WAF

Web Application Firewall

By

Basit Khan

Maryam Malik

Ayesha Mahmood

Submitted to Faculty of Department of Computer Software Engineering, National University of Science and Technology, Islamabad in partial fulfillment for the requirements of a B.E Degree in Computer Software, June 2019.

In the name of Allah, the most merciful, the most beneficent.

ABSTRACT:

Web Applications need to be secured in this era of e-commerce just like any of other business does.With this increase in web applications usage, the vulnerabilities are also increasing with time. Web applications are more prone to attacks, which is becoming a real problem for web application users now-a-days.

In the field of Computer security , a **vulnerability** is a fault/sensitive weakness which can be misused by such as some attacker, to perform unauthorized actions in your system . To exploit a vulnerability, attacker should have tool or connection technique to system weakness.

WAF protects these vulnerabilities. It monitors, and blocks HTTP traffic to and from a web application. It is a type of firewall that controls access to web application(s) from the Internet. It can be implemented as rule set on HTTP conversation. Through customizing the rules accordingly to application, several attacks (XSS, SQL injection, etc. can be identified and blocked. It offers a single source of control for the security of websites, applications, and APIs, hosted across multiple cloud environments. As a cloud based service, this WAF requires no hardware or software to install and maintain.

CERTIFICATE FOR CORRECTNESS AND APPROVAL

It is certified that work contained in the thesis WAF, carried out by Basit Khan, Maryam Malik, Ayesha Mahmood under supervision of Col. Adnan Ahmad khan for partial fulfillment of Degree of Computer Software Engineering is correct and approved.

Approved by

Dr Col. Adnan Ahmad Khan

HOD

Department of CSE

MCS, NUST

Dated: 1 May

DECLARATION

No portion of the work presented in this dissertation has been submitted in support of another award or qualification either at this institution or elsewhere.

DEDICATION

In the name of Allah, the Most Merciful, the Most Beneficent

To our parents, without whose support and cooperation,
a work of this magnitude would not have been possible

To our supervisor Col Adnan Ahmad Khan who has given us great support and valuable suggestions throughout the implementation process.

And finally, to our Friends and siblings for their encouragement.

ACKNOWLEGEMENTS

There is no success without the will of Allah Almighty. We are grateful to Allah, who has given us guidance , strength and enabled us to accomplish this task. Whatever we have achieved, we owe it to Him, in totality. We are also grateful to our parents and family and well- wishers for their admirable support and their critical reviews. We would like to thank our supervisor Col. Adnan Ahmad Khan for his continuous guidance and motivation throughout the course of our project, without his help, we would not have been able to accomplish anything.

Table of Contents

INTRODUCTION

Software Requirements Specification (SRS) provides an overlook alongside purpose, scope, definitions, acronyms, abbreviations, references and overview of the SRS. Main goal of this document is to provide detailed description of the WAF (Web Application Firewall) by defining the problem statement.

1.1 Purpose

This document includes software requirements for WAF. With this increase in web applications usage, the vulnerabilities are also increasing with time. Web applications are more prone to attacks, which is becoming a real problem for web application users now-a-days.

The purpose of this project is to build Web Application Firewall which is a type of firewall that controls access to web application(s) from the Internet. It can be implemented as rules apply on HTTP conversation.Also customizing the rules to accordingly to application, attacks like (XSS, SQL injection, etc.) can be identified and blocked.

The purpose of this document is to state the software requirements for project that is “WAF (Web Application Firewall)”. This document covers all basic features, objectives and attributes of the proposed system. It will give details on objectives and features of the system, the user interfaces of the system, system’s functionality , the constraints under which it must operate and system’s response. For the developer, it will be a reference point during software design, implementation and maintenance.

1.2 Document Conventions

The document is presented here according to standard IEEE format. It follows a convention that involves boldfacing of headings, the use of indentions and numbering for major parts and subparts. It is divided mainly into an overall description,nonfunctional requirements, system features,user interface requirements.

1.2.1 Headings:

Heading are prioritized in a numbered fashion, the highest priority heading having a single digit and subsequent headings having more numbers, according to their level.

All the main headings are titled as follows: single digit number followed by a dot and the name of the section (All bold Calibri (Body), size 18).

All second level subheadings for every sub section have the same number as their respective main heading, followed by one dot and subsequent sub heading number followed by name of the sub section (All bold Calibri (Body), size 16).

Further subheadings, i.e. level three and below, follow the same rules as above for numbering and naming, but different for font size (All bold Calibri (Body), size 14).

1.2.2 Figures:

All figures in this document have captions, and are numbered. Context and flow diagrams are based on UML standards.

1.2.3 Reference:

All references in this document are provided where necessary, however where not present, the meaning is self-explanatory. All ambiguous terms have been clarified in the glossary at the end of this document.

1.2.4 Links to web pages:

All links have been provided with underlined font, the title of the web page or e-book is written at the top of the link and the title may be searched on Google to pinpoint to the exact address.

1.2.5 Basic Text:

All other basic text appears in regular, size 12 Times New Roman. Every paragraph explains one type of data.

1.3 Intended Audience and Reading Suggestions

1.3.1 Intended Audience

It includes:

- **UG Project Evaluation team:**

  The document will provide grounds to evaluation team for checking on progress of the project. It will provide the evaluation team with the scope, requirements and details of the project to be built. It will also be used as basis for the evaluation of the implementation and final project.

- **Developers**

  It will provide guidance to the developers to determine what the requirements are and how they should continue with the project.

- **Project Supervisor**

  It will help the supervisor to supervise the project and guide the team in a better way. This document will be used by him to check whether all the requirements have been understood and in the end whether the requirements have been properly implemented or not.

- **Testers**

The testers of the system can check user requirements from this SRS and develop the test document accordingly.

- **Up gradation Engineers**

  Up gradation engineers can review projects capabilities.They can look for more feature which can enhance the project. It guides for future developments.

  - **End Users**

    This document can be read by the end users if they wish to know what the project is about and what requirements have been fulfilled in this project

1.3.2 Reading Suggestions

The SRS begins with the title and table of contents.Each main heading is succeeded by a number of sub headings, which are all in bold format. The product overview is given at the start, succeeded by the complete detailed features, includes functional and non-functional requirements. Detailing of interface is also given. This SRS ends with appendices, including a glossary.

.

1.4 Project Scope

This project will assist users who are most concerned with their web application's security and want their data to remain protected. It will be done by letting WAF automatically filter out illegitimate traffic based on rule sets that you specify. It looks at both GET and POST-based HTTP requests and applies a rule set, such as the OWASP Top 10 vulnerabilities to determine what traffic to block, challenge, or let pass through. It can block cross-site scripting attacks, and SQL injections.t offers a single source of control for the security of websites, applications, and APIs, hosted across multiple cloud environments.As a cloud-based service, our WAF requires no hardware or software to install and maintain.

For	Web Applications
What	Web Application Firewall act as a security wall between web page(customer's website) and internet
The	WAF(Web Application Firewall)

Is	Web Service/Web Application
That	Provides protection to Web Applications by analysing the HTTP traffic to your website.

1.5 References

1.5.1 IEEE Computer Society Conventions:

- **System Requirements Specification Template providing the format:**

  http://www.cse.msu.edu/~cse870/IEEEXplore-SRS-template.pdf

OVERALL DESCRIPTION

2.1 Product Perspective

Web Applications need to be secured in this era of e-commerce just like any of other business does. There may be some concerns related to security and regulations if the data being stored is of personal kind. If the personal data of any client/user gets in hand of a hacker he could black mail that user.Also the financial data being in wrong hands could lead to huge loses to the company and much more. Similarly ,If a website gets down due to exploitation of any vulnerability it will affect the company's reputation and financial matters.This shows how valuable is the security for web applications and therefore there is a need for some mechanism to secure them.

WAF gives security to websites frombasic common vulnerabilities like SQL injection attacks, cross-site scripting, etc with no changes to your existing infrastructure.

2.2 Product Function

Our WAF protects your web properties from the OWASP top 10 vulnerabilities. These OWASP rules are implemented to protect web applications from a wide range of attacks . WAF rules that you can apply with the click of a button.

- We will provide a website where user can login/sigup.
- The user can choose the features to implement from the website.
- The HTTP traffic will then get analyzed based on those rules.
- The WAF will detect anything out of the ordinary (suspicious user behaviour).

2.3 User Classes and Characteristics

Below points describes the kind of users of WAF. There are explanations of the users:

2.3.1 Companies

Companies having online businesses working on their own web applications, needs to protect their websites (including the confidential data) from third parties/hackers.

2.3.2 Government Officials

Government Departments can also use this WAF for their websites .

2.3.3 Tester (occasional user)

The testers of the system can check user requirements from this SRS and develop test scenarios accordingly.

2.3.4 Developers

The developers will use this at the developing time and at the time of any defect occurred in the product during maintenance.

2.3.5 Documentation Writers

The document can serve as a future reference for other versions of the SRS.

