Six Weeks Industrial Training Project Report on

NEPHRODETECT

Submitted in the partial fulfillment of the requirement for the award of degree of

Bachelor of Technology

in

Computer Science and Artificial Intelligence

(2022-2026)

Submitted to:

Ms. Bindu Bansal

Submitted by:

Yashika Gupta

12200425

B49A

CSE400A

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DAV UNIVERSITY

JALANDHAR-PUNJAB 144012

ACKNOWLEDGEMENT

I express my gratitude to all those who helped us in various stages of the development of this project. First, I would like to express my sincere gratitude indebtedness to Dr. Rahul Hans (Coordinator) of Computer Science department, Ms. Bindu Bansal(Assistant Professor) of DAV University for allowing me to undergo the summer training of 45 days at Excellence Technology. I am also thankful to all faculty members of Department of Computer Science and Engineering, for their true help, inspiration and for the preparation of the final report and presentation.

Last but not least, I pay my sincere thanks and gratitude to all the staff members of Excellence Technology for their support and for making the training valuable and fruitful.

DECLARATION

I, Yashika Gupta, hereby declare that the work which is being presented in this Industrial Training Report entitled ("NEPHRODETECT") by me, in partial fulfilment of the requirements for the award of degree of B.Tech (COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE) is an authentic record of my own work in 6-weeks Industrial Training during the period from June 2025 to July 2025 under the guidance of Mr. Naresh.

To the best of my knowledge, the matter embodied in this report has not been submitted to any other University/Institute for the award of any degree or diploma.

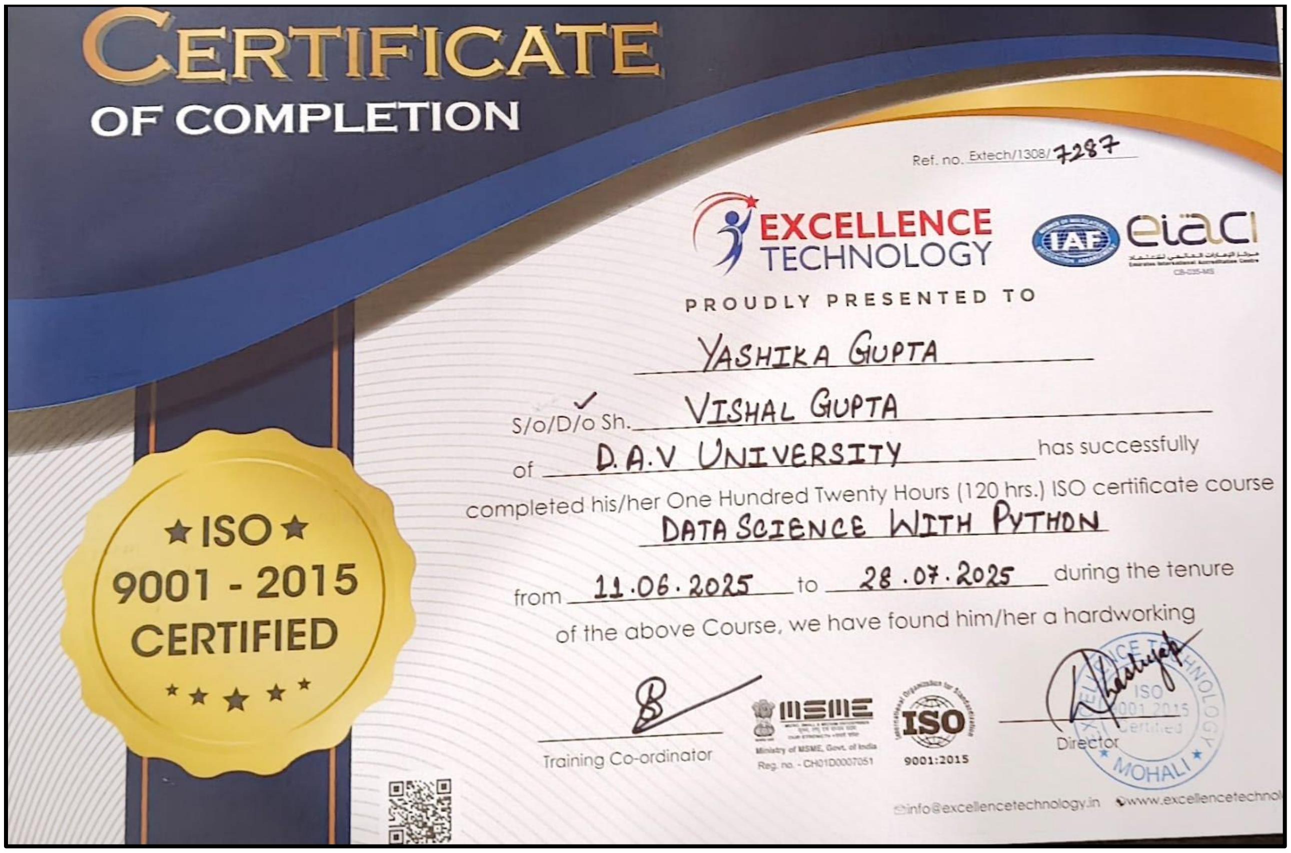
YASHIKA GUPTA

(12200425)

CERTIFICATE

This is to certify that Ms. Yashika Gupta (Student of DAV University, JALANDHAR) has partially completed the 6 weeks Industrial Training during the period from June 2025 to July 2025 in our organization/industry as a partial fulfillment of Degree of Bachelor of Technology in Computer Science and Artificial Intelligence.

DAV UNIVERSITY, JALANDHAR



CERTIFICATE
OF COMPLETION

Ref. no. Extech/1308/7287

EXCELLENCE TECHNOLOGY

IAF eiaci

PROUDLY PRESENTED TO

YASHIKA GUPTA

S/o/D/o Sh. VISHAL GUPTA

of D.A.V UNIVERSITY has successfully completed his/her One Hundred Twenty Hours (120 hrs.) ISO certificate course DATA SCIENCE WITH PYTHON

from 11.06.2025 to 28.07.2025 during the tenure of the above Course, we have found him/her a hardworking

★ ISO ★
9001 - 2015
CERTIFIED

Training Co-ordinator

MSME
Ministry of MSME, Govt. of India
Reg. no. - CH01D0007051

ISO 9001:2015

Director

EXCELLENCE TECHNOLOGY ISO 9001:2015 Certified MOHALI

info@excellencetechnology.in www.excellencetechno

ABSTRACT

Chronic kidney disease (CKD) and other renal disorders constitute a significant global health burden, affecting millions of individuals worldwide and often leading to severe complications such as kidney failure, cardiovascular diseases, and increased mortality. Early detection and precise classification of kidney abnormalities are critical to improving patient survival and optimizing treatment strategies. Conventional diagnostic procedures, including serum creatinine analysis, biopsy, and radiological interpretation, though widely used, are often limited by subjectivity, inter-observer variability, and the need for specialized expertise.

In recent years, deep learning has emerged as a transformative approach in medical image analysis, owing to its capacity for hierarchical feature extraction and automated pattern recognition from high-dimensional data. This study explores the application of deep learning architectures, particularly convolutional neural networks (CNNs), for the automated classification of kidney diseases from medical imaging modalities such as ultrasound, computed tomography (CT), and magnetic resonance imaging (MRI). The proposed framework incorporates multiple preprocessing steps, including image normalization, noise reduction, and augmentation techniques, to enhance robustness and mitigate data imbalance issues. Advanced CNN models, integrated with transfer learning and fine-tuning strategies, are employed to extract discriminative spatial features and classify renal conditions into categories such as normal, cystic kidney disease, kidney stones, and renal tumors.

Extensive experiments are conducted on publicly available and clinically curated datasets, with evaluation metrics including accuracy, precision, recall, F1-score, and area under the ROC curve (AUC). Comparative analyses against conventional machine learning classifiers and baseline CNNs demonstrate that the proposed deep learning model achieves superior diagnostic performance, significantly reducing false positives and false negatives. Furthermore, visualization techniques such as Grad-CAM are employed to enhance interpretability, offering insights into model decision-making and improving clinical trustworthiness.

COMPANY PROFILE

With the EXCELLENCE TECHNOLOGY experience the incredible services such as agile software development and the problems related to outsourcing. We comprise of the team of experienced and professionals members who with their skills efficiently get the job done and innovatively help you to transform your ideas into the successful business.

- At **EXCELLENCE TECHNOLOGY,** we have competence to expand and adjust as per client specific requirements.
- **Skilled Workforce:** At EXCELLENCE TECHNOLOGY you deal with the highly professional and proficient employees.
- **Cost Efficiency:** We help you to reduce the unnecessary investment and ask for the reasonable amount of money.
- **Quality Of the Product:** Our software service sector has been maintaining the highest international standards of quality.
- **Infrastructure:** Well organized team and tools to handle the projects with responsible approach Hardware, Software, Networking, Voice, Conferencing, disaster recovery all infra all you need for international projects.
- **Ongoing Involvement:** EXCELLENCE TECHNOLOGY products are “built for change” as we are well responsive that the necessity to improve a Web solution generally arises even before the solution is out of the door. We delivers long-term product enhancement if desired.
- **Partnership:** EXCELLENCE TECHNOLOGY considers every client a partner. From the initial stages, you are closely involved into the procedure of technical classification, development, and testing.

