DEMOCRATIC AND PEOPLE'S REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
UNIVERSITY OF TIARET

THESIS

Introduced to:

MATHEMATICS AND COMPUTER SCIENCE FACULTY
DEPARTMENT OF COMPUTER SCIENCE

For the graduation of:

MASTER

Specialty: Software engineering

By:

ZEGAI Houari
CHARFAOUI Younes

On the subject

RSA SecurID Token Compatible Random Number Generation For Android and 3D Secure

Publicly Defended on .. / 09 / 2020 in Tiaret in front of the jury composed of:

Mr. BERBER El-Mehdi	Grade	University MAA	President
Pr. DAHMANI Youcef	Grade	University Professor	Supervisor
Mr. BAKAR Khaled	Grade	University MAA	Examiner

2019-2020

Acknowledgements

All thanks and praises go to Allah, the Almighty and the Merciful, for granting us the wisdom and the health to complete this work during this challenging time.

We would like to acknowledge the help and the supervision provided to us by Pr. Youcef DAHMANI for his availability, recommendations, and confidence. His office door was always open whenever we ran into a trouble spot or questioned our project. His direct involvement and encouragement throughout the study influenced the positive progress and made us stronger and better students.

A Special Thanks to Mr. BAKAR Khaled and Mr. BERBER El Mehdi, It is a great pleasure to have you as the jury for this work. We want to express our deep gratitude to you having kindly accepted to examine our modest work.

We would like to express our appreciation to Google for its search engine and the Stack exchange community for many useful inputs and valuable comments.

A big thanks to the teachers and the Computer Science and Mathematics faculty administration who watched over our training and our follow-up in our studies.

Finally, we address our thanks to all those who contributed to this work's outcome with their advice, constructive feedback, or encouragement.

Dedication

To my family. A special feeling of gratitude to my loving parents, Whose words of encouragement and push for tenacity ring in my ears. My brothers' Fares, Abdallah, and Abdelkader have never left my side and are very special.

To my friends who supported me throughout the process. I will always appreciate everything they did. I will not mention their names so as not to forget some of them. Thank you all.

- Houari ZEGAI

Dedication

I dedicate my work to my mother and father: I couldn't have done this work without your inherited genes. A special feeling of gratitude to my parents, whose words of encouragement and push for tenacity ring in my ears. Thanks to their tender encouragement and their great sacrifices, even though they still think that I am insane and mentally ill (computer science and mathematics symptoms), they could create a loving and favorable environment for my studies' pursuit.

To my very beloved only sister Lina, and my dear brothers Abdelatif and Fouad, without whom, this work would have been completed three months earlier.

I dedicate this work to all my family members, to my only GrandMother, to my aunts and uncles; I feel blessed to belong to such a family that provides support and guidance throughout life.

I also dedicate this work to my many friends and my promotion of Master 2 Computer Science 2019/2020, who have supported me throughout the process. I will always appreciate all they have done, from helping me be a better person, proofreading my works, and teaching me several technological pieces. And especially those who bought me food in exchange to help them understand several computer science subjects (such as database normalization and solving deadlock with semaphores).

Last but not least, I dedicate this work to everyone who wonders If I'm dedicating this work to them.

- Younes CHARFAOUI

Abstract

An electronic payment system helps a customer make a payment to a retailer or service provider electronically. Payment gateways, a portal between consumers and payment processors, use different authentication methods during online payment to secure a customer's payment details, typically debit or credit card information. However, since a merchant already can access the payment information in any manner, the encryption offered by a payment gateway can not fully secure the client's payment details. Not all merchants provide their customers with a safe payment system and stick to it while maintaining a standard payment policy. This also opens the payment information received by a customer to the possibility of being hacked or misused by retailers or stolen by hackers or spammers. In this work, we present a stable solution to payment processes in which payment information about a customer is secured. Upon authentication of the transaction, a client sends his contact details and authorization directly to a payment gateway submitting an invoice to the correct seller. Our research's fundamental goal is part of the modernization of payment systems in Algeria.

Keywords: Strong two-factor authentication, Prime numbers, Random number generation, Distribution of cryptographic keys and electronic certificates, RSA cryptography, Elliptic Curve Cryptography.

ملخص

يساعد نظام الدفع الإلكتروني العميل على إجراء الدفع للتاجر أو مزود الخدمة إلكترونيًا. تستخدم بوابات الدفع , وهي بوابة بين المستهلكين ومعالجي الدفع , إجراءات مصادقة مختلفة أثناء الدفع عبر الإنترنت لتأمين تفاصيل دفع العميل , وعادةً ما تكون معلومات بطاقة الخصم أو الائتمان. ومع ذلك , نظرًا لأن التاجر لديه بالفعل إمكانية الوصول إلى معلومات الدفع بأي طريقة , لا يمكن للتشفير الذي توفره بوابة الدفع تأمين تفاصيل الدفع للعميل بشكل كامل. في الواقع , لا يوفر جميع التجار لعملائهم نظام دفع آمنًا ويلتزمون به مع الحفاظ على سياسة دفع قياسية. يؤدي هذا أيضًا إلى فتح معلومات الدفع التي يتلقاها العميل لإمكانية التعرض للقرصنة أو إساءة الاستخدام من قبل تجار التجزئة أو السرقة من قبل المتسللين ومرسلي البريد العشوائي. في هذا البحث , نقدم حلاً مستقرًا لعمليات الدفع حيث يتم تأمين معلومات الدفع الخاصة بالعميل. عند المصادقة على المعاملة , يرسل العميل تفاصيل الاتصال الخاصة به والتفويض مباشرة إلى بوابة الدفع, ويقدم فاتورة إلى البائع الصحيح. الهدف الرئيسي لبحثنا هو جزء من تحديث أنظمة الدفع في الجزائر.

الكلمات الرئيسية: مصادقة قوية بعاملين , أرقام أولية , توليد أرقام عشوائية , توزيع مفاتيح التشفير والشهادات الإلكترونية , تشفير RSA , تشفير منحنى إهليلجي.

Contents

List of Figures

List of Tables

List of Acronyms

3-D Secure Three Domain Secure. vii, viii, 1, 33–41, 70, 90

3DS Three Domain Secure. xi, 34, 37–41

ACS Access Control Server. xi, xii, 38–41, 48, 50, 52, 77, 78, 86–88, 90

AES Advanced Encryption Standard. 27, 62–64

AHS Authentication History Server. xii, 89

AReq Authentication Request Message. viii, 39–41

ARes Authentication Response Message. viii, 39, 41

ATM Automated Teller Machine. 12

CA Certificate Authority. 13

CReq Challenge Request Message. viii, 39, 40

CRes Challenge-Response Message. viii, 40

CSPRNG Cryptographically Secure PRNG. vii, 26

CVV card verification value. 36

DBMS Database Management System. 47

DES Data Encryption Standard. 62, 63

DHKE Diffie–Hellman Key Exchange. 61

DS Directory Server. 38, 40

DSA Digital Signature Algorithm. 62, 64

EC Elliptic Curve. 61

ECC Elliptic Curve Cryptography. 27, 61, 62

ECDH Elliptic Curve Diffie–Hellman. 60, 61

ECDSA Elliptic Curve Digital Signature Algorithm. 64

ER Entity Relationship. xi, 48–50

HTTPS Hypertext Transfer Protocol Secure. 18

IDE Integrated Development Environment. 67, 72

Java EE Java Enterprise Edition. 68

JVM Java virtual machine. 67, 72

JWT Json Web Token. 65

LCG Linear Gongruential Generator. 25

MVC Model – view – controller. 69, 71

MVVM model - view - ViewModel. 71

NIST National Institute of Standards and Technology. 31, 62, 64

OO Object-oriented. 50

PCG Permuted Congruential Generator. ix, xi, 25, 56, 62

POM Project Object Model. 68

PReq Preparation Request Message. viii, 40

PRes Preparation Response Message. viii, 40

PRNG Pseudorandom Number Generator. vii, 24–28, 30, 32

REST Representational State Transfer. 67, 68

RNG Random Number Generator. vii, ix, 21–32, 55–58, 90

RReq Results Request Message. viii, 40

RRes Results Response Message. viii, 40

SMS Short Message Service. 16

SP-network Service Provider network. 63

SQL Structured Query Language. 70

SSH Secure Shell. 63

SSL Secure Sockets Layer. 17, 64

TLS Transport Layer Security. 18, 63

TRNG True Random Numbers Generator. vii, 23, 24, 26

UI User Interface. 69

UML Unified Modeling Language. 4

US United States. 63

VCS Version Control System. 72, 73

Introduction

Problematic

Despite the increasing development of new Information and communication technologies and ongoing advancement in technology with mobile payments, it is necessary to note that online payment has its features and challenges.

The most crucial question of online payment is the parties' security. And to ensure a high level of security, a system must address online transaction issues, from the risk of interception of information to the fraudulent transactions made by hackers and spammers.

Because of the open and international nature of online payment networks, this will lead to more issues concerning customers and merchants' security worldwide. Some protocols are suggested to protect these business transactions. The most popular one is 3-D Secure, developed by EMVCo (Europay MasterCard and Visa Card).

Another aspect concerning the security of electronic payment is the authentication of the various partners in this process. Multiple solutions are suggested but often imply that the customer has some knowledge of the computer tool and remembers his different passwords to access the expected service.

The present work is twofold: Implementation of the protocol 3-D Secure as specified by EMVCo and a newly proposed method to authenticate customers. We have proposed a solution that allows any user with a mobile phone to make purchases without worrying about a password to access the different electronic payment platforms. The solution is based on generating a couple of random numbers that identify the mobile uniquely and, consequently the customer. The generation of random numbers is a challenging work that needs skills in mathematics. Our solution is based on The PCG algorithm.

Challenges and working hypothesis

The new system's challenges and objectives are primarily security, and payment instruments are also another concern for electronic payment. Nevertheless, the high degree of technicality associated with it constitutes an excellent opportunity to develop entirely innovative payment systems.

For a safe and efficient system, an electronic payment system must meet the following basic requirements:

- Authentication and integrity of messages
- Confidentiality
- Availability
- Non-repudiation

Thesis Overview

In this thesis, we propose an online payment system in which a customer's payment information is sent directly to a payment gateway, instead of sending it through a merchant. This approach prevents a customer's payment information from being manipulated and compromised by a merchant.

We will rely on the 3 Domain Secure architecture to implement a highly secure environment between the merchant and the payment gateway.
The mobile authentication is used alongside a state of the art method that allows this authentication to use random numbers as a second factor for the Two Factor authentication mechanism.

Objectives and Advantages of the new system

Objective

- Bring together the players in the trade (bank, customer, merchant) by using an open and secure network
- Allow traders to register and administer a remote account
- Record information on transactions carried out
- Allow fraud detection and risk-based analysis of transactions.
- Make the management of transactions user-friendly
- Authenticate and validate users identity

Advantage

- Customer support
- Cost reduction
- Detect fraudulent transactions
- Make the bank card available to the customer as a secure payment tool

Subject delimitation

Any scientific research work is limited by time, effort, and space to facilitate its understanding and exploitation. Thus, it is within the IT and electronic payment services that provide user authentication and validation of payment requests in a highly secure environment.

Used Techniques and Procedures

Within this project's context, we used the UML method to model our new system's analysis. We made an iterative and incremental approach based on constraints we have and centered on software architecture to build the system. We used the Agile methodologies to wrap all that in a single framework to analyze the constraints and requirements, model the new system, and build an efficient and secure network.

We also relied on many existing documents and standards in the approach we used to create and design a state of the art system.

Thesis Organization

To summarize our work, we organized this thesis into six chapters. Each chapter will introduce a part of our work that was needed to implement the new system. Here is a map of our thesis chapters:

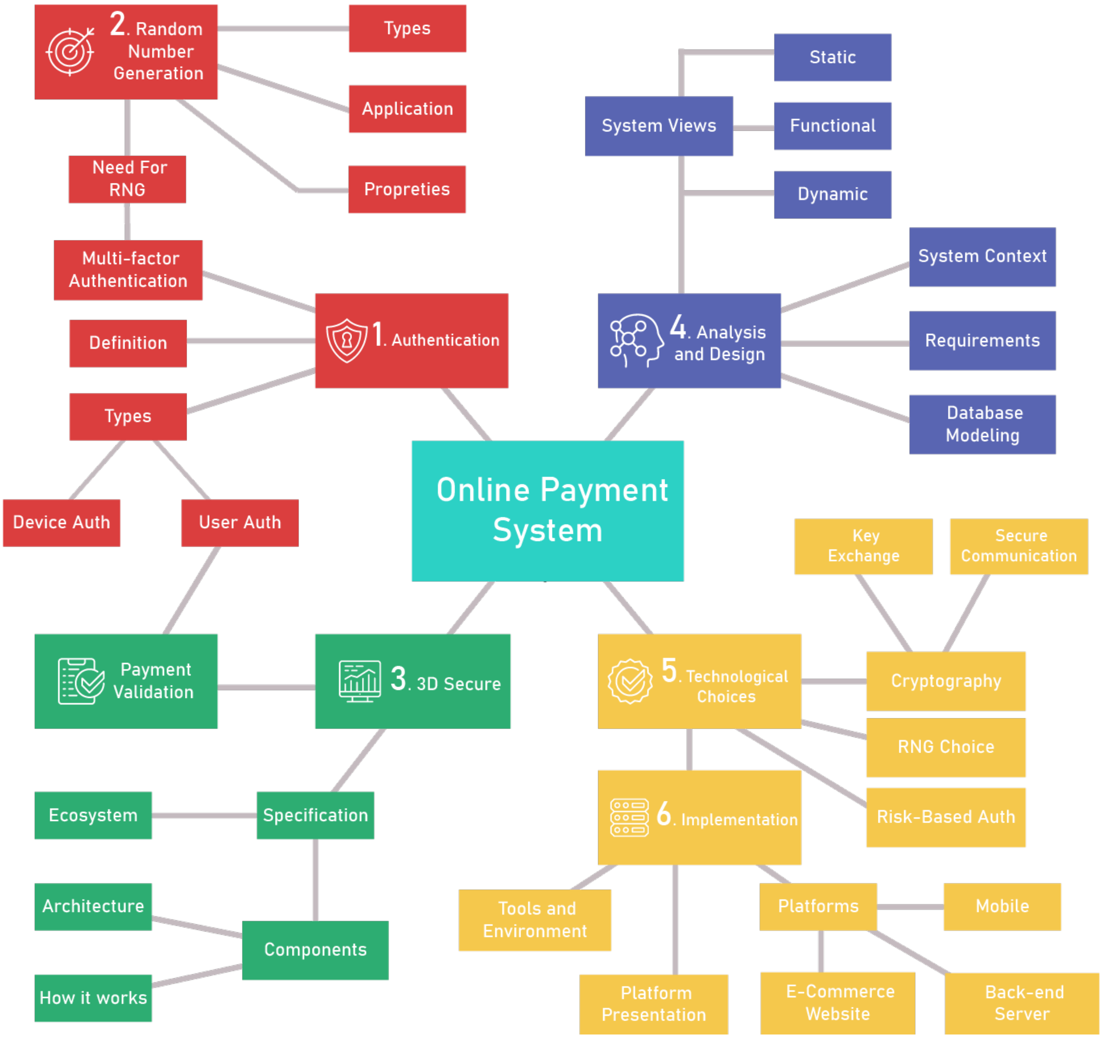


Figure 1: Thesis Organization

Chapter 1

Authentication

1.1 Introduction

Identification, authentication, and authorization, These three terms are elements of information security.

- The first stage is identification: It recognizes information about the user, for example, log in.
- The second stage is the authentication: This is the process of verifying user information like the password.
- The third stage is the authorization: Here the user rights are checked, and access is determined.

This chapter will focus on Authentication and introduce it to you in an easily understandable way.

1.2 Definition of Authentication

Authentication is the method of deciding whether someone or something is really who or what they claim to be. Authentication is accomplished in the real world, through face-to-face contact, displaying IDs, passports, or different kinds of certificates.

In the technology fields, the Authentication provides access control for systems by checking users' credentials in a database of authorized users or a data authentication server [3].

1.3 User Authentication

For most information security cases, user authentication is the fundamental building block and the key defensive line. It is the basis for most forms of access control and device accountability. User authentication requires two features:

First, the user identifies himself to the system by presenting a credential, such as a user ID.

Second, the system verifies the user by the exchange of authentication information.

For example, user Alice may have "HouYou" as an identifier. This information must be stored on any server or computer system that Alice wishes to use and may be known to system administrators and other users. Typical authentication knowledge associated with this user ID is a password that is kept secret (only known to Alice and the system).

When no one can get or guess Alice's password, a combination of Alice's user ID and password would enable administrators to set up Alice's access controls and monitor her behavior. Because Alice's ID is not a secret, system users can send her email, but because her password is a secret, no one can pretend to be Alice.

In essence, identification is how the asserted identity of the device is given by the user; authentication by the user is how the legitimacy of the claim is created.
The User Authentication is different from the Message Authentication since the Message Authentication is a process that requires the parties to check that the sender is authentic and that the contents of the received message have not been changed. [4]
Before discussing user authentication approaches, we need to clarify that authentication helps us identify and check users' identities. This authentication is used in various technology systems to grant access to their resources. One of these systems are the online payment system (our new system) that permits the users to make transactions.
This is a big field in which many research and method we proposed in order to make this process secure, fast, and user friendly. One of these protocols is 3 Domain Secure 2.0 Protocol developed by EMVCo to enable consumers to authenticate themselves with their card issuer when making cards, not present transactions; this authentication protocol requires a whole chapter to explain it. Thus, chapter three will be a detailed description of the protocol of 3 Domain Secure so that we can implement it in our new system.

1.3.1 Approaches to User Authentication

Conventionally, user authentication is based on one of four categories; these types can be mixed to offer a better and stronger authentication, as follows:

1.3.1.1 Something that one knows

Known also as "Personal knowledge," Examples include text password, PIN, graphical passwords (for example, drawing a pattern on a touchscreen device), and personal knowledge questions (for example, "What is your favorite pet name?").

1.3.1.2 Something that one owns

Known also as “Physical possession” The examples include RSA SecurID, IP Address, Smartcard, Mobile phones, and USB tokens hardware. The user can verify the possession of a USB token by plugging it into the device from which they authenticate. The control of a mobile phone can be verified by the website verifier sending the user a randomly generated text message and requesting the user to type it into the website.

1.3.1.3 Something that one is

Known also as “Physical biometrics,” The examples include fingerprint, iris (eye), and facial or voice recognition.

1.3.1.4 Something the user does

Examples include user recognition through typing rhythm, the dynamic usage of the touchscreen or mouse, or gait analysis (e.g., through gathering mobile accelerometer data while the user is walking). [5]

When we use two methods from different categories, this is called 2-Factor authentication; this is often referred to as ”strong authentication” and should be used in secure areas. [6]

1.3.2 Authentication methods

1.3.2.1 Password-based Authentication

The password-based authentication scheme is a commonly used line of defense against intruders. Virtually all multi-user systems, network-based websites, Web-based e-commerce sites, and other related services allow a user to have not only a name or identifier (ID) but also a password. The program compares the password for that user ID to a previously stored password, preserved in the system's password file.

The password authenticates the identity of the user signing into the system. Besides, the ID offers the following security:

- The ID decides whether the user is allowed to enter a device. In specific systems, entry is permitted only to people who already have an ID registered on the list
- The ID sets out the privileges granted to the user. Some users may have the status of an administrator or "superuser" that allows them to read files and perform functions that are mainly protected by the operating system, and some systems have anonymous, or guest accounts, and administrators of these accounts have rights more restricted than others
- The ID is used in what is termed discretionary control of access. For example, a user may permit them to read files owned by that user by listing the other user's IDs [4]

1.3.2.2 Token-based Authentication

A token is an authentication tool that authenticates the recipient by embedding the necessary authorization (like a password) into the token. The concept of token-based authentication is when you show yourself to the device. You need to have the token in your hands.

The machine won't remember you without the token, whether the token was misplaced, loaned, or stolen. Here is a description of tokens' fundamental properties from a protection point of view [7]:

- A person must physically hold the token to use it

- It is hard to replicate a good token
- A person can lose a token and accidentally lose access to a resource
- Persons can identify stolen tokens by taking stock of the tokens they would have in their hands

Tokens are usually split into two categories: active and passive. In all cases, the token contains a specific key, and to create a working copy of a given token, one must copy the base key. A passive token is simply a dedicated base secret storage device. [5, p256]

Examples we might find in our lives around us include, for example, ATM cards, mechanical keys, and advanced equipment such as ”input keys.” Under different circumstances, an active token can produce different outputs.

For example, an active token may engage in an authentication challenge-response protocol or have other crypto-functions that use the secret base token.

Active tokens have historically been either commercial one-time authentication tokens or smart cards, although other versions have emerged connected to existing ports on desktop.

1.3.2.3 Biometrics

This technology gathers unique physiological or behavioral attributes of a person for storage in a database or comparison with data already found in a database.

Like the reference test technique, a biometric is identified as a unique, measurable, biological feature or characteristic for automatically recognizing or verifying a human being’s identity. The statistical study of these physical features has become known as the biometrics method. [8]

These days, biometric technologies are usually used for security purposes to evaluate human characteristics. Five of the most common biometric physical patterns examined for safety purposes are fingerprint, eye, face, voice, and hand. The biometrics industry uses a widely quoted description to help explain this point. There are three health standards [9]:

- The lowest security level is classified as something you have in your possession, such as an ID badge with a photograph on it.
- The second security level is something you recognize, such as the PIN code to your bank account card on ATM or a computer login password
- The highest degree of protection is something that you are and something that you do. Essentially, this is what biometric technology is all about

1.3.2.4 Certificates

A certificate is a letter signed by a third party who has public confidence. The entities must trust the certification authority using the certificates issued by that trusted party. The certificate ensures that the public key is connected to the designated individual. Thus, ensuring that the approved party has been established is the crucial prerequisite for public-key techniques. The format X.509 V3 public-key certificate is the most widely used [10].

The certificate includes edition, serial number, signature, issuer, validity, subject matter, subject Public Key Information, subject Unique ID, issuer Unique ID, extensions, signature algorithm, and the signature value.

Authentication of the certificate is also done through the process of public-key authentication. Instead of only sending a public key to the client, it sends out a certificate that includes a public key. In short, the authentication of the certificate works as follows:

- The client sends the user certificate (including the user's public key) to the server

- The server uses a CA certificate to test the validity of the user certificate
- The server uses the user certificate to verify whether login is enabled or not from its mapping file(s)
- Finally, if the connection is allowed, the server will ensure that the user uses a challenge to have a valid private key

This approach is more secure than conventional public-key authentication since the program checks that a trustworthy Certificate Authority (CA) has provided the user certificate. Moreover, certificate authentication is more convenient since the server needs no local archive of the users' public keys [11].

