. In the hypothetical scenario of a failure in the Prediction service, the rest of the system's functionality will remain intact during the incident. This scenario with a monolithic architecture will bring the whole application into an unusable state.

6.1.5 Flexible scaleability factor

This is not sorely a feature of microservices but a combined feature brought by some additional design choices from within the code itself. Both of the services are implemented using the actor model [Hewitt (2015)]. The actor model allows, in combination with the microservice model, our services to act as a distributed system with multiple nodes; this allows us to scale different services when demand changes dynamically and automatically, ensuring the performance non-functional requirement we set back in 3.3. An example of this can be when multiple users simultaneously use the most advanced and CPU-Intensive algorithm, the ResNet(see 12). If the system determines that the load is beyond some threshold, it can spin multiple instances of the same service. The instances will coordinate themselfs automatically and split the work that needs to be done into equal amounts, reducing the response time. This is implemented using the Heroku-autoscale feature and will be discussed in chapter 11

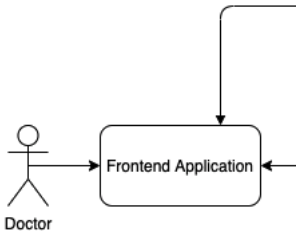
6.1.6 Enchanced security

Having multiple and distinct microservice enhances security, as the malicious compromise of one microservice does not imply the compromise of the whole system. In the hypothetical scenario of a malicious attacker may breach the Prediction Service, then the data of the Prediction Service will be at risk, but not the data from the Information System and vice versa.

6.2 Frontend Web App

The Frontend component has the responsibility of being the edge in our system.

Figure 6.2: Frontend Application



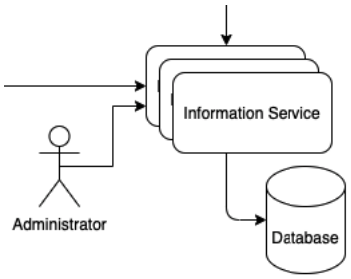
Every action from our end-users, will be channeled through the frontend application. Our frontend is a web-based application(for more information please see 9) and handles the application infastructure via a well designed stateful[Barth (2011)], Json-based[T. Bray (2014)] Https[Rescorla (2000)] RESTFull[R. Fielding (2014)] protocol (for more information please see chapter 10).

6.3 Backend

The backend services are the backbone of our application. They handle all the logic behind the application, from the saving and retrieval of patients, scans, images, and notifications, to the prediction and classification of the ultrasound images. There are two services with distinct areas of interest and different purposes, the Information Backend(also known as 'Information Service' to the simplified diagrams) and Task Backend(Also known as Predictor Service).

6.3.1 Information Backend

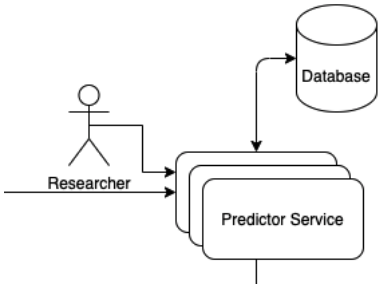
Figure 6.3: Information Service



The Information Backend has the responsibility to perform all the logic apart from the prediction itself. It includes the functionality of keeping the associations of Patients, Scans, and the information composing those entities. It also includes the notification system and authentication services. Finally, it includes a small application for use by the NOC[1] for administrator purposes.

6.3.2 Task Backend

Figure 6.4: Predictor Service



The Task Backend has the responsibility of performing the predictions based on the received ultrasound scan images. After completing a prediction, Task Backend should communicate with Information Backend to inform the user about the completion of the scan. Finally, it includes a small application for use by the researcher to gather the data and the feedback from the end-users.

[1] Network operations center

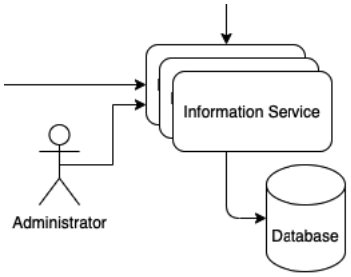
Chapter 7

The Administrators Perspective

7.1 Introduction

In this section, we will briefly go through the application interface for the administrator of the system. The administrator of the system has special rights and is assigned by the hospital that uses the software. It needs to belong to the NOC[1] of the hospital, and it is responsible for the maintenance of the software in the DevOps Level.

Figure 7.1: Information Service



7.2 Administrator Panel

The administrator has special tools for maintaining the system and intervene in its internals, a special administrator panel that gives access to a plethora of features that needs to be handled with care.

