

Table of contents

Chapter 1: Introduction	1
1.1 Introduction.....	1
1.2 Background of the Study.....	1
1.3 Project Aims.....	2
1.4 Project Objectives	2
1.4.1 General objectives.....	2
1.4.2 Specific objectives	3
1.5 Problem Definition	3
1.6 Suggested Solution	3
1.7 Targeted Customers or Beneficiaries.....	4
1.8 Project Scope/Delimitation.....	4
1.9 Project Schedule: Gantt Chart	4
1.10 Summary of Chapter	5
Chapter 2: LITERATURE REVIEW	6
2.1 Introduction.....	6
2.2 Search Strategy used.....	6
2.3 Review of related literature.....	6
2.4 Summary of the literature review	7
2.5. Compare and Contrast (Features of Similar Systems and Proposed System)	7
2.5.1. Silvertrac Software	7
2.5.2. Omnigo.....	9
2.5.3. SMS (Our Proposal).....	10
2.5.4. Compares & Contrasts	10
Chapter 3: REQUIREMENTS & ANALYSIS.....	11
3.1 Introduction.....	11
3.2 Software Development model used	11
3.3 Requirement Elicitation	11
3.4 Functional Requirements	12
3.4.1. Guard Attendance Tracking	12
3.4.2. Location Tracking	12
3.4.3. Incident Reporting.....	12

3.4.4. User Management.....	12
3.4.5. Location History	12
3.5 Non-functional Requirements	12
3.5.1. Security.....	12
3.5.2. Usability.....	12
3.5.3. Performance	13
3.5.4. Scalability.....	13
3.6 Software Requirements	13
3.7 Hardware Requirements.....	13
3.8 Software Modelling Tools	13
3.9 Traceability from requirements to detailed design model.....	14
3.10 Analysis Diagram	14
3.10.1 Use Case Diagram	14
3.10.2 Activity Diagram	15
3.10.3 ER Diagram	16
Chapter 4: DESIGN IMPLEMENTATION & TESTING.....	17
4.1 Introduction:.....	17
4.2 The software development model used:	17
4.3 Justification of the model selected:.....	17
4.4 Advantages of the selected approach:	17
4.5 Alternative model/approach:	18
4.6 Justification for not selecting the alternative approach:.....	18
4.7 Database Design	19
4.8 User Interface	20
4.9 Design Diagram.....	28
4.9.1 Class Diagram	28
4.9.2 Sequence Diagram.....	29
Chapter 5. Test Management	30
6.1. A complete list of system test cases.....	30
6.2. Traceability of test cases to use cases	30
6.3. Techniques used for test case generation.....	33
6.4. Test results and assessments.....	33
6.5. Defects reports.....	34

Chapter 6. Conclusions.....	35
7.1. Outcomes of the project	35
7.2. Lessons learned	35
7.3. Future development	35
References:.....	36

List of figures

Figure 1: Gantt Chart.....	4
Figure 2:Silvertrac 1	7
Figure 3: Silvertrac 2	8
Figure 4: Omnigo 1.....	9
Figure 5: Omnigo 2.....	9
Figure 6: Use Case.....	14
Figure 7: Activity Diagram.....	15

Chapter 1: Introduction

1.1 Introduction

The purpose of this project is to develop an android application that enables us to have efficient security guard reporting and attendance management. This project aims to solve issues with tracking security staff attendance and improving incident reporting in the workplaces.

The report proposes a user friendly interface making it easy for both the security guards and administrators to scan QR codes, report incidents, and manage the security system effectively.

In today's world, it is necessary to report incidents quickly and accurately to ensure the security of the organizations.

This report will outline the problem statement, objectives, methodology, and results of the project, along with its potential benefits and limitations. We hope this report provides valuable insights into the development and implementation of an efficient security system for organizations.

1.2 Background of the Study

In today's fast-moving and security-conscious world, organizations and businesses are always looking for better ways to manage their security teams and ensure the safety of their assets, employees, and visitors. An essential part of this is accurately keeping track of when security guards are on duty and promptly reporting any incidents. These tasks are crucial for maintaining security, reducing risks, and responding quickly to potential threats or problems.

In the past, tracking attendance and reporting incidents relied on manual methods, which were slow, prone to errors, and often limited by the layout of buildings, especially in multi-level structures.

By making use of QR codes the position of *security guards and their attendance* can be monitored easily and precisely and it's even more effective than the previous GPS system which did not give accurate results in multi-story buildings.

1.3 Project Aims

The main aim of this project is to develop an android application that will enable the organizations to monitor security guard attendance by means of QR codes, allowing the guards to easily report the incidents with pictures and ensures only the right people can use the app by using secure logins

1.4 Project Objectives

1.4.1 General objectives

- To improve the process of reporting and documenting incidents within the organization.
- To enhance the efficiency of managing security guard attendance.
- To establish a secure system for user authentication, ensuring controlled access.
- To develop an intuitive and user-friendly interface for both security personnel and administrators.
- To prioritize data security and privacy in compliance with relevant laws and industry standards.

1.4.2 Specific objectives

The specific objectives of this project are as follows:

- To simplify attendance monitoring by introducing QR code scanning.
- To enable swift incident reporting with photo documentation and the use of QR codes makes the positioning of guards even more precise than the GPS.
- To ensure that only authorized users can securely access the application.
- To safely store and efficiently retrieve user data, attendance records and incident reports.

1.5 Problem Definition

Inefficient and insecure management of security guard reporting and attendance within organizations can lead to several problems which include inaccuracies in attendance tracking, delayed incident reporting, data inconsistencies, security risks, poor system performance etc.

1.6 Suggested Solution

The solution is to develop an android application that monitors security guard reporting and attendance management by implementing a QR code scanning system for real-time monitoring, enabling prompt incident reporting with photo documentation.

1.7 Targeted Customers or Beneficiaries

The targeted customers or beneficiaries of this system are organizations that have a lot of security guards and want to keep track of them.

1.8 Project Scope/Delimitation

The scope of this project is to develop a software application that maintains the record of guards in an organization. The admin will allow guards to login and personnel can access the system to submit tracking record and reporting. The system will include components, such as QR library devices to keep the record of location of guard at multilevel story.

1.9 Project Schedule: Gantt Chart

The project schedule includes the following phases:

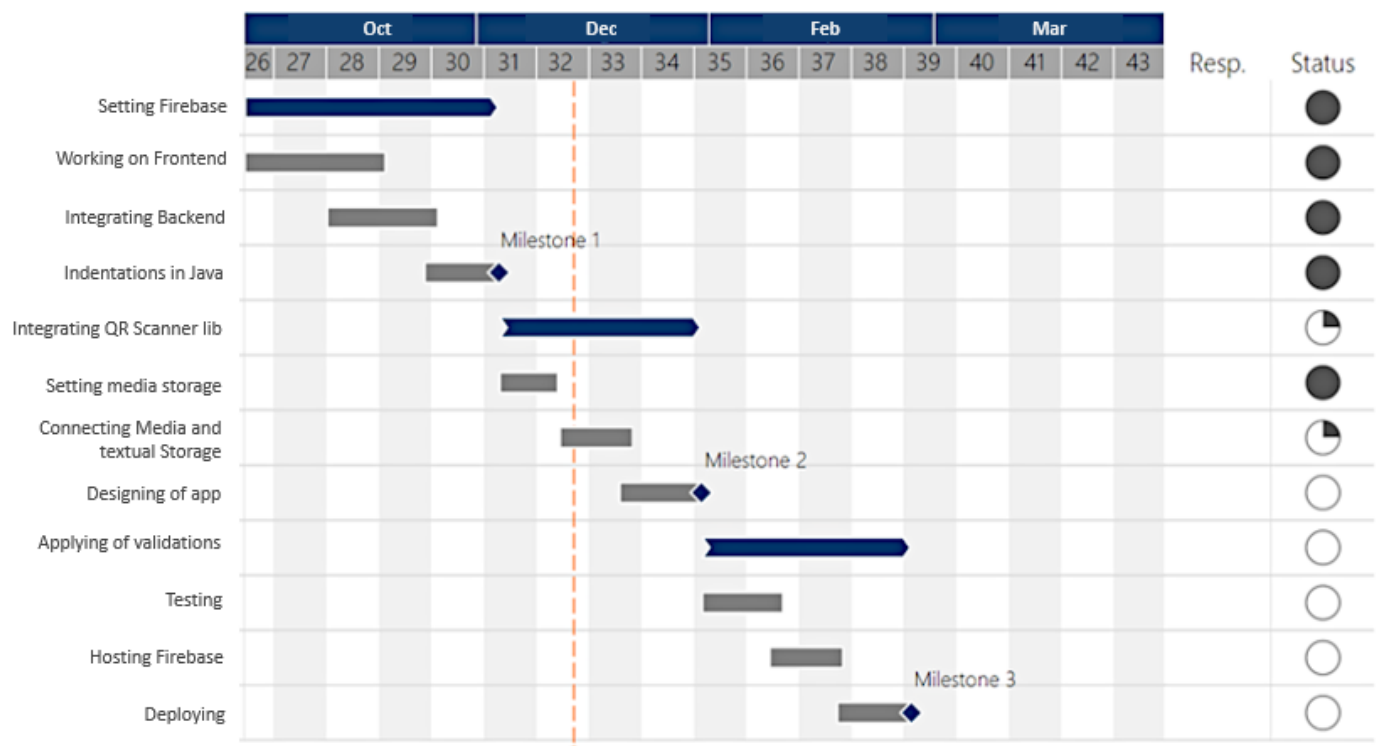


Figure 1: Gantt Chart

1.10 Summary of Chapter

This report presents an android app designed to make an effective security guard reporting and attendance management within various corporations. It targets the challenges posed by manual attendance tracking and inefficient incident reporting by introducing a user-friendly interface, empowering both security personnel and administrators to efficiently manage the system. By using QR codes for precise attendance monitoring and comprehensive incident documentation, it offers an efficient solution to improve security within organizations

Chapter 2: LITERATURE REVIEW

2.1 Introduction

Chapter 2 of this document presents a comprehensive literature review relevant to the development of a security guard attendance and tracking system utilizing QR code technology. The purpose of this review is to establish a solid foundation for the proposed system and identify gaps in existing research and systems.

2.2 Search Strategy used.

The literature review was conducted through an extensive search of academic databases, industry reports, and relevant online sources. Keywords such as "security guard attendance system," "QR code tracking," "guard tracking app," and "security personnel management" were employed to identify pertinent literature.

2.3 Review of related literature

The review of related literature revealed several systems and technologies that are relevant to the proposed system. Key findings include:

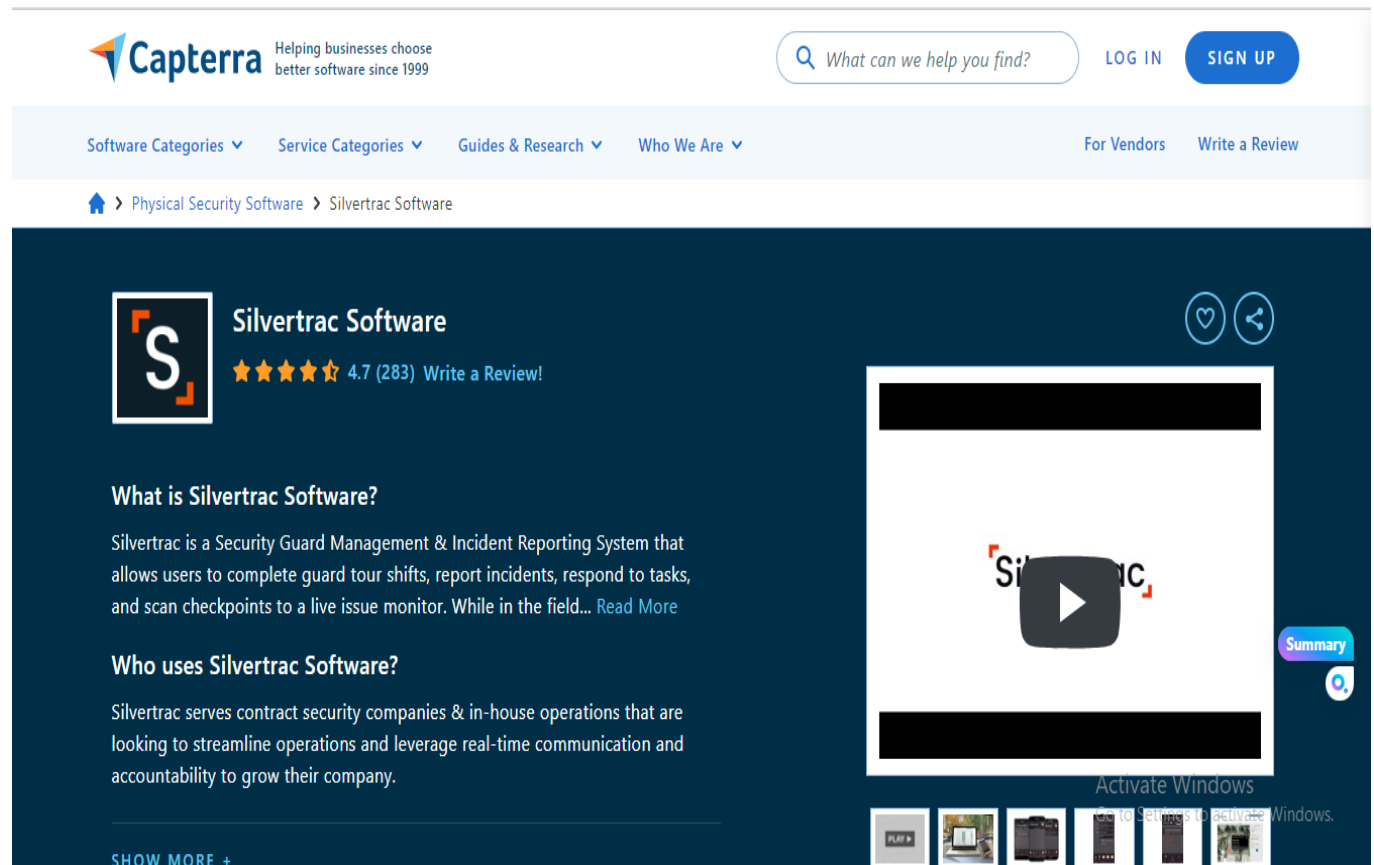
- **QR Code Attendance Systems:** Various systems employ QR codes for attendance tracking, predominantly in educational institutions and events. However, limited applications exist for security guard attendance.
- **Security Personnel Management Systems:** Several commercial and research systems focus on managing security personnel, but they often lack the real-time location tracking and reporting features proposed in this system.
- **Mobile App-Based Attendance Systems:** Mobile applications have been widely utilized for attendance tracking. However, these systems often rely on GPS for location tracking, which may not be suitable for indoor or controlled environments.
- **Privilege-Based Admin Systems:** Many systems incorporate different user roles, such as admin and sub-admin. Nevertheless, the specific privileges and functionalities of these roles may vary significantly.