1.3.2.5 Mutual authentication

Mutual Authentication A two-way authentication consisting of a client and a server is often named. The client has to prove their identity to the server, and the server demonstrates their identity to the client before any application begins. In neither client process nor server process requires user interaction. So that the client can trust the actual organization or entity's certification chain and build the connection between them. The importance of using mutual authentication is presented like this: "Mutual authentication ensures that not only is the person behind the keyboard who he claims to be, but also proves that the server he is communicating with is who it claims to be.

Mutual authentication protects the confidentiality of sensitive information by ensuring that the user's service is genuine." [12] It is also used in online banking and e-business. By using mutual authentication, users trust the Certificate of a Trade Association or an individual in the Certificate Chain, ensuring users trust the third party authority.

For example, when we buy from Amazon, we do not have to pay money directly to the store owner because we don't trust the private offline payment and the owner of the show. So Amazon offered an online payment service-PayPal, which means we can send money through PayPal as long as we get products from the shop owner, and then PayPal will transfer money to the show owner. PayPal plays a third party in the online business to protect the rights of the customer. The figure below shows the mutual authentication model:

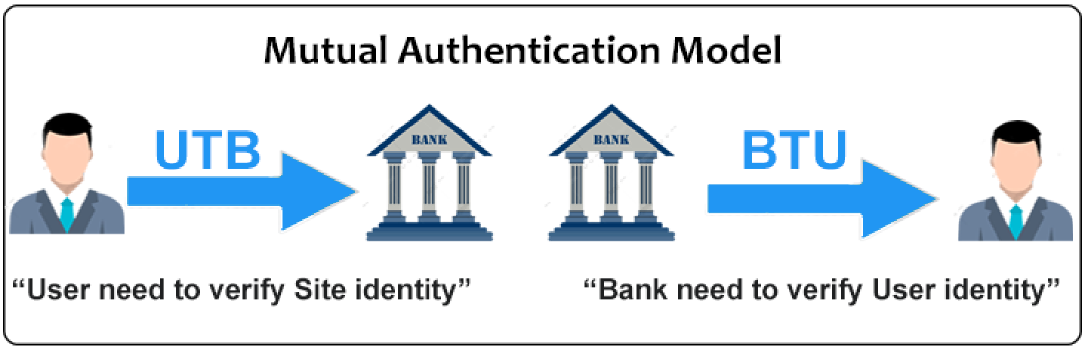

Mutual Authentication Model
UTB
BANK
BANK
BTU
"User need to verify Site identity"
"Bank need to verify User identity"

Figure 1.1: Mutual authentication model

1.3.2.6 Multi-factor Authentication

Multi-factor authentication (MFA) is a secure process in which access is given to a user device only after the successful display of two or more elements of the authentication categories for providing a secure and better way for authenticating users. As mentioned above, the four most common types of authentication factors are:

- Something that one knows
- Something that one owns
- Something that one is
- Something the user does

The purpose of MFA is to verify legal users to protect their private information by offering complex protection and, at the same time, making it more difficult for unauthorized persons to obtain access to it.

The benefits of implementing MFA are to provide a robust way of authenticating applications. Owing to vulnerability or data loss, each factor is not available at any time; the system will also offer authentication services using uncommitted authentication factors. From the attackers' point of view, they have to conquer several obstacles to get through the goal system [13].

We are going to use this type of authentication in our new system, specifically, we will use two factor authentication method, the first factor is the normally used email and password, and the second factor, is some random special numbers generated for each user, this also requires a specific study, How we can generate random number? In the following chapter, we will talk in details about the types and method by which the computer can generate pseudo random number for different uses, and in our case, its the second factor of the two factor authentication that will help us identify each client alone.

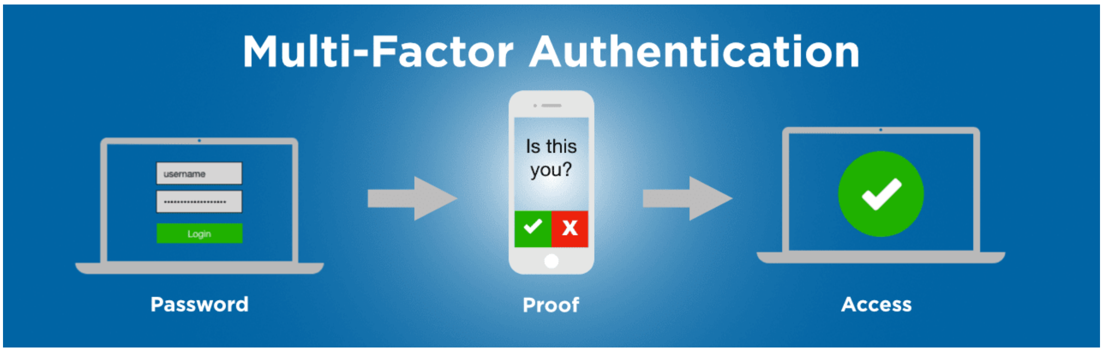


Figure 1.2: Multi-Factor Authentication [1]

1.3.2.7 One Time Password Authentication

One-time password authentication is usually applied in addition to password authentication to implement two-factor authentication (2FA). In this concept, the user needs to provide two types of data for entering the system: something that he knows (for example, a password), and something that he owns (for example, a device for generating one-time passwords). The presence of two factors can significantly increase the security level, which is required for certain types of web applications.

Another popular scenario for using one-time passwords is additional user authentication during the execution of essential actions: transferring money, changing settings, etc.

There are various sources for creating one-time passwords. Most popular:

- Hardware or software tokens can generate one-time passwords based on the secret key entered into them and the current time. Secret keys of users, which are a factor of ownership, are also stored in the server, which allows you to check the entered one-time passwords. An example of a hardware implementation of tokens - RSA SecurID, software - Google Authenticator application.

- Randomly generated codes are transmitted to the user via SMS or another communication channel. In this situation, the ownership factor is the user's phone (more precisely, a SIM card attached to a specific number).

- Printout or scratch card with a list of pre-formed one-time passwords. For each new login, you must enter a unique one-time password with the specified number.

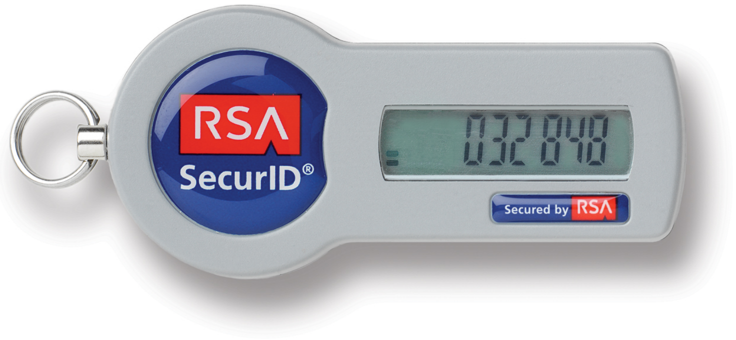


Figure 1.3: The RSA SecurID hardware token generates a new code every 30 seconds [13]

In web applications, this authentication mechanism is often implemented through the extension of forms authentication: after the primary authentication with a password, a user session is created. However, in the context of this session, the user does not have access to the application until he performs additional authentication with a one-time password.

1.4 Machine Authentication

Machine authentication is the process of general authentication of a machine when the machine tries to view or exchange information or perform some other kind of physical interaction. Machine authentication happens in multiple ways in various IT environments, which usually requires a ”Digital Certificate,” such as the SSL protocol (used on the Internet).

1.4.1 Device authentication

Device authentication requires authentication of a device to a verifier, such as a remote server. For example, a cell phone has to authenticate itself, usually by a cryptographic key held on a SIM card, to connect to a mobile network.

Some systems may introduce two-stage authentication that incorporates the local user authentication stage with the application authentication stage; for example, when paying a credit card with a chip-and-pin technology, the payment terminal first communicates to the payment network authentication server to validate the authenticity of the credit card itself (i.e., device authentication).
Users are then authenticated by inserting a numerical PIN that is verified locally by the payment device (i.e., local user authentication). Two-stage authentication can also be used on web authentication; device authentication can also occur between devices (i.e., machine-to-machine authentication).

1.4.2 Server Authentication

Server authentication requires the authentication of remote servers to users. This is achieved on the web through HTTPS built on the public key infrastructure of TLS.
For example, when a user visits the website www.univ-tiaret.dz using HTTPS, the web browser authenticates the webserver to guarantee that a hacker is not intercepting the communication between the website (www.univ-tiaret.dz) and the web browser.
Upon successful authentication, web browsers may display a visual hint in the browser's border region (which can not be changed by site information being displayed), such as a green lock symbol, as a signal to users that the web server has been authenticated.

1.5 Conclusion

Several authentication systems are actually in use. If categorized based on usability and security, most of them fall into the category of security that ensures the protection of the user's account using the second factor but lacks proper usability. Therefore, for strong authentication, we need to use more than one element, such as a one-time password combined with Password-based authentication methods.

Chapter 2

Random Number Generators

2.1 Introduction

In this chapter, we will first start with a definition of randomness and specifically random numbers, their different applications in computer science, and how to generate them.

Next, we will move to different types and properties of Random number generators; then, we explore existing algorithms and and the one we chose to implement in our solution.

We will conclude this chapter with conventional ways of testing the quality of Random number generators.

2.2 Definition of Randomness

Although the word random is usually used in everyday language, defining what randomness is can be very challenging, especially to define it mathematically.

The Oxford dictionary defines it as the following:

“Randomness [NOUN]: The quality or state of lacking a pattern or principle of organization; unpredictability". [14]

Our discussion for this thesis will focus on randomness in numbers, and specifically, on random number sequence, since, in a sense, there is no such thing as a random number; for example, is two a random number?
Instead, we are talking about a sequence of independent random numbers with a specified distribution, and that suggests simply that each number was just generated by chance, having nothing to do with other sequence numbers, and that each number has a specified likelihood of falling within a given range of values. [15]
A generally accepted and fundamental definition for a random number sequence is as follow:
"A random number sequence follows a uniform distribution over all possible values, and each number is independent of the previously generated numbers." [16]

2.3 Definition

Since we have a basic definition of the random numbers, we are now wondering how we can make computers generate these numbers, of course, that will be throughout algorithms, we call these algorithms Random Number Generator (RNG) for short).
A definition of states that "A random number generator is a computer procedure that scrambles the bits of a current number or set of numbers to produce a new number, in such a way that the result appears to be randomly distributed among the set of possible numbers and independent of the previously generated numbers". [16]
Some researchers refer to the notion of inputs as a core concept in RNG "A random number generator is an algorithm that produces a sequence of numbers

based on an initial seed or using continuous input. This sequence must appear random to any observer". [17]

Thus, we define a random number generator as any system that creates random sequences that respect the definition of randomness, as mentioned earlier, or we can also say that these generators must meet specific statistical requirements for randomness.

The statistical requirements of each RNG can vary significantly, and depends on the context. The RNG used in cryptography is different from RNG used in simulation. The first one has strict requirements like the past sequences should not be discovered nor repeated in the next numbers. In contrast, in the other context, it is desirable to obtain the same random sequence several times to reproduce the simulation.

Every Processor in a computer is typically deterministic, that is, any algorithm execution with the same input should have equal outputs – this is the opposite of randomness. That means there is no algorithm capable of producing random numbers alone. Furthermore, as solutions, we should either introduce a real source of randomness into computers or try to cheat on the user by producing number sequences, which only seems random. Thus, we have two RNG types. We will cover them in depth next.

2.4 RNG Types

We have seen in the previous point that computers are deterministic so that we should either use a real source of randomness and introduce them to computers or generate numbers that look like random numbers. Thus, we have two main types of RNG, **True** RNG and **Pseudo** RNG.

2.4.1 TRNG

TRNG (or Nondeterministic RNG, Hardware RNG) is a type of RNG that tries to get randomness using random events in the real world to create its sequences. We call these random events **entropy sources** and these sources of entropy makes a true random number generator **unpredictable**.

2.4.1.1 TRNG Characteristics

Besides the unpredictability, the most known characteristic of TRNG is that its output cannot be reproduced again. [18]
Another characteristic of TRNGs is that they are non-periodic where the sequence numbers pattern is never repeated.

2.4.1.2 Problems of Using TRNG

The major disadvantage of these generators is that they rely on hardware because they use real-world phenomena like:

- Air pressure.
- Atmospheric noise.
- Radioactive source.
- Quantum systems.
- Mouse movements.
- Coin flipping.
- Rolling of dice.
- CPU clock.

- Computer Fan Noise.

To measure these phenomena, they use some physical devices capable of recording these kinds of events. So, these generators are vulnerable to physical attacks that can bias the sequences of numbers [19].
Furthermore, physical devices are usually vulnerable to damage over time. They can also have construction errors that can inevitably bias the sequences generated. [19]
Another related problem is that the amount of real random data is always limited. If we need a lot of random data, we have to wait; which is unacceptable for many applications. [19]

2.4.1.3 TRNG in the Real-world

The website Random.org and HotBits offer True RNG, using Atmospheric noise and Radioactive decay, respectively.
There are many other examples; some TRNG models include [20]:

- Noise based RNG.
- Free running oscillator RNG.
- Quantum RNG.

Finally, TRNG are suitable for many applications, such as government lotteries, simulation, gambling.

2.4.2 Pseudorandom Number Generator (PRNG)

This method of RNG works with deterministic algorithms to generate long sequences of numbers that appear statistically random, which are, in fact, entirely determined by an initial value, known as a seed.

We can define a Pseudo RNG more formally [21] by a structure (S, m, f, U g) where:

- S is a finite set of states.
- m is a probability distribution on S, called the initial distribution.
- $f\colon S \to S$, a transition function.
- U, a finite set of output symbols.
- $g\colon S \to U$, an output function.

Then the generation of random numbers is as follows:

- Generate the initial state (called the seed) s0 according to m and compute u0 = g(s0)
- Iterate for i = 1, 2, 3, …, si = f(si–1), ui = g(si).

The state transition function f and the output function **g** are the core key of an efficient PRNG,

Examples of PRNG Here are some of the known PRNG:

- Linear Gongruential Generator (LCG) [22].
- Mersenne Twister [23].
- Xorshift [24].
- Squares [25].
- Permuted Congruential Generator (PCG) [26].

We will talk about some of these algorithms in chapter six.

2.4.3 Cryptographically Secure PRNG (CSPRNG)

For cryptographic purposes, we have a unique kind of generator that is relevant for stream ciphers. This generator is a particular type of PRNG, where it is defined informally with an additional property; it is **unpredictable** [18]. Formally, for a given integer n , the output numbers of the generator si, si+1, ... , si+n−1, it is computationally **infeasible** to compute the following numbers si+n, si+n+1,... [18]

In our thesis, we are not going to use TRNG, and not a CSPRNG, just a regular PRNG, and that for multiple reasons, one of these reasons is the reproducibility of the random numbers.

2.5 RNG – Application

Random numbers are essential and relevant in computer science, they are also a crucial element to a large number of areas, including global optimization, probabilistic computation, artificial intelligence, sampling, cryptography, computational creativity, modeling, simulation, robotics, gambling, games and many more.

We can use computers to simulate natural phenomena, in engineering and the natural sciences, simulation is used extensively, and random numbers are required to make these things real. The simulation covers many fields, from the study of biological processes and nuclear physics (where particles are subject to random collisions) to operations research (where people come into, say, an airport at random intervals). [15]

On the other case, RNG uses this general term to describe any computational algorithm that employs random numbers or random sampling [15], different methods are use like the Monte Carlo simulations, which uses experiments with random numbers to evaluate mathematical expressions, and the experimental units are the random numbers. [27]

Another application of RNG is cryptography where many security algorithms and protocols need random numbers and bits to remain secure. Algorithms like AES, RSA, and ECC have been using random numbers to be challenging to break. For that, randomness is essential because it eliminates any reasoning. Therefore, it drops any predictability that the attacker might get.

The quality of the random numbers used directly to determine the security strength of the system. We require randomness in cryptography for the following applications [28]:

- Private keys for digital signature algorithms.
- Keys and initialization values for encryption.
- PIN and password generation.
- Keys for keyed MAC algorithms.
- Values to be used in entity authentication mechanisms.
- Values to be used in keys establishment protocols.
- The nonce for cryptographic communication.

2.6 Properties of RNG

PRNG have some properties, these can be either critical or desired. The following points depicts some of them:

1. **Randomness**: The most noticeable property a random number generator should satisfy is that it should be random; we usually interpret it to mean

that a random number generator should comply with statistical expectations regarding random systems [26].

2. **Uniformity**: That means the appearance of any number is equally probable everywhere in the distribution; in other words, uniformity requires that after a generator completes a full period, all outputs will have occurred the same number of times. [26]

3. **Independence**: That is, the current random number is not related to the previously generated numbers.

4. **Period**: It is the most fundamental mathematical concept underlying any PRNG. Any deterministic algorithm executed using finite memory must have a finite number of states, and thus any RNG algorithm must have a fixed period, after which it will repeat [26]; for that, we need algorithms that have a sufficiently long period, to make sure that no wrap-around over the cycle can occur in practice. [29]

5. **Efficiency**: run fast and use only a small amount of memory. [29]

6. **Portable**: works the same in different software/hardware environments [29]

7. **Reproducibility**: The generator should be able to generate the same stream of random numbers for the same seed, so it will be able to reproduce the same sequence as many times as we want [29].

8. **Seekability**: It is the availability of efficient jump-ahead methods that can quickly compute sj+a given sj, for any large a and any j. [29], not all PRNGs are seekable, for instance, many cryptographically secure random number generators are not seekable (over the long term) by design—we do

not want people to be able to use a backward jump to discover numbers that they produced in the past [26]. However, it is advantageous to have this property.

2.7 Testing and Evaluating

In the previous section, we have talked about randomness property, we said that we need to have a generator that can supply random sequences that have both independency and uniform distribution.

It is crucial to test any generator that claims to produce random sequences; for that, we have countless various tests for RNG and the sequences they produce. We can divide these tests into two distinguished groups: empirical tests and theoretical tests.

- Empirical tests are carried out on a sequence generated by an RNG and require no knowledge of how the RNG produces the sequence; these tests are based on a mathematical understanding of random system behavior. In practice, empirical testing has felled impressive mathematical credentials on random number generators.

- Theoretical tests, they are the best when they exist, are a priori tests requiring knowledge of the RNG structure, but not necessarily generating the sequence. [15]

Mainly we will focus on Empirical tests here, and we will give just a brief description of each test. In the following points, we will explore several empirical tests belonging to Statistical Tests and Statistical Test Suites.

2.7.1 Statistical Tests

The first approach to test random number generators is to use the vast collection of well-known statistical formulas.

In this method, each test examines a different quality that the PRNG should have. We gain confidence in any generator only after it passes a considerable number of tests.

Here are some of the statistical tests:

1. **Chi-squared test.** Karl Pearson initially published the Chi-square test in 1900; This test is used to ensure that the numbers are uniformly distributed in a sequence. We can apply this test to see whether our RNG generated numbers follow a uniform distribution. [15]

2. **The Kolmogorov-Smirnov Test.** The Kolmogorov-Smirnov (KS) test has its origins in a 1933 paper by A. N. Kolmogorov, it was designed to test the equality of probability distributions by quantifying the distance between an observed distribution function of the sample and the combined distribution function of the reference distribution,

   We can use this test to check the uniformity of the RNG generated numbers. [15]

3. **The runs test.** An indispensable quality for a random sequence is that it does not contain patterns. This test can be used to verify whether the generated sequence has an upward or downward trend or some cyclical pattern. [15]

4. **Gap Test.** Count the number of digits occurring in repetitions of a given digit and then use the Chi-square or Kolmogorov-Smirnov method to equate with the number of gaps predicted. [15]

2.7.2 Statistical test suites

Test suites are nothing but a set of statistical tests for measuring the quality of a random number generator; they work by performing some statistically well-understood task using a candidate generator and then checking the plausibility of the results by producing a p-value to help support or reject the null hypothesis. Here are the most widely used test suites

1. **NIST [30]** NIST One of the available suites for testing random number generators is the NIST suite. The purpose of this suite is to test bit sequences while keeping in mind that passing all NIST tests implies that a generator is a fit for cryptographic purposes (it contains fifteen well-documented statistical tests).

   Since cryptography has the strictest specifications for randomness, a generator that passes the NIST suite is also suitable for all other purposes. And when a generator fails the NIST suite, it could still be random enough to work in fields such as simulation, since the importance of using less than entirely random information are meager.

2. **TestU01 [31]** L'Ecuyer in 2007 made a significant enrichment to the world of random number generator testing when they designed the TestU01 statistical test suite.

   This test suite includes a large number of previously independently published tests, and they used them at scale, and it broadly expanded the scope and thoroughness of the RNG testing process.

3. **Diehard [32]** Diehard is a widely used test suite for random number tests invented by George Marsaglia in 1995. It is an update for the original random number test suite, Knuth's tests.

   Knuth designed its test before cryptography became an important industry, and the suite was later deemed too easy to pass for situations where enormous quantities of random numbers were needed.

Diehard by design is harder to pass than the Knuth test suite, achieving the role of a general-purpose battery for non-randomness detection.

2.8 Conclusion

The notion of building a system that generates randomness can seem like an inconsistency; even with that, over the last decades, researchers have developed many algorithms that can generate random numbers which can be served as a source of randomness for many applications.

In this chapter, we have described some characteristics of Random number generators, their types, and properties, we have also seen that Pseudo RNG does not produce real random numbers, instead, a random-looking number that can pass the statistical tests.

We also saw that the uniformity, independence, long period, and efficiency is the key to build a powerful PRNG.

Chapter 3

Three Domain Secure

3.1 Introduction

Nowadays, the fastest-growing field of payment is digital commerce, as more wired devices become payment devices. Consumers have several ways of paying than ever before, whether through a browser, a smartphone app, or a connected device. By the next five years, the 30 billion connected devices today will reach 75 billion. [33]

So when a customer makes a digital purchase in a unconventional store, it is much difficult to check both the transaction and the user identity. Industry research reports that half of the declined digital trade transactions are actually legal due to alleged fraud. [34]

Thus, it is more important than ever that the industry goes on to invest in innovative approaches to prevent fraud. Helping issuers and merchants to differentiate between good and bad transactions would mitigate fraud, thus allowing transactions to continue to occur at lightning speed. 3-D Secure is a significant step in this endeavor, which will help deter fraud and promote digital trade with fast, reliable authentication.