2.3.6 Figure

Both users and their interaction with the system shownbelow

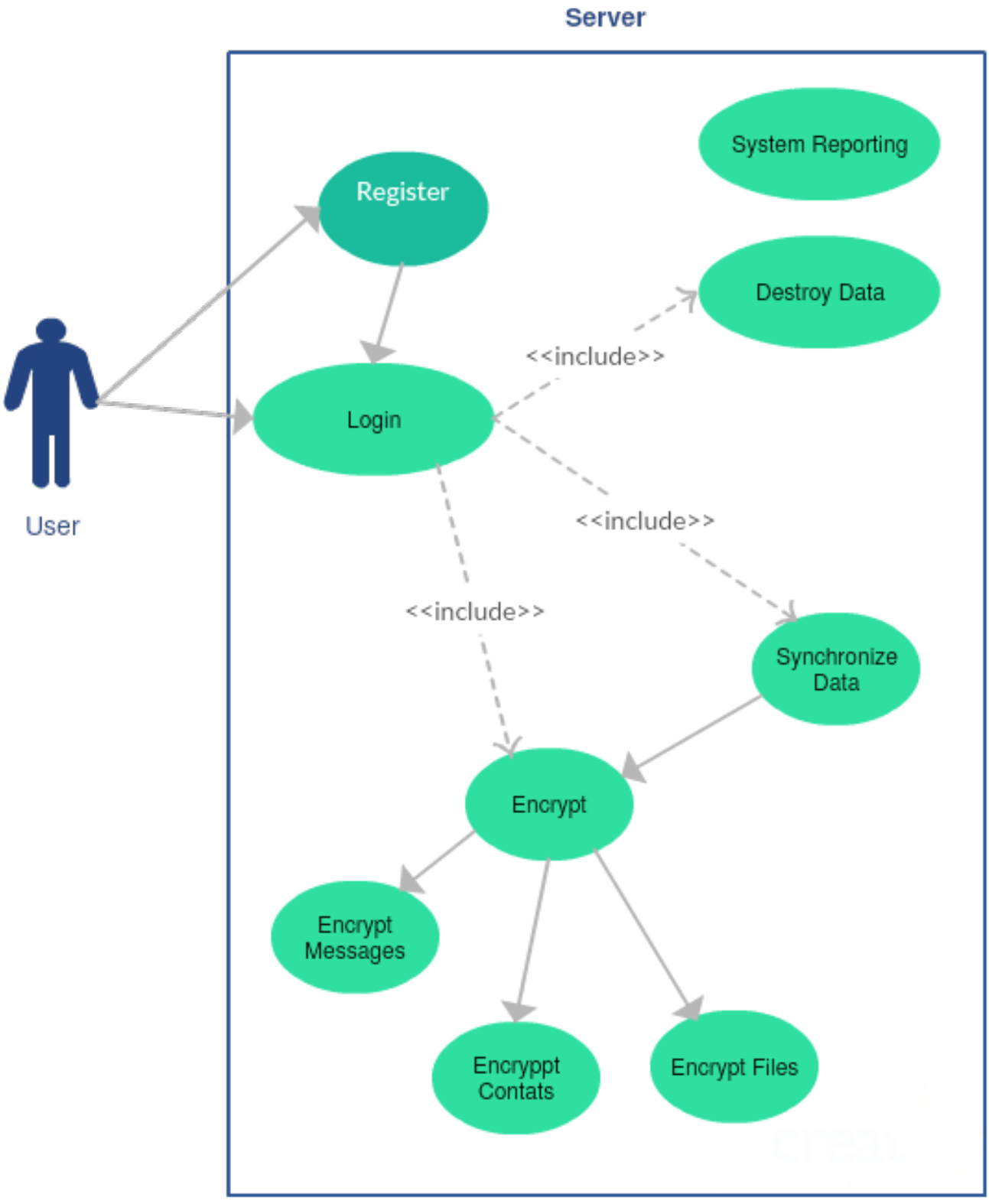

Server
System Reporting
Register
Destroy Data
<<include>>
Login
User
<<include>>
<<include>>
Synchronize Data
Encrypt
Encrypt Messages
Encryppt Contats
Encrypt Files

Figure 1.0

2.4 Operating Environment

- Web Application
- Cloud Deployment

2.5 Design and Implementation Constraints

- Not deployed as a server/hardware device.
- User needs to subscribe to our website.
- User needs to select features/rules form the site to implement.

2.6 User Documentation

- A user manual will be provided which will help new users to get started with the Web Application Firewall. The user manual will provide the instructions on how to deploy the WAF features.
- A summary will also be provided to the user which will highlight the product features and limitations.

2.7 Assumptions and Dependencies(add more)

- The website needs to get subscribed to be able to protect your web content.
- The users may remain unaware of the functionality of WAF as the responsibility is on shoulders of third party.

EXTERNAL INTERFACE REQUIREMENTS

3.1 User Interfaces

- Main menu for navigation will be used.□
- Interface will be user friendly and the standard English-US will be used

3.2 Hardware Interfaces

- Incoming traffic will be monitored before directing it to the server.

3.3 Communication Interfaces

- Internet will be used as medium of communication to the servers.(should we mention cloud?)

System Features

This section tells about the system features of WAF.

4.1 Accessing the Main Menu

4.1.1 Description and priority

After logging in /signing up to the WAF website the features to be implemented will be displayed as rules in the main menu.

Its priority will be high as without this feature the application will not be navigable and the user will not be able to select desired actions.

4.1.2 Stimulus/Response Sequence

1. Open the website.
2. Login/signup to your account
3. Access the main menu.

4.1.3 Functional requirements

1)User shall access the website and create an account.
2)The different options available shall be
a)Customized rules to be deployed
b)Select one's own desired rules to deploy
c)Help menu
d) logout

REQ-3: At any time user can logout from the website whenever he wants to.
.

4.2 Features To be deployed

4.2.1 Monitors Traffic

It scrutinizes inbound traffic for threats and outbound traffic for sensitive data.

4.2.2 Data Leak Prevention

It aids in Data Leak Prevention by looking for sensitive data output from the application such as credit card numbers and other application specific sensitive data.

4.2.3 Mitigates the risk of unknown attacks

It mitigates the risk of unknown attacks by watching for unusual or unexpected patterns in the traffic and defending against them.

4.3 Rules Offered

4.3.1 Description and priority

This holds much importance as is a vital feature.

4.3.2 Stimulus/Response Sequence

1. The user selects Rules to be deployed option from main menu.

2. User selects the rules available from the list, based on their priority.

3. System then lets user to view the security benefits offered by that rule.

4.3.3 Functional requirements

REQ-4: List of the rules shall be displayed for user to select.
REQ-5: Set of instructions shall be provided on how to subscribe the rule after selecting a particular rule.
REQ-7: User shall be able to reconsider a rule through a prompt.
REQ-8: Security benefits of the rule shall be displayed when that particular rule is selected.
REQ-9: User shall be able to cancel a particular selected rule and move to main menu when required

4.4 HELP MENU

4.4.1 Description and Priority

Help menu holds a medium priority. It will contain all the instructions needed to use the application.

4.4.2 Stimulus/Response Sequences

- User clicks on Help Menu from Main Menu
- An instruction manual is displayed to guide the user.

4.4.3 Functional Requirements

REQ-10: Choosing Help Menu option shall show Instruction Manual.

NON FUNCTIONAL REQUIREMENTS

5.1 Safety Requirements

The use of the WAF has no harms whatsoever; nor does it have any possibility of loss or damage that might be inflicted with no changes to your existing infrastructure.

5.2 Performance Requirements

- It shall not crash by accident.
- It allows for faster response to varying attacks.
- The features/rules can be implemented with speed and ease .

5.3 Security Requirements

- WAF will not ask for personal details.
- Personal data of any company/web application will not be compromised in any way.

5.4 Software Quality Attributes

5.4.1 Usability

The top most priority is given to user interface. The website will be easy to use and appealing to the user.

5.4.2 Reliability

It shall provide reliability to the user. The WAF will run stably with all the features mentioned above available and executing perfectly. It shall be tested and debugged completely. All exceptions shall be well handled.

5.4.3 Portability

In API, portability can be defined as “compatibility of application with platform”. As our WAF is implemented as a (cloud based product)website therefore it will be compatible with all latest browser versions.

5.4.4 Flexibility

New requirements can easily be entertained by the design and architecture of website even at some later stage of modification.

Scalability

The WAF is expected to handle many users at a time.

5.4.5 Availability

The services will be available 24/7, provided user is having an online connection.

5.5 Business Rule

The WAF website will be made available online for universal accessibility

BIBLIOGRAPHY

Similar Projects at MCS

1. BitVise XTS-AES Based Disk Encryption Software
   by Myra Khalid, LaraibZahid and Usama Ahmad

2. A similar approach was made by Amit Banerjee, Muhamadul Hassan, MD. Auhidur Rahman and Rajesh Chapagain, Department of Computer Science, South Asian University, New Delhi 110021, India.

   CLOAK: A Stream Cipher Based Encryption Protocol for Mobile Cloud ComputingLink: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8016572

SOFTWARE DESIGN AND DEVELOPMENT

7.1 Introduction

In computer security, a **vulnerability** is a weakness which can be exploited by a Threat Actor, such as an attacker, to perform unauthorized actions within a computer system. To exploit a vulnerability, an attacker must have at least one applicable tool or technique that can connect to a system weakness.

WAF protects these vulnerabilities. It checks on , and put a stop on HTTP traffic accessing the web application . This is a type of firewall that controls access to web application(s) from the Internet. It can be developed to apply a set of rules to an HTTP conversation. By customizing the rules to your application, many attacks (XSS, SQL injection, etc. can be identified and blocked.

