GMR Institute of Technology
An Autonomous Institute Affiliated to JNTUK, Kakinada

GMRIT
Training Tomorrow's Engineers Today

To Detect and Prevent Vulnerabilities in Microservices and Application Programming Interface

A project report submitted in partial fulfillment of the

requirementfor the award of degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted by

T Anusha (19341A05G7)	**G Naga Manmitha (20345A0513)**
Sharon Kota (19341A05F7)	**S Laxman Rao (19341A05F2)**
N Ramesh Kumar (20345A0514)	**P Ganesh (20345A0518)**

Under the esteemed guidance of

Ms. Santhoshini Sahu

Assistant Professor, Dept. of CSE

GMR Institute of Technology

An Autonomous Institute Affiliated to JNTUK, Kakinada

(Accredited by NBA, NAAC with 'A' Grade & ISO 9001:2015 Certified Institution)

GMR Nagar, Rajam-532127,
Andhra Pradesh, India
November 2022

GMR Institute of Technology
An Autonomous Institute Affiliated to JNTUK, Kakinada

GMRIT
Training Tomorrow's Engineers Today

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the thesis entitled **“To detect and prevent Vulnerabilities in Microservices and Application Programming Interface”** submitted by **T Anusha (19341A05G7), G Naga Manmitha (20345A0513), Sharon Kota (19341A05F7), S Laxman Rao (19341A05F2), N Ramesh Kumar (20345A0514), P Ganesh (20345A0518)** has been carried out in partial fulfillment of the requirement for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **GMRIT, Rajam** affiliated to **JNTUK, Kakinada** is a record of bonafide work carried out by them under my guidance & supervision. The results embodied in this report have not been submitted to any other University or Institute for the award of any degree.

Signature of Supervisor
Ms. Santhoshini Sahu
Assistant Professor
Department of CSE
GMRIT, Rajam.

Signature of HOD
Dr. A. V. Ramana
Professor & Head
Department of CSE
GMRIT, Rajam.

The report is submitted for the viva-voce examination held on ………………..

Signature of Internal Examiner

Signature of External Examiner

ACKNOWLEDGEMENT

It gives us an immense pleasure to express deep sense of gratitude to my guide **Ms. Santhoshini Sahu,** Assistant Professor, Department of Computer Science and Engineering for her whole hearted and invaluable guidance throughout the project work. Without her sustained and sincere effort, this project work would not have taken this shape. She encouraged and helped us to overcome various difficulties that we have faced at various stages of our project work.

We would like to sincerely thank our Head of the department **Dr. A. V. Ramana**, for providing all the necessary facilities that led to the successful completion of our project work.

We would like to take this opportunity to thank our beloved Principal **Dr.C.L.V.R.S.V.Prasad**, for providing all the necessary facilities and a great support to us in completing the project work.

We would like to thank all the faculty members and the non-teaching staff of the Department of Computer Science and Engineering for their direct or indirect support for helping us in completion of this project work.

Finally, we would like to thank all of our friends and family members for their continuous help and encouragement.

T Anusha	**19341A05G7**
G Naga Manmitha	**20345A0513**
Sharon Kota	**19341A05F7**
S Laxman Rao	**19341A05F2**
N Ramesh Kumar	**20345A0514**
P Ganesh	**20345A0518**

ABSTRACT

Microservices is an organizational approach for the development of software, in which programs are subdivided into smaller independent services. Application Programming Interface also known as API, is an interface that allows two programs to communicate with each other. Whenever you operate an app like Facebook, send an urgent message, or check the weather on your phone, you are using an application working with APIs. Nowadays, special styles of computing mechanisms which provides in-depth services inclusive of mechanical, aerospace, civil and environmental engineering, packages are regularly deployed on the cloud. As it offers a handy on-call version for renting sources and clean-to-use flexible infrastructures, these are a part of API. The main advantage in the usage of microservices is the capability to provide security for network segregated web application. Microservice architecture is used in an application to design a modular layout, whereby every microservice has single, unique functionality and can be independently controlled and deployed. The overall study comprises experiences of various authors regarding the vulnerabilities in microservices and APIs. The prime motive is to attain enough knowledge about all the security concerns related to APIs and microservices.

Keywords: Microservices, Kubernetes, API, Spring Boot Framework, Docker, Buckets and Tokens.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS & ABBREVIATIONS

API	:	Application Programming Interface
XSS	:	Cross site Scripting
CSRF	:	Cross site Request Forgery
HA	:	High Availability
DDoS	:	Distributed Denial of Service
DoS	:	Denial of Service
CPS	:	Cyber Physical Systems
URI	:	Uniform Resource Identifier
MTD	:	Moving Target Defense
IDS	:	Intrusion Detection System
IPS	:	Intrusion Prevention System
MTLS	:	Mutual Transport Layer Security
IoT	:	Internet of Things
SOA	:	Service Oriented Architecture
REST	:	Representational State Transfer
MIPaRT	:	Microservices integrated overall performance and Reliability checking out
VERCASM-CPS	:	Vulnerability Analysis and Cyber Risk Assessment for Cyber Physical Systems

1. INTRODUCTION

Microservice Framework Structure also known as Microservices, is an architectural approach that organizes software into a collection of services that can be managed, tested, loosely coupled, and run independently. These services are organized around commercial enterprises abilities that are owned by small individual groups. This architecture enables fast, effective, and reliable delivery of huge and complex applications. Microservices are migrated from the architecture called Monolithic. A monolithic architecture is a traditional model of a software program, which is built as a unified unit that is self-contained and independent from other applications. In a monolithic architecture, each component and its associated components must all be present for code to be executed or compiled and for the software to run.

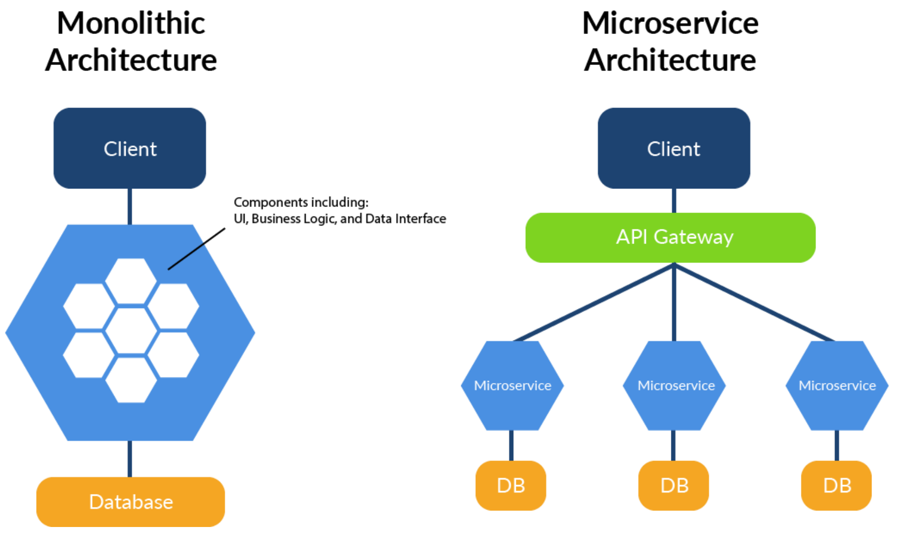


Figure 1.1: Migration of Monolithic to Microservices

Monolithic is a tightly coupled architecture whereas Microservices is a loosely coupled architecture. In the context of APIs, the phrase Application or Utility refers to any software with unique characteristics. Interface is a notion of an agreement between two different programs. This agreement specifies how two services communicate among one another with the help of request and response headers. Some common API vulnerabilities are damaged consumer authentication, unsuitable asset control, excessive information publicity, loss of assets and charge restricting. Application Programming Interface Security (APIs) refers to the mechanism of stopping or mitigating attacks on APIs. Consequently, it is very critical to defend every action the user performs. An API is an interface that defines how efficiently the software program interacts. It controls the patterns of requests that occur among programs, how those requests are made, and the varieties of information formats that are used. APIs are used in Internet of Things (IoT) packages and on websites. They regularly gather and manage facts and permit the person to enter information that gets processed within the environment housing the API.

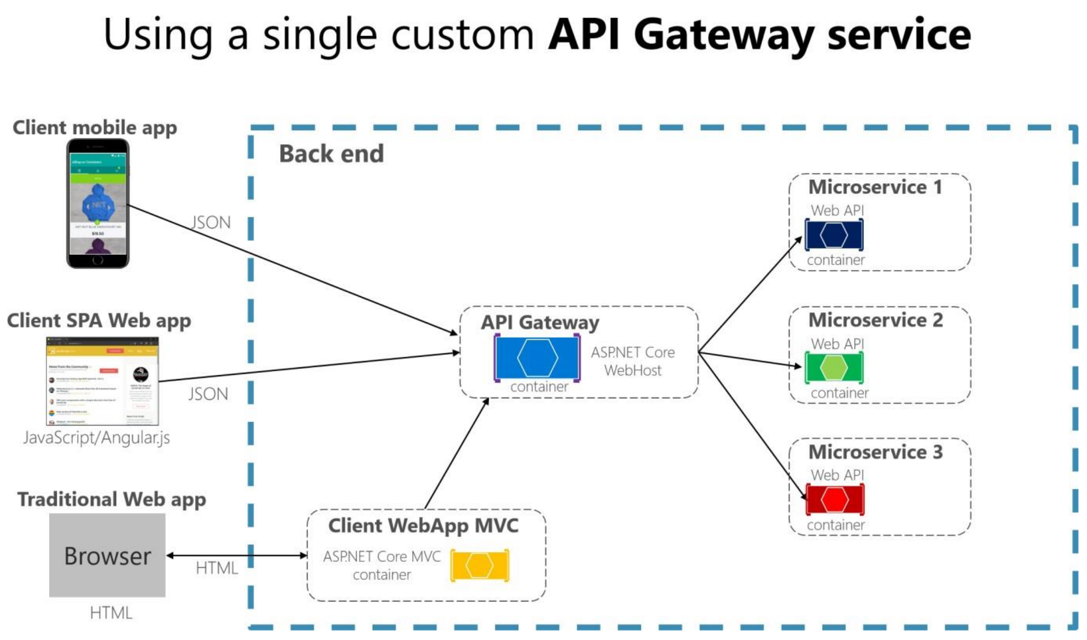


Figure 1.2: API gateway acts as an interface between Applications and Microservices

2. LITERATURE SURVEY

[1] Vayghan, L. A., Saied, M. A., Toeroe, M., & Khendek, F. (2021). A Kubernetes controller for managing the availability of elastic microservice based stateful applications. *Journal of Systems and Software*, *175*, 110924.

Microservice based architecture was growing rapidly using Kubernetes, a free software platform designed to manage containerized complete packages based on microservices. The impact of these experiments shows that adaptable behavior of Kubernetes could not met the high availability requirements. In other cases, Kubernetes could not assure carrier recovery. As a result, in few situations HA system controller integrated with Kubernetes was used. For the purpose of application state transfer and automatic provider redirection actions a proper architecture was needed in microservices for that another way of enabling service healing was implemented, and it was further used to the restore moves of Kubernetes. The outcomes of this examination displays that the stated solution enhances the healing time of microservices to gradually decrease. The primarily designed architecture is based on packages that are implemented by using 50% of highly useful resources. Their utilization of the solution was associated with 2N redundancy model, in which everystandby micro service safeguards the best live micro service. In terms of work to be implemented in future, this issue can be discussed by using and implementing other redundancy models that helps to standardize the percentage of a standby micro service for example the service is one among several lively micro service instances.

[2] Torkura, K. A., Sukmana, M. I., Kayem, A. V., Cheng, F., & Meinel, C. (2018, December). A cyber risk based moving target defense mechanism for microservice architectures, *2018 IEEE*

Moving target defense mechanism was used to resolve the drawbacks of the presence of vulnerabilities in microservices. The process involves changing of targets in a random fashion, so that it becomes difficult and confusing for the attacker to trace their respective targets. It includes attacks such as deception, evasion, and polymorphism. This method primarily deals with the evaluation of risks on microservices. Consequently, the loopholes present in these applications could be easily mitigated. Moving game defense was prone to numerous security risks because of docker image vulnerabilities, installed malware, and defectsin configuration. A mechanism that transforms the special device components to create uncertainty for attackers was called as Moving Target Defense (MTD). As a result of which, one could limit the chances of successful attacks. Diversification index- a metric in MTD, offers the intensity of diversification that was needed for microservices. Diversification index $D = TM/n$. Here, TM is range of microservices which was different and n denotes the number of microservices. This method offers 70% of fulfillment fee for resolving the issues of shared vulnerabilities in microservices.

[3] Northern, B., Burks, T., Hatcher, M., Rogers, M., & Ulybyshev, D. (2021). VERCASM-CPS: Vulnerability Analysis and Cyber Risk Assessment for Cyber-Physical Systems. *Information*, *12*(10), 408.

It was very important to defend one's information from different kinds of cyber-attacks in the process to provide increased security, protection, trust, avoid failure, and limit losses. Many attacks were being performed on the physical infrastructure of the Cyber Physical Systems (CPS). So, in order to prevent these attacks from happening, it was essential to research the configuration of CPS in automated mode. This helps in locating the CPS components which were suspected the most and were modified or put in place properly. Moreover, a public database had been taken to alter the most frequently used components in a CPS. The comparison was passed off among the specific CPS configuration to calculate the level of cyber damage encountered. The solution supports the implementation of Representational State Transfer (REST) in microservice design and also includes Analytics provider, API Gateway and a Discovery carrier. These are the three microservices that have their personal function defined within the machine. RESTful method was used in every microservice which can communicate and exchange information with different components. The controlled shifting target protection was incorporated with the VERCASM-CPS (Vulnerability Analysis and Cyber Risk Assessment for Cyber Physical Systems). Hence, the cyber chance evaluation engine helps in the replacement of inclined CPS components into different automated modes.

[4] Yarygina, T., & Otterstad, C. (2018, June). A game of microservices: Automated intrusion response. In *IFIP International Conference on Distributed Applications and Interoperable Systems* (pp. 169-177). Springer, Cham.

Increase in usage of services would lead to the increase of assaults on microservices. Several tactics already had been in use for detecting the intrusions but they consume extra time. So, there was a need for adaptable intrusion reaction mechanism i.e., game theoretic approach. This technique takes comparatively less time for tracing the vulnerabilities. Two types of systems known as Intrusion Detection System and Intrusion Prevention System are utilized. IDS was a process of monitoring and scanning the entire system and to verify if there was any malicious activity going on and to generate the alerts if any attack was detected whereas, IPS tries to block the intrusions that are detected. Once an intrusion was detected, respective prevention mechanisms must be taken as soon as possible to prevent the impact of an attack. IPS consists of a distributed set of functionalities that helps in tracking imperative entity referred to as the MicroserviceGame Engine (µGE). The µGE collects data from the network community and then it carries out the appropriate moves based on a search tree. This process helps to attain the possible results immediately after an intrusion had been detected. The µGE segregates the entire information which was collected and then builds a game tree. Simultaneously, the actual network adjustments take place and the illustration in the µGE will be up to date. The nodes in the tree are placed and eliminated as they convey to the microservice game engine.

[5] Akbulut, A., & Perros, H. G. (2019, June). Software versioning with microservices through the API gateway design pattern. In *2019 9th International Conference on Advanced Computer Information Technologies (ACIT)* (pp. 289-292). IEEE.

API gateway, a well-known microservice layout sample can handle the virtual hardware configurations for packing containers and the appropriate methods used. Many drawbacks are present in the traditional architectural style i.e., the monolithic structure. To overcome these drawbacks, microservices came into picture as an advancement of the conventional method. Microservices permit the users to divide the complete application into independent services, each having their own specific characteristics. The approximate versioning of software was a process of assigning model names to specific services. Uniform Resource Identifier (URI) was one of the methods for versioning; however it offers a very big URI blueprint which can ultimately become unmanageable. The other method is versioning inside the HTTP header, which was generally preferable than URI. It offers API model, Request header and receiver version. The proposed technique took 27% less hosting cost.

[6] Donham, J. (2018, September). A domain-specific language for microservices. In *Proceedings of the 9th ACM SIGPLAN International Symposium on Scala* (pp. 2-12).

Strato, a gadget constructed on Twitter was not only used to run a mass number of microservices hosted in a carrier computing platform but also includes buying and selling bit autonomy for lots of leverage. There were hundreds of divisions that run in the process of manufacturing at Twitter community. Some of them were like dealing with an average pinnacle-degree request utility logic. Microservices in Strato could be configured with the use of additives and could find ordinary styles of infrastructure at websites like Twitter. For instance, accessing any kind of data in the storage layer of Twitter, big apple etc. The Strato gadget has been under evolution for three years and almost 2 years in manufacturing traffic. Teams utilize Strato and several microservices strolling on this floor. StratoQL includes type checking, synthesis checking, subtyping, conversion, and asynchrony. Strato was a platform which supports flexible utility logic and implies to call for a few sorts of programming language wherein users can get good judgment explicitly.

[7] Somashekar, G., & Gandhi, A. (2021, April). Towards optimal configuration of microservices. In *Proceedings of the 1st Workshop on Machine Learning and Systems* (pp. 7-14).

The rise of microservice architecture in an application was further decomposed into one-of-a-kind interacting modules. These modules could be used to provide agility, scalability, and fault isolation capabilities. The use of a cluster with four servers where every server is composed of 24 hyper cores, 40 GB of memory, and 250 GB of disk area as a part of experimental setup. For the assessment, latency of ninety-five percentile was achieved as an overall performance metric. Other metrics could also be applied to measure the efficiency of the cluster. There were around six optimization algorithms used for evaluation. Two algorithms were related to heuristic-based probabilistic algorithms, the other two deals with evolutionary algorithms stimulated via populace primarily based biological evolution, and the remaining two were related to the sequential model based totally upon the optimization algorithms.

[8] Baarzi, A. F., Kesidis, G., Fleck, D., & Stavrou, A. (2020, April). Microservices made attack-resilient using unsupervised service fissioning. In *Proceedings of the 13th European workshop on Systems Security* (pp. 31-36).

An unmanaged, non-intrusive, alert message detection approach and fissioning based response mechanism had been used. The preliminary results states that by using the detection and protection mechanism, customers can a) successfully pick out the attacks. b) Decrease the impact of the attack on valid users by means of three XXX as compared with an incident wherein no defensive mechanism in the area was present. The study of DoS assault exposure and protection was performed notably beyond functionality that helps to have a look at only specialized mechanisms from previous paintings of software layer and DoS assaults. The process includes sensitivity evaluation which helps to reduce the effect on the number of quarantined replica pods and uses one at a time that is taken to perceive the attacker. The process specifies having too many quarantine pod replicas and then using it on one unused quarantine node may cause resource competition and many of the pods can be affected. So, it was necessary to provide security for inactive users who were legitimate at the node which was restrained. In addition to this, different size reducing tactics were used to perceive a division of microservices. This impacts the stop-to-stop software latency.

[9] Mateus-Coelho, N., Cruz-Cunha, M., & Ferreira, L. G. (2021). Security in microservices architectures. *Procedia Computer Science, 181,* 1225-1236.

The security in Microservices Architectures and the process which illustrated how to provide numerous safety strategies that were used presently in applications, to keep away from the threats in microservices had been discussed. Microservices structure was a newbie evolution of the monolithic structure. Microservices architecture was intrinsically linked to symbiosis that was already incorporated with box-primarily based deployment because none of these packing containers want controls for embedded operating structures. The process includes in creating different calls made to the resources in the operating systems, packages program interface and used in safety techniques like password complexity, authentication, and internet safety to keep away from the attacks like injection, cross-website online scripting, and broken access control.

[10] Torkura, K. A., Sukmana, M. I., Cheng, F., & Meinel, C. (2018, August). Cavas: Neutralizing application and container security vulnerabilities in the cloud native era. In *International Conference on Security and Privacy in Communication Systems* (pp. 471-490). Springer, Cham.

There were several security issues in container technology like Docker and Kubernetes. These issues have severe consequences on the security capability of containers. The study proposes a solution to overcome the security challenges by introducing a very impactful technique known as shift-left, which involves continuous checking during the development and integration phases of the microservices. Protection gateway acts as a threat manipulator gateways which enforce certain safety rules. It was a concept of Cloud conscious vulnerability assessment machine. This solution performs various kinds of operations like vulnerability detection, threat prioritization and resource allocation. It also disclosed the utility layer vulnerabilities. This technique follows five steps 1) Protection evaluation: It discovered the vulnerabilities present in microservices and provided security benefits like protection controls. 2) Security policies: It incorporated predefined guidelines for protection automation and control. These regulations provided protection to the services successfully. 3) Tamper evidence: the evidence should be isolated from viable attacks. It needs to be secured in such a way that they can no longer be discovered using the core services only. 4) Effectively resolves the technology: It was a thought to effectively resolve the technologies used in growing microservices. 5) Validate Vulnerability facts: validation of the vulnerable facts through photo vulnerability scanners.

[11] Heinrich, R., Van Hoorn, A., Knoche, H., Li, F., Lwakatare, L. E., Pahl, C., ... & Wettinger, J. (2017, April). Performance engineering for microservices: research challenges and directions. In *Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering Companion* (pp. 223-226).

There were several problems associated with performances, continues checking out, tracking, modeling of microservices, and handling the need for new overall performance checking out strategies for microservices. The microservices architectural fashion was an extension of the service orientated architecture (SOA) and had a smaller number of its features like scalability, flexibility and portability. The units which were of major importance in the utilization of microservices were RESTful net APIs which were mainly intended in improving the development, shipping, and deployment of the software. Microservices were deployed before the total-scale integration testing which includes all the applicable Microservices within the utility is done. Performance testing was made simpler and quicker with the process of effectively taking comments from operators and then explicitly incorporating logs that could change events into the choices that decided whether a deviation in runtime behavior was classified as an anomaly or not it was a promising solution. As massive microservices installations may also consist of lots of subdivided instances, it became an overtaking because of the dimensions of the models.

[12] Camilli, M., Guerriero, A., Janes, A., Russo, B., & Russo, S. (2022, May). Microservices integrated performance and reliability testing. In *Proceedings of the 3rd ACM/IEEE International Conference on Automation of Software Test* (pp. 29-39).