Note 1. *To connect to the administrator panel, we need to connect to the following addresses*

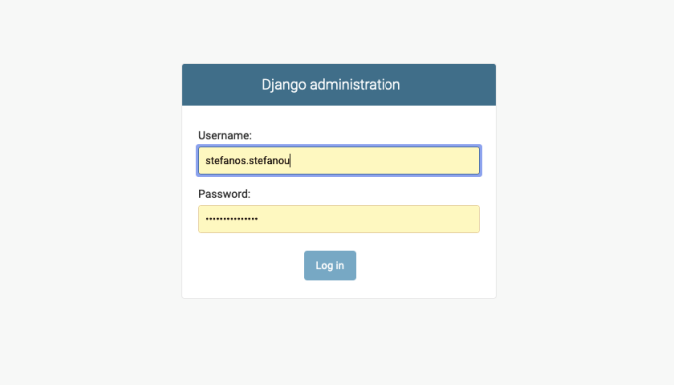
(if online) https://uortmc-infobe.herokuapp.com/admin/
(local machine)http://127.0.0.1:3001/admin

Please refer to the chapter 13 and chapter 14 for more details around how to connect.

[1] Network Operations Center

7.2.1 Admin Panel Login

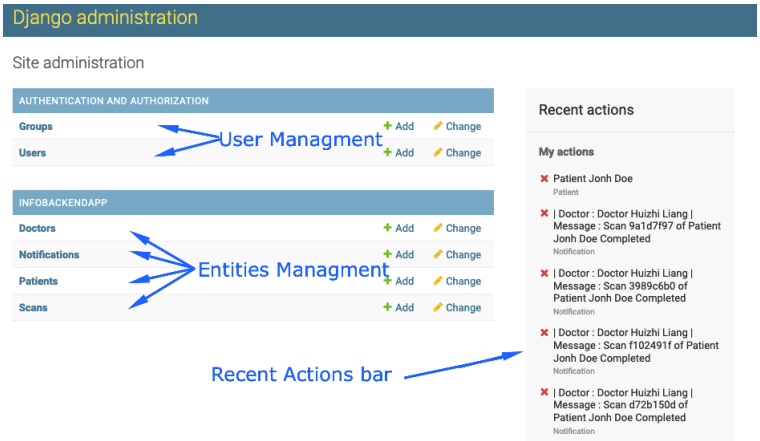
Figure 7.2: Administrator Panel



The login screen is the first screen that an admin should encounter. The information transmitted into the Information Service is encrypted using HHTPS[Rescorla (2000)] and transformed to an salted[Manber (1996)] MD5 Hash[Rivest (1992)] for maximum possible security.

7.2.2 Admin Panel Features

Figure 7.3: Administrator Panel-Home



After the login sequence is completed, the administrator will be redirected to the panel's home page; there, it has available all the functionality needed to perform changes on the system. An administrator has the right to alter the system's properties as well as the entity's attributes. We can add, alter and delete entities at will, using the Entities Management.

Figure 7.4: Example deletion of a scan

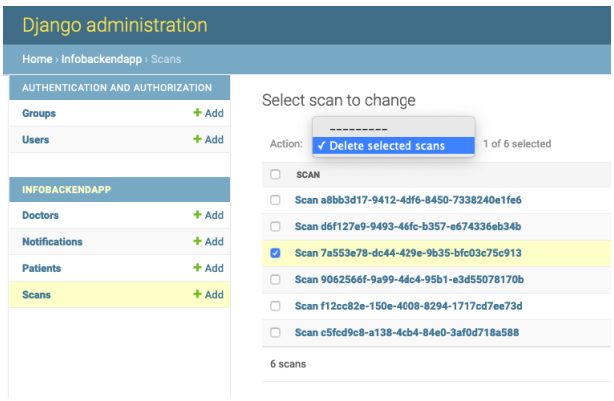
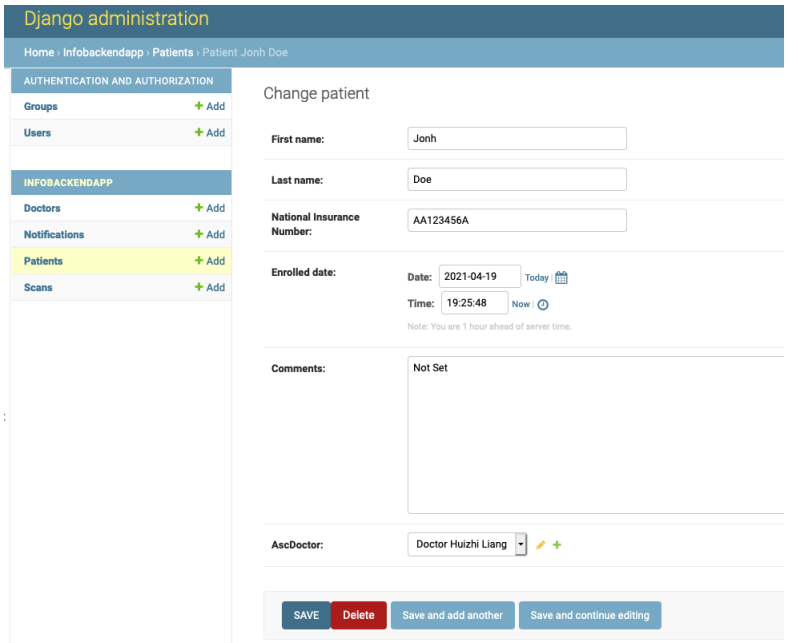


