

Lanzhou Jiaotong University  
School of Electronic and Information Engineering  
Jiuyuan Huo's research group



# Technical Report: A distributed management system for landslide prediction and classification based on WSNs and IPv6

By: Hamzah Murad Mohammed Al-Neshmi, Tom, James, Lydia  
Supervised By: Prof. Jiuyuan Huo

# **Chapter One**

## **Introduction**

## **1. Abstract**

Landslides have always been a threatening geological hazard towards humans and infrastructure. The lives and infrastructure losses are unskippable. The need for a scalable, flexible, robust system to monitor, predict and classify such hazards is indispensable. A system of hardware and software components has been developed using the latest technologies to provide a clear study and substantial help for decision-making authorities. WSNs, management websites, and machine learning models are designed to cover these needs. Sensors send data to a database, and the machine learning models study these data and provide predictions and classifications. These data are shown to the disaster prevention and management authorities in graphs and figures to help make fast responses that could save lives and protect the human-made infrastructure.

## **2. Introduction**

Due to the need for an administration interface for analyzing the data we got from the environmental field using our own IoT devices, a website has been built using the most advanced technologies available when writing this document. This project consists of many parts and layers, which will be fully described in this document. The WSNs are fully connected, and the storage server has the database for storing all the data we got in many columns and connected tables. The distributed system architecture is chosen. The distributed system contains multiple nodes that are physically separate but linked together using the network. All the nodes in this system communicate with each other and handle processes in tandem. Each of these nodes contains a small part of the distributed operating system software. All the nodes in the distributed system are connected to each other. So nodes can easily share data with other nodes. More nodes can easily be added to the distributed system i.e., and it can be scaled as required. Failure of one node does not lead to the failure of the entire distributed system. Other nodes can still communicate with each other.

Based on the used architecture, this system contains many parts, and all are pretty separated physically, i.e., database, machine learning models, and the website front-end are connected throughout the network using RESTful APIs and JSON. A RESTful API is an architectural style for an application program interface (API) that uses HTTP requests to access and use data. That data can be used to GET, PUT, POST, and DELETE data types, which refers to reading, updating, creating, and deleting operations concerning resources. An API for a website is a code that allows two software programs to communicate with each other. The API spells out the proper way for a developer to write a program requesting services from an operating system or other application. A RESTful API also referred to as a RESTful web service or REST API, is based on representational state transfer (REST), which is an architectural style and approach to communications often used in web services development. REST technology is generally preferred over other similar technologies. This tends to be the case because REST uses less bandwidth, making it more suitable for efficient internet usage. RESTful APIs can also be built with programming languages such as JavaScript or Python. A RESTful API breaks down a transaction to create a series of small modules. Each module addresses an underlying part of the transaction. This modularity provides developers with a lot of flexibility, but it can be challenging for developers to design their REST API from scratch. A RESTful API uses

commands to obtain resources. The state of a resource at any given timestamp is called a resource representation. A RESTful API uses existing HTTP methodologies defined by the RFC 2616 protocol.

Security has been taken to consideration by applying Laravel Passport, Laravel Passport provides a full OAuth2 server implementation for your Laravel application in a matter of minutes. Passport is built on top of the League OAuth2 server that is maintained by Andy Millington and Simon Hamp. There is no way to avoid the topic of RESTful APIs when building backend resources for a mobile application or using any of the modern JavaScript frameworks. If by chance you are unaware, an API is an interface or code in this case, that allows two software programs to communicate with each other. Notably, it does not maintain session state between requests, hence, you will need to use tokens to authenticate and authorize users of your application. Laravel makes building such a resource easy with a predefined provision for you to secure it appropriately.

### **3. Goals**

The main goal of developing this website is to provide a management tool and an interface to allow managers and decision-makers to have a clearer picture of the collected data and many other essential goals summarized as follows:

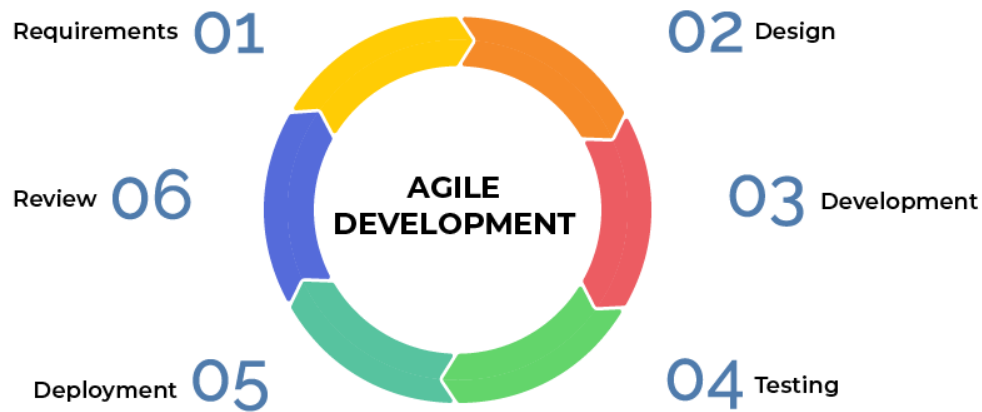
- a. Allow users to manage the system -hardware side talking-.
- b. Provide many roles, permissions, and user types.
- c. The website provides a much clearer picture of the data throughout the digrams and graphic figures.
- d. The website notifies the administration when there is an emergency or abnormal reads.
- e. The system analyzes the data and gives predictions and classifications using well-optimized machine and deep learning models.
- f. The system's design provides flexibility and availability by running on all kinds of devices and screens.
- g. The website provides a level of security as these collected data could be private.
- h. Provide an interactive interface for discussions between researchers, users, and administrators through comments.

### **4. Methodology**

Agile is the used methodology in this project. In software development, agile (sometimes written Agile) practices involve discovering requirements and developing solutions through the collaborative effort of self-organizing and cross-functional teams and their customer(s)/end-user(s). It advocates adaptive planning, evolutionary development, early delivery, and continual improvement, and it encourages flexible responses to change.

Most agile development methods break product development work into small increments that minimize up-front planning and design. Iterations, or sprints, are short time frames (timeboxes) that typically last from one to four weeks. Each iteration involves a cross-functional team working in all functions: planning, analysis, design, coding, unit testing, and acceptance testing. At the end of the iteration, a working product is demonstrated to stakeholders. This minimizes overall risk and allows the product to adapt to changes quickly. An iteration might not add enough functionality to warrant a market release, but the goal is to have an available release (with minimal bugs) at the end of each iteration. Through incremental development, products have room to "fail often and early" throughout

each iterative phase instead of drastically on a final release date. Multiple iterations might be required to release a product or new features. Working software is the primary measure of progress.



**Figure 1 Agile Methodology**

## 5. Tools

### a. Software Tools

Table 1 shows the used software used to accomplish the project.

**Table 1 Software tools**

Tool	Version	Usage
Python	3.7 and 2.7	Python has been used with the MQTT protocol to send the data, and it has been used to analyze the data (server-side)
Raspbian		The operating system used in the Raspberry pi 3
MQTT protocol		The protocol used to send the collected data from the sensors to the database
MySQL	5.7.28	The used database
Apache server	2.4.41	The used server
PHP	7.2	The language which been used to develop the logical functions of the website
PhpMyAdmin	4.9.2	Used to view the database
Laravel	5.8	Used to manage the

		backend of the website
AdminLTE		Used as the front-end of the backend.
Bootstrap	4	Used to maintain a flexible front-end
Javascript		Used to maintain a more interactive environment for the website user
HTML	5	
CSS	3	Used for providing a more interactional website
Flask		Used to send and receive data between Laravel and the machine learning models
Sklearn		A library for developing the machine learning models
ViewJS		Provide more interaction between the user and the website
Visual Stideo Code		Used as an IDE

## **b. Hardware Tools**

### **6. The research Organization**

Chapter one outlines the essential points of this project. The chapter started with an introduction and an overview of the technologies we have used to deliver this tool. Chapter two will discuss the system analysis and try to get the full description of what we are about to do, use cases, and many other diagrams. Chapter three transcribe the analysis results into a design, charts where we can get much information from, helping in the next chapter. Chapter four will discuss the implementation and the techniques that we used to build such a system.

# **Chapter Two**

## **Analysis**

## 1. Introduction

Generally speaking, such a system consists of two different parts is quite complex to analyze. Its requirements are precise and well-defined. Based on the gathered requirements, other analysis parts are easy to get and understand. Requirements are divided into two parts, functional and non-functional requirements. The functional requirements are those descriptions of the services that the software must offer. It describes a software system or its component. A function is nothing but inputs to the software system, its behavior, and outputs. It can be a calculation, data manipulation, business process, user interaction, or any other specific functionality which defines what function a system is likely to perform. Non-functional requirements

Non-functional Requirements define system attributes such as security, reliability, performance, maintainability, scalability, and usability. They serve as constraints or restrictions on the design of the system across the different backlogs.

Both functional and non-functional requirements are divided into those related to the system and the user. System requirements -also called specifications- are what the system can do. The user specifications are those what the user wants the system to do. From those requirements, we can get a vivid picture of the system and its primary functions.

## 2. Feasibility study

An estimation is made of whether the identified user needs may be satisfied using current software and hardware technologies. The purpose of feasibility is to define the potential solutions to the defined problem. The feasibility study will examine whether the system will be cost-effective and it's going to give the cost back as benefits. Our approach is essential to provide helping hands to those decision-makers in areas around dams and highways which are places most vulnerable to landslides' hazards; therefore, we can say that the system is needed in such sites so as to reduce the cost of such threat as much as possible.

## 3. Hardware Cost:

Chip	Description

## 4. Data gathering

The data is gathered through a WSNs randomly thrown in a selected site, those sensors' location is post-known using GPS, they are traced, and their location must be sent to the servers instantaneously. Sensors start sending the data to the server using MQTT protocol throw a SIM card connected to the network. Data is sent using a well-defined structure, and each sent record has all the information of the sensor which sent it and the collected data.

## 5. Sensors



We have used many sensors so as to give as many deformation factors as possible in order to make precise predictions and classifications.

Sensors	Description

## 6. The requirements specifications

### a. User requirements

#### i. Functional requirements

1. The system should collect the data of factors that cause the deformation.
2. The system should send this data to a database server that could securely store this data.
3. The system should analyze the collected data using machine/deep learning models.
4. The system should present this data using graphs and diagrams.
5. The system should have an admin panel that is ultimately a different panel than that of the users'.
6. The system should allow interaction between different kinds of users.
7. The system should allow posting reports, comments, and newsletters.
8. The system should allow all administrations to manage and control all system functions fully.
9. The system offers to sign up new users and classify them.

#### ii. Non-functional requirements

1. Speed.
2. Usability.
3. Efficiency.
4. Acceptability.
5. Security.

### b. System requirements

#### i. Functional requirements

1. The system should collect the data of factors that cause the deformation.
  - a. Sensors collect data.
  - b. Sensors send all the data and the information related to the sensor through an MQTT protocol.
  - c. The data is received by the database server and successfully stored.
2. The system should send this data to a database server that could securely store this data.
  - a. The system uses a SIM card to connect the WSNs with the other system parts.

- b. A secure connection is established with the other parts using password and username secured MQTT protocol.
  - c. The system makes sure of the received data and stores it.
- 3. The system should analyze the collected data using machine/deep learning models.
  - a. The system uses the collected data to produce a pre-processed dataset.
  - b. The system uses the new dataset as an input to the well-optimized machine/deep learning model.
  - c. Results are sent to the next stage, where they are ready to be shown to the user.
- 4. The system should present this data using graphs and diagrams.
  - a. The system gets the analyzed data as a JSON file.
  - b. The system uses the JSON file to create charts
  - c. The system shows notifications and warning alarms in case of abnormal activities.
- 5. The system should have an admin panel that is ultimately a different panel than that of the users'.
  - a. The system must have permissions, roles, and a variety of user types.
  - b. An administrator can give and take out permissions.
  - c. An administrator can review all posts and comments by other users.
  - d. An administrator can send a newsletter email.
  - e. An administrator can create new sensors, sites, users, and posts. Almost entirely manage and modify the system.
- 6. The system should allow interaction between different kinds of users.
  - a. The system allows discussions on a post using comments.
  - b. User can manage their comments
- 7. The system should allow posting reports, comments, and newsletters.
- 8. The system should allow all administrations to manage and control all system functions fully.
- 9. The system offers to sign up new users and classify them.
- 10. The system considers security as a leading factor during sharing data between different system parts using tokens, passports, and encryption.

