Software Design Specifications of the

“Unstable Bluff” Detection System

CS250 – Group 05

March 10, 2023

Group Members:

John Fajardo

Dennis Reardon

Mohammad Nabi Noorzai

THE SYSTEM OVERVIEW

The main goal and objective of this system are to handle the monitoring and observation over the Bluffs in the area of Delmar, California for risk assessment threat tests. At a set interval, the system will utilize multiple cameras and produce high-quality images that will be processed and stored for later review by a facility Admin who will determine the severity of the change made to the bluff and make the decision on what to do next. This is performed to avoid accidental injury to beach visitors nearby and prevent damage to trains and railway tracks.

This product will also show and provide an interface, especially for the user who always monitors bluffs or threats using the camera feeds to define and detect threat levels up to 5 that will also alert the local authorities. The Unstable Bluff system will also provide a lot of information about when it comes to each photo will be 32-bit timestamp and 32-bit geolocation and including the implication of the level of the threat 1 through 5.

THE DESIGN METHOD

This system will allow users to focus completely on the task and will integrate a database design in order to keep every month's worth of every hour's data and one photo every month that is older than one month for the whole year.

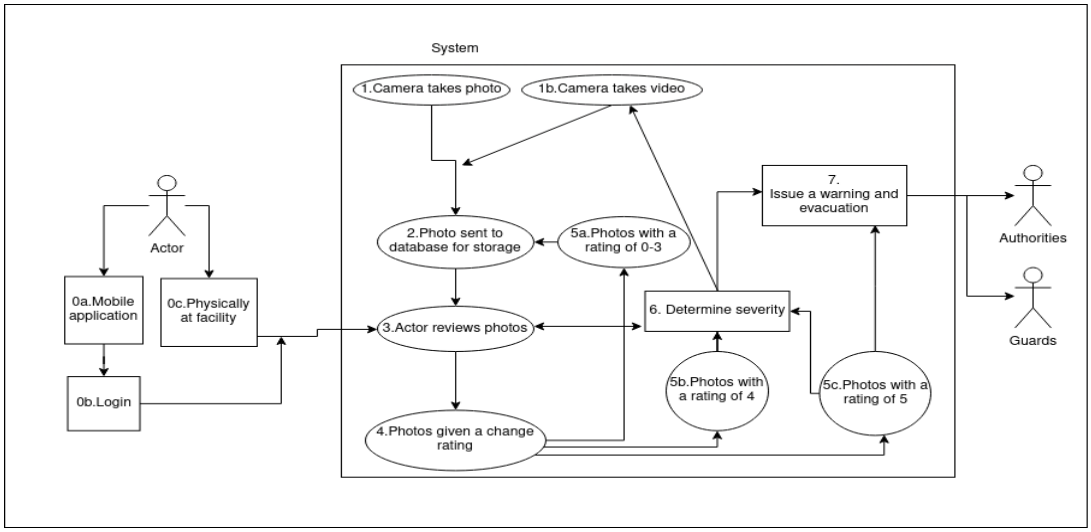
THE HARDWARE

This system will also run from a local facility test that will be able to receive all the information from the surveillance or monitoring cameras that combine with including the repeaters, routers, Wireless AP, and IP network. Amtrak will also give phone access to cellular security cameras that will transfer via the service of the cell phone data or lines.

