Final Dissertation
To obtain the title of:

Master Degree in
Cryptography and Information Security

Submitted by:

Yassine MEKDAD

Entitled:

3-D Secure Enrollment Process Implementation: Activation During Shopping & Registration Using ACS Back Office

Defended on October 21st 2016, in front of the jury composed by:

Mr. El Mamoun SOUIDI	**Chairman, Professor at Faculty of Sciences, Rabat (Dissertation adviser).**
Mrs. Ghizlane ORHANOU	**Professor at Faculty of Sciences, Rabat, (Examiner).**
Mr. Sidi Mohamed DOUIRI	**Professor at Faculty of Sciences, Rabat, (Examiner).**
Mr. Abdelghani ROUSSI	**Project Manager at S2M (Internship adviser).**

Academic year 2015/2016

Faculté des Sciences, 4 Avenue Ibn Battouta B.P. 1014 RP, Rabat – Maroc
Tel +212 (0) 37 77 18 34/35/38, Fax : +212 (0) 37 77 42 61, http://www.fsr.ac.ma
LabMIA: http://www.fsr.ac.ma/mia, Master C2SI : http://masterc2si.com

Dedication

In memory of my defunct grandfather **HAJ AHMED LAGHRISSI**. You left fingerprints of grace on my life. You shall not be forgotten.

This dissertation is dedicated to my mother and my father who encouraged me in this study over a number of years, for their continuous effort, giving and permanent support to reach achievement.This would not have been possible without their conforting words and motivation.

My thanks to my brother and all my family who give me support during my college years,

I would like to thanks also my friends for giving me their best moments marked by happiness and gladness.

I dedicate also this dissertation to **Pr NADIA MOATADID** the most helpful and understandable professor that I have had during my freshman year of college.

MEKDAD YASSINE

Acknowledgment

In term of this work, we would like to thank everyone who has given help to achieve this project

In the beginning of this dissertation, we would like to sincerely thank our advisor, ***Professor EL MAMOUN SOUIDI,*** *our professor for his guidance, support and his confidence on us, as well as his generosity and quality of teaching without forgetting his precious instructions and advices.*

We are grateful toward our external advisor ***PM Abdelghani ROUSSI*** *within S2M Company by whom we learned more professional qualities and strategic plan to lead this project in order to reach its objectives*

We express as well our gratitude to jury members' ***Ghizlane ORHANOU, Sidi Mohamed DOUIRI & EL MAMOUN SOUIDI*** *for accepting to judge this work.*

We would like to thank our professors of **Master Cryptography & Information Security**, *who were always encouraging us to work by providing us what we need from their experience and* skills.

Abstract

This document is a summary of our graduation project conducted at the edition department of S2M, to obtain the Master diploma of Cryptography and Information Security. Online shopping is becoming more and more interesting for clients because of the ease of use and the large choice of products and facilities. As a consequence, 2.3 billion clients have been identified in 2011. This fast increase has been accompanied by various frauds. Several electronic payment have been proposed to fight this issues and reduce the huge number of threats and the 3D-Secure protocol is becoming a standard for payment on the internet. For this reason the continuous improvement of the 3D-Secure project in S2M Company is imperative and necessary beside the global evolution and extension of technologies and users of Internet. This paper proposes the various elements studied on our internship which is about 3D-Secure passing by the steps and respected standards as well as the results of implementing a paramount module in this massive project which is the enrollment process.

Key words Cryptography, Security, 3-D Secure, S2M, Enrollment, Payment.

Contents

List of Tables

List of abreviation

Abreviation	**Meaning**
S2M	Société Maghrébine de Monétique
3-D Secure	Three Domain Secure
ACS	Access Control Server
ADS	Activation During Shopping
MPI	Merchant Plug-in
AHS	Authentication History Server
MVC	Model View Controller
DAO	Data Access Object
DBMS	DataBase Management System
JSP	Java Server Pages
Mkey	Master Key
BIN	Bank Identifier Number
PAN	Personal Account Number
RS	Registration Server
CVV	Card Verification Value
ECI	Electronic Commerce Indicator
CA	Certificat Authority
SSL	Secure Socket Layer
OTP	One Time Password

Table 1 – List of abreviation

List of Figures

Introduction

Internet computer network was constructed by a set of cooperative organizations to link billions of devices worldwide. It's the most exciting and interesting technological fields of our time and it takes a lot of complex software and hardware to make this global system seem so simple. Nevertheless the unlimited expansion of the Internet and the current Internet security issues were not anticipated. For this reasons the Internet threats became prolific especially when the matter is an online payment process.

The e-commerce related issues is important because it includes important issues; the merchant must be paid, the customer wants to ensure the service will be provided by the merchant and the bank must ensure there no identity theft to protect his client. Data should therefore remain confidential.Such an exchange is complex because it involves multiple stakeholders (customer, merchant,Banks) and numerous technical and scientific aspects (computer security,database, smart cards, ...).
It is in this context that our final internship studies conducted at (S2M) for a period of 6 months.During this internship, we were called upon to perform a secure enrollment process for 3-D Secure which ensures the identity of the cardholder credit card for every transaction.

The current document will explore the different phases of our final project graduation which consist of studying the specifications of the Enrollment process in the 3-D Secure payment protocols according to Visa, MasterCard specifications, designing and developing all the features of this process fulfilling all the necessary requirements related to the existent project in S2M Solutions with full respect to the Visa and MasterCard standards. The current report is structured around four chapters starting by introducing the **host organization**, catching a glimpse of its history, internal structure and its achievements.

In the second chapter we will talk generally about the **3-D Secure protocol** and the different entities that take part of an online purchase as well as getting into the section related to the enrollment process which is the module that will be taken place with implementing its features specified as two phases called **Activation During Shopping and Registration Using ACS Back Office** which are respectively the titles of the third and fourth chapter depicting the different steps to build this module including studies of the specifications, requirements, security measures, tools and technologies used for the development without forgetting to mention the main frameworks in addition to the methods we used to make this project useful, efficient and secure.

Chapter 1

Host organization

1.1 General presentation

Société Maghrébine de Monétique (S2M)is an electronic banking company created in Morocco since 1983. Nowadays, S2M performs its operations in 35 countries in Africa, Asia, Europe and Oceania. The company is specializing in edition and integration of payment solutions, customizing payment instruments (cards,vouchers etc ..), desktop publishing (Enveloping), as well as marketing and maintenance of electronic payment terminals.

Figure 1.1 – S2M Logo

Pioneer of the very first credit card applications in North Africa, S2M has profiled itself, for over thirty years, as one of the leading players in developing and implementing payment solutions, personalising means of payment, publishing and mailing.

