**Software Design Specification**

**for**

**Eyewitness Website**

**Version 2.0 approved**

**Prepared by Khyati Chaudhari, William Craddock, and Aman Patel**

**Western Kentucky University**

**3/2/2022**

**Table of Contents**

[**1.**](#_vx1227) **Introduction 1**

[1.1](#_3fwokq0) Purpose 4

[1.2 Product Scope 4](https://docs.google.com/document/d/1ZN4K7yj2KTT-MUmPa1mnIQiUX_1MxKYo/edit#heading=h.1v1yuxt)

[1.3](#_4f1mdlm) Definitions, Acronyms, and Abbreviations 4

[1.4](#_2u6wntf) References 4

[1.5](#_19c6y18) Overview 4

[**2.**](#_3tbugp1) **Glossary 4**

[**3.**](#_28h4qwu) **Use Cases 4**

[3.1](#_nmf14n) Actors 4

[3.2](#_37m2jsg) List of Use Cases 4

[3.3](#_1mrcu09) Use Case Diagrams 4

[3.4](#_46r0co2) Use Cases 4

[**4.**](#_2lwamvv) **Design Overview 4**

[4.1](#_111kx3o) Introduction 4

[4.2](#_3l18frh) System Architecture 4

[4.3](#_206ipza) System Interface 4

[4.4](#_4k668n3) Constraints and Assumptions 4

[**5.**](#_2zbgiuw) **System Object Model 4**

[5.1](#_1egqt2p) Introduction 4

[5.2](#_3ygebqi) Subsystems 4

[5.3](#_2dlolyb) Subsystems Interface 4

[**6.**](#_sqyw64) **Object Descriptions 4**

[6.1](#_3cqmetx) Objects 4

[**7.**](#_1rvwp1q) **Object Collaboration 4**

[7.1](#_4bvk7pj) Object Collaboration Diagram 4

[**8.**](#_2r0uhxc) **Data Design 4**

[8.1](#_1664s55) Entity Relationship Diagram 4

8.2 Conceptual Design 4

[**9.**](#_3q5sasy) **Dynamic Model 4**

[9.1](#_25b2l0r) Sequence Diagrams 4

[9.2](#_kgcv8k) State Diagrams 4

[**10.**](#_34g0dwd) **Non-Functional Requirements 4**

[10.1](#_1jlao46) Performance Requirements 4

[10.2](#_43ky6rz) Design Constraints 4

[**11.**](#_2iq8gzs) **Supplementary Documentation 4**

[11.1](#_xvir7l) Tools Used to Create Diagrams 4

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Software Design Document | 3/1/2022 | Design Document | <2.0> |
|  |  |  |  |

# **1. Introduction**

## **1.1 Purpose**

In this document we are giving system specific requirements for the website we are building for our final project. The website is meant to help law enforcement when dealing with a case where an eyewitness has seen a suspect and helps eyewitnesses better identify suspects. Our website will be a key tool joining eyewitness information to law enforcement directly. In this document we have discussed our overall design about the project and how we are working on it to make it work. First we will talk about different use cases to explain why we have chosen the functional requirement that we have chosen. Next we will give a brief overview of the design of our system and then explain how we designed the core architecture of the system. Later on we will describe our three subsystems of the project. And give the interface explanation of those subsystems, then we will explain the objects that we are using later on and at last before giving explanation on the non-functional requirements we will give detailed explanation on the database design using ER diagram. Finally we will give all the supplemental reading.

## **1.2 Product Scope**

As mentioned in the abstract of our project, we are building this website to help witnesses better help our law enforcement with the piece of information they have about the crime or criminal. Moreover, our website will help establish a tool where witnesses can instantly help law enforcement with the case by identifying the picture of the criminal from the criminal database. Scope of the project as of now is limited to the suspect identification and not help with other case related information. But the future goal for the project will be to make it where law enforcement can use it as their primary workstation.

## **1.3 Definitions, Acronyms, and Abbreviations**

## **1.4 References**

## **1.5 Overview**

1. [Introduction](#_vx1227)
2. [Glossary](#_3tbugp1)
3. [Use Cases](#_28h4qwu)
4. [Design Overview](#_2lwamvv)
5. [System Object Model](#_2zbgiuw)
6. [Object Descriptions](#_sqyw64)
7. [Object Collaboration](#_1rvwp1q)
8. [Data Design](#_2r0uhxc)
9. [Dynamic Model](#_3q5sasy)
10. [Non-Functional Requirements](#_34g0dwd)
11. [Supplementary Documentation](#_2iq8gzs)

# **2. Glossary**

# **3. Use Cases**

In this section we will go in detail on who will be using the finished product, also who are the people who will be interacting with the website on a daily basis. Along with that there are some scenarios in which we gave an explanation why we need certain features and what these features will do to help our actors on a daily interaction with a website. Also we have given diagrams of all those use cases to help readers of this document help understand use cases a little more.

## **3.1 Actors**

As mentioned earlier this section is for the Actors. Actors are the people and machines that work together to make our product functional.

