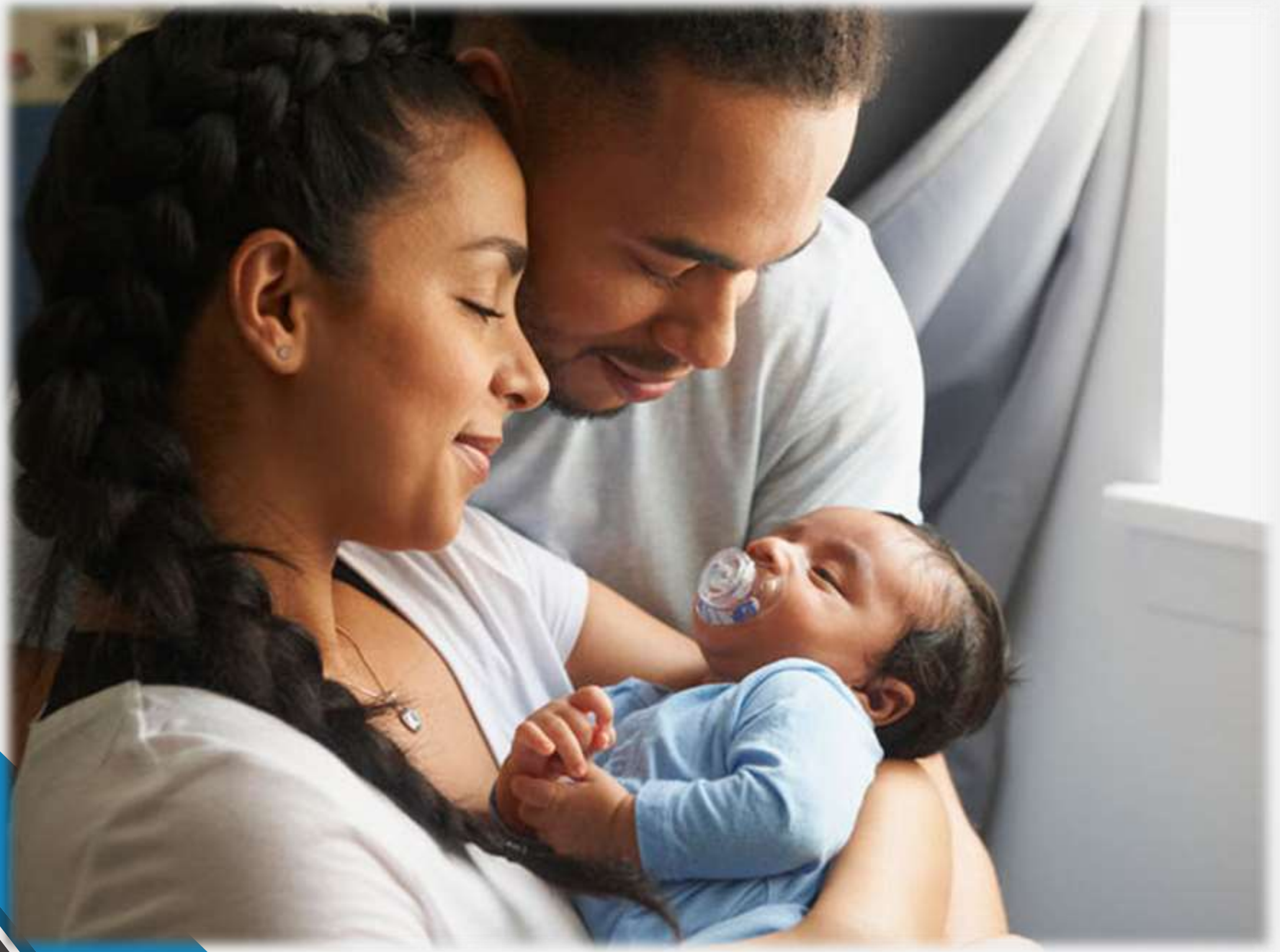
A MECHANISM TO KEEP THE IMMUNIZATION HIGH OF NEWBORNS WITH CRUCIAL TECHNIQUES VIA REMOTE CONSULTATION

(TMP-23-103)



Progress after proposal presentation

- Changes of the components
- Information gathering
  - ✓ Field visits
  - ✓ Interviews
  - ✓ Research papers, tutorials
- Implementations (basic models)

11/6/2023 2

Changes of the components

Component Name	Before Proposal presentation	After Proposal presentation
Decentralized patient information system	• Permission-based access	• Permission-based access control to ensure that only authorized parties can access the data. • Improving Novellty
Chatbot Application	• Medical knowledge base prediction for diseases • Suggestions of remedies • System notifications for the vaccinations • Chat history	• Identify skin infections using image processing • As the novel feature, uploading an image option was introduced by the panel

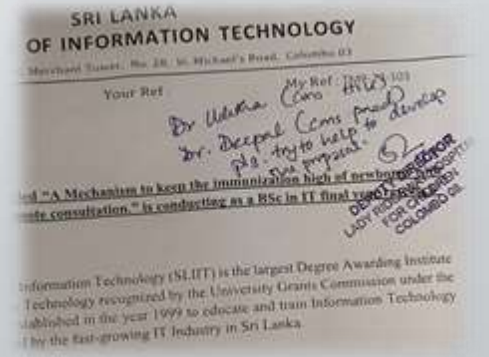
Changes of the components

Component Name	Before Proposal presentation	After Proposal presentation
Nutrition Level and malnutrition detection	• Prediction of nutritional level using image processing and calculations. • Predict the nutritional level using height, weight, and head circumference	• Predict the growth level without using images but behavioral aspects • Component name changed to "**Growth Level Prediction**"
Infant Sickness Identification through Video Processing	• Remote consultation between the pediatrician and the parents with infants. • Predict the sickness and suggest the solutions and remedies.	• Video processing participate only parents with the infants • Only identify the sickness and abnormal behavior and notify it through the system

Information Gathering

Interviews

- **Deputy Director - Lady Ridgeway Hospital**
- **Dr. Uditha - IT Consultant (LRH)**
- **Dr. Deepal - Pediatrition (LRH)**
- **The pediatric clinic was visited at LRH**
- **Visited the Castle Hospital and handed over the necessary documentation for approval to do field visits**



IT20015244 | Wijesinghe R.M.U.S

Decentralized Patient Registration.

Software Engineering

Background

Decentralized Patient (Newborns) Registration (DPR)

- After the proposal was submitted, the panel asked for an outstanding novelty to be highlighted. Information was gathered from experts and doctors to gather information on how to develop the registered patient, doctor and hospital. And if you want to view the patient records, you need to request the public key from the patient. They said to collect new novelty like this.

- I would implement a permission-based access control mechanism to ensure that only authorized parties can access the data.

- I implement the system to allow parents and health care providers to manage a child's medical and personal information in an efficient and accessible manner. For example, if a child needs to see a specialist, the specialist can quickly access the child's medical history and provide the necessary treatment.

- By encapsulating all of these features into a single component, the DPR component could be easily integrated into a larger decentralized healthcare system.

System Workflow

- **In this application is powered by IPFS, where patients' medical records are stored on the distributed file system, not owned by any centralized entity. A patient can access his or her records by interacting with a smart contract on the Ethereum blockchain, forming a digital identity of the patient on the decentralized network.**

- **The client first connects with MetaMask, and uses smart contract to mint a patient or doctor block, registered by the wallet address.**

- **The client can upload a record file to IPFS, which address is linked to a patient block in ETH chain. The client can get all record addressed stored in a patient block from smart contract, and get a record file by its address from IPFS.**

- **The health provider can search for a patient's records using the address, and upload a new record for the patient. The patient can also view his or her records.**

- **A patient or a doctor can access the patient's records by interacting with a smart contract on the Ethereum blockchain.**

Research Gap

Research paper title	year	Used Ethereum Blockchain for Decentralized system	Used interacting with the Ethereum network using Web3.js	Accessible only to authorized parties, such as parents or healthcare providers.
01-Centralised versus Decentralized Management of Patients' Medical Records	2009	No(only research paper)	No	Yes
02-Decentralized Patient-Centric Report and Medical Image Management System Based on Blockchain Technology and the Inter-Planetary File System.	2022	Yes (Only focuses in lab test , not implement and)	Yes	Yes
03-Decentralized Electronic Medical Records.	2019	Yes(Only focuses on the adults used in hyperledger)	-	Yes
04-Digital and Decentralized Management of Patient Data in Healthcare Using Blockchain Implementations.	2021	Yes(Only focuses on the adults not implement,only research paper)	Yes	Yes

Research Question

- The existing electronic health records management systems suffer from security and privacy vulnerabilities, as patient data is often stored in centralized databases prone to unauthorized access and data breaches. Our research project aims to address this critical issue by leveraging the decentralized and immutable nature of blockchain technology.
- How will the system ensure the privacy and security of the data collected?
- How will the system ensure that authorized parties have access to the necessary information?
- How will the system interface with existing healthcare systems and databases?

Research Novelty

- Permission-based access control to ensure that only authorized parties can access the data.
- Our solution incorporates advanced cryptographic techniques such as zero-knowledge proofs and homomorphic encryption to enable secure and privacy-preserving data sharing among authorized healthcare providers while keeping patient identities and sensitive information confidential.
- The use of blockchain technology ensures that the data is immutable and tamper-proof, providing a high level of security.
- We introduce a novel access control mechanism using smart contracts, allowing patients to grant granular access permissions to healthcare providers on a need-to-know basis. The access control rules are enforced autonomously, ensuring data integrity and preventing unauthorized data access.

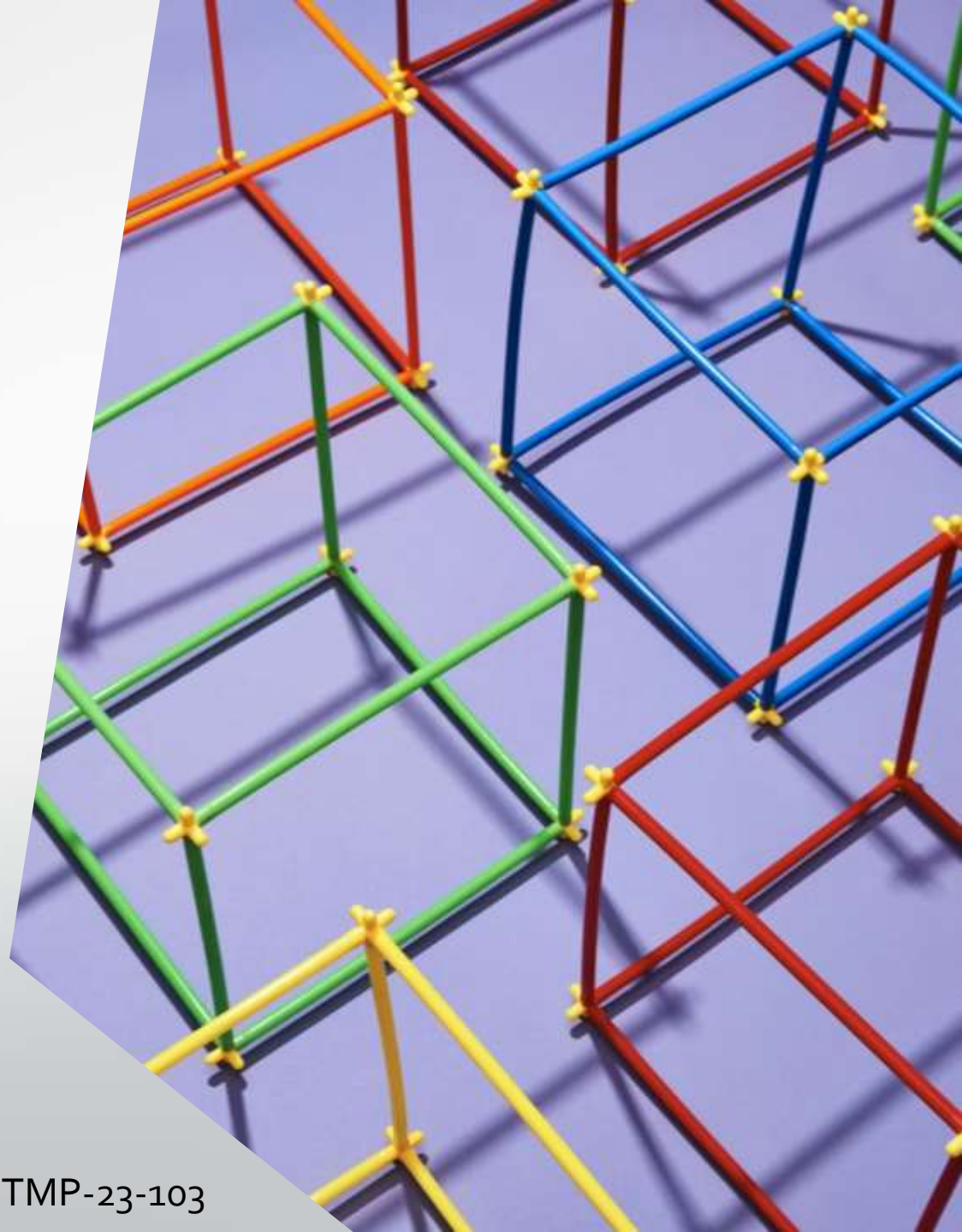
Specific Objectives & sub-Objectives

Specific Objective

The goal of using a decentralized system for the registration of newborns and parents' details is to provide a secure, transparent, and efficient way to manage critical information related to newborns and their parents.

