RENBANK
SOLUTIONS ARCHITECTURE

Prepared for :
APDS7311 POE

Prepared by :
4 September 2024

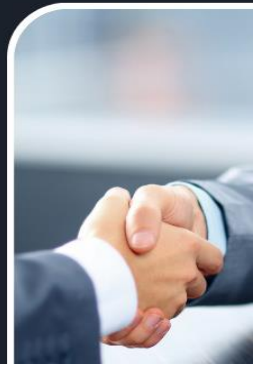


TABLE OF CONTENTS

Development Team Members: 3
Bank Name And Logo: 3
Introduction: 4
Renbank International Payment System Architecture: 5
Flow Of Data In Our System: 6
How We Plan To Secure Data Input And Data In-Transit: 7
How We Will Secure The Information Provided As Input: 7
How We Will Secure The Data In Transit: 8
How We Plan To Harden Against Attacks: 9
Session Hijacking Attacks: 9
Clickjacking Attacks 10
Sql Injection Attacks 10
Cross-Site Scripting (XSS) Attacks 11
Man-In-The-Middle (MITM) Attacks 11
Distributed Denial Of Service (DDoS) Attacks 12
MobSF Implementation: 14
ScoutSuite Implementation: 14
Conclusion: 15
Reference List: 16

DEVELOPMENT TEAM MEMBERS:

1. Rachael Moreira – ST10034334
2. Mayuran Chettiar – ST10155387
3. Jarod Naidoo – ST10174534
4. Devon Duerholz – ST10043202
5. Renier Coetzer – ST10055559

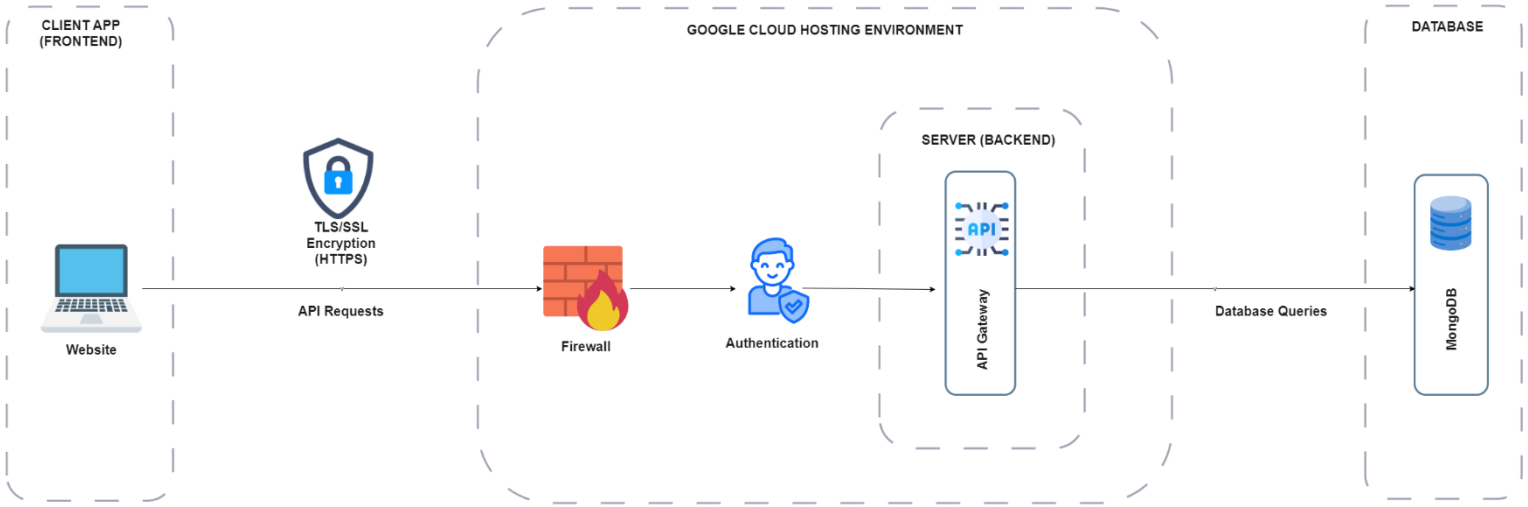
BANK NAME AND LOGO:

INTRODUCTION:

To meet particular business needs, solution architecture (SA) develops and provides technological solutions. It provides a clear plan to improve processes and a framework to make it happen. The documentation helps guide implementation and ensures everything fits smoothly (Ardoq, 2024).

