Design Document

for

surakshIIT

Version 0.1

Prepared by

Group Name: **GITHUB SPAMMERS**
Course: **CS253**
Mentor TA: **Mr. Aman Aryan**
Date: **30-01-2022**

Group Details

Name	**Roll No.**	**Email ID**
Aarchie	200004	aarchi20@iitk.ac.in
Arpit Kumar	200189	arpitk20@iitk.ac.in
Harsh Jain	200412	harshj20@iitk.ac.in
Kajal Deep	200483	kajald20@iitk.ac.in
Kuldeep Singh Chouhan	200530	kuldeepsc20@iitk.ac.in
Mannem Rishwan Reddy	200562	mannemr20@iitk.ac.in
Rohit	200813	rohitg20@iitk.ac.in
Sandeep Kumar Bijarnia	200856	sandeepb20@iitk.ac.in
Udit Prasad	201055	uditp20@iitk.ac.in
Yash Raj Mittal	201148	yashrajm20@iitk.ac.in

Revisions

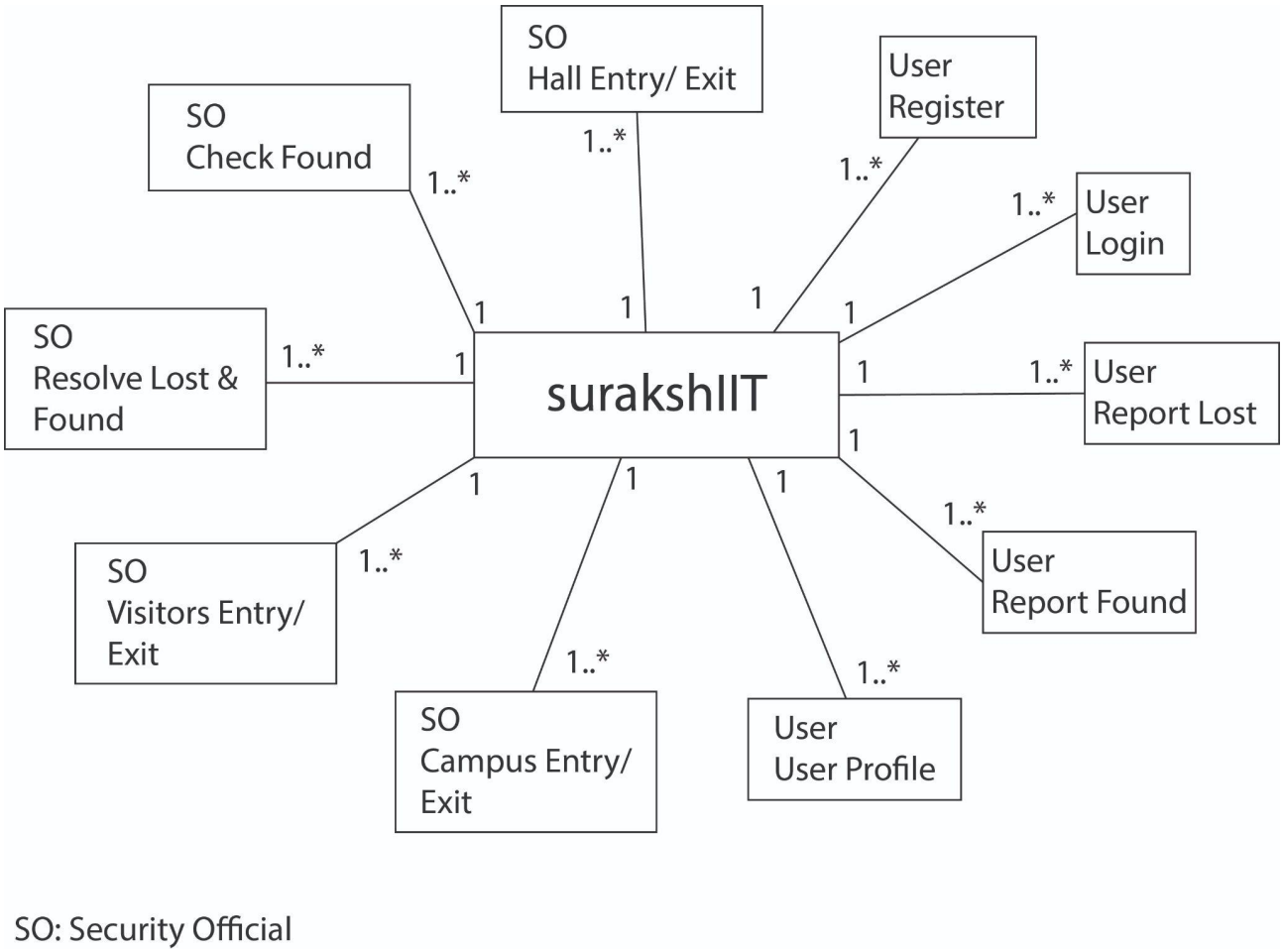
Version	Primary Author(s)	Description of Version	Date Completed

0.1	Udit Prasad Aarchie Kajal Deep Mannem Rishwan Reddy Arpit Kumar Kuldeep Singh Chouhan Sandeep Kumar Bijarnia Harsh Jain Rohit Yash Raj Mittal	Initial Draft	15/02/22

1 Context Design

1.1 Context Model

A **context model** (or context modelling) defines how context data is structured and maintained (It plays a crucial role in supporting efficient context management).



Context Model

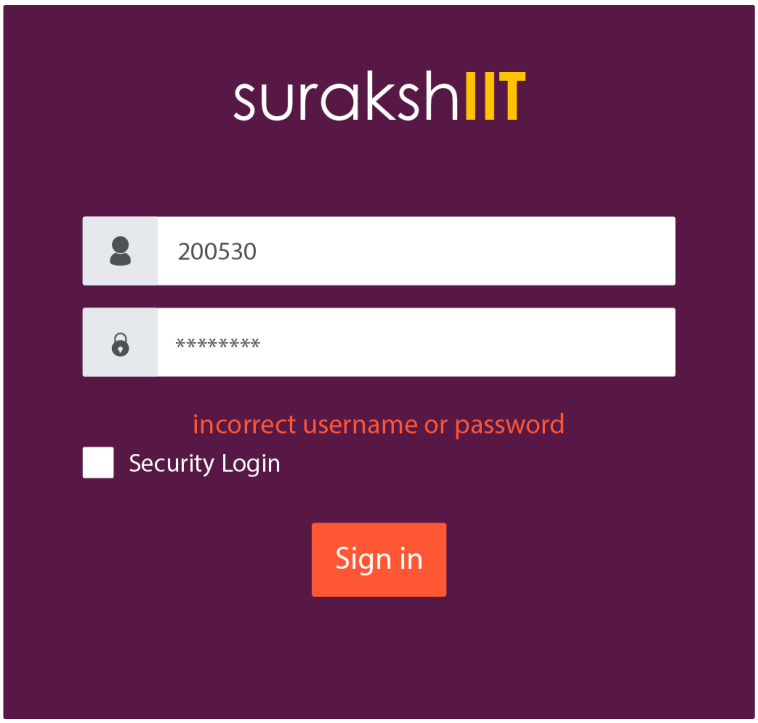
1.1.1 A brief explanation of SurakshIIT Context Model:

- **User Register:** A **User** will register in the SurakshIIT database, whereas SurakshIIT will receive registration requests from multiple users.

- **User Login:** A **User** will log in in SurakshIIT software, whereas SurakshIIT will receive Login requests from numerous users.
- **User Report Lost:** Users will report Lost items in SurakshIIT, whereas SurakshIIT will receive lost item entries from multiple users.
- **User Report Found:** Users will register unknown items found in SurakshIIT, whereas SurakshIIT will receive these entries from various users.
- **User Profile:** Users will access and update profile information in SurakshIIT software, while SurakshIIT will receive similar requests from multiple users.
- **SO Campus Entry or Exit:** Security Officials will update information about those Non-Security Officials willing to enter or exit Campus in SurakshIIT software, whereas SurakshIIT will receive such entries from multiple SO deployed at various locations.
- **SO Visitors Entry or Exit:** Security Officials will update visitors willing to enter or exit Campus in the SurakshIIT database, whereas SurakshIIT will receive such entries from multiple SO deployed at various locations.
- **SO Resolve Lost & Found:** Once an unknown object is reported lost or an unknown object is found, SO will update the DB.
- **SO Check Found:** Once the student is identifying a lost item, SO will verify whether the lost item belongs to the student or not and update data in the SurakshIIT database; SurakshIIT will receive such requests from multiple SO's.
- **SO Hall Entry or Exit:** If a student wants to visit another Hall then the SO will update the data of the student in the SurakshIIT database, and SurakshIIT will receive such requests from multiple SO's
- Object is being reported found then security officials will update relevant information in SurakshIIT whereas SurakshIIT will receive such entry requests from multiple SO deployed at various locations.

1.2 Human Interface Design

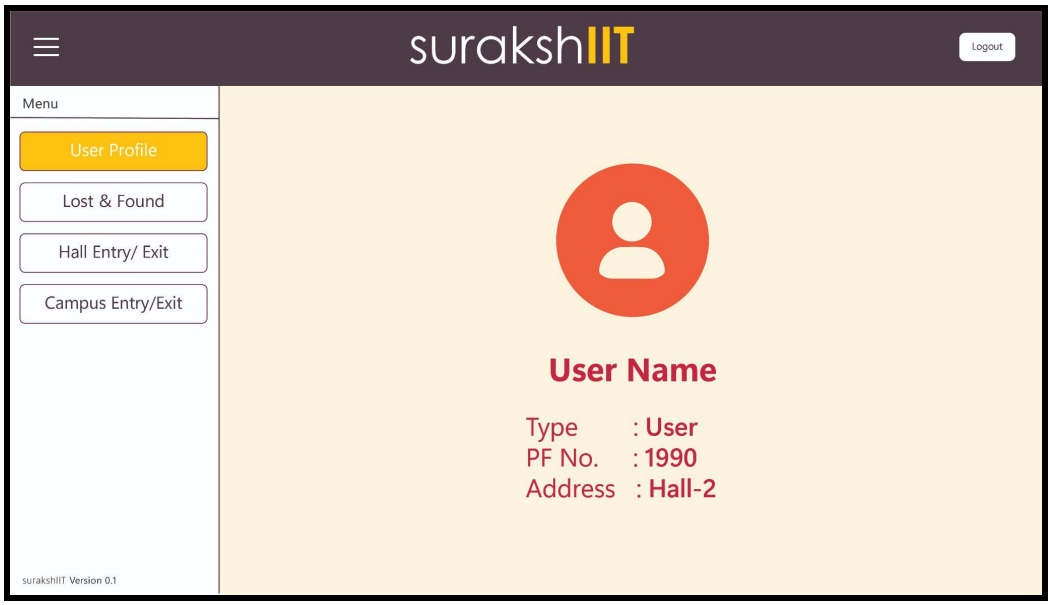
- Users will interact with the system through a website, which would have a simple to use UI and work cross-platform on different OS and browsers.
- When the website opens, User is required to log in using his UIN and Password. For SO, one need to tick the check box for Security Login.



Login Screen

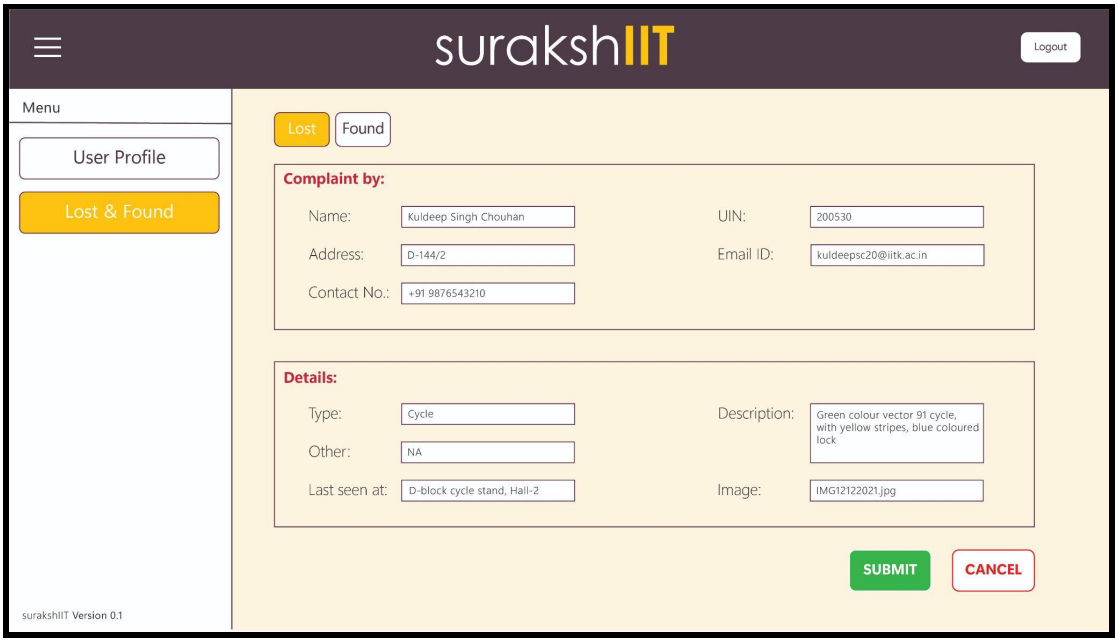
- After Login User will see his profile, with his details extracted from the CC. There would be a menu to find the **Lost & Found** section. There would be some other options, but they would be visible only to the **SO**. These would be for the **Entry/ Exit** at the **Hall** and **Campus Gates.**