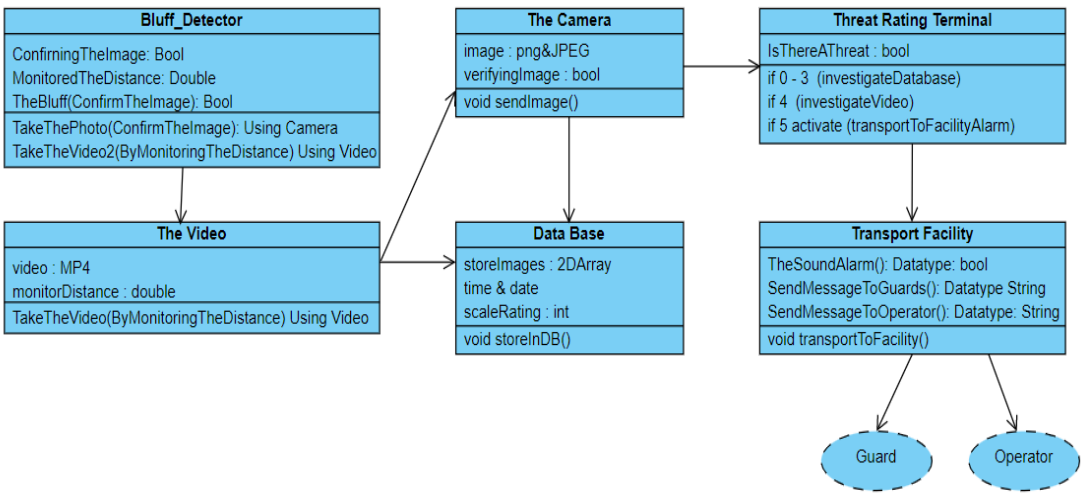
THE MISC

This is an original model test in order to make it better to establish the achievability process of the implementation of this system from the other side of the coastline of California.

THE SOFTWARE ARCHITECTURE DIAGRAM



THE UML CLASS DIAGRAM



THE INDIVIDUALITY CLASS OF THE SYSTEM

THE_BLUFF_DETECTOR

The main purpose of the Bluff detector is really to test the image whether the photo is being captured or not using the camera to see if it's a bluff. If it is determined that it is the bluff, then the photo will be captured instantly.

The_Bluff_Detector() is the one that will be utilizing the Camera() and the Video() inside its own class.

a. CHARACTERISTICS

ConfirmingTheImage: MP3 & MP4: Datatype: bool

- This will determine the Boolean variable that will really return True if the photo/image has been taken. Therefore, this variable also passed from the class of Camera.

MonitoredTheDistance: double

- This double variable will store the Distance of every bluff that has been detected and monitored by the System in feet. The maximum distance value that the camera should capture is 300 feet.

b. FUNCTIONALITY

TheBluff(ConfirmTheImage): bool

- TheBluff() is the one that will really check and capture the image to see if it is a bluff in the seaside to see as well if it goes against the algorithm, then if it proves to be True then it will return the ConfirmTheImage as a true.

TakeThePhoto(ConfirmTheImage): CAMERA

- TakeThePhoto() will be the one who will check especially when the Boolean check is true and satisfied, then it will automatically take a picture for the camera and store the photo/image inside of it.

TakeTheVideo(ByMonitoringTheDistance): VIDEO

- TakeTheVideo() will be the one that will monitor the distance of every bluff that has been detected and monitored by the System in feet and it will call the function TakeThePhoto(). This is also the class that inherits The_Bluff_Detector class.

THE CAMERA

The main purpose of the camera is to take pictures of the bluff. While The_Bluff_Dectector() is the main class that will really inherit the CAMERA in order to verify and confirm the ConfirmImage() variable and also the function TakeThePhoto(). With the help of The Transport facility class, we can determine that it will really inherit the ReadersScalingNumbers() variable that will also determine that every image that has been taken coming from the function TakeThePhoto() and then the conclusion will end up the image or photo will really store everything in the Database.

a. CHARACTERISTIC

ConfirmingTheImage: bool

- whenever a new image is received by the system, it will activate the VerifyingImage() operation which will confirm that the image received a threat scale rating from ReadersScalingNumbers() and that the image saved this value for logging purposes. This will be assigned either true or false indicating that the photo in question has been given a rating or not.

b. FUNCTIONALITY

TakeThePhoto(ConfirmTheImage): CAMERA

- TakeThePhoto(ConfirmTheImage) will check and see if ConfirmTheImage() is true. If it returns true, then the CAMERA will take a photo of file type png or JPEG depending on the file size and send that image to the transport facility system that will store that image for processing.