Our team at RenBank, an international bank, has developed the solution architecture for the bank's internal international payment system. This document begins with an overview of the system architecture and then details the data flow, covering everything from user registration and login to making and verifying international payments. It includes a diagram and a detailed discussion on securing both input data and data in transit. The document also outlines our plans to protect the portal against various attacks. Finally, it includes screenshots of MobSF and ScoutSuite configurations, concluding with a summary and reference list.

RENBANK INTERNATIONAL PAYMENT SYSTEM ARCHITECTURE:

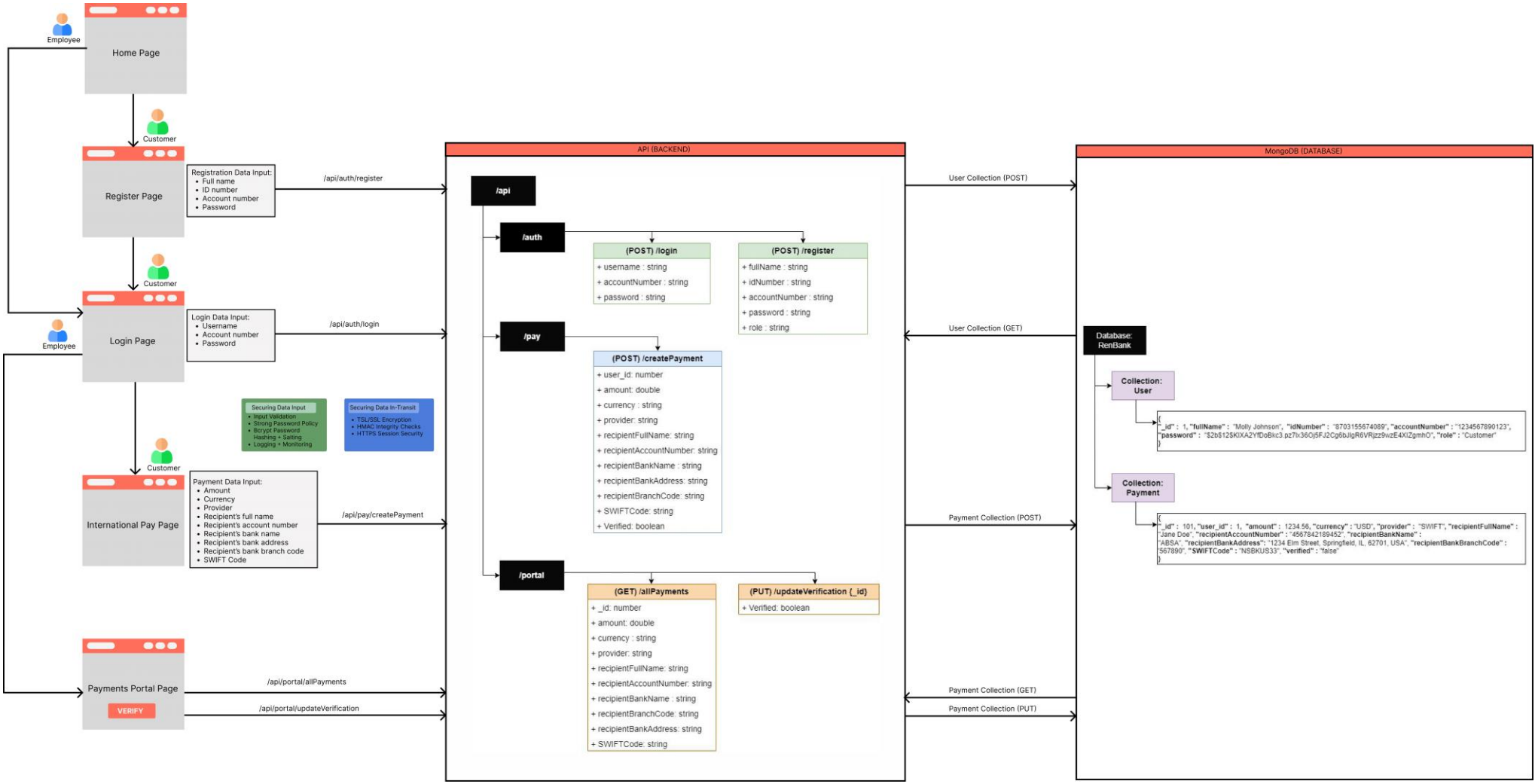


The above diagram shows the full architecture overview of RenBank's international payment system. This system adopts a client - server web application architecture. This architecture organizes applications by dividing tasks between two types of systems:

1. **Servers:** Provide resources or services.
2. **Clients:** Request those resources or services (GeeksForGeeks, 2024).

In this setup, when a client (like the website in our RenBank case) needs data, it sends a request to the server (our backend API) over the internet using Google Cloud for hosting. Google Cloud is chosen for its scalability, which adjusts to varying traffic loads, its reliability, which ensures continuous service availability, and its strong security measures to protect data (CRM Masters, 2022). The server processes the request, communicates with the MongoDB database, and sends the requested data back to the client (GeeksForGeeks, 2024).

FLOW OF DATA IN OUR SYSTEM:



HOW WE PLAN TO SECURE DATA INPUT AND DATA IN-TRANSIT:

HOW WE WILL SECURE THE INFORMATION PROVIDED AS INPUT:

1. **Input validation:**
   Input validation ensures that the data has been checked and cleaned to ensure correct formatting and prevent SQL injections and any other input manipulation tactics. This done by breaking down the inputted value and validating the contents of the data to disregard any malicious inputs.

   Input validation will be applied to all inputted values except password, because the software will have no access to the contents of the password and that will remain private to the user (Clarke, 2009).

2. **Password Hashing and Policies:**
   Through the use of a cryptographic technique like Bcrypt, a plain-text password is hashed into a distinct, fixed-length string of characters, known as password hashing. Instead of the actual password being saved in the database, this hashed password is what makes sure that the original password cannot be easily retrieved, even in the event that the database is hacked.