Sub Objectives

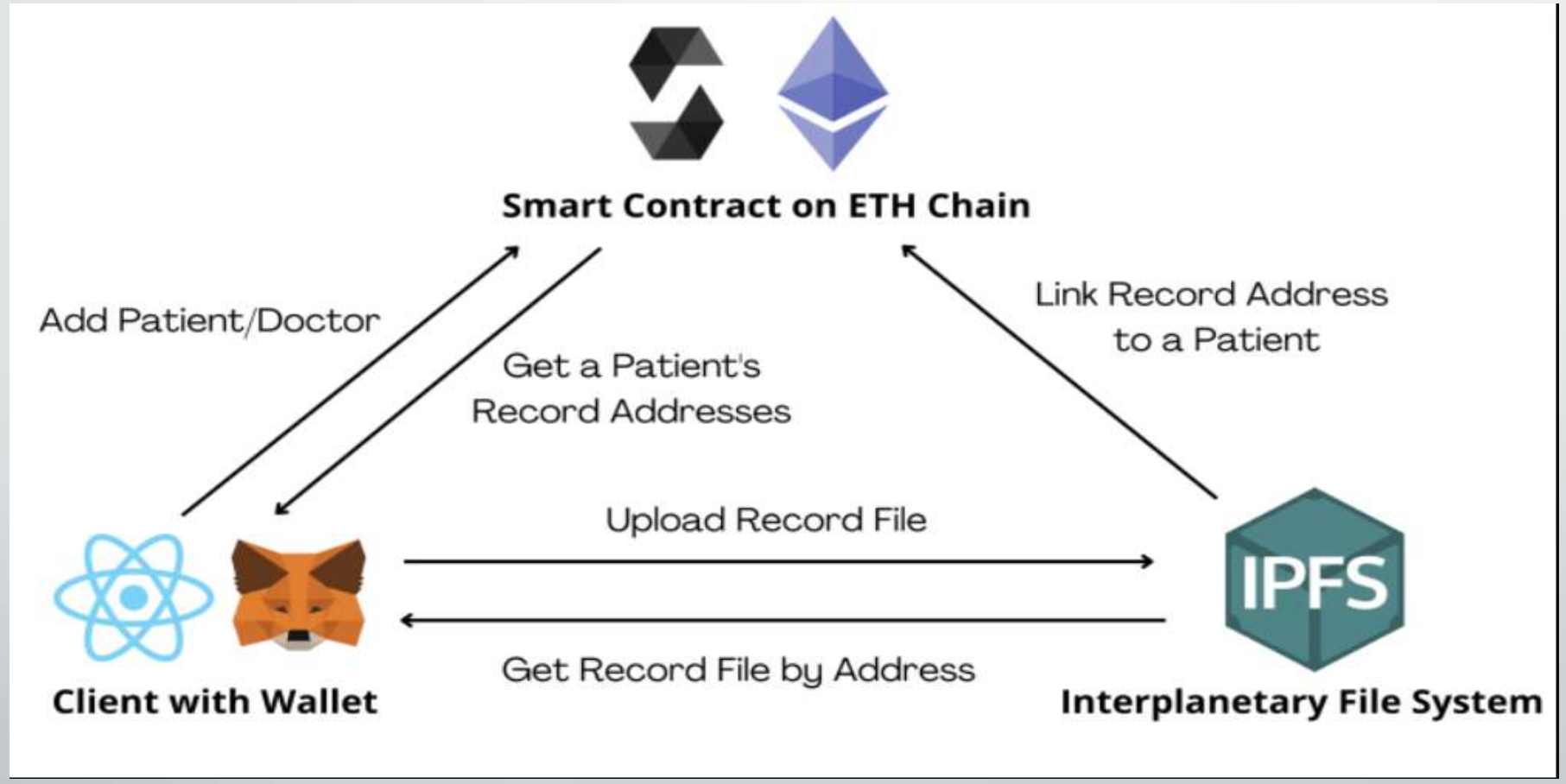
- By leveraging decentralized technologies such as blockchain or decentralized databases, this system can enhance security and privacy, prevent data tampering, and improve data accessibility.
- The smart contract stores the newborns' and parents' details on the Ethereum blockchain, ensuring that the data is secure and immutable.
- This system also allows parents and healthcare providers to manage the child's medical and personal information in a transparent and secure way, while providing access to relevant parties as needed.



Methodology

Component overview Diagram

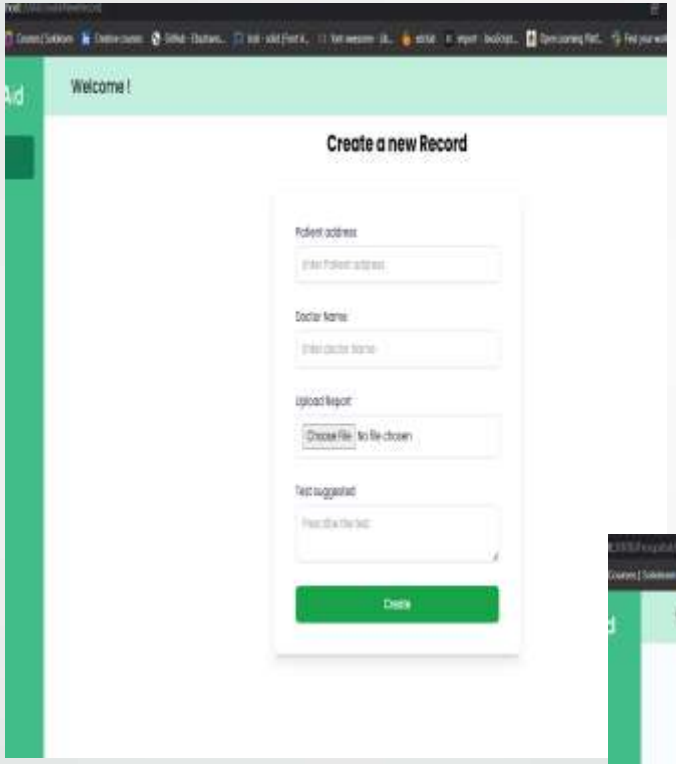
Component Overview Diagram

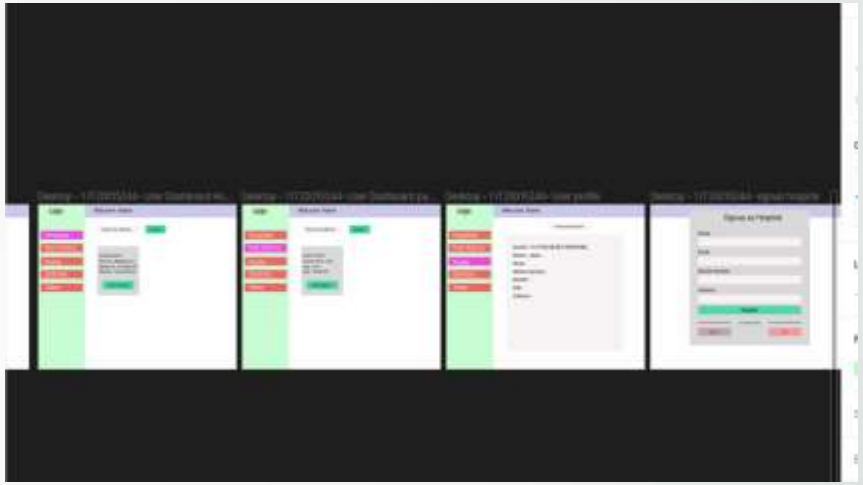


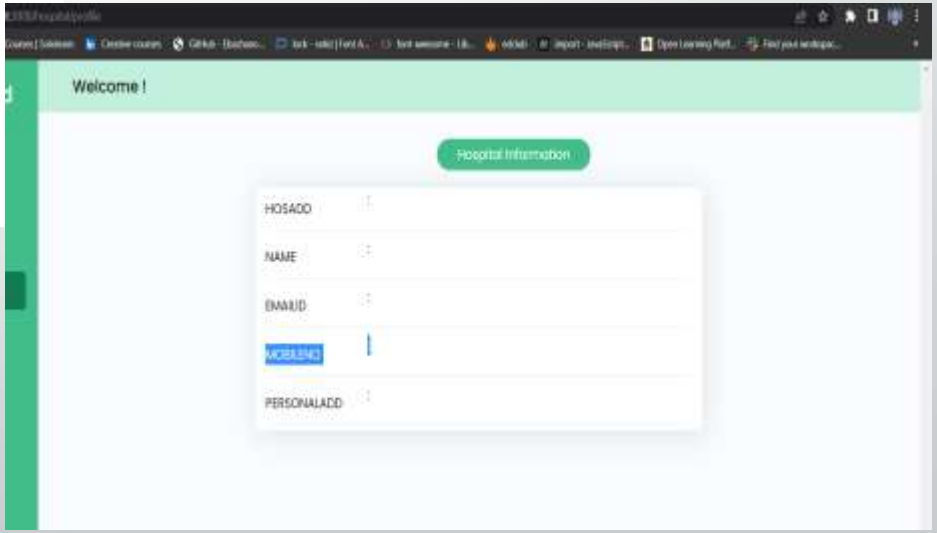
Implementation

Developing Frontend and Backend

- Frontend with: HTML, CSS, ReactJs, Tailwind.
- Backend with : Blockchain
- Storage : Web3.Storage, InterPlanetary File System(IPFS)