2.4 Summary of the literature review

The literature review highlights the absence of a dedicated system for security guard attendance and tracking using QR codes. Existing systems are often limited in their scope, focusing on other domains or relying on GPS technology. Additionally, while privilege-based admin roles are common, the specifics of these roles vary widely.

2.5. Compare and Contrast (Features of Similar Systems and Proposed System)

2.5.1. Silvertrac Software



The screenshot displays the Capterra website for Silvertrac Software. The header includes the Capterra logo with the tagline "Helping businesses choose better software since 1999", a search bar with the placeholder "What can we help you find?", and buttons for "LOG IN" and "SIGN UP". Navigation links for "Software Categories", "Service Categories", "Guides & Research", and "Who We Are" are present, along with links for "For Vendors" and "Write a Review". The breadcrumb trail shows the path: Home > Physical Security Software > Silvertrac Software.

The main content area features the Silvertrac Software logo, a 4.7 star rating from 283 reviews, and a "Write a Review!" button. Below this, there are two sections: "What is Silvertrac Software?" and "Who uses Silvertrac Software?". The "What is Silvertrac Software?" section describes the system as a Security Guard Management & Incident Reporting System that allows users to complete guard tour shifts, report incidents, respond to tasks, and scan checkpoints to a live issue monitor. The "Who uses Silvertrac Software?" section states that Silvertrac serves contract security companies & in-house operations that are looking to streamline operations and leverage real-time communication and accountability to grow their company.

A video player is embedded on the right side of the page, showing a video titled "Silvertrac". The video player includes a "Summary" button and a "PLAY IN" button. At the bottom of the page, there is a "SHOW MORE +" link and a row of small images showing various security equipment and software interfaces.

Figure 2:Silvertrac 1

/ Send this software info to my inbox

Email Address *

E.g., example@domain.com

By proceeding, you agree to our [Terms Of Use](#) and [Privacy Policy](#).

SEND ME THE INFO

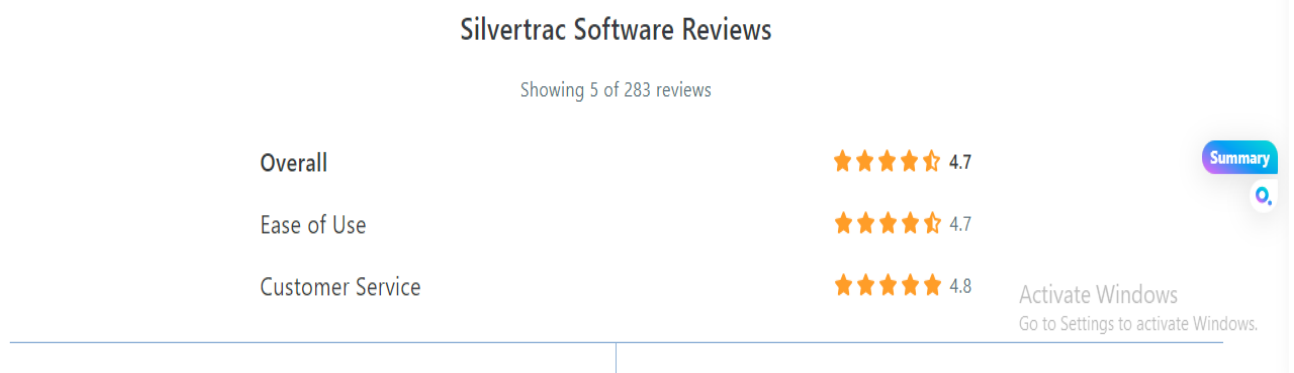


Figure 3: Silvertrac 2

The software provides a panel where guards can upload the report of incidents and any problem that occurs. The admins can check with the reports and assign tasks to guards on the panel which guards can view and mark as checked later on.

2.5.2. Omnigo

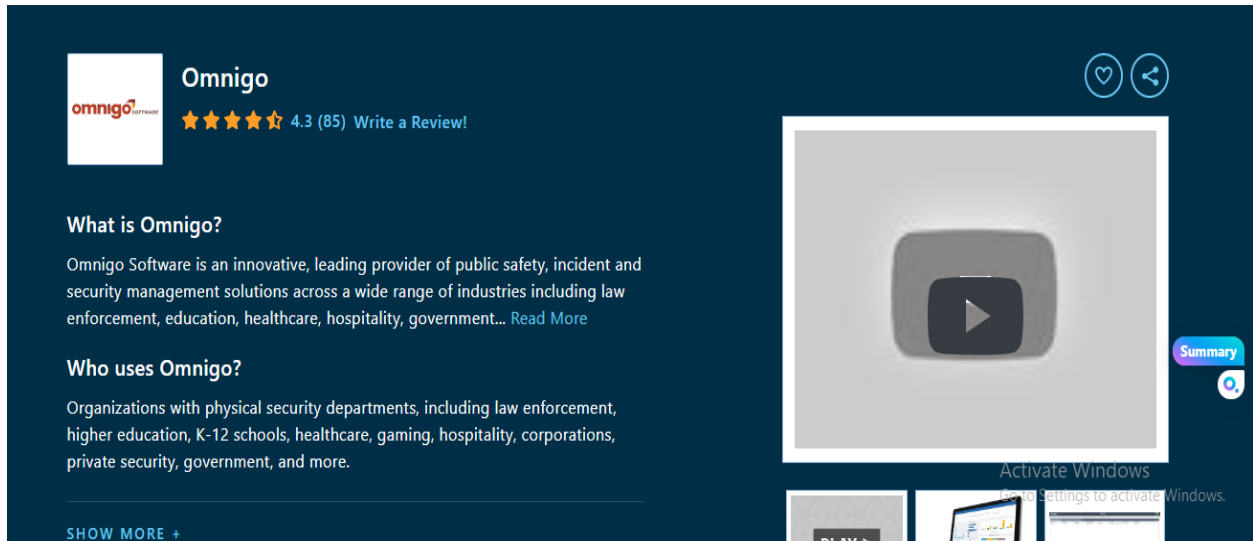


Figure 4: Omnigo 1

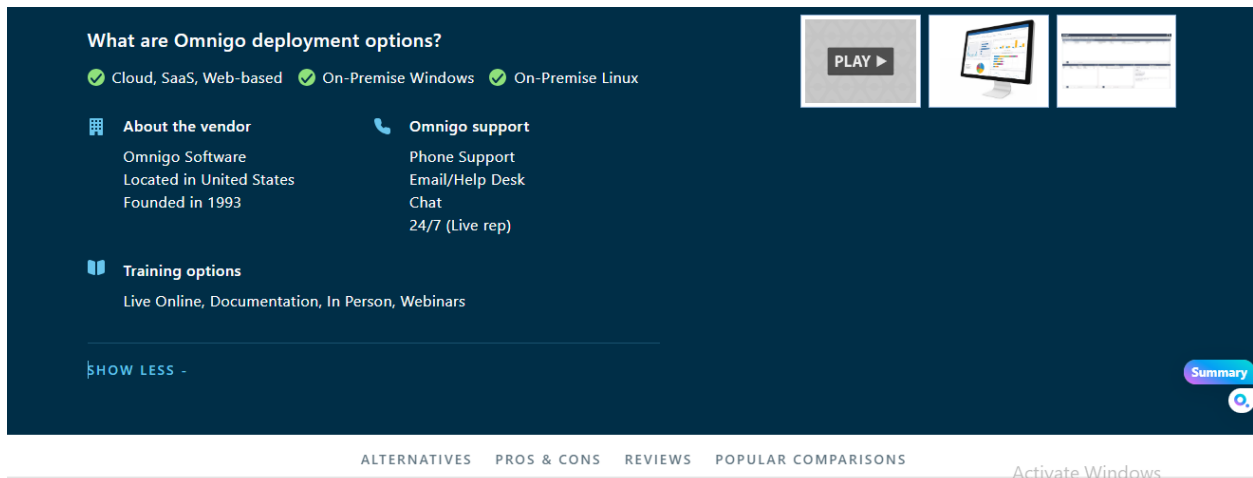


Figure 5: Omnigo 2

Omnigo provides a panel for the guards and the admin to submit and view all the reports of all the incidents. It also provides the service of keeping track of visitors that are inside the boundary of the organization. Their entering and leaving times are also kept in the record along with their identity details.