7.2 Purpose

This document includes software design for WAF. It specifies the detailed architectural design of Web Application Firewall which is being developed. It will act as a guideline for developers and all the other stakeholders throughout the development. Document include classes and their inter-relationships, use cases with detailed descriptions, sequence diagrams, activity diagrams and various other.

7.3 SCOPE

This project will assist users who are most concerned with their web application's security and want their data to remain protected. It will be done by letting WAF automatically filter out illegitimate traffic based on rule sets that you specify. It looks at both GET and POST-based HTTP requests and applies a rule set, such as the OWASP Top 10 vulnerabilities to determine what traffic to block, challenge, or let pass through. It can block cross-site scripting attacks, and

SQL injections. It offers a single source of control for the security of websites, applications, and APIs, hosted across multiple cloud environments. As a cloud-based service, our WAF requires no hardware or software to install and maintain.

7.4 DEFINITION, ACRONYMS AND ABBREVIATIONS

UML: The Unified Modeling Language (UML) is a general-purpose modeling language in the field of software engineering, which is designed to provide a standard way to visualize the design of a system.

SDS: Software Design Specification
GUI: Graphical User Interface
WBS: The project management Work Breakdown Structure
WAF: Web Application Firewall
OWASP: Open Web Application Security Project
Services= Rules
Web application=app
Sign up=Register

7.5 REFERENCES:

•**Use Case Modeling Guidelines,** which documents the guidelines used to develop the use case model specifying the functional requirements in this specification.
https://en.wikipedia.org/wiki/Sequence_diagram
https://en.wikipedia.org/wiki/Component_diagram
https://www.google.com/search?q=cloudflare&rlz=1C1CHBF_enPK818PK818&oq=cloudflare&aqs=chrome..69i57j0j35i39l2j0l2.3332j0j7&sourceid=chrome&ie=UTF-8
https://modsecurity.org/;

DOCUMENT OVERVIEW

8.1 WORK BREAKDOWN STRUCTURE:

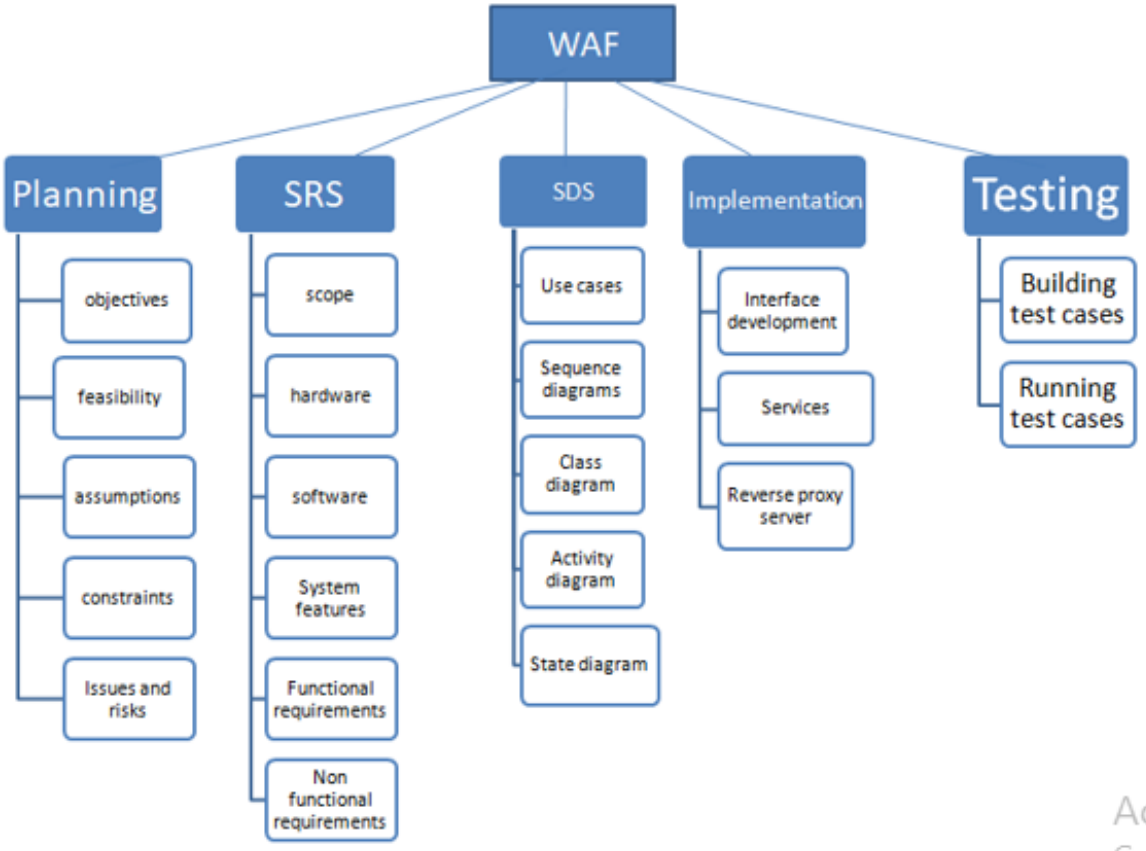


Fig 1.6.1: WorkbreakDown

8.2 SYSTEM ARCHITECTURE DESCRIPTION:

Detailed description of system architecture and design pattern which this system is going to use is discussed later in the document. ” Design Decisions and Tradeoffs‟. This Section gives overview of application, its higher and lower levels details and user interfaces.

8.3 Structure and Relationships:

This section covers the overall technical description of **WAF**. It shows the working of application in perspective of different point-of-views and also shows relationships between different components.

8.3.1 System Block Diagram:

The diagram(s) show the higher level description of the application(s), generic working of the application(s) and interaction with the user. User interacts with WAF App and then choose among different options given in interface and finally the selected WAF service is provided to the user.

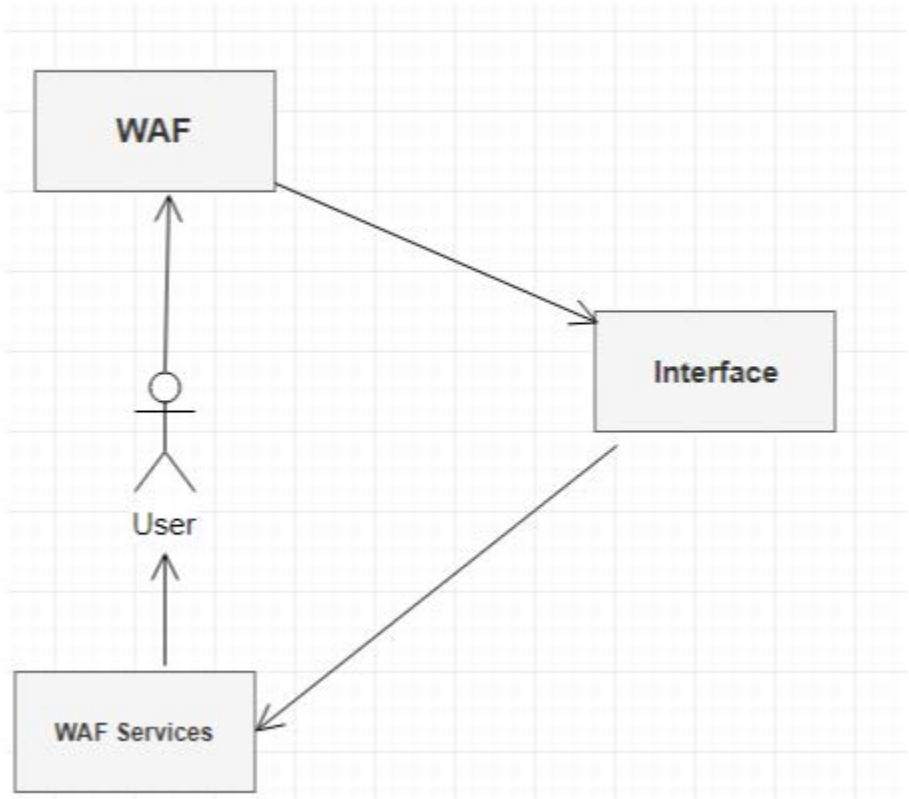


Fig 2.1.1.1: Block Diagram

8.3.2 Component Diagram

Main Components are:

- User Interface(UI)
- User

- Application

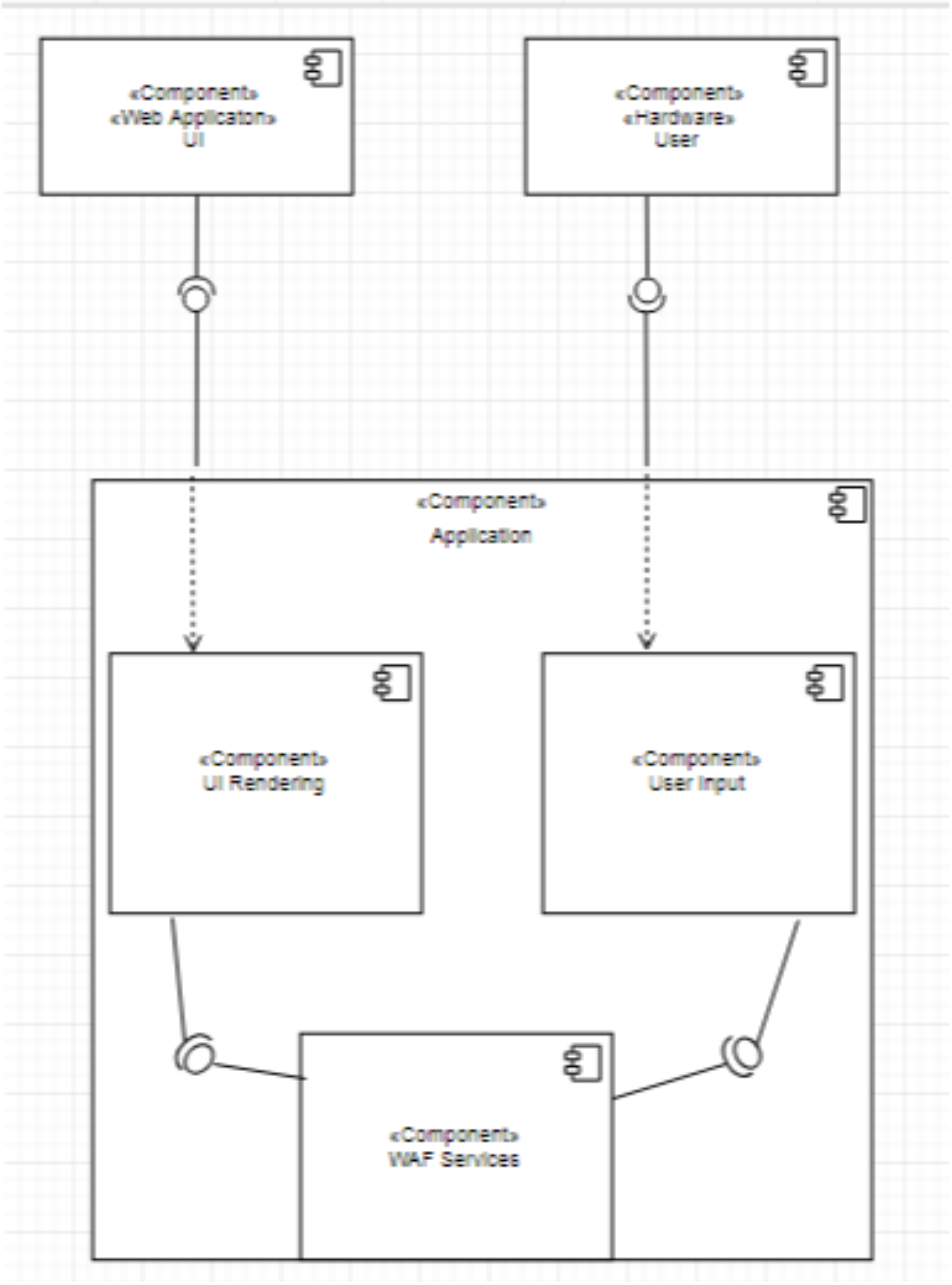


Fig 2.1.2.1: Component Diagram

8.3.3 User View (Use Case diagram)

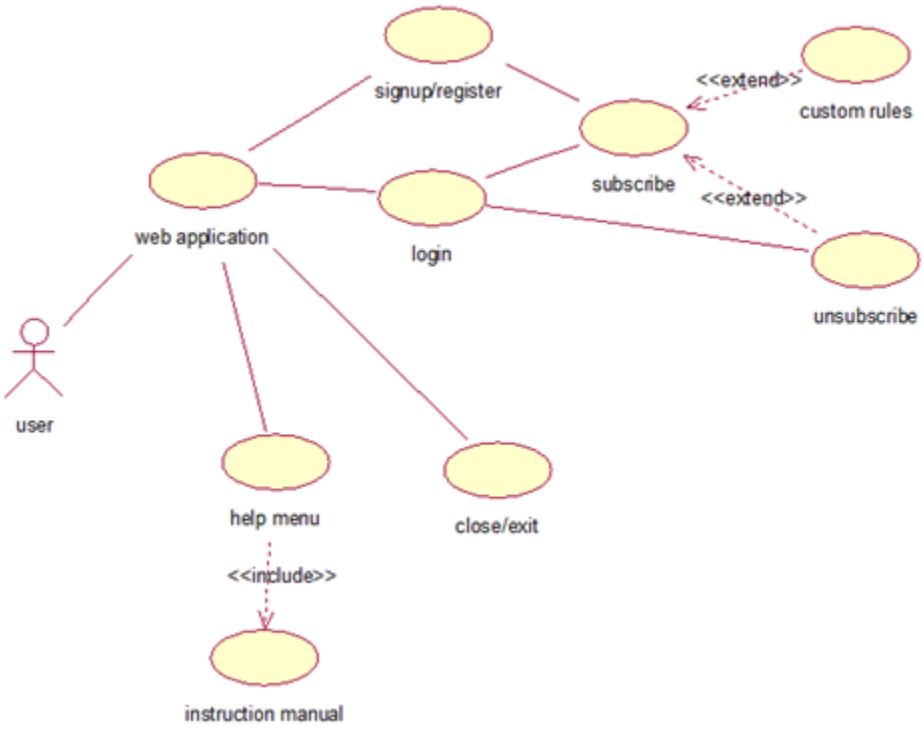


Fig 2.1.3.1: Use Case Diagram

WEB APPLICATION

USE CASE	WEB APPLICATION
Actor	User
Normal Flow	The application provides the service without any error.
Alternative Flow	Service gets down due to system failure or becomes offline redirecting to exit the page.
Pre -Condition	User visits the app page.
Post-Condition	Different menu options are displayed like (i)Login (ii)Sign up (iii)Exit (iv)Help
Extends	N/A
Includes	N/A
Assumptions	User has a working internet connection.

SIGN UP

USE CASE	SIGN UP
Actor	User
Normal Flow	User successfully register himself by providing required details.
Alternative Flow	User is not able to register due to invalid personal information or page errors.
Pre -Condition	User is visiting the page and he does not have an account.
Post-Condition	Account is created and User can subscribe and customize Rules.
Extends	N/A
Includes	N/A
Assumptions	User has a working internet connection.

LOG IN

USE CASE	LOG IN
Actor	User
Normal Flow	Logs in successfully
Alternative Flow	Error occured due to wrong username/password. Error message is displayed.
Pre -Condition	User has an account already registered.
Post-Condition	Menu is displayed to user. (i)Subscribe (ii)Unsubscribe (iii)Custom Rules

Extends	N/A
Includes	N/A
Assumptions	User has a working internet connection. Signed up.

HELP

USE CASE	Help Menu
Actor	User
Normal Flow	User finds the answer to what he is looking for.
Alternative Flow	Instruction manual does not get loaded and provide not so helpful information.
Pre -Condition	logged in
Post-Condition	N/A
Extends	N/A
Includes	Instruction manual
Assumptions	User has some knowledge about the application.

SUBSCRIBE

USE CASE	SUBSCRIBE
Actor	User
Normal Flow	User easily subscribe to a specific rule and continue working with it.
Alternative Flow	Error occurs if user doesn't fulfils the criteria to use the services.
Pre -Condition	User have an account
Post-Condition	Rules subscribed and WAF gets implemented. user can customize rules as well.
Extends	N/A
Includes	N/A
Assumptions	Logged in or signed up

CLOSE/EXIT

USE CASE	Close/Exit
Actor	User
Normal Flow	The application terminates.
Alternative Flow	Does not exit due to errors or incomplete activity.
Pre -Condition	Active Usage of the app.
Post-Condition	Directed out of the application.
Extends	N/A
Includes	N/A
Assumptions	User has a working internet connection.

UNSUBSCRIBE

USE CASE	Unsubscribe
Actor	User
Normal Flow	User no longer can use the unsubscribed service.
Alternative Flow	The services remain subscribed even the user does not want them anymore.
Pre -Condition	Active Usage of the app. Some services subscribed
Post-Condition	User may left with no more service to use.
Extends	subscribe
Includes	N/A
Assumptions	User has a working internet connection.

CUSTOM RULES

USE CASE	Custom Rules
Actor	User
Normal Flow	User can ask for customized solution to their problem or attack they face.
Alternative Flow	Unable to entertain the user request
Pre -Condition	Subscribed to services.

	Active usage of app.
Post-Condition	User will be able to use the solution ahead for similar attacks
Extends	subscibe
Includes	N/A
Assumptions	User has a working internet connection.

