# CHAPTER - 1 INTRODUCTION

Automated car damage assessment is a revolutionary approach to evaluating vehicle damage using advanced technologies such as artificial intelligence (AI), computer vision, and machine learning. Traditionally, car damage assessment has relied on manual inspections by trained professionals, which can be time-consuming and subjective. However, with the advent of automated systems, the process has been streamlined, leading to faster and more accurate evaluations.

By leveraging computer vision techniques, automated car damage assessment systems can analyze high-resolution images or videos of damaged vehicles. These images are processed to extract key information, such as the location and extent of the damage, and to identify different types of damages, such as scratches, dents, or structural issues. The processed data is then fed into AI algorithms and machine learning models trained on extensive datasets to analyze the damage and generate comprehensive reports.

The use of automated car damage assessment offers numerous benefits. Firstly, it significantly reduces the time required for evaluation, allowing for faster processing of insurance claims and facilitating quicker repairs. Secondly, the automated systems provide consistent and objective assessments, eliminating human error and subjective judgments. This leads to more reliable evaluations and fairer settlement of claims. Additionally, automated assessment helps reduce costs for insurance providers and vehicle owners by preventing overestimation or underestimation of repair costs.

As the technology continues to advance, automated car damage assessment is expected to become increasingly prevalent in the automotive industry. Its ability to improve efficiency, accuracy, and customer experience makes it an invaluable tool for insurance companies, repair shops, and vehicle owners. By harnessing the power of AI and computer vision, automated car damage assessment is poised to revolutionize the way vehicle damages are evaluated, ensuring a more streamlined and effective process for all stakeholders involved.

## PURPOSE

The purpose of automated car damage assessment is to streamline and optimize the process of evaluating vehicle damages. By leveraging advanced technologies such as computer vision, machine learning, and artificial intelligence, it aims to replace time-consuming manual inspections with automated analysis of high-resolution images or videos. The primary goal is to improve efficiency and accuracy in assessing the extent and severity of car damages, resulting in faster claims processing and more informed decision-making for insurance companies, repair shops, and vehicle owners.

Furthermore, automated car damage assessment seeks to enhance customer experience by providing a seamless and convenient claims process. With automated systems, customers can submit damage reports remotely and receive prompt evaluations, eliminating the need for physical inspections and reducing wait times.

## AIM

The goal is to eliminate the need for manual inspections and subjective judgments, replacing them with automated analysis of images or videos. The aim is to provide objective and consistent assessments of car damages, enabling faster claims processing, more informed decision-making for insurance companies, and quicker repairs for vehicle owners.

## OBJECTIVE

The objective of automated car damage assessment is to provide a reliable, efficient, and accurate method for evaluating and analyzing vehicle damages.

By utilizing advanced technologies like computer vision, machine learning, and artificial intelligence, the objective is to replace time-consuming and subjective manual inspections with automated systems that can assess car damages based on high-resolution images or videos. The primary objective is to eliminate human error and inconsistencies, ensuring consistent and objective

## BACKGROUND OF PROJECT

The background of automated car damage assessment stems from the increasing demand for efficient and accurate evaluation of vehicle damages in the automotive industry. Traditional methods relied heavily on manual inspections, which were time-consuming, subjective, and prone to human error.

However, with the advancements in computer vision, machine learning, and artificial intelligence, automated car damage assessment has emerged as a promising solution. The availability of high-resolution imaging devices, coupled with sophisticated algorithms and AI models, allows for the analysis and interpretation of visual data to determine the extent and severity of car damages.

This background has driven the development and adoption of automated car damage assessment systems, aiming to streamline the evaluation process, improve accuracy, reduce processing time, and enhance overall customer experience in dealing with car damages and insurance claims.

# CHAPTER - 2 LITERATURE REVIEW

## INTRODUCTION

In this literature survey several methods have been proposed for detection of car damage.

Srimal et al. [ 4 ] proposed a solution which uses 3D Computer Aided Design for the discernment of car damage from the picture, the system only detect damage at edge portion only. Detection of the car damage through CAD software requires some knowledge about the software.

8S Gontscharov et al [ 5 ] ,the proposed system designed by using YOLO(you only look once) algorithm to detect tha car damage, Here the multi sensor data fusion technique is allows to locate the portion of damage more accurately and performs detection faster compared to other algorithms which is fully automatic and doesn’t require much human intervention.

Phyu Mar Kyu et al [ 3], the proposed system uses deep learning based algorithm are VGG16 and VGG19 damaged car detection in the real world. This algorithm notice the severity of the damaged car based on the location. Finally, the author concludes that L2 regularization work greater.

Girish N et al [ 2 ], the proposed system uses vehicle damage detection technique depends on transfer learning and mask RCNN, The mask regional convolution neural network determines a damaged car by its position and estimate the depth of the damage. A Neela Madheswari et al [ 1 ], the proposed system uses convolution neural network is use to accept that image contains a car damage or not. It take as great opportunities to attempt by classifying the car damage into different classes.

## EXISTING SYSTEM

The old system or process of car damage assessment involved manual inspections conducted by trained professionals. When a vehicle was damaged, it would be taken to a repair shop or an insurance company where an expert would physically examine the car to evaluate the extent of the damage. This process typically required visual inspection, measuring tools, and expertise in identifying different types of damages such as dents, scratches, or paint chips. The professional would then provide an assessment report detailing the location and severity of the damages, along with an estimated cost of repairs. This manual approach was time-consuming, subjective, and prone to human error, as it relied on the expertise and judgment of the individual conducting the assessment. It often resulted in delays in the claims process and inconsistencies in evaluations.



**Fig 2.2: Existing system**

.

## PROPOSED SYSTEM

Detect the car damage using photo taken at the accident scene is very useful to reduce the cost of processing insurance claims, as well as provide greater convenience for vehicle users. The following methods are used in the proposed system.

* + 1. Dataset Explanation.
    2. Describing the level of damage.
    3. CNN Model.
    4. VGG16 Algorithm.

### Dataset Explanation

Data preparation is very costly depending on the demand of marking the data. VGG16 can be used to need as a true image in an input. Cross-validation is an approximate for our models to takes a more time since, it is very costly to train the VGG16 for many years. Consequently, split the dataset arbitrarily into distinct set for training and validation. Car is to train for multiple times. At the end train and test can be split for similar images. In this dataset we use more different types of car images. Report our three collected datasets are following.

* + - * Image Net dataset - Vehicle
      * Dataset - All the three dataset are contained train and validation of damaged and undamaged cars.

### Describing the Level of Damage

Damaged car can be defined by their incidence. We think about each damaged part into small, average, severe. The categorization of the damaged car levels as follows.

* + - * Small Damage - creaks in headlight.
      * Average Damage - Damage in car doors.
      * Severe Damage - damage of air bags.

### CNN Model

CNN is one of the neural network it is used for processing the image and segmentation of the image. In this project we use a convolution neural network model for detect the image contains a car. CNN is also used to analyses the damage of the car.

### VGG16 Algorithm

The Image Net Large Scale Visual Recognition Challenge is one of the visions of computer. They contain two jobs. Initial is to detect things within an image called object localization. Next is to classifying the images called image classification.

CNN is the one of the best vision model planning. In VGG16 contains four layers they are convolution, max pooling, and fully connected SoftMax. In this algorithm 16 refers to contain 16 layers.

# CHAPTER - 3

**SYSTEM ANALYSIS**

## HARDWARE AND SOFTWARE REQUIREMENTS

## Hardware Requirements:

* Processor: Minimum Core i5 processor
* RAM: 2GB (Minimum) or 8GB (Recommended)
* Hard Disk: 512GB

## Software Requirements:

* Operating System: Windows 10 or 11
* Technology: Python version 2020.3.1
* Anaconda: Edition 2020.11

## SOFTWARE REQUIREMENTS SPECIFICATION

## Python Overview

Python is a high-level, interpreted, interactive, and object-oriented scripting language.

Python is designed to be highly readable. It uses English keyword frequently whereas other languages use punctuation, constructions than other languages.