   Salting involves adding a random string of characters to the password before hashing. This guarantees that two users with the same password will have different hashed results.

   Strong password policies demand the creation of complex passwords that must have a minimum length of characters, a combination of capital and lowercase letters, digits, and special characters. Furthermore, the policy will forbid using previous passwords and require frequent password changes (Lukic, 2024).

3. **Logging and Monitoring:**

   Logging and monitoring involve tracking system activities and analysing them to detect anomalies or security incidents. Logging securely stores events such as user logins, transactions, and errors for auditing purposes. Constant monitoring examines logs in real-time or almost real-time to spot questionable activity, like unauthorised access attempts.

   Together, they provide visibility into a system's processes and aid in the early identification of breaches as well as ensuring the maintenance of a system's

integrity and security. When malicious users attempt to manipulate the login or transaction forms, the attempts will be logged and monitored to detect the malicious activity and put it to a stop (IBM, 2021).

HOW WE WILL SECURE THE DATA IN TRANSIT:

1. **Encryption:**

   Secure Sockets Layer (SSL) and Transport Layer Security (TLS) are essential for protecting user passwords and payment information in RenBank. Asymmetric encryption is the method used by this encryption protocol to create a secure link between the user's browser and RenBank's server.

   First, TLS/SSL uses a public and private key pair to encrypt the transferred data when a user registers in or makes a payment. The data is encrypted by the public key, making it unreadable to anyone other than RenBank's server's private key.

   Second, TLS/SSL guarantees that no tampering has occurred with the data while it is being transmitted. Additionally, it verifies the server's authenticity to guarantee that users are connecting to the real RenBank server and not a fake one.

   Thirdly, this protects sensitive information like usernames, passwords, account numbers, and payment details from eavesdropping and man-in-the-middle attacks (Dickens, 2021).

2. **Integrity Checks:**

   User and RenBank server data integrity is guaranteed by HMAC (Hash-based Message Authentication Code). RenBank, for instance, creates an HMAC using a cryptographic hash function and combines the data with a secret key when a user provides sensitive data, such as login credentials or payment information. The HMAC is then sent with the data.

