Applied Machine Learning Predictive Analytics to SQL Injection Attack Detection and Prevention

[1,*]T.P.Latchoumi, [2]Manoj Sahit Reddy, [3]K.Balamurugan

[1]Assistant Professor, Department of Computer Science and Engineering, VFSTR(Deemed to be University), Andhra Pradesh, India.

[2]Student, Department of Computer Science and Engineering, IIITDharwad, Dharwad, India.

[3]Professor, Department of Mechanical Engineering, VFSTR (Deemed to be University), Andhra Pradesh, India.

*Corresponding Author: tplatchoumi@gmail.com, +918754830690

Abstract

These days the world is very much dependent on web applications. Hence providing security to these applications is of great importance. Information is maintained in the backend databases in the majority of applications. Among the vulnerabilities is the Structured Query Language Injection Attack (SQLIA). There are several applications to retrieve session/HTTP cookies nowadays. There is quite a range of techniques used to stop these attacks.The proposed work discusses the flaws in a few of these techniques that handle these attacks and implement an efficient hashing technique to prevent this technique. To overcome the above-mentioned attacks, the machine learning concept with the Support Vector Machine (SVM) algorithm was introduced. It is used to detect and prevent SQL injection. In this technique, the SVM algorithm will be trained with all possible malicious expressions and then generate the model. Whenever a user gives any new query then SVM will be applied to that model to predict whether a given query contains any malicious expressions or not. If the user invents the new technique then also SVM can detect that malicious expression by matching with a minimum number of syntax.

Keywords: Support Vector Machine, Web application, Structured Query Language Injection Attack, Simple Object Access Protocol.

I. Introduction

In recent years, much research has been conducted not only at educational institutions but also to prevent needle attacks. Below are some of the preventive measures recommended by researchers [1]. Vulnerabilities in the Web can compromise personal information and other valuable resources. When a user tries to submit a request to a web server, he does this using Hypertext Markup Language (HTML) forms, Uniform Resource Positions (URLs), or other fields where data can be entered. The unfiltered form allows users to use SQL injection. This is because the form data submitted to the database is processed and processed without checking. SQL Hall of Fame Search [2] reports the latest trends in SQLIA data triggers. The ability to protect your backend database from SQLIA in the big data era is a subject-based issue. SQL injection is a type of attack that enables an attacker to access or alter data by inserting query language code composed of a web form login form. A SQL injection vulnerability could allow an attacker to send commands directly to the base database of the web application, removing privacy and functionality[3].

SVM is a set of methods of supervisory learning based on the theory of statistical learning and used for classification and regression tasks. As a classification system, SVM is a global model of classification that generates non-overlapping parts and uses all characteristics in general[4]. The partitions are shared in one pass, creating linear and linear partitions. SVM is based on maximum margin and linear discriminant, similar to a probabilistic approach, but without considering the dependencies between qualities. The basic idea of the SVM classifier is to choose this approach, the max bridge plane [5].

A type of SQL detection system built into the cloud environment that protects web applications in cloud deployments and provides dynamic analysis and input filtering. First, this method gets the SQL keywords through a lexical order analysis of the SQL statement. Then analyze the syntax of the SQL statement to create a rule expression. Finally, a miniature view based on the attack detection model defined by the SQL syntax arrangement was passed [6]. To prevent SQL injection attacks, many techniques such as content filtering, attack testing, and defensive coding are used to identify and prevent a subset of SQL filtering vulnerabilities. SQL checks will recognize this as a malicious query. SQL and SQL are analytical methods to prevent injection. An intermediate proxy is used to translate SQL into the default language. Queries paid by the attacker will be stopped by the database parser [7].

Python is a programming language at a high level that is easy to read and execute. For commercial applications, it is open-source and free. It is called a Python, Ruby, or Perl scripting language and is mostly used to build web applications and interactive web content. It is possible to evaluate the scripts (.py files) immediately written in Python [8,9]. Save the file as a compiled program (.py file) that is used in other Python programs as a programming block that can be expressed. This document contributes to a representative dataset that includes the ability to teach a predictable control learning model through the SQLIA SVM (Support Vector Machine) algorithm to prevent malicious web requests from accessing the target backend database [10]. It also offers, on large data networks, an SQLIA discovery and blockchain environment.