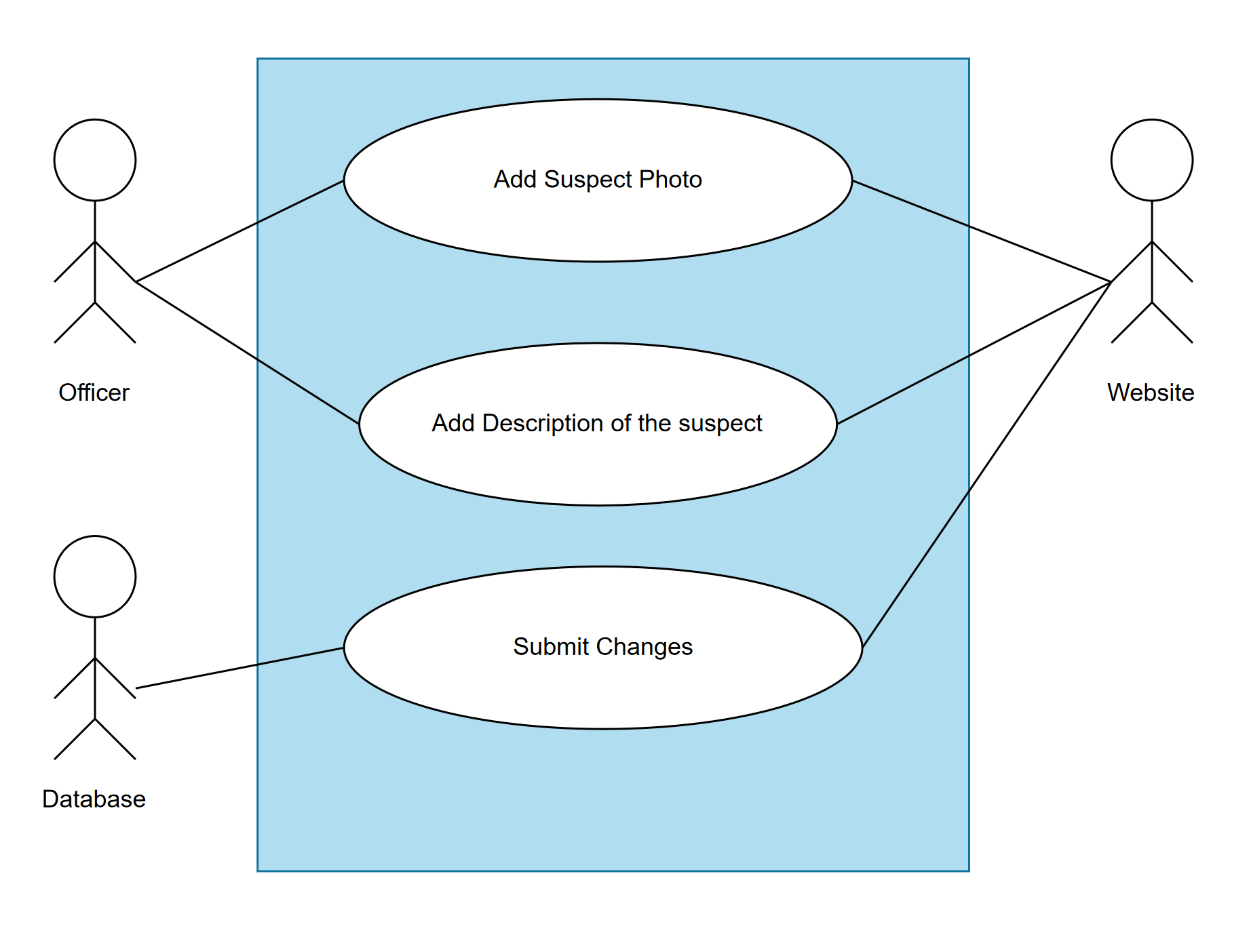
1. Police officer: Is our most important part of this product and also who will be purchasing the product from us. Police officers will have admin power on the database along with all the information. They will have the authority to add or delete photos from the database. Add or delete suspects and also cases. They would be able to create a witness and assign a case and lineup to a witness.
2. Eye Witness: This actor will be our product's end user since they are the reason the product is being built. They will get their login credentials from the police officer using which they would be able to login to their account, select a list of suspects and then submit it to the law enforcement database.
3. Database: This is not a living actor but is one of the most important ones after an officer. Database is where all the data is stored at, from officer’s login credentials to suspects photos everything is stored in the database at its right places which we will talk about in a little bit. For the development phase this database will be hosted on our local computer.
4. Web Server: This is also a machine actor, website will be hosted locally also just like a database. All the information from the user is given and verified with the database from the website and web server helps us to do so.
5. Development team: This will be us who are developing the product and maintaining it when necessary. We will be also responsible for creating new officers until all the officers are up to speed with the website.

## **3.2 List of Use Cases**

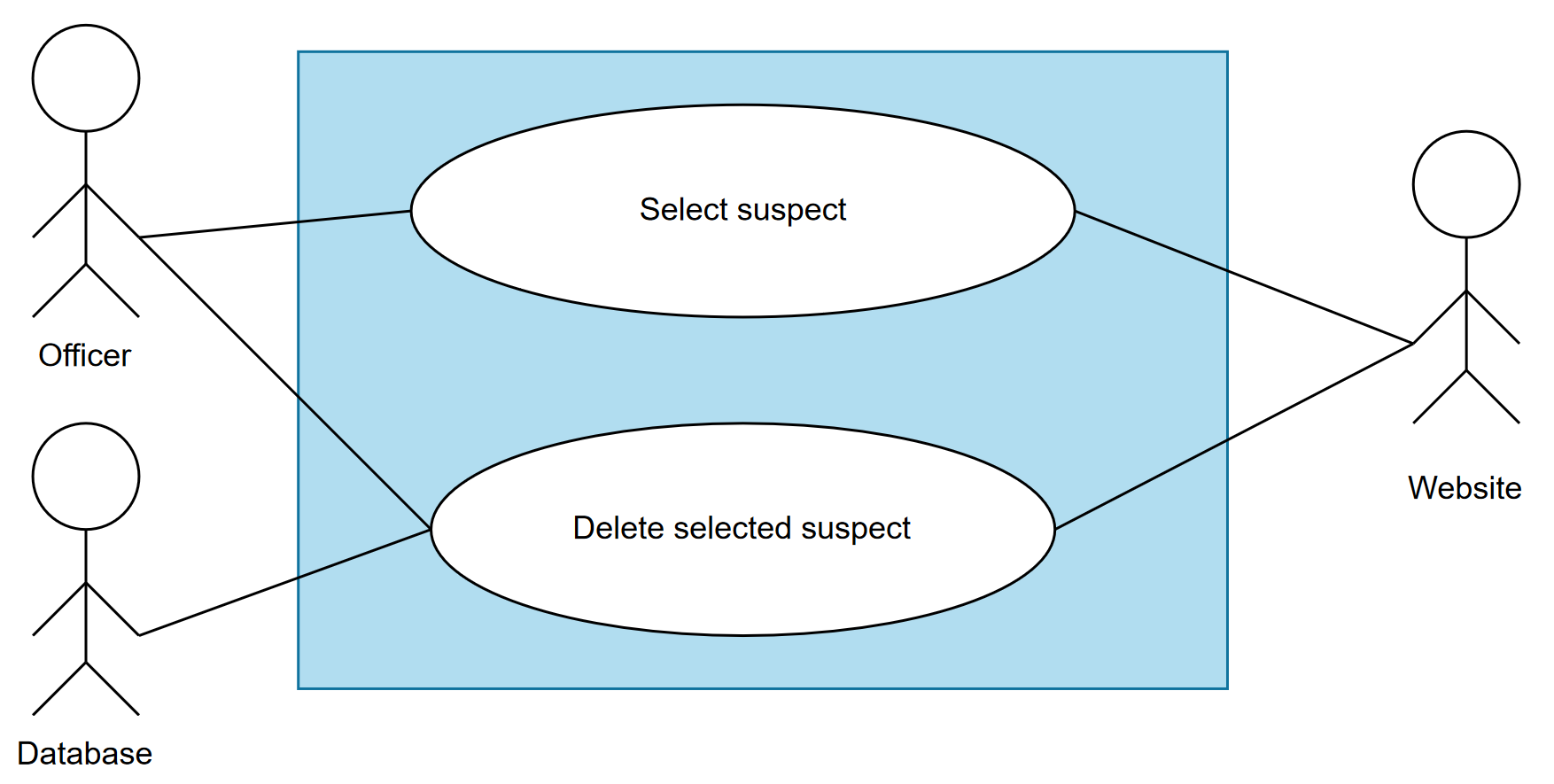
1. Adding a suspect to the database: In this use case we will discuss what actors are involved when we add a new suspect into the database and then we will give a detailed explanation about the use case by giving a table.
2. Removing a suspect from a database: In this use case we will discuss what actors are involved when we remove a suspect from the database and then we will give a detailed explanation about the use case by giving a table.
3. Adding a witness: In this use case we will discuss what actors are involved when we add a new witness into the database and then we will give a detailed explanation about the use case by giving a table.
4. Removing a witness: In this use case we will discuss what actors are involved when we remove a witness from the database and then we will give a detailed explanation about the use case by giving a table.
5. Adding a lineup for witness: In this use case we will discuss what actors are involved when we add a new lineup for witness into the database and then we will give a detailed explanation about the use case by giving a table.
6. Submitting identified suspects: In this use case we will discuss what actors are involved when a witness submits identified suspects into the database and then we will give a detailed explanation about the use case by giving a table.

