PRISONPRO

An Innovative Prison Management System

A MINI PROJECT REPORT

Submitted by

ORRIN LEROY H 220701191

NANDHA KUMAR P 220701180

In partial Fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE

RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)

THANDALAM

CHENNAI – 602105

2023-2024

BONAFIDE CERTIFICATE

Certified that this project report " **PRISONPRO : An Innovative Prison Management System** " is the bonafide work of

"**ORRIN LEROY H 220701191 , NANDHA KUMAR P (220701180)** "

Who carried out the project work under my supervision.

Submitted for the Practical Examination held on ___________________

SIGNATURE	**SIGNATURE**
Dr.R.SABITHA	**Mrs.D.KALPANA**
Professor and II Year Academic Head	**Assistant Professor,**
Computer Science and Engineering,	**Computer Science and Engineering,**
Rajalakshmi Engineering College	**Rajalakshmi Engineering College**
(Autonomous),	**(Autonomous),**
Thandalam, Chennai - 602 105	**Thandalam, Chennai - 602 105**

INTERNAL EXAMINER EXTERNAL EXAMINER

ABSTRACT

PrisonPro is a system designed to enhance prison management and improve sentence prediction accuracy. By integrating data analytics and machine learning, it streamlines inmate record management, tracks rehabilitative progress, and optimizes resource allocation. The system accurately forecasts sentence lengths, aiding judicial decisions and ensuring fairer outcomes. It also monitors inmate behaviour, assesses risks, and plans individualized rehabilitation programs. With robust reporting capabilities, PrisonPro provides real-time insights and detailed analytics, promoting efficiency, transparency, and fairness in the correctional system.

TABLE OF CONTENTS

1.INTRODUCTION

1.1 INTRODUCTION

In the evolving landscape of criminal justice, the management of correctional facilities and the accuracy of sentencing predictions are crucial to ensuring both operational efficiency and fairness. The integration of modern technologies such as data analytics, machine learning, and web-based applications can significantly enhance these processes. This project report introduces the **PrisonPro** system, a comprehensive solution designed to address the needs of prison management and sentence prediction.

PrisonPro combines a user-friendly interface with robust backend functionalities to streamline the management of inmate data and provide accurate sentence predictions. By leveraging advanced machine learning models and secure data storage practices, the system aims to improve the overall efficiency of prison operations and support judicial bodies in making informed sentencing decisions.

1.2 OBJECTIVES

The primary objectives of the Prison Management project are as follows:

- **Develop a Secure User Authentication System:**

    i) Implement robust user registration and login functionalities to ensure that only authorized personnel can access the system

    ii) Utilize bcrypt for secure password hashing to protect user credentials.

- **Create a Comprehensive Inmate Data Management System:**

    i) Design a centralized database using MongoDB to store detailed inmate records, including names, crimes committed, ages, and sentences. Allow users to organize recipes into categories for better management.

    ii) Ensure that the model provides accurate and consistent predictions to support fair judicial decision-making.

- **Build an Interactive and User-Friendly Web Interface:**

    i) Utilize Streamlit to create an intuitive web application that allows users to easily navigate and utilize the system's features. Maintain a comprehensive list of ingredients to facilitate recipe creation.

    ii) Ensure that the interface supports seamless interaction for tasks such as adding and viewing inmate records and generating sentence predictions.

- **Integrate Advanced Sentence Prediction Capabilities:**

  i) Develop and implement a machine learning model capable of predicting prison sentence durations based on crime type, severity, and criminal history.Ensure that images are stored and displayed appropriately within the recipe details.

  ii) Ensure that the model provides accurate and consistent predictions to support fair judicial decision-making.

- **Promote Transparency and Fairness in the Justice System:**

  i) Ensure that the sentence prediction model is transparent and based on objective criteria to support fair sentencing outcomes.

  ii) Provide detailed analytics and reporting capabilities to support data-driven decision-making and policy formulation.

- **Ensure Data Security and Privacy:**

  i) Implement stringent security measures to protect sensitive inmate and user data from unauthorized access.

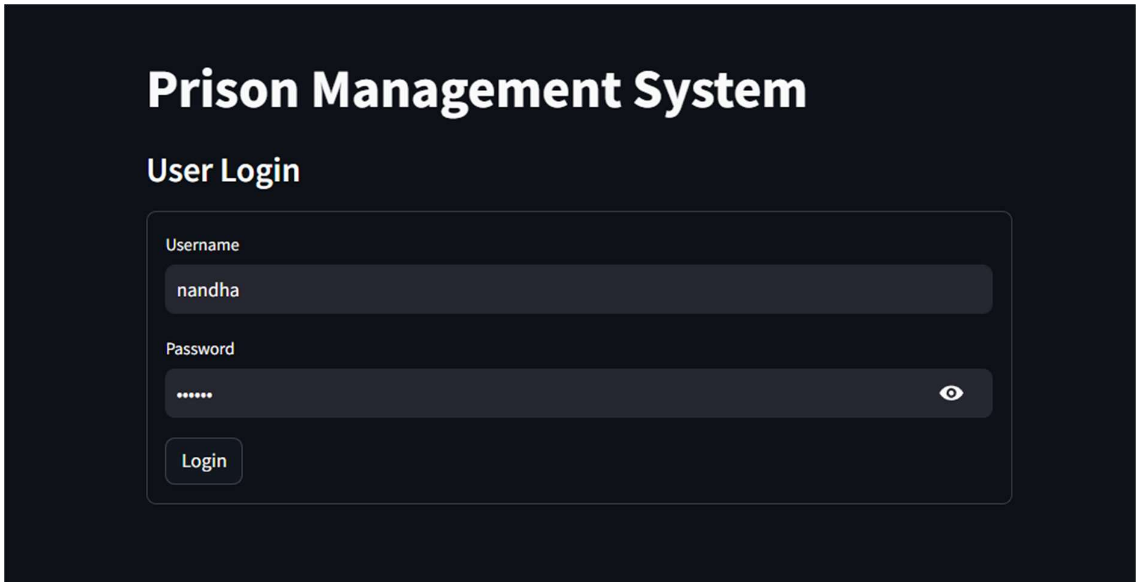
  ii) Comply with relevant data protection regulations to ensure the privacy and confidentiality of all stored information.

1.3 MODULES

The Prison Management project is divided into several modules, each responsible for different aspects of the system. The key modules are as follows:

- **User Authentication Module:**

    i) Registration: Allows new users to create an account by providing their details.
    ii) Login: Authenticates users based on their credentials and provides access to the system.
    iii) Logout: Allows users to securely log out of the application.