Figure 7.5: Example alteration of a patient



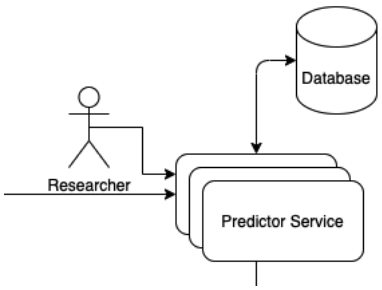
Chapter 8

The Researchers Perspective

8.1 Introduction

In this section, we will briefly look at the available features for the researcher of the project. Every scientist working in prediction models for thyroid nodule classification may upload its algorithm on the platform and receive helpful feedback about its performance using the Researcher panel explained below. Visually the researcher's panel is nearly identical to the administrator panel explained in chapter 7 but offers access to the different data than the administrator panel. This is done to reduce costs and reuse the similar functionality developed for the administrator panel. The researcher panel is provided by the Predictor Service(Task Backend).

Figure 8.1: Predictor Service



8.2 Researcher Panel

The researcher panel provides an interface to the researcher to view its algorithm outputs and performance in an easy and user-friendly manner.

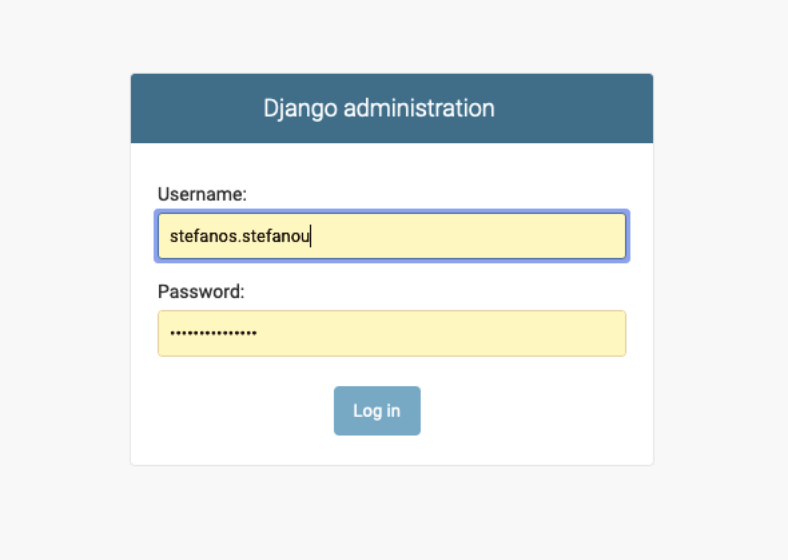
Note 2. *To connect to the researcher panel, we need to connect to the following addresses*

(if online) https://uortmc-taskbe.herokuapp.com/admin/
(local machine)http://127.0.0.1:3002/admin

Please refer to the chapter 13 and chapter 14 for more details around how to connect.

8.2.1 Login screen

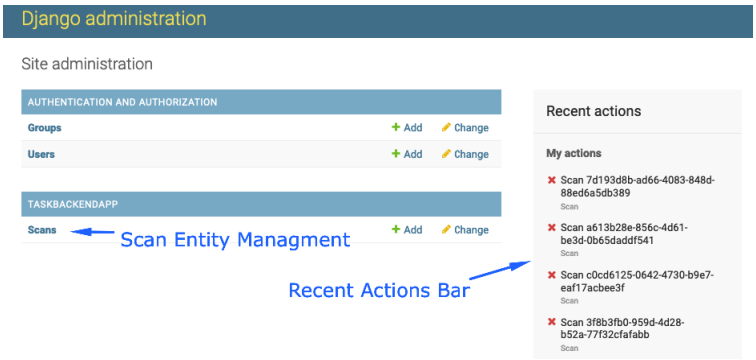
Figure 8.2: Researcher panel login screen



The login screen is the first screen that an researcher should encounter. The information transmitted into the Predictor Service is encrypted using HHTPS[Rescorla (2000)] and transformed to an salted[Manber (1996)] MD5 Hash[Rivest (1992)] for maximum possible security.