A proposed method- MIPaRT (Microservices integrated overall performance and Reliability checking out), was used to robotically looked at the performance and reliability evaluation of micro service systems collectively. Evolutionary adjustments caused changes within the behavior of the actors or added the new actors and the Operational adjustments may took placed which reveal problems and trigger a new Dev cycle. Scalability and overall performance were the satisfactory attributes of microservices that created maximum of the testing time in the demanding situations. Processes for performance testing followed one of the following strategies: trying out to satisfy single person needs or to satisfy a scalability requirement.The 3 degrees of MIPaRT are i) definition of the operational placing ii) exvivo testing and iii) integration analysis observed by means of visualization, performance, and reliability estimation. It came at the fee of the availability of microservices specs, service usage and system monitoring facts that are available in RESTful APIs.

[13] de Almeida, M. G., & Canedo, E. D. (2022). Authentication and Authorization in Microservices Architecture: A Systematic Literature Review. *Applied Sciences*, *12*(6), 3023.

The methods, technologies and challenges regarding the authorization and authentication techniques in microservices had been in implementation. The overall structure allows the communication between microservices have been in an entangled relationship which made it more difficult to control a whole mechanism and due to this they were preparing unique micro service the drawback was that they could been accessed without authorization. Mechanisms along with OAuth 2.0, API Gateway, JWT and technologies like Spring Boot Framework were used to develop the authentication architecture for the small microservices. In case of authentication between microservices, Mutual Transport Layer Security

(MTLS) was used. This guarantees the statistical confidentiality, integrity and legitimately identified the consumer. Most of the demanding situations were related to the complexity in the process of enforcing security in each micro service which increases overall complexity and the assaults may also increase. The API gateway was used here because the intermediate medium among the consumer, microservices and the SSO (single sign up) permits the person to authenticate himself in a most effective way as soon as they got admission to manage mechanisms like RBAC and ABAC which are also could been used in the authorization system.

[14] Hannousse, A., & Yahiouche, S. (2021). Securing microservices and microservice architectures: A systematic mapping study. *Computer Science Review, 41*, 100415.

Systematic mapping had been carried out to categorize the threats on Micro service architecture (MSA) with the security needs. The ultimate purpose of the research was to found and classify the attempts of attacks in detecting and preventing the security risks recognized. A few MSA threats that could took placed were primarily physical system-based attacks, information attacks, infrastructure attacks, software program assaults, auditing, access management, and mitigation. Prevention strategies could be used to attempt the process of recovery from the above referred type of assaults even before occurring. Verification and validation techniques also could be used to make sure that the safety of the microservices might be based on the overall performance evaluation and case research. Non-stop tracking of the microservices and using encryption strategies could be carried out to prevent exposure of confidential and touchy information.

[15] Di Francesco, P., Lago, P., & Malavolta, I. (2019). Architecting with microservices: A systematic mapping study. *Journal of Systems and Software, 150*, 77-97.

One must significantly improve the modern-day nation of art for expertizing in the scientific research on architectural growth of the microservices. The contributions compose of a reusable framework used to categorize, compare, and evaluate the working of models by using the reference architectures that were particular for microservices. The overall working of the infrastructure leads in the reduction of the mechanism's safety that was already involved in building, deploying, and running microservices. It also involves in managing community latency, unreliability, fault tolerance, records consistency, load balancing and transaction management. Microservices could run on any of a machine that could provide the perfect working system, a container engine, a virtualized environment, and a container engine on pinnacle of a virtualized environment. The maximum ordinary design styles while architecting with microservices were API Gateway, Proxy, submit and discovery patterns. The Gaps for destiny research go towards security and real- time communication in areas like IoT and cell.

[16] Leila Abdollahi Vayghan,Mohamed Aymen Saied, Maria Toeroe, Ferhat Khendek,published in 2019.DOI:- 10.1109/QRS.2019.00034.Microservice Based Architecture: Towards High-Availability for StatefulApplications with Kubernetes.

Kubernetes was a platform available to everyone, which did not show the difficulties faced in using microservices even as they had been built in the integrated form for their availability. The cut built integrated Kubernetes helped stateful microservices in identifying the issues. This work proposed a quick fix which complemented Kubernetes with a notion controller enabling built-in segregated microservices to process the state replication and to automate the services to redirect the helpful entities via management of labels that were secondary. The outcomes displayed that this solution helped in improving the processing duration of microservices, built in segregated microservices up-to 55% and even as much as integrated built-in microservices. In built-in microservices they used software that may cause built-inbox failure and pod process failure which may happens earlier than repair. Therefore, the software may be capable of enhancing the recovery time by means of 55% to 70%. As future integrated work, they will built-in software in the case of packages that had already got a replication capability.

[17] Isil Karabey Aksakalli , Turgay celik , Ahmet Burak Can , Bedir Tekinerdogan , Recevied in revised form 3 may 2021,Accepted 19 may 2021,Available online 7 june 2021.

Microservices was one kind of architectural pattern which was used to divide huge structures into smaller and independent functional divisions providing high compatibility. The main mission have been achieved in microservices architectural design was regularly mentioned in writings i.e., to identify the decay of service blocks. Major drawbacks that were recognized include modules have been deployed at respective platforms where the execution takes place, and to followed the verbal exchange styles. The paper reviewed a complete of 239 papers. Among those, authors decided on 38 as primary researched associated with the described studies questions. Microservices instances were deployed to provide exceptional Infrastructure factors which include box, and Orchestration structures (for example, Docker Swarm, Kubernetes and so on). Such balanced structures were typically incorporated to improve the flexibility and adaptability of utility classification.

[18] Salibindla, J. (2018). Microservices API security. *International Journal of Engineering Research & Technology*, *7*(1), 277-281.

Microservices were a version provider orientated architecture fashion which systemizes a utility into a class of services which were loosely connected. Microservices was enormously easy and compact to develop. The cause of the paper was to offer protection to the Restful API by using authentication of the customer's access. Among all the demanding situations, one was to build a Restful API having knowledge throughout the authorization and authentication strategies hence could get right of entry to the APIs (Microservices) which were restricted by means of putting in a blacklist of IP addresses. The OAuth 2.0 was a protocol used for authentication and this framework allows 3rd party software to achieve automatic access to an HTTP carrier. Username and password were contained and preserved in a token which was saved on server side. To keep away from this sort of data manipulation problems, HTTP Signatures were used which lets the consumer to signal the complete HTTP Message, these signatures were brought into picture by websites likeFacebook, Google, etc.

[19] Janes, A., & Russo, B. (2019, October). Automatic performance monitoring and regression testing during the transition from monolith to microservices. In *2019 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)* (pp. 163-168). IEEE.

The process experimented about the PPTAM+, a device that could constantly examine the degradation of a system for some duration of a transition to microservices. This device could constantly screen every microservice and additionally gave indication for the performance that's lost at the general degradation. One technique was used for classic programs to screen that was using software without enhancing the supply code and another way was to interact with the accessibility APIs in each cutting-edge working device. e.g.: windows, Mac OS, and Linux. PPTAM+, integrates a utility overall performance monitoring tool that collects performance statistics and provides interactive visualizations of the results. The performance evaluation was done by the stakeholders. The remarks supplied by the above monitoring tool may be used to relayout or efficaciously allocate the gadget capabilities with microservices. With this method, assessmentof the coupling between microservices and monitoring it over the transition could been achieved.

[20] Nacha Chondamrongkul, Jing Sun and Ian Warren. Published in 2020. IEEE. DOI:-10.1109/ICSA-CS50368.2020.00024. Automated Security Analysis for Microservice Architecture.

It was difficult in pronouncing the computerized security evaluation for Microservices structure hence the automated way was used. It was too challenged error-prone because it then called for guide analysis on the design version to pick out safety threats and trace viable assault situations. Here Ontology technology defined the safety traits, without enhancing and rebuilding the source code of the tool and then it became useful in every other generation gear. This was also used to the things this ware not cited here. The benefit was result of Ontology reasoning facilitates us to pick out a connector prone to DOS attack. The initial assessment had been conducted on extraordinary models to assess the accuracy of this technique. They had discovered that the precision of detection was predicated on measuring how correct the security traits were defined and their scope at the greater comprehensive evaluation. It was yet required to understand its performance higher.

[21] Nguyen, Q., & Baker, O. F. (2019). Applying Spring Security Framework and OAuth2 To Protect Microservice Architecture API. *J. Softw.*, *14*(6), 257-264.

The process carried out opportunity of making use of OAuth2 Framework and a Spring Security Framework to relaxed micro service APIs which actually had been developed on over the Spring Framework. The complete safety assessments on person used individual testing mechanisms later trying them manually these techniques were helpful for the mitigation of the loop holes and their effect thus these loopholes and their exploitation were available to watch and which had been helpful in checking the effectiveness of the OAuth2 and spring boot safety mechanism and they helps in providing privacy to the APIs that are built over the spring framework. This research implements a stock management gadget, the usage of MSA device for studying connection among any two mechanisms. The projects outcomes display the correctly identified assets by using CSRF assaults, XSS assault and Brute force assaults. Beside destiny, the developer wants to increase their study in the increase of safety for a complete API which includes the privacy of different utility areas which includes business, records get entry to layer. As such extra complete API safety solution was helpful for the Java-primarily based micro service-based application.

[22] Flora, J. (2020, October). Improving the security of microservice systems by detecting and tolerating intrusions. In *2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)* (pp. 131-134). IEEE.

This process actually implement for the improvement of the safety in the architectures of microservices via study and dependent incrimination of the overall technologies used for exploiting, bearing, tracing the protective attacks. An intrusion detection system (IDS) was actually a mechanism that displays units' community traffic for suspicious interest and alerts whilst such activity was found. The host-based IDSs are as relevant as the container technologies. The primary objective of these studies was to provide the functionality to assure a dependable and truthful within the existence of protective vulnerabilities. The very last result for an incorporated technique for securing microservices was the use of client-based vulnerability inspection in the microservices. A method or a layout which used further technologies for tolerance of the intrusions in complete microservice dependent systems was always been helpful.

[23] Sun, Y., Nanda, S., & Jaeger, T. (2015, November). Security-as-a-service for microservices-based cloud applications. In *2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)* (pp. 50-57). IEEE.

The endorse which was developed because of safety as a service necessity for microservice dependent cloud applications, the model proposed an API known as Flow Tap, an initial method which was used in the infrastructure of the which allows user to give their support to the high-quality online internet controlling. An easy to access surveillance and on demand implementation structure of the internet visitors in comfortable packages of a cloud had been constructed. The issues were only exacerbated through the cloud; on this account these infrastructures doesn't had overall management on the network. The community complexity delivered with the aid of the huge wide variety of microservices greatly would increase the problem in tracking the overall infrastructures privacy. The services were regularly deployed for the overall consistency for one another.

[24] Zaheer, Z., Chang, H., Mukherjee, S., & Van der Merwe, J. (2019, April). eztrust: Network-independent zero-trust perimeterization for microservices. In *Proceedings of the 2019 ACM Symposium on SDN Research* (pp. 49-61).

eZTrust, a network unbiased framework for microservices. A leverage eBPF, the advanced Berkeley Packet extracted out, and to identified real functionalities variables and then implemented in line with verification and pointing. As such, in a belief that the technology would commonly help for the developmentsin microservice finding out new technologies along with the present application programs transfer improvement capacities. The process undertook OVS glide-dependent software packet authorization, in which the packets were further divided into subdivisions which were no longer dependent totally on header fields of a packet, however primarily dependent upon microservice views.

[25] Wang, Y., Kadiyala, H., & Rubin, J. (2021). Promises and challenges of microservices: an exploratory study. *Empirical Software Engineering*, *26*(4), 1-44.

The major intention at accumulating and categorizing exceptional practices, demanding situations, and some current solutions for those demanding situations and the practitioners were employed via experts who are effectively developing microservice-based packages. Then it also recognizes numerous challenges that were nearer to the green controlling of a usual program throughout offerings an aid to the developer editions. These challenges collected the usage of a combined-approach and had a look at which the practitioners correctly adopted microservice based architectures. They also looked at blanketed keen reviews held with 21 developers then to write a continuation survey which acquires 37 responses pleasurable to the accepting norms. They discovered many challenges that have been faced frequently confronted by using developers who used microservices, inclusive of dealing with programs communicated among microservices and handling different component versions. In addition to this they also identified capacity in further stages, the network involves the similarity that used capable software programs which helped in improving the software practices for the development of a complete microservice based systems.

[26] Daoud, M., El Mezouari, A., Faci, N., Benslimane, D., Maamar, Z., & El Fazziki, A. (2021). A multi-model based microservices identification approach. *Journal of Systems Architecture, 118,* 102200.

Microservices had been majorly in used because of their abundant skills for addressed projects for dividing the inflexible information based computational mechanisms into tiny, abundant, and weekly-coupled functionalities. The interactive clustering mechanisms that was used in the capacity based microservices with advanced machine gaining knowledge of strategies including Herbal Language Processing was used in the development of novel sports credence's. A route had been an evolution of BP's functionalities source for the process to extract beneficial info that would be complemented. The total of thirty primary fashions had been already described. They developed, established a different-version technique for 20 microservices identified. The representations had been identified as an architectural, record, semantics, seize among a BP‟s sports.

[27] Vayghan, L. A., Saied, M. A., Toeroe, M., & Khendek, F. (2019). Kubernetes as an availability manager for microservice applications. *arXiv preprint arXiv:1901.04946*.

Examining greater model performs greater programs for evaluation of the feature known as availability of the Kubernetes that gave you designed application for its controlled microservices. The process played comparative assessment in Availability Management Framework (AMF), it provided a validated answer because the intermediate layer provides high availability for managing the application. The experiments were carried out in private cloud surroundings, thinking about specific failure situations, configurations, redundancy models, and to assess Kubernetes from the attitude of the provision of the applications gone for walks on Kubernetes. They also discovered that including repetition would notably got reduced time to time because components were recuperated immediately after discovering the errors by the Kubernetes andit would never rely upon restoring the defective unit.

[28] Vayghan, L. A., Saied, M. A., Toeroe, M., & Khendek, F. (2018, July). Deploying microservice based applications with kubernetes: Experiments and lessons learned. In *2018 IEEE 11th international conference on cloud computing (CLOUD)* (pp. 970-973). IEEE.

Final goal of the process was to create architecture to allow excessive availability, High availability (HA) with Kubernetes for micro service-based programs. The process examined the provision attainable via Kubernetes below its default configurations. Kubernetes, additionally (K8s), which was a free form device used in automatic dividing, ranging, and controlling applications which uses containers. The K8s reacts to a pod failure automatically through the beginning of a fresh pod and then it would be predicted for increasing the overall accessibility for the set of systems supplied. After that, as soon as a brand-new pod was introduced up with the aid of the deployment controller, the pod is took into consideration and repaired. A total of four parameters were used to evaluate Kubernetes availability: response time, restore time, outage time and healing time.

[29] Pereira-Vale, A., Fernandez, E. B., Monge, R., Astudillo, H., & Márquez, G. (2021). Security in microservice-based systems: A multivocal literature review. *Computers & Security*, *103*, 102200.

The description of the architecture and consequences of literature review expressed by different authors about the safety answers which were made known for a complete system functioning with microservices. Here, safety answers discussed within every study had been categorized as versions in popular security techniques, scopes (Threat Modeling, Data Management, etc.), and had been associated to safety contexts (stumble on, mitigate/prevent, react, recover from attack). This approach proposed few approaches to achieve a complete system based microservice security. Hence, primarily depending upon those results they had a look and encouraged several directions for the research. The outcomes of this were abundant thesis related to authorization and authentication to manage. It was not always unexpected, as primary safety procedures and a comfortable device was have been presented. The detection of attacks and preventing them were some common protection techniques used within a complete microservice system studies.

[30] Arzo, S. T., Scotece, D., Bassoli, R., Barattini, D., Granelli, F., Foschini, L., & Fitzek, F. H. (2022). MSN: A Playground Framework for Design and Evaluation of MicroServices-Based sdN Controller. *Journal of Network and Systems Management*, *30*(1), 1-31.

The description of the microservices primarily based on the decomposition structure, was introduced for the Software Defined Networking (SDN) paradigm to facilitate the improvement of scalability, reliability, andagility. The features of SDN along with flow in a dynamic manner managed and increased the chances of reconfiguring the community as per the application requirements. It made the process available to 5G next-era IIoT (Industrial IoT) networks. The MSN framework was proposed which provides a path for modern wave techniques based on microservices that were majorly used in the following generation of SDNnetwork using 5G technologies. The interconnection among the microservices with each other would be developed by reading distinctive protocols consisting of relaxation, gRPC, and Websocket.

3. METHODOLOGY

This work is to trace the vulnerabilities that are present in the microservices then to reduce the effect of the attacks in APIs. There will be a need to develop few applications with the help of Kubernetes, Spring Boot Framework and React JS. In this project we need 5 web applications built on different technologies.

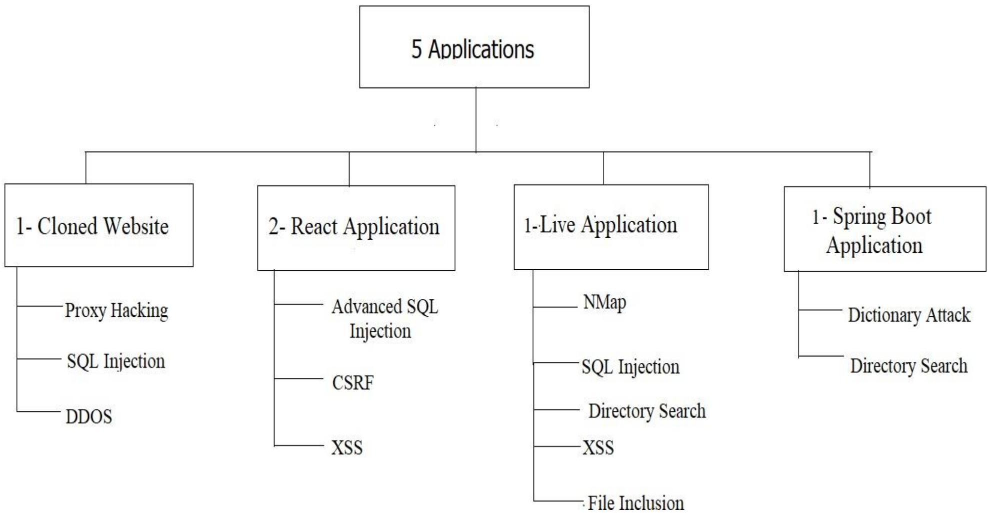


Figure 3.1: Flowchart for Applications and Followed Attacks

React JS helps in the development of the front-end applications. The creation of flexible and strong interfaces is the work of react JS. An open source framework, Spring Boot Framework is a mechanism that uses while developing applications in the most dynamical and simplest way. This framework is utmost helpful in development of the java stand-alone applications. There are few back-end mechanisms that have to be used while developing a perfect set of microservices such as Oracle and MongoDB. Oracle (also RDBMS) is used in computing the grid frameworks and also essentially used in the warehouses which have large amount of data. The oracle allows users to view, edit, modify or delete the data by an authorized user in a more efficient way compared to the other databases. MongoDB a NoSQL database is capable of storing the very large amount of data and it also provides the user friendly environment with better data accessing experiences. Container technology will be used in Kubernetes (K8s) these Kubernetes will allow to manage the applications remotely. Another technology called as Docker technology is similar to Kubernetes having the single difference is that Docker allows users to put the data jointly for the further uses. This will be helpful while developing the applications.

Live Applications are public applications that can be accessed by anyone. In this project we have taken the website called “AltoroMutual”. It is a banking application that can handle transactions like money withdrawal, deposit etc. This is a developed website which means it is already in use. On these applications, few attacks are performed using various tools and identified the vulnerabilities. The attacks are Nmap, Directory Search, File Inclusion, Sql injection, XSS.

Nmap:

To learn which ports are open and what those rules are, a tool called Nmap can be used. Nmap scans the network that a computer is connected to and outputs a list of ports, device names, operating systems, and several other identifiers that help the user understand the details behind their connection status. It can also be used to gain access to uncontrolled ports on a system. The attacker needs to run Nmap on the targeted system and look for vulnerabilities or loopholes in the system. The loop holes can further be exploited to gain access to sensitive information.

For performing attacks and identifying the vulnerabilities linux is more flexible than the other operating systems. In the linux terminal, nmap should be installed by giving the command like

sudo apt install nmap

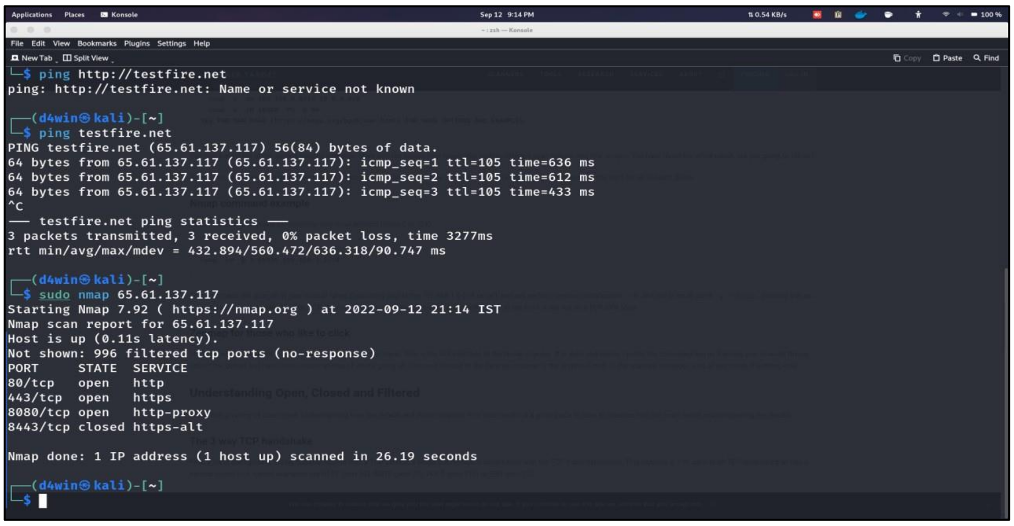
A command is used for determine which ports are open, whose services are being used.

sudo nmap IP address

In this experiment, IP address of the banking application is 65.61.137.117

By using the above command, all information related to that IP address will be displayed.

sudo nmap 65.61.137.117 Figure 3.2: Nmap



```
└─$ ping http://testfire.net
ping: http://testfire.net: Name or service not known

┌──(d4win㉿kali)-[~]
└─$ ping testfire.net
PING testfire.net (65.61.137.117) 56(84) bytes of data.
64 bytes from 65.61.137.117 (65.61.137.117): icmp_seq=1 ttl=105 time=636 ms
64 bytes from 65.61.137.117 (65.61.137.117): icmp_seq=2 ttl=105 time=612 ms
64 bytes from 65.61.137.117 (65.61.137.117): icmp_seq=3 ttl=105 time=433 ms
^C
--- testfire.net ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 3277ms
rtt min/avg/max/mdev = 432.894/560.472/636.318/90.747 ms

┌──(d4win㉿kali)-[~]
└─$ sudo nmap 65.61.137.117
Starting Nmap 7.92 ( https://nmap.org ) at 2022-09-12 21:14 IST
Nmap scan report for 65.61.137.117
Host is up (0.11s latency).
Not shown: 996 filtered tcp ports (no-response)
PORT     STATE  SERVICE
80/tcp   open   http
443/tcp  open   https
8080/tcp open   http-proxy
8443/tcp closed https-alt

Nmap done: 1 IP address (1 host up) scanned in 26.19 seconds

┌──(d4win㉿kali)-[~]
└─$ 
```

Figure 3.2: Nmap

File Inclusion:

The vulnerabilities in file inclusion allow an attacker to read and sometimes execute files on the victim server or, as is the case with Remote File Inclusion, to execute code hosted on the attacker‟s machine. The vulnerability occurs due to the use of user-supplied input without proper validation.