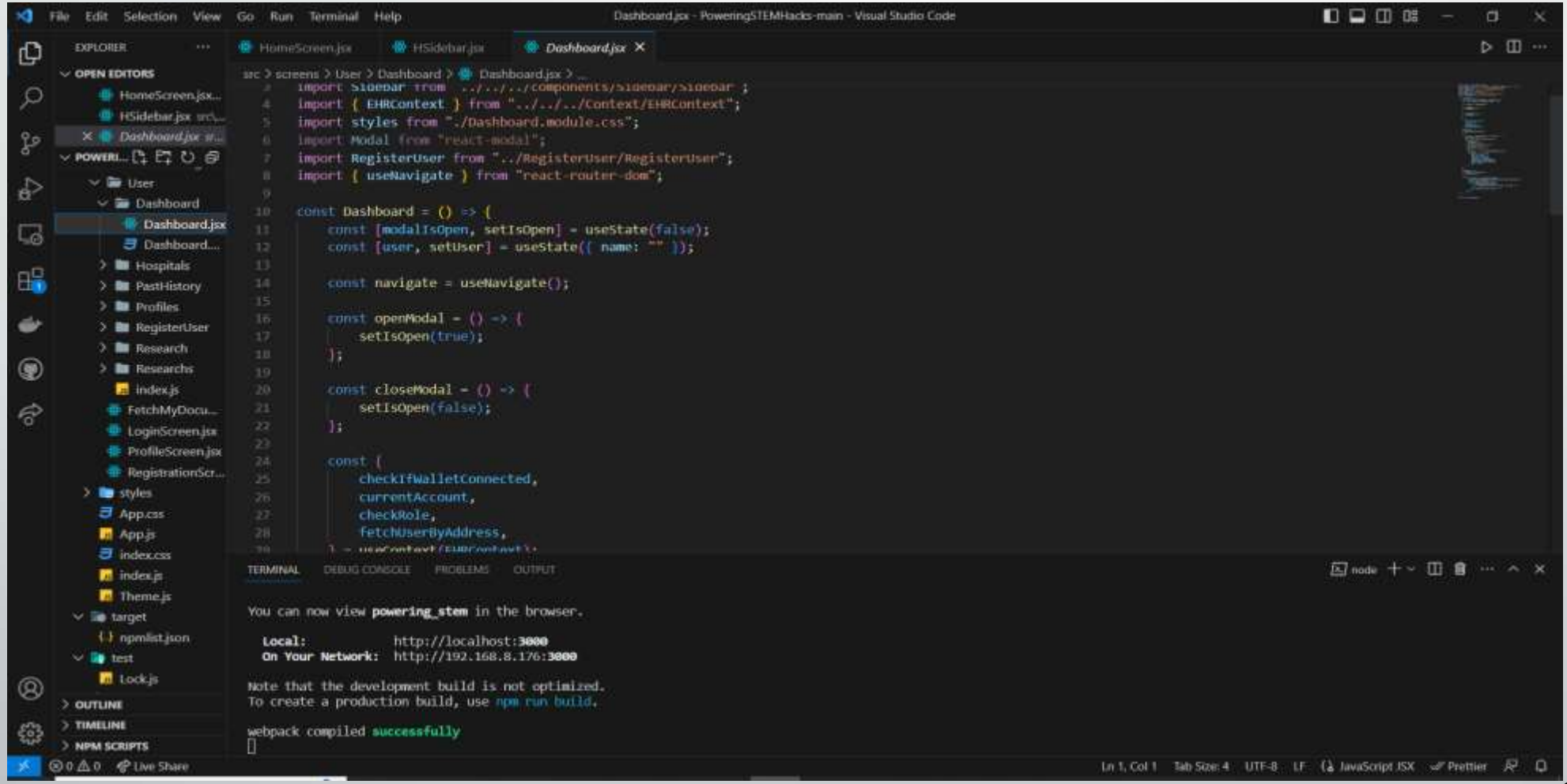






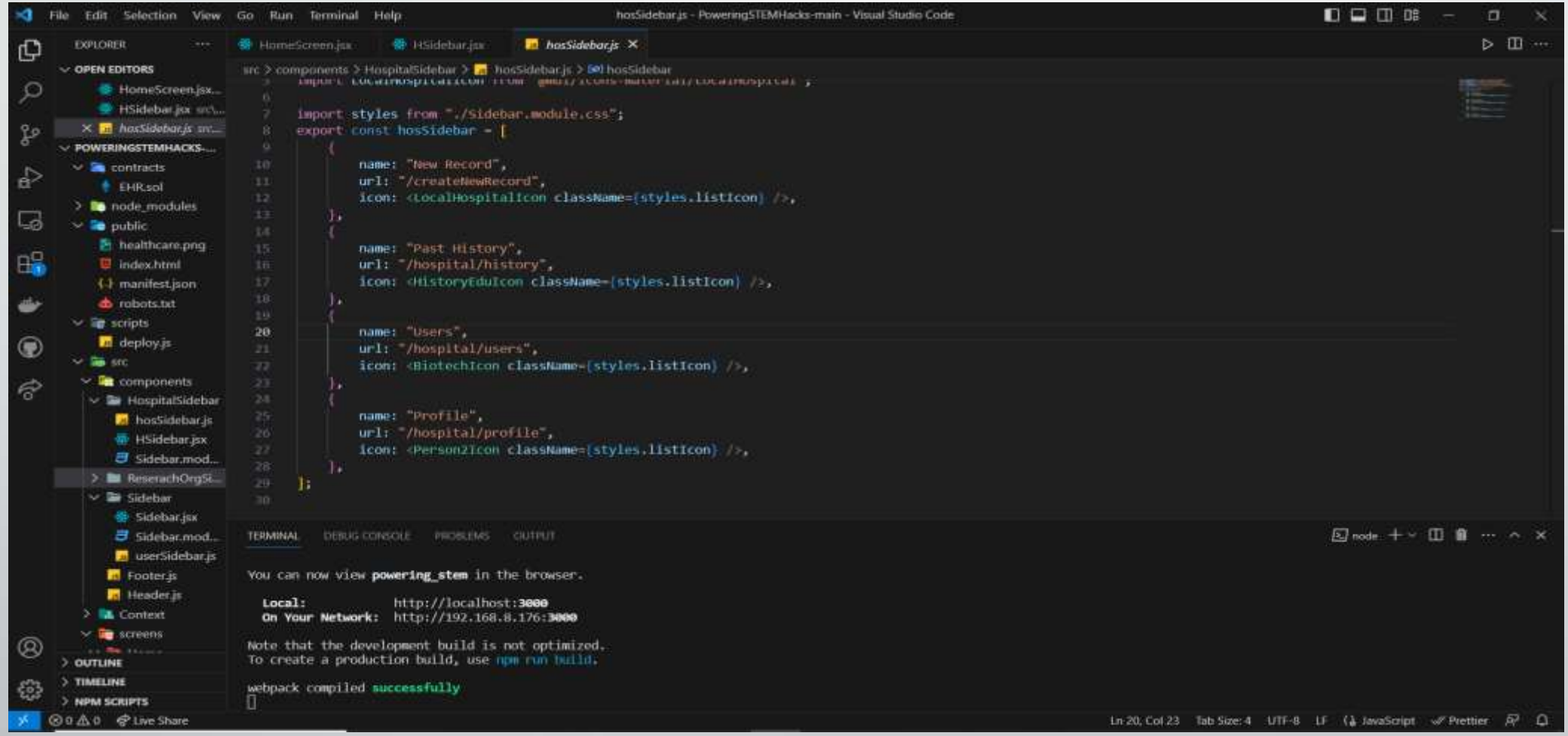
Implementation

Used Dashboard



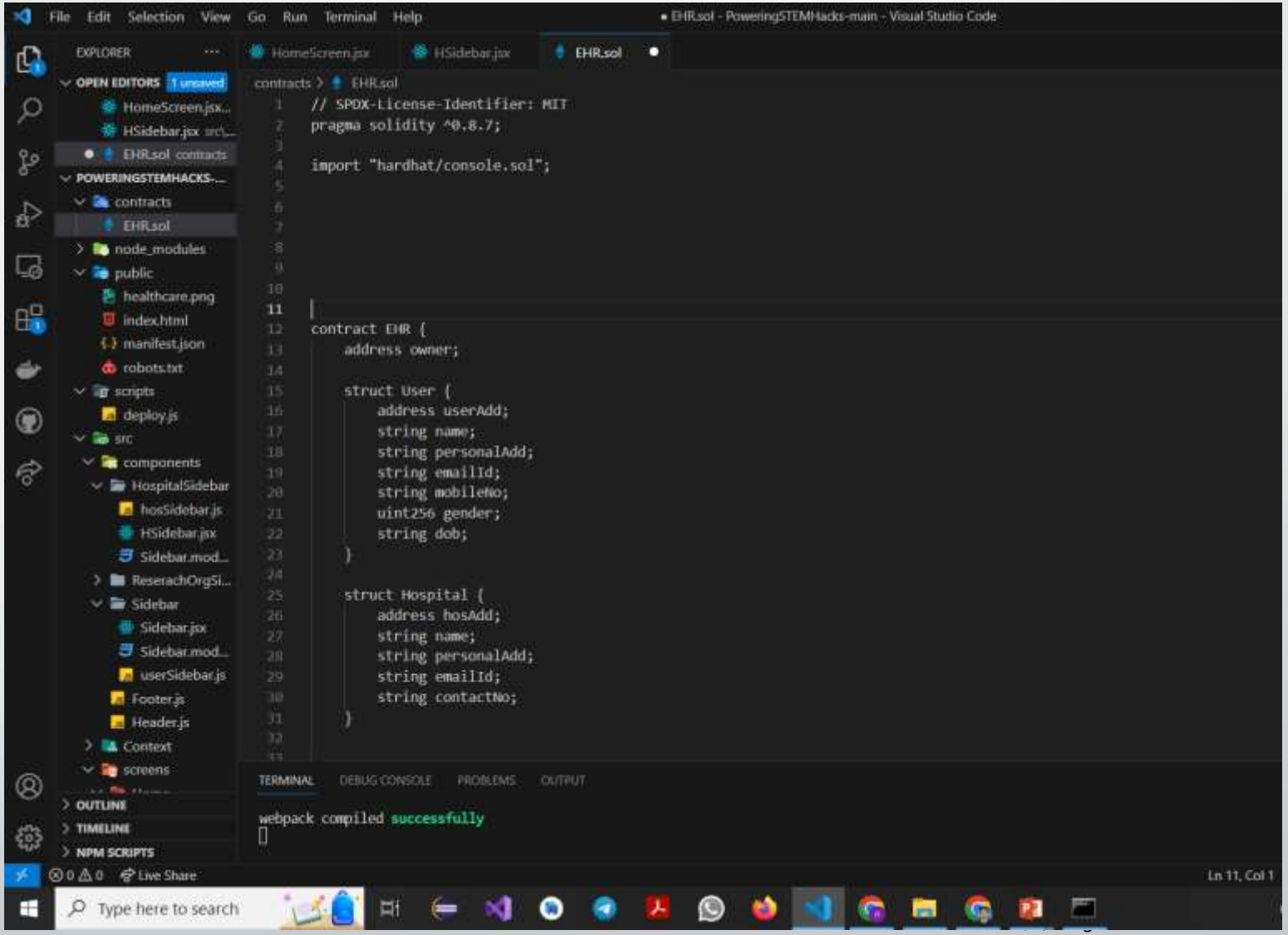
Implementation

Hospital Dashboard



Implementation

Smart Conract code



Technology Involved

Frontend

- React

Backend

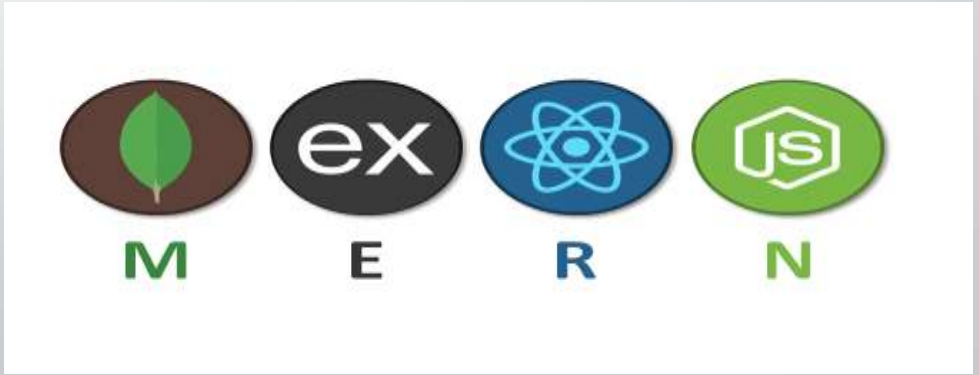
- Node.js and Express
- Python, Springboots

Libraries

- Web3.js Ethereum network
- Mongoose and Axios

Database

- MongoDB, MySql
- Ethereum Based Blockchain



Project Requirements

Functional requirements

- Distributed storage: The system must be able to store data in a distributed manner, with multiple copies of data spread across nodes in the network.
- Decentralized consensus: The system must use a consensus algorithm to agree on the state of the network without relying on a central authority. This could be achieved using a blockchain, DAG, or other consensus mechanisms.
- P2P communication: The system must be able to facilitate communication between nodes without relying on a centralized server. This could be achieved using a peer-to-peer (P2P) network.
- User authentication and access control: The system must provide secure authentication and access control mechanisms to ensure that only authorized users can access and modify data

Non-Functional requirements

Accuracy, Reliability, Performance, Usability , Accessibility,

References

1. Secure decentralized electronic health records sharing system based on blockchains: https://www.frontiersin.org/articles/10.3389/fbloc.2021.732112/full
2. Secure decentralized electronic health records sharing system based on blockchains: https://www.sciencedirect.com/science/article/pii/S1319157821001051.
3. C C Darshan Thimmaiah, D. S. (2019). Decentralized Electronic Medical Records. IJRAR, 7.
4. Catherine Quantin, G. C. (2009). Centralised versus Decentralised Management of Patients' Medical Records. researchgate, 6.
5. Jihui Shi 1, S. K. (2022). A Novel Block Chain Method for Urban Digitization. International Journal of Environmental, 19.
6. Syed Agha Hassnain Mohsan, A. R. (2021). Decentralized Patient-Centric Report and Medical Image. International Journal of Environmental, 18.
7. Zhang P, White J, Schmidt DC, Lenz G and Rosenbloom (2018), FHIR Chain: Applying Blockchain to Securely and Scalably Share, Computational and Structural Biotechnology Journal, 16: 267-278
8. Crypt Bytes Tech (2017), Medical Chain - A blockchain for electronic health records. https:// medium.com/crypt-bytes-tech/medicalchain-a-blockchain-for-electronic-health-recordseef181ed14c2.
9. Ekblaw A, Azaria A, Halamka JD and Lippman A (2016), A Case Study for Blockchain in Healthcare: “MedRec” prototype for electronic health records and medical research data. MIT Media Lab

IT20147778| Jayawickrama N.D.D

Health Informatics Chatbot Application

Software Engineering

Component Discussion

- **Introduction**
- The proposed component is to assist the parents of babies who are having a hard time managing their time.
- The system enables parents to ask questions about their infants' behaviors, any medicine, related remedies, or any other assumption or methods of how exactly to react to an emergency situation.
- By the image processing technique, the chatbot can identify the skin infections of the kid by analyzing a clear uploaded photo.
- Chat bot will also notify the vaccination days according to the data records it gets from the decentralized system.
- From this specific component, the main target is to introduce an assistant that is available beyond the time, helps to manage the time, to reduce the unnecessary cost of doctor appointments, and assist the parents in their problematic situations as a trained health care agent.
- **User Workflow**
- Parent approach to the chatbot provided the problematic situation they have faced.
- The chatbot system takes the input as data, asks relevant questions to understand the situation, and comes to a final prediction.
- Then the suggestions will be displayed as an answer to the problem. This suggestion can be advice, a prediction of a disease, a history record, a remedy, a medicine, or a contact detail of a relevant doctor.

Component Discussion

- **Benefits for Parents**
- Time and cost Management: This system will reduce the unnecessary amount of time and the cost for the travel to hospital.
- Early Detection: By capturing and analyzing the parents' data, the system can predict the sicknesses and suggest solutions.
- Notification system: Parents receive notifications for the relevant vaccination dates.
- Personalization: Chatbot is created for each child, and it is unique for each user with the credentials
- Availability: Unlike doctors, the system is available for the user without any appointments and without time consideration.
- De-stress mental health: System existence is to decrease the problematic and emerging situation and to smoothly handle it.

- **Impact and Future Scope**
- The system touches the sensitive area of the health industry by predicting the sickness of infants.
- This method can be recognized as remote consultation via telecommunication, and with the reliability of the prediction, the system may perform an important role as an assistant for the parents of the newborn child.
- Through the training model that contributes with AI and ML algorithms, Image processing techniques, and NLP processing techniques, the accuracy, reliability, and user-friendliness o the system can be optimized.

Information Gathering

- **Number of interview sessions were conducted to finalize the component by the field specialists.**

✓ Dr. Christine Buddhini – Panadura Hospital

✓ Dr. Thilini Amarathunga - Muttur Hospital

- **Research was conducted after the comments of the panel and the specialists.**
  - **Existing systems**
  - **Research papers**
  - **Video tutorials**
- **For the implementation, tutorial videos, blogs, and chatbot documentation are followed.**

Component Overview Diagram

Software Engineer

Chatbot Modeling Language

Intent Package

uses

Execution Package

uses

Backend Models

Intent Recognition NLU Providers

Chatbot Runtime

Backend Services

GUI-to-text

Chatbot user

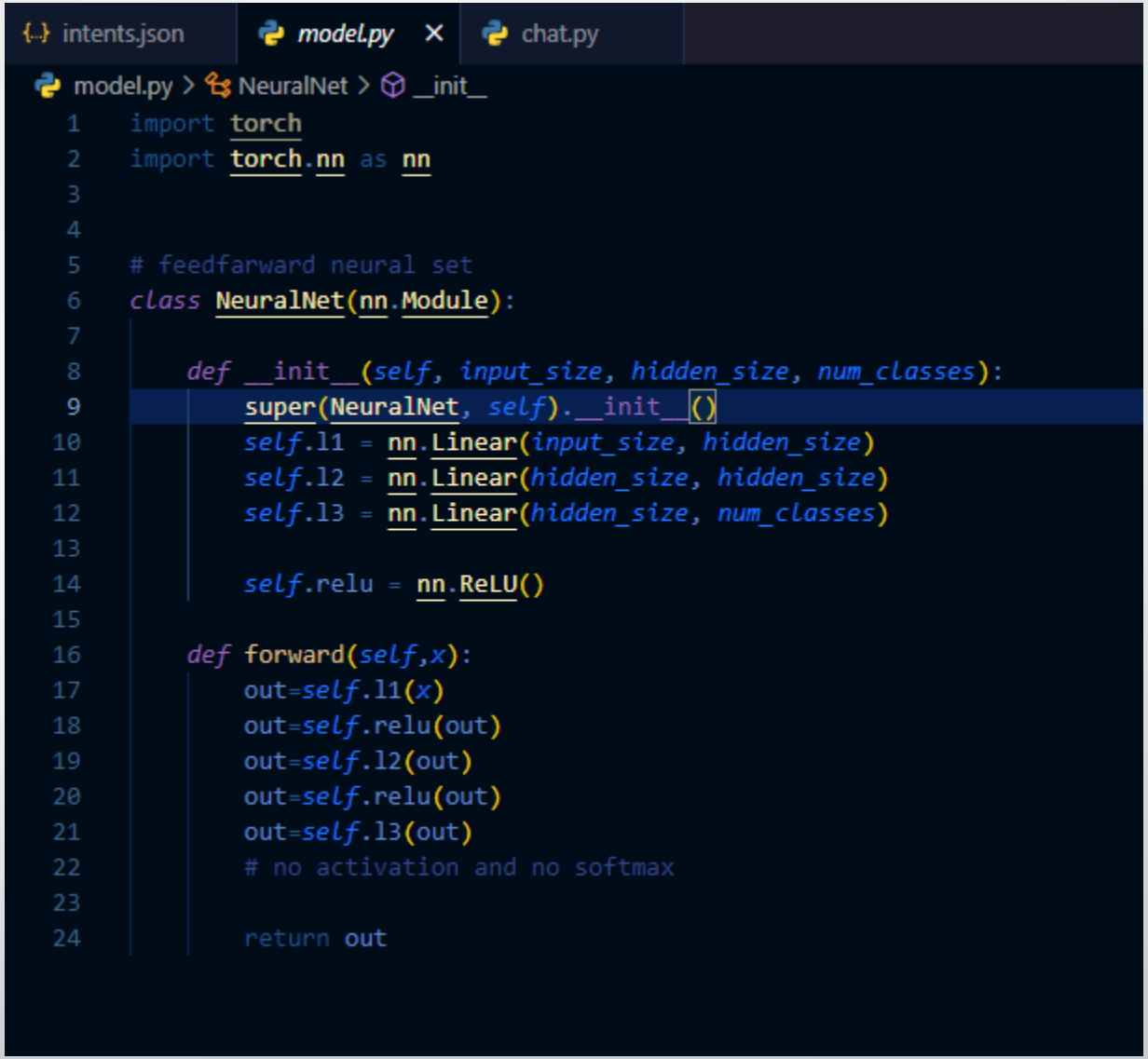
Model Implementation

Intents file

This file consists of the training dataset to train the model. All the data is hard-coded into a text file with tags, patterns, and corresponding responses

