T.C.

MARMARA UNIVERSITY

FACULTY of ENGINEERING

COMPUTER ENGINEERING DEPARTMENT

CSE4197 Engineering Project I

CSE 4197 - Analysis and Design Document

Title of the Project

DAMAGEWIZ: COST CALCULATION OF CARS DAMAGED USING ARTIFICIAL INTELLIGENCE

January 2, 2023

Group Members

150120531 - Hasan Şenyurt

150119761 - Cem Anaral

150119727 - Melisa Durmuş

Supervised by

Assoc. Prof. Mustafa AĞAOĞLU

1. INTRODUCTION

1.1 Problem Description and Motivation

People in today's world struggle greatly with traffic accidents. Our project focuses on the topic of damage assessment of the accident-related cars. The car may need to be towed to an expert for damage assessment, which adds time to the damage assesment process. If not, the car owner must somehow bring the vehicle to an expert to estimate the cost of the damaged parts. These processes are costly, time-consuming, and prone to human mistake. With the application we will develop, we hope to address this problem with the use of images of damaged cars.

The cost of predicting damaged parts is a concern in addition to the loss of time during damage assessment. In various mechanic shops, the same car parts are priced differently. Drivers thus struggle to obtain the best or most affordable car parts. In our project, the user will be shown how much car parts cost in mechanics.

The problem we have mentioned is worth solving because doing so would cut down on wasted time and automate the labor-intensive damage assessment process.

1.2 Scope of the Project

Two essential aspects of this project need to be elaborated in order to understand the scope.

First aspect is about drivers. Features of our project for drivers include identifying the brand and model of the cars as well as damage assessment using four images that the driver took after the accident or that they had previously stored on their mobile devices. Left, right, back, and front of the car must all be captured on camera. Same images will be used for brand and damage detection.

Application will first try to identify the brand of the car. The driver will be able to verify if the car brand is correct or not after detection by viewing the detected brand name.

Drivers must manually enter the car brand if the detected brand is wrong. If the brand detected is correct, the application will try to detect the model and the same process will also be performed for model detection.

Brand and model are significant elements because each part of a car differs depending on the brand and model. Following brand and damage identification, the driver will be shown the required replacement parts in the order of lowest overall cost of damaged parts. The application will look for the least expensive replacement parts for each damaged car part. The application will include a list of the total cost of the damaged parts in mechanic shops.

Second aspect is about mechanic shops. A marketplace will be developed for mechanic shops. They will be able to upload their car parts along with their names, prices, and descriptions. These mechanic shops will be pinned on the map of our application. Mechanic shops will be sorted based on their locations or prices by the driver's choice.

Only automobiles are in our scope. Vehicles except automobiles such as trucks, motorcycles, bikes etc are not in our concern.

1.3 Definitions, acronyms, and abbreviations

- **GUI:** Graphical User Interface program.
- **CNN:** Convolutional Neural Network
- **RPN:** Region Proposal Network
- **RoI:** Region of Interest
- **API:** Application Programming Interface
- **RELU:** Rectified Linear Unit
- **UML:** Unified Modelling Language
- **GPS:** Global Positioning System

- **R-CNN:** Region-based Convolutional Neural Networks
- **YOLO:** You Look Only Once
- **Google Colab:** It is a product offered by Google Research. Especially suitable for machine learning, data analysis and education.
- **F1-score:** In the statistical analysis of binary classification, the F score or F measure is a measure of the accuracy of a test.
- **Machine Learning:** Techniques that let computers gain knowledge from the past.
- **Deep Learning:** Deep learning is a type of machine learning that uses a network of neurons (called a "deep neural network"). These networks can learn more complex tasks than traditional machine learning networks.

2. LITERATURE SURVEY

We searched for some resources that are similar to our project. These resources are about determining if the car is damaged, determining the make and model, and determining where the damage is.

2.1 Vehicle Brand Detection

By using a classification method based on deep neural networks, Kunduraci and Kahramanli [1] have suggested a method for detecting car brands. They discovered that the Faster R-CNN approach, which is based on deep neural networks, had an accuracy rate of 67.66%. Two submodules comprise faster R-CNN. A deep fully convolutional network serves as the first module, proposing regions, and the second module makes use of these regions. Faster R-CNN is the second module's component. The first module, known as RPN, is where the feature network is obtained. RoI pooling is used to downsize proposed regions.

2.2 CarNet.AI

CarNet.AI [2] is an enterprise-level API created by LastStar Ltd. that can identify car types and brands. They begin by identifying each car in the image and drawing a bounding box around it. The next step is to filter out the bounding boxes that, depending on their dimensions, are not acceptable for detection. The third stage is to decide on a detection plan. There are three different approaches to pick from. They either identify every bounding box, the one closest to the image's center, or just the biggest one. Finding the subclasses is the fourth step. They go through each bounding box that was chosen in the third stage and identify their classes. Only bounding boxes with the class "vehicle" are forwarded to the final stage. The features of the cars are discovered in the final step. The make, model, color, and angle of the car are among these features.

2.3 Vehicle Make and Model Recognition

A two-stage framework was suggested by Tafazzoli and Frigiue [3] to identify the brand and model of autos. They employed multiple instance learning to classify objects with excellent accuracy. To increase the accuracy of the first stage, they just identify the brand of the car from the logo in the second stage rather than both the brand and the model in the first.

A paradigm for supervised learning called multiple instance learning deals with ambiguously labeled data in classification and regression problems. Instead of using fixed-length feature vectors as instances like traditional supervised learning does, multiple instance learning analyzes packets of instances within each packet.

2.4 Damage Identification of Selected Car Parts

A two-level machine learning-based approach has been suggested [4] by them to identify the front bumper, rear bumper, and car wheels as three car parts. The first model is for categorizing auto parts, and the second is for identifying damage. The first model has a training accuracy of 94.84% and a validation accuracy of 81.25%. The training accuracy of the second model is 97.16%, and its validation accuracy is 49.28%.

They used images from Google as their dataset to categorize car parts. For their models, they used convolutional neural networks with six layers (excluding poolings). The following are the 6 layers: Convolution2, Pooling2, Convolution2, Pooling2, Convolution2, Pooling2, RELU, and Softmax are the steps in the process.

The names of the car parts represent the initial model's output (Rear Bumper, Front Bumper, Car Wheel). The names of the damaged and undamaged auto parts are the second model's output (Undamaged Front Bumper, Undamaged Rear Bumper, Undamaged Car Wheel, Damaged Front Bumper, Damaged Rear Bumper, Damaged Car Wheel).

Kunduraci and Kahramanli, just did brand detection. CarNet.Ai, Tafazzoli and Frigiue did both brand detection and model detection. Chua and his friends did only damaged part detection. In our project we will do all of them and also we will make damage percentage detection.

3. PROJECT REQUIREMENTS

3.1 Functional Requirements

3.1.1 Login System

3.1.1.1 Description

Users are of two types: mechanics and drivers. We need to know how it is entered into the system.