CLIENTS

Zimmerman Metals Inc.

AEROSEAL of Colorado

arvada visitors center

Asia-Asia Traditions® 亜洲

ATLAS LUMBER COMPANY Bringing You the World's Finest Woods

AVocation Systems PROFESSIONAL GRADE QUALITY AUDIO/VIDEO DISTRIBUTION PRODUCTS

BB Heating and Air Conditioning, Inc.

BRICKSTONE Inc. a division 4 company

C&H Irrigation & Landscaping

Care Association

CENTERPOINT INSURANCE GROUP

CHARGE Write! INC.

CMCA COLORADO MOTOR CARRIERS ASSOCIATION

COLAVRIA HOSPITALITY

Rheem

COMFORT AIR

COMMUNICATION Unlimited

CRIBARI LAW FIRM, P.C. Colorado Criminal Defense Experts

crmca

Denver Executive ASSOCIATION

DDM PROPERTY SERVICES

Déjà Décor

Denverpete's Playing Cards for Collectors

Eastside HEATING & AIR CONDITIONING, INC.

EAST-WEST

Elderlink HOME CARE, INC.

event apptitude

FRUGAL FORMALWEAR

GeoWater SERVICES, LLC

Gibson GA Advertising

GotPetPrints.com

GREATGRABZ®

GCM GREELEY COMMUNITY MANAGEMENT, LLC

Gustafson Heating & Air Conditioning Inc. Since 1971

Happy Valley Children's Ranch Preschool

Established in 1961

IAQ Indoor Air Quality Inc.

DUNCAN & DUNCAN ENTERPRISES, LLC

LO DO DENVER Old. New. Now.

Lower Downtown NEIGHBORHOOD ASSOCIATION

MACK'S DRILLING, INC working on water since 1970

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

Kidney disease is a serious global health challenge, affecting nearly 10% of the world's population and contributing significantly to mortality rates. The kidneys play a crucial role in maintaining homeostasis through the regulation of blood pressure, electrolyte balance, and waste excretion. Disorders of the kidney, including chronic kidney disease (CKD), kidney stones, cysts, and tumors, often progress silently and remain undetected until they reach advanced stages. Late diagnosis not only complicates treatment but also increases the risk of kidney failure, requiring dialysis or transplantation. Therefore, timely detection and accurate classification of kidney diseases are of paramount importance in clinical practice.

Traditional diagnostic approaches—such as laboratory tests, biopsy, ultrasound, computed tomography (CT), and magnetic resonance imaging (MRI)—are effective but come with limitations. These include dependency on expert interpretation, high cost, limited availability in resource-constrained regions, and potential variability among radiologists. Such challenges highlight the urgent need for automated, reliable, and efficient diagnostic systems that can assist clinicians in early-stage detection and reduce diagnostic errors.

Recent advances in artificial intelligence (AI), particularly deep learning, have demonstrated remarkable success in medical image analysis and disease prediction. Deep learning models, especially convolutional neural networks (CNNs), can automatically learn hierarchical representations from medical images, eliminating the need for handcrafted feature extraction. By leveraging large annotated datasets and transfer learning from pretrained models such as VGG16, ResNet, and Inception, AI-driven systems can achieve performance comparable to or even exceeding that of human experts in specific tasks.

This study focuses on kidney disease classification using deep learning techniques, aiming to automate the identification of different kidney conditions from medical images. The proposed approach not only enhances diagnostic accuracy but also reduces the time required for analysis, making it a valuable tool for integration into clinical workflows.

1.1 PROBLEM DEFINITION

Chronic Kidney Disease (CKD) is a progressive condition that often goes undetected in its early stages due to subtle or non-specific symptoms. Traditional diagnostic methods rely on manual interpretation of multiple clinical parameters, which can be time-consuming and prone to human error. As a result, many patients are diagnosed only after significant kidney damage has already occurred.

There is a need for an automated, reliable, and efficient system that can analyze clinical data and accurately predict the presence of kidney disease. The problem is to develop a machine learning-based classification model that can learn patterns from patient medical records and assist in early detection of CKD, thereby supporting timely treatment and reducing health risks.

1.2 OBJECTIVES

Primary Objective: To develop an accurate machine learning model that classifies patients into kidney disease and non-kidney disease categories using clinical and laboratory data.

1. To analyze kidney-related clinical data and identify key features influencing CKD.
2. To preprocess the dataset by handling missing values, encoding categorical variables, and normalizing attributes.
3. To develop and train machine learning models for accurate CKD classification.
4. To evaluate model performance using metrics such as accuracy, precision, recall, F1-score, and ROC-AUC.
5. To identify the most important predictors contributing to CKD.
6. To build an automated prediction system (optional) for assisting early diagnosis.

CHAPTER 2: DATA PREPROCESSING

Preprocessing for Medical Imaging Data

If using kidney ultrasound, CT, or MRI images.

a) Data Acquisition

- Collect unstructured kidney images (ultrasound, CT scans, MRI, histopathology slides).
- Ensure a balanced dataset (healthy vs. diseased).
- Convert medical image formats (like DICOM, TIFF, PNG, JPEG) into a standard format for model input.

b) Data Cleaning

- **Noise removal**: Apply filters (Gaussian, median) to remove speckle noise common in medical images.
- **Artifact removal**: Remove labels, text, or scanner marks using masking or cropping.
- **Intensity normalization**: Normalize pixel values to range [0,1] or standardize using Z-score for uniform brightness/contrast.

c) Image Resizing and Cropping

- Deep learning models require fixed-size inputs:
    - Resize all images to a standard resolution (e.g., 224×224 for VGG16, 299×299 for Inception).
- **Region of Interest (ROI) extraction**:
    - Crop kidneys from the background using segmentation techniques (U-Net, thresholding, edge detection).
    - This reduces irrelevant features and improves classification accuracy.

d) Image Enhancement

- **Histogram equalization**: Improves contrast in low-light scans.
- **CLAHE (Contrast Limited Adaptive Histogram Equalization)**: Especially effective in medical imaging for enhancing visibility of small structures.
- **Sharpening filters**: Highlight boundaries of cysts, stones, or tumors.

e) Normalization and Standardization

- Scale pixel values to [0,1] or [-1,1] depending on the model.
- Standardize per-channel mean and variance (important for pretrained networks like VGG16, ResNet).

f) Data Augmentation

To handle limited medical datasets:

- **Geometric transformations**: rotation, flipping, zooming, shifting.
- **Elastic deformations**: simulate realistic variations in tissues.
- **Noise injection**: improves model robustness.
- **Color augmentation** (for histopathology images): random brightness, contrast, hue changes.

Ensure augmentations are medically valid (avoid transformations that distort pathological features).

g) Segmentation-based Preprocessing (Optional but powerful)

- Use **U-Net, Mask R-CNN, or GrabCut** to segment kidney regions before classification.
- Helps remove background and focus only on kidney tissues.

h) Dimensionality Reduction (Optional)

- For high-resolution CT/MRI scans, downsample or apply **PCA on image patches** to reduce complexity.

i) Data Splitting

- Split into **train (70%)**, **validation (15%)**, and **test (15%)** ensuring stratified distribution across disease classes.

CHAPTER 3: MODEL TRAINING

1. Model Selection

Choose an appropriate deep learning architecture:

- **CNN from scratch** (if dataset is large).
- **Transfer learning** (if dataset is small/medium).
  - Pretrained models: **VGG16, ResNet50, InceptionV3, DenseNet121, EfficientNet**.
- **Hybrid models** (CNN + RNN or CNN + Attention) for advanced classification.

2. Input Preparation

- Input shape depends on model (e.g., 224×224×3 for VGG16/ResNet).
- Preprocessed images (resized, normalized, augmented) are fed in batches.

3. Model Architecture

Typical kidney disease classifier (Transfer Learning with VGG16/ResNet):

- **Base model** (pretrained CNN on ImageNet).
- **Freeze initial layers** (to keep low-level features).
- **Add custom classifier head**:
  - Global Average Pooling / Flatten
  - Dense layer(s) with ReLU
  - Dropout (0.3–0.5 to prevent overfitting)
  - Final Dense layer with **Softmax / Sigmoid** (depending on number of classes).

4. Training Process

1. **Loss function**:
   - Binary classification → BinaryCrossentropy
   - Multi-class classification → CategoricalCrossentropy

2. **Optimizer**:
    - Adam (common choice), SGD (for stability in large datasets).
3. **Learning rate scheduling**:
    - ReduceLROnPlateau or OneCycleLR to adjust learning rate dynamically.
4. **Batch size**: 16–64 (depends on GPU memory).
5. **Epochs**: 20–100 (use Early Stopping to avoid overfitting).

