Radar security

Submitted By

Umair Sajid	18F-0279
Misha Hafeez	19F-0381
Ummey Haani Shad	19F-0399

Supervise By

Dr. Muhmmad Umar Aftab

Co supervise By

Ma’am Sahar Ajmal

Acknowledgement

This documentation consumed a lot of work, research, and dedication. It was not possible without help of our teachers and supervisors. We would like to extend our sincere gratitude to them.

We are thankful to FAST National University of Computer and Emerging Sciences for their support and for providing necessary guidance concerning project.

We are also grateful to both our supervisors Dr. Muhmmad Umar Aftab for provision of expertise, and technical support. Without their superior knowledge and experience, the documentation would never have brought the quality outcomes, and thus their support has been essential.

We shall soon start the implementation of the project after the acceptance of documents.

Intellectual Property Right Declaration

Radar security is the sole property of the National University of Computer and Emerging Science and is protected under the intellectual property right laws. It can only be considered/used for purposes like extension for further enhancement, product development, adoption for commercial/organizational usage, etc., with the permission of the University

This above statement applies to all students and faculty.

Date: _______________

Student 1

Name: Misha Hafeez

Signature: _________________

Student 2

Name: Umair Sajid

Signature: _________________

Student 3

Name: Ummey Haani Shad

Signature: _________________

Supervisor (Faculty)

Name: Dr. Muhmmad Umar Aftab

Signature: _________________

Co-Supervisor (Faculty)

Name: Ma’am Sahar Ajmal

Signature: _________________

Anti-Plagiarism Declaration

This is to declare that the above publication produced under the:

Title: Radar security

is the sole contribution of the author(s) and no part hereof has been reproduced on as it is basis (cut and paste) which can be considered as **Plagiarism.** All referenced parts have been used to argue the idea and have been cited properly. We will be responsible and liable for any consequence if violation of this declaration is determined.

Date: _______________

Student 1

Name: Misha Hafeez

Signature: _________________

Student 2

Name: Umair Sajid

Signature: _________________

Student 3

Name: Ummey Haani Shad

Signature: _________________

Table of Contents

Table of Figures

Tables of Tables

1. Introduction

This document collects, analyzes, and defines the high-level needs and features of the Radar security. It focuses on the capabilities and facilities needed by the stakeholder. Radar security is a software solution which will detect the inside attacks. Radar security will take datasets then it will apply ML-Based algorithms to detect the intruder. Moreover, Radar security is the system having all capabilities and could attempt to detect possible incidents. Radar security operator need not to configure the system, it automatically learns the behavior of many subjects and can be left to run unattended. Since it contains no knowledge, some would say prejudice, about how an intrusion would manifest itself. It has the possibility of catching novel intrusions, as well as variation of know intrusion.

2. Vision Document

TABLE 1 VISION DOCUMENT

Version	Date	Description	Authors
V1.0	30/09/2022	Radar security	Misha Hafeez Umair Sajid Ummey Haani Shad

2.1 Problem Statement

TABLE 2 PROBLEM STATEMENT

The problem of	Mostly intrusion detection systems work on signature-based anomaly detection system. Specifying the detection signatures is a highly qualified, and time- consuming task.
That affects	It has the possibility of catching novel intrusions, as well as variation of known intrusion in no time.
The impact of which is	Radar security will have a great impact on the security of organizations by reducing the cost of security systems.
A successful solution would be	An ML-based technique applied to such a problem which will result in successful detecting of the intruders.

2.2 Business Opportunity

Problems that require continuous, constrained, and deterministic optimization can be resolved to an optimal solution using the ML-based techniques that we will implement and map to our problem. Such problems exist in many real-life situations, which include different constraints. Organizations require security when it comes to the safety of sensitive data. Our ML-based technique will help them find an optimal solution to their hassles so they can have the utmost security.

2.3 Objectives

- Radar security will help avoid malicious threats.
- Radar security will detect the inside threats which are hard to detect.
- Radar security will secure the sensitive information of authorized users from malicious user.
- Radar security will notify the System Security officer (SSO) about the malicious activities.
- Radar security will provide an interactive interface to manage and view security system.

2.4 Scope

Radar security will provide multiple features and use ML-based techniques to generate an efficient and real-life solution. Radar security will allow the user to log in, so it becomes usable. After logging in, the System security officer will be able to see the detected intruders (if any). The other features include user logs, activity logs, network logs, attack alarms, and filtering network logs. The Radar also will **capture the** user behavior and it will also **analyze** its behavior.

2.5 Constraints

2.5.1. Usability

The system will be user friendly by providing a graphic user interface that will be easy to use and understand for anyone with basic computer knowledge. The text will be readable from at least one meter. Non-technical users will be able to use the system.

2.5.2. Reliability

Once the software is installed it won't be lost in case of any kind of system failure.

2.5.4. Maintainability

Software will be easy to maintain and update. If one of functionality of this application update, it will not affect other features.

2.6 Stakeholders and User Description

The software is to detect intrusion in any organization. This software is expected to provide the following features.

- Login for System Security Officer.
- Provide Interactive User Interface.
- Collects the data of a user such as user logs, activity logs, and network logs in the data engine.
- Compare the user behavior with a predefined threshold with the help of a detection engine.
- Generate alarm in case of intrusion through the decision engine.
- Notification system to inform the System Security Officer about the intrusion through the portal.

(The stakeholders for this software are developers and the users of this software

2.6.1. Market Demographics

For organizations that deploy any system for the security of their sensitive information from any kind of intrusion, RADAR can be turned to cater to the needs of such organizations. The most common issue faced by organizations or system security officers is the threat of intrusion on their sensitive information from inside attackers. Thus, Radar can be proved as a viable software in such organizations, it helps them by detecting any kind of malicious activity so that they can take countermeasures timely. However, the target market for Radar is any organization that wants to secure its sensitive data or information.

2.6.2. Stakeholders Summary

TABLE 3 STAKEHOLDER SUMMARY

Name	Description	Responsibilities
FAST NUCES	This stakeholder has all rights to Radar security.	The stakeholder is responsible for rights and permission for development.
Development Team	The following stakeholder is responsible for designing and developing this product, the stakeholder will be analyzing the development and management of the product at certain levels as well.	The stakeholder has the responsibility to document the product efficiently and has the responsibility to meet the deadlines in the development. The stakeholder is responsible for maintaining a complete record of the product. Stakeholders will be solely responsible for the design schemes diagrams and architectures or during the production code.
Supervisor	The stakeholder includes one supervisor and one co-supervisor. Both act as project managers to make the teamwork.	They are responsible to manage the development teams and are responsible for managing the development needed. They act as a filter before the deadline and will be acting as leads for this complete project. They will also be marking the progress of the project.

2.6.3. User Environment

- One person will lead the development and two people will assist him to accomplish the project.
- The total duration of this project is One year (Two Semesters)
- We will complete our documentation within 2 months and will implement that in 10 months.
- We will be using python and the necessary frameworks to make the software

3. System Requirements Specification

Present section deals with Software Requirements Specification (SRS) for Fast Fitness center. It will cover the description of the website that is to be developed concerning standards of SRS documents in the software industry.

