**Major Project Report**

**on**

**RESPIRATORY ANALYSIS DETECTION OF VARIOUS LUNG INFECTIONS USING COUGH SIGNAL**

**Submitted in partial fulfilment of the degree of the**

**Bachelor of Technology in**

# COMPUTER SCIENCE AND ENGINEERING

**BY**

B. VARSHITHA 19C41A05B9

G. LIKHITHA 19C41A05A2

K. AKHIL 19C41A05A7

Under the Guidance of

Dr V. Janaki

Professor



**Department of computer Science and Engineering**

# JAYAMUKHI INSTITUTE OF TECHNOLOGICAL SCIENCES

**NARSAMPET, WARANGAL - 506 332**

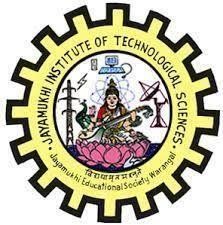
**(Autonomous, Affiliated to JNTUH, Accredited by NBA, NAAC-A)**

# JAYAMUKHI INSTITUTE OF TECHNOLOGICAL SCIENCES

**NARSAMPET, WARANGAL - 506 332**

**(Autonomous, Affiliated to JNTUH, Accredited by NBA, NAAC-A)**

# CERTIFICATE



This is to certify that the project report entitled “**RESPIRATORY ANALYSIS DETECTION OF VARIOUS LUNG INFECTIONS USING COUGH SIGNAL**” is Bonafede work of the students **B. VARSHITHA (19C41A05B9), G. LIKHITHA (19C41A05A2) & K. AKHIL (19C41A05A7)** submitted in partial fulfilment of the requirement for the award of the degree of bachelor of Technology in COMPUTER SCIENCE & ENGINEERING during the academic year 2022-2023.

**GUIDE**  **Head, Department**

**PRINCIPAL**

**ACKNOWLEDGEMENT**

We would especially like to express our extreme gratitude and sincere thanks to our guide **Dr V. Janaki**, Principal, Department of Computer Science and Engineering for her enthusiastic and innovative guidance and support.

We are extremely thankful to **Dr P. Srinivas Rao**, Head, Department of Computer Science and Engineering for the support and encouragement accorded to carry out this project work.

It gives us immense pleasure in expressing sincere and deep sense of gratitude the **Prof.V. Janaki,** Principal, Jayamukhi Institute of Technological sciences for the facility made available for the progress and completion of our work.

We are also thankful to our management for providing all the facilities for completing the project.

**THANK YOU**

1. **VARSHITHA**

**G. LIKHITHA**

1. **AKHIL**

# ABSTRACT

Large number of people die every year of Pulmonary chronic lung diseases irrespective of their age. Lung sound analysis has been a key diagnostic aid to accurately detect pulmonary diseases. Earlier, manual detection was used which was not a dependable method to detect lung diseases due to various reasons like low audibility and difference in perceptions of different physicians for different sounds. Modern computerized analysis yield results with much higher accuracy and thus a better treatment can be given to patients suffering from various kinds of lung diseases. These disorders include Asthma, Bronchitis, Emphysema, Tuberculosis and Pneumonia. Some of the symptoms are wheezing, shortness of breath, bronchi and chronic cough. In this project we are using respiratory audio data-set to predict various diseases such as Asthma, Pneumonia, Bronchitis and many more. To implement this project, we have taken disease diagnosis data-set and respiratory audio data-set and then extract features from all audio data set and then trained a convolution neural network (CNN) algorithm model. After training model, we can upload any new test data to predict disease from it.

**CONTENTS**

**TITLES PAGE NO**

1. **INTRODUCTION 1**
2. **LITERATURE SURVEY 2 -8**
   1. KEY DUPLICATION WITH IBBE
   2. SERVER LESS DISTRIBUTED FILE SYSTEM
   3. THE GOOGLE FILE SYSTEM
   4. CONVERGENT KEY MANAGEMENT
   5. SOFTWARE ENVIRONMENT
   6. WHY CHOOSE PYTHON
3. **SYSTEM ANALYSIS 9**
   1. EXISTING SYSTEM
   2. PROPOSED SYSTEM
4. **FEASIBILITY STUDY 10**
   1. ECONOMICAL FEASIBILITY
   2. TECHNICAL FEASIBILITY
   3. SOCIAL FEASIBILITY
5. **SYSTEM REQUIREMENTS 11**

5.1 HARDWARE REQUIREMENTS

5.2 SOFTWARE REQUIREMENTS

1. **SYSTEM DESIGN 12 - 18**
   1. SYSTEM ARCHITECTURE
   2. DATAFLOW DIAGRAM
   3. UML DIAGRAMS
2. **IMPLEMENTATION** 19
   1. MODULES
   2. SAMPLE CODE
3. **SYSTEMTESTING**  20 - 23
   1. UNITTESTING
   2. INTEGRATION TESTING
   3. ACCEPTANCETESTING
4. **INPUT DESIGN AND OUTPUT DESIGN** 24 - 25
   1. INPUTDESIGN
   2. OUTPUTDESIGN
5. **SCREENSHOTS**   **26 - 30**
6. **CONCLUSION 31**
7. **BIBLOGRAPHY 32**

## 1. INTRODUCTION

Pulmonary disorder is the inability of a person to breathe normally. Manual analysis used in the past only gave an approximate idea of the disorder and hence a very rough treatment was given. This was working out well in the past. Drastic increase in pollution and non-healthy habits of people has given rise to more complex diseases and need a very accurate estimation of the extent of disease. This accuracy can only be bought by automation of the analysis. Researchers observed that the difference between sounds made by infected lungs and the normal healthy lungs could serve as a very good tool for the detailed study and detection of the disease. Recording the Lung sounds, filtering them from the Heart sounds and other noises and studying the waveform of the filtered Lung sound has been the de facto way of performing the analysis. Many methods are given for the filtering and processing of the Lung sounds. Brief review of the previous papers reveals several methods of filtering and examining the LS. The most challenging task in the analysis is the separation of HS from the LS due to spectral and temporal overlap between the two sounds. Filtering techniques used are Modulation Domain filtering [5] that filters the temporal trajectories of short-term spectral components. Signal analysis is carried out by segmentation into consecutive overlapping frames and performing Fourier transform. Combination of adaptive-frequency domain filtering [10] where a very simple method is described that involves subtracting heart sounds from combination of heart and lung sound.

## 2. LITERATURE SURVEY

### **2.1 Respiratory sound analysis for diagnostic information**

**Authors: Rutuja Mhetre and U.R. Bagal**

**Abstract:**  The most important concern in the medical domain is to consider the interpretation of data and perform accurate diagnosis. The bronchitis, pneumonia and many other pulmonary diseases causes respiratory disorders which affects respiratory systems. Diagnosis of these diseases is facilitated by pulmonary auscultation by using stethoscope. This method depends on individuals hearing capability, experience and ability to differentiate the sounds. The quantitative measurement and permanent record of the related parameters is difficult. The recording and analysis of the respiratory sounds may quantify the changes in abnormal respiratory sounds in respiratory disorder. The signal processing techniques may be used for diagnostic information.

**2.2Automatic Wheezing Detection Based on Signal Processing of**

**Spectrogram and Back - Propagation Neural Network**

**Authors: Bor-Shing Lin, Huey-Dong Wu and Sao-Jie Chen**

**Abstract:** Wheezing is a common clinical symptom in patients with obstructive pulmonary diseases such as asthma. Automatic wheezing detection offers an objective and accurate means for identifying wheezing lung sounds, helping physicians in the diagnosis, long-term auscultation, and analysis of a patient with obstructive pulmonary disease. This paper describes the design of a fast and high-performance wheeze recognition system. A wheezing detection algorithm based on the order truncate average method and a back-propagation neural network (BPNN) is proposed. Some features are extracted from processed spectra to train a BPNN, and subsequently, test samples are analysed by the trained BPNN to determine whether they are wheezing sounds. The respiratory sounds of 58 volunteers (32 asthmatic and 26 healthy adults) were recorded for training and testing. Experimental results of a qualitative analysis of wheeze recognition showed a high sensitivity of

0.946 and a high specificity of 1.0.

**2.3 Cough sound analysis and objective correlation with spirometry and clinical diagnosis**

**Authors:** **Gowrisree Rudraraju,Shubha Deepti Palreddy**

**Abstract:** In India, there are 100 million people who suffer from various

respiratory problems; globally it is about 1–1.2 billion. The main problem attributed to the prevalence of respiratory diseases is lack of cost-effective and lab-free methods for early diagnosis. Spirometry is the standard clinical test procedure for detection of respiratory problems, but it requires repetition, and is also expensive and not available in rural areas. Cough sounds carry vital information about the respiratory system and the pathologies involved. Through this study, we detail how a combination of standard signal processing features and domain-specific features play a key role in distinguishing cough patterns. We could establish a relationship between cough pattern and respiratory conditions including widened airway, narrowed airway, fluid filled air sacs, and stiff lungs. Further, cough sound characteristics are correlated to the airflow parameters of spirometry. Our results show strong correlation of cough sound characteristics with airflow characteristics including FEV1, FVC and their ratios, which are important in identifying the type of lung diseases as either obstructive (obstruction in airway) or restrictive (restricts lung expansion). We have constructed a machine learning model to predict obstructive versus restrictive pattern, and validated it using K-fold cross-validation based on ground truth data. With a pattern prediction accuracy of 91.97%, sensitivity of 87.2%, and specificity of 93.69%, our results are encouraging.