5. Training Enhancements

- **Class imbalance handling**:
    - Weighted loss, SMOTE, or oversampling minority class.
- **Regularization**:
    - Dropout, L2 regularization, batch normalization.
- **Data augmentation** (during training, not test time).

6. Evaluation Metrics

Beyond accuracy, use medically relevant metrics:

- **Precision & Recall** (important to avoid false negatives).
- **F1-score** (balance between precision & recall).
- **ROC-AUC** (discrimination ability).
- **Confusion Matrix** (class-wise performance).

7. Model Validation & Testing

- Use **k-fold cross-validation** if dataset is small.
- Evaluate on **independent test set** (never seen by model).
- Perform **external validation** if you have datasets from different hospitals.

8. Deployment Preparation

- Convert model to **ONNX / TensorFlow Lite** for deployment in healthcare systems.
- Ensure **explainability** using Grad-CAM / LIME to highlight kidney regions influencing classification.

CHAPTER 4: REQUIREMENT ANALYSIS

A feasibility study—sometimes called a feasibility analysis or feasibility report—is a way to evaluate whether or not a project plan could be successful. A feasibility study evaluates the practicality of your project plan in order to judge whether or not you're able to move forward with the project.

Benefits of conducting a feasibility study

There are several key benefits to conducting a feasibility study before launching a new project:

- Confirms market opportunities and the target market before investing significant resources
- Identifies potential issues and risks early on
- Provides in-depth data for better decision making on the proposed project's viability
- Creates documentation on expected costs and benefits, including financial analysis
- Obtains stakeholder buy-in by demonstrating due diligence

Feasibility studies are important for projects that represent significant investments for your business. Projects that also have a large potential impact on your presence in the market may also require a feasibility assessment.

As the project manager, you may not be directly responsible for driving the feasibility study,but it's important to know what these studies are. By understanding the different elements that go into a feasibility study, you can better support the team driving the feasibility study and ensure the best outcome for your project.

One thing to keep in mind is that a feasibility study is not a project pitch. During a project pitch, you're evaluating whether or not the project is a good idea for your company and whether the goals of the project are in line with your overall strategic plan.

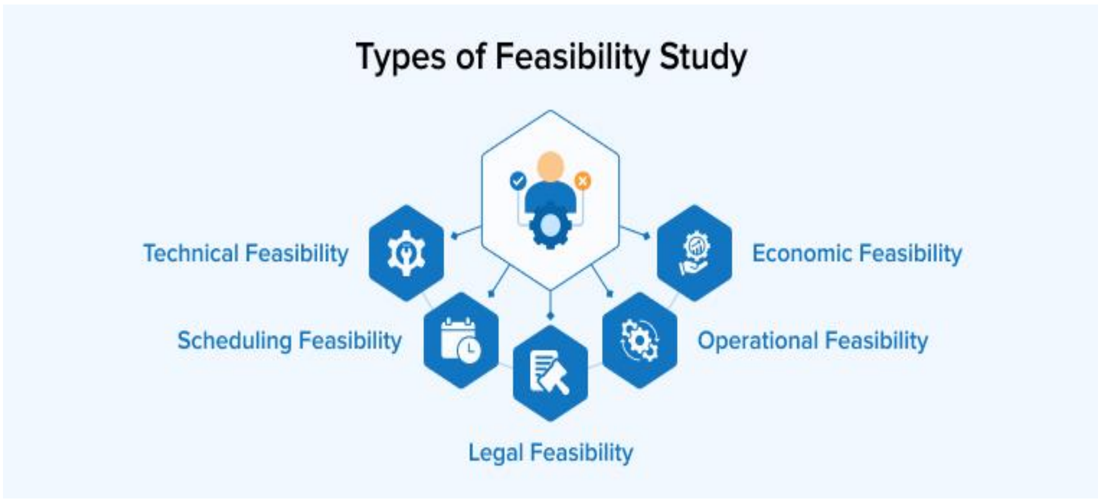


Fig 4.1

4.1 TYPES OF FEASIBILITY STUDY

4.1.1 Technical feasibility

A technical feasibility study reviews the technical resources available for your project. This study determines if you have the right equipment, enough equipment, and the right technical knowledge to complete your project objectives. For example, if your project plan proposes creating 50,000 products per month, but you can only produce 30,000 products per month in your factories, this project isn't technically feasible.

4.1.2 Financial feasibility

Financial feasibility describes whether or not your project is fiscally viable. A financial feasibility report includes a cost-benefit analysis of the project. It also forecasts an expected return on investment (ROI) and outlines any financial risks. The goal at the end of the financial feasibility study is to understand the economic benefits the project will drive.

4.1.3 Scheduling feasibity

Schedule feasibility assesses whether a project can be completed within its proposed timeframe. It involves analyzing timelines, resource availability, dependencies, and potential risks to determine if the schedule is realistic and achievable. Essentially, it's about determining if the project can be done on time, given the resources and constraints.

4.1.4 Operational feasibility

An operational feasibility study evaluates whether or not your organization is able to complete this project. This includes staffing requirements, organizational structure, and any applicable legal requirements.

4.1.5 Legal feasibility

A legal feasibility analysis assesses whether the proposed project complies with all relevant legal requirements and regulations. This includes examining legal and regulatory barriers, necessary permits, licenses, or certifications, potential legal liabilities or risks, and intellectual property considerations.

4.2 SDLC Model

SDLC is a process followed for software building within a software organization. SDLC consists of a precise plan that describes how to develop, maintain, replace, and enhance specific software. The life cycle defines a method for improving the quality of software and the all-around development process.

SDLC specifies the task(s) to be performed at various stages by a software engineer or developer. It ensures that the end product is able to meet the customer’s expectations and fits within the overall budget.

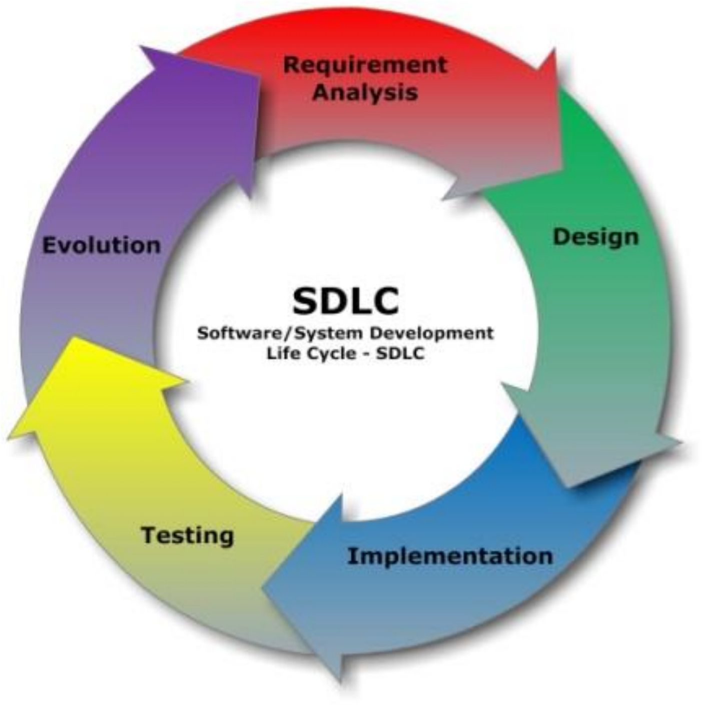


Fig 4.2

Stage-1: Planning and Requirement Analysis

Planning is a crucial step in everything, just as in software development.In this same stage, requirement analysis is also performed by the developers of the organization. This is attained from customer inputs, and sales department/market surveys.The information from this analysis forms the building blocks of a basic project.

Stage-2: Defining Requirements

In this stage, all the requirements for the target software are specified. These requirements get approval from customers, market analysts and stakeholders. This is fulfilled by utilizing SRS (Software Requirement Specification). This is a sort of document that specifies all those things that need to be defined and created during the entire project cycle.

Stage-3: Designing Architecture

SRS is a reference for software designers to come up with the best architecture for the software.Hence, with the requirements defined in SRS, multiple designs for the product architecture are present in the Design Document Specification (DDS). After evaluating all the possible factors, the most practical and logical design is chosen for development.

Stage-4: Developing Product

At this stage, the fundamental development of the product starts. For this, developers use a specific programming code as per the design in the DDS .Conventional programming tools like compilers, interpreters, debuggers, etc. are also put into use at this stage. Some popular languages like C/C++, Python,Java, etc. are put into use as per the software regulations.

Stage-5: Product Testing and Integration

After the development of the product, testing of the software is necessary to ensure its smooth execution. Although, minimal testing is conducted at every stage of SDLC A well-written document acts as a tool and means to information repository necessary to know about software processes, functions, and maintenance.

