UnFold It

Automated Machine Learning Web-App

Ideation Document

1.0

April 20, 2020

Submitted in full partial of the requirements of
Minor Project (SUJECT CODE)
Instructor: Prof. Divya Kumari Sharma

Team: The Unfolders
(Group - X)

Roll Number	Name	Position	Contribution
1706119	Avinash Soni	Team Member	Front-End Static Content Creator and Team Management
1706122	Ayush Agrawal	Team Leader	ML Specialization, Generating required models and scripts
1706124	Biplav Adhikary	Team Member	Web Developer, Server-Side Coding, handling Dynamic Pages and Project Integration
1706130	Gulam Muhammad	Team Member	Content Creator, and official Documentation handling

School of Computer Engineering
Kalinga Institute of Industrial Technology
Deemed to be University
Bhubaneswar-751024
Autumn 2019

KIIT Deemed to be University

School of Computer Engineering
Bhubaneswar, ODISHA
751024

CERTIFICATE

This is to certify that the project entitled

UnfoldIt 'Automated Machine Learning Web-App'

is a record of Bonafede work carried out by

AVINASH SONI - 1706119
AYUSH AGRAWAL - 1706122
BIPLAV ADHIKARY - 1706124
GULAM MUHAMMAD - 1706130

in the partial fulfilment of the requirement for Minor Project (6th Semester) from Department IT (School of Computer Science and Engineering) at KIIT Deemed to be university, Bhubaneswar. This work is done during year 2018-2019, under our guidance.

Date: 25 April, 2020

Professor Divya Kumari Sharma

ACKNOWLEDGEMENT

We would like to express our profound gratitude to Prof. Divya Kumari Sharma, for her expert guidance and encouragement throughout which helped us in accomplishing this project. We would also like to thank the CS, T&P and CAAS Department for allowing us to conduct the project. We came to know a lot many new things during the course of this project. And therefore, we are really thankful to them. This project would not have been accomplished successfully without everyone's cooperation and efforts.

ABSTRACT

With the vast and continuous increase in the amount of data in our digital world, it has been acknowledged that the number of knowledgeable data scientists cannot scale to address these challenges. Thus, there was a crucial need for automating the process of building good machine learning models. In the last few years, several techniques and frameworks have been introduced to tackle the challenge of automating the process of Combined Algorithm Selection and Hyper-parameter tuning (CASH) in the machine learning domain. The main aim of these techniques is to reduce the role of the human in the loop and fill the gap for non-expert machine learning users by playing the role of the domain expert. In this project we present a comprehensive survey for the state-of-the-art efforts in tackling the CASH problem along Auto-Visualization. In addition, we highlight the research work of automating the other steps of the full complex machine learning pipeline (AutoML) from data understanding till model deployment. Furthermore, we provide comprehensive coverage for the various tools and frameworks that have been introduced in this domain. Finally, we discuss some of the research directions and open challenges that need to be addressed in order to achieve the vision and goals of the AutoML process.

Keywords

Dataset, Classifier, Visualizer, Automation, Automated ML, Hyper-Parameters, CASH

Table of Contents

1. Introduction

1.1. Purpose

Data is increasing day by day. Over 2.5 quintillion bytes of data are created every single day, and it's only going to grow from there. But there exists only a selected few people who has the knowledge and skill about to process this data and find out meaningful insight from this data. Suppose a user has sample datasets, but he/she doesn't have any means to process this data, i.e., extract information out of this data and visualize this data properly to understand the data. Sometimes all we want is only to predict the trends from this data, not diving deep into technical stuffs.

1.2. Scope of Project

Any person, be it student, a businessman or someone ranking as high as an IT Engineer who has appropriate data collected with him but lacks the competency to analyze the data, or he doesn't want to invest time and effort into coding lines of code for simple visualization or prediction can use our Web-App to find appropriate insights on this data. This will give him a brief idea and information about his collected data.

1.3. Motivation

In recent years, the demand for machine learning experts has outpaced the supply, despite the surge of people entering the field. To address this gap, there have been big strides in the development of user-friendly machine learning software that can be used by non-experts. The first steps toward simplifying machine learning involved developing simple, unified interfaces to a variety of machine learning algorithms (e.g. H2O, TPOT).

Organizations generally face the following issues:

- Availability of expert machine learning developers
- Efficiently scaling trained models to production environment
- Cost

To address this issue, we want to develop a system, where you won't have to invest in an ML expert and waste effort and time in writing lines of code to for simple predictions and visualizations.

1.4. Related Works

The following the some works based on the concept of Automated Machine Learning:

- **Eclipse Arbiter (Java):** grid and random search, and genetic search for neural architectures.
- Auto-WEKA (Java)
- BayesOpt (C++)

- Featuretools: a good library for automatically engineering features from relational and transactional data
- **auto-sklearn:** a drop-in replacement for scikit-learn estimators.
- MLBox: distributed data processing, cleaning, formatting, and state-of-the-art algorithms such as LightGBM and XGBoost. It also supports model stacking, which allows you to combine an information ensemble of models to generate a new model aiming to have better performance than the individual models.
- **Xcessive**: A web-based application for quick, scalable, and automated hyperparameter tuning and stacked ensembling in Python
- **TPOT:** genetic programming to find the best performing ML pipelines, and it is built on top of scikit-learn
- Advisor
- Hyperopt
- Hyperopt-sklearn
- Spearmint
- SMAC3
- RoBO
- BayesianOptimization
- Scikit-Optimize
- HyperBand
- Optunity
- ATM
- Cloud AutoML
- SigOpt
- H2O
- DataRobot

1.5. References

- Hands-On Machine Learning – Aurellien Geron
- Natural Language Processing Python – Deepti Chopra
- Python - https://docs.python.org/3/
- Flask - https://flask.palletsprojects.com/en/
- Wikipedia - www.wikipedia.com
- HTML, CSS, JavaScript - https://www.w3schools.com/

- AutoViz Visualizer – https://danrothdatascience.github.io/datascience/autoviz.html
- Articles - StakeHolderMap
- Diagrams – Created using LucidChart Web Application

2. Objective

We, by means of our project, want to find a way for users to inspect, analyze. visualize and predict results from his data by automating all his tasks. All the user has to do is first upload the train dataset, and choose a few options, and later on the test dataset and we'll do everything else for him. Here exact steps that the user needs to follow:

▪ Visit our website where you will have the facility to upload the train dataset (preferably pre-processed one).

▪ After that you have to choose your problem, whether it is a classification or a regression problem.

▪ Next, you'll will have to identify the predictive attributes. With this, the data collection phase will be over.

▪ Now, everything will be done by our Automation Engine. We'll determine the hyper-parameters and the best possible Machine Learning model for your provided dataset. This marks the end of the Modelling phase.

▪ Now all you need to do is just upload the actual dataset whose values need to be predicted or classified.

▪ Using the previously created model, you can now visualize your data and you will be able to see his predictions for the uploaded the test set.

3. Definitions and Overview

3.1 AutoML Interface

The H2O AutoML interface is designed to have as few parameters as possible so that all the user needs to do is point to their dataset, identify the response column and optionally specify a time constraint or limit on the number of total models trained.

In both the R and Python API, AutoML uses the same data-related arguments, **x, y, training_frame, validation_frame,** as the other H2O algorithms. Most of the time, all we need to do is specify the data arguments. We can then configure values for max_runtime_secs and/or max_models to set explicit time or number-of-model limits on your run.

