DISEASE PREDICTION MODEL

A PROJECT REPORT for
Mini Project-I (K24MCA18P) Session (2024-25)

Submitted by

HARSHITA CHAUHAN (20241011610009)
HIMANSHI SHARMA (202410116100091)
GUNJAN SHARMA (202410116100080)

Submitted in partial fulfilment of the Requirements for the Degree of

MASTER OF COMPUTER APPLICATION

Under the Supervision of Ms. Divya Singhal Assistant Professor

Submitted to

DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad Uttar Pradesh-201206

(DECEMBER- 2024)

CERTIFICATE

Certified that **Harshita Chauhan (2426MCA315), Himanshi Sharma (2426MCA206)**, **Gunjan Sharma (2426MCA631)** has/ have carried out the project work having "**Disease Prediction Model**" (**Mini Project-II, ID201B**) for **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself/herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Ms. Shruti
Assistant Professor
ComputerApplications
KIET Group of Institutions, Ghaziabad
Disease Prediction Model

Dr. Akash Rajak
Dean Department of
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Harshita Chauhan
Himanshi Sharma
Gunjan Sharma

ABSTRACT

It is a system which provides the user the information and tricks to take care of the health system of the user and it provides how to search out the disease using this prediction. Now a day's health industry plays major role in curing the diseases of the patients so this is often also some quite help for the health industry to inform the user and also it\'s useful for the user just in case he/she doesn't want to travel to the hospital or the other clinics, so just by entering the symptoms and every one other useful information the user can get to grasp the disease he/she is affected by and also the health industry may also get enjoy this method by just asking the symptoms from the stoner and entering within the system and in only many seconds they'll tell the precise and over to some extent the accurate conditions. This Disease Prediction Using Machine Learning is totally through with the assistance of Machine Learning and Python programming language and also using the dataset that\'s available previously by the hospitals using that we are going to predict the diseases.

Keywords: Disease Diagnosis, Medical Dataset, Health Industry, Symptom Analysis

ACKNOWLEDGEMENTS

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor, **Ms. Shruti** for her guidance, help, and encouragement throughout my project work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to Dr. Arun Kumar Tripathi, Professor and Dean, Department of Computer Applications, for his insightful comments and administrative help on various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Our project, the "Disease Prediction System Using Machine Learning," addresses a critical issue in healthcare – the timely and accurate diagnosis of diseases. The traditional healthcare approach often focuses on one disease at a time, leading to fragmented analyses and delayed interventions. This system revolutionizes disease prediction by utilizing machine learning algorithms, specifically the Random-Forest-Classifier, to predict a range of diseases based on a set of symptoms.

The significance of this project lies in its potential to impact people's lives positively. By enabling a swift and accurate prediction of diseases, individuals can receive early warnings and timely medical attention. This not only contributes to more effective treatments but also aids in preventing the escalation of health conditions, ultimately reducing mortality rates.

The user-friendly interface, built with Tkinter, facilitates easy interaction, making it accessible to a wide range of users. Individuals can input their symptoms, and the system provides instant predictions, empowering them with valuable health insights. This project serves as a valuable tool for both individuals monitoring their health and healthcare professionals seeking quick and accurate preliminary assessments.

In essence, the "Disease Prediction System Using Machine Learning" leverages advanced technology to address a pressing healthcare challenge, offering a practical solution that can positively impact public health outcomes.

1.2 PROJECT

The "Disease Prediction using Machine Learning" project emerges as a pioneering endeavour, seamlessly intertwining advancements in machine learning with the complexities of healthcare diagnostics. In a realm where the demand for rapid and accurate disease identification is ever-increasing, this project takes a quantum leap towards a future

where technology becomes an indispensable ally in the pursuit of better health outcomes. At its core, the project seeks to harness the predictive prowess of machine learning, specifically employing the Random Forest Classifier, to revolutionize the landscape of disease diagnosis.

1.3 AIM

The aim of this study is to test the proposed hypothesis that supervised ML algorithms can improve health care by the accurate and early detection of diseases. In this study, we investigate studies that utilize more than one supervised ML model for each disease recognition problem. This approach renders more comprehensiveness and precision because the evaluation of the performance of a single algorithm over various study settings induces bias which generates imprecise results. The analysis of ML models will be conducted on few diseases located at heart, kidney, breast, and brain.

1.4 BACKGROUND

Traditional healthcare diagnostics often grapple with challenges related to time consumption, subjectivity, and the potential for human biases. In this era of technological evolution, there arises an opportunity to transcend these limitations through the fusion of data-driven approaches and medical science. This project's genesis lies in the recognition of the transformative potential that machine learning holds in expediting and enhancing the accuracy of disease prediction. By automating diagnostic processes, we aspire to create a healthcare paradigm that not only meets but exceeds contemporary standards.

1.5 OBJECTIVES

This project is underpinned by a set of overarching objectives that drive its development and implementation:

- **Automated Diagnosis:** Introduce an automated diagnostic system that outpaces conventional manual methods, offering swifter and more objective disease predictions.

- **Enhanced Accuracy:** Utilize machine learning algorithms, notably the Random Forest Classifier, to elevate the precision of disease predictions, mitigating the errors associated with subjective human assessments.
- **User-Friendly Interface:** Develop a user-friendly interface, powered by Tkinter, ensuring accessibility for both healthcare professionals and individuals seeking preliminary health assessments.

1.6 SCOPE

The scope of this project transcends the mere prediction of diseases; it extends to revolutionizing the diagnostic landscape. Envisaged as a versatile tool, the system accommodates a diverse range of symptoms and diseases, making it adaptable to various medical scenarios. By providing a user-friendly interface, the project endeavors to democratize healthcare insights, ensuring accessibility without compromising the depth and accuracy of disease predictions.