S2M proposes the most complete and reliable offer on the market:

- Payment solutions: SELECT System®
- Magnetic stripe and EMV card applications
- E-commerce solutions
- Branch specific solutions: oil companies, telco, distribution, hotels. . .
- POS solutions (payment, loyalty . . .)
- Outsourcing: Payment, credit and loyalty solutions (ASP mode)
- Personalization of payment means: bank cards, private label cards (payment and loyalty)
- Publishing and mailing: document printing and personalization, folding and mass mailing
- POS supply and maintenance

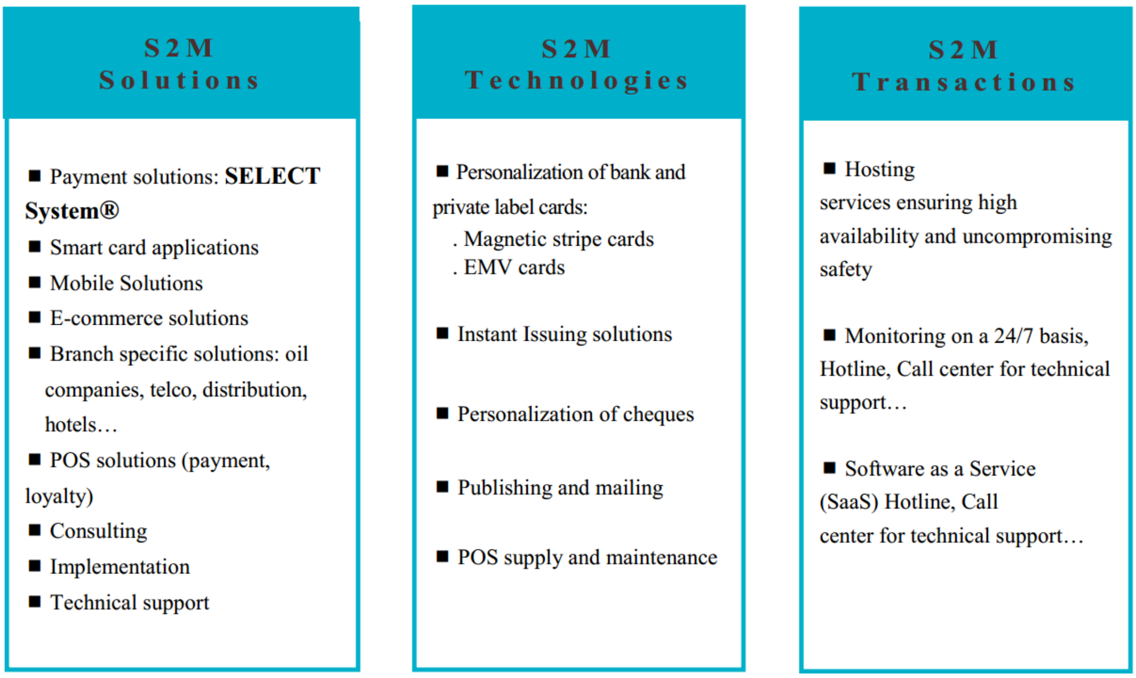
S2M also stands for a broad range of services:

- Implementation
- Consulting
- Training
- Assistance
- Support and maintenance

S2M has established an impressive international reputation: over a hundred private and public institutions, in more than 30 countries in Africa, Asia, Europe and Oceania rely on its solutions and services.

S2M is organized into three entities:

- **S2M Solutions**: specializing in the development, integration and maintenance of electronic payment solutions.
- **S2M Technologies**: offers payment means of personalization services (card check) as well as desktop publishing and maintenance services of electronic payment terminals.
- **S2M Transactions**: specializing in outsourcing of electronic payment solutions (including hosting services, operations and back office processing)



S2M Solutions

- Payment solutions: **SELECT System®**
- Smart card applications
- Mobile Solutions
- E-commerce solutions
- Branch specific solutions: oil companies, telco, distribution, hotels…
- POS solutions (payment, loyalty)
- Consulting
- Implementation
- Technical support

S2M Technologies

- Personalization of bank and private label cards:
  . Magnetic stripe cards
  . EMV cards
- Instant Issuing solutions
- Personalization of cheques
- Publishing and mailing
- POS supply and maintenance

S2M Transactions

- Hosting services ensuring high availability and uncompromising safety
- Monitoring on a 24/7 basis, Hotline, Call center for technical support…
- Software as a Service (SaaS) Hotline, Call center for technical support…

Figure 1.2 – Entities of S2M

1.2 History of S2M

For more than three decades, S2M has anticipated the market needs trends and requirements by constantly innovating and upgrading its solutions and services.

In **1983**, the very first MultiPac version, developed for banks and for credit card processors, makes its reputation as one of the most function-rich system.

1985 is the start of the “personalized bank cards” activity.

In **1997**, S2M is given the 1st Moroccan Export Award, first symbol of international recognition

1999 is the year of S2M Technologies Centre is certified ISO 9002.

In **2000**, S2M introduces its global secure payment solution for e-Commerce on the market.

In **2002**, S2M launches its new, native EMV version 6, which allows S2M to assist banks in their migration towards the EMV standards and provides more interoperability. This latest version takes the best from technological and functional: multithreading, Work-Flow processing, user-friendliness and increased security and availability.

In **2005**, and in order to emphasize the functional and technological break, S2M renames its payment system solution to become **SELECT System®** (Secure ELEctronic Card Transactions System). **SELECT System®** has been designed and developed to comply with the EMV standards and to bring new functionalities and modules answering the customer needs. **SELECT System®** is in fact a new product that brings an unequalled functional wealth.

In **2006**, the bank cards and private label cards personalization unit of S2M Technologies obtains the **VISA VSDC and MasterCard M-Chip** certifications which authorize it to personalize EMV chip cards from then.

2007 is marked by the start up of S2M new Production Centre, certified **ISO 9001 Version 2000**, VISA and MasterCard. This 2000 sqm centre required an investment of 20 million Moroccan Dirhams (nearly 2, 5 million USD) excluding all production equipments.

In **January 2009**, S2M Technologies gets the **ISO 9001 Version 2008** certification.

In **June 2010**, S2M achieves the PA-DSS 1.2 certification for its payment system solution: **SELECT System®**.

In **December 2011**, S2M became the first Moroccan software editor, to deploy and certify its own procedures **CMMI ® Level 4**. This certification has enabled S2M consolidate the best practices recognized worldwide.

In **2014**, S2M Transactions was certified **PCI-DSS**(Payment Card Industry Data Standard Security).

2015: S2M opened the Oceania market by helping a major customer in New Zealand implement its new payment system solution. Moreover, S2M created S2M Transactions GCC (a Joint Venture with Alruaya Holding in the Gulf Cooperation Council) in Bahrain for an increased proximity and service in the region.

1.3 Assets of S2M

S2M has an undeniable know-how in the 4 areas of credit card applications:
— Software development and engineering.
— Implementation, support and maintenance of software package.
— Bank cards and private label cards personalization.
— Personalization and processing of chequebooks.