   The data is genuine and unchanged if, upon receiving it, RenBank's server uses the same secret key to recalculate the HMAC and matches it with the received HMAC. By doing this, you can prevent unwanted changes to the data during transmission and guarantee that it is exactly what was sent (Solomon, 2023).

3. **Session Security:**

   Session security is important for protecting user data during an active session on the RenBank website. For example, when a user logs in, RenBank gives the user a unique and random session ID that is transmitted securely using HTTPS and this ID ensures that the user's session is recognized and maintained securely on the server.

   The session ID is protected during transit and is invalidated upon logout or inactivity, furthermore this done to prevent unauthorized access by ensuring sessions are short-lived and only active while the user is on it (Mol, 2020).

HOW WE PLAN TO HARDEN AGAINST ATTACKS:

SESSION HIJACKING ATTACKS:

Session hijacking is the term used to describe when a hacker impersonates you and gains control of your online session. Hackers accomplish this by taking information, such as cookies, that websites use to recognise you. Once they have control, they can access your accounts, see your personal information, or even make changes as if they were you (AO Kaspersky Lab, 2023).

Our Implementation Plan to Counteract This Attack:

- **Use HTTPS:** HTTPS will be enforced across the entire site to encrypt the data between users and servers, using TLS/SSL encryption to secure all data in transit. This will help secure sensitive information such as user information when registering for a new account or payment information (AO Kaspersky Lab, 2023).

- **Session Timeout:** We will implement session timeouts to limit the session duration. This limits the time that a session is active, reducing risks of session hijacking (AO Kaspersky Lab, 2023).

- **Monitor for Suspicious Activity:** This involves detecting unauthorized access attempts through validating user credentials and checking for multiple successive login attempts that failed. Through this monitoring, we will be able to identify suspicious activity, potentially being a session hijacking attack (AO Kaspersky Lab 2023).

CLICKJACKING ATTACKS

Clickjacking is a trick wherein a hacker hides a risky link or button below something that appears secure on a webpage. When you click on what looks as if a everyday button or hyperlink, you are actually clicking on something harmful without understanding it. This can result in unwanted behaviours, like sharing your personal information or allowing the hacker to manipulate your PC (Rydstedt, 2022).

Our Implementation Plan to Counteract This Attack:

- **Use X-Frame-Options Header:** Setting the X-Frame-Options HTTP header to DENY or SAMEORIGIN is essential for preventing an application from being embedded within an iframe on another website. This is essential for a payment application, as attackers ought to create a malicious site that embeds the application's content within a hidden iframe and trick users into clicking on the invisible iframe to perform accidental actions like authorizing payments or changing account information (Rydstedt, 2022).

- **Content Security Policy (CSP):** The Content-Security-Policy header offers a further layer of defence with the aid of specifying which domains can load content to your web page, consisting of iframes, styles, scripts, and images. This can prevent attackers from injecting malicious scripts or embedding our website into their own. For example, "script-src 'self'" would allow scripts to load only from our domain, preventing malicious JavaScript from executing (Rydstedt, 2022).

SQL INJECTION ATTACKS

SQL Injection is a hacking method where an attacker tricks a website into running harmful code in its database. They do this by entering malicious SQL code into input fields, like login forms or search boxes. If the website isn't properly protected, the attacker can use this trick to access, change, or delete sensitive data in the database (Contrast Security, 2024).

Our Implementation Plan to Counteract This Attack:

- **Filter Database Inputs:** Find and remove harmful code from user inputs. Validating inputs for any harmful or suspicious characters that might return data or “mess” with the database and act as queries (Kime, 2023).

- **Restrict Database Code:** Reduce the number of database procedures and code to avoid accidental database queries and investigation. As per the diagram the procedures are already listed (getting payment information, getting login details etc.) so the chances of malicious SQL code getting added without being authorized/checked are slim (Kime, 2023).

- **Restrict Database Access:** Limit access to prevent unwanted data access, exfiltration, or deletion by using access control measures. This is a feature that MongoDB has available (Kime, 2023).