THE VIDEO

The main purpose of the video is to really take videos and monitor the Bluffs with a distance of 300 feet including characteristics and functionality, The_Bluff_Detector() is the one that will utilize Video() and will inherit its own class.

a. CHARACTERISTIC

MonitoredTheDistance: MP4: Datatype: bool

- The video will always be recorded and saved in the database. If any threat is detected from the bluff erosion, the image camera will be called to take pictures to determine the threat level.

b. FUNCTIONALITY

TakeTheVideo(ByMonitoringTheDistance): VIDEO

- TakeTheVideo(): the video will be recorded constantly 24/7 and stored in DataBase as MP4. Any threat that is detected by the video will call for the TakeThePhoto() function. This will inherit from its own class.

THE TRANSPORT FACILITY

a. CHARACTERISTIC

ReadingTheScale: int

- This will determine that the level threat from 1-5
- Send an alert that the risk level is 0-3 then investigate the Database.
- Send an alert when the scale level number is 4 then investigate Video.
- If the risk level is 5 it will activate the alarm to the Transport Facility and it will send an alarm to Guards and Operators.

b. FUNCTIONALITY

TheSoundAlarm(): Datatype: bool

- The sound alarm will be triggered immediately if the threat is at the maximum dangerous level which is level 5. This will call TheSoundAlarm() function, SendMessageToOperator() function, and the SendMessageToGuard() function at the same time.

SendMessageToGuards(): Datatype String

- If the threat is between 3 and 4 the system will send a message to the guard that "We have a threat, please investigate".

THE INFRASTRUCTURE AND SET UP OF THE APPLICATION OF THE BLUFF SYSTEM

Step 1:

All of the cameras installed along the bluff will need to be tested to ensure that they are properly taking photos at set intervals and when motion is detected. They will all be connected to the transport facility via the help of the installed repeaters, routers, and wireless AP necessary for the transport facility to receive them for later processing. This system will also provide a 32-bit stamp logging the time and place the photo was taken so further investigations have a clear understanding of when and where the bluff may have changed in that time.

Step 2:

The transport facility which will be housing the database will be outfitted with a wireless connection to receive the photos and videos from the camera. Amtrak will additionally provide service so mobile phones may login to the computer system located on-site and be made to have all photos and videos viewable and mark notable ones with ratings should the system detect

motion for further involvement. A technician team must confirm that cameras are both sending all photos at their set times to the database and that the database is recording the time and place of the photo. They must also confirm that logging into the system through mobile is accessible and photos can be viewed and marked with a threat level.

Step 3:

The technician team must then test the system in place for determining threat levels including reviewing a photo after a threat level has been made to confirm the system is indeed recording these changes. Once that is completed, they will need to confirm that all cases of the threat system are in working order. This includes correctly sending the photo back into storage should the rating be between 0-3, and opening the appropriate video relating to the photo reviewed should it receive a score of 4 to get further knowledge into the bluffs change. And finally, should a threat level of 5 be received by the system a message or call must be placed to authorities nearby alerting them of the danger posed to those nearby the beach and to any possible trains passing along the rail tracks. Guards will also be notified via an alarm system placed at points of importance that will alert them of the present danger.

TEST DEVELOPMENT PLAN AND THE TIMELINE

Overview and Duration of Each Phase

1. Research and planning - 1 week
2. System Design - 2 weeks
3. System Development - 8 weeks
4. Testing - 1 week

5. System Development and Maintenance for EVER

1). Researching and Planning

a. Research on the BLUFF DETECTION system and liability that there is land movement that can harm people and properties.

b. Establishment of hardware and software requirements.

c. Develop a plan including timeline, budget, and team member duties.

TASK:

a. Research on the BLUFF DETECTION system and liability that there is land movement that can harm people and properties.

b. Establishment of hardware and software requirements.

c. Develop a plan including timeline, budget, and team member duties.

ROLES AND RESPONSIBILITIES

John: Conduct research on Bluff sliding detection systems, establish hardware requirements.

Dennis: Research software design and development project plan.

Nabi: Develop plan and timeline to meet project budget and requirements.