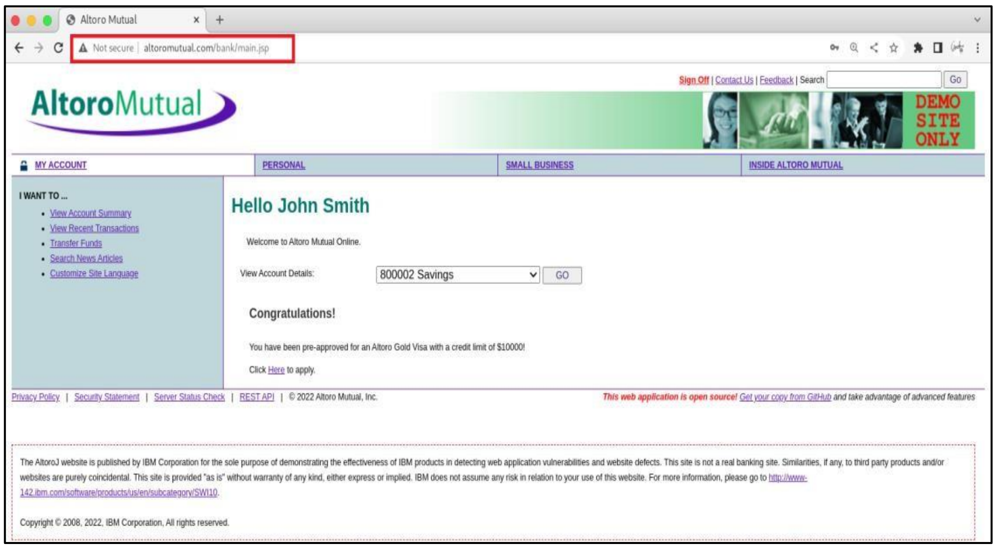
There are two types of File Inclusion vulnerabilities. They are:

- Local file inclusion (also known as LFI) is the process of including files, that are already locally present on the server, through the exploiting of vulnerable inclusion procedures implemented in the application. This vulnerability occurs, for example, when a page receives, as input, the path to the file that has to be included and this input is not properly sanitized, allowing directory traversal characters (such as dot-dot-slash) to be injected. Although most examples point to vulnerable PHP scripts, we should keep in mind that it is also common in other technologies such as JSP, ASP and others.
- Remote file inclusion (RFI) is an attack targeting vulnerabilities in web applications that dynamically reference external scripts. The attacker's goal is to exploit the referencing function in an application to upload malware (e.g., backdoor shells) from a remote URL located within a different domain.

This attack creates a way for an attacker to access admin details through the user details. An attacker logs into the application by testing different usernames and password. Due to poor password sanitization it is possible for an attacker to access admin details through user details after logging into the page. This leads to leakage of all user details and sensitive data.

Altoromutual.com/bank/main.jsp

It is a user’s web page, by testing different passwords, one can loin to the page. This picture shows the page after login to the application.


Altoro Mutual

Not secure | altoromutual.com/bank/main.jsp

Sign Off | Contact Us | Feedback | Search Go

AltoroMutual

DEMO SITE ONLY

MY ACCOUNT | PERSONAL | SMALL BUSINESS | INSIDE ALTORO MUTUAL

I WANT TO ...

- View Account Summary
- View Recent Transactions
- Transfer Funds
- Search News Articles
- Customize Site Language

Hello John Smith

Welcome to Altoro Mutual Online.

View Account Details: 800002 Savings GO

Congratulations!

You have been pre-approved for an Altoro Gold Visa with a credit limit of $10000!

Click Here to apply.

Privacy Policy | Security Statement | Server Status Check | REST API | © 2022 Altoro Mutual, Inc.

This web application is open source! Get your copy from GitHub and take advantage of advanced features

The AltoroJ website is published by IBM Corporation for the sole purpose of demonstrating the effectiveness of IBM products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided "as is" without warranty of any kind, either express or implied. IBM does not assume any risk in relation to your use of this website. For more information, please go to http://www-142.ibm.com/software/products/us/en/subcategory/SWI10.

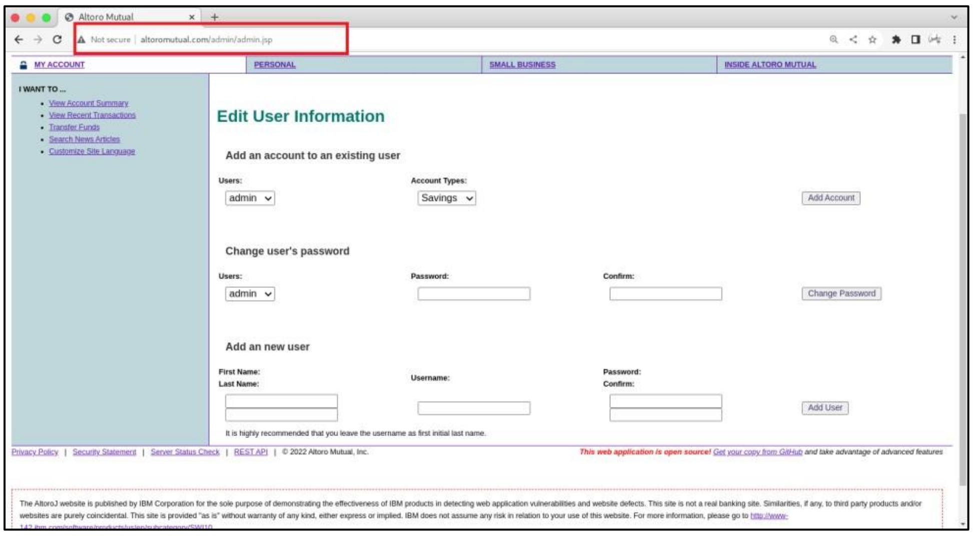
Copyright © 2008, 2022, IBM Corporation, All rights reserved.

Figure 3.3: Before the file inclusion attack

The directory path can be changed for accessing the admin page by changing like

Altoromutual.com/admin/admin.jsp

By changing the path of an url, admin page will be displayed. So an attacker can access all the users details. It will lead to perform some other attacks too. To stop this attack, password sanitization should be done. It will validate the entered password by checking the rules. If password sanitization is done, then there is no wayto perform file inclusion attack.


Altoro Mutual

Not secure | altoromutual.com/admin/admin.jsp

MY ACCOUNT | PERSONAL | SMALL BUSINESS | INSIDE ALTORO MUTUAL

I WANT TO ...

- View Account Summary
- View Recent Transactions
- Transfer Funds
- Search News Articles
- Customize Site Language

Edit User Information

Add an account to an existing user

Users: admin | Account Types: Savings | Add Account

Change user's password

Users: admin | Password: | Confirm: | Change Password

Add an new user

First Name: Last Name: | Username: | Password: Confirm: | Add User

It is highly recommended that you leave the username as first initial last name.

Privacy Policy | Security Statement | Server Status Check | REST API | © 2022 Altoro Mutual, Inc.

This web application is open source! Get your copy from GitHub and take advantage of advanced features

The AltoroJ website is published by IBM Corporation for the sole purpose of demonstrating the effectiveness of IBM products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided "as is" without warranty of any kind, either express or implied. IBM does not assume any risk in relation to your use of this website. For more information, please go to http://www-142.ibm.com/software/products/us/en/subcategory/SWI10

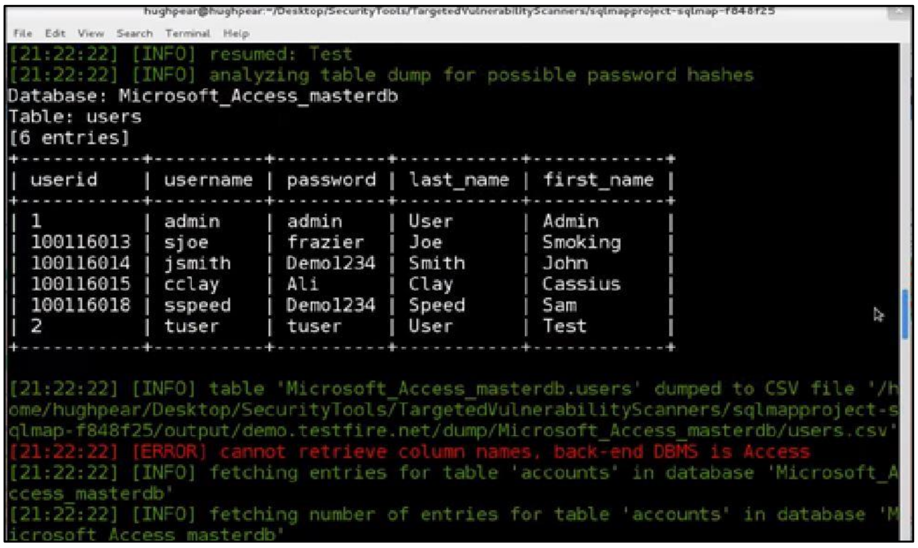
Figure 3.4: After the file inclusion attack

SQL injection:

SQL injection is also a type of injection technique which can be used to destroy your entire database. This attack is most commonly used as a hacking technique. It is generally done by placing malicious code in SQL statements through a web page input. SQL injection usually occurs when you ask a user for input, like their username/user id, and instead of a name/id, the user gives you an SQL statement that you will unknowingly run on your database.

- Ex: SELECT * FROM Users WHERE UserId = 105 OR 1=1;
- The SQL above is valid and will return ALL rows from the "Users" table, since **OR 1=1** is always TRUE.

This attack would be done by using SQLMap is an open source penetration testing tool that automates the process of detecting and exploiting SQL injection flaws and taking over of database servers. It comes with a powerful detection engine, many niche features for the ultimate penetration tester and over data fetching from the database, to accessing the underlying file system and executing commands on the operating system through out-of-band connections. The database for Altoromutual banking application is Microsoft_Access_masterdb which stores all the data about the application. Users is a table in the database.By doing this attack through SQLMap on the database, it returns all data of users like username, password, userid, lastname and firstname.


hughpear@hughpear:~/Desktop/SecurityTools/TargetedVulnerabilityScanners/sqlmapproject-sqlmap-f848f25
File Edit View Search Terminal Help

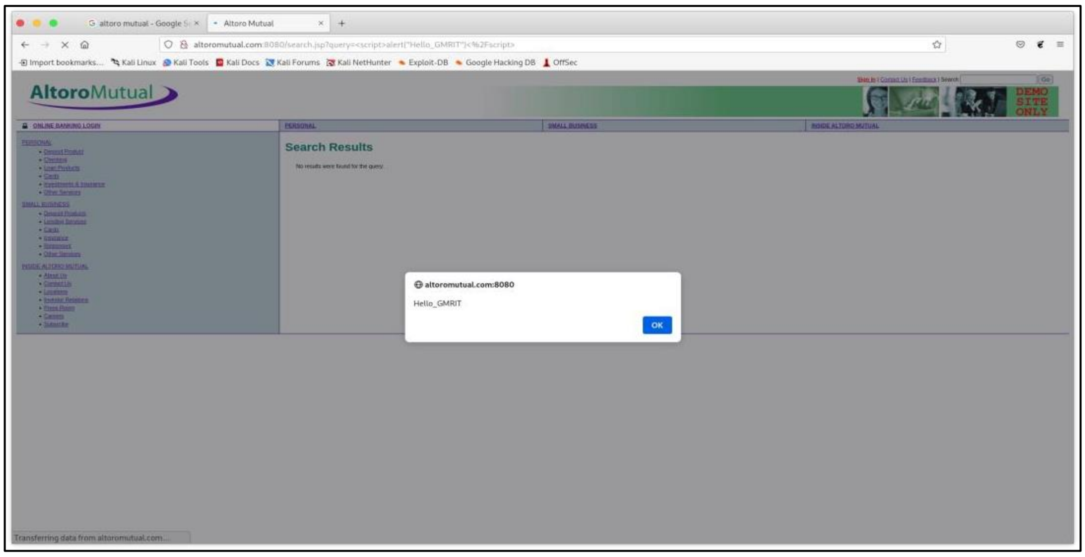
```
[21:22:22] [INFO] resumed: Test
[21:22:22] [INFO] analyzing table dump for possible password hashes
Database: Microsoft_Access_masterdb
Table: users
[6 entries]
+-----------+----------+----------+-----------+------------+
| userid    | username | password | last_name | first_name |
+-----------+----------+----------+-----------+------------+
| 1         | admin    | admin    | User      | Admin      |
| 100116013 | sjoe     | frazier  | Joe       | Smoking    |
| 100116014 | jsmith   | Demo1234 | Smith     | John       |
| 100116015 | cclay    | Ali      | Clay      | Cassius    |
| 100116018 | sspeed   | Demo1234 | Speed     | Sam        |
| 2         | tuser    | tuser    | User      | Test       |
+-----------+----------+----------+-----------+------------+

[21:22:22] [INFO] table 'Microsoft_Access_masterdb.users' dumped to CSV file '/h
ome/hughpear/Desktop/SecurityTools/TargetedVulnerabilityScanners/sqlmapproject-s
qlmap-f848f25/output/demo.testfire.net/dump/Microsoft_Access_masterdb/users.csv'
[21:22:22] [ERROR] cannot retrieve column names, back-end DBMS is Access
[21:22:22] [INFO] fetching entries for table 'accounts' in database 'Microsoft_A
ccess_masterdb'
[21:22:22] [INFO] fetching number of entries for table 'accounts' in database 'M
icrosoft_Access_masterdb'
```

Figure3.5: SQL Injection

Cross-Site Scripting (XSS):

XSS attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it.

In this attack a script is injected to the application by writing the java script code. So that, it will display an alert message. It would be done by detecting where the parameters are passing like entering login details, searching for anything, etc. Passing parameters allows attackers to inject their code to execute a malicious attack on the victim's system. The process of finding the passing parameters would be done by using Zed proxy it is a penetrating testing tool. It is a security testing tool that helps to find potential vulnerabilities in a web application. It has a spidertool that performs web crawling and checks where the parameters are going. It generates a report containing a list of parameters passed. The report is sent to BurpSuite and a link is established between BurpSuite and Zed Proxy. The attackers then intercept the requests and inject their code into the application.


altoromutual.com:8080/search.jsp?query=<script>alert("Hello_GMRIT")<%2Fscript>

Search Results

No results were found for the query.

altoromutual.com:8080

Hello_GMRIT

altoro mutual - Google S
Altoro Mutual
Import bookmarks... Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
AltoroMutual
DEMO SITE ONLY
ONLINE BANKING LOGIN
PERSONAL
SMALL BUSINESS
INSIDE ALTORO MUTUAL
OK
Transferring data from altoromutual.com...

Figure 3.6: XSS attack

Cloned website:

Cloned website refers to an exact copy of an original website. If the user login with his credentials into the original website then automatically that sensitive data will be stored in the cloned website. A number of attacks can be performed on a cloned website. Some of these attacks include SQL injection, DDOS (Distributed Denial of Service), Proxy Hacking attacks using Wireshark. Each attack has its own functionality and those can be done by using the above mentioned tools. In this project, Smart Cities Mission website is taken and performed these attacks. This website is done by taking the original website as reference. There is another way to copying the website by using a curl tool. It clones the original website and runs it on local host. The website works even without internet connection.

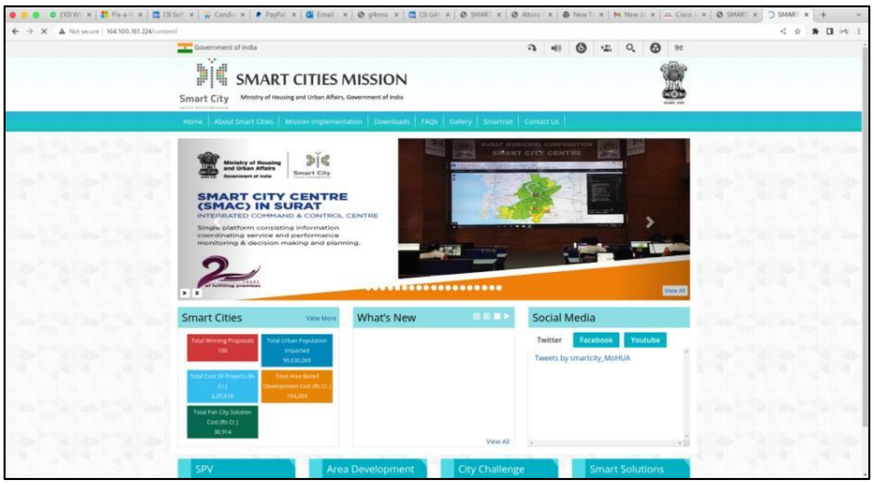

SMART CITIES MISSION
Smart City
Ministry of Housing and Urban Affairs, Government of India
Home | About Smart Cities | Mission Implementation | Downloads | FAQs | Gallery | Smartnet | Contact Us
SMART CITY CENTRE (SMAC) IN SURAT
Smart Cities
What's New
Social Media
SPV
Area Development
City Challenge
Smart Solutions

Figure 3.7: Original Smart Cities Mission Website

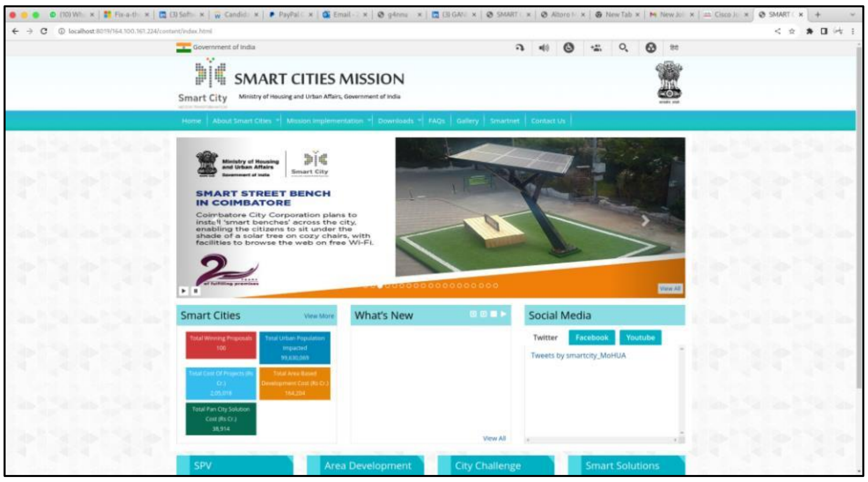

SMART CITIES MISSION
Smart City
Ministry of Housing and Urban Affairs, Government of India
Home | About Smart Cities | Mission Implementation | Downloads | FAQs | Gallery | Smartnet | Contact Us
SMART STREET BENCH IN COIMBATORE
Smart Cities
What's New
Social Media
SPV
Area Development
City Challenge
Smart Solutions

Figure 3.8: Cloned Smart Cities Mission Website

Proxy Hacking:

Proxy hacking is a technique used to attack genuine and original web pages by replacing them with proxies or clones in the results page. The Motive of this attack is to take advantage of the targets system. The assaulter makes a copy of the web pages on a proxy server and uses different ways like linking the replicated websites from the external website to increase the search engine ranking. By doing this, the search engine removes the original website by analyzing the rankings. The solution for this attack is, limiting the network connection from the free and open proxy servers.

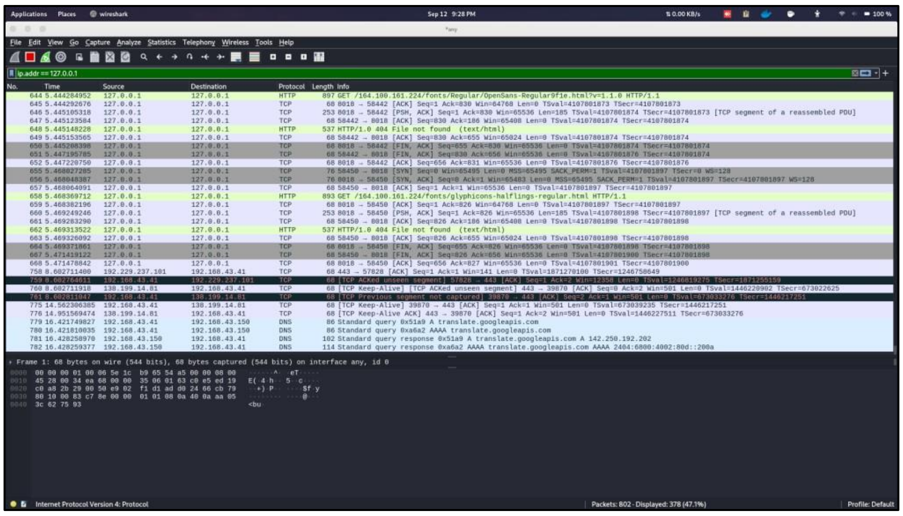
Distributed denial-of-service (DDoS):

A DDoS attack is a malicious attempt to disrupt the services or the normal traffic of a targeted server, or network by sending the target or its surrounding infrastructure with a number of requests or a flood of Internet traffic. DDoS attacks are done by using multiple compromised computer systems as the sources of attack traffic. The exploited machines may include computers and other network resources such as devices connected through IoT. It prevents any traffic from arriving to the destination. These machines or computers which are compromised will have been infected with malware, allowing them to be controlled remotely by the attacker. These group of computers infected with malware can also be called a Botnets orRobot Network.