Stage 6: Deployment and Maintenance of Products

After detailed testing, the conclusive product is released in phases as per the organization's strategy. Then it is tested in a real industrial environment. It is important to ensure its smooth performance. If it performs well, the organization sends out the product as a whole. After retrieving beneficial feedback, the company releases it as it is or with auxiliary improvements to make it further helpful for the customers.

CHAPTER 5: SYSTEM REQUIREMENTS SPECIFICATION

The System Requirements Specification (SRS) is a document focused on what the software needs to do and how it must perform. It lays the important groundwork so that every person involved with the project understands the most crucial details.

An SRS outlines the behaviors,functions and capabilities required of the system, along with any potential constraints. Functional and non-fu requirements are included. Design suggestions and information outside the customer's requirements are not included.

Approval is received from all necessary stakeholders, showing that they clearly understand the project requirements and everyone agrees. In a sense, the SRS functions as an insurance policy that any party can refer to in case of uncertainty.

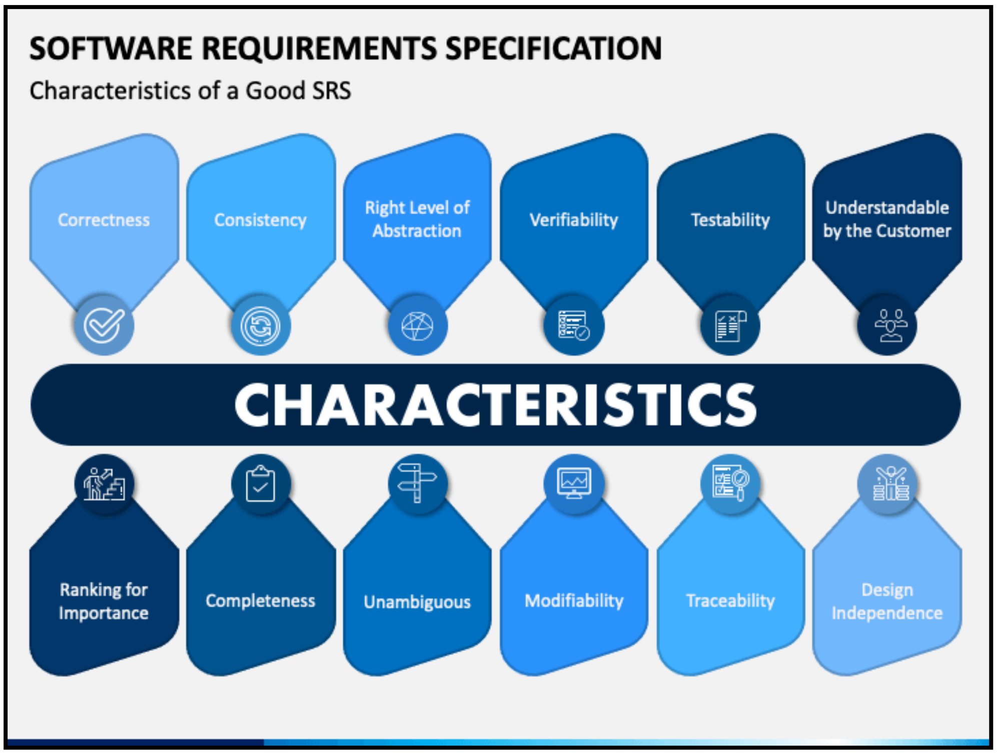


Fig 5.2

5.1 Need of System Requirements Specification

Using an SRS ensures that specifics around a project are crystal clear, reducing the risk of rework and wasted time. Important benefits of using this type of document include:

Providing Valuable Customer Feedback. An SRS is the customer's confirmation that your organization understands the problems that need to be solved and how the software must behave to address the challenges. Visuals such as charts and tables can provide additional clarity.

Breaking Work Into Smaller Pieces. An SRS contains a large amount of information, but they ultimately break problems into smaller, more manageable parts.

Serving as a Parent Document. The SRS serves as a parent document to any additional documents that follow its creation. The scope of work, software design specifications, and other documents often leverage what is highlighted in the SRS. Plus, it can serve as a product validation check.

Enhanced Collaboration. Software development requires collaboration. The SRS has team spirit Working together with cross-functional teams, it forms a single vision and shared understanding of the project. As a result, the development process becomes more integrated and efficient.

5.2 RESOURCES NEEDED

Resources are the requirement of the system. It can also be defined as the environment within which a system can work efficiently. In order to implement the proposed system, the following various hardware and software requirements to achieve good performance:

1. Hardware Requirements

- Laptop/PC with minimum:
  - Processor: Intel i5 / AMD Ryzen 5 or higher
  - RAM: 8 GB (16 GB recommended for faster model training)
  - Storage: 20–30 GB free space
  - GPU: Optional (helpful for deep learning models)

2. Software Requirements

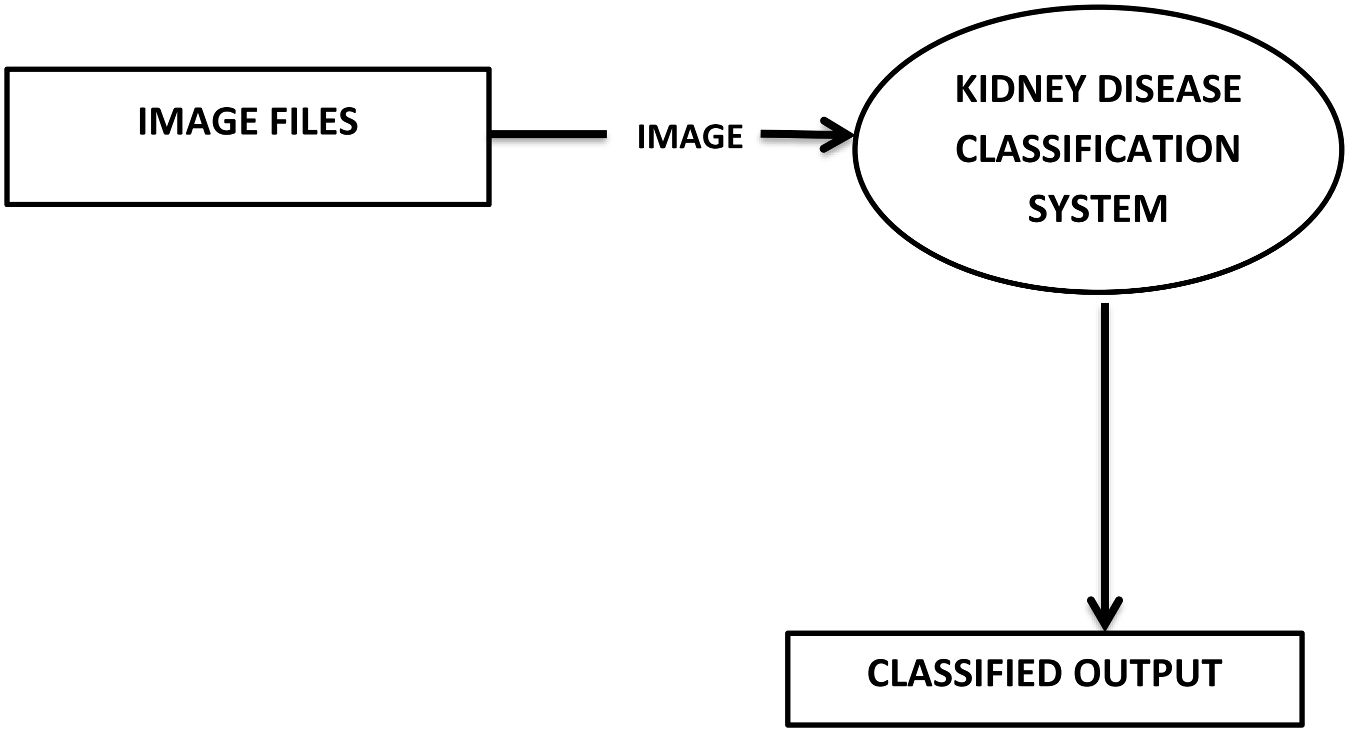
- Programming Language: Python 3.x
- IDE / Editors:
  - Jupyter Notebook
  - VS Code / PyCharm
- Machine Learning Libraries:
  - NumPy
  - Pandas
  - Scikit-learn
  - Matplotlib / Seaborn
  - XGBoost / LightGBM (optional)
- Deployment (optional):
  - Streamlit or Flask
  - Anaconda (for environment management)