## **3.3 Use Case Diagrams**

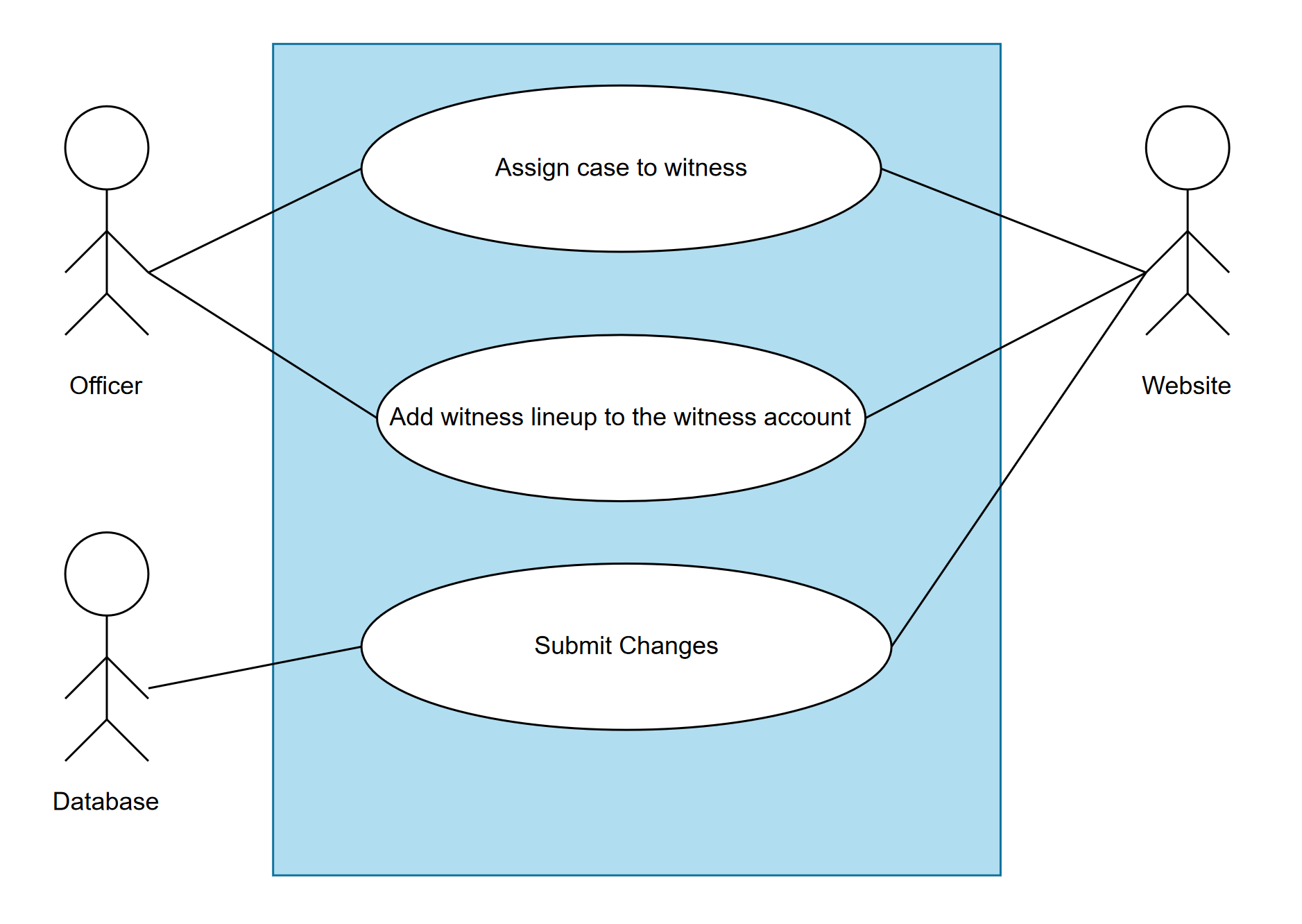
Adding suspect to the database