Prison Management System

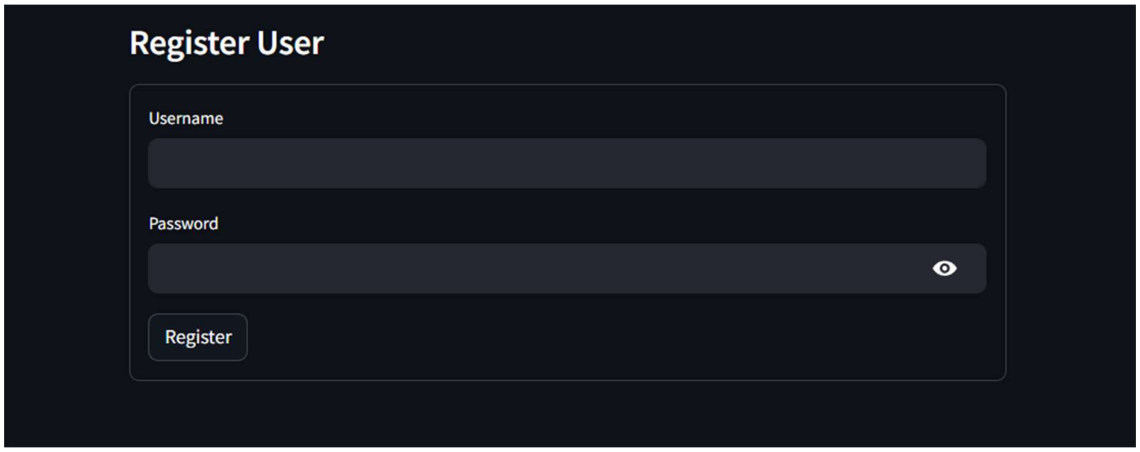
User Login

Username

nandha

Password

••••••



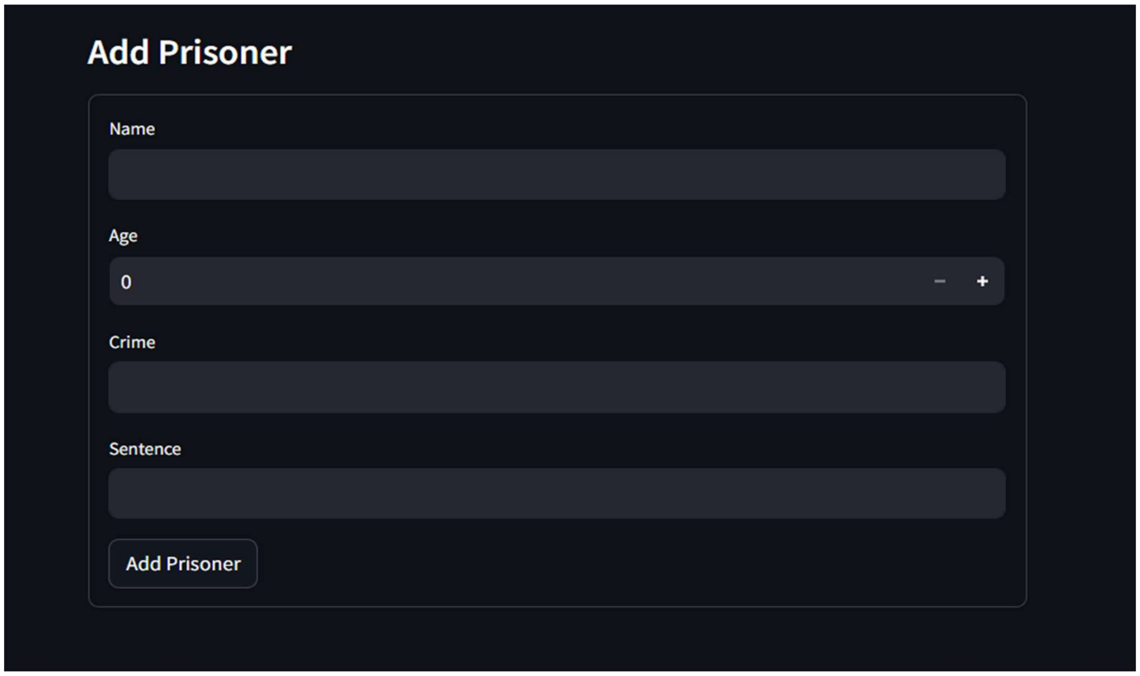
Register User

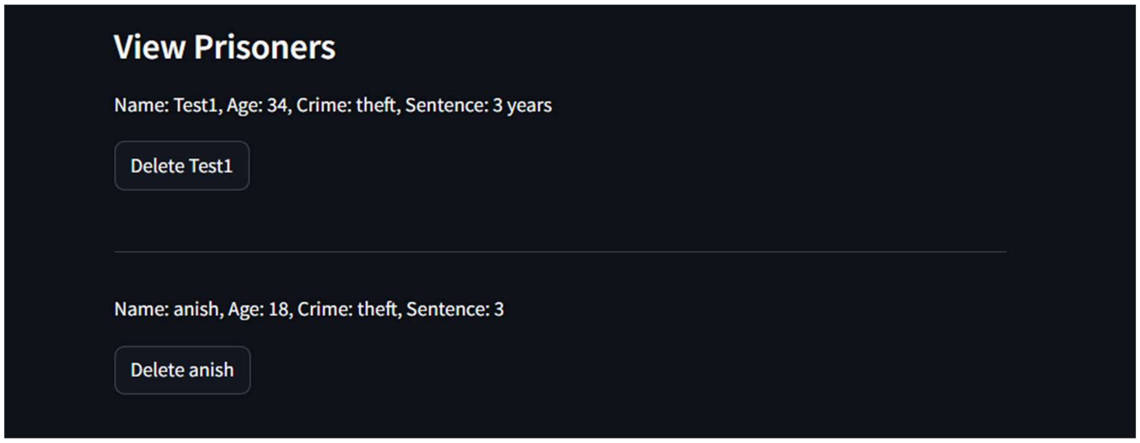
Username

Password

- **Inmate Data Management Module:**

    i) Inmate Records: Stores detailed information about inmates, including names, crimes committed, ages, and sentences.
    ii) Data Entry: Allows authorized personnel to add new inmate records.
    iii) Data Retrieval and Update: Facilitates easy retrieval and updating of inmate information.
    iv) Data Deletion: Enables authorized users to delete inmate records when necessary.