3.1.1.2 Input

Information is obtained from users to register in the system.

3.1.1.3 Processing

According to the incoming information, users' profiles are created.

3.1.1.4 Outputs

The system changes shape according to the user type. If the user is a driver, he/she will see a different screen, if mechanical, he/she will see a different screen.

3.1.1.5 Error/Data Handling

If invalid information is entered while registering, the user is warned and asked to enter the information again.

3.1.2 Image Processing

3.1.2.1 Description

Images should be taken from the user to detect damage.

3.1.2.2 Input

Images should be taken from all 4 sides of the car.

3.1.2.3 Processing

Images should be processed by the program we developed and damage detection should be done.

3.1.2.4 Outputs

The user should be shown where the damage is, the percentage, and which mechanics can have it repaired at what price.

3.1.2.5 Error/Data Handling

If the images are blurred, the user is prompted to take the photos again.
If the brand cannot be determined due to damage, the user is prompted to enter the brand manually.

3.1.3 Marketplace

3.1.3.1 Description

A marketplace should be created for mechanics to promote themselves.

3.1.3.2 Input

Mechanics upload their products to the system.

3.1.3.3 Processing

Products are collected in a database. When necessary, product information is provided to the user.

3.1.3.4 Outputs

Which mechanic owns which product and price information is shown to the user over the detected damage.

3.1.3.5 Error/Data Handling

If the desired product is not available in any of the mechanics, the user is informed.

3.1.4 Filtering Mechanism

3.1.4.1 Description

Users should be provided with a filtering feature according to the features they want to filter.

3.1.4.2 Input

The information about which filtering feature he/she wants to use is obtained from the user. eg closest, cheapest mechanics.

3.1.4.3 Processing

Data is retrieved from the database according to the user's request.

3.1.4.4 Outputs

The mechanic shops with the desired feature are listed to the user.

3.1.4.5 Error/Data Handling

There should be no error in this part.

3.1.5 Rating Mechanism

3.1.5.1 Description

Users should be able to look at their ratings to have an idea about the reliability of the mechanic shops they will contact.

3.1.5.2 Input

Users will be able to rate the mechanic shops they have used before.

3.1.5.3 Processing

Information about which mechanic shop got which score is kept.

3.1.5.4 Outputs

Users are shown the information about which mechanic shop is rated and how much.

3.1.5.5 Error/Data Handling

There should be no error in this part.

3.2 Nonfunctional Requirements

3.2.1 Performance

- The photos taken from the user must be processed, the damage must be determined correctly and the user must be informed about the damage.
- While determining the damage, it should be done quickly without waiting for the user.
- Mechanics' products need to be displayed in their profile.
- The filtering and rating mechanisms should work smoothly.

3.2.2 Reliability

- The product information of the mechanics should be kept up to date. A non-existent product should not be displayed in the list.
- damage assessment should be done correctly, wrong assessment should not be presented to the user

3.2.3 Usability

- The user interface should be easy to understand.
- The user is offered the opportunity to be easily accessible from the application.
- The product information of the mechanics should be kept up-to-date and easily viewed by the user.

3.2.4 Security

- User login information must be kept confidential.
- The license plate information in the car photos we receive from users should be kept confidential.

3.2.5 Maintainability

- While determining the make and model of the car, the brand information that was damaged due to the damage cannot be determined from the photo and the user is asked to enter it manually.
- While detecting damage, if blurry photos are taken from the user, the user will be prompted to upload the photos again.
  Errors encountered in this way will find a solution and the program will run smoothly.

3.2.6 Portability

- Users will be able to access the program on a computer with an internet connection.
- It can be used on Android mobile phones with an internet connection.