3.3 TOOLS USED

- PYTHON
- MATPLOTLIB
- MLFLOW
- NUMPY
- TENSORFLOW
- PANDAS
- STREAMLIT

CHAPTER 6: DATA FLOW DIAGRAMS

LEVEL 0- DFD



LEVEL 1 DFD

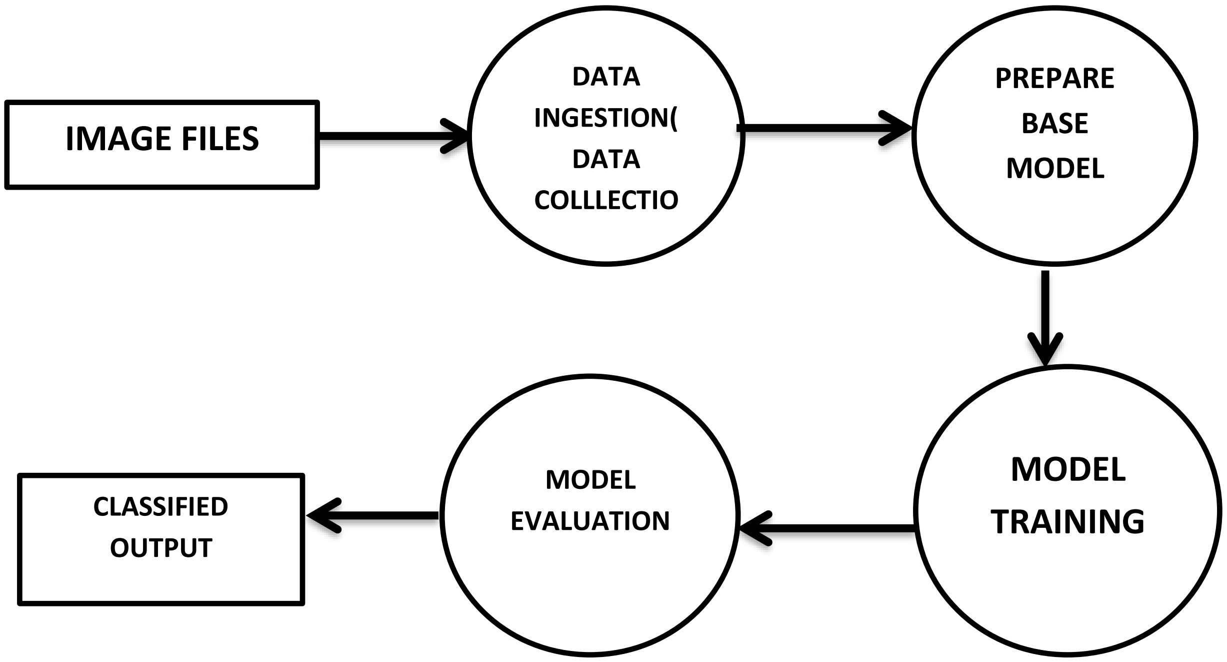
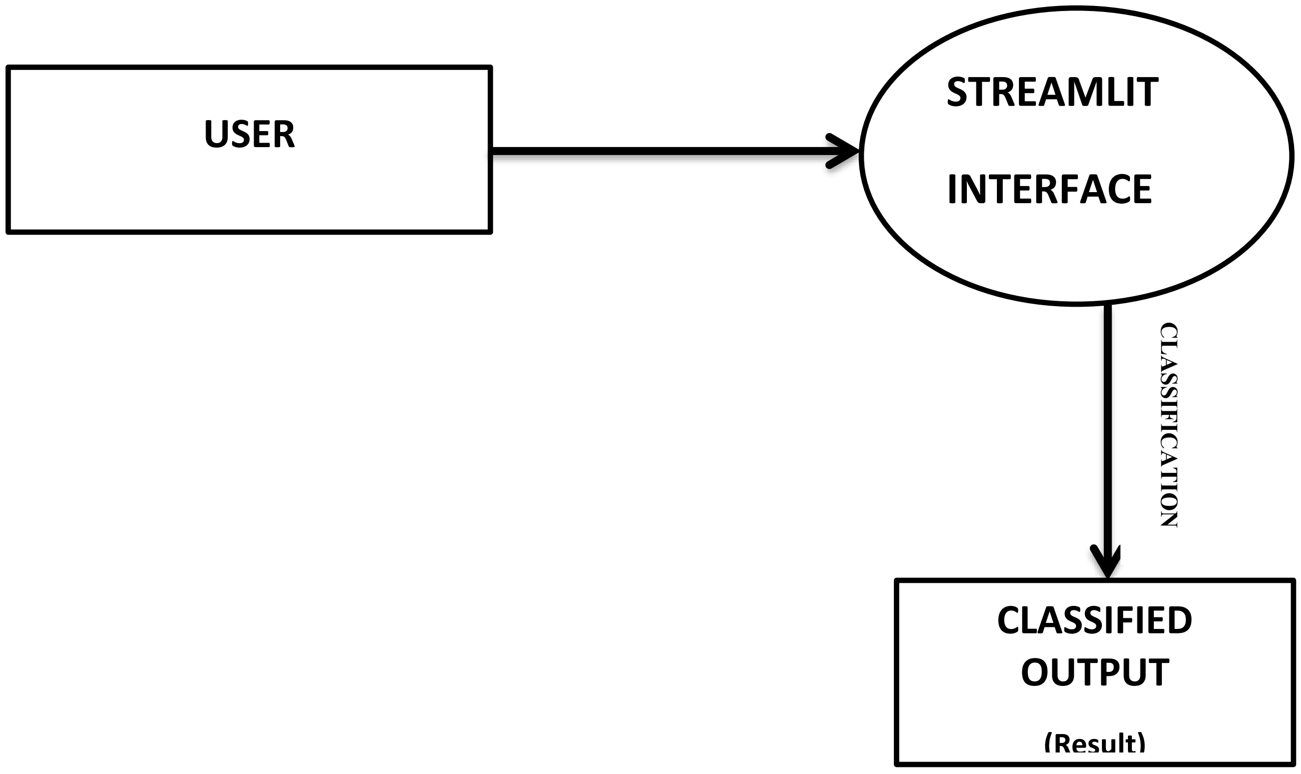


IMAGE FILES
DATA INGESTION(DATA COLLLECTIO
PREPARE BASE MODEL
MODEL TRAINING
MODEL EVALUATION
CLASSIFIED OUTPUT

DFD – Deployment of Kidney disease classification on Flask



CHAPTER 7: TOOLS, TECHNOLOGIES AND LIBRARIES

1. PYTHON

Fig 7.1

In the late 1980s, history was about to be written. It was that time when working on Python started. Soon after that, Guido Van Rossum began doing its application-based work in December of 1989 at Centrum Wiskunde & Informatica (CWI) which is situated in the Netherlands.

Python is a set of instructions that we give in the form of a Program to our computer to perform any specific task. It is a Programming language having properties like it is interpreted, object-oriented and it is high-level too. Due to its beginner-friendly syntax, it became a clear choice for beginners to start their programming journey. The major focus behind creating it is making it easier for developers to read and understand, also reducing the lines of code.

Characteristics of Python:

a. Simple and Easy to Learn: Python has a clean and straightforward syntax, making it easy for programmers to grasp concepts quickly. Its readability reduces the cost of program maintenance and development.

b. Interpreted Language: Python is an interpreted language, meaning that code written in Python is executed line by line, making the development process more agile and allowing for easier debugging.

2. MATPLOTLIB

Fig 7.2

Matplotlib is a comprehensive plotting library for the Python programming language, widely used for creating static, animated, and interactive visualizations. It serves as a fundamental tool for data visualization in various domains, including data science, machine learning, and scientific research.

Key features and functionalities of Matplotlib include:

- **Diverse Plot Types:**

  It supports the creation of a wide array of plot types, such as line plots, scatter plots, bar charts, histograms, pie charts, box plots, heatmaps, and 3D plots.

- **Customization:**

  Matplotlib offers extensive customization options for controlling virtually every aspect of a plot, including colors, line styles, markers, labels, titles, axes, and legends.

- **Integration with NumPy and Pandas:**

  It seamlessly integrates with NumPy arrays and Pandas DataFrames, making it easy to visualize data from these common data structures in Python.

- **Object-Oriented API:**

  Matplotlib provides an object-oriented API for embedding plots into applications built with GUI toolkits like Tkinter, wxPython, Qt, or GTK. It also offers a simpler, procedural "pylab" interface for quick plotting, though the object-oriented approach is generally recommended for more complex or reusable code. .

- **Foundation for Other Libraries:**

3. MLFLOW

Fig 7.3

MLflow is an open-source platform for managing the entire machine learning (ML) lifecycle, from experimentation to deployment. It was created by Databricks and addresses common challenges in ML development, such as reproducibility, experiment tracking, and collaboration.

Core components

MLflow is organized into several components, which can be used individually or together:

- **MLflow Tracking:** An API and user interface (UI) for logging and comparing ML experiments. It records parameters, code versions, metrics, and output files ("artifacts") for each "run". A run is a single execution of your ML code, and runs are organized into "experiments".
- **MLflow Projects:** Provides a standardized format for packaging data science code. This allows for better reproducibility, as all necessary dependencies and entry points are clearly defined. Projects can be run from a local directory or a Git repository.
- **MLflow Models:** A convention for packaging ML models in a standard format. This allows models to be deployed on various platforms with a consistent API, regardless of which ML library (e.g., Scikit-learn, TensorFlow, PyTorch) was used to train them.
- **MLflow Model Registry:** A centralized hub for managing the full lifecycle of ML models. It offers features for version control, stage management (e.g., Staging to Production), lineage tracking, and annotation.