As we embark on a detailed exploration of this project's background, methodologies, software requirements, and potential impact, we aim to illuminate the trajectory of "Disease Prediction using Machine Learning" within the broader context of healthcare innovation. Through this exploration, we envisage a future where technology and healthcare seamlessly converge, offering unprecedented advancements in disease prediction and diagnostics.

Detection of the disease, numerous methodologies will be evaluated such as KNN, NB, DT, CNN, SVM, and LR. At the end of this literature, the best performing ML models in respect of each disease will be concluded.

CHAPTER 2

FEASIBILITY STUDY AND SYSTEM REQUIREMENTS

2.1 FEASIBILITY STUDY

The **feasibility study** for the Disease Prediction Model is a critical assessment to determine whether the project can be successfully developed, implemented, and sustained. It systematically evaluates the technical, operational, and economic dimensions of the project, ensuring that the proposed system meets its goals within the constraints of resources, time, and practicality. This section elaborates on these dimensions in detail, expanding their scope and providing insights into the project's potential viability.

2.1.1 Technical Feasibility

Technical feasibility focuses on determining whether the technological infrastructure, tools, and expertise required to build and operate the Disease Prediction System are available and adequate.

2.1.1.1. Core Technologies

The "Disease Prediction System" leverages state-of-the-art machine learning algorithms and tools.

- **Programming Language and Libraries:** Python, being versatile and developerfriendly, forms the backbone of this project. Libraries like Scikit-learn, Pandas, NumPy, and TensorFlow provide robust support for data manipulation, machine learning, and model evaluation.
- **Scikit-learn**: Ideal for implementing algorithms like Random Forest, Naïve Bayes, and Support Vector Machines.
- **TensorFlow and Keras**: Useful for deep learning models, such as Convolutional Neural Networks (CNN).
- **Tkinter:** Used for building a user-friendly graphical user interface (GUI).
- **Hardware Requirements**:

  The project requires minimal computational resources during deployment. For development and training, a standard system with the following specifications suffices:

  - Processor: Multi-core (i5 or higher).
  - RAM: At least 8 GB (16 GB recommended).
  - Storage: SSD with a capacity of at least 256 GB for fast data processing.

- Cloud environments such as Google Collab or AWS Sage Maker can be leveraged to train machine learning models efficiently.

- **Data Availability**: Public datasets containing disease-related information, such as symptoms and diagnoses, are critical. Repositories like the UCI Machine Learning Repository or Kaggle provide datasets to build and train models. Additionally, data from hospitals and clinics can be integrated for increased accuracy.

2.1.1.2. Model Selection

- **Algorithm Choice:**
  - Random Forest Classifier: Reliable for multi-class classification problems.
  - Naïve Baayes: Works well with categorical data like symptoms.
  - CNN: Exceptional for image-based diagnostics, such as X-rays or MRIs.
  - K-Nearest Neighbors (KNN): Effective for quick, low-complexity predictions.
  -
- **Model Training and Testing:** Models will undergo rigorous training using labeled datasets. Techniques like k-fold cross-validation ensure robustness and generalizability.

2.1.1.3. Software Environment

The system can be developed and deployed using:

- **Operating Systems:** Windows 11, Linux, or macOS.
- **Development Environments:** IDEs such as Jupyter Notebook, Spyder, and PyCharm.
- **Version Control:** GitHub or GitLab to maintain code integrity and manage collaborative development.

2.1.1.4. Scalability

The proposed system is highly scalable. It can handle larger datasets or more complex algorithms by:

- Upgrading hardware (e.g., GPU support for neural networks).
- Utilizing cloud computing for real-time data processing and prediction.

In conclusion, the technical infrastructure for the Disease Prediction System is wellestablished, making the project technically feasible.

2.1.2 Operational Feasibility

Operational feasibility assesses whether the system can function effectively in its intended environment, considering usability, stakeholder needs, and alignment with current practices.

2.1.2.1 Stakeholder Analysis

- **Primary Users:**
  - Individuals seeking preliminary diagnoses based on symptoms.
  - Healthcare professionals requiring quick diagnostic support.

- **Secondary Users:**
  - Researchers and data scientists analyzing trends in disease prevalence.
  - Hospital administrators optimizing patient workflows.

2.1.2.2 User Interface and Experience

The system's GUI is designed for simplicity and efficiency. Built using Tkinter, it includes:

- **Input Fields**: Allowing users to enter symptoms in a structured manner.
- **Output Display:** Providing disease predictions along with confidence scores.
- **Accessibility Features:** Ensuring compatibility with assistive technologies for differently-abled users.

2.1.2.3 System Integration

- **Healthcare Integration:** The system can be integrated into existing electronic health record (EHR) systems, enabling seamless data transfer.
- **Telemedicine Platforms**: This system can enhance online consultations by providing preliminary diagnoses.
- **API Availability:** A Flask API enables third-party integration, extending the system's reach to mobile apps and web services.

2.1.2.4 Ethical and Privacy Considerations

Given the sensitive nature of health data, the system adheres to strict privacy standards:

- Data Encryption: Ensuring secure transmission and storage of patient data.
- Compliance with Regulations: Adhering to HIPAA (Health Insurance Portability and Accountability Act) and GDPR (General Data Protection Regulation).

- Anonymization: Protecting patient identities in shared datasets.

2.1.2.5 Training and Support

Operational success depends on user education. Resources include:

- Online tutorials and documentation.
- Customer support helplines for troubleshooting.