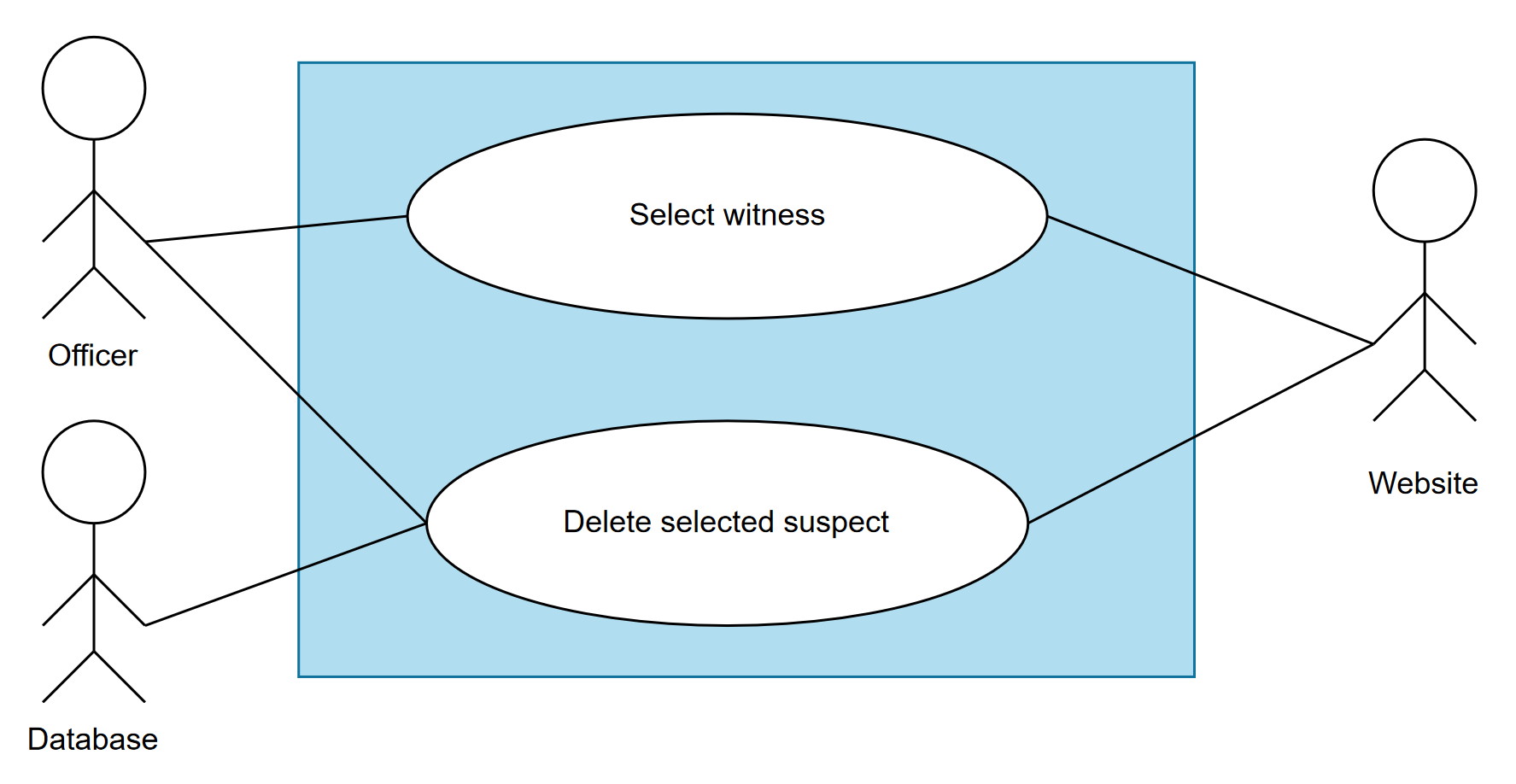
Removing a suspect from a database



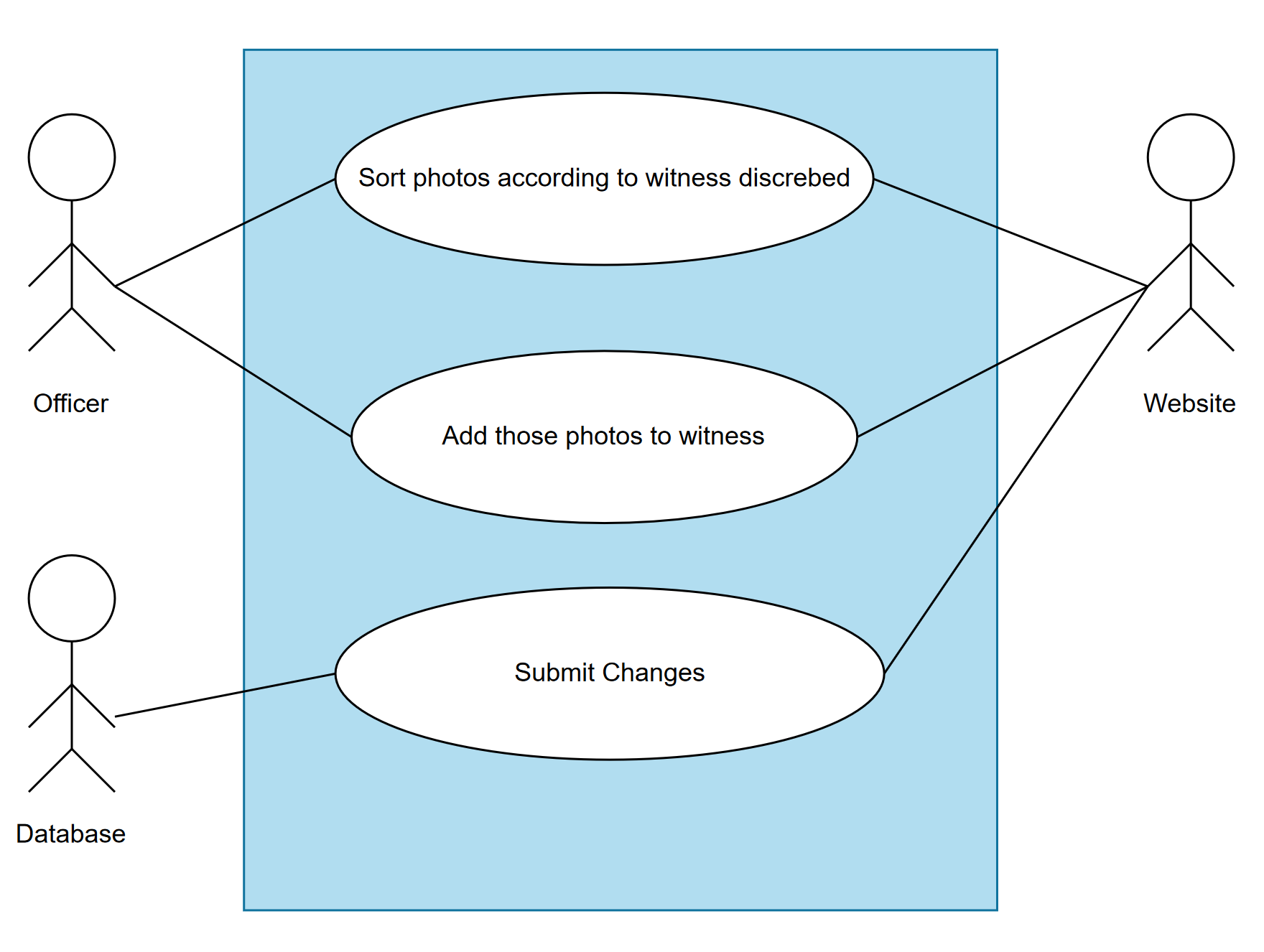
Adding a witness:



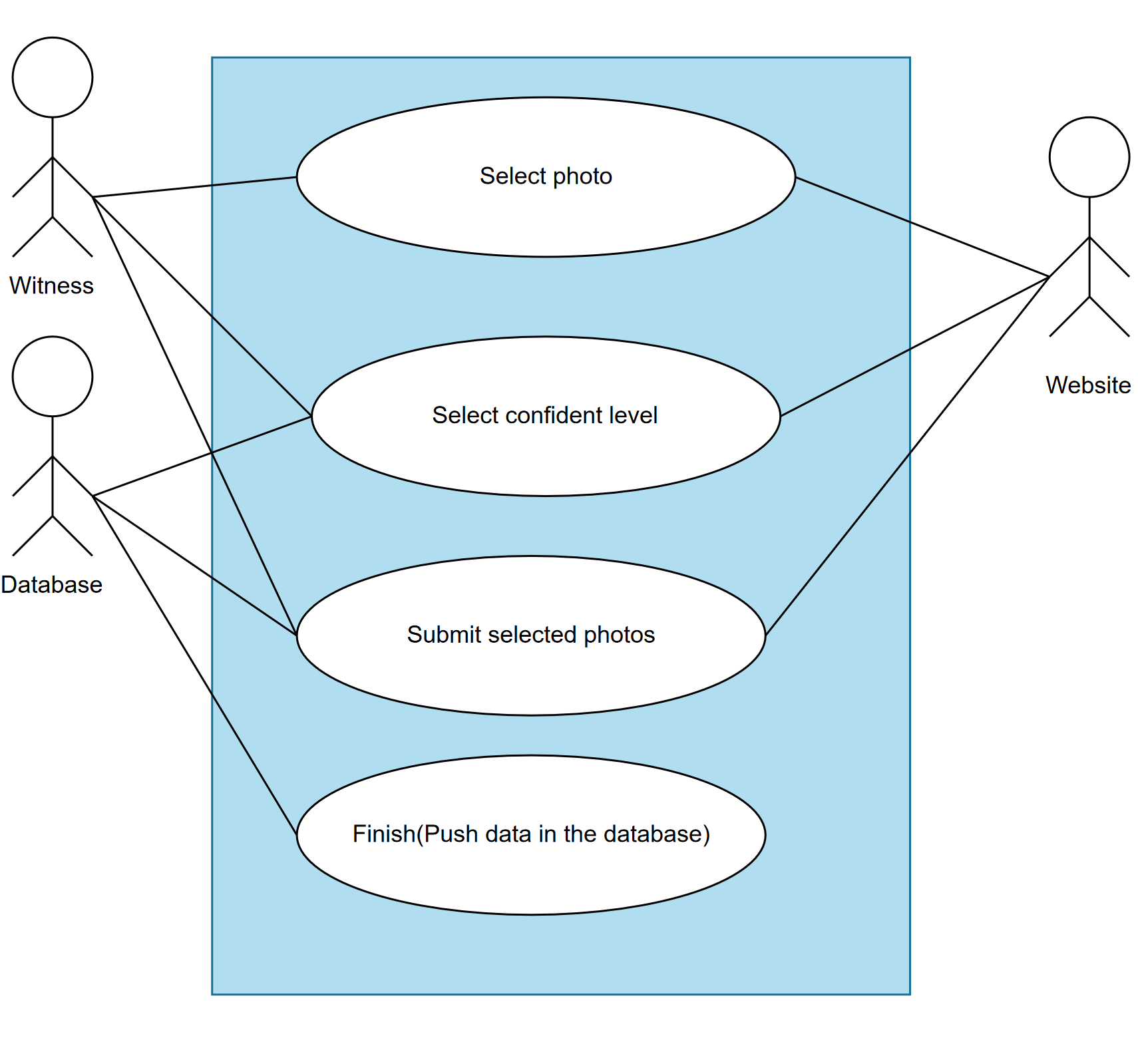
Removing a witness



Adding a lineup for witness



Submitting identified suspects



## **3.4 Use Cases**

| Use Case: Eye-witness Website | | | |
| --- | --- | --- | --- |
| ID: UC1 | | | |
| Actors:Officer, Web server, Database server | | | |
| Preconditions: none | | | |
| Primary Scenario:  Officer selects the add suspect button.  Officer selects the photo of the suspect.  Officer adds a description of the suspect.  Officer saves changes into the database. | | | |
| Secondary Scenario:  Queryset error.  Invalid file type. | | | |

| Use Case: Deleting suspects | | | |
| --- | --- | --- | --- |
| ID: UC2 | | | |
| Actors:Officer, Database server | | | |
| Preconditions: | | | |
| Primary Scenario:  Officer selects the suspect.  Officer deletes the suspect.  Officer saves the changes. | | | |
| Secondary Scenario:  Queryset error. | | | |

