**Project Requirements Specification**

**MoonEyes: Case Management System for Metro Detective Agency**

Senior Design I Summer 2023

Kylie Callison

Erika Valle-Baird

Kendall Gesek

Pamela Shahu

Table of Contents

[**1.0 Introduction 4**](#_sakdae6o8zsd)

[1.1 Goals and objectives 4](#_83lsguvh9hlf)

[1.2 Statement of scope 4](#_t3n72uclias0)

[1.3 Software context 5](#_7b5tswhseni9)

[1.4 Major constraints 6](#_20cqw5x467so)

[**2.0 Usage scenario 7**](#_vii5fmd6vbvg)

[2.1 User profiles 7](#_wn87goy9azx5)

[2.2 User stories 7](#_6um56btj1faz)

[2.3 Special usage considerations 8](#_qrvlx6xxzw5t)

[**3.0 Data Model and Description 9**](#_tb52bh956xd7)

[3.1 Data Description 10](#_ostvs7y628he)

[3.1.1 Data objects 10](#_me4kq5zdoy5c)

[3.1.2 Relationships 10](#_v2qf89aqwey8)

[3.1.3 Complete data mode 10](#_2tqccg98ze1f)

[3.1.4 Data dictionary 11](#_92qlp9r4oubi)

[**4.0 Functional Model and Description 12**](#_mmzvdin54qtx)

[4.1.2 Actors 27](#_oaupff9o1jhc)

[4.2 Software Interface Description 28](#_xckb84o5zq6v)

[4.2.1 External machine interfaces 28](#_pajmk6aqsnh2)

[4.2.2 External system interfaces 28](#_puf8tx72bz64)

[4.2.3 Human interface 28](#_sdpy3ccsj23t)

[4.3 Use Case Diagrams 29](#_axe51cbpnpoj)

[5.1 Description for software behavior 30](#_7crc4x3tjaxl)

[5.1.1 Events 32](#_y86t9swit79i)

[5.1.2 States 32](#_911xn84cufbj)

[5.2 State Transition Diagrams 32](#_del0rgo1ugsr)

[5.3 Control specification (CSPEC) 33](#_dhq90fti7ct)

[**6.0 Restrictions, Limitations, and Constraints 34**](#_ojeueotqyrj4)

[**7.0 Validation Criteria 35**](#_sog04b32pz3r)

[7.1 Classes of tests 35](#_lask6w4768tw)

[7.2 Expected software response 35](#_ao285iy3m3t6)

[7.3 Performance bounds 35](#_ttvcjx9njv2h)

[**8.0 Appendices 36**](#_van8nf5fpw3w)

[8.1 System traceability matrix 36](#_z6ohpg78fpbl)

[8.2 Product Strategies 36](#_ryi9pbp385qi)

[8.3 Analysis metrics to be used 37](#_16xsrunkwl2j)

[8.4 Supplementary information (as required) 38](#_3bf489sum9g9)

[8.5 References 39](#_jlkdyzdtbna8)

# 1.0 Introduction

## 1.1 Goals and objectives

Our senior design project is MoonEyes: a case management system for a private detective agency. The main purpose of this project is to create a process that allows agents from Metro Detective Agency to submit and track case information(including client, subject and agent details). These exact details are specified in [1.2 Statement of Scope](#_t3n72uclias0). Overall, the main goals and objectives of MoonEyes are as follows:

* To provide a searchable database of all case information
* To improve organization and centralize case and client details
* To improve information exchange between agents
* To increase efficiency when searching for case details
* To ensure that all data is kept secure
* To provide new marketing functionally via mass email

Additionally, the goal of this use case document is to serve as a guide when we begin developing the system. Referencing this document will help us keep track of the product requirements and user stories throughout the development timeline. This document will also allow our development team and the client to assess the quality of our finished product and ensure we have met all previously established requirements.

## 1.2 Statement of scope

As mentioned previously, this software is to serve as a case management system for agents within the Metro Detective Agency.

The following requirements were given to us for MoonEyes:

* A way in which agents can log in with credentials
* A way in which agents can remotely access the system
* A way in which an agent can enter information about a client, case, subject or agent into a database.
* Ability to search for any parameter within the database
* A way in which data that was previously entered into the database can be clearly viewed
* A way in which agents can view recent submissions or changes
* A way in which an admin user can view audit logs
* A way in which PDF reports can be generated to contain specific case/client/subject information
* A way in which a mass email can be generated and sent out to a subset of clients

**Database Creation**

In order to accommodate the requirements for client, case, agent and subject tracking, a database will need to be created. Specific details that need to be tracked for each table are described below. Furthermore, this is also the information that agents should be allowed to enter in forms and be stored in the database to search. These are additional requirements that specify the type of information that will be submitted within the database and also the parameters for the search functionality.

* **Client Tracking**

Client name, address, phone number, attorney information,and additional notes.

* **Case Tracking**

Case numbers, Purpose, date, reports, photos/videos, and additional notes.

* **Subject Tracking**

Subject name, associates, phone number, place of work, vehicle information (license plate number, type of car), lawyer information, locations visited, repeated locations, background reports, photos/videos, and any other additional notes.

* **Agent Tracking**

Badge number and case numbers.

## 1.3 Software context

**Deployment**

For this software to become accessible to agents with remote access, we will need to eventually deploy onto a cloud-based web hosting service such as Amazon EC2 or Azure. We must make sure all developments are compatible with these services. Additionally, we must ensure that any other services we might use, such as the database platform (MySQL) are compatible with the hosting service as well. Also, it is likely that users will need to be able to use the system simultaneously. Currently, we do not believe many people will have access to the system at once but expanding the amount of users could cause issues which we need to address during development.

**Security**

Because this system will likely be storing personal information about clients. Subjects and agents, security is an important consideration that needs to be integrated into the design

## 

## 1.4 Major constraints

**Funding**

As mentioned, we will need to deploy MoonEyes on a cloud-based hosting service. This means it will cost anywhere from $10-50/month for our client.

**Time**

Because this project is for our senior design project we have a strict eight-month development timeline. It is possible that we might want to make improvements or add additional features down the road that are incompatible with the amount of time left. Additionally, all members of the team will also be balancing class and work responsibilities along with our project.

**Experience**

Because our group consists of students with limited professional development experience, we may have to do more independent and team research than a typical development team.

.