Screen for User



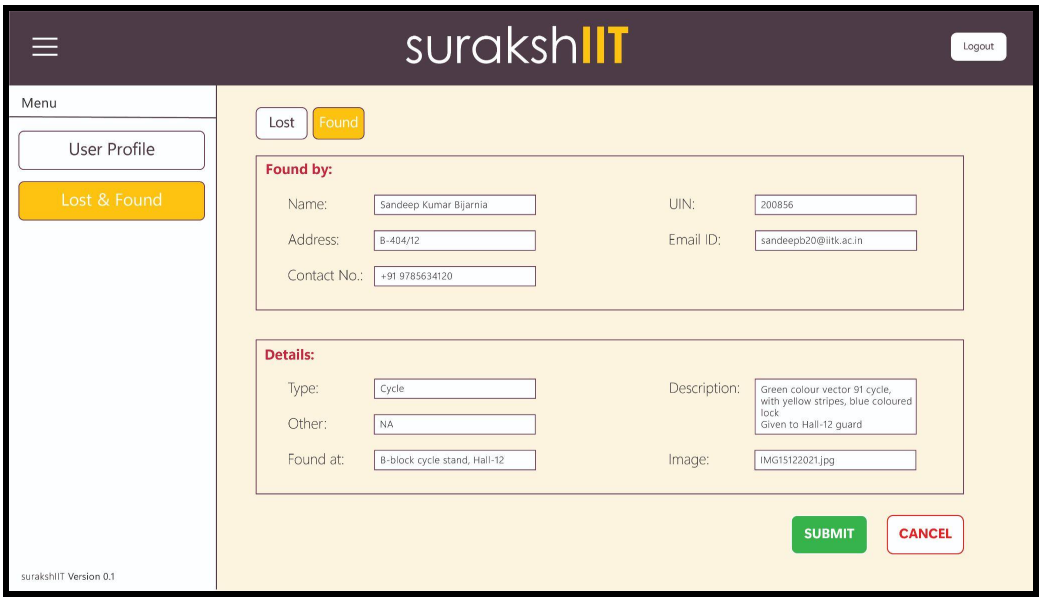
Screen for Security Official

- For reporting a lost item, the User needs to go to the **Lost & Found** tab and click on the **Lost** button. There he needs to fill in details of the lost item, and his details like Roll Number, Address, Contact Number and Email ID would be auto-filled from the data extracted from the CC. There would be some predefined types of lost things, which are the most lost objects, like cycles, ID Card, etc. Both SO and User would access this portal and could use this to report any lost item.



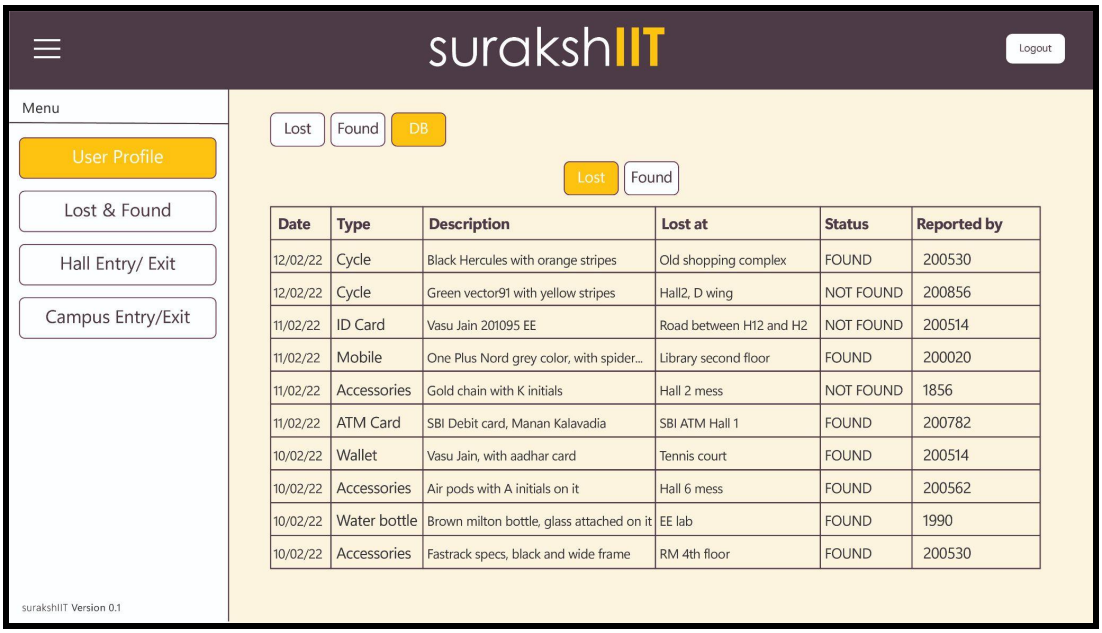
Section for reporting Lost things

- Same applies to the Found portal, which could be opened by clicking the **Found** button.



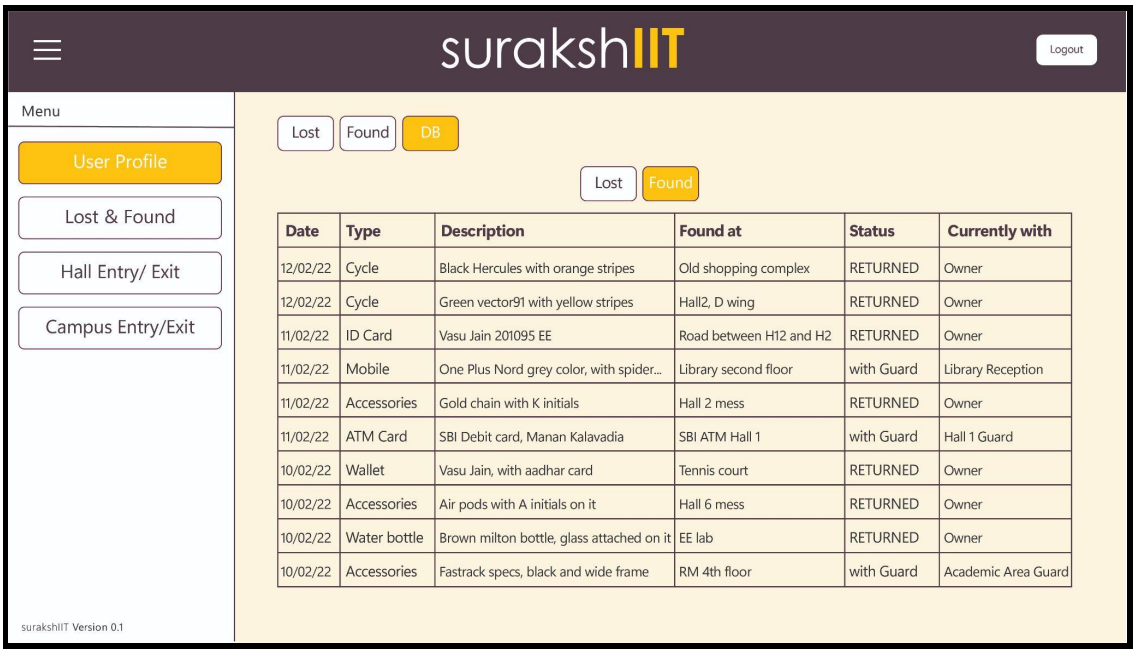
Section for updating Founded items

- For SO, there would be an extra tab, DB, which would showcase a list of lost and found items, with their brief description. Use the 'Lost' and 'Found' tab for switching between the lists. Keywords **FOUND** and **NOT FOUND** would be used to differentiate between which items have been found. Meanwhile, **RETURNED** would illustrate that item has been sent back to the owner and **with Guard** stats that the thing is with a guard and can be obtained from him.



Date	Type	Description	Lost at	Status	Reported by
12/02/22	Cycle	Black Hercules with orange stripes	Old shopping complex	FOUND	200530
12/02/22	Cycle	Green vector91 with yellow stripes	Hall2, D wing	NOT FOUND	200856
11/02/22	ID Card	Vasu Jain 201095 EE	Road between H12 and H2	NOT FOUND	200514
11/02/22	Mobile	One Plus Nord grey color, with spider...	Library second floor	FOUND	200020
11/02/22	Accessories	Gold chain with K initials	Hall 2 mess	NOT FOUND	1856
11/02/22	ATM Card	SBI Debit card, Manan Kalavadia	SBI ATM Hall 1	FOUND	200782
10/02/22	Wallet	Vasu Jain, with aadhar card	Tennis court	FOUND	200514
10/02/22	Accessories	Air pods with A initials on it	Hall 6 mess	FOUND	200562
10/02/22	Water bottle	Brown milton bottle, glass attached on it	EE lab	FOUND	1990
10/02/22	Accessories	Fastrack specs, black and wide frame	RM 4th floor	FOUND	200530

Table showcasing the Lost items

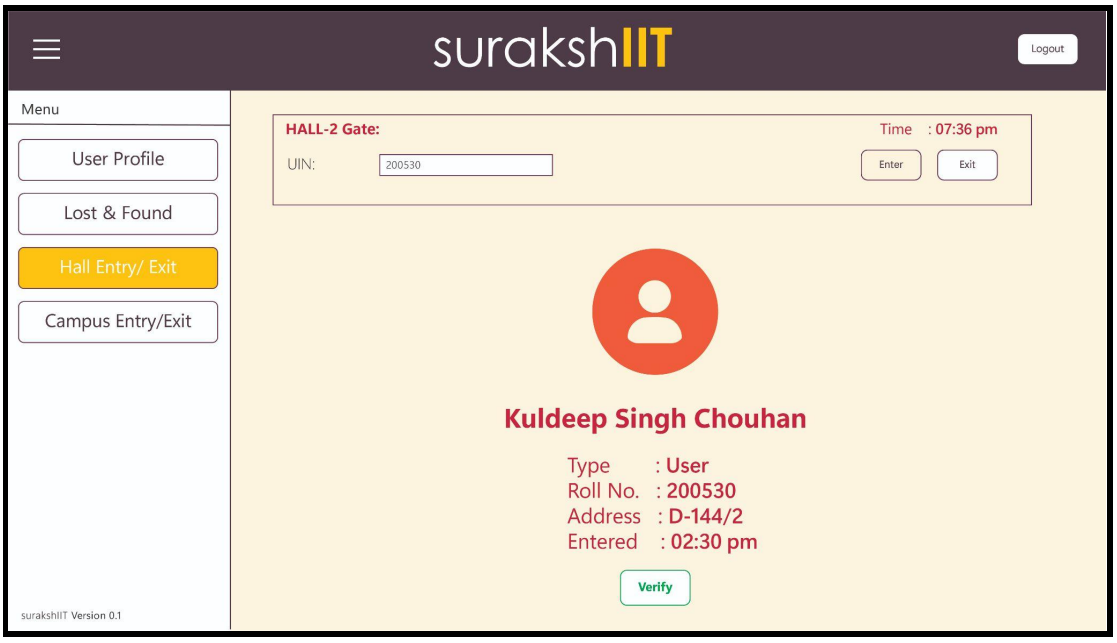


Date	Type	Description	Found at	Status	Currently with
12/02/22	Cycle	Black Hercules with orange stripes	Old shopping complex	RETURNED	Owner
12/02/22	Cycle	Green vector91 with yellow stripes	Hall2, D wing	RETURNED	Owner
11/02/22	ID Card	Vasu Jain 201095 EE	Road between H12 and H2	RETURNED	Owner
11/02/22	Mobile	One Plus Nord grey color, with spider...	Library second floor	with Guard	Library Reception
11/02/22	Accessories	Gold chain with K initials	Hall 2 mess	RETURNED	Owner
11/02/22	ATM Card	SBI Debit card, Manan Kalavadia	SBI ATM Hall 1	with Guard	Hall 1 Guard
10/02/22	Wallet	Vasu Jain, with aadhar card	Tennis court	RETURNED	Owner
10/02/22	Accessories	Air pods with A initials on it	Hall 6 mess	RETURNED	Owner
10/02/22	Water bottle	Brown milton bottle, glass attached on it	EE lab	RETURNED	Owner
10/02/22	Accessories	Fastrack specs, black and wide frame	RM 4th floor	with Guard	Academic Area Guard

Table showcasing the Founded items

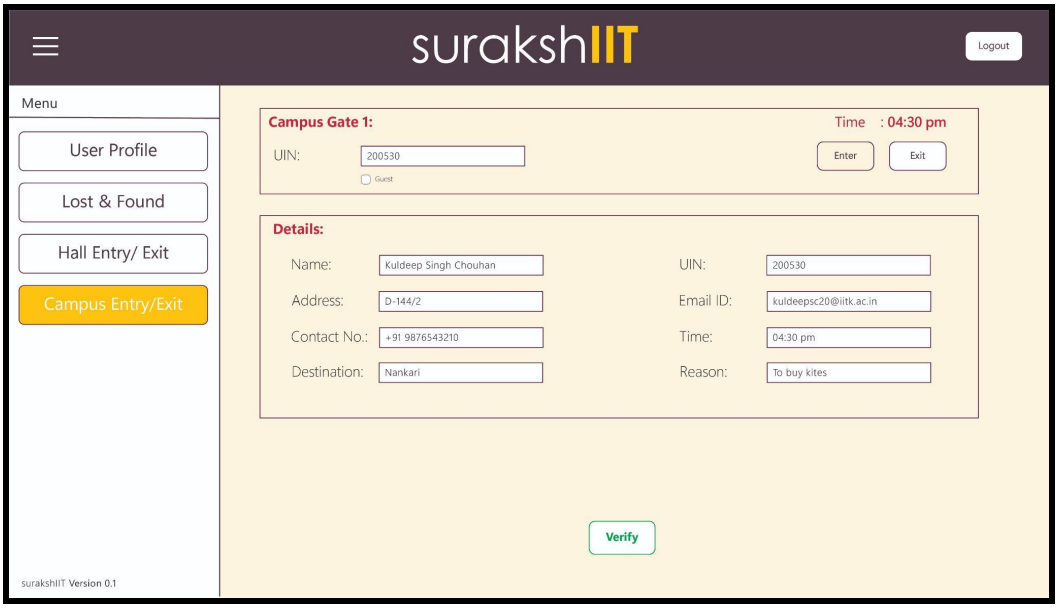
- For entrance in Halls, User needs to go to the **Hall Guard,** and he/she will enter their Roll Number and will verify his identity. The time of entrance will be stored in the DB. This time will show up at the time, the person will exit from the Hall.

