High Level Design (HLD)

Deep EHR: Chronic Disease Prediction Using Medical Notes

Revision Number: 1.0
Last date of revision: 01/06/2021

Ravi Chaurasia
Khushali Shah

Document Version Control

Date Issued	Version	Description	Author
01th June 2021	1.0	First Version of Complete HLD	Ravi Chaurasia

Contents

Abstract

With high increase in vehicles on the road, problem of traffic congestion and accidents has increased substantially. To overcome these problems, the in-depth analysis of causes such as number of traffic rules followed, is required. Therefore, continuous monitoring of traffic on highways and huge roads is mandatory. An Automatic Traffic Control System can prove to be a solution to above mentioned problems.

Automatic Traffic Counter Control can also help in drawing inferences from the recorded data.

1 Introduction

1.1 Why this High-Level Design Document?

The purpose of this High-Level Design (HLD) Document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding, and can be used as a reference manual for how the modules interact at a high level.

The HLD will:

- Present all of the design aspects and define them in detail
- Describe the user interface being implemented
- Describe the hardware and software interfaces
- Describe the performance requirements
- Include design features and the architecture of the project
- List and describe the non-functional attributes like:
    - Security
    - Reliability
    - Maintainability
    - Portability
    - Reusability
    - Application compatibility
    - Resource utilization
    - Serviceability

1.2 Scope

The HLD documentation presents the structure of the system, such as the database architecture, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non-technical to mildly-technical terms which should be understandable to the administrators of the system.

1.3 Definitions

Term	*Description*
EHR	Electronic Health Record
Database	Collection of all the information monitored by this system
IDE	Integrated Development Environment
AWS	Amazon Web Services

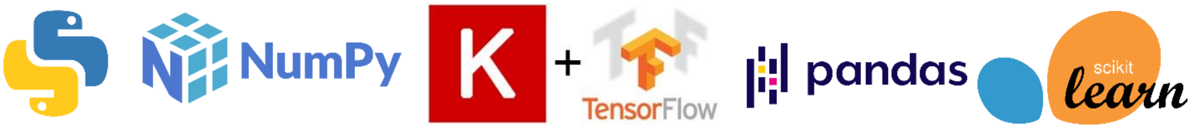
2 General Description

2.1 Product Perspective

The Deep EHR system is a web application which will detect the diseases at earliest for better disease management, improved interventions, and more efficient health-care resource allocation using previous EHR records available. SQL is used to retrieve, insert, delete, and update the database. Here the system store each and every data given by user or received in request to the MySQL/MongoDB database.

2.2 Tools used

Python programming language and frameworks such as Numpy, Pandas, Scikit-learn, TensorFlow, Keras are used to build the whole model.



- PyCharm is used as IDE.
- For visualization of the plots, Matplotlib, Seaborn and Plotly are used.
- AWS is used for deployment of the model.
- Tableau/Power BI is used for dashboard creation.
- MySQL/MongoDB is used to retrieve, insert, delete, and update the database.
- Front end development is done using HTML/CSS
- Python Django is used for backend development.
- GitHub is used as version control system.

2.3 Constraints

The Deep HER system must be user friendly, as automated as possible and users should not be required to know any of the workings.

2.4 Assumptions

The main objective of the project is to predict the onset of disease for new cases based on the information in the EHR by using Machine Learning and Deep Learning techniques. It is also assumed that all aspects of this project have the ability to work together in the way the designer is expecting.