# 2.0 Usage scenario

## 2.1 User profiles

**Users**:

Admin

User

System

**There will be two levels of users:**

Full Control:

Administrator

Read Only:

General Public

## 

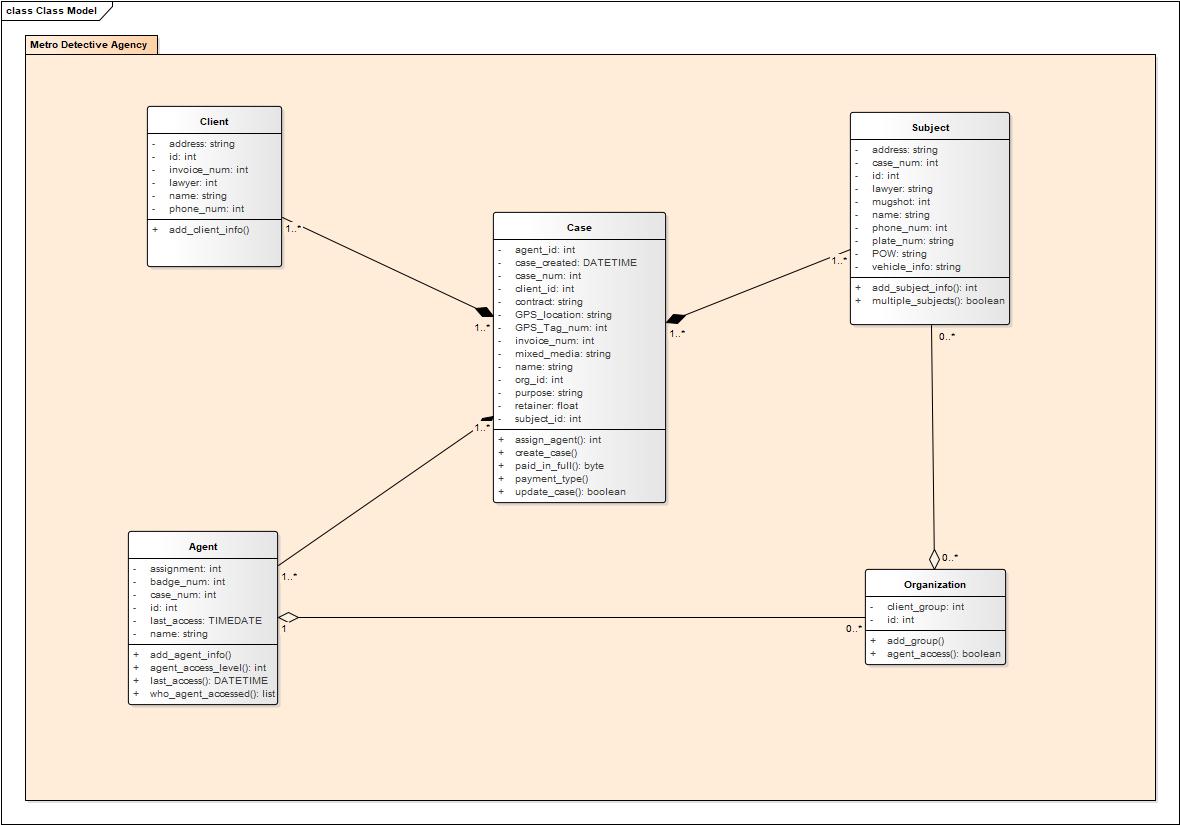
## 2.3 Special usage considerations

**Accessibility**

Because the people using this software may be unfamiliar with case management systems, we should make sure the interface is easily navigable and understandable. We need to clearly display all information needed and possibly offer a help section or tutorial on how to use the software.

# 3.0 Data Model and Description

This section describes information domain for the software

***The main classes of the Metro Detective Agency include:***

**Client**: A person who employs the Metro Detective Agency for a case.

**Case**: Each case will hold information containing its purpose, locations, media, retainers, agents working on each case, client for the case, and subjects for each case.

**Agent**: An employee of Metro Detective Agency who works on cases. An agent is allowed access to groups of clients under organization if admin has cleared them to do so.

**Organization**: Each organization contains groups of clients categorized by admin.

## 3.1 Data Description

Data objects that will be managed/manipulated by the software are described in this section.

### 3.1.1 Data objects

Data objects and their major attributes are described.

**Client**

The Client table contains all information about the agency's clients. Once a client is added to the system it is assigned a unique identifier(*clientID*). Each client will also have additional information attached such as their email, address, lawyer and then any tags.

**Case**

The Case table contains all details about the case. Once a case is added to the database it is assigned a unique case number(*caseNum*). Each case will also generate a timestamp *ts* and a *status* attribute as well( starts as open ‘open’). Each case attribute will have fields for relevant client, subject and agent IDs, which will eventually allow for a user of the system to click on and navigate to those objects. Additionally, each case will also contain details like: purpose, locations, media, retainers.

**Subject**

The subject table contains all information about the subject of a case. This could include someone like a suspect in a certain case or maybe a missing person or family member of a client. Once a case is added to the database it is assigned a unique number(*subjectID*).

**Agent**

The agent table contains relevant information about an agent such as their name and badge number and also what cases they have been assigned. It is also assigned a unique agentID for linking purposes.

### 3.1.2 Relationships

Relationships among data objects are described using an ERD- like form. No attempt is made to provide detail at this stage.

### 

### 

### 3.1.3 Complete data model

An ERD for the software is developed

### 

**Client/Case**

For a case to exist there must be at least one pre-existing *clientID* that will become attached to the case in the *clientIDs* field. A client does not necessarily have to have a case attached but it is an option that will be saved in the *caseIDs* field.

**Subject/Case**

For a subject to exist there must be at least one pre-existing *caseID* that will become attached to the subject in the *subjectIDs* field*.* A case does not require a subject but it is an option that will be saved in the *subjectIDs* field.

**Agent/Case**

Agents do not necessarily have any *caseIDs* assigned to them and cases do not need any agent to be attached.

**Tags**

Tags can be created and optionally linked to any case, client, subject or agent.