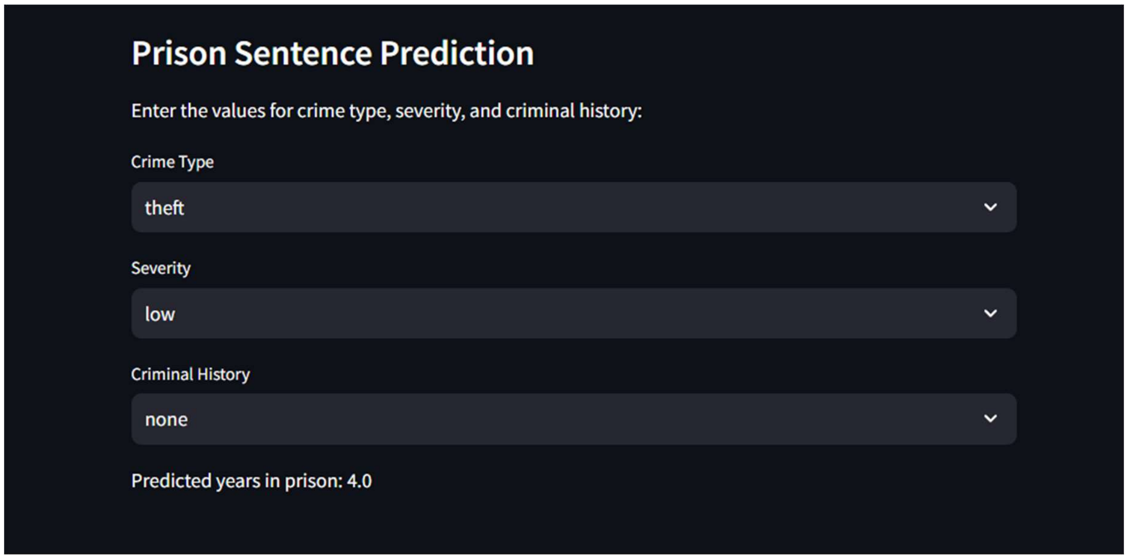

View Prisoners

Name: Test1, Age: 34, Crime: theft, Sentence: 3 years

Name: anish, Age: 18, Crime: theft, Sentence: 3

- **Sentence Prediction Module:**

    i) Input Parameters: Accepts inputs such as crime type, severity, and criminal history.
    ii) Prediction Algorithm: Uses a pre-trained machine learning model to predict the likely sentence duration.
    iii) Prediction Output: Displays the predicted sentence length to the user in an easily interpretable format.


Prison Sentence Prediction

Enter the values for crime type, severity, and criminal history:

Crime Type: theft

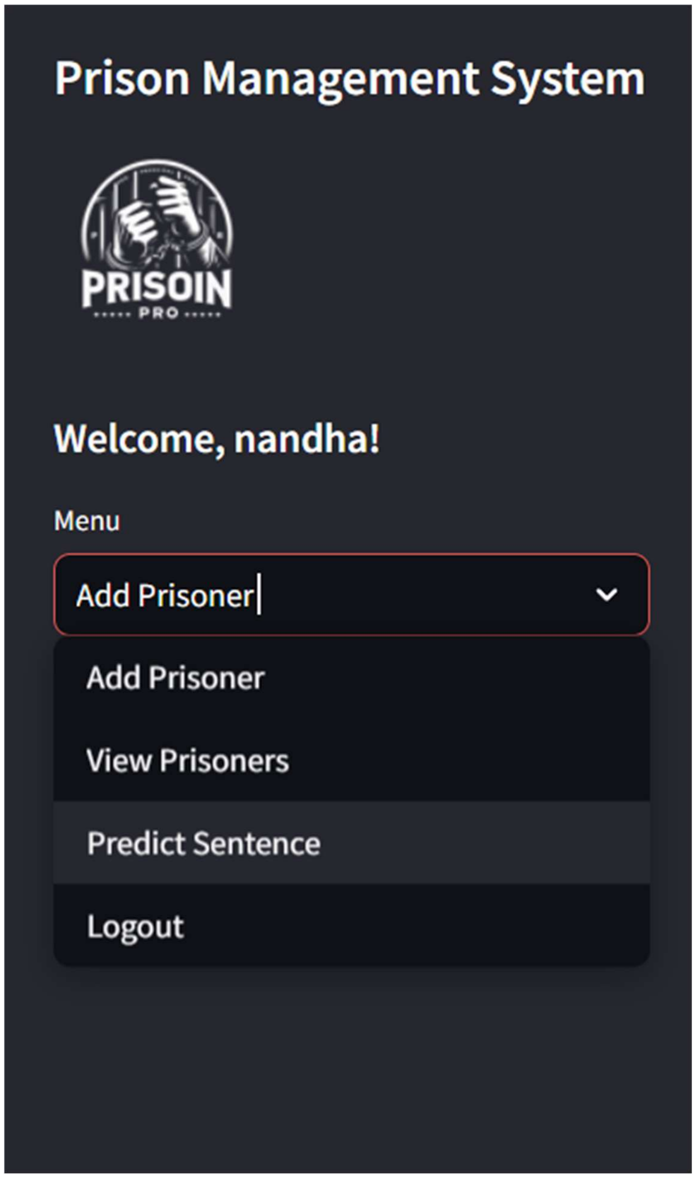
Severity: low

Criminal History: none

Predicted years in prison: 4.0

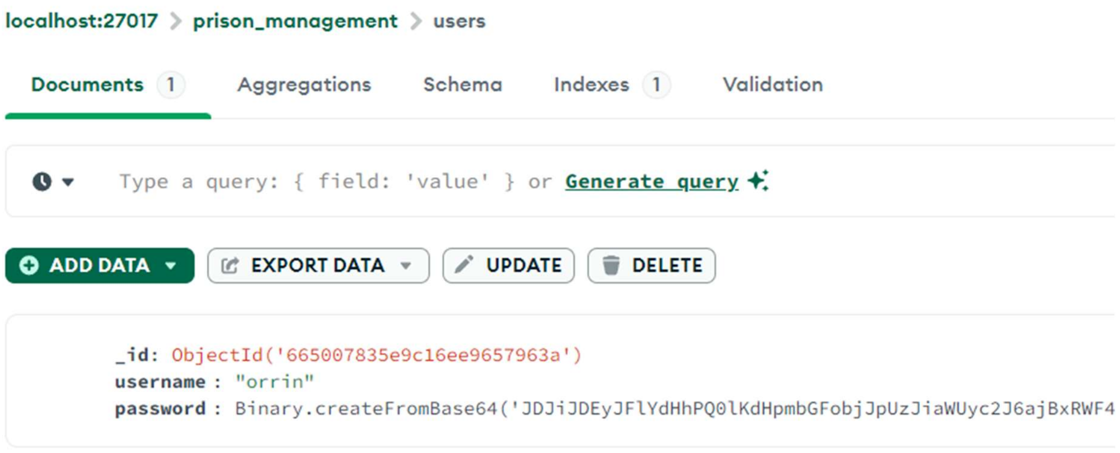
- **Web Interface Module:**

   i) User Dashboard: A central interface where users can access all system functionalities.
   ii) Form Handling: Interactive forms for user registration, login, adding prisoners, and predicting sentences.
   iii) Navigation: Simple navigation to different sections of the application, such as adding/viewing prisoners and predicting sentences