Verification of the person entering

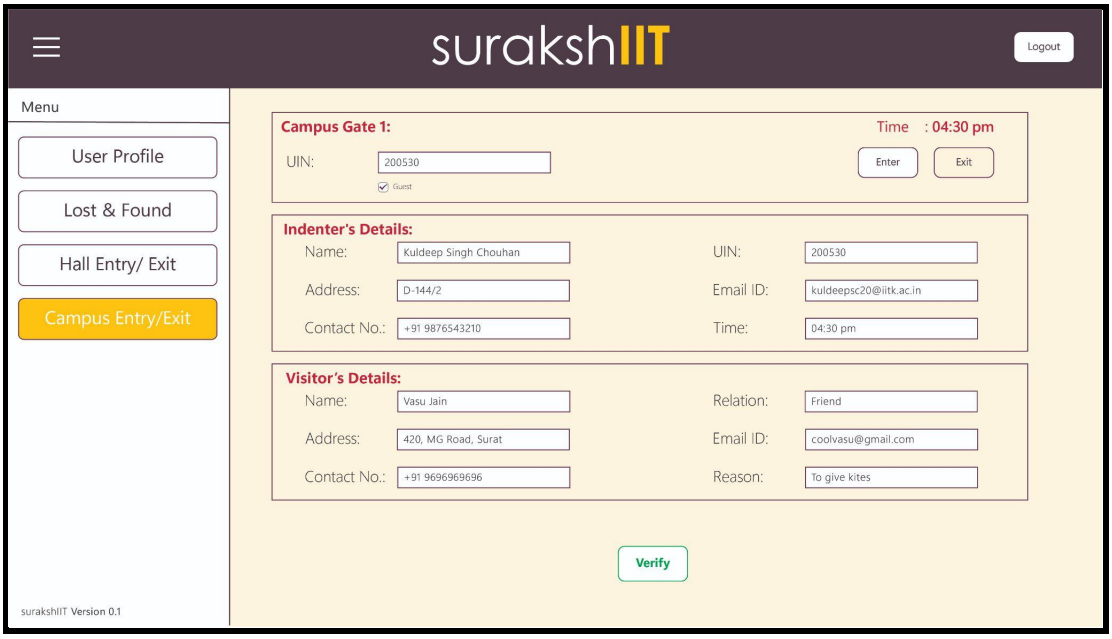


Verification of the person exiting

- For **Campus Entry/Exit** similar steps would be taken. The user has to enter the **destination**, where he is going and the **reason** for the same. For a guest, one needs to tick the check box **Guest**. The visitor needs to enter their **Intender's detail** along with their information. For a vehicle's entry, it would record its number along with the DL number of the driver.



Exit portal for a registered User



Guest's entry from Campus Gate

2 Architecture Design

Model-View-Controller Pattern

This architecture pattern is mainly used for Web applications. As our software is also a web application, we utilize this architecture pattern.

Model

We have four different databases for our application:

- User Database
- Lost and Found Database
- Hall entry/exit Database
- Campus entry/exit Database

Controller

We have three main parts here:

- Authorization
- Requests
- Validation

In Authorization, we have Login.

Requests will be sent according to different needs like

- Lost and Found
- Hall entry/exit
- Campus entry/exit

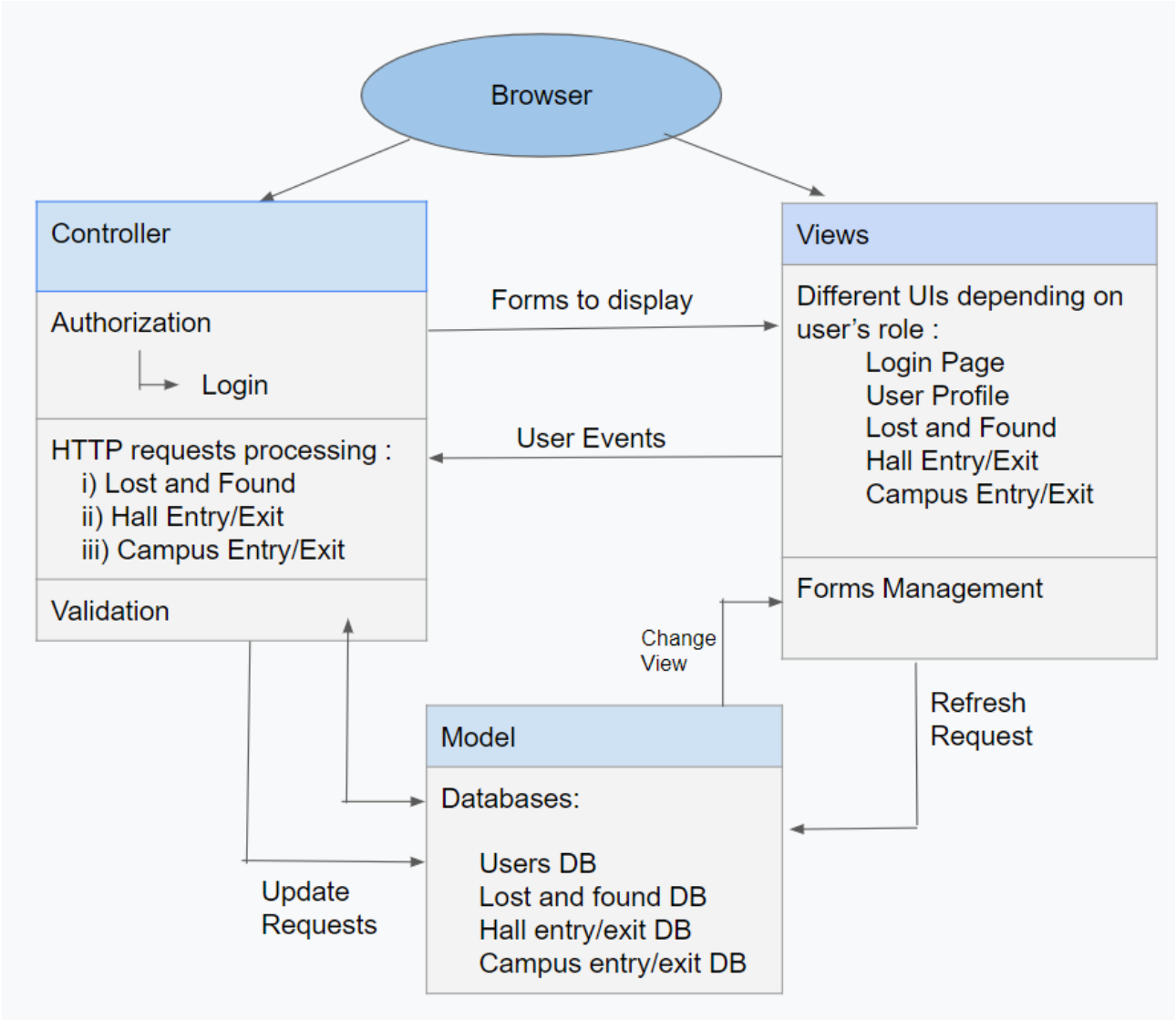
View

We will have different views according to the permissions given to the user
Mainly we have:

- Register and a Login view
- User Profile view
- Lost and Found Portal view
- Hall and Campus entry/exit View

Now, we will look at the links between these three

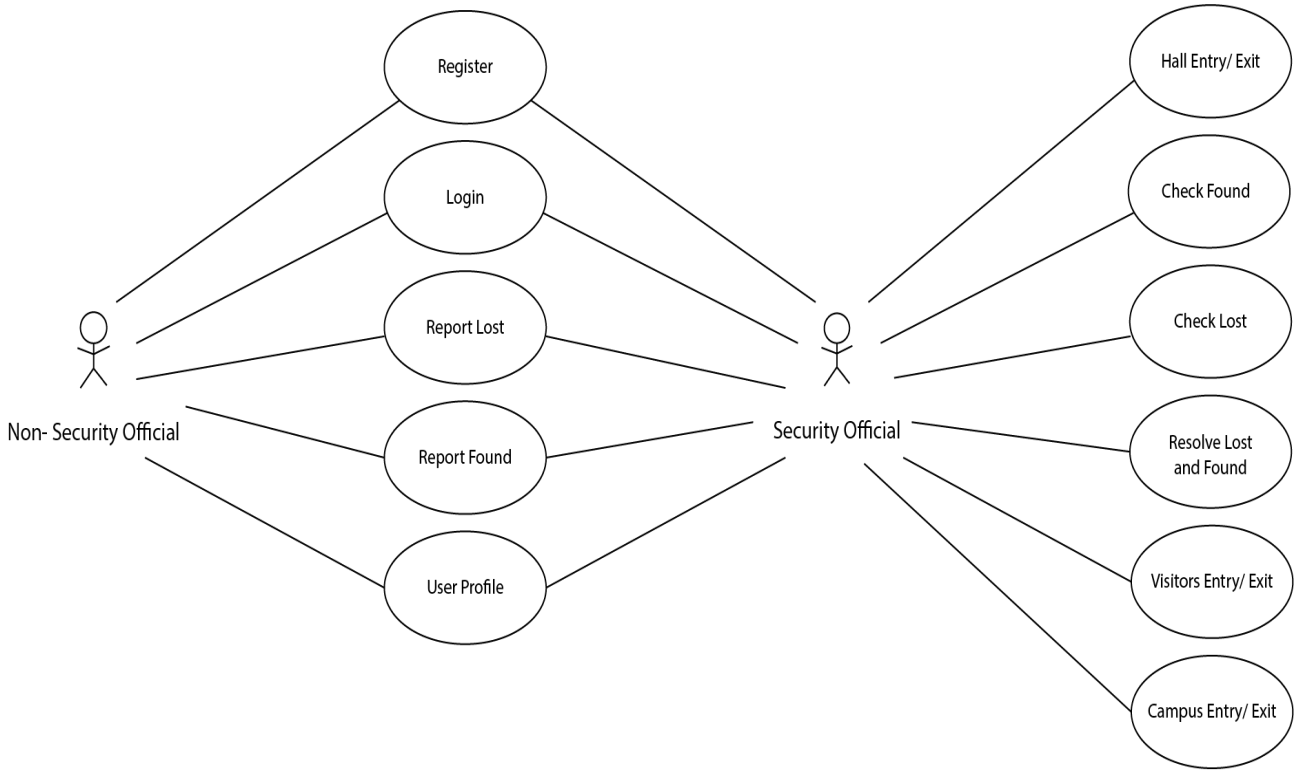
- The controller will communicate with the Model for Authorization and Validation of the user for sending requests to the Model to update the databases.
- The controller will communicate with views to control the user interface, and from user can send different commands to the controller using views.
- Then the model will communicate with the views to change the views.



