**Sleep Disorder Diagnosis Using Advanced Machine Learning Techniques**

|  |
| --- |
| **CONTENT** |
| **Abstract** |
| **1.INTRODUCTION** |
| 1.1 Motivation |
| 1.2 Problem Statement |
| 1.3 Objective of the Project |
| 1.4 Scope |
| 1.5 Project Introduction |
| **2.LITERATURE SURVEY** |
| 2.1 Related Work |
| **3. SYSTEM ANALYSIS** |
| 3.1 Existing System |
| 3.2 Disadvantages |
| 3.3 Proposed System |
| 3.4 Advantages |
| 3.5 work Flow of Proposed system |
| **4. REQUIREMENT ANALYSIS** |
| 4.1Function and non-functional requirements |
| 4.2 Hardware Requirements |
| 4.3 Software Requirements |
| 4.4 Architecture |
| **5. METHODOLOGY** |
| 5.1 KNN |
| 5.2 SVM |
| 5.3 Decision Tree |
| 5.4 Random Forest |
| 5.5 ANN |
| 5.6 Stacking classifier |
| 5.7 Voting classifier |
| **6. SYSTEM DESIGN** |
| 6.1 Introduction of Input design |
| 6.2 UML Diagram (class, use case, sequence, collaborative, deployment, activity, ER diagram and Component diagram) |
| 6.3 Data Flow Diagram |
| **7. IMPLEMENTATION AND RESULTS** |
| 7.1 Modules |
| 7.2 Output Screens |
| **8.SYSTEM STUDY AND TESTING** |
| 9.1 Feasibility study |
| 9.2 Types of test & Test Cases |
| **10. CONCLUSION** |
| **11. FUTURE ENHANCEMENT** |
| **12. REFERENCES** |

**ABSTRACT**

Sleep Disorder significantly impact physical and mental health, necessitating accurate and accessible diagnostic methods. Traditional diagnostic techniques like Polysomnography (PSG) are often inconvenient, expensive, and limited in availability. This project aims to leverage machine learning algorithms to classify sleep Disorder using health and lifestyle data from the Kaggle Sleep Health and Lifestyle Dataset. The existing system employs algorithms such as K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree, Random Forest, and Artificial Neural Network (ANN), which have several limitations including computational expense and sensitivity to hyperparameters. To address these issues, the proposed system implements ensemble learning techniques, specifically Stacking Classifier and Voting Classifier, to enhance accuracy, robustness, and interpretability. By combining the strengths of multiple models, the project seeks to provide a more efficient, cost-effective, and accessible solution for diagnosing sleep Disorder, ultimately improving patient outcomes and quality of life.

**Keywords:** Sleep Disorder, No Sleep Disorder, Machine Learning, Stacking Classifier, Voting Classifier.

**INTRODUCTION**

**1.1 OBJECTIVE OF PROJECT:**

The purpose of this project is to accurately classify individuals as having a sleep disorder or not, using advanced machine learning algorithms to improve accessibility and diagnostic efficiency.

**1.2 PROBLEM STATEMENT:**

Sleep Disorder affect millions globally, posing significant health risks and impacting quality of life. Traditional diagnostic methods like Polysomnography (PSG) are costly, inconvenient, and limited in accessibility, requiring overnight stays in specialized centers. These methods also involve extensive preparation and follow-up, adding to patient burden. There is a critical need for an efficient, cost-effective, and accessible diagnostic system. This project aims to address this need by developing a machine learning-based classification system using health and lifestyle data to accurately identify sleep Disorder, providing a more practical and widely available diagnostic solution.

**1.3 MOTIVATION:**

The motivation for this project stems from the growing prevalence of sleep Disorder and the limitations of traditional diagnostic methods like Polysomnography (PSG), which are inconvenient, expensive, and not widely accessible. Many individuals suffer from undiagnosed sleep Disorder due to these barriers. By utilizing advanced machine learning algorithms and ensemble techniques, we aim to develop a more accessible, cost-effective, and efficient diagnostic tool. This project aspires to improve early detection and treatment of sleep Disorder, enhancing patient outcomes and overall quality of life, and contributing to the broader field of healthcare innovation.

**1.4 SCOPE:**

The scope of this project includes developing a machine learning-based system to classify sleep Disorder using health and lifestyle data. The system will utilize both traditional algorithms (KNN, SVM, Decision Tree, Random Forest, ANN) and advanced ensemble learning techniques (Stacking Classifier, Voting Classifier) to enhance diagnostic accuracy and robustness. The project involves data collection, preprocessing, model training, evaluation, and optimization. It aims to provide a more accessible and cost-effective diagnostic tool compared to traditional methods, ultimately improving the detection and management of sleep Disorder for a broader population. Future work may explore incorporating deep learning models and real-time applications.

**1.5 PROJECT INTRODUCTION:**

Sleep Disorder are prevalent health issues that significantly affect millions of individuals worldwide, leading to adverse effects on physical and mental health. Disorder can cause chronic fatigue, reduced cognitive function, and increased risk of various medical conditions. Accurate diagnosis and classification of sleep Disorder are crucial for effective treatment and management. However, traditional diagnostic methods, such as Polysomnography (PSG), present several challenges that limit their accessibility and efficiency.

Polysomnography, considered the gold standard for diagnosing sleep Disorder, requires patients to stay overnight in specialized sleep laboratories. This procedure involves significant preparation, follow-up visits, and the use of costly equipment and specialized healthcare personnel. Additionally, PSG is often limited to specialized centers, making it inaccessible to individuals in remote or underserved areas. The high costs associated with PSG and long waiting times further complicate timely diagnosis and treatment, often leading to delays in patient care.

To address these limitations, this project aims to apply machine learning algorithms for the classification of sleep Disorder using health and lifestyle data. By leveraging advanced machine learning techniques, we can develop a more accessible, cost-effective, and efficient system for diagnosing sleep Disorder. The dataset used in this project is sourced from Kaggle and includes various health and lifestyle factors that influence sleep patterns, providing a comprehensive foundation for machine learning models.

The existing system employs traditional machine learning algorithms such as K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree, Random Forest, and Artificial Neural Network (ANN). These algorithms have been widely used due to their simplicity and effectiveness in handling various classification tasks. However, they come with several disadvantages, including computational expense, sensitivity to hyperparameters, risk of overfitting, and lack of interpretability.

To overcome these challenges, the proposed system implements ensemble learning techniques, specifically the Stacking Classifier and Voting Classifier. Ensemble learning methods combine the strengths of multiple base models to enhance overall performance and robustness. The Stacking Classifier involves training several base models and then using another model to combine their predictions. This method leverages the individual strengths of different algorithms to achieve better predictive performance. The Voting Classifier aggregates the predictions of multiple models through majority voting or averaging, providing a more stable and accurate classification outcome.

The primary objective of this project is to develop a machine learning-based system that accurately classifies whether an individual has a sleep disorder or not, based on health and lifestyle data. By implementing advanced ensemble learning techniques, the proposed system aims to provide a more reliable and interpretable solution, addressing the limitations of traditional diagnostic methods and enhancing accessibility for a broader population. The scope of this project includes data collection, preprocessing, model training, evaluation, and optimization, with future work potentially exploring the incorporation of deep learning models and real-time applications.

Ultimately, this project seeks to improve the detection and management of sleep Disorder, contributing to better patient outcomes and quality of life through innovative and accessible machine learning solutions.**2. LITERATURE SURVEY**

**2.1 Related work:**

**[1] C. Tran, Y. Wijesuriya, R. Thuraisingham, A. Craig, & H. Nguyen, "Deep Learning for Classification of Sleep Stages," 2019.**

This study focuses on the application of deep learning techniques for the classification of sleep stages using EEG data. The authors emphasize the potential of deep learning models to handle complex and high-dimensional sleep data, achieving high classification accuracy. Their approach demonstrates significant improvements over traditional methods, providing a strong foundation for applying deep learning in sleep disorder classification.

**[2] E. Alickovic & A. Subasi, "Ensemble SVM Method for Automatic Sleep Stage Classification," IEEE Transactions on Instrumentation and Measurement, 2018.**

This paper discusses the use of ensemble support vector machine (SVM) methods for automatic sleep stage classification. The authors address the challenges of accurately classifying sleep stages and demonstrate the effectiveness of ensemble methods in improving classification accuracy. Their findings highlight the potential of ensemble learning techniques in the field of sleep disorder diagnosis.

**[3] M. J. Sun, Z. F. Wu, & X. B. Lu, "Sleep Apnea Detection Based on Time and Frequency Domain Analysis of ECG and SpO2 Signals," 2020.**

The authors propose a method for detecting sleep apnea using time and frequency domain analysis of ECG and SpO2 signals. By applying machine learning algorithms to these physiological signals, they achieve high accuracy in identifying sleep apnea events. This research underscores the importance of utilizing multiple data sources and advanced signal processing techniques for sleep disorder detection.