## 7. System modeling

Usually, models are used to deal with complexity through abstraction and graphical views. We use the UML-standard modeling language in the software domain, which provides several pictorial diagrams to show the system from different perspectives. And it has been considered attractive to model complex embedded systems. This model contains one class diagram built to represent a static and dynamic aspect of the deformation prediction and classification system.

a. Website design

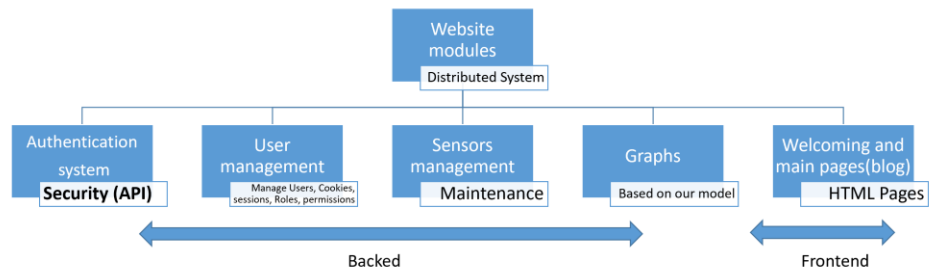


Figure 2 Website design

b. Distributed system architecture

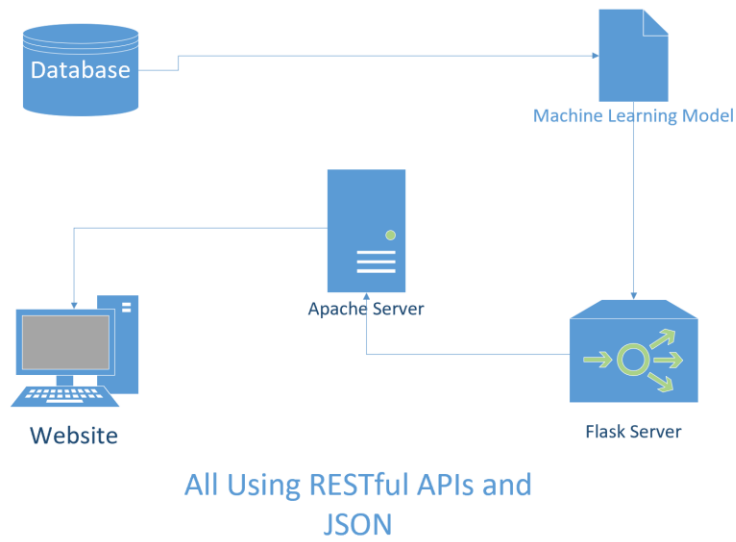


Figure 3 systems' components interaction using RESTful APIs and JSON

# **Chapter Three**

## **Design**

# 1. Introduction

As indicated in the analysis chapter about the importance of the embedded systems and the importance of the analyses stage. The design stage (System design) is no less critical than analyzing step. If the analyses stage shows and lists the functions of the system to the stakeholders in a high level of abstraction using some diagrams such as use case diagram, the design stage come to show and details how the system can meet these functions, and it is providing a clear vision for the developers and programmers. The design stage includes the architecture design and the system design. Architecture is concerned with representing the system's structure and gives a high level of abstraction view of the system. System design is concerned with showing the interaction between the system's components and the degree of coupling between them.

## 2. Hardware design

### 3. Database design

The database consists of 23 tables, and foreign keys connect some. The design is fully illustrated in figure 4.

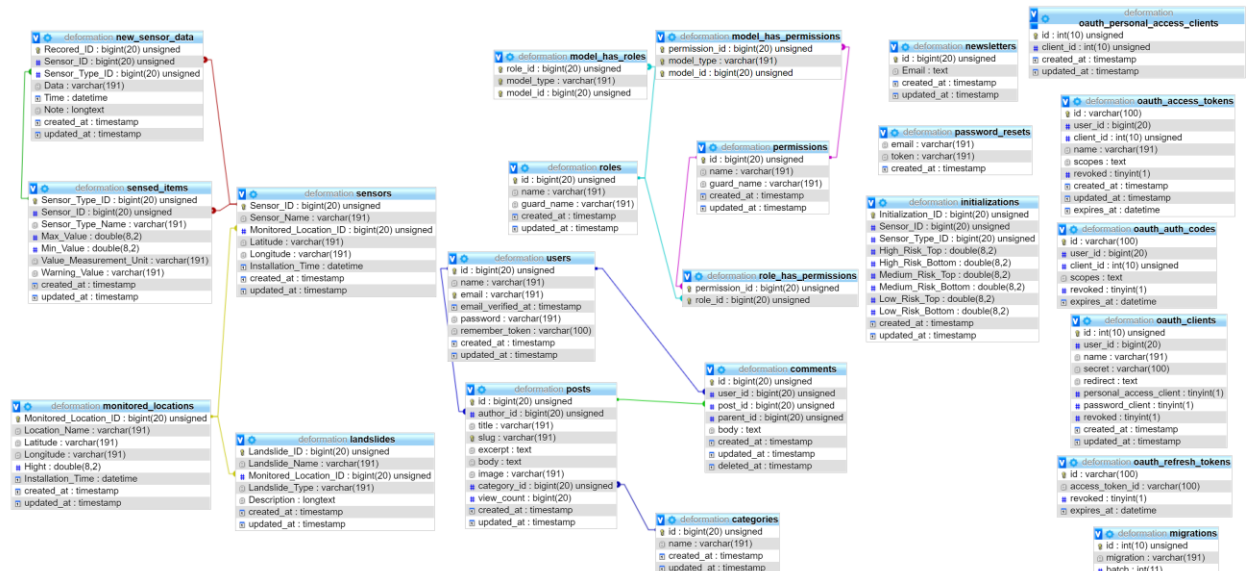
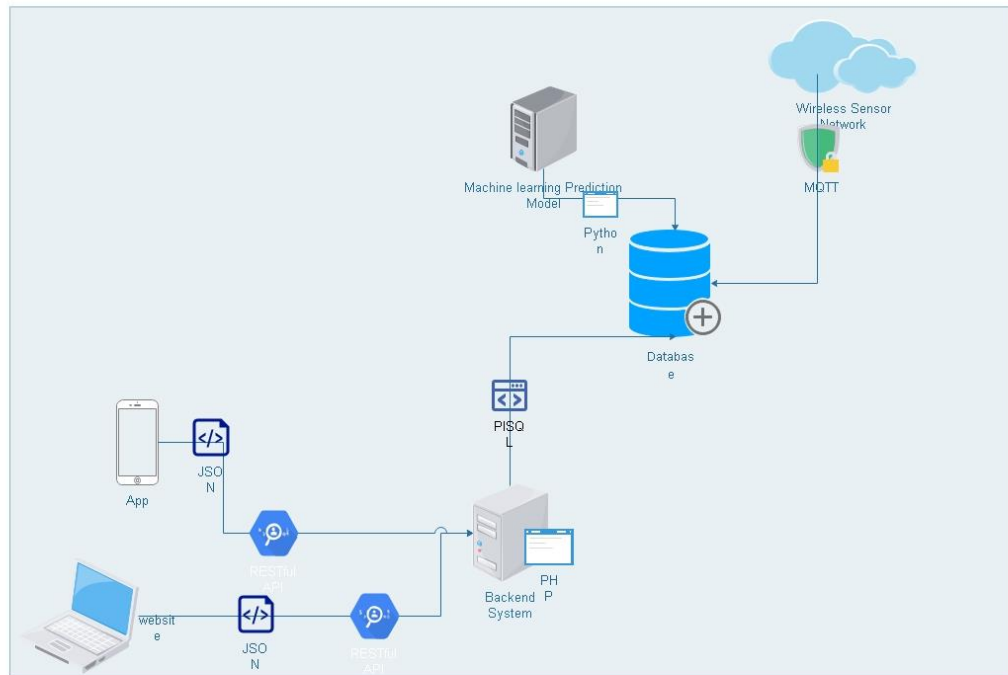


Figure 4 Database Design

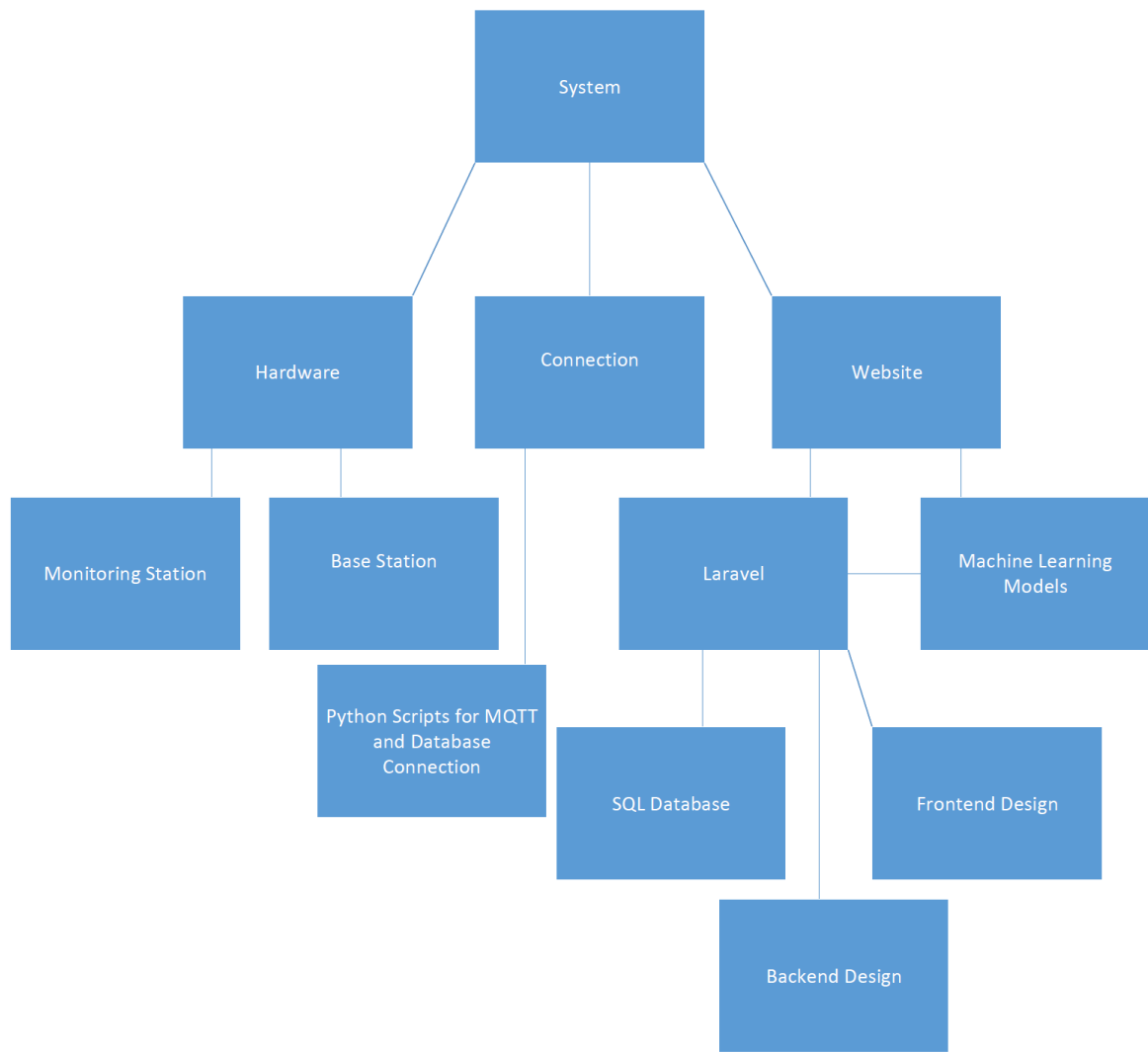
## 4. System design



**Figure 5 system architecture**

## 5. System Architecture design

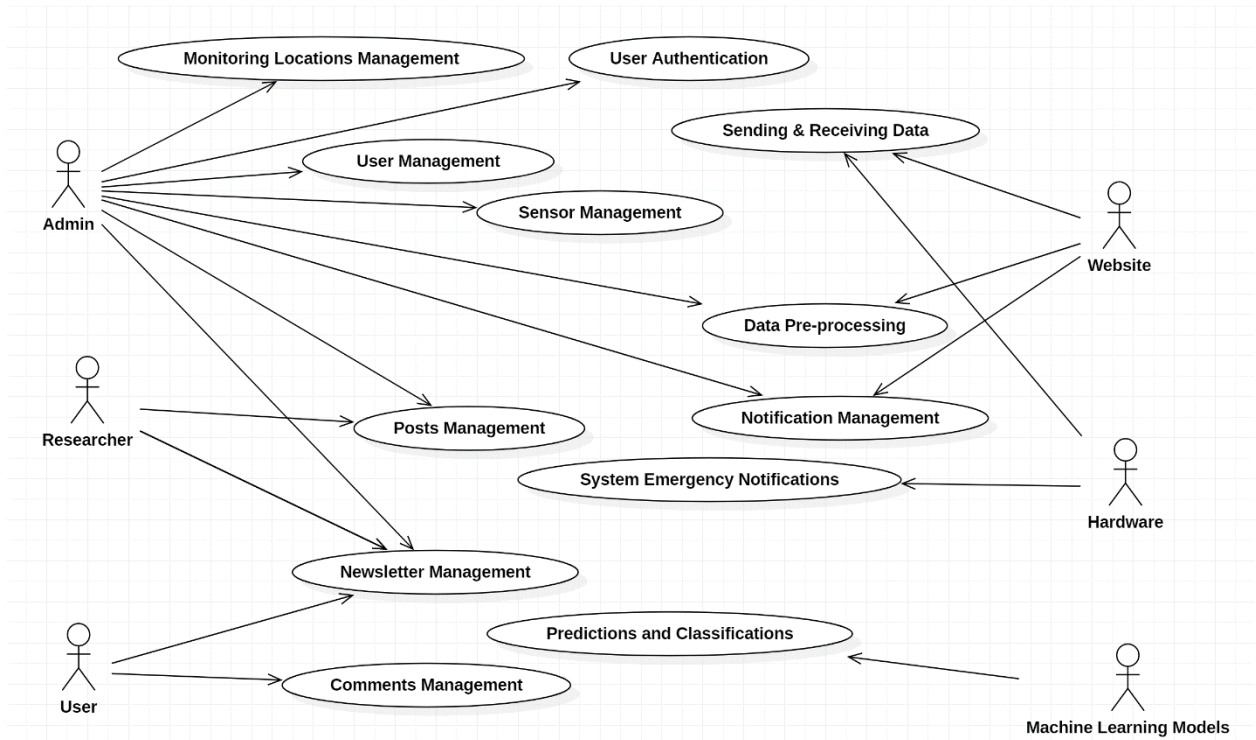
This diagram is showing the system's subsystems, to create a clear understanding of the system components and all the related relationships between the subsystems, each and every system sub-component somehow connected to one or two other components.