With these measures, the project ensures operational feasibility, addressing both user needs and ethical considerations.

2.1.3 Economic Feasibility

Economic feasibility evaluates whether the benefits of the project outweigh its costs, making it financially viable.

2.1.3.1 Development Costs

The costs involved in the system's development include:

- Human Resources: Salaries for developers, data scientists, and domain experts.
- Hardware and Software: Acquisition of systems, cloud services, and licensed tools if necessary.

Estimated development cost breakdown:

- Initial Investment: $10,000 - $15,000 (including salaries, hardware, and software).
- Cloud Services: $200/month for training and hosting (scalable based on usage).

2.1.3.2 Maintenance Costs

- Regular Updates: Retraining models with new data.
- Bug Fixes and Feature Enhancements: Ensuring smooth functionality.
- Technical Support: Offering ongoing user assistance.

2.1.3.3 Revenue Generation

The system can generate revenue through:

- Licensing: Hospitals and clinics can subscribe to the service.
- Freemium Model: Basic features for free, with advanced analytics offered at a premium.

- API Monetization: Charging third-party developers for access to the prediction API.

2.1.3.4 Cost-Benefit Analysis Benefits

include:

- Reduction in healthcare costs by minimizing unnecessary hospital visits.
- Improved patient outcomes through early diagnosis.
- Revenue opportunities from commercial partnerships and licensing.

In summary, the system's economic benefits far outweigh its costs, ensuring financial feasibility.

2.1.4 Additional Considerations

2.1.4.1 Risk Analysis

- Technical Risks:
  - Insufficient data quality leading to inaccurate predictions.
  - Overfitting or bias in machine learning models.
- Operational Risks:
  - Resistance from healthcare professionals accustomed to traditional diagnostics.
- Mitigation Strategies:
  - Regular model evaluation and updates.
  - User feedback mechanisms to improve system performance.

2.1.4.2 Long-term Vision

The system's future enhancements include:

- Integration with wearable devices for real-time health monitoring.
- Expansion to include rare diseases and mental health conditions.
- Multilingual support for global accessibility.

2.2 SYSTEM REQUIREMNETS

Software Environment is a technical specification of requirement of software product. This specifies the surroundings for development, operation and protection of the product.

Technology used:

- Python
- Tkinter
- NumPy
- Pandas
- Scikit-Learn
- Random-Forest-Classifier
- Matplotlib

2.2.1 Python: Python is a high-level, interpreted, and general-purpose programming language. It was created by Guido van Rossum and first released in 1991. Python emphasizes readability and ease of use, and it has a simple and straightforward syntax that makes it accessible for beginners while maintaining power and flexibility for experienced developers. Key features of Python include:

1. Readability: Python code is designed to be easy to read and write, making it accessible for developers at all levels.
2. Interpreted: Python is an interpreted language, which means that the Python code is executed line by line, and there is no need for a separate compilation step.
3. High-level: Python abstracts many low-level details, making it easier to focus on solving problems rather than dealing with system-specific complexities.
4. Dynamically Typed: Python is dynamically typed, meaning that variable types are interpreted at runtime, making it more flexible but also requiring careful attention to variable types.
5. Versatile: Python supports multiple programming paradigms, including procedural, object-oriented, and functional programming.

2.2.2 Tkinter: Tkinter is the standard GUI (Graphical User Interface) toolkit for Python. It facilitates the creation of desktop applications with graphical interfaces. Tkinter is based on the Tk GUI toolkit and provides a set of tools and widgets for building interactive applications.

It is known for its simplicity, making it accessible for beginners, while also offering the flexibility for more advanced GUI development. Tkinter supports cross-platform development, enabling applications to run on Windows, macOS, and Linux. Developers can use various widgets such as buttons, labels, entry fields, and more to design the user interface. Tkinter follows an event-driven programming model where actions or events trigger specific functions. The integration of Tkinter with Python allows developers to seamlessly combine the power of Python with GUI elements. The toolkit is customizable, allowing developers to tailor the appearance and behaviour of widgets to suit the application's requirements. Overall, Tkinter is a versatile and widely used library for creating graphical user interfaces in Python.

2.2.3 NumPy: NumPy, short for Numerical Python, is a fundamental package for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of high-level mathematical functions to perform operations on these arrays. NumPy is a cornerstone library in the Python data science ecosystem and is extensively used in various fields such as physics, statistics, engineering, and machine learning.

NumPy's primary object is the `ndarray` (n-dimensional array), which is a flexible and efficient container for homogeneous data. It allows for mathematical operations on entire arrays without the need for explicit looping. This vectorized approach not only simplifies the code but also improves performance.

The library offers a plethora of mathematical functions, including linear algebra operations, Fourier analysis, random number generation, and more. It also supports broadcasting, a powerful mechanism that allows operations on arrays of different shapes and sizes, enhancing flexibility in computations.

NumPy is designed to seamlessly integrate with C, C++, and Fortran code, enabling users to incorporate performance-critical components into their Python programs.

2.2.4 Pandas: Pandas is an open-source data manipulation and analysis library designed for Python. It provides high-performance, easy-to-use data structures, namely Series and Data Frame, built on top of NumPy. Pandas is particularly suited for working with structured and tabular data.

The Data Frame, a two-dimensional labelled data structure, is a central component of Pandas. It allows for the organization and manipulation of data in a tabular form, similar to a spreadsheet or a SQL table. The Data Frame includes labelled axes (rows and columns), facilitating intuitive data indexing and manipulation.

Pandas excels in data cleaning, exploration, and analysis tasks. It offers functionalities for data alignment, missing data handling, grouping, aggregation, merging, and reshaping. The library is widely used in data science, finance, economics, and other domains where efficient data manipulation is crucial.