- **Maintain Applications and Databases:** Maintain fully updated and patched databases. When feasible, upgrade (Kime, 2023).

CROSS-SITE SCRIPTING (XSS) ATTACKS

XSS attacks occur when an attacker injects malicious scripts into webpages viewed by other users. These scripts can steal data or manipulate the content (Bilginç IT Academy, 2024).

Our Implementation Plan to Counteract This Attack:

- **Input Validation and Sanitization:** We can ensure all input fields are properly validated through the use of custom regex patterns to restrict the type of data that can be entered. For example, custom regex patterns will be applied to validate sensitive information such as account numbers and passwords. These patterns will enforce strict formatting rules, ensuring that only valid and secure inputs are accepted, thereby reducing the risk of injection attacks and data breaches (Bilginç IT Academy, 2024).

- **Output Encoding:** This involves encoding all user-generated content before displaying it on the page to prevent execution of injected scripts (Bilginç IT Academy, 2024).

MAN-IN-THE-MIDDLE (MITM) ATTACKS

MITM attacks involve an attacker intercepting and potentially altering communication between the user and the server (Expert Panel, 2024).

Our Implementation Plan to Counteract This Attack:

- **Public Key Pinning:** By implementing certificate pinning it ensures that the client only trusts specific certificates when communicating with our server (Expert Panel, 2024).

- **Force TLS/SSL Encryption with a Trusted CA:** We can ensure that all data transmitted between our users and our servers is encrypted using TLS/SSL to protect against interception. A trusted Certificate Authority (CA) ensures that our website is secure. We will use Google Cloud's Certificate Authority Service to manage these certificates. When we use certificate pinning, we ensure that only certificates from our trusted CA are accepted, adding an extra layer of security (Google, 2019).

- **End-to-end encryption:** We utilize tools that encrypt data until it leaves the user and reaches its destination, ensuring that it cannot be accessed or tampered with along the way (Google, 2019).

DISTRIBUTED DENIAL OF SERVICE (DDOS) ATTACKS

DDoS attacks overwhelm a server with traffic, making it unavailable to legitimate users (Indusface, 2019).

Our Implementation Plan to Counteract This Attack:

- **Limiting Number of Requests from an IP Address within a Certain Time Window:** Rate limiting is a technique that helps prevent Distributed Denial of

Service (DDoS) attacks by restricting the number of requests a single IP address can make to a server within a specified time window, because in a DDoS attack, many requests are sent from different IP addresses or from a single IP address, which overwhelms the server's resources and disrupts its services, therefore by implementing rate limiting, the server can limit the number of requests from each IP address, preventing excessive usage.

For example, a rate limiter might allow only 100 requests per 15 minutes per IP address, therefore once this limit is exceeded, any extra requests from the same IP address within that time window are rejected with an error response. This protection method is effective in stopping attackers who rely on brute force by spamming the server with requests from a single machine (imperva, 2024).

- **Limiting the Size of Data Packages:** Limiting the size of data packages is a beneficial technique to mitigate DDoS attacks, especially those involving large data packages, which are designed to use up the server's resources, because during a DDoS attack, attackers might send unusually large HTTP requests with large data payloads attacked, such as oversized JSON objects or files, thereafter these large requests can overwhelm the server's memory and processing capacity, therefore slowing it down or causing it to crash.

  Thus, by setting limits on the size of incoming data, to for example 1kb, the risk of such large, resource-intensive requests reaching the server will be reduced and if an incoming request exceeds our specified size limit, the server will reject the request and respond with an error, preventing the server from needing to process the massive data packages (NetScout, 2024).

MOBSF IMPLEMENTATION:

Please review PDF document titled “MobSF_Report” to view our MobSF implementation screenshots.

SCOUTSUITE IMPLEMENTATION:

Please review PDF document titled “ScoutSuite_Analysis” to view our ScoutSuite implementation screenshots.

CONCLUSION:

Through our comprehensive solutions architecture for RenBank, which covers the international payment system, data flow, security measures for data input and transmission, portal hardening strategies, and the implementation of MobSF and ScoutSuite, we are confident in our ability to deliver a secure and reliable international payment system.