2). SYSTEM DESIGN

TASK:

a. Collect data for requirements and specifications.

b. Software Architecture Diagrams

c. Class Diagrams

ROLES AND RESPONSIBILITIES

John: Collect data for requirements and establish documents.

Dennis: Develop a software architecture diagram with specifications and explanations.

Nabi: Develop Class diagrams with specifications and explanations.

3). SYSTEM DEVELOPMENT

TASK:

a. Bluff_Detector

b. The Video

c. The Camera

d. Data Base

e. Threat Rating Terminal

f. Transport Facility

ROLES AND RESPONSIBILITIES

John: Develop code for Bluff_detector and The Video classes.

Dennis: Develop code The Camera and Data Base classes.

Nabi: Develop code for the Threat Rating Terminal and Transport Facility classes.

4). TESTING

TASK:

a. Unit tests.

b. Functionality/Integration test

c. System test

ROLES AND RESPONSIBILITIES

John: Conduct Unit Tests and fix any issue.

Dennis: Conduct Functionality/Integration test and fix any issue.

Nabi: Conduct System test and fix any issue.

5). SYSTEM DEVELOPMENT AND MAINTENANCE

TASK:

a. Add new functionality or features when needed to add.

b. Hardware maintenance.

c. Keep the system up and running.

ROLES AND RESPONSIBILITIES

John: Add functionality or features.

Dennis: Hardware Maintenance.

Nabi: Keep the system up and running.

THE TEST PLAN VERIFICATION

a. **Functionality Test**

   Taking A Picture Test Plan:

- This function will determine the TakeThePhoto() class function of Camera. This test plan will determine whether the Camera is able to take/capture a picture of the unstable bluff especially when the bluff is being detected by The Bluff Detector System. Then if the test fails no picture will be taken and no photo will be stored in the Database.

CAMERA.TakeThePhoto(bool ConfirmTheImage)

TESTING PLAN CASE #1

- **If passed (ConfirmingTheImage):** Photo is **taken/captured**, therefore this test was successfully passed, and The Bluff Detector System is able to capture the bluff and the photo will be stored in the Database. The TakeThePhoto() function will return true.

TESTING PLAN CASE #2

- **If fail (ConfirmingTheImage):** Capturing photo **is an error,** this test was not successfully passed, and the Bluff Detector is not able to capture the bluff. The TakeThePhoto() function does not take an int variable type and it causes an error. Therefore, with this error The Bluff Detector System was not able to take a picture of the bluff, and nothing will be stored in the Database.

b. Functionality Test

Taking A Video Test Plan:

This function will determine the TakeTheVideo() class function of Camera. This test plan will determine whether the Camera is able to take a video of the unstable

bluff especially when the bluff is being detected by The Bluff Detector System. Then if the test fails no video will be taken and nothing will be stored in the Database.

CAMERA.TakeTheVideo(Double ByMonitoringTheDistance)

TESTING PLAN CASE #1

- **If the function TakeTheVideo() is activated,** then video is recorded by double the length of the distance needed to make an accurate deduction and the Bluff system in this particular area will be saved as an MP4 file for delivery to the database.

TESTING PLAN CASE #2

- **If the function TakeTheVideo() is not called** upon by its previous system Bluff Detector then it means there was not enough movement in the bluff to cause an activation of the camera causing the function to be ignored. In the event this happens the overall system will simply progress to the next stage which is taking a photo.

c. Functionality Test

Sending Alarm to The Facility Test Plan:

- This function the will determine the TheSoundAlarm() that will send and alarm to the guard or operator. This test plan also will determine that to indicate the Transport Facility to make an action when the bluff is detected that needs to take an action.

TransportFacility.TheSoundAlarm(): Datatype bool

TESTING PLAN CASE #1

- **Should the function pass and the function:** TheSoundAlarm() return the variable True, the TransportFacility system will call upon the operation transportToFacility(). In addition, calls will be made to the functions SendMessageToGuards() and SendMessageToOperator() indicating that an alarm has been triggered and immediate action must be taken to determine how severe the threat truly is.