3.1 List of Features

- Dashboard
- Data collection
- Data computation
- Decision engine

3.2 Functional Requirements

Following functional requirements corresponding to the features are discussed here

3.2.1 Dashboard

- System security officer shall be able to see the system logs of all users
- System security officer shall be able to see the user logs of all active users
- System security officer shall be able to see the alarm
- System security officer shall be able to manage the history of alarm
- System security officer shall be able to see the alarm history of alarm

3.2.2 Data collection

- Operational system shall be able to collect the logs of user
- Operational system shall be able to send the logs to the data computation engine

3.2.3 Data engine

- Operational system shall be able to compute the data logs

3.2.4 Decision engine

- Operational system shall be able to run the decision engine to get the result about intrusion.

3.3 Non-Functional Requirements

Non-Functional requirements’ proper elicitation is important in any project’s success.

3.3.1. User Friendly Interface

Just like any other website the user interface matters a lot. Providing with a convenient interface to the user is necessary because without easy-to-use interface the user will be not feel comfortable in using the website.

3.3.2. Responsiveness

The interface is supposed to be responsive so that using features becomes fast and efficient. Our job will be to make sure that the code can be made effective and efficient to add maximum responsiveness to the website.

3.3.3. Interactivity

The interface should be fully functional so that each tab corresponds to the specific events in the web application. In our case the healthiest interface should be of webpage as it will be most used area. Moreover, it must be learnable.

3.3.4. Security

Our web-based project is highly secure because each user accesses his/her own data.

4. Use Case Diagram

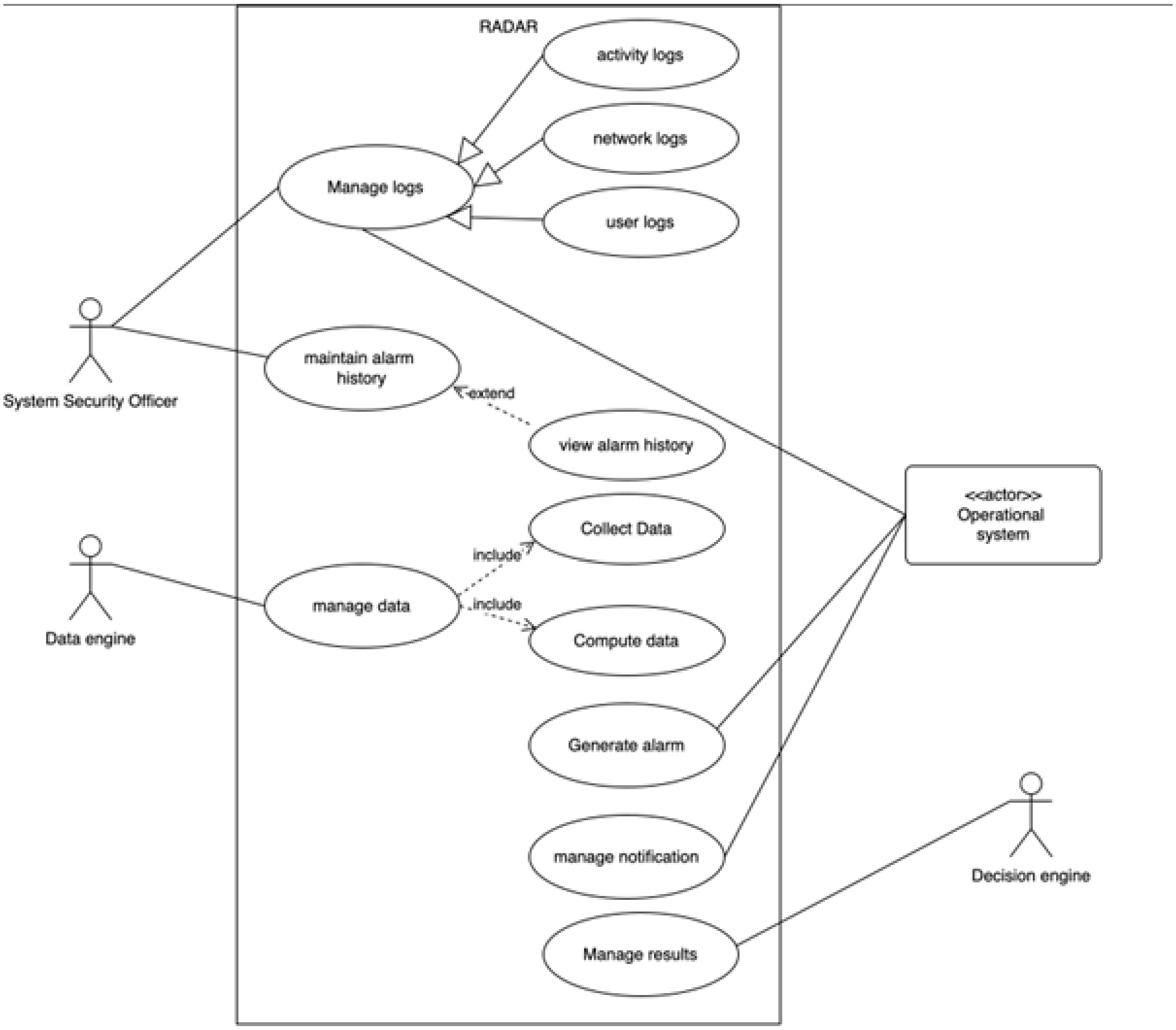

FIGURE 1 USE CASE DIAGRAM

5. High Level Use case

UC-1: Maintain Alarm History

TABLE 4 MAINTAIN ALARM HISTORY

Use Case:	Maintain Alarm history
Actors:	System Security Officer
Type:	Primary
Description:	System security officer will be able to see or modify history of previous attack alarms and this will be displayed on the dashboard.

UC-2: Generate Alarm

TABLE 5 GENERATE ALARM

Use Case:	Generate Alarm
Actors:	Operational System
Type:	Secondary
Description:	System will generate alarm after the detection of intruder by decision engine.

UC-3: Manage Notifications

TABLE 6 MANAGE NOTIFICATIONS

Use Case:	Manage Notifications
Actors:	Operational System
Type:	Secondary
Description:	System will be able to manage all the notifications which will be sent to System security officer after alarm generation.