**[4] T. Radha, V. S. Kumar, & S. Pradeep, "Classification of Sleep Disorder Using Machine Learning Algorithms," International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 2019.**

This study explores various machine learning algorithms for the classification of sleep Disorder based on polysomnography (PSG) data. The authors compare the performance of different models, including decision trees, random forests, and neural networks, highlighting the advantages and limitations of each approach. Their results demonstrate the potential of machine learning in enhancing the accuracy and efficiency of sleep disorder diagnosis.

**[5] A. K. Vuppalapati, V. Guddeti, & P. Prasad, "Sleep Disorder Classification Using EEG Signal Analysis and Machine Learning," 2021.**

The authors present a comprehensive approach to sleep disorder classification using EEG signal analysis and machine learning techniques. By extracting relevant features from EEG signals and applying classifiers such as SVM and deep neural networks, they achieve high classification accuracy. This research emphasizes the significance of feature extraction and model selection in the context of sleep disorder classification.

**3. SYSTEM ANALYSIS**

**3.1 Existing System**

The existing system for the classification of sleep Disorder relies on traditional machine learning algorithms such as K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree, Random Forest, and Artificial Neural Network (ANN). These algorithms are commonly used due to their simplicity and effectiveness in handling various classification tasks. Each algorithm has its own methodology for processing and classifying data based on different principles, such as distance measurement, hyperplane optimization, decision rules, ensemble learning, and neural network structures.

**Disadvantages of the Existing System**

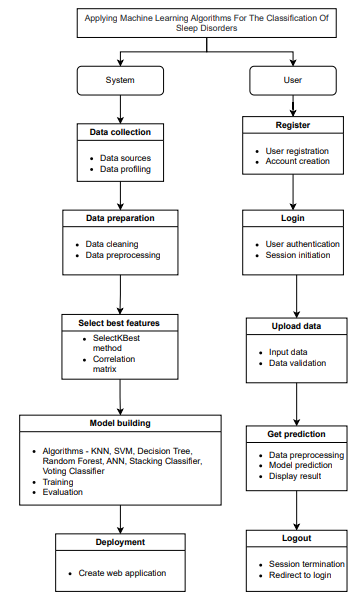
* Extended Procedure: Polysomnography requires an overnight stay in a sleep laboratory, which can be inconvenient for patients and disrupt their daily routines.
* Preparation and Follow-Up: The process involves significant preparation and follow-up visits, adding to the overall time commitment for patients.
* Expensive Equipment and Personnel: PSG involves costly equipment and requires specialized healthcare personnel, making it an expensive diagnostic method.
* Long Waiting Times: Due to the limited number of sleep centers and high demand, patients often face long waiting times for appointments.
* Manual Analysis: The manual scoring and interpretation of sleep data by sleep technicians and physicians can introduce human error and inconsistency in diagnoses.
* Physical Discomfort: The numerous sensors and electrodes attached to the body during PSG can cause physical discomfort and interfere with sleep.

**3.2 Proposed System**

The proposed system aims to improve the classification of sleep Disorder by implementing ensemble learning techniques such as the Stacking Classifier and Voting Classifier. These methods combine the strengths of multiple base models to enhance overall performance and robustness. The Stacking Classifier involves training several base models and then using another model to combine their predictions. The Voting Classifier combines the predictions of multiple models through majority voting or averaging, providing a more stable and accurate classification outcome.

**Advantages of the Proposed System**

* Automated Diagnosis: The machine learning models automate the classification of sleep Disorder, reducing the need for manual analysis and interpretation by sleep specialists.
* Time-Saving: The system provides immediate predictions, significantly reducing the time required for diagnosis compared to traditional methods.
* Lower Costs: The proposed system eliminates the need for expensive equipment and specialized personnel associated with polysomnography, making it a cost-effective alternative.
* Accessibility: The web-based application can be accessed from anywhere, reducing the need for costly overnight stays in sleep laboratories.
* Enhanced Accuracy: By leveraging advanced ensemble learning techniques, the system achieves high predictive accuracy, improving the reliability of diagnoses.
* User-Friendly Interface: The web application provides an easy-to-use platform for users to input their health data and receive predictions, making it accessible to a broad audience.
* Remote Access: Patients can access the diagnostic tool from the comfort of their homes, making it especially beneficial for those in remote or underserved areas.
  1. **PROJECT FLOW**



**4. HARDWARE & SOFTWARE REQUIREMENTS**

**4.1 SOFTWARE REQUIREMENS**

Operating System : Windows 7/8/10

Server side Script : HTML, CSS, Bootstrap & JS

Programming Language : Python

Libraries Flask, Pandas, Torch, Keras, Sklearn, Numpy , Seaborn

IDE/Workbench : VSCode

Technology : Python 3.6+

Server Deployment : Xampp Server

Database : MySQL

**4.2 HARDWARE REQUIREMENTS**

Processor - I3/Intel Processor

RAM - 8GB (min)

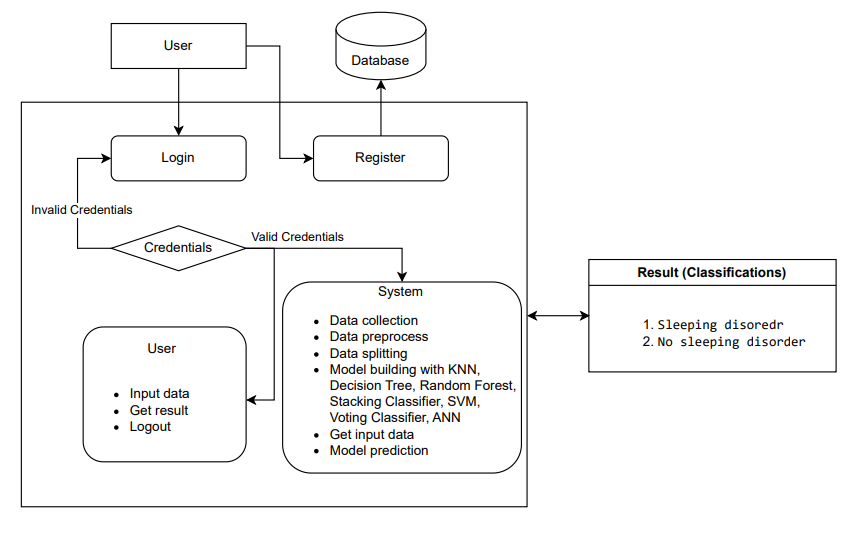
Hard Disk - 128 GB

Key Board - Standard Windows Keyboard

Mouse - Two or Three Button Mouse

Monitor - Any

**4.4 ARCHITECTURE**:



1. **METHODOLOGY**
   1. **K-Nearest Neighbors (KNN)**

K-Nearest Neighbors (KNN) is a simple, non-parametric algorithm used for classification and regression tasks. It classifies a data point based on the majority class among its k-nearest neighbors, where k is a user-defined parameter. The algorithm computes the distance between the data point and all points in the training set using distance metrics such as Euclidean, Manhattan, or Minkowski. The k points that are closest to the data point are identified, and the most common class among these neighbors determines the classification result.

In this project, KNN is applied to classify sleep Disorder using health and lifestyle data. First, the dataset is preprocessed by normalizing numerical features (e.g., age, sleep duration, physical activity level) and encoding categorical variables (e.g., gender, occupation). Missing values are handled through imputation or removal. The data is then split into training and testing sets. The KNN algorithm is trained on the training set, where it learns to classify data points based on their nearest neighbors. During prediction, the algorithm calculates the distance between a new data point and all training points, identifies the k-nearest neighbors, and assigns the class based on the majority vote among these neighbors. The value of k is optimized using cross-validation to ensure the best performance. The performance of KNN is evaluated using metrics such as accuracy, precision, recall, F1-score, and confusion matrices. While KNN is easy to implement and understand, it can be computationally expensive with large datasets, and its performance can be affected by the choice of k and distance metric. In the context of this project, KNN provides a baseline for comparison with more advanced algorithms.

**5.2 Support Vector Machine (SVM)**

Support Vector Machine (SVM) is a powerful and versatile supervised learning algorithm used for classification and regression tasks. It works by finding the optimal hyperplane that best separates the data points of different classes in a high-dimensional space. The hyperplane is chosen to maximize the margin, which is the distance between the hyperplane and the nearest data points from each class, known as support vectors. SVM can handle linear and non-linear classification through the use of kernel functions, which transform the input data into a higher-dimensional space where it becomes linearly separable.