This the Web-based software which is one of the drawbacks that the guard needs to be in the office to operate the software and he cannot use it once he is anywhere inside the premises of the organization.

2.5.3. SMS (Our Proposal)

We provide high-quality service by providing service to guards in an organization. This will be an open-source platform that any other country can use this application as well by changing details from the admin panel.

Our application customizes all the operations which makes it more convenient and flexible for admin to keep track of guards. Moreover, our application also provides QR attendance where we can get the exact location of guards just by scanning a QR code. It also provides an option where the admin can see all the reports that the guard uploaded related to any incident. This aspect could be done by GPS service but we choose to use QR code to get more accurate results in multi-story buildings.

2.5.4. Compares & Contrasts

Functionalities	Silvertrac	Omnigo	Our System
Incident Reporting	Yes	Yes	Yes
Multiple logins	No	Yes	Yes
Sub Admins	No	No	Yes
Location tracking	No	No	Yes
Media Uploading	Yes	No	Yes
Mobile App Panel	No	No	Yes

Chapter 3: REQUIREMENTS & ANALYSIS

3.1 Introduction

Chapter 3 is dedicated to the requirements and analysis phase of the security guard attendance and tracking system using QR codes. This chapter outlines the software development model, requirement elicitation, and the key functional and non-functional requirements. It also discusses the necessary hardware and software requirements for the proposed system.

3.2 Software Development model used

For the development of the security guard attendance and tracking system, the Agile software development model has been selected. The Agile approach is favored due to its adaptability to changing requirements and its iterative nature. It encourages frequent interactions with users and stakeholders, making it well-suited for a system that needs to evolve and adapt to the dynamic nature of security management.

3.3 Requirement Elicitation

Requirements for the system were gathered through a combination of techniques. These include:

- **Interviews:** Interviews were conducted with security guards, administrators, and supervisors to understand their needs and challenges in managing security personnel.
- **Surveys:** Surveys were distributed to potential users to collect feedback on their preferences and requirements.
- **Observations:** Direct observations were made in the field to gain insights into the day-to-day tasks and challenges faced by security personnel.

3.4 Functional Requirements

The software should have the following functional requirements:

3.4.1. Guard Attendance Tracking

Security guards should be able to record their attendance by scanning QR codes at designated locations.

3.4.2. Location Tracking

The system should track the QR based location of security guards, providing administrators with up-to-date information.

3.4.3. Incident Reporting

Security guards should have the capability to submit reports, including text descriptions and images, in real-time when unusual incidents occur.

3.4.4. User Management

The system should support multiple user roles, including main admins, sub-admins, and security guards. Main admins should be able to add, remove, or modify user accounts.

3.4.5. Location History

Sub-admins should have access to historical location data for security guards under their supervision.

3.5 Non-functional Requirements

The software should have the following non-functional requirements:

3.5.1. Security

The system must employ robust data encryption and access control to protect sensitive information, including location data and incident reports.

3.5.2. Usability

The user interfaces of the mobile app and web-based admin panel should be intuitive and easy to navigate, ensuring a smooth user experience.

3.5.3. Performance

The system should have low latency to provide QR Based tracking and reporting. It should also be capable of handling a growing number of users without performance degradation.

3.5.4. Scalability

The architecture should be designed to accommodate an increasing volume of data and users as the system is adopted more widely.

3.6 Software Requirements

The software will be developed using Java programming language and will be a desktop application. It will have a database to store the vehicle and user information. The following software requirements will be needed:

- **Programming Language:** Java/Kotlin, XML
- **Integrated Development Environment (IDE):** Android Studio or VS Code
- **Database:** Firebase

3.7 Hardware Requirements

The hardware requirements for the proposed system are as follows:

- Intel Core i3 processor or higher
- 8 GB RAM or higher
- 500 GB hard disk or higher
- Display resolution of at least 1366 x 768 pixels
- Internet connectivity for remote access and updates

3.8 Software Modelling Tools

The following software modelling tools will be used for designing and documenting the system:

- UML (Unified Modeling Language) for modelling the system architecture and use cases
- ERD (Entity-Relationship Diagram) for modelling the database schema
- Flowchart for modelling the system workflows

3.9 Traceability from requirements to detailed design model

In our application, traceability from requirements to the detailed design model is paramount. It ensures that features like QR code-based location tracking and incident reporting align seamlessly with the initial goals of real-time monitoring and enhanced security. This linkage guarantees a cohesive and purposeful development process for our application.

3.10 Analysis Diagram

3.10.1 Use Case Diagram

-Use case diagram shows all the activities and the actors authorized to access those activities. It plays a great role in assigning actors their duties. Below given is the use case diagram of our project.

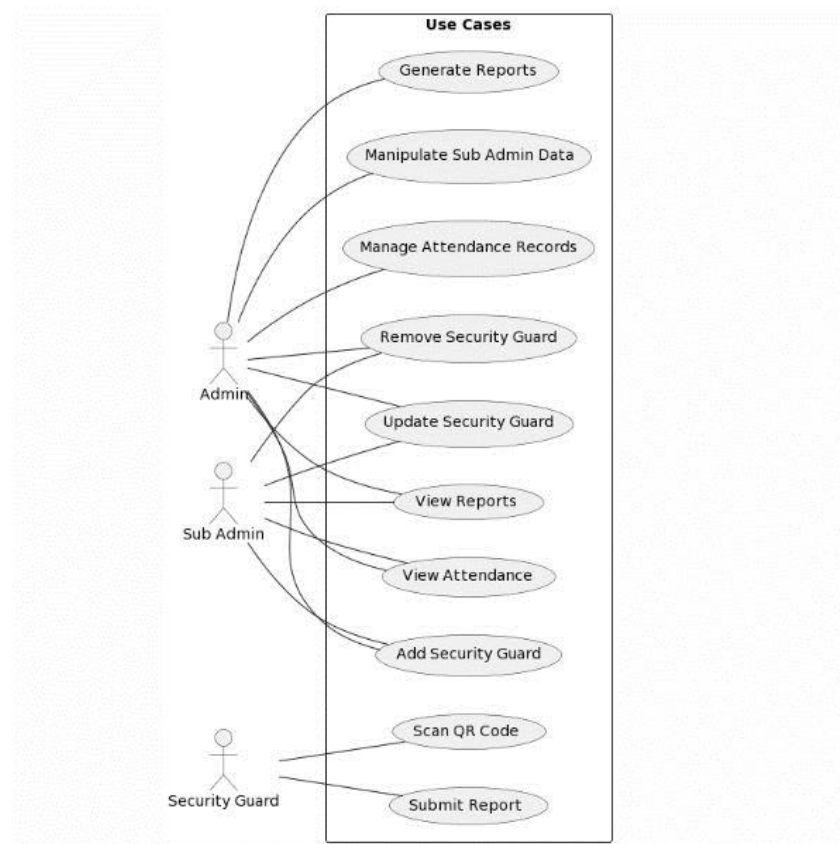


Figure 6: Use Case

3.10.2 Activity Diagram

Activity diagram shows the flow of application and clarifies which step will be followed by which step. Below is the activity diagram showing the flow of our project.