8.3.4 SEQUENCE DIAGRAMS

The sequence diagrams of working WAF is given below:

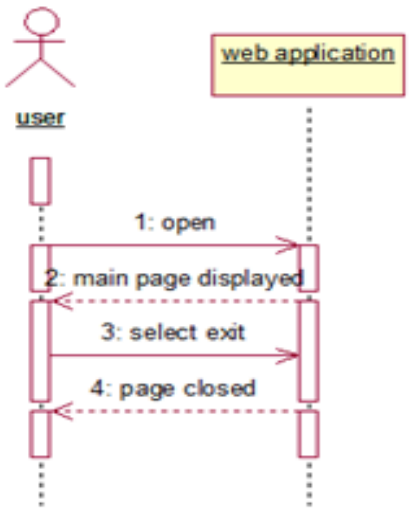


Fig 2.1.4.1: Sequence Diagram

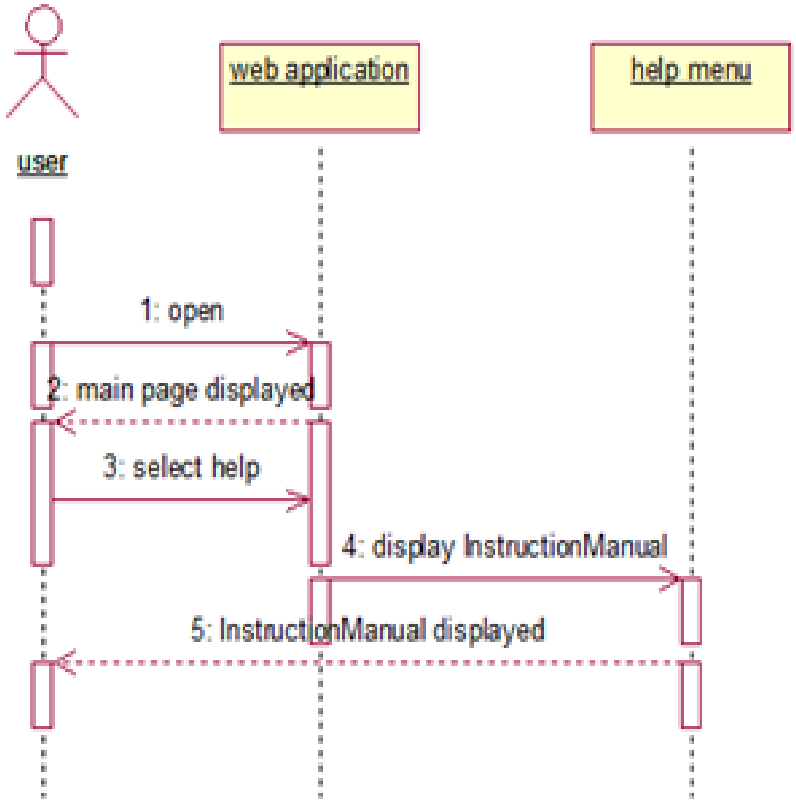


Fig 2.1.4.2: Sequence Diagram

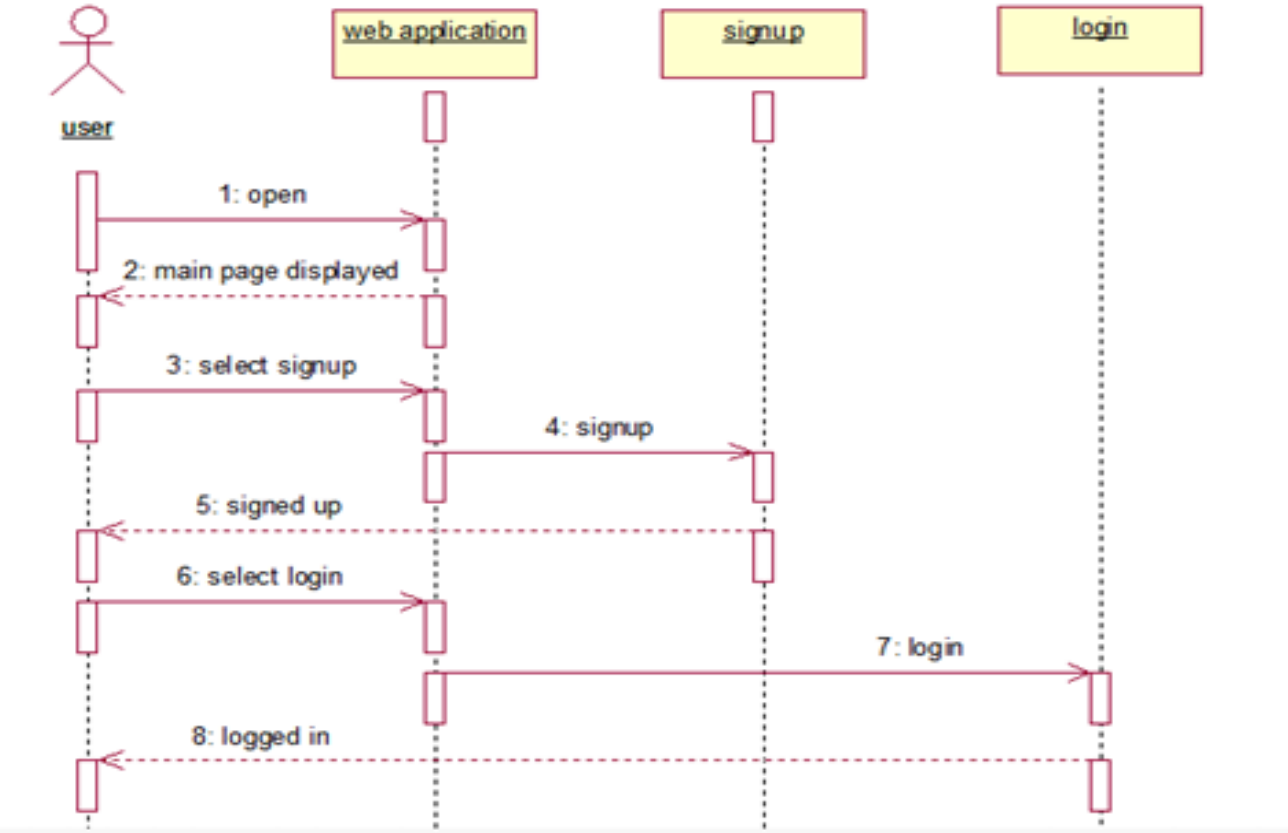


Fig 2.1.4.3: Sequence Diagram

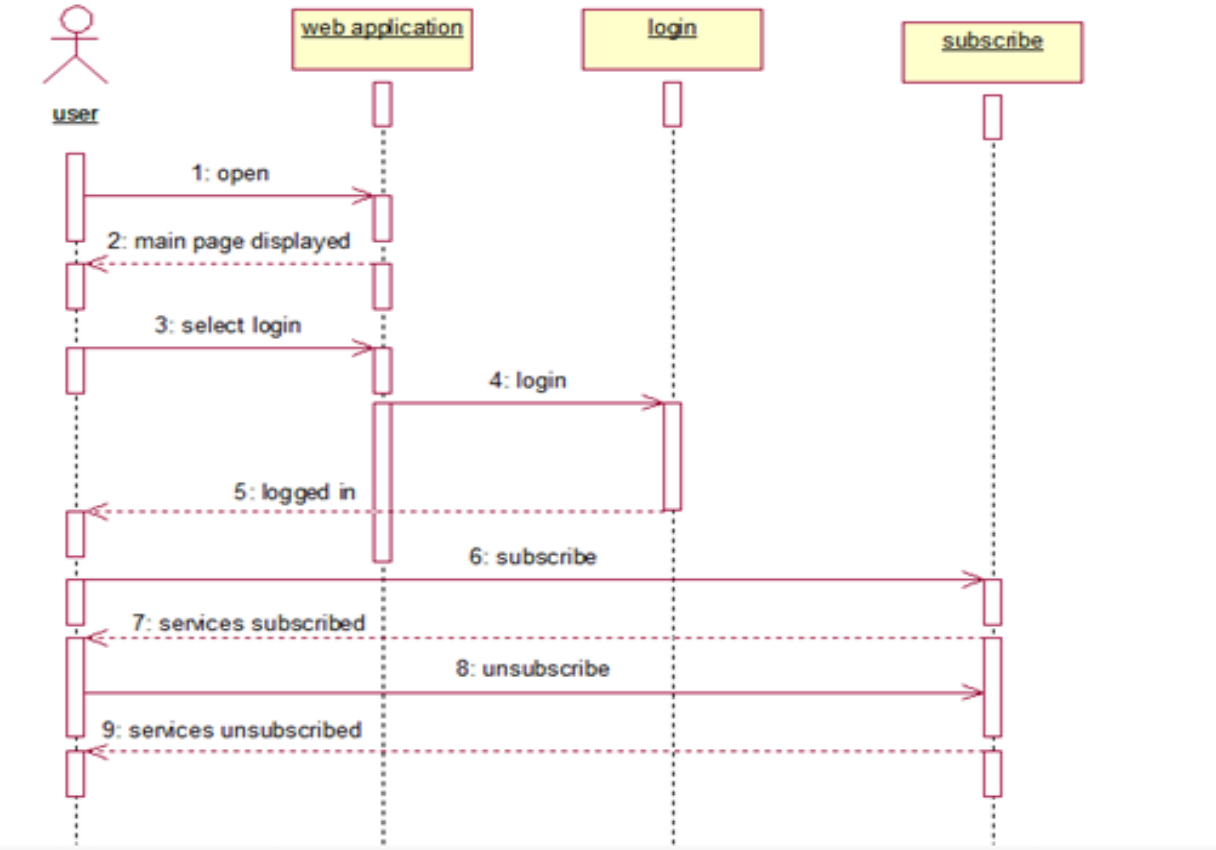


Fig 2.1.4.4: Sequence Diagram

8.3.5 Logical View (State diagram)

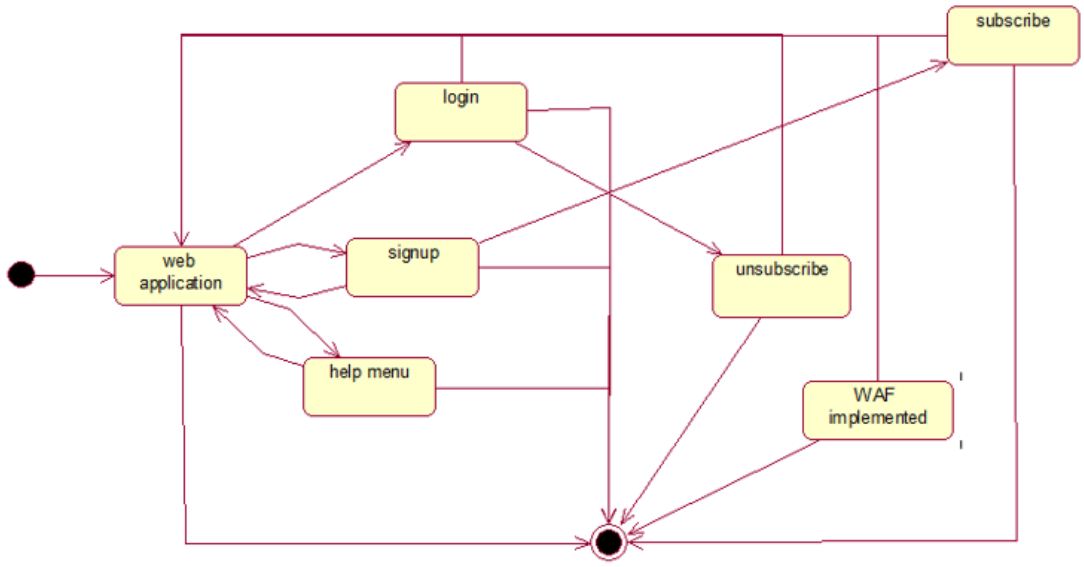


Fig 2.1.5.1: State Diagram

8.3.6 Dynamic View (Activity Diagram)

In activity diagram, the dynamic view of the system is shown. All the activities are shown concurrently with their respective start and end states

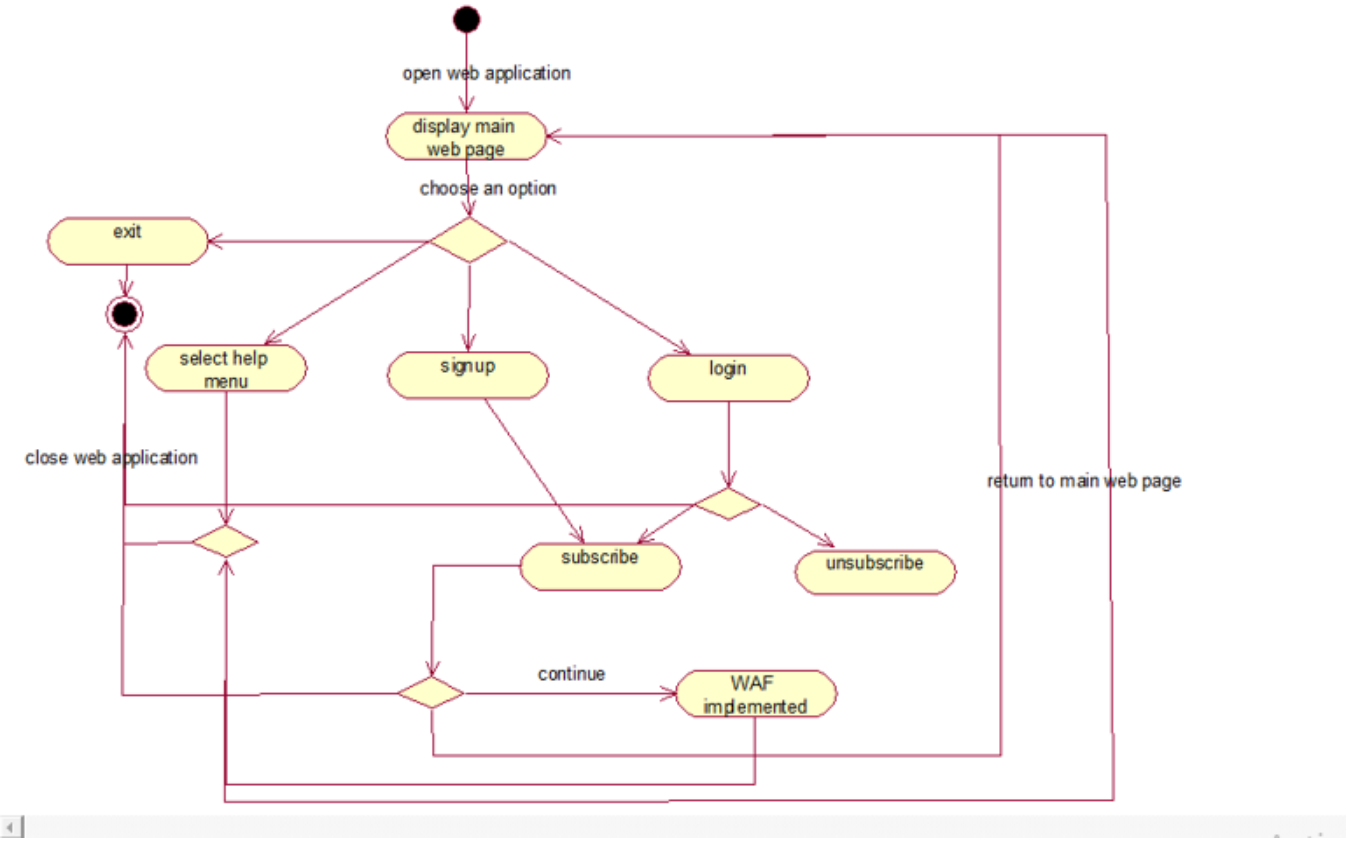


Fig 2.1.6.1: Activity Diagram

8.3.7 IMPLEMENTATION VIEW (CLASS DIAGRAM)

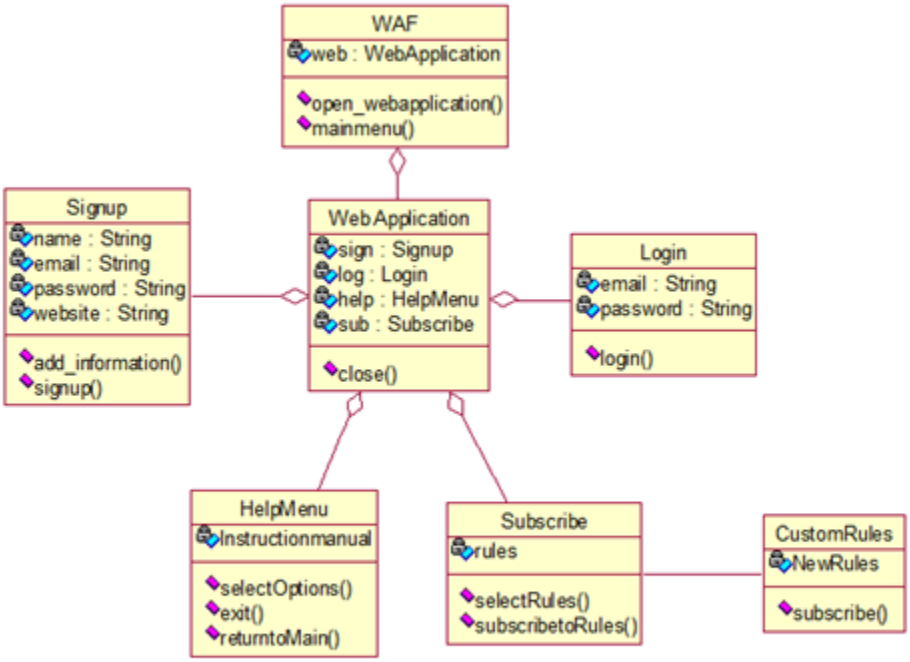


Fig 2.1.7.1: Class Diagram

Name	**Description**
WAF	It is the main class of system. It creates the object of WebApplication. It has two functions open_webapplication and main menu.
Web application	This holds the objects of Sign up, login ,helpmenu and subscribe.All of these are initial or main steps toward using the main functionality of WAF.

	User can exit as well by closing the application.
Sign up	This class provides the functionality for first time user to register their details. It has attributes like name ,string,password and website. User will provide valid inputs to get successfully signed up
Login	Assuming that User has already signed up, this class lets him to enter the inputs to email and password attribute and he gets the access to all services. It has the Login function for doing this.
Help Menu;	Holds the instruction manual and further more the functionality of closing the help menu and return to main page . It helps you choose among different options
Subscribe	This class allow the user to select the wanted rules/services and subscribe to them.

8.3.8 Structure Chart

This chart shows the breakdown of the application to its lowest manageable levels. It shows the modules and their corresponding functions which this application will implement. This chart basically shows the structure breakdown of the application starting from main modules to specific functions.