6. Expanded Use Case

EUC-1: Manage Results

TABLE 7 MANAGE RESULTS

Use Case:	Manage results
Actors:	Decision engine
Type:	Secondary
Pre-condition:	Any activity should be performed by the user to have the data on which computations would be done.
Post-condition:	The computed data would be there to pass on the next engine.
Main Success Scenario	1. Decision engine will get computed data from the data engine. 2. Decision engine will also be able to get previous record from the system. 3. The computed data will help to take decision on computation basis. 4. Decision engine will generate result details and pass to the operational system so that it will take countermeasures. 5. System Security Officer would also be able to view the result details.
Alternate Scenario	System fails to detect any intruder.
Special Requirements:	There must be computed data so that decision engine can generate results efficiently.
Frequency of occurrence	High

EUC-2: Manage Data

TABLE 8 MANAGE DATA

Use Case:	Manage Data
Actors:	Data engine
Type:	Primary
Pre-condition:	Any activity should be performed by the user to have the data on which computations would be done.
Post-condition:	The computed data would be there to pass on the next engine.
Main Success Scenario	1. The computed data will help to detect the user. 2. The computed data will help to take decision on computation basis. 3. The computed data then ultimately help to generate alarm in case of any malicious activity. 4. System Security Officer would also be able to view all the data logs on dashboard.
Alternate Scenario	System fails to extract any user data. System will again fetch the data.
Special Requirements:	1. There must be performed any activity by the user. 2. Data engine must work properly.
Frequency of occurrence	High

EUC-3: Manage logs

TABLE 9 MANAGE LOGS

Use Case:	Manage logs
Actors:	System Security Officer, Operational system
Type:	Primary, Secondary
Pre-condition:	Any activity should be performed by the user to have the data so that SSO can view the logs, which computations would be done.
Post-condition:	SSO will be able to view logs and operational system can maintain the logs such as activity logs, network logs, user logs.
Main Success Scenario	1. SSO will be able to view logs of any user. 2. Operational system will be able to maintain logs. 3. The updated logs will help to detect intruder.
Alternate Scenario	System fails to maintain logs and will not be able to detect any intruder, also SSO can't be able to view updated logs.
Special Requirements:	Data engine must work properly.
Frequency of occurrence	High

7. System Sequence diagram

Manage Result

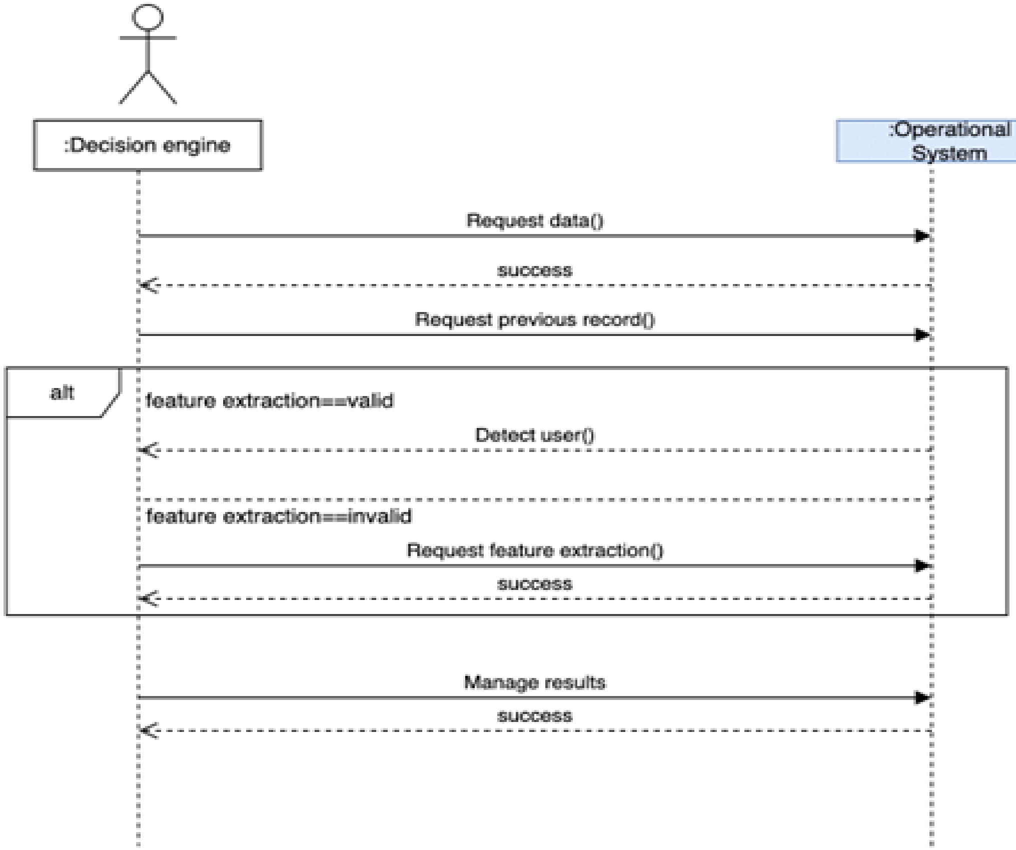
:Decision engine
:Operational System
Request data()
success
Request previous record()
alt
feature extraction==valid
Detect user()
feature extraction==invalid
Request feature extraction()
success
Manage results
success