4. SYSTEM DESIGN

4.1 UML Use Case Diagram

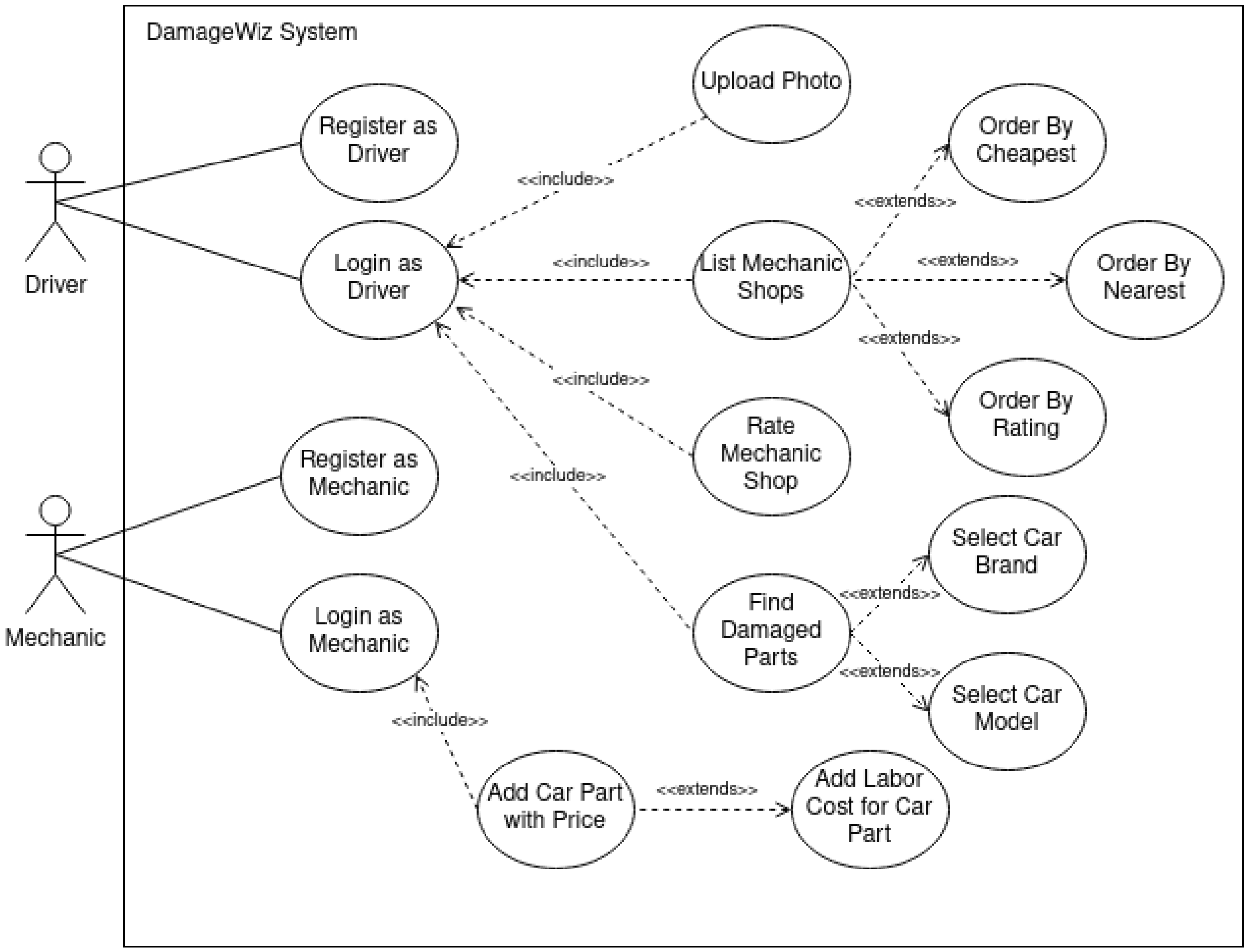

Figure 1: Uml Use Case Diagram

4.2 UML Class Diagram

4.2.1 Marketplace Application

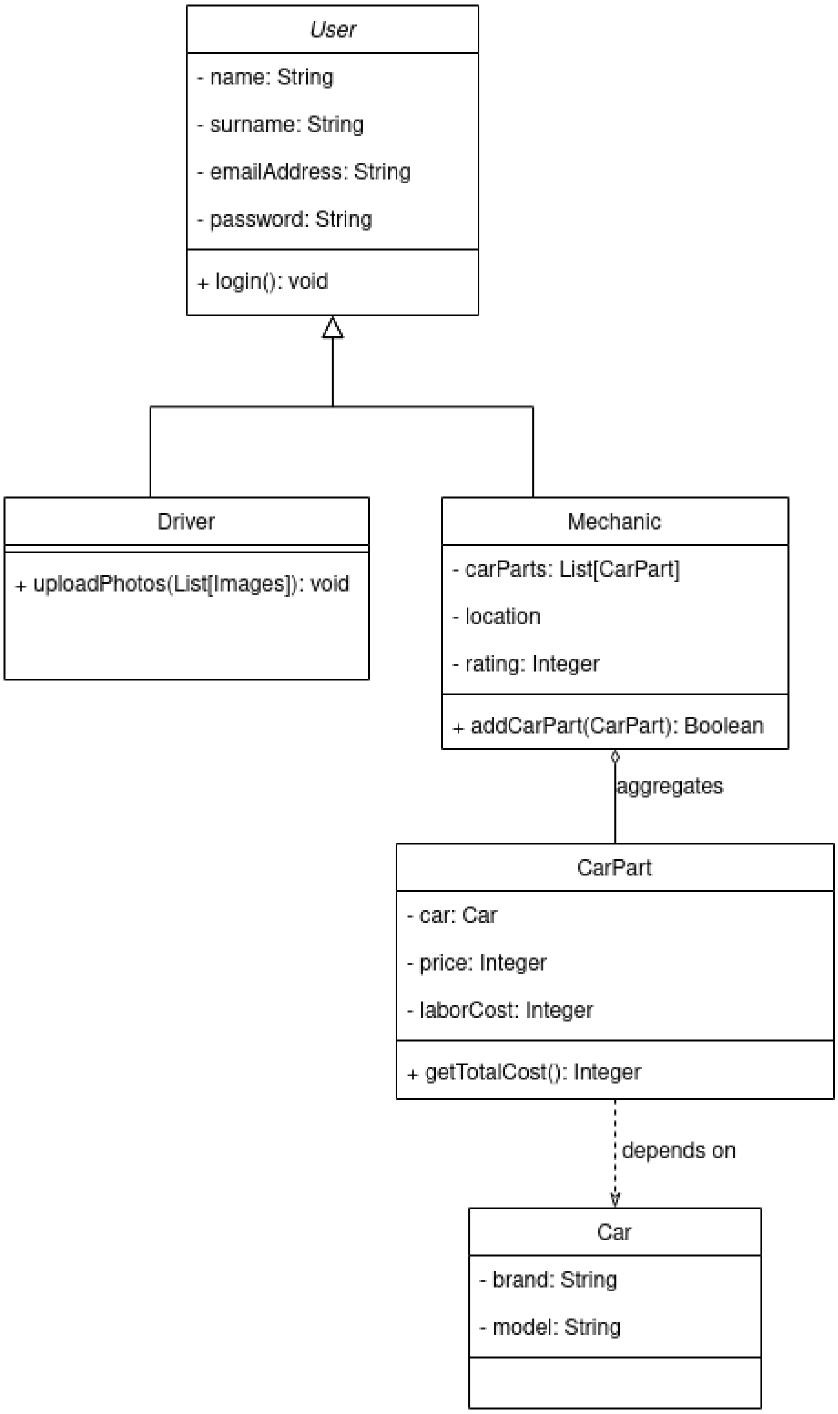

Figure 2: Uml Class Diagram - Marketplace

4.2.2 Detection API

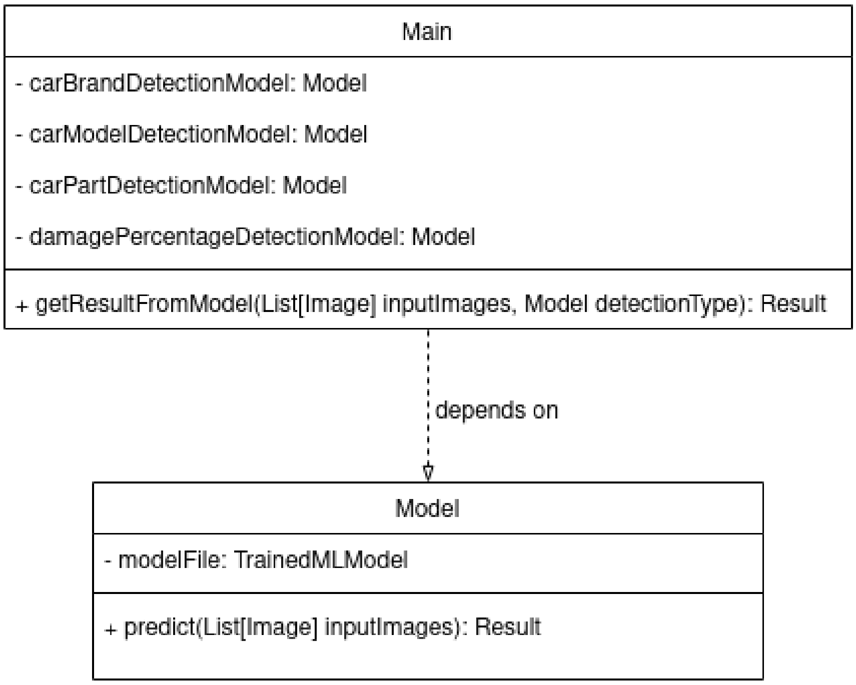

Figure 3: Uml Class Diagram - Detection

4.3 User Interface

4.3.1 Login

Figure 4: Login Screen

4.3.2 Register

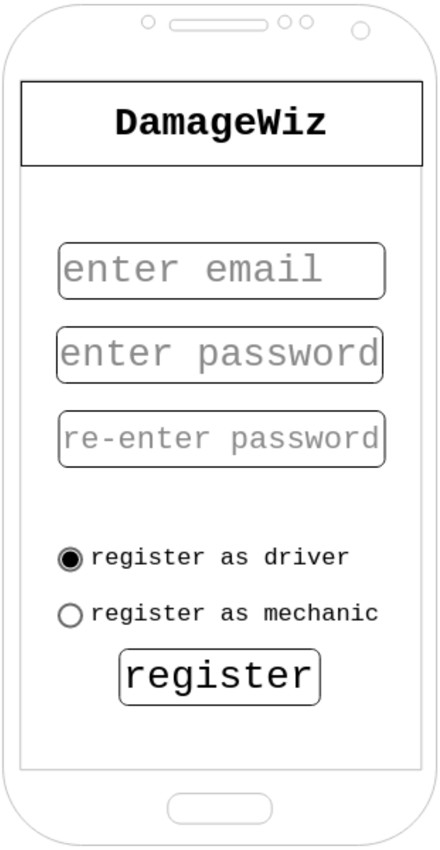

Figure 5: Register Screen

4.3.3 Upload Photos (Driver point of view)

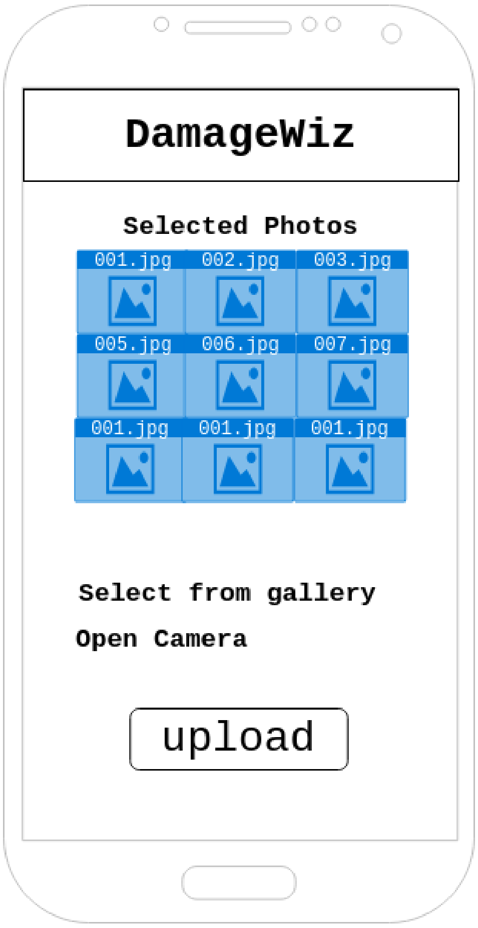

Figure 6: Photo Upload Screen

4.3.4 Confirm Detection Results (Driver Point of View)

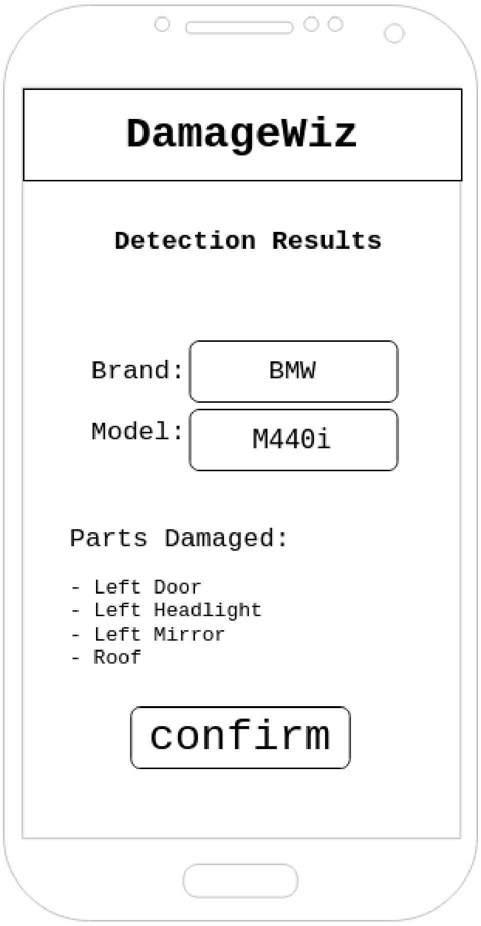

Figure 7: Confirmation Screen

4.3.5 List Mechanic Shops (Driver)

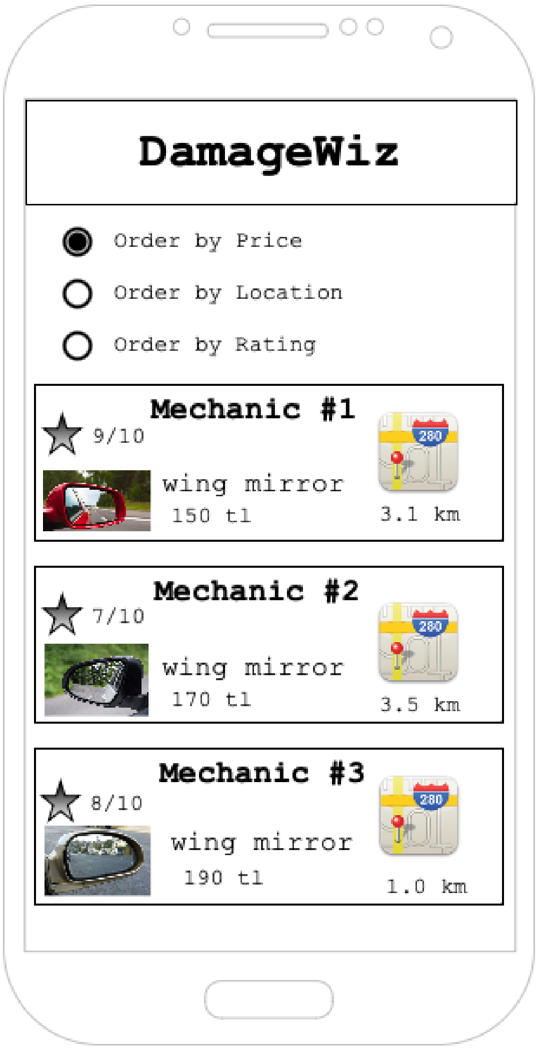

Figure 8: Listing Mechanic Shops

4.3.6 Add Car Parts (Mechanic)

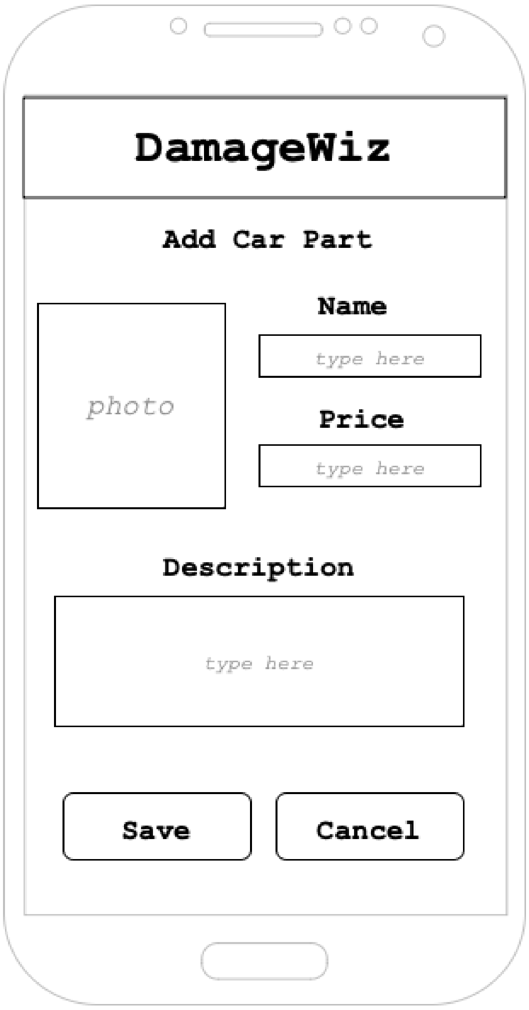

Figure 9: Add Car Part Screen

4.4 Test Plan

Each part of the project will be developed separately, and they will be integrated during the development process. Testing and monitoring all of the project's results. Duration of testing is one month.