8.2.2 Researcher Panel Features

Figure 8.3: Researcher Panel-Home



After the login sequence is completed, the researcher will be redirected to the panel's home page; there, it has available all the functionality needed to perform debbuging into the algorithm under develepoment, such as real-time logging capability. By selecting the scan in question can have access to the required information

Figure 8.4: Researcher Panel-List of scans

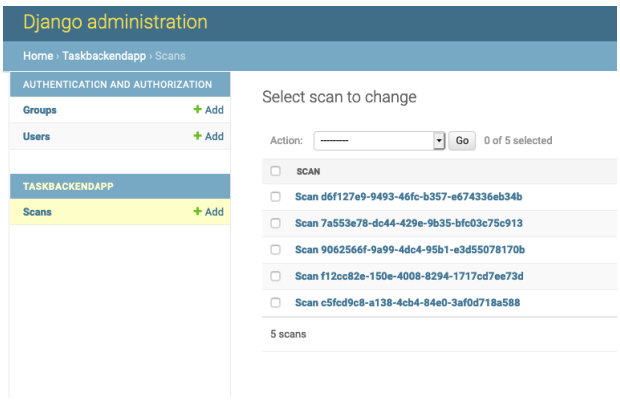
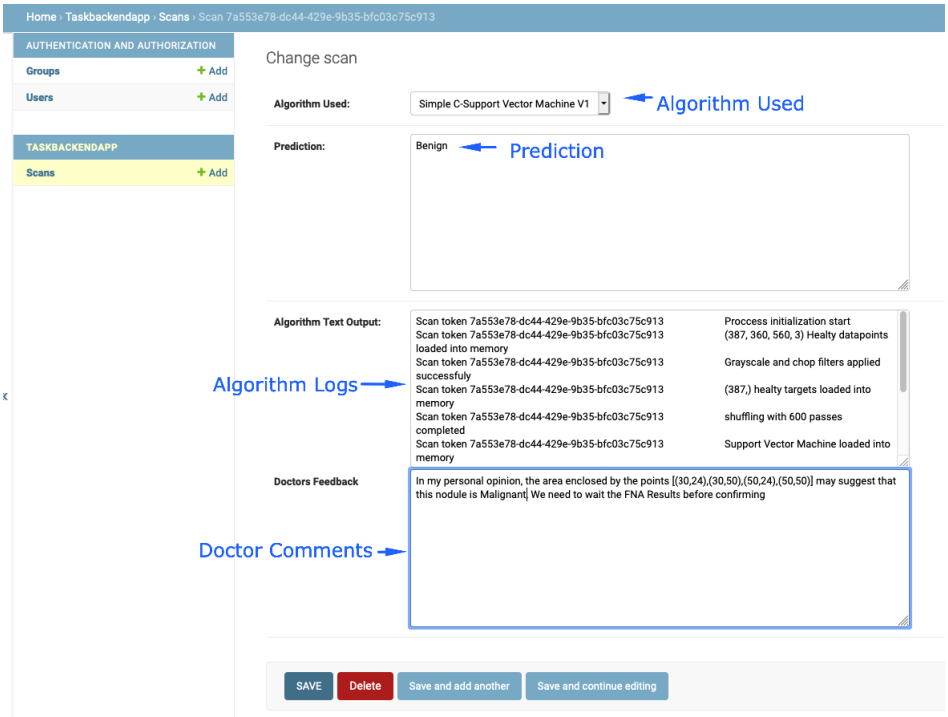


Figure 8.5: Researcher Panel-Scan Details Example



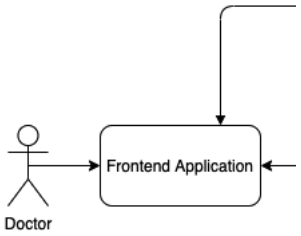
The researcher then can improve the algorithm based on the feedback provided from the doctors, as well as to troubleshoot possible errors using the real-time logging capability.

Chapter 9

The Frontend Application

9.1 Introduction

Figure 9.1: Frontend Application



This chapter will look at the technology stack, internals, and points of interest of the frontend application. The frontend application's purpose is to serve as a visual middleware between the end-user (The Doctor) and the system itself. It propagates the actions of the user into the system by using plain HTTPS Requests, via a stateful[Barth (2011)], Json-based[T. Bray (2014)] Https[Rescorla (2000)] RESTFull[R. Fielding (2014)] protocol (for more information, please see chapter 10).