In summary, while NumPy provides a foundation for numerical operations and array manipulation, Pandas builds on top of it to offer high-level data structures and tools for data analysis in a tabular format. Together, they form a powerful combination for various scientific and data analysis applications in Python.

2.2.5 Scikit-Learn: Scikit-Learn, also known as sklearn, is an open-source machine learning library for Python. It is built on NumPy, SciPy, and Matplotlib and provides simple and efficient tools for data analysis and modelling. Scikit-Learn includes a wide array of machine learning algorithms for tasks such as classification, regression, clustering, dimensionality reduction, and more.

Key features of Scikit-Learn:

1. **Consistent Interface**: Scikit-Learn provides a unified and consistent interface for various machine learning algorithms. This consistency makes it easy to switch between different models without significant changes to the code.

2. **Data Pre-processing**: The library offers tools for pre-processing data, including handling missing values, scaling features, and encoding categorical variables.

3. **Model Evaluation:** Scikit-Learn includes functions for evaluating the performance of machine learning models through metrics such as accuracy, precision, recall, F1-score, and more.

4. **Cross-Validation**: Cross-validation techniques help in assessing a model's performance by splitting the data into multiple subsets for training and testing.

5. **Hyperparameter Tuning:** Scikit-Learn provides tools for hyperparameter tuning, allowing users to optimize model performance by selecting the best set of hyperparameters.

6. **Pipeline:** The `Pipeline` class enables the creation of a streamlined workflow, incorporating data pre-processing, feature engineering, and model training in a single, coherent process.

2.2.6 Random Forest Classifier: Random Forest is an ensemble learning method based on decision tree classifiers. It constructs a multitude of decision trees during training and outputs the class that is the mode of the classes (classification) or the mean prediction (regression) of the individual trees.

Key characteristics of Random Forest:

1. **Ensemble Learning:** Random Forest is an ensemble of decision trees. It builds multiple trees and merges them to get a more accurate and stable prediction.

2. **Bootstrap Aggregating (Bagging):** Each tree in the forest is trained on a random subset of the training data, and the final prediction is an average (classification) or a weighted average (regression) of the individual tree predictions.

3. **Feature Randomness:** Random Forest introduces randomness by considering only a subset of features at each split in a tree. This helps in decorrelating the trees and preventing overfitting.

4. **Robustness:** Random Forest is robust to outliers and noisy data. It tends to generalize well to unseen data, making it a popular choice for various machine learning tasks.

2.2.7 Matplotlib: It is a library for creating static, animated, and interactive visualizations in Python- programming language. In this project it is used for creating simple plots, sub-plots and its object is used alongside with the seaborn object to employ certain functions such as show, grid etc. A %matplotlib inline function is also used for providing more.

2.2.8 Working Environment

Hardware Configuration:

- Processor: P III 700 MHz
- RAM: 64 MB RAM
- Hard Disk Drive: 20 GB HDD
- Keyboard: 104 keys
- Mouse: HP Mouse
- Monitor: fourteen” digital colour monitor
- Display Type: VGA

Software Configuration:

- Operating System: Windows 11
- Web Browser: Chrome
- Designing Tool: Spyder

CHAPTER 3

SOFTWARE REQUIREMENT SPECIFICATION

3.1 FUNCTIONAL REQUIREMENT

- User registration (User/Organizer)
- User login/logout/update Profile
- Request for Blood
- Add News
- Contact Details
- Refer a friend.
- Search for event
- By Distance
- By District or City
- By Date or Time
- By House number or Phone number
- 24

3.2 NON-FUNCTIONAL REQUIREMENT

- Security
- Availability
- Performance
- User Satisfaction
- Backup

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION TO SYSTEM DESIGN

The identified entity or the characteristic may be entered within the code; conversely there are many System designs in the manner of planning a brand-new machine or to replace the present device. Simply, device layout is just like the blueprint for building, it specifies all the capabilities which can be within the completed product. System design segment follows device analysis section. Design is involved with figuring out functions, statistics streams among the one's functions, keeping a file of the layout choices and supplying a blueprint for the implementation phase. Design is the bridge between system analysis and system implementation. Some of the essential fundamental concepts involved in the design of application software are:

4.1.1 Abstraction Standard Verification

Abstraction is used to construct answers to hassle without having to take account of the tricky info of the diverse thing sub troubles. Abstraction lets in machine clothier to make stepwise purification, which at each level of the design may also hide useless details associated with representation or implementation from the surrounding.

Standard is involved with decomposing of principal module into accurately described attainable units with well-defined interfaces the various gadgets. This enhances layout clarity, which in turn eases implementation, Debugging, Testing, Documenting and Maintenance of the software product. Standard regarded on this feel is a valuable tool in the production of big software program tasks.

Verification is a fundamental concept in software layout. A design is verifiable if it could be confirmed that the design will bring about implementation that satisfies the consumer's necessities. Verification is of two kinds specifically. Verification that the software necessities evaluation satisfies the purchaser's desires. Verification that the layout satisfies the requirement analysis. Some of the vital factors of quality that are to be taken into consideration within the design of software program are:

4.1.2 Reliability

The software program must behave strictly in line with the unique specification and ought to be characteristic easily under everyday conditions.

4.1.3 Docility

The software program needs to be able to adapt without difficulty to modifications in the specification.