TESTING PLAN CASE #2

- **If by chance the function returns the variable False to the function:** TheSoundAlarm(), Then the function will not activate its alarm and both the guards and the operator will not receive a message as the image did not earn a threat rating above a 3. This system will then proceed to call upon the next system called the "Treat Rating Terminal."

d. Alerting The Actors

Testing The Alarms For Guard/Operator

- TheSoundAlarm() function is False by default. When we want to test it, the threat level should be 5 then we expect it to become True. Then

TheSendMessage() function will be called and it should send messages to both Guards and Operators.

TransportFacility.TheSendMessage(): Dataype string

TESTING PLAN CASE #1

- **If passed (SendMessageToGuards):** Test case will be executed when we expect the function TheSoundAlarm() **passes** True which by default is False. Then the function: TheSendMessage() to Guard should be called and it will return or send the variable type string, “We have a threat, please investigate”. This should send a message to the guard. Same thing for Operator when the function TheSoundAlarm() **passes** True then the function: TheSendMessage() to Operator should be called and it will return or send the variable type string, “We have a threat, please investigate”.

TESTING PLAN CASE #2

- **If fail (SendTheMessageToGuards):** If in any case TheSoundAlarm() function **fails** or returns false the both TheSendMessage() functions for Guard and for Operator will not be called and they will not get the messages.

e. Threat Rating Terminal

ReadingTheScaleInt: Datatype int

- **(int ReadingTheScale):** The if conditions will determine the level of threat:

If the first if-condition is executed it will determine if the threat level is between (0, 3) it will alert the Operator to “Investigate Database”. If the second if-else condition is executed it will determine if the threat level is between (3, 4) this will alert the Operator to “Investigate the video”.

TransportFacility.ReadingTheScaleFrom(0, 5)

TransportFacilty.TheSoundAlarm: True

- **(Bool TheSoundAlarm):** If the else condition is executed it will determine the threat level (5) which is max danger or emergency level and this will call the TransportFacility.TheSoundAlarm() function.

TakeTheVideo(Double) → The_Bluff_Detector.TakeTheVideo(Double) → The_Bluff_Detector.TakeAPhoto(Bool) → The_Bluff_Detector.TheBluff(Bool) → Camera.TakeThePhoto(Bool) → TransportFacility.ReadingTheScale(int) → TransportFacility.TheSoundAlarm() → TransportFacility.SendMessageToTheGuards(Bool)

TESTING PLAN CASE #1

- If the function passed, (Double) → (Double) → (bool) → (bool) → (int) → (boo). This test was passed because the parameter function is correct that corresponds to the to each method that makes the result being successful when using the case scenario.

TESTING PLAN CASE #2

- If the function fails, (Video) → (Bool) → (Bool) → (Bool) → (int) → (Bool). This is a compile error. Because the TakeTheVideo() function method does really fed the proper parameter, So it's going to deal with the wrong data information and data type. The system will only not get the real information for a Boolean and then also the desired effect and it will not also achieve the information for the guards or rail operators.

Our Diagram is satisfactory and doesn't necessarily need to be changed. Therefore, the only need to be changed should in this Software Design Specification is all the test cases just in case.

THE DESIGN 2.0 DATA MANAGEMENT

- For the sake of the redundancy mitigation, the first diagram will really show and include the key distinction of the SQL that we will be utilizing. When it comes to the further charts, we will be using multiple databases because having them separated will be beneficial for

THE DATABASE VIDEO

- For The Database video will be the one who largely implements the SQL in order to utilize the table and also the entities while putting some additional information into the document for the storage of the video itself. SQL is a perfect fit for Video storage because it is really not required to be dynamic.

THE TABLE CAMERA - This is the operation that will really hold the Data relevant to the Camera.