### 

### 

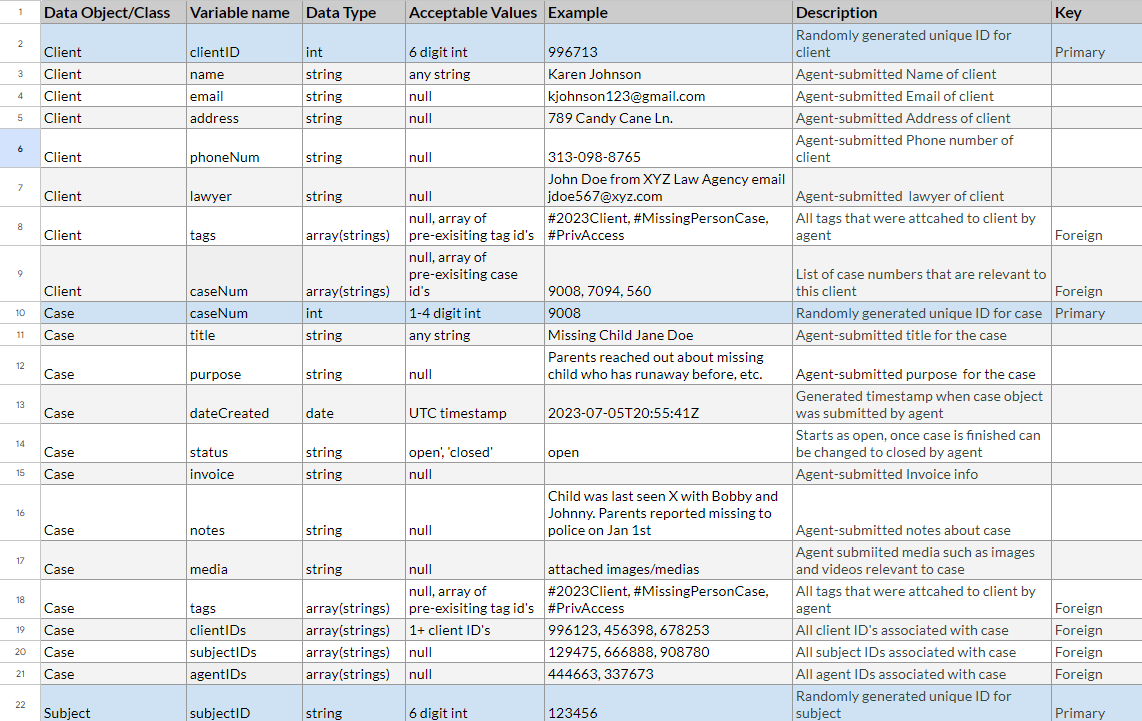
### 3.1.4 Data dictionary

The Complete Moon Eyes data dictionary is maintained in electronic form here.

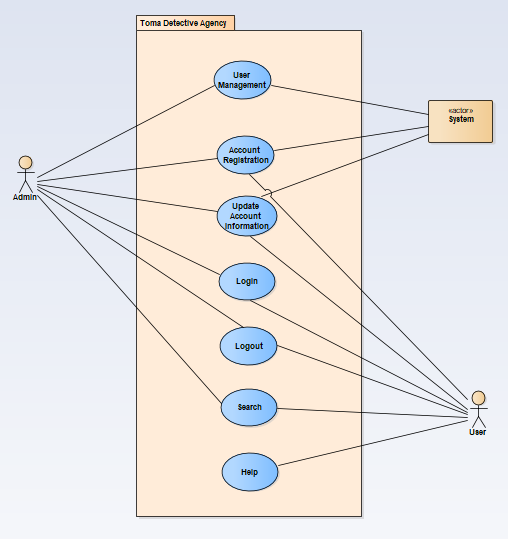
[MoonEyes Data Dictionary](https://docs.google.com/spreadsheets/d/1m3bAlzV34hWUjziYH2o5Z25da6-P0TZZdgUkvvmdSmQ/edit?usp=sharing)

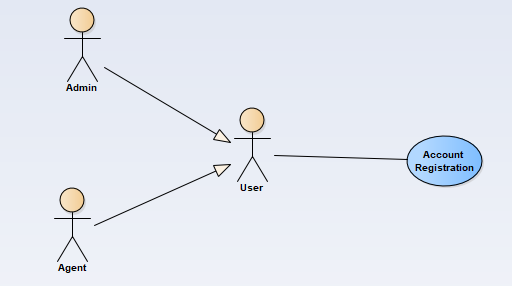
**Shared URL:** <https://docs.google.com/spreadsheets/d/1m3bAlzV34hWUjziYH2o5Z25da6-P0TZZdgUkvvmdSmQ/edit#gid=0>

**Preview**



# 4.0 Functional Model and Description





| **Use Case Name:** | **Account Registration** | |
| --- | --- | --- |
| **Actors:** | Agent, Admin | |
| **Description:** | Agent | Admin |
| This function allows agents to register for an account with the system. The system will check the username, given by the user, is not a duplicate name in the current system’s database. The system will require necessary user data to create an account. | An employee of the system must be allowed access to generate reports, analyze user behavior, perform user management, view and edit customer information, and assist agents.. |
| **User Story:** | As an agent, I need to have secure login via a username and password. | As an admin, I need to have secure admin access. |
| **Preconditions:** | The user does not have an account. | |
| **Scenario Description:** | Agent | Admin |
| 1. The use case starts when an agent indicates that he or she wants to register.  2. The system requests a username and password.  3. The agent enters a username and password.  4. The system checks that the username would not be a duplicate of any existing registered usernames.  5. The system requests information about the agent, specific to the account type.  6. The system starts a login session and displays a welcome message based on the username. | 1. The use case starts when a user indicates that he or she wants to register.  2. The system requests a username and password.  3. The user enters a username and password.  4. The system checks that the username would not be a duplicate of any existing registered usernames.  5. The system requests information about the user, specific to the account type.  6. The system starts an admin login session and displays a welcome message based on the username. |
| **Post Conditions:** | Agent | Admin |
| The agent is now registered and can access the system.  The agent can now obtain case information and update their account information.  The agent is now registered and can access the system. | The user can now obtain case information, update their account information, and get logged information.  The user is now registered and can access the system. |
| **Special Requirements:** | The user must not be banned from the system.  The user is not already a registered user of the system. | |

| **Use Case Name:** | **Account Login** | |
| --- | --- | --- |
| **Actors:** | Agent, Admin | |
| **Description:** | Agent | Admin |
| This function allows agents to login to the system. To confirm an agent’s identity, the system will require both a username and password. | An employee must login to the system to edit account information, generate reports, or perform user management. |
| **Preconditions:** | · The user is registered and has a username and password.  · The user’s account is not locked or admin must be contacted for user verification.  · A user is logged into their account after inputting their username, password, and selecting the login button. | |
| **Scenario Description:** | 1. The use case starts when a user indicates that he wants to login.  2. The system requests the username and password.  3. The user enters their username and password.  4. The system verifies the username and password against all registered users.  5. The system starts a login session and displays a welcome message based on the user’s account type. | |
| **Alternative Flow:** | 4.a:  if the username is invalid, the use case goes back to step 2.  4.b:  if the password is invalid the system requests that the user re-enter the password. When the user enters another password the use case continues with step 4 using the original username and new password.  4.c:  If three incorrect inputs for password is signed, user’s account is locked until account and behavior are verified, signaling an admin to release it. | |
| **Post Conditions:** | · The user can now obtain case and client information from the database .  · The user remains logged in until the “Logout” button is specified. | |
| **Exceptions:** | · Cancel button is pressed returning the user to the main screen without being logged in.  · The user is registered. | |