**2.4 Detection of Cough and Adventitious Respiratory Sounds in Audio**

**Recordings by Internal Sound Analysis**

**Authors: Bruno Rocha, L. Mendes**

**Abstract:** We present a multi-feature approach to the detection of cough and adventitious respiratory sounds. After the removal of near-silent segments, a vector of event boundaries is obtained and a proposed set of 126 features is extracted for each event. Evaluation was performed on a data set comprised of internal audio recordings from 18 patients. The best performance (F-measure = 0.69 ± 0.03; specificity = 0.90 ± 0.01) was achieved when merging wheezes and crackles into a single class of adventitious respiratory sounds.

**2.5 SOFTWARE ENVIRONMENT**

**Python** is a high-level, interpreted scripting language developed in the late 1980s by Guido van Rossum at the National Research Institute for Mathematics and Computer Science in the Netherlands. The initial version was published at the alt. Sources [newsgroup](https://en.wikipedia.org/wiki/Usenet) in 1991, and version 1.0 was released in 1994.

Python 2.0 was released in 2000, and the 2.x versions were the prevalent releases until December 2008. At that time, the development team made the decision to release version 3.0, which contained a few relatively small but significant changes that were not backward compatible with the 2.x versions. Python 2 and 3 are very similar, and some features of Python 3 have been back ported to Python 2. But in general, they remain not quite compatible.

Both Python 2 and 3 have continued to be maintained and developed, with periodic release updates for both. As of this writing, the most recent versions available are 2.7.15 and 3.6.5. However, an official [End of Life date of January 1, 2020](https://pythonclock.org/) has been established for Python 2, after which time it will no longer be maintained. If you are a newcomer to Python, it is recommended that you focus on Python 3, as this tutorial will do.

Python is still maintained by a core development team at the Institute, and Guido is still in charge, having been given the title of BDFL (Benevolent Dictator For Life) by the Python community. The name Python, by the way, derives not from the snake, but from the British comedy troupe [Monty Python’s Flying Circus,](https://en.wikipedia.org/wiki/Monty_Python%27s_Flying_Circus) of which Guido was, and presumably still is, a fan. It is common to find references to Monty Python sketches and movies scattered throughout the Python documentation.

**2.6 WHY CHOOSE PYTHON**

If you’re going to write programs, there are literally dozens of commonly used languages to choose from. Why chooses Python? Here are some of the features that make Python an appealing choice.

**Python is Popular**

Python has been growing in popularity over the last few years. The 2018 [Stack Overflow Developer Survey](https://insights.stackoverflow.com/survey/2018) ranked Python as the 7th most popular and the number one most wanted technology of the year. [World-class software development countries around the globe use Python every single day](https://realpython.com/world-class-companies-using-python/).

According to [research by Dice](https://insights.dice.com/2016/02/01/whats-hot-and-not-in-tech-skills/) Python is also one of the hottest skills to have and the most popular programming language in the world based on the [Popularity of Programming Language Index](https://pypl.github.io/PYPL.html).

Due to the popularity and widespread use of Python as a programming language, Python developers are sought after and paid well. If you’d like to dig deeper into [Python salary statistics and job opportunities, you can do so her](https://dbader.org/blog/why-learn-python)e.

**Python is interpreted**

Many languages are compiled, meaning the source code you create needs to be translated into machine code, the language of your computer’s processor, before it can be run. Programs written in an interpreted language are passed straight to an interpreter that runs them directly.

This makes for a quicker development cycle because you just type in your code and run it, without the intermediate compilation step.

One potential downside to interpreted languages is execution speed. Programs that are compiled into the native language of the computer processor tend to run more quickly than interpreted programs. For some applications that are particularly computationally intensive, like graphics processing or intense number crunching, this can be limiting.

In practice, however, for most programs, the difference in execution speed is measured in milliseconds, or seconds at most, and not appreciably noticeable to a human user. The expediency of coding in an interpreted language is typically worth it for most applications.

### **Python is Free**

The Python interpreter is developed under an OSI-approved open-source license, making it free to install, use, and distribute, even for commercial purposes.

A version of the interpreter is available for virtually any platform there is, including all Flavors of Unix, Windows, macOS, smart phones and tablets, and probably anything else you ever heard of. A version even exists for the half dozen people remaining who use OS/2.

### **Python is Portable**

Because Python code is interpreted and not compiled into native machine instructions, code written for one platform will work on any other platform that has the Python interpreter installed. (This is true of any interpreted language, not just Python.)

### **Python is Simple**

As programming languages go, Python is relatively uncluttered, and the developers have deliberately kept it that way.

A rough estimate of the complexity of a language can be gleaned from the number of keywords or reserved words in the language. These are words that are reserved for special meaning by the compiler or interpreter because they designate specific builtin functionality of the language.

Python 3 has 33 keywords, and Python 2 has 31. By contrast, C++ has 62, Java has 53, and Visual Basic has more than 120, though these latter examples probably vary somewhat by implementation or dialect.

Python code has a simple and clean structure that is easy to learn and easy to read. In fact, as you will see, the language definition enforces code structure that is easy to read.

But It’s Not That Simple For all its syntactical simplicity, Python supports most constructs that would be expected in a very high-level language, including complex dynamic data types, structured and functional programming, and [object-oriented programming.](https://realpython.com/python3-object-oriented-programming/)

Additionally, a very extensive library of classes and functions is available that provides capability well beyond what is built into the language, such as database manipulation or GUI programming.

Python accomplishes what many programming languages don’t: the language itself is simply designed, but it is very versatile in terms of what you can accomplish with it.

#### **Conclusion**

This section gave an overview of the **Python** programming language, including:

* A brief history of the development of Python
* Some reasons why you might select Python as your language of choice

Python is a great option, whether you are a beginning programmer looking to learn the basics, an experienced programmer designing a large application, or anywhere in between. The basics of Python are easily grasped, and yet its capabilities are vast. Proceed to the next section to learn how to acquire and install Python on your computer.

**Python** is an [open source](https://simple.wikipedia.org/wiki/Open_source) [programming language](https://simple.wikipedia.org/wiki/Programming_language) that was made to be easy-to-read and powerful. A [Dutch](https://simple.wikipedia.org/wiki/Netherlands) programmer named [Guido van Rossum](https://simple.wikipedia.org/wiki/Guido_van_Rossum) made Python in 1991. He named it after the television show [Monty Python's Flying Circus.](https://simple.wikipedia.org/wiki/Monty_Python%27s_Flying_Circus) Many Python examples and tutorials include jokes from the show.

Python is an interpreted language. Interpreted languages do not need to be [compiled](https://simple.wikipedia.org/wiki/Compiled_language) to run. A program called an [interpreter](https://simple.wikipedia.org/wiki/Interpreter_(computing)) runs Python code on almost any kind of computer. This means that a programmer can change the code and quickly see the results. This also means Python is slower than a compiled language like [C,](https://simple.wikipedia.org/wiki/C_(programming_language)) because it is not running [machine code](https://simple.wikipedia.org/wiki/Machine_code) directly.

Python is a good programming language for beginners. It is a high-level language, which means a programmer can focus on what to do instead of how to do it. Writing programs in Python takes less time than in some other languages.

Python drew inspiration from other programming languages like C, [C++,](https://simple.wikipedia.org/wiki/C%2B%2B) [Java,](https://simple.wikipedia.org/wiki/Java_(programming_language)) [Perl,](https://simple.wikipedia.org/wiki/Perl) and [Lisp.](https://simple.wikipedia.org/wiki/LISP)

Python has a very easy-to-read syntax. Some of Python's syntax comes from C, because that is the language that Python was written in. But Python uses whitespace to delimit code: spaces or tabs are used to organize code into groups. This is different from C. In C, there is a [semicolon](https://simple.wikipedia.org/wiki/Semicolon) at the end of each line and curly braces ({}) are used to group code. Using whitespace to delimit code makes Python a very easy-to-read language.

**Python use [change / change source]**

Python is used by hundreds of thousands of programmers and is used in many places. Sometimes only Python code is used for a program, but most of the time it is used to do simple jobs while another programming language is used to do more complicated tasks.