Model

This code uses forward propagation of neural network technique and **relu** function to apply weights, and bias terms in terms to reduce the difference between actual output and predict output



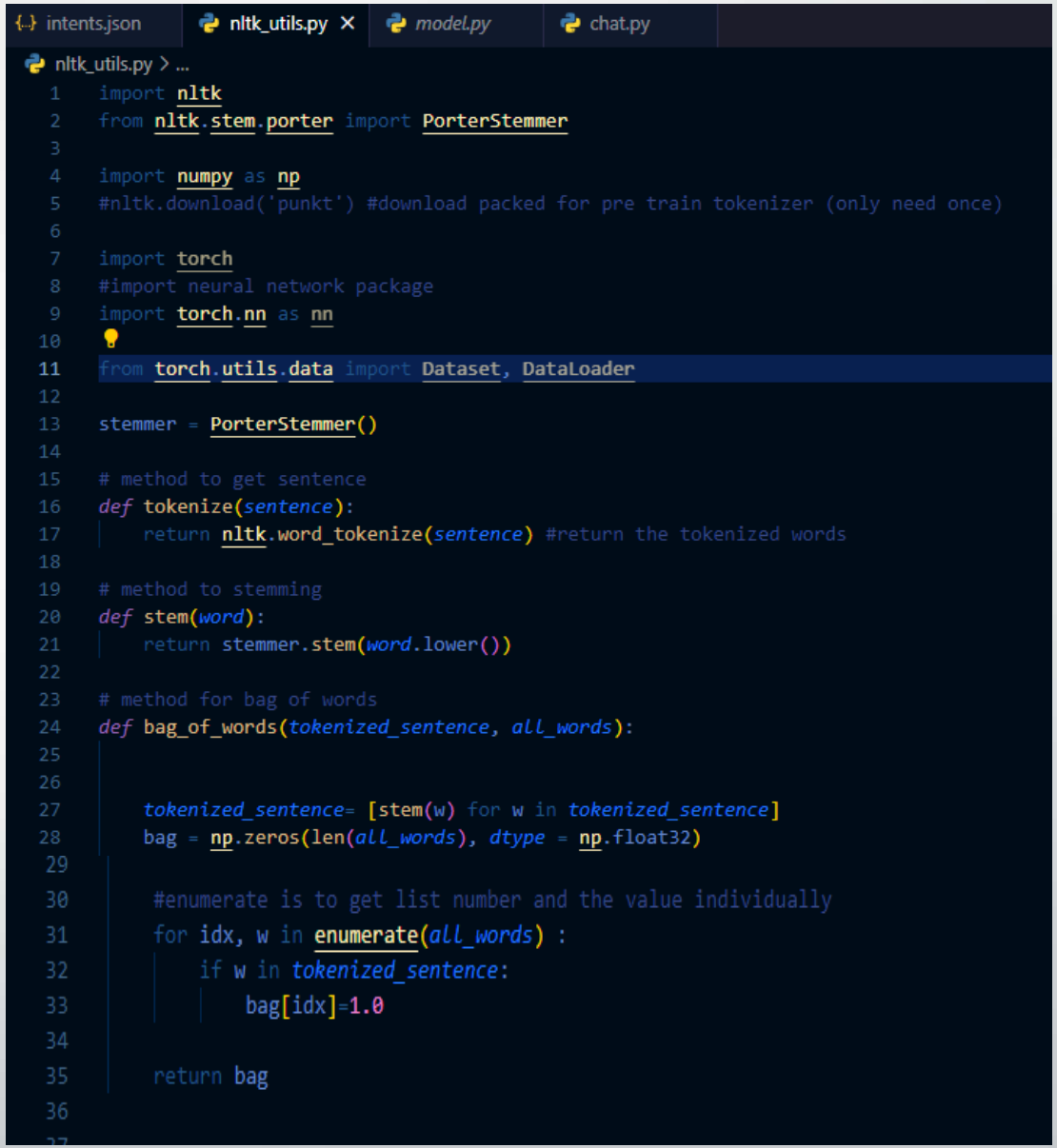
NLTK

NLTK stands for Natural Language Toolkit. This technique is used for tokenization, stemming, and getting the bag of words

Tokenization -Split strings into meaningful units

Stemming –Generate the root form by cutting the end off

Bag_of_words – the final outcome of binary values of stemmed words



Training file

- The intent file,
- NLTK,
- Numpy,
- Pytorch,
- The dataset
- the model,

of the system are imported into the training file. The file is to train the model by the dataset by using each technique

- **numpy**: A library for numerical operations and array manipulation.
- **Pytorch:** open-source machine learning library that provides a flexible framework for building and training deep learning models efficiently.
- **Epoch**: refers to a single pass through the entire training dataset during the training process of a model. It helps optimize the model's parameters by updating them based on the calculated loss and the chosen optimization algorithm.

```python
batch_size = 8
hidden_size = 8
output_size = len(tags)
input_size = len(X_train[0])
learning_rate = 0.001
#complete iteration of a dataset during training phase
num_epochs = 1000

# print(input_size, output_size)

class ChatDataset(Dataset):

    def __init__(self):
        self.n_samples = len(X_train)
        self.x_data = X_train
        self.y_data = y_train

    # support indexing such that dataset[i] can be used to get i-th sample
    def __getitem__(self, index):
        return self.x_data[index], self.y_data[index]

    # we can call len(dataset) to return the size
    def __len__(self):
        return self.n_samples

dataset = ChatDataset()
train_loader = DataLoader(dataset=dataset,
                          batch_size=batch_size,
                          shuffle=True,
```

```python
        xy.append((w, tag))

# stem and lower each word
#ouit the signs and get the words out from sentence
ignore_words = ['?', '.', '!']
all_words = [stem(w) for w in all_words if w not in ignore_words]

#sorting to ignore mutiplication of words
all_words = sorted(set(all_words))
tags = sorted(set(tags))
#print(tags)

# print(len(xy), "patterns")
# print(len(tags), "tags:", tags)
# print(len(all_words), "unique stemmed words:", all_words)

# create training data
X_train = []
y_train = []
for (pattern_sentence, tag) in xy:
    # X: bag of words for each pattern_sentence
    bag = bag_of_words(pattern_sentence, all_words)
    X_train.append(bag)
    # y: PyTorch CrossEntropyLoss needs only class labels, not one-hot
    # numbering the words inorder
    label = tags.index(tag)
    y_train.append(label)

X_train = np.array(X_train)
```

train.py > ...

```python
import numpy as np
import random
import json

import torch
import torch.nn as nn
from torch.utils.data import Dataset, DataLoader

from nltk_utils import bag_of_words, tokenize, stem
from model import NeuralNet

with open('intents.json', 'r') as f:
    intents = json.load(f)
    # print(intents) #-> to check

all_words = [] #for all the words from sentences
tags = [] # tags as in intents file
xy = [] #empty list that will be hold tags and patterns after the process

# loop through each sentence in our intents patterns
# with the key "intents", all there in intents file
for intent in intents['intents']:
    tag = intent['tag']
    # add to tag list
    tags.append(tag)
    for pattern in intent['patterns']:
        # tokenize each word in the sentence
        w = tokenize(pattern)
        # add to our words list
```

intents.json | nltk_utils.py | train.py | chat.py

train.py > ...

```python
dataset = ChatDataset()
train_loader = DataLoader(dataset=dataset,
                          batch_size=batch_size,
                          shuffle=True,
                          num_workers=0)

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

model = NeuralNet(input_size, hidden_size, output_size).to(device)

# Loss and optimizer
criterion = nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(model.parameters(), lr=learning_rate)

# Train the model
for epoch in range(num_epochs):
    for (words, labels) in train_loader:
        words = words.to(device)
        labels = labels.to(dtype=torch.long).to(device)

        # Forward pass
        outputs = model(words)
        # if y would be one-hot, we must apply
        # labels = torch.max(labels, 1)[1]
        loss = criterion(outputs, labels)

        # Backward and optimize
```

intents.json | nltk_utils.py | train.py | chat.py

train.py > ...

```python

    if (epoch+1) % 100 == 0:
        print (f'Epoch [{epoch+1}/{num_epochs}], Loss: {loss.item():.4f}')


print(f'final loss: {loss.item():.4f}')

# create dictionary to save the data
data = {
"model_state": model.state_dict(),
"input_size": input_size,
"hidden_size": hidden_size,
"output_size": output_size,
"all_words": all_words,
"tags": tags
}

FILE = "data.pth"
#serialized and save in to pricled file
torch.save(data, FILE)

print(f'training complete. file saved to {FILE}')
```

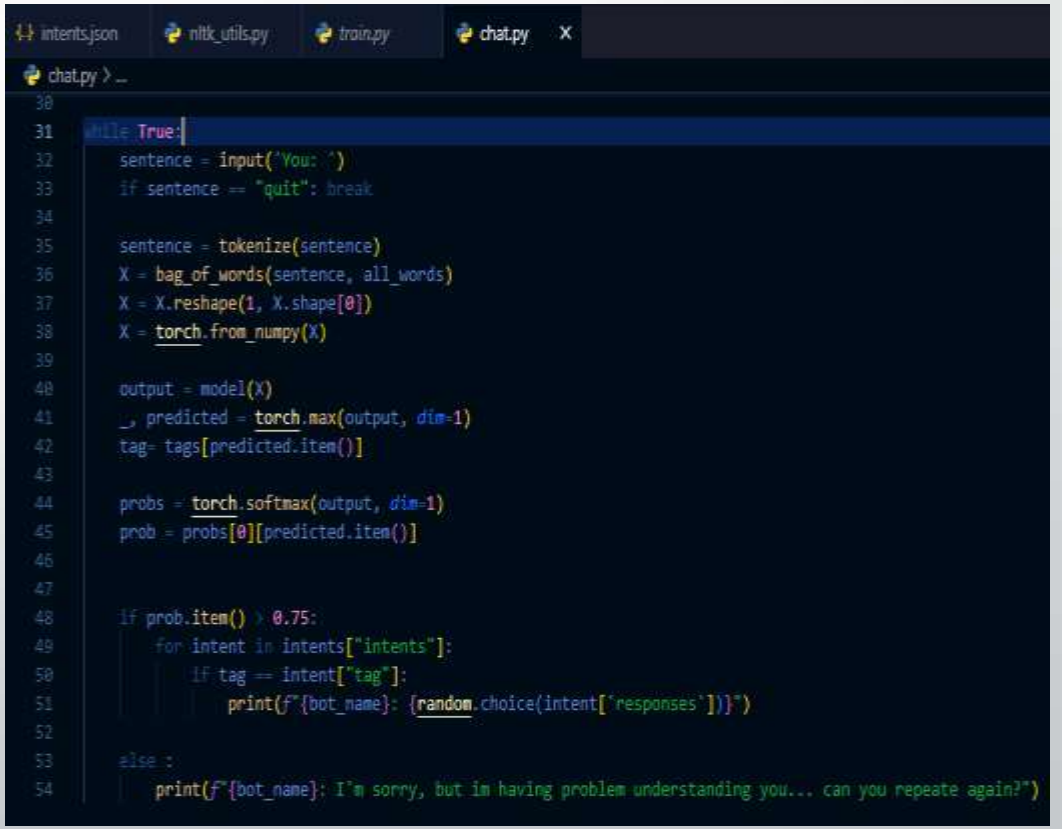
Chat file

This code uses by the user interaction

intents.json | nltk_utils.py | train.py | chat.py

chat.py > ...

```python
import random #to make random choice from possible answers
import json
import torch
from model import NeuralNet
from nltk_utils import bag_of_words, tokenize


device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

with open('intents.json','r') as a:
    intents = json.load(a)

#load the aved data from data file
FILE = "data.pth"
data= torch.load(FILE)

input_size = data["input_size"]
hidden_size = data["hidden_size"]
output_size = data["output_size"]
all_words = data["all_words"]
tags = data["tags"]
model_state = data["model_state"]

model = NeuralNet(input_size, hidden_size, output_size).to(device)
model.load_state_dict(model_state)
model.eval()

bot_name = "Nanny_Bot"
print("Let's chat! type 'quit' to exit")
```