Wireshark:

Wireshark is a network protocol analyzer, or an application that captures packets from a network connection, such as from your computer to your home office or the internet. Wireshark is the most often-used packet sniffer in the world. The three things done by wireshark are:

- **Packet Capture:** Wireshark listens to a network connection in real time and then grabs entire streams of traffic – quite possibly tens of thousands of packets at a time.
- **Filtering:** Wireshark is capable of slicing and dicing all of this random live data using filters. By applying a filter, you can obtain just the information you need to see.
- **Visualization:** Wireshark, like any good packet sniffer, allows you to dive right into the very middle of a network packet. It also allows you to visualize entire conversations and network streams.


Applications
Places
wireshark
Sep 12 9:28 PM
0.00 KB/s
100 %
*any
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Internet Protocol Version 4: Protocol
Packets: 802 · Displayed: 378 (47.1%)
Profile: Default

```
ip.addr == 127.0.0.1
No.  Time          Source           Destination      Protocol Length Info
644 5.444284952  127.0.0.1        127.0.0.1        HTTP   897 GET /164.100.161.224/fonts/Regular/OpenSans-Regular9f1e.html?v=1.1.0 HTTP/1.1
645 5.444292676  127.0.0.1        127.0.0.1        TCP     68 8018 → 58442 [ACK] Seq=1 Ack=830 Win=64768 Len=0 TSval=4107801873 TSecr=4107801873
646 5.445105318  127.0.0.1        127.0.0.1        TCP    253 8018 → 58442 [PSH, ACK] Seq=1 Ack=830 Win=65536 Len=185 TSval=4107801874 TSecr=4107801873 [TCP segment of a reassembled PDU]
647 5.445123584  127.0.0.1        127.0.0.1        TCP     68 58442 → 8018 [ACK] Seq=830 Ack=186 Win=65408 Len=0 TSval=4107801874 TSecr=4107801874
648 5.445148228  127.0.0.1        127.0.0.1        HTTP   537 HTTP/1.0 404 File not found  (text/html)
649 5.445153565  127.0.0.1        127.0.0.1        TCP     68 58442 → 8018 [ACK] Seq=830 Ack=655 Win=65024 Len=0 TSval=4107801874 TSecr=4107801874
650 5.445208398  127.0.0.1        127.0.0.1        TCP     68 8018 → 58442 [FIN, ACK] Seq=655 Ack=830 Win=65536 Len=0 TSval=4107801874 TSecr=4107801874
651 5.447195785  127.0.0.1        127.0.0.1        TCP     68 58442 → 8018 [FIN, ACK] Seq=830 Ack=656 Win=65536 Len=0 TSval=4107801876 TSecr=4107801874
652 5.447220750  127.0.0.1        127.0.0.1        TCP     68 8018 → 58442 [ACK] Seq=656 Ack=831 Win=65536 Len=0 TSval=4107801876 TSecr=4107801876
655 5.468027285  127.0.0.1        127.0.0.1        TCP     76 58450 → 8018 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=4107801897 TSecr=0 WS=128
656 5.468048387  127.0.0.1        127.0.0.1        TCP     76 8018 → 58450 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=4107801897 TSecr=4107801897 WS=128
657 5.468064091  127.0.0.1        127.0.0.1        TCP     68 58450 → 8018 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=4107801897 TSecr=4107801897
658 5.468369712  127.0.0.1        127.0.0.1        HTTP   893 GET /164.100.161.224/fonts/glyphicons-halflings-regular.html HTTP/1.1
659 5.468382196  127.0.0.1        127.0.0.1        TCP     68 8018 → 58450 [ACK] Seq=1 Ack=826 Win=64768 Len=0 TSval=4107801897 TSecr=4107801897
660 5.469249246  127.0.0.1        127.0.0.1        TCP    253 8018 → 58450 [PSH, ACK] Seq=1 Ack=826 Win=65536 Len=185 TSval=4107801898 TSecr=4107801897 [TCP segment of a reassembled PDU]
661 5.469283290  127.0.0.1        127.0.0.1        TCP     68 58450 → 8018 [ACK] Seq=826 Ack=186 Win=65408 Len=0 TSval=4107801898 TSecr=4107801898
662 5.469313522  127.0.0.1        127.0.0.1        HTTP   537 HTTP/1.0 404 File not found  (text/html)
663 5.469326092  127.0.0.1        127.0.0.1        TCP     68 58450 → 8018 [ACK] Seq=826 Ack=655 Win=65024 Len=0 TSval=4107801898 TSecr=4107801898
664 5.469371861  127.0.0.1        127.0.0.1        TCP     68 8018 → 58450 [FIN, ACK] Seq=655 Ack=826 Win=65536 Len=0 TSval=4107801898 TSecr=4107801898
667 5.471419122  127.0.0.1        127.0.0.1        TCP     68 58450 → 8018 [FIN, ACK] Seq=826 Ack=656 Win=65536 Len=0 TSval=4107801900 TSecr=4107801898
668 5.471478842  127.0.0.1        127.0.0.1        TCP     68 8018 → 58450 [ACK] Seq=656 Ack=827 Win=65536 Len=0 TSval=4107801901 TSecr=4107801900
758 8.602711400  192.229.237.101  192.168.43.41    TCP     68 443 → 57828 [ACK] Seq=1 Ack=1 Win=141 Len=0 TSval=1871270100 TSecr=1246758649
759 8.602764611  192.168.43.41    192.229.237.101  TCP     68 [TCP ACKed unseen segment] 57828 → 443 [ACK] Seq=1 Ack=2 Win=12358 Len=0 TSval=1246819275 TSecr=1871255159
760 8.602711918  138.199.14.81    192.168.43.41    TCP     68 [TCP Keep-Alive] [TCP ACKed unseen segment] 443 → 39870 [ACK] Seq=0 Ack=2 Win=501 Len=0 TSval=1446220902 TSecr=673022625
761 8.602811047  192.168.43.41    138.199.14.81    TCP     68 [TCP Previous segment not captured] 39870 → 443 [ACK] Seq=2 Ack=1 Win=501 Len=0 TSval=673033276 TSecr=1446217251
775 14.562306385 192.168.43.41    138.199.14.81    TCP     68 [TCP Keep-Alive] 39870 → 443 [ACK] Seq=1 Ack=1 Win=501 Len=0 TSval=673039235 TSecr=1446217251
776 14.951569474 138.199.14.81    192.168.43.41    TCP     68 [TCP Keep-Alive ACK] 443 → 39870 [ACK] Seq=1 Ack=2 Win=501 Len=0 TSval=1446227511 TSecr=673033276
779 16.421749827 192.168.43.41    192.168.43.150   DNS     86 Standard query 0x51a9 A translate.googleapis.com
780 16.421810035 192.168.43.41    192.168.43.150   DNS     86 Standard query 0xa6a2 AAAA translate.googleapis.com
781 16.428250970 192.168.43.150   192.168.43.41    DNS    102 Standard query response 0x51a9 A translate.googleapis.com A 142.250.192.202
782 16.428259377 192.168.43.150   192.168.43.41    DNS    114 Standard query response 0xa6a2 AAAA translate.googleapis.com AAAA 2404:6800:4002:80d::200a

› Frame 1: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) on interface any, id 0
0000  00 00 00 01 00 06 5e 1c  b9 65 54 a5 00 00 08 00   ······^· ·eT····
0010  45 28 00 34 ea 68 00 00  35 06 01 63 c0 e5 ed 19   E(·4·h·· 5··c····
0020  c0 a8 2b 29 00 50 e9 02  f1 d1 ad d0 24 66 cb 79   ··+)·P·· ····$f·y
0030  80 10 00 83 c7 8e 00 00  01 01 08 0a 40 0a aa 05   ········ ····@···
0040  3c 62 75 93                                        <bu·
```

Figure 3.9: Wireshark

Application created with the help of ReactJS, MongoDB:

This application is a shopping website, which contains login page, adds to cart details and product details etc. The data that is entered in the login page by the customer like username, password and email id those will be saved in the database. The database used in this application is Mongo DB which stores large amount of data. The data connectivity is done to mongo db. ReactJS is a JavaScript library used for building a responsive user interface. It is one of the technologies of MERN stack. It can be used on client and server side as well as with other frameworks. React uses virtual DOM which is a JavaScript object. This will improve apps performance, since JavaScript virtual DOM is faster than the regular DOM. Mongo DB is a document-oriented NoSQL database used for high volume data storage. It makes use of collections and documents. Each database contains collections which in turn contains documents. Each document can be different with a varying number of fields. The size and content of each document can be different from each other. Data connectivity links disparate data sets and applications, including data from different identity spaces. The input data will be stored in the database due to the database connectivity. The attacks performed on this website are Advanced SQL injection, CSRF (Cross-Site Request Forgery) and XSS (Cross- site Scripting).

Application created with the help of ReactJS, Oracle Database:

This Application is based on ReactJS with Oracle Dtabase connectivity. It‟s Application contains the login page and registration page, when users or Admin login into the login page by their credentials. It can be verified and pass into the login page and which there credentials are stored in OracleDB. Here database will be taken is Oracle 21c express edition which is latest version and having the high performance and security features. And it will store the large data and we can easy fetch the data and provide the security to the data. In this application we used Xamp server to control the database and Nodejs which was backend to the reactjs Frontend, Here we developed both frontend and backend side. ReactJS is a JavaScript library used for building a responsive user interface. It is one of the technologies of MERN stack. It can be used on client and server side as well as with other frameworks. React uses virtual DOM which is a JavaScript object. This will improve apps performance, since JavaScript virtual DOM is faster than the regular DOM. The attacks performed on this website are Advanced SQL injection, CSRF (Cross-Site Request Forgery) and XSS (Cross- site Scripting).

Cross- Site Request Forgery (CSRF):

CSRF is a type of a cyber-attack which forces authenticated users to submit a request to web application against which they are currently authenticated. These attacks aim at exploiting the trust a web application has in a user who is authenticated. CSRF attacks are successful as the web application cannot differentiate between a request generated by an individual user and a request generated by a user without their consent.

a) Changing of password

b) The attacker's motive for carrying out a CSRF attack is to force the user to submit a state-changing record. Some examples are-

c) Submitting a transaction

Advanced SQL Injection:

SQL Injection is a web-based security vulnerability that allows attackers to see data they shouldn't be able to see, by allowing the attacker to interfere with an application's database requests by injecting malicious SQL injection payloads. For implementing the SQL Injection, SQL language will help to retrieve the data inside the database. This will be done by using payloads; a payload is malware that the threat actor intends to deliver to the victim. It is a piece of code that executes when hackers exploit vulnerability. The assaulter sends information to a target by integrating with payloads.

After performing above attacks on various types of web applications, we designed an Ecommerce application using spring boot technology by providing more security and repeated those attacks.

Spring Boot Framework:

Java Spring Framework or Spring Boot Framework is a popular, open source, enterprise-level framework for creating standalone, production-grade applications that run on the Java Virtual Machine (JVM). Spring Framework offers a dependency injection feature that lets objects define their own dependencies that the spring container later injects into them. This enables developers to create modular applications consisting of loosely coupled components that are ideal for microservices and distributed network applications. The main goal of Spring Boot Framework is to reduce Development, Unit Test and Integration Test time and to ease the development of Production ready web applications very easily compared to existing Spring Framework, which really takes more time.

Advantages of Spring Boot Framework:

- It is very easy to develop Spring Based applications with Java.

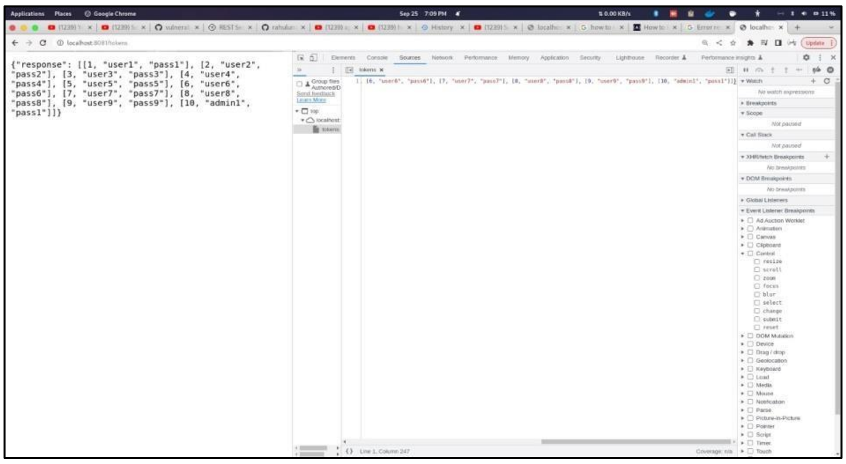
- It reduces lots of development time and increases productivity.

- It avoids writing lots of Code, Annotations and XML Configuration.
- It is very easy to integrate Spring Boot Application with its Spring Ecosystem like Spring JDBC, Spring Data, Spring Security etc.
- It follows “Opinionated Defaults Configuration” approach to reduce Developer effort.
- It provides Embedded HTTP servers like Tomcat, etc. to develop and test our web applications very easily.
- It provides CLI (Command Line Interface) tool to develop and test Spring Boot Applications from command prompt very easily and quickly.
- It provides lots of plugins to develop and test Spring Boot Applications very easily using Build Tools like Maven and Gradle.
- It provides many plugins to work with embedded and in-memory Databases very easily.

Spring boot framework has many advantages, but there are chances of attacks. There are some possible attacks which are currently happening on the spring boot websites. They are

Password in clear text:

An attacker situated in the user's ISP or the application's hosting infrastructure could also perform this attack. The Vulnerabilities that result in the disclosure of users' passwords, if passwords are in plain text, the security would be compromised by anyone having a glance at it.

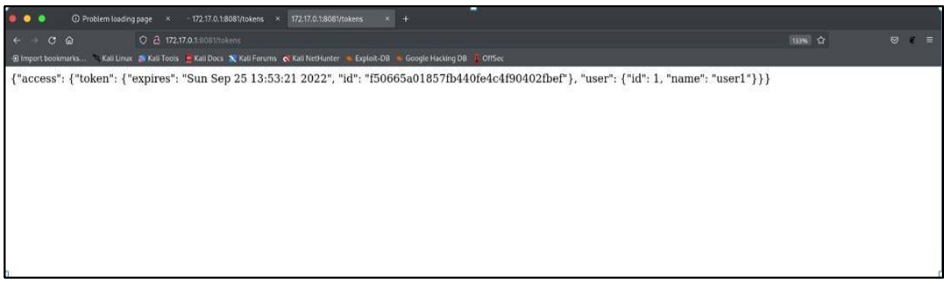


```
{"response": [[1, "user1", "pass1"], [2, "user2",
"pass2"], [3, "user3", "pass3"], [4, "user4",
"pass4"], [5, "user5", "pass5"], [6, "user6",
"pass6"], [7, "user7", "pass7"], [8, "user8",
"pass8"], [9, "user9", "pass9"], [10, "admin1",
"pass1"]]}
```

Figure 3.10: Password inclear text

Insecure Direct Object Reference (IDOR):

This attack occurs when an application provides direct access to objects based on user-supplied input. As a result of this vulnerability attackers can bypass authorization and access resources in the system directly, for example database records or files.

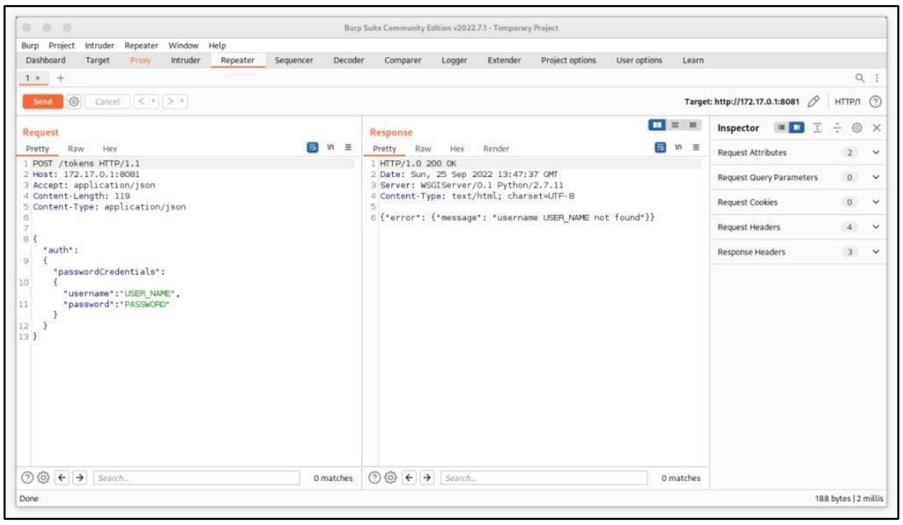


```
{"access": {"token": {"expires": "Sun Sep 25 13:53:21 2022", "id": "f50665a01857fb440fe4c4f90402fbef"}, "user": {"id": 1, "name": "user1"}}}
```

Figure 3.11: IDOR

Improper Implementation of Authorization Token:

In this attack, the access token is sent from the OAuth service to the client application via the user's browser as a URL fragment. The client application then accesses the token using JavaScript. The trouble is, if the application wants to maintain the session after the user closes the page, it needs to store the current user data (normally a user ID and the access token).



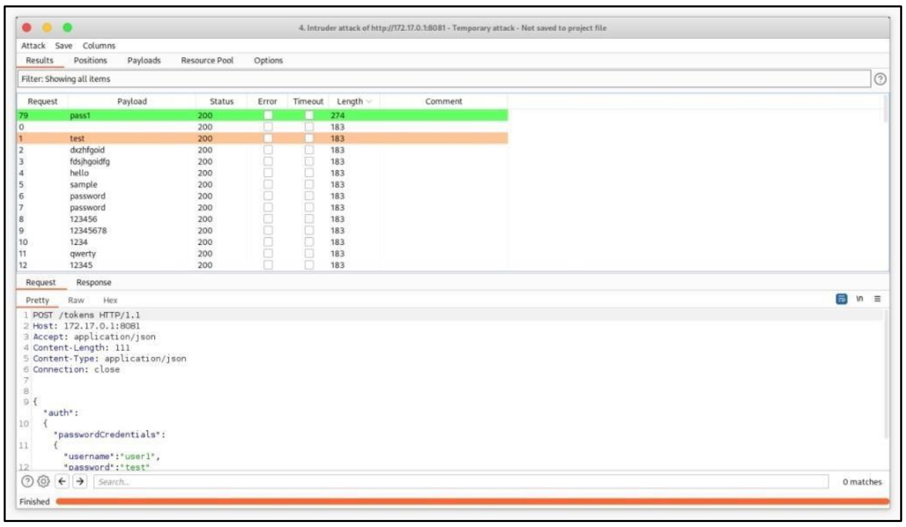
```
POST /tokens HTTP/1.1
Host: 172.17.0.1:8081
Accept: application/json
Content-Length: 119
Content-Type: application/json


{
  "auth":
  {
    "passwordCredentials":
    {
      "username":"USER_NAME",
      "password":"PASSWORD"
    }
  }
}
```

```
HTTP/1.0 200 OK
Date: Sun, 25 Sep 2022 13:47:37 GMT
Server: WSGIServer/0.1 Python/2.7.11
Content-Type: text/html; charset=UTF-8

{"error": {"message": "username USER_NAME not found"}}
```

Figure 3.12: Improper Implementation of Authorization Token

Privilege Escalation:

Every account that interacts with a system has some privileges. Standard users typically have limited access to system databases, sensitive files, or other resources. In some cases, users have excessive access to sensitive resources it leads to attackers can manipulate weaknesses of the system to increase privileges.


4. Intruder attack of http://172.17.0.1:8081 - Temporary attack - Not saved to project file
Attack
Save
Columns
Results
Positions
Payloads
Resource Pool
Options
Filter: Showing all items
Request
Response
Pretty
Raw
Hex
Search...
0 matches
Finished

Request	Payload	Status	Error	Timeout	Length	Comment
79	pass1	200	☐	☐	274	
0		200	☐	☐	183	
1	test	200	☐	☐	183	
2	dxzhfgoid	200	☐	☐	183	
3	fdsjhgoidfg	200	☐	☐	183	
4	hello	200	☐	☐	183	
5	sample	200	☐	☐	183	
6	password	200	☐	☐	183	
7	password	200	☐	☐	183	
8	123456	200	☐	☐	183	
9	12345678	200	☐	☐	183	
10	1234	200	☐	☐	183	
11	qwerty	200	☐	☐	183	
12	12345	200	☐	☐	183	

```
POST /tokens HTTP/1.1
Host: 172.17.0.1:8081
Accept: application/json
Content-Length: 111
Content-Type: application/json
Connection: close


{
  "auth":
  {
    "passwordCredentials":
    {
      "username":"user1",
      "password":"test"
```

Figure 3.13: Privilege Escalation

In this project, a Spring Boot Application is implemented by providing external security to protect against frequent attacks.

Transport Layer Security (TLS):

It encrypts data sent over the Internet to ensure that eavesdroppers and hackers are unable to see what you transmit which is particularly useful for private and sensitive information such as passwords, credit card numbers, and personal correspondence. The data is encrypted and not shown to the others.

Username Enumeration:

User enumeration is when a malicious actor can use brute-force techniques to either guess or confirm valid users in a system. User enumeration is often web application vulnerability, though it can also be found in any system that requires user authentication. Two of the most common areas where user enumeration occurs are in a site's login page and its, Forgot Password.

Information Exposure through Server Headers:

The information is available in header fields and can be acquired using a web browser to make a simple HTTP request to any web application. It is often called the web server banner and is ignored by most people with the exception of malicious ones. Attackers can perform banner grabbing using even simple TCP tools like telnet or net cat. Then they launch targeted attacks against your web server and version. In addition, if a particular web server version is known to be vulnerable to a specific exploit, the attacker would just need to use that exploit as part of their assault on the target web server.