9.2 Technology Stack

The Frontend application uses the following frameworks and libraries

- React v18.0
- Boostrap v4
- Axios Requests
- Hansontable v8.3.2

9.2.1 React

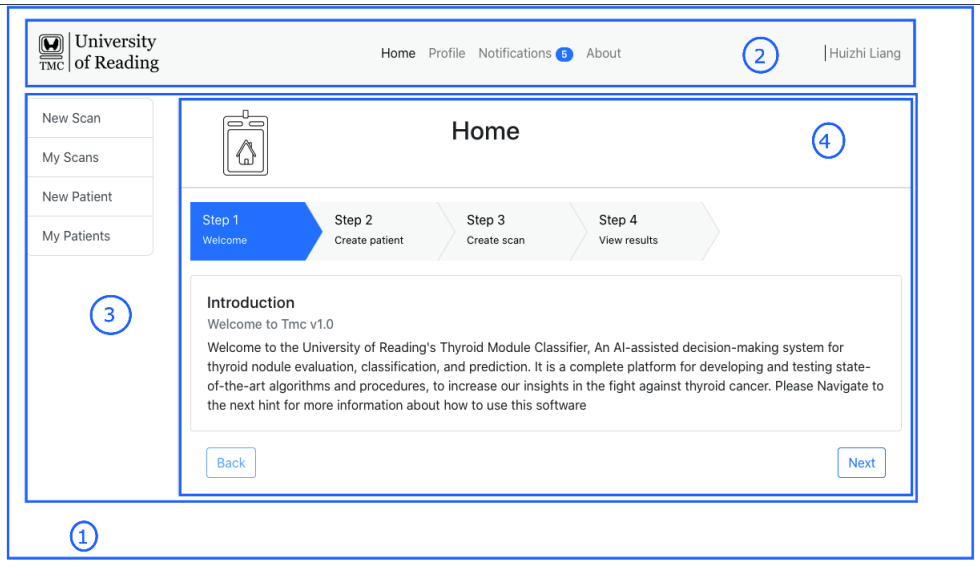
React.js is an open-source javascript framework for developing frontend applications. It is created by Facebook and maintained by the open-source community as well as from some individual companies. It encourages the creation of applications with well-defined state and

state transitions by composing lightweight and reusable UI Elements called 'Components.' The system's behavior is modeled strictly by events generated due to a state transition, and the components should act accordingly. Our Frontend Application contains 16 independent components that communicate with each other by callbacks. An exhaustive list of the components is given below.

Body	The entry-point application-wide component
About	The About page
Home	The Home Page
Hints	The Instruction manual at home page
Login	The login form
MyPatients	The patients list
MyScans	The scan list
NewPatient	The new patient form
NewScan	The new scan form
Notifications	The notifications list
PatientView	The patiet's detailed view
ScanView	The scan's detailed view
Profile	The users profile page
Nav	The Navigation Bar
Alert	The Alert message box
Content	Main view and Action bar

An example screen with its respective components highlighted is shown below...

Figure 9.2: React Components on example screen

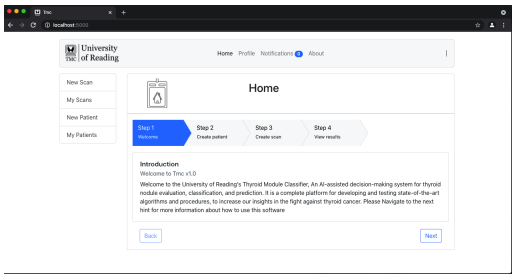


...where the marked areas are ascosiated on the respective components based on the mappings below.

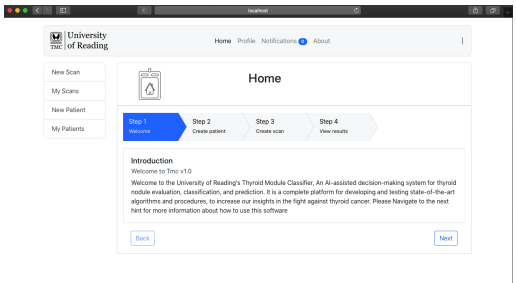
1	Body
2	NavBar
3	Content
4	Hints

9.2.2 Boostrap

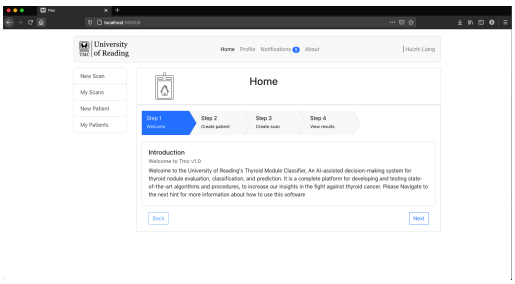
Bootstrap is an open-source CSS and Javascript web development network. It contains web templates, as well as forms, buttons, and navigation elements. Boostrap is used extensively throughout the frontend application to enable a user-friendly experience and cross-compatibility to various browsers and mobile devices.