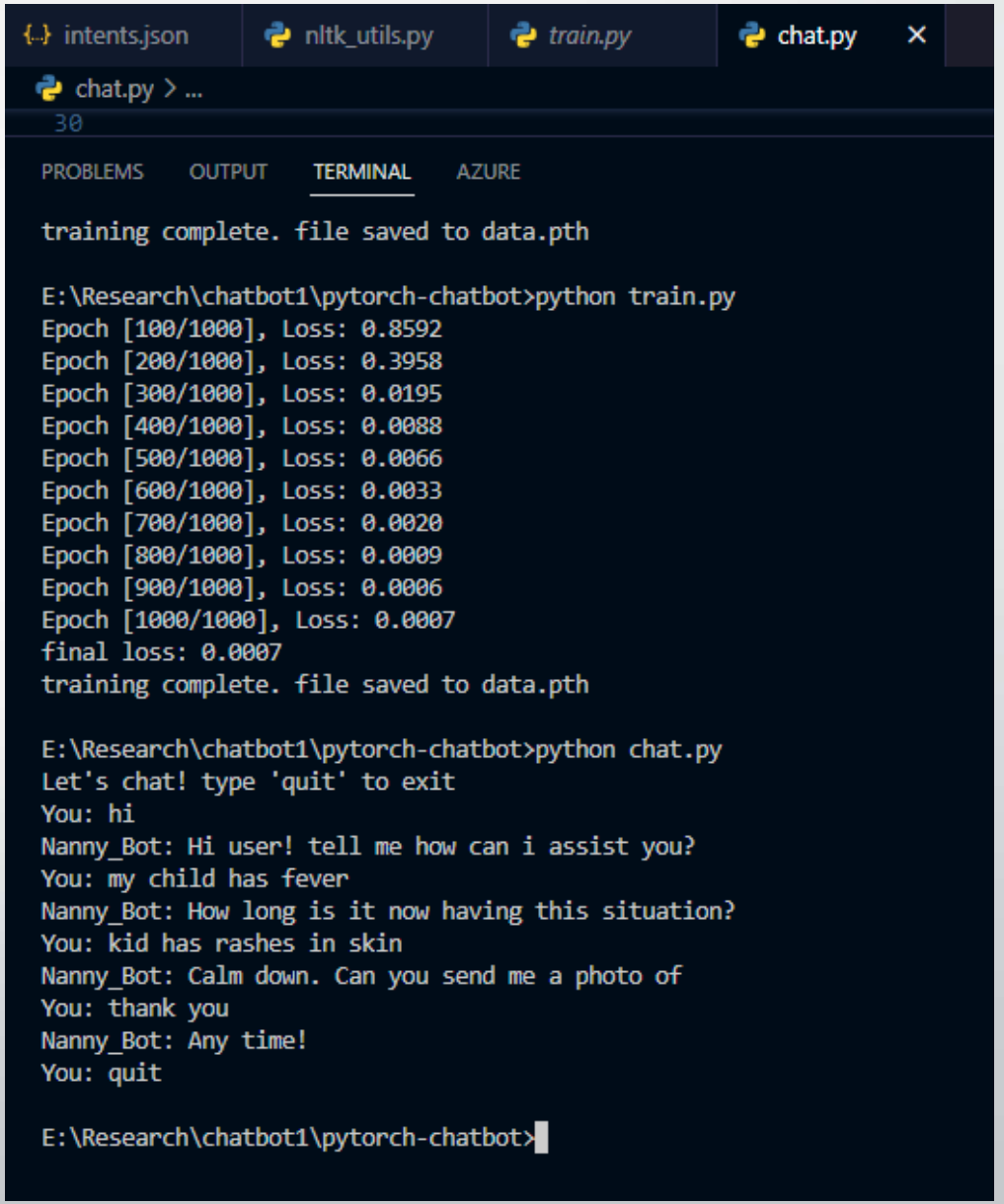


```python
while True:
    sentence = input('You: ')
    if sentence == "quit": break

    sentence = tokenize(sentence)
    X = bag_of_words(sentence, all_words)
    X = X.reshape(1, X.shape[0])
    X = torch.from_numpy(X)

    output = model(X)
    _, predicted = torch.max(output, dim=1)
    tag= tags[predicted.item()]

    probs = torch.softmax(output, dim=1)
    prob = probs[0][predicted.item()]


    if prob.item() > 0.75:
        for intent in intents["intents"]:
            if tag == intent["tag"]:
                print(f"{bot_name}: {random.choice(intent['responses'])}")

    else :
        print(f"{bot_name}: I'm sorry, but im having problem understanding you... can you repeate again?")
```

Final output

- Training result

```
E:\Research\chatbot1\pytorch-chatbot>python train.py
Epoch [100/1000], Loss: 1.0175
Epoch [200/1000], Loss: 0.0666
Epoch [300/1000], Loss: 0.0782
Epoch [400/1000], Loss: 0.0036
Epoch [500/1000], Loss: 0.0052
Epoch [600/1000], Loss: 0.0009
Epoch [700/1000], Loss: 0.0008
Epoch [800/1000], Loss: 0.0012
Epoch [900/1000], Loss: 0.0005
Epoch [1000/1000], Loss: 0.0003
final loss: 0.0003
training complete. file saved to data.pth

E:\Research\chatbot1\pytorch-chatbot>
```



IT20405090 | Vanhoff R. L.

Growth Level Prediction

Software Engineering

Background

- After the proposal presentation the component was asked to change to a growth level predictor instead of a nutritional level identification and the functionality was changed from image processing to training a model to predict the growth level, by the panel.
- Information was gathered from the specialists and doctors in order to gather information for the predictions to make it more accurate and to discuss on how the functionality can be done.
- But after the process, it was stated that the growth level cannot be measured without the height, weight and age-appropriate other measures with just the behavioral milestones.
- As the final decision, the component was changed to a prediction function where according to the given measures of a child, the growth level can be predicted by the model.

Information Gathering

- Several interviews were held with the doctors to discuss about the changed function before finalizing the component.

✓ **Dr. Udara Ariyasinghe - Arogya Family Medical Center**

✓ **Dr. Thilini Amarathunga - Muttur Hospital**

- Investigations were done by exploring the available systems, past research papers, tutorials and other related materials to get a clear understanding about doing predictions using a model.
- Referred online materials such as YouTube videos, online surveys, WHO documentation and other medicine related documentation to use the correct information to train the model and start the implementations.

Component Discussion

The Function's Primary Goal - The purpose is to give parents an automated method for determining and tracking their infants' growth levels. The function estimates the developmental stage based on several factors about a kid, including weight, height, age, and other development milestones.

The following are the function's main objectives:

- **Development Level Prediction**: The Decision Tree Regressor method, a machine learning approach, is used by the function to forecast newborns' development levels. The function trains a regression model to precisely forecast the growth level based on the input measurements by examining the provided dataset, which comprises development milestones data for children aged 0–5 years.
- **Precautionary Measures:** If the expected growth level suggests overgrowth or undergrowth, the function is intended to help parents take the appropriate safety measures. Parents can better comprehend their baby's development and seek the necessary medical advice or intervention by having insights about the child's growth trajectory.
- **User-Friendly Interfaces**: The function is incorporated into a user-friendly front-end web application that is a full-stack web application. Parents may easily input their baby's measurements into the interface to get a prediction of growth level. To make it simple for parents to understand and respond to the results, the interface also displays the forecast as a % along with corresponding categories of poor, fair, good, and excellent.

By emphasizing these crucial elements, the function intends to give parents a useful and instructive tool for tracking their child's development and encouraging proactive healthcare management.

Component Discussion

User Workflow

- **Enter Child's Information**: Provide the child's age, height, weight, and select or input information about their developmental milestones in various domains, such as motor skills, problem-solving, communication, and emotional development.
- **Predict Growth Level**: Click the "Predict the Growth" button to initiate the prediction process. The system will process the entered data and calculate the child's growth level.
- **View Result**: Once the prediction is complete, view the predicted growth level displayed on the screen. Take note of the result and any recommendations provided by the application.

Impact and Future Scope

Impact:

- Empowers parents with valuable insights into their child's progress.
- Enables early detection of developmental delays or potential issues.
- Facilitates timely interventions and support.
- Promotes proactive parenting by offering personalized recommendations.
- Contributes to informed decision-making for child development.

Future Scope:

- Calculation of different milestones and personalized recommendations.
- Expansion of assessed milestones and incorporation of machine learning techniques.
- Integration of a comprehensive growth database for accurate comparisons.
- Inclusion of educational resources and interactive features.
- Continuous evolution and advancements as a valuable parenting tool.

System Workflow

- **Data collection**: The system begins by gathering a dataset that includes information on the developmental stages of children aged 0 to 5. The growth prediction model is trained using the dataset as its foundation.
- **Data preprocessing**: To extract pertinent characteristics, the collected dataset is preprocessed. The algorithm in this instance chooses particular metrics like weight, height, age, emotional development, social skills, emotional issues, and growth rate. In addition to addressing missing values, categorical variables must be encoded using LabelEncoder, and the data must be split into training and testing sets using train_test_split.
- **Model Training**: The system trains the growth predictor model using the DecisionTreeRegressor method from the scikit-learn package. The selected characteristics and accompanying growth levels are included in the training dataset, which is used to train the model.
- **Model Evaluation**: Using the testing dataset, the model is assessed after training. To evaluate the model's success, the system predicts the growth levels for the testing data and contrasts them with the actual growth levels. The accuracy_score measure is used to determine the model's accuracy.

System Workflow

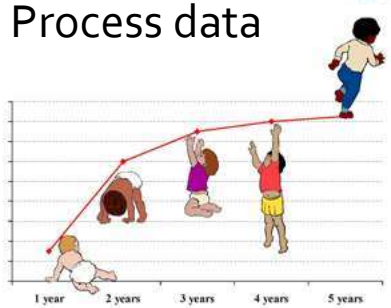
- **Model Persistence**: After the model has been trained and assessed, it is stored using the pickle library in a serialized manner. The model is saved in a file along with the label encoders required to encode categorical variables.
- **Front-End Integration**: A full-stack online application incorporates the growth prediction model. Parents can pick certain observations or activities pertaining to their baby's growth and development using checkboxes on the front-end interface.
- **Prediction Calculation**: The system determines the score based on the number of selected actions when the user checks the appropriate checkboxes. The ratio of the number of chosen activities to the total number of potential actions is multiplied by 100 to get the score, which is then expressed as a percentage.
- **Forecast Interpretation**: Based on the estimated %, the algorithm chooses one of four growth forecast categories (poor, fair, good, or great). For example, 80% for exceptional, 60% for good, and 40% for fair, are used as specific thresholds to determine which category a forecast belongs in.
- **Results Presentation**: The system shows the user the growth projection category. Based on the chosen observations, the category represents the baby's growth stage. On the front-end interface, the results are displayed in an approachable manner, giving parents useful information about their child's development and the essential safety measures to be implemented.

Component Overview Diagram

Input data will be entered by the parents

Output will be displayed as a prediction

Process data



Machine Learning Model

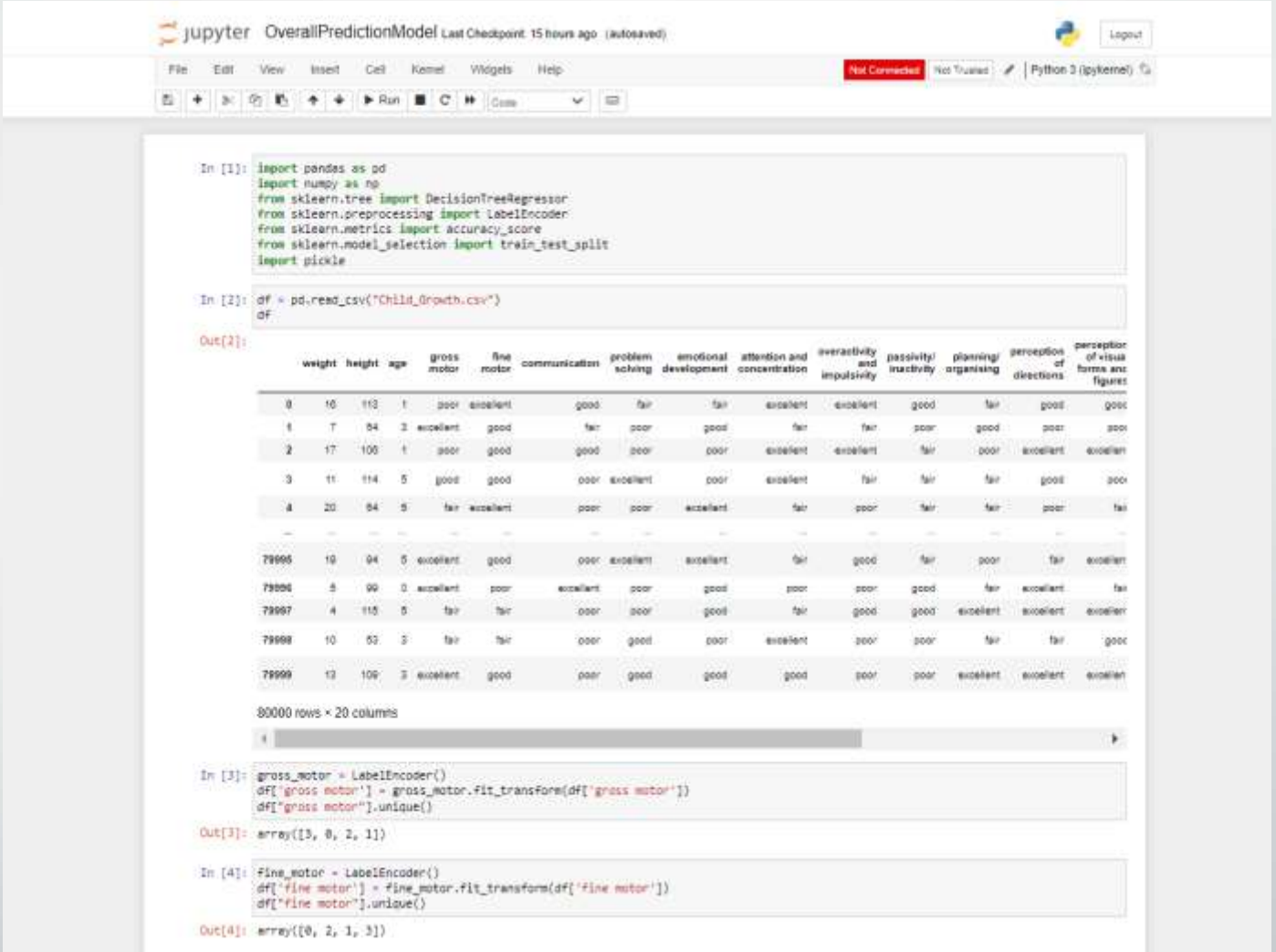
Train the Machine Learning model using a dataset