- Checking to see if brand detection is functioning properly. We will upload blurry images to see whether our modal can detect the brand of the car. Also, we will send images of cars, which will have brands that are not in our dataset. The reason for doing this is to see if our application will work correctly or not because the manual entry screen has to be shown to the user in this case. Data augmentation can be done and a new proper dataset can be found if results of the test are not good enough. We will use F1-score to compute accuracy.

- Testing of damage assessment and damaged part detection. Damage part detection is the one of the most critical parts of our project because the parts that will be listed on the driver's screen in the application will be fetched from the database according to the detected damaged parts. So, we will test this function by uploading different parts of lots of different cars. If the results of tests are not sufficient, we will label more data to increase accuracy.

- Checking the affected parts' damage percentage. Damage percentage will give drivers an idea about damage. They can decide whether the damaged part must be replaced or repaired according to that value. We will choose cars with different damage size to test damage percentage calculation. If we cannot manage to calculate the percentage correctly enough, we may classificate damage size as minor, moderate and major. Test results will be the best factor to decide that.

- Testing of the product screen that will occur after damaged parts of the car matches with products of mechanic shops. Database queries and different scenarios will be tested. Also, filtering and rating will be tested.

- There will be testing conducted on additional features of the website and mobile application. Frontend issues will be tested by using the application on different mobile phones. GPS feature will be tested by adding temporary mechanic shops to the map.

5. SOFTWARE ARCHITECTURE

5.1 Data Flow Diagram

We have 2 databases in our application. Mechanic database stores mechanics information, such as car parts with prices. User info database stores credentials of drivers and mechanics. Also the user info database contains ratings of mechanics.

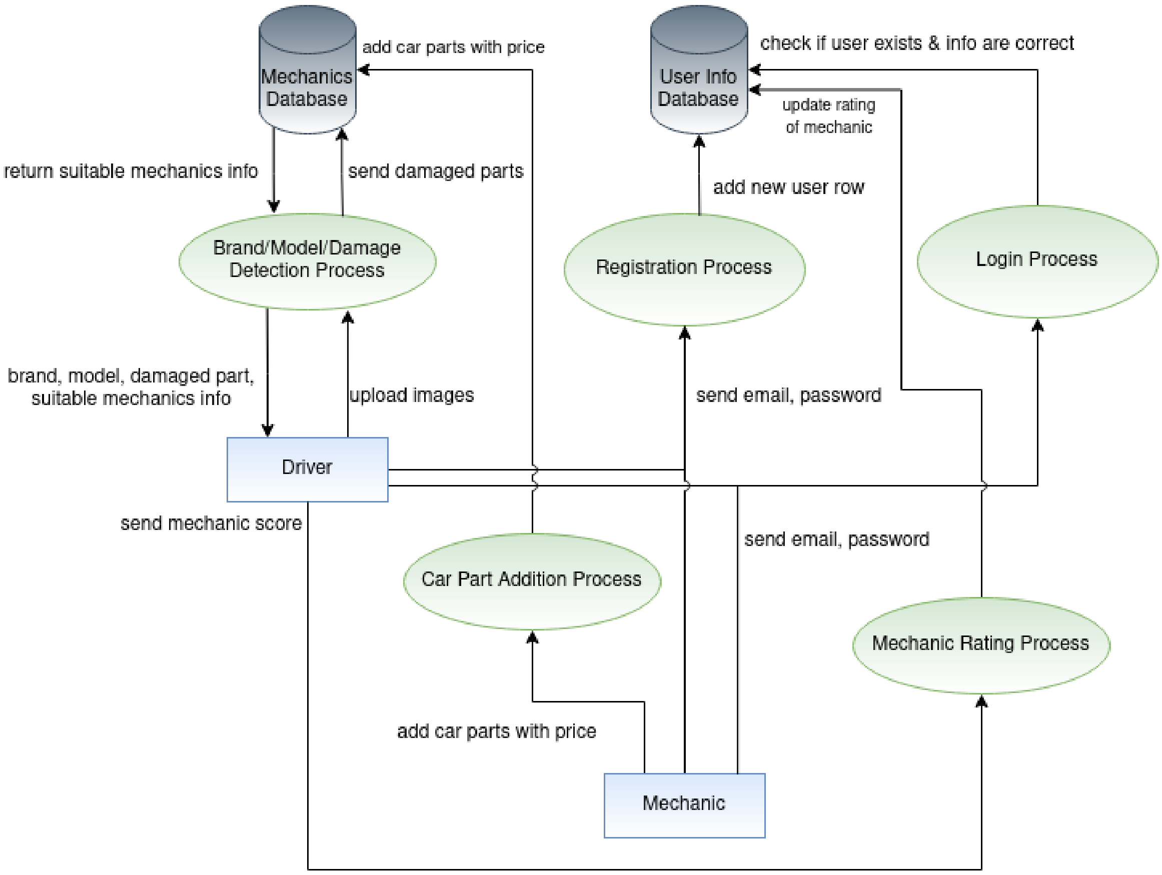

Figure 10: Data Flow Diagram

5.2 Control Flow Diagram

First damaged car photos taken from the driver. Then we confirm that the car brand and model is detectable. If the car brand is not detected from the photos, drivers will be prompted to upload new photos or drivers give brand and model info manually. Afterwards we check if the car is damaged or not. If the car is damaged, we localize the damage and calculate the percentage. If not, then the program stops.