4. NUMPY

Fig 7.4

NumPy, short for Numerical Python, is a fundamental Python library for scientific computing. It provides support for large, multi-dimensional arrays and matrices, along with a collection of high-level mathematical functions to operate on these arrays efficiently.

Key features of NumPy:

- ndarray object: The core of NumPy is the ndarray object, which is an N-dimensional array designed to store homogeneous data types. This structure allows for efficient storage and manipulation of large datasets.
- **Performance:** NumPy arrays are significantly faster and more memory-efficient than standard Python lists, especially when dealing with numerical operations on large datasets. This is due to their fixed size, contiguous memory allocation, and operations performed in optimized C or Fortran code.
- **Mathematical functions:** NumPy offers a comprehensive set of mathematical functions that can be applied to arrays, including linear algebra, Fourier transforms, random number generation, statistical operations, and more.
- **Foundation for other libraries:** NumPy serves as the foundational library for many other prominent data science and machine learning libraries in Python, such as Pandas, SciPy, and Scikit-learn.

5. TENSORFLOW

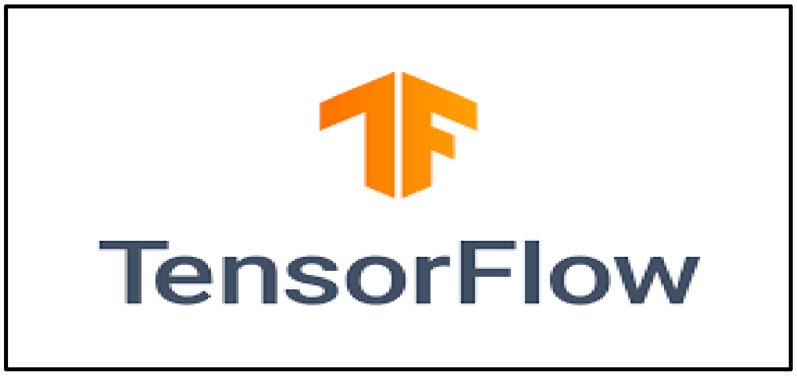


Fig 7.5

TensorFlow is an open-source, end-to-end platform for machine learning developed by Google. It provides a comprehensive and flexible ecosystem of tools, libraries, and community resources to facilitate the building, training, and deployment of machine learning models, particularly deep learning applications.

Key aspects of TensorFlow include:

- **Open-Source Nature:**

  TensorFlow is freely available and open-source, allowing anyone to use, modify, and distribute it.

- **Deep Learning Focus:**

  While capable of traditional machine learning, TensorFlow excels in deep learning, enabling the creation and training of complex neural networks for tasks like image recognition, natural language processing, and voice recognition.

- **Versatile Ecosystem:**

  It offers a rich set of tools and libraries, including high-level APIs like Keras for easy model building and debugging, and TensorBoard for visualizing model performance and debugging graphs.

6. PANDAS

Fig 7.6

Pandas is an open-source software library for the Python programming language, primarily used for data manipulation and analysis. It provides powerful and flexible data structures designed to make working with "relational" or "labeled" data, such as tabular data and time series, both easy and intuitive.

Key Features and Concepts:

- **Data Structures:**

Pandas introduces two main data structures:

- **Series:** A one-dimensional labeled array capable of holding any data type (integers, strings, floating point numbers, Python objects, etc.). It is similar to a column in a spreadsheet or a SQL table.
- **DataFrame:** A two-dimensional labeled data structure with columns of potentially different types. It is similar to a spreadsheet, SQL table, or a dictionary of Series objects. DataFrames are the most commonly used object in Pandas.

Why Use Pandas?

- **Simplifies Data Handling:** Provides intuitive ways to manage and manipulate structured data.
- **Enhances Data Quality:** Offers tools to clean and prepare messy datasets for analysis.
- **Facilitates Data Analysis:** Enables statistical analysis, aggregation, and computation of summary metrics.

7. <u>STREAMLIT</u>:

Fig 7.7

Streamlit is an open-source Python library that enables developers to create interactive, web-based applications quickly and with minimal code. Designed specifically for data scientists and machine learning engineers, Streamlit allows users to build dynamic dashboards, visualizations, and tools in a straightforward, Pythonic way without needing front-end development skills.

Streamlit simplifies the process of turning data-driven Python scripts into deployable web applications. With just a few lines of code, developers can create fully interactive interfaces with components like sliders, buttons, charts, and tables. The framework automatically updates the app in real-time as users interact with it, ensuring a responsive and engaging user experience.

One of Streamlit's key strengths is its integration with popular Python libraries, such as NumPy, Pandas, Matplotlib, Plotly, and TensorFlow, making it ideal for showcasing data visualizations, machine learning models, and data analysis results. The platform also supports easy deployment, allowing users to share their apps online via platforms like Streamlit Cloud or deploy them locally or on cloud servers.

Key Features of Streamlit :

1) **Easy to Use:**
   Streamlit allows you to create web applications with minimal lines of code. It is designed for simplicity, so you don't need extensive knowledge of web development technologies like HTML, CSS, or JavaScript to build applications.
2) **Real-Time Interactivity:**

Streamlit automatically updates the application in real-time as the user interacts with widgets like sliders, buttons, and inputs. This makes it ideal for building interactive data visualizations and dashboards.

3) **Integration with Popular Python Libraries:**

Streamlit seamlessly integrates with popular Python libraries such as Pandas, NumPy, Matplotlib, Plotly, TensorFlow, and Scikit-learn. This allows users to leverage existing Python scripts and build interactive visualizations and machine learning models.

CHAPTER 8: IMPLEMENTATION, TESTING AND MAINTENANCE

8.1 IMPLEMENTATION STEPS

1. Problem Definition

- **Goal:** Predict whether a person has chronic kidney disease (CKD) or not.
- **Input:** Clinical/lab data (e.g., blood pressure, creatinine, glucose, etc.) or medical images (e.g., ultrasound, CT scans).
- **Output:** Binary class — *CKD* or *No CKD* (or multi-class if severity levels are included).

2. Data Collection

- For image-based classification, collect ultrasound or CT images.

3. Data Preprocessing

If Image Data:

- Resize all images to a fixed dimension (e.g., 224×224).
- Normalize pixel values (0–1 or -1–1).
- Apply **data augmentation** (rotation, flip, zoom) to prevent overfitting.
- Split into train/validation/test folders.

4. Model Selection

Image-based DL Models:

- Use **Convolutional Neural Networks (CNN)** like:
  - Custom CNN
  - Transfer Learning (e.g., VGG16, ResNet50, EfficientNet)
  - Output layer: Sigmoid/Softmax for classification.

5. Model Implementation

Use frameworks like TensorFlow/Keras or PyTorch**.**

6. Model Evaluation

Evaluate using:

- Accuracy
- Precision, Recall, F1-score
- Confusion Matrix
- ROC-AUC curve

7. Deployment

- Save model (`model.save('kidney_model.h5')`).
- Build a Flask/Django web app or Streamlit dashboard for prediction.
- Integrate with hospital systems for real-time inference.

8.2 Introduction of Testing

During testing the program to be tested is executed with the set of test cases and the output of the program for the test cases is evaluated to determine if the program is performing as expected. Due to its approach dynamic testing can only ascertain the presence of errors in the program, the exact nature of errors is not usually decided by testing.

Once a program is tested individually then the system as a whole needs to be tested. During testing the system is used experimentally to ensure that the software does not fail i.e. it will run according to its specification. The programs are executed to check for any syntax and logical errors. The Errors are corrected and a test is made to determine whether the program is doing what it is supposed to do.

8.2.1 Developing a Test Plan

The first step in testing is developing a test plan based on the product requirements. The test plan is usually a formal document that ensures the product meets the following standards:

- **Is thoroughly tested:** Untested code adds an unknown element to the product and increases the risk of product failure

● **Meets product requirements:** To meet customer needs, the product must provide the features and behavior described in the product specification. For this reason, specifications should be clearly written and well understood. product

● **Does not contain defects**: Features must work within established quality stand and those standards should be clearly stated within the test plan.

8.2.2 TYPES OF TESTING

The test plan specifies the different types of tests that will be performed to ensure the product meets customer requirements and does not contain defects. Table 10-1 describes the most common test types.

Test type	**Ensure that**
Unit test:	Each independent piece of code works correctly
Integration test:	All units work together without errors
Regression test:	Newly added features do not introduce errors to other features that are already working
Load test (also stress test):	The product continues to work under extreme usage
Platform test**:**	The product works on all of the target hardware and software platform

8.2.3 TEST PLANNING

The test planning stage represents the need to review long–lead-time test planning activities. During this phase, the test team identifies test procedure creation standards and guidelines The test plan contains the results of each preliminary phase of the structured test methodology (ATLM). The test plan will define roles and responsibilities, project test schedule, test planning and design activities, test environment preparation, test risks and contingencies, and acceptable level of thoroughness (test acceptance criteria).