**Figure 6 System Architecture Design**

## **6. Use Case Diagram**

This diagram shows the system's functional specifications, which have been mentioned in the Analysis part. In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. While a use case itself might drill into a lot of detail about every possibility, a use-case diagram can help provide a higher-level view of the system. It has been said before that "Use case diagrams are the blueprints for your system".



**Figure 7 Systems Use Cases Diagram**

Here we can see that our system has four actors representing the main functional actors that interfere with the primary system's functions. A use case means each system functions, and many actors can modify one single use case.

Up next, there will be an explanation of the most critical system function, which is the prediction and classification; we can see that only the Machine learning Models actor – showed in the use cases diagram- is the one which affects the prediction and classification use case.

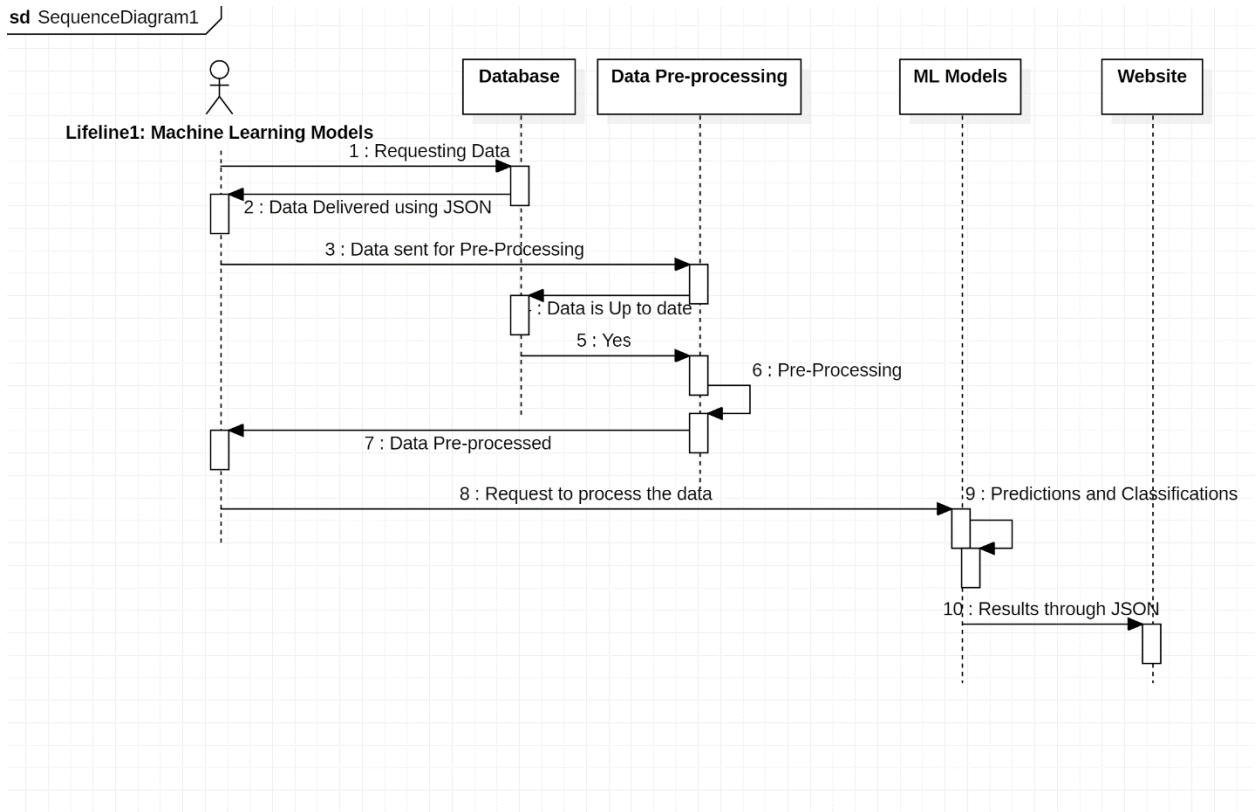
## 7. Prediction and Classification Sequence diagram

A sequence diagram depicts the interaction between objects in sequential order, i.e., how these interactions occur. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

A Sequence Diagram is an interaction diagram that details how operations are carried out -- what messages are sent and when. Sequence diagrams are organized according to time. The time progresses as you go down the page. The objects involved in the operation are listed from left to right according to when they take part in the message sequence.

In Figure 7, a sequence diagram of the most important system functions of this system which Prediction and classification machine models, is illustrated, and the time based steps are shown as well.





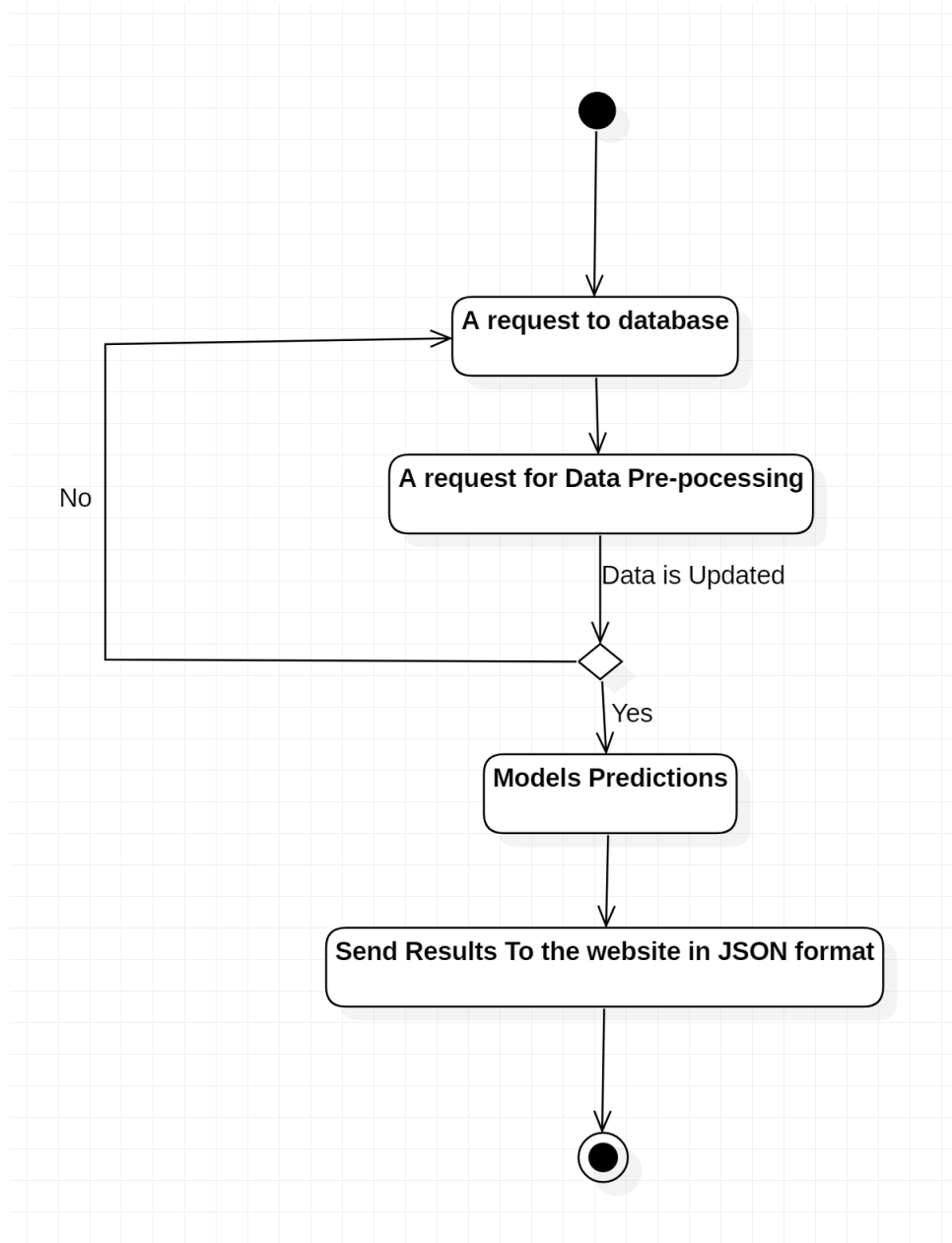
**Figure 8 Prediction and Classification Machine Learning Models Sequence Diagram**

## 8. Activity Diagram of the Prediction and Classification Machine Learning Models Use Case

We use Activity Diagrams to illustrate the flow of control in a system and refer to the steps involved in the execution of a use case. We model sequential and concurrent activities using activity diagrams. So, we basically depict workflows visually using an activity diagram. An activity diagram focuses on condition of flow and the sequence in which it happens. We describe or depict what causes a particular event using an activity diagram.

UML models basically three types of diagrams, namely, structure diagrams, interaction diagrams, and behavior diagrams. An activity diagram is a behavioral diagram i.e. it depicts the behavior of a system. An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed. We can depict both sequential processing and concurrent processing of activities using an activity diagram. They are used in business and process modelling where their primary use is to depict the dynamic aspects of a system.

Figure 8 Shows the activity diagram of the use case we mentioned before, the use case first asks for the data, if the data is updated, it'll be sent to the data pre-processing activity. As the data is pre-processed the models now can sent the results as a JSON format which will be sent to the website for making the interactive innovative charts for making decision making more easier and precise.



**Figure 9 Prediction and Classification Machine Learning Models Use Case Activity Diagram**

# **Chapter Four**

## **Implementation**

## 1. Introduction

Software development is the process of conceiving, specifying, designing, programming, documenting, testing, and bug fixing involved in creating and maintaining applications, frameworks, or other software components. Software development is a process of writing and maintaining the source code, but in a broader sense, it includes all that is involved between the conception of the desired software through to the final manifestation of the software, sometimes in a planned and structured process

Implementation is the part of the process where software engineers actually program the code for the project. Software testing is an integral and important phase of the software development process. This part of the process ensures that defects are recognized as soon as possible. In some processes, generally known as test-driven development, tests may be developed just before implementation and serve as a guide for the implementation's correctness.

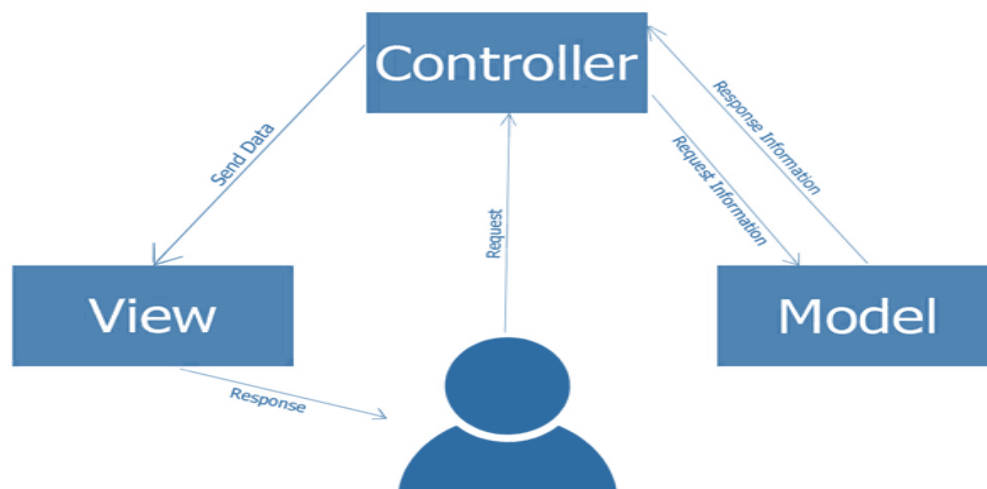
## 2. Implementation Methodology

MVC is short for Model, View, and Controller. MVC is a popular way of organizing your code. The big idea behind MVC is that each section of your code has a purpose, and those purposes are different. Some of your code holds the data of your app, some of your code makes your app look nice, and some of your code controls how your app functions.