3 Design Details

3.1 Functional Architecture

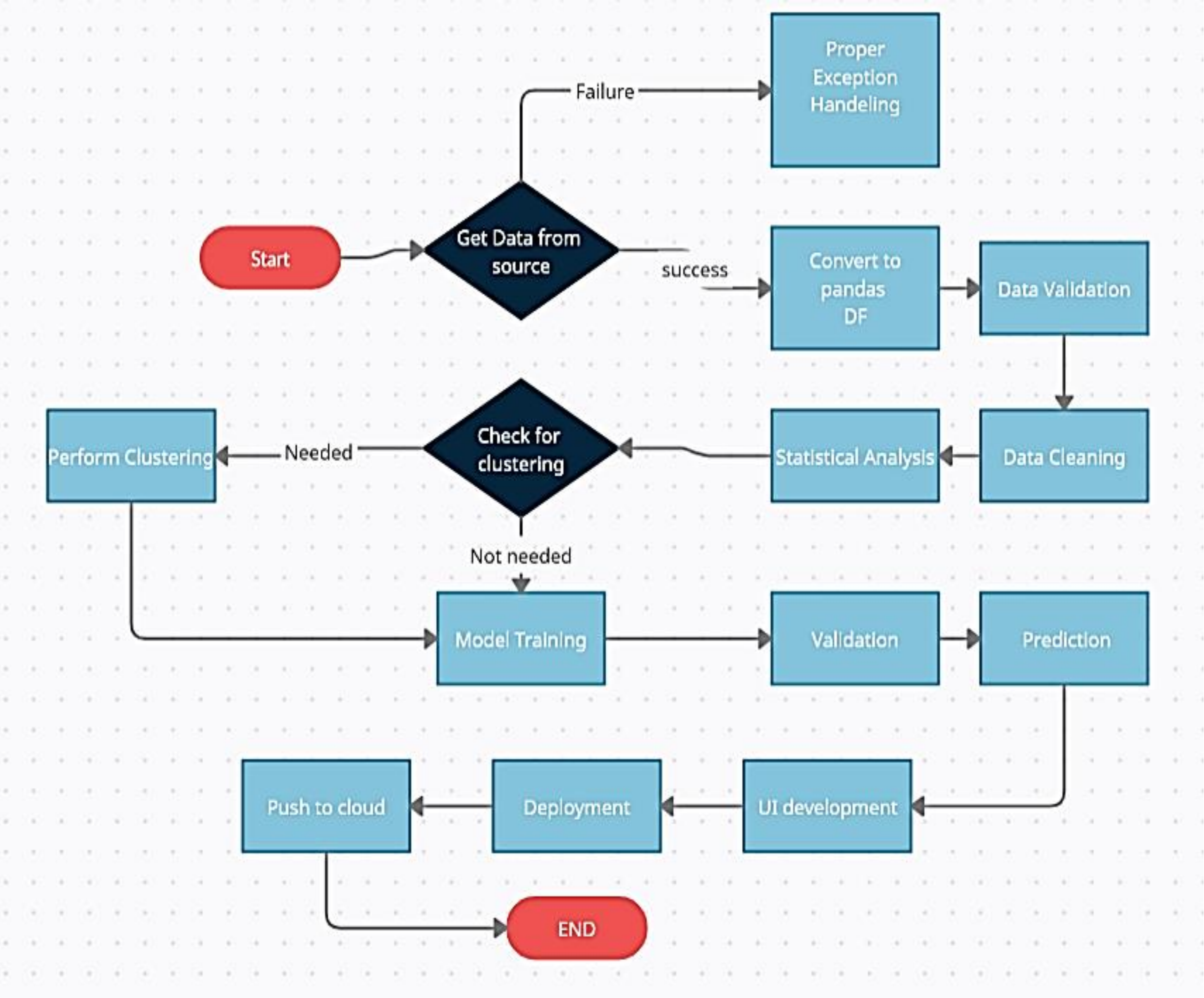


Figure 1: Functional Architecture of Deep EHR

3.2 Database Design

System needs to store every request into database and you need to store in such a way that if you want to retrain a model it should be easy to retrain model with new data as well.

Initial Step-By-Step Description:

1. The User choose the disease.
2. The User gives required information.
3. The system store each and every data given by user or received in request to the database.