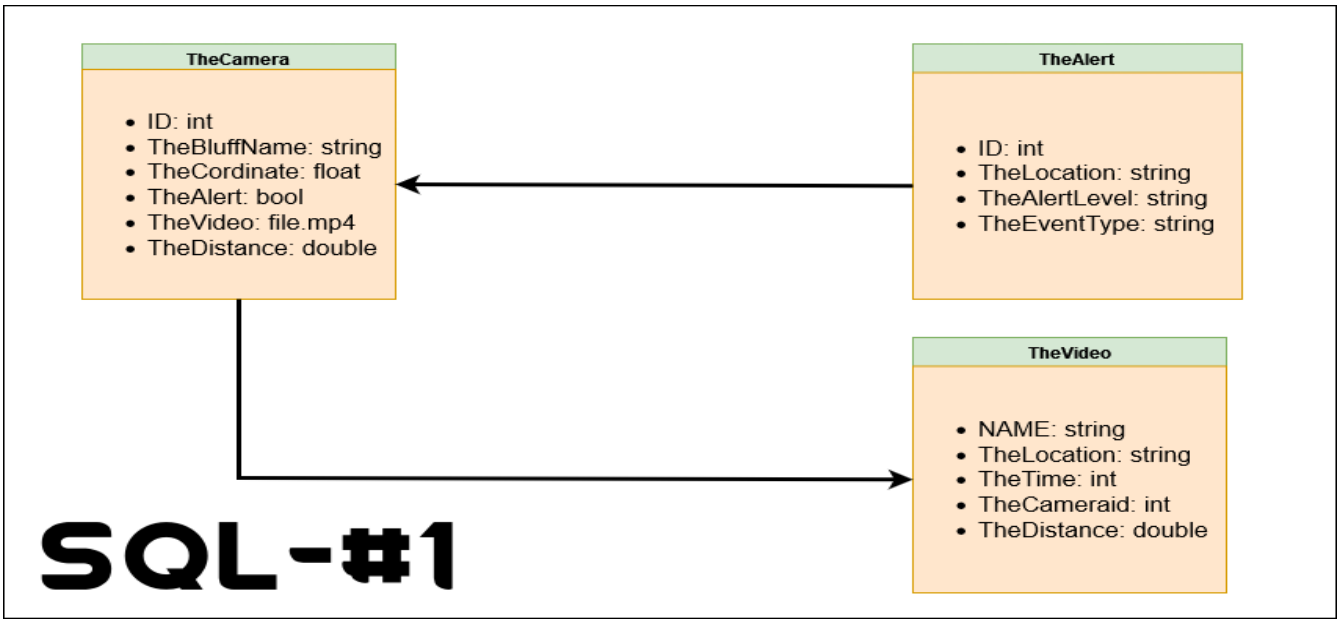
THE ENTITTIES - TheCamera ID, TheBluff Name, TheCo-ordinates (From -180 throughout 180 Geographically), The Alert Level(TheBluff_Safety), then ThePicture from TheVideo (This will taken from the corresponding TheVideo using the method call)

THE DOCUMENT VIDEO - This is the operation that will hold the data relevant to the Video files of video type format.

THE ENTITTIES: TheName: (Camera Name), TheLocation, TheTime (0:00-24:00 including with the date, dd/dd/yyyy), TheCamera ID, TheDistance (x.xx)

THE TABLE ALERT - This is the operation that will hold the data relevant for information of the TheAlert information that is being provided by TheCamera.

THE ENTITTIES - TheCamera ID: The Location, TheAlertLevel (Scale 1-5), TheEvenType (TheSafety/TheCaution/TheDanger)



THE DATABASE ALERT

- For The Database Alert operation is used to store information relating to giving alerts and crucial information when the bluff requires reviewing.

THE TABLE ALERT - For The Alert operation will contain the data for critical events, alertness level to noticeable bluff changes and the location of the bluff that resulted in the elevated alert status and the specific camera identification that located it.