4.1.4 Recyclable

The software program ought to be evolved using a modular technique, which allows modules to be reused by way of other utility, if practical. System Design briefly describes the idea of gadget design, and it has four sections. The first segment briefly describes the capabilities that the device is going to provide to the consumer and the outputs that the proposed device is going to provide. The second phase namely Logical Design describes the Data Flow Diagrams, which clearly show the information actions, the procedures and the statistics sources, and sinks, E-R diagrams which are the general logical design of the database, and high degree method structure of the device. The procedure of layout includes “conceiving and making plans out in the mind” and making a drawing pattern, or comic strip of the device. In software program design there are two styles of essential sports, Conceptual Design and Detailed Design. Conceptual or logical or outside layout of software program involves conceiving, making plans out, and specifying the externally observable characteristics of a software product. These characteristics encompass person presentations, outside data assets, practical traits and high-degree method structure for the product. Details or inner layout entails conceiving, making plans out, and specifying the inner shape and processing info of the software program product. The purpose of internal layout is to specify internal shape, processing details, blueprint of implementation, checking out, and maintenance sports. One of the critical fundamental standards of software design is general. A standard device is composed of interfaces for a few of the units. Well known complements layout readability, which in turn eases implementation, debugging, checking out, documentation, and maintenance of the software product. The different fundamental standards of software design encompass abstraction, structure, facts hiding, concurrency and verification. The use of structuring lets in decomposition of a big device into smaller, extra potential gadgets with nicely described

relationships to the alternative gadgets. The device design is verifiable if it may be assessed that the design will result in an implementation that satisfies the purchaser's requirements.

4.1.5 Preliminary Design

Preliminary design is involved with deriving an average picture of the device. Deriving entire system into modules and submodules at the same time as preserving Cohesion and Coupling factors in thoughts. Tools, which help in the first layout procedure, are Data Flow Diagrams.

4.1.6 Code design

The motive of code is to ease the identification and retrieval of gadgets of statistics. A code is an ordered collection of symbols designed to supply specific identification of an entity or characteristic. To obtain identification there should be best one place wherein there is an area within the code for each aspect this is to be recognized. This jointly one of a kind characteristics need to be built into any coding machine. The codes for this gadget are designed with two features in thought. Optimum human orientated use and system efficiency. Length of the code range from length of one to period of five characteristics:

The code shape is precise; making sure that handiest one fee of the code with a single that means may be correctly applied to a given entity or attributes.

The code shape is expansible considering the boom of its set of entities and attributes. The code is concise and brief for recording, verbal exchange, transmission and storage policies.

They have a uniform size and format. The codes are simple so that the consumer can effortlessly apprehend it. The codes also are versatile i.e., it is easy to alter to mirror vital changes in condition, chart scrappy and relationships of the encode entities.

The codes also are effortlessly garage for producing reviews in a predetermined order of layout. The codes are also stable and do not require being often up to date thereby selling consumer efficiency. The codes are also meaningful. They are also operable i.e., they're good enough for gift and anticipate records processing both for gadget and human use.

4.1.7 Input Design:

Input layout is part of normal system design, which requires incredibly careful attention. The essential targets of input design are:

- To produce a value-powerful method of input. To gain the highest possible level of accuracy.
- To make certain that the input is acceptable to and understood by using the consumer staff.

In this gadget input monitors are designed very cautiously so that no faulty statistics will be inputted in the database. The facts are made as clean as possible. For simplifying the records access many facilities are given. Each display in this gadget is eased by many controls so that the user can without problems work with this device.

4.1.8 Output Design

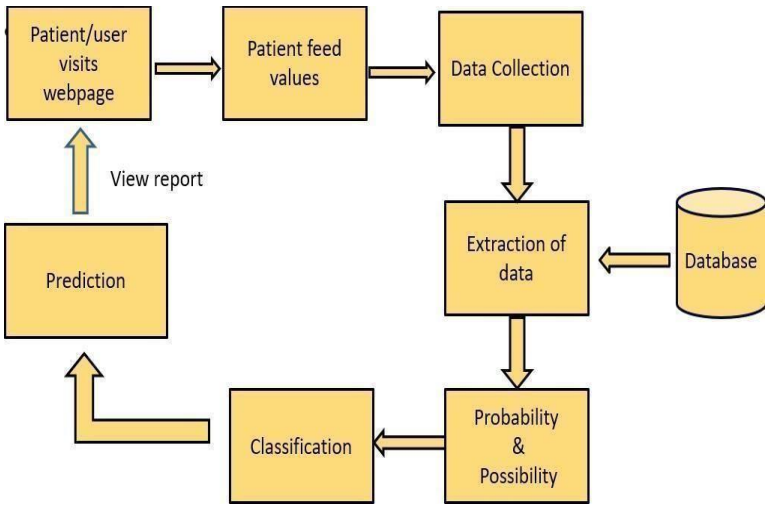
Outputs from systems are required normally to communicate the outcomes of processing to users. They are also to supply a permanent tough copy of those results for the later session.