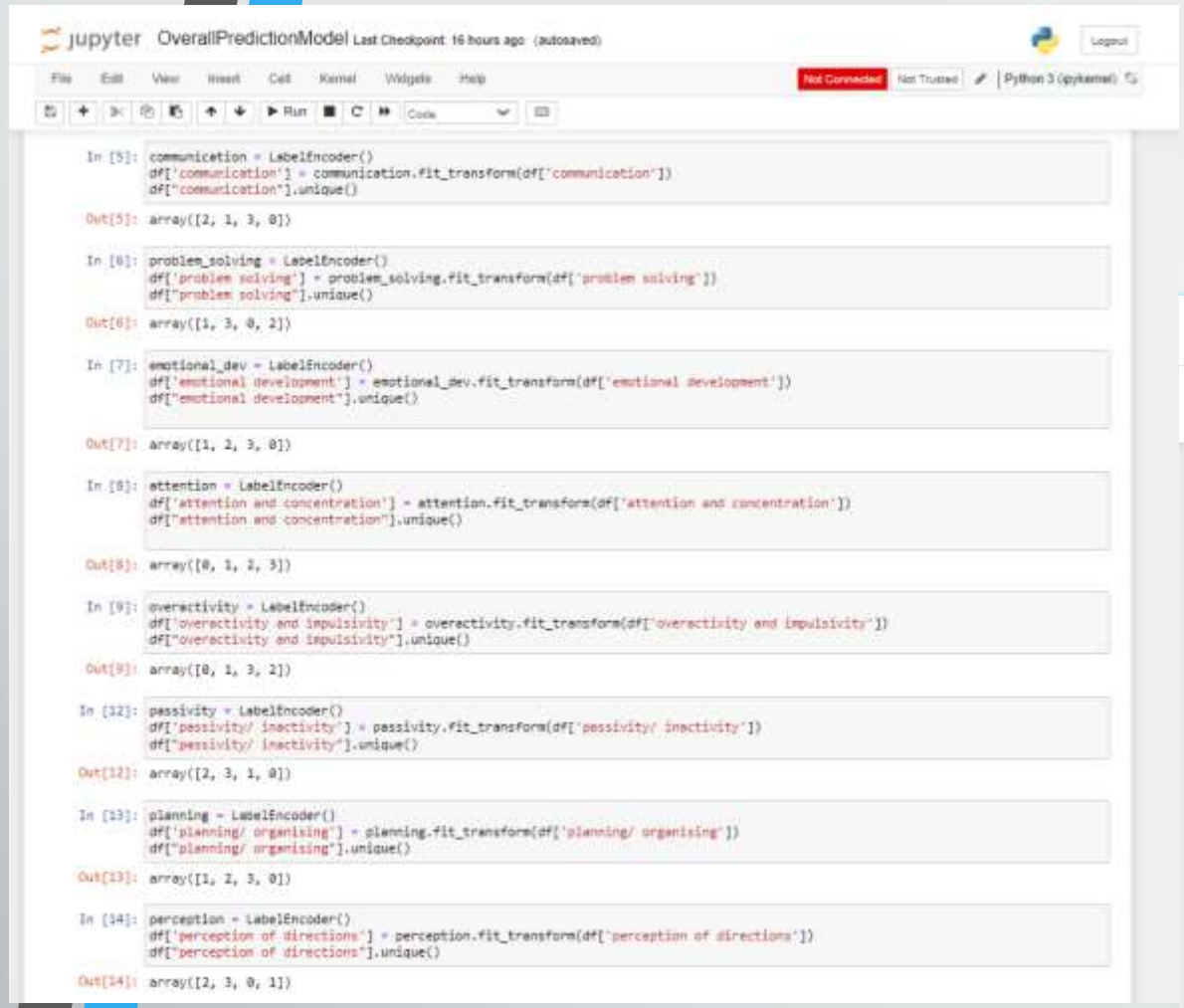
Implementation

Developing and training a model

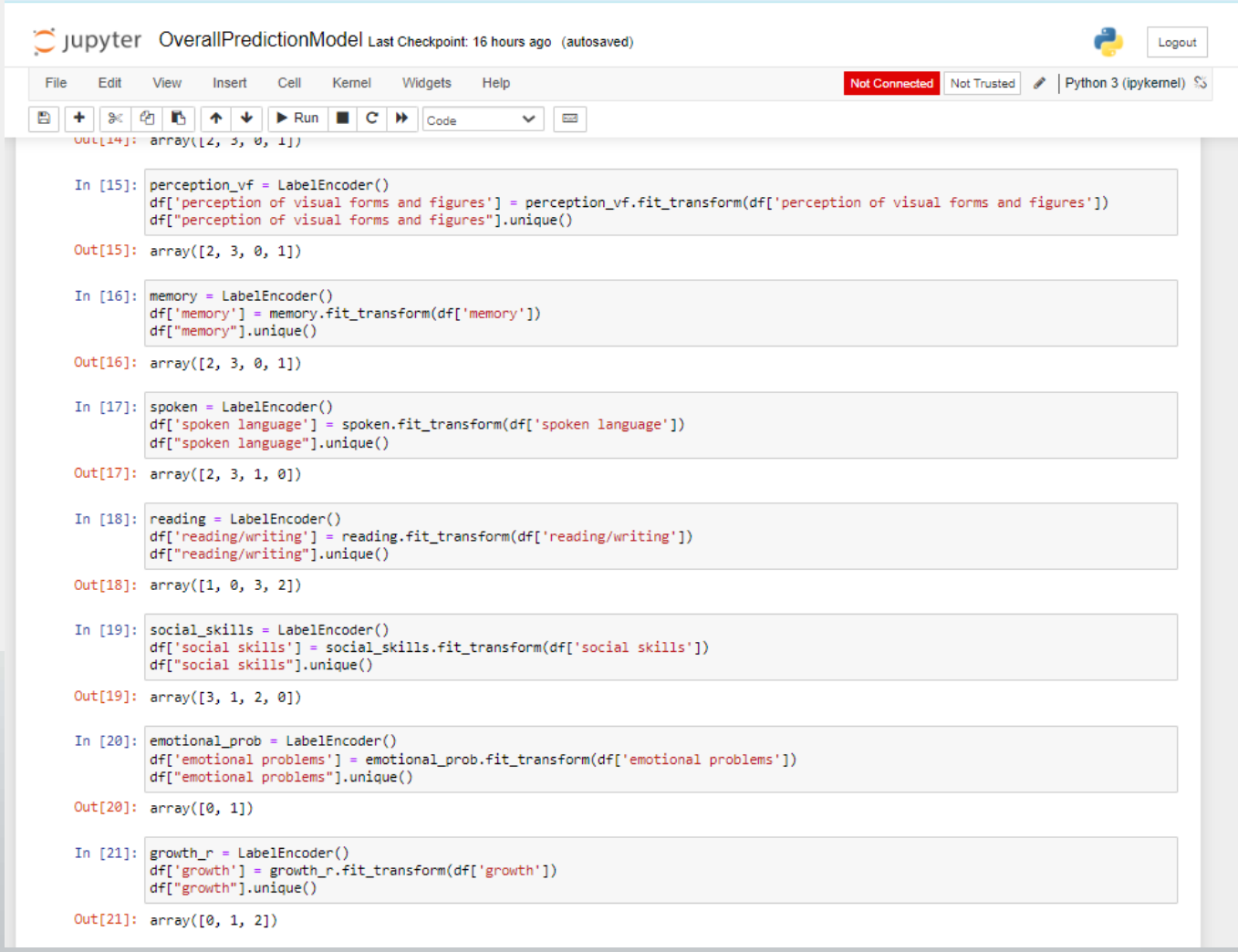
Libraries:

- **numpy**: A library for numerical operations and array manipulation.
- **pandas**: A library for data manipulation and analysis.
- **sklearn.tree.DecisionTreeRegressor**: A class from scikit-learn library for implementing decision tree regression.
- **sklearn.preprocessing.LabelEncoder**: A class from scikit-learn library for label encoding categorical variables.
- **sklearn.metrics.accuracy_score**: A function from scikit-learn library for computing accuracy scores.
- **sklearn.model_selection.train_test_split**: A function from scikit-learn library for splitting data into training and testing sets.
- **pickle**: A module for serializing and deserializing Python objects.





- The code loads a dataset from a CSV file using pd.read_csv from pandas library.
- The dataset is then filtered to select specific columns relevant to the model.
- Missing values are dropped using **dropna** function.
- Categorical variables are encoded using **LabelEncoder**.



Data set reference - https://www.kaggle.com/datasets/salmanahmad1980/child-growth-measurements

- The code splits the preprocessed data into input features (X) and the target variable (y).
- The train_test_split function is used to divide the data into training and testing sets, with a test size of 20% and a random state of 0.
- **A decision tree regressor model** is instantiated using **DecisionTreeRegressor.**
- The model is trained using the training data with the fit method.

```
In [23]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
         dec_tree_reg = DecisionTreeRegressor(random_state=0)

In [24]: dec_tree_reg.fit(X_train, y_train)
         y_pred_train = dec_tree_reg.predict(X_train)
         y_pred_test = dec_tree_reg.predict(X_test)

In [25]: train_accuracy = accuracy_score(y_train, y_pred_train.round())
         test_accuracy = accuracy_score(y_test, y_pred_test.round())

In [26]: print("Training Accuracy:", train_accuracy)
         print("Testing Accuracy:", test_accuracy)

         Training Accuracy: 1.0
         Testing Accuracy: 0.335625
```

Accuracy

```
In [28]: # Save the model and LabelEncoders in a dictionary
         data = {
             "model": dec_tree_reg,
             "gross_motor":gross_motor,
             "fine_motor":fine_motor,
             "communication":communication,
             "problem_solving":problem_solving,
             "emotional_dev":emotional_dev,
             "attention":attention,
             "overactivity":overactivity,
             "passivity":passivity,
             "planning":planning,
             "perception":perception,
             "perception_vf":perception_vf,
             "memory":memory,
             "spoken":spoken,
             "reading":reading,
             "social_skills":social_skills,
             "emotional_prob":emotional_prob,
             "growth": growth_r
         }

In [29]: # Save the dictionary as a pickle file
         with open('overAllModel.pkl', 'wb') as file:
             pickle.dump(data, file)
```

Selection of Decision Tree Regressor

- Highest accuracy score and lowest error compared to other algorithms.

```
In [12]: #trying the different machine learning models
         from sklearn.linear_model import LinearRegression
         linear_reg = LinearRegression()
         linear_reg.fit(X, y.values)

Out[12]: LinearRegression()

In [13]: y_pred = linear_reg.predict(X)

In [14]: from sklearn.metrics import mean_squared_error, mean_absolute_error
         import numpy as np
         error = np.sqrt(mean_squared_error(y, y_pred))

In [15]: error

Out[15]: 0.8156768147782453
```

```
In [21]: from sklearn.model_selection import GridSearchCV

         max_depth = [None, 2,4,6,8,10,12]
         parameters = {"max_depth": max_depth}

         regressor = DecisionTreeRegressor(random_state=0)
         gs = GridSearchCV(regressor, parameters, scoring='neg_mean_squared_error')
         gs.fit(X, y.values)

Out[21]: GridSearchCV
         estimator: DecisionTreeRegressor
         DecisionTreeRegressor

In [17]: regressor = gs.best_estimator_

         regressor.fit(X, y.values)
         y_pred = regressor.predict(X)
         error = np.sqrt(mean_squared_error(y, y_pred))
         print(error)

         0.8156377061858167
```

```
In [19]: from sklearn.ensemble import RandomForestRegressor
         random_forest_reg = RandomForestRegressor(random_state=0)
         random_forest_reg.fit(X, y.values)

Out[19]: RandomForestRegressor(random_state=0)

In [20]: y_pred = random_forest_reg.predict(X)
         error = np.sqrt(mean_squared_error(y, y_pred))
         print(error)

         0.4164121319965191
```

```
]: from sklearn.tree import DecisionTreeRegressor
   dec_tree_reg = DecisionTreeRegressor(random_state=0)
   dec_tree_reg.fit(X, y.values)

]: DecisionTreeRegressor(random_state=0)

]: y_pred = dec_tree_reg.predict(X)

]: import numpy as np
   from sklearn.metrics import mean_squared_error, mean_absolute_error
   error = np.sqrt(mean_squared_error(y, y_pred))
   print(error)

   0.3083555172199778
```

Infant Growth Level Prediction

We need some information to predict the growth level of the child according to the development milestones

Age of the Infant

0 — 5

Height (to the nearest integer in centimeters)

0

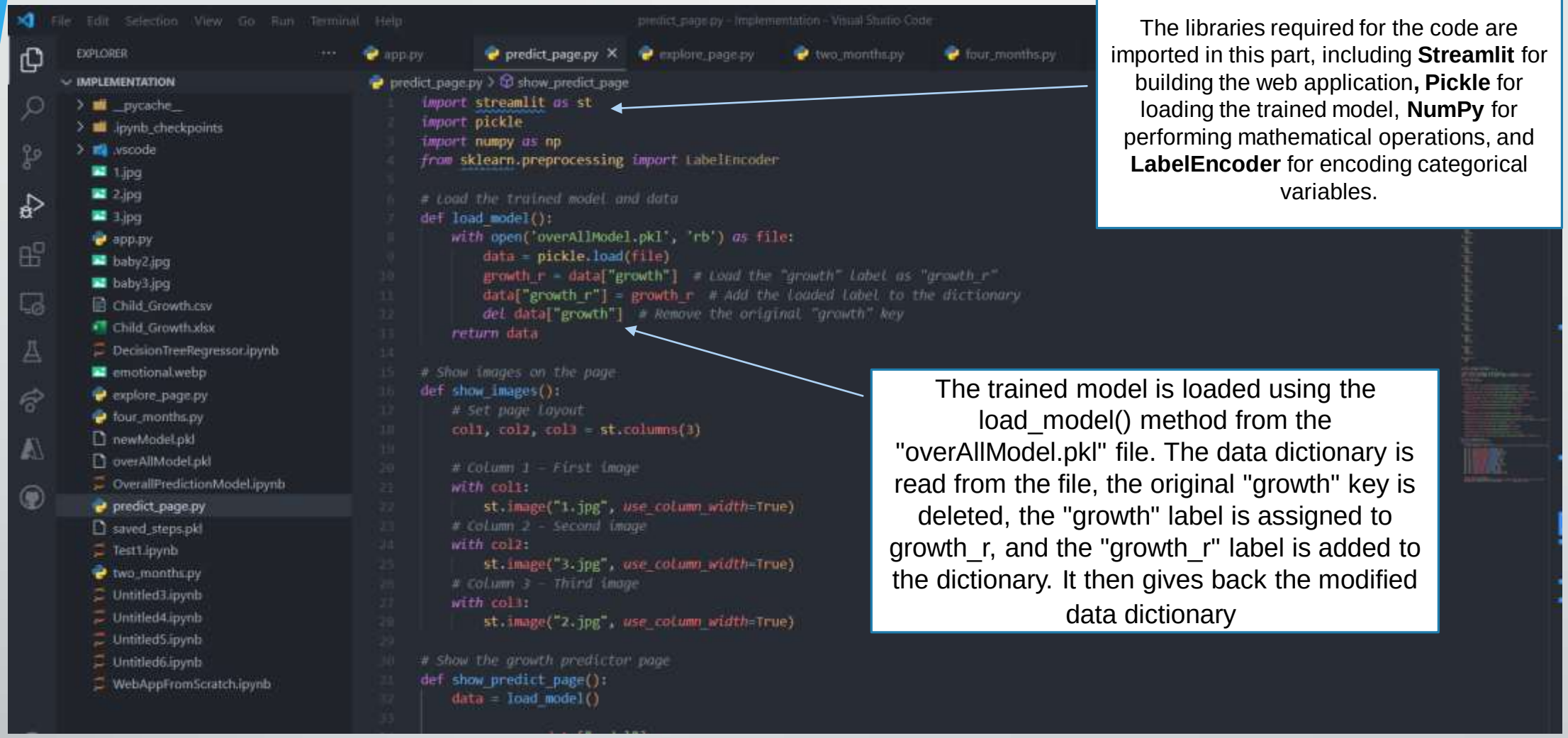
Weight (to the nearest integer in kilograms)