| Use Case: Case assign | | | |
| --- | --- | --- | --- |
| ID: UC3 | | | |
| Actors:Officer, database server | | | |
| Preconditions:  Witnesses already exist. | | | |
| Primary Scenario:  Officer selects the witness.  Assign the case to a witness.  Assign the lineup for witnesses to identify.  Officer sorts the photos according to what the witness told.  Submit the changes to the database. | | | |
| Secondary Scenario:  Witness not found  Create new witness  Sorting error | | | |

| Use Case: Deleting witness | | | |
| --- | --- | --- | --- |
| ID: UC3 | | | |
| Actors:Officer, Database server | | | |
| Preconditions: Witness already exists | | | |
| Primary Scenario:  Officer selects the witness.  Officer deletes the witness.  Officer saves the changes. | | | |
| Secondary Scenario:  Queryset error. | | | |

| Use Case: Assigning lineup to a witness | | | |
| --- | --- | --- | --- |
| ID: UC5 | | | |
| Actors: Officer, Database, Witness | | | |
| Preconditions:witness and suspects exists | | | |
| Primary Scenario:  Witness describes the suspects  Accordingly officer sorts all the suspects  Assigns that lineup to the witness’s profile  Submit changes to the database | | | |
| Secondary Scenario:  Witness not found. | | | |

| Use Case: Identifying suspects | | | |
| --- | --- | --- | --- |
| ID: UC6 | | | |
| Actors:Witness. database | | | |
| Preconditions:Officer already created a lineup for witness | | | |
| Primary Scenario:  Suspect login to account  Suspect selects their case  Suspect starts looking at the photos  Suspects selects the photo  Suspects gives confidence level to the photo  Suspect submits the changes. | | | |
| Secondary Scenario: | | | |

# **4. Design Overview**

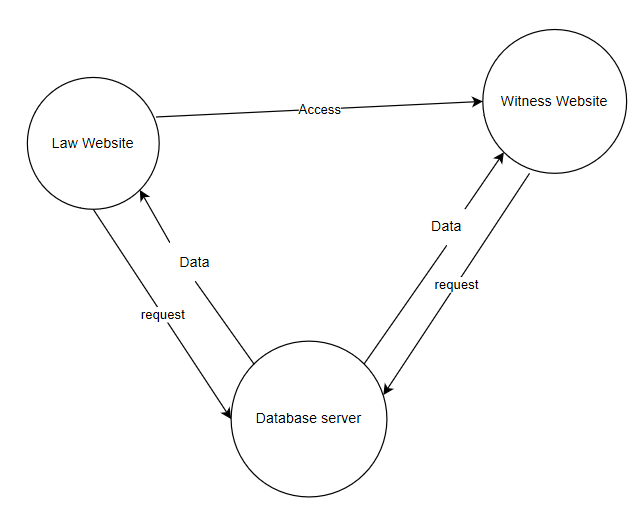
## **4.1 Introduction**

In this section we will explain about the overall design of our website and how the internal architecture is designed to help both law enforcement and witnesses better help each other. We will first explain about our system architecture, then we will go over the system interface and finally we will talk a little bit about constraints and assumptions.

## **4.2 System Architecture**

Our system is divided in three parts

* Law Enforcement website
* Witness website
* Database server



Above diagram shows how all the aspects of the project will work together to function properly as a whole. We are creating this project entirely on django so we are writing all the scripting required for the project in Python and using sqlite for the database. We will be creating a form for adding stuff in and then for witnesses to submit the selected photos. We will discuss the individual objects that we created in the section later.

## **4.3 System Interface**

Overall system is built in a way that all the parts of the system have to work to make everything work properly. So for instance here is the flow of how all the functionality should work.

## **4.4 Constraints and Assumptions**

We are using raw data which was provided to us by Dr. Baker for the purpose of this project. And the biggest problem at this point is to add all the data that we have into our database. We have not thought of a system which can add all the data automatically into the database but at this point we will have to add everything manually which will take a lot of our time and energy.But accept that we don't think we might have any issues.

# **5. System Object Model**

## **5.1 Introduction**

As we mentioned earlier that our system is divided in three different subsystems as follows

* Law Enforcement website
* Witness website
* Database server

Here is the explanation of those.

## **5.2 Subsystems**

1. Law Enforcement Site: This will be the primary site of the project and also the most useful one. Only law enforcement officers will be allowed to login into this part of the site. And this site will be capable of doing stuff like adding new suspects, witnesses and even case.This is the website where all the new witness login information will be added as well. Other than adding everything they have permission to delete everything they can add. We are creating this website using Django so we have created necessary forms and views to display everything from the database to our html pages. We will discuss these forms and views in the coming section.
2. Photo Library for Witnesses in Database: This will be within our Law Enforcement site but this is a very integral part of our project. After adding a new witness officer will create a lineup of suspects from the complete criminal database. Now this lineup will be easily able to modify if something has to be changed.
3. Database: Probably the most important part of our product which is the entire database. Our database is hosted locally. And we will talk about that later in the documentation. Following is the diagram of how each part of the system works with each other.

## **5.3 Subsystems Interface**

Law Enforcement Interface:

This interface will allow officers to login to their account. Do all the major admin related tasks like adding/deleting witnesses, suspects and cases. They will also be able to change witness login credentials. Following is the rough layout of how our law enforcement will look like.

Eye-witness Interface:

This will be for the witnesses, no witness will be directly able to create their own account but officers will have to create for them and once they have login information, they will be able to login and then identify suspects and then submit those to the database.

Database:

This part of our system won't be visible to witnesses and officers also will be seeing a very simplified version of the raw data. But more about databases in the future sections.