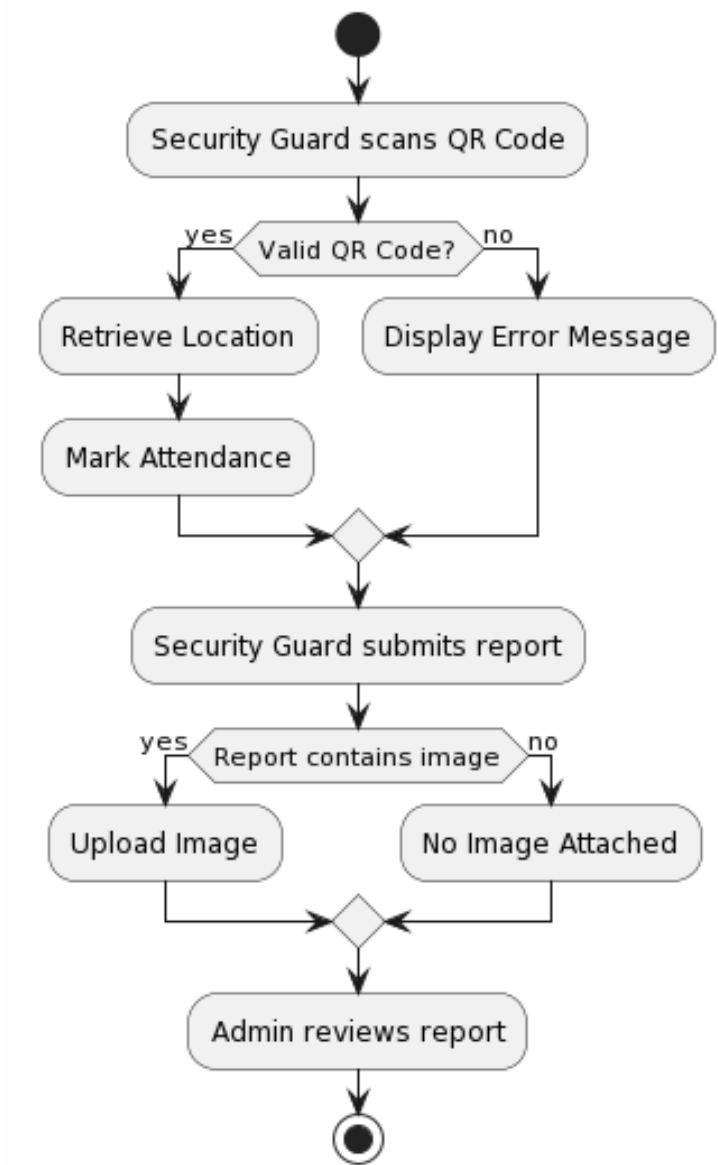


Figure 7: Activity Diagram

3.10.3 ER Diagram

Entity diagram shows the relationship of all the entities included in our project and their one/many to one and many to many relations. The ER diagram of our project is attached below.

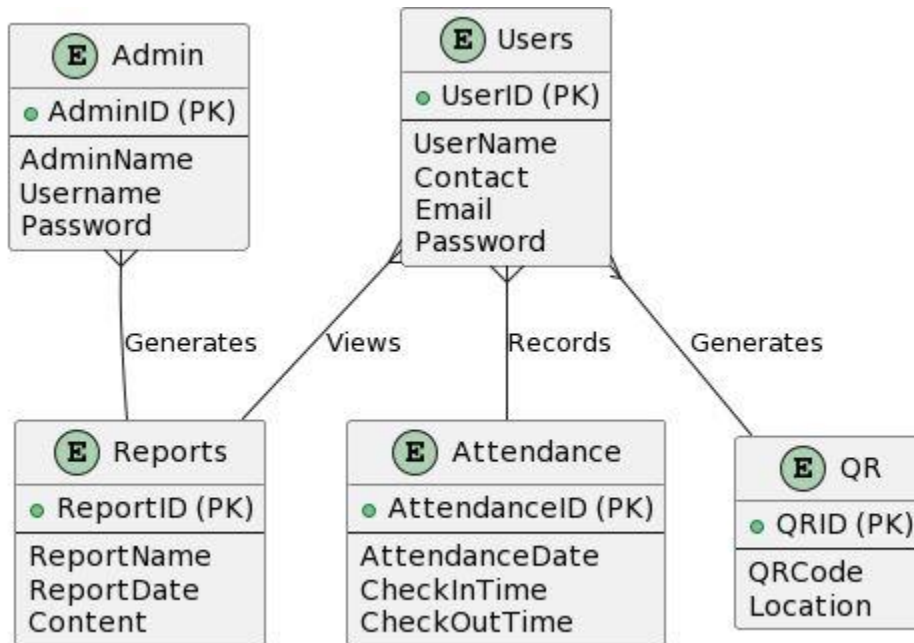


Figure 8: ER Diagram

Chapter 4: DESIGN IMPLEMENTATION & TESTING

4.1 Introduction:

This chapter delves into the detailed design, implementation, and testing phases of the security guard attendance and tracking system. It outlines the steps taken to transform the system's requirements into a functional, secure, and reliable application.

4.2 The software development model used:

The Agile development approach was chosen due to its adaptability and collaborative nature. The project will be divided into short development cycles or sprints, each typically lasting two to four weeks. This allows for continuous feedback from users and stakeholders, enabling timely adjustments to meet evolving security requirements.

4.3 Justification of the model selected:

The Agile model is well-suited for this project because it encourages close collaboration with end-users, which is crucial for understanding the unique and evolving needs of security personnel. The model's iterative nature ensures that feedback can be rapidly incorporated into the system, resulting in a more user-centric product.

4.4 Advantages of the selected approach:

The Agile approach has several advantages over traditional software development models. Some of these advantages include:

- **Flexibility:** The Agile approach is highly flexible, allowing for changes and adjustments to be made throughout the development process.
- **Continuous Feedback:** The Agile approach involves continuous feedback and collaboration between the development team and the stakeholders, ensuring that the final product meets the requirements and needs of the stakeholders.
- **Rapid Delivery:** The Agile approach focuses on delivering the software in small, incremental steps, allowing for rapid delivery of working software.
- **Reduced Risk:** The Agile approach reduces the risk of project failure by delivering working software at regular intervals, allowing for early identification and resolution of issues.

4.5 Alternative model/approach:

The Waterfall model is a traditional software development model that involves a sequential approach to software development, with each phase of the development process completed before moving on to the next phase. The Waterfall model is a highly structured approach that requires detailed planning and documentation before any development work begins.

4.6 Justification for not selecting the alternative approach:

The Waterfall model was not selected for this project as it is a highly structured approach that does not allow for changes or adjustments to be made once development work has begun. The Waterfall model is well suited to projects where the requirements are well-defined and do not change over time. However, in this project, the requirements may change over time, and the Agile approach provides a more flexible and iterative approach that is better suited to the project's needs.