Graphical representation of Architecture-pattern

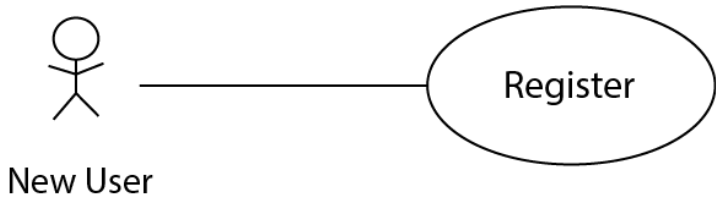
3 Object Oriented Design

3.1 Use Case Diagrams

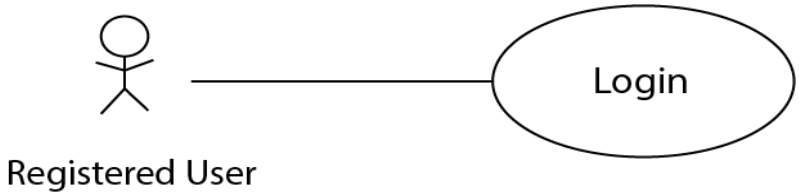


High level Use case diagram of surakshIIT

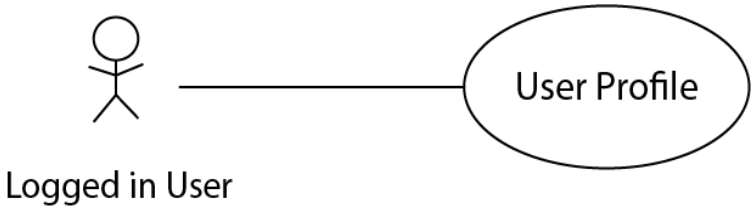
Now we will see various use case diagrams for different functionalities in the later part of this subsection