Its teams are involved in all stages of a credit card applications project:

— Consulting (audits, gap analysis, detailed studies),
— Software development and implementation,
— Payment, credit and loyalty solutions outsourcing,
— Testing Certification with the associations (Europay, VISA, MasterCard) support and maintenance.

Certified by VISA and MasterCard, S2M covers through its offer all the management functions related to bank cards: from cardholder to merchant until the processing of transactions between banks and international transactions.

At the leading edge of technology
The production centre is equipped with modern and powerful equipment. Furthermore, a continuous investment plan allows to reinforce the centres outputs from year to year and to equip them with high-tech hardware and software means.

Physical security

- Access control to the production areas; video monitoring and recording 24/7; intrusion detectors and infra-red barriers; electric group
- Safe deposit
- 3 distinct units, dedicated by activity: cards, cheques, publishing and mailing
- Back-up site completely operational, as protected as the main production centre

Logical security

- Firewall and information processing systems for controlled access in order to protect the network, electronic documents...
- Secure transfer of the files to be processed
- Security and traceability of access and processings
- Back-Up of the production servers

1.4 organizational chart

This figure represents the structure of S2M, the relationships and relative ranks of its parts and positions/jobs.

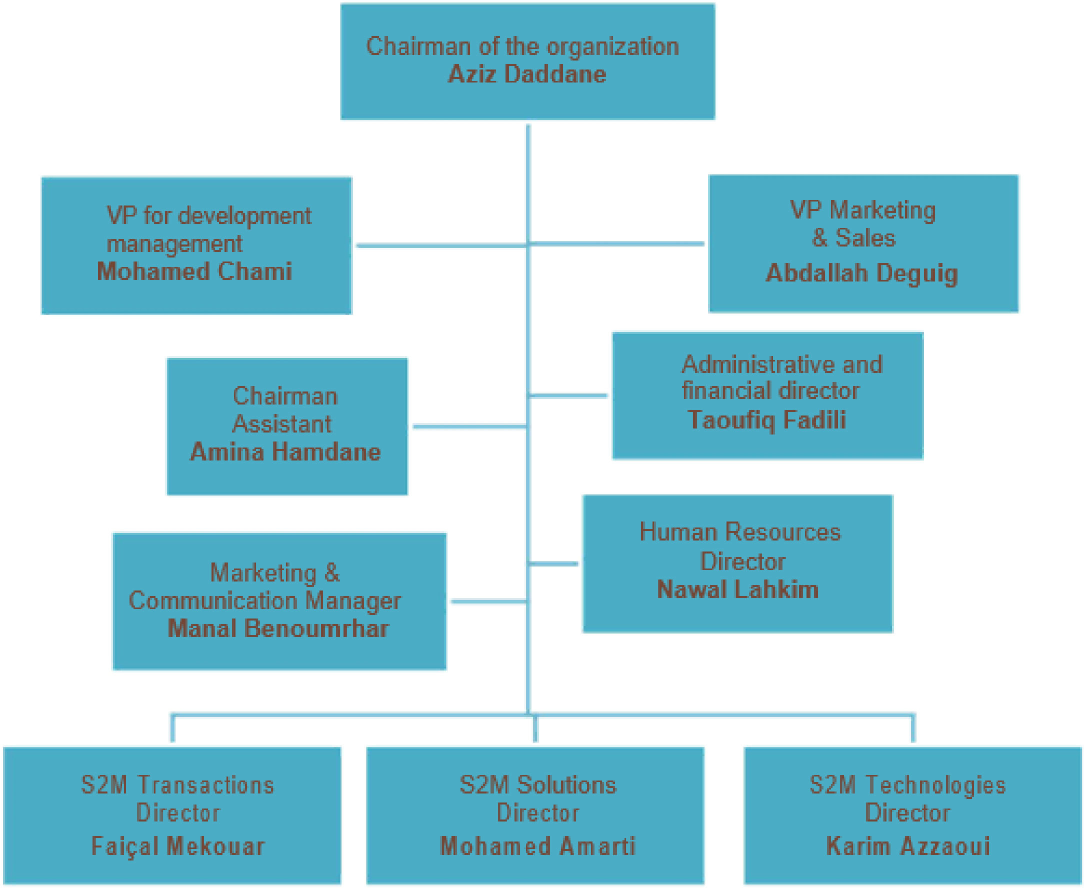
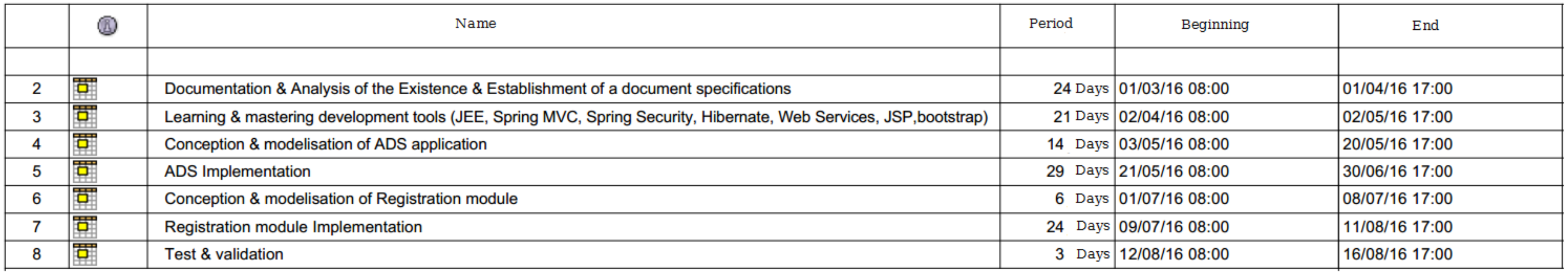


Figure 1.3 – Organizational chart of S2M

1.5 Project schedule planning

To carry out the realization of our project, we have adopted an approach that ensures control of project development. Which allow us not to make mistake of groping several paths. Something that leads to a result certain and controllable. This figure below shows the diagram of the different steps of the project.



	ⓘ	Name	Period	Beginning	End
2	▦	Documentation & Analysis of the Existence & Establishment of a document specifications	24 Days	01/03/16 08:00	01/04/16 17:00
3	▦	Learning & mastering development tools (JEE, Spring MVC, Spring Security, Hibernate, Web Services, JSP,bootstrap)	21 Days	02/04/16 08:00	02/05/16 17:00
4	▦	Conception & modelisation of ADS application	14 Days	03/05/16 08:00	20/05/16 17:00
5	▦	ADS Implementation	29 Days	21/05/16 08:00	30/06/16 17:00
6	▦	Conception & modelisation of Registration module	6 Days	01/07/16 08:00	08/07/16 17:00
7	▦	Registration module Implementation	24 Days	09/07/16 08:00	11/08/16 17:00
8	▦	Test & validation	3 Days	12/08/16 08:00	16/08/16 17:00

Figure 1.4 – Gantt Diagram

Chapter 2

3-D Secure protocol

2.1 Introduction

The fast advancing global information infrastructure, including information technology and computer networks such as the Internet and telecommunications systems, enable the development of electronic commerce at a global level. Everyone involved in electronic commerce whether a merchant, bankcard Acquirer, processor or payment gateway know the challenges faced with authenticating card-not-present Internet-based payments. We can't obtain and validate a customer's signature, nor we can record the contents of the magnetic stripe on the card.