| **Use Case Name:** | **Account Logout** |
| --- | --- |
| **Actors:** | Agent, Admin |
| **Description** | The agent or admin is logged into their account on the system. |
| **Preconditions:** | The agent or admin is logged into their account on the system. |
| **Triggers** | Account Logout is triggered when a user, currently logged into the system, selects the logout button. |
| **Scenario Description:** | The user selects the “Logout” button on the main menu. |
| **Post Conditions:** | The user is logged out of their account. |
| **Exceptions:** | The user does not select logout and is therefore logged out of the system after sixty minutes of inactivity. |

| **Use Case Name:** | **Search** |
| --- | --- |
| **Actors:** | User |
| **Description:** | This function allows an admin to search for case information. The system will check the information provided against cases in the database. The system will display all cases that match the provided information. |
| **Preconditions:** | · The user is registered.  · The user is logged in to their account. |
| **Scenario Description:** | 1. The “Search” button has been selected by the user.  2. The system displays entry fields for the user to input for search.  3. After a field has been entered, the system checks the details entered against information in the database.  4. If there are matches, the system displays matching case numbers to the details entered.  5. The user can then select which case they would like to view.  6. The system displays that case information.  7. User inputs invalid search criteria |
| **Alternative Flows:** | 3a. A list of cases that match field information, as designated by the user, is displayed for the user to browse and choose from  3b.There are no cases matching user’s specifications, return to step 1. |
| **Post Conditions:** | · All cases matching the search parameters are returned to the user who requested cases with certain criteria. |
| **Triggers:** | · Search is triggered when a user selects the search button after inputting search criteria. |

| **Use Case Name:** | **User Management** |
| --- | --- |
| **Actors:** | **Admin** |
| **Description:** | This function provides admin with the ability to edit case information, add cases, and save cases. Admin must be allowed to access reports, analyze user behavior, view and edit information, and assist users. |
| **Preconditions:** | The user is logged in, the user’s account is an admin account. |
| **Triggers:** | User management is triggered when an admin selects the edit button, allowing the user to edit case information.  It is also triggered upon adding and saving cases. |
| **Scenario Description:** | 1. The use case starts when an admin selects the “User Management” button.  2. The admin is then displayed with options to edit or add cases, view or edit client information, assist a user, and audit logs. |
| **Alternative Flow:** | 2a.Admin attempts to save a duplicate case. Met with duplicate message error.  2b. Admin edits a case file using invalid information. Met with invalid information message. |
| **Postconditions:** | Any requested reports have been sent to a printing device or saved remotely.  All added and edited client information has been saved to the database. |
| **Exceptions:** | · The user is registered.  · Cancel button is pressed returning the admin to the main screen without saving editing of user accounts. |

| **Use Case Name:** | **Update Account Information** |
| --- | --- |
| **Actors:** | **User, Admin** |
| **Triggers:** | The update account information button has been pressed. |
| **Summary:** | A user can edit their account information**.** |
| **Entry Conditions:** | · The user has an existing account.  · The user is logged in to the system. |
| **Flow of Events:** | 1. The “Edit Account” button has been selected by the user.  2. The system displays current account information, allowing for input from the user.  3. The user selects “Done”, submitting the changes. |
| **Alternative Flow:** | 3a. The input by the user included a character that was not recognized as allowable input for that given field, returns user to step 2. |
| **Exit Conditions:** | The account has updated information per the user input. |
| **Special Requirements:** | · The user is registered. |

| **Use Case Name:** | **Help** |
| --- | --- |
| **Actors:** | User |
| **Description** | This function provides a user with information pertaining to the functionality of specific sections within the system. |
| **Preconditions:** | The user or admin is logged into their account on the system. |
| **Triggers** | Help is triggered when a user selects the help button. |
| **Scenario Description:** | · The user selects the “Help” button.  · Headers for each function are displayed to the user’s screen.  · The user selects which function they would like information on.  · The system presents a guideline to the screen describing the functionality and how to use that function. |
| **Post Conditions:** | None |
| **Exceptions:** | None |

### 4.1.2 Actors

Admin

An admin is any user of the system who is an employee and holds an admin account. An admin is informed about any changes to the system.

Agent

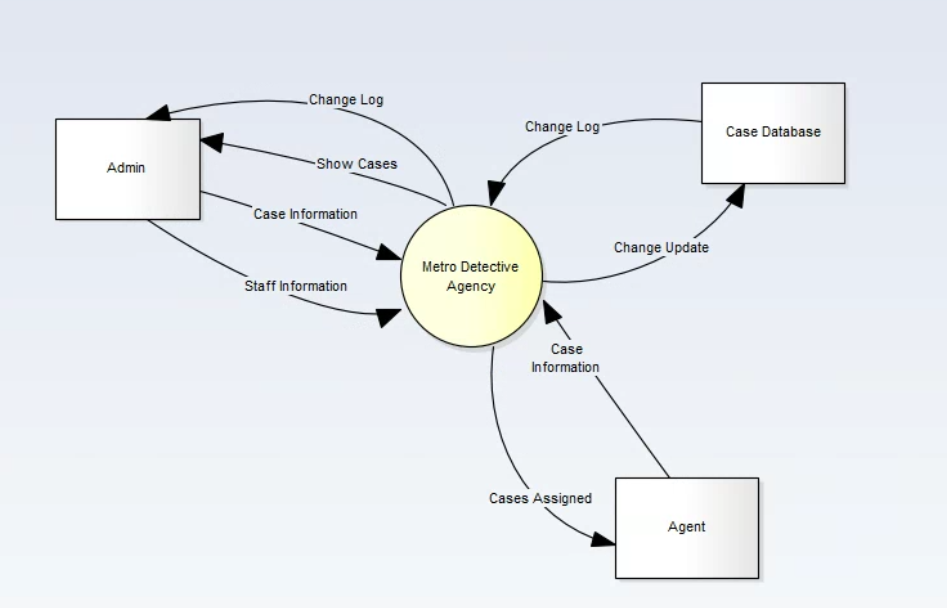
An agent is any user of the system who has a username, password, is an employee of Metro Detective Agency, and holds an account.

System

The system is a database that contains client and agent information. The system is informed about client and agent updates so that is can provide up to date information.

## 

## 4.2 Software Interface Description

Metro Detective Agency’s software interface with the outside world are described below with a graphical depiction of a data flow context level 0 diagram.

**System Externals**

**Agent**

An agent is any user of the system who is an employee, who is not an admin, nor holds an admin account. An agent may search for specific cases, update their information, and update case information. An agent who provides name, address, or other case details in a search query will receive the results of such a query if there is a match. An agent who searches or changes anything in the database will be logged for log records admin receives.