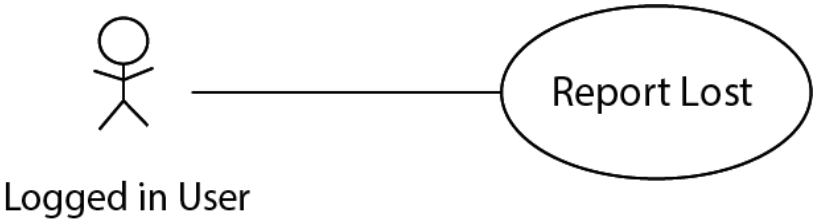
Register Use Case



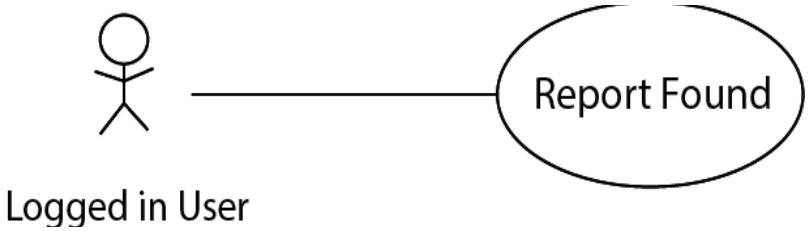
Login use case



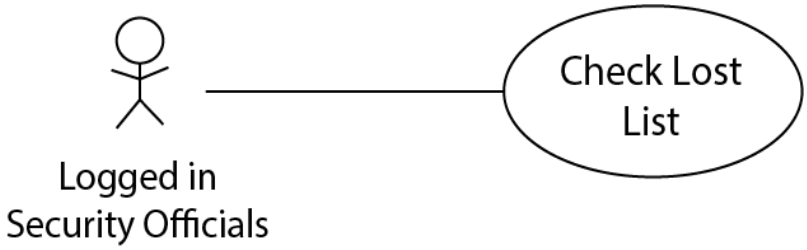
Fetch user Profile Use case



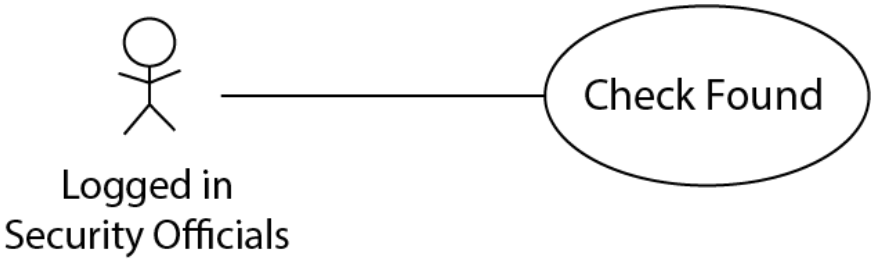
Report Lost Use Case



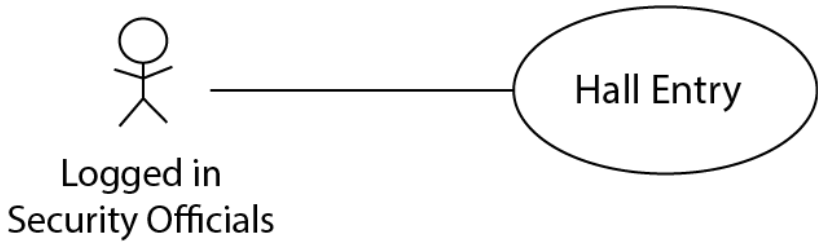
Report Found Use Case



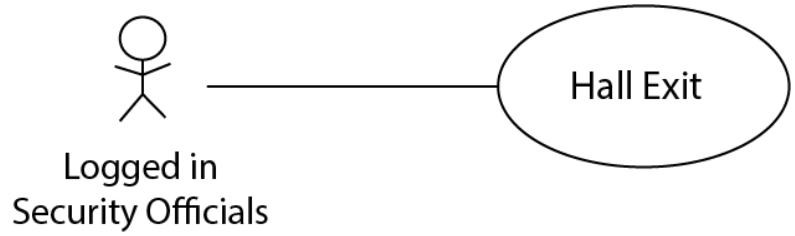
Check Lost items Use case



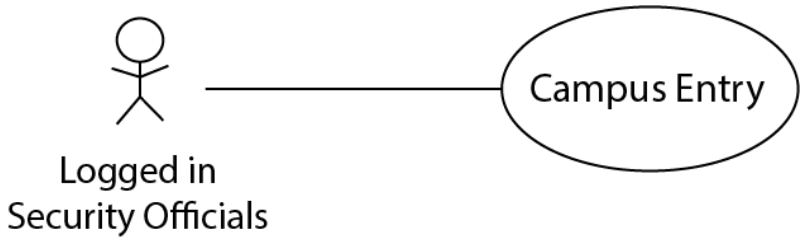
Check Found items Use case



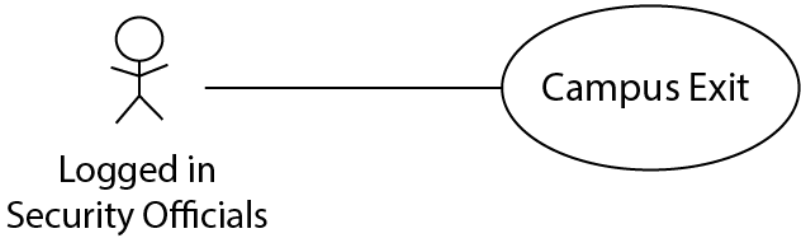
Hall entry use case



Hall exit use case

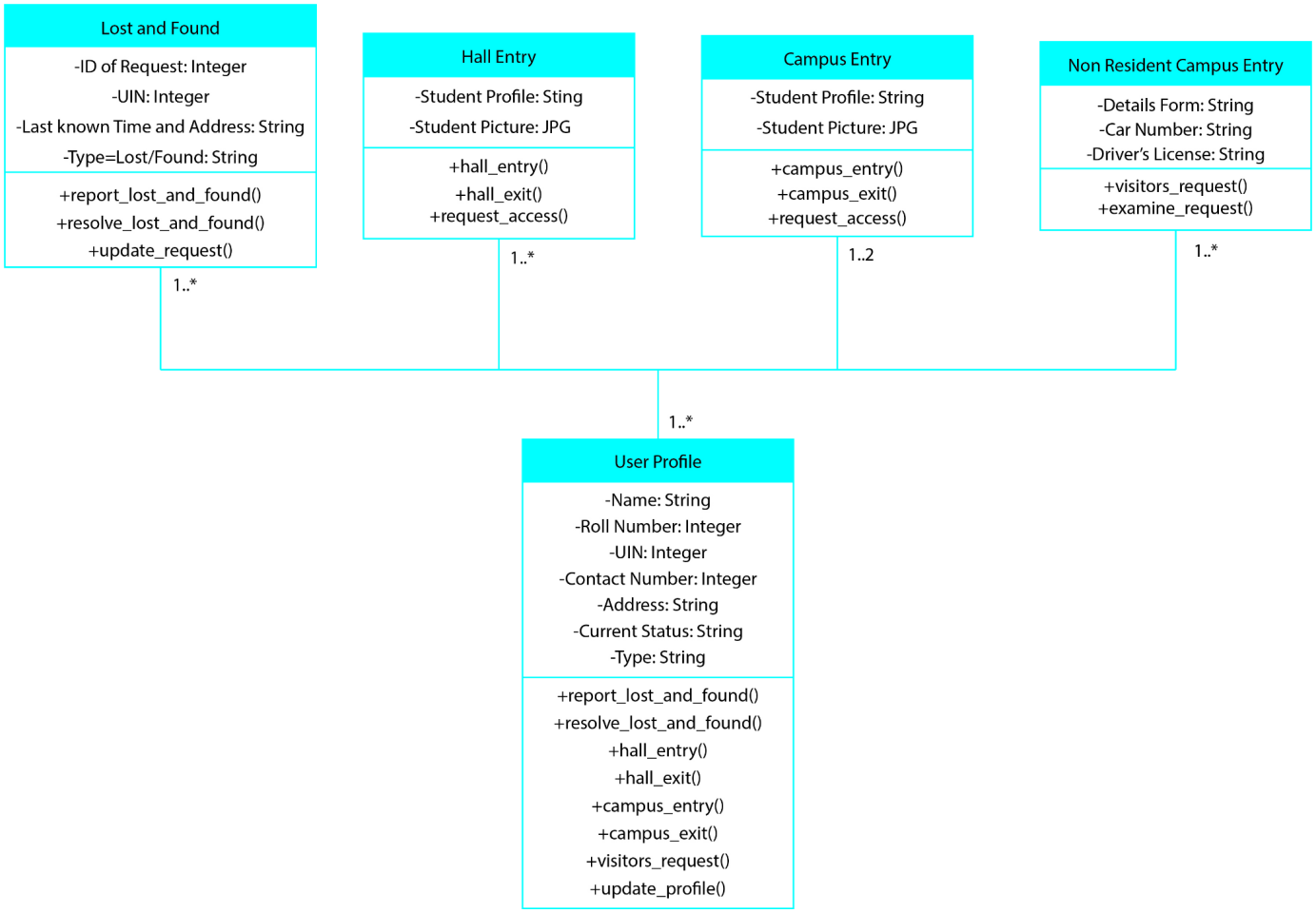


Campus entry use case



Campus exit use case

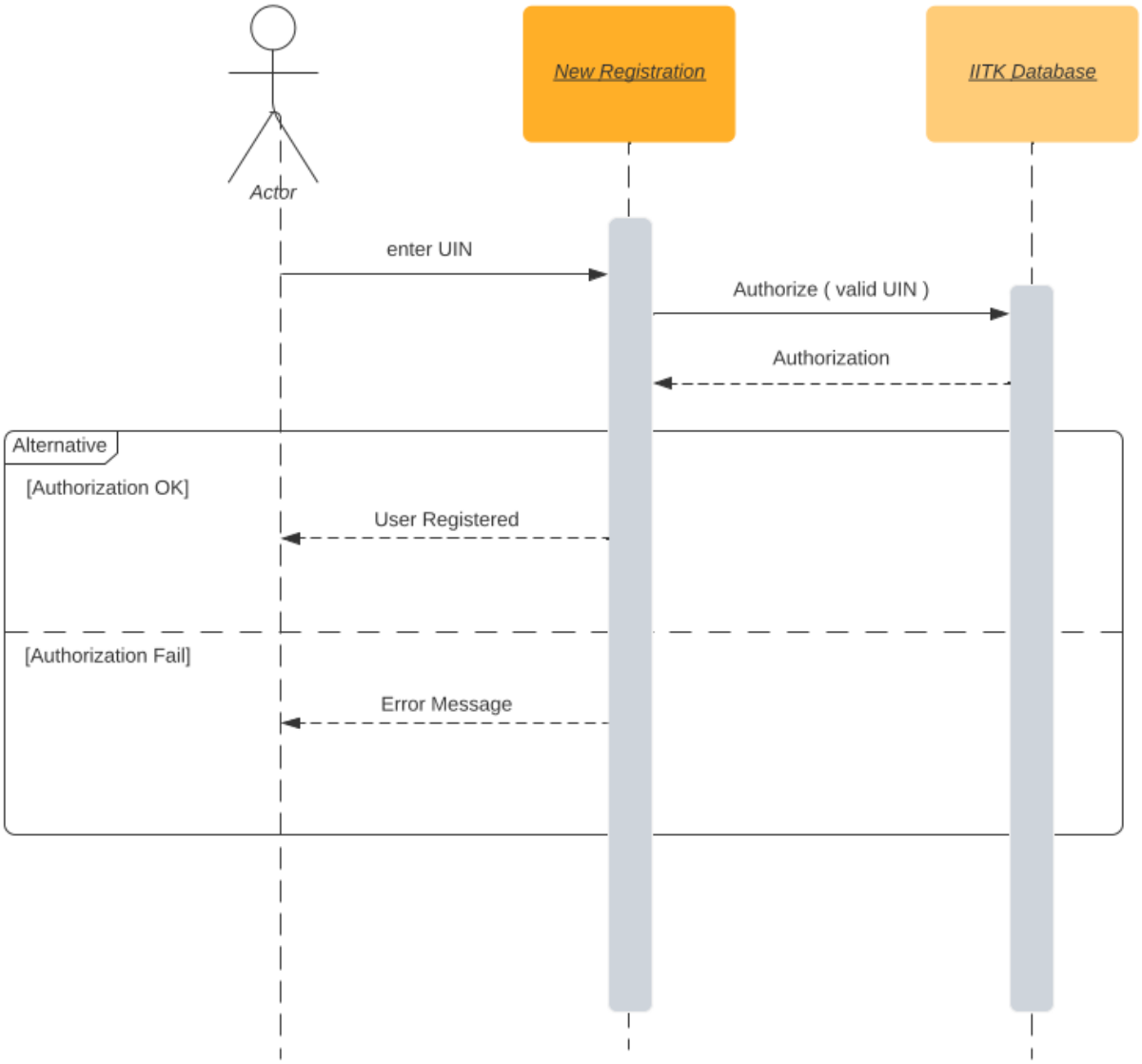
3.2 Class Diagrams



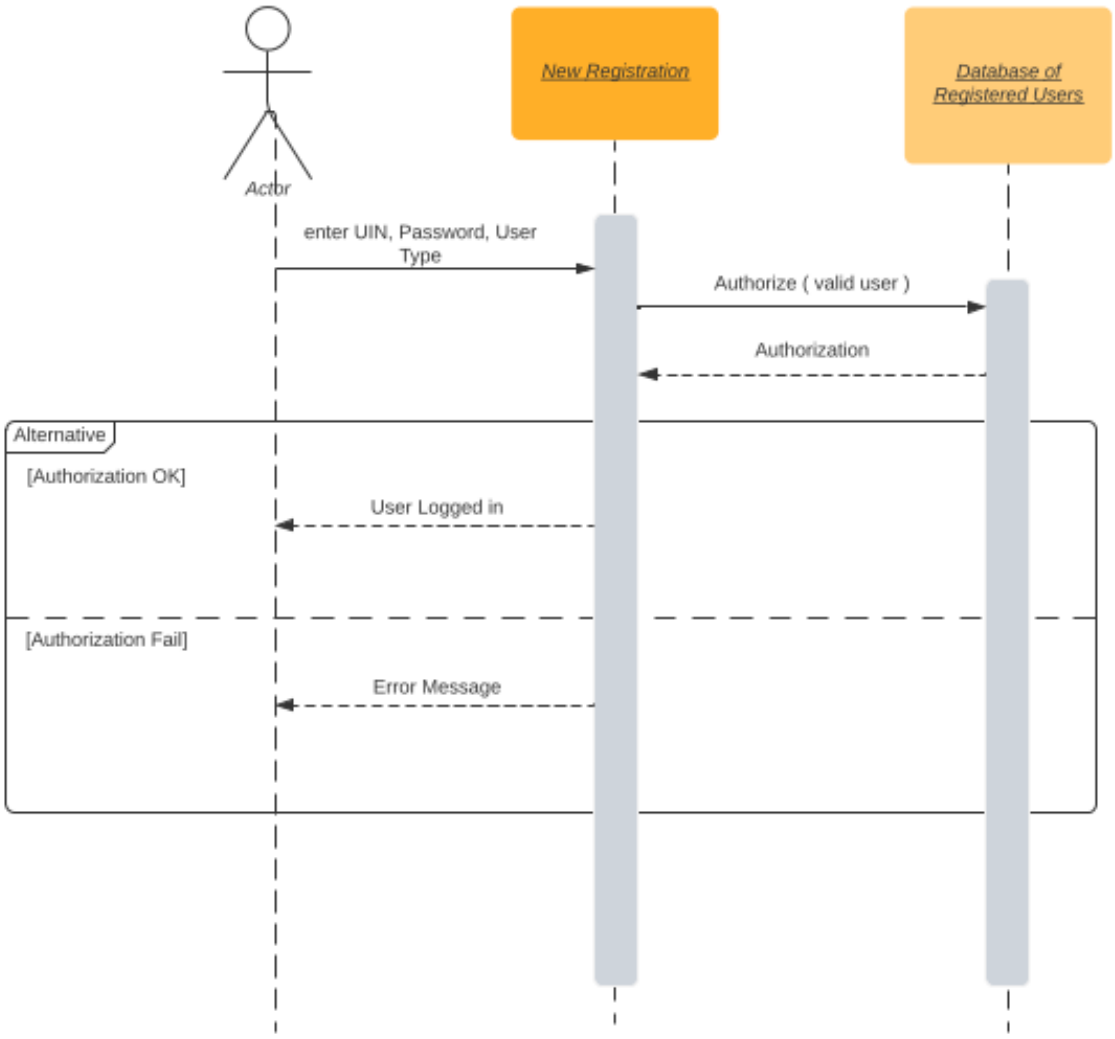
This is how all the objects in the project will interact with each other. Depending on the access given, the profile object will have a set of functions that will make calls to different objects and request services. For ex., from the profile, we use report_lost_and_found(), which will create a Lost and Found object with its ID, time, type, etc., to uniquely identify it, which will also have its function like update_request().

3.3 Sequence Diagrams

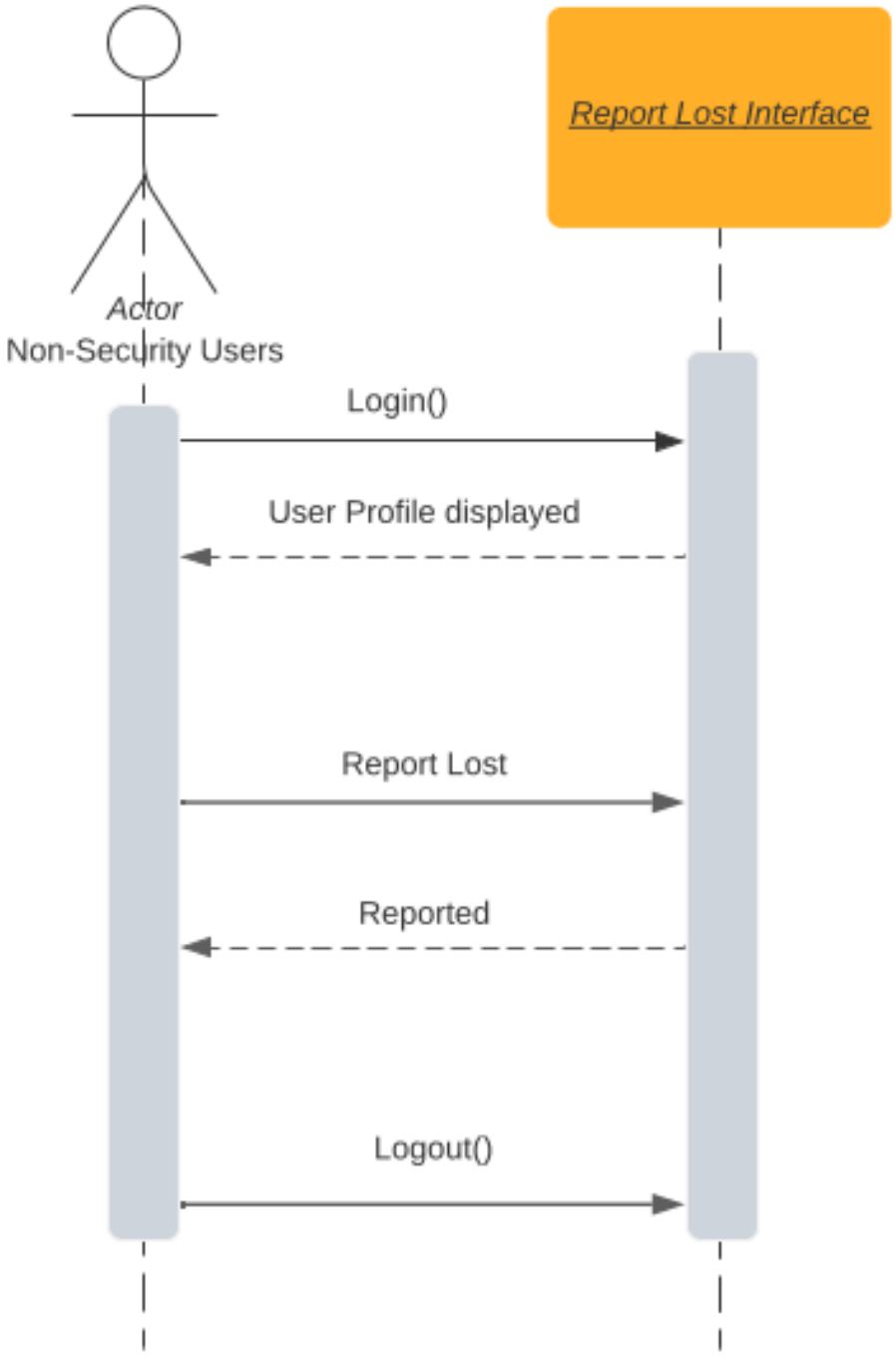
3.3.1 Register Sequence Diagram



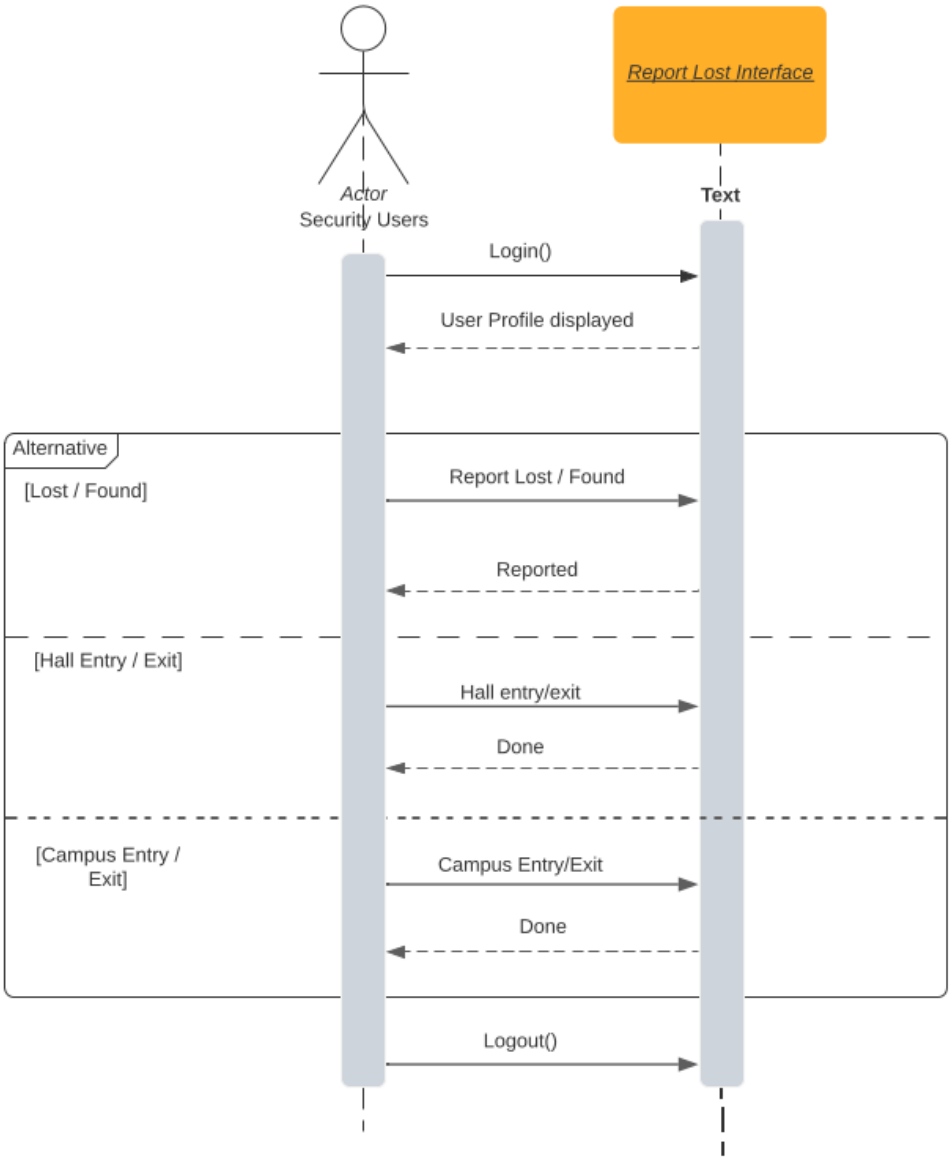
3.3.2 Login Sequence Diagram



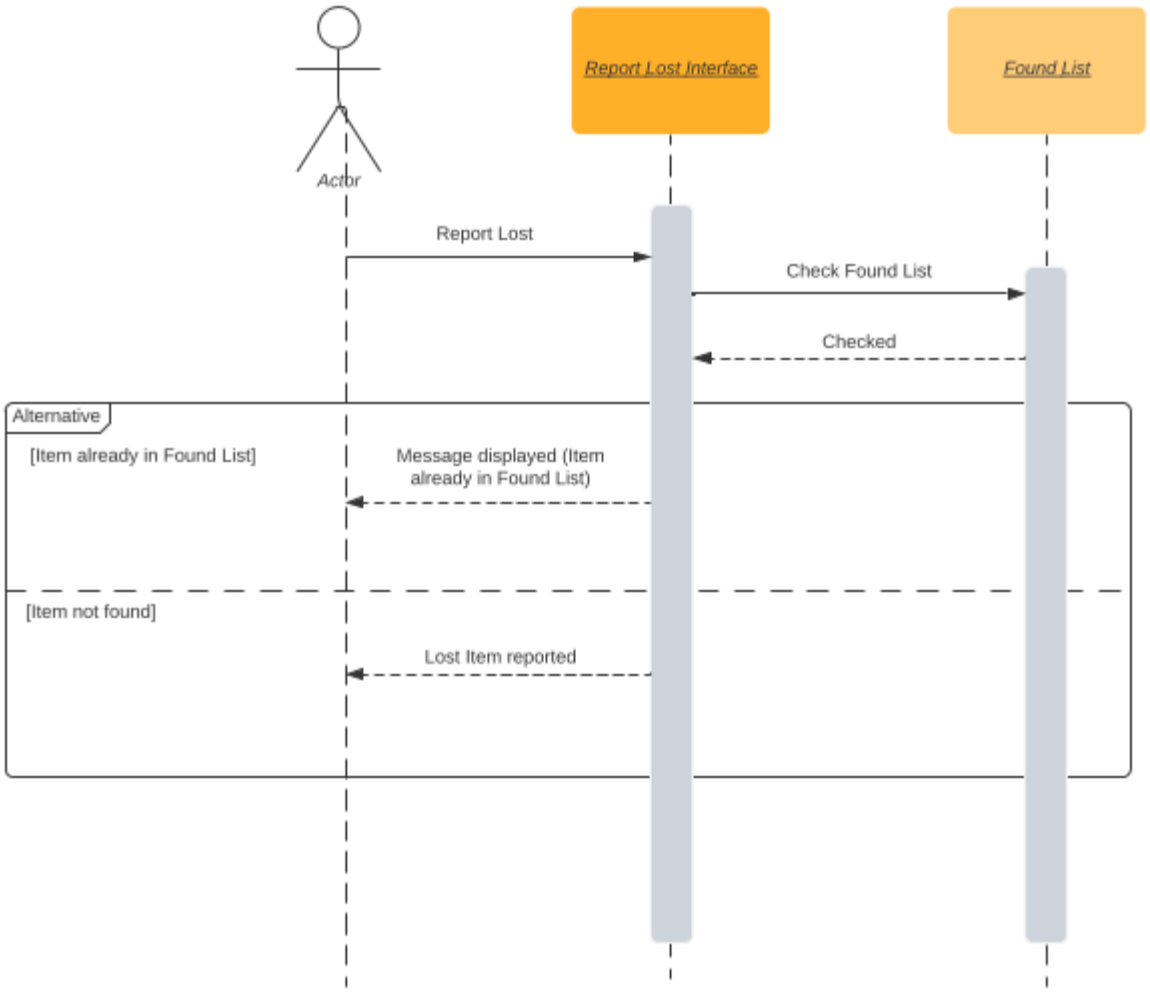
3.3.3 High level Sequence Diagram for Non-Security Officials