To reduce the number of disputed online purchases, there is a need for a means to enable Issuers to verify that **the actual cardholder is the person performing the Internet purchase**. This process has been termed **Payer Authentication**.

In early 2001, VISA introduced a security protocol called 3-D Secure to improve transaction performance online and to accelerate the growth of electronic commerce through increased consumer confidence.
3-D Secure is an XML-based protocol designed to be an extra security layer for online credit and debit card transactions. It was originally developed by **Arcot Systems**, with the intention of improving the security of Internet payments and is offered to customers under the name Verified by Visa.

This chapter summarizes the roles of participants in the S2M 3D-Secure program and provides a high level review of the 3D Secure technology components.

2.2 Components and participants

The 3D-Secure Specification is developed to improve transaction performance online and to accelerate the growth of electronic commerce. The 3D-Secure project implemented within S2M is an authentication program based on this specification. The 3-D Secure protocol is a technical platform that includes technical specifications and requirements for issuers, acquirers, and merchants. In addition to utilizing the widely supported Internet technology Secure Sockets Layer (SSL) encryption to protect payment card information during transmission over the Internet, 3-D Secure uses cardholder authentication to verify the parties involved in the transaction.

2.2.1 Components

3-D Secure in a nutshell stands for 'Three Domain Model' for secure payment systems. The model divides payments into three distinct 'domains':

- **The Issuer Domain:** systems and functions of the Issuer and its cardholders.
- **The Acquirer Domain:** systems and functions of the Acquirer and its merchants.
- **The Interoperability Domain:** systems and functions that enable the Issuer Domain and the Acquirer Domain to interoperate and authenticate each other worldwide.

This figure depicts the three domains of 3-D Secure protocol

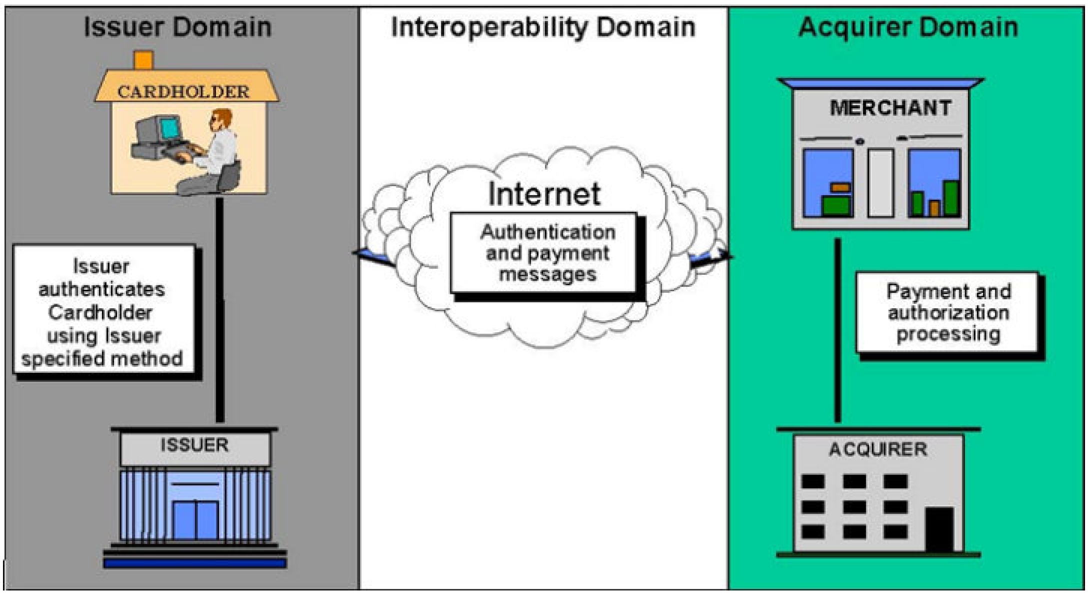

Issuer Domain
Interoperability Domain
Acquirer Domain
CARDHOLDER
MERCHANT
Internet
Authentication and payment messages
Issuer authenticates Cardholder using Issuer specified method
Payment and authorization processing
ISSUER
ACQUIRER

Figure 2.1 – Overview of 3-D Secure Protocol

2.2.2 Program participants

This table summarizes the roles of participants in the 3D-Secure program:

Participant Description	Role in 3DS
Issuer: Financial institution that issues cards to cardholders	-Manages cardholder participation and registration in 3DS program -validates cardholder at the time of each online purchase; -provides digitally signed response to merchant for each authenticated transaction
Cardholder: Account holder of payment card	-Uses card to pay for purchases over the Internet. -Registers one time only. -Provides password at time of purchase.
Acquirer: Financial institution that contracts with merchants for acceptance of payment cards	-Registers merchants for 3DS; -Ensures that merchants originating Internet transactions are operating under a merchant Agreement with acquirer in accordance with the business rules and technical requirements for the 3DS program.
Merchant: Offers merchandise or services at a website and accepts payment from a cardholder who makes purchases over the Internet	-Operates software to support 3DS program. This software is referred to as merchant server software or as a merchant Server Plug-in (MPI). May develop and implement its own MPI or may obtain technology products and consulting services (including software integration into the merchant's commerce environment) from a technology provider.
Global Network Transaction (VisaNet as an Example): Systems and services, including the Visa Integrated Payments System	-Verifies issuer's authentication results. Routes authorization requests to issuers and sends responses to acquirers for return to merchants.

Table 2.1 – Roles of 3DS Program Participants

2.3 Software components

The 3-D Secure protocol divides the authentication process into three parts or "domains" according to the participants involved:

- **Issuer Domain:** Issuers and cardholders.
- **Acquirer Domain:** Acquirers and merchants.
- **Interoperability Domain:** The operated systems that connect the issuer and acquirer Domains.

