

Final Year Project

Decentralized Governance of

Smart Transportation Using Blockchain

Final Evaluation

FYP Team

M. Shahid Hussain 16I-0126

M. Ibrahim Malik 16I-0187

Abdul Haseeb 16I-0296

Supervised By

Dr. Muhammad Asim

FAST School of Computing

National University of Computing and Emerging Sciences,
Islamabad, Pakistan

Acknowledgements

This report has become a reality with a kind support and help of many individuals. We would like to extend our sincere thanks to all of them. Foremost, we want to offer this endeavor to Allah Almighty for the wisdom He bestowed upon us, the strength, peace of our minds and good health in order to finish this iteration. We would like to express our gratitude towards our project supervisor Dr. Muhammad Asim for always helping us and guiding us throughout the iteration.

Table of Contents

1. Introduction	8
2. Project Vision	9
Problem Statement.....	9
Business Opportunity.....	9
Objectives	10
Project Scope	10
Constraints.....	10
Key High-Level Goals and Problem of Stakeholders	11
3. Software Requirements Specifications.....	11
List of Features.....	11
Functional Requirements.....	12
FR01: Identify damaged roads	12
FR02: Identify car crashes	12
FR03: Update location on dashboard	12
FR04: Analyze reports from social media	12
FR05: Analyze reports from mobile application	13
FR06: Verify report authenticity	13
Quality Attributes	13
Non-functional requirements	13
NFR01: Identify severity of pothole.....	13
NFR02: Identify severity of accident.....	13
NFR03: Identify other types of road damage.	14

4. High Level Use Cases	14
UC01.....	14
UC02.....	15
UC03.....	15
UC04.....	16
UC05.....	16
UC06.....	17
UC07.....	17
Use Case Diagram	18
5. Iteration Plan.....	19
6. Iteration 1	19
Expanded Use Cases	19
UC01.....	19
UC02.....	20
UC03.....	21
System Sequence Diagram.....	22
Use Case: Detect Pothole.....	22
Use Case: Display Damage Points	23
Use Case: View Damage Points.....	23
Operation Contracts	24
C01	24
C02	24
C03	24
C04	25

C05	25
C06	25
C07	26
C08	26
Sequence Diagrams.....	27
7. Iteration – 2	30
Expanded Use Cases	30
UC01.....	30
UC02.....	31
UC03.....	32
System Sequence Diagrams	33
Use Case: Label Image	33
Use Case: Verify Detected Pothole	34
Use Case: Insert Data	34
Operation Contracts	35
C01	35
C02	35
C03	36
C04	36
C05	37
Sequence Diagrams.....	38
Use Case: Label Image	38
Use Case: Verify Detected Pothole	39
Use Case: Insert data	40

Activity Diagram.....	41
Domain Model	42
Class Diagram.....	43
Architecture Diagram.....	44
Package Diagram.....	45
Deployment Diagram	46
8. Iteration – 3 & 4.....	46
9. High Level Use Cases	46
Use Case Diagram	47
Expanded Use Cases	48
UC01.....	48
System Sequence Diagram.....	48
Use Case: Detect Pothole.....	48
Activity Diagram.....	50
Domain Model	50
Operation Contracts	49
C01	49
C02	49
Sequence Diagrams.....	49
Use Case: Detect Pothole.....	49
Class Diagram.....	51
Architecture Diagram.....	51
10. Implementation Details	53
11. User Manual	54

Desktop	54
Android Application	60

1. Introduction

Deteriorated roads induce vehicle damage, traffic congestion, and driver discomfort which influences traffic management of smart cities. Specifically, they slowdown the traffic and create choke points. About one third of the accidents are caused by deteriorated roads and car crashes. Therefore, such things need to monitored and resolved for a smooth flow of traffic.

A smart city's basic principle is to use all available resources efficiently, but to report any damages on the road, a manual process is used which requires a survey team analyzing the city manually for damaged roads and signs. This gets expensive and labor intensive as a human is reporting all damages. This process can be effective to some extent; however, its drawbacks prevent it from being the best solution.

Traffic needs to move at all times, so a system that can allow such problems to be fixed quickly is required. An automated system that can detect and report damages quickly would tremendously help in minimizing the deterioration. Hence, an autonomous and intelligent system of detecting the damage is a desirable and required feature of a smart city infrastructure. By using the latest technologies like Blockchain as it is a decentralized, distributed and public digital ledger that is used to record transactions across many smart cities, and machine learning and image processing techniques to detect road damages to make our system state of the art.

2. Project Vision

Smart cities provide modern infrastructures and services by deploying principles of information and communication technologies. The need for a more efficient and safe transportation system is a growing demand for all these cities. Damaged roads and signs are some factors that contribute to vehicle damage, slower traffic, congestion and crashes.

In this project, we propose a modern road and car crash detection system that uses block chain technology to maintain a robust decentralized system of verification and prioritization, machine learning and image processing techniques to detect road damage, and a dashboard to display the verified and labeled road damages.

Problem Statement

Deteriorated roads induce vehicle damage, traffic congestion, and driver discomfort which influences traffic management of smart cities. Specifically, they slow down traffic and create choke points. About one third of the accidents are caused by deteriorated roads and car crashes. Therefore, such things need to be monitored and resolved for a smooth flow of traffic.

Business Opportunity

Our product will be a component of a smart city. The smart city projects in Pakistan would be able to use this to efficiently detect and repair road damage. The costs associated with doing this the traditional ways have been insurmountably high, but our system will allow the administration to effectively repair the damages.

Objectives

- Improve smart city infrastructure.
- Repair damaged roads and give assistance to car crash victims as soon as possible.
- Maintain a robust decentralized system of verification.
- Maintain a dedicated dashboard for identifying positions of damage.
- Update incoming queries on dashboard.

Project Scope

This documentation is associated with DG Smart Transport. The project is currently aimed towards detecting urban damage to ensure efficient flow of traffic in order to make cities safer and more secure.

Constraints

The primary constraint of our system is the lack of a dash cam culture in Pakistan, as only a small minority owns them already, and an even smaller minority among them would be interested in joining this program.

Stakeholders Description

1. Administration.
2. Citizens.

Key High-Level Goals and Problem of Stakeholders

Stakeholder	Goal	Problem
Administration	The administration would like to minimize road damage and vehicular crashes and congestion due to the aforementioned road damages.	An efficient method for detecting and reporting road damage does not exist in the country as of now.
Citizen	The citizens would like an easy way to report damaged roads to the administration so necessary action can be taken immediately.	An efficient method for reporting damaged roads does not exist, which requires minimum effort.

3. Software Requirements Specifications

List of Features

- Identify damaged roads and car crashes.
- Update the location of each point on dedicated dashboard.
- Maintain a decentralized chain of all points.
- Analyze incoming reports/queries from social media and mobile application.
- Verify the authenticity of any report/query.

Functional Requirements

FR01: Identify damaged roads

FR01-01	The camera shall take a picture of the pothole with geotags.
FR01-02	The picture shall be fed into the machine learning model with its geotags.
FR01-03	The pothole shall be detected in the image if it exists.

FR02: Identify car crashes

FR02-01	The camera shall take a picture of the car crash with geotags.
FR02-02	The picture shall be fed into the machine learning model with its geotags.
FR02-03	The car crash shall be detected in the image if it exists.

FR03: Update location on dashboard

FR03-01	The location shall be sent to the dashboard with the image.
FR03-02	The dashboard shall display the image with its location.

FR04: Analyze reports from social media

FR04-01	Reports about damaged roads and car crashes shall be generated from social media accounts.
FR04-02	A web scraping algorithm shall get all relevant reports.
FR04-03	The machine learning model shall analyze the images for any relevant information.

FR05: Analyze reports from mobile application

FR05-01	Reports about damaged roads and car crashes shall be generated from an application install base
FR05-02	The machine learning model shall analyze the images for any relevant information.

FR06: Verify report authenticity

FR06-01	Reports shall be verified using a blockchain thread.
FR06-02	Each separate report from the same location shall be appended to the same blockchain thread.
FR06-03	The longest thread shall have the highest priority.

Quality Attributes

- Correctness
- Availability
- Efficiency
- Reliability
- Maintenance

Non-functional requirements***NFR01: Identify severity of pothole.***

After detecting presence of potholes, the severity of pothole shall be detected. This will aid in prioritizing the fixing process.

NFR02: Identify severity of accident.

After detecting accident, the severity of accident shall be detected. This will aid in prioritizing which accidents need priority.

NFR03: Identify other types of road damage.

After identifying base line road damage, other urban damage such as damaged signs and damaged traffic lights can also be detected. This will aid in identifying more day to day damages, making city transportation safer and more secure.

4. High Level Use Cases***UC01***

Use Case	Detect pothole
Actors	Camera
Type	Primary
Description	The camera is the source of input which passes image/video to model, which detects the pothole.

UC02

Use Case	Detect accident
Actors	Camera
Type	Primary
Description	The camera is the source of input which passes image/video to the model, which detects the accident.

UC03

Use Case	Label image
Actors	Machine learning model
Type	Secondary
Description	The camera passes the images as input to the machine learning model, which detects the potholes/accidents (if any) and then labels that particular image.

UC04

Use Case	Insert data
Actors	Blockchain system
Type	Secondary
Description	The model detects any potholes/accidents and sends geotagged images to the proximity algorithm which inserts the data into the relevant thread of the blockchain system.

UC05

Use Case	Verify detected pothole
Actors	Blockchain system
Type	Secondary
Description	The camera detects any potholes/accidents and sends geotagged images/videos. The blockchain system stores the data into the relevant thread and verifies that the system does not already hold any other instance of the same point reported.

UC06

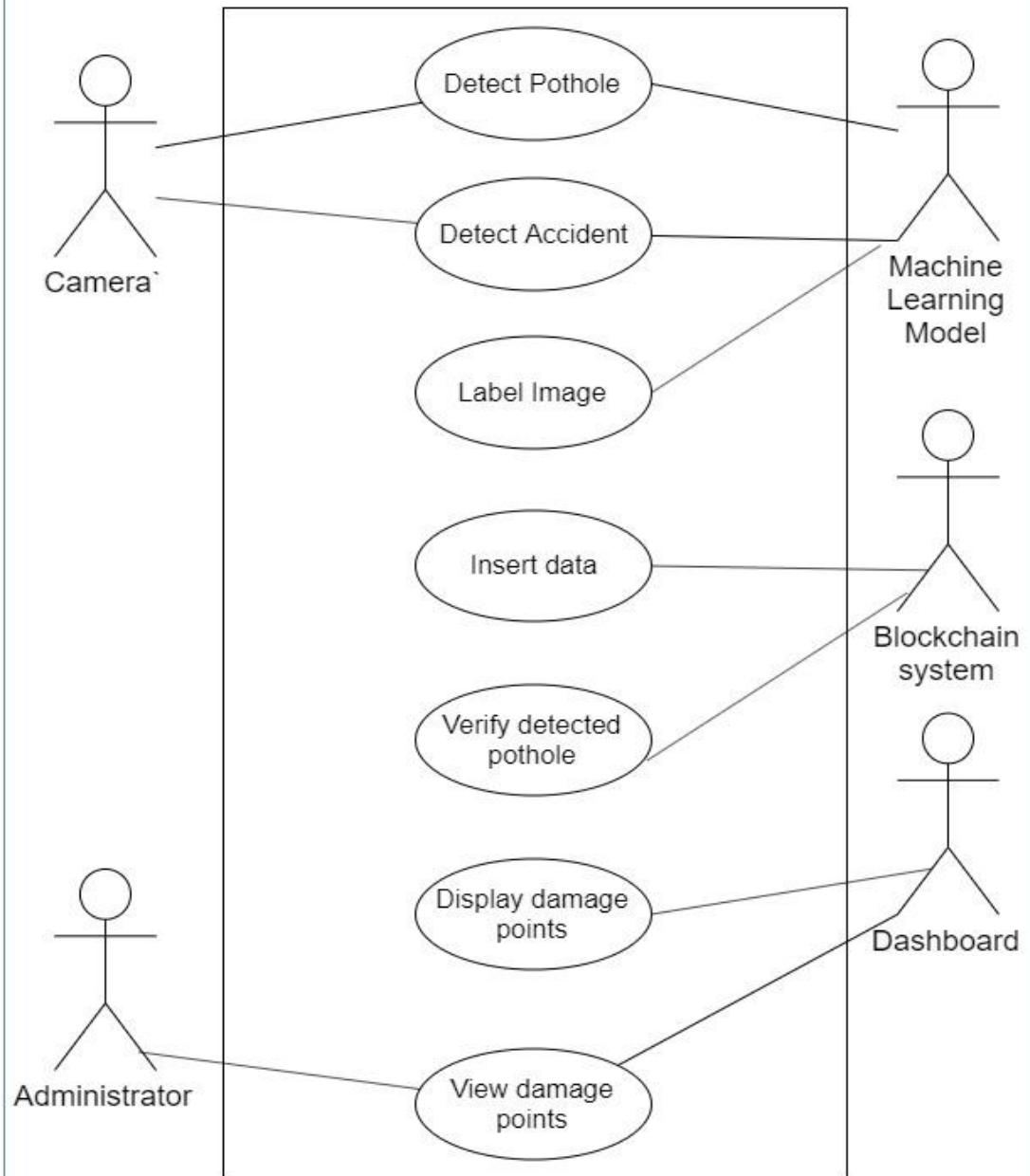
Use Case	Display damage points
Actors	Dashboard
Type	Secondary
Description	The camera detects any potholes/accidents and sends geotagged images/videos. The blockchain system stores and verifies the data. The verified data points are then shown on dashboard with their locations.