* **Python is Interpreted:** Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
* **Python is Interactive:** You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
* **Python is Object-Oriented:** Python supports the Object-Oriented style or technique of programming that encapsulates code within objects.
* **Python is a Beginner's Language:** Python is a great language for beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

## HISTORY OF PYTHON

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol68, Small Talk, Unix shell, and other scripting languages. Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL). Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

## Python Features

Python's features include

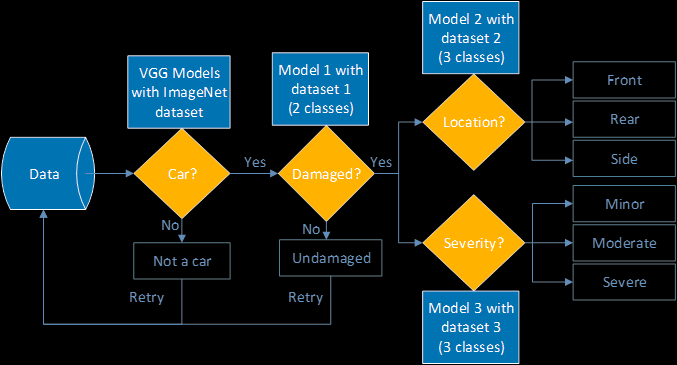
* **Easy-to-learn:** Python has few keywords, a simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
* **Easy-to-read:** Python code is more clearly defined and visible to the eyes.
* **Easy-to-maintain:** Python's source code is fairly easy-to-maintained.
* **A broad standard library:** Python's bulks of the library are very portable and cross platform compatible on UNIX, Windows, and Macintosh.
* **Interactive Mode:** Python has support for an interactive mode that allows interactive testing and debugging of snippets of code.
* **Portable:** Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* **Extendable:** You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
* **Databases:** Python provides interfaces to all major commercial databases.
* **GUI Programming:** Python supports GUI applications that can be created and ported to many system calls, libraries, and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

# CHAPTER - 4 SYSTEM DESIGN

## ARCHITECTURE

The architecture of automated car damage assessment typically involves several key components. Firstly, high-resolution images or videos of the damaged vehicle are captured using cameras or other imaging devices. These visuals serve as input data for the assessment system. Next, computer vision techniques are applied to process the images, including tasks such as image segmentation to identify and isolate the damaged areas from the rest of the vehicle. Feature extraction algorithms may also be employed to extract relevant information from the images.

The processed data is then fed into machine learning models and artificial intelligence algorithms. These models have been trained on large datasets of labeled car damage images, allowing them to recognize patterns and classify different types of damages accurately. The models analyze the input data, evaluate the severity of the damages, and generate a detailed assessment report.

The architecture may also include a database to store and manage the collected data and assessment reports. Additionally, a user interface or application can be developed to facilitate the interaction between users (such as insurance companies, repair shops, or vehicle owners) and the automated car damage assessment system. The system's architecture aims to integrate computer vision, machine learning, and AI algorithms to automate the assessment process and provide accurate, consistent, and efficient car damage evaluations.

***Fig. 4.1 System Architecture for heart attack prediction system***

# Data Flow and Control Flow diagrams

## Control Flow Diagram:

A **Control Flow Graph (CFG)** is the graphical representation of control flow or [computation](https://www.geeksforgeeks.org/cyclomatic-complexity/) [during the execution of programs](https://www.geeksforgeeks.org/cyclomatic-complexity/) or applications. Control flow graphs are mostly used in static analysis as well as compiler applications, as they can accurately represent the flow inside of a program unit. The control flow graph was originally developed by Frances E. Allen.

### Characteristics of Control Flow Graph:

* + Control flow graph is process oriented.
  + Control flow graph shows all the paths that can be traversed during a program execution.
  + Control flow graph is a directed graph.
  + Edges in CFG portray control flow paths and the nodes in CFG portray basic blocks. There exist 2 designated blocks in Control Flow Graph:

### Entry Block:

Entry block allows the control to enter into the control flow graph.

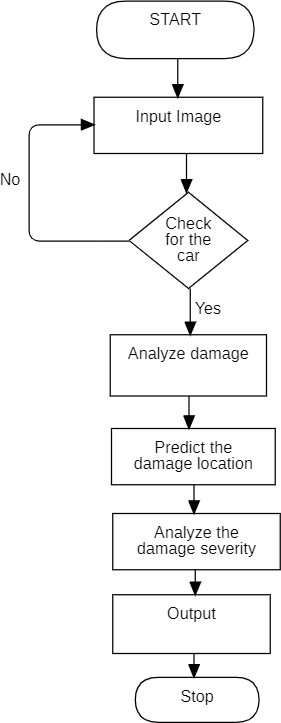
### Exit Block:

Control flow leaves through the exit block.

Hence, the control flow graph is comprised of all the building blocks involved in a flow diagram such as the start node, end node and flows between the nodes.

Control flow diagrams typically use symbols and arrows to represent different types of operations and the flow of control between them. Some common symbols include:

* Start/end symbol: This symbol represents the beginning and end of the process or program.
* Process symbol: This symbol represents an operation or step in the process. It may also include additional information about the operation, such as inputs and outputs.
* Decision symbol: This symbol represents a decision point in the process where the flow of control may split into different paths depending on a condition.
* Connector symbol: This symbol represents the connection between different parts of the diagram.



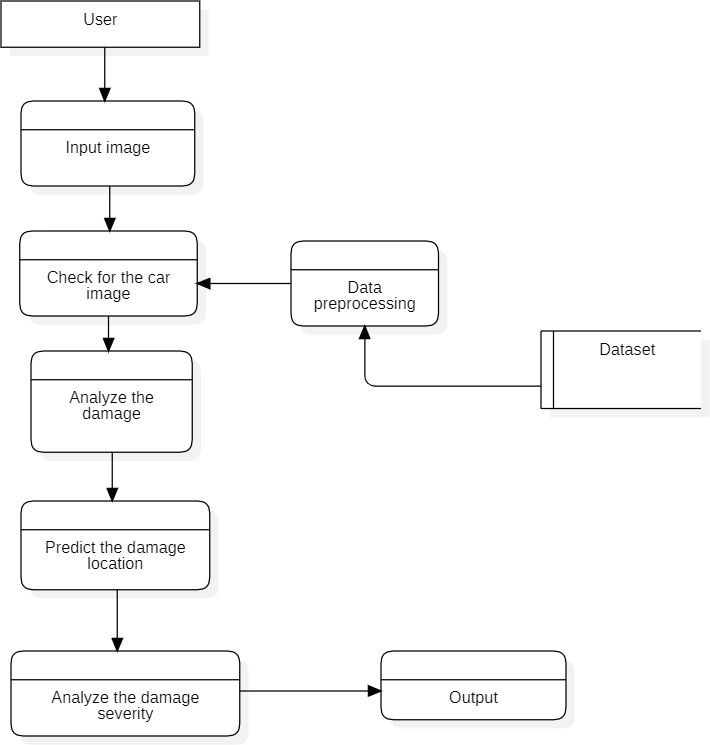
***Fig. 4.2 Control flow diagram for automated car damage assessment***

**Description:** The above diagram represents the control for diagram for automated car damage assessment. It shows the graphical representation of control flow or computing during the execution of programs or applications. Here we have one start symbol, process symbol, decision symbols, connector symbols. The process will start with the start symbol after that the user will give input image and it checks for weather it is a car or not. If it is a car, it will analyze damage, predict the damage location and analyze the damage severity.

## Data flow diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both.

It shows how data enters and leaves the system, what changes the information, and where data is stored.

The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.

***Fig. 4.3 Data flow control diagram for automated car damage assessment***

**Description:** The above diagram represents the data flow diagram for automated car damage assessment. It is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both. Here the user as to give input image and check weather it is a car or not by taking the data from the dataset then it analyzes the damage.

## UML DIAGRAMS

The Unified Modelling Language (UML) is used to specify, visualize, modify, construct and document the artifacts of an object-oriented software-intensive system under development. UML offers a standard way to visualize a system's architectural blueprints, including elements such as

* Actors
* Business processes
* Logical components
* Activities
* Programming language statements
* Database schemas and reusable software components.

UML combines best techniques from data modelling (entity relationship diagrams), business modelling (workflows), object modelling, and component modelling. It can be used with all processes, throughout the software development life cycle, and across different implementation technologies. UML has synthesized the notations of the Booch method, the Object modelling technique (OMT), and Object-oriented software engineering (OOSE) by fusing them into a single, common, and widely usable modelling language. UML aims to be a standard modelling language that can model concurrent and distributed systems.

## USE CASE DIAGRAM:

A use case diagram is used to represent the dynamic behaviour of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system.