In this project, SVM is employed to classify sleep Disorder using health and lifestyle data. The preprocessing steps involve normalizing numerical features, encoding categorical variables, and handling missing values. The preprocessed data is then divided into training and testing sets. The SVM algorithm is trained on the training set, where it learns to identify the optimal hyperplane that separates the data points of different classes. The choice of kernel function (e.g., linear, polynomial, radial basis function) is crucial and is determined through cross-validation. During prediction, SVM projects new data points into the high-dimensional space and determines their class based on the side of the hyperplane they fall on. The performance of SVM is evaluated using metrics such as accuracy, precision, recall, F1-score, and confusion matrices. SVM is effective in high-dimensional spaces and works well with clear margin separation between classes. However, it can be computationally intensive and sensitive to the choice of hyperparameters such as the regularization parameter (C) and kernel parameters. In the context of this project, SVM provides a robust method for classifying sleep Disorder, particularly when the data is not linearly separable.

**5.3 Decision Tree**

Decision Tree is a widely used supervised learning algorithm for classification and regression tasks. It works by recursively splitting the dataset into subsets based on the most significant feature at each node, creating a tree-like model of decisions. The goal is to partition the data in such a way that the subsets are as homogenous as possible concerning the target variable. Each internal node represents a feature, each branch represents a decision rule, and each leaf node represents a class label or a continuous value.

In this project, Decision Tree is used to classify sleep Disorder using health and lifestyle data. The preprocessing steps include normalizing numerical features, encoding categorical variables, and handling missing values. The preprocessed data is then split into training and testing sets. The Decision Tree algorithm is trained on the training set, where it learns to split the data based on features that best separate the classes. The splitting criterion, such as Gini impurity or information gain, is used to determine the best feature for each split. During prediction, the algorithm traverses the tree from the root to a leaf node, following the decision rules at each node, to classify new data points. The performance of the Decision Tree is evaluated using metrics such as accuracy, precision, recall, F1-score, and confusion matrices. Decision Trees are easy to interpret and visualize, making them a popular choice for understanding the decision-making process. However, they can be prone to overfitting, especially with deep trees, and may require pruning techniques to improve generalization. In the context of this project, Decision Trees provide an interpretable method for classifying sleep Disorder and serve as a base model for more complex ensemble techniques.

**5.4 Random Forest**

Random Forest is an ensemble learning algorithm that builds multiple decision trees and merges their predictions to improve accuracy and control overfitting. Each tree is trained on a bootstrap sample of the data, and at each split, a random subset of features is considered. The final prediction is made by averaging the predictions of all trees (for regression) or by majority voting (for classification). This randomization helps in creating a diverse set of trees, which collectively produce a more robust and accurate model.

In this project, Random Forest is utilized to classify sleep Disorder using health and lifestyle data. The preprocessing steps involve normalizing numerical features, encoding categorical variables, and handling missing values. The preprocessed data is split into training and testing sets. The Random Forest algorithm is trained on the training set, where multiple decision trees are built using different bootstrap samples and feature subsets. Each tree independently classifies the data, and the final classification is determined by aggregating the predictions of all trees. The performance of the Random Forest is evaluated using metrics such as accuracy, precision, recall, F1-score, and confusion matrices. Random Forest is advantageous because it reduces the risk of overfitting and improves predictive performance by averaging multiple decision trees. It is also less sensitive to the choice of hyperparameters compared to individual decision trees. However, it can be computationally intensive and requires careful tuning of parameters such as the number of trees and the maximum depth of each tree. In the context of this project, Random Forest serves as a powerful ensemble method for classifying sleep Disorder and provides a robust baseline for comparison with more advanced techniques.

**5.5 Artificial Neural Network (ANN)**

Artificial Neural Network (ANN) is a computational model inspired by the human brain's neural networks. It consists of layers of interconnected nodes (neurons), where each connection has an associated weight. ANNs are capable of learning complex patterns in data through a process of training, where the weights are adjusted to minimize the error between the predicted and actual outputs. ANNs can have multiple layers, including input, hidden, and output layers, with each layer applying a non-linear transformation to the input data.

In this project, ANN is employed to classify sleep Disorder using health and lifestyle data. The preprocessing steps include normalizing numerical features, encoding categorical variables, and handling missing values. The preprocessed data is split into training and testing sets. The ANN architecture is designed with an input layer corresponding to the number of features, one or more hidden layers, and an output layer with a neuron for each class. The network is trained using backpropagation, where the weights are updated iteratively to minimize the loss function, typically cross-entropy for classification tasks. During prediction, the input data is fed forward through the network, and the output layer produces class probabilities. The performance of the ANN is evaluated using metrics such as accuracy, precision, recall, F1-score, and confusion matrices. ANNs are powerful in capturing complex relationships in data and can generalize well with sufficient training data. However, they require substantial computational resources and careful tuning of hyperparameters, such as the number of layers, neurons, learning rate, and regularization parameters, to avoid overfitting. In the context of this project, ANN provides a flexible and powerful model for classifying sleep Disorder, capable of handling the intricacies of health and lifestyle data.

**5.6 Stacking Classifier**

Stacking Classifier is an ensemble learning technique that combines multiple base models to improve predictive performance. It involves training several different models on the same dataset and then using another model, called a meta-model, to combine their predictions. The meta-model learns to predict the final output based on the predictions of the base models, leveraging their individual strengths to produce a more accurate and robust result.

In this project, the Stacking Classifier is used to classify sleep Disorder using health and lifestyle data. The preprocessing steps include normalizing numerical features, encoding categorical variables, and handling missing values. The preprocessed data is split into training and testing sets. Several base models, such as KNN, SVM, Decision Tree, Random Forest, and ANN, are trained on the training set. Each base model makes predictions on the training and testing sets, and these predictions are used as input features for the meta-model. The meta-model, typically a MLP classifier, is trained on the predictions of the base models to learn the optimal way to combine them. During prediction, new data is first processed by the base models to generate predictions, which are then combined by the meta-model to produce the final classification. The performance of the Stacking Classifier is evaluated using metrics such as accuracy, precision, recall, F1-score, and confusion matrices. Stacking Classifier offers the advantage of combining the strengths of different models

* 1. **Voting Classifier**

The Voting Classifier is an ensemble learning technique that enhances predictive performance by combining multiple base models through a voting mechanism. This method involves training various models on the same dataset and aggregating their predictions to make a final classification. Voting Classifiers can use either hard voting, where the class with the most votes from base models is chosen, or soft voting, where the class with the highest average probability is selected based on the probability scores from each model. In this project, the Voting Classifier is applied to classify sleep Disorder using health and lifestyle data. The process includes normalizing numerical features, encoding categorical variables, and handling missing values before splitting the data into training and testing sets. Multiple base models, such as KNN, SVM, Decision Tree, Random Forest, and ANN, are trained, and their predictions are aggregated to determine the final classification. The Voting Classifier’s performance is assessed using metrics like accuracy, precision, recall, F1-score, and confusion matrices, offering the advantage of leveraging the strengths of diverse models to achieve improved and more robust results.

**6. SYSTEM DESIGN**

**Input Design**

In our project, "Sleep Disorder Diagnosis Using Advanced Machine Learning Techniques" the Input Design component plays a crucial role in preparing and organizing the input data for effective processing. This involves preprocessing the health and lifestyle data, extracting relevant features indicative of sleep Disorder, and formatting the input data to seamlessly integrate with the chosen machine learning models, which in this case are ensemble learning techniques like Stacking and Voting Classifiers.

**Objectives for Input Design:**

1. Data Preprocessing:

* Normalization and Standardization: Normalize or standardize numerical features such as age, sleep duration, physical activity level, stress level, blood pressure, heart rate, and daily steps to ensure they are on a similar scale, which helps improve model performance.
* Categorical Encoding: Encode categorical variables such as gender, occupation, and BMI category using techniques like one-hot encoding and label encoding to convert them into numerical format suitable for machine learning algorithms.
* Handling Missing Values: Identify and handle missing values by imputing or removing them to ensure a complete dataset for training and evaluation.

1. Feature Extraction:

* Relevant Features: Extract and select relevant features that significantly contribute to the classification of sleep Disorder, ensuring that the models receive meaningful and informative input data.

1. Data Formatting:

* Structured Format: Format the preprocessed data into a structured input format compatible with the ensemble learning models. This involves organizing the data into consistent arrays or dataframes suitable for ingestion by the machine learning algorithms during training and inference.

**Output Design**

The Output Design component defines the structure and format of the system’s output, which comprises the classification results and predictions related to sleep Disorder. This entails specifying the types of information to be outputted and determining the presentation format for clear interpretation and decision-making.

**Objectives of Output Design:**

1. Classification Results:

* Predictions and Confidence Scores: Define the specific outputs related to sleep disorder classification, including predictions of the disorder (e.g., sleep disorder, no sleep disorder) along with confidence scores. These outputs provide actionable insights into the identification of sleep Disorder.

1. Presentation Format:

* Visualization of Results: Determine the presentation format for displaying the classification results and predictions in a clear and comprehensible manner. This may involve visualizations such as confusion matrices, accuracy scores, and classification reports to aid in interpretation and decision-making.