3.2 3-D Secure

Protected user authorization protocol for CNP (card-not-present) operations. This technology is designed to keep payments of goods and services on the Internet safe, protecting cardholders and issuers from card fraud. Initially, the protocol was developed and proposed by VISA [35].

Visa introduced the 3D Secure 1.0 in the early of mid-2000's and mentioned it as Verified by Visa, followed by other credit card schemes (Including MasterCard SecureCode, American Express SafeKey and JCB J/Secure). It is an XML-based messaging protocol allowing cardholders to authenticate with their card issuer when making online transactions of card-not-present (CNP). [36]

This version enables users to take advantage of password protection by adding a key to their card that is prompted when a transaction is initiated by the authentication process. Although the improvements that 3-D Secure 1.0 brings to the market in the fight against theft, the technology that only facilitates browser purchases has been limited as it was released years before smartphones, and the protocol is not designed to deliver good mobile user experience or to use the latest authentication methods. [37]

The payments industry recognized the need to develop a new, 3-D Secure specification because of the other disadvantages and represent current and future business needs better.

Therefore, as of January 2015, EMVCo, which is an entity that is jointly owned by major global payment solution companies American Express, UnionPay, JCB, Discover, MasterCard, and Visa, were developing the EMV 3DS 2 specification and 3-D Secure 2 were released in October 2016. This new version has emerged intending to address the disadvantages of its predecessor [38][39].

3.2.1 3-D Secure 2 Specification

EMVCo published EMV 3-D Secure core functions specification of version 2. The new specification provides an internationally interoperable framework that encourages the customer experience across all electronic business channels and connected devices when a cardholder is authenticated. As mentioned on the site of EMVCo [40], here are the specifications:

- Supports specific app-based purchases on mobile and other consumer devices and traditional browser-based e-commerce channels.
- Improves the consumer experience by enabling intelligent risk-based decisions that encourage frictionless consumer authentication.
- Delivers industry-leading security features.
- Specifies the use of multiple options for step-up authentication, including one-time passcodes as well as biometrics via out-of-band authentication.
- Details functionality that enables merchants to integrate the authentication process seamlessly into their checkout experiences for both app and browser-based implementations.
- Offers performance advancements for end-to-end message processing.
- Adds a non-payment message category to provide cardholder verification details to support various non payment activities.

3.2.2 How 3-D Secure 2 works

3-D Secure provides an authentication data link between digital merchants, payment networks, and financial institutions in order to be able to monitor and

exchange more transaction intelligence.
This protocol adds an additional step of user authorization when paying for purchase from an online store. The first step uses the card information (the expiration date, the name of the cardholder, and the card verification value (CVV).
In the second step, using the 3-D Secure protocol, the store's website shows the card bank's page, which proposes to enter an additional security code. The client can receive through a text message in his mobile phone, with the help of a card of one code or a particular device, and the code can be permanent, pre-established by the client.

3.2.3 3-D Secure 2 Ecosystem Components and Architecture

The three domains are the merchant/acquirer domain, interoperability domain, and the issuer domain. The following diagram is of the 3-D Secure 2 process workflow:

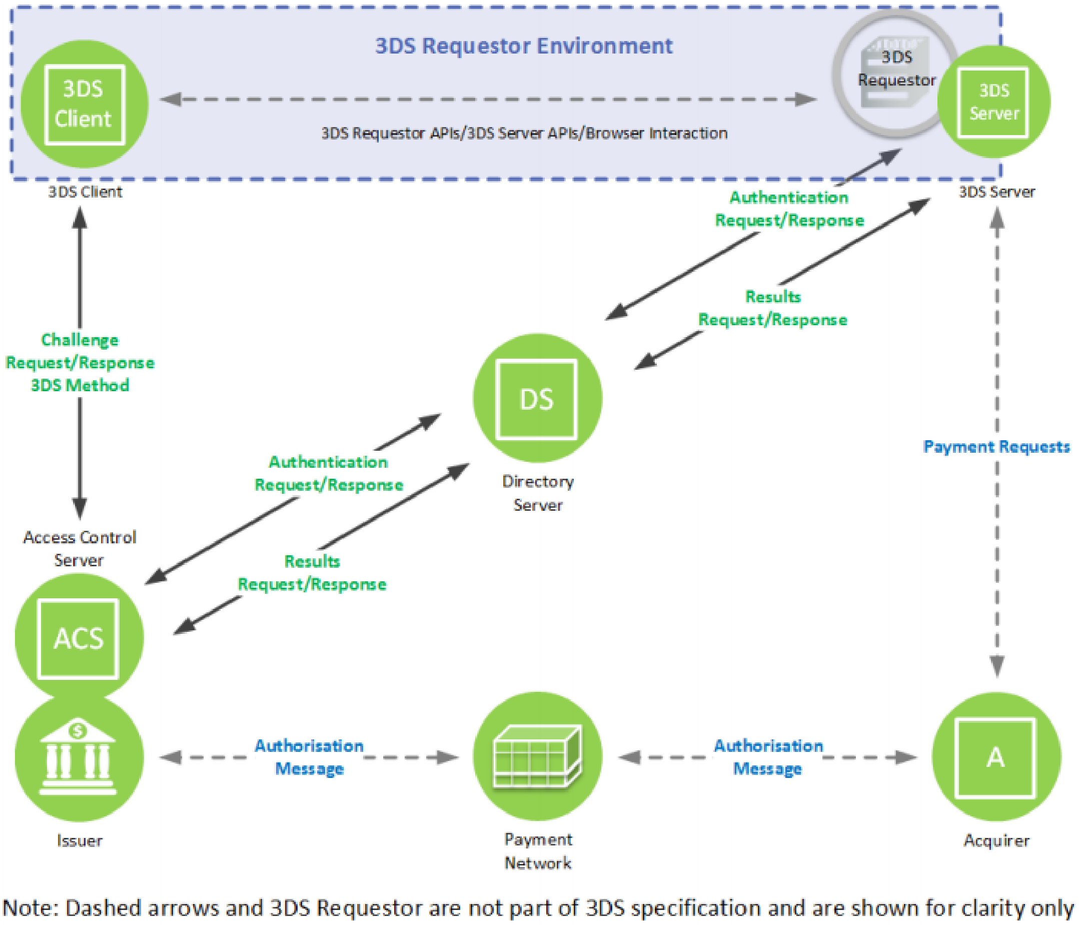

3DS Requestor Environment
3DS Client
3DS Requestor APIs/3DS Server APIs/Browser Interaction
3DS Requestor
3DS Server
3DS Client
3DS Server
Authentication Request/Response
Results Request/Response
Challenge Request/Response 3DS Method
DS
Directory Server
Payment Requests
Authentication Request/Response
Results Request/Response
Access Control Server
ACS
Authorisation Message
Authorisation Message
A
Issuer
Payment Network
Acquirer
Note: Dashed arrows and 3DS Requestor are not part of 3DS specification and are shown for clarity only

Figure 3.1: EMV 3DS Architecture. [2]

3.2.3.1 Acquirer Domain

The components of the acquirer domain are as follows [37]:

3DS Requestor Environment The 3DS Requestor Environment is a collective term for components managed by the 3DS Requestor supporting 3-D Secure. The Environmental components of 3DS Requestor include:

- **3DS Requestor**: The 3DS Requestor initiates the authentication request

(AReq) message and is the conduit for the 3-D Secure data from the Consumer Device. For example, this may represent the existing Merchant web server for online shopping.

- **3DS Client**: The 3DS Client is a part of customer Device that initiates 3-D Secure authentication. For example, in payment authentication, the 3DS Client is integrated into the Merchant Checkout as part of online shopping experience.
- **3DS Server**: The 3DS Server provides the functional interface that handles online transactions and simplifies communication between the 3DS Requestor and the Directory Server (DS).

3.2.3.2 Interoperability Domain

The Interoperability Domain consists of:

Directory Server Managed by a payment network. The DS performs a number of functions including Authentication of the 3DS Server with the ACS, the validation of the 3DS Requester (if it is registered and trusted), and maintains account and ACS routing data, and routing messages between the 3DS Server and the ACS.

3.2.3.3 Issuer Domain

The components of the acquirer domain are as follows:

- **Cardholder**: The Cardholder uses a Consumer Device to provide account information. If required, the Cardholder is required to give additional information for authentication.

- **Access Control Server (ACS)** The ACS includes the authentication rules and is controlled by the Issuer. ACS functions include: verifying whether a card number and the device type is eligible for 3-D Secure authentication, and authenticating the Cardholder or confirming the account information.

3.2.4 3-D Secure 2 Messages

This section introduces the messages defined for 3-D Secure [41].

3.2.4.1 Authentication Request Message (AReq)

The AReq is the initial message in the 3-D Secure authentication flow. In order to authenticate the Cardholder. The 3DS Server forms the AReq message. It may include Cardholder, payment, and Device information for the transaction. Just one AReq message is sent per authentication.

3.2.4.2 Authentication Response Message (ARes)

The ARes is the ACS response of the Sender to the message AReq. It could indicate that the Cardholder was authenticated, or that further information of the Cardholder is needed to complete the authentication. There is just one message with ARes per transaction.

3.2.4.3 Challenge Request Message (CReq)

The 3DS Server sends the CReq message. It initiates the interaction of the cardholder in a challenging flow and can be used to bring the authentication data from the holder. There is only one CReq message sent per challenge.

3.2.4.4 Challenge-Response Message (CRes)

The CRes message is the ACS response to the CReq message. It is the result of the Cardholder authentication. There is just one CRes response to every challenge.

3.2.4.5 Results Request Message (RReq)

The message on the RReq communicates the authentication or verification results. ACS transfers the request via the DS to the 3DS server. There is just one RReq message per each AReq message. The RReq message is not used in a Frictionless transaction.

3.2.4.6 Results Response Message (RRes)

The RRes message confirms receipt of the RReq message. The message is sent to the ACS via the DS by the 3DS Server. There is only one RRes message sent for each RReq message.

3.2.4.7 Preparation Request Message (PReq)

The PReq message is sent to the DS from the 3DS server to request information about the ACSs and the DS. This message is not part of the 3-D Secure Message Authentication Flow.

3.2.4.8 Preparation Response Message (PRes)

The PRes message is a DS reply to the PReq message. The 3DS Server can use the PRes message to store information about ACSs and DS. This message is not part of the 3-D Secure Message Authentication Flow.

3.2.5 3-D Secure 2 Authentication Flows

In this section, we will describe the flows specified for EMV 3-D Secure authentication.

3.2.5.1 Frictionless Flow

Frictionless Flow is one of the fundamental differences between 3DS 1 and 3DS 2. It initiates a 3-D Secure authentication process and consists of an AReq and an ARes message. It allows issuers to achieve sufficient authentication without communicating with the cardholder based on risk-based authentication implemented in the ACS. [37][42]

3.2.5.2 Challenge Flow

The Frictionless Flow switches to the Challenge Flow in case the ACS assess that further Cardholder interaction is required to complete the authentication. For example, a challenge could be needed if the transaction is considered high-risk, over certain thresholds.

3DS Requestors agree to continue with the challenge or to terminate the process of 3-D Secure authentication.

3.2.5.3 Non-payment Authentication

Non-payment authentication is one of the features added with 3DS 2, enabling the protocol to be used for mobile wallet cardholder identification and verification and not just browser-based payments. This flow is equivalent to the 3-D Secure 2 authentication flow during a web-based purchase but does not include payment-specific steps such as initiating payment, confirming, etc. [43]

3.2.6 New authentication options

3.2.6.1 Risk-based Authentication

Risk-based Authentication is the process by which Issuer can enforce frictionless payments for low-risk transactions. It allows the issuer to have the risk algorithm applied and the collection of risk rules generated accordingly. Issuers may implement their algorithm or use independent vendors for such implementation of risk engines. [44]
Usually, including the evaluation of the history of Cardholder/merchant transaction and transaction data, such as amount, location, and device information. [45]

3.2.6.2 Biometric authentication

Payment systems also widely encourage the use of biometric authentication tools such as fingerprint matching, voice, and facial recognition. This technology is becoming mature enough to offer a high level protection and efficiency, as well as useful consumer experience at checkout. [37]

3.3 Conclusion

3D Secure standard was proposed to reduce fraud and provide added security to online payments. It introduces “frictionless authentication” and improves customers’ purchase experience because of Strong Customer Authentication (SCA). After we discussed how 3D secure protocol works, and after discussing our authentication method, we are ready to dive into the analysis and design of the new system we are going to build. In the next chapter, we will talk about the different uses cases and the system’s context alongside the different processes that illustrate the dynamic aspects of the system.

Chapter 4

Analysis And Design

4.1 Introduction

Analysis and design steps are the foundations of the system we will create; these steps gives a precise and clear idea of our system, defining its scope and the overall operations.
In the next section, we present several diagrams that describe the different aspects of the our system and its operation (use cases, classes, sequences). The analysis shows a total abstraction that is independent of any technology or implementation.
This analysis is based on the results of the study of the existing, which allowed us to assess the current online banking systems, by perceiving the customer's needs, and to describe the good behavior that the new system must-have.

4.2 System Context

We will use a context diagram to describe the system's context and boundaries to be modeled, which are simply the inside and outside of the new system and its relationship with external entities.

He are the high-level components to make our system:

- Merchant Integration Management.
- Account Management.
- Transactions Management.
- Balance Management.
- Authentication System.
- Fraud Detection System.
- Payment Approval System.
- History Management.

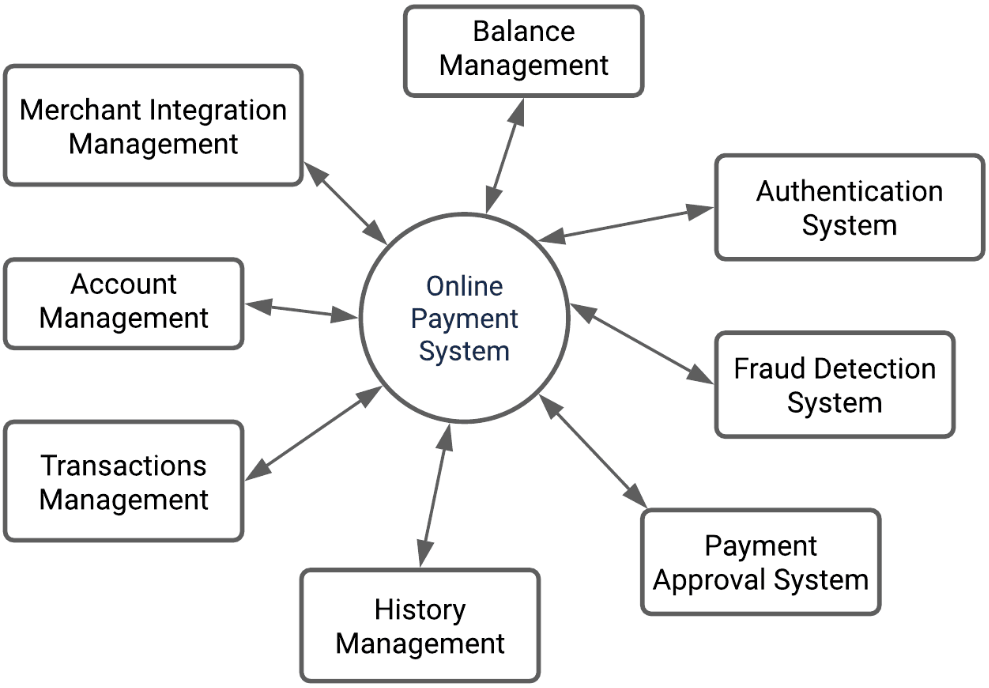


Figure 4.1: Context diagram of our system

4.3 Use Case Diagram

A use-case model describes a system's functional requirements in terms of use cases. It is a model of the system's intended functionality (use cases) and environment (actors). Use cases enable you to relate what you need from a system to how it delivers on those needs.

A use case is a description of the system behavior. That description is written from the perspective of a user who has just ordered the system to do some particular thing. A use case captures the visible sequence of events that a system goes through in response to a single user stimulus. [46]

After analyzing famous payment systems and the new system's requirements, we have created the following use case diagram:

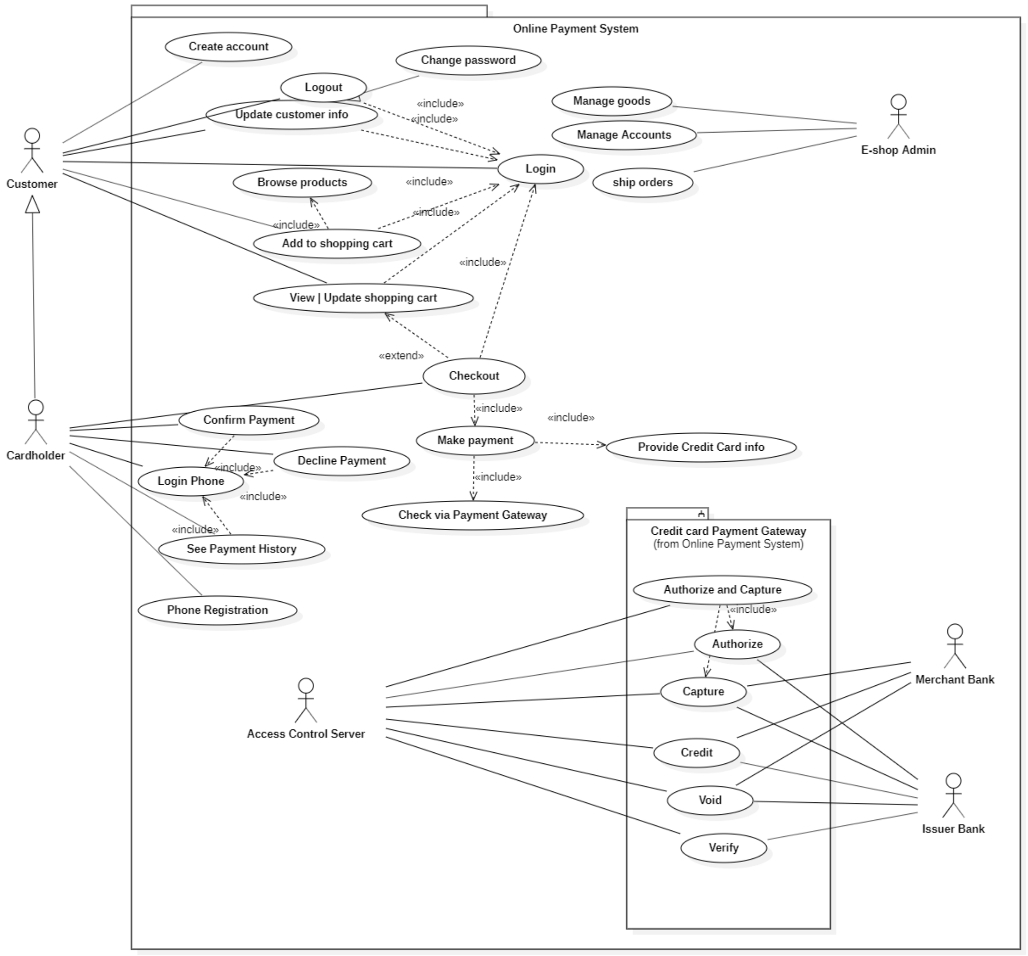


Figure 4.2: Use case diagram of our system

As the diagram suggests, we have several actors that interact in our system to provide the functionalities in a secure and performant way.

The primary actors for the system are the Merchant and the Cardholder, alongside different system actors that help accomplish the processes.

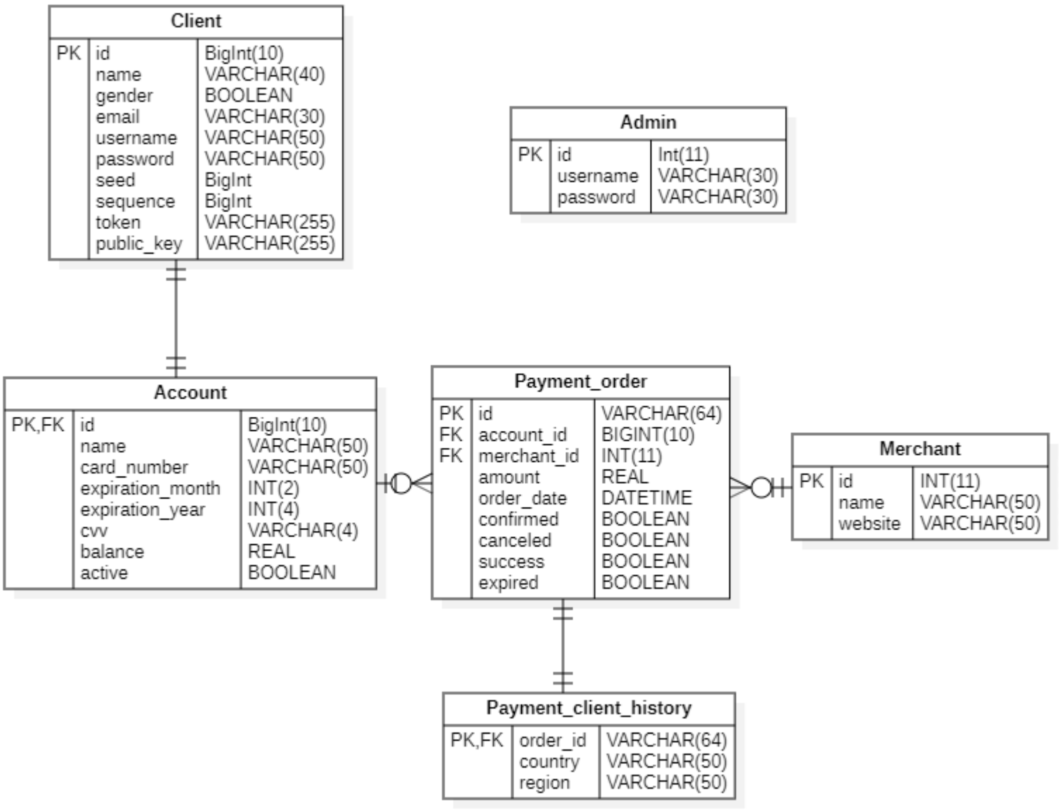
The merchant submits some credit card transaction request to the credit card payment gateway on behalf of a cardholder, and the bank of the cardholder may approve or reject the transaction based on different criteria.

4.4 Entity-Relationship Diagram

An Entity-relationship model (ER model) is generally used to describe the data structure of a given system, with the use of a diagram called the ER Diagram. This model will be the blueprint of a database that can later be implemented as a database in any DBMS.