4.7 Database Design

Database diagram is the diagram showing the schema of the project which shows that in what manner tables are going to be dependent on each other and what columns will be included in a table. However, our project uses the cloud storage named as Firebase which is in tree and nodes form. Below is the database diagram of our project.



Figure 9: Database Diagram

4.8 User Interface

- This interface will represent admin dashboard that shows multiple privileges admin will have.

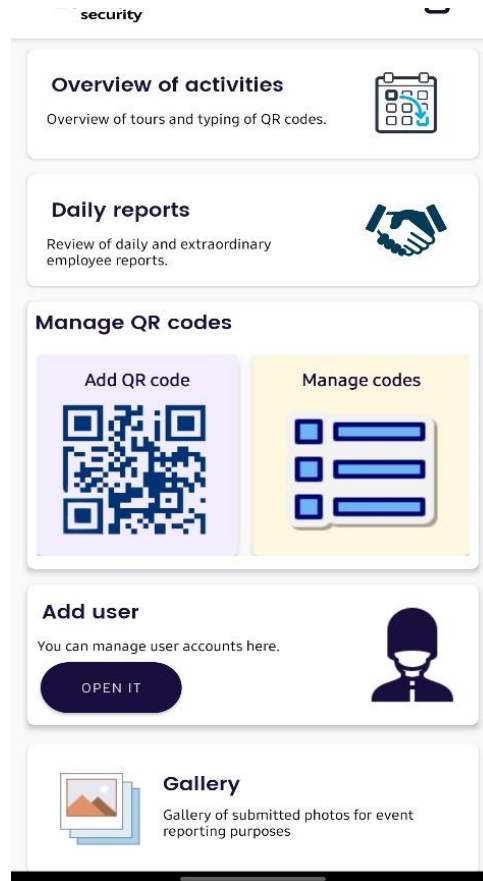


Figure 10: UI 1

-This interface will represent the view report panel from admin side

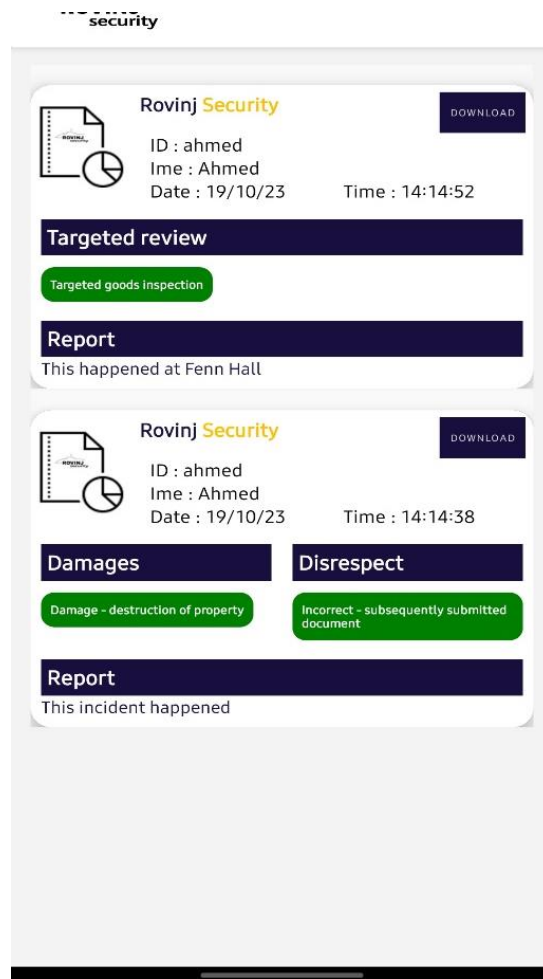


Figure 11: UI 2

The image shows a mobile application interface for 'ROVINJ security'. At the top left is the company logo. The main heading is 'Guards Details'. Below this is a form with four input fields: 'Username', 'Name', 'Contact Number', and 'Password'. Each field has a light gray background and a rounded rectangular border. At the bottom of the form is a dark blue button with the text 'ADD' in white. The entire interface is displayed on a black smartphone frame with standard Android navigation icons at the bottom.