MVC is a way to organize your code's core functions into their own, neatly organized boxes. This makes thinking about your app, revisiting your app, and sharing your app with others much easier and cleaner. The Model component corresponds to all the data-related logic that the user works with. This can represent either the data that is being transferred between the View and Controller components or any other business logic-related data. For example, a Customer object will retrieve the customer information from the database, manipulate it and update it data back to the database or use it to render data.

The View component is used for all the UI logic of the application. For example, the Customer view will include all the UI components such as text boxes, dropdowns, etc. that the final user interacts with. Controllers act as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output. For example, the Customer controller will handle all the interactions and inputs from the Customer View and update the database using the Customer Model. The same controller will be used to view the Customer data.

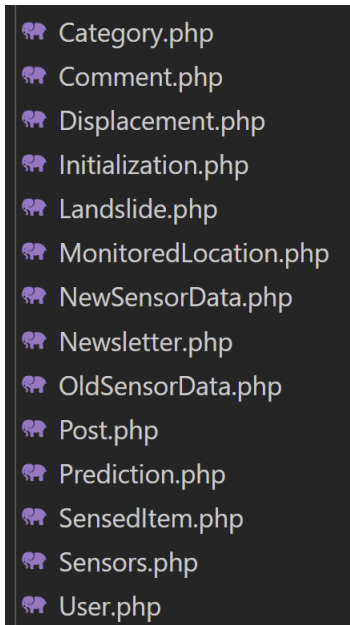
In our case, we have applied Laravel which is a web development technology was built based on the MVC pattern. This gives us even more fixable environment to implement our project and makes us finding bugs even more quickly than normal web development patterns. Having this pattern applied gives us more freedom along with the distributed systems concept to use lots of other technologies.



**Figure 10 MVC programming Technique**

### 3. MVC implementations hands-on-code

Developing the website was even an easier job having MVC applied, it separates the logic from the other parts of the project. With using Laravel, our Models, Views, and Controllers where as shown in the three following Pictures.



**Figure 11 The projects Models**

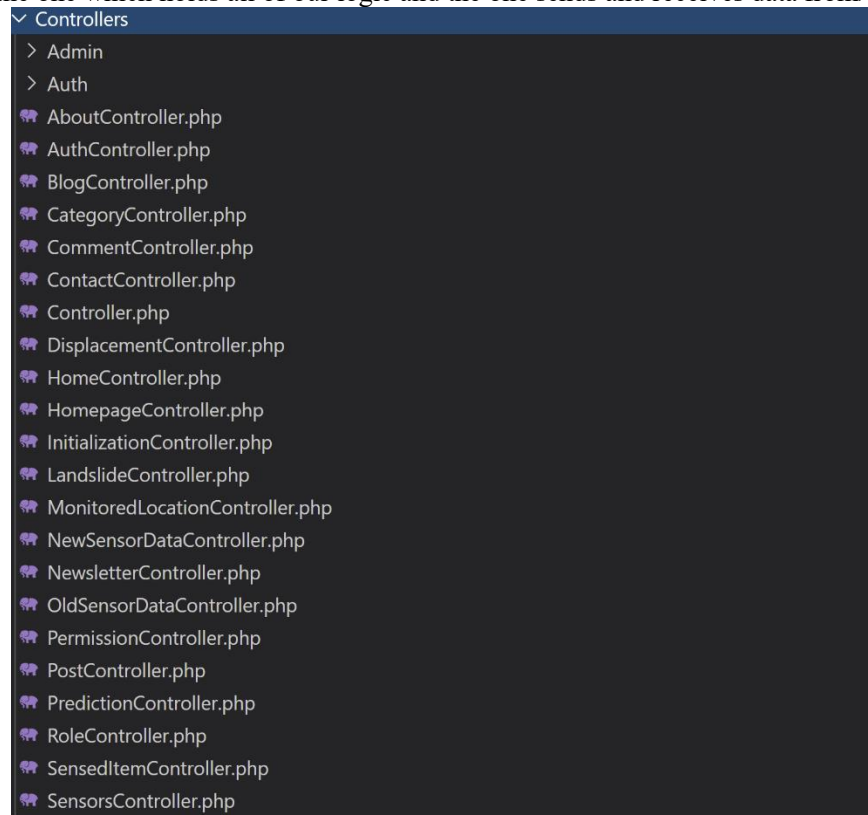
Models works as a man-in-the-middle between Controllers and Database, they are the one responsible for updating the database and the database relations between the tables, as well as saved programmer-built code. In each file shown in figure 10, we wrote the code which connects the controllers to the database.

An example of the models code is shown in figure 11, it is a PHP and Laravel blade code.

```
app > 🐘 NewSensorData.php > ...
1  <?php
2
3  namespace App;
4  use App\Sensors;
5  use App\SensedItem;
6  use Illuminate\Database\Eloquent\Model;
7
8  class NewSensorData extends Model
9  {
10     protected $fillable = [
11         'Sensor_ID',
12         'Sensor_Type_ID',
13         'Data',
14         'Time',
15         'Note'
16     ];
17     protected $primaryKey = "Recored_ID";
18     public function sensor(){
19         return $this->belongsTo('App\Sensors', 'Sensor_ID');
20     }
21     public function type(){
22         return $this->belongsTo('App\SensedItem', 'Sensor_Type_ID');
23     }
24 }
```

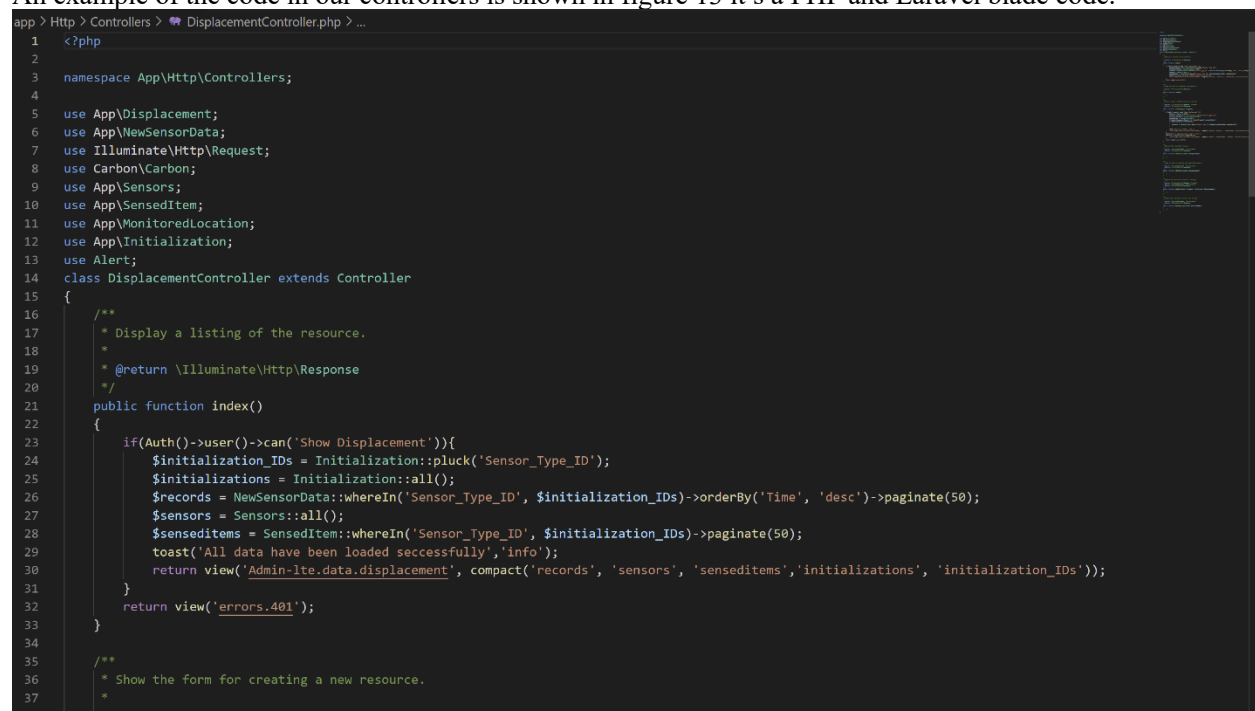
**Figure 12 Models Code Example (Newsletter Model)**

Controllers are the one which holds all of our logic and the one sends and receives data from/to views.



**Figure 13 Controllers**

An example of the code in our controllers is shown in figure 13 it's a PHP and Laravel blade code.



**Figure 14 Controller Code Example**

At the end we have the views which are the ones responsible for the interaction with the users. Figure 14 shows a small example of the code we wrote, it's HTML, CSS, JavaScript, Bootstrap, ViewJS and Laravel blade code.

```

resources > views > Admin-lte > blog > post > create.blade.php > ...
1 @include('Admin-lte.partials.header')
2 <div class="wrapper">
3     @include('Admin-lte.partials.nav')
4     @include('Admin-lte.partials.slider')
5
6
7 <!-- Content Wrapper. Contains page content -->
8 <div class="content-wrapper">
9     <!-- Content Header (Page header) -->
10    <div class="content-header">
11        <div class="container-fluid">
12            <div class="row mb-2">
13                @include('partials.alerts')
14            </div><!-- /.row -->
15        </div><!-- /.container-fluid -->
16    </div>
17 </div>
18 <!-- /.content-header -->
19 <!-- Main content -->
20 <section class="content">
21    <div class="container-fluid">
22        <div class="row">
23            <div class="col-md-12">
24                <div class="card">
25
26                    <!-- /.card-header -->
27                    <div class="card-body">
28                        <form method="post" action="/admin/adminpost" enctype="multipart/form-data">
29                            @csrf
30                            <div class="form-group">
31                                <label for="Author_name">{{ __('lang.Author_N') }}</label>
32                                <select name="Author_name" class="form-control" required>
33                                    <option value="">{{ __('lang.Select') }} </option>
34                                    @foreach($users as $user)
35                                        <option value="{{ $user->id }}">
36                                            {{ $user->id }} - {{ $user->name }}
37                                        </option>

```

Figure 15 Views Example

## 4. Script programming architecture

### a. Hardware code explanation

### b. Hardware-Database script

This script is receiving the data from the MQTT server and store it to our database. Refer to figure 15.

```

connect.py
1 import mysql.connector
2 import paho.mqtt.client as mqttClient
3 import time
4 import datetime
5 from datetime import date
6
7 def on_connect(client, userdata, flags, rc):
8     if rc == 0:
9         print("Connected to broker")
10
11         global Connected # Use global variable
12         Connected = True # Signal connection
13
14     else:
15         print("Connection failed")
16
17 def on_message(client, userdata, message):
18     #printing out
19     print("Message received: " + str(message.payload))
20     #data_processing
21     data = message.payload.decode("utf-8")
22     x = data.split("/")
23     sensor_ID = x[0]
24     sensor_Type_ID = x[1]
25     Time_Y = x[2]
26     Time_M = x[3]
27     Time_D = x[4]
28     Time_H = x[5]
29     Time_M = x[6]
30     Time_S = x[7]
31     data = str(x[8])

```

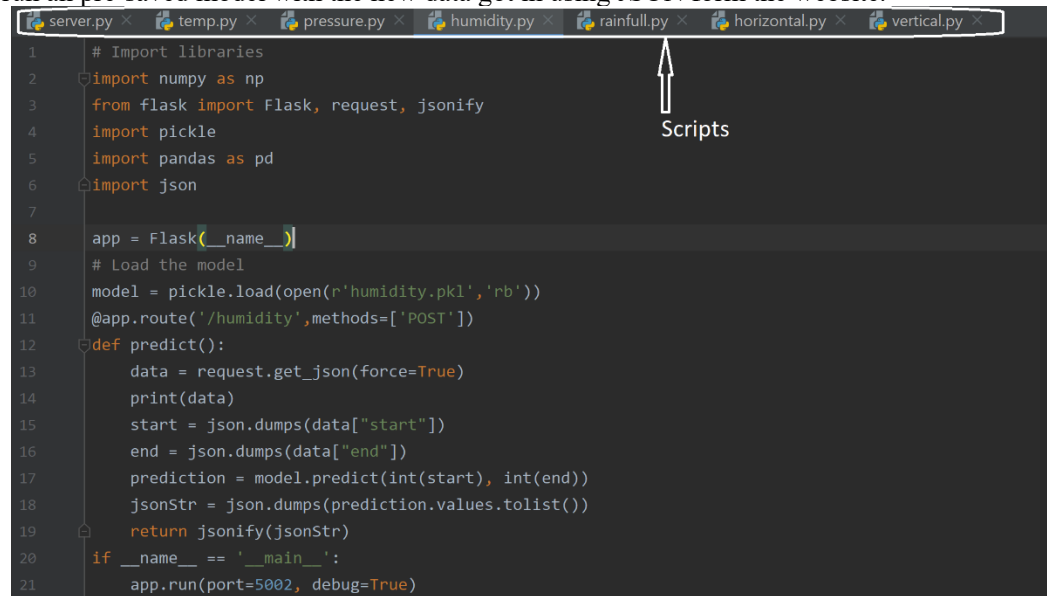
Figure 16 Hardware-To-Server Data Script

### c. Machine learning models' scripts

In these scripts, we have used flask technology to run a stand-alone server in order to run the

python-based models and send the results back to the website using JSON.