- **Database Management Module:**

  i) Data Storage: Securely stores user and inmate data.
  ii) Query Handling: Efficiently handles database queries for retrieving, updating, and deleting records.
  iii) Data Security: Implements security measures to protect sensitive data


localhost:27017 > prison_management > users
Documents 1
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
ADD DATA
EXPORT DATA
UPDATE
DELETE

```
_id: ObjectId('665007835e9c16ee9657963a')
username : "orrin"
password : Binary.createFromBase64('JDJiJDEyJFlYdHhPQ0lKdHpmbGFobjJpUzJiaWUyc2J6ajBxRWF4
```

2. SURVEY OF TECHNOLOGIES

2.1 SOFTWARE DESCRIPTION

The Prison Management project utilizes various software technologies to ensure a robust and scalable system. The core technologies include Python for server-side scripting, MongoDB for database management.

2.2 LANGUAGES

The Prison Management project leveraged several programming languages and technologies to build the system. This language was chosen for its specific strengths and contributions to different aspects of the project.

2.2.1 PYTHON

PYTHON is a widely-used open-source scripting language suited for web development. It was employed for server-side scripting to handle data processing, database interactions, and dynamic content generation. Key features of PYTHON utilized in this project include:

- Server-side scripting
- Form handling
- Database connectivity using PyMongo
- Session management

LIBRARIES: -

1) STREAMLIT :
   Streamlit is an open-source app framework designed to create and share data applications quickly and effortlessly. It allows users to build custom web applications for machine learning and data science projects with minimal coding. Streamlit apps are created using Python scripts, enabling rapid prototyping and seamless integration of interactive widgets and visualizations.

2) TENSORFLOW-KERAS :
   TensorFlow Keras is a high-level API for building and training deep learning models within the TensorFlow framework. It simplifies the creation of neural networks by providing intuitive and user-friendly tools for defining, training, and evaluating machine learning models

3) BCRYPT :
   Bcrypt is a password-hashing function designed to be computationally intensive, providing enhanced security against brute-force attacks. It automatically handles salt generation and includes an adjustable work factor to adapt to increasing computational power over time.

4) PYMONGO :
   PyMongo is a Python distribution containing tools for working with MongoDB, a NoSQL database. It provides an intuitive and flexible interface for connecting to MongoDB, performing CRUD operations, and managing collections and documents.

5) NUMPY :
   NumPy is a fundamental package for scientific computing with Python, offering powerful array operations and mathematical functions. It provides a convenient interface for handling large datasets efficiently and performing complex numerical computations with ease.

2.2.2 TOOLS USED: -

1) MONGODB: -

MongoDB is a leading NoSQL database renowned for its flexibility, scalability, and ease of use. Its document-oriented structure allows for storing data in a JSON-like format, enabling developers to work with dynamic schemas effortlessly. With features like high availability, horizontal scalability, and robust query capabilities, MongoDB is a preferred choice for modern applications handling large volumes of data, real-time analytics, and geospatial queries.

2) VISUAL STUDIO CODE: -

Visual Studio Code (VS Code) is a lightweight, open-source code editor developed by Microsoft, renowned for its versatility and extensive feature set. It offers a customizable and intuitive interface with support for various programming languages, extensions, and debugging tools. With features like IntelliSense, Git integration, and built-in terminal, VS Code enhances developer productivity, making it a favourite among developers for building web, mobile, and cloud applications.

3) GITHUB: -

GitHub is a web-based platform for version control using Git, enabling collaboration and code sharing among developers worldwide. It provides a centralized repository for managing and tracking changes to code, facilitating seamless collaboration through features like pull requests, code reviews, and issue tracking. With its vast community, robust documentation, and integration with popular development tools, GitHub has become an essential platform for software development, fostering innovation and open-source contributions across various domains.

3.REQUIREMENTS AND ANALYSIS

3.1 REQUIREMENT SPECIFICATION

- User Requirements:

  The system should be accessible via web browsers and mobile devices. Users should be able to create, edit, and delete inmate details, add inmates, and predict sentence.

- System Requirements:

  The system should ensure data integrity and provide a secure environment for user information.

3.2 HARDWARE AND SOFTWARE REQUIREMENTS

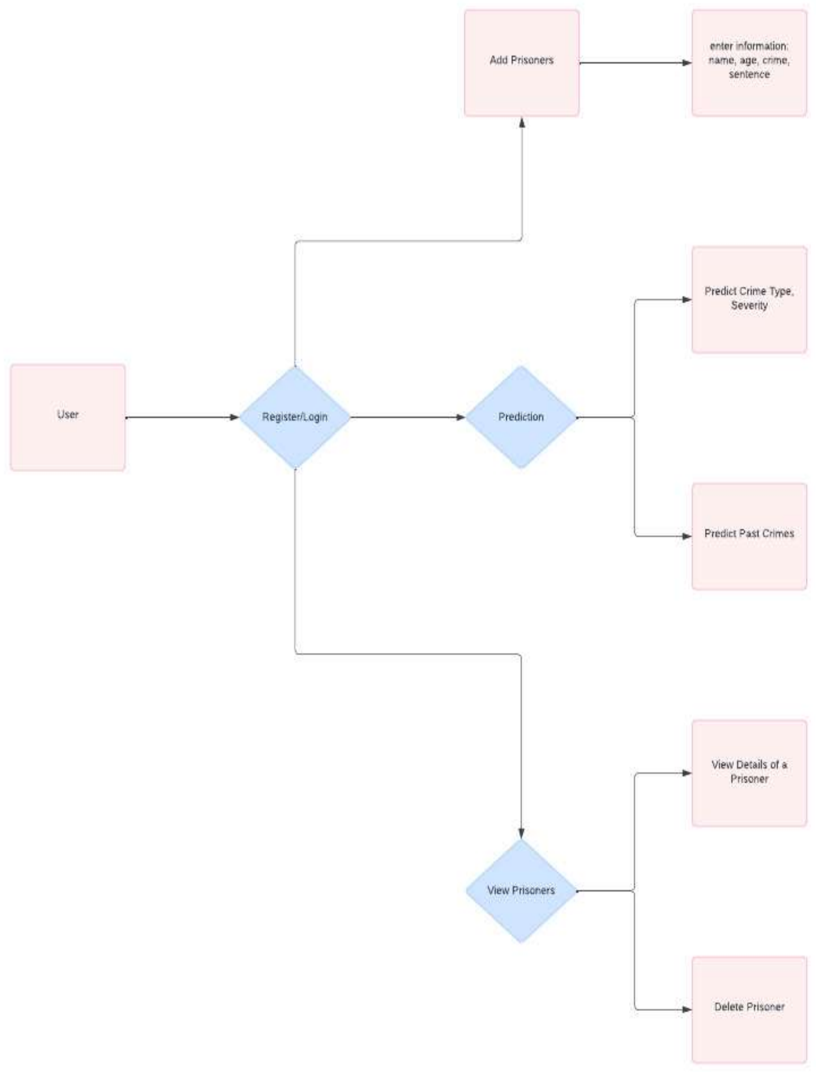
- Hardware:

  i) Server with minimum 4GB RAM and 100GB storage.
  ii) Client devices with internet access.