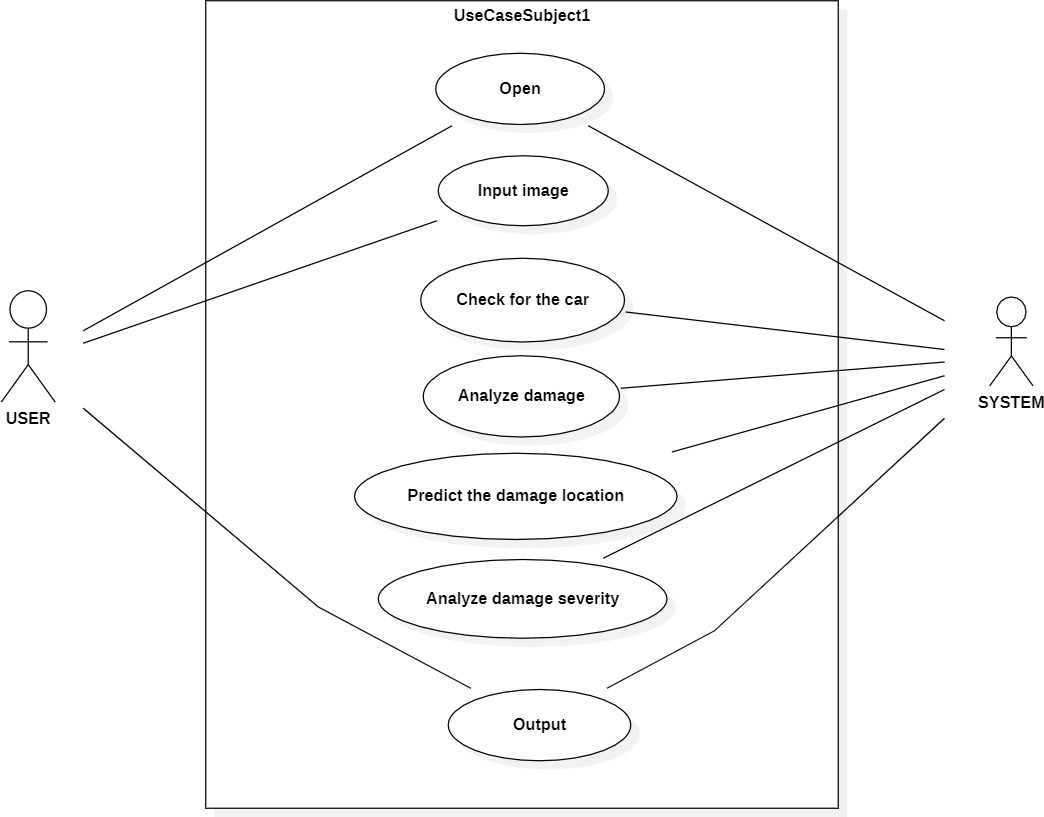
Following are the purposes of a use case diagram given below

* It gathers the system's needs.
* It depicts the external view of the system.
* It recognizes the internal as well as external factors that influence the system.
* It represents the interaction between the actors.

Following are some rules that must be followed while drawing a use case diagram:

A pertinent and meaningful name should be assigned to the actor or a use case of a system.

* **Use cases:** Horizontally shaped ovals that represent the different uses that a user might have.
* **Actors:** Stick figures that represent the people actually employing the use cases.
* **Associations:** A line between actors and use cases. In complex diagrams, it is important to know which actors are associated with which use cases.



***Fig 4.4 Use case Diagram for automated car damage assessment***

**Description:** The above diagrams represent use case diagram for automated car damage assessment. It showsthe all-possible interaction between the actors and the use cases. We have two actors a user andsystem. User has to enter the details in the user interface provided. System will collect the imageentered by the user. Then system will match the given image with the data image entered by the user andclassification of image is done. After classifying the image, the system will analyze the damage, it predicts the damage location and analyze the damage severity.

## CLASS DIAGRAM:

Class diagram is a static diagram. It represents the static view of an application. The class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. The class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modelling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. The class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

The purpose of the class diagram is given below:

* Analysis and design of the static view of an application.
* Describe the responsibilities of a system.
* The base for component and deployment diagrams.
* Forward and reverse engineering.

The following points should be remembered while drawing a class diagram:

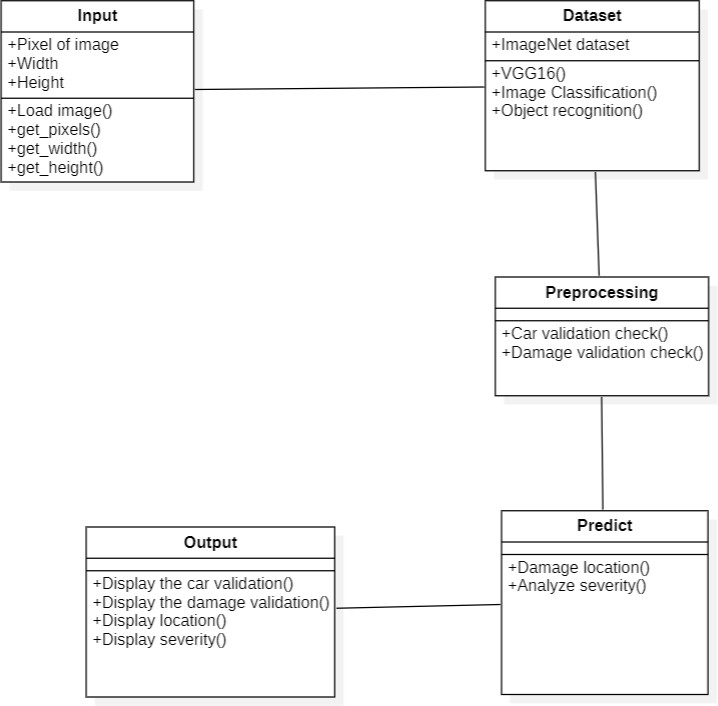
* The name of the class diagram should be meaningful to describe the aspect of the system.
* Each element and its relationships should be identified in advance.
* Responsibility (attributes and methods) of each class should be clearly identified

For each class, a minimum number of properties should be specified, as unnecessary properties will make the diagram complicated.

Use notes whenever required to describe some aspect of the diagram.

The class diagram is the main building block of [object-oriented](https://www.wikiwand.com/en/Object-oriented_programming) modeling. It is used for general [conceptual modeling](https://www.wikiwand.com/en/Conceptual_model) of the structure of the application, and for detailed modeling, translating the models into [programming code.](https://www.wikiwand.com/en/Programming_code) Class diagrams can also be used for [data](https://www.wikiwand.com/en/Data_modeling) [modeling](https://www.wikiwand.com/en/Data_modeling). The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.

In the diagram, classes are represented with boxes that contain three compartments:

* + The top compartment contains the name of the class. It is printed in bold and centered, and the first letter is capitalized.
  + The middle compartment contains the attributes of the class. They are left-aligned and the first letter is lowercase.

***Fig 4.5 Class Diagram for automated car damage assessment***

**Description:** This above diagram represents the class diagram for automated car damage assessment. We have 5 classes with different class labels namely input, dataset, preprocessing, predict and output. Each class have different attributes and operations to perform. User class provides input to system. Dataset class does the training and testing of data operation.

## SEQUENCE DIAGRAM:

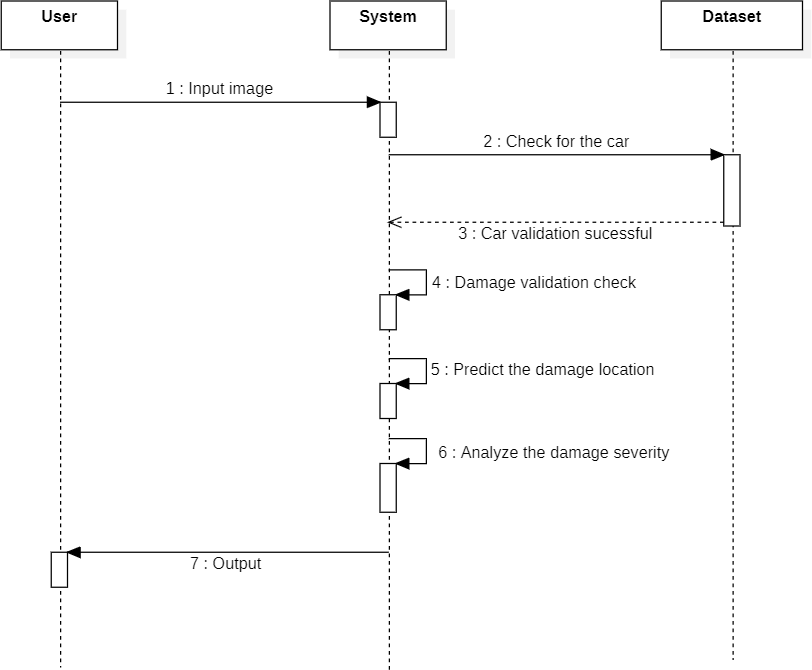
A sequence diagram is the most commonly used interaction diagram. Interaction diagram **–** An interaction diagram is used to show the interactive behaviour of a system. Since visualizing the interactions in a system can be a cumbersome task, we use different types ofinteraction diagrams to capture various features and aspects of interaction in a system.

**Sequence Diagrams –** A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

## Sequence Diagram Notations –

1. **Actors –** An actor in a UML diagram represents a type of role where it interacts with the system and its objects. It is important to note here that an actor is always outside the scope of the system we aim to model using the UML diagram.
2. **Lifelines –** A lifeline is a named element which depicts an individual participant in a sequence diagram. So basically, each instance in a sequence diagram is represented by a lifeline. Lifeline elements are located at the top in a sequence diagram. The standard in UML for naming a lifeline follows the following format – Instance Name: Class Name.
3. **Messages –** Communication between objects is depicted using messages. The messages appear in a sequential order on the lifeline. We represent messages using arrows. Lifelines and messages form the core of a sequence diagram.
4. **Delete Message** – We use a Delete Message to delete an object. When an object is deallocated memory or is destroyed within the system, we use the Delete Message symbol. It destroys the occurrence of the object in the system. It is represented by an arrow terminating with a x.
5. **Self-Message –** Certain scenarios might arise where the object needs to send a message to itself. Such messages are called Self Messages and are represented with a U-shaped arrow.
6. **Reply Message –** Reply messages are used to show the message being sent from the receiver to the sender. We represent a return/reply message using an open arrowhead with a dotted line. The interaction moves forward only when a reply message is sent by the receiver.