UC07

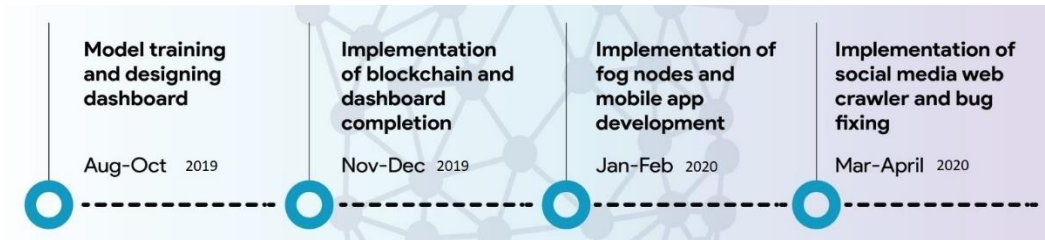
Use Case	View damage points
Actors	Administrator
Type	Primary
Description	The dashboard shows points of damage with their location. The administrator can view the points of damage on the dashboard and take the corresponding action.

Use Case Diagram

DGSmartTransport



5. Iteration Plan



6. Iteration 1

Expanded Use Cases

UC01:

Use Case	Detect Pothole
Scope	Efficient Smart Transportation System
Level	User's Goal
Primary Actor	Camera
Stakeholders and Interests	Camera: Needs to submit good quality, geotagged images for model to easily detect photos. Model: Wants a fast and highly accurate algorithm for detecting potholes as fast as possible.
Preconditions	Camera sends input to model.
Postconditions	Model detects potholes in image received from camera.

Main Success Scenario	<p>Camera takes video input and separates it frame by frame into images.</p> <p>Camera passes each image into model.</p> <p>Model analyzes each image and identifies ones with potholes in them.</p>
------------------------------	--

UC02:

Use Case	Display damage points.
Scope	Efficient Smart Transportation System
Level	User's Goal
Primary Actor	Dashboard
Stakeholders and Interests	<p>Dashboard: Wants to show all points of damage clearly.</p> <p>Administrator: Wants to view all points of damage easily and effortlessly.</p>
Preconditions	Model detects pothole in geotagged image.
Postconditions	Dashboard displays location of pothole detected by using the geotag provided.
Main Success Scenario	<p>Camera takes video input and separates it frame by frame into images.</p> <p>Camera passes each image into model.</p> <p>Model analyzes each image and identifies ones with potholes in them.</p>

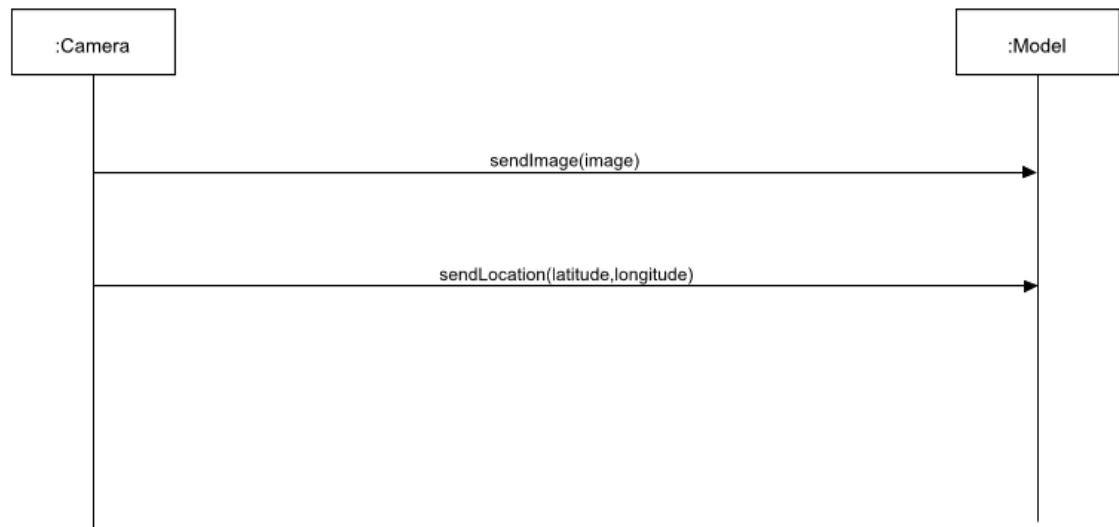
	Dashboard displays the location of each pothole detected by using geotag provided with image.
--	---

UC03:

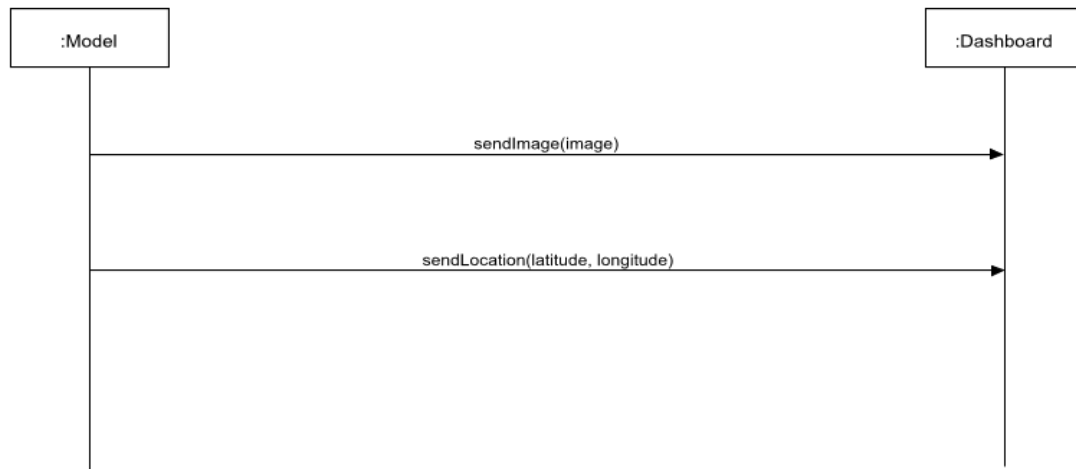
Use Case	View damage points.
Scope	Efficient Smart Transportation System
Level	User's Goal
Primary Actor	Administrator
Stakeholders and Interests	Administrator: Wants to view all points of damage easily and effortlessly.
Preconditions	Camera sends input to model.
Postconditions	Model detects pothole in geotagged image. Dashboard displays location of pothole detected.
Main Success Scenario	<p>Camera takes video input and separates it frame by frame into images.</p> <p>Camera passes each image into model.</p> <p>Model analyzes each image and identifies ones with potholes in them.</p> <p>Dashboard displays the location of each pothole detected by using geotag provided with image.</p> <p>Administrator views the location of each damage point according to his/her preference.</p>

System Sequence Diagram

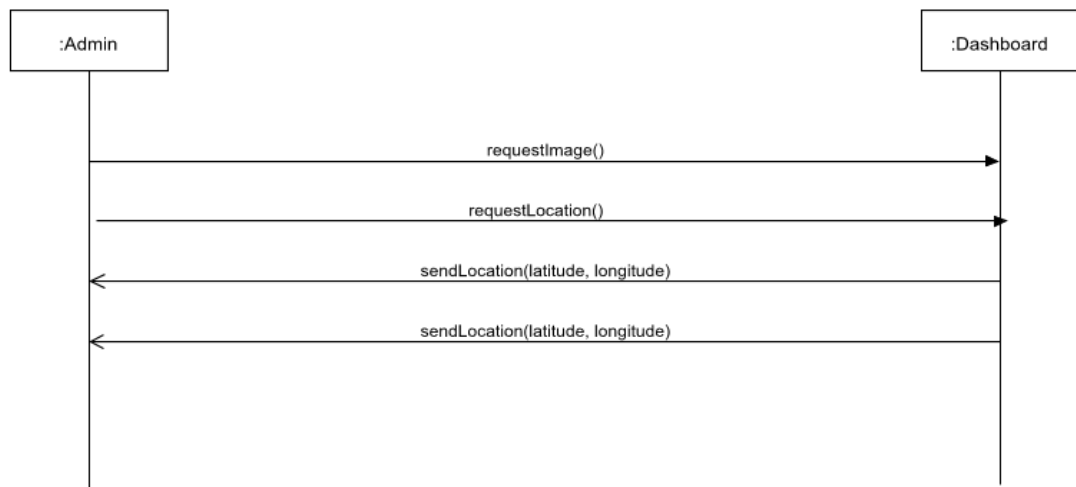
Use Case: Detect Pothole



Use Case: Display Damage Points



Use Case: View Damage Points



Operation Contracts

C01

Name: sendImage

Responsibility: Send image from camera to ML model.

Type: System

Use case: Detect Pothole

Pre-conditions:

- Camera has captured an image or video.

Post-conditions:

- Model will use image or video for pothole detection.

C02

Name: sendLocation

Responsibility: Send location of image from camera to ML model.

Type: System

Use case: Detect Pothole

Pre-conditions:

- Camera has captured a geotagged image or video.

Post-conditions:

- Model will use geotags to forward them to dashboard.

C03

Name: sendImage

Responsibility: Send image from ML model's output to dashboard.

Type: System

Use case: Display damage point

Pre-conditions:

- Model has detected a pothole in the given image.

Post-conditions:

- Dashboard will display image.

C04

Name: sendLocation

Responsibility: Send location from ML model's output to dashboard.

Type: System

Use case: Detect Pothole

Pre-conditions:

- Model has detected a pothole at the given location in the image.

Post-conditions:

- Dashboard will display Coordinates on a maps interface.

C05

Name: requestImage

Responsibility: Ask for pothole image from dashboard.

Type: System

Use case: View damage points

Pre-conditions:

- Dashboard has the image the admin requests.

Post-conditions:

- Administrator will use image to check up on pothole intensity.

C06

Name: requestLocation

Responsibility: Ask for pothole location from dashboard.

Type: System

Use case: View damage points

Pre-conditions:

- Dashboard has the location the admin requests.

Post-conditions:

- Administrator will use the returned location to check up on pothole location.

C07

Name: sendImage

Responsibility: Send image from dashboard to administrator.

Type: System

Use case: View damage points

Pre-conditions:

- Administrator has requested access to pothole image.

Post-conditions:

- Administrator will use image to check up on pothole intensity.

C08

Name: sendLocation

Responsibility: Send location of pothole from camera to model.

Type: System

Use case: View damage points

Pre-conditions:

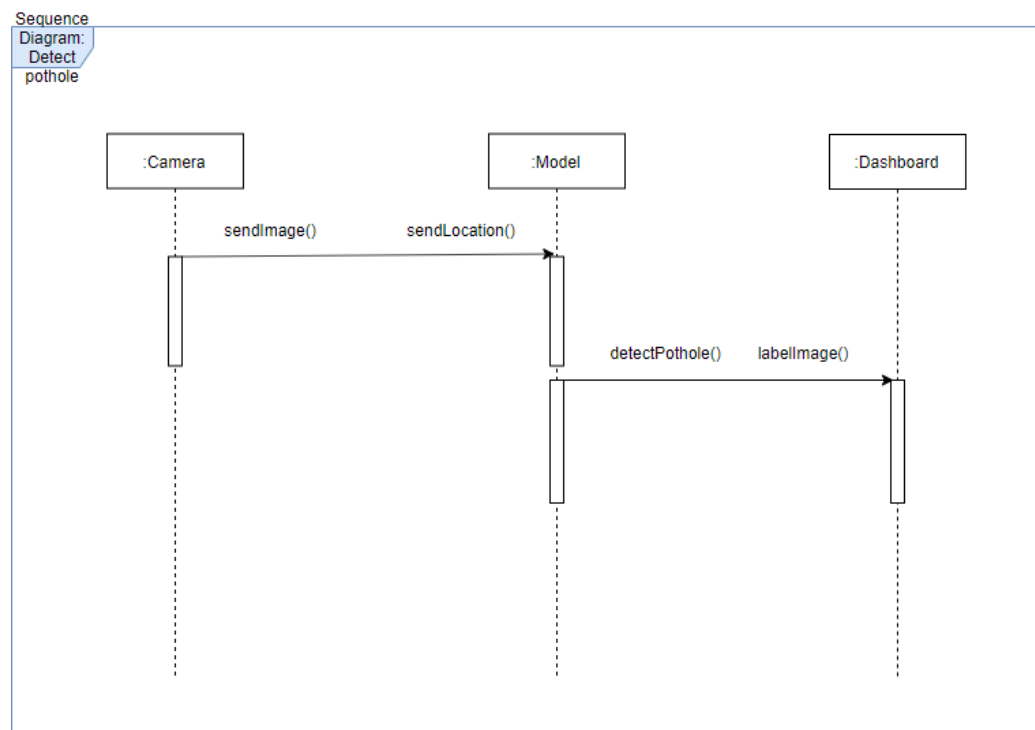
- Administrator has requested access to pothole location

Post-conditions:

- Administrator will use the returned location to check up on pothole location.