FIGURE 2 MANAGE RESULT

Manage logs

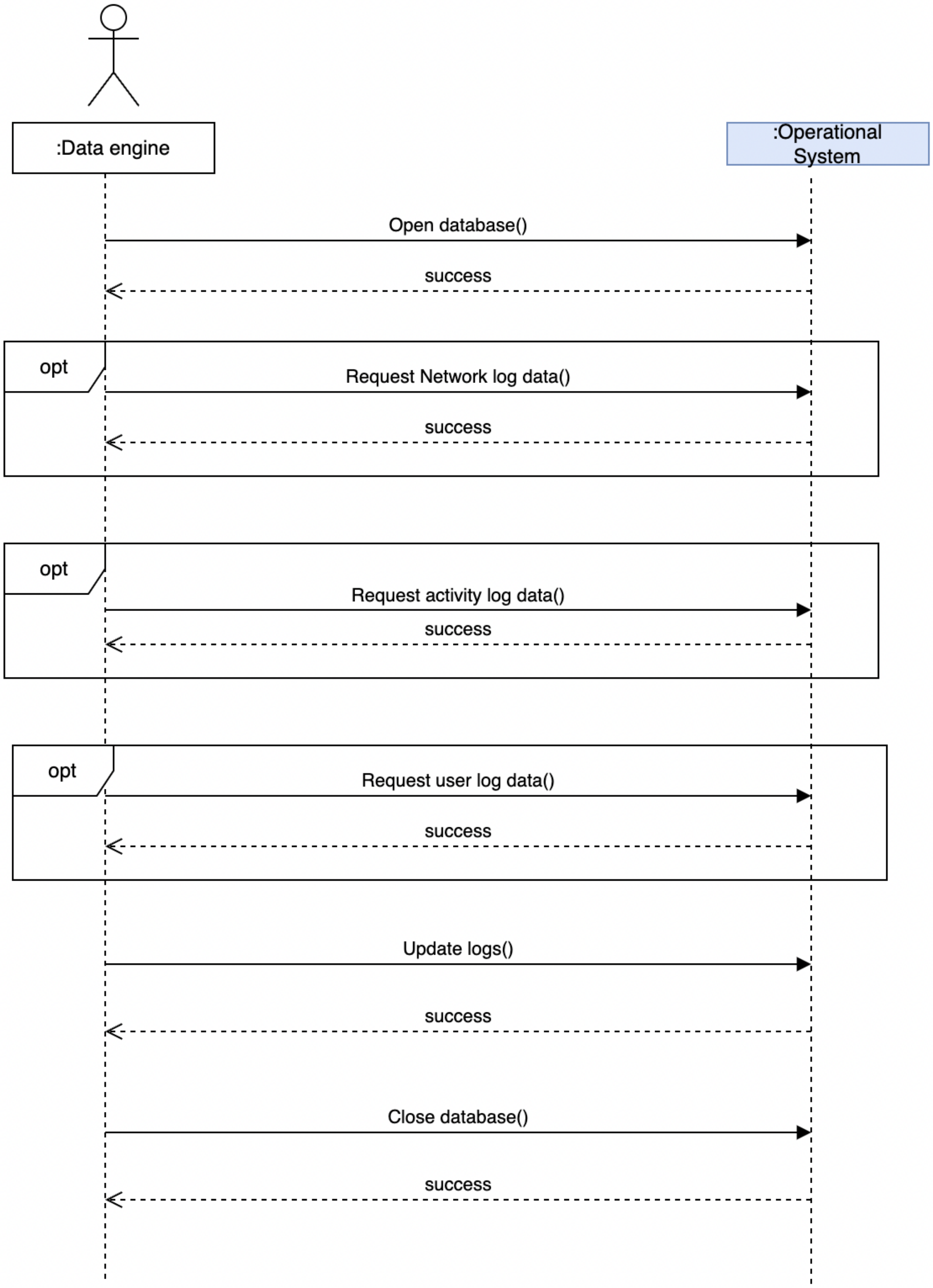

FIGURE 3 MANAGE LOGS

Manage logs(sso)

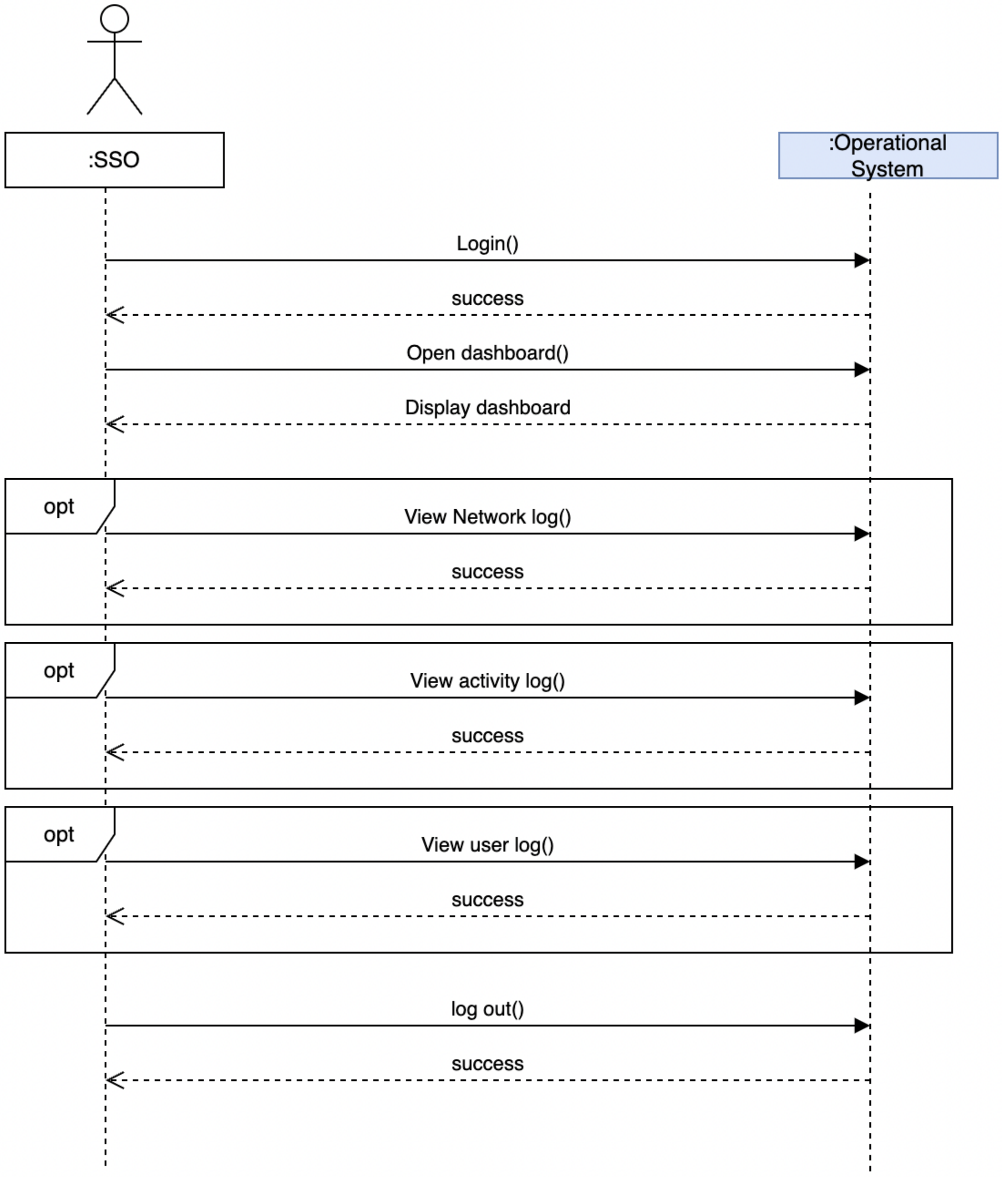

FIGURE 4 MANAGE LOGS(SSO)