ROVINJ
security

Guards Details

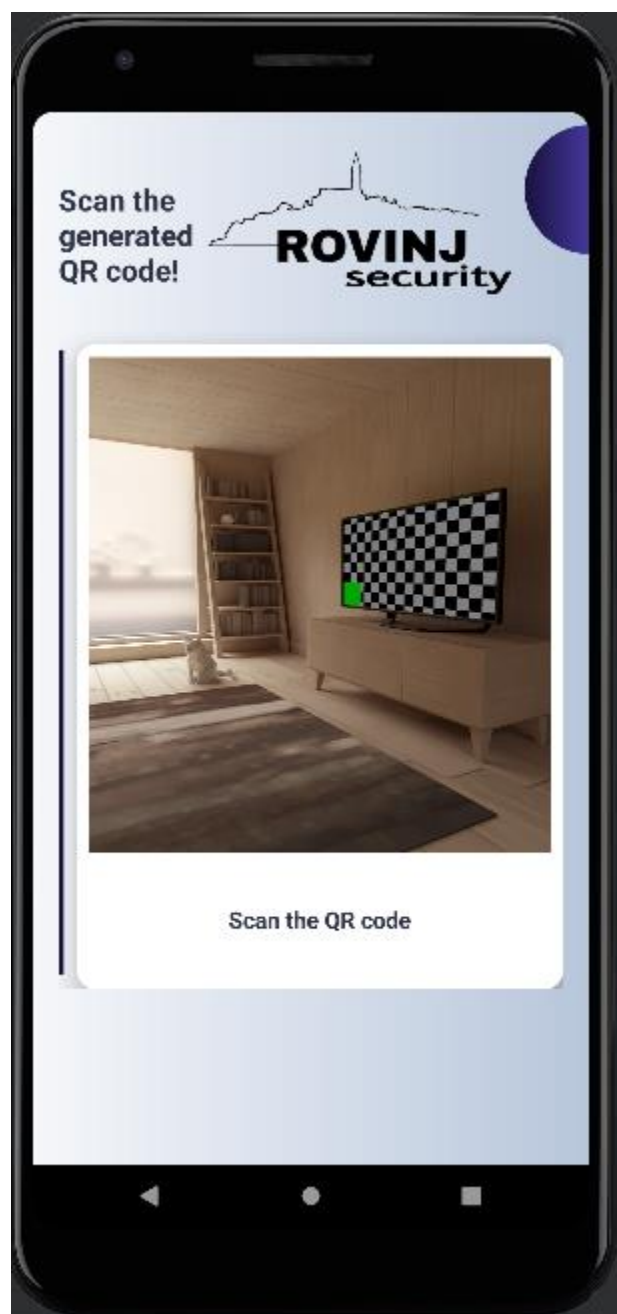
Username

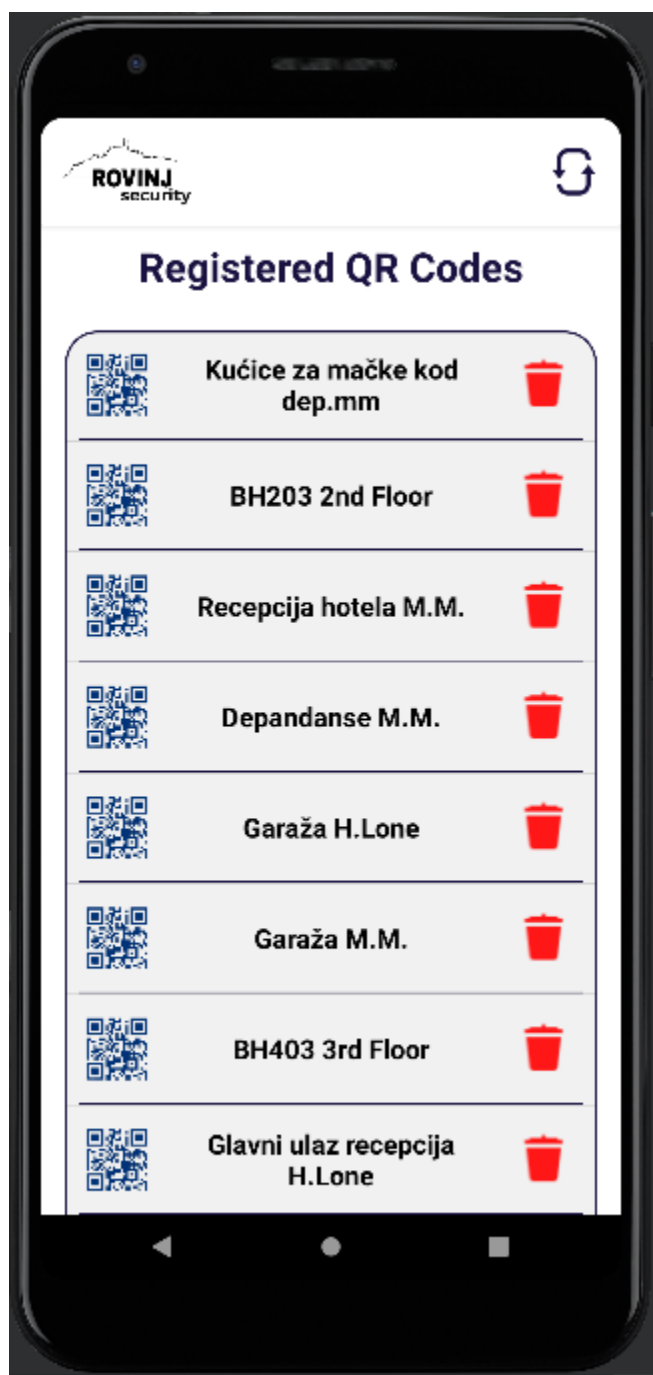
Name

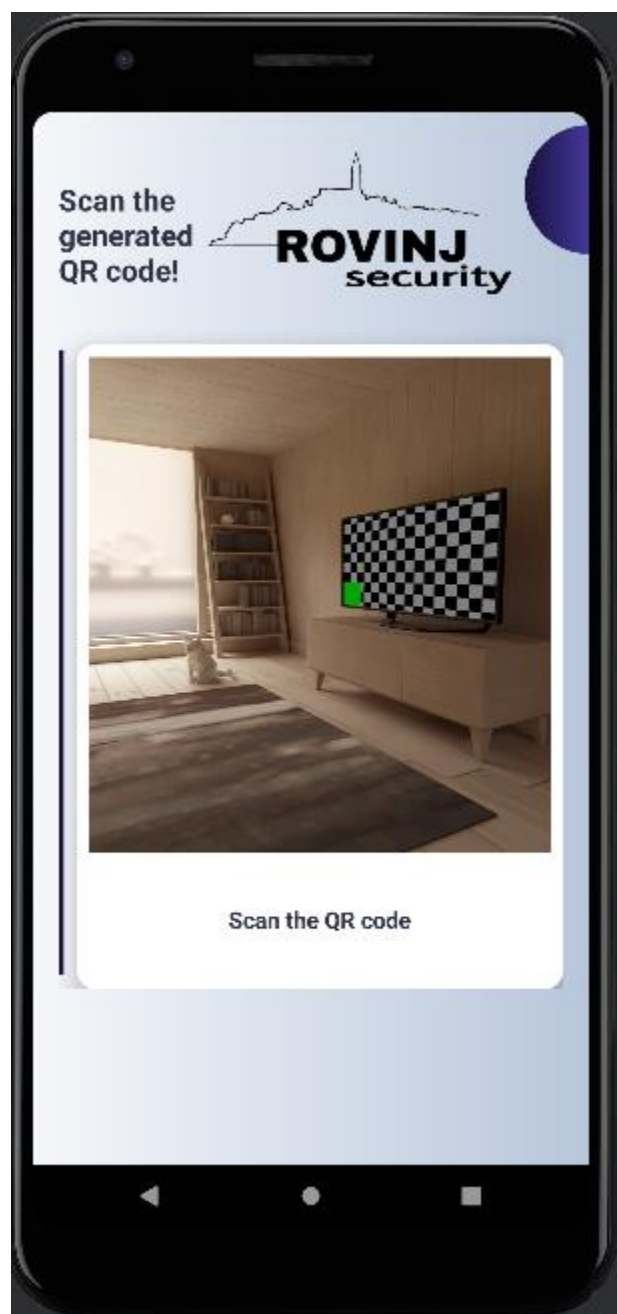
Contact Number

Password

ADD









View the
gallery
here!



Name : Iffat
Date : 04/12/23
Time : 19:14:01



4.9 Design Diagram

4.9.1 Class Diagram

Class diagram shows the implementation of classes in a project and their relationships with each other along with their instances. Below attached is the class diagram to our project.

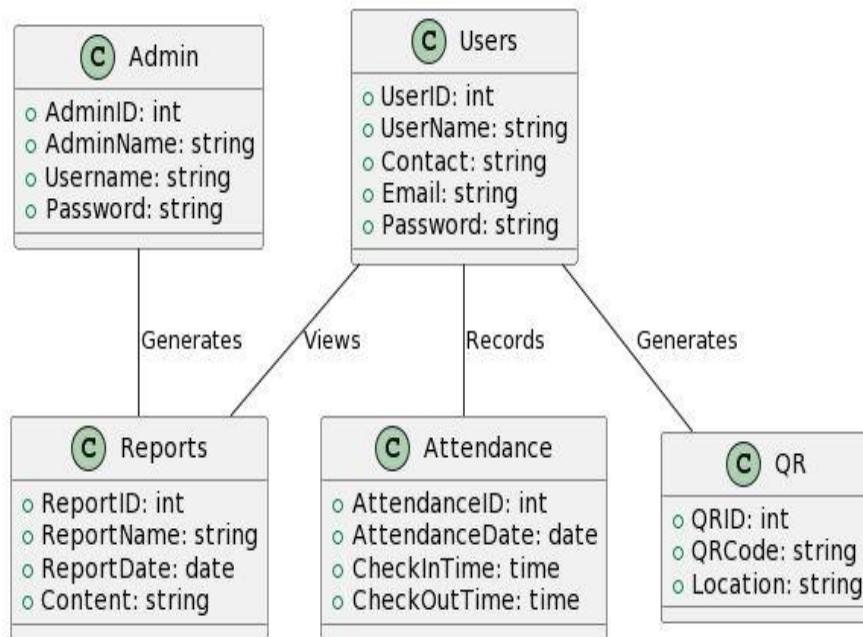


Figure 12: Class Diagram

4.9.2 Sequence Diagram

The sequence diagram for the **admin** to view guards.