We wrote 7 scripts for each seven main factors affect the occurrence of landslides. The script run an pre-saved model with the new data got in using JSON form the website.



```
1 # Import libraries
2 import numpy as np
3 from flask import Flask, request, jsonify
4 import pickle
5 import pandas as pd
6 import json
7
8 app = Flask(__name__)
9
10 # Load the model
11 model = pickle.load(open(r'humidity.pkl', 'rb'))
12 @app.route('/humidity', methods=['POST'])
13 def predict():
14     data = request.get_json(force=True)
15     print(data)
16     start = json.dumps(data["start"])
17     end = json.dumps(data["end"])
18     prediction = model.predict(int(start), int(end))
19     jsonStr = json.dumps(prediction.values.tolist())
20     return jsonify(jsonStr)
21
22 if __name__ == '__main__':
23     app.run(port=5002, debug=True)
```

Figure 17 Flask Technology

## 5. Website Frontend

Our website has two main frontend views, one is the main website which is shown to all users and the second one is the backend view, which is shown to those with the permissions to admin and manage the websites, it shows them all the functions and gives them a full management to the website.

Let's show some of the frontend views. Beside that, the system is designed based on Bootstrap 4 and CSS3 along with the latest web development technologies. The website is totally responsive, meaning that, the html pages can adapt all screens including phones screens to bug TV screens.

### a. Main page

Here in this main website page, we can see all the important news as categorized posts, it has a slider for the most important news and a place where visitors can read the news of the posts. The footer contains lot's of information about the project as well as the newsletter subscription part.



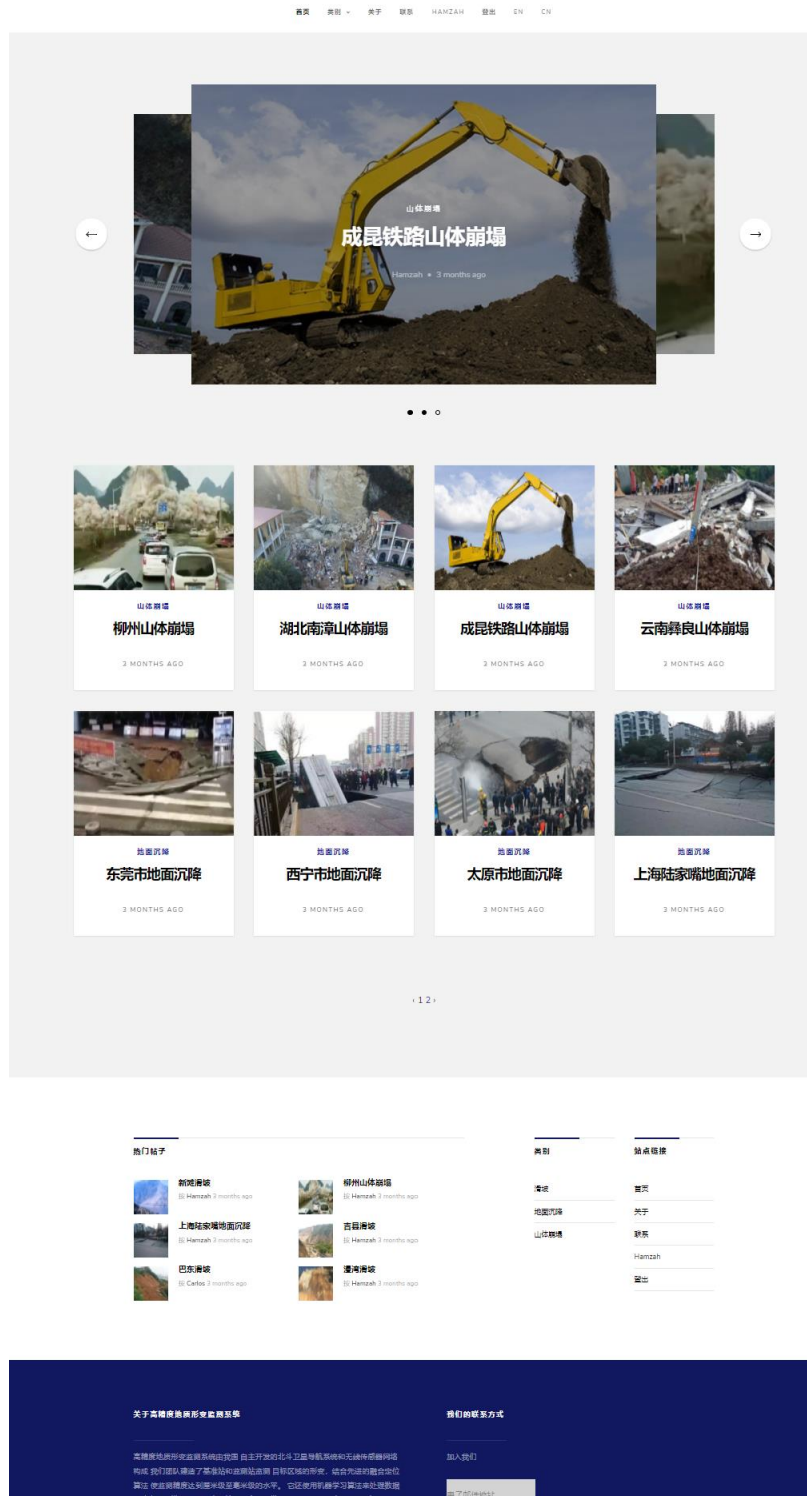
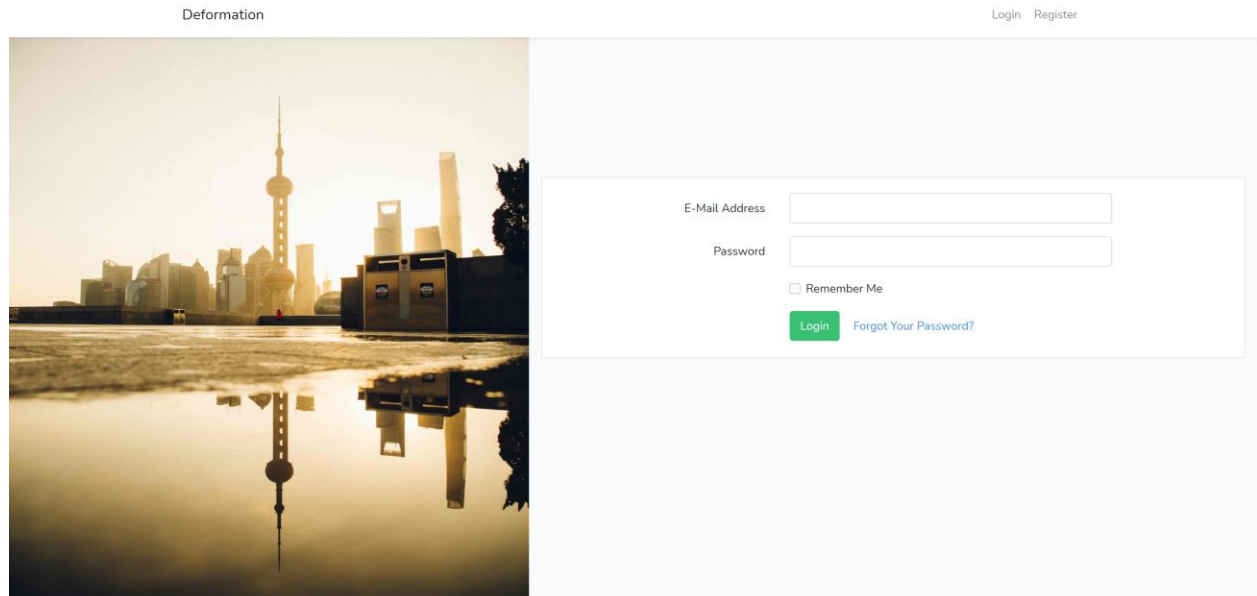


Figure 18 Main Page

b. Login Page

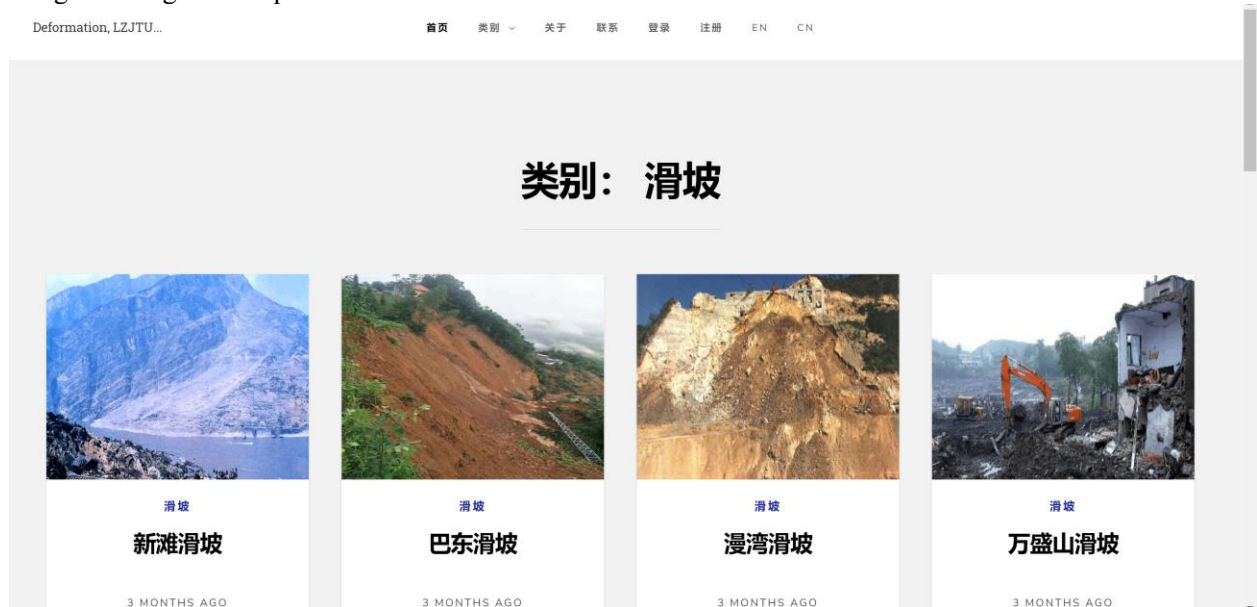
In this page, users can login. This page is designed using the latest bootstrap version available at the time of writing this document. Meaning that, it is designed for different sizes of screens.



**Figure 19 Login Page**

c. **Post Categories Page**

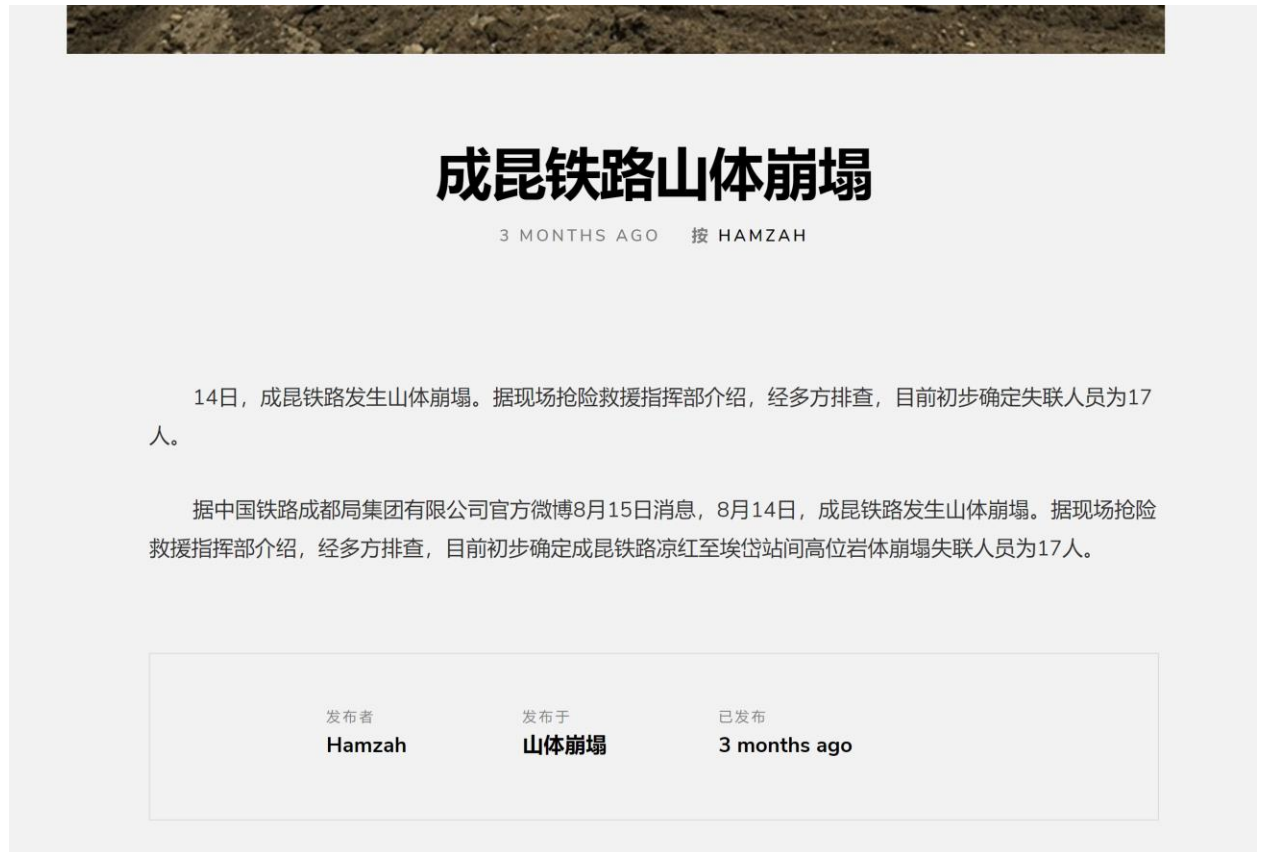
User can easily browse the post posted by different users by category, the posts are displayed with the following information, author, picture, post title and easy to read date. This interface is also designed using bootstrap.