The various kinds of outputs are needed through this device are given below:

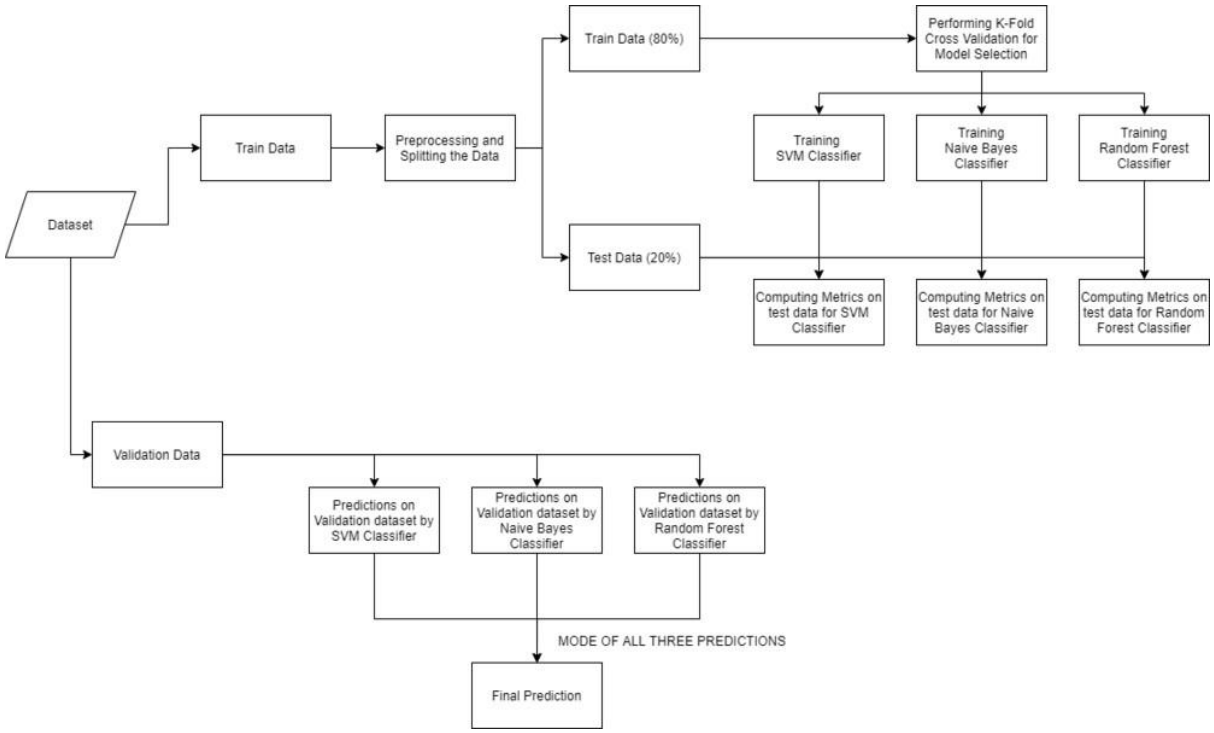
External outputs, whose vacation spot is out of doors the concern, and which require powerful attention because they, assignment the photograph of the concern. Internal outputs, whose vacation spot is in the subject, and which require careful design due to the fact they may be the consumer's most important interface within the system.

Operation outputs, whose use is only inside the computer branch, E.G., application listings, use statistics and many others, Interactive outputs, which includes the user in speaking directly with the computers.