**Admin**

An admin is any user of the system who is an employee and holds an admin account. An admin is informed about case information. An admin also obtains logs of any changes made throughout the system. An admin who provides name, address, or other case details in a search query will receive the results of such a query if there is a match.The admin may update cases, change their information, and update and change agent information.

**Case Database**

The case database is a database that contains customer, case, and agent information. The case database is informed about case, customer, and agent updates so that it can provide up to date information.

5.0 Behavioral Model and Description

A description of the behavior of the software is presented.

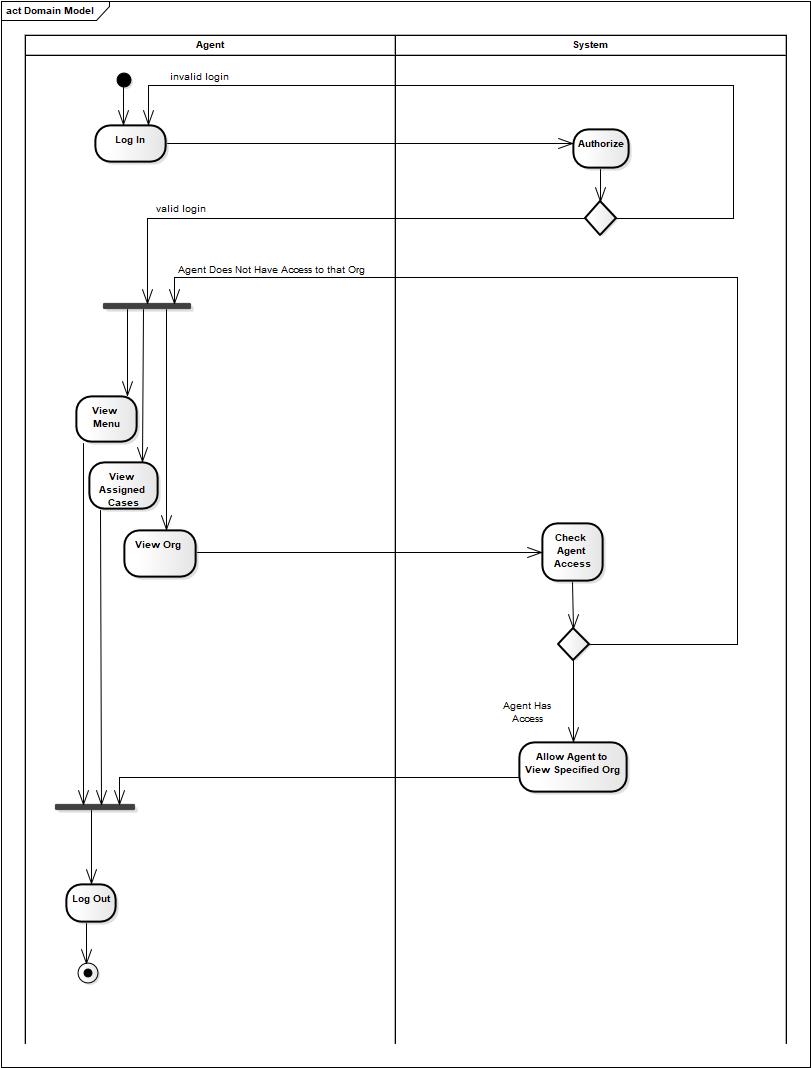
## 5.1 Description for software behavior

A detailed description of major events and states is presented in this section.

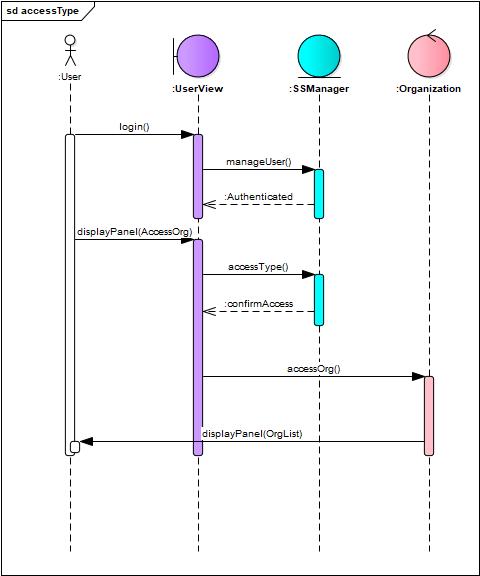
**Activity diagram**

Activity diagram for an agent accessing an organization. After logging in and confirming login validity, an agent is presented with different menu options. Upon choosing to via an organization, the system checks the agent’s access type. If the system confirms the agent is allowed to access that organization, it allows the agent to via cases associated with the organization. If the system determines the agent does not have access to the organization, they are returned to the regular menu.

\*\*This is a behavior diagram :)



Sequence Diagram: Agent accessing an organization.



### 5.1.1 Events

A listing of events (control, items) that will cause behavioral change within the system is presented.

* **User Registration**

When a user provides the required information like name, email, password, and other necessary information successfully registered for an account, it triggers the registration of a new user profile within the system.

* **User Login**

When a user enters their username and password, get successfully log into the system. It triggers the authentication process and grants users' access to their accounts

* **User Logout**

When a user clicks the logout button it triggers the ending of the user's session and restricts their access to the system.

* **Case Creation**

When an agent enters the required information and creates a new case within the system, it triggers the storage of case details in the database.

* **Case Update**

When an agent updates an existing case with new information or modifies existing details, it triggers the system to update the case record in the database.

* **Search Query**

When a user enters the details of a specific case to search for information on that case, it triggers the system to fetch and display relevant cases.

* **Report Generation**

When an agent generates PDF reports that contain case, client, or subject information, it triggers the system to gather the relevant data and generate the report in the desired format.

* **Email Generation**

When an agent starts a mass email to a subset of clients, it triggers the system to get the necessary email content and send it to the designated recipients.

* **Admin Access**

When an admin user logs into the system, it triggers the activation of administrative functionalities and grants access to additional features such as user management and audit logs.

* **Audit Logs**

Changes like adding, editing, or deleting case information to the database trigger the logging of these activities in the audit logs for future reference and accountability.

### 5.1.2 States

A listing of states (modes of behavior) that will result as a consequence of events is presented.

* **System Initialization**

This state represents the initial state of the system when it is started or reset. It sets up the necessary components and prepares the system for user interaction.

* **User Registration**

This state occurs when a user fills out the registration form and successfully gets registered for an account. The system creates a new user profile and stores the user's information in the database.

* **User Login**

This state occurs when a user enters their credentials (username and password) and successfully logs into the system. The system grants access to the user's account.

* **User Logout**

This state occurs when a user chooses to log out of the system. The system ends the user's session and returns to the login state.