8. Activity Diagram

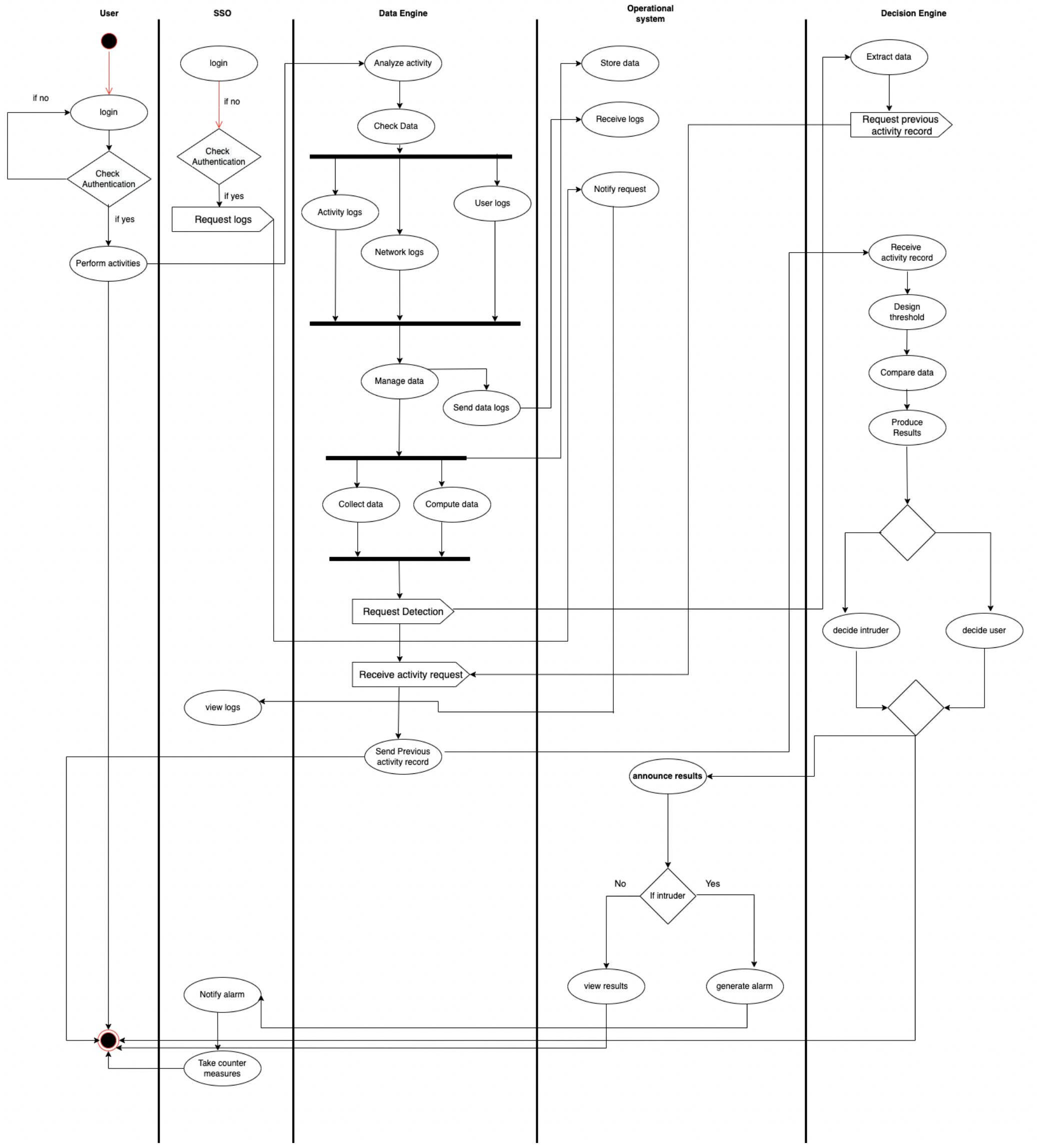

FIGURE 5 ACTIVITY DIAGRAM

9. Domain Model

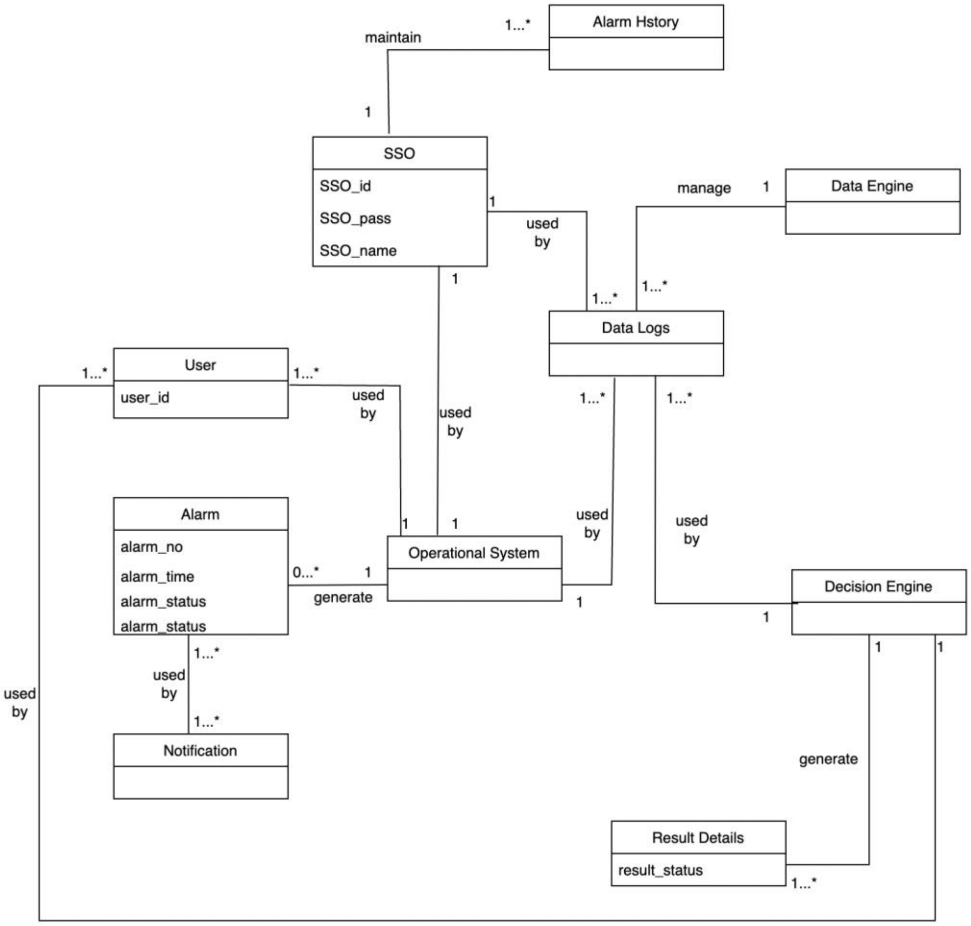

FIGURE 6 DOMAIN MODEL

10. Data Flow Diagram

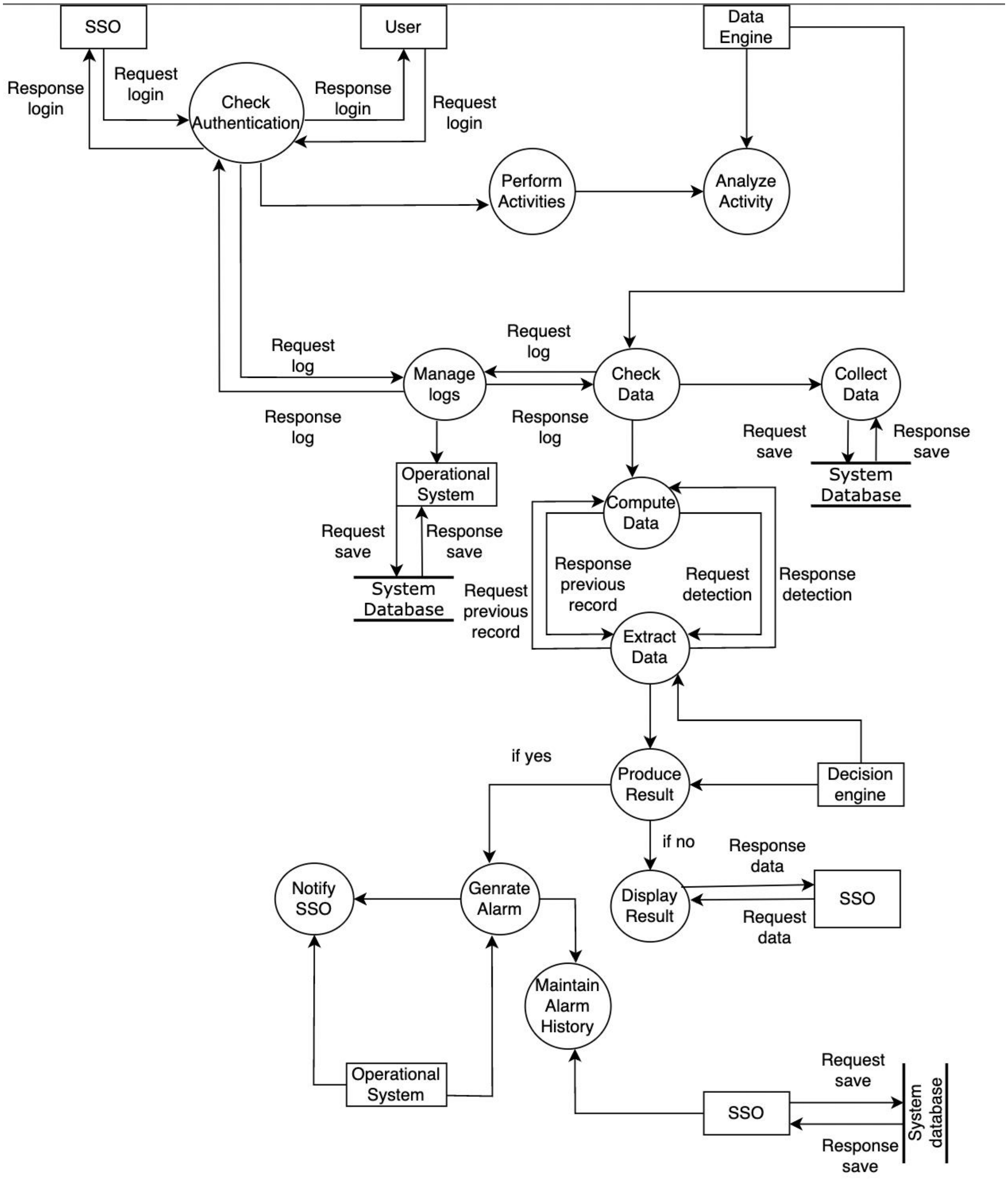

FIGURE 7 DATA FLOW DIAGRAM

11. Class Diagram

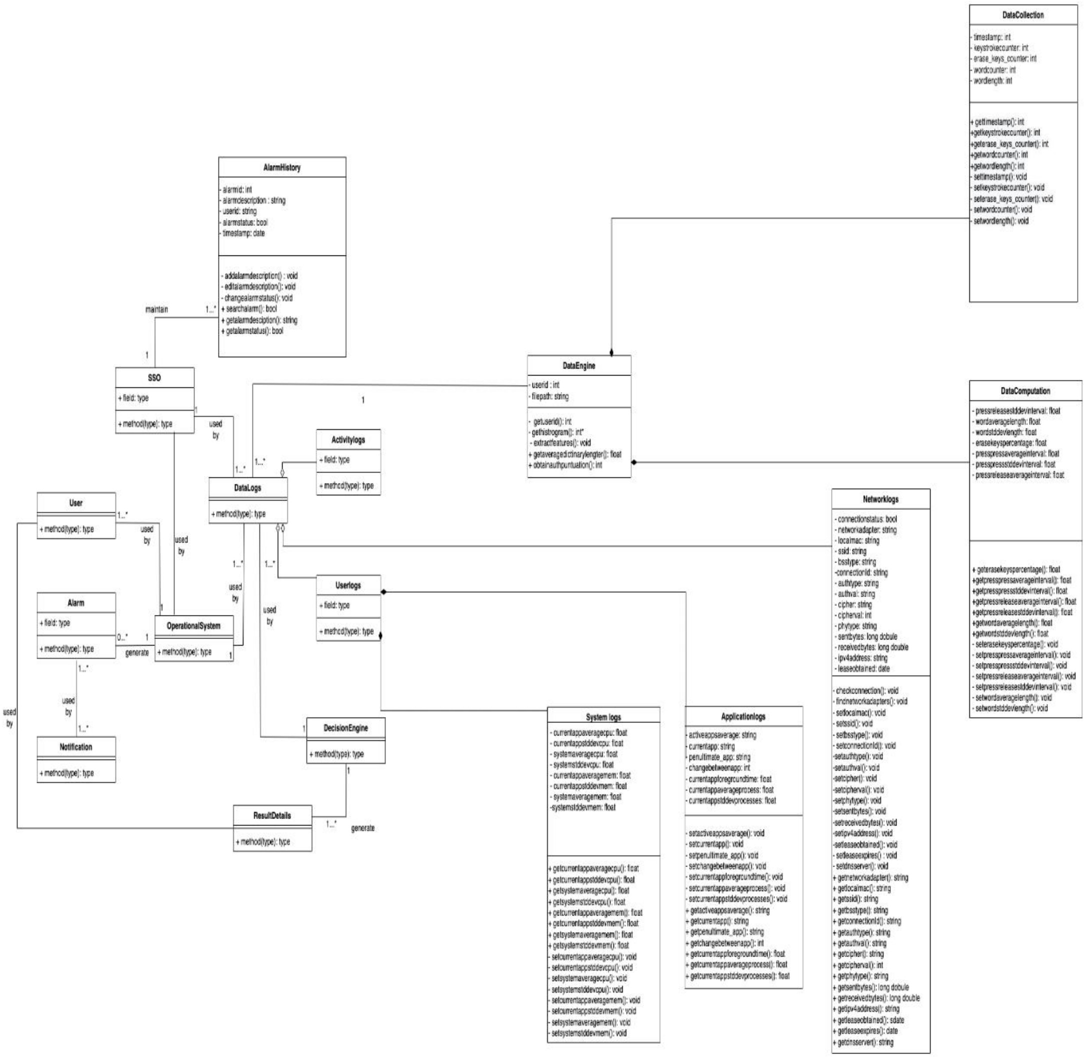
DataCollection
- timestamp: int
- keystrokecounter: int
- erase_keys_ocunter: int
- wordcounter: int
- wordlength: int
+ gettimestamp(): int
+getkeystrokecounter(): int
+geterase_keys_counter(): int
+getwordcounter(): int
+getwordlength(): int
- settimestamp(): void
- setkeystrokecounter(): void
- seterase_keys_ocunter(): void
- setwordcounter(): void
- setwordlength(): void
AlarmHistory
- alarmid: int
- alarmdescription : string
- userid: string
- alarmstatus: bool
- timestamp: date
- addalarmdescription() : void
- editalarmdescription(): void
- changealarmstatus(): void
+ searchalarm(): bool
+ getalarmdesciption(): string
+ getalarmstatus(): bool
maintain
1...*
1
SSO
+ field: type
+ method(type): type
1
used by
DataEngine
- userid : int
- filepath: string
- getuserid(): int
- gethistrogram(): int*
- extractfeatures(): void
+ getaveragedictinarylengter(): float
+ obtainauthpuntuation(): int
1
1...*
1...*
Activitylogs
+ field: type
+ method(type): type
DataLogs
+ method(type): type
DataComputation
- pressreleasestddevinterval: float
- wordaveragelength: float
- wordstddevlength: float
- erasekeyspercentage: float
- presspressaverageinterval: float
- presspressstddevinterval: float
- pressreleaseaverageinterval: float
+ geterasekeyspercentage(): float
+getpresspressaverageinterval(): float
+getpresspressstddevinterval(): float