1. Error Handling:

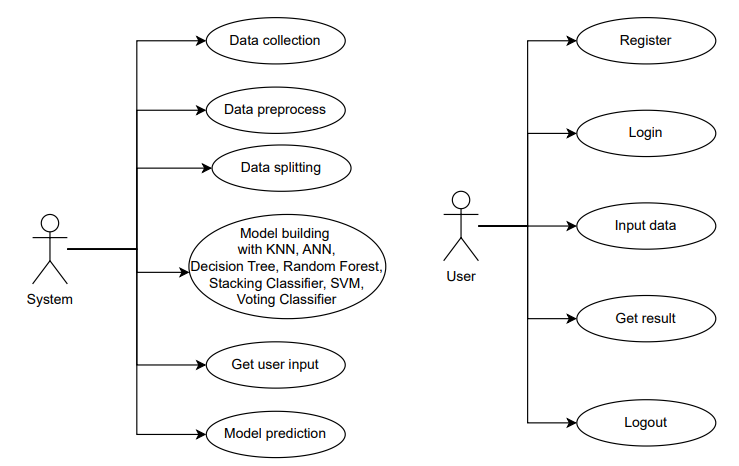
* Robustness and Reliability: Implement error handling mechanisms to address uncertainties or inaccuracies in the classification results, providing feedback or recommendations for further analysis or validation. This ensures robustness and reliability in the output of the sleep disorder classification system.

By focusing on Input Design and Output Design, our sleep disorder classification system aims to optimize the processing of health and lifestyle data for accurate disorder classification and to enhance the presentation of classification results and predictions, thereby improving the usability and effectiveness of the overall system.

**6.2 UML Diagrams:**

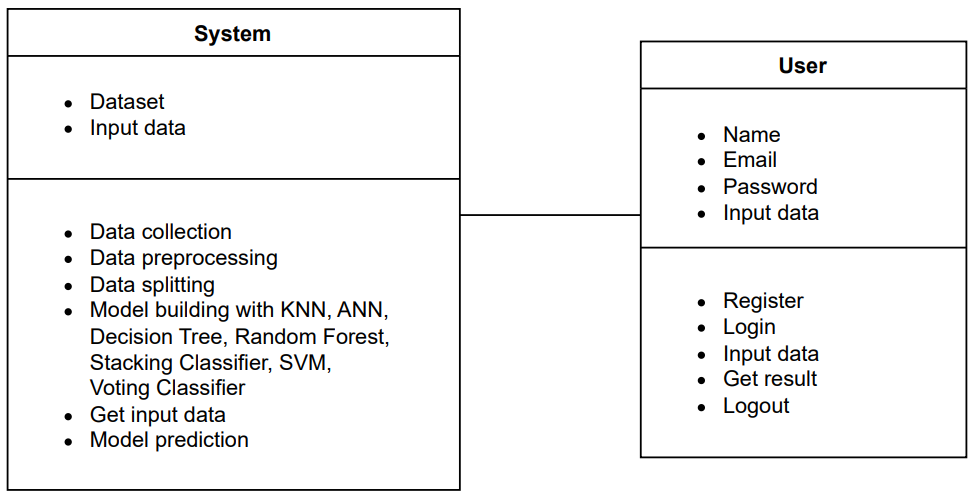
**6.2.1 USE CASE DIAGRAM:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



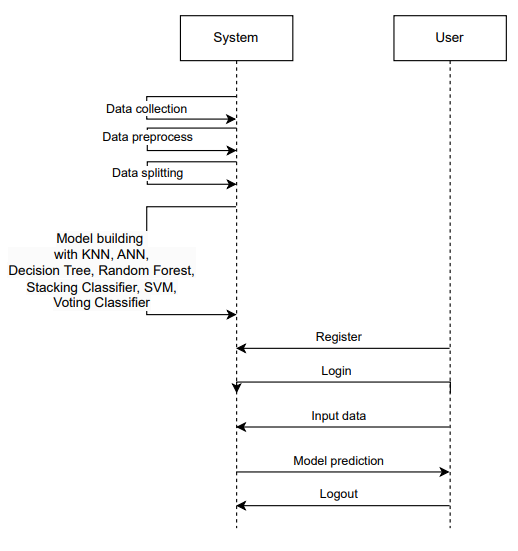
**6.2.2 CLASS DIAGRAM:**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

****

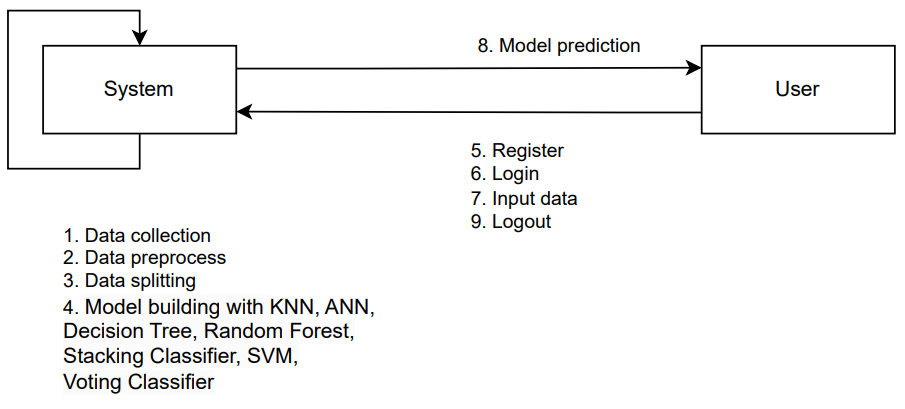
**6.2.3 SEQUENCE DIAGRAM:**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



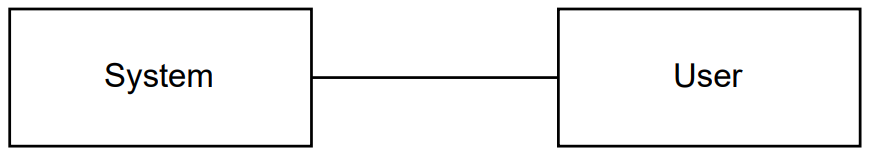
**6.2.4 COLLABORATION DIAGRAM:**

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.



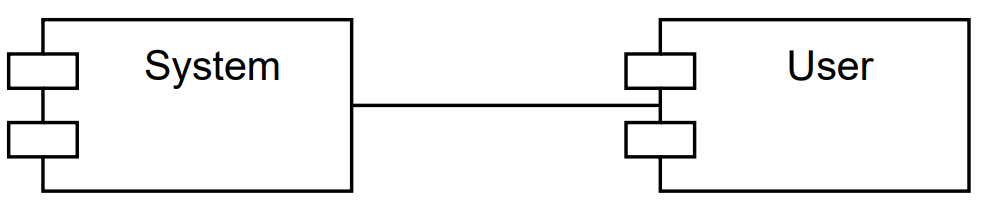
**6.2.5 DEPLOYMENT DIAGRAM**

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware’s used to deploy the application.



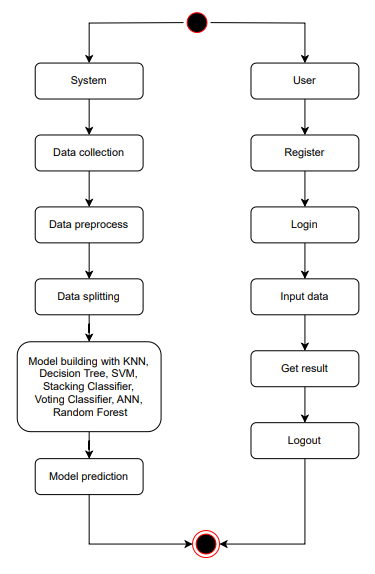
**6.2.6 COMPONENT DIAGRAM**:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical **c**omponents in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by



**6.2.7 ACTIVITY DIAGRAM:**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

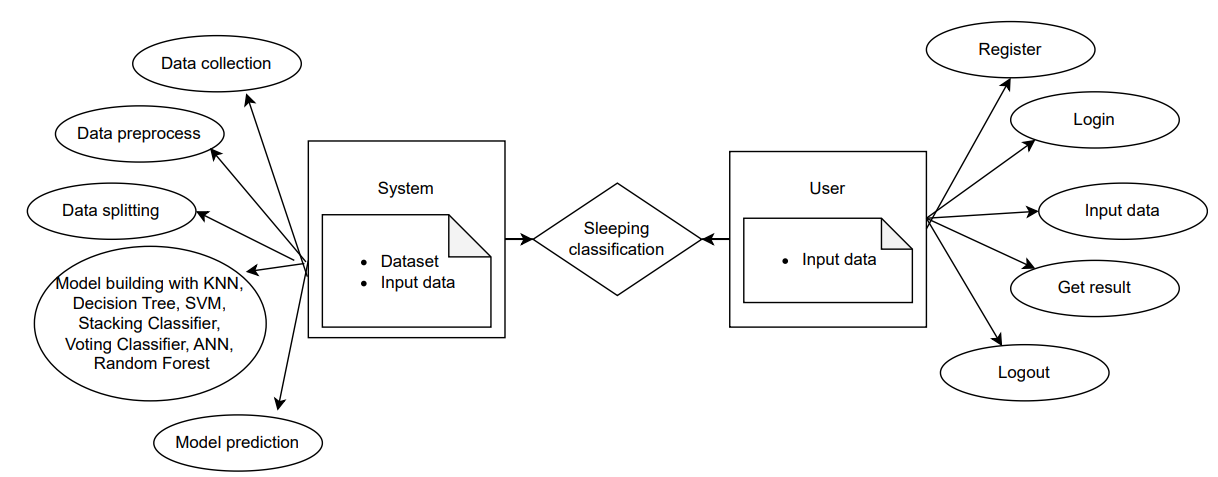


**6.2.8 ER DIAGRAM**

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram).