Its [standard library](https://simple.wikipedia.org/w/index.php?title=Standard_library&action=edit&redlink=1) is made up of many [functions](https://simple.wikipedia.org/wiki/Computable_function) that come with Python when it is installed. On the [Internet](https://simple.wikipedia.org/wiki/Internet) there are many other [libraries](https://simple.wikipedia.org/w/index.php?title=Library_(computing)&action=edit&redlink=1) available that make it possible for the Python language to do more things. These libraries make it a powerful language; it can do many different things.

Some things that Python is often used for are:

* Web development
* Scientific programming

• Desktop [GUIs](https://simple.wikipedia.org/wiki/GUI)

* Network programming
* [Game](https://simple.wikipedia.org/wiki/Video_game) programming

## 

## 3. SYSTEM ANALYSIS

**3.1 EXISTING SYSTEM:**

The lungs are important organs in the respiratory system and used for gas exchange (oxygen and carbon dioxide). When we breathe. Our lungs transfer oxygen from the air into the blood, and carbon dioxide from the blood into the air. Cough is the most common symptom of several respiratory diseases. Cough is a defence mechanism of the body which prevents the respiratory tract from inhaling foreign materials accidentally or those produced internally by infection, it is characterized as wet when the sounds carry features indicative of mucus; in the absence of perceivable wetness, it is called dry. Changes in the character of the cough sound can reflect pathological situations in the lungs. Pathological situations arise due to some conditions like obstruction, restriction, and combined patterns.

**3.2 PROPOSED SYSTEM:**

In this project we are using respiratory audio data-set to predict various diseases such as Asthma, Pneumonia, Bronchitis and many more. To implement this project, we have taken disease diagnosis data-set and respiratory audio data-set and then extract features from all audio data-set and then trained a convolution neural network (CNN) algorithm model. After training model, we can upload any new test data to predict disease from it.

## 

## 4. FEASIBILITY STUDY

The feasibility of the project is analyzed 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
  1. **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.

* 1. **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.

* 1. **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.

**5. SYSTEM REQUIREMENTS**

* 1. **HARDWARE REQUIREMENTS:**

|  |  |  |  |
| --- | --- | --- | --- |
| • | System | : | Pentium Dual Core. |
| • | Hard Disk | : | 120 GB. |
| • | Monitor | : | 15’’ LED |
| • | Input Devices | : | Keyboard, Mouse |
| • | Ram | : | 1 GB |

* 1. **SOFTWARE REQUIREMENTS:**

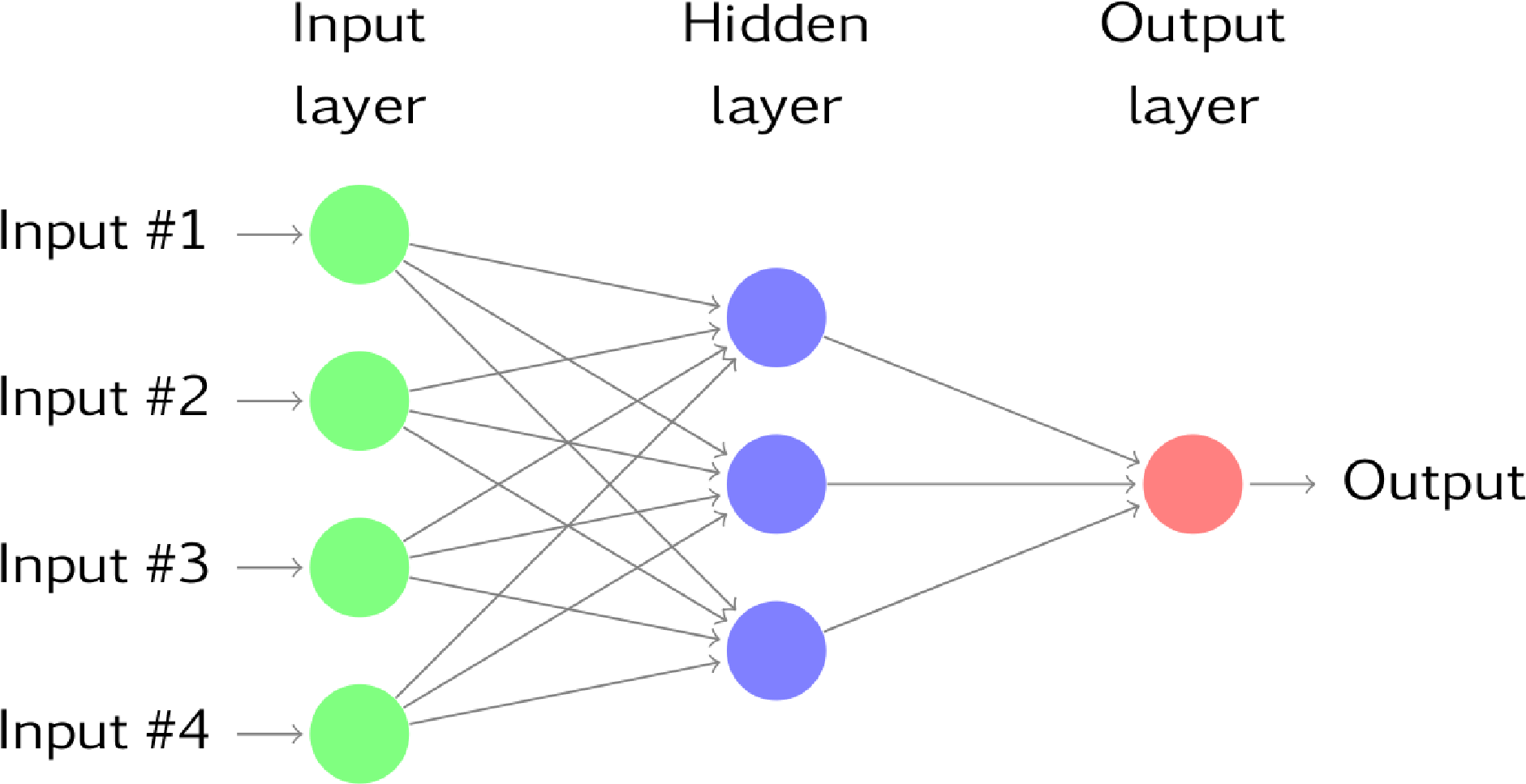
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| • | Operating system |  | : | Windows 10 |
| • | Coding Language |  | : | python |
| • | Tool |  | : | PyCharm |
| • | Database |  | : | MYSQL |
| • | Server |  | : | Flask |

### 

### **6. SYSTEM DESIGN**

**6.1 SYSTEM ARCHITECTURE:**

See below image with layers



**6.2 DATA FLOW DIAGRAM:**

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

**DATA FLOW DIAGRAM:**

**User**

**Check**

**Unauthorized user**

**YES NO**

Upload Test Audio & Predict Disease

Extract Features from Audio Dataset

Train CNN Algorithm

CNN Accuracy & Loss Graph

Upload Test Audio & Predict Disease

Exit

**End process**

**6.3 UML DIAGRAMS:**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

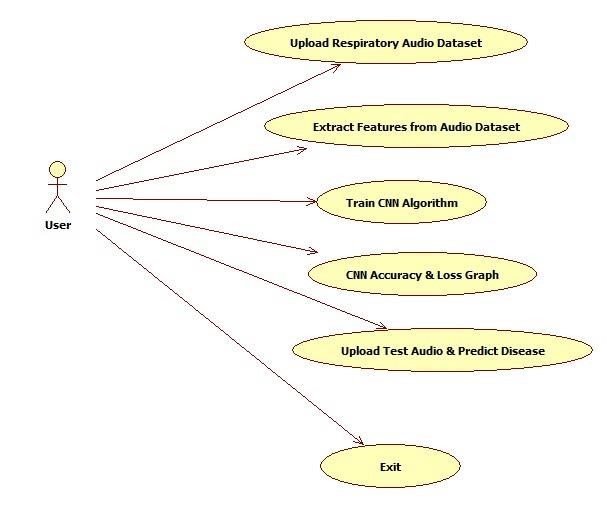
**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Integrate best practices.

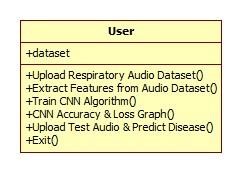
**USE CASE DIAGRAM:**

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.