After exploring all the possible attacks and techniques to protect from those attacks, an ecommerce web application is created that can withstand all those attacks. The web application is built using Spring Boot technology. Java Spring Framework is a popular, open source, enterprise-level framework for creating standalone, production-grade applications that run on the Java Virtual Machine (JVM). Java Spring Boot is a tool that makes developing web application and microservices with Spring Framework faster and easier through three core capabilities:

1. Auto configuration
2. An opinionated approach to configuration
3. The ability to create standalone applications

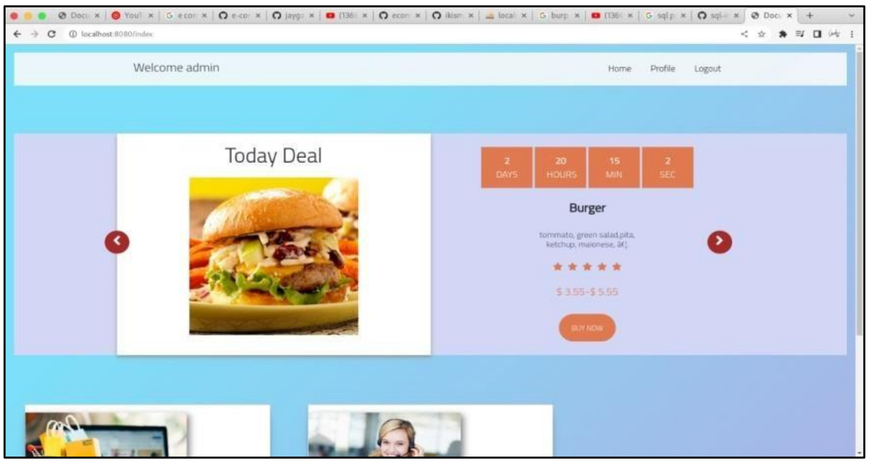
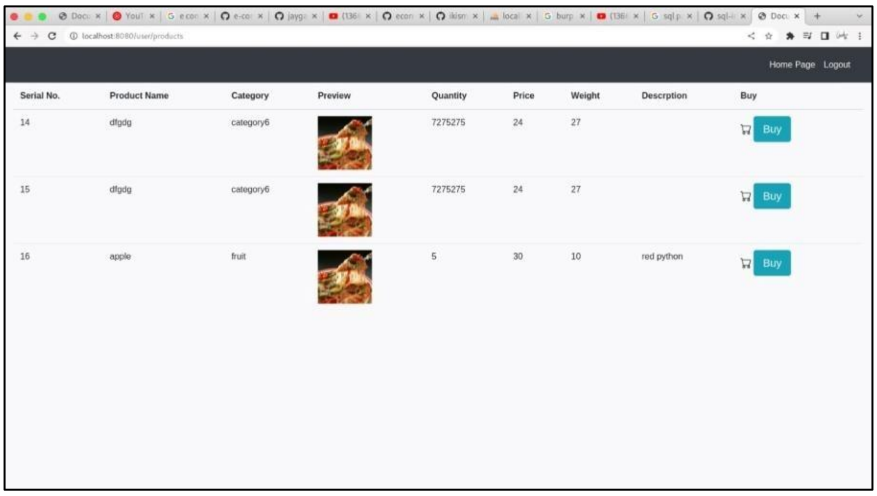


Figure 3.14: Home page of the website



Serial No.	Product Name	Category	Preview	Quantity	Price	Weight	Descrption	Buy
14	dfgdg	category6		7275275	24	27		Buy
15	dfgdg	category6		7275275	24	27		Buy
16	apple	fruit		5	30	10	red python	Buy

Figure 3.15: Add to cart page

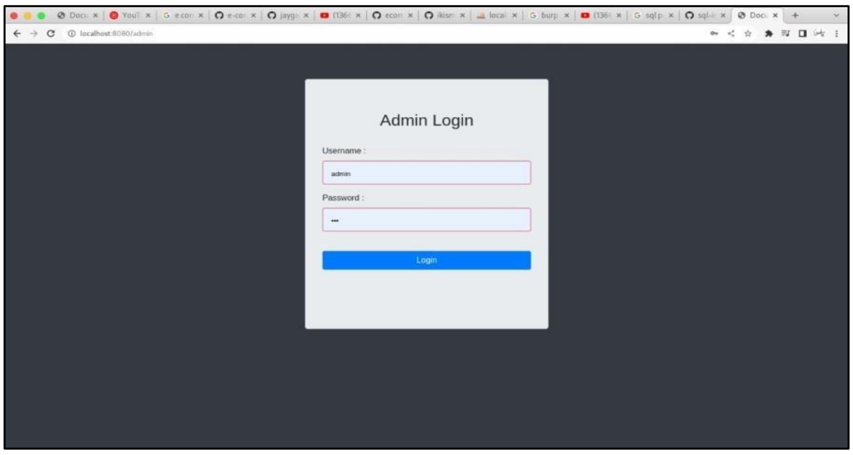
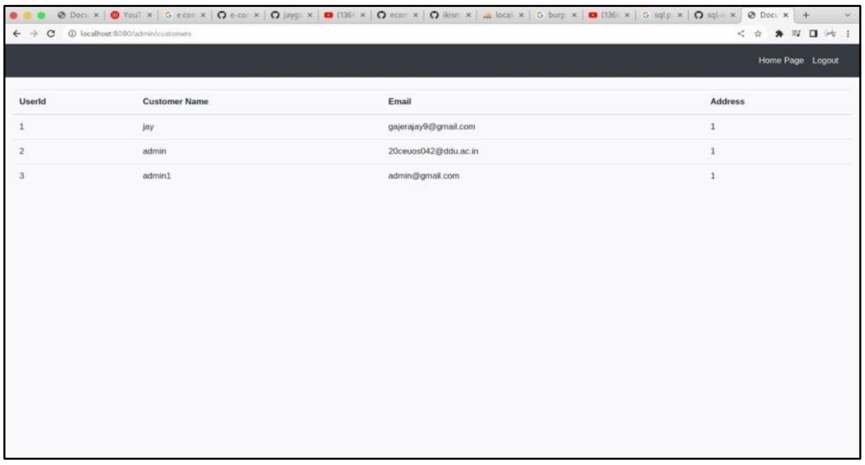
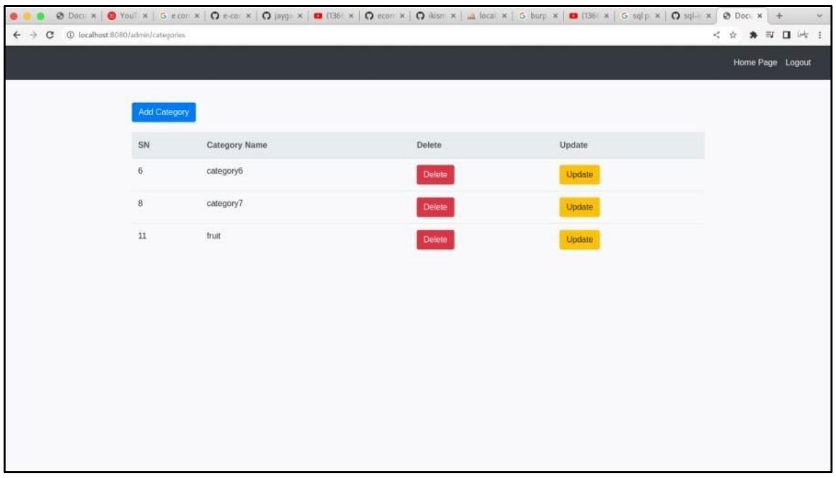


Figure 3.16: Login page



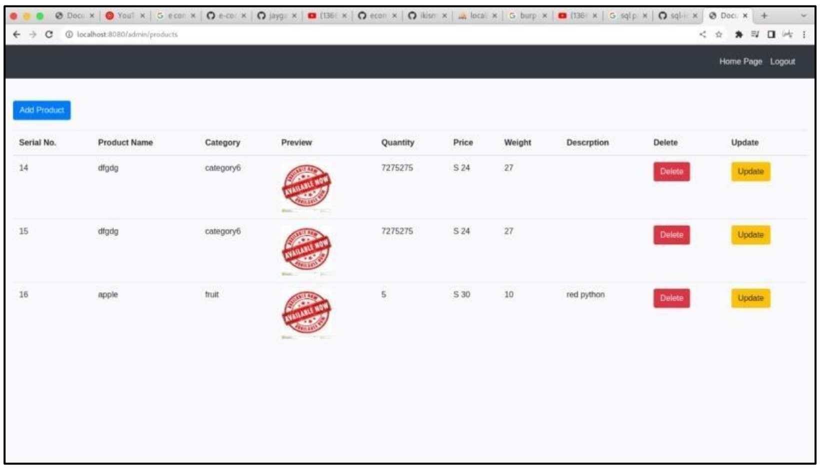
UserId	Customer Name	Email	Address
1	jay	gajerajay9@gmail.com	1
2	admin	20ceuos042@ddu.ac.in	1
3	admin1	admin@gmail.com	1

Figure 3.17: Customers details



SN	Category Name	Delete	Update
6	category6	Delete	Update
8	category7	Delete	Update
11	fruit	Delete	Update

Figure 3.18: Category page



Serial No.	Product Name	Category	Preview	Quantity	Price	Weight	Descrption	Delete	Update
14	dfgdg	category6		7275275	S 24	27		Delete	Update
15	dfgdg	category6		7275275	S 24	27		Delete	Update
16	apple	fruit		5	S 30	10	red python	Delete	Update

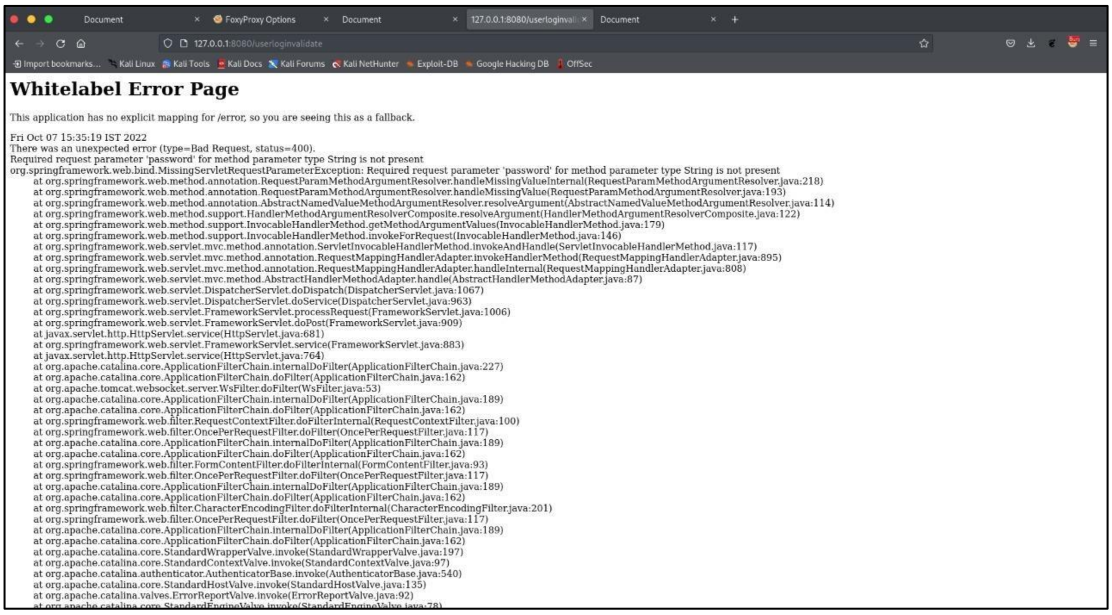
Figure 3.19: Add to cart

Above images are the website pages. The user has to register in the application and they can access the website by adding products to the cart. They can purchase products using the payments webpage.

4.EXPERIMENTATIONS AND RESULTS

In this project, the application is implemented with Spring Boot embedded with various security counter measures. The attacks were replicated on a spring boot ecommerce application, however the attacks did not work on this application. Because, this application is created to protect against all kinds of attacks. The performed attacks on this application are:

SQL Injection with payloads:



Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

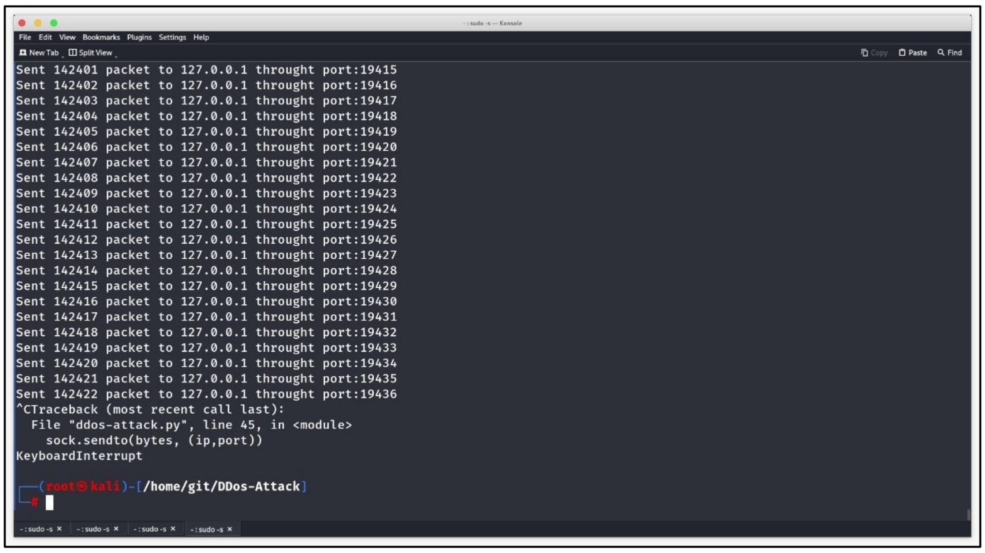
```
Fri Oct 07 15:35:19 IST 2022
There was an unexpected error (type=Bad Request, status=400).
Required request parameter 'password' for method parameter type String is not present
org.springframework.web.bind.MissingServletRequestParameterException: Required request parameter 'password' for method parameter type String is not present
	at org.springframework.web.method.annotation.RequestParamMethodArgumentResolver.handleMissingValueInternal(RequestParamMethodArgumentResolver.java:218)
	at org.springframework.web.method.annotation.RequestParamMethodArgumentResolver.handleMissingValue(RequestParamMethodArgumentResolver.java:193)
	at org.springframework.web.method.annotation.AbstractNamedValueMethodArgumentResolver.resolveArgument(AbstractNamedValueMethodArgumentResolver.java:114)
	at org.springframework.web.method.support.HandlerMethodArgumentResolverComposite.resolveArgument(HandlerMethodArgumentResolverComposite.java:122)
	at org.springframework.web.method.support.InvocableHandlerMethod.getMethodArgumentValues(InvocableHandlerMethod.java:179)
	at org.springframework.web.method.support.InvocableHandlerMethod.invokeForRequest(InvocableHandlerMethod.java:146)
	at org.springframework.web.servlet.mvc.method.annotation.ServletInvocableHandlerMethod.invokeAndHandle(ServletInvocableHandlerMethod.java:117)
	at org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter.invokeHandlerMethod(RequestMappingHandlerAdapter.java:895)
	at org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter.handleInternal(RequestMappingHandlerAdapter.java:808)
	at org.springframework.web.servlet.mvc.method.AbstractHandlerMethodAdapter.handle(AbstractHandlerMethodAdapter.java:87)
	at org.springframework.web.servlet.DispatcherServlet.doDispatch(DispatcherServlet.java:1067)
	at org.springframework.web.servlet.DispatcherServlet.doService(DispatcherServlet.java:963)
	at org.springframework.web.servlet.FrameworkServlet.processRequest(FrameworkServlet.java:1006)
	at org.springframework.web.servlet.FrameworkServlet.doPost(FrameworkServlet.java:909)
	at javax.servlet.http.HttpServlet.service(HttpServlet.java:681)
	at org.springframework.web.servlet.FrameworkServlet.service(FrameworkServlet.java:883)
	at javax.servlet.http.HttpServlet.service(HttpServlet.java:764)
	at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:227)
	at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:162)
	at org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:53)
	at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:189)
	at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:162)
	at org.springframework.web.filter.RequestContextFilter.doFilterInternal(RequestContextFilter.java:100)
	at org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter.java:117)
	at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:189)
	at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:162)
	at org.springframework.web.filter.FormContentFilter.doFilterInternal(FormContentFilter.java:93)
	at org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter.java:117)
	at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:189)
	at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:162)
	at org.springframework.web.filter.CharacterEncodingFilter.doFilterInternal(CharacterEncodingFilter.java:201)
	at org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter.java:117)
	at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:189)
	at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:162)
	at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:197)
	at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:97)
	at org.apache.catalina.authenticator.AuthenticatorBase.invoke(AuthenticatorBase.java:540)
	at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:135)
	at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:92)
	at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:78)
```

Figure 4.1: SQL injection with payloads

SQL Injection is a type of attack performed using SQL Map, an open source tool, with 108 payloads. But the attack failed and the databases on that website were not exposed. Protecting the database by verifying all the login details and no one is allowed to enter and access the database. By doing this one can save their sensitive data, login details etc.

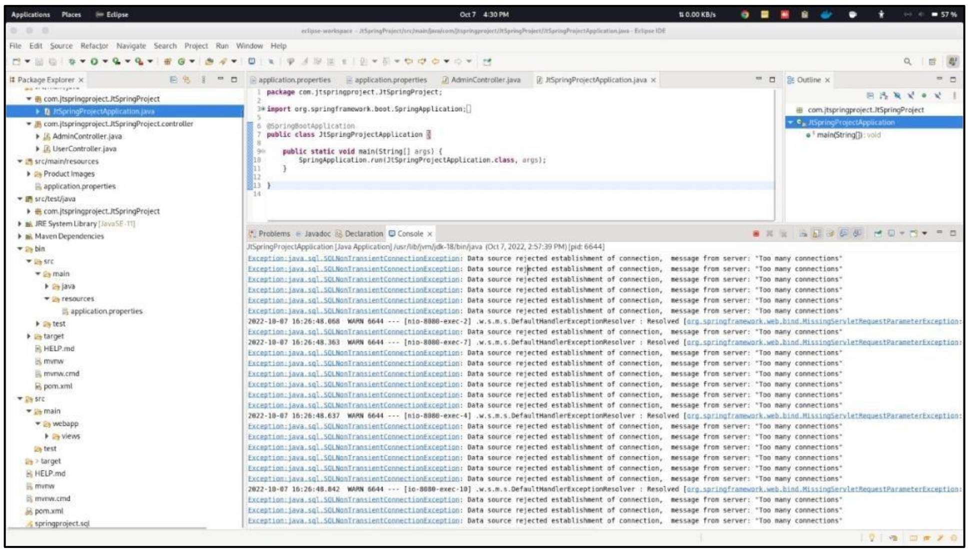
DDOS Attack:

Distributed denial of service is a type of attack where the main objective of this attack is to disrupt the traffic. The assaulter sends many requests to the targeted website IP address and this leads to server damage, data leakage. This attack implemented on the application through IP address but it failed to execute because it accepts only few requests, it observes the frequency of IP address, if the limit is exceeded, it blocks the IP address.



```
Sent 142401 packet to 127.0.0.1 throught port:19415
Sent 142402 packet to 127.0.0.1 throught port:19416
Sent 142403 packet to 127.0.0.1 throught port:19417
Sent 142404 packet to 127.0.0.1 throught port:19418
Sent 142405 packet to 127.0.0.1 throught port:19419
Sent 142406 packet to 127.0.0.1 throught port:19420
Sent 142407 packet to 127.0.0.1 throught port:19421
Sent 142408 packet to 127.0.0.1 throught port:19422
Sent 142409 packet to 127.0.0.1 throught port:19423
Sent 142410 packet to 127.0.0.1 throught port:19424
Sent 142411 packet to 127.0.0.1 throught port:19425
Sent 142412 packet to 127.0.0.1 throught port:19426
Sent 142413 packet to 127.0.0.1 throught port:19427
Sent 142414 packet to 127.0.0.1 throught port:19428
Sent 142415 packet to 127.0.0.1 throught port:19429
Sent 142416 packet to 127.0.0.1 throught port:19430
Sent 142417 packet to 127.0.0.1 throught port:19431
Sent 142418 packet to 127.0.0.1 throught port:19432
Sent 142419 packet to 127.0.0.1 throught port:19433
Sent 142420 packet to 127.0.0.1 throught port:19434
Sent 142421 packet to 127.0.0.1 throught port:19435
Sent 142422 packet to 127.0.0.1 throught port:19436
^CTraceback (most recent call last):
  File "ddos-attack.py", line 45, in <module>
    sock.sendto(bytes, (ip,port))
KeyboardInterrupt

┌──(root㉿kali)-[/home/git/DDos-Attack]
└─#
```

Figure 4.2:DDoS attack



```java
package com.jtspringproject.JtSpringProject;

import org.springframework.boot.SpringApplication;

@SpringBootApplication
public class JtSpringProjectApplication {

    public static void main(String[] args) {
        SpringApplication.run(JtSpringProjectApplication.class, args);
    }

}
```

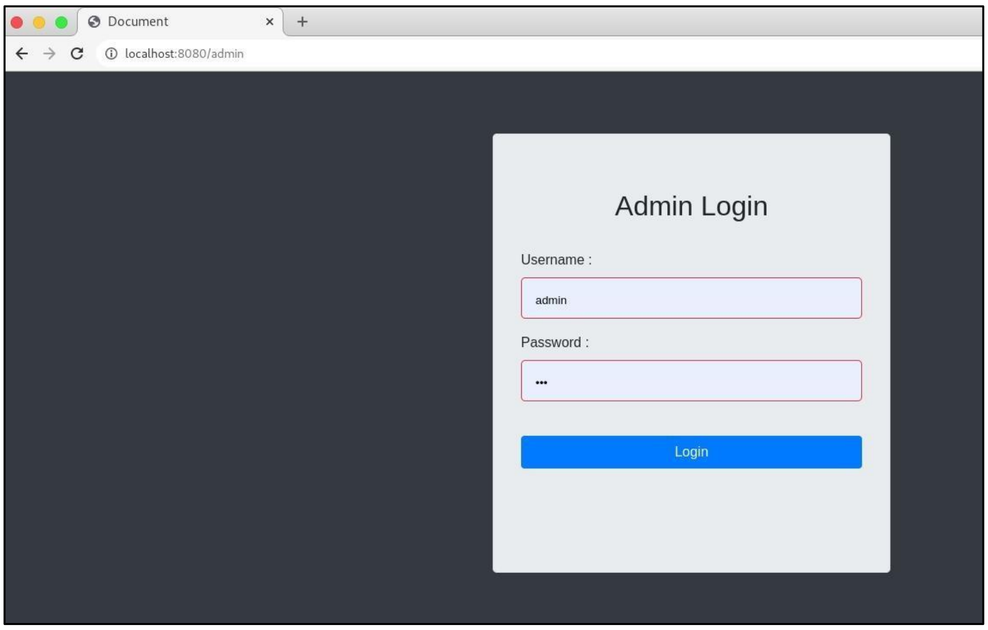
```
JtSpringProjectApplication [Java Application] /usr/lib/jvm/jdk-18/bin/java (Oct 7, 2022, 2:57:39 PM) [pid: 6644]
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
2022-10-07 16:26:48.068  WARN 6644 --- [nio-8080-exec-2] .w.s.m.s.DefaultHandlerExceptionResolver : Resolved [org.springframework.web.bind.MissingServletRequestParameterException:
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
2022-10-07 16:26:48.363  WARN 6644 --- [nio-8080-exec-7] .w.s.m.s.DefaultHandlerExceptionResolver : Resolved [org.springframework.web.bind.MissingServletRequestParameterException:
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
2022-10-07 16:26:48.637  WARN 6644 --- [nio-8080-exec-4] .w.s.m.s.DefaultHandlerExceptionResolver : Resolved [org.springframework.web.bind.MissingServletRequestParameterException:
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
2022-10-07 16:26:48.842  WARN 6644 --- [io-8080-exec-10] .w.s.m.s.DefaultHandlerExceptionResolver : Resolved [org.springframework.web.bind.MissingServletRequestParameterException:
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
Exception:java.sql.SQLNonTransientConnectionException: Data source rejected establishment of connection,  message from server: "Too many connections"
```

Figure 4.3: Blocking the IP address after six requests

After a DDoS attack on a web application, it often blocks the incoming IP address. The above image shows that it accepts only six requests, if the limit is exceeded it will block the IP address.