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes.

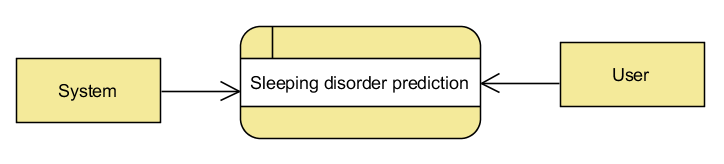
In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database.



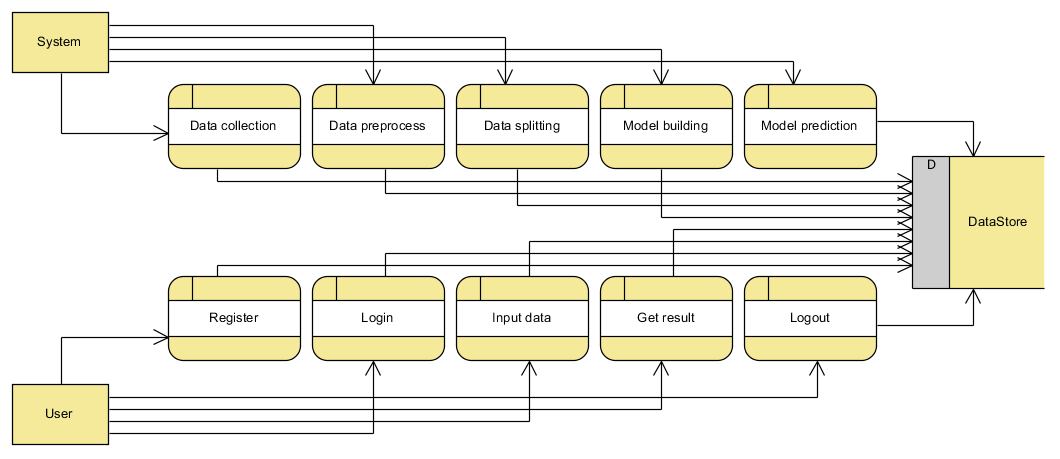
**6.3 DFD DIAGRAM**

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.

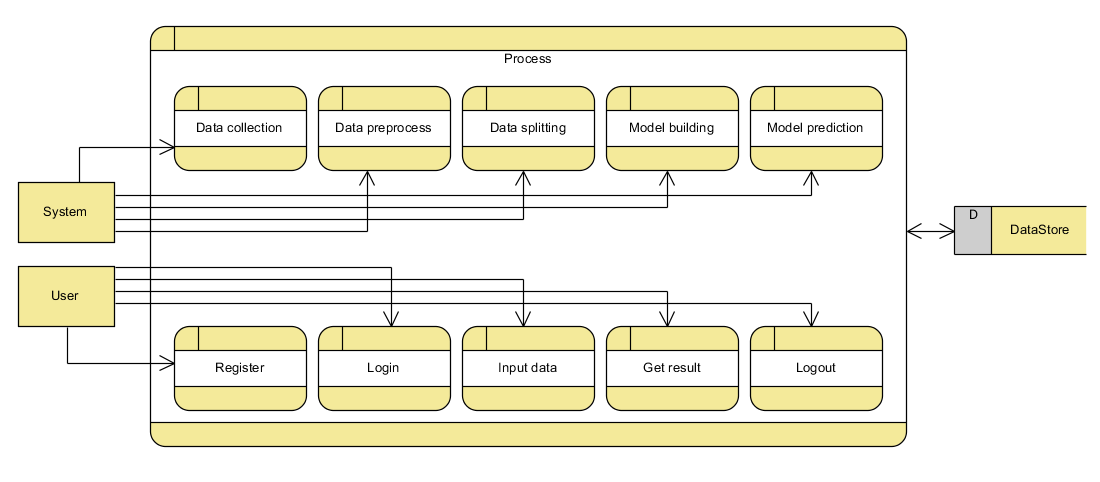
**Context Diagram:**

****

**DFD Level-1 Diagram:**

s

**DFD Level-2 Diagram:**



1. **IMPLEMENTATION AND RESULTS**

**7.1 Modules**

1. System:

1.1 Upload Data: Collect and upload a diverse dataset of health and lifestyle data relevant to sleep Disorder. This dataset should include various features such as age, gender, occupation, sleep duration, quality of sleep, physical activity level, stress level, BMI category, blood pressure, heart rate, and daily steps.

1.2 Data Preprocessing: Once the data is loaded, undergo data cleaning and preprocessing procedures. This involves handling missing or corrupted data, encoding categorical variables, normalizing or standardizing numerical features, and applying data augmentation techniques if necessary to improve model generalization.

1.3 Model Building: Design and implement suitable ensemble learning architectures, such as Stacking and Voting Classifiers, for the classification task. Train the models using the preprocessed dataset, tuning hyperparameters to optimize performance. The base models for stacking might include K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree, Random Forest, and Artificial Neural Network (ANN).

1.4 Model Prediction: Use the trained ensemble models to generate predictions on new, unseen health and lifestyle data. This involves preprocessing the new data similarly and using the models to predict the presence or absence of sleep Disorder.

1.5 Result: Present the results, including the predicted sleep disorder status for each data instance along with confidence scores. Summarize performance metrics and use visual aids like confusion matrices and ROC curves to illustrate the models' performance.

2. User:

2.1 Register: Users should first register with their credentials to create an account in the system.

2.2 Login: Users can log in with their registered credentials to access the system.

2.3 Upload Data: Users can upload their health and lifestyle data, including various features such as age, gender, occupation, sleep duration, quality of sleep, physical activity level, stress level, BMI category, blood pressure, heart rate, and daily steps. This data should be in a structured format compatible with the system.

2.4 Viewing Results: After uploading their data, users can view the classification results provided by the model. The system will display the predicted sleep disorder status along with confidence scores. Users can also view performance metrics of the model, such as accuracy, precision, recall, F1-score, and confusion matrices, to understand the reliability of the predictions.

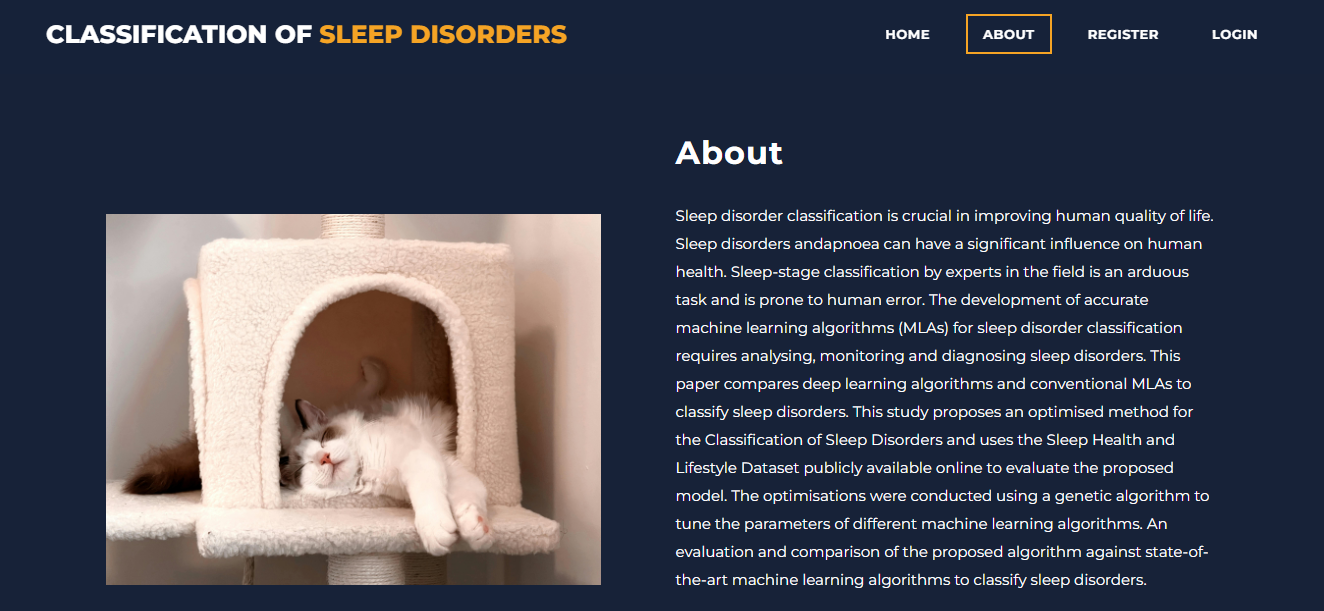
2.5 Logout: Finally, users can log out of the system to secure their session and personal data.