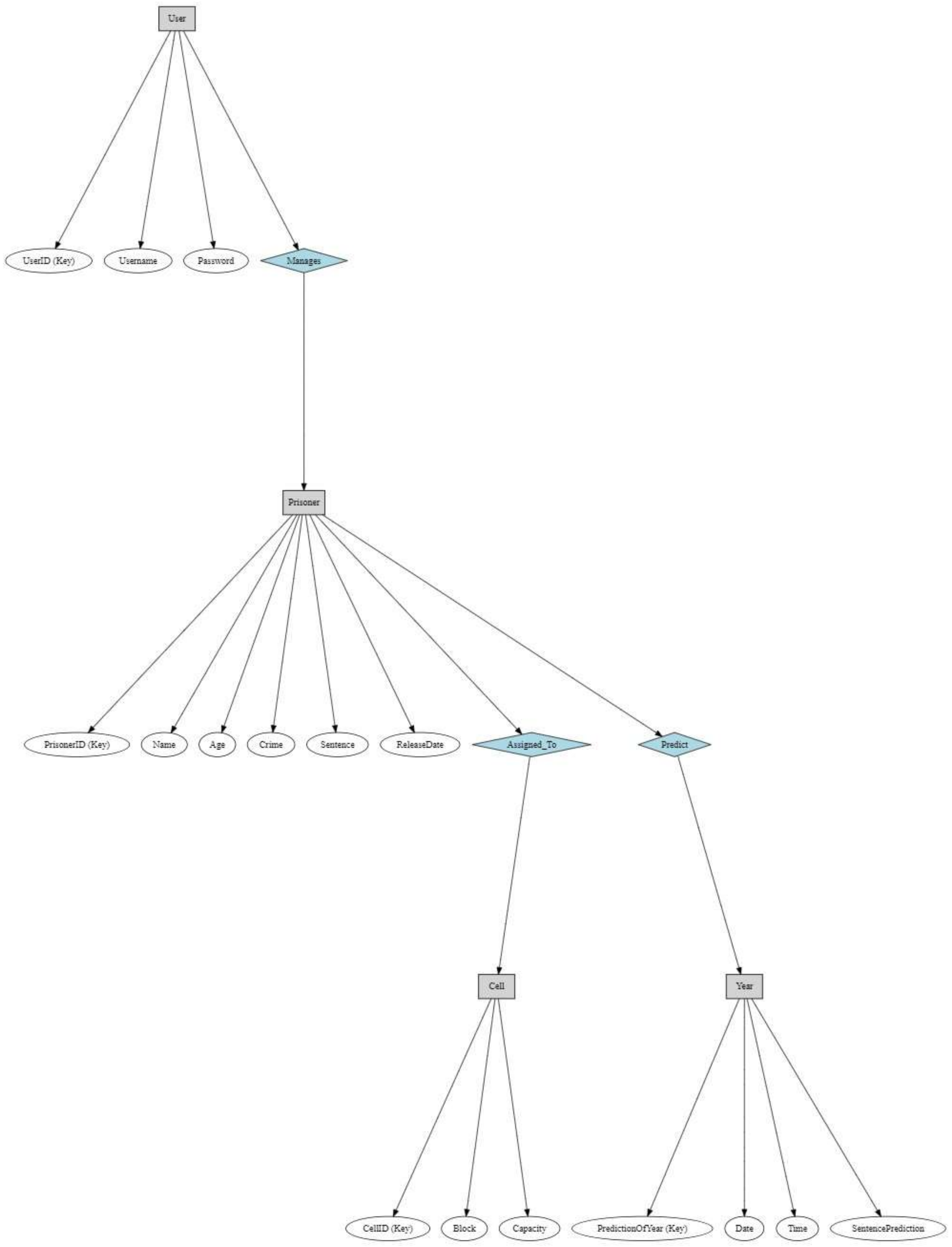
- Software:

  i) Operating System: Linux/Windows
  ii) Web Server: Apache
  iii) Database: MongoDB
  iv) Languages: Python
  v) Libraries: Streamlit, Bcrypt, Tensorflow Keras, Numpy, PyMongo

3.3 ARCHITECTURE DIAGRAM


User
Register/Login
Add Prisoners
enter information: name, age, crime, sentence
Prediction
Predict Crime Type, Severity
Predict Past Crimes
View Prisoners
View Details of a Prisoner
Delete Prisoner

3.4 ER DIAGRAM



4.PROGRAM CODE

4.1 Deep Learning Model:-

Pre-Processing:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler

data = pd.read_csv('prison_data.csv')

# Encode categorical variables
label_encoder = LabelEncoder()
data['crime_type'] = label_encoder.fit_transform(data['crime_type'])
data['severity'] = label_encoder.fit_transform(data['severity'])
data['criminal_history'] = label_encoder.fit_transform(data['criminal_history'])

X = data[['crime_type', 'severity', 'criminal_history']]
y = data['years']

# Split into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Standardize the features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Building;-

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

# Build the neural network model
model = Sequential([
    Dense(64, input_dim=X_train.shape[1], activation='relu'),
    Dense(32, activation='relu'),
    Dense(1)
])

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')
```

```python
# Summary of the model
model.summary()
```

Training and Evaluatiion:-

```python
# Train the model
history = model.fit(X_train, y_train, epochs=50, batch_size=10, validation_split=0.2)

# Evaluate the model
loss = model.evaluate(X_test, y_test)
print(f'Test Loss: {loss}')

# Predict using the model
y_pred = model.predict(X_test)

# Optionally, compare predicted vs actual values
import matplotlib.pyplot as plt

plt.scatter(y_test, y_pred)
plt.xlabel('Actual Years')
plt.ylabel('Predicted Years')
plt.title('Actual vs Predicted Years in Prison')
plt.show()
```

Execution:-

```python
import numpy as np


input_data = np.array([[1, 0, 0]])  # 2D array with shape (1, 3)
prediction = model.predict(input_data)
prediction = np.round(prediction)
print(prediction)
```

4.2 WebPage:-

```
import streamlit as st
from pymongo import MongoClient
import bcrypt
from tensorflow.keras.models import load_model
import numpy as np

# Connect to MongoDB
client = MongoClient('mongodb://localhost:27017/')
db = client['prison_management']
users_collection = db['users']
prisoners_collection = db['prisoners']

# Load the deep learning model
model = load_model('my_model.h5')

# User registration
def register_user():
    st.subheader('Register User')
    with st.form(key='register_form'):
        username = st.text_input('Username')
        password = st.text_input('Password', type='password')
        submit_button = st.form_submit_button(label='Register')
        if submit_button:
            if users_collection.find_one({'username': username}):
                st.error('Username already exists!')
            else:
                hashed_password = bcrypt.hashpw(password.encode('utf-8'), bcrypt.gensalt())
                user_data = {
                    'username': username,
                    'password': hashed_password
                }
                users_collection.insert_one(user_data)
                st.success('User registered successfully!')