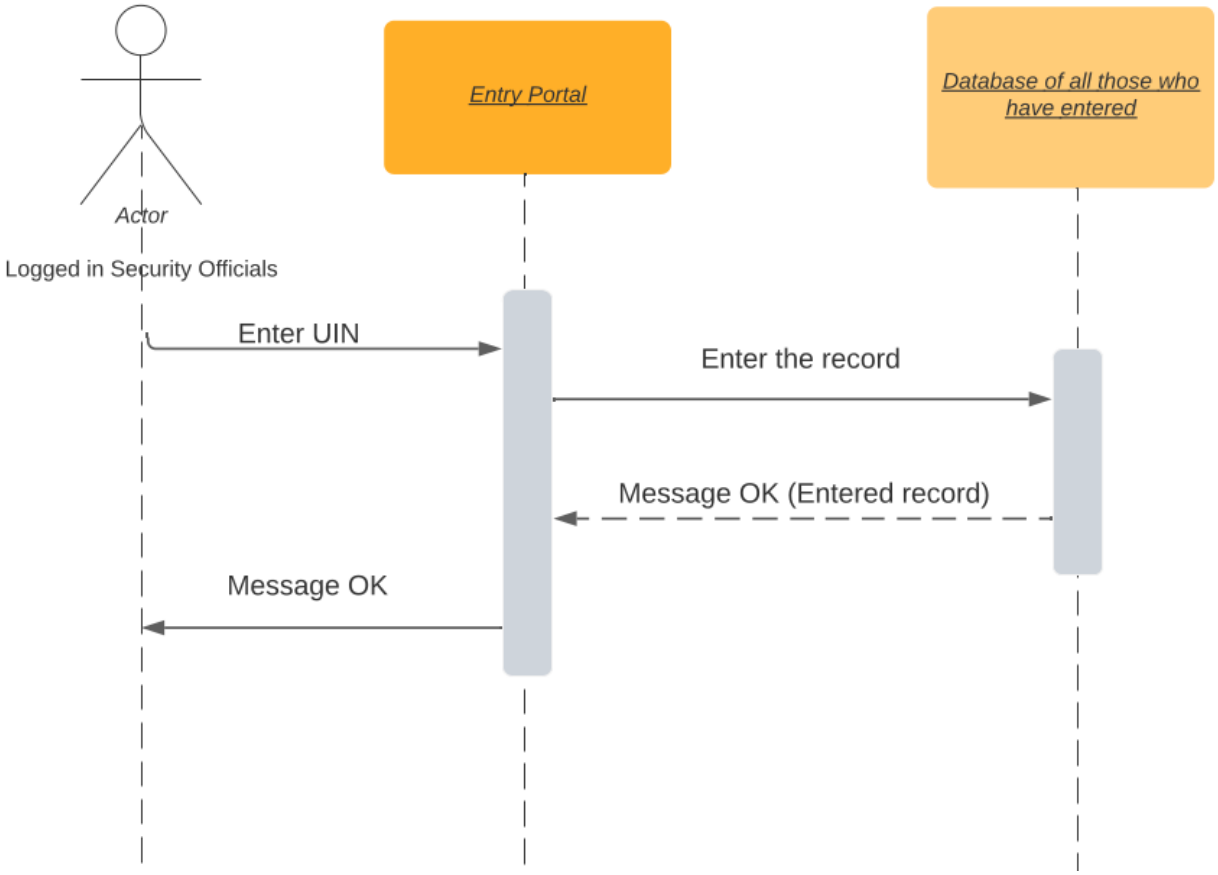
3.3.4 High level Sequence Diagram for Security Officials