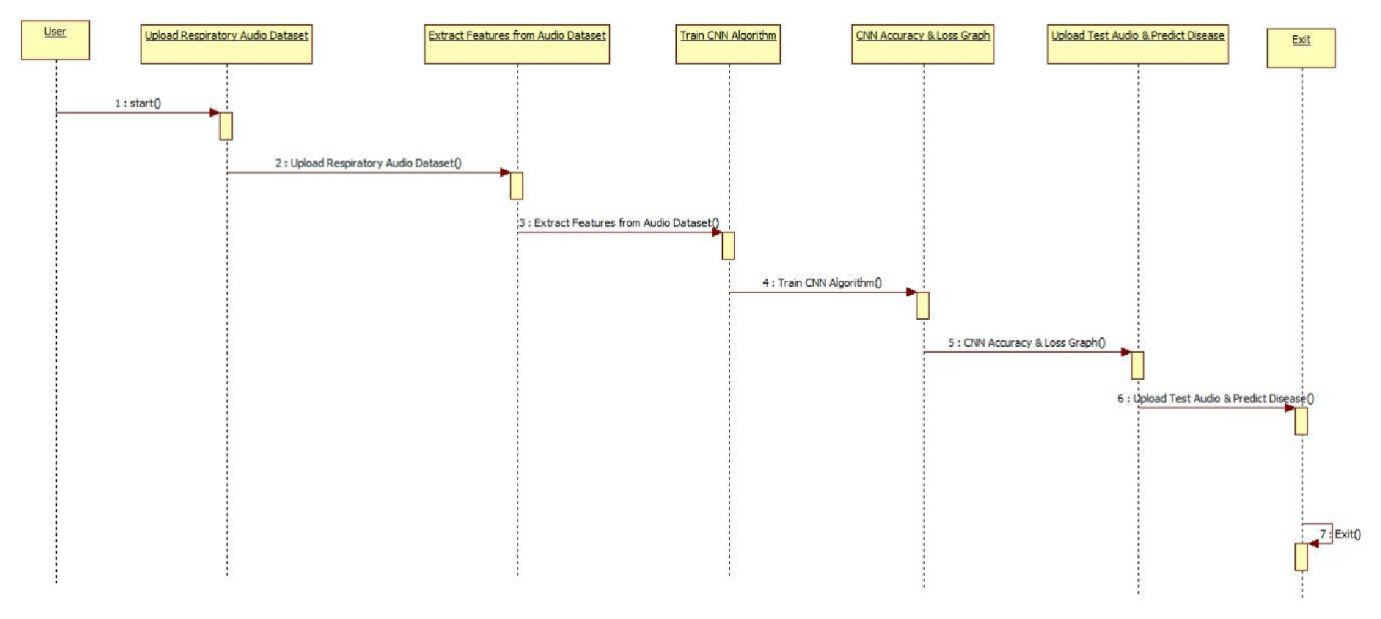
**CLASS DIAGRAM:**

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.



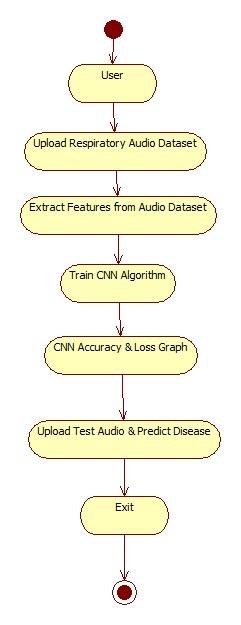
**SEQUENCE DIAGRAM:**

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.



**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.



### 

### **7. IMPLEMENTATION**

**7.1 MODULES:**

* Upload Respiratory Audio Data-set
* Extract Features from Audio Data-set
* Train CNN Algorithm
* CNN Accuracy & Loss Graph
* Upload Test Audio & Predict Disease
* Exit

**MODULES DESCRIPTION:**

1. Upload Respiratory Audio Data-set: using this module we will upload disease diagnosis data-set and respiratory audio data-set
2. Extract Features from Audio Data-set: using this module we will extract features from both datasets and then build training data-set
3. Train CNN Algorithm: using above train data-set we will train CNN model and then build a trained model and this model can be used to predict disease from any new test audio files
4. CNN Accuracy & Loss Graph: using this module we will display comparison graph between accuracy and loss of CNN trained model
5. Upload Test Audio & Predict Disease: using this module we will upload test audio files and then apply CNN trained model on that test audio to predict disease

#### 8. 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 test. Each test type addresses a specific testing requirement.

**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.

**Functional test:**

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.

**System Test:**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**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.

**8.1 Unit Testing:**

Unit testing is usually conducted as part of a combined code and unit test phase of the software life-cycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach:**

Field testing will be performed manually and functional tests will be written in detail.

**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.

* 1. **Integration Testing**

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.

* 1. **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.

# 9. INPUT DESIGN AND OUTPUT DESIGN

**9.1 INPUT DESIGN:**

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy.

Input Design considered the following things:

* What data should be given as input?
* How the data should be arranged or coded?
* The dialog to guide the operating personnel in providing input.
* Methods for preparing input validations and steps to follow when error occur.

**OBJECTIVES:**

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2.It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

**9.2 OUTPUT DESIGN:**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision making.

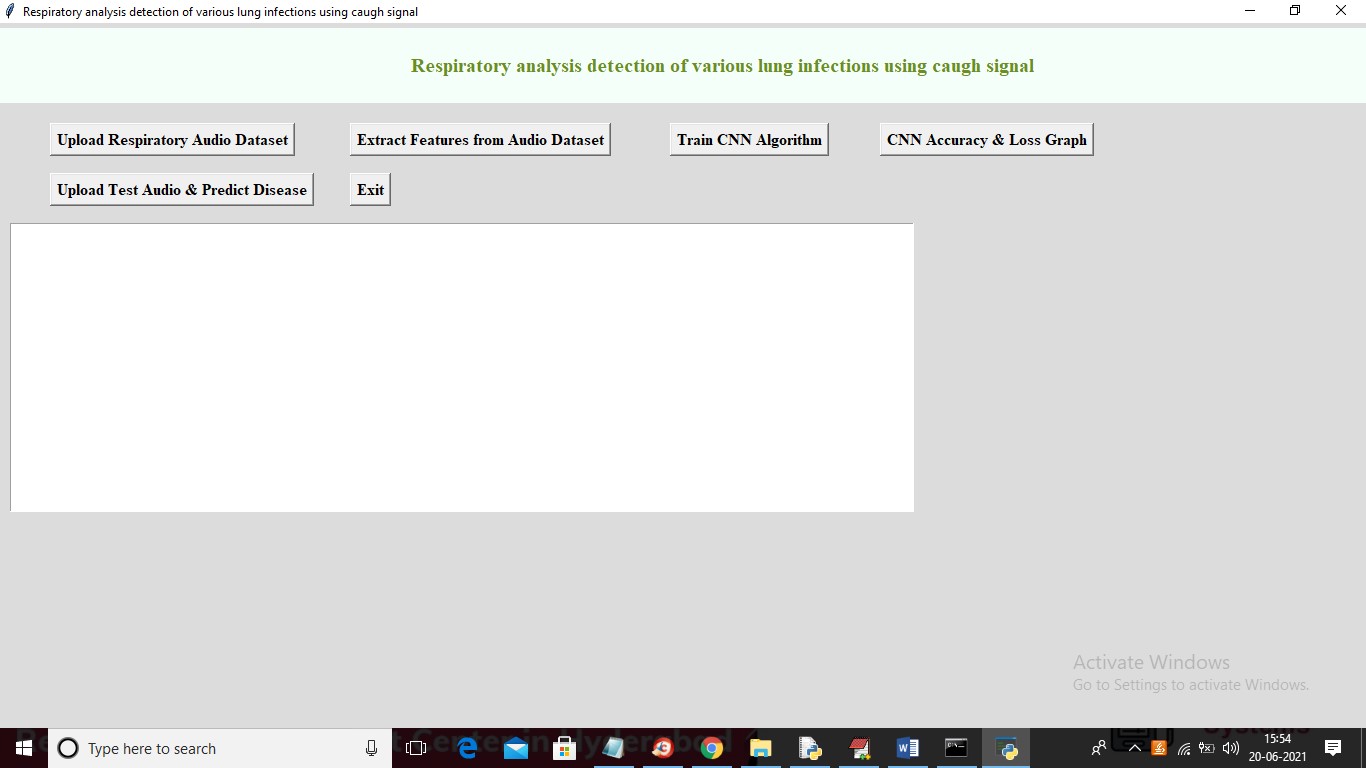
1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
2. Select methods for presenting information.
3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