+getpressreleaseaverageinterval(): float
+getpressreleasestddevinterval(): float
+getwordaveragelength(): float
+getwordstddevlength(): float
- seterasekeyspercentage(): void
- setpresspressaverageinterval(): void
- setpresspressstddevinterval(): void
- setpressreleaseaverageinterval(): void
- setpressreleasestddevinterval(): void
- setwordaveragelength(): void
- setwordstddevlength(): void
User
+ method(type): type
1...*
used by
used by
Networklogs
- connectionstatus: bool
- networkadapter: string
- localmac: string
- ssid: string
- bsstype: string
-connectionId: string
- authtype: string
- authval: string
- cipher: string
- cipherval: int
- phytype: string
- sentbytes: long dobule
- receivedbytes: long double
- ipv4address: string
- leaseobtained: date
- checkconnection(): void
- findnetworkadapters(): void
- setlocalmac(): void
- setssid(): void
- setbsstype(): void
- setconnectionId(): void
-setauthtype(): void
-setauthval(): void
-setcipher(): void
-setcipherval(): void
-setphytype(): void
-setsentbytes(): void
-setreceivedbytes(): void
-setipv4address(): void
-setleaseobtained(): void
- setleaseexpires() : void
- setdnsserver(): void
+ getnetworkadapter(): string
+ getlocalmac(): string
+ getssid(): string
+ getbsstype(): string
+ getconnectionId(): string
+ getauthtype(): string
+ getauthval(): string
+ getcipher(): string
+ getcipherval(): int
+ getphytype(): string
+ getsentbytes(): long dobule
+ getreceivedbytes(): long double
+ getipv4address(): string
+ getleaseobtained(): sdate
+ getleaseexpires(): date
+ getdnsserver(): string
1...*
1...*
used by
used by
Userlogs
+ field: type
+ method(type): type
Alarm
+ field: type