# User login
def login_user():
    st.subheader('User Login')
    with st.form(key='login_form'):
        username = st.text_input('Username')
        password = st.text_input('Password', type='password')
        submit_button = st.form_submit_button(label='Login')
        if submit_button:
            user = users_collection.find_one({'username': username})
            if user and bcrypt.checkpw(password.encode('utf-8'), user['password']):
                st.session_state['logged_in'] = True
                st.session_state['username'] = username
```

```python
                st.success('Login successful!')
                st.experimental_rerun()
            else:
                st.error('Invalid username or password')

# Add a prisoner
def add_prisoner():
    st.subheader('Add Prisoner')
    with st.form(key='add_prisoner_form'):
        name = st.text_input('Name')
        age = st.number_input('Age', min_value=0, max_value=150)
        crime = st.text_input('Crime')
        sentence = st.text_input('Sentence')
        submit_button = st.form_submit_button(label='Add Prisoner')
        if submit_button:
            prisoner_data = {
                'name': name,
                'age': age,
                'crime': crime,
                'sentence': sentence
            }
            prisoners_collection.insert_one(prisoner_data)
            st.success('Prisoner added successfully!')

# View all prisoners
def view_prisoners():
    st.subheader('View Prisoners')
    prisoners = prisoners_collection.find()
    prisoners_list = list(prisoners)

    if len(prisoners_list) == 0:
        st.write('No prisoners found.')
    else:
        for prisoner in prisoners_list:
            st.write(f"Name: {prisoner['name']}, Age: {prisoner['age']}, Crime: {prisoner['crime']}, Sentence: {prisoner['sentence']}")
            if st.button(f"Delete {prisoner['name']}", key=prisoner['_id']):
                prisoners_collection.delete_one({'_id': prisoner['_id']})
                st.experimental_rerun()
            st.write('---')

# Predict prison sentence
def predict_sentence():
    st.subheader("Prison Sentence Prediction")
    st.write("Enter the values for crime type, severity, and criminal history:")

    crime_type_options = ["theft", "assault", "robbery"]
    severity_options = ["low", "medium", "high"]
    criminal_history_options = ['none', 'minor', 'repeat']

    crime_type = st.selectbox("Crime Type", options=crime_type_options, index=0)
    severity = st.selectbox("Severity", options=severity_options, index=0)
    criminal_history = st.selectbox("Criminal History", options=criminal_history_options, index=0)

    input_data = np.array([[crime_type_options.index(crime_type), severity_options.index(severity),
criminal_history_options.index(criminal_history)]])

    prediction = model.predict(input_data)
    prediction = np.round(prediction)

    st.write(f"Predicted years in prison: {prediction[0][0]}")

# Main function
def main():
    st.title('Prison Management System')

    if 'logged_in' not in st.session_state:
        st.session_state['logged_in'] = False

    if st.session_state['logged_in']:
        st.sidebar.header(f"Welcome, {st.session_state['username']}!")
        menu_options = ['Add Prisoner', 'View Prisoners', 'Predict Sentence', 'Logout']
        selected_menu = st.sidebar.selectbox('Menu', menu_options)

        if selected_menu == 'Add Prisoner':
            add_prisoner()
        elif selected_menu == 'View Prisoners':
            view_prisoners()
        elif selected_menu == 'Predict Sentence':
            predict_sentence()
        elif selected_menu == 'Logout':
            st.session_state['logged_in'] = False
            st.experimental_rerun()
    else:
        auth_options = ['Login', 'Register']
        selected_auth = st.sidebar.selectbox('Authenticate', auth_options)

        if selected_auth == 'Login':
            login_user()
        elif selected_auth == 'Register':
            register_user()