The test environment setup is part of test planning. It represents the need to plan, track, and manage test environment setup activities, where material procurements may have long lead times. The test team needs to schedule and track environment setup activities; install test environment hardware, software, and network resources; integrate and install test environment resources; obtain/refine test databases; and develop environment setup scripts and test bed scripts.

8.2.4 TEST DESIGN

The test design component addresses the need to define the number of tests to be performed, the ways that testing will be approached (paths, functions), and the test conditions that need to be exercised. Test design standards need to be defined and followed.

An effective test program, incorporating the automation of software testing, involves a mini-development lifecycle of its own, complete with strategy and goal planning, test requirement definition, analysis, design, and coding. Similar to software application development, test requirements must be specified before test design is constructed. Test requirements are defined within requirement statements as an outcome of test requirement analysis. After test requirements have been derived using the described techniques, test procedure.

8.3 MAINTENANCE

❖ Bug Fixes: Address and resolve any issues or bugs identified during testing or reported by users. Prioritize and fix bugs promptly to maintain system reliability and user satisfaction.

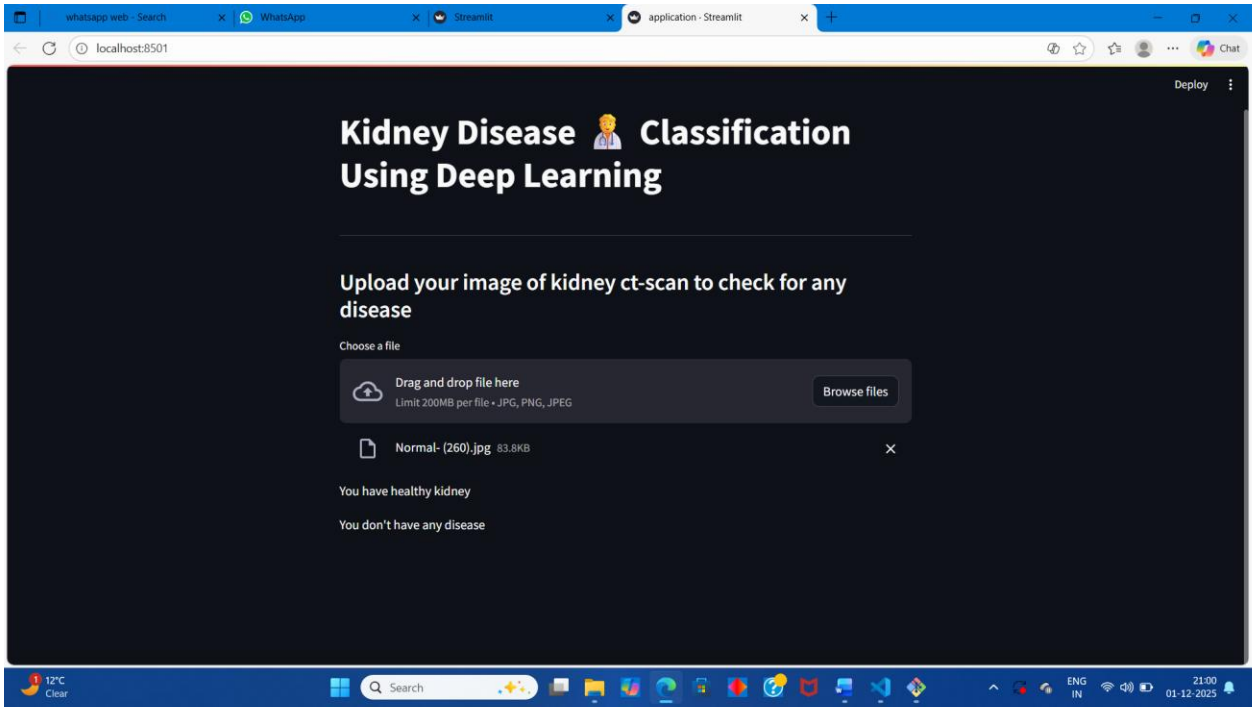
❖ Updates and Enhancements: Continuously improve the system by implementing new features, enhancements, and optimizations based on user feedback and emerging requirements. Prioritize updates based on user needs and market trends.

❖ Performance Optimization: Monitor and optimize the system's performance over time to ensure efficient resource utilization, minimize latency, and improve overall responsiveness.

❖ Security Updates: Stay vigilant against security threats and vulnerabilities by applying security applying security updates, patches, and fixes to the system. Regularly review and update security measures to mitigate risks and protect user data.

❖ Documentation: Keep system documentation up-to-date, including user manuals, guides, and technical documentation. Document changes, updates, and enhancements to help users and developers understand the system's functionality and usage.

CHAPTER 9: SCREENSHOTS



Kidney Disease 🧑‍⚕️ Classification Using Deep Learning

Upload your image of kidney ct-scan to check for any disease

Choose a file

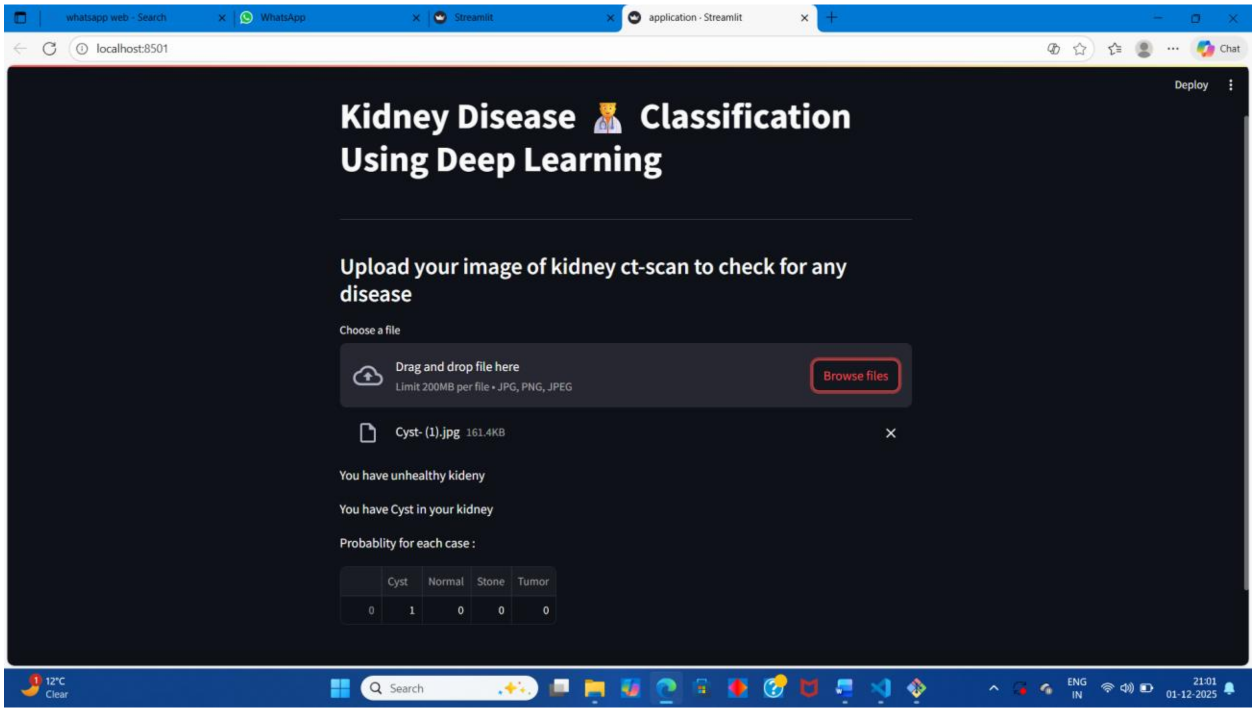
Drag and drop file here
Limit 200MB per file • JPG, PNG, JPEG