An ER diagram presents the relationship between entity sets. An entity in terms of DBMS is a table in the database, so by showing the relationship amongst tables and their attributes, the ER diagram exposes a complete logical structure of a database.

In our system, the main entities are the users of the system, the merchant, payment transaction and histories, and orders to pay. Besides several other classes we used to accomplish the new system functionality, here is an ER diagram representing this concept:



Client

PK	id	BigInt(10)
	name	VARCHAR(40)
	gender	BOOLEAN
	email	VARCHAR(30)
	username	VARCHAR(50)
	password	VARCHAR(50)
	seed	BigInt
	sequence	BigInt
	token	VARCHAR(255)
	public_key	VARCHAR(255)

Admin

PK	id	Int(11)
	username	VARCHAR(30)
	password	VARCHAR(30)

Account

PK,FK	id	BigInt(10)
	name	VARCHAR(50)
	card_number	VARCHAR(50)
	expiration_month	INT(2)
	expiration_year	INT(4)
	cvv	VARCHAR(4)
	balance	REAL
	active	BOOLEAN

Payment_order

PK	id	VARCHAR(64)
FK	account_id	BIGINT(10)
FK	merchant_id	INT(11)
	amount	REAL
	order_date	DATETIME
	confirmed	BOOLEAN
	canceled	BOOLEAN
	success	BOOLEAN
	expired	BOOLEAN

Merchant

PK	id	INT(11)
	name	VARCHAR(50)
	website	VARCHAR(50)

Payment_client_history

PK,FK	order_id	VARCHAR(64)
	country	VARCHAR(50)
	region	VARCHAR(50)

Figure 4.3: ER Diagram of ACS Database

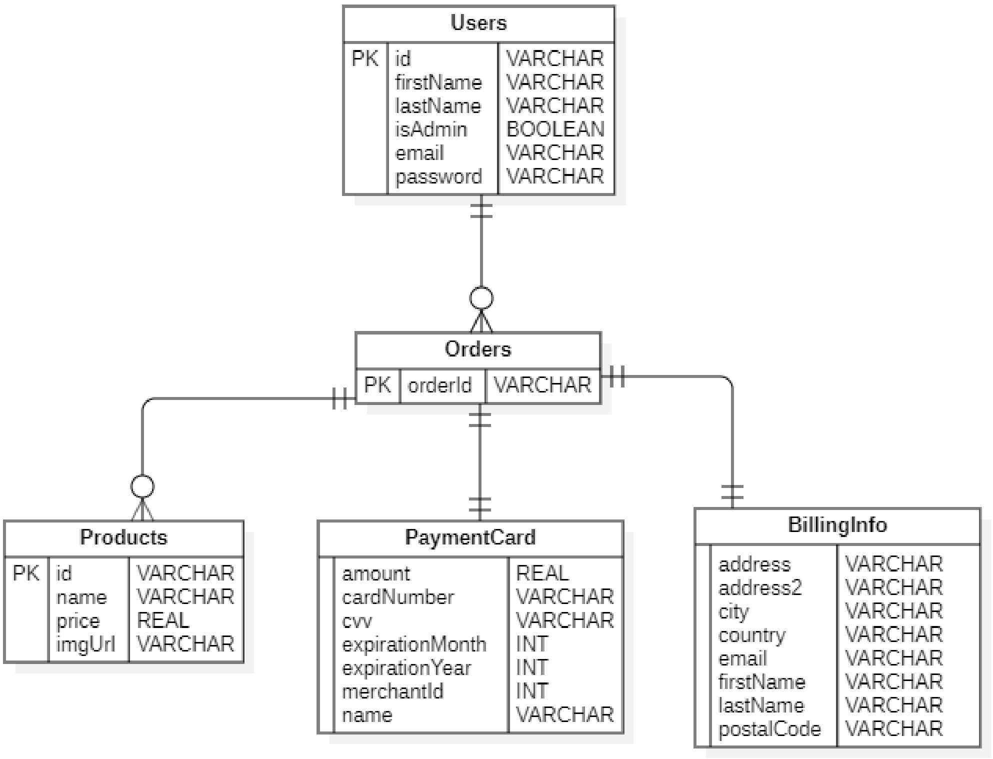


Figure 4.4: ER Diagram of E-commerce Database

4.5 Class diagram

The class diagram is a central modeling technique that runs through nearly all object-oriented methods. This diagram describes the types of objects in the system and various kinds of static relationships between them.

Relationships

- There are three principal kinds of relationships which are essential:
- Association: represent relationships between instances of types (a person works for a company, a company has several offices).

- Inheritance - the most apparent addition to ER diagrams for use in OO. It has an immediate correspondence to inheritance in OO design.

- Aggregation: Aggregation, a form of object composition in object-oriented design.

One of the most critical subsystems in our system is the ACS. Here is its class diagram:

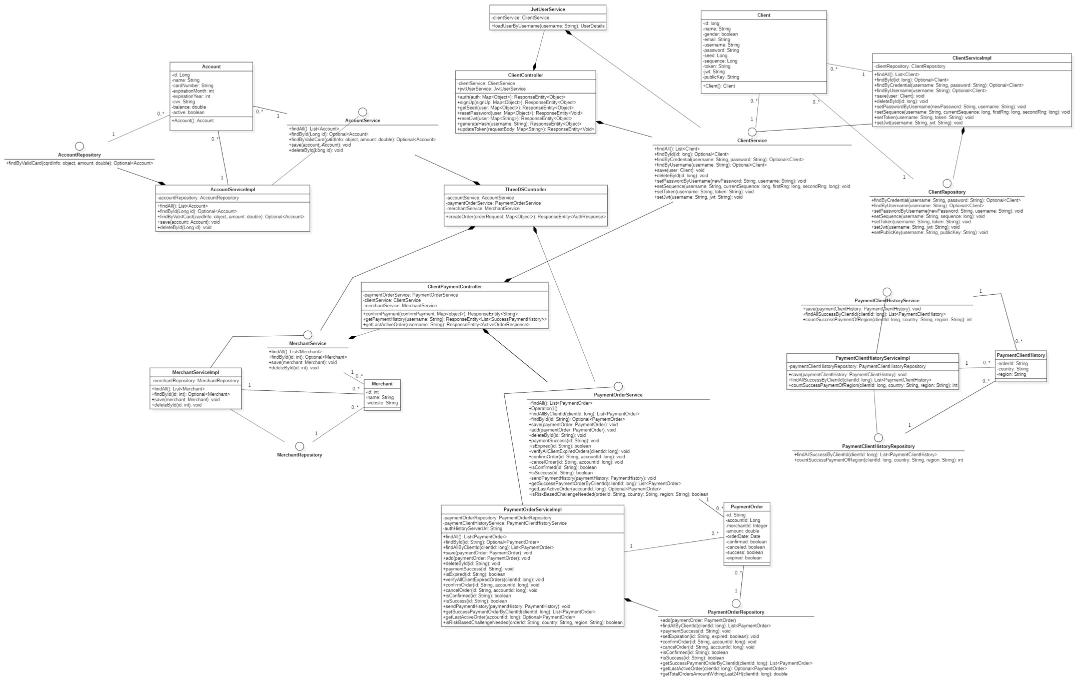

JwtUserService
-clientService: ClientService
+loadUserByUsername(username: String): UserDetails
Client
-id: long
-name: String
-gender: boolean
-email: String
-username: String
-password: String
-seed: Long
-sequence: Long
-token: String
-jwt: String
-publicKey: String
+Client(): Client
ClientServiceImpl
-clientRepository: ClientRepository
+findAll(): List<Client>
+findById(id: long): Optional<Client>
+findByCredential(username: String, password: String): Optional<Client>
+findByUsername(username: String): Optional<Client>
+save(user: Client): void
+deleteById(id: long): void
+setPasswordByUsername(newPassword: String, username: String): void
+setSequence(username: String, currentSequence: long, firstRng: long, secondRng: long): void
+setToken(username: String, token: String): void
+setJwt(username: String, jwt: String): void
Account
-id: Long
-name: String
-cardNumber: String
-expirationMonth: int
-expirationYear: int
-cvv: String
-balance: double
-active: boolean
+Account(): Account
ClientController
-clientService: ClientService
+jwtUserService: JwtUserService
+auth(auth: Map<Object>): ResponseEntity<Object>
+signUp(signUp: Map<Object>): ResponseEntity<Object>
+getSeed(user: Map<Object>): ResponseEntity<Object>
+resetPassword(user: Map<Object>): ResponseEntity<Void>
+resetJwt(user: Map<String>): ResponseEntity<Object>
+generateHash(username: String): ResponseEntity<Object>
+updateToken(requestBody: Map<String>): ResponseEntity<Void>
AccountService
+findAll(): List<Account>
+findById(Long id): Optional<Account>
+findByValidCard(cardInfo: object, amount: double): Optional<Account>
+save(account: Account): void
+deleteById(Long id): void
AccountRepository
+findByValidCard(cqrdInfo: object, amount: double): Optional<Account>
ClientService
+findAll(): List<Client>
+findById(id: long): Optional<Client>
+findByCredential(username: String, password: String): Optional<Client>
+findByUsername(username: String): Optional<Client>
+save(user: Client): void
+deleteById(id: long): void
+setPasswordByUsername(newPassword: String, username: String): void
+setSequence(username: String, currentSequence: long, firstRng: long, secondRng: long): void
+setToken(username: String, token: String): void
+setJwt(username: String, jwt: String): void
ClientRepository
+findByCredential(username: String, password: String): Optional<Client>
+findByUsername(username: String): Optional<Client>
+setPasswordByUsername(newPassword: String, username: String): void
+setSequence(username: String, sequence: long): void
+setToken(username: String, token: String): void
+setJwt(username: String, jwt: String): void
+setPublicKey(username: String, publicKey: String): void
AccountServiceImpl
-accountRepository: AccountRepository
+findAll(): List<Account>
+findById(Long id): Optional<Account>
+findByValidCard(cardInfo: object, amount: double): Optional<Account>
+save(account: Account): void
+deleteById(Long id): void
ThreeDSController
-accountService: AccountService
-paymentOrderService: PaymentOrderService
-merchantService: MerchantService
+createOrder(orderRequest: Map<Object>): ResponseEntity<AuthResponse>
ClientPaymentController
-paymentOrderService: PaymentOrderService
-clientService: ClientService
-merchantService: MerchantService
+confirmPayment(confirmPayment: Map<object>): ResponseEntity<String>
+getPaymentHistory(username: String): ResponseEntity<List<SuccessPaymentHistory>>
+getLastActiveOrder(username: String): ResponseEntity<ActiveOrderResponse>
PaymentClientHistoryService
+save(paymentClientHistory: PaymentClientHistory): void
+findAllSuccessByClientId(clientId: long): List<PaymentClientHistory>
+countSuccessPaymentOfRegion(clientId: long, country: String, region: String): int
PaymentClientHistoryServiceImpl
-paymentClientHistoryRepository: PaymentClientHistoryRepository
+save(paymentClientHistory: PaymentClientHistory): void
+findAllSuccessByClientId(clientId: long): List<PaymentClientHistory>
+countSuccessPaymentOfRegion(clientId: long, country: String, region: String): int
PaymentClientHistory
-orderId: String
-country: String
-region: String
PaymentClientHistoryRepository
+findAllSuccessByClientId(clientId: long): List<PaymentClientHistory>
+countSuccessPaymentOfRegion(clientId: long, country: String, region: String): int
MerchantService
+findAll(): List<Merchant>
+findById(id: int): Optional<Merchant>
+save(merchant: Merchant): void
+deleteById(id: int): void
MerchantServiceImpl
-merchantRepository: MerchantRepository
+findAll(): List<Merchant>
+findById(id: int): Optional<Merchant>
+save(merchant: Merchant): void
+deleteById(id: int): void
Merchant
-id: int
-name: String
-website: String
MerchantRepository
PaymentOrderService
+findAll(): List<PaymentOrder>
+Operation1()
+findAllByClientId(clientId: long): List<PaymentOrder>
+findById(id: String): Optional<PaymentOrder>
+save(paymentOrder: PaymentOrder): void
+add(paymentOrder: PaymentOrder): void
+deleteById(id: String): void
+paymentSuccess(id: String): void
+isExpired(id: String): boolean
+verifyAllClientExpiredOrders(clientId: long): void
+confirmOrder(id: String, accountId: long): void
+cancelOrder(id: String, accountId: long): void
+isConfirmed(id: String): boolean
+isSuccess(id: String): boolean
+sendPaymentHistory(paymentHistory: PaymentHistory): void
+getSuccessPaymentOrderByClientId(clientId: long): List<PaymentOrder>
+getLastActiveOrder(accountId: long): Optional<PaymentOrder>
+isRiskBasedChallengeNeeded(orderId: String, country: String, region: String): boolean
PaymentOrderServiceImpl
-paymentOrderRepository: PaymentOrderRepository
-paymentClientHistoryService: PaymentClientHistoryService
-authHistoryServerUrl: String
+findAll(): List<PaymentOrder>
+findById(id: String): Optional<PaymentOrder>
+findAllByClientId(clientId: long): List<PaymentOrder>
+save(paymentOrder: PaymentOrder): void
+add(paymentOrder: PaymentOrder): void
+deleteById(id: String): void
+paymentSuccess(id: String): void
+isExpired(id: String): boolean
+verifyAllClientExpiredOrders(clientId: long): void
+confirmOrder(id: String, accountId: long): void
+cancelOrder(id: String, accountId: long): void
+isConfirmed(id: String): boolean
+isSuccess(id: String): boolean
+sendPaymentHistory(paymentHistory: PaymentHistory): void
+getSuccessPaymentOrderByClientId(clientId: long): List<PaymentOrder>
+getLastActiveOrder(accountId: long): Optional<PaymentOrder>
+isRiskBasedChallengeNeeded(orderId: String, country: String, region: String): boolean
PaymentOrder
-id: String
-accountId: Long
-merchantId: Integer
-amount: double
-orderDate: Date
-confirmed: boolean
-canceled: boolean
-success: boolean
-expired: boolean
PaymentOrderRepository
+add(paymentOrder: PaymentOrder)
+findAllByClientId(clientId: long): List<PaymentOrder>
+paymentSuccess(id: String): void
+setExpiration(id: String, expired: boolean): void
+confirmOrder(id: String, accountId: long): void
+cancelOrder(id: String, accountId: long): void
+isConfirmed(id: String): boolean
+isSuccess(id: String): boolean
+getSuccessPaymentOrderByClientId(clientId: long): List<PaymentOrder>
+getLastActiveOrder(clientId: long): Optional<PaymentOrder>
+getTotalOrdersAmountWithingLast24H(clientId: long): double
0..*
1

Figure 4.5: Class diagram of ACS

The class diagram is a central modeling technique that runs through nearly all object-oriented methods. This diagram describes the types of objects in the system and various kinds of static relationships between them.

4.6 Sequence diagram

The Sequence Diagram models the collaboration of objects based on a time sequence. It shows how the objects interact with others in a particular scenario of a use case. With the advanced visual modeling capability, you can create a complex sequence diagram in a few clicks. Besides, some modeling tools such as Visual Paradigm can generate a sequence diagram from the flow of events you have defined in the use case description.

We will present three principal sequence diagrams the are the core processes in our new system:

1. **Payment Workflow:** When a given user makes a payment, a complex process will rise in the background, from identifying and authenticating the user to assessing a risk-based score of the transaction. A complete picture can be shown in the next diagram

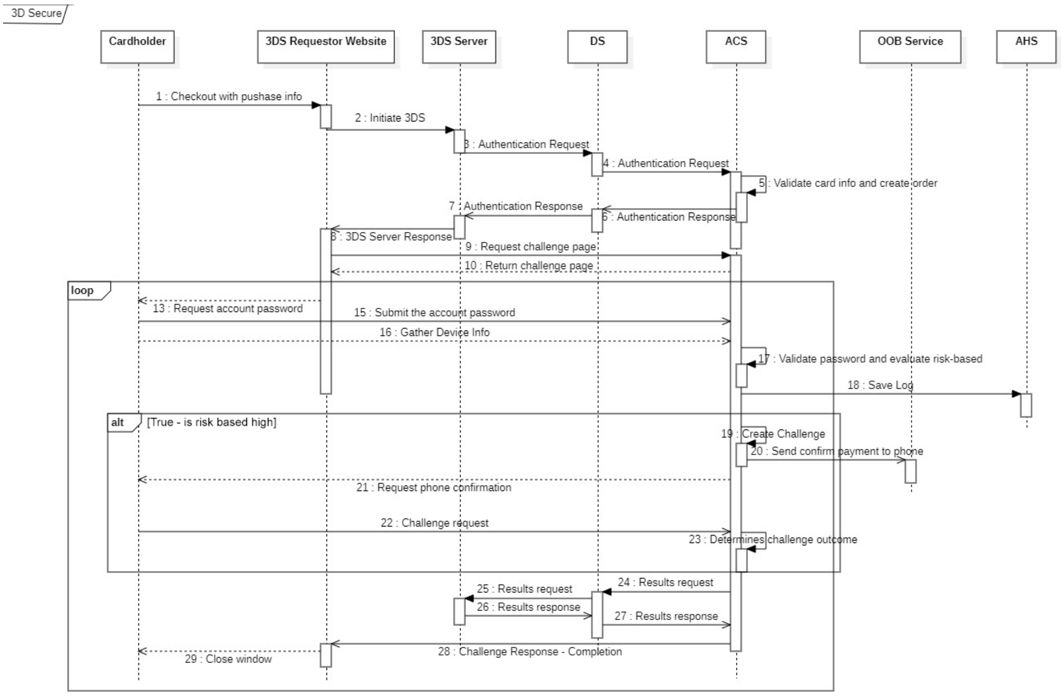

3D Secure
Cardholder
3DS Requestor Website
3DS Server
DS
ACS
OOB Service
AHS
1 : Checkout with pushase info
2 : Initiate 3DS
3 : Authentication Request
4 : Authentication Request
5 : Validate card info and create order
7 : Authentication Response
6 : Authentication Response
8 : 3DS Server Response
9 : Request challenge page
10 : Return challenge page
loop
13 : Request account password
15 : Submit the account password
16 : Gather Device Info
17 : Validate password and evaluate risk-based
18 : Save Log
alt
[True - is risk based high]
19 : Create Challenge
20 : Send confirm payment to phone
21 : Request phone confirmation
22 : Challenge request
23 : Determines challenge outcome
24 : Results request
25 : Results request
26 : Results response
27 : Results response
28 : Challenge Response - Completion
29 : Close window

Figure 4.6: Sequence diagram of ACS

2. **Registration Workflow:** When new users subscribe into the application, our system will exchange several parameters to assure the security of our users, such as the exchange of the public keys to ensure strong cryptography, and the seed and sequence, which are parameters for generating random numbers that are entirely different in each user. Here is a complete diagram that presents this process:

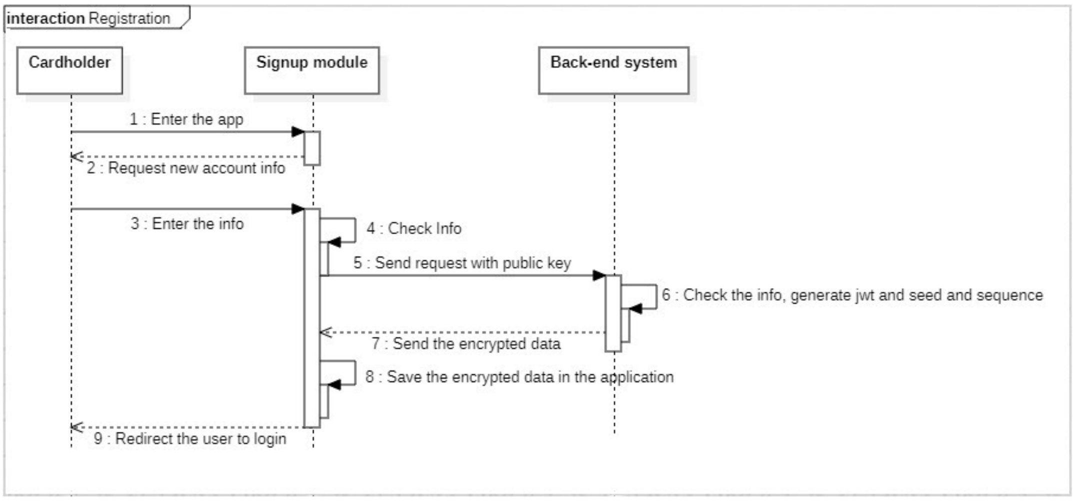

interaction Registration
Cardholder
Signup module
Back-end system
1 : Enter the app
2 : Request new account info
3 : Enter the info
4 : Check Info
5 : Send request with public key
6 : Check the info, generate jwt and seed and sequence
7 : Send the encrypted data
8 : Save the encrypted data in the application
9 : Redirect the user to login

Figure 4.7: Sequence diagram of Registration

3. **Login:** Authenticating users is also a critical task. Many things need to happen to ensure our cardholders' security, and we can resume all the process in the following sequence diagram:

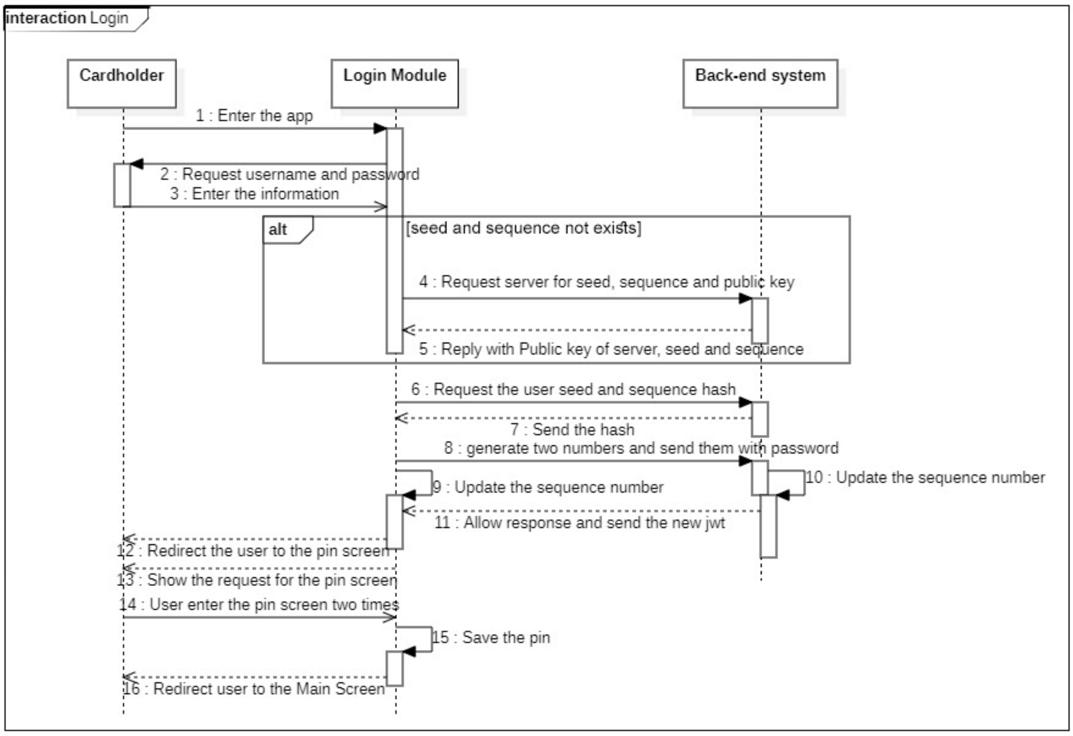

interaction Login
Cardholder
Login Module
Back-end system
1 : Enter the app
2 : Request username and password
3 : Enter the information
alt
[seed and sequence not exists]
4 : Request server for seed, sequence and public key
5 : Reply with Public key of server, seed and sequence
6 : Request the user seed and sequence hash
7 : Send the hash
8 : generate two numbers and send them with password
9 : Update the sequence number
10 : Update the sequence number
11 : Allow response and send the new jwt
12 : Redirect the user to the pin screen
13 : Show the request for the pin screen
14 : User enter the pin screen two times
15 : Save the pin
16 : Redirect user to the Main Screen

Figure 4.8: Sequence diagram of Login

4.7 Conclusion

In this chapter, we started the design of our new system. We first defined the requirements and functionalities of the system. We saw the generic design phase in which we detailed each module's study by determining the different types of classes and entities. Finally, we saw the system from the dynamic perspective, how each component interacts to deliver highly secure communication and authentication for payment purposes.