II. Proposed system

The proposed detection model may report vulnerabilities in web applications. As a result, this model can reduce the likelihood that SQLIA will launch in your web application. Machine learning with SVM algorithms is used to prevent SQLIA runtime monitoring. The solution behind this technique is to detect and prevent SQLIA outages when the home page of each application is transferred to a test page as shown in Figure 1.

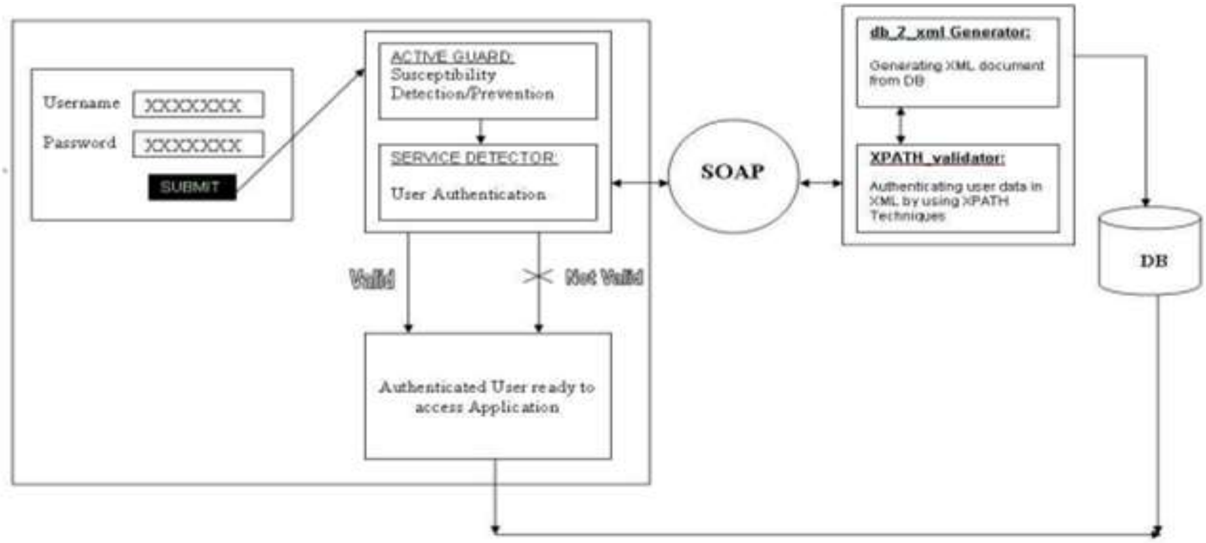

Username XXXXXXXX
Password XXXXXXXX
SUBMIT
ACTIVE GUARD:
Susceptibility Detection/Prevention
SERVICE DETECTOR:
User Authentication
Valid
Not Valid
Authenticated User ready to access Application
SOAP
db_2_xml Generator:
Generating XML document from DB
XPATH_validator:
Authenticating user data in XML by using XPATH Techniques
DB

Figure 1: Proposed architecture

Straight lines and straight-line sections are created by sharing SVM sections in one pass. SVM is based on maximum margin and linear discriminant, similar to a probabilistic approach, but without considering the dependencies between qualities. The basic idea of the SVM classifier is to use this approach, i.e. select the bridge plane with the largest edges, as shown in Figure 2.