3.3 Web Application Architecture

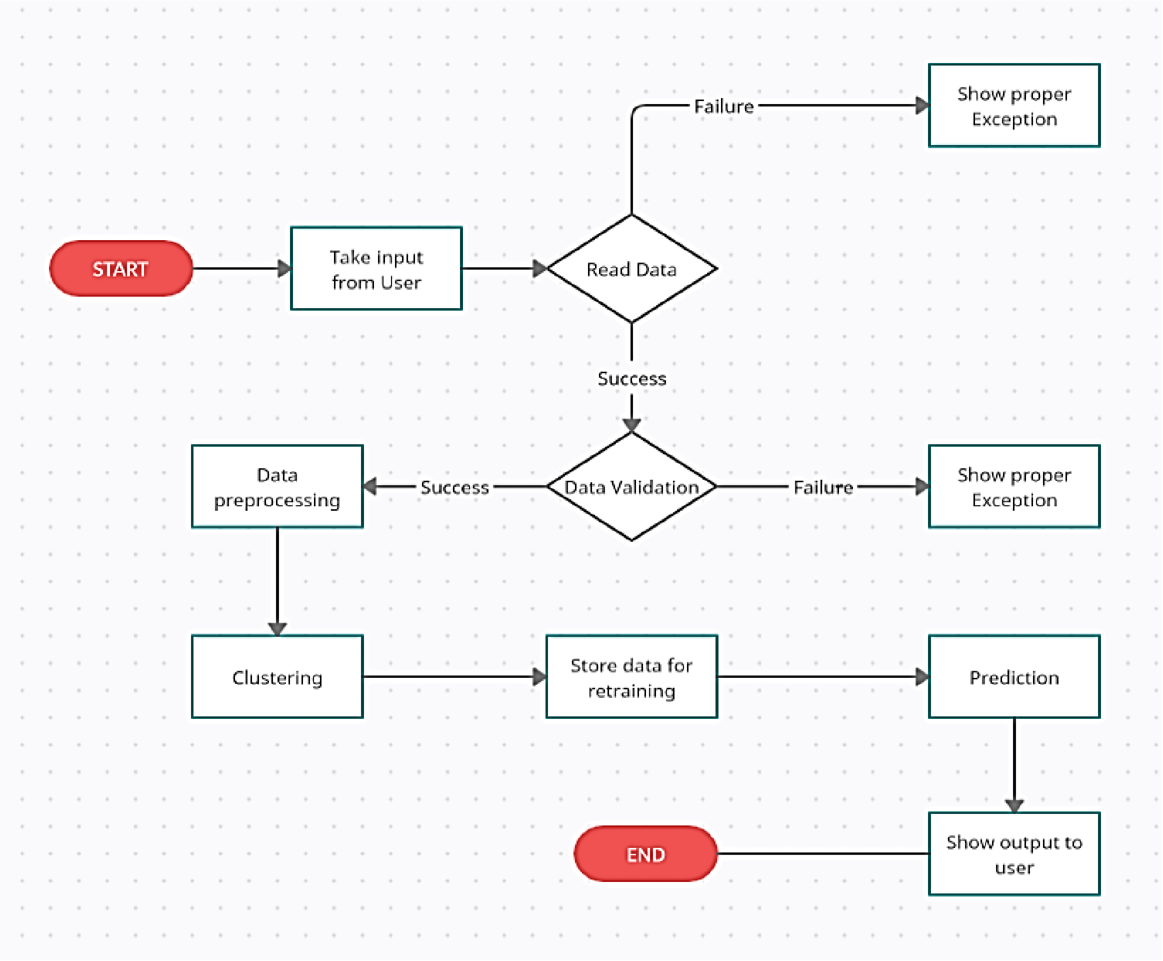


Figure 2: Web application architecture

The user interface is a very simple plain layout with little graphics. It will display information very clearly for the user and will primarily output information to the user through HTML pages. Also, all the details for the user input will be provided.

3.4 Event log

The system should log every event so that the user will know what process is running internally.

Initial Step-By-Step Description:

1. The System identifies at what step logging required
2. The System should be able to log each and every system flow.
3. Developer can choose logging method. You can choose database logging/ File logging as well.
4. System should not hang even after using so many loggings. Logging just because we can easily debug issues so logging is mandatory to do.

3.5 Error Handling

Should errors be encountered, an explanation will be displayed as to what went wrong? An error will be defined as anything that falls outside the normal and intended usage.

3.6 Help

The ‘Help’ option is provided in web application for guiding users regarding maximum range of valid inputs required for predicting a particular disease.

3.7 Performance

Deep EHR system is used for predicting the disease onset, it should be as accurate as possible. So that it will not mislead the user. Also, model retraining is very important to improve the performance.

3.8 Security

Since the Deep EHR system consists of patient’s demographic data, the information should be secured.

3.9 Reusability

The code written and the components used should have the ability to be reused with no problems.

3.10 Application compatibility

The different components for this project will be using Python as an interface between them. Each component will have its own task to perform, and it is the job of the Python to ensure proper transfer of information.

3.11 Resource utilization

When any task is performed, it will likely use all the processing power available until that function is finished.

3.12 Deployment

Microsoft Azure
Google Cloud
amazon web services™

4 Dashboards

Dashboards will be implemented to display and indicate certain KPIs and relevant indicators for the disease.

As and when, the system starts to capture the historical/periodic data for a user, the dashboards will be included to display charts over time with progress on various indicators or factors

4.1 KPIs (Key Performance Indicators)

Key indicators displaying a summary of the person’s health and physique as compared to a normal individual with similar basic traits

1. Time and workload reduction using the EHR model.
2. Comparison of accuracy of model prediction and doctor’s prediction.
3. Number of times a patient visits the hospital.
4. Time between symptom onset and detection of illness/visit to hospital.
5. Immunity of patient (based on previous illnesses).
6. Vaccines the patient has taken.
7. Length of stays in hospital.

5 References

1. ‘Deep EHR: Chronic Disease Prediction using Medical Notes’ by Jingshu Liu, Zachariah Zhang and Narges Razavian