In each security point that we mentioned in this chapter, and the previous ones, we have worked on it separately, In the following chapter, we will present the different technological choices to address each security point, that way we ensure having a secure environment.

Chapter 5

Technological Choices

5.1 Introduction

Throughout this thesis, we have used many technologies and standards to achieve our work.

With all the choices available and the additional research and works, it is hard to choose what works best regarding various aspects such as performance and security; we will describe the technologies and their uses in our overall system in this chapter.

5.2 RNG Choice

We used the user's password with our two-factor authentication solution, plus two randomly generated random numbers. These two numbers can uniquely identify a single user because that pair of numbers will not be generated by any user other than this one.

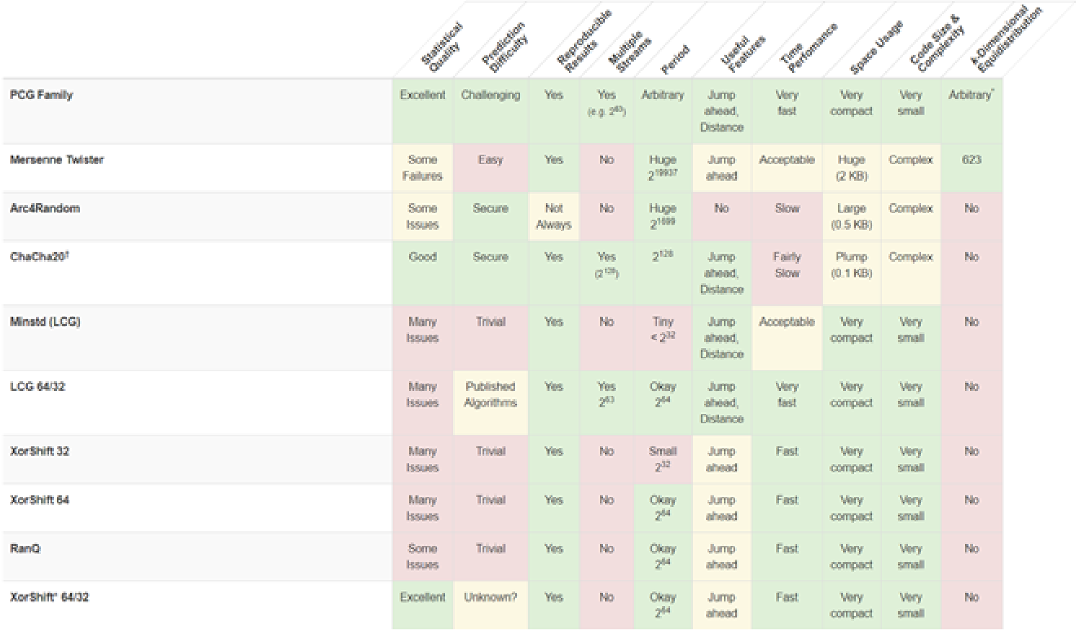
To generate numbers, we had to select an algorithm that will generate these numbers, plus additional property, which is the non-repetition of any two numbers in any series.

We had decided to choose the PCG [47] algorithm in our solution for generating random numbers, correctly, the XSH-RR 64/32 version.

5.2.1 Permuted Congruential Generator (PCG)

PCG was released in 2014; it is a pseudorandom number generation algorithm that applies an output permutation function to improve the statistical properties of a linear congruential generator modulo-2n. PCG achieves excellent statistical efficiency by utilizing fast and compact code, alongside small states and other useful assets.

Here is a comparison provided by the PCG official [48] web site:



	Statistical Quality	Prediction Difficulty	Reproducible Results	Multiple Streams	Period	Useful Features	Time Performance	Space Usage	Code Size & Complexity	k-Dimensional Equidistribution
PCG Family	Excellent	Challenging	Yes	Yes (e.g. 2^{63})	Arbitrary	Jump ahead, Distance	Very fast	Very compact	Very small	Arbitrary*
Mersenne Twister	Some Failures	Easy	Yes	No	Huge 2^{19937}	Jump ahead	Acceptable	Huge (2 KB)	Complex	623
Arc4Random	Some Issues	Secure	Not Always	No	Huge 2^{1699}	No	Slow	Large (0.5 KB)	Complex	No
ChaCha20†	Good	Secure	Yes	Yes (2^{128})	2^{128}	Jump ahead, Distance	Fairly Slow	Plump (0.1 KB)	Complex	No
Minstd (LCG)	Many Issues	Trivial	Yes	No	Tiny $< 2^{32}$	Jump ahead, Distance	Acceptable	Very compact	Very small	No
LCG 64/32	Many Issues	Published Algorithms	Yes	Yes 2^{63}	Okay 2^{64}	Jump ahead, Distance	Very fast	Very compact	Very small	No
XorShift 32	Many Issues	Trivial	Yes	No	Small 2^{32}	Jump ahead	Fast	Very compact	Very small	No
XorShift 64	Many Issues	Trivial	Yes	No	Okay 2^{64}	Jump ahead	Fast	Very compact	Very small	No
RanQ	Some Issues	Trivial	Yes	No	Okay 2^{64}	Jump ahead	Fast	Very compact	Very small	No
XorShift* 64/32	Excellent	Unknown?	Yes	No	Okay 2^{64}	Jump ahead	Fast	Very compact	Very small	No

Figure 5.1: PCG Comparison to other algorithms

PCG uses a linear congruential generator as the state-transition function, The output uses permutation functions on tuples to produce a much more random output than the RNG's internal state. [47]

It has a period of 2^{64}, 64 bits for the internal state, and an output of 32 bits; for the speed benchmark performance, it produces a random number in 0.66 nanoseconds. [47]

It comes with many great features such as multiple streams and seek-ability (moving forward or backward in the sequence) [47], which we need in our solution for efficiency search.

Before jumping to the code, we should mention that to generate random numbers, we need two inputs, something called the seed, and the other is the sequence number or the increment. Using these two numbers, the algorithm will create the inner state and use the sequence to generate each number. To ensure that each user has its own number sequence, each user needs to have a unique seed number among all the users, and this feature is assured by the server whenever a new user is created in the system; it will generate unique numbers.

Here is the code source of the 32 bit version of the algorithm:

```c
#include <stdint.h>
// Or something seed-dependent
static uint64_t state = 0x4d595df4d0f33173;
static uint64_t const multiplier = 6364136223846793005u;
// Or an arbitrary odd constant
static uint64_t const increment = 1442695040888963407u;

static uint32_t rotr32(uint32_t x, unsigned r) {
  return x >> r | x << (-r & 31);
}

uint32_t pcg32(void) {
  uint64_t x = state;
  unsigned count = (unsigned)(x >> 59); // 59 = 64 - 5

  state = x * multiplier + increment;
  x ^= x >> 18; // 18 = (64 - 27)/2
  return rotr32((uint32_t)(x >> 27), count); // 27 = 32 - 5
}

void pcg32_init(uint64_t seed) {
  state = seed + increment;
  (void)pcg32();
}
```

5.3 Risk-Based Authentication

One of the significant advantages of using 3 Domain Secure version 2.0 is evaluating each transaction with a risk score to see whether it needs extra checks, so less disruptive authentication will occur and better user experience.

Another term for this concept is Frictionless authentication. It allows businesses and their payment provider to send more data elements on each transaction to the cardholder's bank, such as contextual data.

Suppose this data is enough for the bank to trust the real cardholder. In that case, the latter is making the purchase, the transaction goes through the **frictionless** flow, and the authentication is completed without additional requirements.

Nevertheless, if the bank decides it needs further proof, the transaction is sent through the ”challenge” flow, and in our case, the customer will do the challenge in his phone with our mobile application.

To assess whether a transaction requires an additional verification to make sure that the real cardholder is making the purchase, we have used the customer location and the transaction amount.

The location is used to see if this is the usual place the customer is shopping from, if a new site appears, we check it and then add it to the trusted places for further purchases.

The amount is also used to check if the transaction exceeds a specific value we set or if the previous transaction's sum exceeds another specified value in the last 24 hours.

Here is a formal pseudo-algorithm for our risk-based authentication in our new system:

```
inputs: credit_card_information, user_region, transaction_amount
algorithm:
 if (user_region is in previous_approved_regions):
   if (transaction_amount > normal_threshold or
     (previous_amounts in last 24 hours) > threshold_of_24_hours):
     do_validation_check()
   else:
     do frictionless authentication
 else:
    result = do_validation_check()
    if(result == true):
    add user_region to previous_approved_regions
```

5.4 Key Exchange Protocols

We want to exchange the data between our back-end and mobile applications securely in our solution, so we need to apply cryptography to protect the user data. The first thing to tackle when implementing cryptography is to set a key to use for both parties that encrypt and decrypt the data; we are talking about key exchanges.

Key-exchange protocols are mechanisms that allow two parties already communicating over an insecure channel to generate a shared secret key without revealing it to someone listening to their conversation.

Our work has used the Elliptic Curve Diffie–Hellman Key Exchange or **ECDH**, which relies on Elliptic curves' power alongside the Diffie and Hellman Protocol. The allows two parties having each an elliptic curve public-private key pair, to establish a shared secret key over an insecure channel. [49]

ECDH is very similar to the classical **Diffie–Hellman Key Exchange (DHKE)** algorithm, but it uses **ECC point multiplication** instead of **modular exponentiations**. The basic of ECDH is the property of EC points: **(a * G) * b = (b * G) * a**

Suppose we have two **secret numbers a** and **b** (which means two **private keys**, belonging to Alice and Bob) and an elliptic curve with generator point **G**. In that case, we can exchange over an insecure channel the values **a * G** and **b * G** (the result of the multiplication is the **public keys** of Alice and Bob), and then we can derive a shared secret:

secret = (a * G) * b = (b * G) * a

The above equation takes the following form:

$alicePubKey * bobPrivKey = bobPubKey * alicePrivKey =$ **secret**

The ECDH algorithm (Elliptic Curve Diffie–Hellman Key Exchange) is trivial:

1. **Alice** generates a **random** ECC key pair: **alicePrivKey**, alicePubKey = alicePrivKey * G}
2. **Bob** generates a **random** ECC key pair: **bobPrivKey, bobPubKey** = bobPrivKey * G
3. Alice and Bob **exchange their public keys** through the insecure channel (e.g., over the Internet)
4. **Alice** calculates **shared key** = bobPubKey * alicePrivKey
5. **Bob** calculates **shared Key** = alicePubKey * bobPrivKey
6. Now both **Alice** and **Bob** have the same **sharedKey** == bobPubKey * alicePrivKey == alicePubKey * bobPrivKey

The other vital thing to mention about elliptic curve cryptography is the key size; the elliptic curve requires less key length than the popular algorithm RSA to achieve the same security level. [50]

The choice of key bit lengths in the range of 160–256 in ECC will provide security equivalent to 1024–3072-bit in RSA.

Here is a table that shows the comparison:

Minimum size (bits) of Public Keys			Key Size Ratio
Security (bits)	DSA / RSA	ECC	ECC to RSA / DSA
80	1024	160-223	1:6
112	2048	224-255	1:9
128	3072	256-383	1:12
192	7680	384-511	1:20
256	15360	512+	1:30

Table 5.1: RSA and Elliptic Curve Key Size Comparison

5.5 Cryptography Algorithm

Now we have a shared key between the server and the mobile application; we can encrypt and decrypt the data sent between these two parties. Primarily, we need that encryption to secure the first connection to exchange the seed and the sequence numbers required by the PCG Algorithm to generate two numbers.

Many algorithms provide proper security encryption and excellent performance results in both encryption and decryption operation. We decided to use the NIST recommended algorithm Advanced Encryption Standard (AES) to provide highly secure cryptography using the previously generated key.

5.5.1 Advanced Encryption Standard

AES is the most widely used symmetric cipher today. AES has a robust mathematical structure, unlike DES, the basics of most of its operations are on arith-

metic in the finite fields F28 and F2.
Even though the term ”Standard” in its name only refers to US government applications, the AES block cipher is also mandatory in several industry standards and is used in many commercial systems. [49]
The commercial standards that include AES are the Internet security standard IPsec, TLS, the Wi-Fi encryption standard IEEE 802.11i, and the secure shell network protocol Secure Shell (SSH), the Internet phone Skype and numerous security products around the world. To date, there are no attacks better than brute-force known against AES. [49]

5.5.1.1 How it works

AES is a block cipher that does not rely on the Feistel cipher; instead, it is designed as a substitution-permutation network or SP-network. However, AES does have several similarities with DES. Block ciphers based on the SP-network design consist of a series of rounds, each consisting of a key addition phase, a substitution phase, and a permutation phase.
The idea is that the permutation phase aims to produce an avalanche effect by spreading out differences in input to other parts of the state as quickly as possible, performing a diffusion process. The substitution phase is the main non-linear component, and this aims to introduce as much non-linearity, or confusion, into the output as possible.

5.6 Digital Signature

In our solution, we also want to validate the authenticity of the two parties’ messages. When it comes to verifying digital messages’ integrity and authenticity, we have to use digital signature algorithms to know whether a transmitted message

was altered or not.
Multiple powerful algorithms can provide such a feature. We need balance in terms of efficiency and security, for that we used the Elliptic Curve Digital Signature Algorithm (ECDSA) alongside SHA256.
Here are NIST-approved digital signature algorithms

- **DSA.**
- **RSA.**
- **ECDSA (our choice).**

While RSA is a secure algorithm, Elliptic curves have several advantages over RSA. In particular, in the absence of strong attacks against elliptic curve cryptosystems, bit lengths in the range of 160–256 bit can be chosen, which provide security equivalent to 102 –3072-bit RSA schemes. [49]

5.7 Secure Communication Choices

To secure communication between our server and different entities in the system, we used the SSL certificate scheme to provide highly secure transmission. That way, our users' information are protected against any attacks.
We used a default configuration to create a certificate that contains the following parameters: '

- The cryptographic algorithm to generate the key pair is RSA;
- The size of the key used is 2048 bits.
- The signature algorithm is SHA256 with RSA.
- The encryption and decryption algorithm is AES 256

5.8 Token Authentication

To protect our server endpoint for unallowed calls, we used the token authentication mechanism to prevent access to only the allowed parties to call the endpoint. Our user's application requests the endpoints for login, payment history, payment confirmation, and other services; we want only our users to access it. Any registered user will have a token to make him allowed to request the endpoints.
To be exact, we used Json Web Token (JWT), an open standard (RFC 7519). It works like the following:

- The server generates a **token** that certifies the user identity.
- Sends the token to the client.
- The client will send the **token** back to the server for every subsequent request.
- The server verifies the token to know that the request comes from a particular identity.

5.9 Conclusion

This chapter discussed several security issues and how to solve them using some technologies and protocols. In the next chapter, we will see the software, languages, and frameworks we used during the development and some of our application screenshots.

Chapter 6

Implementation

6.1 Introduction

After we had talked about the analysis and design, expressing the needs and modeling the system to be developed, we will move to the final phase of our graduation project, this phase will discuss the big and the subtitles of how we had implemented these models to create our solution to the problem.
The implementation of our project requires a number of technology and development tools. We have chosen them on the basis of their characteristics that we deem suitable to meet our needs.
We will see our work environment, the software, and IDE's we used then we move to the languages, frameworks, and database systems, and finally, we will wrap up this chapter by presenting the main interfaces of the different users of our application.

6.2 Mobile and Web Platforms

During this project, we have developed mainly three software that interact with each other to provide the described functionalities and features to the users.

Ecommerce: We first developed the E-Commerce website, which provides the customers with some books to buy, this website will help us demonstrate the use of our payment system throughout buying books and checkout.

Mobile Application: We have also implemented an Android application that responds to our customers need, it has the ability to register new users, check their identity, confirm and decline payment request, and show users histories, we will present some of the screenshots later on.

Back-end Server: This is the main and the core component in our system, it interacts with the Android application throughout RESTful requests, it has all the security measures required to provide a highly secure environment to our users, and to provide payment functionalities to merchant websites (like the e-commerce we built).

6.3 Tools and Technologies

6.3.1 Server

Intellij IDEA: IntelliJ IDEA is an Integrated Development Environment (IDE) for JVM languages. It is developed by JetBrains, and it provides great features like code completion and navigation, integration with build/packaging tools like maven and gradle, It supports version control systems like Git, and many other features.

Java: Java is a class-based, object-oriented programming language. It is a general-purpose programming language intended to let application developers write once, run anywhere, meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. it also provides a high level of security in code and scalability in entreprise systems.

Apache Maven: Maven is a build automation tool mainly used for Java applications, and is a tool for software comprehension and project management. Maven is able to manage the build and lifecycle, report, and the documentation of a project from a central piece of information, based on a Project Object Model (POM) concept.

Spring: Spring is the most popular open source application development framework for enterprise Java. Used to produce high-performance, testable, and reusable code. Spring is used to create web applications over the Java EE platform. The Spring Framework targets to facilitate the use of Java EE development and promote good programming practices.

Spring Boot: Spring Boot is a Java-based open source framework that is used to build stand-alone and production ready spring applications that you can just run. This offers efficient management of REST web services, back-end server and enterprise application in general.

Spring Data: The goal of Spring Data is to provide a common and reliable, Spring-based programming model for data access while maintaining the unique characteristics of the underlying data store. It facilitates the use of technologies for data access, relational and non-relational databases, and more. It is an umbrella project that includes multiple subprojects unique to a given database.

The projects are being developed by working with a lot of the companies and developers behind these exciting technologies.

Spring MVC: The Spring MVC is a framework that provides architecture for the Model – view – controller (MVC) that divides the related program logic into three interconnected elements which is the input logic, business logic, and UI logic, while having a loose coupling between them.

Spring Security: Spring Security is an efficient and easily configurable framework for authentication and access control. It is the standard used to secure Spring-based apps.
Spring Security is a framework which aims to provide Java applications with both authentication and authorisation. As all Spring projects, Spring Security's true strength lies in how quickly it can be extended to fulfill personalized specifications.

Web Technologies

1. **HTML5 (Hypertext Markup Language 5):** is a markup language used to describe the structure and content of web pages.

2. **CSS3 (Cascading Style Sheets):** is a language for managing the presentation and formatting of web pages. (Positioning of elements, fonts, colors, sizes, etc ...).

3. **JavaScript (JS):** it is an object oriented scripting language used to energize web pages and allow user interaction. It is run at the browser level.

4. **Bootstrap Framework:** Bootstrap is a CSS Framework which has a grid to facilitate the management of the formatting of HTML pages. It also offers

design components based on HTML and CSS with a Responsive design that allows a display that adapts to the size of the screen, whether it is a tablet, a smartphone or a computer.

5. **Thymeleaf:** Thymeleaf is a modern server-side Java template engine capable of rendering HTML, XML, JavaScript , CSS, and even plain text, for both web and standalone environments. Thymeleaf's main objective is to provide an elegant and highly-maintainable way to build models. It strengthens internal collaboration and fills the divide between design and engineering teams, which also helps them to thoroughly test models. We used this template engine in the 3D-Secure challenge confirm payment form.

MySQL: MySQL is a Database Management Systems. It consists of a data and rights definition language as well as a data manipulation language. It has the advantage of being portable (it can be compiled on several platforms like Windows, Unix... etc.). Moreover, it is easy to use, standard (it uses SQL), robust, free, and it is fully supported by many programming languages.

JavaFX: JavaFX is a Java framework for building RIA (Rich Internet Application) created by Sun Microsystems which now belongs to Oracle, JavaFX becomes the official graphical user interface (GUI) creation tool of the Java language, for all kinds of application (mobile applications, desktop applications, embedded systems, etc.). JavaFX contains a variety of tools, including audio and video media, 2D and 3D graphics. We have used this framework to build management applications for our 3-D Secure system.

6.3.2 E-Commerce

Visual Studio Code: Visual Studio Code is Microsoft's free source-code editor. it provides debugging support, smart code completion, syntax highlighting, code refactoring, and more. We used this software to develop the Angular web application in our system.

Angular: Angular is an open-source framework created by the Google team, based on TypeScript and used in the front-end to create efficient and sophisticated single-page web applications. Angular is a complete rewriting from the same team that constructed AngularJS. It aims to simplify the development and testing of such applications by providing a framework for client-side MVC and MVVM architectures, together with components commonly used in wealthy Internet applications.

Typescript: TypeScript is an application-scale JavaScript language. It adds optional types of JavaScript that support utilities on any Operating system for JavaScript large-scale apps on any browser and host. TypeScript compiles for readable, and standards-based JavaScript.