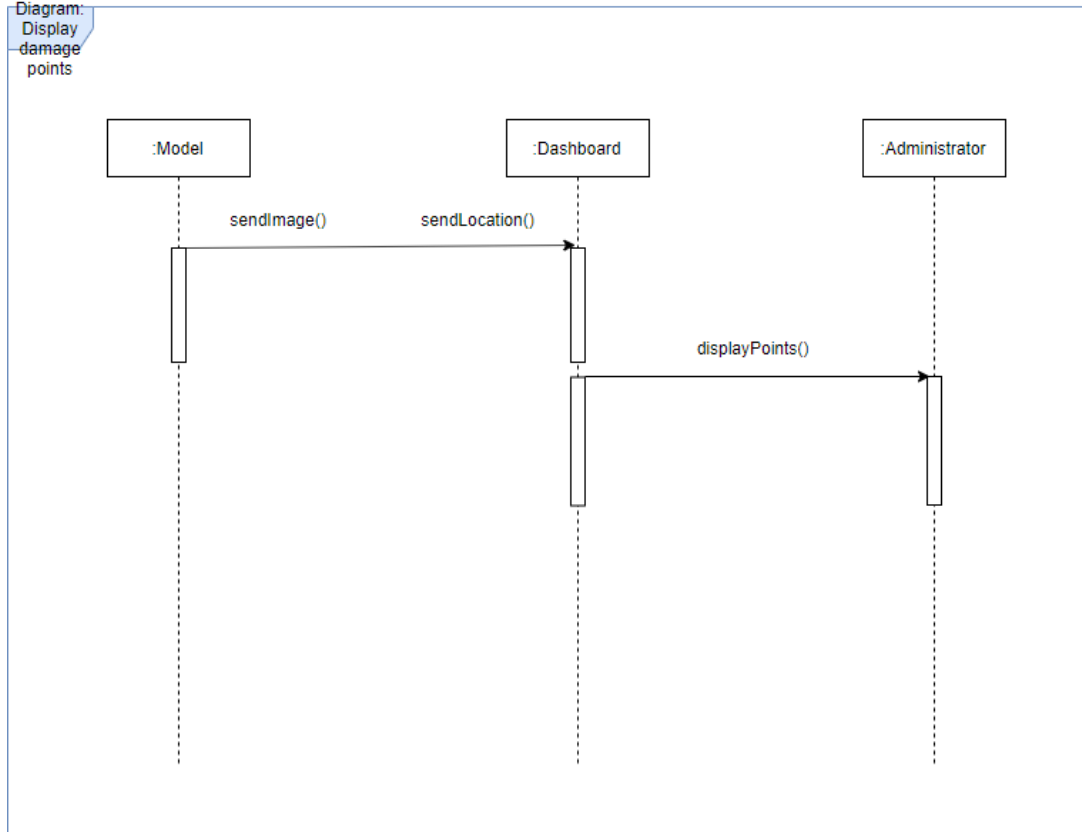
Sequence Diagrams

Use Case: Detect Pothole



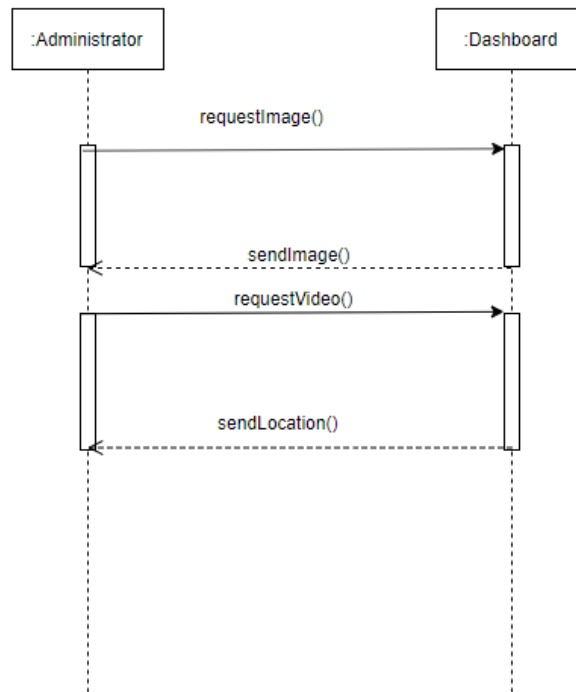
Use Case: Display damage points

Sequence
Diagram:
Display
damage
points



Use Case: View damage points

Sequence
Diagram:
Display
damage
points



7. Iteration – 2

Expanded Use Cases

UC01

Use Case	Label image
Scope	Efficient Smart Transportation System
Level	User's Goal
Primary Actor	Machine learning model
Stakeholders and Interests	Camera: Needs to submit good quality, geotagged images for model to easily detect photos. Model: Wants a fast and highly accurate algorithm for detecting potholes and labels the images.
Preconditions	Camera sends input to model.
Postconditions	Model detects potholes in image received from camera and labels the respective image.
Main Success Scenario	Camera takes video input and separates it frame by frame into images. Camera passes each image into model. Model analyzes each image and identifies ones with potholes in them and returns the labelled image.

UC02

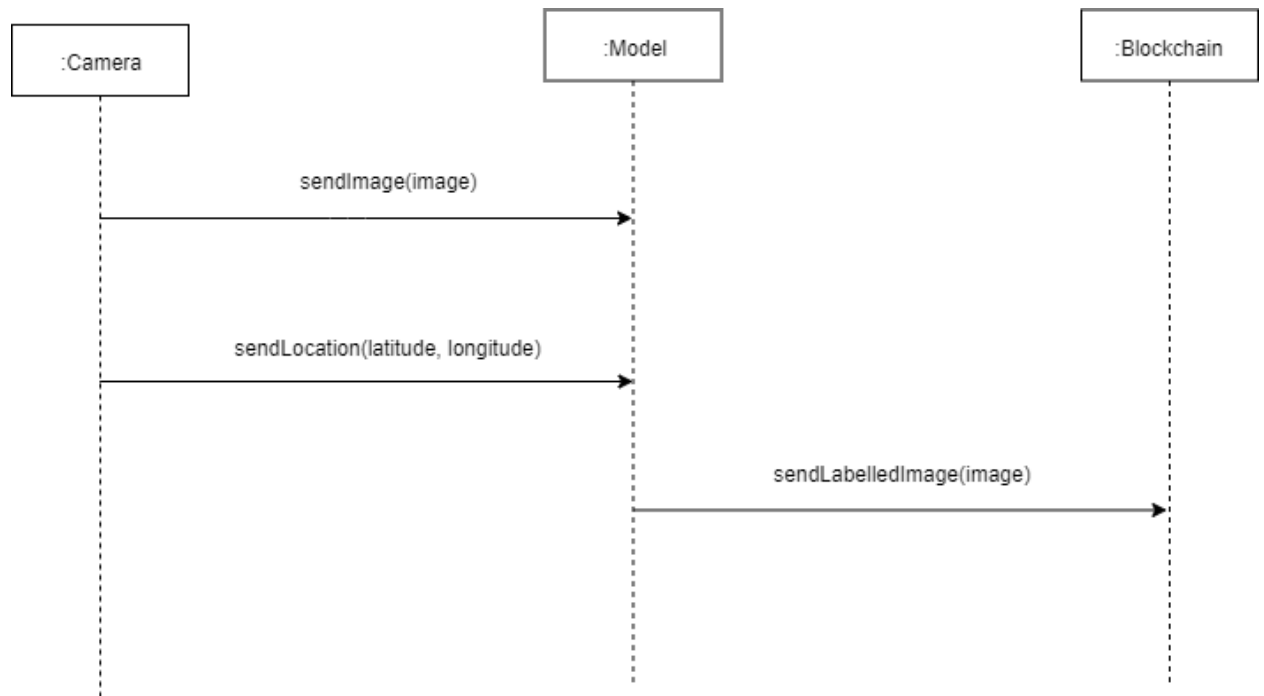
Use Case	Verify detected pothole
Scope	Efficient Smart Transportation System
Level	User's Goal
Primary Actor	Blockchain System
Stakeholders and Interests	Blockchain system: Wants to verify if a particular image in a thread previously exists in the same or any other thread.
Preconditions	Blockchain system identifies the thread of the labelled image.
Postconditions	The blockchain system verifies that the system does not already hold any other instance of the same point reported.
Main Success Scenario	<p>Model analyzes each image and identifies ones with potholes in them and returns the labelled image.</p> <p>Proximity algorithm identifies the thread of the labelled image, and passes the image to the blockchain system.</p> <p>The blockchain system verifies that the system does not already hold any other instance of the same point reported.</p>

UC03

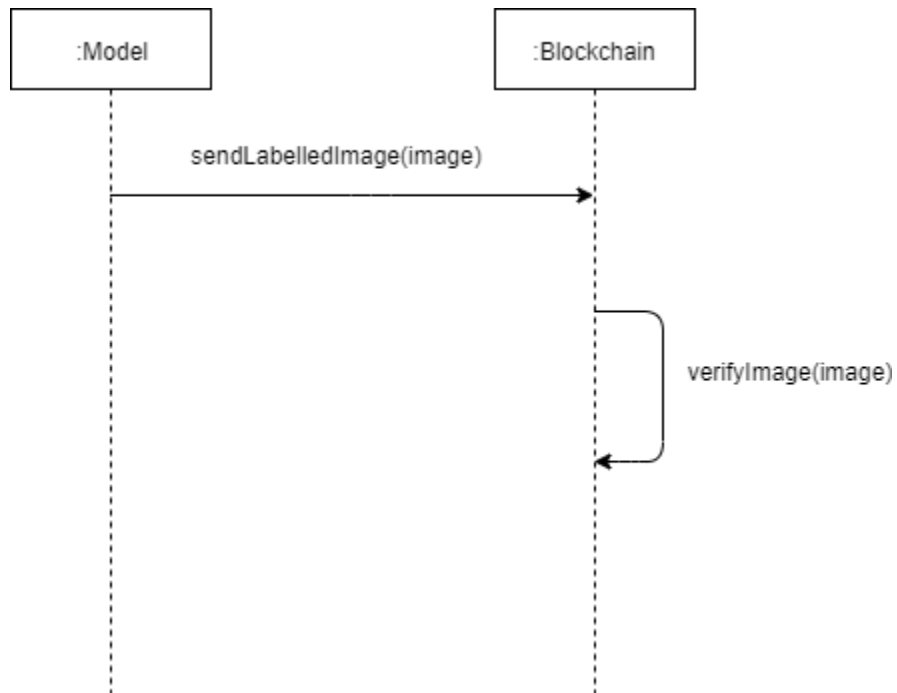
Use Case	Insert data
Scope	Efficient Smart Transportation System
Level	User's Goal
Primary Actor	Blockchain System
Stakeholders and Interests	<p>Proximity algorithm: Needs to calculate in which thread the labelled should be inserted, and passes the image to the relevant thread.</p> <p>Blockchain system: Wants to insert the received labelled image into its corresponding thread.</p>
Preconditions	<p>Model detects potholes in image received from camera and labels the respective image.</p> <p>Proximity algorithm identifies the thread of the labelled image, and passes it onto blockchain system.</p>
Postconditions	The labelled image with geotag is inserted into the relevant thread of the blockchain system.
Main Success Scenario	<p>Camera takes video input and separates it frame by frame into images.</p> <p>Camera passes each image into model.</p> <p>Model analyzes each image and identifies ones with potholes in them and returns the labelled image.</p> <p>Proximity algorithm identifies the thread of the labelled image, and passes the image to the blockchain system.</p>

System Sequence Diagrams

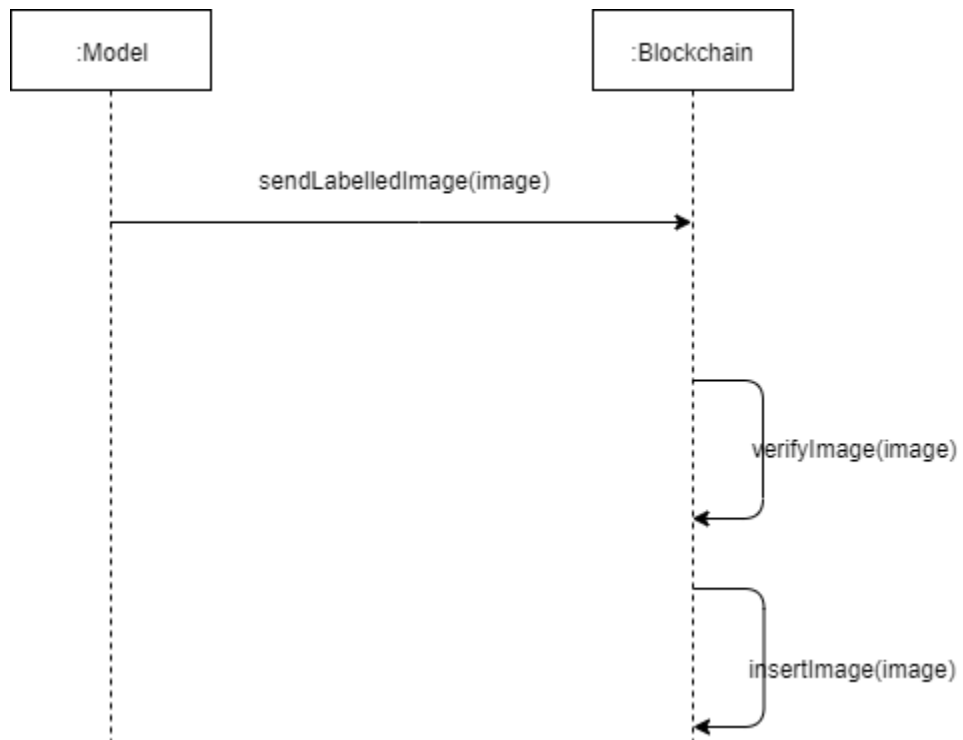
Use Case: Label Image



Use Case: Verify Detected Pothole



Use Case: Insert Data



Operation Contracts

C01

Name: sendImage()

Responsibility: Send image from camera to ML model.

Type: System

Use case: Label Image

Pre-conditions:

- Camera has captured an image or video.

Post-conditions:

- Camera has passed image to the machine learning model.

C02

Name: sendLocation(latitude, longitude)

Responsibility: Send location(coordinates) from camera to machine learning model.

Type: System

Use case: Label Image

Pre-conditions:

- Camera has captured an image or video.

Post-conditions:

- Camera has passed location to the machine learning model.

C03

Name: sendLabelledImage(image)

Responsibility: Send labelled image from model to blockchain if the labelled image has detected road damage.

Type: System

Use case: Label Image, Verify Detected Pothole, Insert Data

Pre-conditions:

- Camera has captured an image or video.
- Camera has passed image and its location(coordinates) to machine learning model.

Post-conditions:

- Machine learning analyzes image and passes the labelled image if detection is made.

C04

Name: verifyImage(image)

Responsibility: Verify the existence of image passed in blockchain thread.

Type: System

Use case: Verify Detected Pothole, Insert Data

Pre-conditions:

- Model identifies pothole detected and sends labelled image to blockchain system.

Post-conditions:

- Blockchain system verifies the existence of image passed in blockchain thread.

C05

Name: insertImage(image)

Responsibility: Insert image in corresponding blockchain thread.