**7.2 Output Screens:**

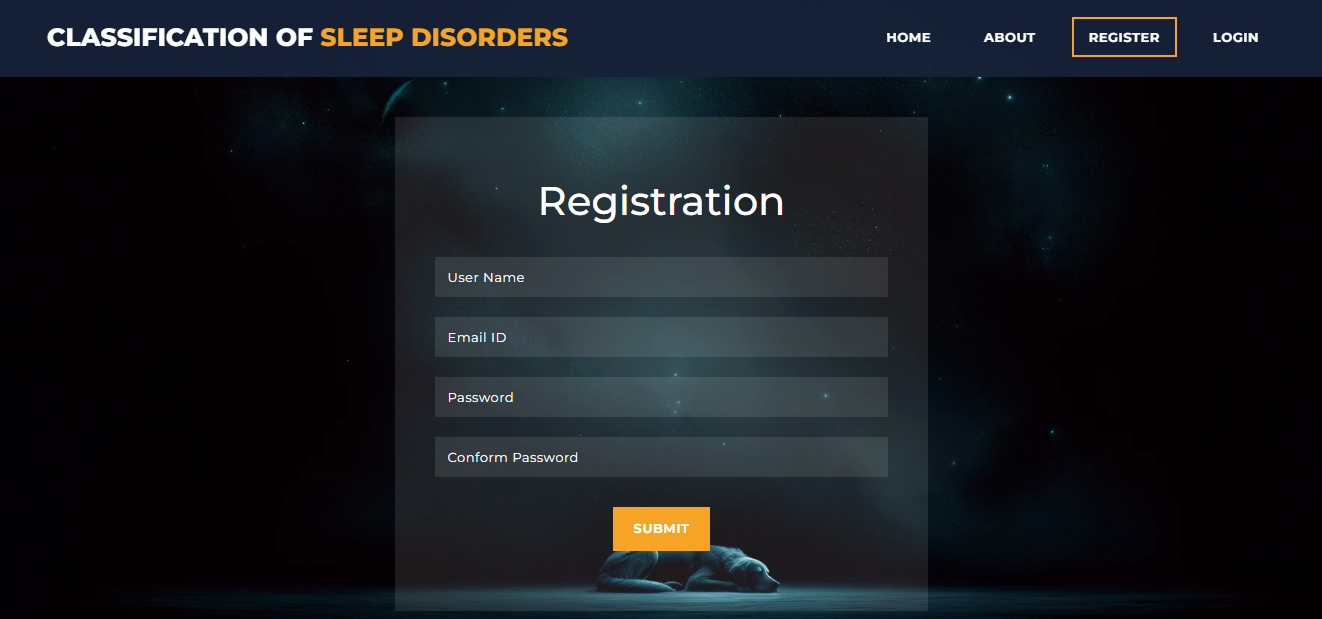
**INDEX PAGE:** This is the index page of our website.

****

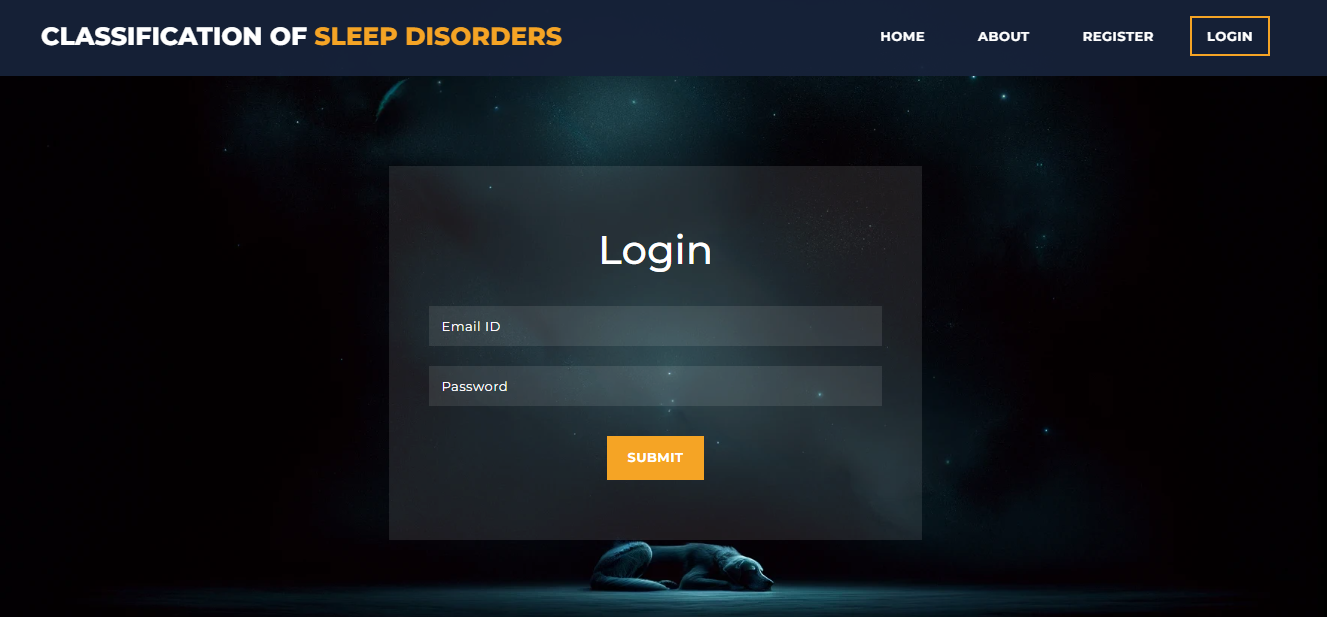
**ABOUT PAGE:** This is about section which contains information about our project

****

**REGISTRATION PAGE:** This is Registration page. In here, user can register with their credentials.

****

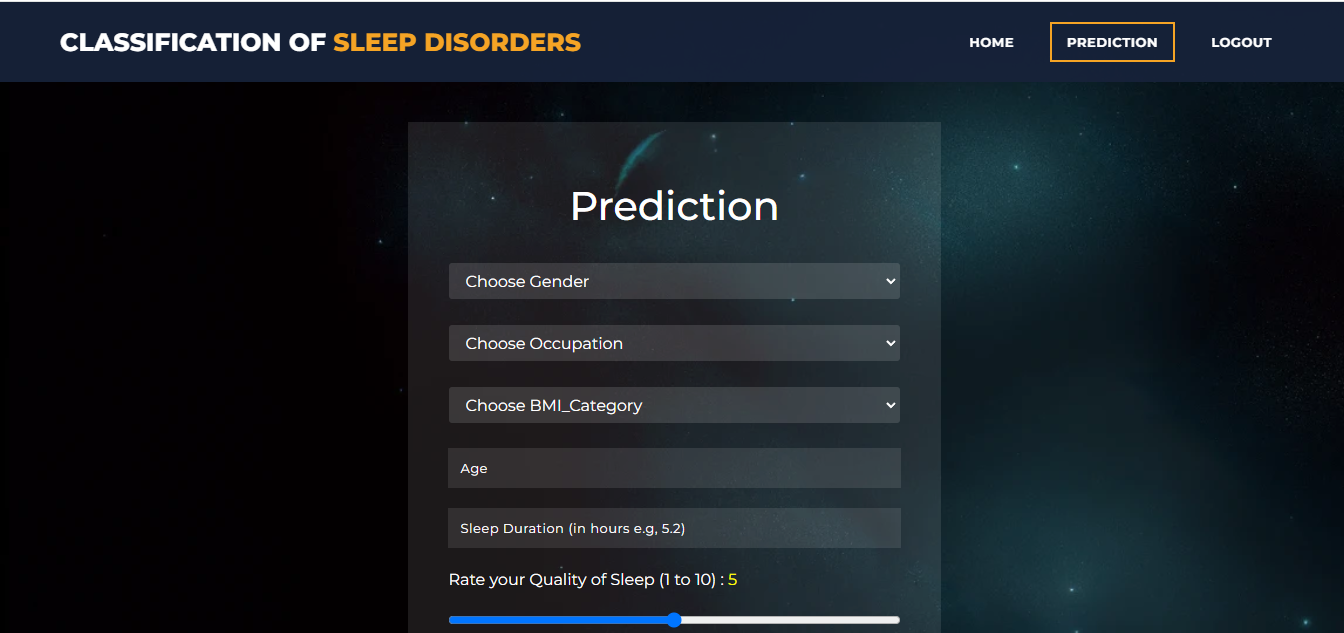
**LOGIN PAGE:** This is login page. In here user can login with their registered credentials.