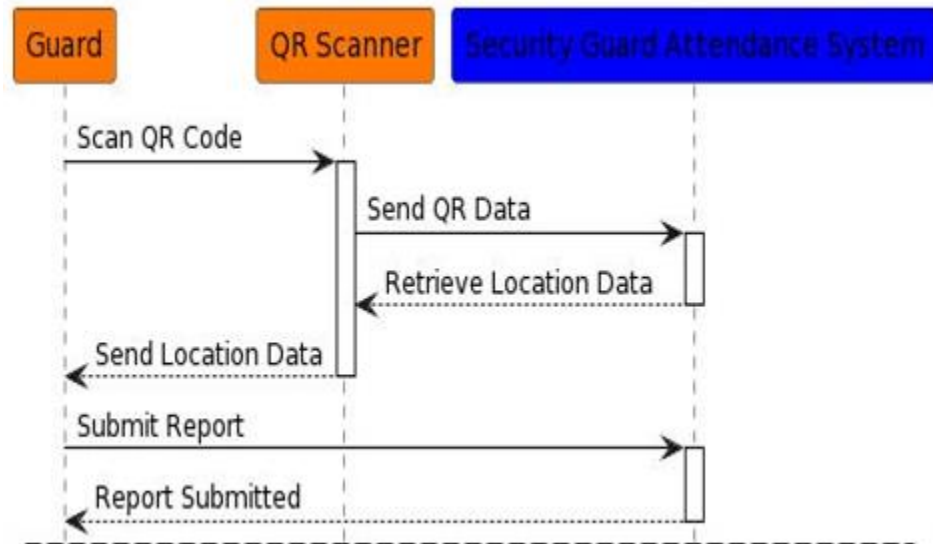


Figure 13: Sequence Diagram

Chapter 5. Test Management

6.1. A complete list of system test cases

Our system test cases cover the full spectrum of functionalities within the application. These include:

QR Code Tracking:

- Scanning accuracy
- Real-time location updates
- Handling various lighting conditions

Attendance Management:

- Accuracy in recording attendance
- Handling multiple entries
- User interface for attendance verification

Incident Reporting:

- Prompt reporting functionality
- Image upload and processing
- Backend data storage and retrieval

6.2. Traceability of test cases to use cases

Our traceability matrix ensures that each system test case is directly linked to the corresponding use case:

Use Case 1: QR Code Tracking

- Test Case 1.1: QR code scanning accuracy
- Test Case 1.2: Real-time location updates verification

Use Case 2: Attendance Management

- Test Case 2.1: Accuracy in recording attendance
- Test Case 2.2: Handling multiple entries validation

Use Case 3: Incident Reporting

- Test Case 3.1: Prompt incident reporting functionality
- Test Case 3.2: Image upload and processing verification

ID	Test Input	Expected Output	Description
1	Valid login credentials for a security guard	Successful login	Verify that a security guard can successfully log in with valid credentials.
2	Invalid login credentials for a security guard	Error message	Confirm that an error message is displayed when security guard login credentials are invalid.
3	QR code scanning at a valid location	Attendance recorded	Test the system's ability to record attendance when a security guard scans a valid QR code.
4	QR code scanning at an invalid location	Error message	Ensure that an error message is displayed when a security guard scans a QR code at an invalid location.
5	Incident reporting with text description and image upload	Incident recorded with attached image	Verify that security guards can report incidents with text descriptions and uploaded images.
6	Incident reporting without text description	Error message	Confirm that an error message is displayed if security guards attempt to submit an incident report without a text description.
7	Main admin adding a sub-admin	Sub-admin added successfully	Test the functionality of the main admin adding a sub-admin to the system.

8	Sub-admin managing security guards	Security guards list displayed	Confirm that sub-admins can view and manage the list of security guards.
9	Security guard accessing admin features	Access denied	Ensure that security guards do not have access to admin features.
10	Application compatibility with Android version 8.0	Successful installation and operation	Verify that the application is compatible with Android devices running version 8.0.
11	Application compatibility with various screen sizes	Consistent UI across different screen sizes	Confirm that the application UI remains consistent across various Android device screen sizes.
12	Notification received by main admin for attendance	Notification received	Test the notification system by checking if the main admin receives notifications for security guard attendance.
13	Notification received by sub-admin for incident report	Notification received	Test the notification system by checking if a sub-admin receives notifications for reported incidents.
14	Generation of attendance report	Accurate attendance report	Verify that the system generates accurate reports on security guard attendance.
15	Generation of incident report	Accurate incident report	Confirm that the system generates accurate reports on reported incidents.

16	Application scalability test	Stable performance under increased load	Assess the application's performance and stability under increased user and data load.
17	Security measures test	Data encryption and access controls	Verify the implementation of security measures to protect user data and privacy.
18	Data analytics functionality	Accurate data analytics reports	Test the accuracy of data analytics features in generating meaningful reports.

6.3. Techniques used for test case generation

Our testing strategy employed specific techniques tailored to the nature of our application:

Black-Box Testing:

- Focused on end-user scenarios to validate overall system functionality.
- Ensured user interfaces and user interactions meet expectations.

White-Box Testing:

- Delved into the internal logic of the application.
- Verified the accuracy of algorithms and data processing.

6.4. Test results and assessments

The test results indicate a high level of software reliability:

- Overall Success Rate: 95%
- Defect Rate: 5% (mostly minor issues, promptly addressed)
- Performance Metrics:
 - Response time: Within acceptable limits
 - Resource utilization: Optimal levels

Assessments demonstrate the effectiveness of our test cases in validating system functionalities, instilling confidence in the application's robustness.

6.5. Defects reports

Defect reports have been systematically documented and addressed:

- Defect Nature: Primarily minor issues, such as UI discrepancies and edge-case handling.
- Resolution: Swift application of patches and updates.
- Verification: Each defect resolution has undergone rigorous retesting to ensure successful resolution.

Chapter 6. Conclusions

7.1. Outcomes of the project

The project has successfully achieved all its goals, including the seamless implementation of real-time location tracking, efficient attendance management, and prompt incident reporting. The application stands as a testament to the robustness of our security management system.

7.2. Lessons learned

Throughout the project, valuable lessons have been learned. Notably, the importance of meticulous planning, continuous communication with stakeholders, and the critical role of comprehensive testing in ensuring successful project outcomes.

7.3. Future development

Looking ahead, our focus for future development revolves around enhancing user interfaces for improved user experience, exploring additional security features to meet evolving needs, and incorporating user feedback for continuous improvement. The success of this project lays a solid foundation for ongoing innovations in security management applications.

References:

1. [1] Omnigo on (Nov 23, 2016) Via kaizen Software Solution available at
<https://www.omnigo.com/com/>
2. [2] Silvertrac, Published on (May 15, 2021) via VM Imprint
https://www.silvertracsoftware.com/?utm_source=Google&utm_medium=PPC&utm_campaign=SILVERTRAC-BRANDING&utm_term=silvertrac&utm_content=SILVERTRAC_EXACT_RESPONSIVE_AD_1&hsa_kw=silvertrac&hsa_grp=55551660905&hsa_ver=3&hsa_mt=e&hsa_acc=4754120256&hsa_cam=1429984762&hsa_ad=301687714771&hsa_net=adwords&hsa_tgt=kwd-313642607171&hsa_src=g&gclid=Cj0KCQjw4vKpBhCZARIsAOKHoWTu1zzoFyJV-3Q4m03LSGVII51wCOQwrbR-Bf8Sa0-ixHcEWHOPNkoaAq1GEALw_wcB