(a) System Works on Chrome



(b) System Works on Safari



(c) System Works on Firefox

Figure 9.3: Boostrap runs consistetly everywhere

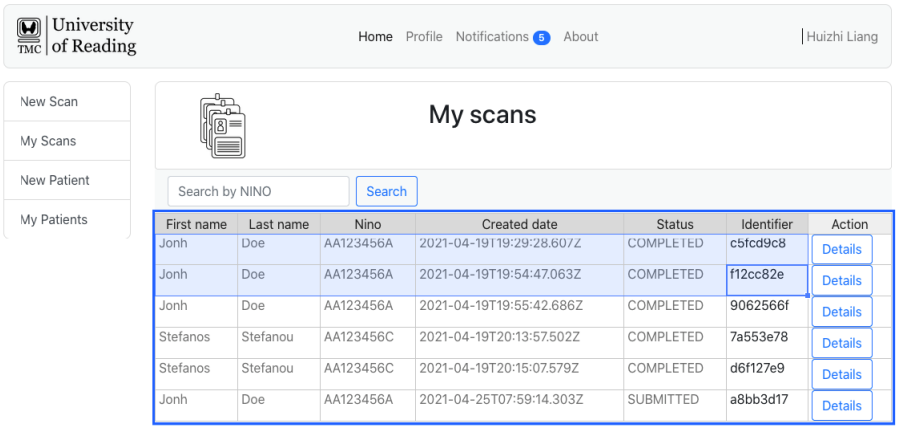
9.2.3 Axios Requests

'Axios is a popular, promise-based HTTP client that sports an easy-to-use API and can be used in both the browser and Node.js.'[Jacques (2018)]. Axios enables the communication of the Frontend Application with the Backend Services; it does this asynchronously, so it does not block the main thread of execution; this feature enables the application to be as smooth and responsive as possible, even under heavy loads(If the backend services are not responding fast enough, functionality non needed communication is not affected.)

9.2.4 Hansontable

Hansontable is a javascript library providing a fully functional MS Excel-style spreadsheet for generic use. We use this spreadsheet on two components, MyPatients and MyScans.

Figure 9.4: Hansontable example with the first two lines selected



Chapter 10

The Application Programming Interface

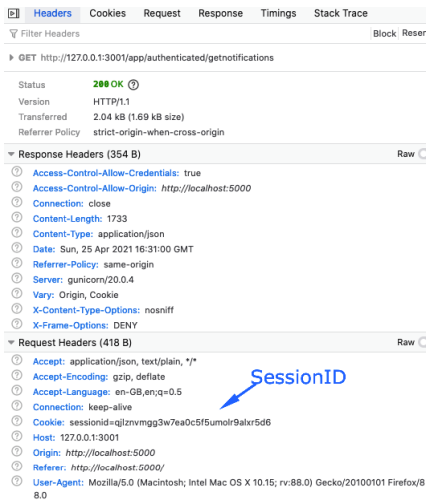
10.1 Introduction

In this section we will briefly discuss the commnunication protocol used by the application to its internal components. The protocol used as mentioned already in the previous chapters, is a designed stateful[Barth (2011)], Json-based[T. Bray (2014)] Https[Rescorla (2000)] RESTFull[R. Fielding (2014)] protocol. Let us explain briefly what those techical terms mean.

10.1.1 Stateful Protocol

Stateful protocol[Barth (2011)] is a protocol capable of recognising and distinquishing between the different requests made by the same host machine. In our application this is essential because the authentication functionality would be impossible otherwise. An authenticated user is always ascosiated with a sessionID. The sessionID[Kaplan (2014)] is a character string that is returned after the authentication is complete and should be attached to every subsequent request for user authentication to work. Our application protocol uses the cookie mechanism to attach the sessionID to every request, ensuring that the Services will recognise the sender. The following image provides an example

Figure 10.1: Session ID Attached to Notifications Request, captured using Firefox-Tools