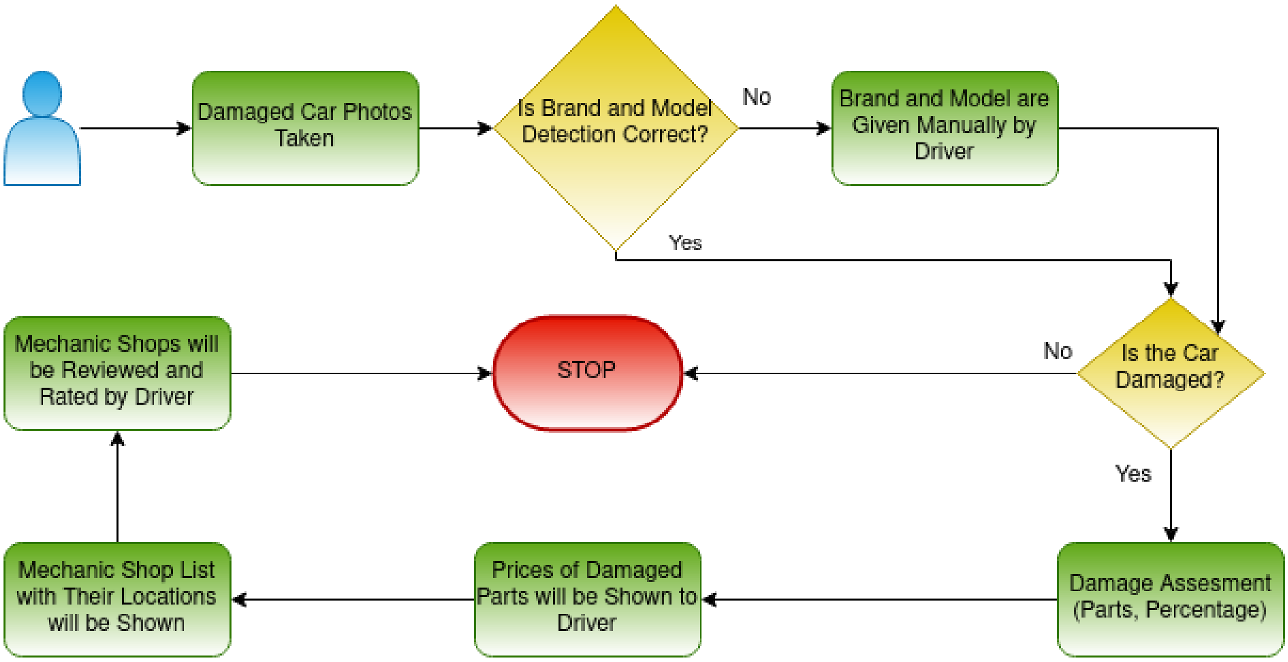

Figure 11: Control Flow Diagram

5.3 Modular Design Diagram

DamageWiz System is composed of three main components: Frontend, Marketplace Backend and Detection Backend.

Frontend component is where we serve UI to drivers and mechanics. Marketplace Backend is used for every action that drivers and mechanics can take. Detections (Brand Detection, Model detection and Damaged Part Detection) are done in the Detection Backend. Marketplace Backend communicates with Detection Backend whenever a detection must be done.

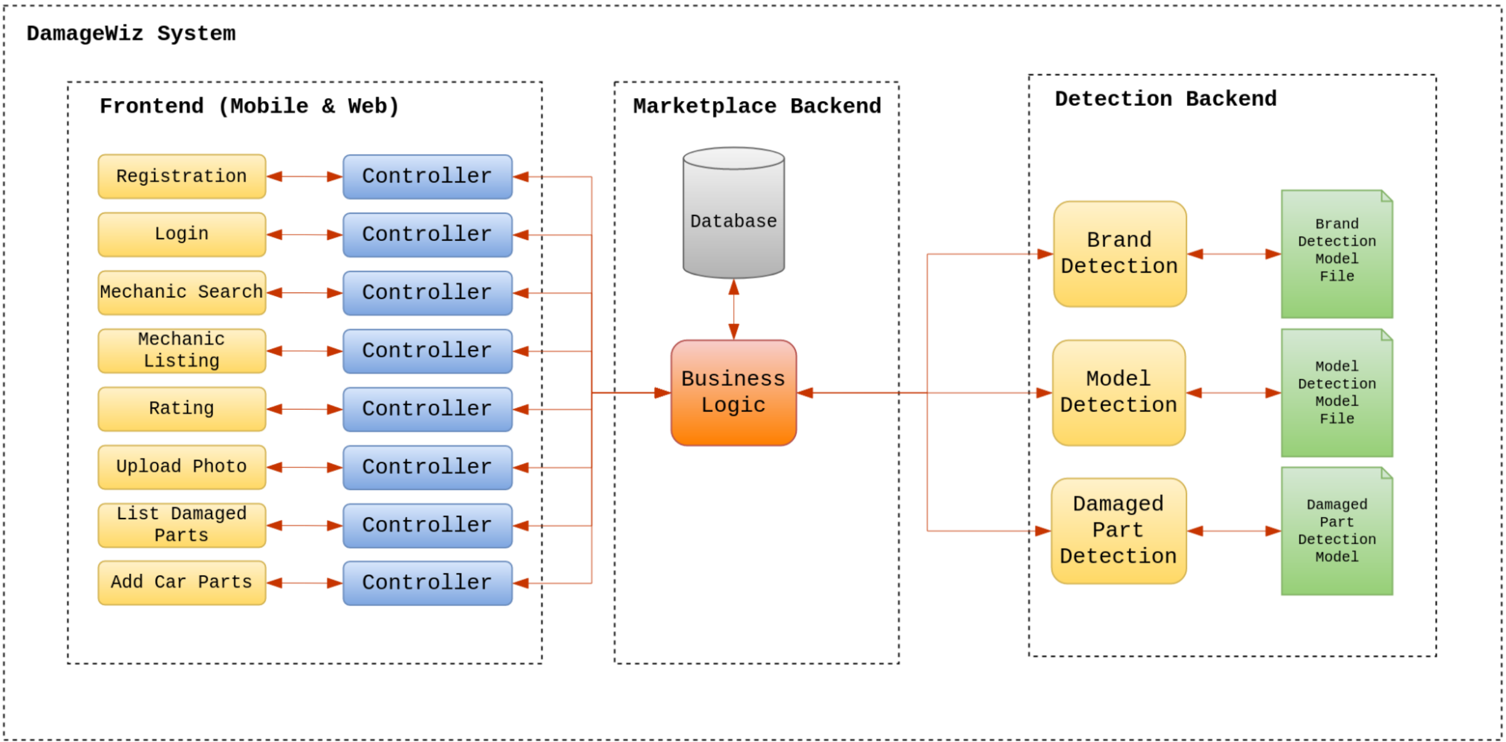

Figure 12: Modular Design Diagram

6. TASKS ACCOMPLISHED

6.1 Current State of the Project

In the current state of the project, these tasks were accomplished:

- Dataset of damaged and undamaged cars are collected from Kaggle [5]. Total dataset includes 2300 photos. 1150 photos for each damaged and undamaged car. We separated data 80% for training, 20% for testing.