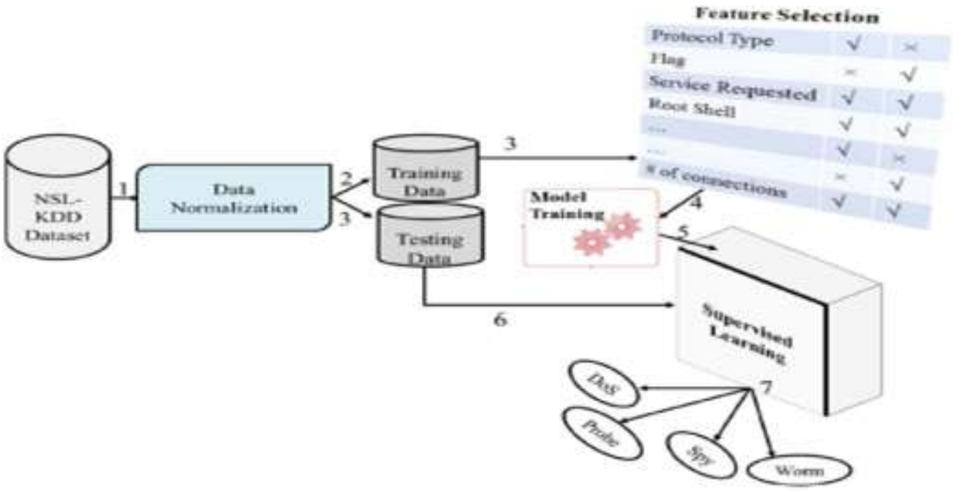
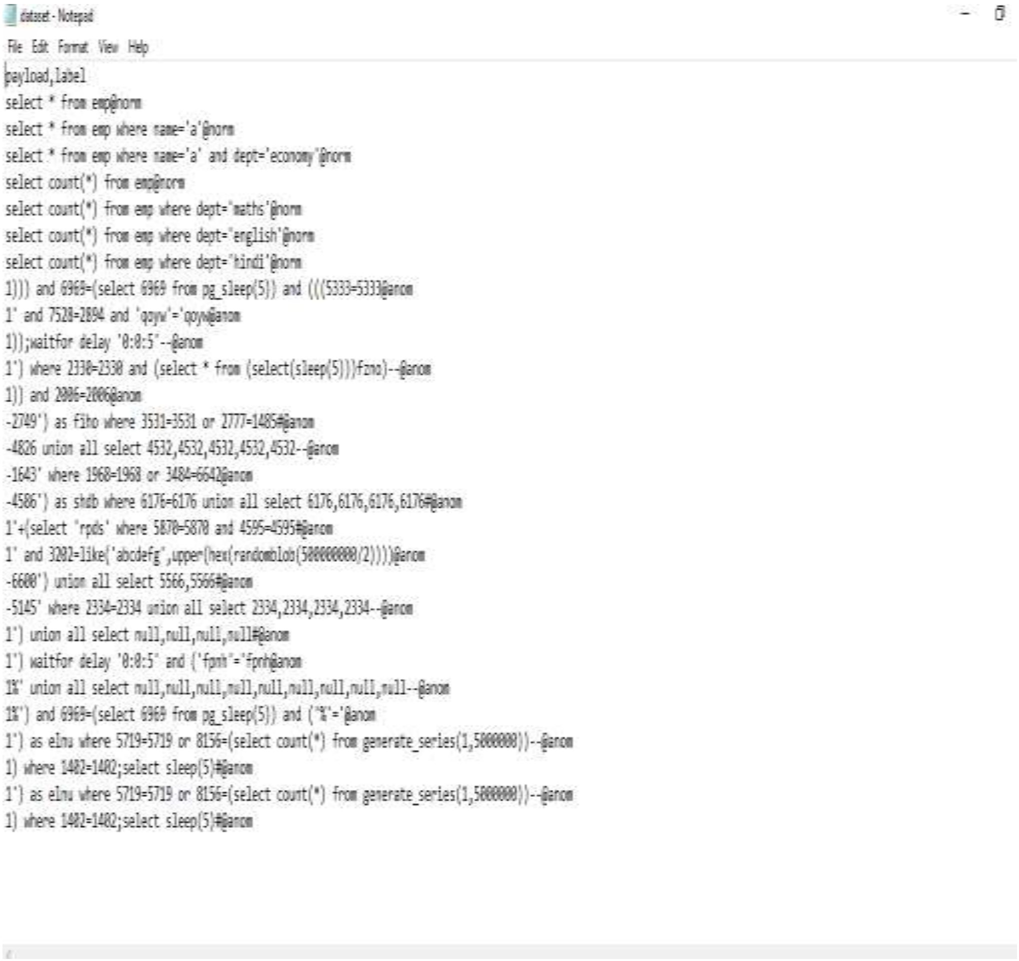


Figure 2: Proposed SVM classifier architecture

Training classifiers used in designing predictive analytics web applications using the level of training data. Attack signatures take the form of SQLIA tokens and SQLIA positive symbols at injection points, but a legitimate web request will take the form of data that the application expects. The vector of training data defined as a matrix or dictionary keyword property (SQLIA negative) and a SQL token (SQLIA positive). The SVM will identify this malicious expression when a user invites a new technique by matching it with a minimum amount of syntax