REFERENCE LIST:

AO Kaspersky Lab. 2023. What is session hijacking and how does it work.

[Online]. Available at: https://www.kaspersky.com/resource-center/definitions/what-is-session-hijacking

[Accessed 1 September 2024]

Ardoq. 2024. Solution Architecture: The Ultimate Guide to What It Is, How It Works, Benefits & Examples, 7 March 2024.

[Online]. Available at: https://www.ardoq.com/knowledge-hub/solution-architecture

[Accessed 25 August 2024].

Bilginç IT Academy. 2024. 7 Layers of Cyber Security You Should Know.

[Online]. Available at: https://bilginc.com/en/blog/7-layers-of-cyber-security-you-should-know-5933/

[Accessed 30 August 2024].

Clarke, J. 2009. Input Validation - an overview | ScienceDirect Topics.

[Online]. Available at: https://www.sciencedirect.com/topics/computer-science/input-validation

[Accessed 28 August 2024].

Contrast Security. 2024. What is SQL Injection How Does It Work.

[Online]. Available at: https://www.contrastsecurity.com/glossary/sql-injection

[Accessed 1 September 2024]

CRM Masters. 2022. Google Cloud Platform – Advantages and Disadvantages, 4 August 2022.

[Online]. Available at: https://crm-masters.com/google-cloud-platform-advantages-and-disadvantages/

[Accessed 26 August 2024].

Dickens, R. 2021. What Are Encryption Protocols and How Do They Work? | Encryption Consulting.

[Online]. Available at: https://www.encryptionconsulting.com/what-are-encryption-protocols-and-how-do-they-work/

[Accessed 28 August 2024].

Expert Panel. 2024. 19 Keys To Detecting And Preventing Man-In-The-Middle Attacks. Forbes, 7 March 2024.

[Online]. Available at: https://www.forbes.com/councils/forbestechcouncil/2024/03/07/19-keys-to-detecting-and-preventing-man-in-the-middle-attacks/

[Accessed 30 August 2024].

GeeksForGeeks. 2024. Client-Server Model, 19 April 2024.

[Online]. Available at: https://www.geeksforgeeks.org/client-server-model/

[Accessed 26 August 2024].

Google. 2019. Cloud Computing Services | Google Cloud.

[Online]. Available at: https://cloud.google.com/

[Accessed 30 August 2024].

IBM. 2021. Logging and monitoring.

[Online]. Available at: https://www.ibm.com/docs/en/ftmswsfm300?topic=program-logging-monitoring

[Accessed 28 August 2024].

imperva. 2024. What is Rate Limiting | Types & Algorithms | Imperva.

[Online]. Available at: https://www.imperva.com/learn/application-security/rate-limiting/

[Accessed 7 October 2024].

Indusface. 2019. 6 Best Practices to Prevent DDoS Attacks.

[Online] Available at: https://www.indusface.com/blog/best-practices-to-prevent-ddos-attacks/

[Accessed on 30 August 2024].

Kime, C. 2023. How to Prevent SQL Injection Attacks. eSecurity Planet, 16 May 2023.

[Online]. Available at: https://www.esecurityplanet.com/threats/how-to-prevent-sql-injection-attacks/

[Accessed 1 September 2024]

Lukic, D. 2024. Hash, salt and verify passwords - Node, Python, Go and Java.

[Online]. Available at: https://supertokens.com/blog/password-hashing-salting

[Accessed 28 August 2024].

Mol, G. 2020. Session Security.

[Online]. Available at: https://beaglesecurity.com/blog/article/session-security.html

[Accessed 28 August 2024].

NetScout. 2024. What is a Large Payload Post DDoS Attack.

[Online]. Available at: https://www.netscout.com/what-is-ddos/large-payload-post
[Accessed 7 Oct. 2024].

Rydstedt, G. 2022. Clickjacking. OWASP Foundation, 2022.

[Online]. Available at: https://owasp.org/www-community/attacks/Clickjacking

[Accessed 1 September 2024]

Solomon, S. 2023. What Is HMAC (Hash-Based Message Authentication Code).

[Online]. Available at: https://frontegg.com/blog/hmac

[Accessed 28 August 2024].