Type: System

Use case: Verify Detected Pothole, Insert Data

Pre-conditions:

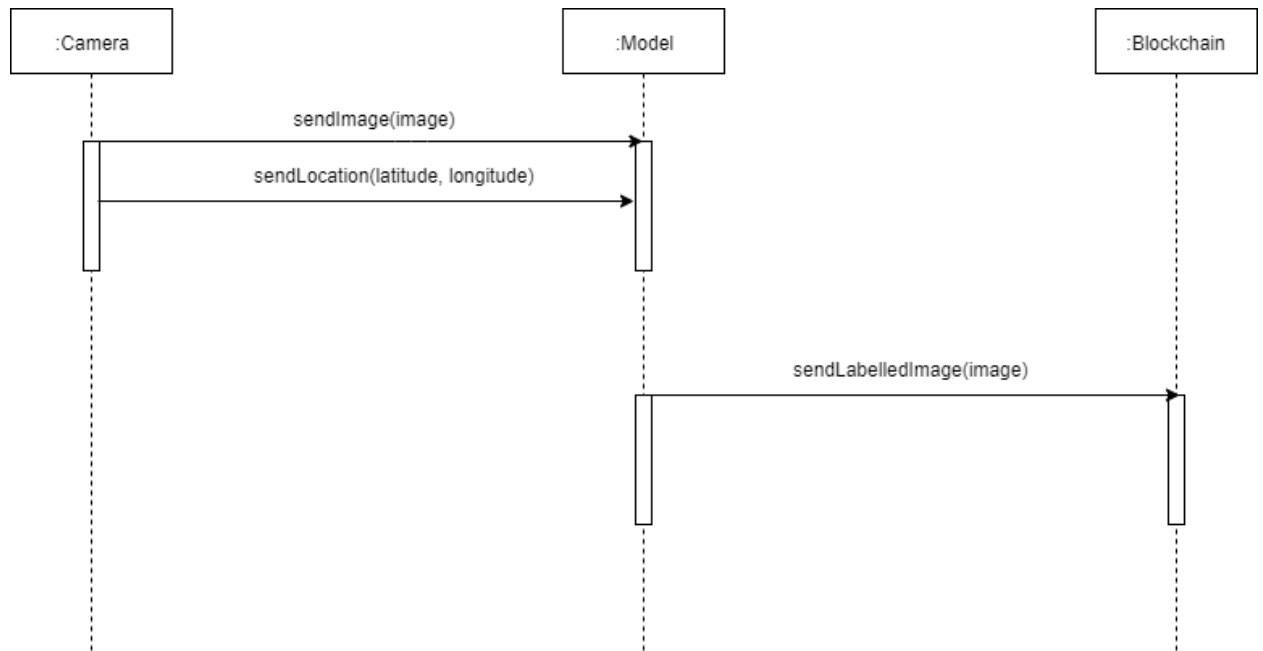
- Model identifies pothole detected and sends labelled image to blockchain system.
- Blockchain system verifies the existence of image passed in blockchain thread.

Post-conditions:

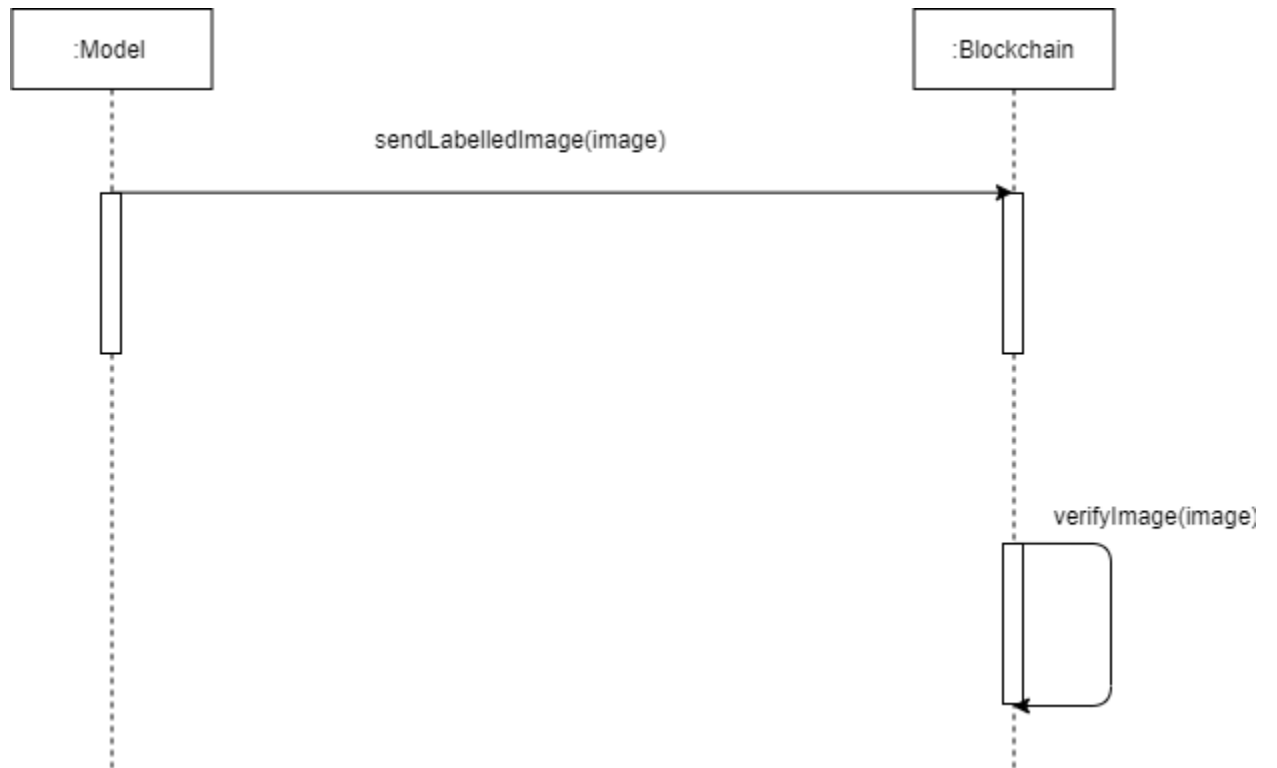
If detected damage point does not already exist, the blockchain system inserts image into corresponding blockchain thread.