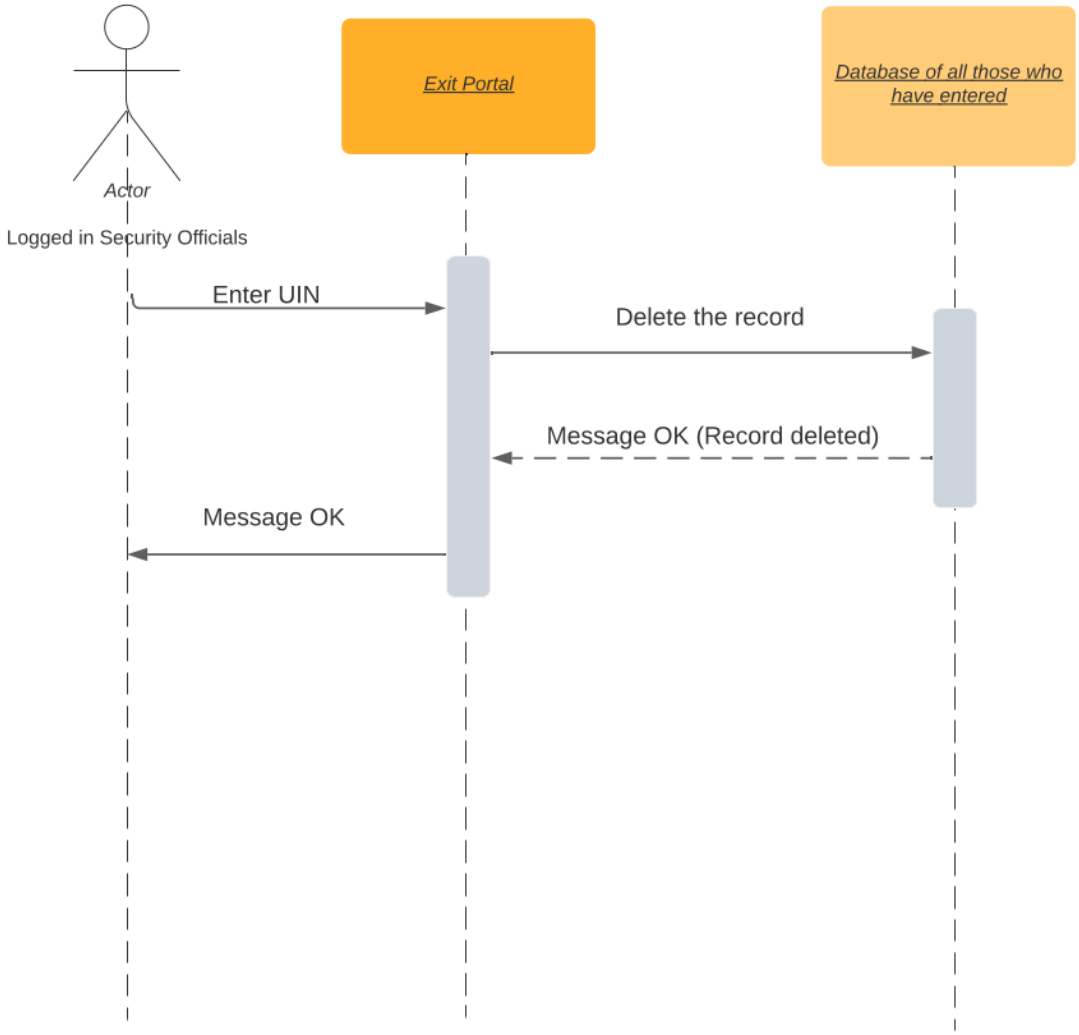
3.3.5 Report Lost Sequence Diagram



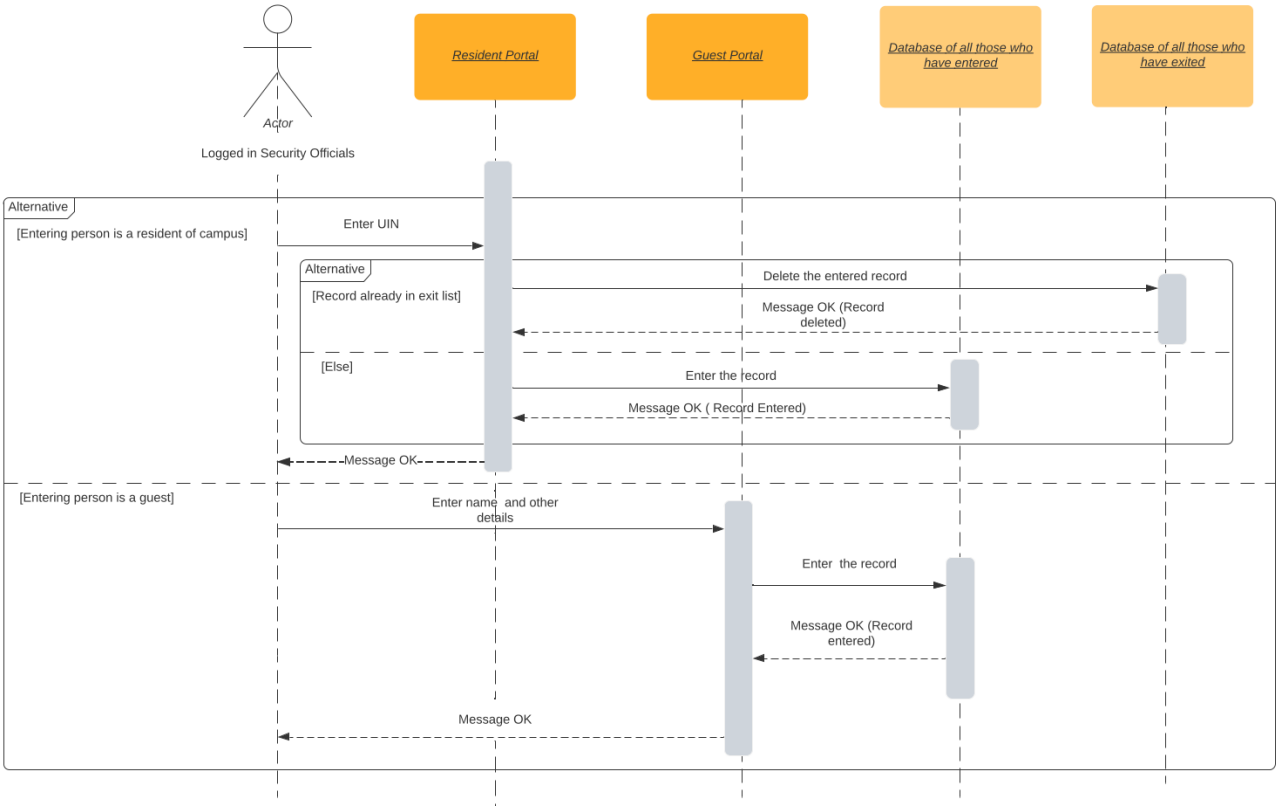
3.3.6 Hall Entry Sequence Diagram



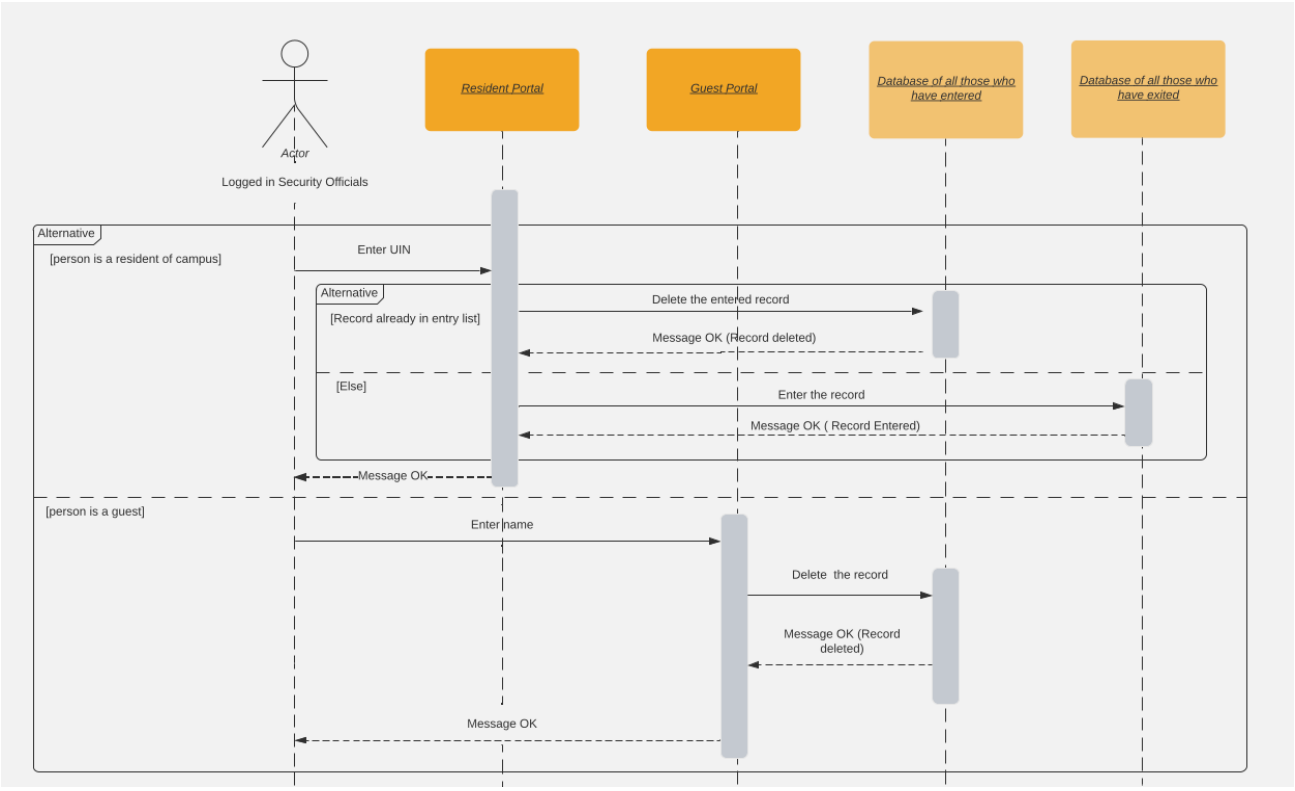
3.3.7 Hall exit Sequence Diagram



3.3.8 Campus Entry Sequence Diagram

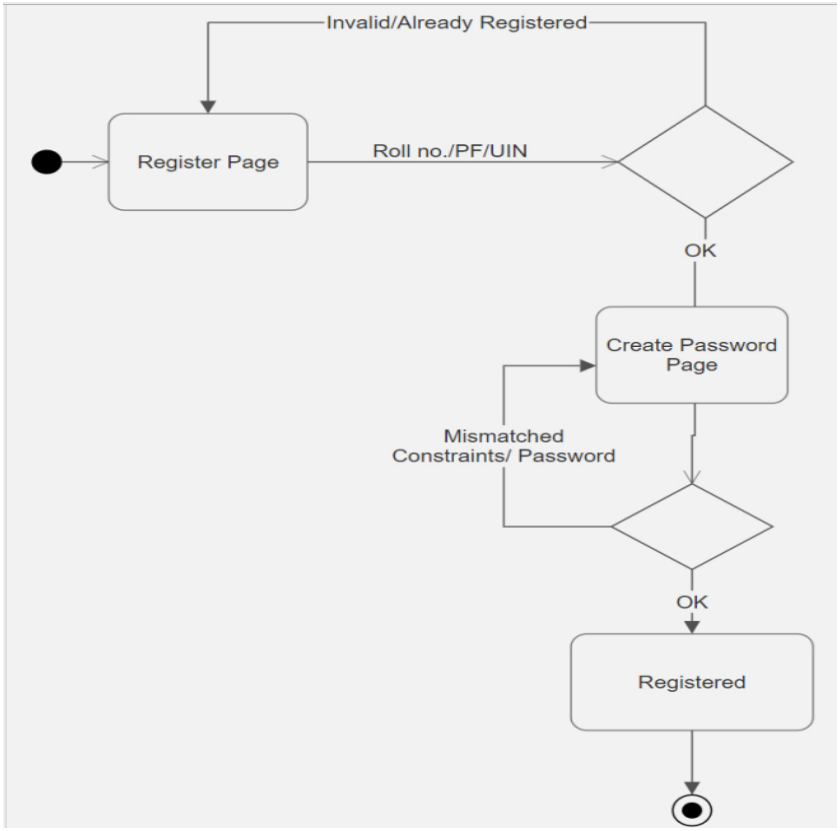


3.3.9 Campus Exit Sequence Diagram

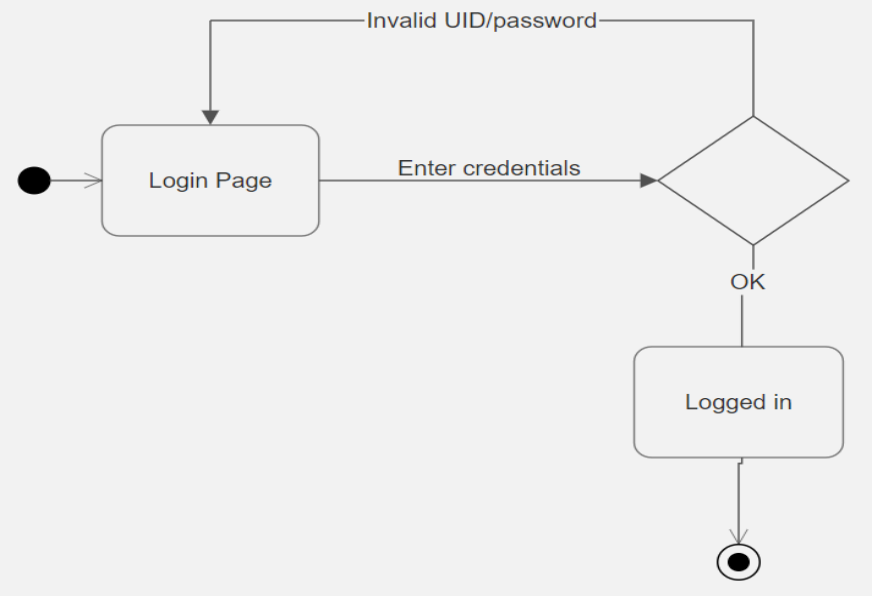


3.4 State Diagrams

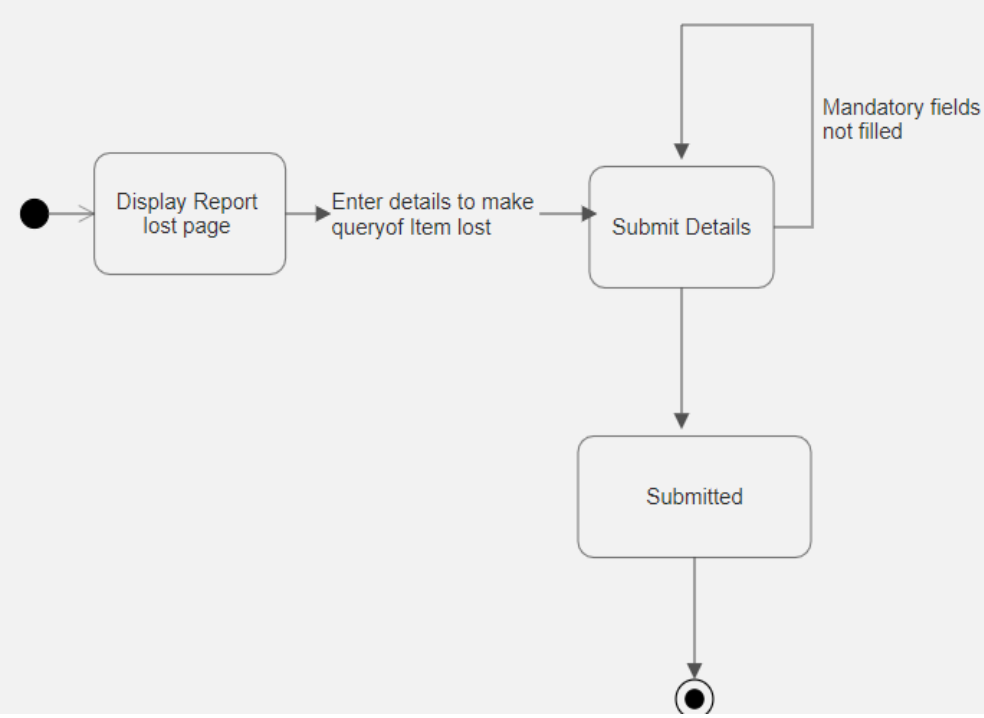
3.4.1 Register



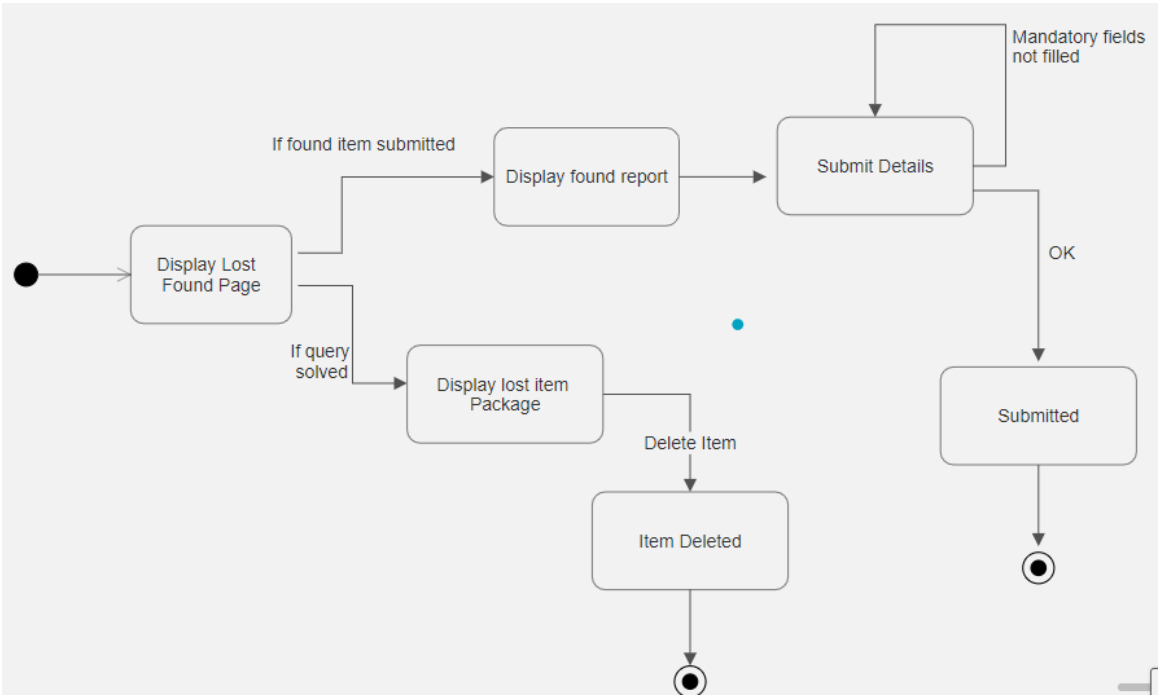
3.4.2 Login



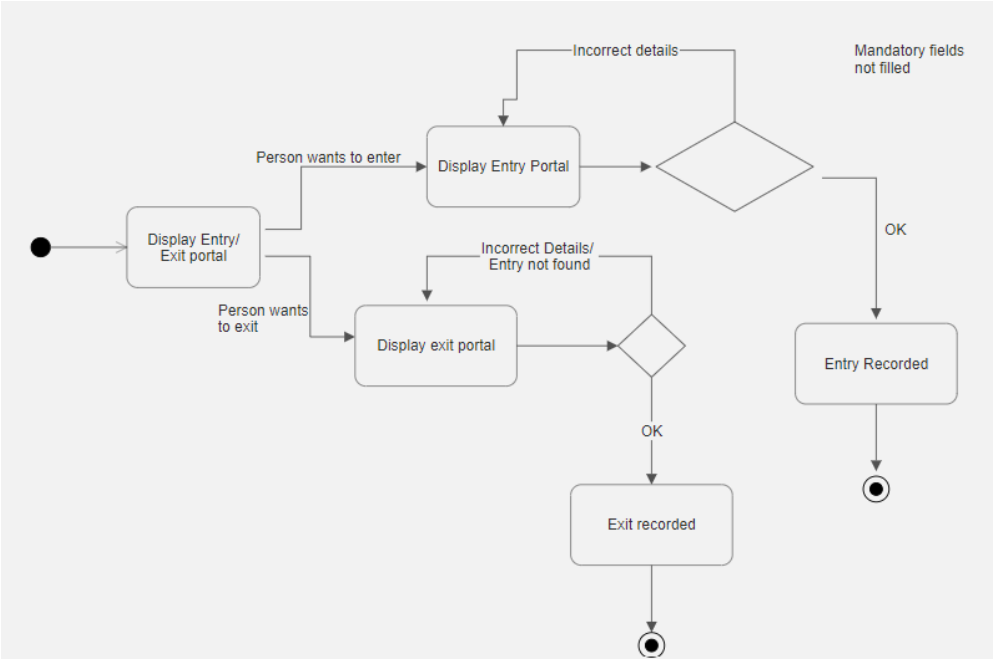
3.4.3 Non-Security Lost and found



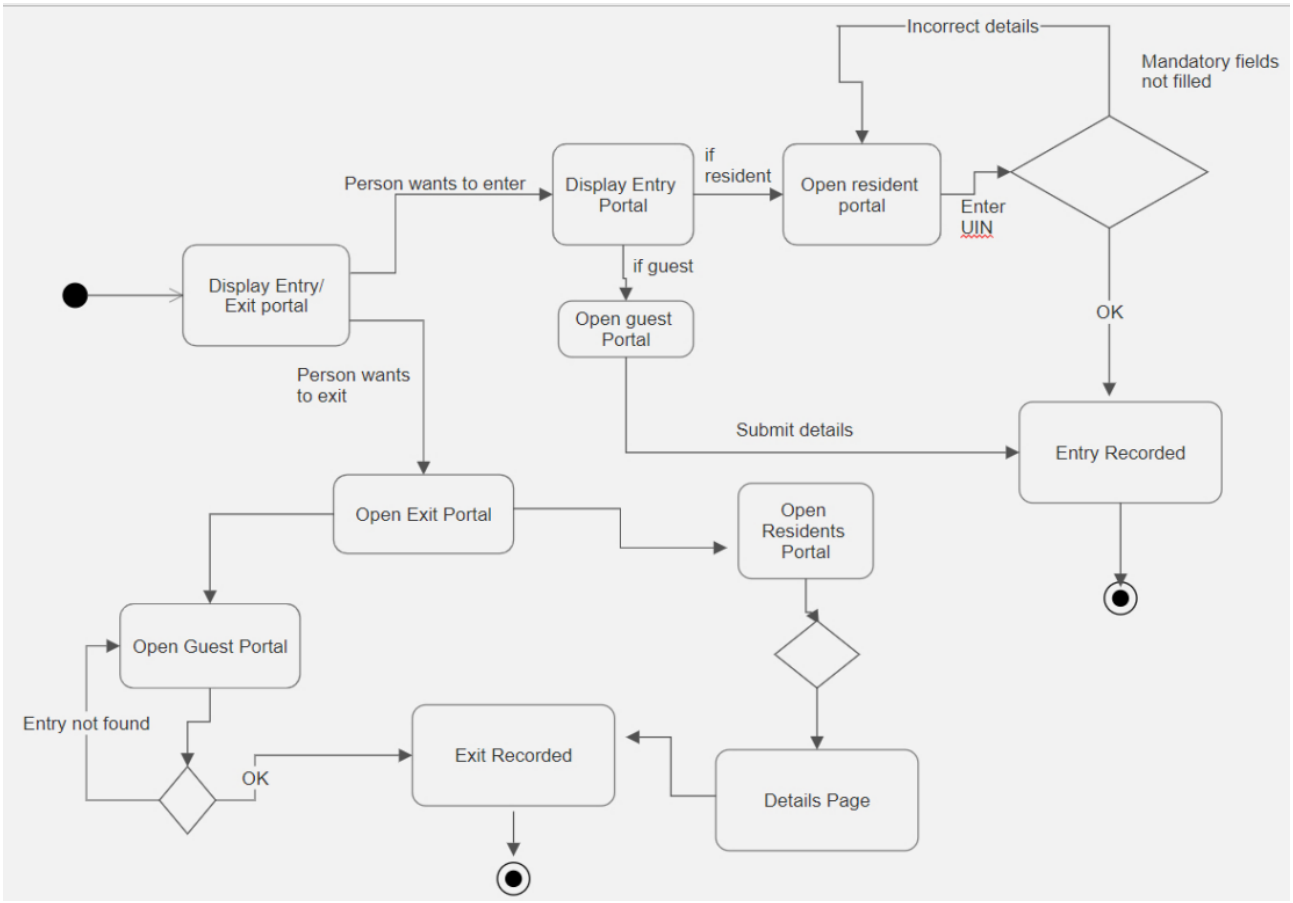
3.4.4 Security official Lost and Found



3.4.5 Hall Entry/Exit



3.4.6 Campus Entry/Exit



4 Project Plan

Steps to create and execute a test plan

1. **Analyse the product or features to be tested**
    - **a.** Proper handling of data in DB. There is always more than one user who needs to access the data simultaneously. There is a considerable amount of data that needs to be searched quickly. Security is equally important; rules need to be enforced for who can access the data.
    - **b.** Proper access of data to each individual according to their role.
    - **c.** No memory leak or overflow. Our top priority is to ensure that all the sensitive data is safe and secure; and that all the information is properly accommodated with minimum garbage.
    - **d.** Protection of user credentials and Appropriate handling of user data.
    - **e.** Is the website accessible on different devices possessed by end-users? Assuming that our website can be accessed by a range of users with different devices, we need to ensure that it runs well on all of them without any glitches. Browsers Compatibility (Cross Browser Testing), Operating System Compatibility and (Cross Platform Testing), Device Compatibility (Cross-Device Testing)
    - **f.** Response time - Can the system sustain peak load time? The site should handle many simultaneous user requests, extensive input data from users, simultaneous connection to DB, heavy load on specific pages, etc.
    - **g.** Availability - Some features may be closed for a certain period for maintenance and updates. Other than that, it should work correctly without any glitches.
    - **h.** Accessible design for users with Disabilities and all sorts of preferences.
    - **i.** How often the application crashes and minimum time to get the system back online.

2. **Test Strategies and approaches to be used**

    - **a.** **Functional testing** -
      Functional testing involves testing the application against the business requirements. It incorporates all test types designed to guarantee each part of a piece of software behaves as expected by using uses cases provided by the design team or business analyst. These Testing methods include -
        1. **Unit testing** - Unit testing is the first level of testing and is often performed by the developers themselves. It is a process to ensure that each component of the software at the code level is functional and works as expected. This usually involves writing unit tests that will call

the methods in each unit and validate them if the required parameters are passed and the expected return value.

2. **Integration testing** - After each unit is thoroughly tested, it is integrated with other units to form modules or components designed to perform specific tasks or activities. These are then tested through integration testing to ensure that entire sections of an application behave as expected. These are usually comprised of a combination of automated functional and manual tests.
3. **System testing** - The system testing method will evaluate a complete and integrated system to ensure that it meets specified requirements. Software functionality is tested end-to-end and is usually conducted by a different testing team than the development team before the product will be pushed into production.
4. **Acceptance testing** - Acceptance testing is the last phase of functional testing and is used to assess whether or not the final piece of software is ready for delivery. This requires testing the product both internally and externally, which means you will need to get it into the hands of your end-users for beta testing with your QA team. Beta testing is important to get real feedback from potential customers and address any eventual usability concerns.

b. **Non-Functional testing** -

Non-functional testing methods include all test types focused on the operational aspects of a piece of software. This includes:

1. **Performance testing-** This technique determines how an application will behave under various conditions. The goal is to test its speed, scalability, responsiveness, and stability in real user situations. Some of the types of performance are Load testing, Stress testing, Endurance testing, and spike testing.