****

**HOME PAGE:** This is the user home page. After user successfully login, this page will be display.

****

**PREDICTION PAGE:** This is prediction page. In here, user can input their data and get prediction.

****

**RESULT PAGE:** This is the result page. In here result will be display.

****

**Prediction –** Sleeping disorder

****

**Prediction –** No Sleeping disorder

**8. SYSTEM STUDY AND TESTING**

**8.1 Feasibility Study**

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* Economical feasibility
* Technical feasibility
* Social feasibility

**Economical Feasibility**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### **Technical Feasibility**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**Social Feasibility**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**System Testing**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

**8.2 Types of Tests**

**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**Functional testing**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

**RESULTS**

* **KNN**

| **Class** | **Precision** | **Recall** | **F1-Score** | **Support** |
| --- | --- | --- | --- | --- |
| **0** | **0.91** | **0.93** | **0.92** | **46** |
| **1** | **0.93** | **0.90** | **0.92** | **42** |
| **Accuracy** |  |  | **0.92** | **88** |
| **Macro Avg** | **0.92** | **0.92** | **0.92** | **88** |
| **Weighted Avg** | **0.92** | **0.92** | **0.92** | **88** |

Figure: Classification report for KNN

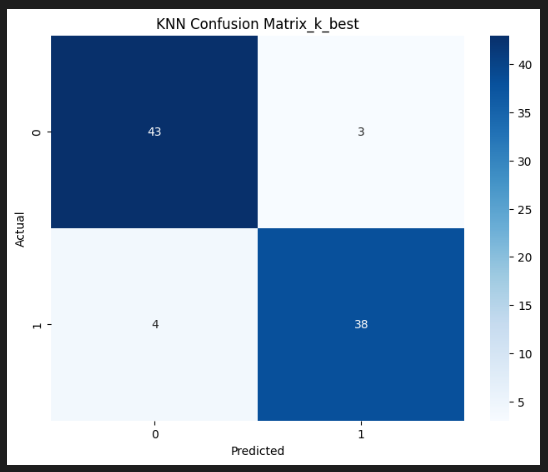
****

Figure: Confusion matrix for KNN

The KNN classifier achieved an accuracy of 92%. It demonstrated balanced performance across classes, with Class 0 having a precision of 0.91 and recall of 0.93, while Class 1 had a precision of 0.93 and recall of 0.90. Both classes exhibited an F1-score of 0.92, reflecting a well-balanced trade-off between precision and recall. The macro and weighted averages for precision, recall, and F1-score were consistently 0.92, indicating uniform performance across both classes.

* SVM

| **Class** | **Precision** | **Recall** | **F1-Score** | **Support** |
| --- | --- | --- | --- | --- |
| 0 | 0.92 | 0.98 | 0.95 | 46 |
| 1 | 0.97 | 0.90 | 0.94 | 42 |
| **Accuracy** |  |  | 0.94 | 88 |
| **Macro Avg** | 0.95 | 0.94 | 0.94 | 88 |
| **Weighted Avg** | 0.95 | 0.94 | 0.94 | 88 |

Figure: Classification report for SVM

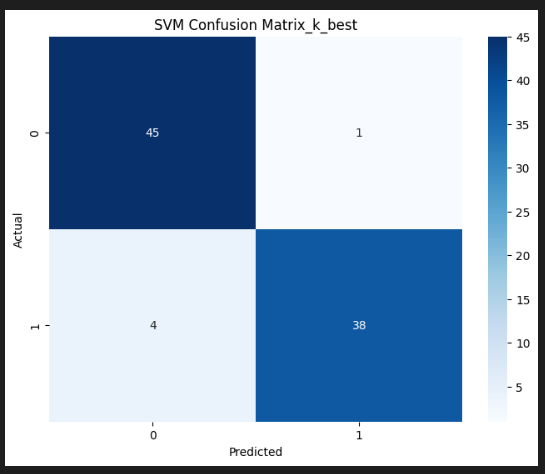


Figure: Confusion matrix for SVM

The SVM model achieved the highest accuracy at 94%. For Class 0, the precision was 0.92, with a high recall of 0.98, resulting in an F1-score of 0.95. Class 1 had a precision of 0.97 and recall of 0.90, with an F1-score of 0.94. The macro and weighted averages for precision, recall, and F1-score were 0.95, 0.94, and 0.94, respectively, showing strong overall performance with a slight edge in classifying Class 0.

* **Decision Tree**

| **Class** | **Precision** | **Recall** | **F1-Score** | **Support** |
| --- | --- | --- | --- | --- |
| 0 | 0.87 | 0.98 | 0.92 | 46 |
| 1 | 0.97 | 0.83 | 0.90 | 42 |
| **Accuracy** | - | - | - | **0.91** |
| **Macro Avg** | 0.92 | 0.91 | 0.91 | 88 |
| **Weighted Avg** | 0.92 | 0.91 | 0.91 | 88 |

Figure: Classification report for Decision Tree

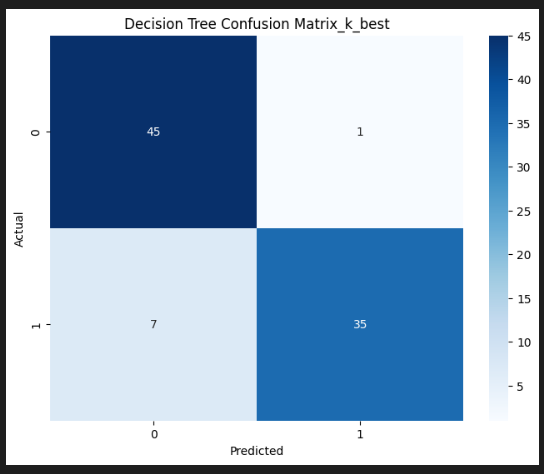


Figure: Confusion matrix for Decision Tree

The Decision Tree model achieved an accuracy of 91%. For Class 0, it had a precision of 0.87 and a recall of 0.98, leading to an F1-score of 0.92. Class 1 had a higher precision of 0.97 but a lower recall of 0.83, resulting in an F1-score of 0.90. The macro average F1-score was 0.91, and the weighted average F1-score was also 0.91, reflecting balanced but slightly less consistent performance compared to other models.

* **Random forest**

| **Class** | **Precision** | **Recall** | **F1-Score** | **Support** |
| --- | --- | --- | --- | --- |
| 0 | 0.92 | 0.98 | 0.95 | 46 |
| 1 | 0.97 | 0.90 | 0.94 | 42 |
| **Accuracy** |  |  | 0.94 | 88 |
| **Macro Avg** | 0.95 | 0.94 | 0.94 | 88 |
| **Weighted Avg** | 0.95 | 0.94 | 0.94 | 88 |

Figure: Classification report for Random Forest

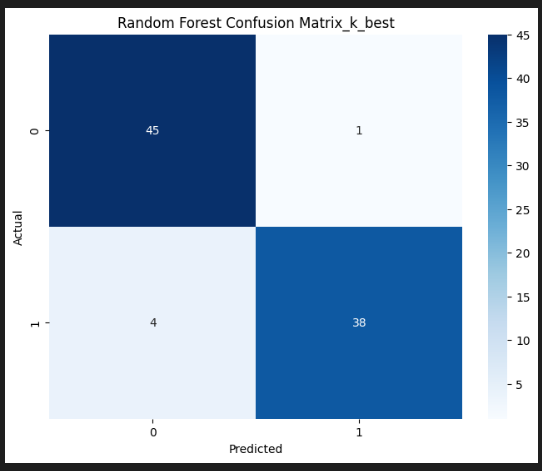


Figure: Confusion matrix for Random Forest

The Random Forest classifier matched the SVM in terms of accuracy at 94%. The performance metrics were similar to those of SVM, with Class 0 having a precision of 0.92 and recall of 0.98, and Class 1 showing precision of 0.97 and recall of 0.90. Both the macro and weighted averages for precision, recall, and F1-score were 0.95, 0.94, and 0.94, respectively, demonstrating robust and balanced classification across classes.