A. Upload Datasets

Datasets used to predict and accuracy for a model which will be built for solving real world problem by viewing, training, testing the data. These Datasets will give past and updated data of models which also help in building predictive Artificial Intelligence projects shown in Figure 3.



```
payload,label
select * from emp@norm
select * from emp where name='a'@norm
select * from emp where name='a' and dept='economy'@norm
select count(*) from emp@norm
select count(*) from emp where dept='maths'@norm
select count(*) from emp where dept='english'@norm
select count(*) from emp where dept='hindi'@norm
1))) and 6969=(select 6969 from pg_sleep(5)) and (((5333=5333@anom
1' and 7528=2894 and 'qoyw'='qoyw@anom
1));waitfor delay '0:0:5'--@anom
1') where 2330=2330 and (select * from (select(sleep(5)))fzno)--@anom
1)) and 2006=2006@anom
-2749') as fiho where 3531=3531 or 2777=1485#@anom
-4826 union all select 4532,4532,4532,4532,4532--@anom
-1643' where 1968=1968 or 3484=6642@anom
-4586') as shdb where 6176=6176 union all select 6176,6176,6176,6176#@anom
1'+(select 'rpds' where 5870=5870 and 4595=4595#@anom
1' and 3202=like('abcdefg',upper(hex(randomblob(500000000/2))))@anom
-6600') union all select 5566,5566#@anom
-5145' where 2334=2334 union all select 2334,2334,2334,2334--@anom
1') union all select null,null,null,null#@anom
1') waitfor delay '0:0:5' and ('fpnh'='fpnh@anom
1%' union all select null,null,null,null,null,null,null,null,null--@anom
1%') and 6969=(select 6969 from pg_sleep(5)) and ('%'='@anom
1') as elnu where 5719=5719 or 8156=(select count(*) from generate_series(1,5000000))--@anom
1) where 1402=1402;select sleep(5)#@anom
1') as elnu where 5719=5719 or 8156=(select count(*) from generate_series(1,5000000))--@anom
1) where 1402=1402;select sleep(5)#@anom
```

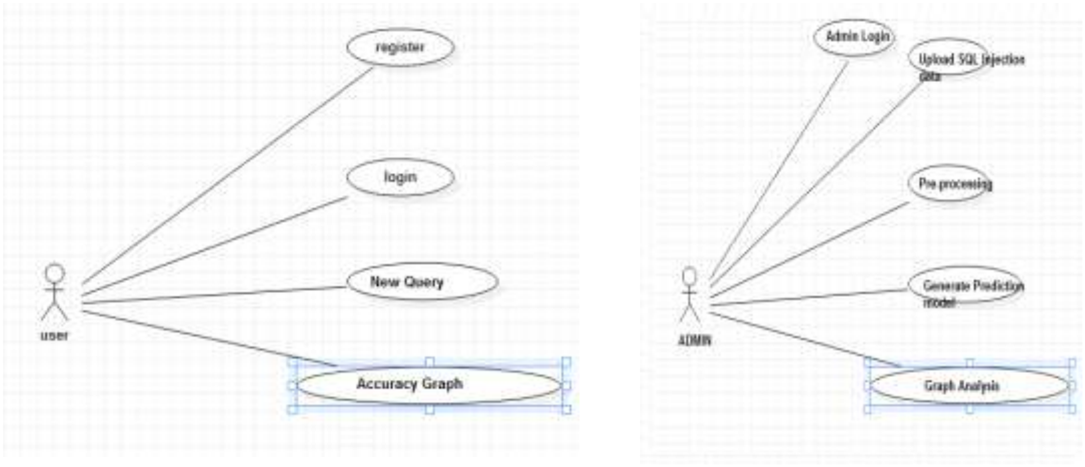
Figure 3: Upload datasets using SQL

2.2 UML Diagrams

The skin will appear on the case diagram after finishing the first task as shown in Figure 4 (a) and (b).

The object of the use scenario can, in short, be determined as follows:

- Used to assemble a system's prerequisites.
- Used to create the appearance of the device.
- System Recognition of external and internal factors influencing the system.
- The relationship between requirements is indicated.