# **6. Object Descriptions**

## **6.1 Objects**

| **App:Police/ Views.py** | **Method Descriptions** |
| --- | --- |
| def Police(): | Connects to police.html in templates |

| **App:Witness/ Views.py** | **Method Descriptions** |
| --- | --- |
| def Witness(): | Connects to Witness.html in templates |

| **App:login / views.py** | **Method Descriptions** |
| --- | --- |
| def Login(): | Connects to Login.html in templates |

| **App:Register/ Views.py** | **Method Descriptions** |
| --- | --- |
| def Register(): | Connects to Registeer.html in templates |

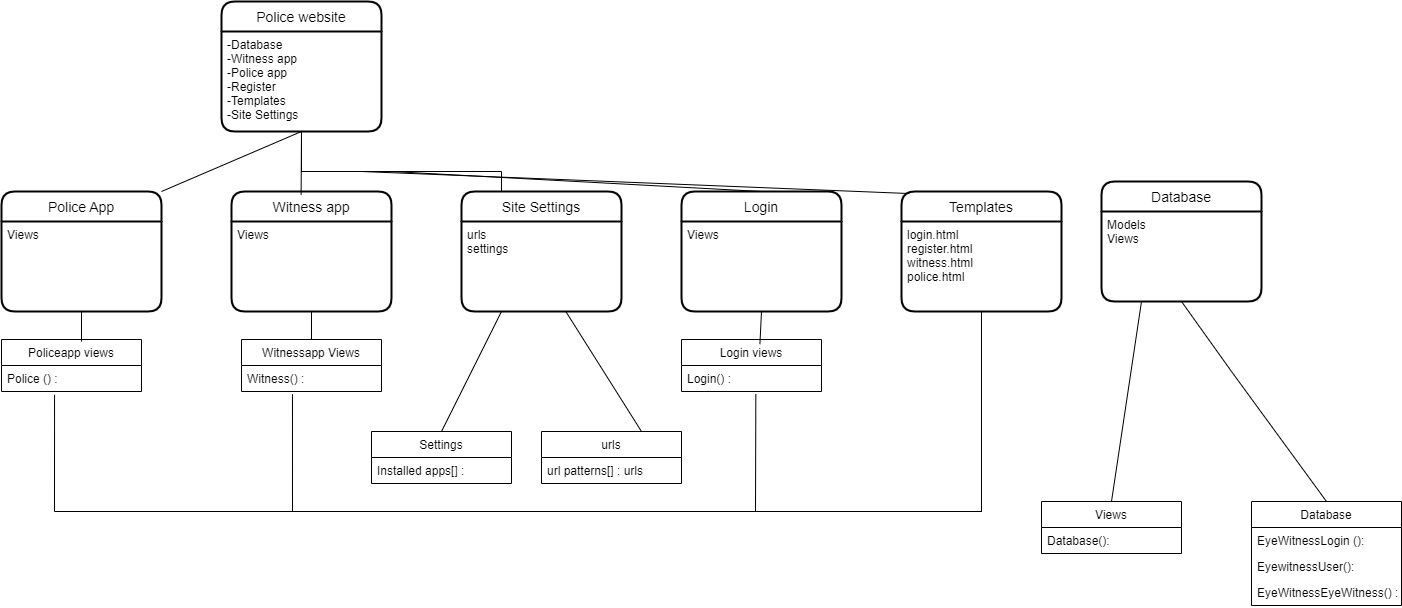
| **App:Databse /Method.py** | **Method Descriptions** |
| --- | --- |
| Witnesslogin() | Sets up database model |
| WitnessUser() | Sets up database model |
| WitnessWitness() | Sets up database Model |

| **App: Settings Url/py** | **Method Descriptions** |
| --- | --- |
| UrlPatterns[] | Sets up pathway for Url recognition |

| **App: Settings /Settings.py** | **Method Descriptions** |
| --- | --- |
| Installed Apps[] | Sets up system to recognize app installation |