This figure illustrates and describes the key software components in each domain

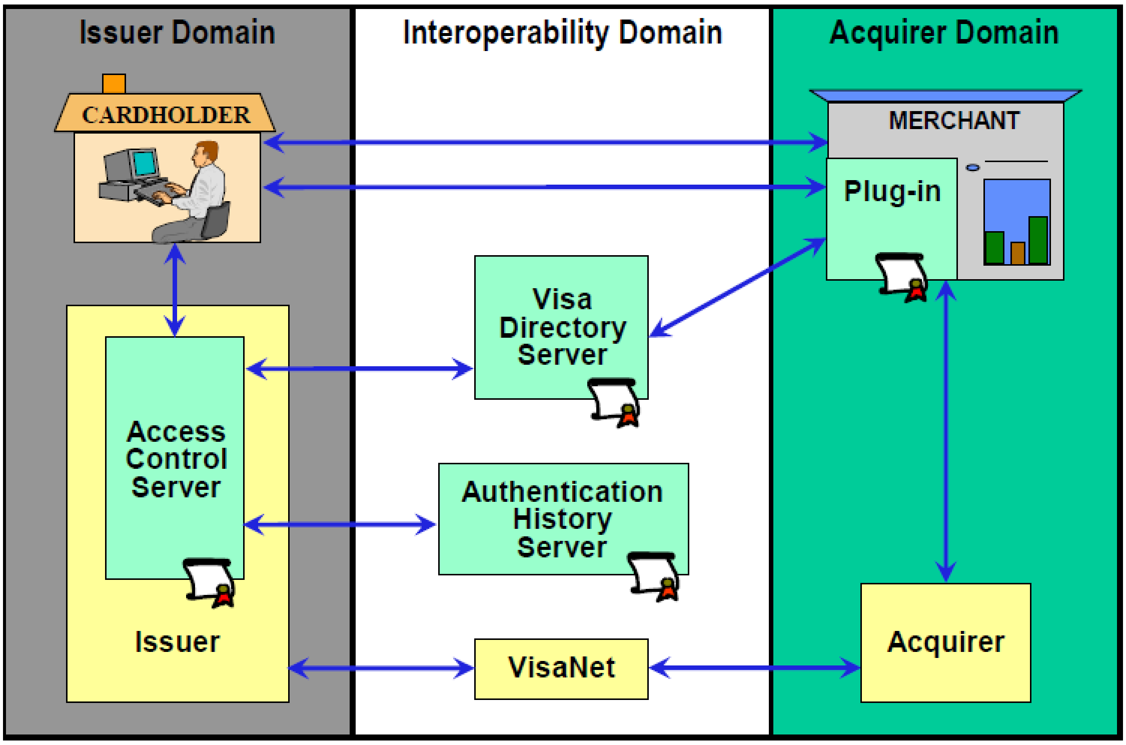


Figure 2.2 – The Three Domains of 3-D Secure

2.3.1 The Issuer Domain

Issuer Access Control Server (ACS)

A server with registered cardholder account and access information. The ACS is operated by the issuer or its processor. It validates cardholder participation in the program, performs cardholder verification (and, for chip cards, card authentication) at time of purchase, and provides digitally signed responses to merchants.

Issuer Registration Server (RS)

A server that manages the cardholder registration process. The server authenticates the cardholder for registration by presenting a series of questions to be answered by the cardholder

and verified by the issuer, or by using a similar identity authentication process. The Registration Server is operated by the issuer or its processor.
NOTE: *The Registration Server is not necessarily a separate server. Registration functionality may be provided by the ACS.*

Issuer Implementation Options

Issuers have two options from which to choose when implementing the 3DS program.

- **Issuer Develops Own Software:** An issuer may develop its own software capabilities using the 3-D Secure technology platform. Visa and MasterCard have published a set of technical specifications for 3-D Secure.
- **Issuer Obtains Software from Vendor:**Alternatively, an issuer may license or purchase software capabilities developed by an approved technology provider for operation in the issuer's data center. In this case, it is important to ensure that the software has passed the 3-D Secure Compliance Testing Process. Vendors successfully completing this testing will receive an acknowledgement of 3-D Secure compliance.

Attempts Access Control Server (AACS)

A server that supports attempts functionality, providing a proof of authentication attempt when authentication is not available. Will respond on behalf of:

- Non-participating issuers.
- Participating issuers with non-enrolled cardholders.
- ACS implementations that are not able to respond.

2.3.2 The Acquirer Domain

Merchant Server Plug-in (MPI)

(Also referred to as *merchant server software*) A software module integrated into merchant websites, used to provide the interface between the 3DS program and the merchant's payment processing software. May verify the issuer's digital signature used to sign authentication responses returned to the merchant, or this verification may be performed by a separate server, the acquirer, or a third party. Provides authentication data fields for inclusion in authorization processing.

2.3.3 The Interoperability Domain

The Directory Server (Visa directory on the picture)

A server operated by an electronic banking center (such as Visa or MasterCard) to route authentication requests from merchants to issuer Access Control Servers and to return the results of authentication.

Authentication History Server

A server operated by electronic banking center to store data about authentication transactions. The Authentication History Server is used to verify authenticated transactions and to provide information during the dispute resolution process.

2.4 3-D Secure Enrollment process

In order to participate on the 3D-Secure protocol, the issuer must make possibilities to cardholders through providing registration options to them.
There are a variety of registration options an issuer can implement to register cardholders. Regardless of the type of registration process selected, issuers should take care in selecting elements to register cardholders in the 3D-Secure program in order to minimize the risk of fraudulent registrations. Issuers should carefully consider the data elements to verify the cardholders identity and must use a combination of both in-wallet and out-of-wallet elements—for example, (in-wallet elements include: **Signature Panel Code** (the Card Verification Value 2 or **CVV2**) and **Expiry Date**; out-of-wallet elements include: Last 4-6 digits of Social Security Number/National Identity Number, Telephone number or Mother's maiden name).

The following are registration options that issuers should consider:

- **Stand-alone registration** positions 3DS project as a separate program offered by the issuer and does not fully integrate with any existing cardholder services or registration processes. Examples:
  - Registration Website – Cardholders are authenticated and registered through a Web interface using existing cardholder information from the issuer's master file.
  - Direct Mail Solicitation – Issuers send direct mail solicitations to cardholders, soliciting them to register for 3DS.
- **Integrated registration** positions 3DS as an enhancement to existing bank programs and incorporates the registration process into processes that cardholders already perform (e.g., account activation, online banking, and customer service). Examples:
  - Online Banking Registration – Cardholders are registered in 3DS from within the issuer's online banking website. The issuer authenticates the cardholder using the existing online banking profile.
  - Registration With New Accounts Process – Issuers may choose to educate and register cardholders for 3DS when the cardholder is applying for a new account.
  - Mass Registration – Cardholders are automatically registered in 3DS as part of a mass registration process.
  - Enrollment process which will be more detailed by the next

When choosing a registration option or options, it is recommended that issuers take advantage of current cardholder behavior and incorporate 3DS registration into existing methods of communication, rather than establishing a new process. Generally, registration processes that are based on existing cardholder behavior will have a greater likelihood of success.

In general the matchless registration option which is adopted by Visa known as Verified by Visa program, MasterCard as Secure Code and almost the majority of electronic banking companies is the Enrollment process which is automatic and preferred registration option specifically by cardholders because this approach is one where the participation on 3DS is simplified and make the forwarding information between ACS and other entities gone faster furthermore the enrollment server can be a part of the ACS.