File Inclusion Attack:

In this attack, the attacker can access the admin data through user details. In this application, the attack is not possible as the details are verified everywhere. The sanitization of details is very important to prevent these types of attacks. Authentication happens when someone wants to login or register. So outsider cannotaccess other's details.



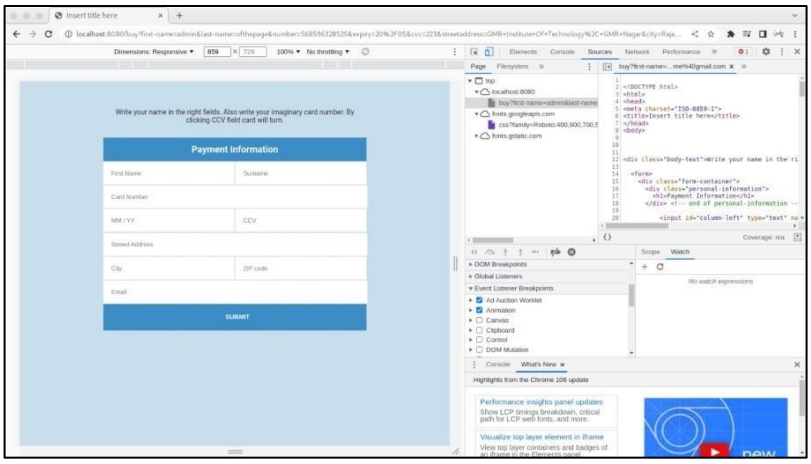
Username : admin

Password : •••

Figure 4.4: Validating the admin login details

Cross Site Request Forgery (CSRF):

Cross-Site Request Forgery (CSRF) is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. With a little help of social engineering, an attacker may trick the users of a web application into executing actions of the attacker's choosing. If the victim is a normal user, a successful CSRF attack can force the user to perform state changing requests like transferring funds, changing their email address, and so forth. If the victim is an administrative account, CSRF can compromise the entire web application. This attack is not possible in this website by making the website more secure by adopting counter measures.



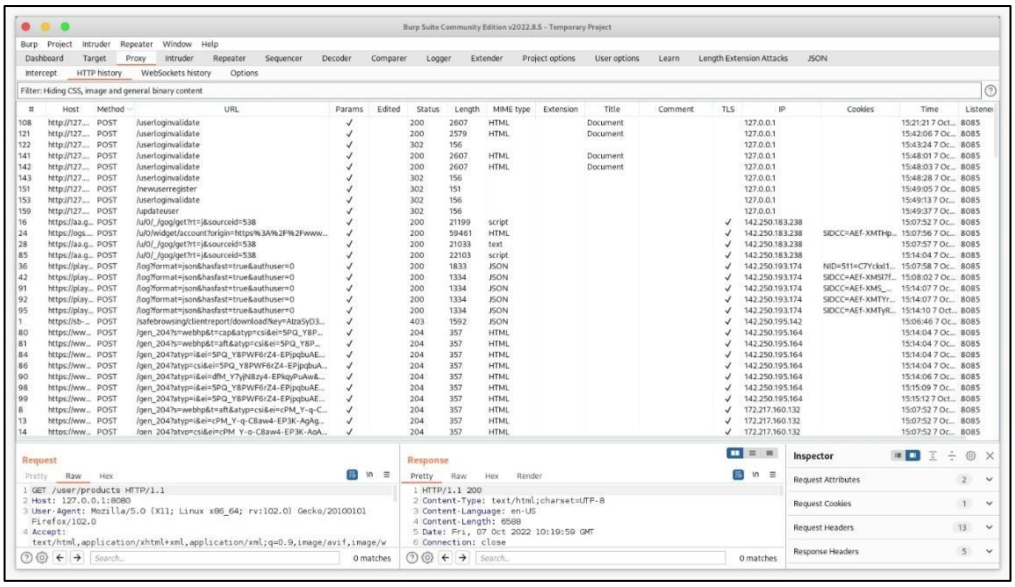
Write your name in the right fields. Also write your imaginary card number. By clicking CCV field card will turn.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>



<div class="body-text">Write your name in the ri

  <form>
    <div class="form-container">
      <div class="personal-information">
        <h1>Payment Information</h1>
      </div> <!-- end of personal-information --

          <input id="column-left" type="text" na
```

Figure 4.5: Blocking CSRF attack

Cross Site Scripting:

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. So they can execute their attack on the trusted website. In this website, this attack is not encouraged as the developer does not provide any kind of parameters.


Burp Suite Community Edition v2022.8.5 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn Length Extension Attacks JSON
Intercept HTTP history WebSockets history Options
Request
Pretty Raw Hex
Response
Pretty Raw Hex Render
Inspector
Search...
0 matches

Filter: Hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status	Length	MIME type	Extension	Title	Comment	TLS	IP	Cookies	Time	Listene
108	http://127....	POST	/userloginvalidate	✓		200	2607	HTML		Document			127.0.0.1		15:21:21 7 Oct...	8085
121	http://127....	POST	/userloginvalidate	✓		200	2579	HTML		Document			127.0.0.1		15:42:06 7 Oc...	8085
122	http://127....	POST	/userloginvalidate	✓		302	156						127.0.0.1		15:43:24 7 Oc...	8085
141	http://127....	POST	/userloginvalidate	✓		200	2607	HTML		Document			127.0.0.1		15:48:01 7 Oc...	8085
142	http://127....	POST	/userloginvalidate	✓		200	2607	HTML		Document			127.0.0.1		15:48:03 7 Oc...	8085
143	http://127....	POST	/userloginvalidate	✓		302	156						127.0.0.1		15:48:28 7 Oc...	8085
151	http://127....	POST	/newuserregister	✓		302	151						127.0.0.1		15:49:05 7 Oc...	8085
153	http://127....	POST	/userloginvalidate	✓		302	156						127.0.0.1		15:49:13 7 Oc...	8085
159	http://127....	POST	/updateuser	✓		302	156						127.0.0.1		15:49:37 7 Oc...	8085
16	https://aa.g...	POST	/u/0/_/gog/get?rt=j&sourceid=538	✓		200	21199	script				✓	142.250.183.238		15:07:52 7 Oc...	8085
24	https://ogs....	POST	/u/0/widget/account?origin=https%3A%2F%2Fwww...	✓		200	59461	HTML				✓	142.250.183.238	SIDCC=AEf-XMTHp...	15:07:56 7 Oc...	8085
28	https://aa.g...	POST	/u/0/_/gog/get?rt=j&sourceid=538	✓		200	21033	text				✓	142.250.183.238		15:07:57 7 Oc...	8085
85	https://aa.g...	POST	/u/0/_/gog/get?rt=j&sourceid=538	✓		200	22103	script				✓	142.250.183.238		15:14:04 7 Oc...	8085
36	https://play...	POST	/log?format=json&hasfast=true&authuser=0	✓		200	1833	JSON				✓	142.250.193.174	NID=511=C7Ycld1...	15:07:58 7 Oc...	8085
42	https://play...	POST	/log?format=json&hasfast=true&authuser=0	✓		200	1334	JSON				✓	142.250.193.174	SIDCC=AEf-XMSl7f...	15:08:02 7 Oc...	8085
91	https://play...	POST	/log?format=json&hasfast=true&authuser=0	✓		200	1334	JSON				✓	142.250.193.174	SIDCC=AEf-XMS_...	15:14:07 7 Oc...	8085
92	https://play...	POST	/log?format=json&hasfast=true&authuser=0	✓		200	1334	JSON				✓	142.250.193.174	SIDCC=AEf-XMTYr...	15:14:07 7 Oc...	8085
95	https://play...	POST	/log?format=json&hasfast=true&authuser=0	✓		200	1334	JSON				✓	142.250.193.174	SIDCC=AEf-XMTyR...	15:14:10 7 Oct...	8085
1	https://sb-...	POST	/safebrowsing/clientreport/download?key=AIzaSyD3...	✓		403	1592	JSON				✓	142.250.195.142		15:06:46 7 Oc...	8085
80	https://ww...	POST	/gen_204?s=webhp&t=cap&atyp=csi&ei=5PQ_Y8P...	✓		204	357	HTML				✓	142.250.195.164		15:14:04 7 Oc...	8085
81	https://ww...	POST	/gen_204?s=webhp&t=aft&atyp=csi&ei=5PQ_Y8P...	✓		204	357	HTML				✓	142.250.195.164		15:14:04 7 Oc...	8085
84	https://ww...	POST	/gen_204?atyp=i&ei=5PQ_Y8PWF6rZ4-EPjpqbuAE...	✓		204	357	HTML				✓	142.250.195.164		15:14:04 7 Oc...	8085
86	https://ww...	POST	/gen_204?atyp=csi&ei=5PQ_Y8PWF6rZ4-EPjpqbuA...	✓		204	357	HTML				✓	142.250.195.164		15:14:04 7 Oc...	8085
90	https://ww...	POST	/gen_204?atyp=i&ei=dfM_Y7yjN8zy4-EPkqyPuAw&...	✓		204	357	HTML				✓	142.250.195.164		15:14:06 7 Oc...	8085
98	https://ww...	POST	/gen_204?atyp=i&ei=5PQ_Y8PWF6rZ4-EPjpqbuAE...	✓		204	357	HTML				✓	142.250.195.164		15:15:09 7 Oc...	8085
99	https://ww...	POST	/gen_204?atyp=i&ei=5PQ_Y8PWF6rZ4-EPjpqbuAE...	✓		204	357	HTML				✓	142.250.195.164		15:15:12 7 Oct...	8085
8	https://ww...	POST	/gen_204?s=webhp&t=aft&atyp=csi&ei=cPM_Y-q-C...	✓		204	357	HTML				✓	172.217.160.132		15:07:52 7 Oc...	8085
13	https://ww...	POST	/gen_204?atyp=i&ei=cPM_Y-q-C8aw4-EP3K-AgAg...	✓		204	357	HTML				✓	172.217.160.132		15:07:52 7 Oc...	8085
14	https://ww...	POST	/gen_204?atyp=csi&ei=cPM_Y-q-C8aw4-EP3K-AgA...	✓		204	357	HTML				✓	172.217.160.132		15:07:52 7 Oc...	8085

Request

```
GET /user/products HTTP/1.1
Host: 127.0.0.1:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101
Firefox/102.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/w
```

Response

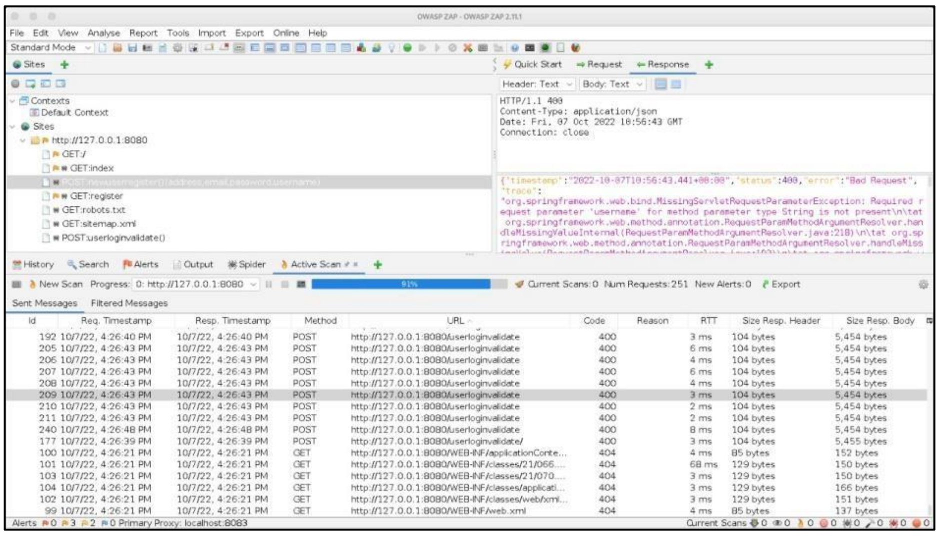
```
HTTP/1.1 200
Content-Type: text/html;charset=UTF-8
Content-Language: en-US
Content-Length: 6588
Date: Fri, 07 Oct 2022 10:19:59 GMT
Connection: close
```

Request Attributes	2
Request Cookies	1
Request Headers	13
Response Headers	5

Figure 4.6: XSS attack

After performing all these attacks, it was confirmed that it is a secure web application and the website was tested on Z Attack Proxy. It is a security testing tool that helps to find potential vulnerabilities in a web application. This tool is the most widely used web scanner in security testing. This tool allows us to do Penetration testing. It is the process of testing our applications for vulnerabilities.

After testing this application in ZAD tool, it shows 91% accuracy with less vulnerabilities. The complete process involves in the development of the secured Spring Boot application. This is capable of defending the several attacks. The solution of this project is to use the Spring Boot framework over other mechanisms to make a more robust and secure application.



Contexts
Default Context
Sites
http://127.0.0.1:8080
GET:/
GET:index
POST:newuserregister()(address,email,password,username)
GET:register
GET:robots.txt
GET:sitemap.xml
POST:userloginvalidate()

```
HTTP/1.1 400
Content-Type: application/json
Date: Fri, 07 Oct 2022 10:56:43 GMT
Connection: close
```

```
{"timestamp":"2022-10-07T10:56:43.441+00:00","status":400,"error":"Bad Request",
"trace":
"org.springframework.web.bind.MissingServletRequestParameterException: Required r
equest parameter 'username' for method parameter type String is not present\n\tat
 org.springframework.web.method.annotation.RequestParamMethodArgumentResolver.han
dleMissingValueInternal(RequestParamMethodArgumentResolver.java:218)\n\tat org.sp
ringframework.web.method.annotation.RequestParamMethodArgumentResolver.handleMiss
```

Progress: 0: http://127.0.0.1:8080 91% Current Scans: 0 Num Requests: 251 New Alerts: 0

Id	Req. Timestamp	Resp. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Header	Size Resp. Body
192	10/7/22, 4:26:40 PM	10/7/22, 4:26:40 PM	POST	http://127.0.0.1:8080/userloginvalidate	400		3 ms	104 bytes	5,454 bytes
205	10/7/22, 4:26:43 PM	10/7/22, 4:26:43 PM	POST	http://127.0.0.1:8080/userloginvalidate	400		6 ms	104 bytes	5,454 bytes
206	10/7/22, 4:26:43 PM	10/7/22, 4:26:43 PM	POST	http://127.0.0.1:8080/userloginvalidate	400		4 ms	104 bytes	5,454 bytes
207	10/7/22, 4:26:43 PM	10/7/22, 4:26:43 PM	POST	http://127.0.0.1:8080/userloginvalidate	400		6 ms	104 bytes	5,454 bytes
208	10/7/22, 4:26:43 PM	10/7/22, 4:26:43 PM	POST	http://127.0.0.1:8080/userloginvalidate	400		4 ms	104 bytes	5,454 bytes
209	10/7/22, 4:26:43 PM	10/7/22, 4:26:43 PM	POST	http://127.0.0.1:8080/userloginvalidate	400		3 ms	104 bytes	5,454 bytes
210	10/7/22, 4:26:43 PM	10/7/22, 4:26:43 PM	POST	http://127.0.0.1:8080/userloginvalidate	400		2 ms	104 bytes	5,454 bytes
211	10/7/22, 4:26:43 PM	10/7/22, 4:26:43 PM	POST	http://127.0.0.1:8080/userloginvalidate	400		2 ms	104 bytes	5,454 bytes
240	10/7/22, 4:26:48 PM	10/7/22, 4:26:48 PM	POST	http://127.0.0.1:8080/userloginvalidate	400		8 ms	104 bytes	5,454 bytes
177	10/7/22, 4:26:39 PM	10/7/22, 4:26:39 PM	POST	http://127.0.0.1:8080/userloginvalidate/	400		3 ms	104 bytes	5,455 bytes
100	10/7/22, 4:26:21 PM	10/7/22, 4:26:21 PM	GET	http://127.0.0.1:8080/WEB-INF/applicationConte...	404		4 ms	85 bytes	152 bytes
101	10/7/22, 4:26:21 PM	10/7/22, 4:26:21 PM	GET	http://127.0.0.1:8080/WEB-INF/classes/21/066....	404		68 ms	129 bytes	150 bytes
103	10/7/22, 4:26:21 PM	10/7/22, 4:26:21 PM	GET	http://127.0.0.1:8080/WEB-INF/classes/21/070....	404		3 ms	129 bytes	150 bytes
104	10/7/22, 4:26:21 PM	10/7/22, 4:26:21 PM	GET	http://127.0.0.1:8080/WEB-INF/classes/applicati...	404		3 ms	129 bytes	166 bytes
102	10/7/22, 4:26:21 PM	10/7/22, 4:26:21 PM	GET	http://127.0.0.1:8080/WEB-INF/classes/web/xml...	404		3 ms	129 bytes	151 bytes
99	10/7/22, 4:26:21 PM	10/7/22, 4:26:21 PM	GET	http://127.0.0.1:8080/WEB-INF/web.xml	404		4 ms	85 bytes	137 bytes

Alerts 0 3 2 0 Primary Proxy: localhost:8083 Current Scans 0 0 0 0 0 0 0 0

Figure 4.7: ZAD tool showing the accuracy of 91%

5. CONCLUSIONS AND FUTURE SCOPE

The detection of vulnerabilities in Microservices and API are examined in this study. The main goal of this research is to detect the vulnerabilities and attacks on the microservices. In this project, we performed various attacks for finding the vulnerabilities. The technologies like Spring Boot framework helps in the development of the Applications. The project enables us to clearly understand all the microservices that were embedded into on application and how the API are handled here, the microservices are often effected by some attacks that can even destroy the whole application this project helps the developers to develop the further microservices in such a way that no attackers find a loop to enter into the secured gateway that has been deployed using the many security measures that were taken to prevent the unauthorized access.

Appendix:

Pom.xml:

```
 <?xml version="1.0" encoding="UTF-8"?>
 <project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"xsi:schemaLocation="http://maven.apache.org/POM
/4.0.0   https://maven.apache.org/xsd/maven-
4.0.0.xsd"><modelVersion>4.0.0</modelVersion>
<parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.6.4</version>
        <relativePath/> <!-- lookup parent from repository -->
</parent>
<groupId>com.jtspringproject</groupId>
          <artifactId>JtSpringProject</artifactId>
            <version>0.0.1-SNAPSHOT</version>
            <name>JtSpringProject</name>
            <description>Spring project for Java Technology</description>
            <properties>
          <java.version>11</java.version>
</properties>
<dependencies>
<dependency>
          <groupId>org.springframework.boot</groupId>
          <artifactId>spring-boot-starter-web</artifactId>
</dependency>
<dependency>
           <groupId>org.springframework.boot</groupId>
          <artifactId>spring-boot-starter-test</artifactId>
<scope>test</scope>
</dependency>
<dependency>
          <groupId>org.springframework.boot</groupId>
          <artifactId>spring-boot-devtools</artifactId>
</dependency>
<dependency>
         <groupId>javax.servlet</groupId>
         <artifactId>jstl</artifactId>
</dependency>
<dependency>
        <groupId>org.apache.tomcat.embed</groupId>
        <artifactId>tomcat-embed-jasper</artifactId>
</dependency>
<dependency>
     <groupId>mysql</groupId>
     <artifactId>mysql-connector-java</artifactId>
     <version>8.0.28</version>
</dependency>
</dependencies>
<build>
<plugins>
<plugin>
        <groupId>org.springframework.boot</groupId>
       <artifactId>spring-boot-maven-plugin</artifactId>
</plugin>
</plugins>
</build>
</project>
```

mvnw.cmd[Docker Image]:

```
@REM Begin all REM lines with '@' in case MAVEN_BATCH_ECHO is 'on'
@echo off
@REM set title of command window
title %0
@REM enable echoing by setting MAVEN_BATCH_ECHO to 'on'
@if "%MAVEN_BATCH_ECHO%" == "on"  echo %MAVEN_BATCH_ECHO%
@REM set %HOME% to equivalent of $HOME
if "%HOME%" == "" (set "HOME=%HOMEDRIVE%%HOMEPATH%")
@REM Execute a user defined script before this one if
not "%MAVEN_SKIP_RC%" == "" goto skipRcPre
@REM check for pre script, once with legacy .bat ending and once with .cmd ending
if exist "%USERPROFILE%\mavenrc_pre.bat" call "%USERPROFILE%\mavenrc_pre.bat" %*
if exist "%USERPROFILE%\mavenrc_pre.cmd" call "%USERPROFILE%\mavenrc_pre.cmd"
%*:skipRcPre@setlocal
set ERROR_CODE=0
@REM To isolate internal variables from possible post scripts, we use another setlocal
@setlocal
@REM ==== START VALIDATION ====
if not "%JAVA_HOME%" == "" goto OkJHomeecho.
echo Error: JAVA_HOME not found in your environment. >&2
echo Please set the JAVA_HOME variable in your environment to match the >&2
echo location of your Java installation. >&2
echo.goto error:OkJHome
if exist "%JAVA_HOME%\bin\java.exe" goto initecho. echo Error: JAVA_HOME is set to an invalid
directory. >&2echo JAVA_HOME = "%JAVA_HOME%" >&2
```