if __name__ == '__main__':
    main()
```

5.EXISTING SOFTWARES

There are several prison management software solutions currently in use, designed to help correctional facilities manage inmate information, staff operations, security, and administrative tasks. Here are some examples:

- **Offender360: -**

  Features: This software provides comprehensive management of inmate records, including booking, classification, housing, and release. It also integrates with other law enforcement systems for better data sharing and operational efficiency.

  - **Use Case:** Typically used in large correctional facilities and government agencies.

- **JailTracker: -**

  Features: JailTracker offers modules for inmate booking, facility management, commissary operations, and medical records. It also includes reporting tools and integrates with other criminal justice systems.

  - **Use Case:** Popular in small to medium-sized jails and detention centers.

- **eOMIS:**

  Features: Developed by Marquis Software, eOMIS covers a range of functions including inmate tracking, medical records management, case management, and parole/probation monitoring.

  - **Use Case:** Widely used by state departments of corrections and large correctional facilities.

- **Spillman Flex:**

Features: This software includes jail management, records management, and computer-aided dispatch. It provides tools for inmate tracking, incident reporting, and resource management.

  - **Use Case:** Often used by sheriff's offices and law enforcement agencies that manage jails and detention centers.

- **Guardian RFID:**

Features: Focuses on real-time inmate tracking and management using RFID technology. It includes tools for inmate headcounts, activity logging, and compliance monitoring.

  - **Use Case:** Often used by sheriff's offices and law enforcement agencies that manage jails and detention centers.

6.NEW ADDITION

PrisonPro has recently added an innovative feature for sentence prediction using AI technology. This new functionality leverages machine learning models to estimate sentence lengths based on various factors, such as crime severity, criminal history, and jurisdictional guidelines. The aim is to provide more consistent and transparent sentencing outcomes. This feature is part of a broader effort to integrate advanced analytics into the criminal justice system, enhancing decision-making processes and improving fairness in sentencing

- **Enhanced Inmate Search**:
  The updated inmate search tool now offers more precise and faster search capabilities, allowing users to locate inmates more efficiently across various correctional facilities in the United States.

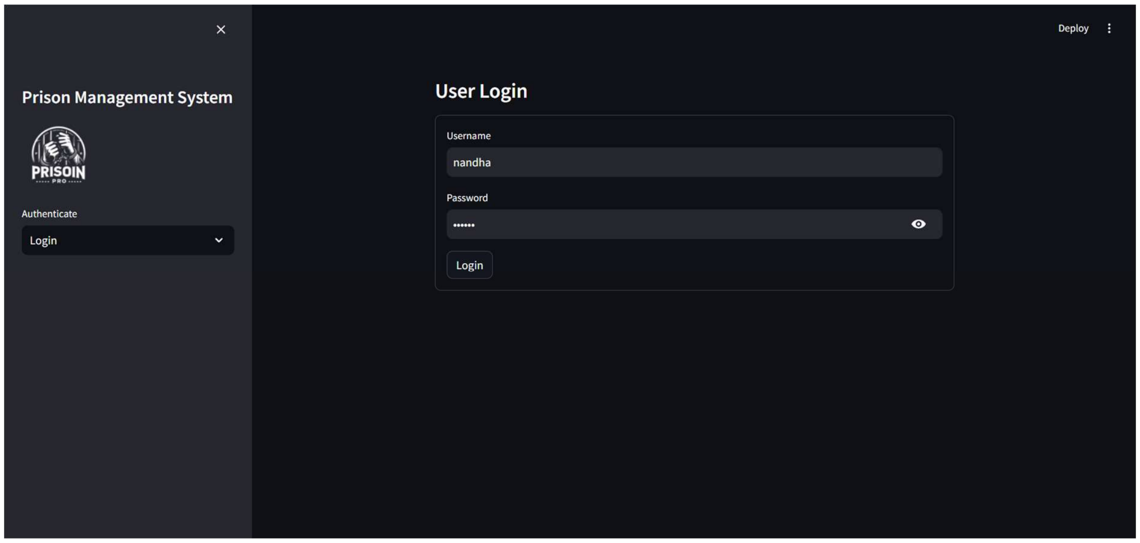
- **Updated Visiting Information**:
  PrisonPro has expanded its database of visiting hours and application procedures for correctional facilities, making it easier for families to plan their visits. This includes detailed instructions on bringing minors to visitations and updated mailing addresses for sending packages and letters to inmates.

- **Improved User Interface**:
  The website has undergone a redesign to offer a more user-friendly interface. This includes easier navigation through different sections such as sending money, setting up phone accounts, and mailing guidelines.