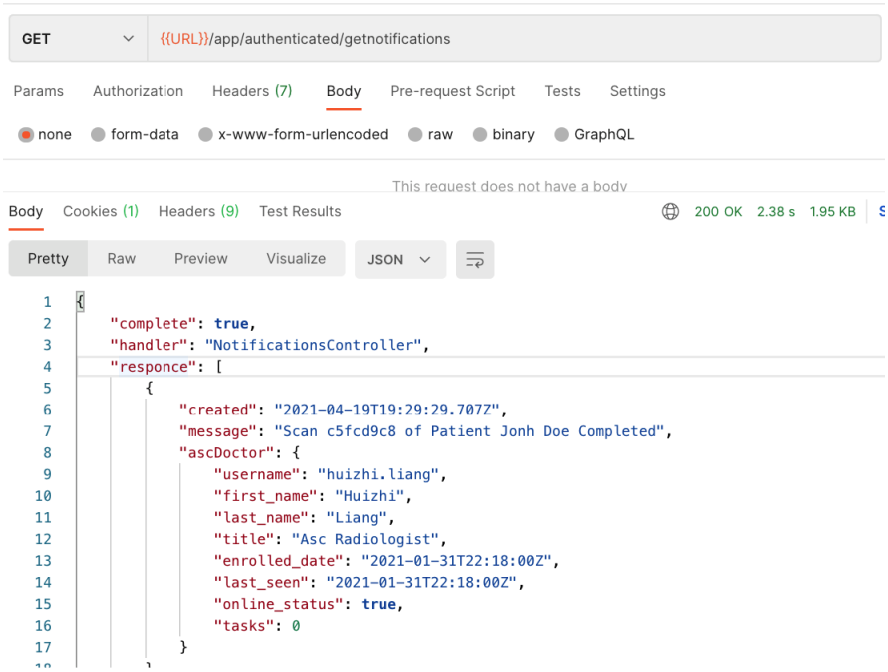


Here, the Session ID is attached to a request from the Frontend Application into the Information Service, for the retrieval of new notifications

10.1.2 Json

The JavaScript Object Notation (or JSON) Data Interchange Format[T. Bray (2014)] is a text-based, lightweight, human-readable language-indepedent data exchange format. We use JSON extensively to design our communication protocol, mainly because of his widely adopted use and availability of decoders. Additionally, both languages involved in the data transaction(javascript for the frontend, python for the backend services) are supporting JSON natively.

Figure 10.2: Capturing the Information service's JSON-Based responce using Postman



10.1.3 HTTPS

HTTPS or HyperText Transfer Protocol Secure version, is a updated version of classic HTTP protocol, using TLS as an additional security layer. TLS encrypts all the underlying data to provide unparallel protection against various malicious attacks. The following images shown the login packages as they sent from the Frontend Application to the Information Service using the WireShark packet analyser.

Figure 10.3: Unecrupted Credentials (HTTPS disabled)

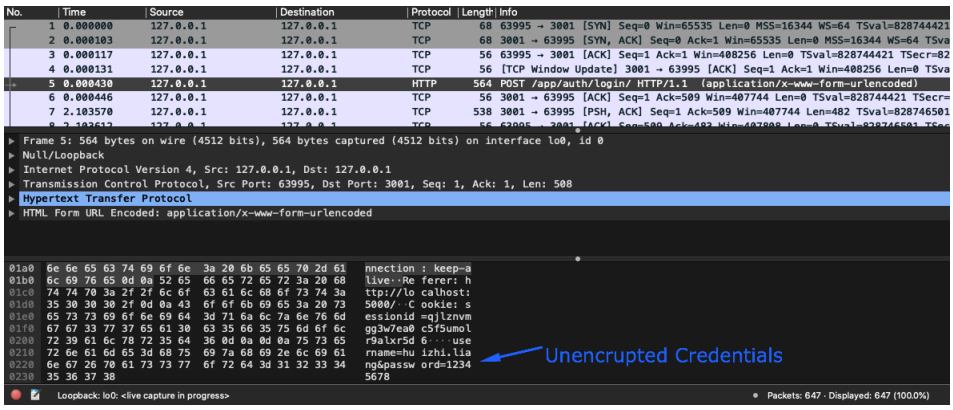
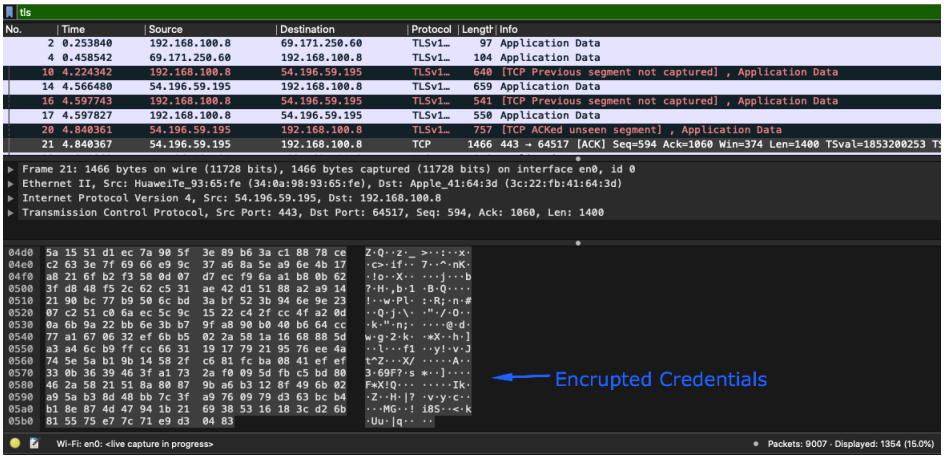