4 (a) Use case diagram of a user

4 (b) Use case diagram of Admin

Figure 4: Use case diagram of user and admin role

Sequence outlines are utilized to catch the dynamic nature yet from an alternate point. The motivation behind the Sequence diagram is to catch the dynamic conduct just as the message stream of a framework and portray the auxiliary association and communication among objects as appeared in Figure 5.

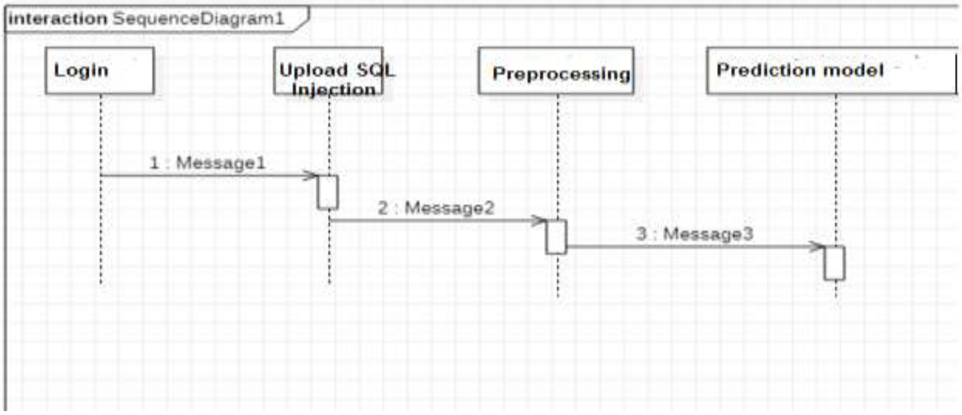


Figure 5: Sequence diagram of the interaction between user and admin

A function can be defined as a function of a system. Consequently, the control flow is transferred from one function to another. This stream may be continuous, branched, or simultaneous. Functional diagrams cover all types of flow control using various components, such as a fork and a hitch. This function is a specific function of the system.

Message flow from one activity to another is not displayed. Functional diagrams are sometimes considered block diagrams. Although the diagrams look like a block diagram, it is not. As shown in Figure 6, it shows different flows in parallel, branched, at the same time and simply.

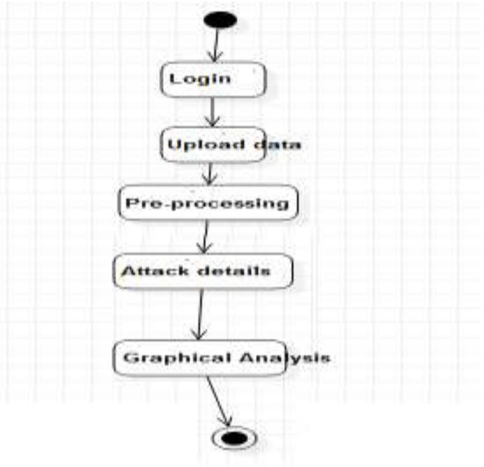


Figure 6: Activity diagram of attack analysis

IV Result Analysis

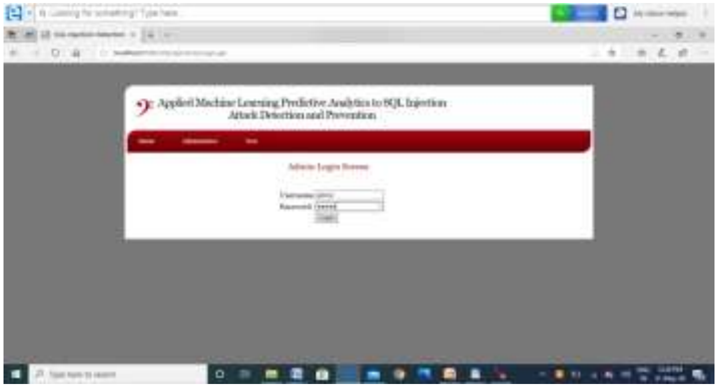
Start the Tomcat server and type localhost: 9595 in the browser link will provide the server as shown in Figure 7. Initiate the SQL injection to prevent the query from attack as shown in Figure 8.