Sequence Diagrams

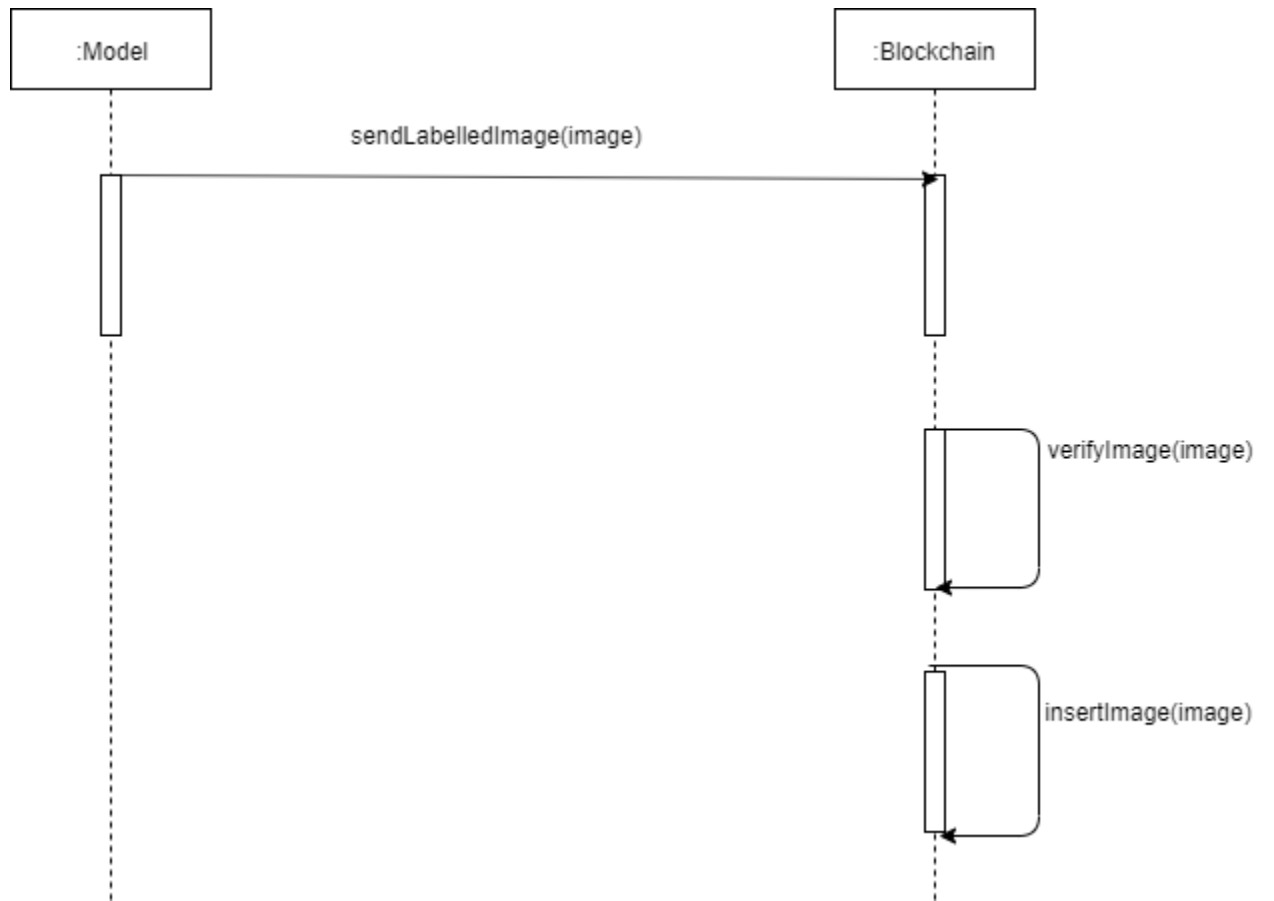
Use Case: Label Image



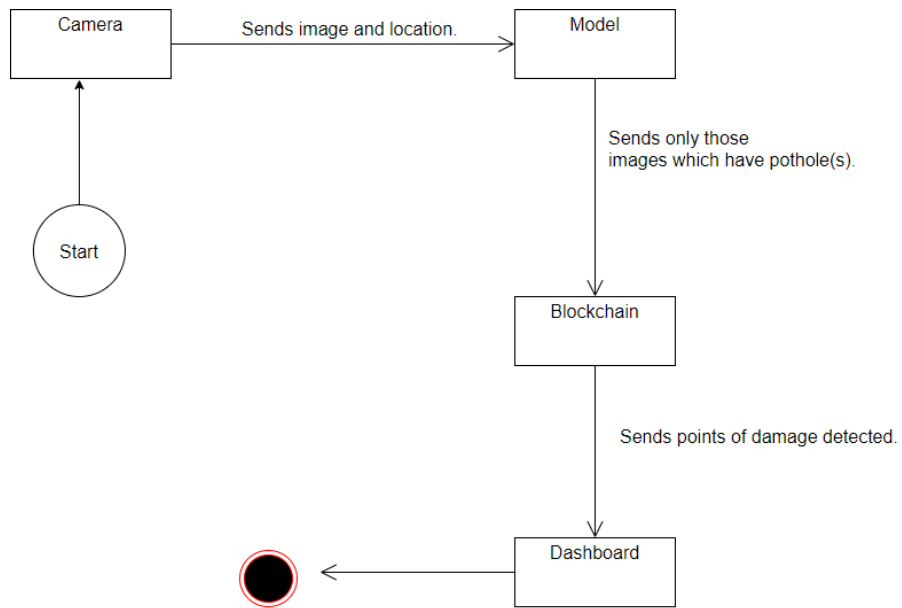
Use Case: Verify Detected Pothole



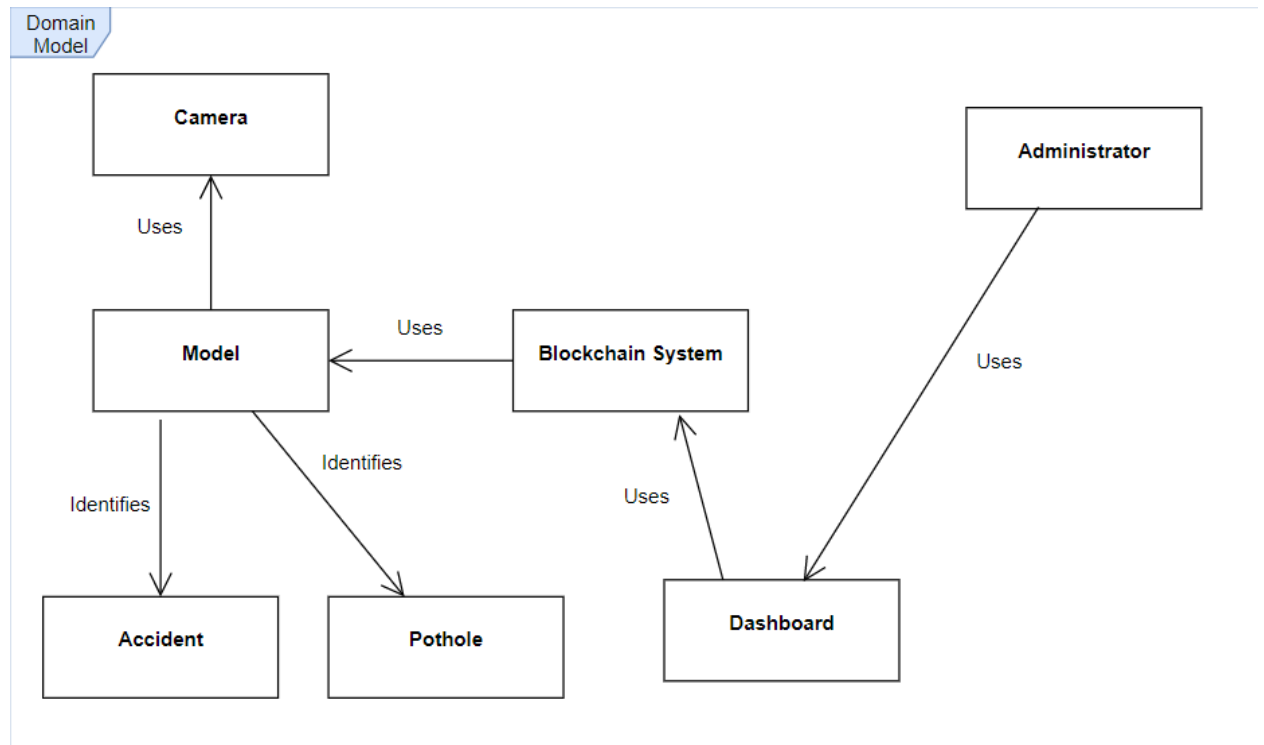
Use Case: Insert data



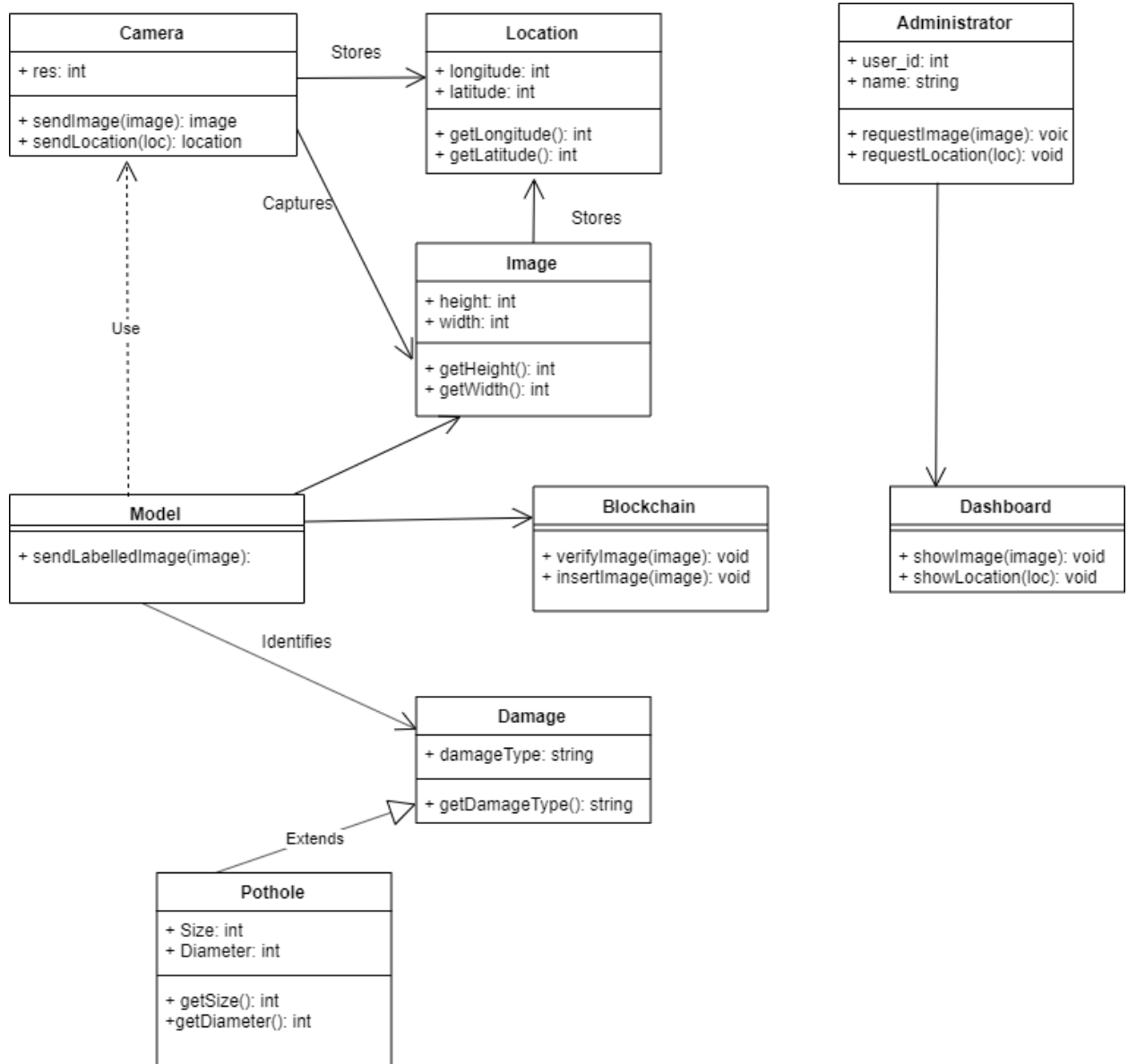
Activity Diagram



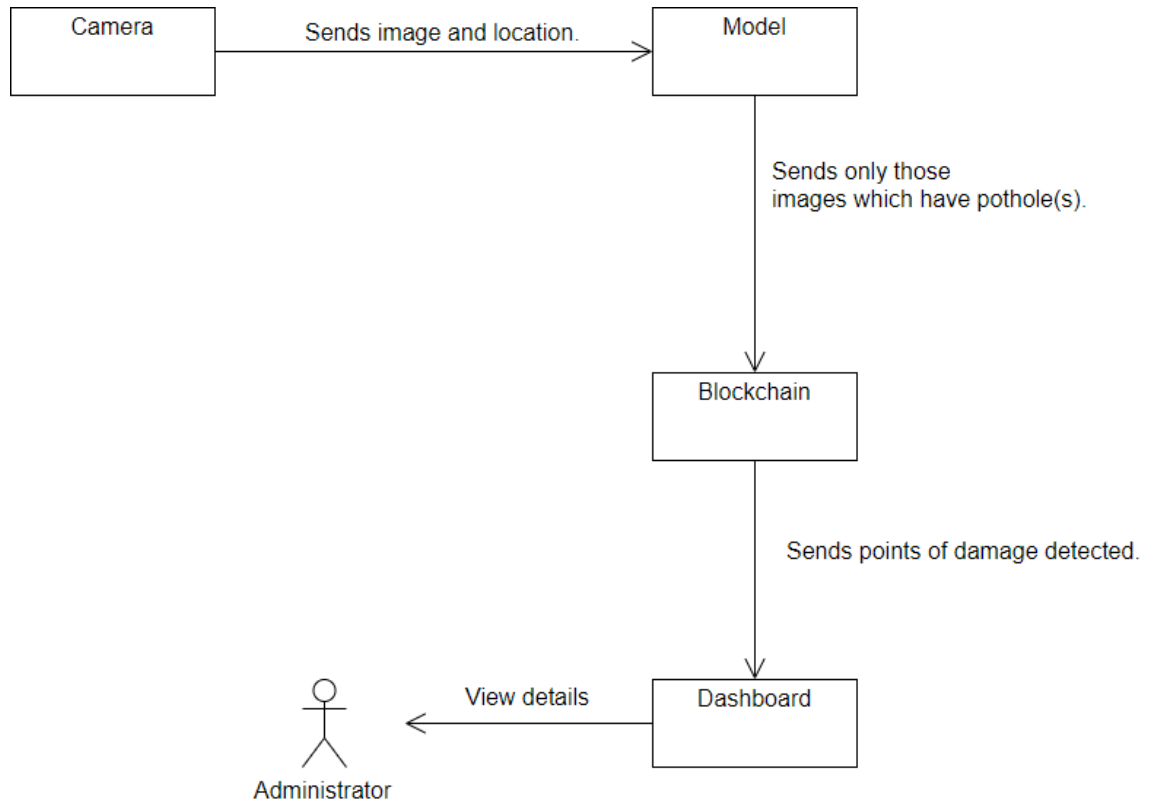
Domain Model



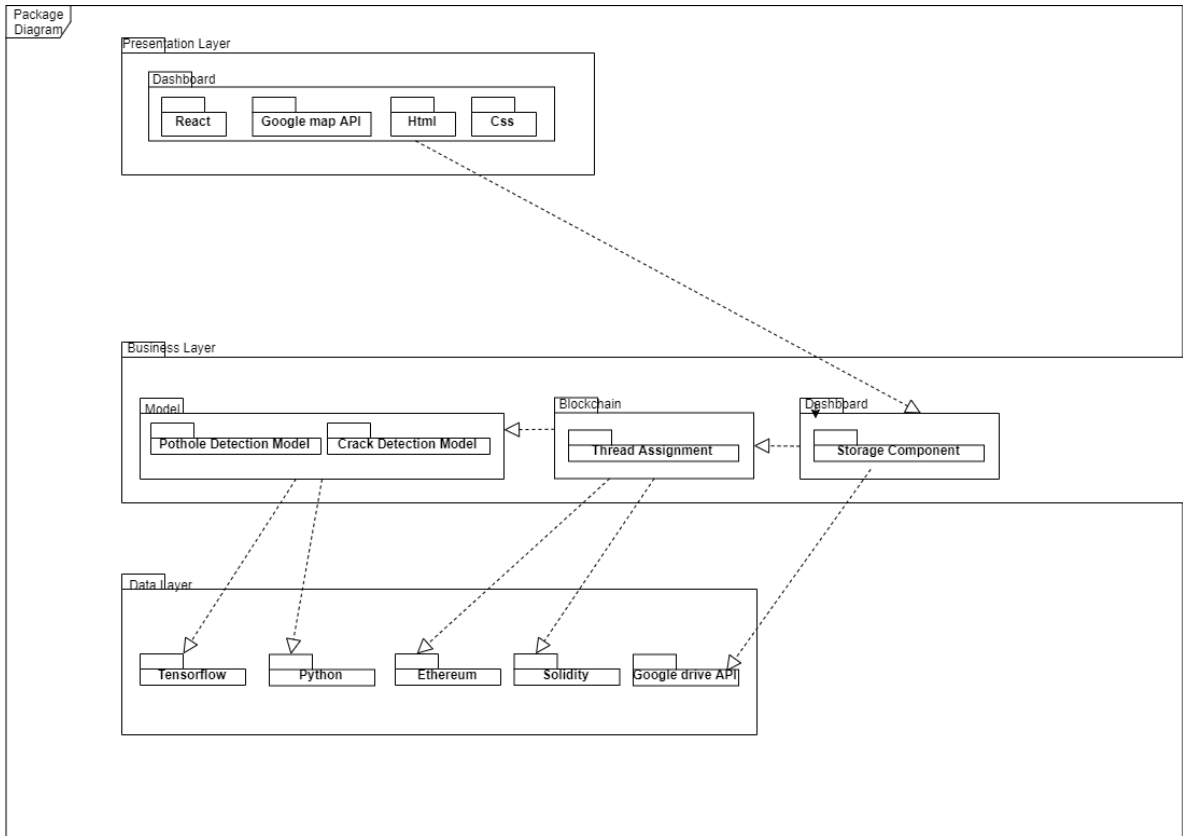
Class Diagram



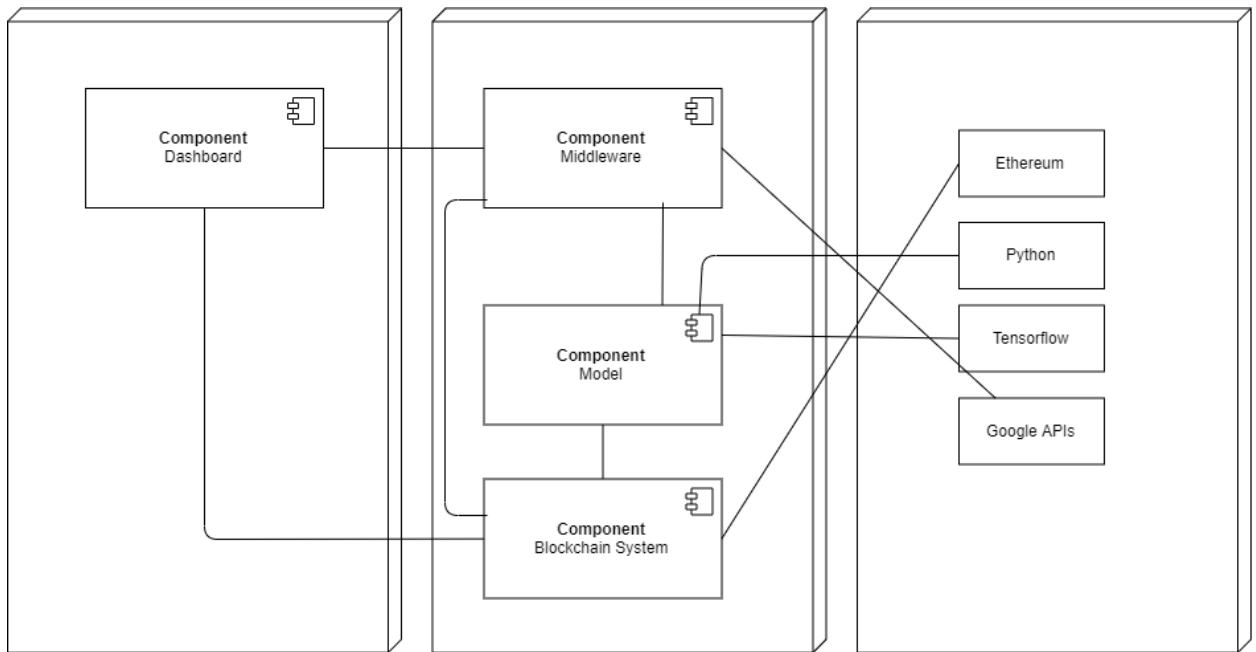
Architecture Diagram



Package Diagram



Deployment Diagram

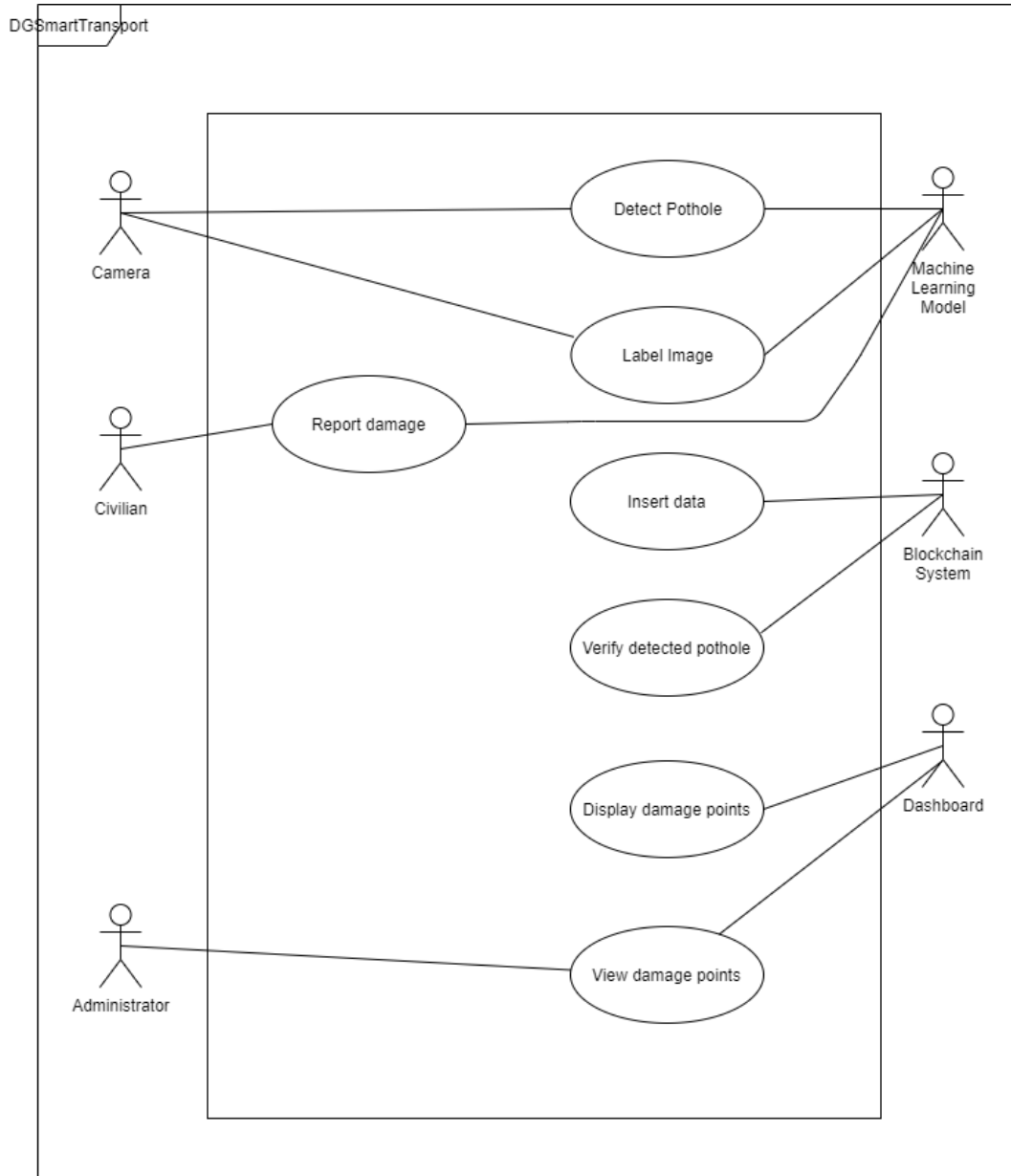


8. Iteration – 3 & 4

9. High Level Use Cases

Use Case	Report damage
Actors	Citizen/ Civilian
Type	Primary
Description	The civilian uses the mobile application to take a picture of, and report any road related damage. The geotagged image is transferred to the machine learning model for detection of damage.

Use Case Diagram



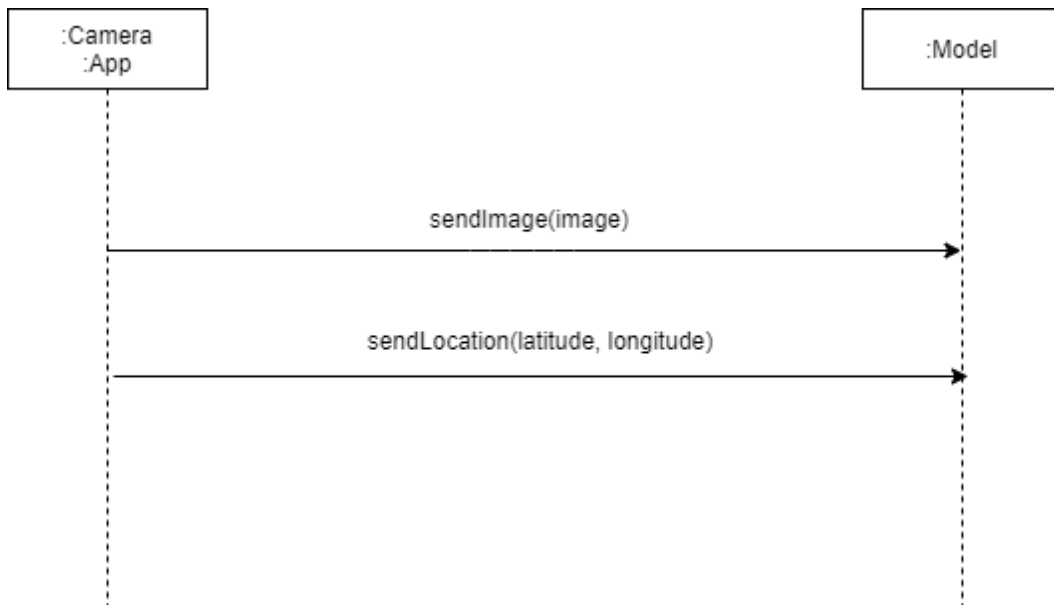
Expanded Use Cases

UC01:

Use Case	Report damage.
Scope	Efficient Smart Transportation System
Level	User's Goal
Primary Actor	Citizen/Civilian
Stakeholders and Interests	Citizen: Wants to report any road related damage so that the relevant authorities can fix it.