3D-Secure Enrollment: Activation During Shopping

Activation During Shopping is an online program, it's the most popular form of cardholder enrollment designed to make Internet purchase transactions safer by authenticating a cardholder's identity at the time of purchase, before the merchant submits an authorization request. 3DS-ADS software installed at the merchant's site activates the cardholder interface during the authentication process.

The goal of 3DS-ADS is to create a level of consumer trust and confidence in online shopping similar to that in the physical shopping environment. It is designed to improve both cardholder and merchant confidence in Internet purchases and to reduce disputes and fraudulent activity related to the use of payment cards. Issuers can benefit from reduced costs associated with the most common types of Internet disputes. The Interface of 3DS-ADS is displayed to the cardholder during registration and each time the cardholder enters a password for authentication at the time of purchase, and may also be displayed by participating merchants.

3DS-ADS supports a variety of Internet access devices including personal computers, wireless devices such as mobile phones, personal digital assistants, and set-top boxes. 3DS-ADS can operate with multiple authentication technologies including passwords, digital certificates, and chip cards. Merchant processing is independent of the authentication techniques adopted by the card issuer. This is a very attractive option as it facilitates a fast adoption rate and has helped enroll a critical mass of cardholders into 3DS program.

Issuers that enroll and activate their cardholders via Activation During Shopping (ADS) will derive substantial benefits in leveraging that same mechanism for password recovery.

2.5 Online Purchases

After enrolling as described in the part of Enrollment, the cardholder is ready to use 3DS at any 3DS-enabled merchant. We take as an example the Verified by Visa program which is developed by Visa for describing the transaction flows

This figure illustrates the purchase transaction flow

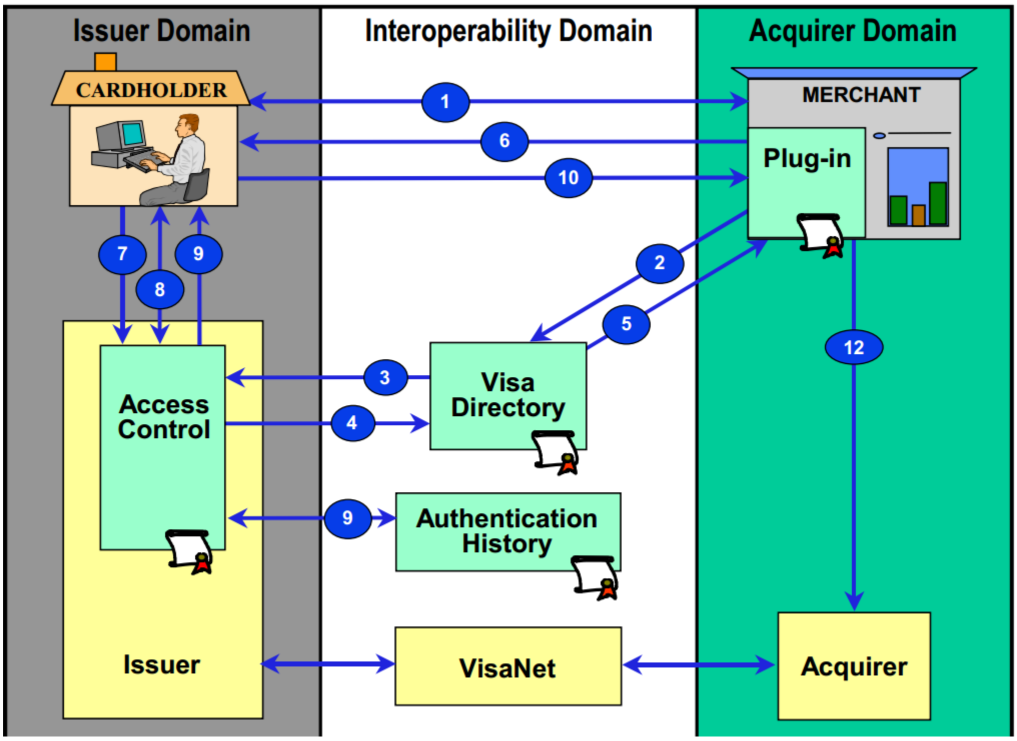


Figure 2.3 – Purchase Transaction Flow

Step 1. Cardholder finalizes purchase.
The cardholder browses at a participating merchant's website, adds items to the shopping cart, provides information required for checkout (by key-entering data or by using an electronic wallet, a merchant one-click service, or some other form-fill method), then clicks "Buy". (The merchant now has all necessary data, including the PAN.)
NOTE: Steps 2-7 are invisible to the cardholder.

Step 2. Merchant Server Plug-in initiates 3-D Secure processing.
When the cardholder clicks "Buy", the merchant Server Plug-in (MPI) is activated. The MPI sends the PAN and other information to the Visa Directory Server.

Step 3. Visa Directory Server processes request.
The Visa Directory Server authenticates the merchant. If merchant authentication is successful, the Visa Directory Server queries the appropriate Access Control Server (ACS) to determine whether authentication is available for the PAN. If merchant authentication fails or if no appropriate ACS is available, the Visa Directory Server creates a response for the MPI and processing continues with Step 5.

Step 4. ACS responds to Visa Directory Server.
The ACS determines whether authentication is available for the PAN, prepares a response, and sends it to the Visa Directory Server.

Step 5. Visa Directory Server forwards response.
The Visa Directory Server forwards the ACS response (or its own) to the MPI. If authentication is available, the response includes the URL of the ACS to which the merchant will send the Payer Authentication Request.

Step 6. MPI sends Payer Authentication Request.
If authentication is not available, then the MPI advises the merchant commerce server that authentication is not available, and processing continues with Step 12. The MPI sends the Payer Authentication Request to the ACS via the cardholder's device, using the URL received in Step 5. The Payer Authentication Request contains information about the purchase transaction.

Step 7. ACS receives Payer Authentication Request.

Step 8. ACS authenticates cardholder.
The ACS authenticates the cardholder using processes applicable to the PAN (password, chip, PIN, etc.). For authentication by password, for instance: If the Cardholder has not yet enrolled in 3DS and if the Issuer supports Activation During Shopping, the Cardholder will instead see Activation During Shopping screens, as described in 3D-Secure Enrollment: Activation During Shopping. The ACS formats a Payer Authentication Response with appropriate values, including authentication status and, if applicable, Electronic Commerce Indicator (ECI) and Cardholder Authentication Verification Value (CAVV). The ACS signs the Payer Authentication Response with the issuer's 3DS Signature Key. (This allows the merchant to verify that the response originated from a valid participating issuer.)

Step 9. ACS sends results.
The ACS returns the signed Payer Authentication Response to the MPI via the cardholder's device. Whether or not authentication was successful, the ACS sends a copy of the Payer Authentication Response, plus related data, to the Authentication History Server. The server provides transaction reporting to issuers and serves as the database of record for dispute resolution.