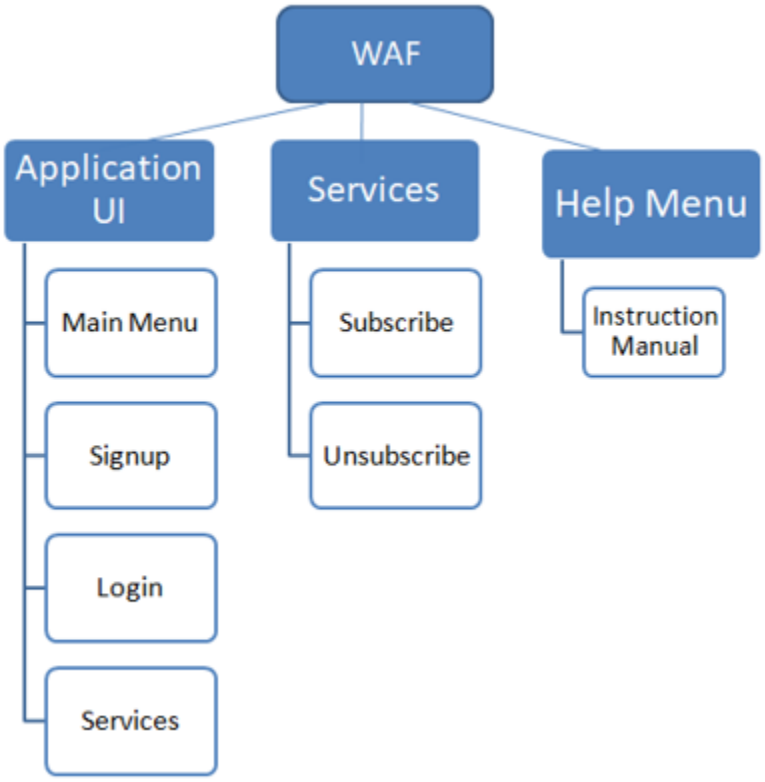


Fig 2.1.8.1: Structure Chart

DETAILED DESCRIPTION OF COMPONENTS

Identification	**Name: User interface**
Type	Component
Purpose	The purpose of this component is to display to the user with all information/result
Function	Interacts with user to get the required inputs. User can navigate through different options.
Subordinates	No subcomponents. **Functional Requirements** Requirement 1:The user will be able to perform all operations by interacting with options presented.
Dependencies	The subcomponent UI RENDERING of the Application will get its input from this module. This input will be further processed renderer.
Interfaces	
Resources	
Processing	
Data	

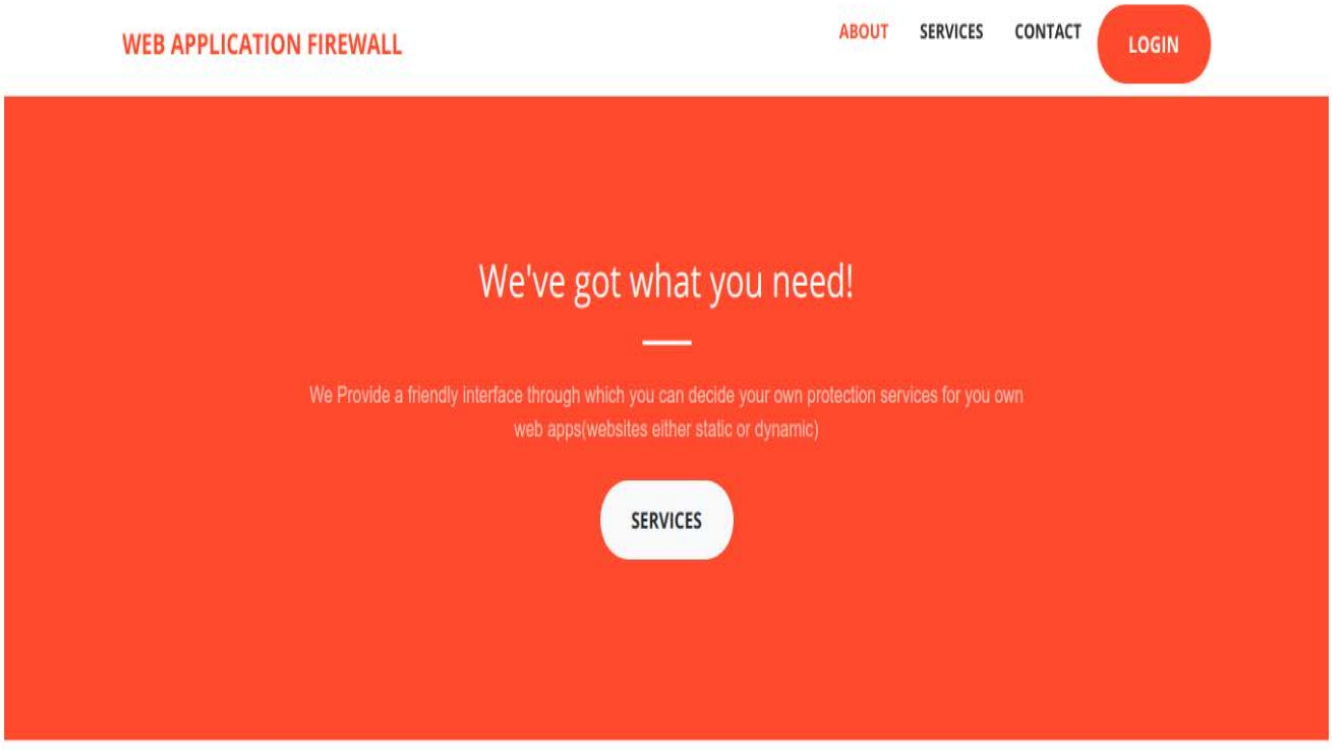
Identification	**Name: User**
Type	Component
Purpose	• The purpose of this component is to get the user inputs from user. • User will choose among different menu options.
Function	User decides the inputs or what services to use.
Subordinates	No subcomponents. **Functional Requirements** Requirement 1:The user will be able to perform all operations by interacting with options presented.

Dependencies	The sub component User input of the Application will get its input from this module. This input will be further processed to see what user is asking for.
Interfaces	Basic web page interface or the user interface
Resources	Laptop and working internet connection
Processing	The processing required for this component is receiving the user's input and giving this input to the input event module of the Application component
Data	User input

Identification	Name: APPLICATION
Type	Component
Purpose	• The purpose of this component is to get the user inputs from user. • To do the required implementation according to user input • To provide WAF services • To render the HTML page.
Function	The whole processing functionality of entertaining the user's requested services.
Subordinates	**Subcomponents:** • User input • WAF services • UI rendering **Functional Requirements** Requirement 1: It should be able to detect user input correctly. Requirement 2:It should be able to map the services/rules accordingly Requirement 3: Finally it should send data to renderer to render the content or results

Dependencies	**Components using this component:** User input UI rendering
Interfaces	The external interfaces interacting with this component are User using laptop and user interface
Resources	Internet usage
Processing	The processing is receiving the user input to the input event module of the Application component and then provide waf service and let user know about it
Data	Float values, integer values, strings

INTERFACE

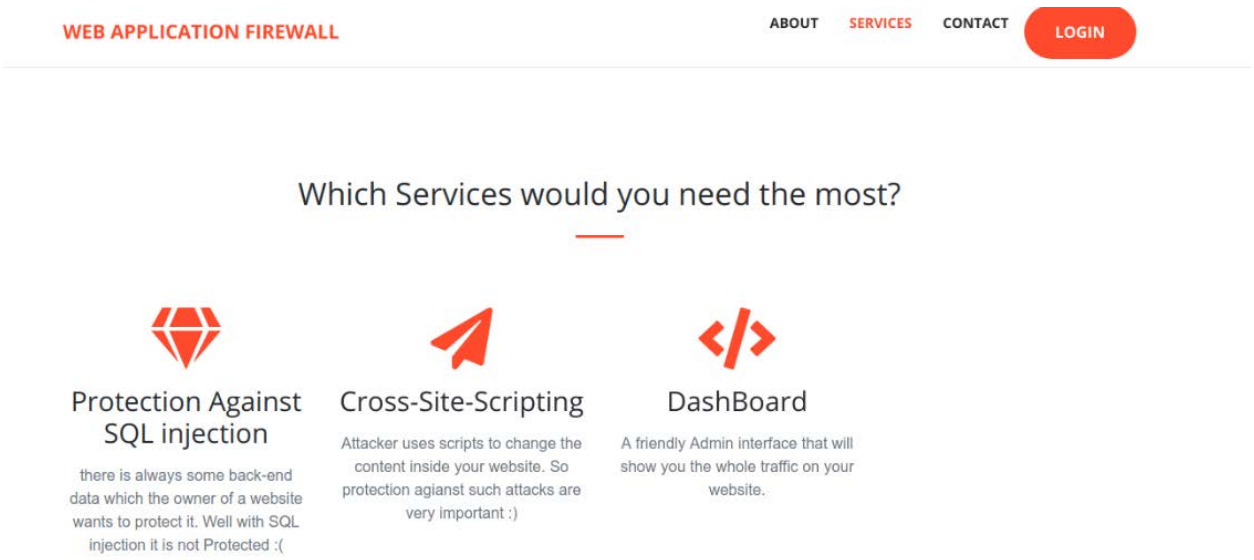


Let's Get In Touch!

bla bla bla bla bl bla bla bla bla bla bla

any number

OurOfficial@mail.com



Reuse and Relationships

WAF is not based on any previous systems neither it's an extension of any other applications at any level. But it can be evolved into a bigger and more complex system with more features and functionality. Developers can also reuse some of the modules of the system.