Preconditions	Camera (or App) sends input to model.
Postconditions	<ol style="list-style-type: none">1. Model detects pothole in geotagged image.2. Dashboard displays location of pothole detected.
Main Success Scenario	<ol style="list-style-type: none">1. User launches the application.2. The app uses the camera to takes photo(s).3. Camera passes results to model for detection.

System Sequence Diagram

Use Case: Detect Pothole



Operation Contracts

C01

Name: sendImage

Responsibility: Send image from camera (or app) to ML model.

Type: System

Use case: Detect Pothole

Pre-conditions:

- Camera has captured an image or video (either dash cam or App camera).

Post-conditions:

- Model will use image or video for pothole detection.

C02

Name: sendLocation

Responsibility: Send location of image from camera to ML model.

Type: System

Use case: Detect Pothole

Pre-conditions:

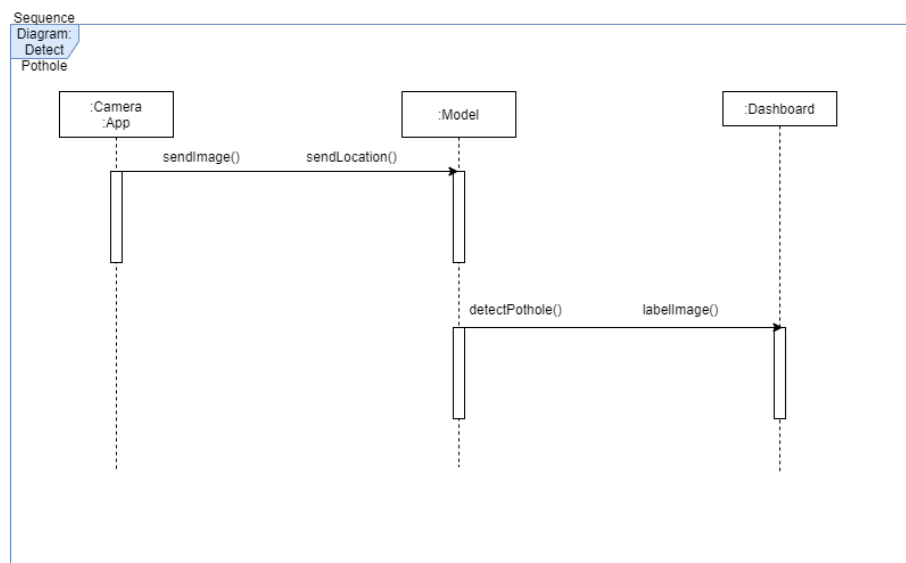
- Camera has captured a geotagged image or video (either dash cam or App camera).

Post-conditions:

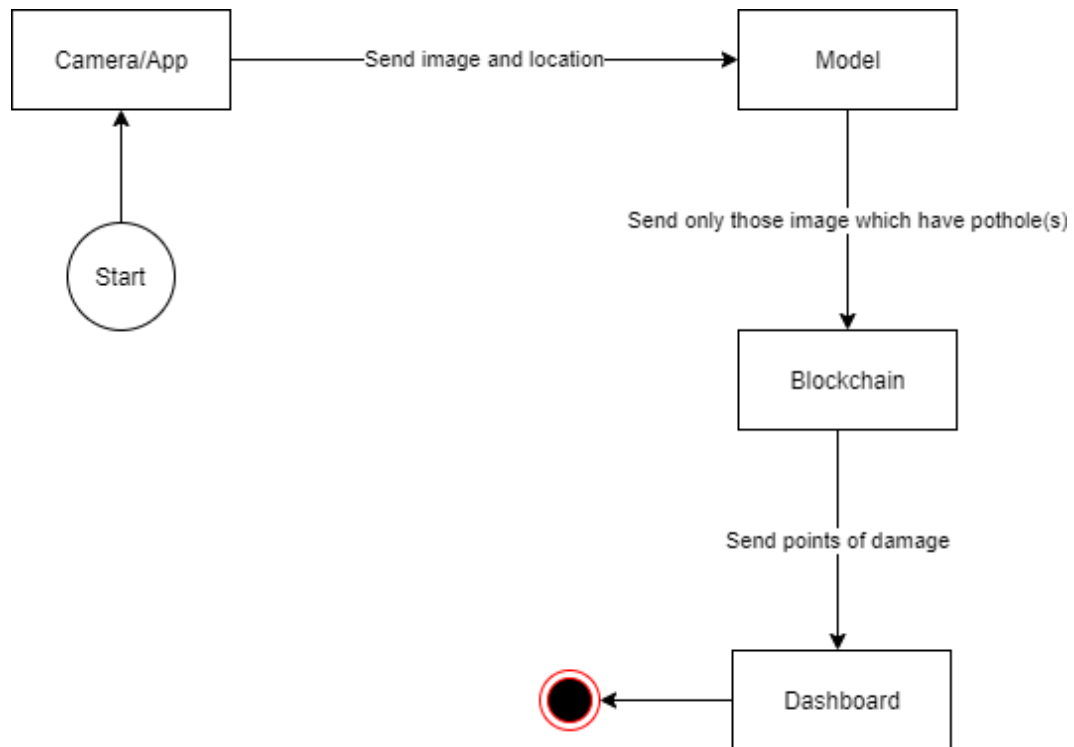
- Model will use geotags to forward them to dashboard.