0

Gross Motor Development Status: poor	How Organized And Planned?: poor
Fine Motor Development Status: poor	How Good At Perception Of Directions?: poor
How Good At Solving Problems?: poor	How Good At Perception Of Visual Forms And Figures?: poor
How Good At Communication?: poor	How Good Is The Memory?: poor
Emotional Development Status: poor	How Good At Reading Skills?: poor
Attention And Concentration: poor	How About Speaking Skills?: poor
Overactivity and Impulsivity: poor	Social Skills Status: poor
Passivity Or Inactivity: poor	Are There Any Emotional Problems?: yes

Predict the Growth

Growth level can be predicted as : Under Growth



The libraries required for the code are imported in this part, including **Streamlit** for building the web application**, Pickle** for loading the trained model, **NumPy** for performing mathematical operations, and **LabelEncoder** for encoding categorical variables.

The trained model is loaded using the load_model() method from the "overAllModel.pkl" file. The data dictionary is read from the file, the original "growth" key is deleted, the "growth" label is assigned to growth_r, and the "growth_r" label is added to the dictionary. It then gives back the modified data dictionary

predict_page.py > show_predict_page

```python
# Show the growth predictor page
def show_predict_page():
    data = load_model()

    regressor = data["model"]
    gross_motor_encoder = data["gross_motor"]
    fine_motor_encoder = data["fine_motor"]
    problem_solving_encoder = data["problem_solving"]
    communication_encoder = data["communication"]
    emotional_dev_encoder = data["emotional_dev"]
    attention_encoder = data["attention"]
    overactivity_encoder = data["overactivity"]
    passivity_encoder = data["passivity"]
    planning_encoder = data["planning"]
    perception_encoder = data["perception"]
    perception_vf_encoder = data["perception_vf"]
    memory_encoder = data["memory"]
    spoken_encoder = data["spoken"]
    reading_encoder = data["reading"]
    social_skills_encoder = data["social_skills"]
    emotional_prob_encoder = data["emotional_prob"]


# Create a mapping dictionary for numerical labels to string labels
    label_mapping = {
        2: "Under Growth",
        0: "Normal Growth",
        1: "Over Growth"
    }
```

predict_page.py > show_predict_page

```python
# widgets-title
st.title("Infant Growth Level Prediction")
# widgets-text
# h3 tag - ###
st.write("""### We need some information to predict the growth level of the child according to the development milestones""")

# Call the function
show_images()

#choices for the dropdowns
gross_motor = (
    "poor",
    "fair",
    "good",
    "excellent"
)

fine_motor = (
    "poor",
    "fair",
    "good",
    "excellent"
)
problem_solving = (
    "poor",
    "fair",
    "good",
    "excellent"
)
```

The primary function that renders the growth predictor page is show_predict_page(). It uses the load_model() method to initially load the model and any other necessary data. Then, it divides up the supplied data into distinct variables for further usage.

predict_page.py > show_predict_page

```python
    )

    emotional_prob = (
        "yes",
        "no"

    )

    # creates a slider for age selection
    age = st.slider("Age of the Infant", 0, 5, 0)

    # creates number input fields for height and weight
    height = st.number_input("Height (to the nearest integer in centimeters)", min_value=0)
    weight = st.number_input("Weight (to the nearest integer in kilograms)", min_value=0)

    # creates select boxes
    col1, col2 = st.columns(2)

    with col1:
        gross_motor_status = st.selectbox("Gross Motor Development Status", gross_motor)

        fine_motor_status = st.selectbox("Fine Motor Development Status", fine_motor)

        problem_solving_status = st.selectbox("How Good At Solving Problems?", problem_solving)

        communication_status = st.selectbox("How Good At Communication?", communication)

        emotional_development_status = st.selectbox("Emotional Development Status", emotional_dev)

        attention_status = st.selectbox("Attention And Concentration", attention)

        overactivity_status = st.selectbox("Overactivity and Impulsivity", overactivity)
```

The slider, select boxes and other components to take the user input, are indicated here.

```python
# Button to trigger prediction
ok = st.button("Predict the Growth")
if ok:
    # Prepare input data for prediction
    X = np.array([[weight, height, age, gross_motor_status, fine_motor_status, problem_solving_status, communication_status, emotional_development_status, attention_s

    X[:, 3] = gross_motor_encoder.transform(X[:, 3])
    X[:, 4] = fine_motor_encoder.transform(X[:, 4])
    X[:, 5] = problem_solving_encoder.transform(X[:, 5])
    X[:, 6] = communication_encoder.transform(X[:, 6])
    X[:, 7] = emotional_dev_encoder.transform(X[:, 7])
    X[:, 8] = attention_encoder.transform(X[:, 8])
    X[:, 9] = overactivity_encoder.transform(X[:, 9])
    X[:, 10] = passivity_encoder.transform(X[:, 10])
    X[:, 11] = planning_encoder.transform(X[:, 11])
    X[:, 12] = perception_encoder.transform(X[:, 12])
    X[:, 13] = perception_vf_encoder.transform(X[:, 13])
    X[:, 14] = memory_encoder.transform(X[:, 14])
    X[:, 15] = reading_encoder.transform(X[:, 15])
    X[:, 16] = spoken_encoder.transform(X[:, 16])
    X[:, 17] = social_skills_encoder.transform(X[:, 17])
    X[:, 18] = emotional_prob_encoder.transform(X[:, 18])


    growth = regressor.predict(X)
    y_pred_string = [label_mapping[label] for label in growth]  # Map numerical labels to string labels
    st.subheader(f"Growth level can be predicted as : " + ", ".join(y_pred_string))
```

Here creates a 2D numpy array X containing the input data for prediction. The values are taken from various variables representing the selected options and input values.
The next set of lines from X[:, 3] = gross_motor_encoder.transform(X[:, 3]) to X[:, 18] = emotional_prob_encoder.transform(X[:, 18]) encode the categorical features in the input data to numerical values using the respective encoders. This is necessary for the machine learning model to make predictions.
growth = regressor.predict(X): This line uses the trained regressor model to make predictions on the prepared input data X. The predicted growth levels are stored in the growth variable.

Work In Progress

Your Baby At Four Months

Social and Emotional Milestones

- [x] Smiles On Their Own To Get The Attention
- [ ] Chuckles (Not Yet A Full Laugh) When Trying to Make Them Laugh
- [ ] Looks At The Parent, Moves Or Make Sounds To Keep The Attention

---Social and Emotional Development: Poor---

Language And Communication Milestones

- [ ] Makes Sounds Like 'ooh', 'aaahh'(cooing)
- [ ] Makes Sounds Back When You Talk To Them
- [x] Turns Head Towards The Sound Of The Voice

---Language And Communication Development: Poor---

Cognitive Milestones (Learning, Thinking, Problem Solving)

- [x] If Hungry, Opens The Mouth When Sees A Bottle Or The Breast
- [x] Look At The Hands With Interest

---Cognitive Development: Excellent---

Movement / Physical Developement Milestones

- [ ] Holds Head Up Steady Without Support When Holding
- [ ] Holds A Toy When Put In The Hands
- [ ] Uses The Arm To Swing At Toys
- [ ] Brings Hands To Mouth
- [ ] Pushes Up Onto Elbows / Forearms When On Tummy

The Development Milestones of a child

There are four different development milestones in a child's life cycle.

- Physical Development - Gross motor and Fine motor skills
- Cognitive Development - Ability to think, reason and solve problems
- Language Development - Communicate using words and gestures, understand language and expressions
- Social and Emotional Development - Express emotions, interactions and attachments

IT20156374| Fernando A.P

Infant Sickness Identification through Video Processing

Software Engineering

Component Discussion

- **Introduction**
- The focus of this research is to develop a system that utilizes video processing techniques to identify abnormal behaviors in infants, aiming to assist parents in recognizing potential illnesses.
- The system enables parents to record videos of their infants and provides guidance on capturing specific behaviors for analysis.
- By employing image processing and video processing techniques, the system analyzes the recorded videos to identify signs of sickness in infants.
- The primary objective of this system is to offer valuable support to parents with babies up to five years old.

- **System Workflow**
- Parent records a video of their infant, following the system's instructions.
- The system processes the recorded video using advanced image and video processing techniques.
- The system detects abnormal behaviors or signs of sickness in the video frames.
- If any abnormal behavior is identified, the system prompts a command or displays a notification message to the parent.

Component Discussion

- **Benefits for Parents**
- Enhanced Monitoring: The system allows parents to actively monitor their infants for potential illnesses.
- Early Detection: By capturing and analyzing abnormal behaviors, the system aids in early detection of sicknesses.
- Prompt Notifications: Parents receive immediate alerts or commands from the system, providing timely guidance and support.
- Peace of Mind: This system empowers parents with a valuable tool to ensure the well-being of their infants.

- **Impact and Future Scope**
- By developing a reliable system for sickness identification in infants, this research aims to contribute to the field of child healthcare and parental support.
- Future enhancements may involve integrating artificial intelligence algorithms to improve the accuracy and efficiency of sickness detection.
- The system could be expanded to include a wider range of sicknesses and developmental abnormalities, catering to a broader age group of children.