- **Community Support and Resources**:
  PrisonPro has bolstered its community support features, providing a platform for users to ask questions and seek advice from others in similar situations. This support network aims to help users navigate the complexities of incarceration rules and procedures.

7. RESULTS AND DISCUSSION

USER LOGIN INTERFACE

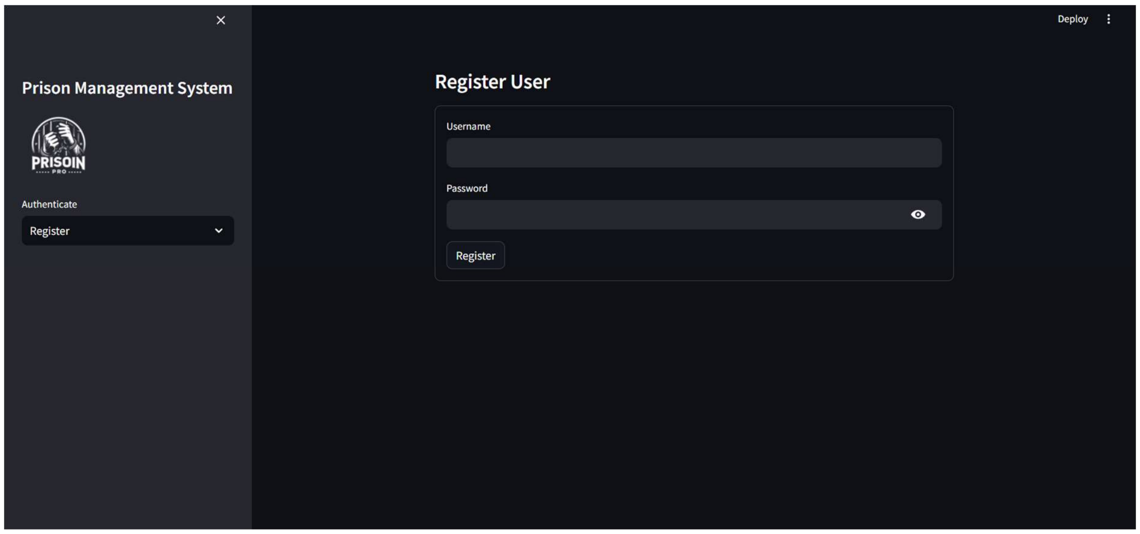


User Login

Username: nandha

Password: ••••••

USER INTERFACE

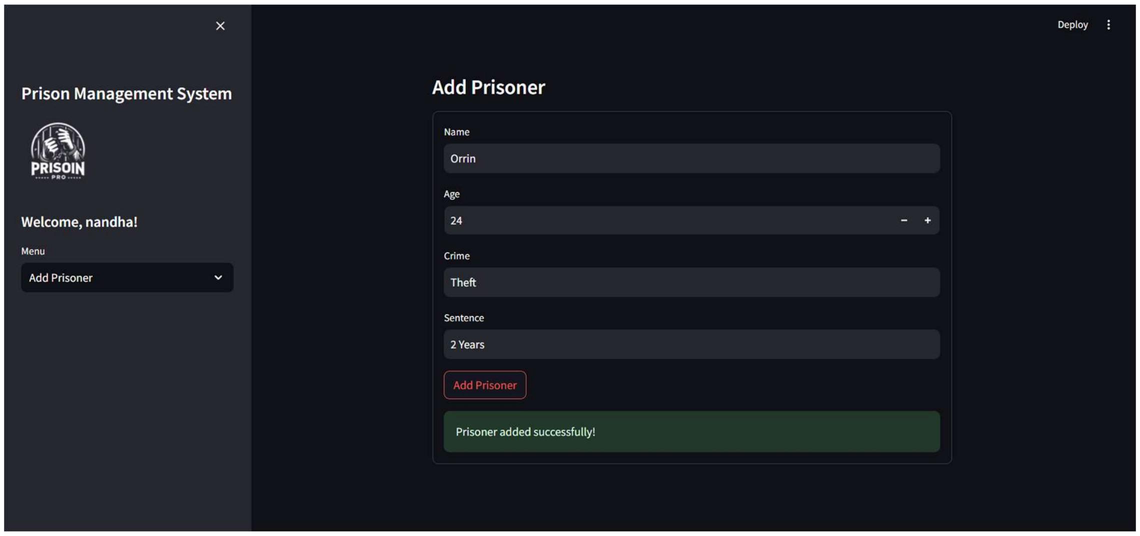


Register User

Username

Password

ADD PRISONER INTERFACE



Welcome, nandha!

Add Prisoner

Name: Orrin

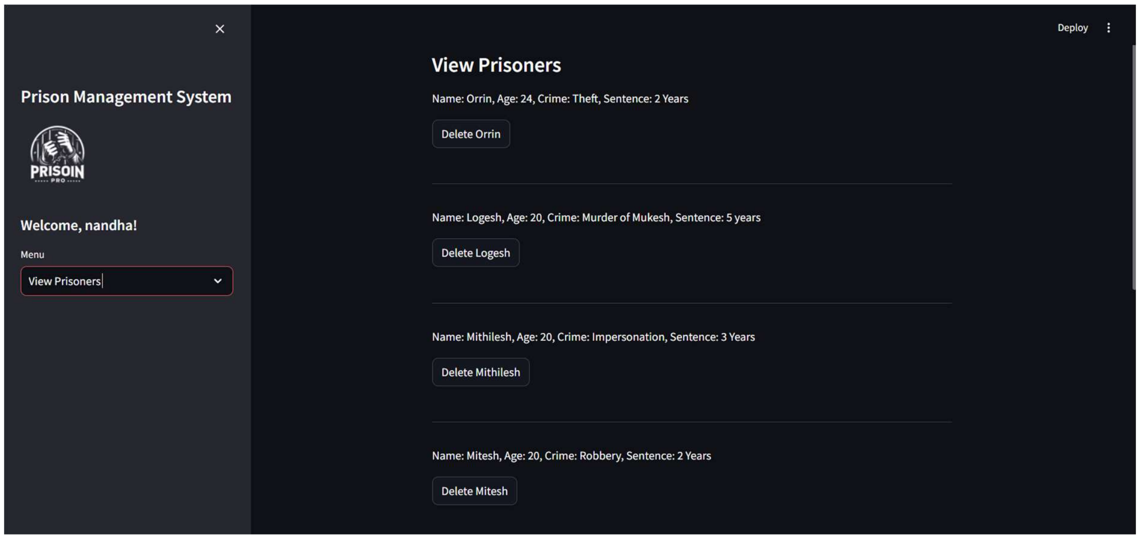
Age: 24

Crime: Theft

Sentence: 2 Years

Prisoner added successfully!

VIEW PRISONER INTERFACE



Welcome, nandha!

View Prisoners

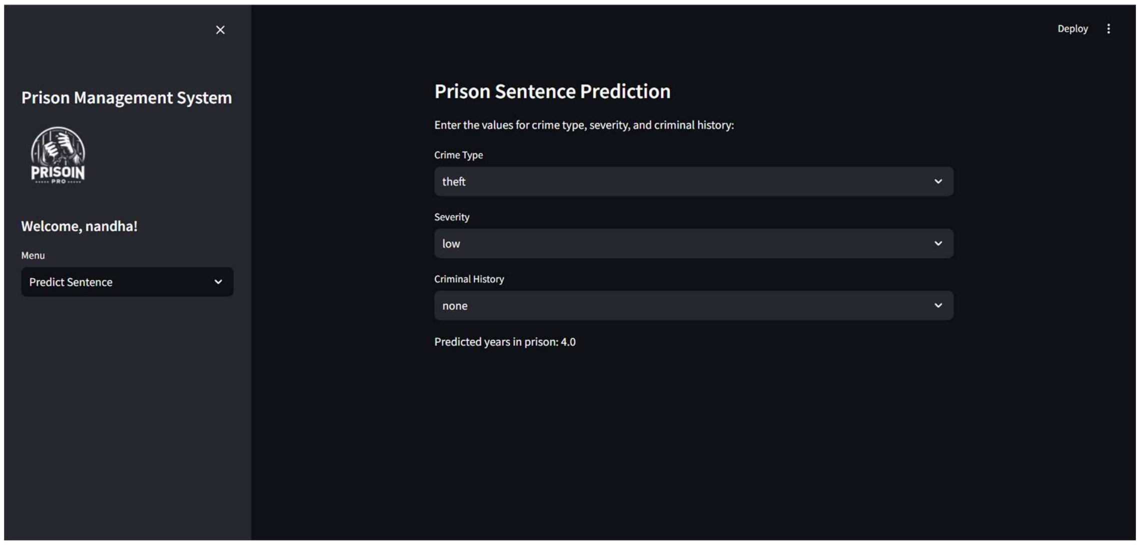
Name: Orrin, Age: 24, Crime: Theft, Sentence: 2 Years

Name: Logesh, Age: 20, Crime: Murder of Mukesh, Sentence: 5 years

Name: Mithilesh, Age: 20, Crime: Impersonation, Sentence: 3 Years

Name: Mitesh, Age: 20, Crime: Robbery, Sentence: 2 Years

PREDICT SENTENCE INTERFACE



Prison Sentence Prediction

Enter the values for crime type, severity, and criminal history:

Crime Type: theft

Severity: low

Criminal History: none

Predicted years in prison: 4.0

8.CONCLUSION

PrisonPro is designed to streamline and enhance the management of correctional facilities through the effective use of modern technology. This comprehensive system covers all critical aspects, including inmate management, security monitoring, staff coordination, and visitor tracking.

The architecture of PrisonPro is built to be resilient and adaptable, allowing correctional facilities to operate efficiently and respond promptly to various situations. The integration of advanced analytics and machine learning capabilities further aids in making informed decisions and anticipating potential issues.

In conclusion, PrisonPro represents a significant step forward in prison management, offering a secure, efficient, and scalable solution that addresses the complex needs of modern correctional facilities. By adopting this system, facilities can ensure better compliance, safety, and operational efficiency, ultimately contributing to a more orderly and secure environment for inmates and staff alike.

9.REFERENCE

Below is a list of references and resources used during the development of the Recipe Cookbook project. These include documentation, libraries, frameworks, and tools that were essential for building and testing the application.

- **Python Documentation:**

Official Python Documentation: https://docs.python.org/3/

- **MongoDB Documentation:**

Official MongoDB Documentation: https://www.mongodb.com/docs/

- **Tensorflow Documentation:**

Official Tensorflow Documentation: https://www.tensorflow.org/api_docs

- **Bcrypt:**

Bcrypt for Python: https://pypi.org/project/bcrypt/

- **Numpy Documentation:**

Official numpy for Python: https://numpy.org/doc/

- **Datacamp :**

Streamlit tutorial: https://numpy.org/doc/

- **Datasets for training:**

Kaggle: https://www.kaggle.com/datasets