Sequence Diagrams

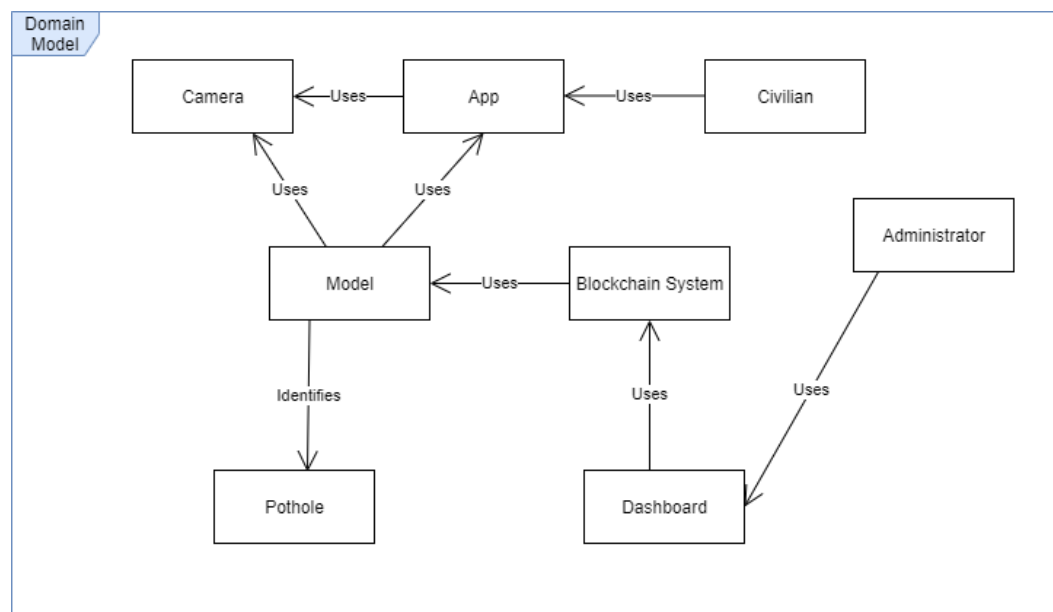
Use Case: Detect Pothole



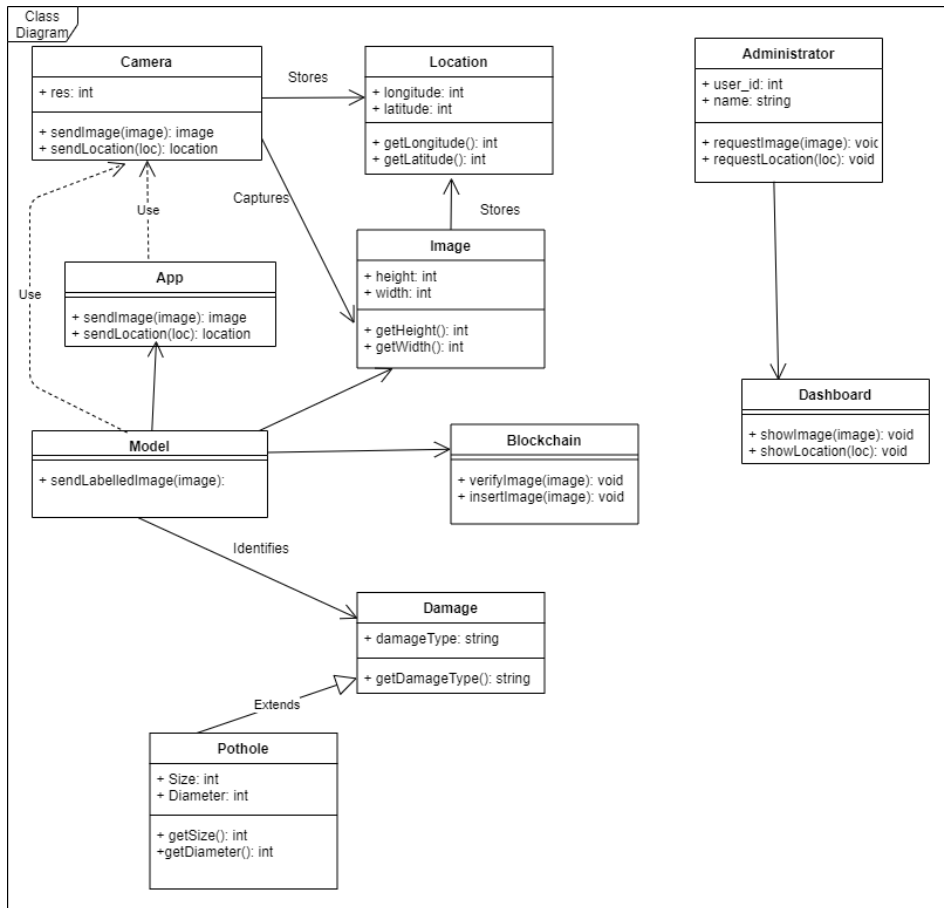
Activity Diagram



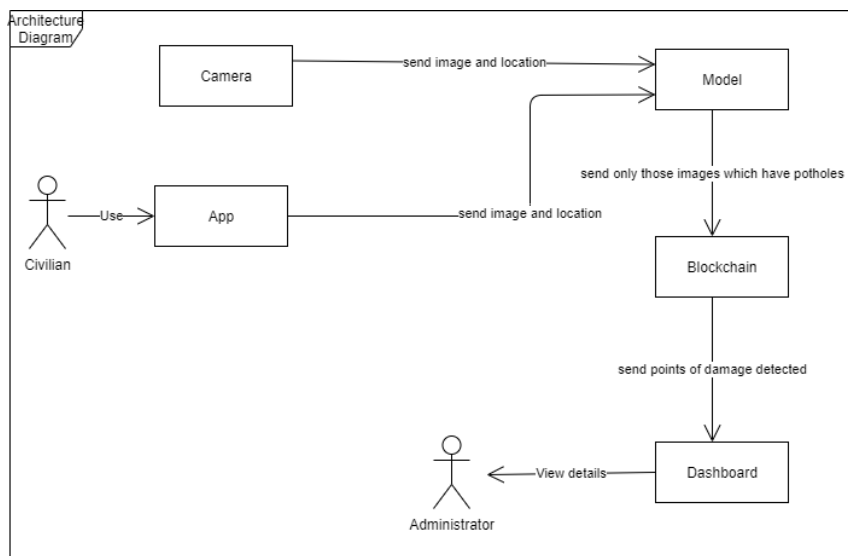
Domain Model



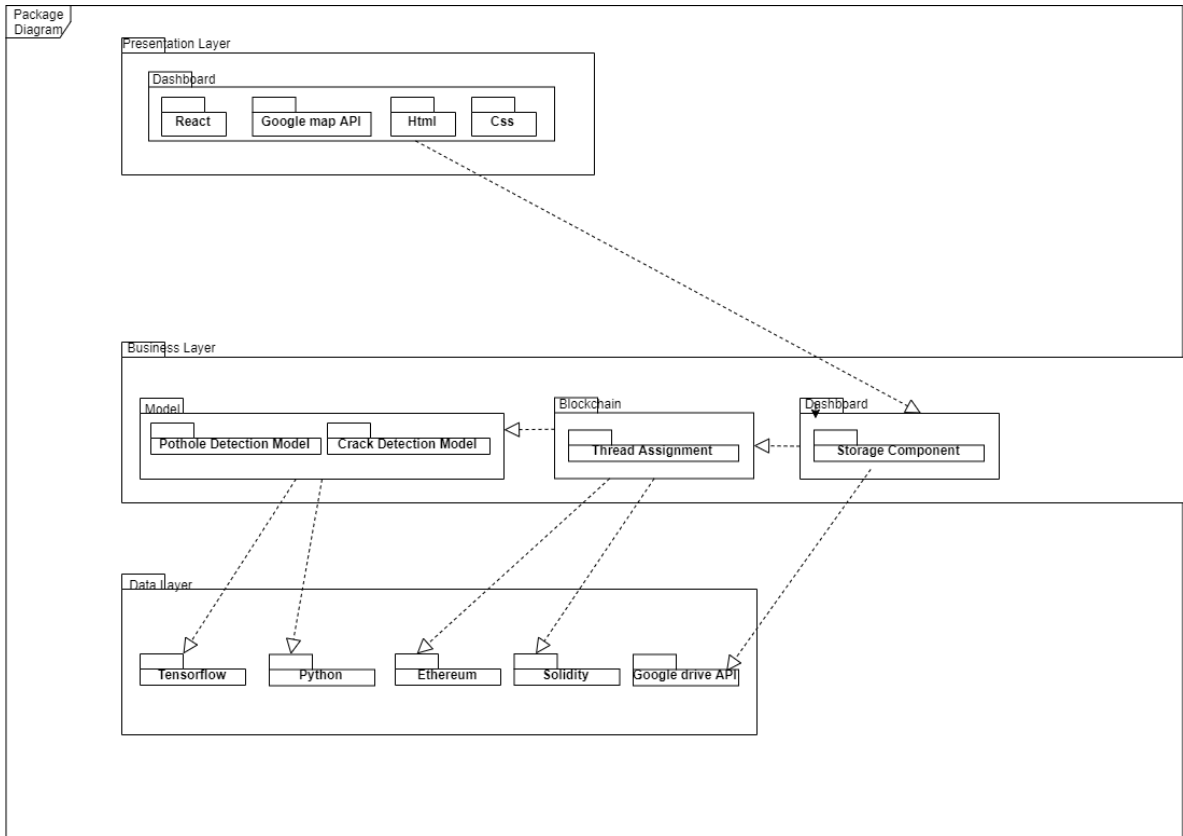
Class Diagram



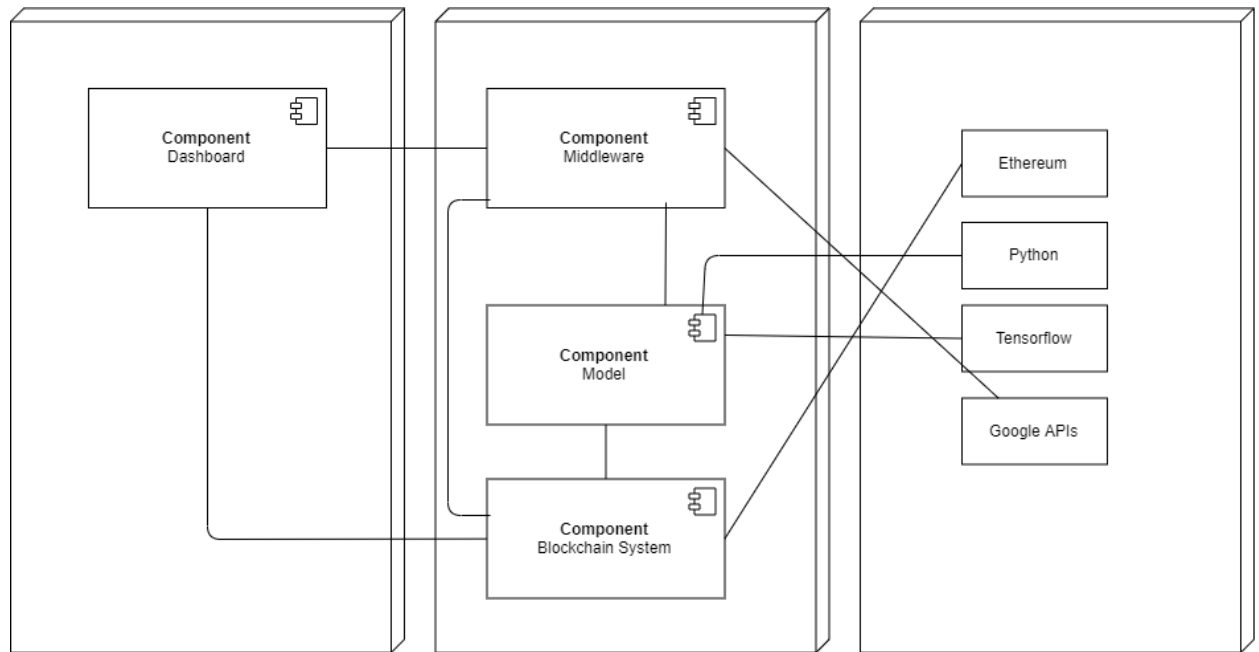
Architecture Diagram



Package Diagram



Deployment Diagram



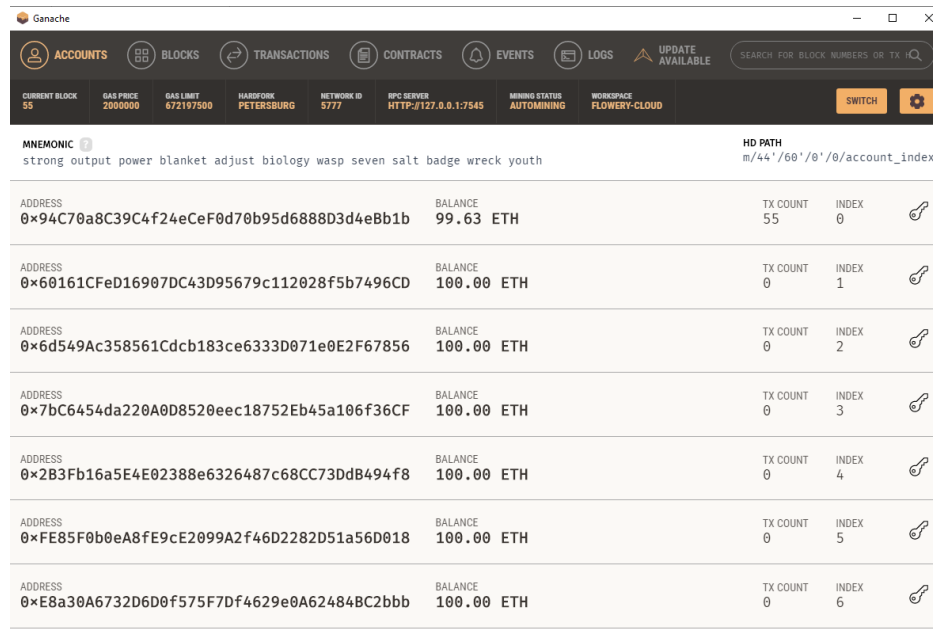
10. Implementation Details

The project has three main components: blockchain, machine learning model and administrative dashboard. The project follows a MVC architectural pattern. The blockchain is based on ethereum. It is a decentralized system, which enables anyone in the network to check the validity of a transaction, also it's distributed so no single point of failure. The machine learning model is trained using tensorflow object detection. The model is based on a convolutional neural network architecture, called resnet. The administrative dashboard is built in react and offers a seamless and intuitive experience.

11. User Manual

Desktop

First we will start our local Ethereum Blockchain using Ganache



The screenshot shows the Ganache desktop application. The top navigation bar includes icons for ACCOUNTS, BLOCKS, TRANSACTIONS, CONTRACTS, EVENTS, LOGS, and an UPDATE AVAILABLE button. Below this, a status bar displays various network metrics: CURRENT BLOCK (55), GAS PRICE (200000), GAS LIMIT (672197500), HARDFORK (PETERSBURG), NETWORK ID (5777), RPC SERVER (HTTP://127.0.0.1:7545), MINING STATUS (AUTOMINING), and WORKSPACE (FLOWERY-CLOUD). The main area displays account information for a mnemonic: "strong output power blanket adjust biology wasp seven salt badge wreck youth". The HD PATH is shown as "m/44'/60'/0'/0/account_index". A table lists six accounts with their addresses, balances, transaction counts, and indices.

ADDRESS	BALANCE	TX COUNT	INDEX
0x94C70a8C39C4f24eCeF0d70b95d6888D3d4eBb1b	99.63 ETH	55	0
0x60161CFed16907DC43D95679c112028f5b7496CD	100.00 ETH	0	1
0x6d549Ac358561Cdcb183ce6333D071e0E2F67856	100.00 ETH	0	2
0x7bC6454da220A0D8520eec18752Eb45a106f36CF	100.00 ETH	0	3
0x2B3Fb16a5E4E02388e6326487c68CC73DdB494f8	100.00 ETH	0	4
0xFE85F0b0eA8fE9cE2099A2f46D2282D51a56D018	100.00 ETH	0	5
0xE8a30A6732D6D0f575F7Df4629e0A62484BC2bbb	100.00 ETH	0	6

Figure 1 Local Ethereum Blockchain Deployed on Ganache

After which we will deploy our Smart Contract using

```
truffle migrate
```

command in powershell in the directory where the smart contract exists. Now the smart contract will be deployed on the local Ethereum Blockchain. After which we can use the deployed smart contract to send and retrieve data of Potholes reported.