Firebase: Firebase is a platform developed by Google for creating mobile and web applications. It has several services that help to build mobile and web applications much faster because of the serverless nature it uses.
We have used the following services from the database:

- We have used the Firebase Firestore Database as a NoSQL database for our eCommerce to store prices and descriptions of items.
- We used Firebase Storage to store the corresponding product images.

- And finally, we also used Firebase cloud messaging to deliver notifications to our users when they need payment confirmation.

6.3.3 Android client

Android Studio: Android Studio is the standard Integrated Development Environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. We used this software to develop and debug the Android application in our system.

Kotlin: Kotlin started at JetBrains, the company behind IntelliJ IDEA, in 2010, it is a cross-platform, statically typed, general-purpose programming language with type inference. It interoperates fully with Java, and the JVM version of Kotlin's standard library depends on the Java Class Library. Still, type inference allows its syntax to be more concise.

Version Control System The VCS, for short, is a system that allows multiple managing revisions of the same unit of information. Software developers use it to keep track of the revision of the project. It can be used by anyone who works with digital assets, Designers, scriptwriters, and so one.
It is useful because:

- Enforce Discipline with VCS
- Archiving Versions of the same files
- Maintain Historical Information
- Enable Collaboration with others
- Recover from accidental deletion or edits

- Conserve Disk Space and managements.

There are many Version Control Systems out there, and Git is an example of VCS we used in our project to manage our code versions to recover in case of error and enable collaboration between us and because it is simple and easy to learn. Also, it provides a graphical Web interface Called GitHub to see and visualize the content.

6.4 Platforms Presentations

As you have noticed, our system is composed of different entities that interact with each other as the following diagram shows, in this section, we will present some of the screenshots of our newly implemented system.

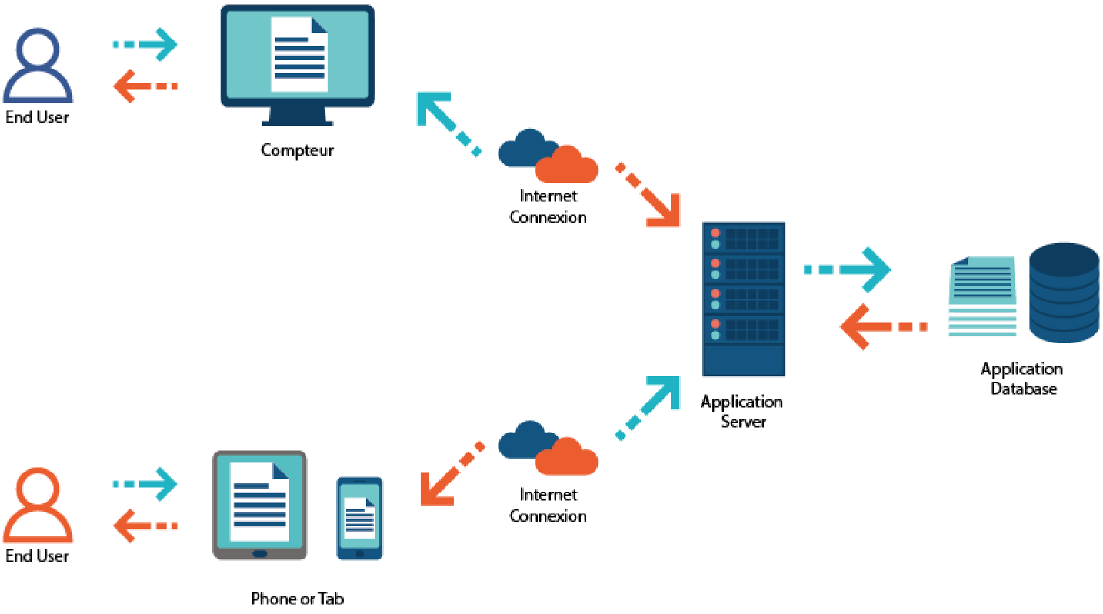


Figure 6.1: Our System View

6.4.1 Ecommerce

To demonstrate the payment process we had to create an E-commerce website that acts as a merchant website, it contains books and has some basic functionality to allow users to checkout and enter their payment preferences to buy books.

Home page (browse goods): From this page, you can select the goods and the quantity you want to buy.

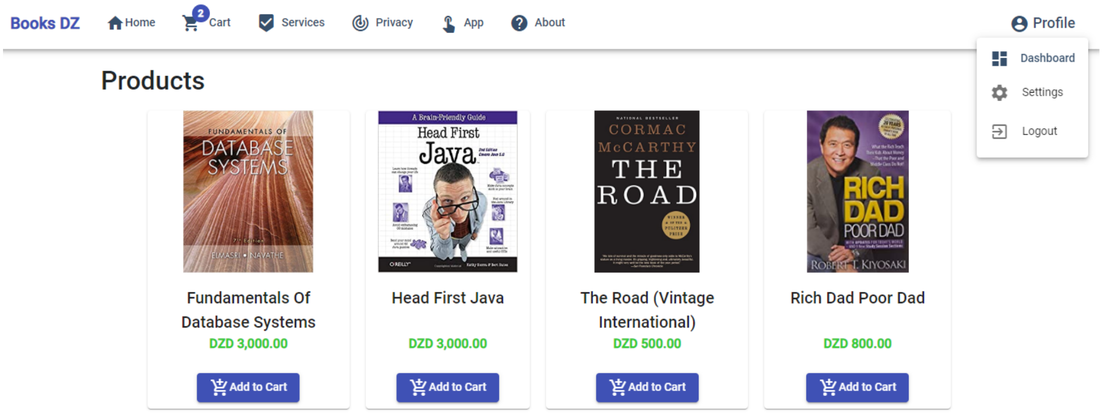
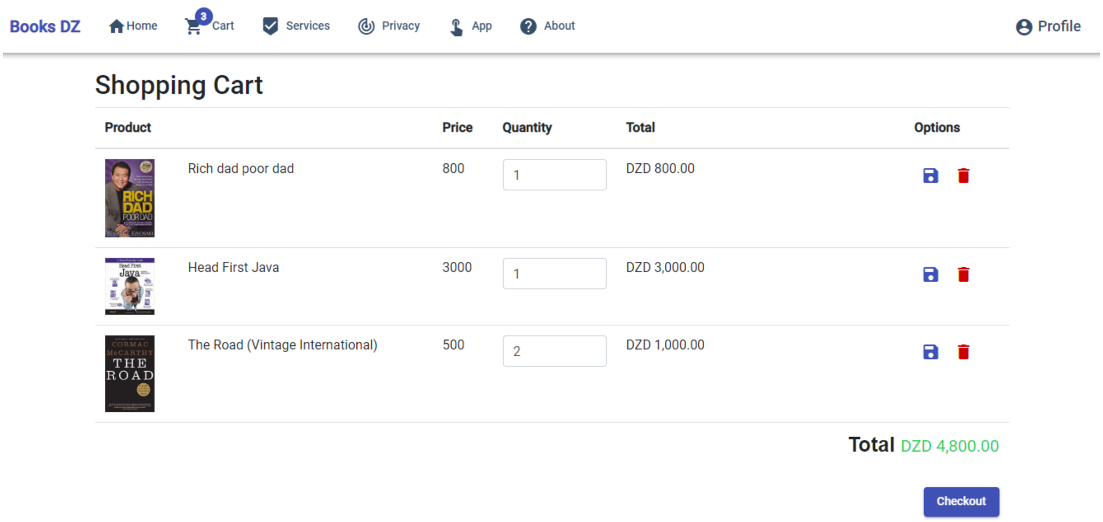


Figure 6.2: Ecommerce - Home page

Shopping Cart page: From this page, you can manage the products selected to buy by deleting or changing their quantity.



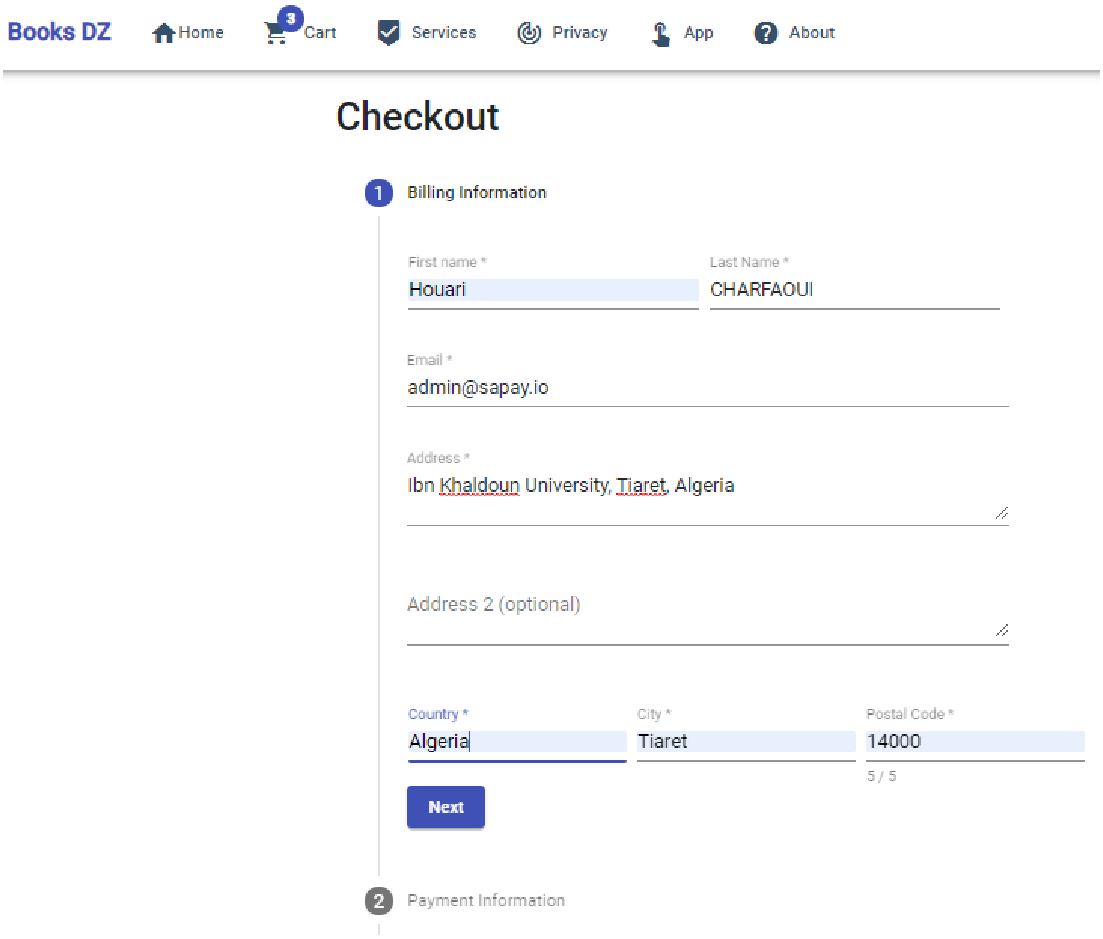
Shopping Cart

Product		Price	Quantity	Total	Options
	Rich dad poor dad	800	1	DZD 800.00	
	Head First Java	3000	1	DZD 3,000.00	
	The Road (Vintage International)	500	2	DZD 1,000.00	

Total DZD 4,800.00

Figure 6.3: Ecommerce - Shopping card page

Checkout 1 - Billing information: This is the first step in the checkout, from this page you fill in the billing information that you want to receive the package to.



Checkout

1 Billing Information

First name *
Houari

Last Name *
CHARFAOUI

Email *
admin@sapay.io

Address *
Ibn Khaldoun University, Tiaret, Algeria

Address 2 (optional)

Country *
Algeria

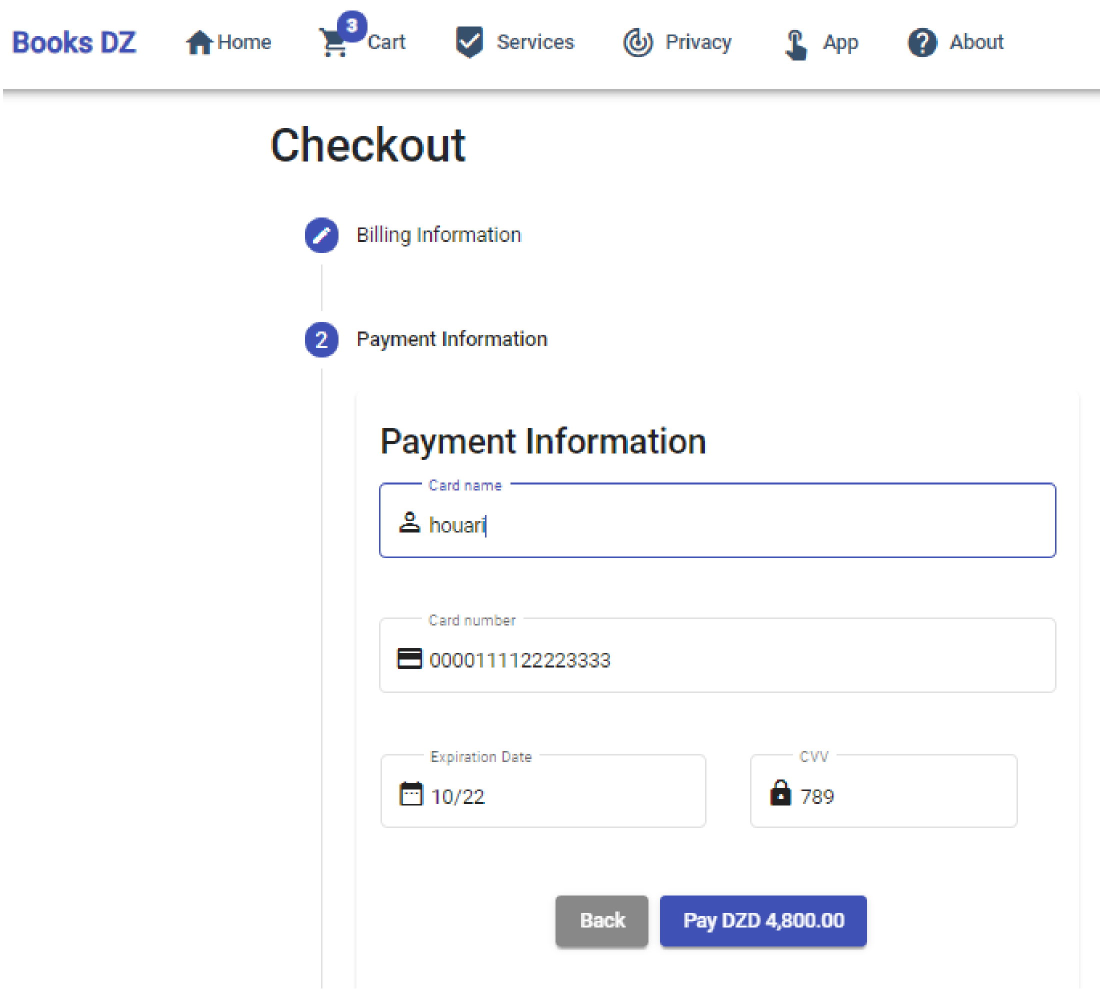
City *
Tiaret

Postal Code *
14000
5 / 5

2 Payment Information

Figure 6.4: Ecommerce - Fill billing information on checkout page

Checkout 2 - Credit Card information: From this page, you fill in the credit card information with which you want to buy the cargo.



Checkout

Billing Information

2 Payment Information

Payment Information

Card name: houari

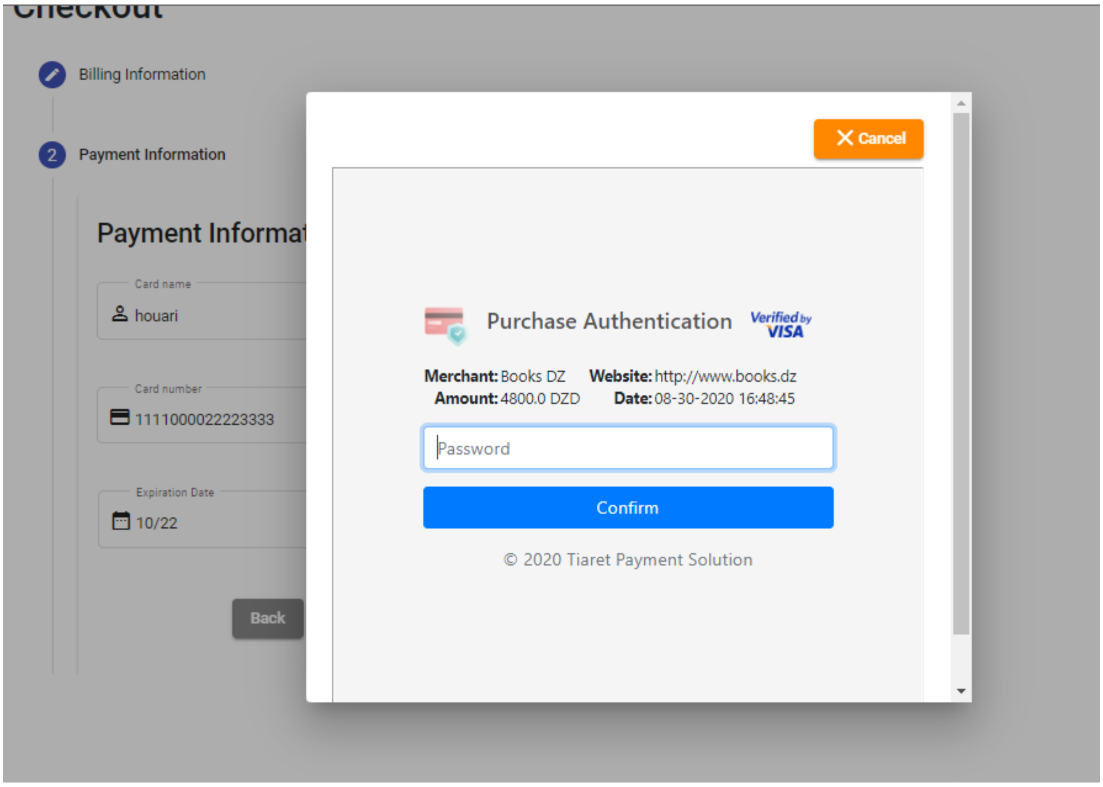
Card number: 0000111122223333

Expiration Date: 10/22

CVV: 789

Figure 6.5: Ecommerce - Fill credit card information on checkout page

Checkout 3 - ACS Purchase Authentication: The ACS (issuer bank) confirm the identity of the cardholder by opening an iframe on the e-commerce website and asked to fill additional information (password). Which means there is a direct contact between the cardholder and the ACS.



Checkout

Billing Information

2 Payment Information

Payment Informat

Card name: houari

Card number: 1111000022223333

Expiration Date: 10/22

Purchase Authentication

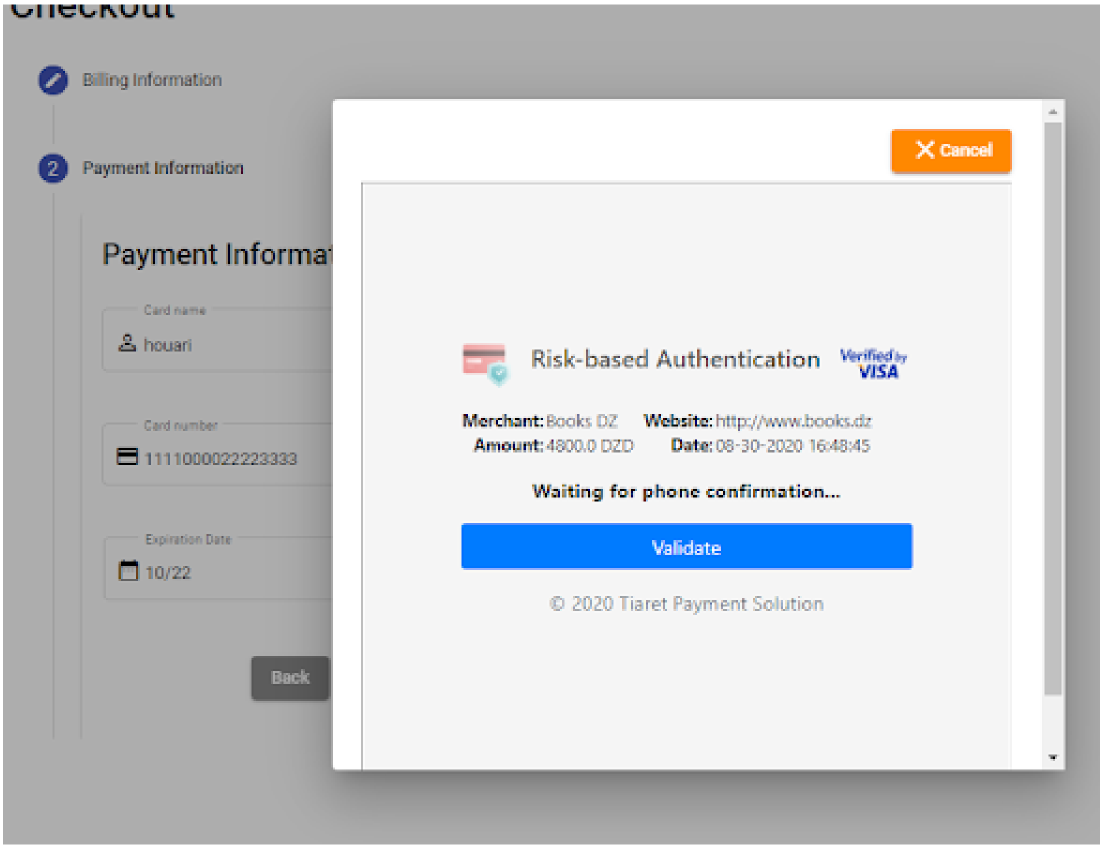
Merchant: Books DZ Website: http://www.books.dz
Amount: 4800.0 DZD Date: 08-30-2020 16:48:45

Password

© 2020 Tiaret Payment Solution

Figure 6.6: Ecommerce - ACS purshase authentication iframe

Checkout 4 - Risk-based Authentication (phone confirmation): ACS assesses risk based and if you cross the threshold, it opens the page bellow and requires a phone confirmation to complete the purchase.