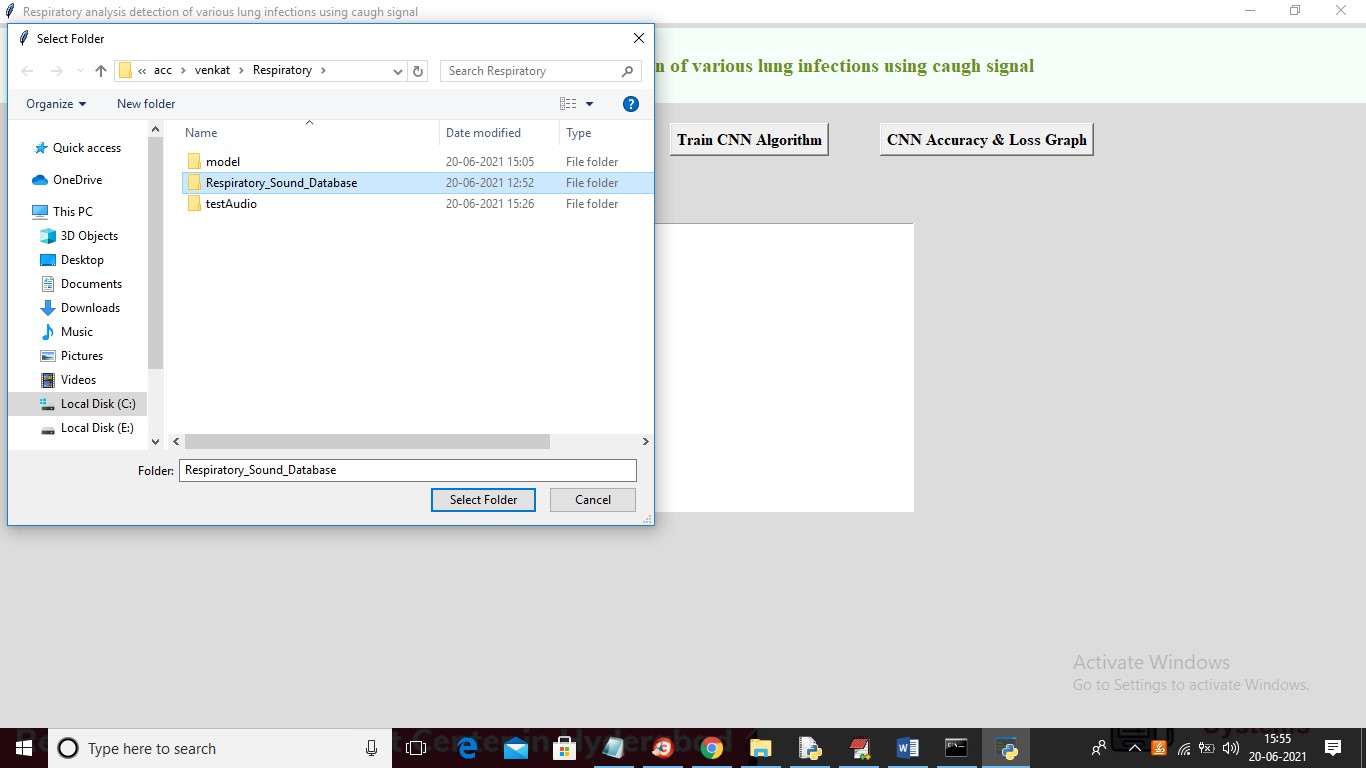
* + Convey information about past activities, current status or projections of the Future.
  + Signal important events, opportunities, problems, or warnings.
  + Trigger an action.
  + Confirm an action.

## 10. SCREENSHOTS

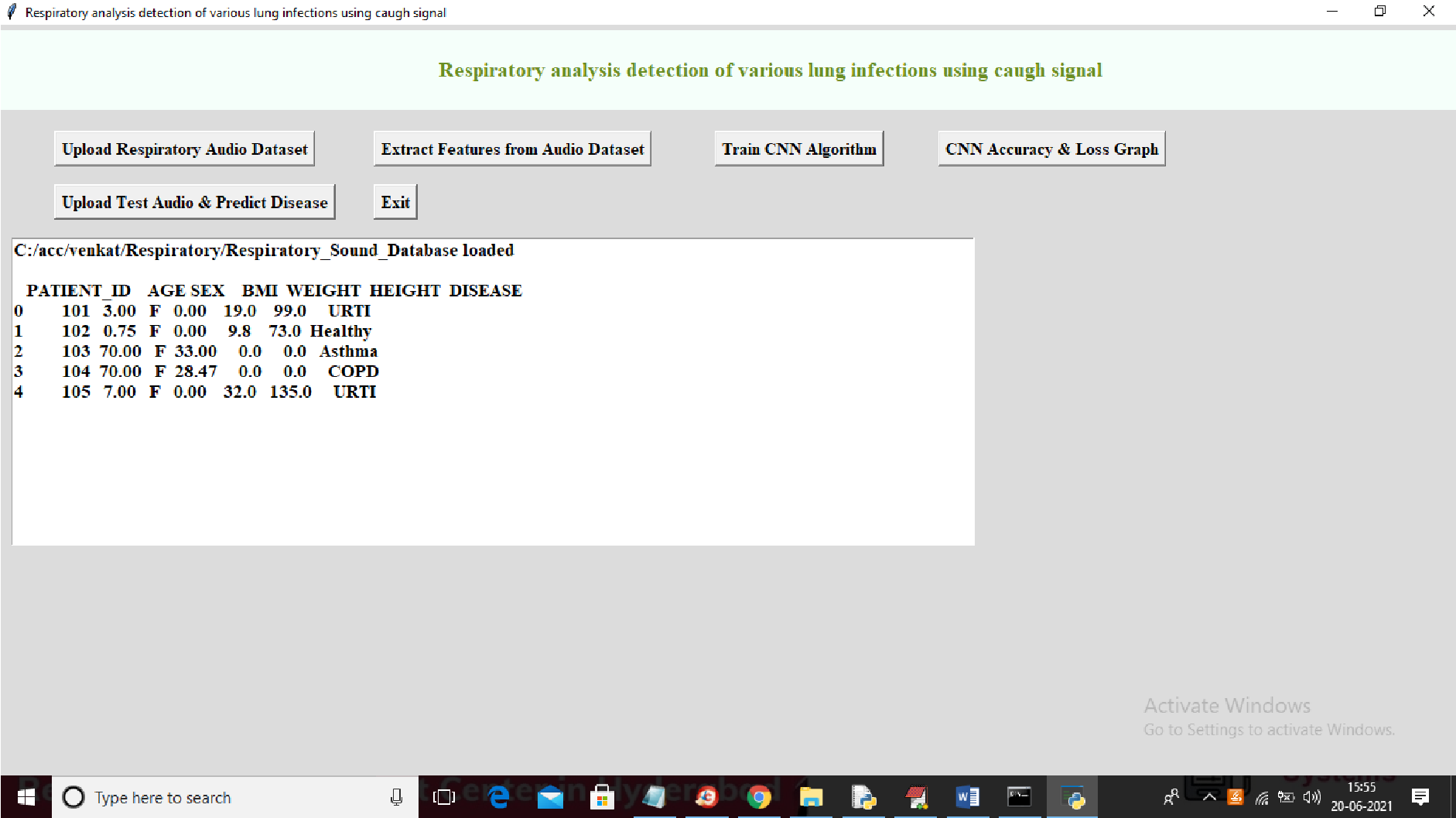
To run project double click on ‘run.bat’ file to get below screen



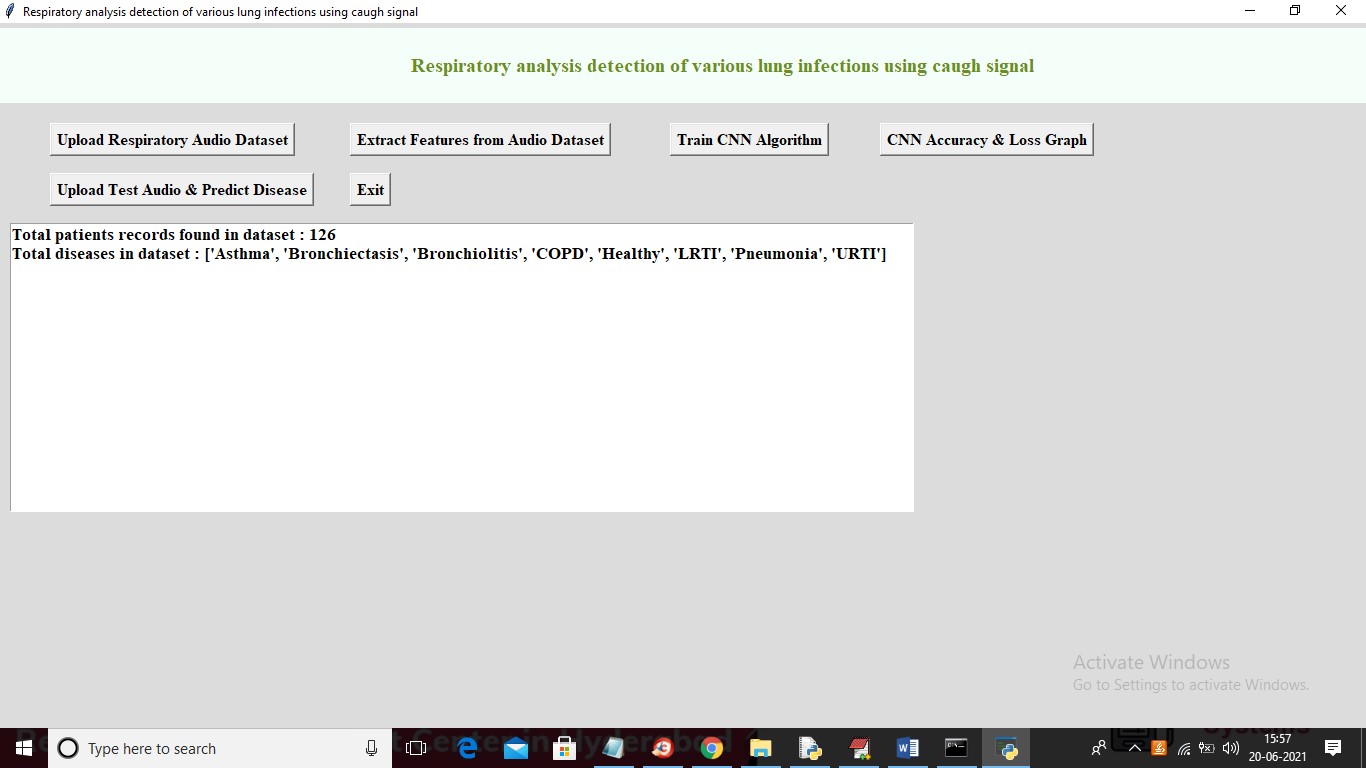
In above screen click on ‘Upload Respiratory Audio Dataset’ button to upload dataset