4.2 DATA-FLOW DIAGRAM

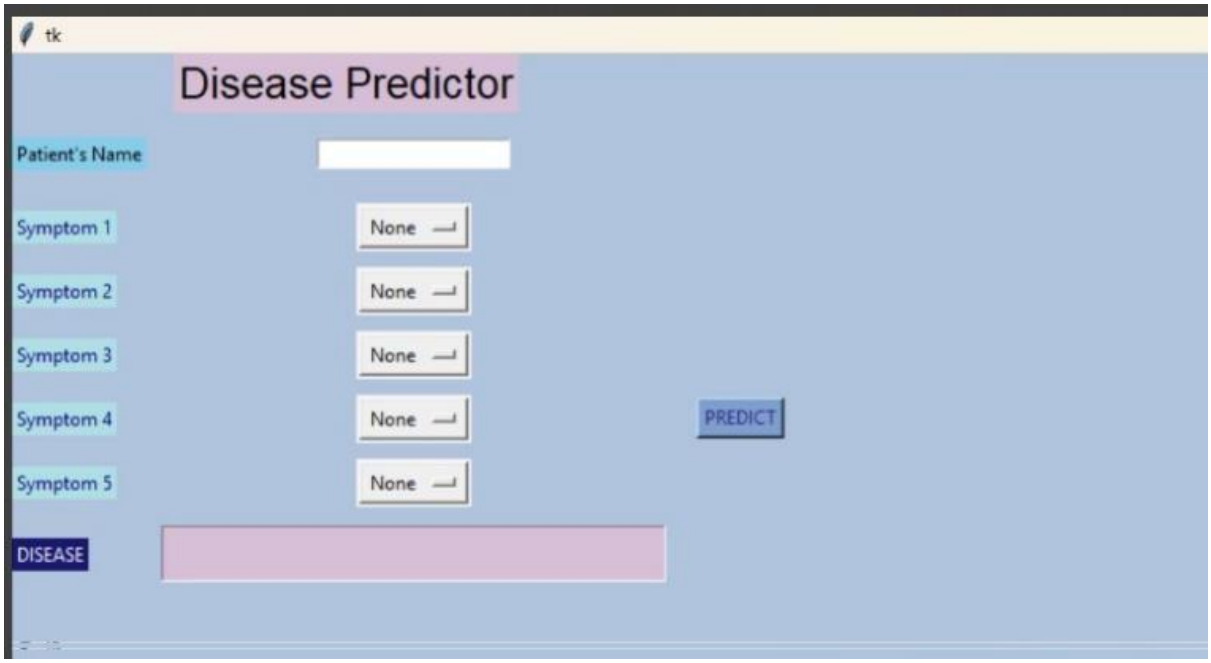


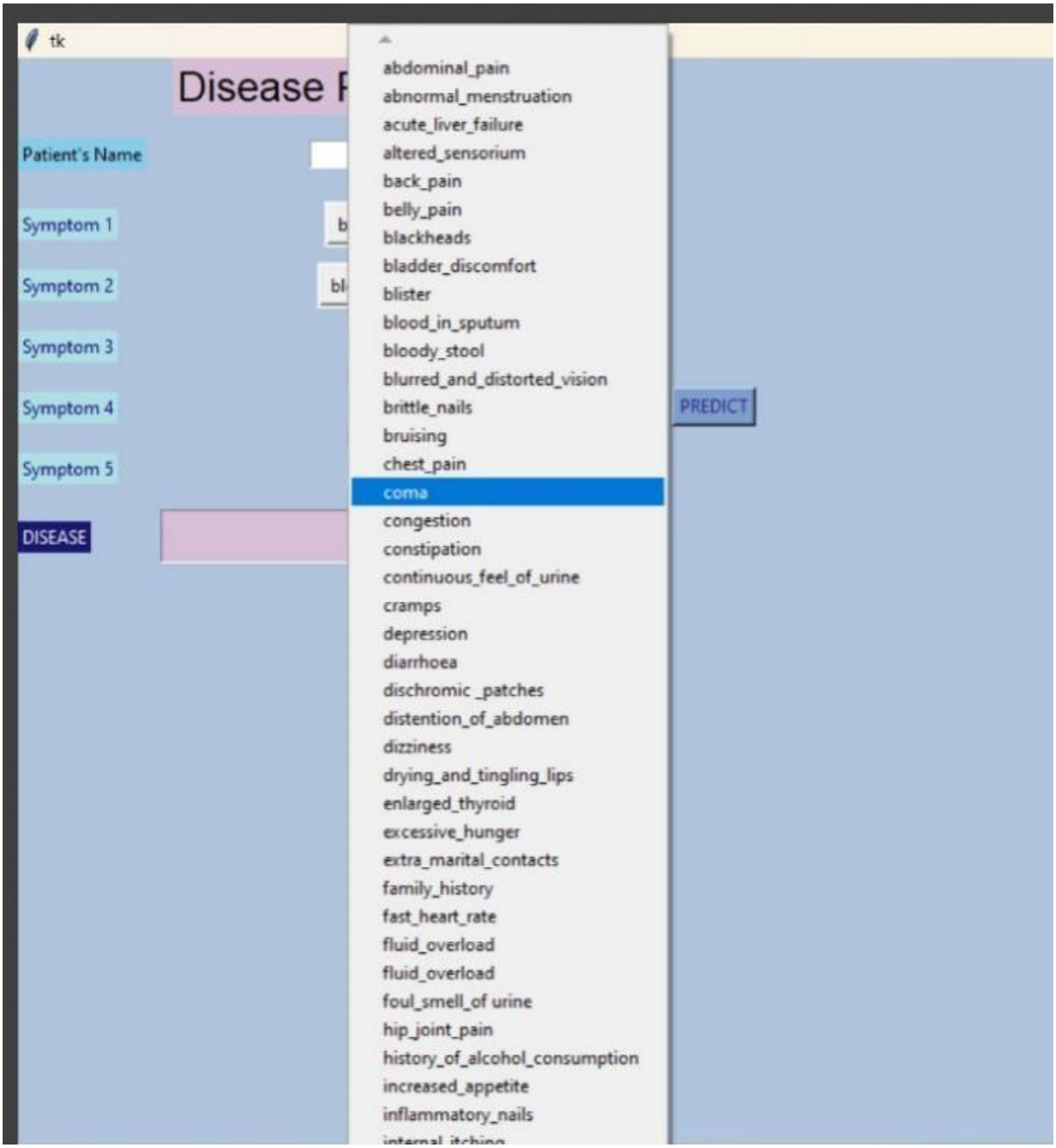
4.2 SYSTEM FLOW DIAGRAM



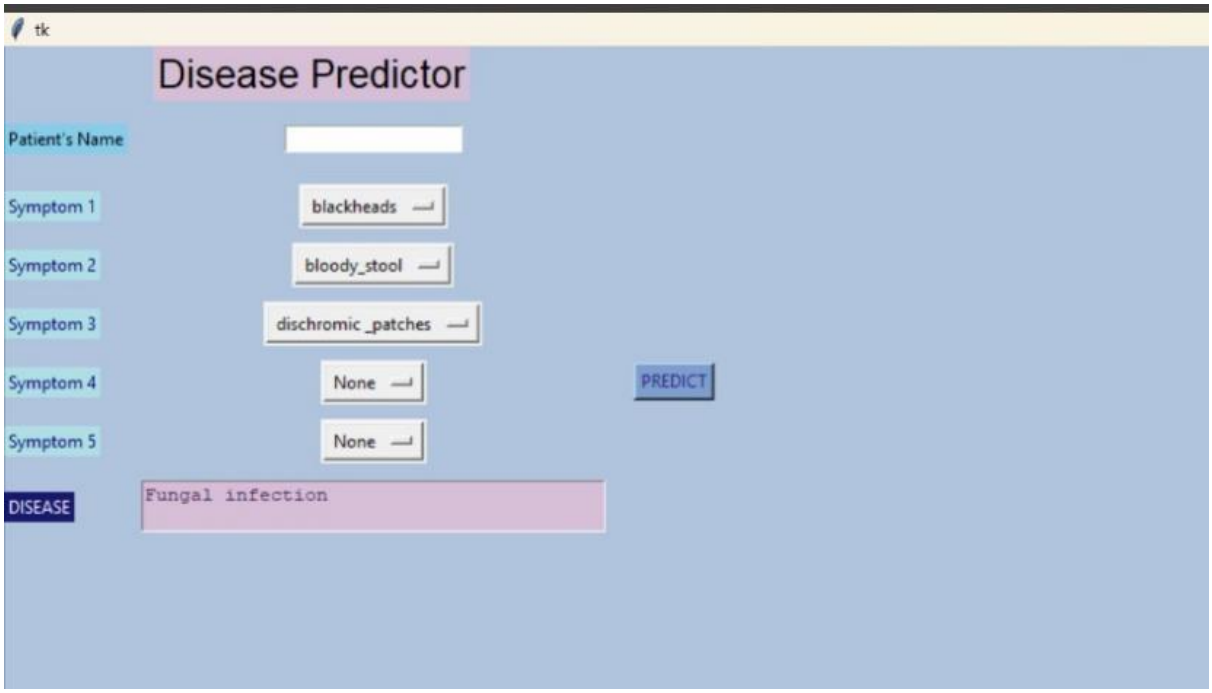
INPUT and OUTPUT DESIGN

5.1 INPUT SCREENSHOTS




tk
Disease P
Patient's Name
Symptom 1
Symptom 2
Symptom 3
Symptom 4
Symptom 5
DISEASE
PREDICT
abdominal_pain
abnormal_menstruation
acute_liver_failure
altered_sensorium
back_pain
belly_pain
blackheads
bladder_discomfort
blister
blood_in_sputum
bloody_stool
blurred_and_distorted_vision
brittle_nails
bruising
chest_pain
coma
congestion
constipation
continuous_feel_of_urine
cramps
depression
diarrhoea
dischromic _patches
distention_of_abdomen
dizziness
drying_and_tingling_lips
enlarged_thyroid
excessive_hunger
extra_marital_contacts
family_history
fast_heart_rate
fluid_overload
fluid_overload
foul_smell_of urine
hip_joint_pain
history_of_alcohol_consumption
increased_appetite
inflammatory_nails

5.2 OUTPUT SCREENSHOTS



CHAPTER 5

SYSTEM TESTING

System testing is the stage earlier than machine implementation in which the device is made mistakes free, and all the needed adjustments are made. The gadget turned into examined with look at information and essential corrections to the machine had been executed. All the reviews were checked with the aid of the person and accepted. The machine became very user friendly with on-line aid to help the user anywhere essential.

Test Plan:

A test plan is a preferred record for the whole undertaking, which defines the scope, technique to be taken, and schedule of checking out, in addition to finding the check object for the complete trying out system, and the personally liable for the special activities of trying out. This report describes the plan for testing the understanding management device.

Major testing activities are:

- Test units
- Features to be tested Approach for testing Test outputs Schedule.
- Personal allocation

Test units:

Test Case specification is an important activity inside the checking out manner. In this undertaking, I have carried out stages of testing.

Unit testing System trying out.

The fundamental devices in Unit trying out are:

- Validating the user request
- Validating the enter given via the person Exception dealing with
- The simple units in System trying out are:
- Integration of all applications is correct or not?

- Checking whether the whole device after integrating is running as expected. The gadget is tested as entire after the unit testing.

Other Testing Strategies:

Alpha Testing:

This turned into carried out at the developer's web page via client. The software is used in an herbal placing with the developer "searching over the shoulder" of the consumer and recording errors and use issues. Alpha checks are performed in controlled surroundings.