**Figure 20 Post Categories Page**

d. **Post and Comments Page**

In this page, users can interact with the posts by writing comments of their opinions, they even can open a discussion, However, they have to be registered in order to write a comment.



**Figure 21 Post Page**

e. Backend main page

In this page, lot's of short cuts and diagrams are shown to the admin, the sensors and sensed items, posts and data, more and more of short cuts. Analytical numbers of the system's main functions and charts of the machine learning predictions all are shown in order to make the process of managing the system easier. The admin can also use the left-side panel to control the look of the website, size of the fonts and many more options.

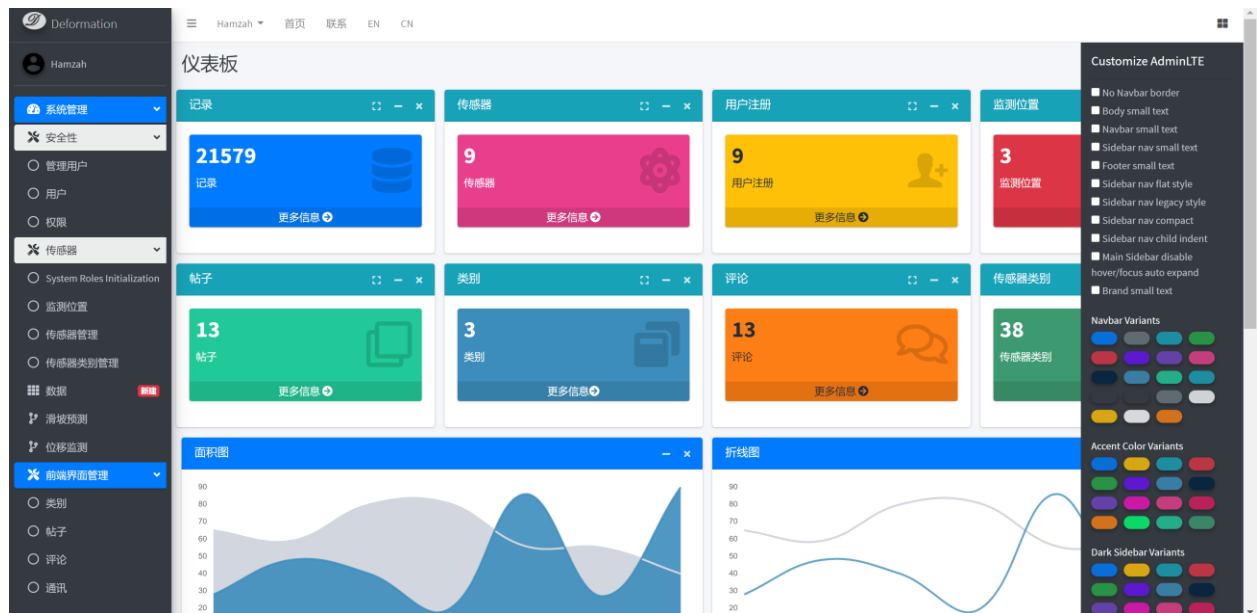


Figure 22 Backend Front page

#### f. User management

In this page the admin or whoever has the permission of managing the users has the ability to edit, add, and delete users.

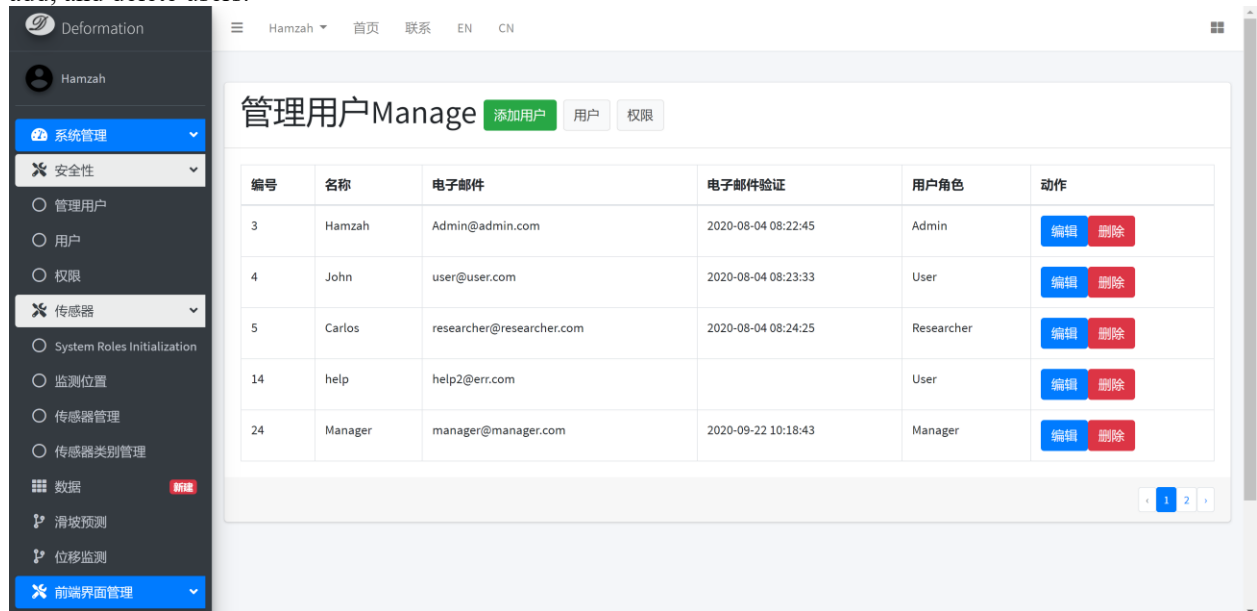
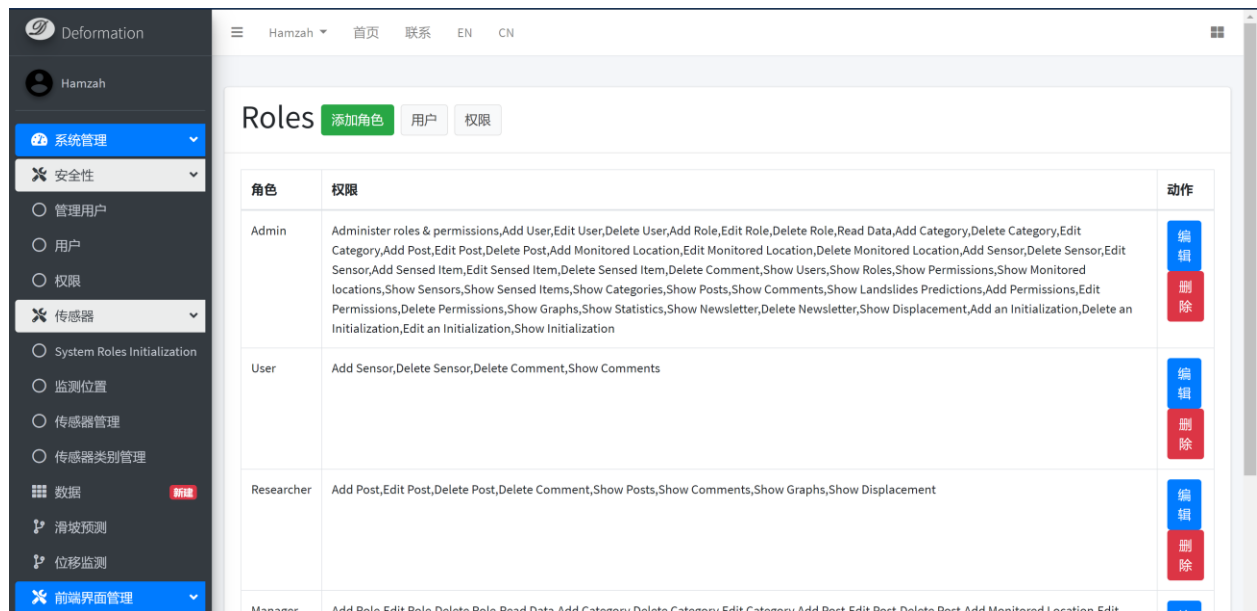


Figure 23 User Management

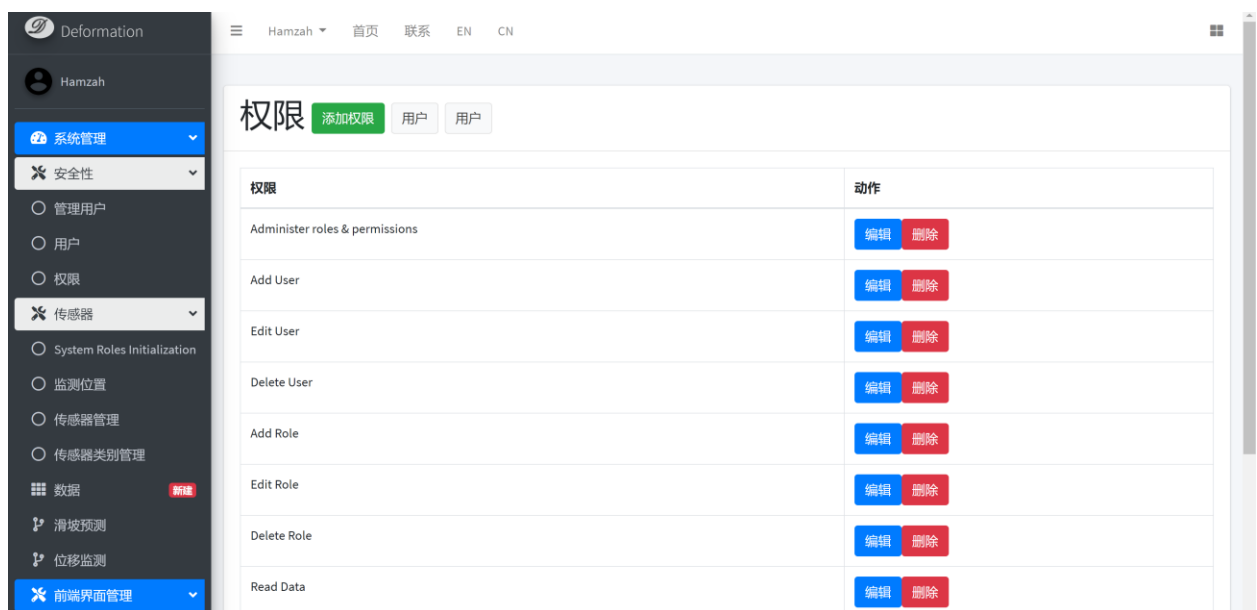
#### g. System Roles Management

In this page, the user who has the permission of controlling the roles in the system can add new roles, edit them and delete as well, here the user can assign roles to different users.



#### h. Permission Management

In this page, as a part of the user management, admin can assign the right permission to the user so that the user can do something on the website, so that a user is assigned a role, and a role has many permissions associated to it.



**Figure 24 Permission Management**

#### i. System Emergency Management

In this page the admin can set up the settings for the emergency system, many columns to fill up, limits, so when there is a new record which hits the limits, the system can react and notify the user to do what's best.