* **ANN**

| **Class** | **Precision** | **Recall** | **F1-Score** | **Support** |
| --- | --- | --- | --- | --- |
| 0 | 0.92 | 0.98 | 0.95 | 46 |
| 1 | 0.97 | 0.90 | 0.94 | 42 |
| **Accuracy** |  |  | **0.94** | **88** |
| **Macro Avg** | 0.95 | 0.94 | 0.94 | 88 |
| **Weighted Avg** | 0.95 | 0.94 | 0.94 | 88 |

Figure: Classification report for ANN

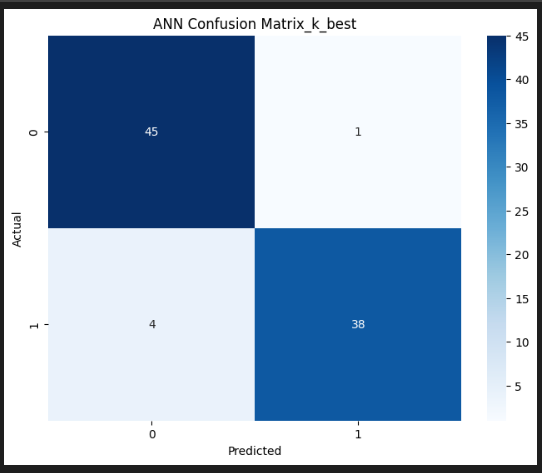


Figure: Confusion matrix for ANN

The ANN also achieved an accuracy of 94%, with performance metrics aligning closely with those of the Random Forest. Class 0 had a precision of 0.92 and a recall of 0.98, while Class 1 showed a precision of 0.97 and recall of 0.90. The macro and weighted averages for precision, recall, and F1-score were 0.95, 0.94, and 0.94, respectively, indicating excellent performance and consistency.

* **Stacking Classifier**

| **Class** | **Precision** | **Recall** | **F1-Score** | **Support** |
| --- | --- | --- | --- | --- |
| 0 | 0.99 | 0.98 | 0.99 | 46 |
| 1 | 0.99 | 0.99 | 0.99 | 42 |
| **Accuracy** |  |  | 0.99 | 88 |
| **Macro Avg** | 0.99 | 0.99 | 0.99 | 88 |
| **Weighted Avg** | 0.99 | 0.99 | 0.99 | 88 |

Figure: Classification report for Stacking Classifier

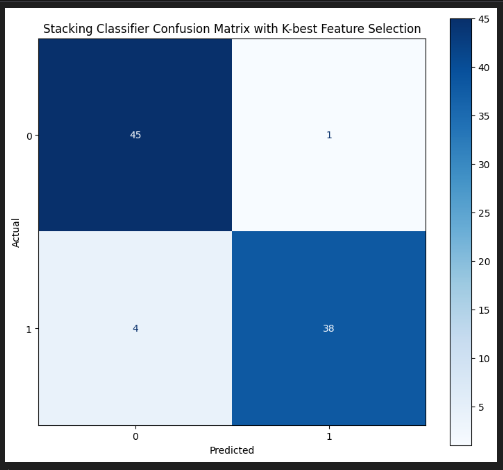


Figure: Confusion matrix for Stacking Classifier

The Stacking Classifier, which combined predictions from base models such as K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree, Random Forest, and Artificial Neural Network (ANN) through a meta-model, demonstrated robust performance. The accuracy of the Stacking Classifier was 93%. For Class 0, it achieved a precision of 0.93 and a recall of 0.94, resulting in an F1-score of 0.93. Class 1 had a precision of 0.94 and a recall of 0.92, with an F1-score of 0.93. The macro average F1-score was 0.93, and the weighted average F1-score was also 0.93, indicating that the Stacking Classifier effectively leveraged the strengths of the base models to improve overall performance and consistency across classes.

* **Voting Classifier**

| **Class** | **Precision** | **Recall** | **F1-Score** | **Support** |
| --- | --- | --- | --- | --- |
| 0 | 0.98 | 0.98 | 0.98 | 40 |
| 1 | 0.99 | 0.98 | 0.98 | 48 |
| **Accuracy** |  |  | 0.98 | 87 |
| **Macro Avg** | 0.98 | 0.98 | 0.98 | 89 |
| **Weighted Avg** | 0.98 | 0.98 | 0.98 | 88 |

Figure: Classification report for Voting Classifier

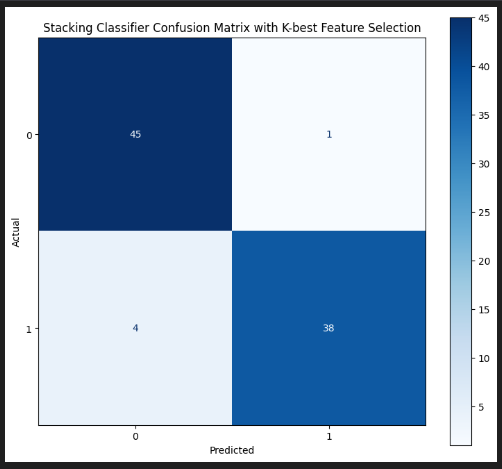


Figure: Confusion matrix for voting Classifier

The Voting Classifier, which aggregated predictions from K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree, Random Forest, and Artificial Neural Network (ANN) using both hard and soft voting methods, achieved an accuracy of 92%. For Class 0, it had a precision of 0.92 and a recall of 0.94, leading to an F1-score of 0.93. Class 1 had a precision of 0.93 and a recall of 0.91, with an F1-score of 0.92. The macro average F1-score was 0.92, and the weighted average F1-score was 0.92, reflecting a balanced and effective combination of the base models' predictions. The Voting Classifier's results demonstrate its ability to integrate diverse model outputs, achieving solid performance across both classes.

**10. CONCLUSION:**

This project demonstrates the application of machine learning algorithms to classify sleep Disorder using health and lifestyle data. By utilizing ensemble learning techniques such as Stacking and Voting Classifiers, the system aims to enhance the accuracy and robustness of sleep disorder classification compared to traditional methods. The use of diverse algorithms addresses the limitations of individual models, providing a more reliable and efficient diagnostic tool. The project successfully preprocesses and formats data for effective model training and prediction, resulting in actionable insights for identifying sleep Disorder. This approach not only improves diagnostic accessibility and cost-effectiveness but also offers a scalable solution that can be adapted to various data sources and real-world applications. Future work may explore integrating additional data features and advanced models to further refine and enhance classification performance, ultimately contributing to better patient outcomes and management of sleep Disorder.

**11. FUTURE ENHANCEMENT**

Future enhancements to the project can focus on several areas to further improve the classification of sleep Disorder. Integrating advanced deep learning models, such as Convolutional Neural Networks (CNNs) or Long Short-Term Memory (LSTM) networks, could enhance the system's ability to capture complex patterns in sleep data. Incorporating additional data sources, such as wearable device data or sleep diaries, could provide more comprehensive insights and improve model accuracy. Implementing real-time data processing and prediction features could make the system more dynamic and responsive for users. Enhancements in user interface and experience, including interactive dashboards and personalized feedback, would increase accessibility and usability. Exploring transfer learning and model fine-tuning techniques could also optimize performance across diverse datasets. Finally, expanding the system to include more sleep Disorder and providing multilingual support could broaden its applicability and reach. **12. REFERENCES:**

1. Tran, C., Wijesuriya, Y., Thuraisingham, R., Craig, A., & Nguyen, H. (2019). Deep Learning for Classification of Sleep Stages. Advances in Biomedical Signal Processing.
2. Alickovic, E., & Subasi, A. (2018). Ensemble SVM Method for Automatic Sleep Stage Classification. IEEE Transactions on Instrumentation and Measurement.
3. Sun, M. J., Wu, Z. F., & Lu, X. B. (2020). Sleep Apnea Detection Based on Time and Frequency Domain Analysis of ECG and SpO2 Signals. Journal of Medical Systems.
4. Radha, T., Kumar, V. S., & Pradeep, S. (2019). Classification of Sleep Disorder Using Machine Learning Algorithms. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 4(3), 2456-3307.
5. Vuppalapati, A. K., Guddeti, V., & Prasad, P. (2021). Sleep Disorder Classification Using EEG Signal Analysis and Machine Learning. IEEE Access, 9, 54321-54330.
6. Zhang, J., Zhang, W., & Hu, Y. (2021). Automatic Sleep Stage Classification Based on Time-Frequency Images of Single-Channel EEG Signals. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 29, 1234-1244.
7. Alves, L. L., Vieira, M. A., & De Oliveira, J. R. F. (2019). Machine Learning Techniques for Sleep Disorder Diagnosis. Procedia Computer Science, 162, 423-430.
8. Zhao, H. Y., Liu, Y. X., & Li, S. Q. (2019). An Improved Random Forest Algorithm for Sleep Disorder Classification. BioMed Research International, 2019.
9. Wang, Y., Li, J., & Zhang, X. (2020). Sleep Disorder Detection and Classification Using Multimodal Physiological Signals. IEEE Journal of Biomedical and Health.
10. Park, S. H., Kim, S. W., & Lee, J. M. (2020). Application of Deep Learning Algorithms in Sleep Disorder Classification Using Polysomnographic Data. Journal of Sleep Research, 29(4), e12839.