+ method(type): type
0...*
1
generate
1
OperationalSystem
+ method(type): type
1
1...*
used by
1...*
used by
Notification
+ method(type): type
System logs
- currentappaveragecpu: float
- currentappstddevcpu: float
- systemaveragecpu: float
- systemstddevcpu: float
- currentappaveragemem: float
- currentappstddevmem: float
- systemaveragemem: float
-systemstddevmem: float
+ getcurrentappaveragecpu(): float
+ getcurrentappstddevcpu(): float
+ getsystemaveragecpu(): float
+ getsystemstddevcpu(): float
+ getcurrentappaveragemem(): float
+ getcurrentappstddevmem(): float
+ getsystemaveragemem(): float
+ getsystemstddevmem(): float
- setcurrentappaveragecpu(): void
- setcurrentappstddevcpu(): void
- setsystemaveragecpu(): void
- setsystemstddevcpu(): void
- setcurrentappaveragemem(): void
- setcurrentappstddevmem(): void
- setsystemaveragemem(): void
- setsystemstddevmem(): void
Applicationlogs
- activeappsaverage: string
- currentapp: string
- penultimate_app: string
- changebetweenapp: int
- currentappforegroundtime: float
- currentappaverageprocess: float
- currentappstddevprocesses: float
- setactiveappsaverage(): void
- setcurrentapp(): void
- setpenultimate_app(): void
- setchangebetweenapp(): void
- setcurrentappforegroundtime(): void
- setcurrentappaverageprocess(): void
- setcurrentappstddevprocesses(): void
+ getactiveappsaverage(): string
+ getcurrentapp(): string
+ getpenultimate_app(): string
+ getchangebetweenapp(): int
+ getcurrentappforegroundtime(): float
+ getcurrentappaverageprocess(): float
+ getcurrentappstddevprocesses(): float
1
DecisionEngine
+ method(type): type
1
ResultDetails
+ method(type): type
1...*
generate