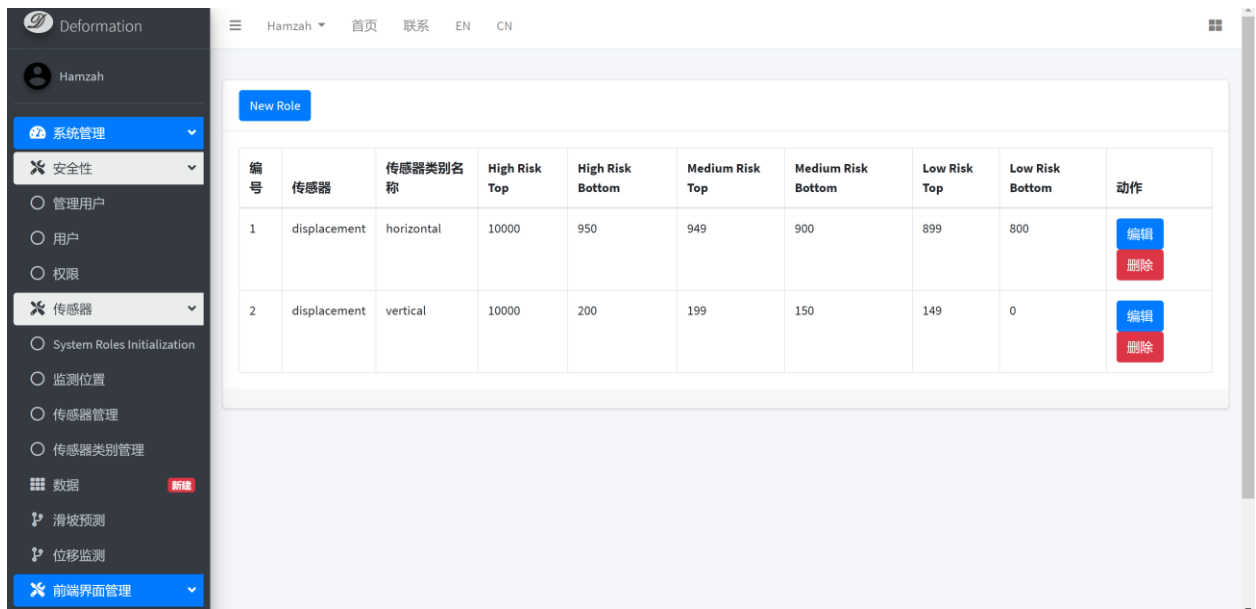
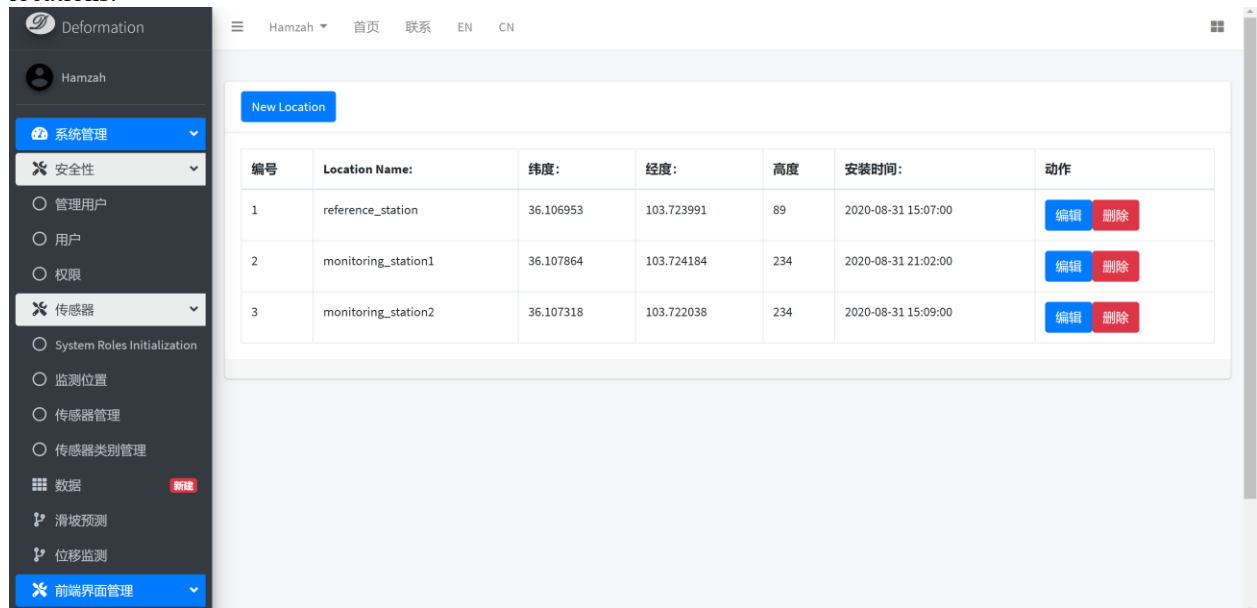


Figure 25 System Emergency Management

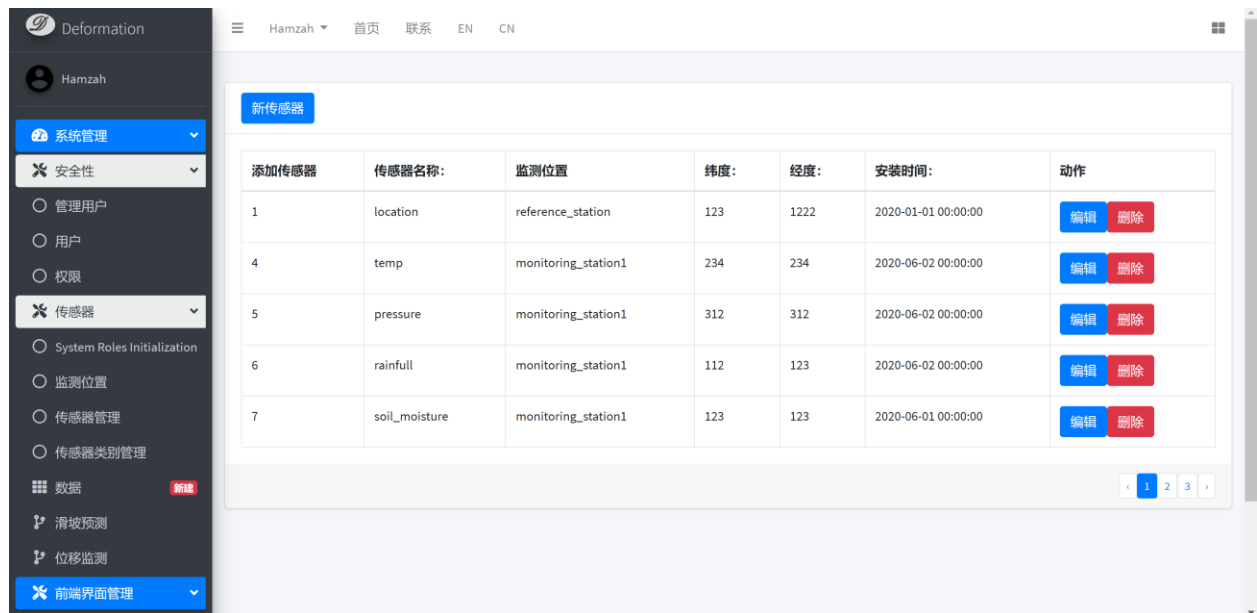
j. Monitoring Location Management

In this page the user can manage the locations and attach sensors to the mentioned locations, they can edit, and delete a location. So sensors where are all located in one place are monitoring locations.



k. Sensor Management

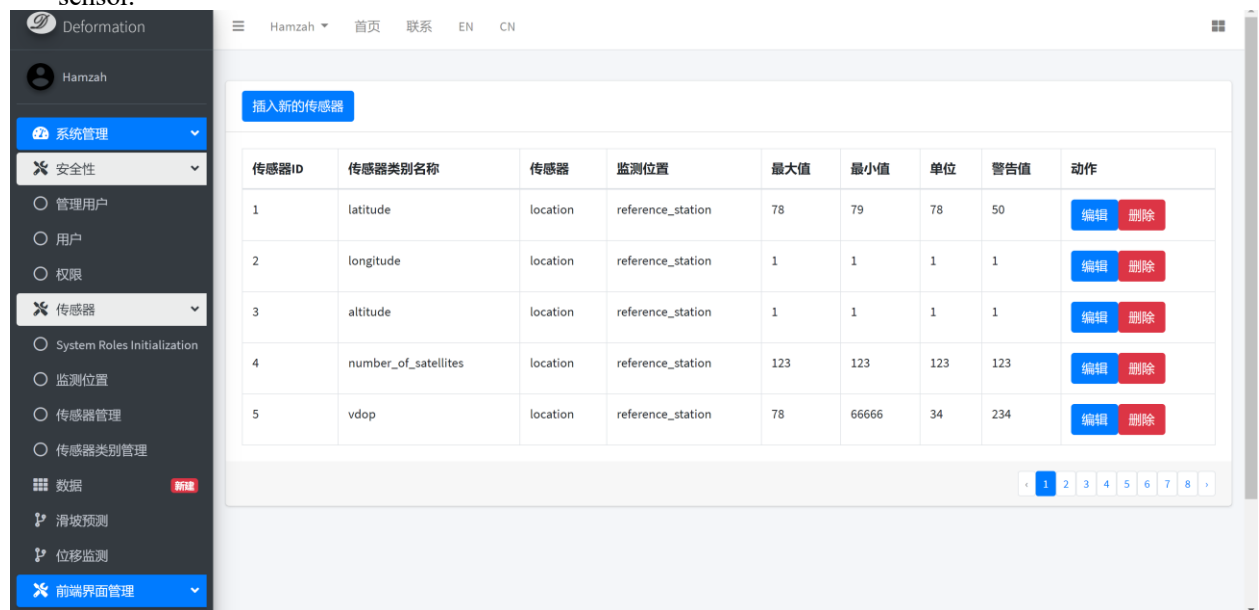
In this page the user can add the information of a new sensor, user is also allowed to edit a previously entered information on a sensor or delete it.



**Figure 26 Sensor Management**

1. Sensed Item Management

In this page, the user is allowed to add sensed items, meaning that each and every sensor has many data to gather, a location sensor as an example can send the altitude and the longitude all together, so the sensor has many different information, in this page, we can attach an sensed item to each sensor.



**Figure 27 Sensed Item Management**

m. Data management

In this page, the user can display all the gathered information as records, in each record it is obviously clear to tell the information gathered and the origin of that record, sensor, sensed item, and the monitoring location. A user can filter the data as they wish.

Deformation

Hamzah

系统管理

安全性

管理用户

用户

权限

传感器

System Roles Initialization

监测位置

传感器管理

传感器类别管理

数据

滑坡预测

位移监测

前端界面管理

Hamzah

首页

联系

EN

CN

All data have been loaded seccessfully

Filter By Sensor:

-- Select One --

Filter By Sensed Item:

-- Select One --

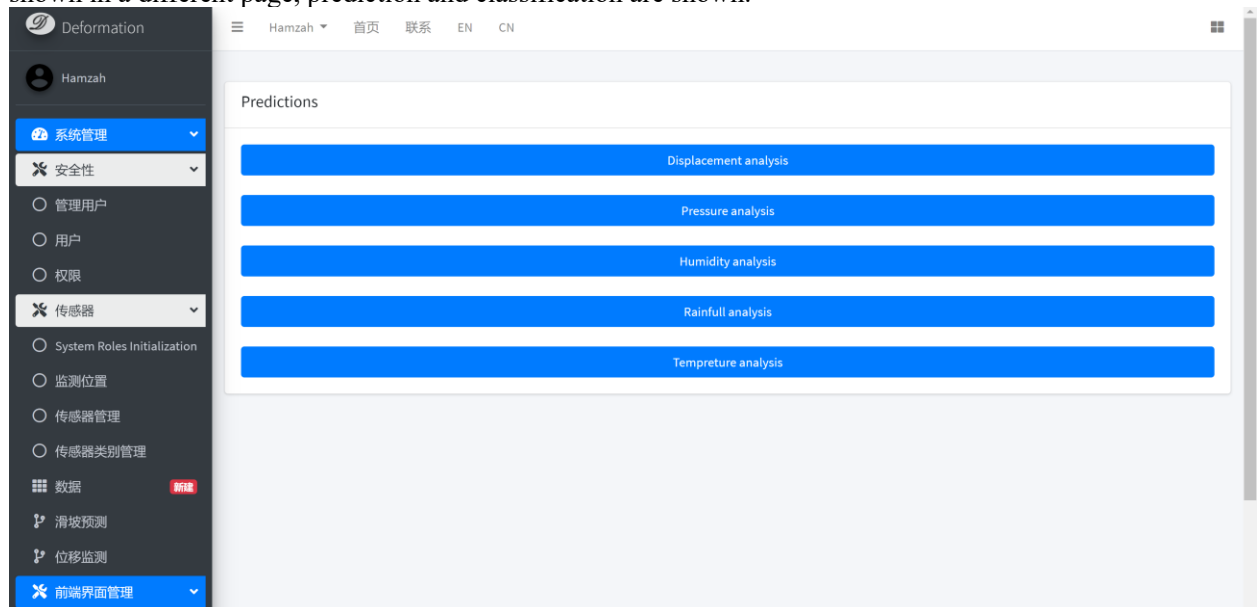
Filter

编号	位置	传感器	传感器类别	数据	时间	注意
22283	1 reference_station	1 location	5 vdop	0.9	2021-01-13 10:39:12	Nothing
22284	1 reference_station	1 location	6 hdop	1.03	2021-01-13 10:39:12	Nothing
22286	2 monitoring_station1	15 displacement	37 horizontal	1140.9173	2021-01-13 10:39:11	Nothing
22282	1 reference_station	1 location	3 altitude	1521.0	2021-01-13 10:39:08	Nothing
22278	2 monitoring_station1	15 displacement	37 horizontal	1142.0863	2021-01-13 10:39:07	Nothing
22275	1 reference_station	1 location	3 altitude	1521.0	2021-01-13 10:39:05	Nothing
22277	1 reference_station	1 location	5 vdop	0.9	2021-01-13 10:39:05	Nothing

**Figure 28 Data Management**

n. Machine Learning Predictions

In this page, the user is allowed to choose what model to apply on the gathered data, we have the displacement analysis, which run the model to analysis the location information and the results are shown in a different page, prediction and classification are shown.