PSEUDOCODE

//To block excessive traffic:

Begin

if network(bandwidth > normal bandwidth){

Attacks happens --->goto 1: // it could be server down for some time or just restart the server

}

else{

Allow access to web application:

}

End

//Checking if the User(ip) is legitimate

```
Begin
   Set A[] as attack packets; // attacker ips
   Set L[] as legitimate packets; // legitimate ips

   if (TTl values are different for a specific ip){

      Add that ip to A[];
Display_errorPage();
      Deny that ip;
   }
   else if (the no.of packets in a given timeinterval is more for an ip){

      Add that ip to A[];
Display_errorPage();
      Deny that ip;
   }
   else if (there are varying ports number for an ip){

      Add that ip to A[];
Display_errorPage();
      Deny that ip;
   }
   else {
      Add that ip to L[];
      Allow that packet with its ip;
Display_Home(); // Display our web app
   }

End
```

// User entered

```
Begin

   connection = True;

while(connection == true){

      if (UserClick == HelpPage){
```

```
goto -->HelpPage // HelpPage.php OR HelpPage.ejs OR HelpPage.html
      }

      else if (UserClick == Services){

goto --> Services

         if (UserClick == ShowServices){

Display_Services();
         }

         else if(UserClick == UploadSite){

Display_Admin_Dashboard();
         }
      }

      else if (UserClick == HomePage){

goto -->Display_Home();
      }

      // User attempts for attacks

      else if (attempt for sqlinjection){

Display_errorPage();
         Add that specific ip to A[];
         Deny that ip always; // could be forever or for specific time
      }

      else if (attempt for XXS){

Display_errorPage();
         Add that specific ip to A[];
         Deny that ip always; // could be forever or for specific time
      }
 //All top 10 flaws of OWASP
      .
      .
      .
      .
      .
// if User wants to close the App
```

```
        else if (UserClick == CloseButton){

          connection == false;
CloseSite();
       }
   }


End
```

TESTING AND EVALUATION

13.1 Introduction

This test plan document describes the appropriate strategies, process and methodologies used to plan, execute and manage testing of the “Web Application Firewall". The test plan will ensure that project meets the customer requirements at an accredited level.

Manual Testing will be followed which includes testing a software manually, i.e., without using any automated tool or any script. In this type, the tester takes over the role of an end-user and tests the software to identify any unexpected behavior or bug. Each Unit will be tested separately and then will be integrated with other units; therefore, Unit Testing and Integration testing will be followed. For each unit, Black box Testing is done and for combined units Acceptance Testing is done.

The test scope includes the Testing of all functional, application performance and use cases requirements listed in the *requirement document.*

Software testing, depending on the testing method employed, can be implemented at any time in the development process. However, most of the test effort occurs after the requirements have been defined and the coding process has been completed.

This document includes the plan, scope, approach and procedure of the testing of software. The pass/fail criteria of the test items are also defined. The document tracks the necessary

information required to effectively define the approach to be used in the testing of the product.

13.2 Test Items

The test items selected for testing include the following

- ☐ Performance
- ☐ Interface
- ☐ User control.

13.3 Features to be tested

The features of our project include the functionality mentioned in our design document. Following features are to be tested keeping in view the test items and system features aforementioned.

1.Web App and Reverse Proxy Server properly installed and Working.

2.Different attacks are generated and finally tested to see if WAF encounters them or not.

3.Major attacks tested were :

- Malicious HTTP traffic
- Database injection
- DOS attack
- Cross Site Scripting

13.4 Approach

Functional Testing will focus on each use case that is included in the version currently being worked on. Testing will mainly consist of execution of test cases written to address the gap identified. It will focus on inputs, outputs and system changes due to the actions.

13.5 Item Pass/Fail Criteria

Details of the test cases are specified in section Test Deliverables. Following the principles outlined below, a test item would be judged as pass or fail.

- Preconditions are met

- Inputs are carried out as specified
- The result works as what specified in output => Pass
- The system doesn't work or not the same as output specification => Fail.

13.6 Suspension Criteria and Resumption Requirements

Testing procedure will be suspended whenever a defect is found that restricts further testing. A corrective measure will be applied depending upon the criticality of the defect and testing will be resumed.

Efforts have been made to remove all and every chance of failure but there are certain unpredictable factors such as network issues, corrupt input data, or system failure that may lead to some issues. Error handling is applied more deeply to cover all these issues but unforeseen circumstances may happen.

13.7 Test Deliverables

13.7.1 Testing tasks

- ☐ Develop Test Cases.
- ☐ Execute tests based on the test cases developed.
- ☐ Report defects during tests if any.
- ☐ Manage the changes made after testing.

13.7.2 Test cases

Following are the Test Cases:

Test Case ID	**TC 1**
Test Case Description	Working Enviroment
Testing Technique	White Box Testing, Black Box Testing.
Preconditions	1. Working Web App. 2. Deployed reverse proxy server 3. Database connected with web app
Steps	
Expected output	Http Traffic of Web App passing through reverse proxy server.
Actual Output	Traffic was successfully redirected .
Status	PASS

HTTP METHOD

Test Case ID	**TC 2**
Test Case Description	Patch Method for Web App
Testing Technique	White Box Testing

Preconditions	1.Rest Client API 2.Web Application 3.Reverse Proxy Server
Steps	1.Send Patch Request.
Expected output	Allowing the request to pass through
Actual Output	The request was successfully implemented on Web App.
Status	PASS

Test Case ID	**TC 3**
Test Case Description	Patch Method for Reverse Proxy Server

Testing Technique	White Box Testing
Preconditions	1.Rest Client API 2.Web Application 3.Reverse Proxy Server
Steps	1.Send Patch Request.
Expected output	Not allowing the request to pass through.
Actual Output	The request was successfully forbidden on Reverse Proxy Server.
Status	PASS

Test Case ID	**TC 4**
Test Case Description	Cross Site Scripting on Web App.
Testing Technique	Black Box testing
Preconditions	1.Rest Client API 2.Web Application 3.Reverse Proxy Server
Steps	1.Sending a script in HTTP request.
Expected output	Allowing the request to pass through.
Actual Output	The request was successfully implemented on Web App.
Status	PASS

Test Case ID	**TC 5**

Test Case Description	Cross Site Scripting on Reverse Proxy Server.
Testing Technique	Black Box testing
Preconditions	1.Rest Client API 2.Web Application 3.Reverse Proxy Server
Steps	1.Sending a script in HTTP request.
Expected output	Not Allowing the request to pass through.
Actual Output	The request was forbidden at port 8080 Reverse Proxy server.
Status	PASS

Test Case ID	**TC 6**
Test Case Description	Database injection on Web App.
Testing Technique	Black Box testing
Preconditions	1.Rest Client API 2.Web Application 3.Reverse Proxy Server

Steps	1.Sending queries for getting data from MongoDB
Expected output	Allowing the request to pass through.
Actual Output	The request was successfully implemented on Web App .
Status	PASS

Test Case ID	**TC 7**
Test Case Description	Database injection on Reverse Proxy Server.
Testing Technique	Black Box testing
Preconditions	1.Rest Client API 2.Web Application 3.Reverse Proxy Server
Steps	1.Sending queries for getting data from MongoDB
Expected output	Not Allowing the request to pass through.
Actual Output	The request was forbidden on reverse Proxy. .

Status	PASS

Test Case ID	**TC 8**
Test Case Description	DOS Attack.
Testing Technique	Black Box testing
Preconditions	1.Rest Client API 2.Web Application 3.Reverse Proxy Server 4.Shell Script for multiple request 5.Allowing 5 requests per second.
Steps	1.Generate 10 requests per second on both servers.
Expected output	Allow request upto specified limit rest are blocked

Actual Output	6 requests were allowed per second.
Status	PASS

Responsibilities, Staffing and Training Needs

14.1 Responsibilities

All developers of the project are responsible for the completion of all components testing and integration testing tasks.

14.2 Staffing and Training Needs:

Basics knowledge of testing strategies and techniques is needed for the testing of the project. Techniques such as Black Box testing, integration testing should be known to developers. All the developers will be testing each other's work and will be actively participating in the development and testing of the project simultaneously.

Schedule

9.1 Important Dates

- o Unit Testing and integration testing will be finished by the end of April, 2019 as will the Development process
- p Acceptance Testing will be performed right after the Development process completes.

Risks and contingencies

16.1 Kernel Risk

By using containers, greater abstraction away from hardware also brings with it the risk of less transparency and control. When something breaks in a system running hundreds of containers, we have to hope that the failure bubbles up somewhere we can detect. If the problem is with the host operating system or underlying hardware, it might be hard to determine.

An outage that could have been resolved in 20 minutes using VMs may take hours or days to resolve with containers if you do not have the right instrumentation.

16.2 Operational Risks

Operational risks will be eliminated by Scheduling daily meetings and regular deadlines to meet the goals of the project as well as provide proper communication within the group.

16.3 Technical risks

Technical risks will be eliminated by keeping the once defined requirements constant.

16.4 Programmatic Risks

In case of a programmatic risk the scope of the project will be limited in order to stay inside the constraints of the project.