Information Gathering

Interviews and discussions about the overall system and individual component

- ✓ Dr.Supun Peris - Ragama Hospital

Resources and study materials to enhance the components

- ✓ Existing Research papers
- ✓ Articles
- ✓ Video tutorials for technologies

Component Overview Diagram

Web cam

Smart phone

Input video of the infants

Video recording by the parents according to the guidelines of the system

Abnormal behaviors detecting

sickness identification

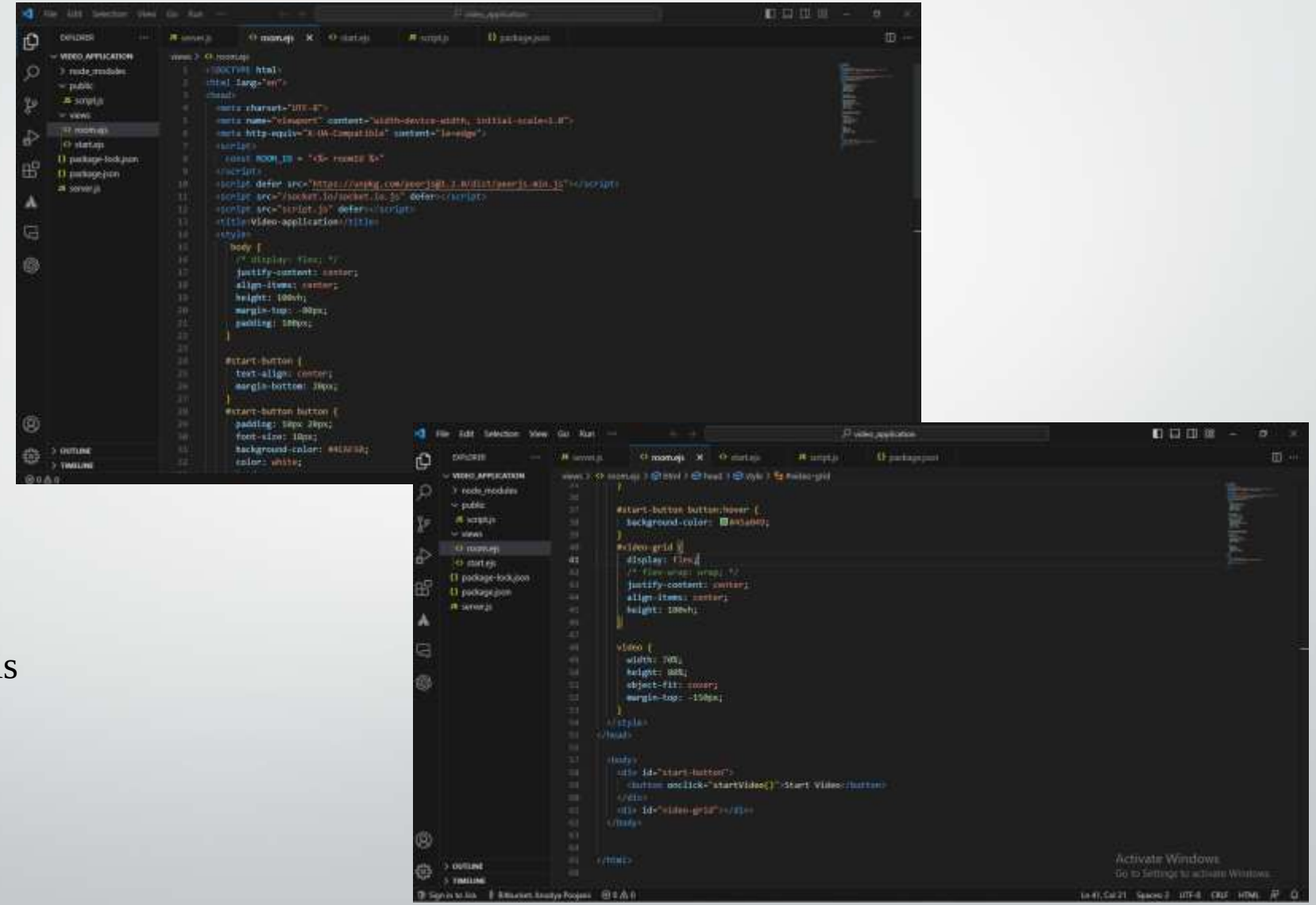
Video processing and analyzing

Results

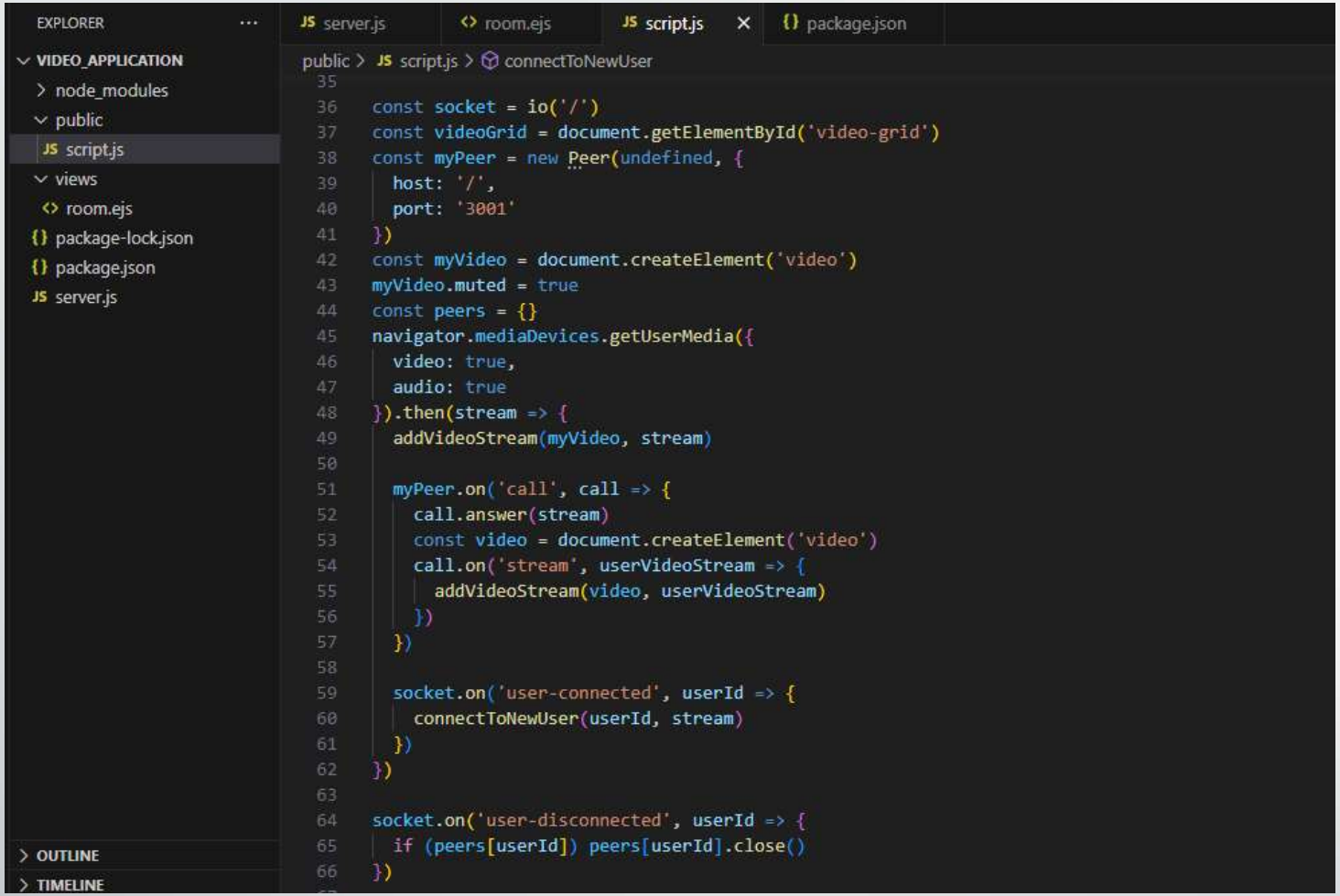
output

Completion of the project

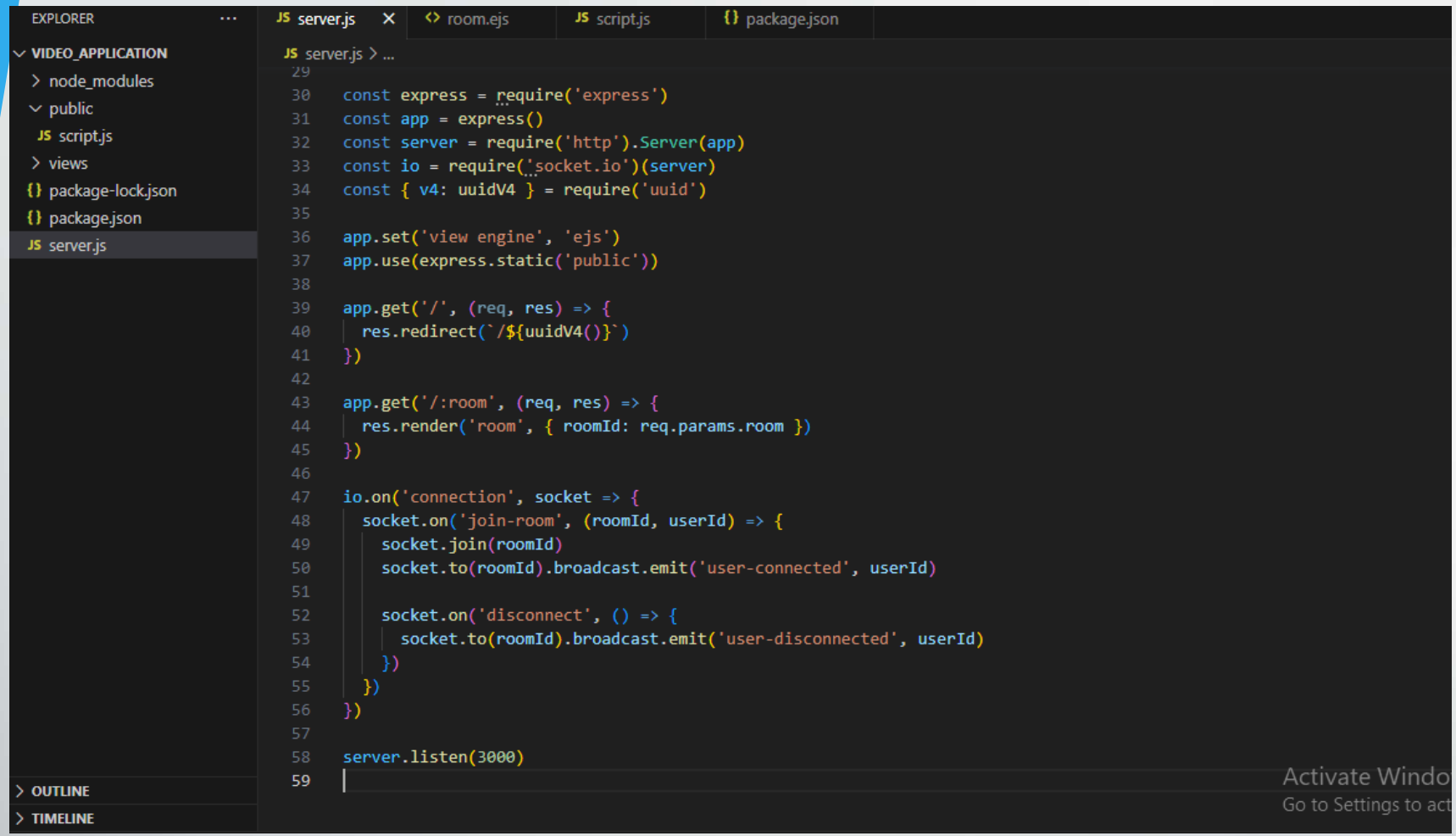
Screenshot of the implementation



Room.ejs – this is one of frontend implementation code segment. This will load the video to the browser.



Script.js – This code sets up a video conferencing application using WebRTC technology. It establishes a connection between clients using socket.io and Peer.js libraries.



Server.js - This server-side code sets up a signaling server using Node.js, Express, and Socket.IO to facilitate communication between clients in the video application.

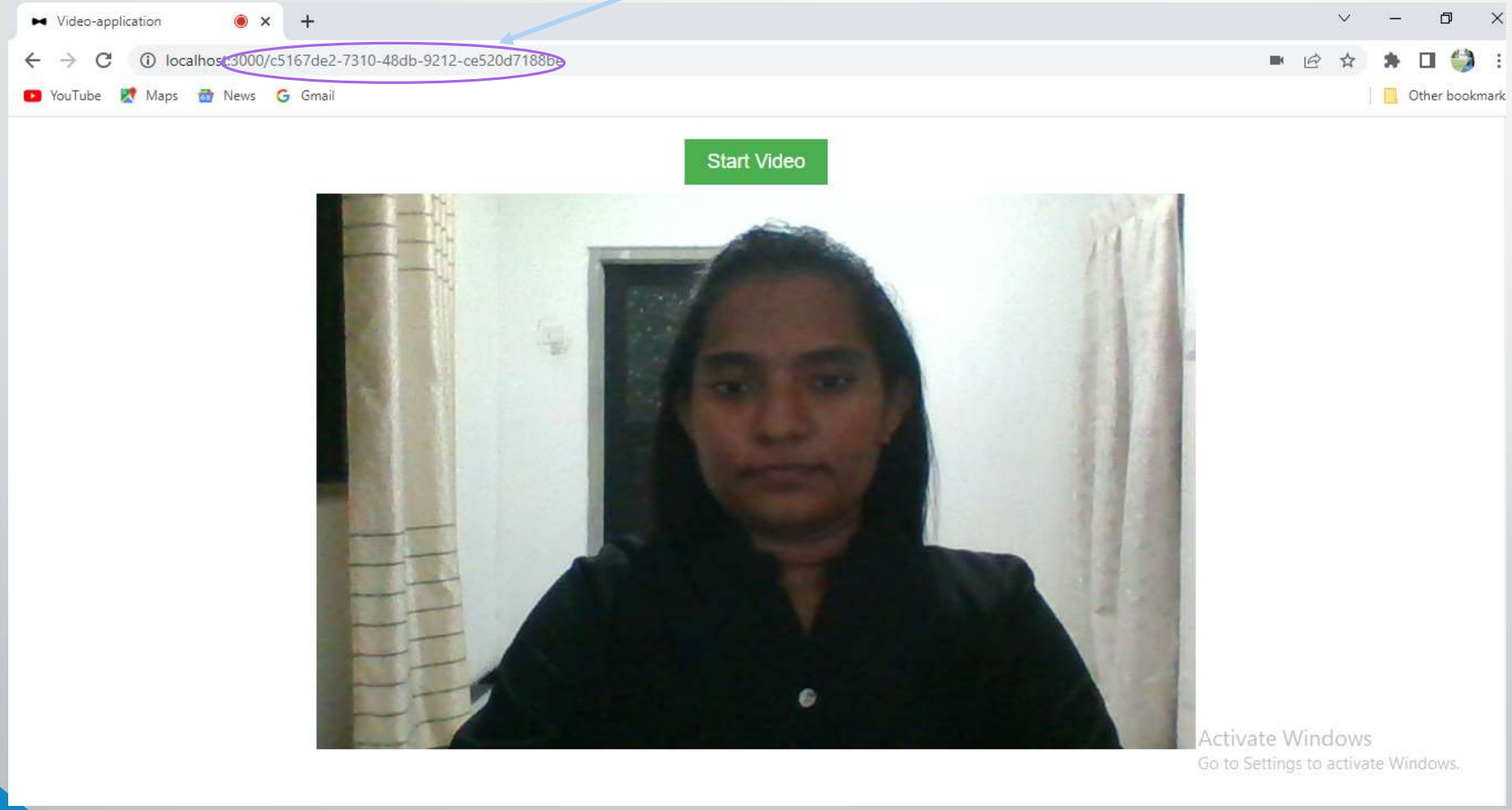
- **Technologies and libraries**

```
{
  "name": "video_application",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  ▷ Debug
  "scripts": {
    "devStart": "nodemon server.js"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "ejs": "^3.1.3",
    "express": "^4.17.1",
    "socket.io": "^2.3.0",
    "uuid": "^8.1.0"
  },
  "devDependencies": {
    "nodemon": "^2.0.4"
  }
}
```

Node.js: A JavaScript runtime environment used to execute server-side JavaScript code.
Express: A web application framework for Node.js that simplifies the development of web applications and APIs.
Socket.IO: A JavaScript library that enables real-time, bidirectional communication between web clients and servers using WebSockets.
EJS: Embedded JavaScript, a templating engine that generates HTML markup with embedded JavaScript code.
UUID

Output of the implementation

Generating video with the id for user



Supportive Information

- Commercialization
- Budget

Commercialization

- The proposed system aims to streamline the registration process for newborns and their parents by utilizing decentralized technology like blockchain. By doing so, the system offers increased security, transparency, and accessibility.
- To meet the functional requirements, the system would need to enable hospitals and healthcare providers to register newborns and parents' details, provide a user-friendly interface for parents to register their information, and ensure that the data is stored accurately on the blockchain.
- Nonfunctional requirements would include performance, scalability, and security, such as ensuring that the system can handle a large volume of registrations and protect data from unauthorized access.
- To commercialize the system, it could be marketed to hospitals and healthcare providers as well as parents who want a secure and accessible way to register their child's details.
- The system could be monetized through subscription or usage fees and could also be integrated with other healthcare systems or applications to provide additional value to users.

Budget

Task	Cost (Rs.)
• Domain Name	5000.00
• Hosting	9000.00
• Backups	5000.00
• Website Strategy	2000.00
• Testing	2000.00
• Other	3000.00
• Maintenance	2000.00
• Marketing	10000.00
Total Cost	**38000.00**

Thank You!

Team TMP 23-103