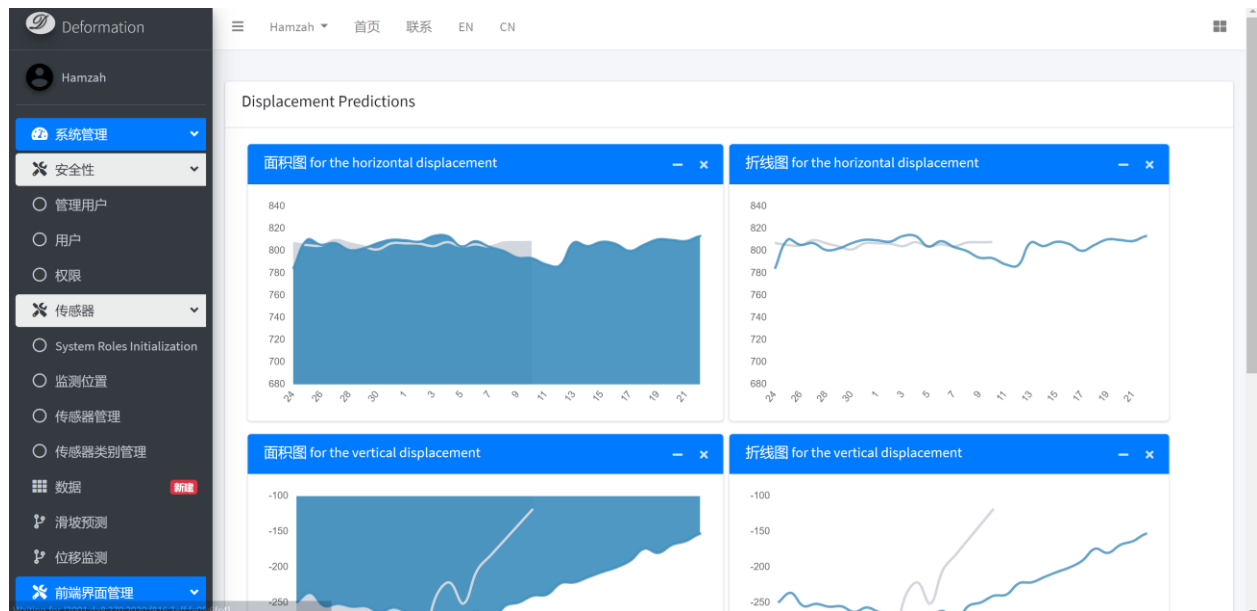


**Figure 29 Machine Learning Predictions**

o. Machine Learning models results

In this page, a user can see the machine learning models' results.





**Figure 30 Machine Learning Results**

p. Emergency data

In this page the user can clearly see the classification of each record, a user can see if there are any dangerous data which requires a specific attention.

编号	位置	传感器	传感器类别	数据	时间	注意
22286	2 monitoring_station1	15 displacement	37 horizontal	1140.9173	2021-01-13 10:39:11	High risk
22278	2 monitoring_station1	15 displacement	37 horizontal	1142.0863	2021-01-13 10:39:07	High risk
22276	2 monitoring_station1	15 displacement	37 horizontal	1141.8622	2021-01-13 10:39:03	High risk
22270	2 monitoring_station1	15 displacement	37 horizontal	1143.4447	2021-01-13 10:39:01	High risk
22266	2 monitoring_station1	15 displacement	37 horizontal	1142.5626	2021-01-13 10:38:59	High risk
22263	2 monitoring_station1	15 displacement	37 horizontal	1140.2499	2021-01-13 10:38:57	High risk
22260	2 monitoring_station1	15 displacement	37 horizontal	1144.1127	2021-01-13 10:38:55	High risk

**Figure 31 Emergency Data**

q. Post Category Management

In this page, we categorized the posts into many categories, here we can create a post category, edit the name of a category or delete a post category. In order to delete an exciting category, we have to delete the posts associated to that category first.

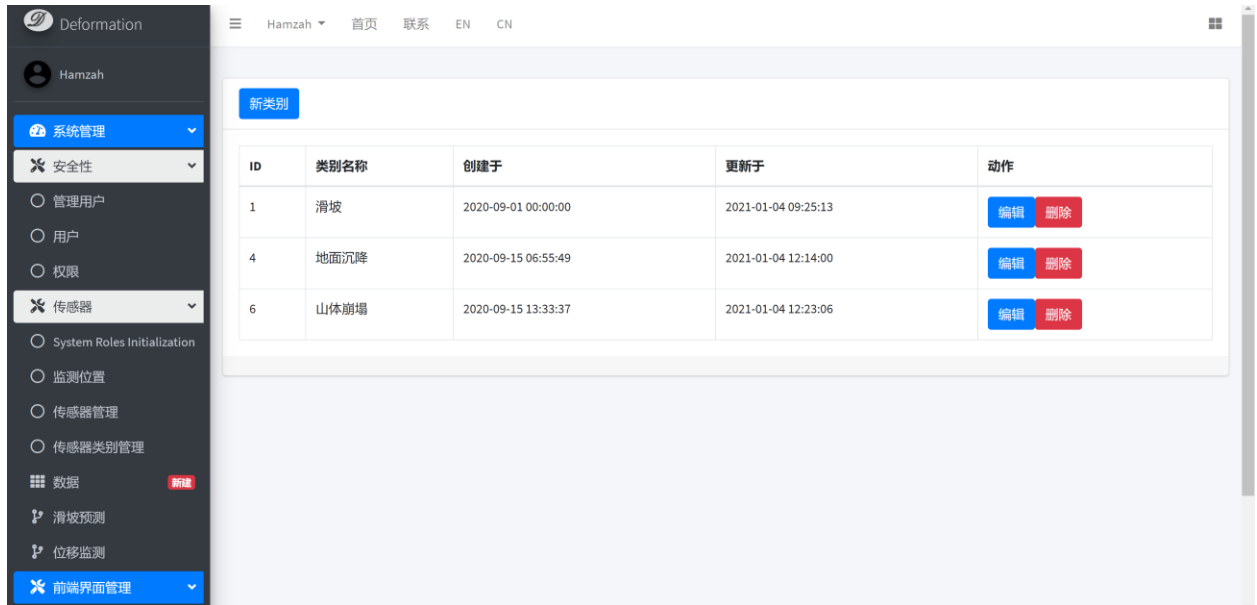


Figure 32 Post Category Management

r. Post management

In this page, a user is allowed to add a new post, this page is simply designed to show all the posts details such as the picture, title, body of the paragraph. A user can edit the post or delete it.

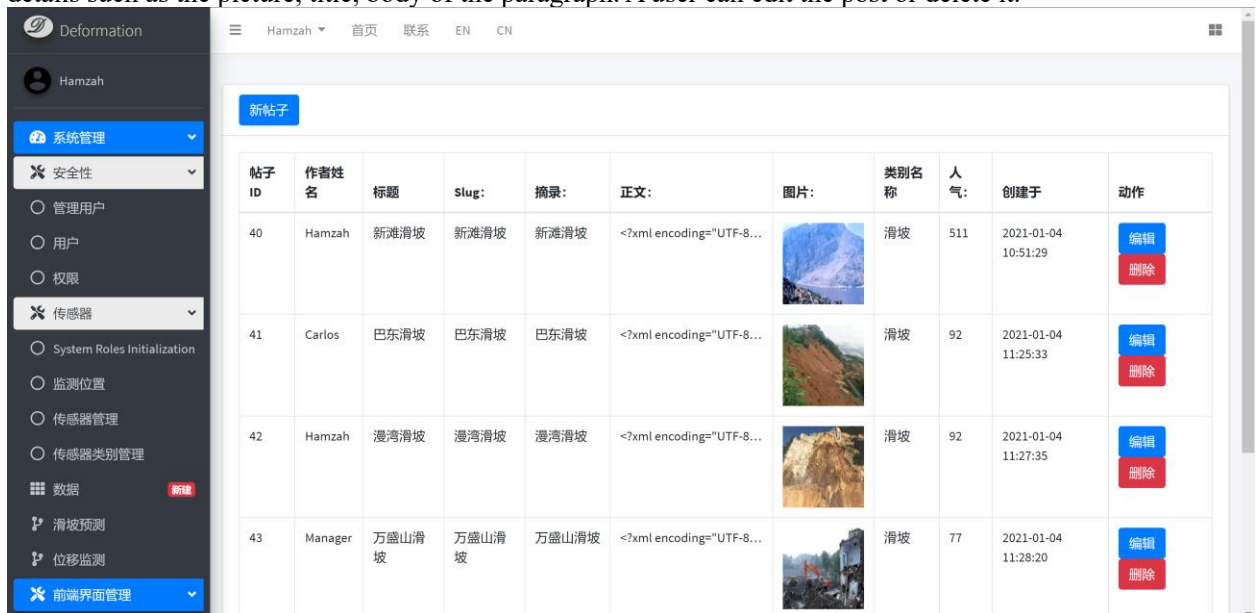
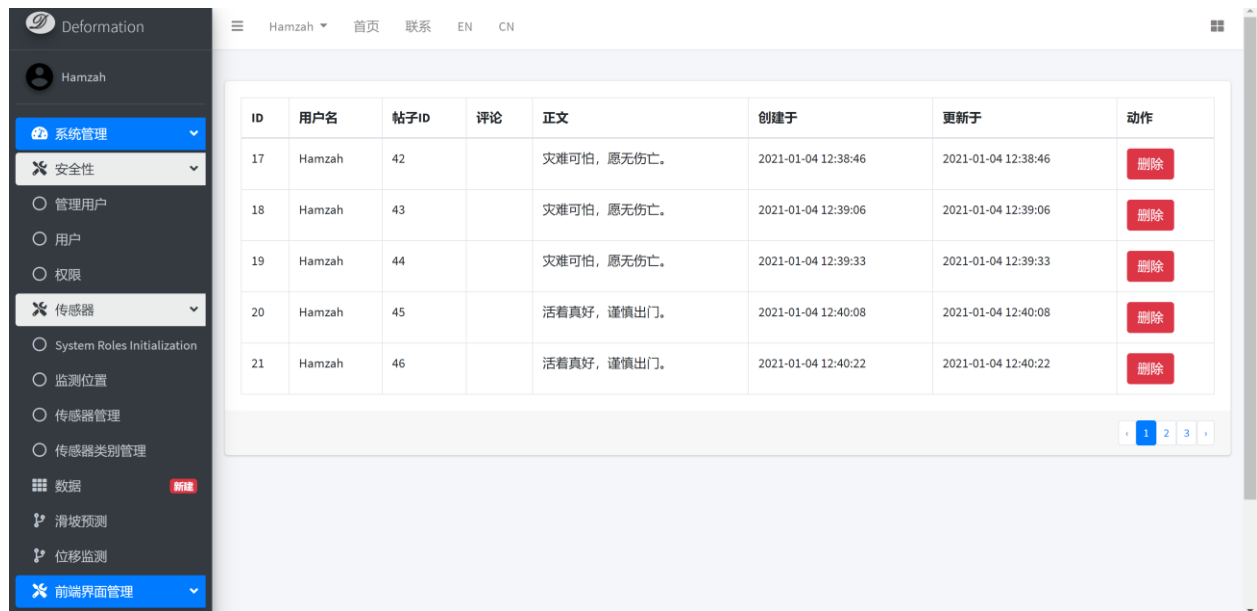


Figure 33 Post Management

s. Comments Management

Admin has the ability to delete the comments while the user can add them to a certain post.



**Figure 34 Comments Management**

## 6. Conclusion

The high-precision geological deformation monitoring system is a merged system of the Beidou satellite navigation system and wireless sensor network independently developed by our research group. Our team built a reference station and a monitoring station to monitor the target area's deformation, combined with advanced fusion positioning algorithms to achieve monitoring accuracy of centimeters to millimeters. It also uses machine learning algorithms to process data to build predictive models so as to realize early warning and prediction of geological disasters. At the same time, the system can also be connected to traditional sensors, such as rainfall sensors, soil moisture sensors, temperature, and humidity sensors, pressure gauges, and upload data to the cloud, and can perform data visualization on the host computer (Web and Android) And remote monitoring, as future work, this system will perform functions such as product display on WeChat.