FIGURE 8 CLASS DIAGRAM

12. State Chart

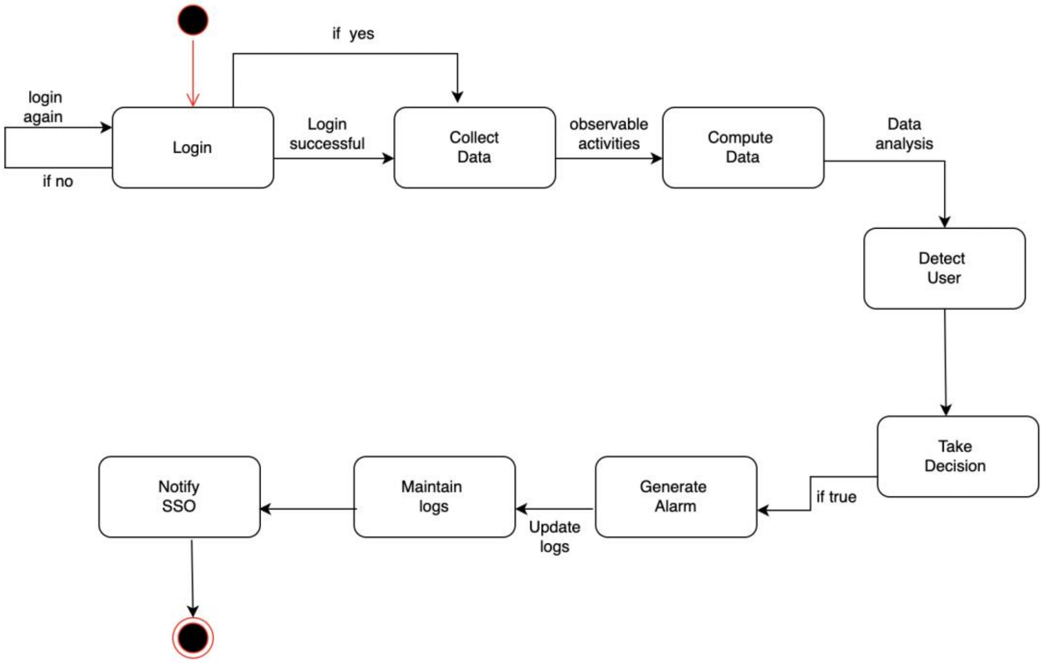

FIGURE 9 STATE CHART

13. Layer Architecture Diagram

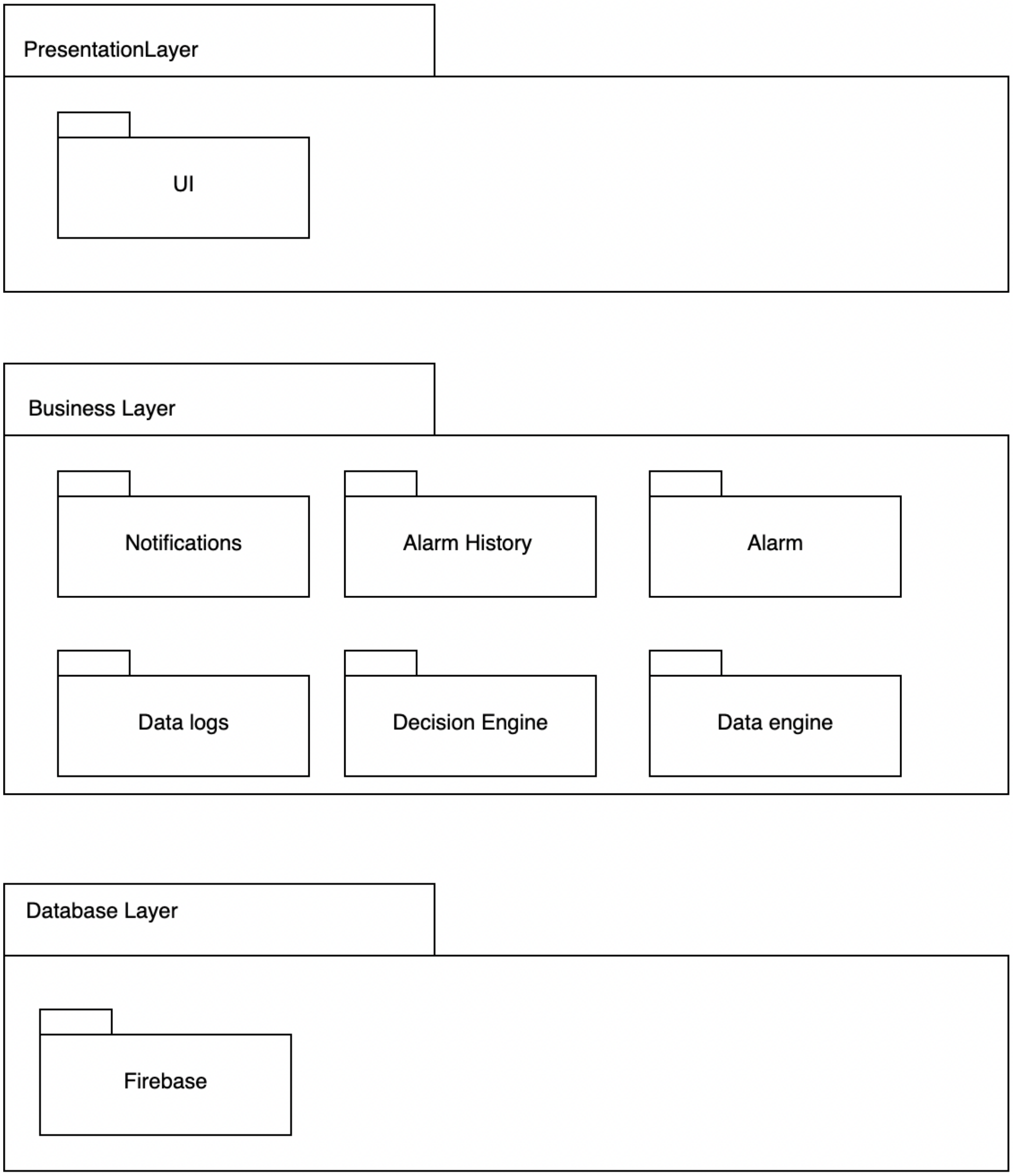

FIGURE 10 LAYER ARCHITECTURE DIAGRAM

14. Deployment

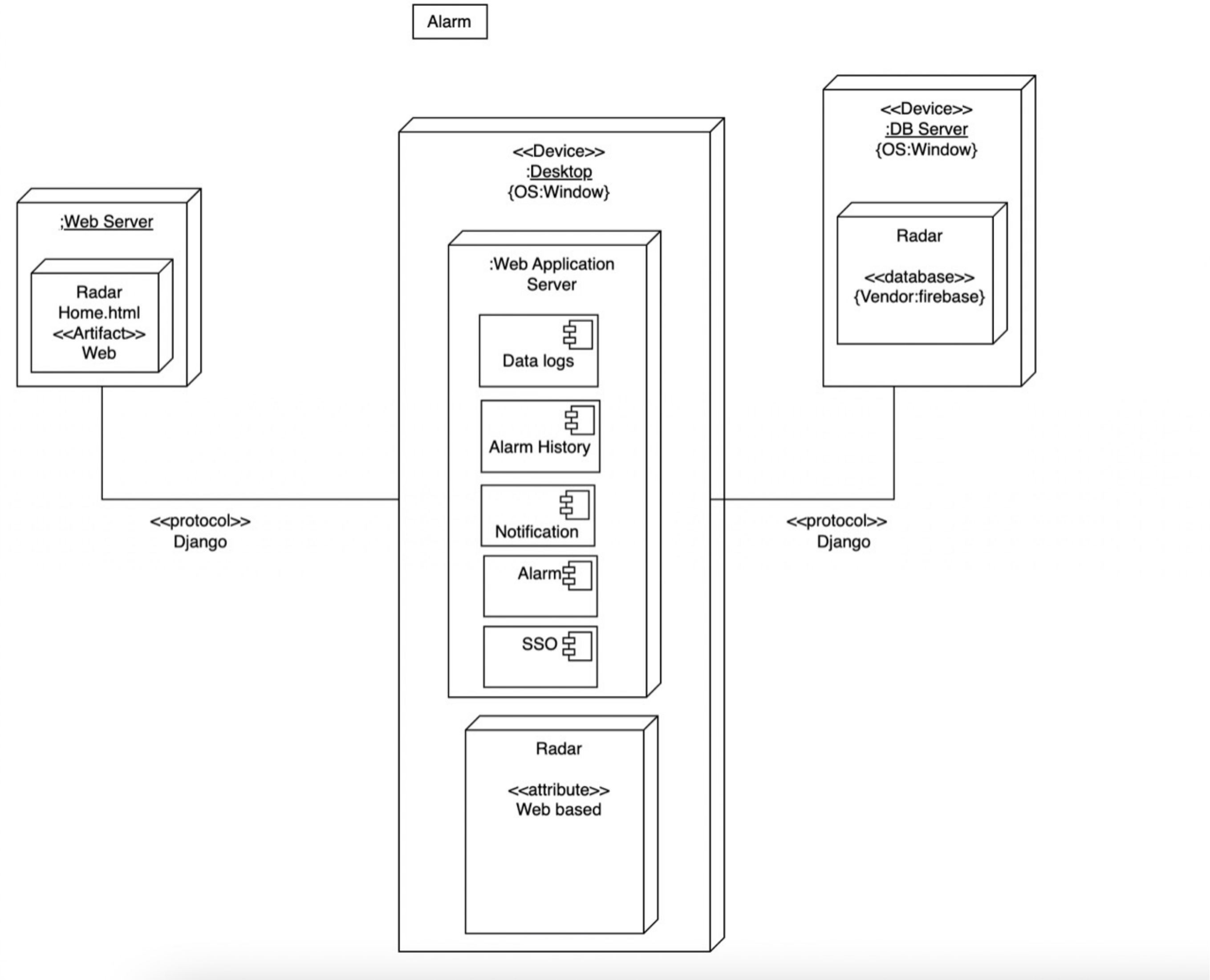

FIGURE 11 DEPLOYMENT DIAGRAM