Figure 10.4: Ecrupted Credentials (HTTPS enabled)



Becomes evident that, by using HTTPS we increase our system security, as credentials and personal information are encrupted before sent over the internet.

10.1.4 RESTFull protocol

Chapter 11

The Backend

...

11.1 ...

....

11.2 ...

...

11.2.1 ...

11.3 Summary

...

Chapter 12

The Prediction Service

...

12.1 ...

....

12.2 ...

...

12.2.1 ...

12.3 Summary

...

Chapter 13

CICD-Versioning-Deployment

...

13.1 ...

....

13.2 ...

...

13.2.1 ...

13.3 Summary

...

Chapter 14

Starting The Application Locally

...

14.1 ...

....

14.2 ...

...

14.2.1 ...

14.3 Summary

...

Chapter 15

Discussion, Conclusion and Future work

...

15.1 ...

....

15.2 ...

...

15.2.1 ...

15.3 Summary

...

Chapter 16

Reflection

...

16.1 ...

....

16.2 ...

...

16.2.1 ...

16.3 Summary

...

References

Barth, A. (2011), 'HTTP State Management Mechanism', Internet Requests for Comments.
URL: *https://tools.ietf.org/html/rfc6265*

Eun Ju Ha, J. H. B. (2021), 'Ha ej, baek jh. applications of machine learning and deep learning to thyroid imaging: where do we stand? ultrasonography. 2021 jan;40(1):23-29. doi: 10.14366/usg.20068. epub 2020 jul 3. pmid: 32660203; pmcid: Pmc7758100.'.
URL: *https://pubmed.ncbi.nlm.nih.gov/32660203/*

Fine Needle Aspiration Biopsy of Thyroid Nodules (2019).
URL: *https://www.thyroid.org/fna-thyroid-nodules/¿*

Hewitt, C. (2015), 'Actor model of computation: Scalable robust information systems'.

International Organization for Standardization (1994), 'Information technology — digital compression and coding of continuous-tone still images: Requirements and guidelines', ISO/IEC 10918-1.

Jacques, N. (2018), 'Introducing axios, a popular, promise-based http client'.
URL: *https://www.sitepoint.com/axios-beginner-guide/*

Kaplan, H. (2014), 'A Session Identifier for the Session Initiation Protocol (SIP)', Internet Requests for Comments.
URL: *https://tools.ietf.org/html/rfc7329*

Leach, P. (2005), 'A Universally Unique IDentifier (UUID) URN Namespace', Internet Requests for Comments.
URL: *https://tools.ietf.org/html/rfc4122*

Manber, U. (1996), 'A simple scheme to make passwords based on one-way functions much harder to crack', *Computers and Security* **15**(2), 171–176.
URL: *https://www.sciencedirect.com/science/article/pii/016740489600003X*

National Insurance Manual (2021).
URL: *https://www.gov.uk/hmrc-internal-manuals/national-insurance-manual/nim39110*

Newman, S. (2020), *Monolith to microservices: evolutionary patterns to transform your monolith*, O'Reilly Media, Inc.

R. Fielding, E. (2014), 'Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content', Internet Requests for Comments.
URL: *https://tools.ietf.org/html/rfc7231*

Rescorla, E. (2000), 'HTTP Over TLS', Internet Requests for Comments.
URL: *https://tools.ietf.org/html/rfc2818*

Rivest, R. (1992), 'The MD5 Message-Digest Algorithm', Internet Requests for Comments.
URL: *https://tools.ietf.org/html/rfc1321*

T. Bray, E. (2014), 'The JavaScript Object Notation (JSON) Data Interchange Format', Internet Requests for Comments.
URL: *https://tools.ietf.org/html/rfc7159*

Yergeau, F. (2003), 'UTF-8, a transformation format of ISO 10646', Internet Requests for Comments.
URL: *https://tools.ietf.org/html/rfc3629*