Step 10. MPI receives Payer Authentication Response.

Step 11. MPI processes response.
The MPI validates the signature on the Payer Authentication Response (either by performing the validation itself or by passing the message to a separate Validation Server), and validates other data in the response. The MPI passes the results of the authentication attempt to the merchant commerce server.

Step 12. Authorization processing.
Based on the data received from the MPI, the merchant commerce server determines whether to proceed with authorization. If the merchant commerce server advises the MPI that

authentication failed, the merchant should request another form of payment from the shopper.

If authorization is appropriate:

- The merchant commerce server sends an authorization request to the merchant's acquirer or third party payment processor. The authorization request includes the Electronic Commerce Indicator (ECI) appropriate to the authentication status which will be defined and detailed in the part of authorization processing.
- The acquirer sends the authorization request, including 3DS authentication information, to the issuer via VisaNet.
- The issuer receives and processes the authorization request. If the issuer generated a CAVV(Card Authentication Verification Value) during the authentication process, the issuer will expect to receive the CAVV in the BASE I authorization message. When the CAVV is passed in BASE I, either the issuer or Visa on its behalf will perform CAVV verification.
- The issuer returns an authorization response. The issuer may choose to decline the authorization request for reasons unrelated to the 3DS authentication (e.g., insufficient open-to-buy)
- If the issuer authorizes the transaction, the merchant displays an order confirmation as usual, providing the cardholder with details about the order, delivery, and the merchant's customer service

Authentication History Transmission

For each Payer Authentication Response generated, whether or not authentication was successful, the ACS must send a copy plus related data to the Authentication History Server, in the form of a **PATransReq**. Some regions may require that the issuer ACS send the **PAReq** and **PARes** to the AHS as well.

2.6 3-D Secure Messages

This section briefly describes each 3-D Secure message involving the issuer.

2.6.1 Verify Enrollment Messages

When the shopper initiates payment, the merchant server software formats a message to the Directory Server to determine whether the shopper's card is enrolled in 3-DS program.

Verify Enrollment Request or VEReq	The merchant server software sends the VEReq messageto the Directory Server to determine whether a particular card number is enrolled and cantherefore be authenticated

Verify Enrollment Response or VERes	The Directory Server returns the VERes message, indicating one of the following responses:
	Y = Authentication Available
	Cardholder Enrolled (or attempts processing available) – URL of issuer ACS is included in response
	N = Cardholder Not Participating
	Cardholder Not Enrolled
	U = Unable to Authenticate

Error message	The VEReq could not be processed because the merchant was unable to provide the appropriate credentials to the Directory Server or for other reasons, such as a field that is formatted incorrectly.

Table 2.2 – Verify Enrollment Messages

2.6.2 Payer Authentication Messages

Upon receiving a Verify Enrollment Response indicating that authentication is available (in the VERes message returned by the Issuer, the PAN Authentication Available field had the value "Y" (indicating "Authentication Available") and all other fields passed all required edits), the merchant server software formats a Payer Authentication Request (PAReq), and forwards it to the issuer ACS whose URL was included in the VERes, via the cardholder's browser.

Payer Authentication Request or PAReq	The merchant server software sends the PAReq message to the issuer ACS to request cardholder authentication. Contains cardholder, merchant, and transaction-specific information
Payer Authentication Response or PARes	The issuer ACS returns the PARes message, including the issuer's authentication decision (as described in the next table)
Error message	The PAReq could not be processed because, for example, a field was formatted incorrectly.

Table 2.3 – Payer Authentication Messages

The following table lists the possible outcomes of payment authentication, and the codes used to communicate the issuer's authentication decision. The values displayed are those inserted in the "Transaction Status" field of the Payer Authentication Response message that is returned to the merchant.

Authentication Result Determined by Issuer ACS	**Transaction Status Values**
Authentication Successful The cardholder's password (or other authentication information) was successfully validated. If the cardholder has a chip card, the chip card cryptogram was also successfully validated	Y
Authentication Failed The cardholder's password (or other authentication information) and/or the chip card cryptogram failed validation.	N
Authentication Could Not Be Performed This is a neutral response status, indicating that the issuer was not able to perform cardholder authentication. Reasons include a misrouted request, a system error, inconsistent transaction data, and so on	U
Attempts Processing Performed Authentication was not available, but functionality was available (through the issuer, Visa or MasterCard, or a third party) to generate a proof of authentication attempt.	A

Table 2.4 – Issuer Authentication Results Values

2.6.3 Authentication History Messages

The Access Control Server (ACS) must send to the Authentication History Server (AHS) a record of each transaction for which a Payer Authentication Response was returned, whether the authentication was successful or not.

Payer Authentication Transaction Request or PATransReq	The issuer ACS sends the **PATransReq** message (which contains the **PARes**) to the Authentication History Server, which archives authentication activity for use by acquirers and issuers for dispute resolution and other purposes. *Note: It may required that the issuer ACS send the PAReq and PARes to the AHS as well.*
Payer Authentication Transaction Response or PATransRes	The Authentication History Server responds with a PATransRes, to advise the ACS that the transaction was received and recorded.
Error message	The **PATransReq** could not be processed because,for example, the message was corrupted.

Table 2.5 – Authentication History Messages

2.7 Authorization Processing

Upon receiving the Payer Authentication Response (as described in **"Online Purchases"** and **"Payer Authentication Messages"**), the MPI:

- Validates the issuer's digital signature using the Root Certificate.
- Validates other fields in the Payer Authentication Response The MPI notifies the merchant commerce server if either validation fails. These transactions are treated as though authentication had failed. The merchant should request another form of payment from the shopper. Visa or MasterCard may not permit a merchant to submit a failed authentication transaction for authorization. If both validations are successful, then the MPI reviews the value in the Transaction Status field of the Payer Authentication Response (PARes) message to determine the payment authentication results, and passes the results to the merchant commerce server, along with the Electronic Commerce Indicator (ECI) and Cardholder Authentication Verification Value (CAVV), if received.

This table below lists the Transaction Status field values and appropriate merchant processing for each.

Results of Authentication Attempt	**Transaction Status**	**Merchant Response**
Authentication Successful	Y	The merchant submits an authorization request including the ECI and CAVV supplied in the PARes.
Authentication Failed	N	The merchant should not submit a failed authentication for authorization.
Authentication Could Not Be Performed	U	The merchant may process an authorization request using the appropriate ECI.
Attempts Processing Performed	A	The merchant submits an authorization request including the ECI and CAVV supplied in the PARes.

Table 2.6 – Transaction Status Values

Electronic Commerce Indicator (ECI)

The Electronic Commerce Indicator (ECI) indicates the level of security used when the cardholder provided payment information to the merchant. It must be set to a value corresponding to the authentication results and the characteristics of the merchant checkout process. The merchant commerce server transmits the authorization request message, including the ECI, to the acquirer or its processor.