CHAPTER 6

EVALUATION

The Disease Prediction System underwent thorough evaluation during the software-testing phase, ensuring each module was meticulously examined for bugs and the accuracy of output. The system's development embraced user-friendliness, with comprehensive documentation provided as online help when needed. The evaluation process involved formal reviews encompassing all system components.

The complete gadget turned into developed the use of the Python, NumPy, Pandas, Tkinter, and Scikit-learn as again end. Python is used to write object-oriented code. NumPy is used for numerical and mathematical operations. Pandas is used to provide data structures for efficient data manipulation and analysis. Tkinter is used to provide a set of tools and libraries for creating graphical interfaces and windows-based applications. Tkinter is widely used due to its simplicity, ease of integration with Python, and cross-platform support. Scikit-learn is used to provide simple and efficient tools for data analysis and modelling, including various machine learning algorithms and utilities. Hence the design of the complete device is user-friendly and easy the implementation has been quite clean.

CHAPTER 7

CONCLUSION

This undertaking has given me an okay opportunity to layout, code, check and implement a utility. This has helped in setting into practice diverse Software Engineering principles and Database Management standards like keeping integrity and consistency of information. Further, this has helped me to examine ORACLE eight, ASP 2. Zero, HTML, VB Script, Adobe Photo store.

7.Zero and Personal Web Server.

I thank my guide for his priceless contribution in guiding me via out the assignment. I additionally thank my dad and mom and friends who have supported and influenced me to complete this mission successfully.

REFERENCES

- S. Cheema, S. Srivastava, P. K. Srivastava and B. K. Murthy, "A standard compliant Blood Bank Management System with enforcing mechanism," 2015 International Conference on
- Computing, Communication and Security (ICCCS)
- M. Sarode, A. Ghanekar, S. Krishnadas, Y. Patil and M. Parmar, "Intelligent Blood Management System," 2019 IEEE Bombay Section Signature Conference (IBSSC), Mumbai, India, 2019
- M. Y. Esmail and Y. S. H. Osman, "Computerized Central Blood Bank Management System (CCBBMS)," 2018 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE), Khartoum, Sudan, 2018
- A. S. Cheema, S. Srivastava, P. K. Srivastava and B. K. Murthy, "A standard compliant Blood Bank Management System with enforcing mechanism," 2015 International Conference on Computing, Communication and Security (ICCCS), Pointe aux Piments, Mauritius, 2015