## Uses of sequence diagrams –

* + Used to model and visualize the logic behind a sophisticated function, operation or procedure.
  + They are also used to show details of UML use case diagrams.

***Fig 4.6 Sequence Diagram for automated car damage assessment***

**Description:** The about represents the sequence diagram for automated car damage assessment. We have 3 lifelines namely user, system and database. Each lifeline plays different role. The sequence diagram shows the sequences of actions taking place in the analyzing system. Firstly, the user gives input image to the system and the system retrieves the image from dataset. Then the system compares the data and runs the algorithm.

### ACTIVITY DIAGRAM

The activity diagram is used to demonstrate the flow of control within the system rather than the implementation. It models the concurrent and sequential activities. The activity diagram helps in envisioning the workflow from one activity to another. It emphasized the condition of flow and the order in which it occurs. The flow can be sequential, branched, or concurrent, and to deal with such kinds of flows, the activity diagram has come up with a fork, join, etc.

It is also termed an object-oriented flowchart. It encompasses activities composed of a set of actions or operations that are applied to model the behavioral diagram.

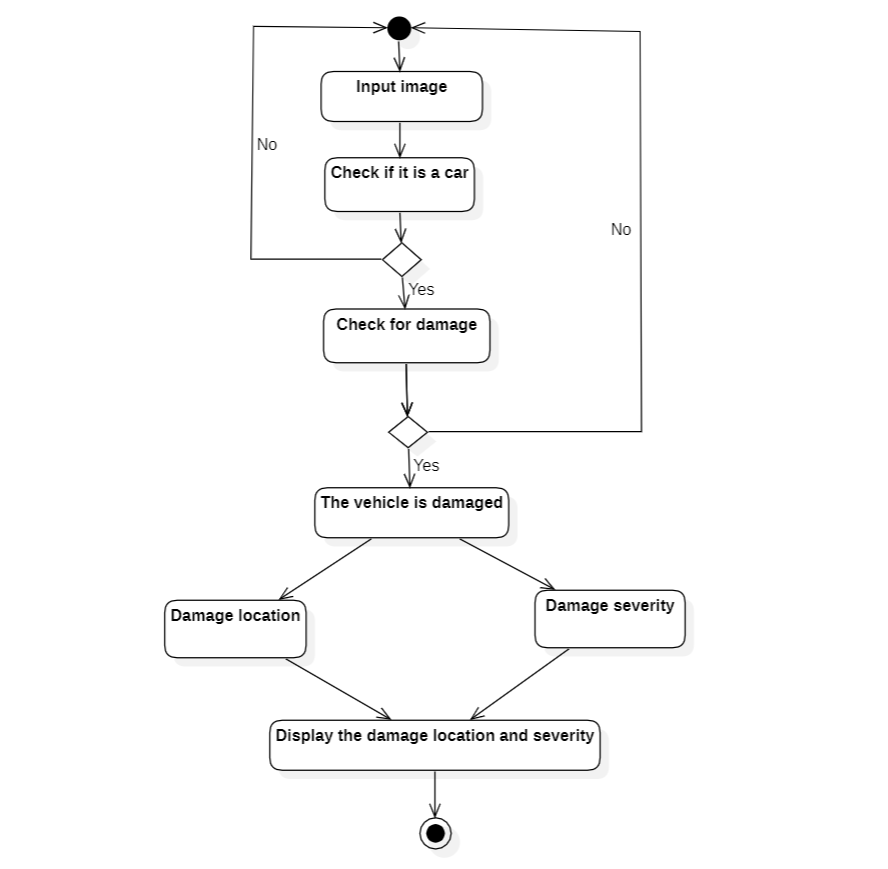
The purpose of an activity diagram can be described as –

* Draw the activity flow of a system.
* Describe the sequence from one activity to another.
* Describe the parallel, branched, and concurrent flow of the system.

## Basic components of an activity diagram

Before you begin making an activity diagram, you should first understand its makeup. Some of the most common components of an activity diagram include:

* + **Action:** A step in the activity wherein the users or software perform a given task. In Lucid chart, actions are symbolized with round-edged rectangles.
  + **Decision node:** A conditional branch in the flow that is represented by a diamond. It includes a single input and two or more outputs.
  + **Control flows:** Another name for the connectors that show the flow between steps in the diagram.
  + **Start node:** Symbolizes the beginning of the activity. The start node is represented by a black circle.
  + **End node:** Represents the final step in the activity. The end node is represented by an outlined black circle.



***Fig 4.7 Activity Diagram for heart attack prediction system***

**Description:** The above diagram represents the activity diagram for automated car damage assessment. This activity diagram shows how the control flows in the damage assessment. The rectangles represent action performed in the analyzing and prediction. First, we start the process by executing the code. Then the user provides the image to the system. Then the system will analyze the damage, predict the damage location and analyze the damage severity. It will display the damage location and severity.

# CHAPTER - 5 IMPLEMENTATION AND RESULT

## METHOD OF IMPLEMENTATION

The implementation phase is where you and your project team actually do the project work to produce the deliverables. The word “deliverable” means anything your project delivers. The deliverables for your project include all of the products or services that you and your team are performing for the client, customer, or sponsor, including all the project management documents that you put together.

The steps undertaken to build each deliverable will vary depending on the type of project you are undertaking, and cannot therefore be described here in any real detail. For instance, engineering and telecommunications projects will focus on using equipment, resources, and materials to construct each project deliverable, whereas computer software projects may require the development and implementation of software code routines to produce each project deliverable. The activities required to build each deliverable will be clearly specified within the project requirements document and project plan.

## TECHNOLOGIES

### Machine Learning:

Machine learning is a subfield of artificial intelligence (AI) that involves the use of algorithms and statistical models to enable computer systems to automatically learn from data, without being explicitly programmed.

The goal of machine learning is to enable computers to improve their performance on a specific task over time by learning from experience. This is typically done by training the machine learning algorithm on a large set of labeled data, which enables the system to identify patterns and make predictions or decisions based on new, unseen data.

Machine learning involves showing a large volume of data to a machine so that it can learn and make predictions, find patterns, or classify data. The three machine learning types are supervised, unsupervised, and reinforcement learning.

### Supervised learning:

This machine learning type got its name because the machine is “supervised” while it's learning, which means that you’re feeding the algorithm information to help it learn. This type of machine learning feeds historical input and output data in machine learning algorithms, with processing in between each input/output pair that allows the algorithm to shift the model to create outputs as closely aligned with the desired result as possible. Common algorithms used during supervised learning include neural networks, decision trees, linear regression, and support vector machines.

### Unsupervised learning:

While supervised learning requires users to help the machine learn, unsupervised learning doesn't use the same labeled training sets and data. Instead, the machine looks for less obvious patterns in the data. This machine learning type is very helpful when you need to identify patterns and use data to make decisions. Common algorithms used in unsupervised learning include Hidden Markov models, k-means, hierarchical clustering, and Gaussian mixture models.

### Reinforcement learning:

Reinforcement learning is the closest machine learning type to how humans learn. The algorithm or agent used learns by interacting with its environment and getting a positive or negative reward. Common algorithms include temporal difference, deep adversarial networks, and Q-learning. Going back to the bank loan customer example, you might use a reinforcement learning algorithm to look at customer information. If the algorithm classifies them as high-risk and they default, the algorithm gets a positive reward. If they don't default, the algorithm gets a negative reward. In the end, both instances help the machine learn by understanding both the problem and environment better.

## Flask Python:

Flask is a web framework, it’s a Python module that lets you develop web applications easily. It’s has a small and easy-to-extend core: it’s a micro framework that doesn’t include an ORM (Object Relational Manager) or such features. It does have many cool features like URL routing, template engine. It is a WSGI web app framework.

Flask has become popular among Python enthusiasts. As of October 2020, it has second most stars on [GitHub](https://en.wikipedia.org/wiki/GitHub) among Python web-development frameworks, only slightly behind [Django](https://en.wikipedia.org/wiki/Django_(web_framework)) and was voted the most popular web framework in the Python Developers Survey 2018, 2019, 2020 and 2021

Flask provides a set of features and capabilities that make it a popular choice for building web applications, including:2

* **Routing:** Flask provides a routing mechanism that maps URL endpoints to Python functions, allowing you to define the behavior of your application based on the URLs that are requested.
* **Templating:** Flask supports template engines like Jinja2, which allow you to generate dynamic HTML pages based on the data passed to them.
* **Request handling:** Flask provides a simple and flexible way to handle HTTP requests and responses, with support for request and response objects, cookies, and sessions.
* **Extensions:** Flask has a large number of extensions available that can add additional functionality to your application, such as database integration, user authentication, and more.
* **Lightweight:** Flask is a lightweight framework that is easy to learn and quick to set up, making it an ideal choice for small to medium-sized web applications.
* **RESTful API development:** Flask is often used to develop RESTful APIs due to its flexibility and ease of use.
* **Testing:** Flask provides a built-in test client that allows you to test your application's functionality and behavior.