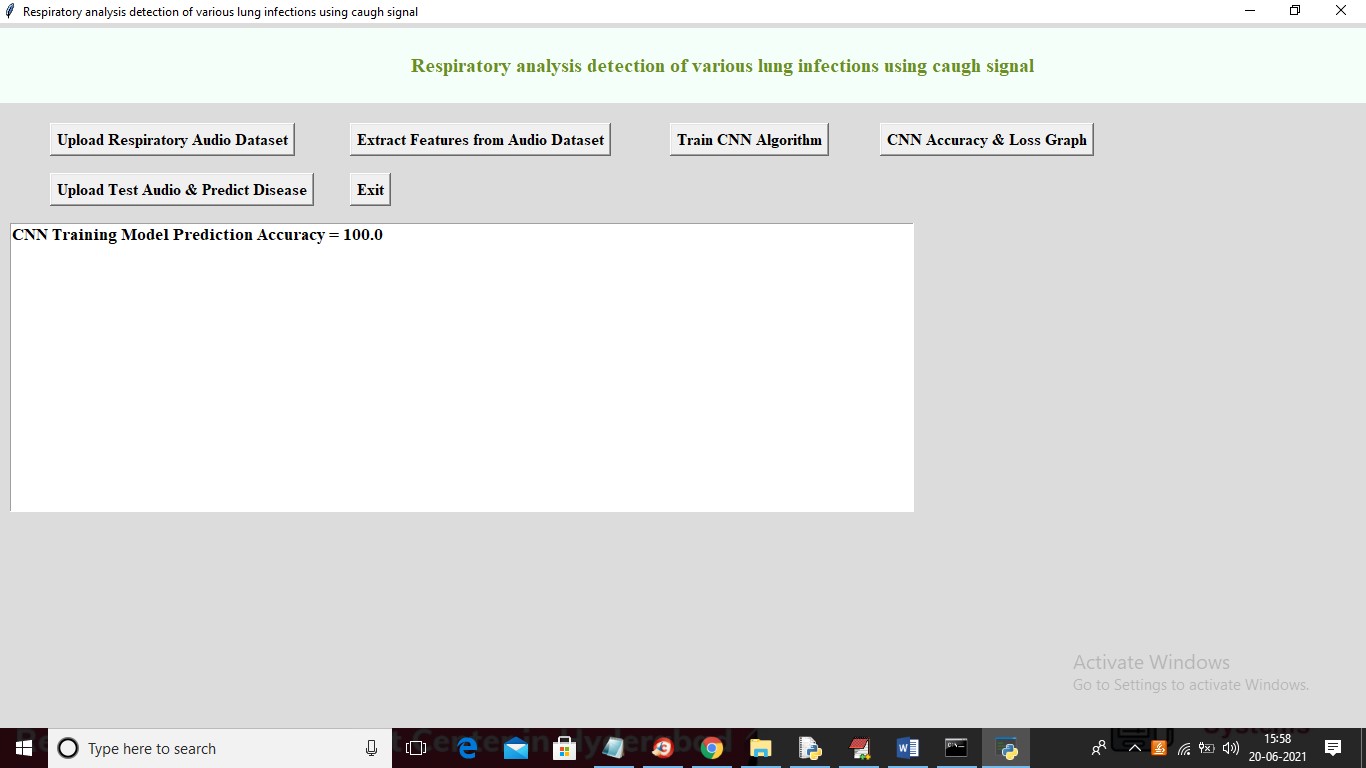
In above screen selecting and uploading entire respiratory sound folder and then click on ‘Select Folder’ button to load dataset and to get below screen



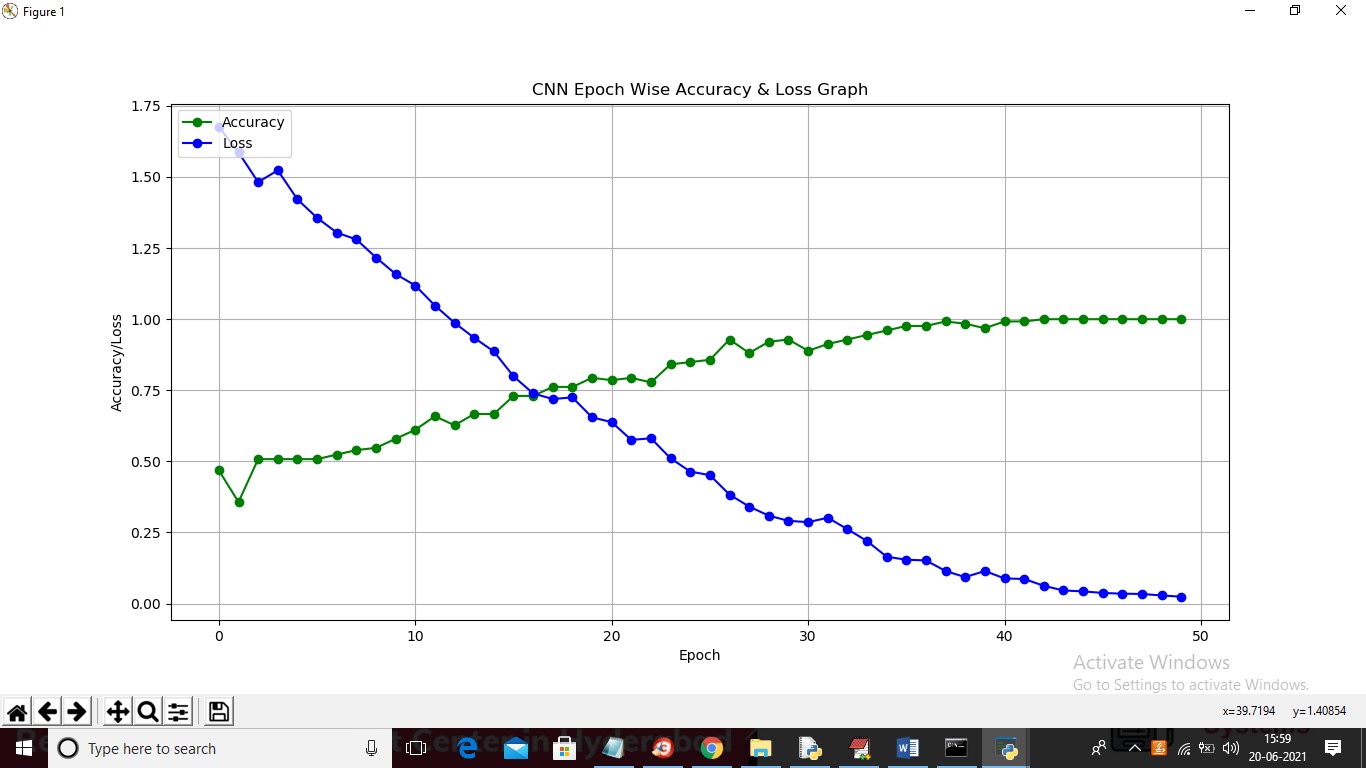
In above screen for each patient, we can see associated with disease diagnose and above disease will be used as class label for each extracted audio features and now click on ‘Extract Features from Audio Dataset' button to extract features from each audio files and then associate detected disease as class label to audio file.



In above screen application found 126 patients audio files and this audio dataset contains 8 different diseases and now dataset is ready and now click on ‘Train CNN Algorithm’ button to train CNN with above data-set and then calculate CNN prediction accuracy.