Figure 7: Activating Tomcat server through localhost

Applied Machine Learning Predictive Analytics to SQL Injection Attack Detection and Prevention

Figure 8: Initiate the SQL Injection

Admin and user have a login page to register and login as shown in Figure 9. And after login into the system, the admin can securely upload datasets as shown in Figure 10.



Applied Machine Learning Predictive Analytics to SQL Injection Attack Detection and Prevention

Admin Login Screen

Username: admin

Password: *****

Figure 9: Admin Login screen

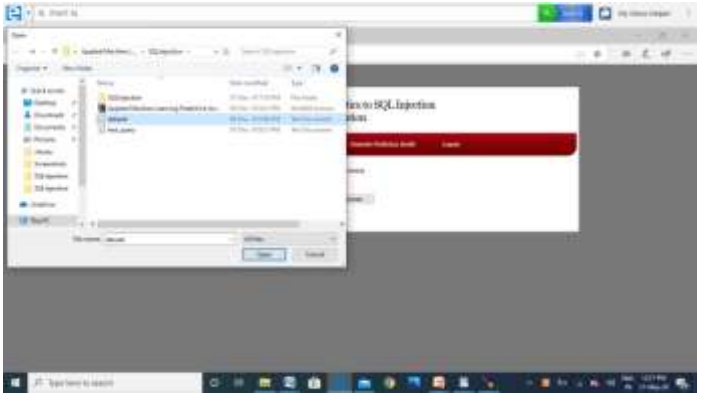
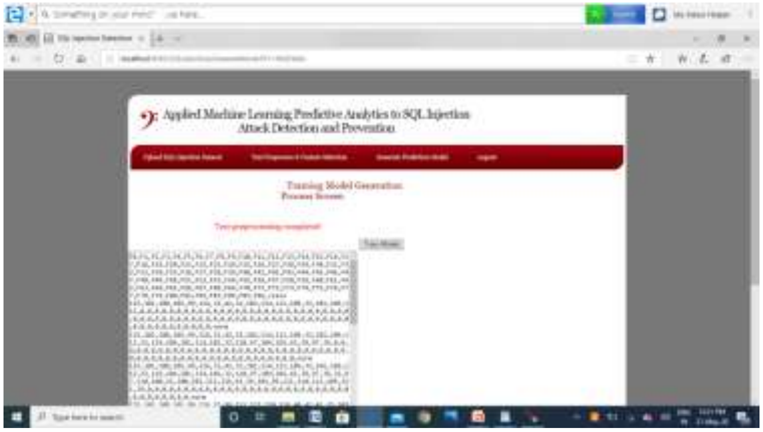


Figure 10: Admin upload the data sets securely

Admin can train the dataset based on machine learning predictive analysis using SQL Injection as shown in Figure 11. And finally, the validation screen to check the attack detection is shown in Figure 12.

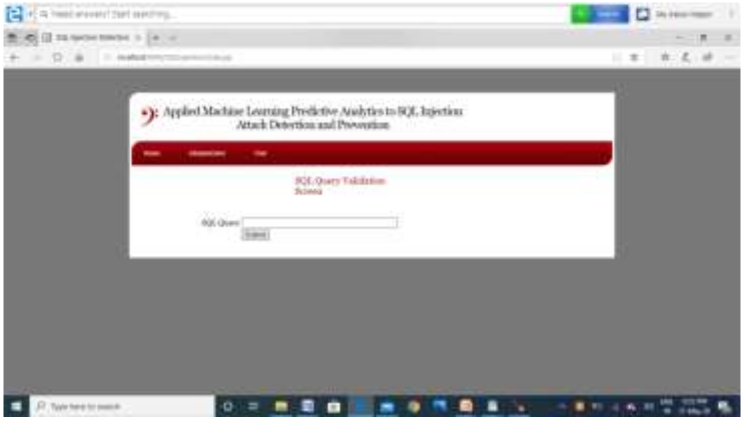


Applied Machine Learning Predictive Analytics to SQL Injection Attack Detection and Prevention

Training Model Generation Process Screen

Text preprocessing completed!