Flask is commonly used to build web applications, APIs, and microservices. It is well- suited for small to medium-sized projects that require a simple and flexible web framework. Flask is also a popular choice for building prototypes and proof-of-concept applications due to its ease of use and quick setup time.

## Bootstrap:

Bootstrap is a popular framework used in web development, but it is not directly related to automated car damage assessment. However, Bootstrap can be utilized in the user interface (UI) development aspect of an automated car damage assessment system.

When building an automated car damage assessment system, a web-based interface can be developed to interact with users and display the assessment results. Bootstrap can be employed to create a responsive and visually appealing UI for this purpose.

Bootstrap provides a collection of pre-designed UI components, such as buttons, forms, tables, and navigation menus, along with a responsive grid system that automatically adjusts the layout based on the screen size. This allows for the creation of a user-friendly interface that works well on different devices and screen resolutions.

Additionally, Bootstrap offers CSS (Cascading Style Sheets) and JavaScript-based functionalities that can enhance the interactivity and user experience of the application. For example, modal windows can be used to display detailed information or images of car damage, tooltips can provide additional context, and validation plugins can ensure the correctness of user inputs.

By utilizing Bootstrap in the development of the user interface, an automated car damage assessment system can have a modern and responsive design, improving the overall user experience and making it easier for users to interact with the system.

## Libraries

* + - 1. **Pickle**

The pickle library in Python is a module that provides a way to serialize and deserialize Python objects. Pickling is the process of converting a Python object into a byte stream that can be stored or transmitted. Unpickling is the reverse process of converting the byte stream back into the original Python object.

The pickle module supports a wide range of Python data types, including lists, dictionaries, sets, classes, functions, and instances. It can also handle circular references and shared references, making it a robust and flexible serialization solution.

## Pandas:

Pandas is an open-source Python Library providing high-performance data manipulation analysis tool using its powerful data structures. The name Pandas is derived from the word panel Data — an Econometrics from Multidimensional data.

Prior to Pandas, Python was majorly used for data munging and preparation. It had very contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyse.

Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

## NumPy:

NumPy is a powerful Python library for numerical computing. It stands for "Numerical Python" and provides support for efficient manipulation of large arrays and matrices of numeric data. NumPy serves as a fundamental building block for various scientific and data analysis libraries in Python.

One of the key features of NumPy is its multidimensional array object called ndarray. This data structure allows for efficient storage and manipulation of large datasets, enabling high-performance mathematical operations on arrays. NumPy provides a wide range of mathematical functions and operations that can be applied element-wise or across the entire array.

## Matplotlib:

Matplotlib is a popular Python library for creating static, animated, and interactive visualizations. It provides a wide range of functions and tools for generating various types of plots, graphs, charts, and images. With Matplotlib, you can create line plots, scatter plots, bar plots, histograms, pie charts, 3D plots, and much more.

## Sklearn:

Scikit-learn, also known as sklearn, is a widely used Python library for machine learning and data mining tasks. It provides a comprehensive set of tools and functionalities for building and applying machine learning models to analyze and predict patterns in data.

Scikit-learn is widely used in various domains, including academia, industry, and research, for tasks such as classification, regression, clustering, and dimensionality reduction. It offers a powerful and flexible framework for applying machine learning techniques to analyze and extract insights from data

## Seaborn:

Seaborn is a Python data visualization library built on top of Matplotlib. It provides a high- level interface for creating aesthetically pleasing and informative statistical graphics. Seaborn is particularly useful for visualizing and exploring complex datasets, as it simplifies the process of creating visually appealing plots with minimal code.

Seaborn simplifies the creation of visually appealing and informative statistical graphics in Python. It is widely used in data analysis, exploratory data visualization, and communication of insights. By leveraging Seaborn's capabilities, you can create publication-quality plots with minimal effort, allowing for effective data exploration and presentation.

## IPython:

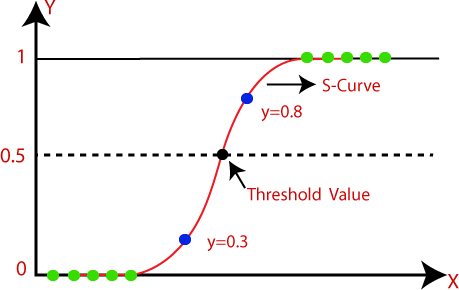
IPython is an interactive command-line interface and a powerful interactive computing environment for Python. It provides an enhanced interactive shell that offers features beyond the standard Python shell, making it more efficient and convenient for interactive programming, data exploration, and scientific computing tasks.

IPython has gained popularity among data scientists, researchers, and developers for its interactive and exploratory capabilities. It offers a rich set of features that enhance productivity, code readability, and collaboration, making it a powerful tool for interactive computing and data exploration in the Python ecosystem.

## ALGORITHMS

### LOGISTIC REGRESSION

* + - 1. Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
      2. Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.
      3. Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.
      4. In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
      5. The curve from the logistic function indicates the likelihood of something such as whetherthe cells are cancerous or not, a mouse is obese or not based on its weight, etc.
      6. Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
      7. Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification.

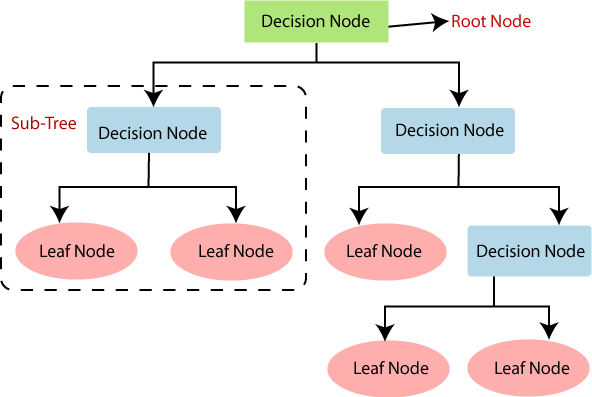


***Fig.5.3.1.1 Graph representing logistic function***

### DECISION TREE ALGORITHM

* + - 1. Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.
      2. In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

The decisions or the test are performed on the basis of features of the given dataset.

* + - 1. It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.
      2. It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
      3. In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.
      4. A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.

***Fig 5.3.2.1 Flowchart of Decision Tree Algorithm***

## SOURCE CODE

### #Car\_check

{

"cells": [

{

"cell\_type": "markdown", "metadata": {},

"source": [

"# Pipe1 : \n",

"## To check if user input image is a car or not. Tested against multiple cnn frameworks to find one with maximum accuracy.\n",

"\n",

"### Data0 - dataset of all images of cars(damaged and undamaged ones)"

]

},

{

"cell\_type": "code", "execution\_count": 1, "metadata": {},

"outputs": [], "source": [ "import os\n", "import h5py\n",

"import urllib.request\n", "import json\n",

"import numpy as np\n", "import pandas as pd\n",

"import matplotlib.pyplot as plt\n", "import seaborn as sns\n",

"import pickle as pk\n",

"from IPython.display import Image, display, clear\_output\n", "from collections import Counter, defaultdict\n",

"from sklearn.metrics import classification\_report, confusion\_matrix"

]

},

{

"cell\_type": "code", "execution\_count": 2, "metadata": {}, "outputs": [

{

"name": "stderr", "output\_type": "stream", "text": [

"Using TensorFlow backend.\n"

]

}

],

"source": [

"from keras import optimizers\n",

"from keras.applications.vgg16 import VGG16\n", "from keras.applications.vgg19 import VGG19\n", "from keras.applications.resnet50 import ResNet50\n",

"from keras.applications.inception\_v3 import InceptionV3\n", "from keras.applications.xception import Xception\n",

"from keras.applications.imagenet\_utils import preprocess\_input, decode\_predictions\n", "from keras.utils.data\_utils import get\_file\n",

"from keras.utils.np\_utils import to\_categorical\n",

"from keras.preprocessing.image import ImageDataGenerator, array\_to\_img, img\_to\_array, load\_img\n",

"from keras.models import Sequential, load\_model\n",

"from keras.layers import Convolution2D, MaxPooling2D, ZeroPadding2D, Activation, Dropout, Flatten, Dense\n",

"from keras.callbacks import ModelCheckpoint, History"

]

},

### #Damaged\_or\_Not

{

"cells": [

{

"cell\_type": "code", "execution\_count": 1, "metadata": {},

"outputs": [], "source": [ "import os\n", "import h5py\n",

"import numpy as np\n", "import json\n",

"import urllib.request\n", "import numpy as np\n",

"import matplotlib.pyplot as plt\n", "import pandas as pd\n",

"import seaborn as sns\n",

"from IPython.display import Image, display, clear\_output\n",

"from sklearn.metrics import classification\_report, confusion\_matrix\n", "\n",

"%matplotlib inline\n", "sns.set\_style('whitegrid')"

]

},

{

"cell\_type": "code", "execution\_count": 2, "metadata": {}, "outputs": [

{

"name": "stderr", "output\_type": "stream", "text": [

"Using TensorFlow backend.\n"

]

}

],

"source": [

"from keras import optimizers\n",

"from keras.applications.vgg16 import VGG16\n",

"from keras.models import Sequential, load\_model, Model\n",

"from keras.layers import Conv2D, MaxPooling2D, ZeroPadding2D, Activation, Dropout, Flatten, Dense, Input\n",

"from keras.regularizers import l2, l1\n",

"from keras.utils.np\_utils import to\_categorical\n",

"from keras.preprocessing.image import ImageDataGenerator, array\_to\_img, img\_to\_array, load\_img\n",

"from keras.callbacks import ModelCheckpoint, History\n", "from keras import backend as K\n",

"from keras.utils.data\_utils import get\_file"

]

}

### #Damage\_location

{

"cells": [

{

"cell\_type": "markdown", "metadata": {},

"source": [

"# Pipe3: \n",

"## To classify location of damage - front, rear or behind."