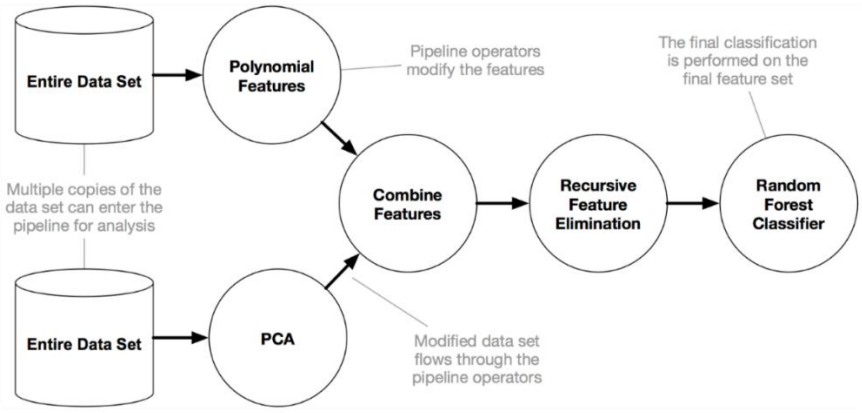
→ **Advantages of AutoML:**

Different organizations and industries have to deal with lot of data, be it related to the products or the employee and this data can be very useful for understanding things properly but every organization can't afford a data scientist.

3.2 TPOT Classification


```
class tpot.TPOTClassifier(generations=100, population_size=100,
                          offspring_size=None, mutation_rate=0.9,
                          crossover_rate=0.1,
                          scoring='accuracy', cv=5,
                          subsample=1.0, n_jobs=1,
                          max_time_mins=None, max_eval_time_mins=5,
                          random_state=None, config_dict=None,
                          template=None,
                          warm_start=False,
                          memory=None,
                          use_dask=False,
                          periodic_checkpoint_folder=None,
                          early_stop=None,
                          verbosity=0,
                          disable_update_check=False)
```

3.2 TPOT Classification



The TPOTClassifier performs an intelligent search over machine learning pipelines that can contain supervised classification models, pre-processors, feature selection techniques, and any other estimator or transformer that follows the scikit-learn API. The TPOTClassifier will also search over the hype parameters of all objects in the pipeline.

By default, TPOTClassifier will search over a broad range of supervised classification algorithms, transformers, and their parameters. However, the algorithms, transformers, and hyper parameters that the TPOTClassifier searches over can be fully customized using the config_dict parameter.

2.1 Product Perspective

Our APP/Website works on the pre-processed datasets which needs more modification for better understanding of it.

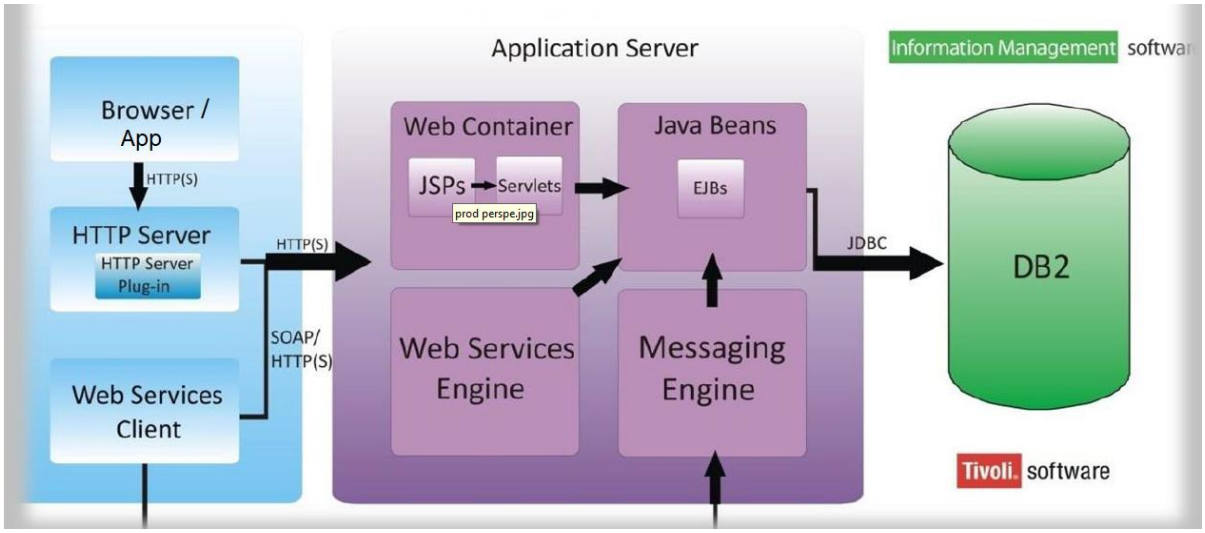


Figure 2.1 - System Environment

2.2 Product Functions

There are several major functions that **UnfoldIt** will perform. Once the user has started the application/Website, the software connects to the company's repository. Once a connection has been established, all other major functions of the software can be performed. **UnfolIt** automatically checks the repository for any changes, and updates all local information accordingly. The results came out from the from the operations performed on the datasets is completely based on the corrective-ness of the provided data by user and after that if the user is still not satisfied by the result then our helping team will come forward to assist the user in getting the desired output.

2.3 User Characteristics

There are no major prerequisites required for using our application. The user must be aware of the basic terms related to Machine Learning, so that he/she can have complete

understanding of the full process that has been performed internally and externally. He should be well acquainted with the use of web browsers in a computer system or navigating through the app.

2.4 Constraints

- Pre-processed data and clean data sets are allowed.
- User must have the basic knowledge that how ML feature works.
- User must be having the under stability of ML output forms.
- Limited to HTTP/HTTPS. This system is working for single server.

2.5 Assumptions and Dependencies

- The user responding to our bot is a genuine user (a human being), not another bot created to replicate your answers on the user's stead.
- The details shared to our bot is accurately correct (like description of his condition). Because only based on the user's response, our bot is going to predict.
- The medicines suggested by our bot is subjected to the availability in the market.
- We are only suggesting basic non-prescription grade medicines to the users.
- The user is having a working internet connection.

This section is catch-all for everything else that might influence the design of the system and that did not fit in any of the categories above.

2.6 Apportioning of Requirements.

The very first version that we will be designing will be having the functionality only to sign up, sing in and chat with the bot and the bot will only be able to suggest the medicines. The later versions will be having the functionality to buy the predicted medicines. There will be a team of doctors available for conversing with the patients in case if the user is not satisfied with the bot's response, the doctor will be replying him once he gets available.

3. Specific Requirements

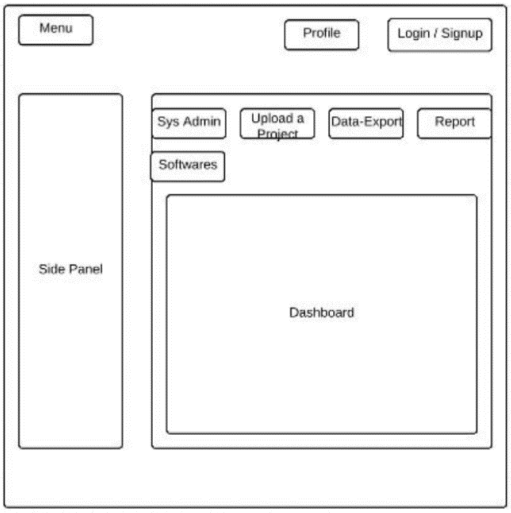
This section contains all of the functional and quality of the system. It gives a detailed description of the system and all its features.

This section provides a detailed description of all inputs into and outputs from the system. It also gives a description of the hardware and software interfaces and provides basic prototypes of the User interface.

3.1.1 User Interfaces

The major interface of our system is:

Dashboard () The dashboard will be accessible with all user types. Here they will perform respective tasks according to their permission level.



3.1.2 Hardware Interfaces

Even though this architecture is hardware-software integrated web architecture, we will not be designing any specific hardware interface to run the system. Our system is a web-based system, so, we will be launching it in several computers online.

3.1.3 Software Interfaces

Front end client on internet:
Name-react JS 16.8.6
We are working on ReactJS due to availability of huge libraries and frameworks for it.