* **Case Creation**

This state occurs when an agent creates a new case within the system. The system stores the case information in the database and transitions to the case-specific state.

* **Case Modification**

This state occurs when an agent updates the information of an existing case with new information or changes existing details. The system updates the case record in the database and remains in the case-specific state.

* **Search Results**

This state occurs when a user performs a search for specific case information. The system fetches and displays the relevant search results.

* **Report Generation**

This state occurs when an agent generates a PDF report containing case/client/subject information. The system gathers the necessary data and generates the report, transitioning to the report-specific state.

* **Email Sending**

This state occurs when an agent starts a mass email to a group of clients. The system prepares and sends the email, transitioning to the email-specific state.

* **Admin Access**

This state occurs when an admin user logs into the system. The system gives the admin access to admin functionalities and transitions to the admin-specific state.

* **Audit Log Recording**

This state occurs when there are changes made to the system, and the system records the activity in the audit logs. The system remains in its current state but updates the audit logs.

## 5.2 State Transition Diagrams

Depict the overall behavior of the system.



## 

## 

## 

## 

## 5.3 Control specification (CSPEC)

Depict the manner in which control is managed by the software.

# **User Authentication**

# The system has its user authentication system to grant access to only authorized users. When users provide the required information like username and password, the system processes it and matches it with the credentials already stored in the database, if it matches with the database system grants access to the user.

# **User Roles and Permissions**

# There are different types of users having different roles. The system assigns the roles to users and each role has specific permissions to the tasks. For example, the admin has more permissions and has all the management access in the system. Agents have permission to create cases. Other users have other permissions according to their roles.

# **Access Control List**

# The system uses the ACL system to control the permission that which user can perform which tasks. For example, only the admin can do user management. Unauthorized users cannot perform any task. Both user and admin can update the account information users cannot update the case information.

# **Session Management**

# This is to control the system access to only authorized users. When the user login to the system it starts a session, and all the activities are held within the session by the specific user and maintain the user’s rights. The session ends when a user logouts.

# **Workflow and Business Rules**

# It’s about the control over the system and all the actions that are performed are in sequence and being performed step by step. It also ensures the required conditions and rules and regulations.

# **Data Validation and Error Handling**

# To maintain the quality performance of the system, error handling and error prevention are implemented. It ensures that the users enter the correct information, for example creating a case where the entered data is correct and in the right data fields.

# **Audit Logs**

# Audit logs keep all the records of user activities on the system. Admin k can view and monitor all the activities of users and can take necessary actions.

# **System Configuration**

# The configuration option is for the admin who can change the settings and rules and policies of the software. Admin can update the rules according to the company’s rules.

# 

# 6.0 Restrictions, Limitations, and Constraints

**Funding**

In order for this database system to be accessible to agents remotely, the system has to be deployed through a cloud-based web service such as Amazon EC2. This service can cost anywhere from $10 to $50 monthly depending on the size of the data being stored and how often the data is accessed. The clients have been made aware of these costs and they have been working towards a solution to budget for these costs and for the team to gain access to the cloud-based web service to develop the system.

**Time**

This project has a strict eight-month development timeline. This will impact any improvements or additional features since the timeline is limited to begin with. Since the project is being developed by students, all members of the team will also have to balance other class and work responsibilities, which can greatly affect the time spent on the project compared to a full-time professional software development team.

**Experience**

The team consists of students with limited professional experience in software development. Because of this, the team will rely heavily on independent research, guidance from resources, and lack some of the skills a professional development team would have.

# 

# 7.0 Validation Criteria

## 7.1 Classes of tests

**Black-Box Testing**

The use of functional Black-Box testing will focus on how the software is working from a user’s perspective. This will include a series of test cases that are designed from each use case to guarantee that the system is operating as intended and satisfying the business requirements . Since it will be from a user-perspective, this testing will be performed later in the development process. A list of the test cases are detailed in Section 7.2 Expected Software Response.

**Unit Testing**

Unit testing is a crucial step in the development of MoonEyes. This type of testing will be used in the development of the system to ensure that each component is working properly before moving on to integration testing, as well as testing boundaries. Some of these boundaries include validating the information being entered by an agent; security measures such as an automatic sign out after 60 minutes of inactivity; and ensuring that the search parameters are displaying the correct information.

**Integration Testing**

After the successful development of unit testing, the focus will shift to integration testing between multiple units. The integration testing will be used to test the interaction between the database, cloud-based web application, and the user interface. The main focal point of the integration testing is to validate the information the agent is entering or changing is updating correctly in the database and the information can be successfully and efficiently queried.

**Performance & Smoke Testing**

Performance testing will be used to ensure the system can perform with multiple agents using it at a time and the data is readily available to access. Further details of the requirements for performance testing can be found in Section 7.3 Performance Bounds. Smoke testing will be used after the system is deployed to ensure the application is running properly in the environment and to determine if more testing is needed.

**Validation Testing**

Lastly, the use of validation testing will validate that all of the requirements of the MoonEyes have been met and implemented in the system. As well as the system is functioning as intended.

## 7.2 Expected software response

As mentioned in Section 7.1, the Black-Box test cases are provided below with the expected results.