]

},

{

"cell\_type": "code", "execution\_count": 1, "metadata": {},

"outputs": [], "source": [

"import os\n",

"import urllib.request\n", "import h5py\n", "import json\n",

"import numpy as np\n", "import pandas as pd\n", "import seaborn as sns\n",

"import matplotlib.pyplot as plt\n", "from collections import Counter \n",

"from IPython.display import Image, display, clear\_output\n",

"from sklearn.metrics import classification\_report, confusion\_matrix\n", "\n",

"%matplotlib inline\n", "sns.set\_style('whitegrid')"

]

},

{

"cell\_type": "code", "execution\_count": 2, "metadata": {}, "outputs": [

{

"name": "stderr", "output\_type": "stream", "text": [

"Using TensorFlow backend.\n"

]

}

### #Damage\_severity

{

"cells": [

{

"cell\_type": "code", "execution\_count": 1,

"metadata": {},

"outputs": [], "source": [ "import os\n",

"import urllib.request\n", "import h5py\n", "import json\n",

"import numpy as np\n", "import pandas as pd\n", "import seaborn as sns\n",

"import matplotlib.pyplot as plt\n", "from collections import Counter \n",

"from IPython.display import Image, display, clear\_output\n",

"from sklearn.metrics import classification\_report, confusion\_matrix\n", "\n",

"%matplotlib inline\n", "sns.set\_style('whitegrid')\n", "sns.set\_palette(\"cubehelix\")"

]

},

{

"cell\_type": "code", "execution\_count": 2, "metadata": {}, "outputs": [

{

"name": "stderr", "output\_type": "stream", "text": [

"Using TensorFlow backend.\n"

]

}

],

"source": [

"from keras import optimizers\n",

"from keras.models import Sequential, load\_model, Model\n",

"from keras.layers import Convolution2D, MaxPooling2D, ZeroPadding2D, Activation, Dense, Dropout, Flatten\n",

"from keras.applications.vgg16 import VGG16\n",

"from keras.callbacks import ModelCheckpoint, History\n", "from keras.utils.np\_utils import to\_categorical\n",

"from keras.preprocessing.image import ImageDataGenerator, array\_to\_img, img\_to\_array, load\_img\n",

"from keras.regularizers import l2, l1"

]

}

## OUTPUT SCREENSHOT:

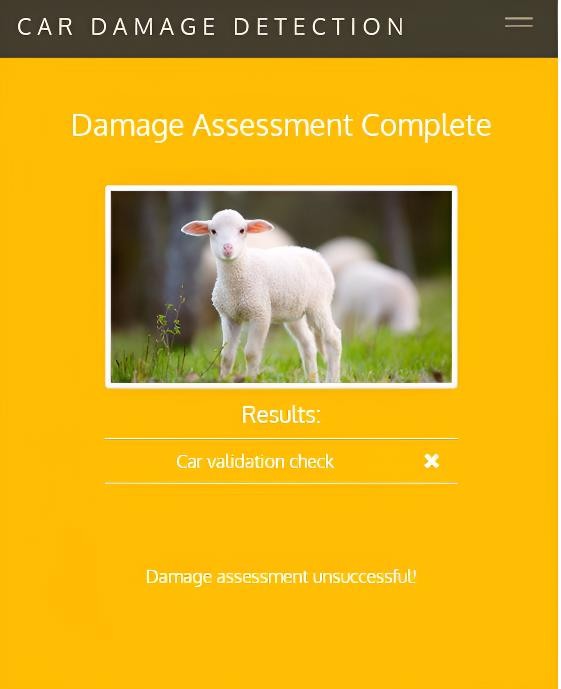
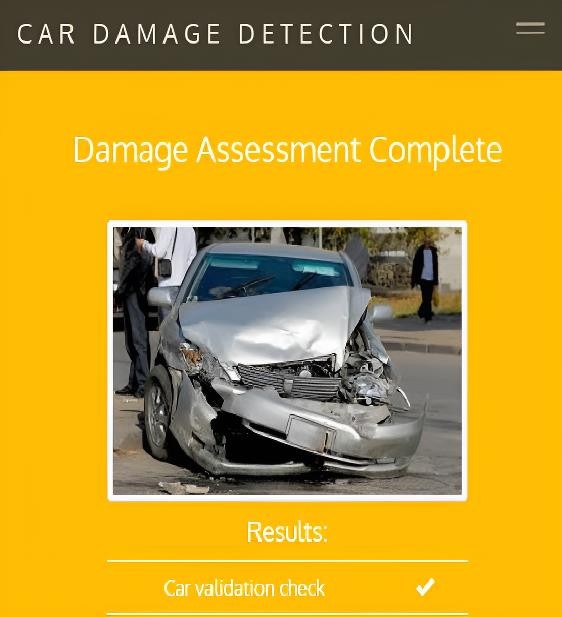
The system takes input as damaged car image. To identify the location and severity of the damage Input car



***Fig 5.5.1: Input Image***

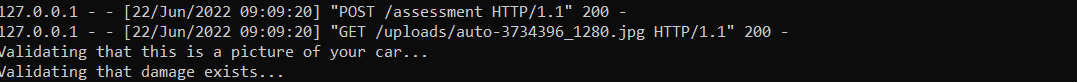
Once the image is loaded then we check that image contains a car or not. If the image contains a car then print the validation this is a picture of your car otherwise, they does not print any result. Shown in the below diagram.

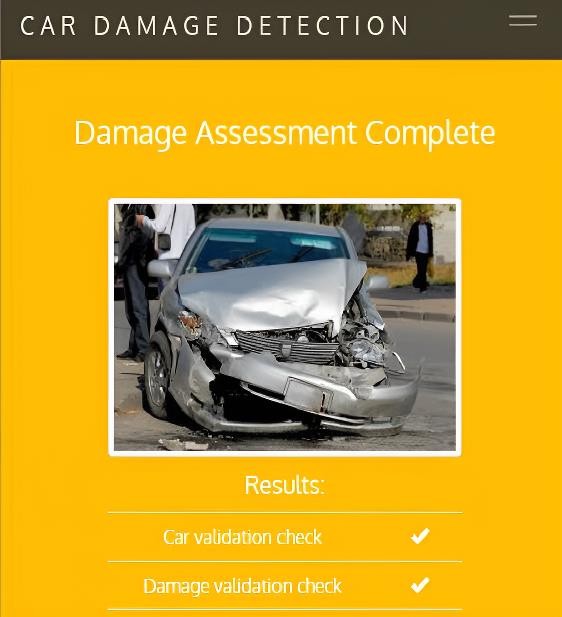




***Fig 5.5.2 Successful car validation Fig 5.5.3 Unsuccessful car validation***

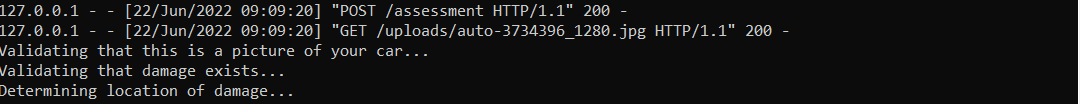
Next have to analyze the damage of the car. Car damage as to be analyze by applying the neural network and check for validation then print the damage validation check is true. That is show in below diagram

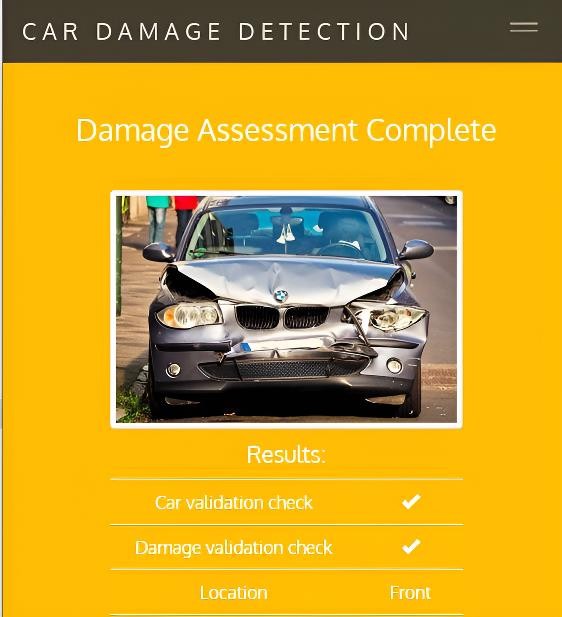




***Fig 5.5.4: Successful damage validation***

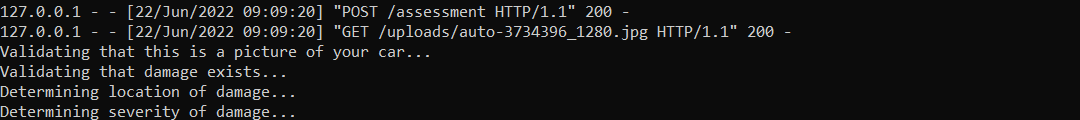
Once the damage validation is check successfully then we have to predict the damage location such as front of the car, rear of the car and side of the car. Shown in the below diagram.