Web Server:
Name- WASCE 3.0, Debian Operating System 10.0

Database Server:
Name- DB2 10.5, Debian Operating System 10.0

Development End:
Name- Node JS 10.0.0
We are working on NodeJS as it uses Native JS which is great for optimizing loading time for

3.1.4 Communications Interfaces

The communication interfaces we will be using are:
Client (customer) on internet will be using HTTP/HTTPS protocol.
Client (system user) on internet will be using HTTP/HTTPS protocol.

3.1.5 Memory Constraints

Client Side:
Minimum disk space should be 512MB of RAM on a PC or 256MB of RAM on Android Device.

Server side:
Minimum disk space should be 2GB/client.

3.1.6 Operations

The normal and special operations required by the user are:

1. The health care operations are certain administrative, financial, legal and quality improvement activities to run its business and to support the core functions of treatment and payment.
2. Case management, care coordination &business management.
3. Streamlining and optimizing utilization of various past assessments of a patient.
4. Reduce re-admissions and encourage regular backing up of data for further information.

3.1.7 Site Adaptation Requirements

1. A well-designed feature to upload previous records and diagnosis treatments of older patients and add them to our database for increasing the future efficiency of our bot by training it with the new datasets.

2. Doctors will have the ability to change the prescribed medicines by our bot for certain serious diseases.

3. After chatting about his/her condition, his/her responses about the disease will be fed to his medical history database from the chat screen internally and automatically.

4. The patient should be able to upload his/her medical test reports (if prescribed any) and it will be analyzed by a doctor and it be shown in the dashboard once analyzed.

3.2 Functional Requirements

This section describes specific features of the software project. If desired, some requirements may be specified in the use-case format and listed in the Use Cases Section

Welcome Page and Dashboard

Figure: 3.2

3.2.1.1 Introduction

This page will basically greet the users for choosing us by using our web-based service or app. And will be having various options to view his uploaded reports, analyzed reports, recent suggestions for medicines and diseases if he/she patient.

And in case if a doctor logs-in, he/she will be having a different version dash-board activity.

3.2.1.2 Inputs

The user will only need to enter into our main website or app interface by clicking on the continue button.

The user will have the ability to view any option shown in the dashboard by clicking on the activity tabs / button for it.

3.2.1.3 Processing

All the processing that will be required is for establishing the connection between the client and the server and in loading of the page when any button is clicked.

3.2.1.4 Outputs

The user will be shown the welcome page on the first loading. And clicking on dashboard activity, it will redirect to the full screen of that activity.

3.2.1.5 Error Handling

No error handling mechanism is required for this section.

Sign-In / Sign Up

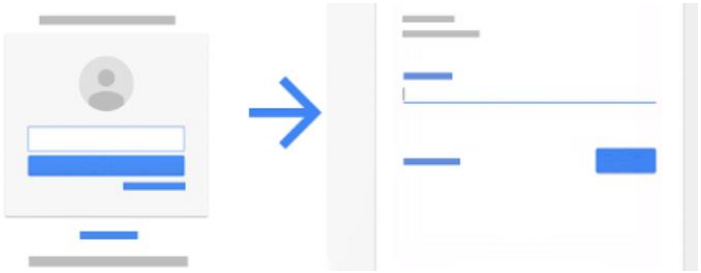


Figure 3.2 B

3.2.2.1 Introduction

The most important functions that will provide users with unique id is through creation of their own account through sign up. And once the sign up is done, the user can sign in back again anytime.

3.2.2.2 Inputs

In case of sign-up, the user will asked questions related to his basic personal details like name, age etc. and his health details medical history which we will store on our server and will be used later by our bot to process inputs.

3.2.2.3 Processing

The submission of the form, storing it in our database in the server will be the processing in case of sign-up. And matching the sign-in content with stored user details will be all that is required for processing.

3.2.2.4 Outputs

The user will now be able to interact with out bot when all this gets done.

3.2.2.5 Error Handling

In case the user fails to sign in due to mismatch, we have the option to reset the password by email or through the mobile number that we earlier stored by just entering the correct username.

Chat Page

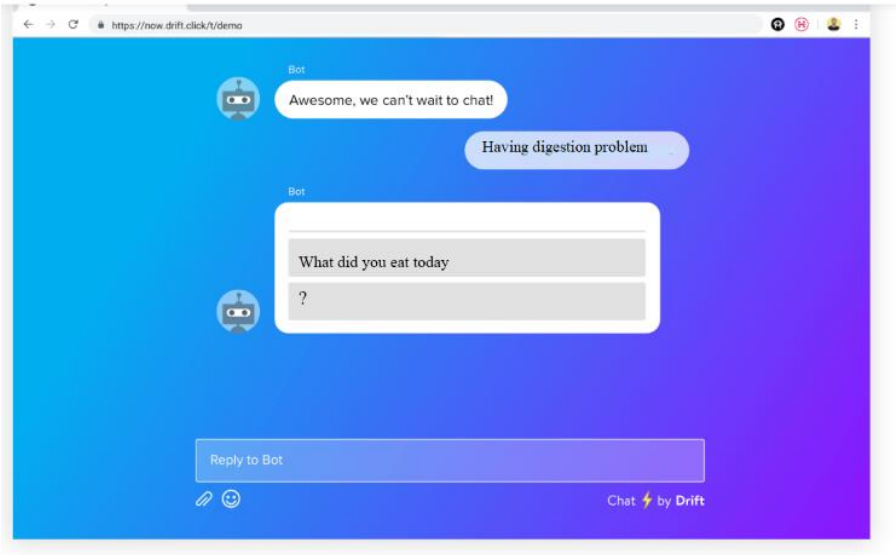


Figure 3.2 C

3.2.3.1 Introduction

Here the user will be able to communicate with our bot.

2.2.1.1 Inputs

The bot will ask various questions to the user to try and learn about the user's problem. The user will need to enter the correct details every time.

2.2.1.2 Processing

All processing required is understanding the user's response by our bot and time required by the NLP engine to predict his response and reply him and at last predict him with the most accurate suggestion to his problem.

2.2.1.3 Outputs

The user will be shown the chat screen with various buttons for him to interact with our bot. There user can either click on the suggestions boxes or choose to manually type his reponse.

3.2.1.5 Error Handling

In case our bot does not understand what the user is trying to do, then the bot will try to give him suggestions for him to choose from. The user can choose any of them or change his response.

In case the of connection problem like connection lost, we will provide the facility for user to continue the chat within 5 minutes if the connection is back again.

3.3 Use Cases

E-Health Care Services Use Cases

The E-Health Care Services has the following sets of use cases:

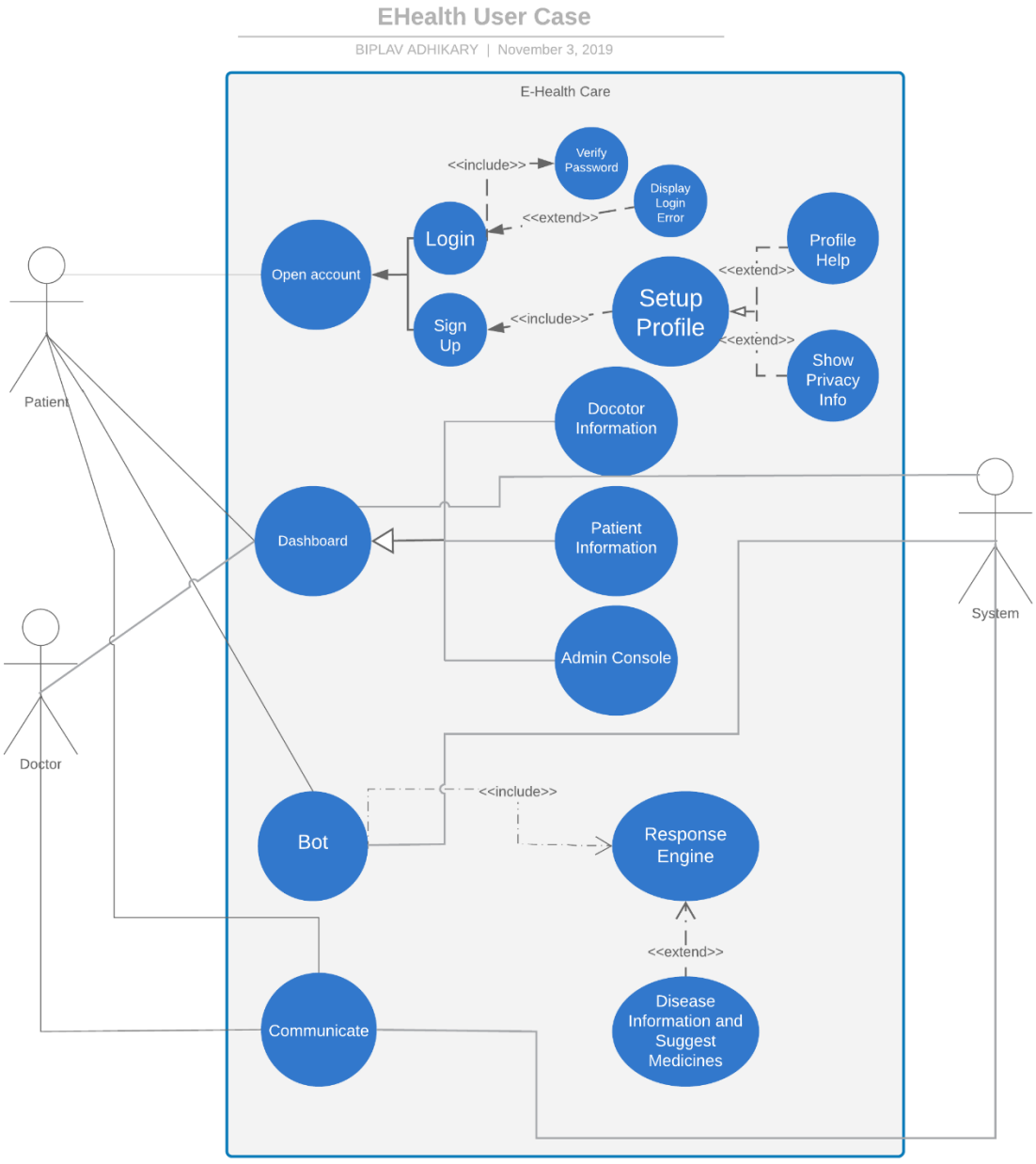


Figure 3.3 – E-Health Care Services Use Cases

Initial Step-By-Step Description

Before this use case can be initiated, the Editor has already accessed the main page of the Article Manager.

1. The Patient/Doctor has to first create an account first. If he does not register an account he/she won't be able to use most of the features in the dashboard.
2. If an account already exists he/she has to log in.
3. The login and sign up cases include multiples other cases.
4. The dashboard gives and view of the option and recent activities of the user.
5. Both doctor and patient will be having their respective dashboards.
6. The user if he is a patient will be able to communicate with the bot via a chat window and the bot will suggest the disease he/she is having and prescribe medicines through the system actor.
7. The patient and the doctor are able to communicate further with the communicate case.

3.4 Classes / Objects

3.4.1 <Class>

Figure 3.4 – E-Health Care Services Class Diagram

3.4.1.1 Attributes

Each class is having their own private, protected and public member variables. There are multiple use of protected attributes in our case so that they can be inherited by the subsequent child classes.

3.4.1.2 Functions

All the methods are made public so as they can be directly accessed by an object of that class.

3.4.2 <Object>

Here's the example of an object of the same class.

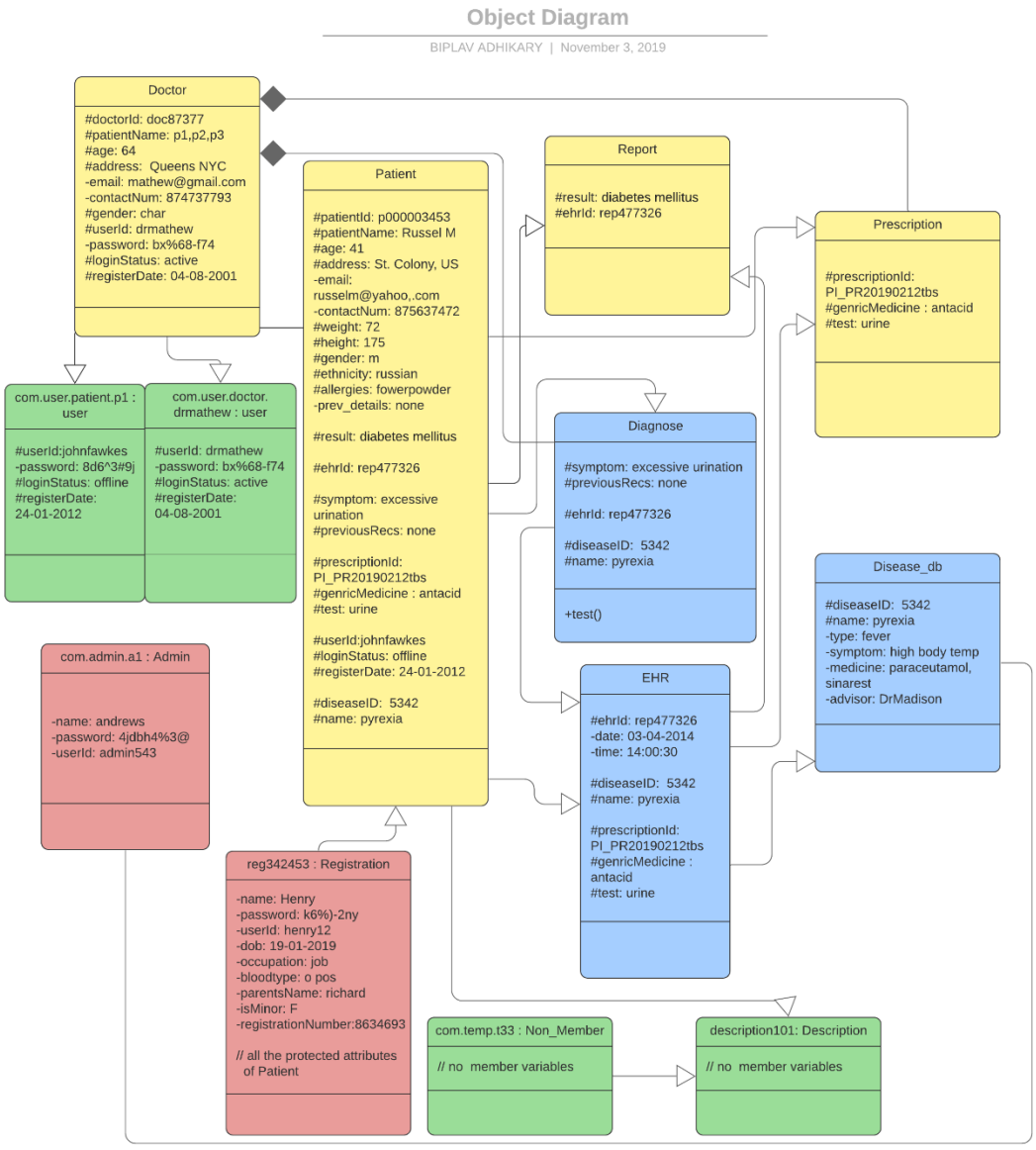


Figure 3.4.2 – Object Diagram

3.5 Non-Functional Requirements

Non-functional requirements may exist for the following attributes. Often these requirements must be achieved at a system-wide level rather than at a unit level. State the requirements in the following sections in measurable terms (e.g., 95% of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, etc).