Ganache

ACCOUNTS

BLOCKS

TRANSACTIONS

CONTRACTS

EVENTS

LOGS

UPDATE AVAILABLE

SEARCH FOR BLOCK NUMBERS OR TX HASH

CURRENT BLOCK
55

GAS PRICE
2000000

GAS LIMIT
672197500

HARDFORK
PETERSBURG

NETWORK ID
5777

RPC SERVER
HTTP://127.0.0.1:7545

MINING STATUS
AUTOMINING

WORKSPACE
FLOWERY-CLOUD

SWITCH

← BACK

Damage

ADDRESS
0x264e471f6b5a57282935a3ef2Ae38F68ab9CD800

BALANCE
0.00 ETH

CREATION TX
0x382EDf42B7D7377D9Fe21e08A7c91d9F097D34ac08Fb9AB40aB28c2cABE8c12

STORAGE

4 items

damageCount : uint 12

deleteCount : uint 2

deleted : {} mapping (not supported yet)

reported : {} mapping (not supported yet)

TRANSACTIONS

TX HASH
0xbbea4b02f1a4f64b2e88d6e0f6f6ca63cea4b88eb7814ce4ca8344126971271b

CONTRACT CALL

FROM ADDRESS
0x94C70a8C39C4f24eCeF0d70b95d6888D3d4eBb1b

TO CONTRACT ADDRESS
Damage

GAS USED
251382

VALUE
0

Figure 2 Smart Contract Deployed on Ethereum

Start by running the model. In order to run the model, run the detection python script using the command:

```
python3 odt.py
```

The model will detect road damages in the video provided and isolate images of them.



Figure 3 Model output

Afterwards, to run the front end run this in the project directory:

```
npm start
```

Open <http://localhost:3000> to view it in the browser.

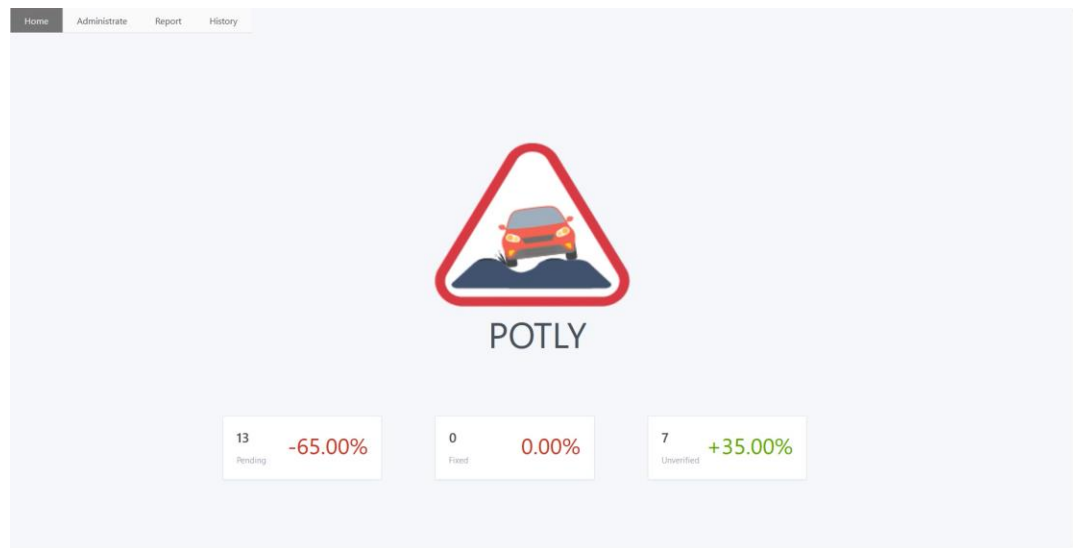


Figure 4 The home page

The home page will appear, showing an overview of damages that are fixed, pending and those that require further verification before they appear in the administrate panel



Figure 5 Administrate page

The administrate page shows a list of all verified reported damages and a google maps view will show where they take place.

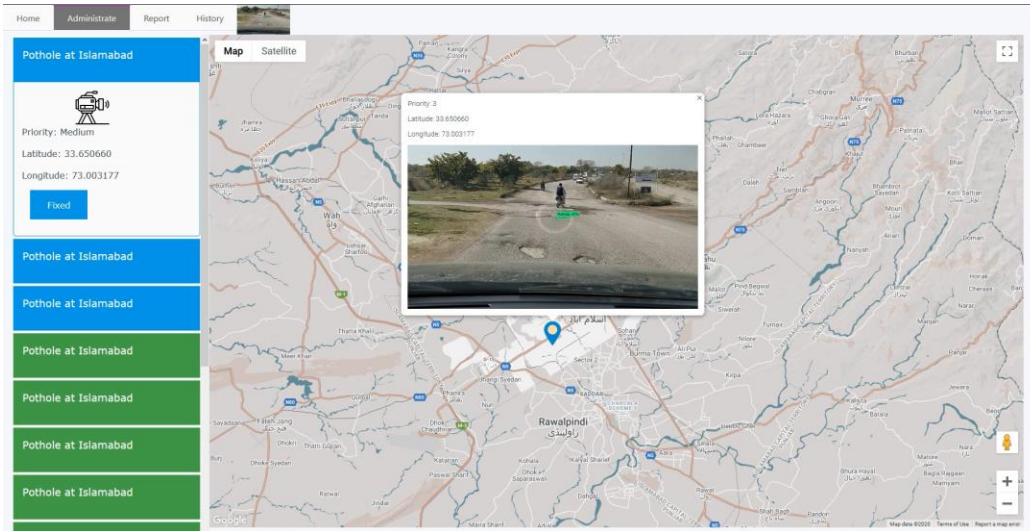


Figure 6 Administrate page, showing a reported road damage


Clicking on an item in the list expands it, giving the administrator details about it, including the source as an icon, and allows them to mark it as fixed. A map marker also appears, and clicking it pops out a small info window that contains an image that showcases the damage on the road. The list and map markers are color coded by priority, with the highest priority items appearing at the top of the list.

ID	Type	Location	Latitude	Longitude	Priority
1	Pothole	Islamabad	33.650660	73.003177	Medium
2	Pothole	Islamabad	33.652768	73.007158	Processing
3	Pothole	Islamabad	33.653123	73.007847	Medium
4	Pothole	Islamabad	33.649606	73.001154	Low
5	Pothole	Islamabad	33.652877	73.007375	Processing
6	Pothole	Islamabad	33.651223	73.004223	Processing
7	Pothole	Islamabad	33.652161	73.006053	Low
8	Pothole	Islamabad	33.650048	73.002018	Processing
9	Pothole	Islamabad	33.651728	73.005205	Processing
10	Pothole	Islamabad	33.653605	73.008729	Medium
11	Pothole	Islamabad	33.620138	73.003608	Insignificant
12	Pothole	Islamabad	33.617574	72.998716	Processing
13	Pothole	Islamabad	33.619048	73.001570	Low
14	Pothole	Islamabad	33.616966	72.997246	Processing
15	Pothole	Islamabad	33.621174	73.005571	Insignificant
16	Pothole	Islamabad	33.624997	73.012577	Low
17	Pothole	Islamabad	33.622076	73.006923	Low
18	Pothole	Islamabad	33.616403	72.996162	Insignificant
19	Pothole	Islamabad	33.622791	73.008618	Insignificant
20	Pothole	Islamabad	33.623836	73.010260	Insignificant

Figure 7 A list of all reports

The reports page is next, and it shows a list of all the damage reports that are currently pending or waiting for approval (Processing). The processing reports are those that are unverified, and don't show on the main Administrative panel.

4	Pothole	Islamabad	33.649606	73.001154	Low
5	Pothole	Islamabad	33.652877	73.007375	Processing
6	Pothole	Islamabad	33.651223	73.004223	Processing
7	Pothole	Islamabad	33.652161	73.006053	Low
8	Pothole	Islamabad	33.650048	73.002018	Processing
9	Pothole	Islamabad	33.651728	73.005205	Processing
10	Pothole	Islamabad	33.653605	73.008729	Medium
11	Pothole	Islamabad	33.620138	73.003608	Insignificant
12	Pothole	Islamabad	33.617574	72.998716	Processing
13	Pothole	Islamabad	33.619048	73.001570	Low
14	Pothole	Islamabad	33.616966	72.997246	Processing
15	Pothole	Islamabad	33.621174	73.005571	Insignificant
16	Pothole	Islamabad	33.624997	73.012577	Low
17	Pothole	Islamabad	33.622076	73.006923	Low
18	Pothole	Islamabad	33.616403	72.996162	Insignificant
19	Pothole	Islamabad	33.622791	73.008618	Insignificant



Priority: **Insignificant** Latitude: **33.622791** Longitude: **73.008618**

[Mark Fixed](#)

20	Pothole	Islamabad	33.623836	73.010260	Insignificant
----	---------	-----------	-----------	-----------	---------------

Figure 8 Clicking a report

Clicking on a report gives a detailed overview of it, including its source and an option to mark it as fixed, similar to the Administrative screen. Fixed items are moved to the history page.


Home

Administrate

Report

History

ID	Type	Location	Latitude	Longitude	Priority	Date
1	Pothole	Islamabad	33.622791	73.008618	Insignificant	06/08/2020, 11:50:50



Priority: **Insignificant**

Latitude: **33.622791**

Longitude: **73.008618**

Undo

Figure 9 Similar GUI as Reports screen

The History page consists of all the reports that were previously marked as fixed. Similar to the Reports page, clicking an item in the list expands it, and gives a detailed overview of the

damage. The administrator here has the power to undo an item that was previously marked as fixed.

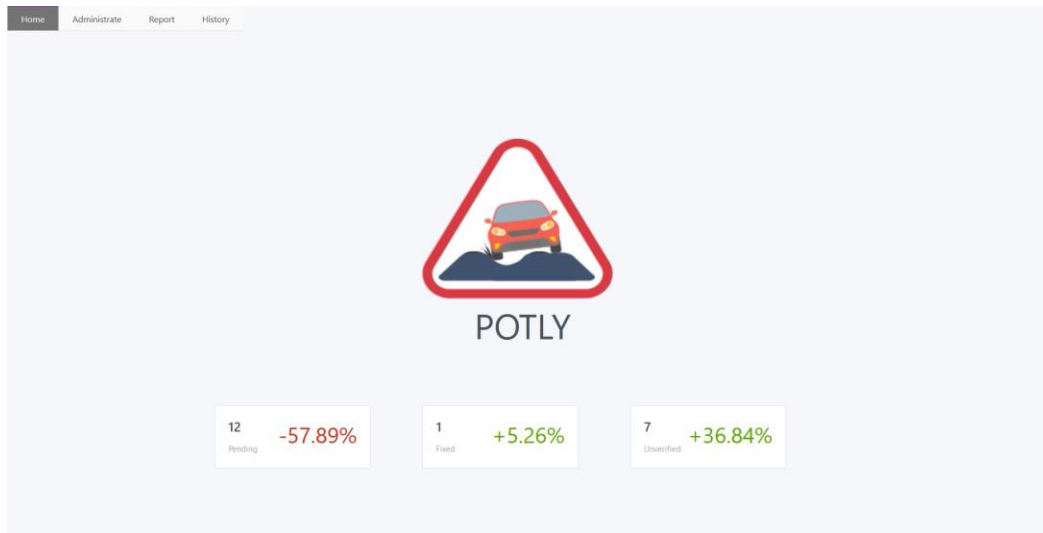
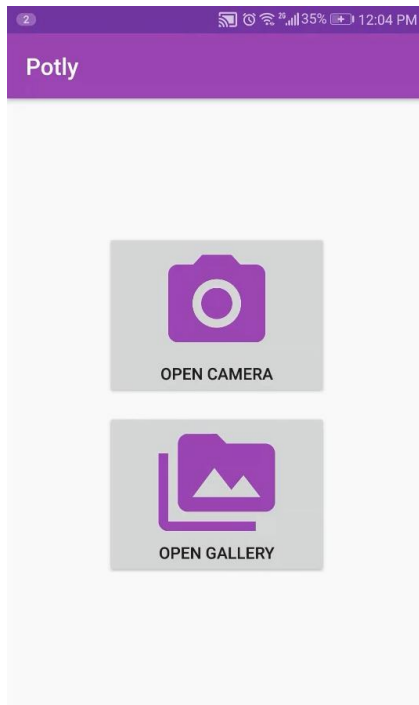


Figure 10 Updated homepage after changes are made

Android Application



The android application was built for the residents of the smart city, and allows them to report damages to the administration. They can pick an image from the camera, or pick an existing image from the gallery to send to the smart city administrators. The images are geotagged, and the model automatically extracts all details from it.

Figure 11 Android App home screen

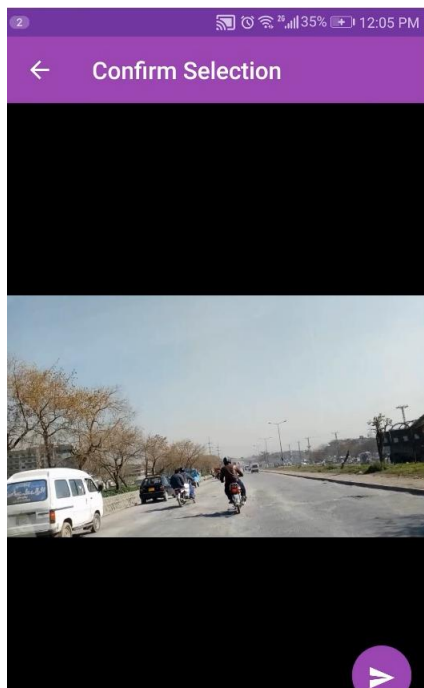


Figure 12 Review the selected image before sending it