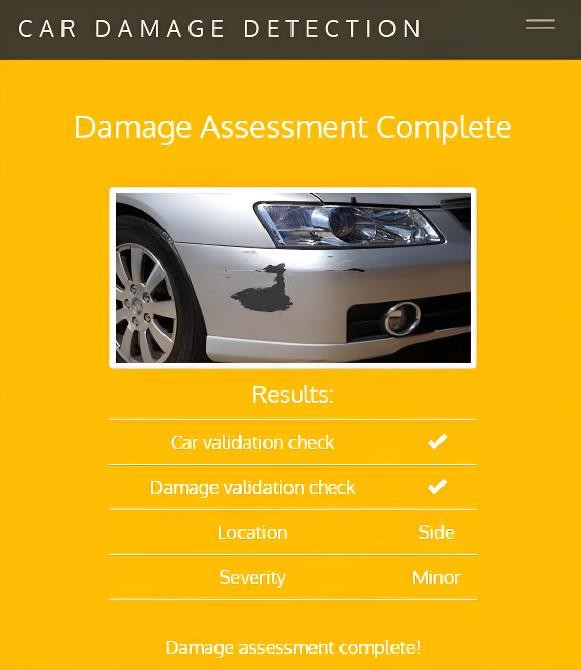




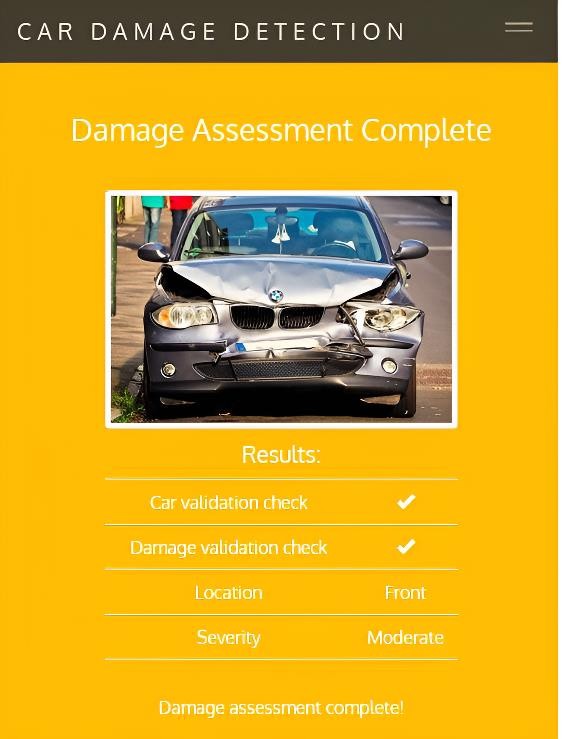
***Fig 5.5.5: Prediction of front location***

After completing the location of the damage prediction then we have to check for severity of the damaged car namely minor, moderate or severe. Shown in the below diagram.

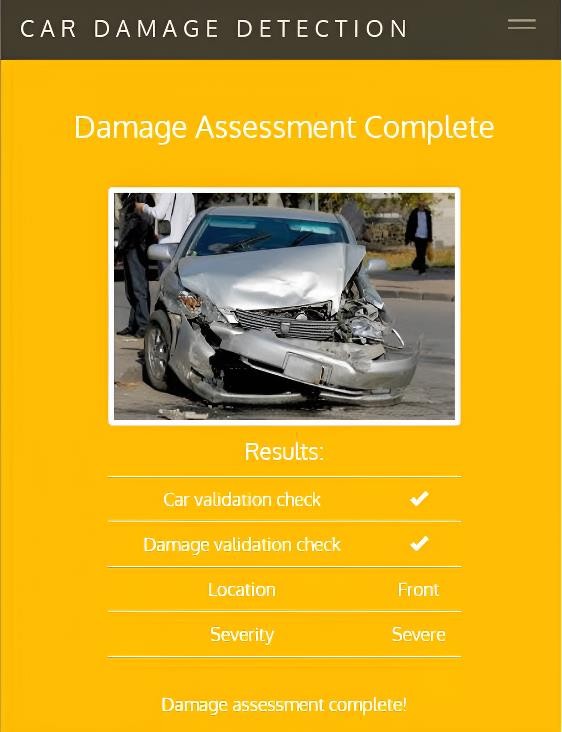




***Fig 5.5.6: Damage assessment complete in minor severity***



***Fig 5.5.7: Damage assessment complete in moderate severity***



***Fig 5.5.8: Damage assessment complete severe severity***

# CHAPTER - 6

**TESTING**

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software Testing also provides an objective, independent view of the software to allow the business to appreciate and understand the risks at implementation of the software. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs.

Software Testing can also be stated as the process of validating and verifying that a software program/application/product:

* Meets the business and technical requirements that guided its design and Development.
* Works as expected and can be implemented with the same characteristics.

## TESTING STRATEGIES

Software level testing can be majorly classified into 4 levels:

1. Unit Testing
2. Integration Testing
3. System Testing
4. Acceptance Testing

## Unit Testing:

Unit Testing is a software testing technique by means of which individual units of software

i.e. group of computer program modules, usage procedures and operating are tested to determine whether they are suitable for use or not. It is a testing method using which every independent module is tested to determine if there are any issue by the developer himself. It is correlated with functional correctness of the independent modules.

Unit Testing is defined as a type of software testing where individual components of software are tested. Unit Testing of software product is carried out during the development of an application. An individual component may be either an individual function or a procedure. Unit Testing is typically performed by the developer.

Unit testing focuses on the building blocks of the software system, that is, objects and subsystems. They are three motivations behind focusing on components. First unit testing reduces the complexity of the overall test activities, allowing us to focus on smaller units of the system. Unit testing makes it easier to pinpoint and correct faults given that few computers are involved in this test.

## Integration Testing:

Integration testing is the second level of the software testing process comes after unit testing. In this testing, units or individual components of the software are tested in a group. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units.

Unit testing uses modules for testing purpose, and these modules are combined and tested in integration testing. The Software is developed with a number of software modules that are coded by different coders or programmers. The goal of integration testing is to check the correctness of communication among all the modules.

In integration testing, testers test the interfaces between the different modules. These modules combine together to form a bigger component or the system. Hence, it becomes very crucial to validate their behaviour when they work together. Apart from the interfaces, they also test the integrated components. Integration testing is the next level of testing after unit testing. Testers do it after completion of the unit testing phase. Integration testing, techniques can be a white box or black box depending on the project requirements.

Integration testing reduces the risk of finding the defects in integrated components in the System testing phase. Integration defects can be complex to fix and they can be time- consuming as well. Finding them early in the cycle eliminates the risk of making too many changes at the System testing phase. As each of the integrating components has been tested in the integration phase, the System testing can focus on end to end and user- specific flows.

## System Testing:

System Testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements. In other words, System Testing means testing the system as a whole. All the modules/components are

integrated in order to verify if the system works as expected or not. System Testing is done after Integration Testing. This plays an important role in delivering a high-quality product.

The purpose of a system test is to evaluate the end-to-end system specifications, Usually, the software is only one element of a larger computer-based system. Ultimately, the software is interfaced with other systems. System Testing is actually a series of different tests whose sole purpose is to exercise the full computer-based system.

In system testing, integration testing passed components are taken as input. The goal of testing is to detect any irregularity between the units that are integrated together. testing detects within both the integrated units and the whole system. The result of testing is the observed behaviour of a component or a system when it is tested

## 6.1.4 Acceptance Testing:

Acceptance Testing is method of software testing where a system tested for acceptability. The major aim of this test is to evaluate the compliance of the system with the business requirements and assess whether it is acceptable for delivery or not.

The Standard definition if Acceptance testing is given as, “It is a formal testing according to user needs, requirements and business processes conducted to determine whether a system satisfies the acceptance criteria or not and to enable the users, customers or other authorized entities to determine whether to accept the system or not”.

Acceptance Testing is the last phase of software testing performed after System Testing And before making the system available for actual use.