3.5.1 Performance

The system must be interactive and the delays involved must be less. So, in every action-response of the system, there are no immediate delays. In case of opening windows forms, of popping error messages and saving the settings or sessions there is **delay much below 2 seconds**. In case of opening databases, sorting questions and evaluation there are no delays and the operation is performed in less than 2 seconds for opening, sorting, computing, posting > 95% of the files. Also, when connecting to the server the delay is based editing on the distance of the 2 systems and the configuration between them so there is high probability that there will be or not a successful connection in less than 20 seconds for sake of good communication.

3.5.2 Reliability

As the system provide the right tools for discussion, problem solving it must be made sure that the system is reliable in its operations and for securing the sensitive details.

3.5.3 Availability

If the internet service gets disrupted while sending information to the server, the information can be sent again for verification.

3.5.4 Security

The main security concern is for users account hence proper login mechanism should be used to avoid hacking. The tablet id registration is way to spam check for increasing the security. Hence, security is provided from unwanted use of recognition software.

Data Transfer

- The system shall use secure sockets in all transactions that include any confidential customer information.
- The system shall automatically log out all customers after a period of inactivity.
- The system shall confirm all transactions with the customer's web browser.
- The system shall not leave any cookies on the customer's computer containing the user's password.
- The system shall not leave any cookies on the customer's computer containing any of the user's confidential information.

Data Storage

- The customer's web browser shall never display a customer's password. It shall always be echoed with special characters representing typed characters.
- The customer's web browser shall never display a customer's credit card number after retrieving from the database. It shall always be shown with just the last 4 digits of the credit card number.
- The system's back-end servers shall never display a customer's password. The customer's password may be reset but never shown.
- The system's back-end servers shall only be accessible to authenticated administrators.
- The system's back-end databases shall be encrypted.

3.5.5 Maintainability

After conducting multiple through tests, we have come up with our initial version. We will be releasing more versions later on. If in case any problem is faced, there in an option to mail the query/feedback to us. We have also provided a feedback windows that you will be getting below the dashboard. Once we reach our target user base, we will also be coming up more features to our services.

3.5.6 Portability

Since our application is both web-based and mobile application based, we have ensured that it will run in 99% of all today's devices

- Android devices with version 4.0+
- IOS 7+
- Windows (XP, Vista,7,8,10) and most popular Linux Distros

3.6 Inverse Requirements

There are no useful inverse requirements for this product.

3.7 Design Constraints

Standard Development Tools

The system shall be built using a standard web page development tool that conforms to either IBM's CUA standards or Microsoft's GUI standards.

Web Based Product

- There are no memory requirements
- The computers must be equipped with web browsers such as Internet explorer.
- The product must be stored in such a way that allows the client easy access to it.
- Response time for loading the product should take no longer than five minutes.
- A general knowledge of basic computer skills is required to use the product

3.8 Logical Database Requirements

- The system shall provide storage of all databases on redundant computers with automatic switchover.
- The system shall provide for replication of databases to off-site storage locations.
- The system shall provide RAID V Disk Stripping on all database storage disks.

3.9 Other Requirements

No other requirements.

4. Analysis Models

This section contains all the necessary Data Flow Diagrams and UML diagrams to explain the necessary working of our model.