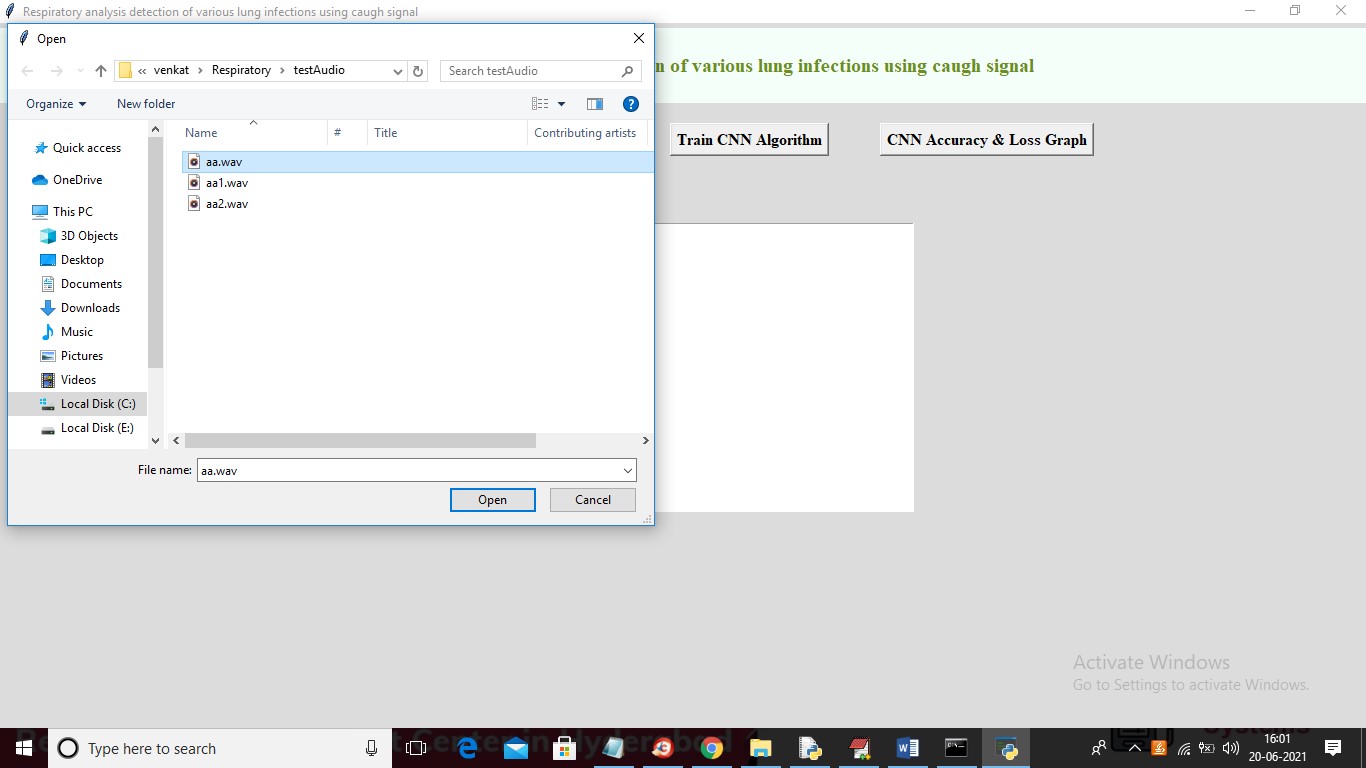


In above screen CNN trained on audio features and got 100% accuracy and now click on ‘CNN Accuracy & Loss Graph’ button to get below graph

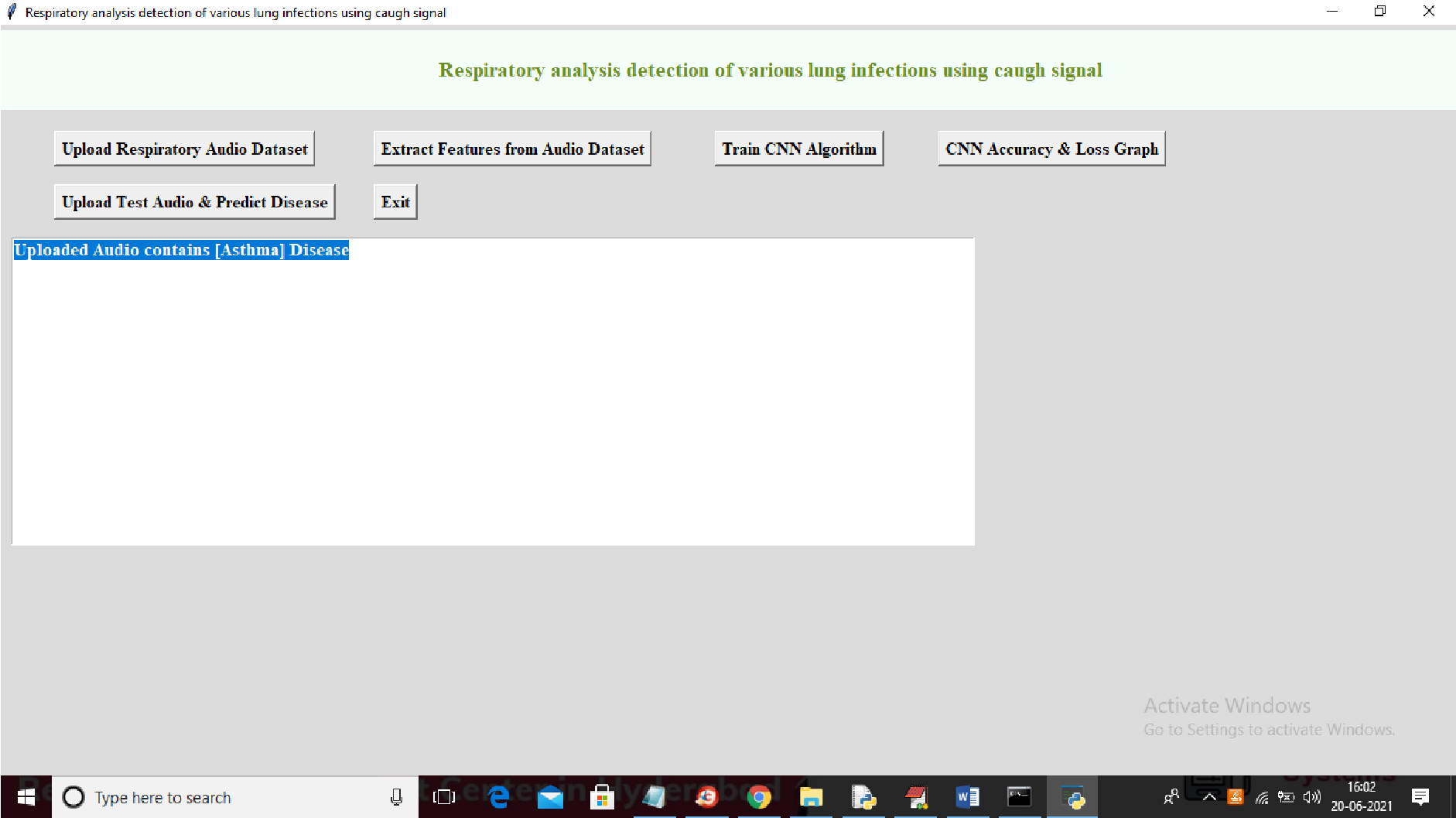


In above graph x-axis represents EPOCH/ITERATIONS and Y-axis represents accuracy and loss values and green line represents accuracy and blue line represents LOSS and we used 50 EPOCHS to train CNN model and we can see with each increasing epoch accuracy get increased and loss value got decrease to 0 and accuracy increased to 100%. Now click on ‘Upload Test Audio & Predict Disease’ button to upload test audio