Card name: houari

Card number: 1111000022223333

Expiration Date: 10/22

Risk-based Authentication

Verified by VISA

Merchant: Books DZ Website: http://www.books.dz

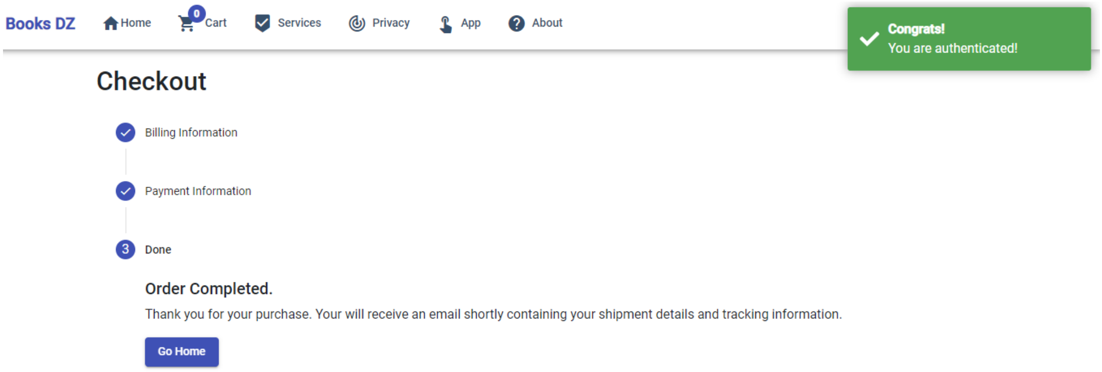
Amount: 4800.0 DZD Date: 08-30-2020 16:48:45

Waiting for phone confirmation...

© 2020 Tiaret Payment Solution

Figure 6.7: Ecommerce - Risk-based Authentication iframe

Checkout 5 - Order Completed: The order completed and payment authorized.



Congrats!
You are authenticated!

Checkout

Billing Information

Payment Information

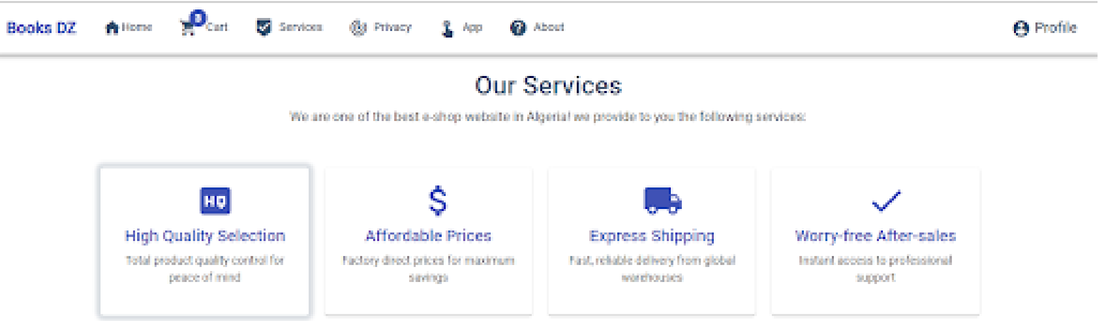
3 Done

Order Completed.

Thank you for your purchase. Your will receive an email shortly containing your shipment details and tracking information.

Figure 6.8: Ecommerce - Order completed page

Our Services page: This page shows our offered services.



Our Services

We are one of the best e-shop website in Algeria! we provide to you the following services:

High Quality Selection
Total product quality control for peace of mind

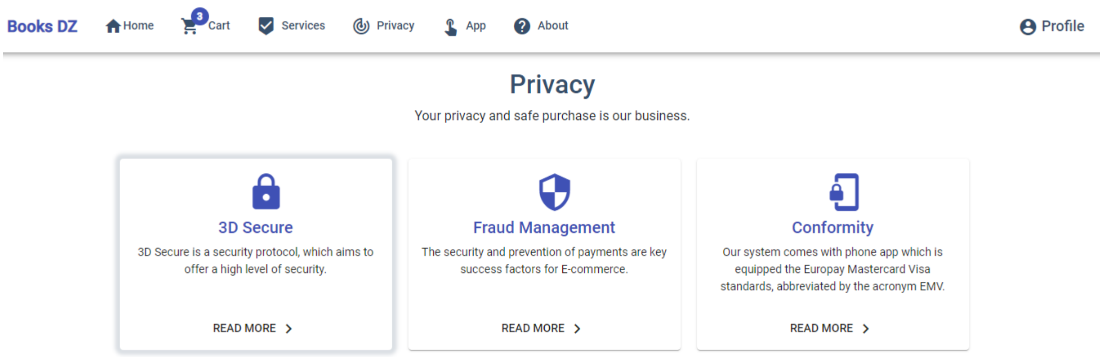
Affordable Prices
Factory direct prices for maximum savings

Express Shipping
Fast, reliable delivery from global warehouses

Worry-free After-sales
Instant access to professional support

Figure 6.9: Ecommerce - Our Services page

Our Privacy page: This page shows the privacy and security we provided to our clients.



Privacy

Your privacy and safe purchase is our business.

3D Secure

3D Secure is a security protocol, which aims to offer a high level of security.

Fraud Management

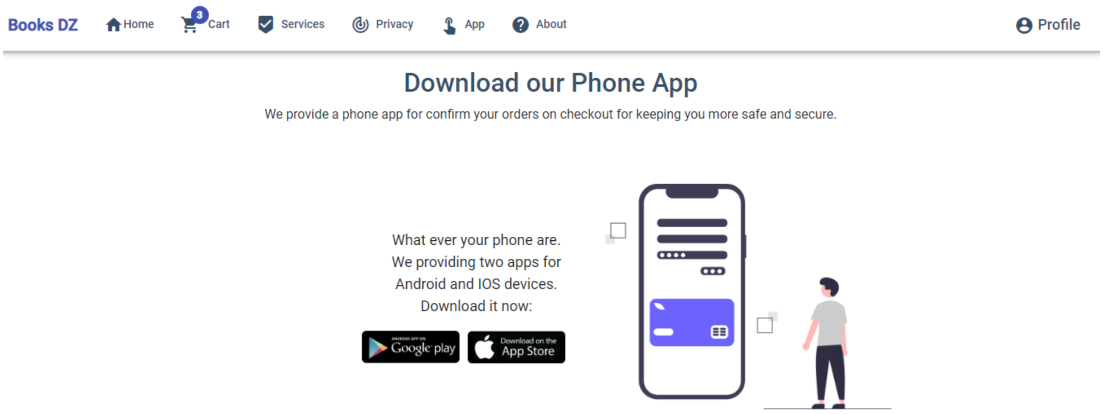
The security and prevention of payments are key success factors for E-commerce.

Conformity

Our system comes with phone app which is equipped the Europay Mastercard Visa standards, abbreviated by the acronym EMV.

Figure 6.10: Ecommerce - Our privacy page

Download our Applications page: From this page you can download our mobile applications.



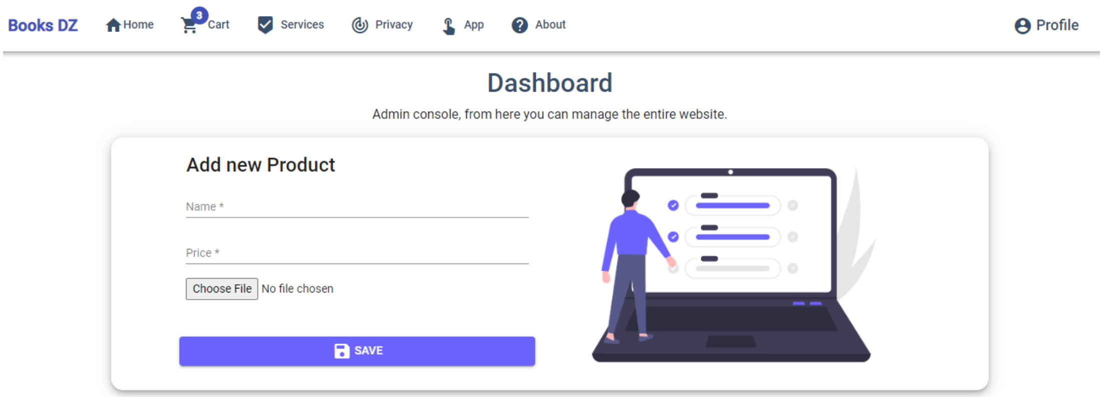
Download our Phone App

We provide a phone app for confirm your orders on checkout for keeping you more safe and secure.

What ever your phone are.
We providing two apps for
Android and IOS devices.
Download it now:

Figure 6.11: Ecommerce - Download our mobile applications page

Admin Dashboard: This page represents the admin dashboard, from here you can manage products (e.g. add new products).

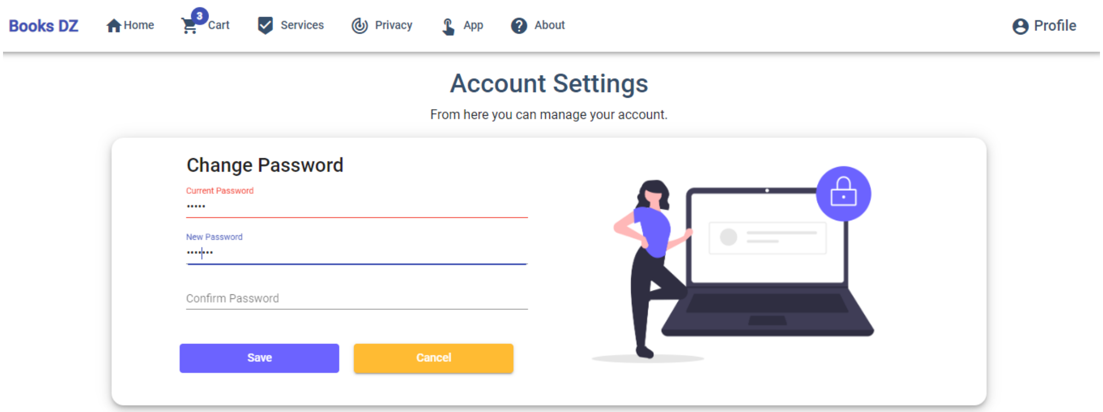


Dashboard

Admin console, from here you can manage the entire website.

Figure 6.12: Ecommerce - Admin Dashboard

Account settings - Change password: From this page you can change the password of your account.



Account Settings

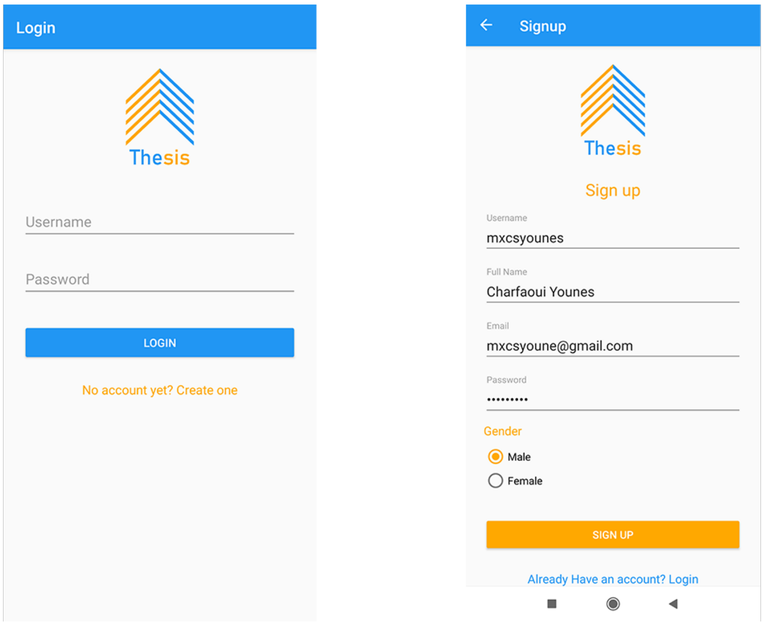
From here you can manage your account.

Figure 6.13: Ecommerce - Change password (Account Settings)

6.4.2 Android client

In the Android application, we show the main screenshots that showcase our result, even though many of the big processing is happening in the background (from generating numbers to authentication and validation of transactions):

Registration and Login: From this page you can login to your account or create a new one.



Login

Thesis

Username

Password

LOGIN

No account yet? Create one

Signup

Thesis

Sign up

Username
mxcsyounes

Full Name
Charfaoui Younes

Email
mxcsyoune@gmail.com

Password
•••••••••

Gender

● Male

○ Female

SIGN UP

Already Have an account? Login

Figure 6.14: Android Client - Login and Registration page

Entering the App: To enter the application, you need to type your PIN code or use your fingerprint for authentication.

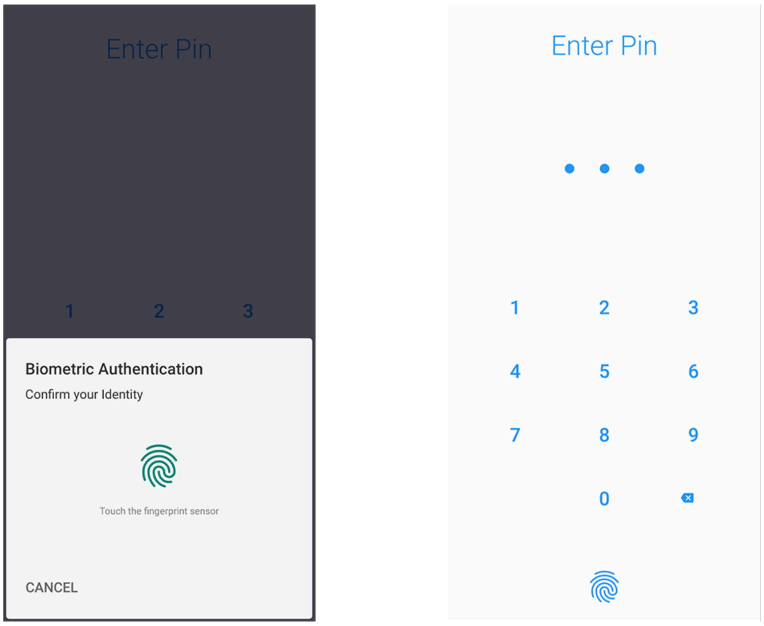
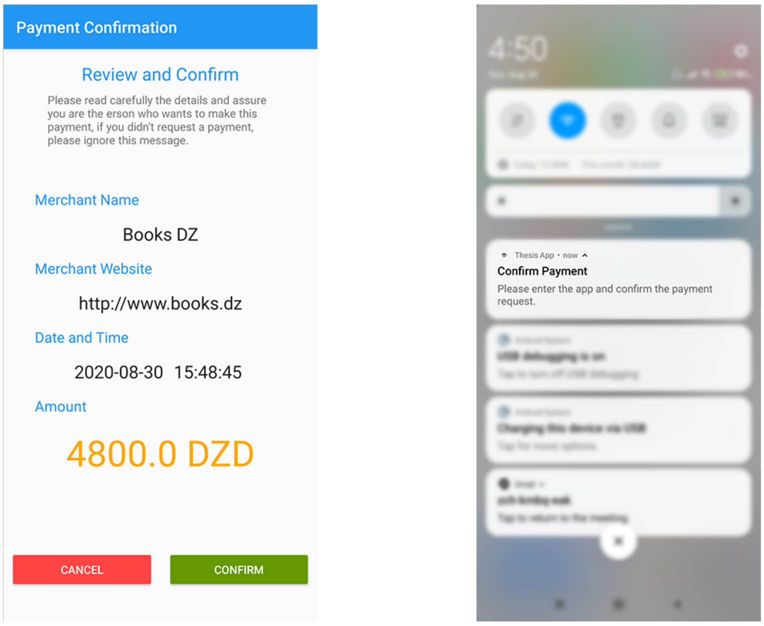


Figure 6.15: Android Client - Pin and fingerprint validation page

Payment Notification and Confirmation: When risk-based required a phone confirmation, you will receive a notification and you can confirm or decline the payment, like the following:



Payment Confirmation

Review and Confirm

Please read carefully the details and assure you are the erson who wants to make this payment, if you didn't request a payment, please ignore this message.

Merchant Name

Books DZ

Merchant Website

http://www.books.dz

Date and Time

2020-08-30 15:48:45

Amount

4800.0 DZD

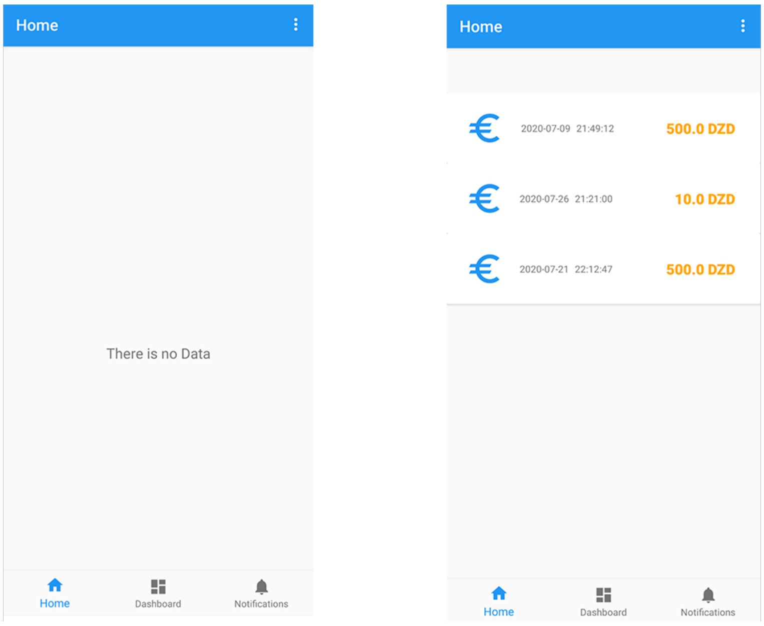
Thesis App • now

Confirm Payment

Please enter the app and confirm the payment request.

Figure 6.16: Android Client - Notification and payment confirmation page

Payment Transactions History: This page shows the transactions history you made on your account.



There is no Data

€	2020-07-09 21:49:12	500.0 DZD
€	2020-07-26 21:21:00	10.0 DZD
€	2020-07-21 22:12:47	500.0 DZD

Figure 6.17: Android Client - Payment transactions history page

6.4.3 Administration

We have also created separate software stacks for managing the system, so we can get an overview of how the system is running, manage clients, and much more, here is some screenshots to demonstrate that.

6.4.3.1 Access Control Server (ACS - Admin)

Home: From this page you can quickly access the different parts of the application,

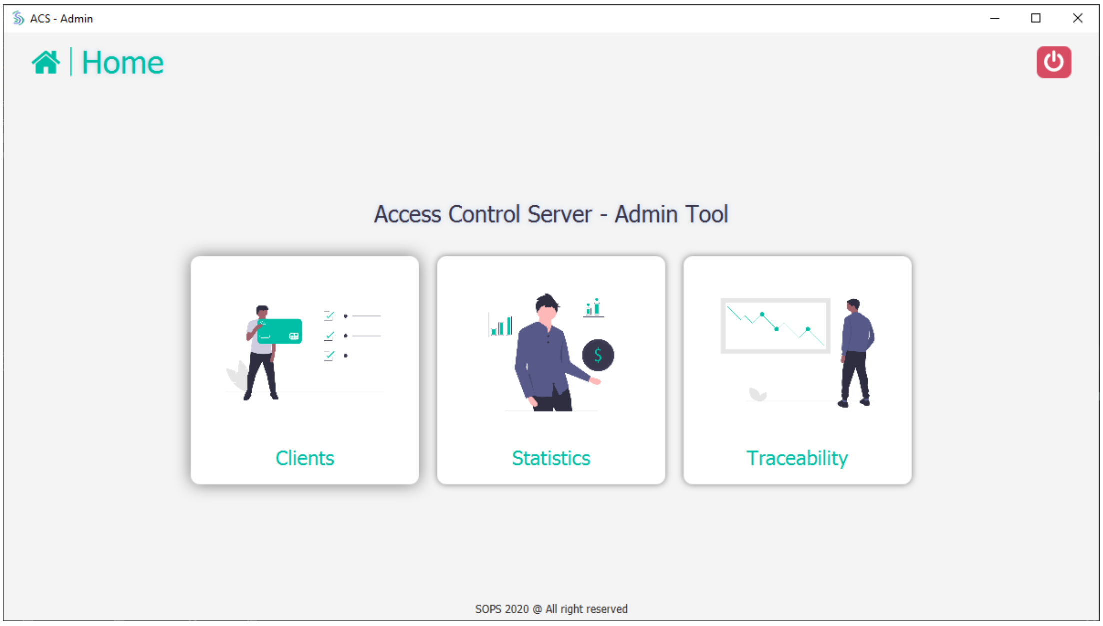
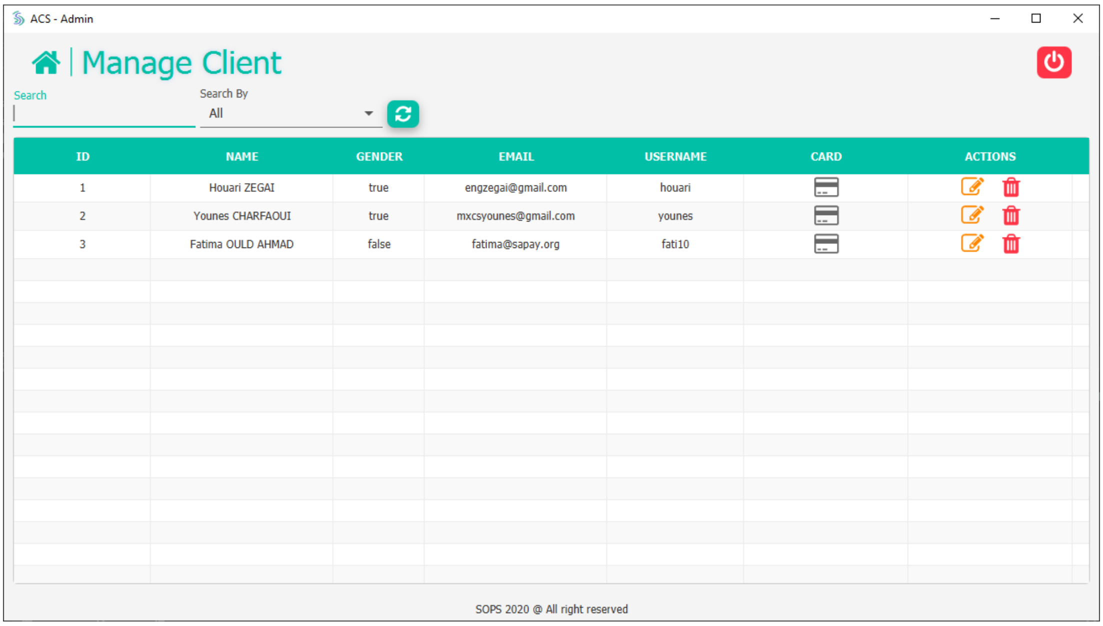

ACS - Admin
Home
Access Control Server - Admin Tool
Clients
Statistics
Traceability
SOPS 2020 @ All right reserved

Figure 6.18: ACS Admin - Home Page

Manage Clients: From this page you can see registered clients and you can create and manage their credit cards.