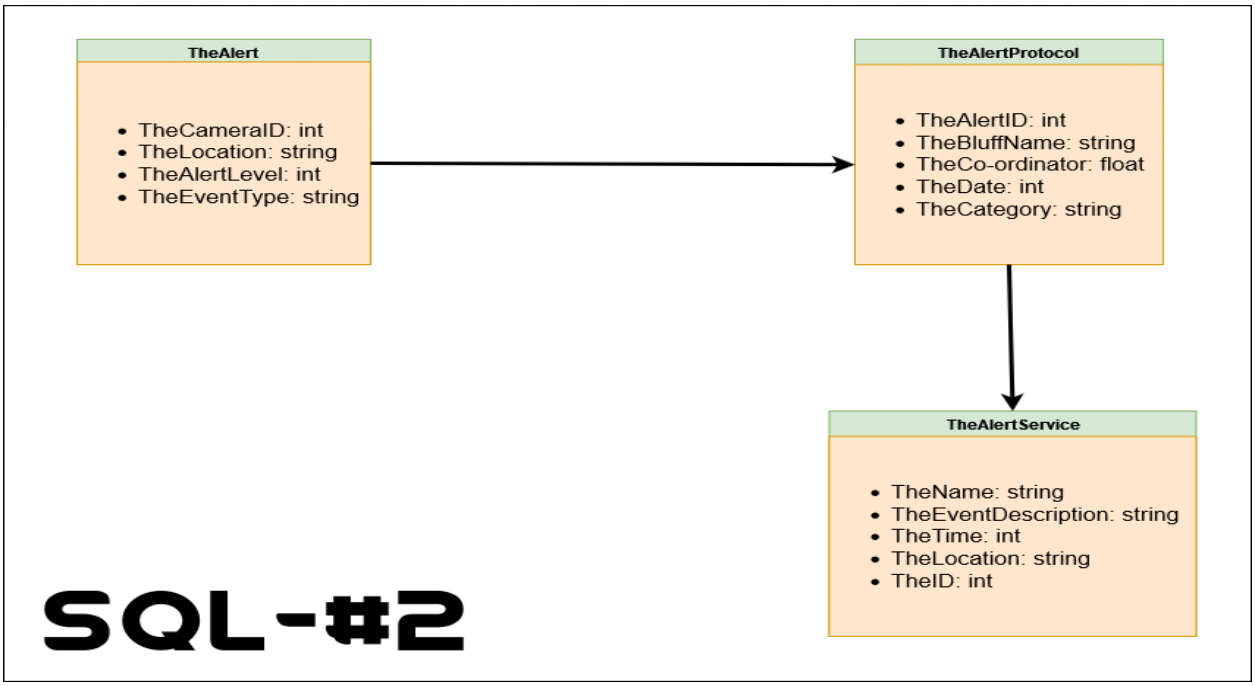
THE ENTITTIES - The Alert operation will contain the data for critical events, alertness level to noticeable bluff changes and the location of the bluff that resulted in the elevated alert status and the specific camera identification that located it.

THE ALERT PROTOCOL - The Alert Protocol operation contains the information for storing the severity and date of the bluff event including the specified name of the severity level to be concerned about. Should the severity be high enough, TheCo-ordinator will be critical in sending and receiving messages to the appropriate actors.

THE ENTITTIES - TheAlertID, TheBluffName, TheCo-ordinator, TheDate, TheCategory

THE ALERT SERVICE - The Alert service operation is needed to store the information needed when sending out messages that are being sent to actors which will go to locations along the bluff based on the severity, location and time for accurate readings.

THE ENTITTIES - TheName, TheEventDescription, TheTime, TheLocation, TheID



THE DATEBASE PERSONNEL - The personnel of the system consist of a manager, operators, clients, and guards. The personnel can create an account on the database that could be too many-to-many relationship tables. Clients additionally can be given accounts with a password to login with.

THE ENTITIES - ThePersonnelID: Name, LastName, Role and ID: username, emailAddress, and password.

THE CLIENT - The personnel of the system consist of a manager, operators, clients, and guards. The personnel can create an account on the database that could be too many-to-many relationship tables.

THE ENTITIES - ThePersonnelID: Name, LastName, Role, ID: username, emailAddress, and password.

THE EMPLOYEE – This will be a SQL because due to the fact of the nature of interaction in addition for the user addition and also subtraction.

THE ENTITIES – TheEntity ID: (TheUnique int identifier), Name(of the person), TheEmployee Number, TheActive(YES/NO)