4.1 Data Flow Diagram (DFD)

4.1.1 Level 0

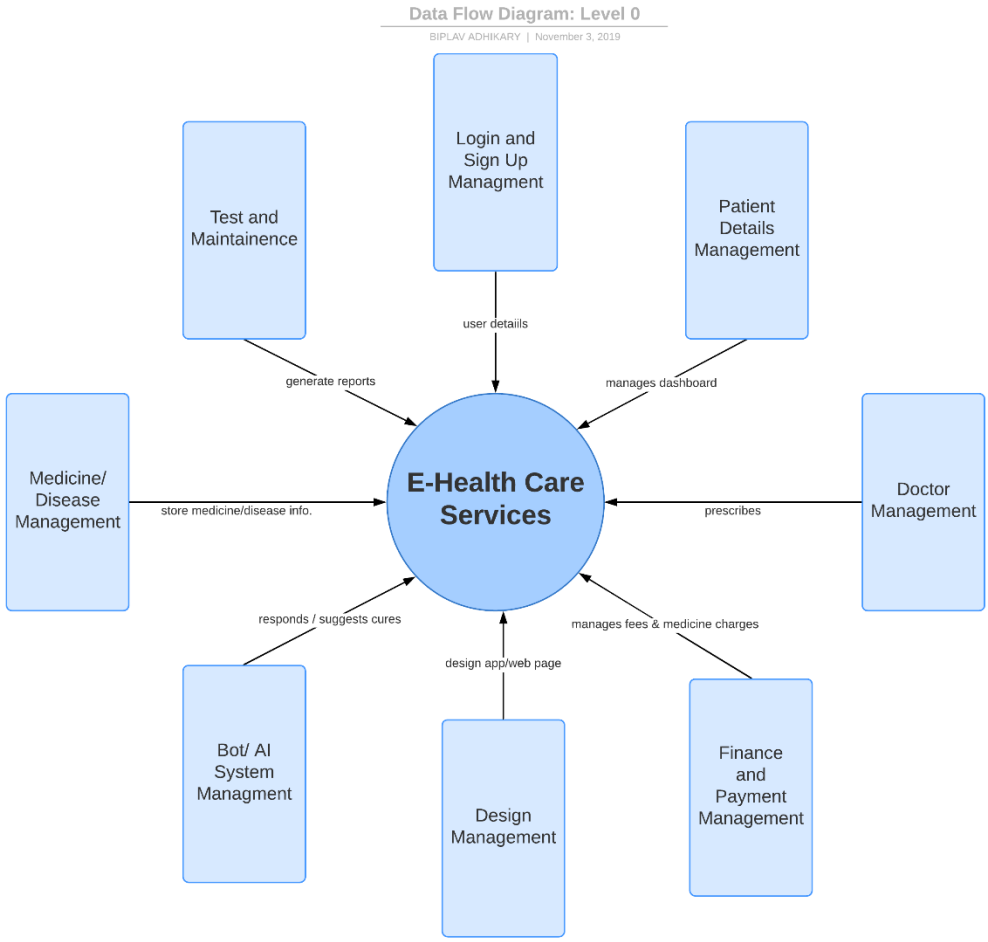


Figure 4.1.1 – E-Health Care Services Level 0 Activity

4.1.2 Level 1

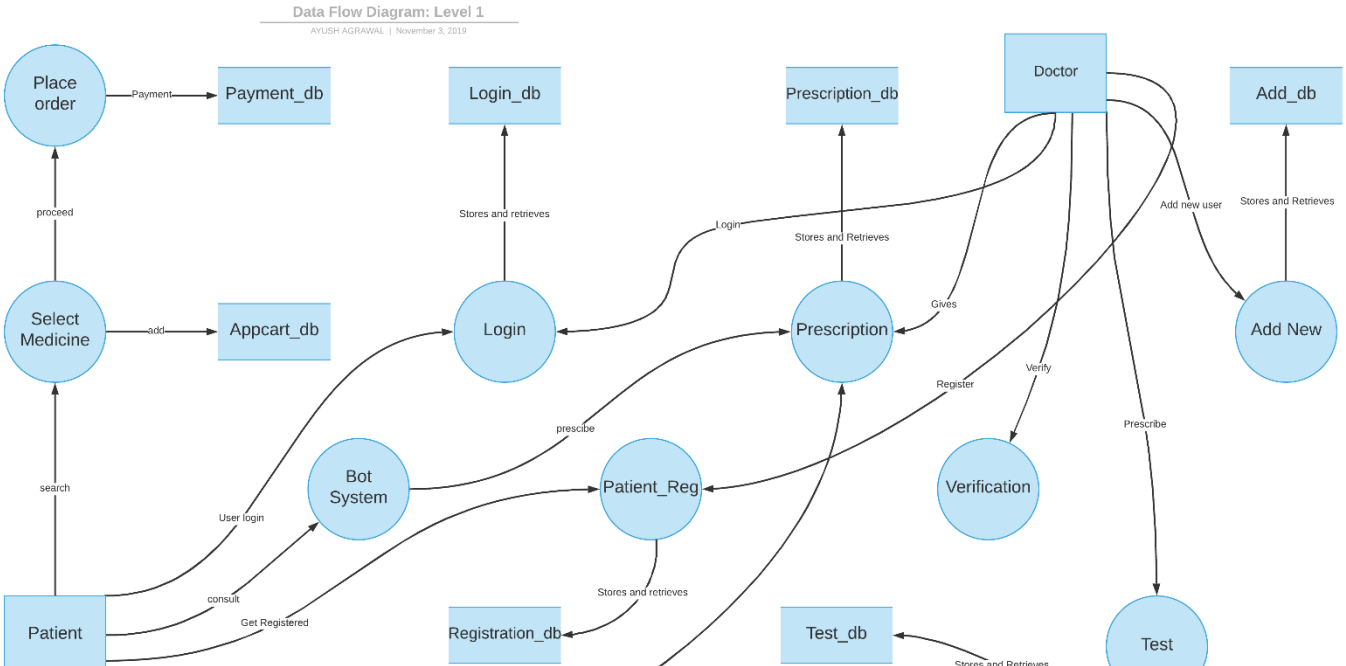


Figure 4.1.2 – E-Health Care Services Level 1 Activity

4.1.3 Level 2

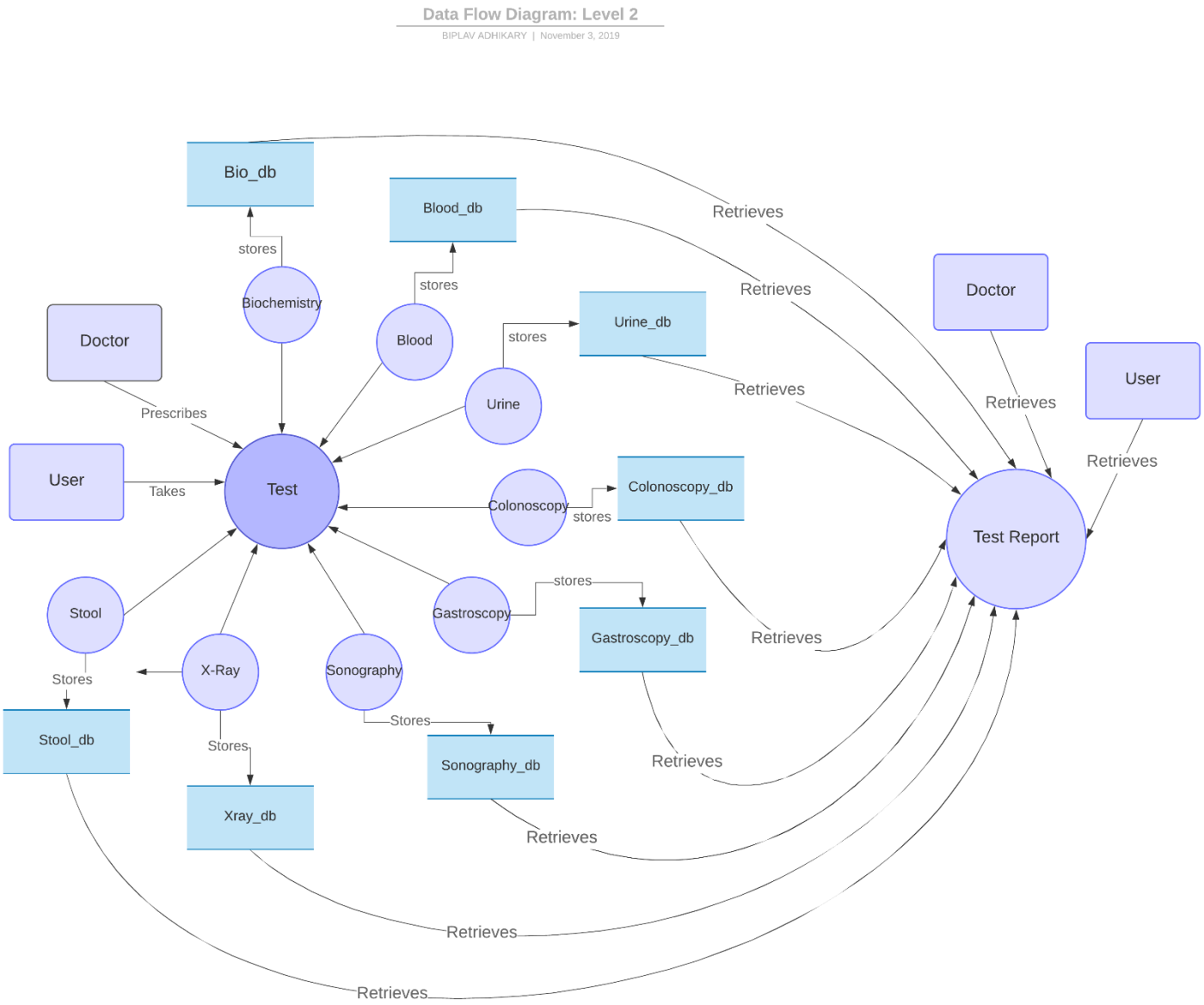


Figure 4.1.3 – E-Health Care Services Level 2 Activity

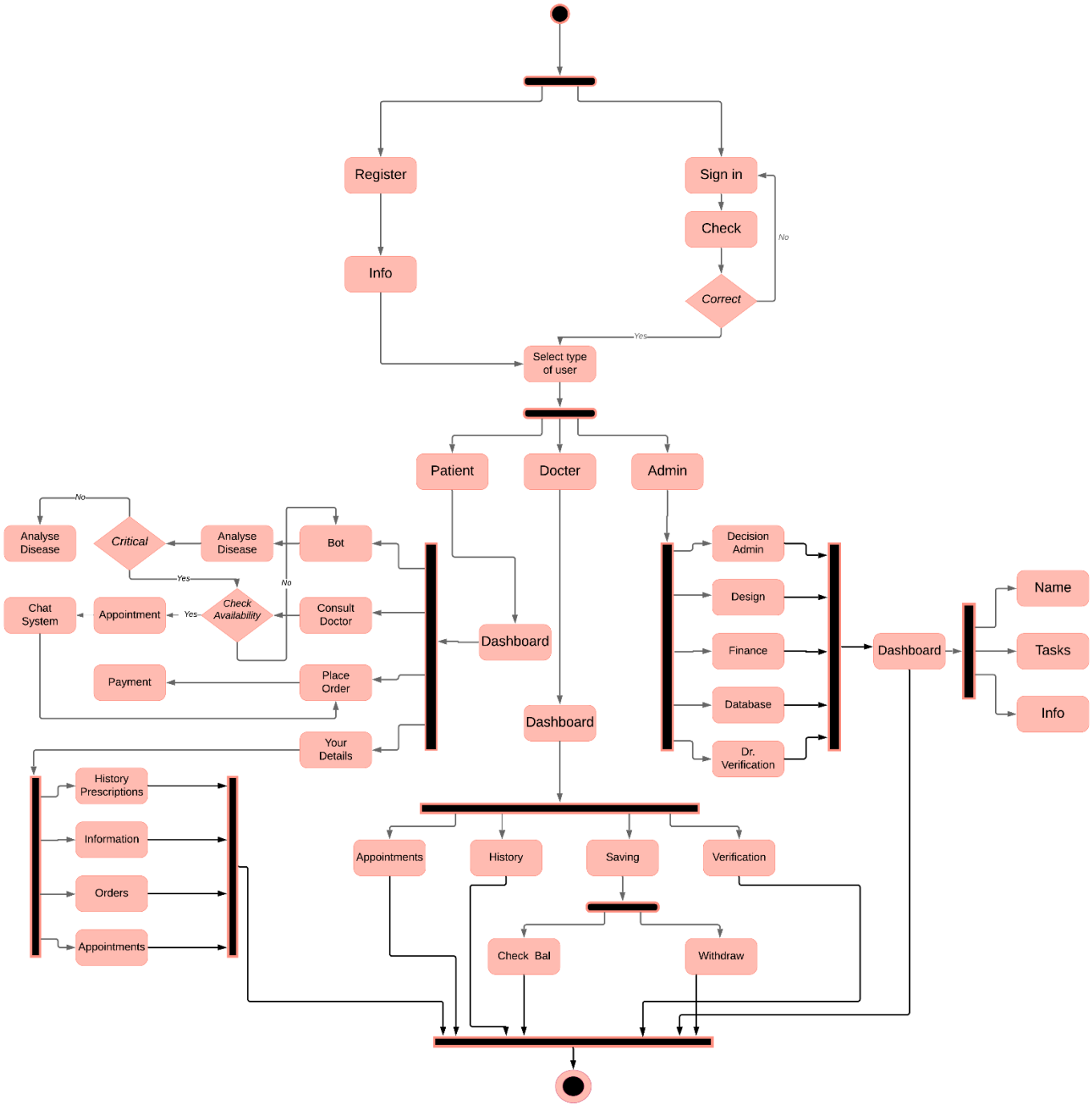

Register
Sign in
Check
No
Info
Correct
Yes
Select type of user
Patient
Docter
Admin
No
Analyse Disease
Critical
Analyse Disease
Bot
Yes
No
Chat System
Appointment
Yes
Check Availability
Consult Doctor
Payment
Place Order
Your Details
Dashboard
Decision Admin
Design
Finance
Database
Dr. Verification
Dashboard
Name
Tasks
Info
Dashboard
History Prescriptions
Information
Orders
Appointments
Appointments
History
Saving
Verification
Check Bal
Withdraw