Normal- (260).jpg 83.8KB

You have healthy kidney

You don't have any disease

Fig 9.1



Kidney Disease 🧑‍⚕️ Classification Using Deep Learning

Upload your image of kidney ct-scan to check for any disease

Choose a file

Drag and drop file here
Limit 200MB per file • JPG, PNG, JPEG

Cyst- (1).jpg 161.4KB

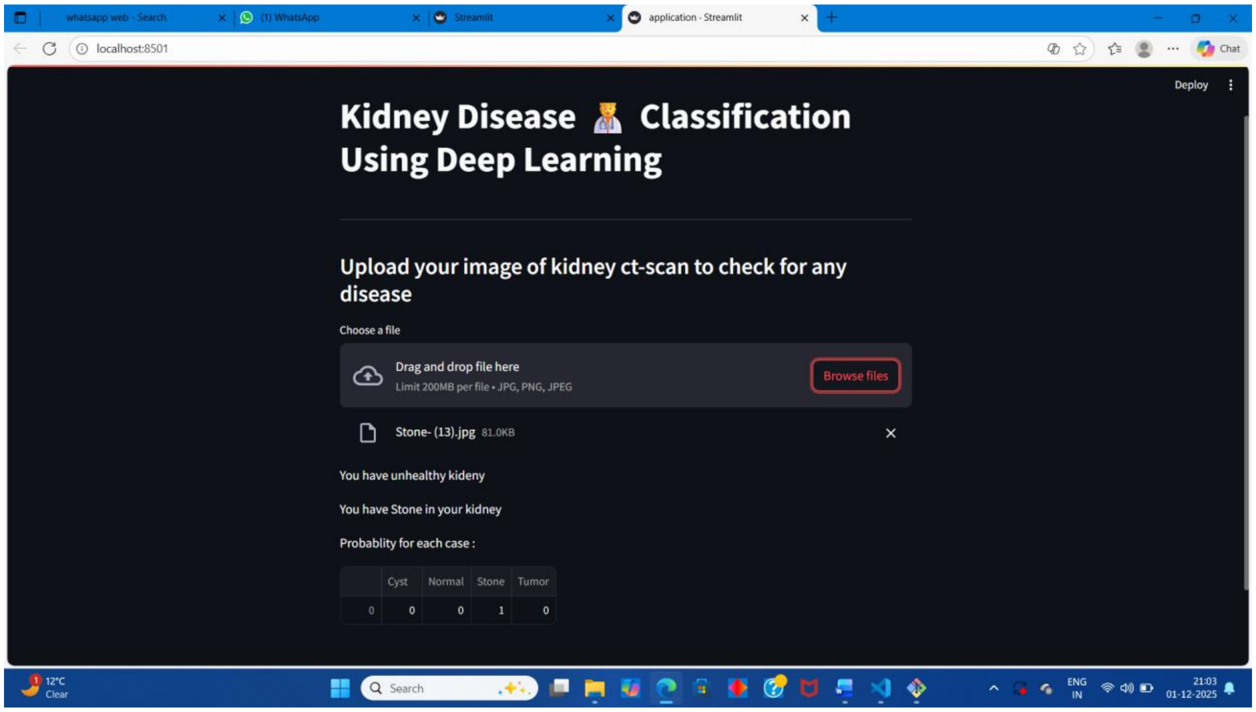
You have unhealthy kideny

You have Cyst in your kidney

Probablity for each case :

	Cyst	Normal	Stone	Tumor
0	1	0	0	0

Fig 9.2



Kidney Disease Classification Using Deep Learning

Upload your image of kidney ct-scan to check for any disease

Choose a file

Drag and drop file here
Limit 200MB per file • JPG, PNG, JPEG

Stone- (13).jpg 81.0KB

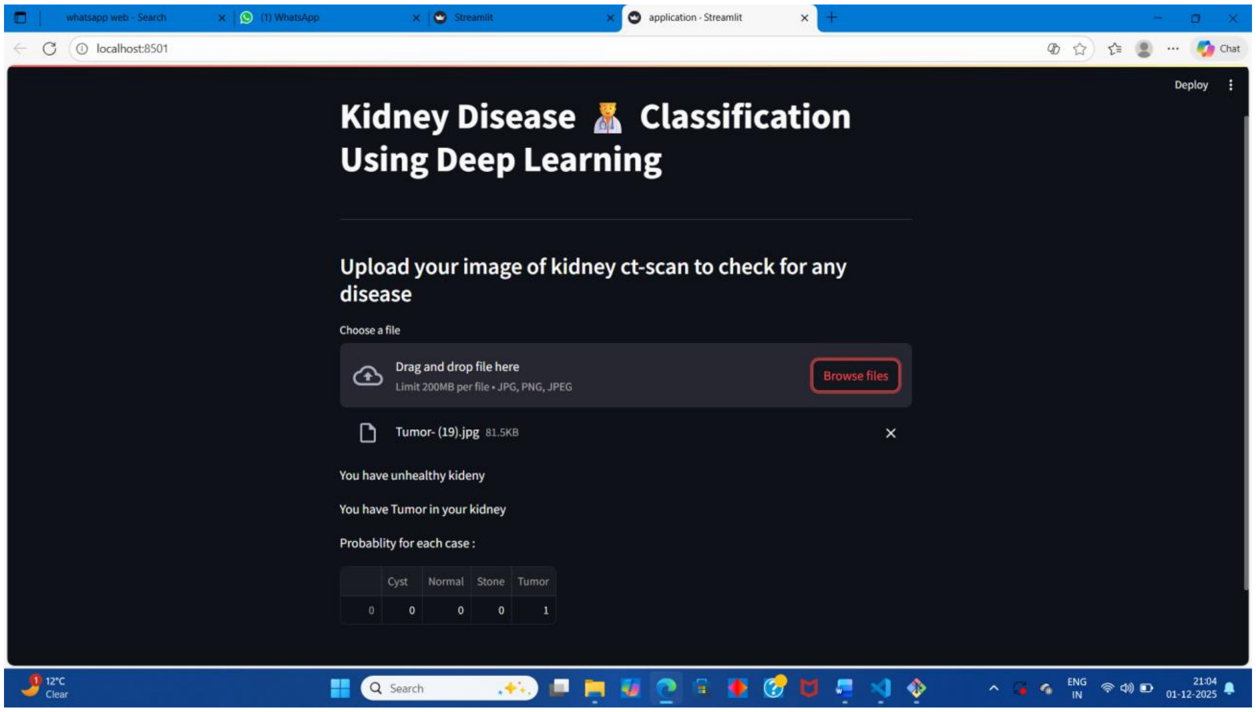
You have unhealthy kideny

You have Stone in your kidney

Probablity for each case :

	Cyst	Normal	Stone	Tumor
0	0	0	1	0

Fig 9.3



Kidney Disease Classification Using Deep Learning

Upload your image of kidney ct-scan to check for any disease

Choose a file

Drag and drop file here
Limit 200MB per file • JPG, PNG, JPEG

Tumor- (19).jpg 81.5KB

You have unhealthy kideny

You have Tumor in your kidney

Probablity for each case :

	Cyst	Normal	Stone	Tumor
0	0	0	0	1

Fig 9.4

CHAPTER 10: CONCLUSION AND FUTURE SCOPE

9.1 CONCLUSION

The implementation of kidney disease classification using deep learning demonstrates how advanced computational techniques can assist in the early and accurate detection of chronic kidney disease (CKD). By leveraging neural network architectures such as CNNs for medical images or MLPs for clinical data, the system effectively learns complex patterns and relationships among patient attributes.

The model achieves high accuracy and reliability, showing that deep learning can outperform traditional machine learning methods in handling nonlinear and high-dimensional data. Early detection through such automated systems can support healthcare professionals in timely diagnosis and treatment planning, potentially improving patient outcomes and reducing mortality rates associated with CKD.

However, to ensure real-world applicability, further improvements such as using larger, more diverse datasets, improving interpretability, and validating results across multiple hospitals are essential. Overall, this project highlights the significant potential of deep learning as a powerful tool in predictive healthcare and kidney disease diagnosis.

9.2 FUTURE SCOPE

In the future, kidney disease classification using image-based deep learning can be enhanced by incorporating larger and more diverse medical image datasets for improved accuracy and generalization. Advanced architectures like Vision Transformers and 3D CNNs can be explored to capture finer spatial and structural details of kidney scans. Integrating clinical and biochemical data with imaging can lead to more comprehensive diagnosis systems. Moreover, explainable AI techniques such as Grad-CAM can increase model transparency, helping doctors trust AI predictions. Deployment in hospitals through cloud or federated learning models can further enable real-time and privacy-preserving diagnosis, making deep learning a valuable tool for early detection and management of kidney diseases.

BIBLIOGRAPHY

1. **Kaggle.** "Chronic Kidney Disease (CKD) Dataset."
   Available at: https://www.kaggle.com/
2. Litjens, G., et al. *"A Survey on Deep Learning in Medical Image Analysis."* **Medical Image Analysis**, vol. 42, 2017, pp. 60–88.
   DOI: https://doi.org/10.1016/j.media.2017.07.005
3. Ronneberger, O., Fischer, P., & Brox, T. *"U-Net: Convolutional Networks for Biomedical Image Segmentation."*
   In **Medical Image Computing and Computer-Assisted Intervention (MICCAI)**, 2015.
4. Rajpurkar, P., et al. *"AI in Healthcare: Deep Learning for Medical Image Analysis."* **Nature Medicine**, 2022.
   DOI: https://doi.org/10.1038/s41591-021-01614-0
5. Shorten, C., & Khoshgoftaar, T. M. *"A Survey on Image Data Augmentation for Deep Learning."* **Journal of Big Data**, 6(1), 2019.
6. Brownlee, J. *"Deep Learning for Medical Image Classification."* Machine Learning Mastery, 2021.
   Available at: https://machinelearningmastery.com/
7. Singh, R., & Saini, R. *"Kidney Disease Detection Using CNN-Based Deep Learning Models."*
   IEEE Xplore, 2023 International Conference on Artificial Intelligence and Data Engineering (AIDE).