| **Test ID** | **Use Case** | **Test Case** | **Procedure** | **Expected Results** |
| --- | --- | --- | --- | --- |
| **1** | Account Registration | Duplicate Emails | Account not created for duplicate emails | Text block with error message regarding an email already registered to an account. |
| **2** | Account Registration | Registering for an Account | Agent being able to register for an account | Screen with username and password creation will appear prompting user to create an account to access the system. Admin will be notified of every new account creation and will verify agent before access is granted to system. |
| **3** | Account Login | Invalid Login Input | Wrong information locks account after 3 invalid entries | Login screen with a text box with username and password input. Login button underneath agent input. Error message will appear after invalid information is entered 3 times. |
| **4** | Account Login | Account Login Successful | Agent enters valid information and can properly access database | Login screen with a text box with username and password input. Login button underneath agent input. Agent enters valid username and password to gain access to the database. |
| **5** | Account Logout | Automatic Logout after Inactivity | Agent is logged out after 60 minutes of inactivity | If the agent does not select the “Logout” button, the system will automatically log the agent out after 60 minutes of inactivity with a warning message in a text block of logging agent out at the 55 minute mark. |
| **6** | Account Logout | Agent is able to logout of system | Agent is able to successfully logout of the system | The agent selects the “Logout” button, they are able to successfully exit the system and returned to the “Login” screen. |
| **7** | Case Management | Invalid Information to edit a case | Agent edits case file with invalid information or duplicate cases | The agent enters invalid information or the system recognizes the duplication of a case, a text box error message will appear rejecting the edits because of invalid information or duplicate cases. |
| **8** | Case Management | Valid Information to edit a case | Agent enters valid information to edit a case | The agent enters valid information when changing the information on a case. After the agent presses the "Save" button, a successfully saved message will appear. |
| **9** | Case Management | Audit Logs | Admin is able to generate an audit log of all the changes made to the system from who edited the system, when they did it, and what was changed. | The admin is able to select a button called “Reports” under their account information and select the “Audit Changes” tab. The admin can choose from a daily, weekly, monthly, or yearly report on all the changes that have occurred in the database. The report will include which agent made the changes, when the changes occurred, and what was changed. |
| **10** | Search | Invalid Search Critera | Agent inputs invalid search criteria | An error message will appear in a text box if the user does not enter valid information in order to conduct a search. |
| **11** | Search | Valid Search Critera | Agent enters valid search criteria | The user enters valid search parameters and the system will display all cases matching the provided information. |
| **12** | Search | Create a PDF | Agent can download a PDF verison of the search results | After an agent completes a search, along the top search bar, there will be a “Save to PDF” button, that will enable the search results to be downloaded to a PDF format for the agent based on the entered search parameters. |
| **13** | Search | Mass Marketing Email | Agent can mass email certain clients based off of search parameters | After an agent completes a search, along the top search bar, there will be a “Email Clients” button, that will allow the agent to send a mass email to all clients that appear in the search. |
| **14** | Update Account Information | Invalid information to update their account | Agent inputs invalid information into any field with attempting to change their account information | An error message in a text box will appear to warn the user that the information entered is not valid and to try to reenter valid information in order to change account information |
| **15** | Update Account Information | Valid information to update their account | Agent inputs valid password change or other account information | The user will be able to edit their account information and press the "Save" button on the bottom of the screen. A "Saved Successfully" message will appear letting the user know the information was successfully updated. |
| **16** | Help | Agent selects the “Help” button | “Help” button provides an agent with the information pertaining to the functionality of a specific sections within the sytem | The agent presses the “Help” button, which would then display a screen of different headers with information about the database. The agent is able to select the header they would like more information or directions on, which would cause a different screen to appear with the information regarding the header topic. |

For the performance testing, we expect the system to be able to handle the expected traffic and usage from the agents and administrator. For the smoke testing, we expect the system to function as intended with no additional testing at the time needed.

For unit and integration testing, we are expecting to run across small errors and bugs that can be easily detected and fixed during the development process.

## 7.3 Performance bounds

The MoonEyes case management software will currently be used by 3-4 agents on a weekly, if not daily basis. There is also the possibility that the number of agents will grow as the business continues to grow. The amount of cases an agent is working on can range from 10 to 300 in a year depending on the complexity of the cases. Some major investigations may require 20 or more hours of labor, while others may only take an hour or two. The system will need to be able to handle the variations in cases, as some may require more storage and usage than others. Also, some of the cases may not contain as much information as others, such as having an agent tied in a case in the beginning stages or the storage of client information as a future contact. The information will still need to be readily accessible via the search feature whether the data entered is limited or plentiful.

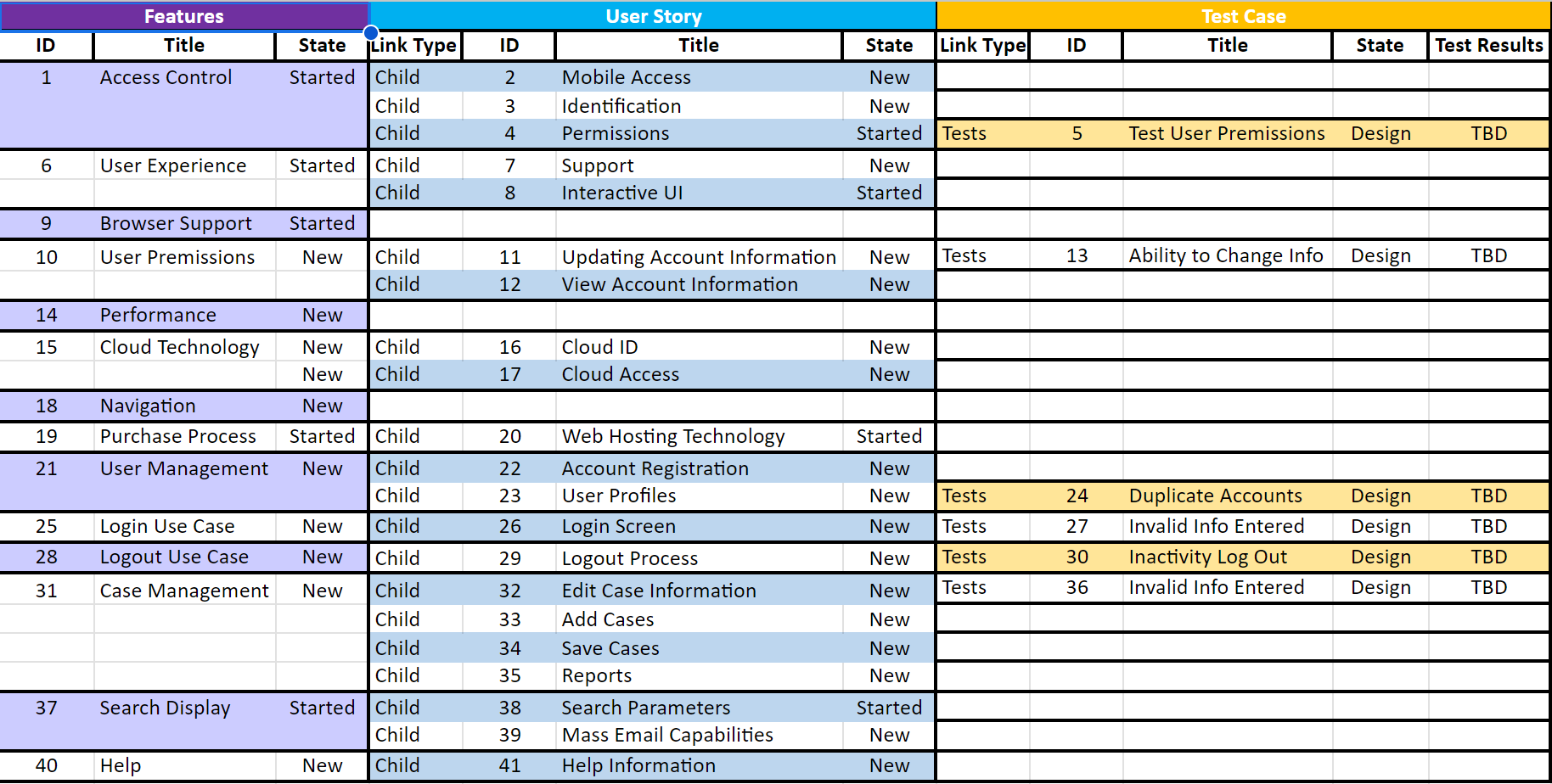
In addition, the administrator of the system will be able to generate audit logs to authenticate the information being changed or added to the system. This audit log will show what the agent has accessed, when the agent accessed the information, and what information was changed or entered. This audit log may need to be generated on a daily, weekly or monthly basis depending on the administrator’s discretion.