Figure 4.2 – E-Health Care Services Activity Diagram

4.3 Sequence Diagrams

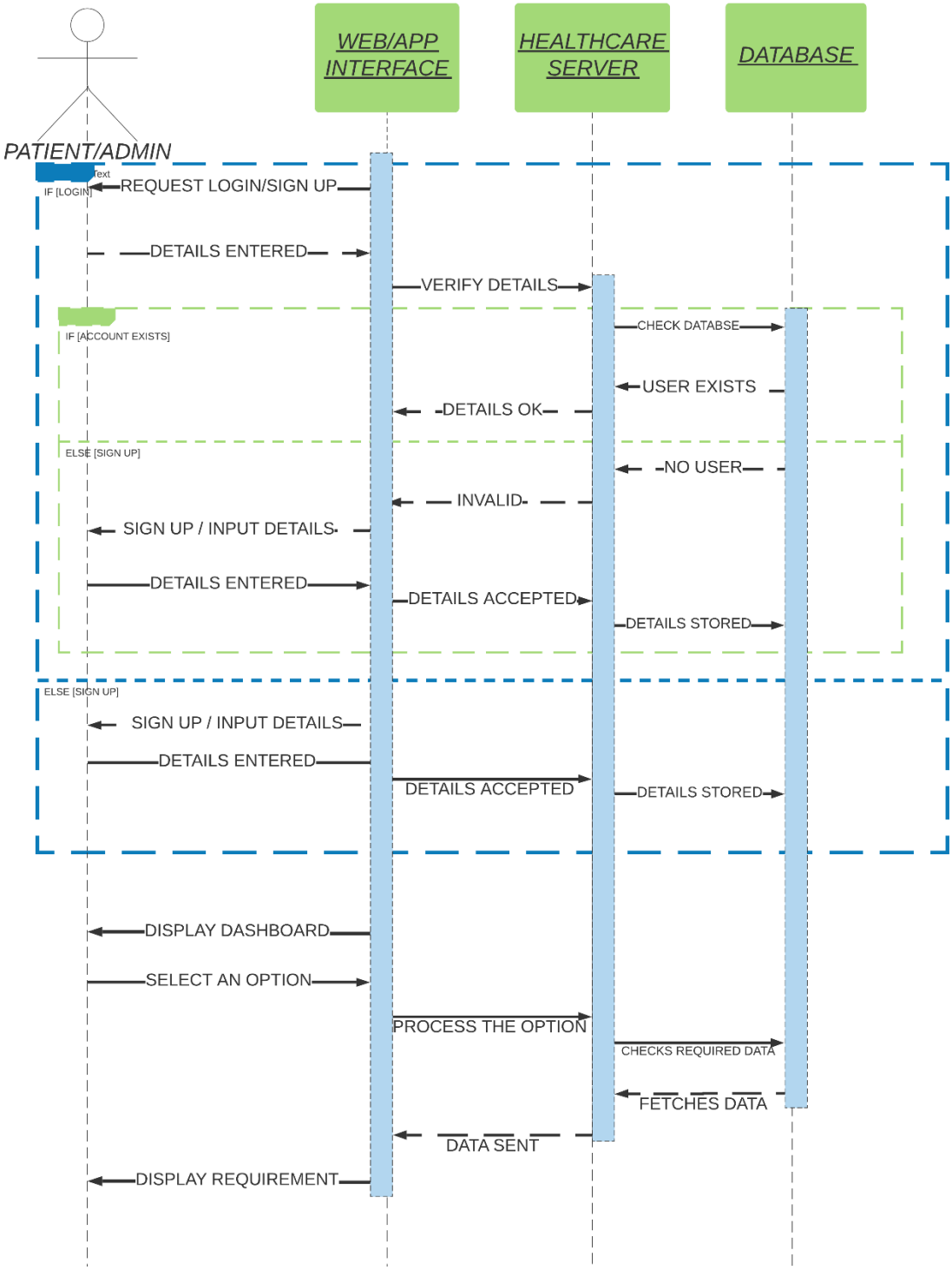


Figure 4.3.1 – Sequence Diagram Patient/Admin

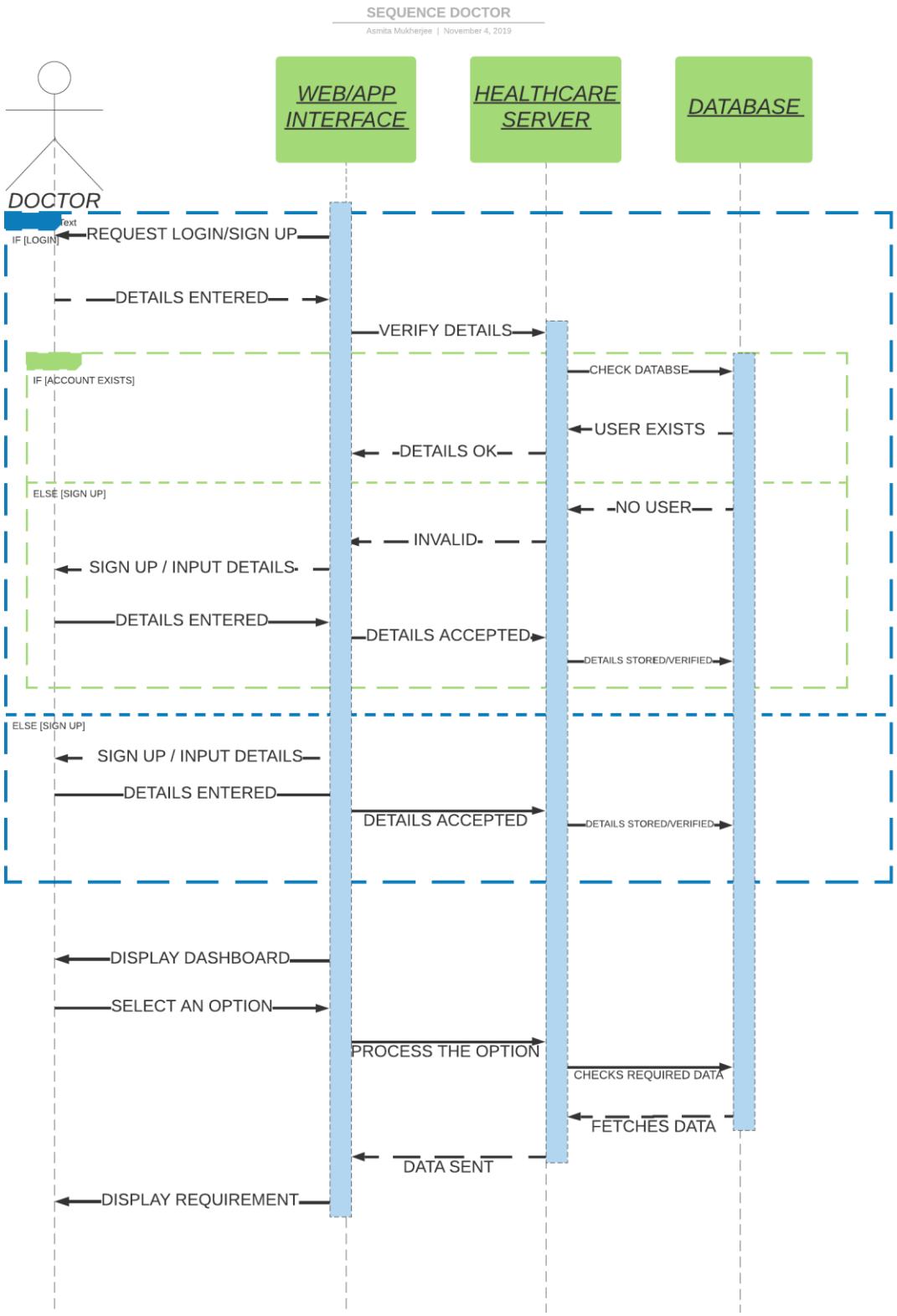


Figure 4.3.2 – Sequence Diagram Doctor

4.4 State-Transition Diagrams (STD)

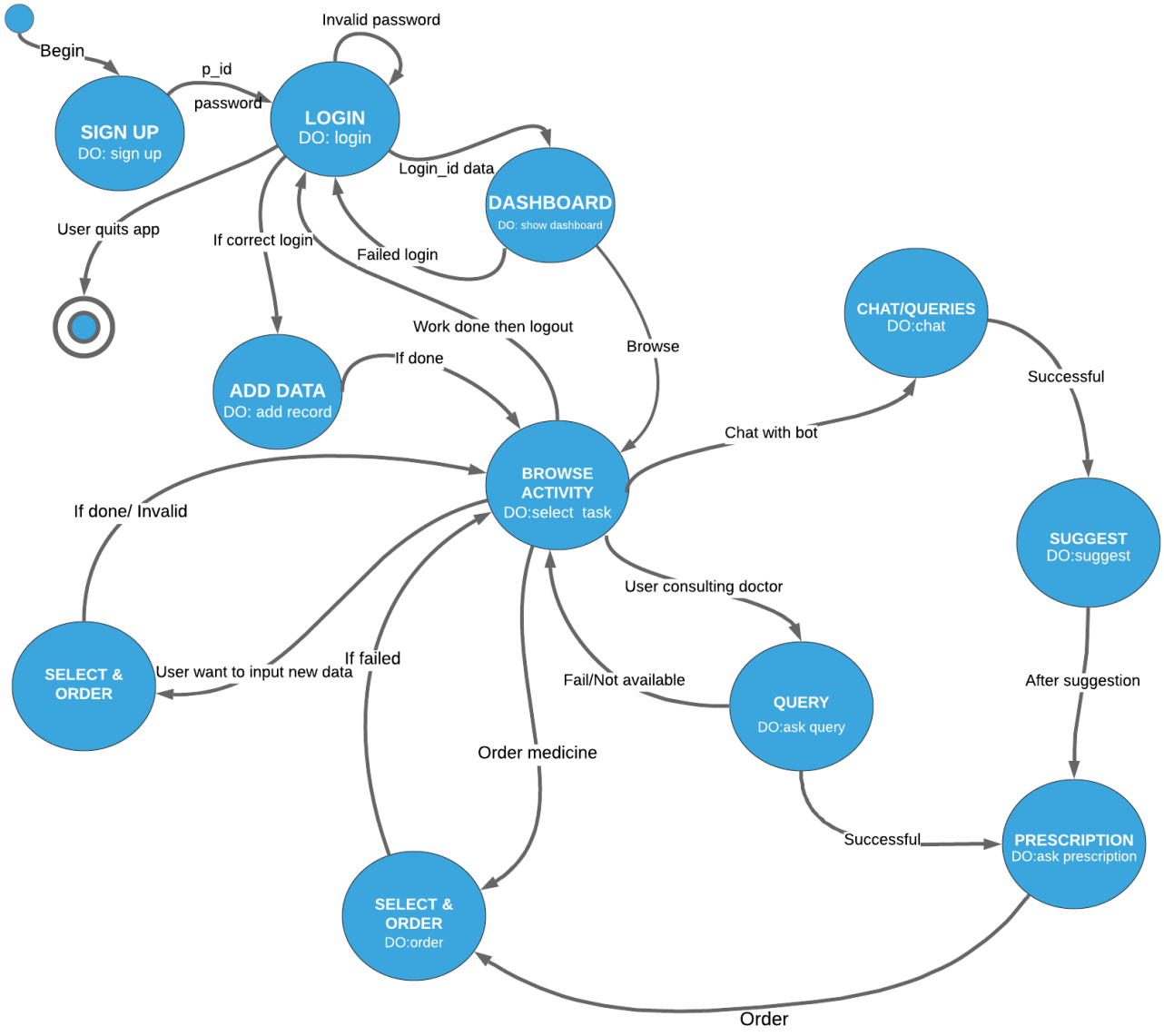


Figure 4.4 – State-Transition Diagram

5. Change Management Process

1. Request for change is made

Changes and issues are captured through Project Team meetings, a working group of external stakeholders including other agencies, meetings of the Project Board, outcomes of approval stages and through monitoring by the Project Team and Project Sponsor.

2. Register and assess the change

Change requests are captured on a central change log that enables monitoring of change levels by the Change Manager.

3. Review and submit RFC to Change Board

If the change is valid the Project Manager assesses the impact of the change with the project team and submits the change via a Request for Change Form to the Change Manager. It is then submitted to the Change Board who assess the change and approve or reject it.

4. Change Board accept or reject the change

The Change Board includes the client and senior managers who have a strategic overview of the contract. They assess whether the change is within program tolerances and are able to approve most changes.

5. Update plans & implement the change

Once the change is approved the change log is updated and stakeholders are informed. The Project Manager then works with the project team to plan the implementation of the change; amending the project and stage plans, sourcing any additional equipment or personnel, updating the configuration items and completing the work package(s).

A. Appendices

A.1 Appendix 1

Team Meeting #1

Date: August 15, 2019
Place: Library CS Building

Summary:

- Project Methodology
- Requirements Discussion
- Scope of the Project
- Target Audience
- Constraints
- User Interface Design
- Hardware Constraints
- Functional Requirements

Team Meeting #2

Date: November 1, 2019
Place: Library CS Building

Summary:

- Discussion on UML Design Process and Software
- Use Case Discussion & Design
- Activity Diagram Discussion and Design
- Sequence Diagram Discussion and Design

Team Meeting #3

Date: November 3, 2019
Place: Reading Room CS Building

Summary:

- Data Flow Design
- State Diagram Discussion & Design
- Completion of other formal documentation

A.2 Appendix 2

Index