2. **Security testing** - This is a growing concern and needs to secure data being used and stored in the software. Security testing is used to determine if the information and data are protected. The goal is to purposefully find loopholes and security risks in the system that could result in unauthorised access or information loss by probing the application for weaknesses. Various types of security testing are Integrity, Confidentiality, Authentication, Authorization, Availability, and Non-repudiation.
3. **Usability testing** - This type of testing is done from an end user's perspective to determine if the system is easily usable. Usability testing is generally the practice of testing how to easy design is to use on a group of representative users
4. **Compatibility testing** - Compatibility testing is used to determine how an application or software will work in different environments. It is used to check whether your product is compatible with multiple

operating systems, platforms, browsers or resolution configurations. The goal is to ensure that the functionality of your software is consistently supported in any environment that you expect your end-users to use.

3. Resource Planning

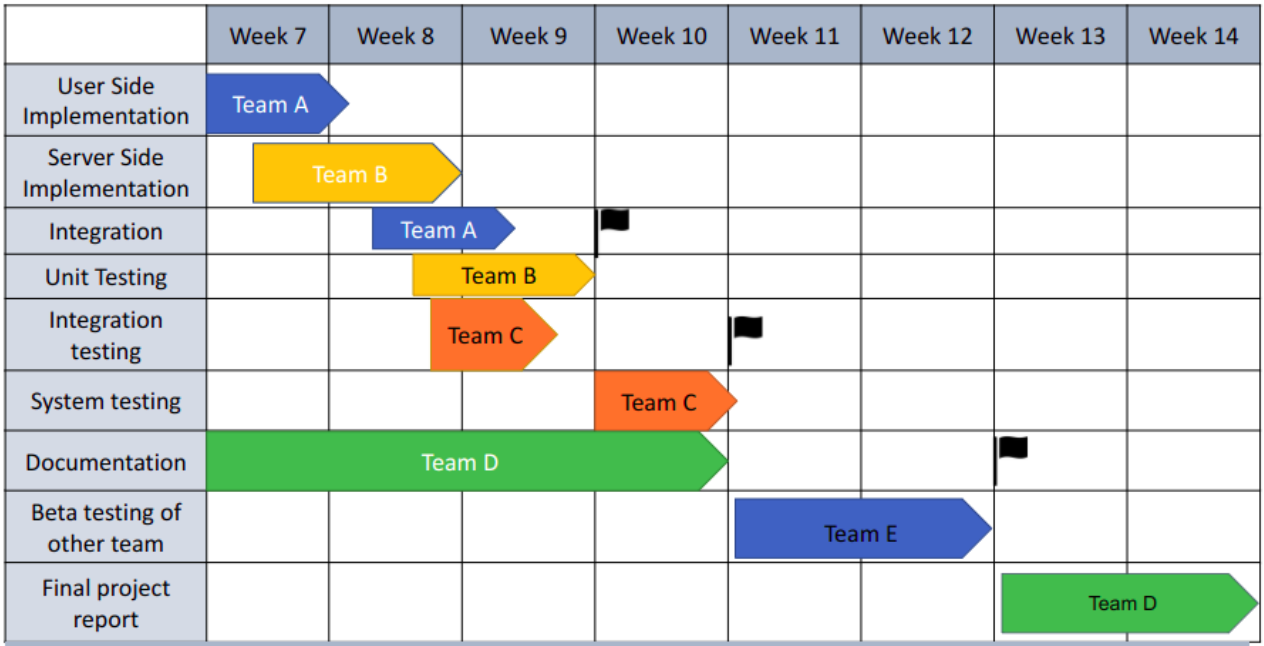
Human Resources:-

No.	Member	Tasks
1	Test Manager	Manage and Supervise the complete testing programme. Define project directions, Acquire appropriate resources while ensuring that the test plan is executed successfully in due time.
2	Tester	Identifies and describes proper test techniques, tools, and automation architecture. He analyses the test approach, execute the tests, log results, and report the defects.
3.	Developer in Test	Implement the test cases, test program, test suite, etc.
4.	Test Administrator	The role of a Test Administrator is to build up and ensure the test environment and assets are adequately managed and maintained. He supports Tester to use the test environment for test execution.
5.	SQA members	Take charge of quality assurance. Check to confirm whether the testing process is meeting specified requirements.

System Resources:-

No.	Resources	Descriptions
1	Server	Install the web application under test. This includes a separate web server, database server, and application server, if applicable.
2	Test tool	The testing tool is to automate the testing, simulate the user operation, and generate the test results. We can use tons of test tools for this project, such as Selenium, QTP, etc.
3.	Network	We need a Network including LAN and Internet to simulate the actual business and user environment.
4.	Computer	The PC which users often use to connect the webserver.

4. Develop a schedule to execute plan and track progress in the project management tool



Team A	Team B	Team C	Team D	Team E
Udit Prasad	Kuldeep Singh Chouhan	Harsh Jain	Aarchie	Aarchie
Kajal Deep	Udit Prasad	Sandeep Kumar Bijarnia	Kajal Deep	Harsh Jain
Sandeep Kumar Bijarnia	Yash Raj Mittal	Rohit	Mannem Rishwan Reddy	Kuldeep Singh Chouhan
	Sandeep Kumar Bijarnia	Arpit Kumar	Arpit Kumar	

Gantt Chart for our Project

5. Test Deliverables

Test deliverables before the testing phase.

1. **Test plans document** - a document that catalogues the test strategy, objectives, deadlines, estimations and resources required.
2. **Test Design specifications** - document specifying the details of the test approach for a feature or combinations of features.
3. **Test cases documents** - the document that catalogues a detailed description of the steps and actions one needs to perform to test one piece of the functionality and the criteria for passing.

Test deliverables during the testing.

1. **Test Data** - includes the data which is required for testing purposes. While executing a test case, data is required as input.
2. **Error logs and execution logs** - one of the most crucial artifacts providing the details of the output of test cases. It indicates the passed and failed test cases and provides info regarding the failure source.
3. **Test Scripts** - used in the automation testing environment. It is a set of instructions that is used to test the application automatically.

Test deliverables after the testing cycle is over.

1. **Test Results/reports** - Test summary featuring the testing objectives, activities, and test results.
2. **Defect Report** - A bug report summarizing all the defects identified, the actions which are causing them, and the expected results instead of the error message.
3. **Installation/ Test procedures guidelines -** the document that helps to install or configure the components that make up the system and its hardware and software requirements.

Appendix A - Group Log

Date	**Activity**
6/02/2022	Discussed upon the design document template
9/02/2022	Discussed and finalised various design aspects of our project
10/02/2022	Distributed the work between different sub-teams
13/02/2022	Created the first draft of the document
15/02/2022	Got the document reviewed by the TA
15/02/2022	Updated the document