Figure 11: Training model generation process screen



Applied Machine Learning Predictive Analytics to SQL Injection Attack Detection and Prevention

SQL Query Validation Screen

SQL Query

Figure 12: SQL query validation check screen

Test Cases

Test Case 1

Test Case Name	Checking an empty login field.
Description	On the login screen, if the Username and Password fields are blank.
Output	When entering the system, a warning window is not displayed asking for a username and password.

Test Case 2

Test Case Name	Check input fields
Description	The unique login and password are set by the administrator. If

	you enter an invalid username or password.
Output	Alert Box Login does not show username or password as invalid.

The current android application is developed using XML, Java, SQL with Firebase connectivity. It can be used by every individual who is in a need of fulfilling their household services. At the time of submitting this application was capable of doing the following:

- Displaying the home screen with different fragments.
- Authentication of the user by using a login screen using Firebase.
- Home screen to display based on user or service provider.
- After the successful login of the user, they can choose the service and book a slot of their particular service provider from the displayed list.
- Add, update, view, and delete the user details.
- After the successful login of the service provider, they can view all the bookings that are booked by the users and can attend them one by one.
- The service provider can also set his preferences to not available if he's too busy or many users had already booked him.
- The service provider can change their particular radius of location for servicing.
- He can set up to 10 km radius.
- Logout and end the session.
- Understanding the connections of the SQLite Database is a tricky part and confusing when dealing with multiple tables within a database.
- Making exact orientation API design levels was a difficult task as there are many types of devices like desktop, tablet, mobile with varying screen sizes, and resolutions.
- Implementing synchronization with Firebase was a challenging task.

V. Conclusion and Future enhancement

In this study, we demonstrated predictive analysis with good results that can be evaluated empirically in the above mixing matrix and ROC chart to detect and prevent SQLIA in a big data environment. When standardizing this thesis compared with the current thesis, the methodology proposed here works in an environment containing a large amount of data, which is not enough in current studies, according to the knowledge of SQLIA. Future research will include the use of a multiclass classifier to define and group various types of SQLIA, as intended.

References

[1] Halfond, W. G., Orso, A., &Manolios, P. (2006, November). Using positive tainting and syntax-aware evaluation to counter SQL injection attacks. In Proceedings of the 14th ACM SIGSOFT international symposium on Foundations of software engineering (pp. 175-185).

[2] Li, Q., Wang, F., Wang, J., & Li, W. (2019). LSTM-based SQL injection detection method for an intelligent transportation system. IEEE Transactions on Vehicular Technology, 68(5), 4182-4191.

[3] Tang, P., Qiu, W., Huang, Z., Lian, H., & Liu, G. (2020). Detection of SQL injection based on artificial neural networks. Knowledge-Based Systems, 190, 105528.

[4] Ezhilarasi, T. P., Dilip, G., Latchoumi, T. P., &Balamurugan, K. (2020). UIP—A Smart Web Application to Manage Network Environments. In Proceedings of the Third International Conference on Computational Intelligence and Informatics (pp. 97-108). Springer, Singapore.

[5] Li, Q., Li, W., Wang, J., & Cheng, M. (2019). A SQL Injection Detection Method Based on Adaptive Deep Forest. IEEE Access, 7, 145385-145394.

[6] Zhang, K. (2019, November). A machine learning-based approach to identify SQL injection vulnerabilities. In 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE) (pp. 1286-1288). IEEE.

[7] Volkova, M., Chmelar, P., &Sobotka, L. (2019, June). Machine Learning Blunts the Needle of Advanced SQL Injections. In MENDEL (Vol. 25, No. 1, pp. 23-30).

[8] Demetrio, L., Valenza, A., Costa, G., &Lagorio, G. (2020, March). WAF-A-MoLE: evading web application firewalls through adversarial machine learning. In Proceedings of the 35th Annual ACM Symposium on Applied Computing (pp. 1745-1752).

[9] Loganathan, J., Janakiraman, S., Latchoumi, T. P., &Shanthoshini, B. (2017). Dynamic Virtual Server for Optimized Web Service Interaction. International Journal of Pure and Applied Mathematics, 117(19), 371-377.

[10] Ahmim, A., Ferrag, M. A., Maglaras, L., Derdour, M., &Janicke, H. (2020). A Detailed Analysis of Using Supervised Machine Learning for Intrusion Detection. In Strategic Innovative Marketing and Tourism (pp. 629-639). Springer, Cham.