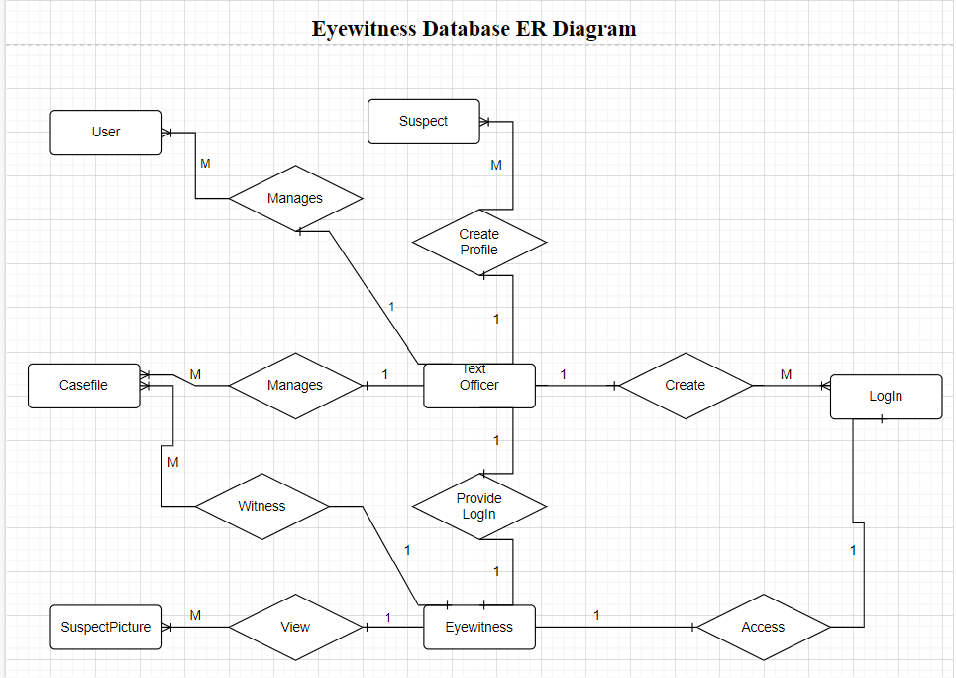
# **7. Object Collaboration**

## **7.1 Object Collaboration Diagram**

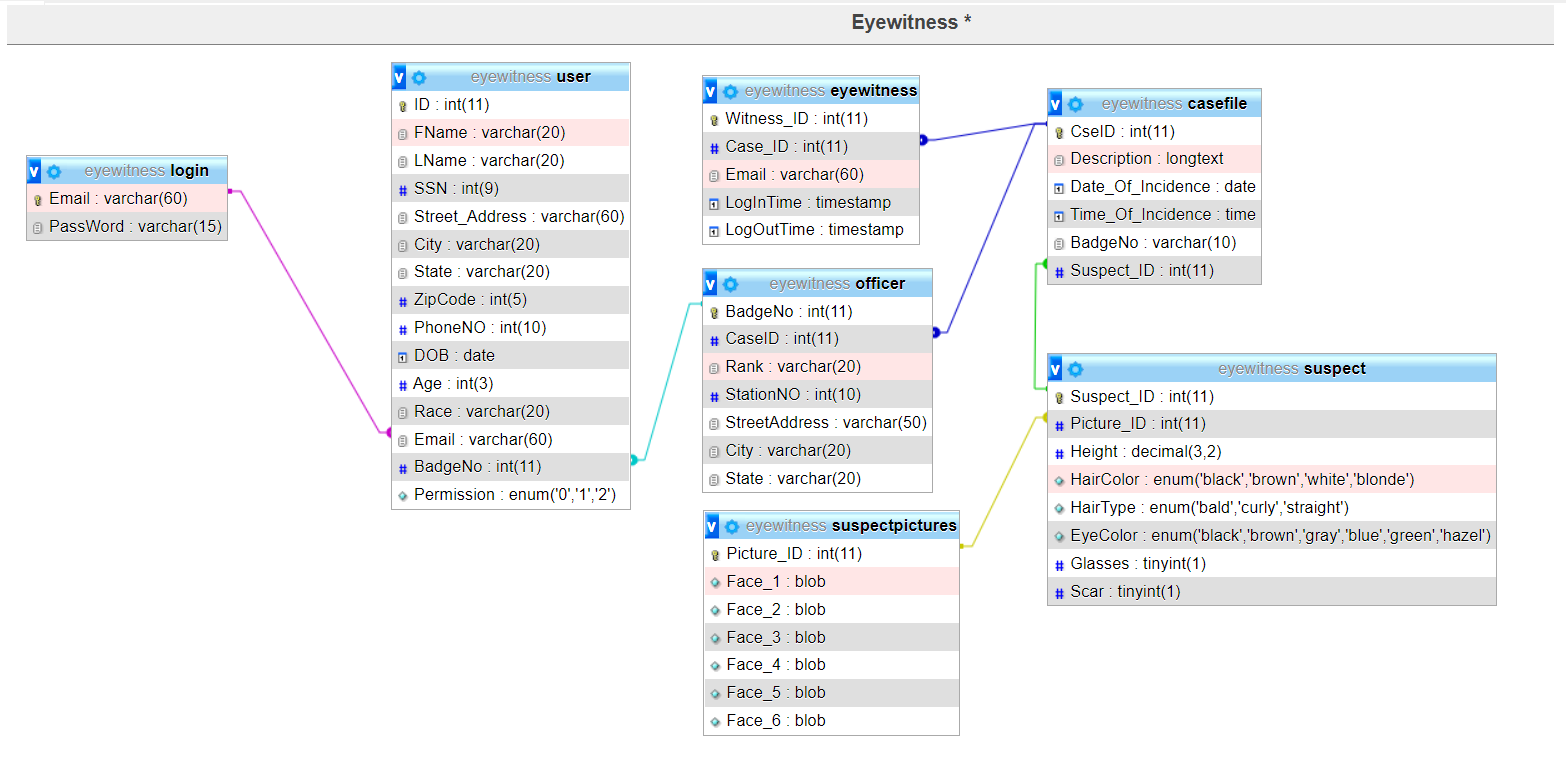


# **8. Data Design**

## **8.1 Entity Relationship Diagram**



**8.2 Conceptual Design**

****

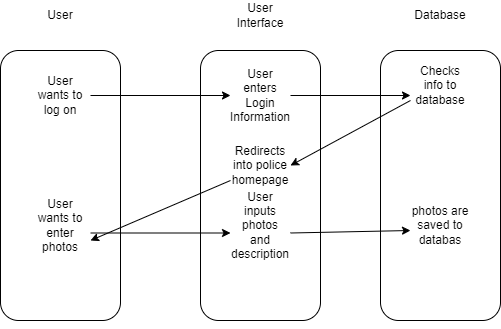
# **9. Dynamic Model**

## **9.1 Sequence Diagrams**

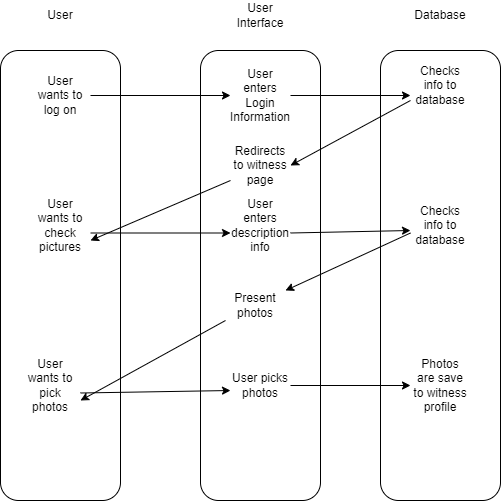


## **9.2 State Diagrams**

9.2.1 Police Diagram



9.2.2 Witness Diagram



# **10. Non-Functional Requirements**

## **10.1 Performance Requirements**

The website will be able to multitask to allow multiple users to interact with the system without having to wait on another user to finish one’s task. Django is the main framework for this website, which has very quick response time for any queries.

For the database requirement we are using SQLite. SQLite provides an excellent development alternative for applications that are predominantly read-only or require a smaller installation footprint. The website will be able to hold and browse through large amounts of data.

Django has one of the best out-of-the-box security systems, and it helps avoid common security issues, such as SQL injection, clickjacking, and cross-site scripting. It is also cross-platform, meaning it can be based on Mac, PC or Linux.

## **10.2 Design Constraints**

The website is built using the Django framework. Django is the open-source framework for backend web applications based on Python, and its main goals are scalability, flexibility, reliability, and simplicity. Also, Django has its own main system for all functions and components, such as HTTP response called “views”. It has an admin panel, which is easier to work with other technical features like, HTTP libraries, Python unit test frameworks, etc.

Because an SQLite database requires no administration, it works well in devices that must operate without expert human support. Client/server database engines are designed to live inside a lovingly attended datacenter at the core of the network. SQLite works there too, but SQLite also thrives at the edge of the network, fending for itself while providing fast and reliable data services to applications that would otherwise have dodgy connectivity.

# **11. Supplementary Documentation**

## **11.1 Tools Used to Create Diagrams**

We have a separate excel sheet for the timeline of the project.But for the languages and tools we didn't have to say a lot.

We used Django server and django environment to create the website and also for the database purpose we are using django's built in database sqlite. Python is the one supported language for django. And here is the list of all the other tools we used to create this documentation.

Please find the other file named “ganttChart.xlsx”.

* Models made with draw.io- https://app.diagrams.net/