The main goal behind acceptance testing is to check whether the developed software product passes The acceptance norms defend on the basis of user and business Requirements, so as to declare it acceptable or non-acceptable for its use by the users.

Acceptance testing is one of the last types of software testing performed over a software or application. It is conducted by a pool of targeted users to ensure the readiness and quality of system from user's perspective, which allows the team to meet their needs and expectations.

# TEST CASE DESIGN

The design of tests for software and other engineering products can be as challenging as The initial design of the product. Test case methods provide the developer with a systematic approach to testing, Moreover, these methods provide a mechanism that can help to ensure the completeness of tests and provide the highest like hood for uncovering errors in software.

Any Engineered product can be tested in either of the two ways:

1. Knowing the specified function that a product has been designed to perform, tests can be conducted. These tests demonstrate whether each function is full operational and at the same time searches for errors in each function.
2. Knowing the internal workings of a product, tests can be conducted to ensure that are performed according to specifications and all internal components hence been adequately exercised.

Test case design methods are divided into two types:

1. White-box testing
2. Black-box testing

## White-Box Testing

White testing, sometimes called glass-box testing is a test, case designed method that uses the control structure of the procedural design to derive test cases. Using white-box testing methods, the s/w engineer can derive test cases that that all independent paths within a modulehave been exercised at once. Exercise all logical decisions on their true and false sides. Execute all loops at their boundaries and within their operational bonds.

Exercise internal data structures to ensure their validity. Basis path testing is a white-box testing technique. The basis path method enables the test case designer to derive a logical complexity measure of a procedural design and use this measure as a guide for defining a basis set are guaranteed to exercise every statement in the program at least one time during testing.

## Black Box Testing:

Black box testing is a type of software testing that focuses on the external behaviour of the software system or application, without knowledge of its internal workings. In other words, it is a testing method where the tester does not have access to the source code, design documents, or internal structures of the system being tested.

In black box testing, the tester treats the system as a "black box" and only focuses on the inputs and outputs of the system, without any knowledge of how the system processes or manipulates the data. The main objective of black box testing is to ensure that the system or application meets the functional and non-functional requirements specified in the projectdocumentation.

Black box testing can be applied at different levels of testing, including unit testing, integration testing, system testing, and acceptance testing. Some common techniques used in black box testing include equivalence partitioning, boundary value analysis, decision table testing, state transition testing, and use case testing.

## TEST CASES

**Table 6.1: Test Case for Excel Sheet Verification:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case#** | **TEST TITLE** | **Test Steps** | **Expected Result** | **Actual Result** | **Status** |
| Test Case\_1 | Input Image validation | 1.Input car Image | Accepted | Input Accepted | Pass |
| 2.Run the code |
| 3.Click on submit |
| Test Case\_2 | Input Image validation | 1.Input animal image | Not accepted | Input not accepted | Fail |
| 2. Run the code |
| 3.Click on submit |
| Test Case\_3 | Damage Detection | 1.Check for damage | Damage Detected | Input accepted | Pass |
| 2.Run the code |
| 3.Click on submit |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case\_4 | Damage type | 1.Check for Dents | Dent detected | Input accepted | Pass |
| 2.Run the code |
| 3.Click on submit |
| Test Case\_5 | Damage type | 1.Check for scratches | Scratches Detected | Input accepted | Pass |
| 2.Run the code |
| 3.Enter Residencetype  field |
| Test Case\_6 | Damage type | 1.Check for glass  breakage | Glass breakage detected | Input accepted | Pass |
| 2.click on  'Glucoselevel' |
| Test Case\_7 | Severity Estimation | 1.Check for severity | Minor severity | Input accepted | Pass |
| 2.Run the code |
| 3.Click on submit |
| Test Case\_8 | Severity Estimation | 1.Check for severity | Moderate severity | Input accepted | Pass |
| 2.Run the code |
| 3.Click on  submit |
| Test Case\_9 | Severity Estimation | 1.Check for  severity | Severe severity | Accepted | Pass |
| 2.Run the code |
| 3.Click on submit |

# CHAPTER - 7 CONCLUSION AND FUTURE SCOPE

## CONCLUSION

In conclusion, automated car damage assessment is a valuable technology that streamlines and enhances the process of evaluating car damage. By leveraging computer vision algorithms, machine learning techniques, and data analysis, automated car damage assessment systems can accurately detect, classify, and assess various types of damage on vehicles.

The purpose of automated car damage assessment is to provide an efficient and objective evaluation of car damage, eliminating the need for manual inspection and reducing human error. The aim is to improve the accuracy and speed of the assessment process, enabling faster insurance claims processing, accurate repair cost estimation, and efficient decision-making for vehicle repairs or replacements.

The use of algorithms such as logistic regression, decision trees, and other machine learning techniques enables automated car damage assessment systems to analyze car images, extract relevant features, and make predictions regarding the presence, type, and severity of car damage. These algorithms play a crucial role in accurately classifying the damage and estimating the extent of the damage.

Automated car damage assessment systems offer several benefits, including increased efficiency, reduced costs, improved accuracy, and consistent assessment results. They streamline the assessment process, saving time for both insurers and car owners. Moreover, they provide objective and unbiased evaluations, eliminating subjective judgments that can occur in manual assessments.

While automated car damage assessment systems have made significant advancements, continuous research and development are essential to improve their performance, expand their capabilities, and address challenges such as robustness to varying lighting conditions.

## FUTURE SCOPE

The future scope of automated car damage assessment is promising, with several exciting developments and advancements on the horizon. Here are some potential areas of growth and improvement in this field:

* **Advanced Computer Vision Techniques:** As computer vision techniques continue to evolve, we can expect more advanced algorithms and models specifically tailored for car damage assessment. These techniques may include improved object detection and segmentation algorithms, better handling of occlusions and complex damage patterns, and the ability to accurately assess damage in varying lighting conditions.
* **Integration of Sensor Data:** Automated car damage assessment can benefit from the integration of sensor data, such as data from cameras, LiDAR, radar, or other sensors present in modern vehicles. By combining visual information with data from these sensors, the assessment systems can gain a more comprehensive understanding of the extent and severity of the damage, leading to more accurate assessments.
* **Deep Learning and Neural Networks:** Deep learning techniques, particularly convolutional neural networks (CNNs), have shown great potential in image analysis tasks. Future developments may involve the use of more complex deep learning architectures to improve the accuracy and robustness of car damage assessment. Additionally, transfer learning and pre-training on large-scale datasets could enable models to generalize better and adapt to different types of damage.
* **Real-time Assessment:** With advancements in processing power and algorithms, real-time car damage assessment systems can become a reality. This would enable immediate assessment of damage during the inspection process, providing instant feedback and facilitating faster decision- making for repairs or insurance claims.
* **Integration with Insurance and Repair Systems:** Future developments may involve seamless integration of automated car damage assessment systems with insurance and repair systems. This would enable automatic generation of insurance claims, estimation of repair costs, and integration with repair shop workflows, streamlining the entire process and reducing administrative overhead.
* **Improved User Interfaces and Accessibility:** User interfaces of automated car damage assessment systems can be further enhanced to be more user-friendly and accessible. This includes intuitive visual interfaces, mobile applications, and the ability to capture and analyze car damage using smartphones or other consumer devices.

# CHAPTER - 8 REFERENCES

## Paper references:

1. Paper: "Deep Learning-Based Car Damage Classification and Assessment" Authors: S. Li, X. Zhang, X. Xie, et al.

Published in: IEEE Transactions on Intelligent Transportation Systems, 2019 Link: <https://ieeexplore.ieee.org/document/8594445>

1. Paper: "Automated Car Damage Assessment Using Convolutional Neural Networks" Authors: A. M. Javed, S. Tahir, M. Y. Javed, et al.

Published in: Proceedings of the International Conference on Big Data, IoT, and Data Science, 2018 Link: <https://ieeexplore.ieee.org/document/8730589>

1. Paper: "Car Damage Detection and Classification using Deep Convolutional Neural Networks" Authors: M. A. Hossain, S. Ahmed, A. A. A. Mamun, et al.

Published in: Proceedings of the International Conference on Computer, Communication, Chemical, Material and Electronic Engineering, 2017

Link: <https://ieeexplore.ieee.org/document/8620925>

## Text Books:

1. "Computer Vision: Algorithms and Applications" by Richard Szeliski
2. "Pattern Recognition and Machine Learning" by Christopher M. Bishop
3. "Deep Learning" by Ian Goodfellow, Yoshua Bengio, and Aaron Courville

## Websites:

1. Verisk: Verisk is a leading data analytics company that offers solutions for automated car damage assessment. Their website provides information on their offerings, including AI-powered tools for accurate and efficient car damage assessment. Visit their website at: <https://www.verisk.com/insurance/claims/car-damage-estimation/>
2. Audatex: Audatex provides automated solutions for vehicle damage assessment and estimating. Their website offers information about their technology and services, including AI-driven solutions for faster and more accurate damage assessment. <https://www.audatex.us/>