ACS - Admin

Manage Client

Search

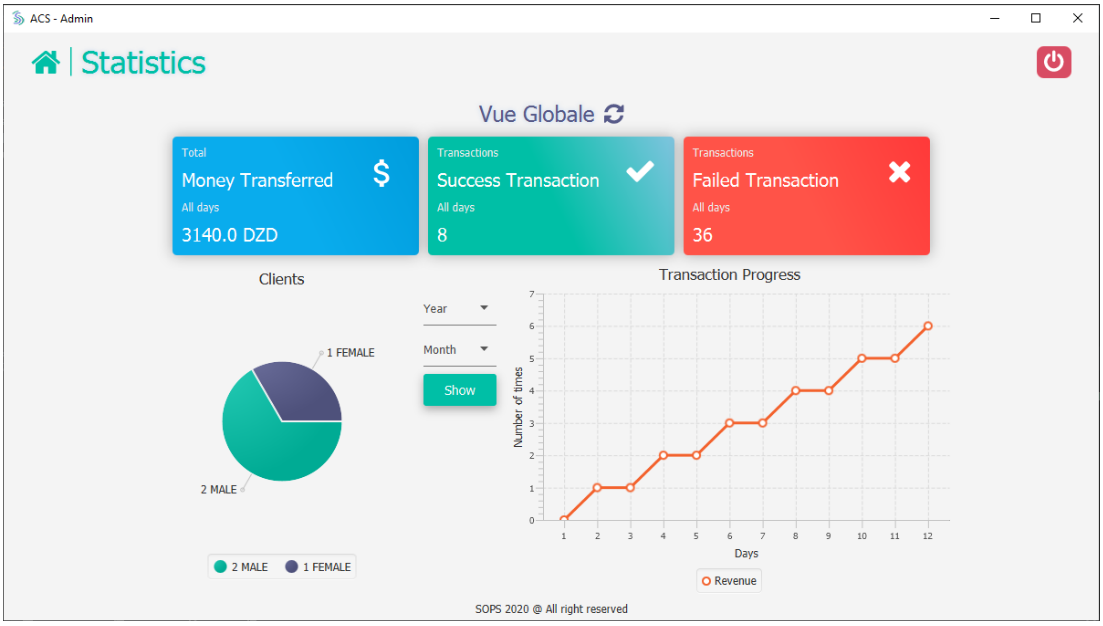
Search By: All

ID	NAME	GENDER	EMAIL	USERNAME	CARD	ACTIONS
1	Houari ZEGAI	true	engzegai@gmail.com	houari	💳	✎ 🗑
2	Younes CHARFAOUI	true	mxcsyounes@gmail.com	younes	💳	✎ 🗑
3	Fatima OULD AHMAD	false	fatima@sapay.org	fati10	💳	✎ 🗑

SOPS 2020 @ All right reserved

Figure 6.19: ACS Admin - Manage Clients page

Statistics: From this page you can see some of global statistics of the access control server.



ACS - Admin

Statistics

Vue Globale

Total
Money Transferred
All days
3140.0 DZD

Transactions
Success Transaction
All days
8

Transactions
Failed Transaction
All days
36

Year

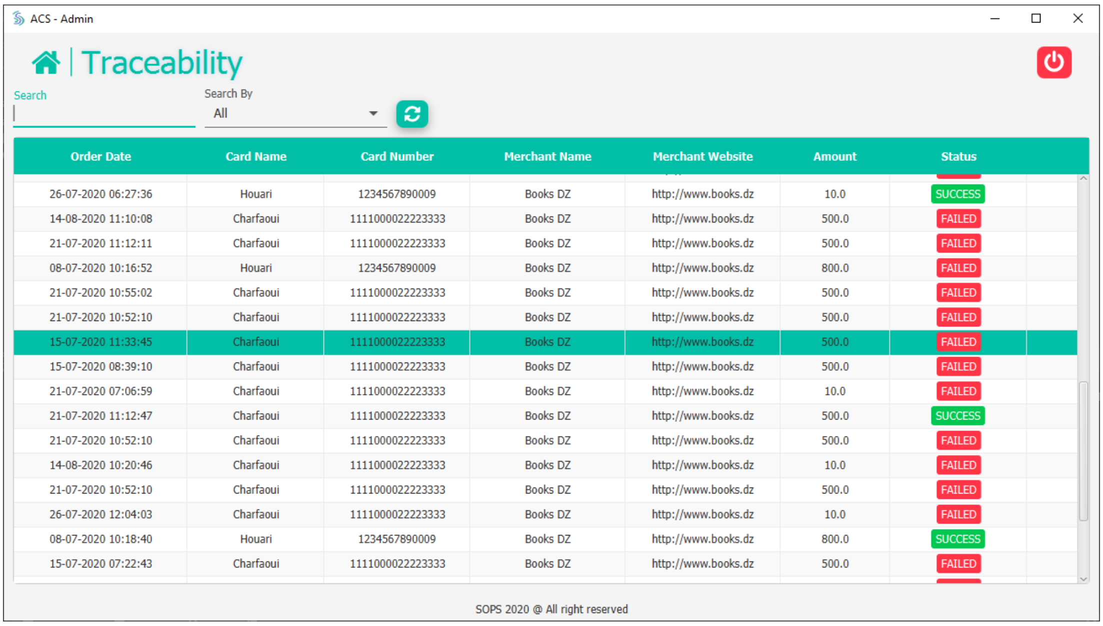
Month

Show

SOPS 2020 @ All right reserved

Figure 6.20: ACS Admin - Statistics page

Traceability (Transactions history): The page below shows the traceability of all past transactions made on our system.



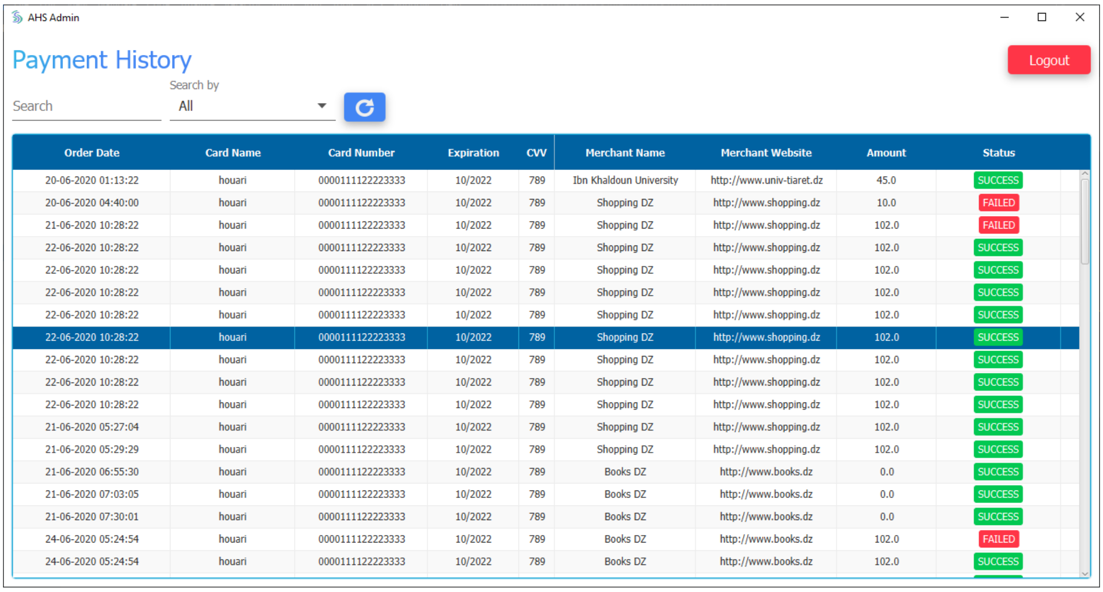
Order Date	Card Name	Card Number	Merchant Name	Merchant Website	Amount	Status
26-07-2020 06:27:36	Houari	1234567890009	Books DZ	http://www.books.dz	10.0	SUCCESS
14-08-2020 11:10:08	Charfaoui	1111000022223333	Books DZ	http://www.books.dz	500.0	FAILED
21-07-2020 11:12:11	Charfaoui	1111000022223333	Books DZ	http://www.books.dz	500.0	FAILED
08-07-2020 10:16:52	Houari	1234567890009	Books DZ	http://www.books.dz	800.0	FAILED
21-07-2020 10:55:02	Charfaoui	1111000022223333	Books DZ	http://www.books.dz	500.0	FAILED
21-07-2020 10:52:10	Charfaoui	1111000022223333	Books DZ	http://www.books.dz	500.0	FAILED
15-07-2020 11:33:45	Charfaoui	1111000022223333	Books DZ	http://www.books.dz	500.0	FAILED
15-07-2020 08:39:10	Charfaoui	1111000022223333	Books DZ	http://www.books.dz	500.0	FAILED
21-07-2020 07:06:59	Charfaoui	1111000022223333	Books DZ	http://www.books.dz	10.0	FAILED
21-07-2020 11:12:47	Charfaoui	1111000022223333	Books DZ	http://www.books.dz	500.0	SUCCESS
21-07-2020 10:52:10	Charfaoui	1111000022223333	Books DZ	http://www.books.dz	500.0	FAILED
14-08-2020 10:20:46	Charfaoui	1111000022223333	Books DZ	http://www.books.dz	10.0	FAILED
21-07-2020 10:52:10	Charfaoui	1111000022223333	Books DZ	http://www.books.dz	500.0	FAILED
26-07-2020 12:04:03	Charfaoui	1111000022223333	Books DZ	http://www.books.dz	10.0	FAILED
08-07-2020 10:18:40	Houari	1234567890009	Books DZ	http://www.books.dz	800.0	SUCCESS
15-07-2020 07:22:43	Charfaoui	1111000022223333	Books DZ	http://www.books.dz	500.0	FAILED

SOPS 2020 @ All right reserved

Figure 6.21: ACS Admin - Traceability (Payment History)

6.4.3.2 Authentication History Server (AHS - Admin)

Transactions History: The page below shows the traceability of all past transactions made on our system.



Payment History

Order Date	Card Name	Card Number	Expiration	CVV	Merchant Name	Merchant Website	Amount	Status
20-06-2020 01:13:22	houari	0000111122223333	10/2022	789	Ibn Khaldoun University	http://www.univ-tiaret.dz	45.0	SUCCESS
20-06-2020 04:40:00	houari	0000111122223333	10/2022	789	Shopping DZ	http://www.shopping.dz	10.0	FAILED
21-06-2020 10:28:22	houari	0000111122223333	10/2022	789	Shopping DZ	http://www.shopping.dz	102.0	FAILED
22-06-2020 10:28:22	houari	0000111122223333	10/2022	789	Shopping DZ	http://www.shopping.dz	102.0	SUCCESS
22-06-2020 10:28:22	houari	0000111122223333	10/2022	789	Shopping DZ	http://www.shopping.dz	102.0	SUCCESS
22-06-2020 10:28:22	houari	0000111122223333	10/2022	789	Shopping DZ	http://www.shopping.dz	102.0	SUCCESS
22-06-2020 10:28:22	houari	0000111122223333	10/2022	789	Shopping DZ	http://www.shopping.dz	102.0	SUCCESS
22-06-2020 10:28:22	houari	0000111122223333	10/2022	789	Shopping DZ	http://www.shopping.dz	102.0	SUCCESS
22-06-2020 10:28:22	houari	0000111122223333	10/2022	789	Shopping DZ	http://www.shopping.dz	102.0	SUCCESS
22-06-2020 10:28:22	houari	0000111122223333	10/2022	789	Shopping DZ	http://www.shopping.dz	102.0	SUCCESS
22-06-2020 10:28:22	houari	0000111122223333	10/2022	789	Shopping DZ	http://www.shopping.dz	102.0	SUCCESS
21-06-2020 05:27:04	houari	0000111122223333	10/2022	789	Shopping DZ	http://www.shopping.dz	102.0	SUCCESS
21-06-2020 05:29:29	houari	0000111122223333	10/2022	789	Shopping DZ	http://www.shopping.dz	102.0	SUCCESS
21-06-2020 06:55:30	houari	0000111122223333	10/2022	789	Books DZ	http://www.books.dz	0.0	SUCCESS
21-06-2020 07:03:05	houari	0000111122223333	10/2022	789	Books DZ	http://www.books.dz	0.0	SUCCESS
21-06-2020 07:30:01	houari	0000111122223333	10/2022	789	Books DZ	http://www.books.dz	0.0	SUCCESS
24-06-2020 05:24:54	houari	0000111122223333	10/2022	789	Books DZ	http://www.books.dz	102.0	FAILED
24-06-2020 05:24:54	houari	0000111122223333	10/2022	789	Books DZ	http://www.books.dz	102.0	SUCCESS

Figure 6.22: AHS Admin - Transactions History

6.5 Conclusion

In this chapter, we have presented the implementation of a prototype of our Payment System platform

We first outlined the different technologies used for the implementation as well as the development environment. We have opted for choices of tools and techniques that are widely used today in the development of web and mobile applications, and finally, we have exposed the basic functionalities offered in our platform through examples.

Conclusion and Perspectives

The present work aims to respond to a need for a method of authenticating equipment and clients by defining a method of generating random numbers, which is the core idea behind the identification and authentication mechanism. As a use case, electronic commerce seems to be a well suited one. To be at the same level as what is done, we have faithfully implemented the 3-D Secure protocol version 2.0 of EMVCo.

We have presented and solved several security points using state of the art techniques, standards, and protocols that exist to achieve a certain level of security of services and servers.

Our work still needs a lot of polishing and maintenance to reach the best security and availability level. Since we were limited by the time and the available resources, we would like to add several other concepts like improving the Random number generation so authentication can happen more rapidly and seamlessly.

Apart from the RNG point, we want to complete the ACS administration system to fully control the system and complete the payment process (interbank communication).

Besides the current Risk-Based assessment that we have implemented, we wanted to create an expert system or train machine learning or deep learning models to detect fraud. This way, we do not rely only on simple data like amounts and regions, and this allows the system to identify in real-time if the customer is makeing the purchase himself or someone is using his information. And at the same point, we also want to validate the merchant's identity to ensure that the merchant is not fraudulent.

Finally, to improve our transactions system's credibility and transparency, we want to add the Blockchain technology in history payments or transactions.

References

[1] C. Wheeler, "Multi-factor authentication."

[2] Mastercard, "Emv 3-d secure - high level overview." `http://www.w3.org/2017/Talks/3ds-overview-tpac2017.pptx`, 2017. (Accessed on 08/12/2020).

[3] P. Elena, "Authentication under constraints," *Department of Computer Science and Engineering, Chalmers University of Technology, Goteborg, Sweden*, p. 3, 2016.

[4] W. Stallings and L. Brown, *Computer security: principles and practice.* Pearson Education, Inc., 2018.

[5] F. Alaca, *Strengthening Password-Based Web Authentication Through Multiple Supplementary Mechanisms.* PhD thesis, Carleton University, 2018.

[6] P. Cervicek, *Authentication in Web Applications.* PhD thesis, MS thesis, Distributed Computing Systems Engineering, 2009.

[7] R. E. Smith, *Authentication: from passwords to public keys.* Addison-Wesley, 2002.

[8] C. Soutar, D. Roberge, A. Stoianov, R. Gilroy, and B. V. Kumar, "Biometric encryption," in *ICSA guide to Cryptography*, vol. 22, McGraw-Hill, 1999.

[9] S. Huopio, "Biometric identification, authorization and access control in open network environment," 1998.

[10] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. T. Polk, *et al.*, "Internet x. 509 public key infrastructure certificate and certificate revocation list (crl) profile.," *RFC*, vol. 5280, 2008.

[11] W. Stallings, *Network security essentials: applications and standards.* Pearson Education India, 2007.

[12] J. Garman, *Kerberos: The Definitive Guide: The Definitive Guide.* " O'Reilly Media, Inc.", 2003.

[13] D. Dasgupta, A. Roy, and A. Nag, *Advances in User Authentication.* Springer, 2017.

[14] "Randomness: Definition of randomness by oxford dictionary on lexico.com also meaning of randomness."

[15] D. E. Knuth, *Art of computer programming, volume 2: Seminumerical algorithms.* Addison-Wesley Professional, 2014.

[16] G. Marsaglia, "Random number generation," in *Encyclopedia of computer science*, 2003.

[17] B. F. Vajargah and R. Asghari, "A novel pseudo-random number generator for cryptographic applications," *Indian Journal of Science and Technology*, 2016.

[18] C. Paar and J. Pelzl, *Understanding cryptography: a textbook for students and practitioners.* Springer Science & Business Media, 2009.

[19] B. Schneier, T. Kohno, and N. Ferguson, *Cryptography engineering: design principles and practical applications.* Wiley, 2013.

[20] M. Stipčević, “Quantum random number generators and their use in cryptography,” in *2011 Proceedings of the 34th International Convention MIPRO*, pp. 1474–1479, IEEE, 2011.

[21] B. F. Vajargah and R. Asghari, “A novel pseudo-random number generator for cryptographic applications,” *Indian Journal of Science and Technology*, 2016.

[22] H. G. Katzgraber, “Random numbers in scientific computing: An introduction,” *arXiv preprint arXiv:1005.4117*, 2010.

[23] M. Matsumoto and T. Nishimura, “Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator,” *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 1998.

[24] G. Marsaglia *et al.*, “Xorshift rngs,” *Journal of Statistical Software*, 2003.

[25] B. Widynski, “Squares: A fast counter-based rng,” *arXiv preprint arXiv:2004.06278*, 2020.

[26] M. E. O’Neill, “Pcg: A family of simple fast space-efficient statistically good algorithms for random number generation,” *ACM Transactions on Mathematical Software*, 2014.

[27] J. E. Gentle, *Random number generation and Monte Carlo methods.* Springer Science & Business Media, 2006.

[28] D. Eastlake, J. Schiller, and S. Crocker, “Randomness requirements for security,” *RFC4086*, 2005.

[29] J. E. Gentle, W. K. Härdle, and Y. Mori, *Handbook of computational statistics: concepts and methods.* Springer Science & Business Media, 2012.

[30] L. E. Bassham III, A. L. Rukhin, J. Soto, J. R. Nechvatal, M. E. Smid, E. B. Barker, S. D. Leigh, M. Levenson, M. Vangel, D. L. Banks, *et al.*, *Sp 800-22 rev. 1a. a statistical test suite for random and pseudorandom number generators for cryptographic applications.* National Institute of Standards & Technology, 2010.

[31] P. L'Ecuyer and R. Simard, "Testu01: Ac library for empirical testing of random number generators," *ACM Transactions on Mathematical Software (TOMS)*, 2007.

[32] G. Marsaglia, "The marsaglia random number cdrom including the diehard battery of tests of randomness, 1995," *URL http://www. stat. fsu. edu/pub-/diehard*, 2008.

[33] T. Alam, "A reliable communication framework and its use in internet of things (iot)," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT)*, vol. 3, no. 5, pp. 450–456, 2018.

[34] "Solving the cnp false decline puzzle: Collaboration is key." `https://merchantriskcouncil.org/resource-center/whitepapers/2016/solving-the-cnp-false-decline-puzzle-collaboration-is-key`, 2016. (Accessed on 08/15/2020).

[35] "3-d secure by visa." `https://usa.visa.com/run-your-business/small-business-tools/payment-technology/visa-secure.html`. (Accessed on 08/20/2020).

[36] "3d secure 2.0 and card schemes." https://3dsecure2.com/card-schemes/. (Accessed on 08/08/2020).

[37] "Modirum - 3-d secure." https://www.modirum.com/3dsecure/. (Accessed on 08/08/2020).

[38] "Emvco to manage next generation of 3d-secure specification." https://www.emvco.com/wp-content/uploads/documents/EMVCo_To_Support_3DS_January-2015.pdf, 2015. (Accessed on 08/13/2020).

[39] "2016 annual emv user meeting to be held in copenhagen." https://www.emvco.com/wp-content/uploads/documents/EMVCo_announces_2016_user_meeting_April-2016.pdf, 2016. (Accessed on 08/13/2020).

[40] "Emvco launches emv 3-d secure 2.0 specification." https://www.emvco.com/wp-content/uploads/2017/05/EMV-3DS-2-Spec-Launch-Final-October-2016.pdf, 2017. (Accessed on 08/06/2020).

[41] EMVCo, "Emv - 3-d secure: Protocol and core functions specification." https://docs.3dsecure.io/3dsv2/_downloads/b412903d6e2c99b7828246fa10db5b3e/EMVCo_3DS_Spec_v220.pdf. (Accessed on 08/19/2020).

[42] "Frictionless flow for charges created using 3d secure 2 (3ds2) : Stripe: Help & support." https://support.stripe.com/questions/frictionless-flow-for-charges-created-using-3d-secure-2-3ds2. (Accessed on 08/13/2020).

[43] "Non-payment authentication with 3d secure 2." https://3dsecure2.com/non-payment-authentication/. (Accessed on 08/07/2020).

[44] GPayments, “Risk-based authentication and 3d secure 2.” https://www.gpayments.com/Portals/0/pdfs/RBA_and_3D_Secure_2.pdf. (Accessed on 08/17/2020).

[45] VISA, “Visa 3-d secure 2.0.” https://technologypartner.visa.com/Download.aspx?id=681. (Accessed on 08/18/2020).

[46] R. C. Martin, “Uml for java (tm) programmers,” 2003.

[47] M. E. O’Neill, “Pcg: A family of simple fast space-efficient statistically good algorithms for random number generation,” *ACM Transactions on Mathematical Software*, 2014.

[48] M. O’Neill, “Pcg, a family of better random number generators.” https://www.pcg-random.org/, Aug 2014. (Accessed on 08/17/2020).

[49] C. Paar and J. Pelzl, *Understanding cryptography: a textbook for students and practitioners*. Springer Science & Business Media, 2009.

[50] B. Schneier, T. Kohno, and N. Ferguson, *Cryptography engineering: design principles and practical applications*. Wiley, 2013.

[51] L. Bottou, “Stochastic gradient descent tricks,” in *Neural networks: Tricks of the trade*, Springer, 2012.

[52] Y. Qi, “Random forest for bioinformatics,” in *Ensemble machine learning*, Springer, 2012.

[53] D. Blackman and S. Vigna, “Scrambled linear pseudorandom number generators,” *arXiv preprint arXiv:1805.01407*, 2018.