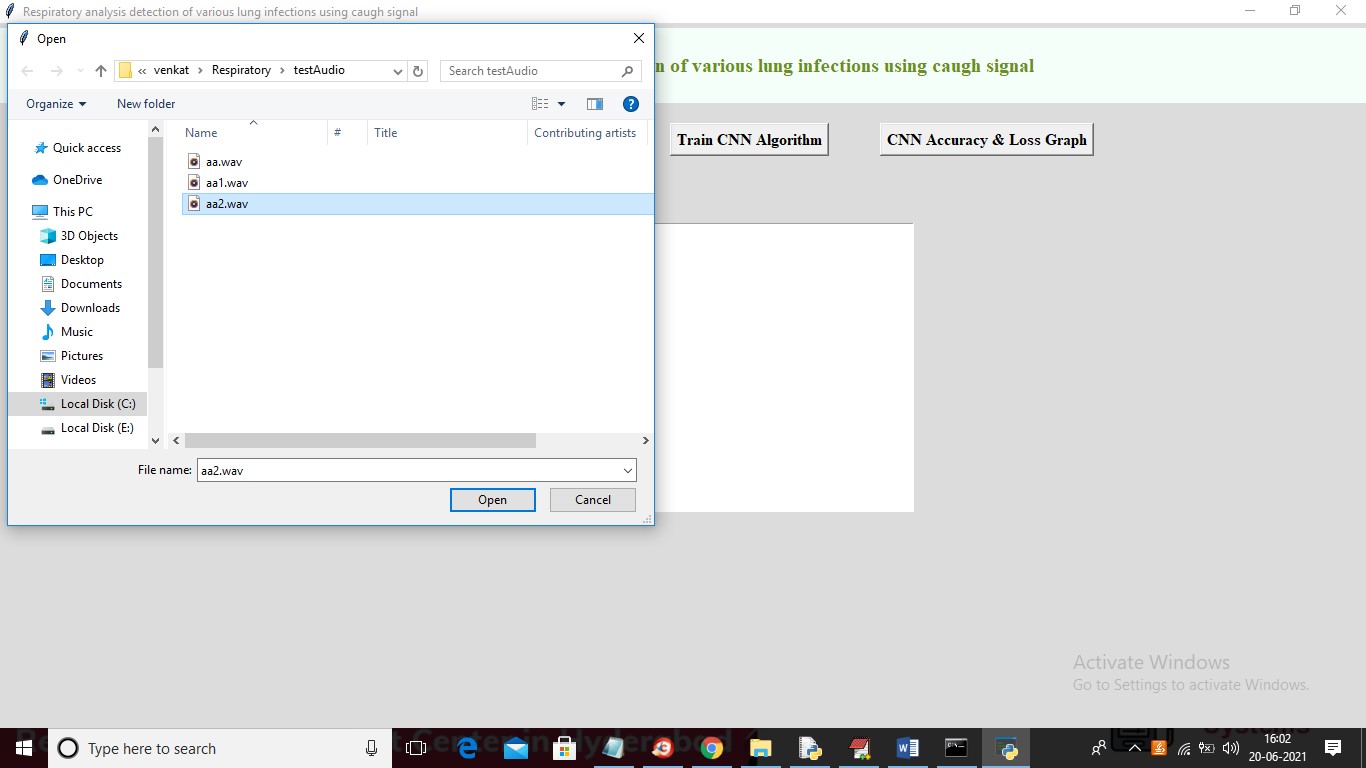
file



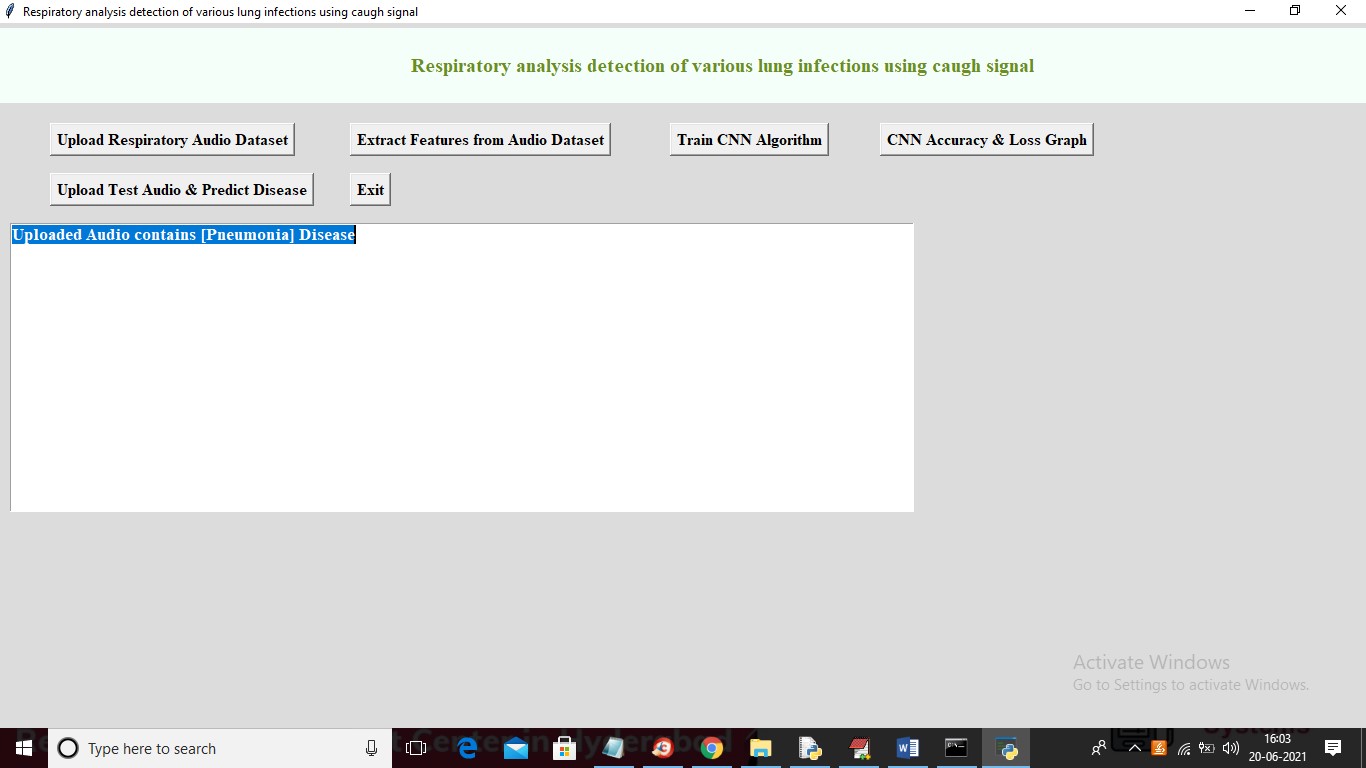
In above screen selecting and uploading ‘aa.wav’ file and then click on ‘Open’ button to get below prediction result



In above screen in blue colour text we can see disease predicted as “ASTHMA” form uploaded audio file and test with other file also



For above selected audio below is the result



In above screen from uploaded audio disease predicted as ‘Pneumonia’ and similarly you can upload other files and predict disease

# 11. CONCLUSION

The lungs are important organs in the respiratory system and used for gas exchange (oxygen and carbon dioxide). When we breathe. Our lungs transfer oxygen from the air into the blood, and carbon dioxide from the blood into the air. To implement this project, we have taken disease diagnosis dataset and respiratory audio dataset and then extract features from all audio dataset and then trained a convolution neural network (CNN) algorithm model. After training model, we can upload any new test data to predict disease from it.

# 

# 12. BIBLIOGRAPHY

1. Pulmonary breath Sounds. East Tennessee State University, November 2002.
2. J. J, Ward. R.A.L.E. Lung Sounds Demo. Med. RRT in Respiratory Care, Canada, 2005.
3. Think labs Digital Stethoscope Lung Library.
4. 3M Littmann Stethoscope Lung Sound Library.
5. Tiago H. Falk, Wai-Yip Chan, Ervin Sejdic´ and Tom Chau, “Spectro-Temporal Analysis of Auscultatory Sounds”, New Developments in Biomedical Engineering, Intech, 2010.
6. Gadge PB and Rode SV, “Automatic Wheeze Detection System as Symptoms of

Asthma Using Spectral Power Analysis”, Journal of Bioengineering & Biomedical Science, 2016.

1. Bor-Shing Lin, Huey-Dong Wu and Sao-Jie Chen, “Automatic Wheezing Detection Based on Signal Processing of Spectrogram and Back - Propagation Neural Network”, Journal of Healthcare Engineering, Vol. 6, No. 4, pp. 649- 672, 2015.
2. L Pekka Malmberg, Leena Pesu, Anssi R A Sovijarvi, “Significant differences in flow standardised breath sound spectra in patients with chronic obstructive pulmonary disease, stable asthma, and healthy lungs”, Thorax, Vol. 50, pp. 12851291, 1995.ss
3. Arati Gurung, Carolyn G Scrafford, James M Tielsch, Orin S Levine and William Checkley, “Computerized Lung Sound Analysis as diagnostic aid for the detection of abnormal lung sounds: a systematic review and meta-analysis”, Respiratory Medicine, Vol. 105, No. 9, pp. 1396–1403, 2011.

[10] Pankaj B. Gadge, Bipin D. Mokal, Uttam R. Bagal, “Respiratory Sound Analysis using MATLAB”, International Journal of Scientific & Engineering Research, Vol. 3, Iss. 5, 2012.

[11] Rutuja Mhetre and U.R. Bagal, “Respiratory sound analysis for diagnostic information”, IOSR Journal of Electrical and Electronics Engineering, Vol. 9, Iss. 5, 2014.

[12] Ria Lestari Moedomo, M. Sukrisno Mardiyanto, Munawar Ahmad, Bachti Alisjahbana, Tjahjono Djatmiko, “The Breath Sound Analysis for Diseases Diagnosis and Stress Measurement”, Proc. of International Conference on System Engineering and Technology, Bandung, Indonesia, 2012.