```
 echo Please set the JAVA_HOME variable in your environment to match the >&2
 echo location of your Java installation. >&2echo.
 goto error
 @REM ==== END VALIDATION ====:init
 @REM Find the project base dir, i.e. the directory that contains the folder ".mvn".
 @REM Fallback to current working directory if not found.
 set MAVEN_PROJECTBASEDIR=%MAVEN_BASEDIR%
 IF NOT "%MAVEN_PROJECTBASEDIR%"=="" goto endDetectBaseDir
set EXEC_DIR=%CD%
 set WDIR=%EXEC_DIR%:findBaseDir
IF EXIST "%WDIR%"\.mvn goto baseDirFound
cd ..
IF "%WDIR%"=="%CD%" goto baseDirNotFound
set WDIR=%CD%
goto findBaseDir:baseDirFound
set MAVEN_PROJECTBASEDIR=%WDIR%
cd "%EXEC_DIR%"
goto endDetectBaseDir:baseDirNotFound
set MAVEN_PROJECTBASEDIR=%EXEC_DIR%
cd "%EXEC_DIR%":endDetectBaseDir
IF NOT EXIST "%MAVEN_PROJECTBASEDIR%\.mvn\jvm.config" goto endReadAdditionalConfig
@setlocal EnableExtensions EnableDelayedExpansion
for /F "usebackq delims=" %%a in ("%MAVEN_PROJECTBASEDIR%\.mvn\jvm.config") do set
JVM_CONFIG_MAVEN_PROPS=!JVM_CONFIG_MAVEN_PROPS! %%a@endlocal & set
JVM_CONFIG_MAVEN_PROPS=%JVM_CONFIG_MAVEN_PROPS%:endReadAdditionalConfigSET
MAVEN_JAVA_EXE="%JAVA_HOME%\bin\java.exe"
set WRAPPER_JAR="%MAVEN_PROJECTBASEDIR%\.mvn\wrapper\maven-wrapper.jar"
set WRAPPER_LAUNCHER=org.apache.maven.wrapper.MavenWrapperMain
setDOWNLOAD_URL=https://repo.maven.apache.org/maven2/org/apache/maven/wrapper/maven-
wrapper/3.1.0/maven-wrapper-3.1.0.jar
FOR   /F    "usebackq    tokens=1,2    delims=="    %%A  IN
("%MAVEN_PROJECTBASEDIR%\.mvn\wrapper\maven-wrapper.properties") DO (

  IF "%%A"=="wrapperUrl" SET DOWNLOAD_URL=%%B

)
@REM Extension to allow automatically downloading the maven-wrapper.jar from Maven-central
@REM This allows using the maven wrapper in projects that prohibit checking in binary data.
 if exist %WRAPPER_JAR% (
     if "%MVNW_VERBOSE%" == "true" (
           echo Found %WRAPPER_JAR%)
     )
else (
    if not "%MVNW_REPOURL%" == "" (SET
        DOWNLOAD_URL="%MVNW_REPOURL%/org/apache/maven/wrapper/maven-
wrapper/3.1.0/maven-wrapper-3.1.0.jar"

    )
if "%MVNW_VERBOSE%" == "true" (
echo Couldn't find %WRAPPER_JAR%, downloading it ...echo
Downloading from: %DOWNLOAD_URL%
    )   powershell -Command "&{"^
                        "$webclient = new-object System.Net.WebClient;"^
"if(-not([string]::IsNullOrEmpty('%MVNW_USERNAME%')-and
[string]::IsNullOrEmpty('%MVNW_PASSWORD%'))) {"^"$webclient.Credentials=new-object
System.Net.NetworkCredential('%MVNW_USERNAME%', '%MVNW_PASSWORD%');"^

"}"^
"[Net.ServicePointManager]::SecurityProtocol=[Net.SecurityProtocolType]::Tls12;$webclient.DownloadFil
e('%DOWNLOAD_URL%', '%WRAPPER_JAR%')"^
"}"
if "%MVNW_VERBOSE%" == "true" (
       echo Finished downloading %WRAPPER_JAR%
    )
 )
@REM End of extension
@REM Provide a "standardized" way to retrieve the CLI args that will
@REM work with both Windows and non-Windows executions.
set MAVEN_CMD_LINE_ARGS=%*
%MAVEN_JAVA_EXE% ^
%JVM_CONFIG_MAVEN_PROPS% ^
%MAVEN_OPTS% ^
%MAVEN_DEBUG_OPTS% ^-classpath %WRAPPER_JAR% ^
"-Dmaven.multiModuleProjectDirectory=%MAVEN_PROJECTBASEDIR%"
^%WRAPPER_LAUNCHER% %MAVEN_CONFIG% %*
if ERRORLEVEL 1 goto error
goto end:error
set ERROR_CODE=1:end
@endlocal & set ERROR_CODE=%ERROR_CODE%
if not "%MAVEN_SKIP_RC%"=="" goto skipRcPost
@REM check for post script, once with legacy .bat ending and once with .cmd ending
if exist "%USERPROFILE%\mavenrc_post.bat" call "%USERPROFILE%\mavenrc_post.bat"

if exist "%USERPROFILE%\mavenrc_post.cmd" call "%USERPROFILE%\mavenrc_post.cmd"
:skipRcPost
@REM pause the script if MAVEN_BATCH_PAUSE is set to 'on'
if "%MAVEN_BATCH_PAUSE%"=="on" pause
 if "%MAVEN_TERMINATE_CMD%"=="on" exit %ERROR_CODE%
```

```
cmd /C exit /B %ERROR_CODE%
```

JtSpringProjectApplication.java:

```
package com.jtspringproject.JtSpringProject;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
@SpringBootApplication
public class JtSpringProjectApplication { public
static void main(String[] args) {
SpringApplication.run(JtSpringProjectApplicati
on.class, args);
    }
}
```

AdminController.java:

```
package com.jtspringproject.JtSpringProject.controller;
import java.sql.*;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;
import com.mysql.cj.protocol.Resultset;
@Controller
 public class AdminController {
        int adminlogcheck = 0;
        String usernameforclass = "";
        @RequestMapping(value = {"/","/logout"})
        public String returnIndex() {
                adminlogcheck =0;
                usernameforclass = "";
                return "userLogin";
        }
 @GetMapping("/index")
        public String index(Model model) {
                if(usernameforclass.equalsIgnoreCase(""))
                        return "userLogin";
                else {
                        model.addAttribute("username", usernameforclass);
                        return "index";
                }
    }

@GetMapping("/userloginvalidate") public String
userlog(Model model) {
                return "userLogin";}
```

```
@RequestMapping(value = "userloginvalidate", method = RequestMethod.POST)

Public String userlogin(@RequestParam("username")
String username,@RequestParam("password") String pass,Model model) {
  try
  {
         Class.forName("com.mysql.jdbc.Driver");
         Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
Statement stmt = con.createStatement();
ResultSet rst = stmt.executeQuery("select * from users where username ='"+username+"' and password
= '"+ pass+"' ;");
if(rst.next()) {
         usernameforclass = rst.getString(2);\
         return "redirect:/index";
              }
else {
        model.addAttribute("message", "Invalid Username or Password");return "userLogin";
        }
}
catch(Exception e)
{
System.out.println("Exception:"+e);
}
return "userLogin";
}
@GetMapping("/admin")
  public String adminlogin(Model model) {
      return "adminlogin";
}
@GetMapping("/adminhome")
public String adminHome(Model model) {
if(adminlogcheck!=0)
return "adminHome";
else
return "redirect:/admin";
}
@GetMapping("/loginvalidate")
public String adminlog(Model model) {
return "adminlogin";
}
  @RequestMapping(value = "loginvalidate", method = RequestMethod.POST)public      String
         adminlogin(    @RequestParam("username")          String  username,@RequestParam("password")
  String pass,Model model) {if(username.equalsIgnoreCase("admin") && pass.equalsIgnoreCase("123")) {
  adminlogcheck=1;
    return "redirect:/adminhome";
  }
  else {
      model.addAttribute("message", "Invalid Username or Password");return
  "adminlogin";
      }
  }
  @GetMapping("/admin/categories")public
  String getcategory() {
  return "categories";}

  @RequestMapping(value = "admin/sendcategory",method = RequestMethod.GET)public
  String addcategorytodb(@RequestParam("categoryname") String catname)
  {
         try
  {
      Class.forName("com.mysql.jdbc.Driver");
  Connection con =
  DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");Statement stmt =
  con.createStatement();
   PreparedStatement pst = con.prepareStatement("insert into categories(name) values(?);");
  pst.setString(1,catname);
  int i = pst.executeUpdate();
  }
  catch(Exception e)
  {
   System.out.println("Exception:"+e);
  }
  return "redirect:/admin/categories";
  }
  @GetMapping("/admin/categories/delete")
  public String removeCategoryDb(@RequestParam("id") int id)
  {
  try
{
      Class.forName("com.mysql.jdbc.Driver");
      Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
Statement stmt = con.createStatement();
PreparedStatement pst = con.prepareStatement("delete from categories where categoryid = ? ;");
pst.setInt(1, id);
int i = pst.executeUpdate();
}
```

```
catch(Exception e)
{
  System.out.println("Exception:"+e);
}
return "redirect:/admin/categories";
}
@GetMapping("/admin/categories/update")
public String updateCategoryDb(@RequestParam("categoryid") int id,
@RequestParam("categoryname") String categoryname){
try
{
     Class.forName("com.mysql.jdbc.Driver");
     Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
     Statement stmt = con.createStatement();
     PreparedStatement pst = con.prepareStatement("update categories set name = ?where categoryid = ?");
    pst.setString(1, categoryname);pst.setInt(2, id);
                          int i = pst.executeUpdate();
}
catch(Exception e)
                  {
                   System.out.println("Exception:"+e);
                  }
                  return "redirect:/admin/categories";
          }
@GetMapping("/admin/products")
public String getproduct(Model model) {return
"products";}
@GetMapping("/admin/products/add") public String
addproduct(Model model) {
                  return "productsAdd";}
@GetMapping("/admin/products/update")
public String updateproduct(@RequestParam("pid") int id,Model model) {String
pname,pdescription,pimage;
int pid,pprice,pweight,pquantity,pcategory;try
{
   Class.forName("com.mysql.jdbc.Driver");
   Connection con =DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
   Statement stmt = con.createStatement(); Statement stmt2 = con.createStatement();
   ResultSet rst = stmt.executeQuery("select * from products where id = "+id+";");if(rst.next())
{
                          pid = rst.getInt(1);
                          pname = rst.getString(2);
```

```
                                pimage
                                =rst.getString(3);
                                pcategory = rst.getInt(4);
                                pquantity = rst.getInt(5);
                                pprice = rst.getInt(6);
                                pweight =  rst.getInt(7);
                                pdescription = rst.getString(8);
                                model.addAttribute("pid",pid);
                                model.addAttribute("pname",pname);
                                model.addAttribute("pimage",pimage);
                                ResultSet rst2 = stmt.executeQuery("select * from categories where categoryid =
 "+pcategory+";");
 if(rst2.next())
 {
        model.addAttribute("pcategory",rst2.getString(2));
}

model.addAttribute("pquantity",pquantity);
model.addAttribute("pprice",pprice);
model.addAttribute("pweight",pweight);
model.addAttribute("pdescription",pdescription);
                                }
}
                        catch(Exception e)
                        {
                                System.out.println("Exception:"+e);
                        }
                        return "productsUpdate";
          }
@RequestMapping(value = "admin/products/updateData",method=RequestMethod.POST)
public String updateproducttodb(@RequestParam("id") int id,@RequestParam("name")
String name, @RequestParam("price") int price, @RequestParam("weight") int weight,
@RequestParam("quantity") int quantity, @RequestParam("description") String description,
@RequestParam("productImage") String picture ){
                        try
                        {
                                Class.forName("com.mysql.jdbc.Driver");
 Connection  con=DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
PreparedStatement pst = con.prepareStatement("update products set name= ?,image
 = ?,quantity = ?, price = ?, weight = ?,description = ? where id = ?;");
                                pst.setString(1, name);
```

```
                pst.setString(2, picture);
                pst.setInt(3, quantity);
                pst.setInt(4, price);
                pst.setInt(5, weight);
                pst.setString(6, description);
                pst.setInt(7, id);
                int i = pst.executeUpdate();
            }
            catch(Exception e)
            {
                System.out.println("Exception:"+e);
            }
            return "redirect:/admin/products";
        }
        @GetMapping("/admin/products/delete")
        public String removeProductDb(@RequestParam("id") int id)
        {
        try
   {
      Class.forName("com.mysql.jdbc.Driver");
    Connection con=
DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");PreparedStatement
pst = con.prepareStatement("delete from products where id = ? ;");pst.setInt(1, id);
                int i = pst.executeUpdate();
                }
            catch(Exception e)
            {
                System.out.println("Exception:"+e);
            }
            return "redirect:/admin/products";
        }
        @PostMapping("/admin/products")
        public String postproduct() {
            return "redirect:/admin/categories";
        }
        @RequestMapping(value = "admin/products/sendData",method=RequestMethod.POST)
        public String addproducttodb(@RequestParam("name")
        String name, @RequestParam("categoryid") String catid, @RequestParam("price") int
 price, @RequestParam("weight") int weight, @RequestParam("quantity") int quantity,
 @RequestParam("description") String description, @RequestParam("productImage") String picture ) {
            try
            {
                Connection
 con=DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
```

```
                        Statement stmt = con.createStatement();
 ResultSet rs = stmt.executeQuery("select * from categories where name =
 '"+catid+"';");
if(rs.next())
{
int categoryid = rs.getInt(1);
PreparedStatement pst=con.prepareStatement("insert      into
products(name,image,categoryid,quantity,price,weight,description) values(?,?,?,?,?,?,?);");
pst.setString(1,name);
pst.setString(2, picture);
pst.setInt(3, categoryid);
pst.setInt(4, quantity);
pst.setInt(5, price);
pst.setInt(6, weight);
pst.setString(7, description);
int i = pst.executeUpdate();
                        }
                }
catch(Exception e)
{
System.out.println("Exception:"+e);
                }
                return "redirect:/admin/products";
        }
        @GetMapping("/admin/customers")
        public String getCustomerDetail() {
                return "displayCustomers";
        }
@GetMapping("profileDisplay")
        public String profileDisplay(Model model) {
                String displayusername,displaypassword,displayemail,displayaddress;
                try
                {
  Class.forName("com.mysql.jdbc.Driver");
Connection con=
DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");Statement stmt =
con.createStatement();
 ResultSet rst = stmt.executeQuery("select * from users where username ='"+usernameforclass+"';");\
                        if(rst.next())
                    {
                        int userid = rst.getInt(1);
                        displayusername = rst.getString(2);
```

```
                                displayemail = rst.getString(3);
                                displaypassword = rst.getString(4);
                                displayaddress = rst.getString(5);
                                model.addAttribute("userid",userid);
                                model.addAttribute("username",displayusername);
                                model.addAttribute("email",displayemail);
                                model.addAttribute("password",displaypassword);
                                model.addAttribute("address",displayaddress);
                                }
                        }
 catch(Exception e)
{
System.out.println("Exception:"+e);
                        }
                        System.out.println("Hello");
                        return "updateProfile";
                }
@RequestMapping(value = "updateuser",method=RequestMethod.POST)
                public String updateUserProfile(@RequestParam("userid") int
 userid,@RequestParam("username") String username, @RequestParam("email") String email,
 @RequestParam("password") String password, @RequestParam("address") String address)

                {
try
{
Class.forName("com.mysql.jdbc.Driver");
Connection con
=DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
 PreparedStatement pst = con.prepareStatement("update users set username= ?,email = ?,password= ?,
 address= ? where uid = ?;");

                                pst.setString(1, username);
                                pst.setString(2, email);
                                pst.setString(3, password);
                                pst.setString(4, address);
                                pst.setInt(5, userid);
                                int i = pst.executeUpdate();
                                usernameforclass = username;
                        }
catch(Exception e)
{
System.out.println("Exception:"+e);
}
return "redirect:/index";
```

```
}
 }
```

UserController.java:

```
 package com.jtspringproject.JtSpringProject.controller;
 import java.sql.*;
 import org.springframework.stereotype.Controller;
 import org.springframework.ui.Model;
 import org.springframework.web.bind.annotation.*;
 import com.mysql.cj.protocol.Resultset;
 @Controller
 public class AdminController {
         int adminlogcheck = 0;
         String usernameforclass = "";
         @RequestMapping(value = {"/","/logout"})
         public String returnIndex() {
                 adminlogcheck =0;
                 usernameforclass = "";
                 return "userLogin";
         }
@GetMapping("/index")
         public String index(Model model) {
                 if(usernameforclass.equalsIgnoreCase(""))
                         return "userLogin";
                 else {
                         model.addAttribute("username", usernameforclass);
                         return "index";
                 }
         }
         @GetMapping("/userloginvalidate")
         public String userlog(Model model) {
                 return "userLogin";
         }
         @RequestMapping(value = "userloginvalidate", method = RequestMethod.POST)
         public String userlogin( @RequestParam("username") String username,
 @RequestParam("password") String pass,Model model) {
         try
                 {
  Class.forName("com.mysql.jdbc.Driver");
 Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
Statement stmt = con.createStatement();
                         ResultSet rst = stmt.executeQuery("select * from users where username =
 '"+username+"' and password = '"+ pass+"' ;");
```

```
                if(rst.next()) {
                        usernameforclass = rst.getString(2);
                        return "redirect:/index";
                        }
                else {
                        model.addAttribute("message", "Invalid Username or Password");
                        return "userLogin";
                }
        }
        catch(Exception e)
        {
                System.out.println("Exception:"+e);
        }
        return "userLogin";
    }
    @GetMapping("/admin")
    public String adminlogin(Model model)
        return "adminlogin";
    }
    @GetMapping("/adminhome")
    public String adminHome(Model model) {
        if(adminlogcheck!=0)
                return "adminHome";
        else
                return "redirect:/admin";
    }
    @GetMapping("/loginvalidate")
    public String adminlog(Model model) {
        return "adminlogin";
    }
    @RequestMapping(value = "loginvalidate", method = RequestMethod.POST)
    public String adminlogin( @RequestParam("username") String username,
 @RequestParam("password") String pass,Model model) {
        if(username.equalsIgnoreCase("admin") && pass.equalsIgnoreCase("123")) {
                adminlogcheck=1;
                return "redirect:/adminhome";
}
        else {
                model.addAttribute("message", "Invalid Username or Password");
                return "adminlogin";
        }
    }
```

```
@GetMapping("/admin/categories")public String
getcategory() {
                return "categories";
}
@RequestMapping(value = "admin/sendcategory",method = RequestMethod.GET)
        public String addcategorytodb(@RequestParam("categoryname") String catname)
        {
                try
                {
                        Class.forName("com.mysql.jdbc.Driver");
                        Connection con=
 DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
                        Statement stmt = con.createStatement();
                        PreparedStatement pst = con.prepareStatement("insert into categories(name)
 values(?);");
pst.setString(1,catname); int i =
pst.executeUpdate();
                        }
                catch(Exception e)
                {
                        System.out.println("Exception:"+e);
                }
                return "redirect:/admin/categories";
        }
@GetMapping("/admin/categories/delete")
        public String removeCategoryDb(@RequestParam("id") int id)
        {
try
                {
Class.forName("com.mysql.jdbc.Driver");
                        Connection con =
 DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
                        Statement stmt = con.createStatement();
                        PreparedStatement pst = con.prepareStatement("delete from categories where
 categoryid = ? ;");
 pst.setInt(1, id);
int i = pst.executeUpdate();
}
catch(Exception e)
                {
System.out.println("Exception:"+e);
```

```
                        }
            return "redirect:/admin/categories";
}
@GetMapping("/admin/categories/update")
            public          String          updateCategoryDb(@RequestParam("categoryid")          int          id,
 @RequestParam("categoryname") String categoryname)

            {
try
{
  Class.forName("com.mysql.jdbc.Driver");
  Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
  Statement stmt = con.createStatement();
 PreparedStatement pst = con.prepareStatement("update categories set name = ?where categoryid = ?");
 pst.setString(1, categoryname);pst.setInt(2, id);
 int i = pst.executeUpdate();
                        }
                        catch(Exception e)
                        {
                                    System.out.println("Exception:"+e);
                        }
                        return "redirect:/admin/categories";
            }
            @GetMapping("/admin/products")
            public String getproduct(Model model) {
                        return "products";
      }
 @GetMapping("/admin/products/add") public
String addproduct(Model model) {
  return "productsAdd";
}
@GetMapping("/admin/products/update")
 public String updateproduct(@RequestParam("pid") int id,Model model) {String
pname,pdescription,pimage;
                        int pid,pprice,pweight,pquantity,pcategory;
                        try
                        {
Class.forName("com.mysql.jdbc.Driver");
Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
Statement stmt = con.createStatement(); Statement stmt2 = con.createStatement();
ResultSet rst = stmt.executeQuery("select * from products where id = "+id+";");
if(rst.next()){
                                    pid = rst.getInt(1);
                                    pname = rst.getString(2);
```