- First stage of the project is deciding if the car is damaged or not by using uploaded car image. DenseNet model is trained for this purpose. Model is created in Python by using Google Colab.

- Simple GUI program was developed as the first prototype of the project. Users can upload the car image which can be damaged or undamaged, then our first prototype will show the results by using the DenseNet model. Results include the answer whether the car is damaged or not, and prediction value. If the prediction value is less than 0.5, it means that the car is damaged. Example can be seen in Figure 13. If the prediction value is greater than 0.5, it means that the car is undamaged. Undamaged car example can be seen in Figure 14 as well.

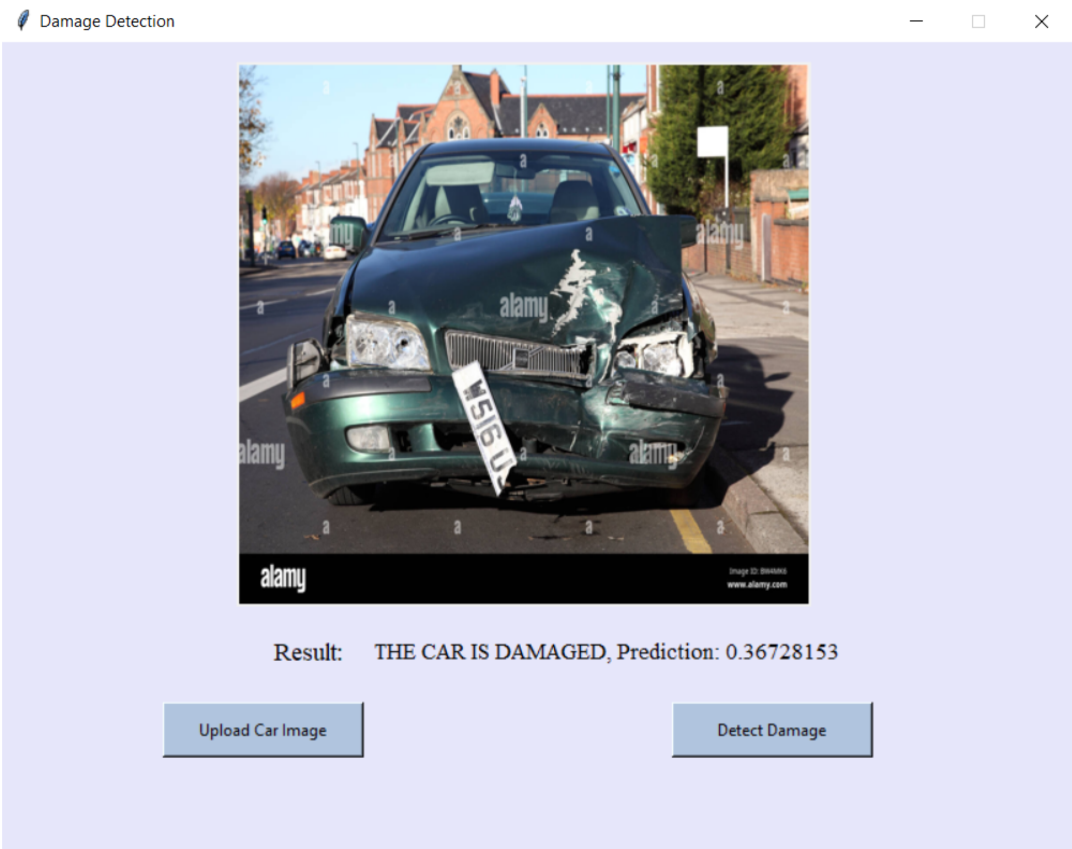

Figure 13: Results of Damaged Car Image on Program.

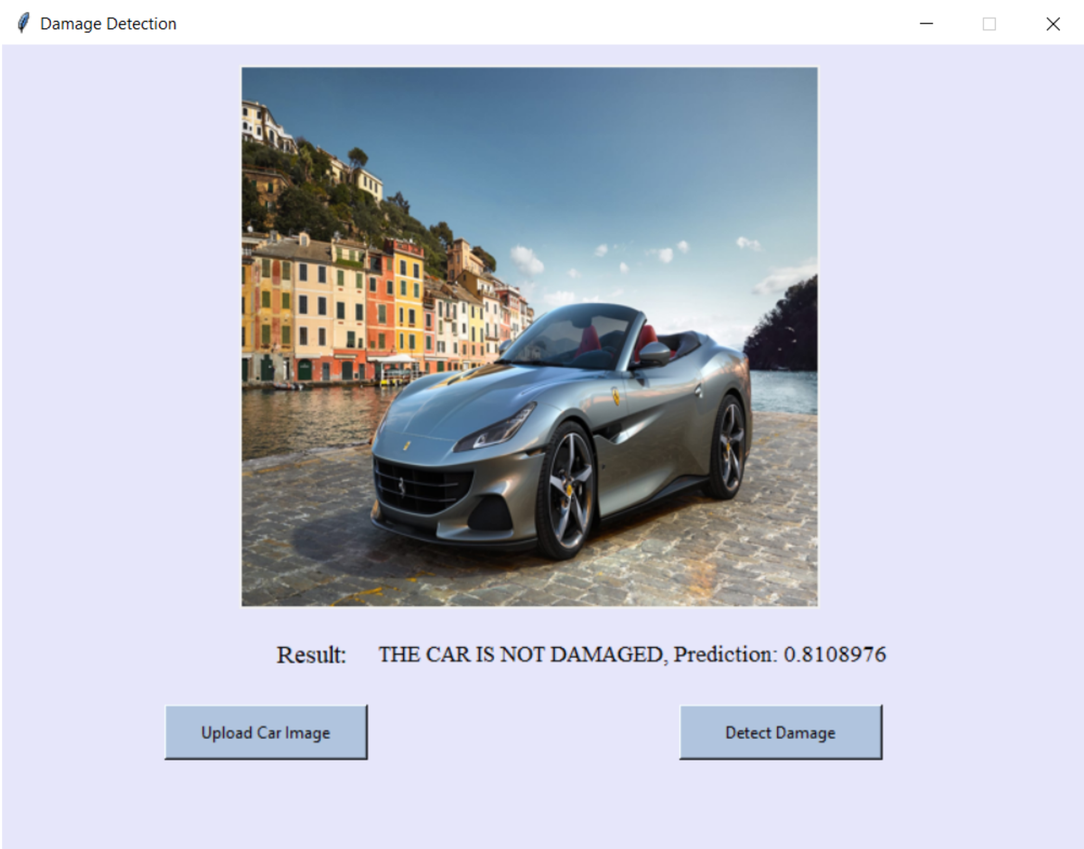

Figure 14: Results of Undamaged Car Image on Program.