# 8.0 Appendices

Presents information that supplements the Requirements Specification

## 8.1 System traceability matrix

A matrix that traces stated software requirements back to the system specification.



## 8.2 Product Strategies

If the specification is developed for a product, a description of relevant product strategy is presented here.

**User-Focused Design**: System users like admins and agents must have a user-friendly and easy-to-use interface so they don’t face any difficulties while performing a task. A survey could be an option to know what exactly a common user is expecting so the system can be developed according to the user’s perspective rather than they learn about it. However, the main purpose is to provide a better user experience.

**Data Security and Privacy:** As the nature of this software is very sensitive and it contains the data of different types like the stakeholder’s information such as admins, clients, and agents therefore the security of this system is one of the most crucial aspects. The system should meet security and protection standards and implement a reliable user authentication that has a secret encryption function to keep the passwords and protection keys safe.

**Remote Access and Cloud Deployment:** According to the requirements system must be available for users from any place and any system like computers or mobile phones so they can perform their tasks remotely. It should be a web-based system hosted on a cloud-based hosting and should have compatibility with all browsers and operating systems.

**Comprehensive Case Tracking**: The agent is one of the main stakeholders of the system, he should be able to add data about the new clients and cases. The system should provide the agent the facility to enter all the relevant details of the case so it must have all the fields in the related form for example case ID, media(images or video), and time and date. It should also have a functionality to search anything like case id or name of client or agent because if the user wants to track any case or client he should be able to do that.

**Reporting and Communication:** The feature of generating reports in pdf form also should be available for agents and they should be allowed to send these reports to clients and related authorities. To communicate with clients and marketing he should also be able to send bulk emails as well.

**Audit Logs and Admin Controls:** There should be also an admin of the system who has access to all the information and changes agents do in the system. Admin should be allowed to edit any case data and can view all the other details related to the changes and modifications. Admin should have all other controls for user management.

**Scalability and Performance:** The system must have the ability to deal with a large number of users. For example if users increase, the system should still work properly and there should be no effect on the performance and speed of the system.

**Documentation and Training**: To provide a better user experience there must be documentation for users to guide them on how to use the system and how they can perform their desired task. The system itself should be responsive so users don’t face any kind of difficulties in use.

**Continuous Improvement and Maintenance:** For the system's betterment and improvement there should be user feedback and in case of bugs and errors it also will help to improve system performance.

## 8.3 Analysis metrics to be used

A description of all analysis metrics to be used during the analysis activity is noted here.  
  
**User Adoption Rate:** To know about the system's acceptability we use the user adoption rate matrix because this is based on the number of sign-ups and signs on the system during a particular time frame. It’s directly proportional to the system's acceptability.

**User Engagement:** User engagement as it’s clear by its name that it’s about the user interaction on the system. We use this metric to calculate how much the system is useful for the users. Therefore if users are using the software more often and doing different activities like using different features of the system that means they are getting used to the system and it’s eventually benefiting them.

**System Performance**: To ensure the full use of the system it’s a must to keep check on system performance. For example, performing different activities like entering a new date or creating a new user system speed should be as per requirement. By this metric analysis, we can make our software more useful and efficient.

**Error Rate:** To improve the software every day and make it more efficient we need to reduce the bugs and errors in the system, that’s why we use this metric to solve the bugs in software. These errors may occur while performing different tasks using the software for example while tracking or creating a new case in the system.

**Search Efficiency:** It's one of the main features of this software because whenever a user would track a case or admin or other users want to see some information or data they will use the search functionality so it should work fast to fetch the information from the database and result should be accurate. Therefore to improve search results we make this metric and analyze the results.

**Security and Privacy:** Security is the one of the main aspects of the system so to ensure the protection of the data from unauthorized users and other malicious attacks we make this matrix to evaluate system capacity to ensure the security and privacy of the data. This metric also analyzes if the software security system meets the standards or not.

## 8.4 Supplementary information (as required)

**User Feedback:** To improve software quality and performance user feedback is very crucial because they are using it in an actual environment and can give a fair opinion about the performance and issues in the system. Surveys, interviews, and feedback forms can be good to gather user feedback about the system. They can highlight the good and bad areas where the system is performing as per requirement is having problems, and they also can send some recommendations to improve the system. Eventually, this will help to improve the system’s quality.

**User Demographics:** For a better understanding of the unique demands and preferences of users of the system it will be helpful to learn about the demographics of the end user. This will help to target the exact needs of a particular group of users. If some users are looking for some special features we can easily understand and will be able to help them by developing that.

**Training and Support Materials:** For better user experience this is the best idea to give the training and some helpful material that will help them to use the system and perform their specific tasks using this software. They could have a better idea about the software and its functionalities.

**System Usage Data:** Collecting data on system usage can provide valuable insights into user behavior and patterns. This data may include the frequency of logins, the most commonly used features, and the time spent on different tasks. Analyzing this information can help identify popular features, potential bottlenecks, or areas where users may need additional support.

Data on users’ interactions with the software and their activities is the best way to learn about user behavior. Based on mostly used features by specific users and how much time they spent on this system, we can sort out what are the areas we can focus on because those are functionalities mostly used by users and we can improve them.

**System Updates and Enhancements:** Changes according to the needs and feedback of the users must be kept on record because that is the record of how we have upgraded the system as per the end user’s needs.

## 8.5 References

This requirement specification was created using the latest Pressman format.The Cyber Rovers “WMITS Software Requirements Specifications” found on McGraw Hill website was used as reference. Additionally, the textbook “ *Software Engineering: A Practitioner's Approach”* was used as a reference throughout the document.

* R. S. Pressman, “Document Templates: Software Requirements Specification,” *R.S. Pressman & Associates, Inc..* [Online]. Available: <http://www.rspa.com/docs/Reqmspec.html>. [Accessed: 14-June-2023].
* “Software requirements specification,” *McGraw Hill Higher Education*. [Online]. Available: <http://www.mhhe.com/engcs/compsci/pressman/graphics/Pressman5sepa/common/cs1/sw-req.pdf>. [Accessed: 14-June-2023].
* Pressman, Roger S., and Bruce R. Maxim. *Software Engineering: A Practitioner's Approach* . Vol. 9, McGraw-Hill Education , 2020.