```
                    pimage = rst.getString(3);
                    pcategory = rst.getInt(4);
                    pquantity = rst.getInt(5);
                    pprice = rst.getInt(6);
                    pweight =  rst.getInt(7);
                    pdescription = rst.getString(8);
                    model.addAttribute("pid",pid);
                    model.addAttribute("pname",pname);
                    model.addAttribute("pimage",pimage);
                    ResultSet rst2 = stmt.executeQuery("select * from categories where categoryid =
 "+pcategory+";");
if(rst2.next())
                    {
                         model.addAttribute("pcategory",rst2.getString(2));
                    }
                    model.addAttribute("pquantity",pquantity);
                    model.addAttribute("pprice",pprice);
                    model.addAttribute("pweight",pweight);
                    model.addAttribute("pdescription",pdescription);
                    }}
               catch(Exception e)
               {
                    System.out.println("Exception:"+e);
               }
               return "productsUpdate";
          }
          @RequestMapping(value = "admin/products/updateData",method=RequestMethod.POST)
          public String updateproducttodb(@RequestParam("id") int id,@RequestParam("name") String
 name, @RequestParam("price") int price, @RequestParam("weight") int weight,
 @RequestParam("quantity") int quantity, @RequestParam("description") String description,
 @RequestParam("productImage") String picture ){
               try
               {
                    Class.forName("com.mysql.jdbc.Driver");
                    Connection con=
 DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");PreparedStatement
 pst = con.prepareStatement("update products set name= ?,image= ?,quantity = ?, price = ?, weight =
 ?,description = ? where id = ?;");pst.setString(1, name);
                    pst.setString(2, picture);
                    pst.setInt(3, quantity);
                    pst.setInt(4, price);
                    pst.setInt(5, weight);
                    pst.setString(6, description);
                    pst.setInt(7, id);
                    int i = pst.executeUpdate();
}
catch(Exception e)
            {
                    System.out.println("Exception:"+e);
            }
            return "redirect:/admin/products";
        }
        @GetMapping("/admin/products/delete")
        public String removeProductDb(@RequestParam("id") int id)
        {
try
            {
                    Class.forName("com.mysql.jdbc.Driver");
 Connectio con = DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
 PreparedStatement pst = con.prepareStatement("delete from products where id = ? ;");
                    pst.setInt(1, id);
                    int i = pst.executeUpdate();
                    }
            catch(Exception e)
            {
                    System.out.println("Exception:"+e);
            }
            return "redirect:/admin/products";
        }
        @PostMapping("/admin/products")
        public String postproduct() {
return "redirect:/admin/categories";
        }
        @RequestMapping(value = "admin/products/sendData",method=RequestMethod.POST)
        public String addproducttodb(@RequestParam("name") String name,
 @RequestParam("categoryid") String catid, @RequestParam("price") int price,
 @RequestParam("weight") int weight, @RequestParam("quantity") int quantity,
 @RequestParam("description") String description, @RequestParam("productImage") String picture ) {
        try
            {
Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
Statement stmt = con.createStatement()
ResultSet rs = stmt.executeQuery("select * from categories where name =
 '"+catid+"';");
```

```
if(rs.next())
{
int categoryid = rs.getInt(1);
                    PreparedStatement pst = con.prepareStatement("insert into
 products(name,image,categoryid,quantity,price,weight,description) values(?,?,?,?,?,?,?);");
                    pst.setString(1,name); pst.setString(2, picture); pst.setInt(3, categoryid);
 pst.setInt(4, quantity); pst.setInt(5, price); pst.setInt(6, weight); pst.setString(7, description);int i =
 pst.executeUpdate();
                    }

              }
              catch(Exception e)
              {
                    System.out.println("Exception:"+e);
              }
              return "redirect:/admin/products";
       }
       @GetMapping("/admin/customers")
       public String getCustomerDetail() {
              return "displayCustomers";
       }
 @GetMapping("profileDisplay")
       public String profileDisplay(Model model) {
              String displayusername,displaypassword,displayemail,displayaddress;
              try
              {
Class.forName("com.mysql.jdbc.Driver");
Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
Statement stmt = con.createStatement();
 ResultSet rst = stmt.exec
 uteQuery("select * from users where username ='"+usernameforclass+"';");if(rst.next())
                    {
                    int userid = rst.getInt(1);
                    displayusername = rst.getString(2);
                    displayemail = rst.getString(3);
                    displaypassword = rst.getString(4);
                    displayaddress = rst.getString(5);
                    model.addAttribute("userid",userid);
                    model.addAttribute("username",displayusername);
                    model.addAttribute("email",displayemail);
                    model.addAttribute("password",displaypassword);
                    model.addAttribute("address",displayaddress);
                    }
                    }
catch(Exception e)
                    {
                              System.out.println("Exception:"+e);
                    }
                    System.out.println("Hello");
                    return "updateProfile";
          }
          @RequestMapping(value = "updateuser",method=RequestMethod.POST)
          public String updateUserProfile(@RequestParam("userid") int
 userid,@RequestParam("username") String username, @RequestParam("email") String email,
 @RequestParam("password") String password, @RequestParam("address") String address)
{
try
{
 Class.forName("com.mysql.jdbc.Driver");
  Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/springproject","root","");
  PreparedStatement pst = con.prepareStatement("update users set username= ?,email= ?,password= ?,
  address= ? where uid = ?;");
 pst.setString(1, username);pst.setString(2, email);
pst.setString(3, password);pst.setString(4,
address); pst.setInt(5, userid);
int i = pst.executeUpdate();
usernameforclass = username;
                    }
catch(Exception e)
{
  System.out.println("Exception:"+e);
}
return "redirect:/index";
}
  }
```

Application.properties:

```
spring.mvc.view.prefix=/views/spring.mvc.view.suffix=.jsp
```

REFERENCES

[1] Vayghan, L. A., Saied, M. A., Toeroe, M., & Khendek, F. (2021). A Kubernetes controller for managing the availability of elastic microservice based stateful applications. *Journal of Systems and Software*, *175*, 110924.

[2] Torkura, K. A., Sukmana, M. I., Kayem, A. V., Cheng, F., & Meinel, C. (2018, December). A cyber risk based moving target defense mechanism for microservice architectures, *2018 IEEE*

[3] Northern, B., Burks, T., Hatcher, M., Rogers, M., & Ulybyshev, D. (2021). VERCASM-CPS: Vulnerability Analysis and Cyber Risk Assessment for Cyber-Physical Systems. *Information*, *12*(10), 408.

[4] Yarygina, T., & Otterstad, C. (2018, June). A game of microservices: Automated intrusion response. In *IFIP International Conference on Distributed Applications and Interoperable Systems* (pp. 169-177). Springer, Cham.

[5] Akbulut, A., & Perros, H. G. (2019, June). Software versioning with microservices through the API gateway design pattern. In *2019 9th International Conference on Advanced Computer Information Technologies (ACIT)* (pp. 289-292). IEEE.

[6] Donham, J. (2018, September). A domain-specific language for microservices. In *Proceedings of the 9th ACM SIGPLAN International Symposium on Scala* (pp. 2-12).

[7] Somashekar, G., & Gandhi, A. (2021, April). Towards optimal configuration of microservices. In *Proceedings of the 1st Workshop on Machine Learning and Systems* (pp. 7-14).

[8] Baarzi, A. F., Kesidis, G., Fleck, D., & Stavrou, A. (2020, April). Microservices made attack-resilient using unsupervised service fissioning. In *Proceedings of the 13th European workshop on Systems Security* (pp. 31-36).

[9] Mateus-Coelho, N., Cruz-Cunha, M., & Ferreira, L. G. (2021). Security in microservices architectures. *Procedia Computer Science*, *181*, 1225-1236.

[10] Torkura, K. A., Sukmana, M. I., Cheng, F., & Meinel, C. (2018, August). Cavas: Neutralizing application and container security vulnerabilities in the cloud native era. In *International Conference on Security and Privacy in Communication Systems* (pp. 471-490). Springer, Cham.

[11] Heinrich, R., Van Hoorn, A., Knoche, H., Li, F., Lwakatare, L. E., Pahl, C., ... & Wettinger, J. (2017, April). Performance engineering for microservices: research challenges and directions. In *Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering Companion* (pp. 223-226).

[12] Camilli, M., Guerriero, A., Janes, A., Russo, B., & Russo, S. (2022, May). Microservices integrated performance and reliability testing. In *Proceedings of the 3rd ACM/IEEE International Conference on Automation of Software Test* (pp. 29-39).

[13] de Almeida, M. G., & Canedo, E. D. (2022). Authentication and Authorization in Microservices Architecture: A Systematic Literature Review. *Applied Sciences*, *12*(6), 3023.

[14] Hannousse, A., & Yahiouche, S. (2021). Securing microservices and microservice architectures: A systematic mapping study. *Computer Science Review*, *41*, 100415.

[15] Di Francesco, P., Lago, P., & Malavolta, I. (2019). Architecting with microservices: A systematic mapping study. *Journal of Systems and Software*, *150*, 77-97.

[16] Leila Abdollahi Vayghan,Mohamed Aymen Saied, Maria Toeroe, Ferhat Khendek,published in 2019.DOI:- 10.1109/QRS.2019.00034.Microservice Based Architecture: Towards High-Availability for StatefulApplications with Kubernetes.

[17] Isil Karabey Aksakalli , Turgay celik , Ahmet Burak Can , Bedir Tekinerdogan , Recevied in revised form 3 may 2021,Accepted 19 may 2021,Available online 7 june 2021.

[18] Salibindla, J. (2018). Microservices API security. *International Journal of Engineering Research & Technology*, *7*(1), 277-281.

[19] Janes, A., & Russo, B. (2019, October). Automatic performance monitoring and regression testing during the transition from monolith to microservices. In *2019 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)* (pp. 163-168). IEEE.

[20] Nacha Chondamrongkul, Jing Sun and Ian Warren. Published in 2020. IEEE. DOI:- 10.1109/ICSA-CS50368.2020.00024. Automated Security Analysis for Microservice Architecture.

[21] Nguyen, Q., & Baker, O. F. (2019). Applying Spring Security Framework and OAuth2 To Protect Microservice Architecture API. *J. Softw.*, *14*(6), 257-264.

[22] Flora, J. (2020, October). Improving the security of microservice systems by detecting and tolerating intrusions. In *2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)* (pp. 131-134). IEEE.

[23] Sun, Y., Nanda, S., & Jaeger, T. (2015, November). Security-as-a-service for microservices-based cloud applications. In *2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)* (pp. 50-57). IEEE.

[24] Zaheer, Z., Chang, H., Mukherjee, S., & Van der Merwe, J. (2019, April). eztrust: Network-independent zero-trust perimeterization for microservices. In *Proceedings of the 2019 ACM Symposium on SDN Research* (pp. 49-61).

[25] Wang, Y., Kadiyala, H., & Rubin, J. (2021). Promises and challenges of microservices: an exploratory study. *Empirical Software Engineering*, *26*(4), 1-44.

[26] Daoud, M., El Mezouari, A., Faci, N., Benslimane, D., Maamar, Z., & El Fazziki, A. (2021). A multi-model based microservices identification approach. *Journal of Systems Architecture*, *118*, 102200.

[27] Vayghan, L. A., Saied, M. A., Toeroe, M., & Khendek, F. (2019). Kubernetes as an availability manager for microservice applications. *arXiv preprint arXiv:1901.04946*.

[28] Vayghan, L. A., Saied, M. A., Toeroe, M., & Khendek, F. (2018, July). Deploying microservice based applications with kubernetes: Experiments and lessons learned. In *2018 IEEE 11th international conference on cloud computing (CLOUD)* (pp. 970-973). IEEE.

[29] Pereira-Vale, A., Fernandez, E. B., Monge, R., Astudillo, H., & Márquez, G. (2021). Security in microservice-based systems: A multivocal literature review. *Computers & Security*, *103*, 102200.

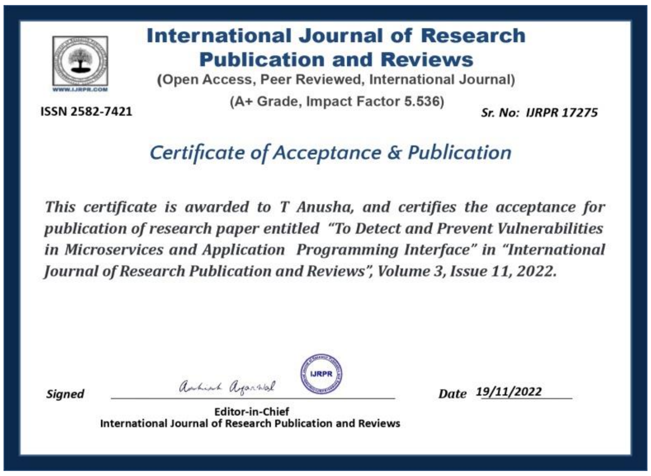
[30] Arzo, S. T., Scotece, D., Bassoli, R., Barattini, D., Granelli, F., Foschini, L., & Fitzek, F. H. (2022). MSN: A Playground Framework for Design and Evaluation of MicroServices-Based sdN Controller. *Journal of Network and Systems Management*, *30*(1), 1-31.

LIST OF PUBLICATIONS

T Anusha, G Naga Manmitha, Sharon Kota, S Laxman Rao, N Ramesh Kumar, P Ganesh, To Detect and Prevent Vulnerabilities in Microservices and Application Programming Interface, International Journal of Research Publication and Reviews, Volume 3 Issue no 11, November 2022.

PAPER LINK: https://ijrpr.com/uploads/V3ISSUE11/IJRPR8061.pdf

CERTIFICATES:



International Journal of Research Publication and Reviews
(Open Access, Peer Reviewed, International Journal)
(A+ Grade, Impact Factor 5.536)

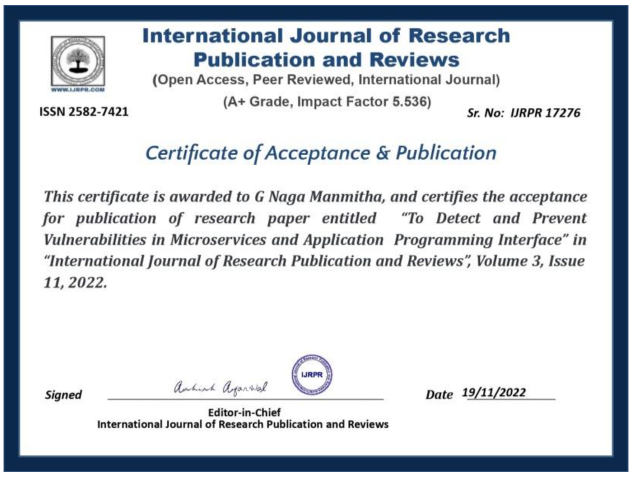
ISSN 2582-7421 *Sr. No: IJRPR 17275*

Certificate of Acceptance & Publication

This certificate is awarded to T Anusha, and certifies the acceptance for publication of research paper entitled "To Detect and Prevent Vulnerabilities in Microservices and Application Programming Interface" in "International Journal of Research Publication and Reviews", Volume 3, Issue 11, 2022.

Signed [illegible] IJRPR *Date* 19/11/2022

Editor-in-Chief
International Journal of Research Publication and Reviews



International Journal of Research Publication and Reviews
(Open Access, Peer Reviewed, International Journal)
(A+ Grade, Impact Factor 5.536)

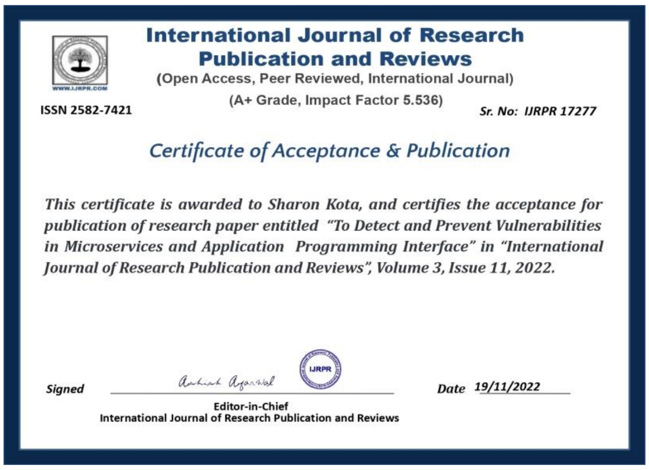
ISSN 2582-7421 *Sr. No: IJRPR 17276*

Certificate of Acceptance & Publication

This certificate is awarded to G Naga Manmitha, and certifies the acceptance for publication of research paper entitled "To Detect and Prevent Vulnerabilities in Microservices and Application Programming Interface" in "International Journal of Research Publication and Reviews", Volume 3, Issue 11, 2022.

Signed [illegible] IJRPR *Date* 19/11/2022

Editor-in-Chief
International Journal of Research Publication and Reviews



International Journal of Research Publication and Reviews
(Open Access, Peer Reviewed, International Journal)
(A+ Grade, Impact Factor 5.536)

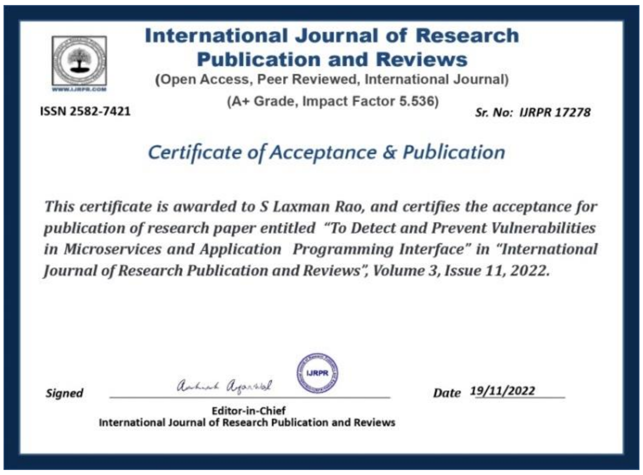
ISSN 2582-7421 *Sr. No: IJRPR 17277*

Certificate of Acceptance & Publication

This certificate is awarded to Sharon Kota, and certifies the acceptance for publication of research paper entitled "To Detect and Prevent Vulnerabilities in Microservices and Application Programming Interface" in "International Journal of Research Publication and Reviews", Volume 3, Issue 11, 2022.

Signed [illegible] IJRPR *Date* 19/11/2022

Editor-in-Chief
International Journal of Research Publication and Reviews



International Journal of Research Publication and Reviews
(Open Access, Peer Reviewed, International Journal)
(A+ Grade, Impact Factor 5.536)

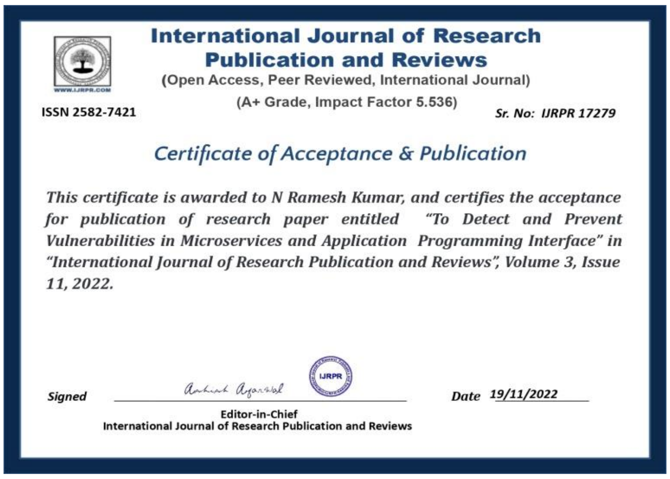
ISSN 2582-7421 *Sr. No: IJRPR 17278*

Certificate of Acceptance & Publication

This certificate is awarded to S Laxman Rao, and certifies the acceptance for publication of research paper entitled "To Detect and Prevent Vulnerabilities in Microservices and Application Programming Interface" in "International Journal of Research Publication and Reviews", Volume 3, Issue 11, 2022.

Signed [illegible] IJRPR *Date* 19/11/2022

Editor-in-Chief
International Journal of Research Publication and Reviews



International Journal of Research Publication and Reviews

(Open Access, Peer Reviewed, International Journal)

(A+ Grade, Impact Factor 5.536)

WWW.IJRPR.COM

ISSN 2582-7421

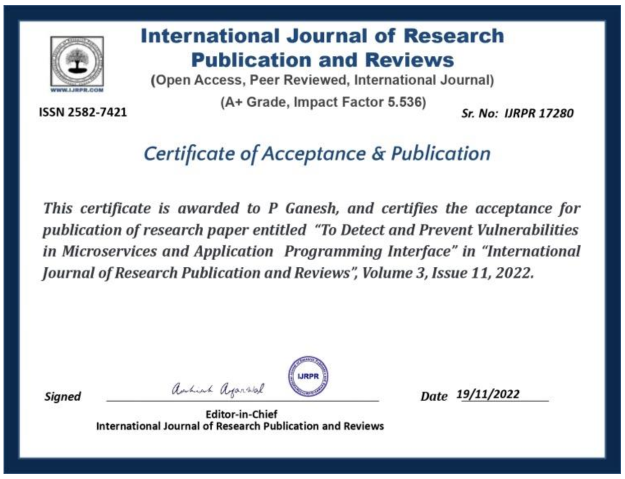
Sr. No: IJRPR 17279

Certificate of Acceptance & Publication

This certificate is awarded to N Ramesh Kumar, and certifies the acceptance for publication of research paper entitled "To Detect and Prevent Vulnerabilities in Microservices and Application Programming Interface" in "International Journal of Research Publication and Reviews", Volume 3, Issue 11, 2022.

Signed [illegible] IJRPR *Date* *19/11/2022*

Editor-in-Chief
International Journal of Research Publication and Reviews



International Journal of Research Publication and Reviews

(Open Access, Peer Reviewed, International Journal)

(A+ Grade, Impact Factor 5.536)

WWW.IJRPR.COM

ISSN 2582-7421

Sr. No: IJRPR 17280

Certificate of Acceptance & Publication

This certificate is awarded to P Ganesh, and certifies the acceptance for publication of research paper entitled "To Detect and Prevent Vulnerabilities in Microservices and Application Programming Interface" in "International Journal of Research Publication and Reviews", Volume 3, Issue 11, 2022.

Signed [illegible] IJRPR *Date* *19/11/2022*

Editor-in-Chief
International Journal of Research Publication and Reviews

19CS705 PROJECT

0 0 16 8

Course Outcomes

At the end of the project work, the students will be able to

1. Identify a contemporary engineering application to serve the society at large
2. Use engineering concepts and computational tools to get the desired solution
3. Justify the assembled/fabricated/developed products intended.
4. Organize documents and present the project report articulating the applications of the concepts and ideas coherently
5. Demonstrate ethical and professional attributes during the project implementation.
6. Execute the project in a collaborative environment.

CO–PO Mapping

COs	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	3	2	-	-	-	3	2	-	-	-	-	-
CO2	3	3	-	-	3	-	-	-	-	-	-	-
CO3	3	3	3	2	-	-	-	-	-	-	2	-
CO4	-	-	-	-	-	-	-	-	-	3		2
CO5	-	-	-	-	-	-	-	3	-	-	-	-
CO6	-	-	-	-	-	-	-		3	-	-	-

3–Strongly linked | 2–Moderately linked | 1–Weakly linked