6.2 Task Log

Our Engineering Project I meetings were held in our advisor's office. Our meetings were nearly between half an hour and forty five minutes.

Meeting 1 (25.09.2022): In our first meeting with our advisor, we talked about details of our project. We tried to determine the scope of the project according to our discussion. Also, the content of the proposal document was discussed.

Meeting 2 (05.10.2022): After delivering the proposal document, our advisor explained sections of the PSD in detail. We decided to write the first three sections of the PSD by next week.

Meeting 3 (12.10.2022): We showed the first three sections of the PSD, and we took note of advice that our advisor gave us. Also, we decided to make a literature survey about machine learning and deep learning algorithms that will be useful for our project. We planned to do research about related works as well until the next meeting.

Meeting 4 (19.10.2022): We found and analyzed very useful papers about Faster R-CNN [6], YOLO [7] and DenseNet [8] models. We told our advisor how we planned to use these models in our project. In papers, we understood the advantages of using these models in object detection.

Meeting 5 (26.10.2022): We did research about related works to compare what we will do in our project. We separated our concerns for related works into four: brand detection, model detection, damaged part detection and damage percentage detection. We showed our related works in the PSD to our advisor, and we decided to make a table that includes these concerns and which related work includes them. This table will summarize related works.

Meeting 6 (02.11.2022): We completed the first six sections of the PSD. We use the papers that we found while doing research about related works last week. Also, we searched for proper datasets for our project to use in the next stages. We found the dataset that includes damaged and undamaged car images. We wrote the methodology and technical approach section of the PSD, and this section contains that dataset and description of deep learning models that we made research about before. We showed our advisor all the PSD sections that we wrote.

Meeting 7 (09.11.2022): Seventh and Eighth section of PSD were written, and shown to the advisor. We discussed what we have done so far in PSD and in our project. We checked spell mistakes together, and noted them.

Meeting 8 (16.11.2022): Last two sections of PSD were written. We showed our advisor these sections, but we did not evaluate all sections yet. We decided to evaluate all sections of PSD after midterm week, before the deadline.

Meeting 9 (30.11.2022): We checked, and edited our PSD for the last time before submission.

Meeting 10 (07.12.2022): Our advisors suggested making a sprint plan for code, ADD, and presentation of our project. We planned to write code first, then prepare a presentation, and finally start ADD. We will develop the first prototype of our project until the next meeting.

Meeting 11 (21.12.2022): We trained a DenseNet model to detect whether a car is damaged or undamaged by using an uploaded image. After getting the DenseNet model, a simple GUI program was developed as the first prototype of our project. We recorded a short video of the first prototype to show in the presentation. We showed our advisor this program.

Meeting 12 (28.12.2022): We prepared a presentation that contains all necessary information about our project. We rehearsed the presentation several times. We got some feedback from our advisor.

6.3 Task Plan with Milestones

Phase 1 - Data Preprocessing: We will find proper datasets:

1.1: Logo dataset for brand detection.

1.2: Damaged and undamaged car for deciding if the car is damaged or not (already found).

1.3: Damaged car parts to detect location of the damaged part. If we cannot find a labeled dataset, we will label damaged and undamaged car images from the dataset and blur their license plates.

Phase 2 - Keras Implementation:

1.1: Faster R-CNN implementation for brand detection

1.2: YOLO implementation for damaged part detection

1.3: DenseNet implementation for damage percentage detection

Phase 3 - Web Application Development:

1.1: MVC .net Core will be used to develop web application.

1.2: Marketplace of mechanic shops will be developed.

Phase 4 - Mobile Application Development:

1.1: Mobile application will be web-based application that will be adapted from phase 3.

1.2: Java will be used for development.

Phase 5 - Testing and Deployment:

1.1: Testing our models to check if they are working correctly.

1.2: Testing marketplace, driver's screen, mechanic shop's screen and all other screens with different scenarios.

1.3: Other tests will be done in different screens of application.

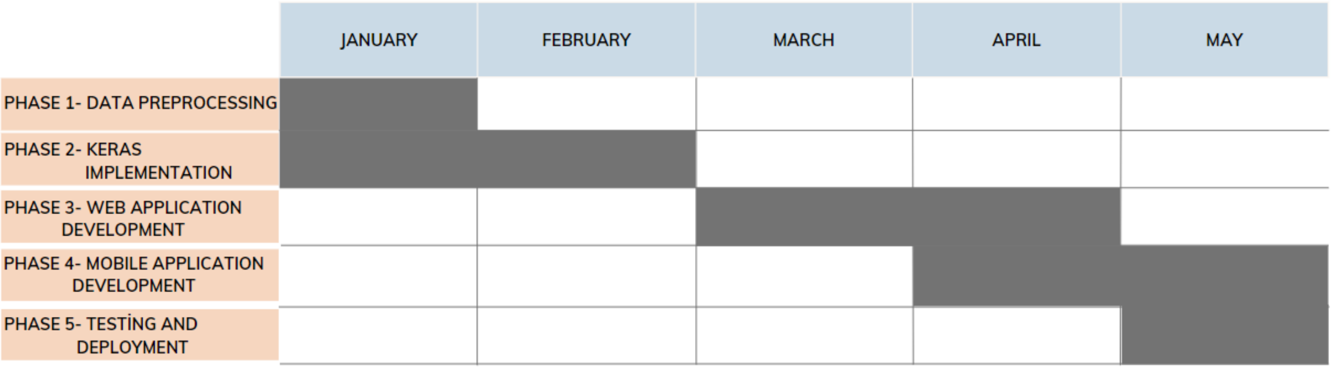

Figure 15: Gantt Chart of Timeline of the Project.

7. REFERENCES

[1] Kunduracı M. F., Kahramanli Örnek, H. "Vehicle Brand Detection Using Deep Learning Algorithms". *International Journal of Applied Mathematics Electronics and Computers 7 (2019)*: 70-74

[2] CarNet.AI [Date of Access: 27/11/2022] [https://carnet.ai/make-model-recognition-api]

[3] Tafazzoli F., Frigui H. "Vehicle Make and Model Recognition Using Local Features and Logo Detection". *International Symposium on Signal, Image, Video and Communications (ISIVC) (2016)*: 353-358

[4] Chua A. C. et al. "Damage Identification of Selected Car Parts Using Image Classification and Deep Learning". *IEEE 13th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM) (2021)*: 1-5

[5] Anuj Shah, "Car Damage Detection" [Date of Access: 10/12/2022] [https://www.kaggle.com/datasets/anujms/car-damage-detection].

[6] Ren S., He K., Girshick R., Sun J. "Faster R-CNN". *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1 (2015)*: 91-99

[7] Gandhi R. "R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms" [Date of Access: 15/11/2022] [https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e]

[8] Huang G., Liu Z., Maaten L., Weinberger K. Q. "Densely Connected Convolutional Networks". *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017):* 2261-2269