NOTE: *The acquirer/merchant must use the correct ECI in both the authorization and clearing messages to ensure liability protection.*

Possible ECI data values are:

- **ECI = 5**: This value is set by the ACS in the Payer Authentication Response message when the cardholder successfully passed 3DS payment authentication.
- **ECI = 6**: This value is set by the merchant when the merchant attempted to authenticate the cardholder using 3DS, but the issuer or cardholder was not participating, or set by an Attempts ACS when the issuer or cardholder was not participating or an issuer ACS was not able to respond.
- **ECI = 7**: This value is set by the merchant when the payment transaction was conducted over a secure channel (for example, SSL/TLS), but payment authentication was not performed, or when the issuer responded that authentication could not be performed (That is, the Issuer returned a PARes message in which the Transaction Status field had the value “U”, indicating “Authentication Could Not Be Performed”) (for example, the ACS was unable to match the account ID from the PAReq to the corresponding VEReq, or payment authentication was attempted on an excluded channel or product).

NOTE: *ECI values in BASE I are two digits – the values above with a leading zero. ECI values in the Single Message System and in BASE II are single digits as shown above and ECI value is also included in the clearing/settlement message.*

Cardholder Authentication Verification Value (CAVV)

The **CAVV** is a cryptographic value derived by the issuer during payment authentication that can provide evidence of the results of payment authentication during an online purchase. Issuers must include the CAVV in each Payer Authentication Response message sent to the MPI in which the Transaction Status value is either "Y" (Authentication Successful) or "A" (Attempts Processing Performed). When a merchant receives a CAVV value in a Payer Authentication Response message from the issuer, the CAVV must be included in the global network transaction (VisaNet as an example) authorization message in order for the merchant to receive chargeback protection for inter-regional transactions. Merchants must be able to send the CAVV to their acquirers. Acquirers must be able to receive the CAVV from participating merchants and correctly transmit the data in the authorization request.

Note: *The CAVV is generated by the ACS using the same algorithm as CVV, but different data elements.*

The issuer uses the CAVV received in the authorization request to confirm that the cardholder was successfully authenticated for this transaction (or that a proof of authentication attempt was created).

2.8 Access Control Server Functions

The following paragraph describes the functions of the Access Control Server (ACS) in 3DS program. The primary functions of the ACS are:

- To ascertain whether 3DS authentication is available for a particular cardholder
- To respond to requests for payment authentication
- To convey payment authentication information to the Authentication

History Server In addition to the ACS functions listed in this chapter, the issuer must provide a means of enrolling cardholders in 3DS. Enrollment may be included in ACS functionality, or may be provided by means of a separate Enrollment Server

2.8.1 Message Processing Functions

The ACS must perform the following message processing functions:

— Receive and process Verify Enrollment Request (VEReq) messages from the MPI via the Directory Server.

— Format Verify Enrollment Response (VERes) messages and send them to the MPI via the Directory Server.

— Receive and process Payer Authentication Request (PAReq) messages from the MPI via the cardholder's device.

— Format and sign Payer Authentication Response (PARes) messages and send them to the MPI via the cardholder's device
Note: To perform this function, the ACS must be able to calculate the cardholder Authentication Verification Value (CAVV), as discussed in **"Cardholder Authentication Verification Value (CAVV)"**.

- Format Payer Authentication Transaction Request (PATransReq) messages and send them to the Authentication History Server.
- Receive and process Payer Authentication Transaction Response (PATransRes) messages from the Authentication History Server Format, send, receive, and process Error messages as required.

2.8.2 Certificate usage and types

Certificate usage

The Access Control Server uses X.509-based certificates as a means to ensure authentication between systems, to encrypt sensitive data, and to sign 3DS messages to serve as a basis for non-repudiation for merchant liability protection.

Certificate types

ACSs must use "Visa Brand CA" or "MasterCard CA" (in the case of "MasterCard") issued Digital Certificates to authenticate to the Interoperability Domain components (a server Digital Certificate to the Directory Server and a client Digital Certificate to the Authentication History Server). ACSs must use a commercial third-party SSL server Digital Certificate to authenticate to the cardholder browser.

Issuers who participate in 3DS have two options for operating an Access Control Server.

1. Member operates its own ACS.
2. Member designates a third-party processor to operate an ACS on its behalf. If the Member elects to operate their own ACS, they need an ACS Member Signature Digital Certificate. If the Member elects to designate a third-party processor to operate an ACS on their behalf, the processor needs an ACS Processor Signature Digital Certificate. The Member or third-party processor that operates the ACS will request and install the relevant Digital Certificate. ACS Member Signature Digital Certificates are used by and for individual Issuers. Third-party processors may use a single ACS Processor Signature Digital Certificate for those Issuers who use a unique Issuer CAVV key. For issuers that share a CAVV key, an ACS processor must use unique ACS Member Signature Digital Certificates. If a third-party processor operates the ACS, the member needs to have a valid third-party processor designation on file with the electronic banking center or the financial services corporation.

Chapter 3

Activation During Shopping

This chapter is dedicated to report the different steps of Activation ADS Program achievement starting by enumerating all the requirements and specifications going up to the part of design and development without forgetting to quote the details of tools and technologies used to produce this module and finally we will shed light on illustrations and demonstrations that depict efficacy and smoothness of the program.

3.1 ADS Requirements Specification

3.1.1 Executive Summary

Project Overview

Activation During Shopping (ADS) is an option which take a part of the 3D-Secure payment protocol and gives cardholders the opportunity to enroll in 3D-Secure while they are shopping at a participating Merchant site. This present project has as purpose to achieve designing and developing the features of this option on the enrollment phase of 3D-Secure.

Purpose and Scope of this Specification

The principal purpose of this project is to mitigate the risks of fraud and adding an extra layer of security then facilitate the cardholder participation into the 3D-Secure protocol during online purchase. Online shoppers can take the initiative to enroll in 3D-S and validating their identity at once through the ADS interface which will be developed.

3.1.2 Product/Service Description

The cardholders can benefit from this service which allow them to activate there participation to 3D-S from there devices just with providing the answers to some questions that appear in the ADS Interface.
When a cardholder uses his or her card for an Internet purchase, a pop-up window displays his or her basic cardholder information and a one-time password is sent to the cardholder via

SMS in order to verify his or her identity (this OTP is required to enroll the cardholder in 3D-Secure). During all subsequent Internet transactions, an OTP is sent to the cardholder, who must use it to complete the purchase. Activation also supports cards with multiple cardholders.

3.1.3 Requirements

System requirements for designers to design a system satisfying the requirements and testers to verify that the system satisfies the requirements. We describe at this section input and output into/from the system, and functions that are performed by the system.

Functional Requirements

- Ensure integration of the service in the enrollment server
- Ensure interaction with access control server
- During the activation process, an OTP must be sent to the cardholder via SMS Gateway
- Before the ADS begin, the enrollment server must verify the identity of the cardholder
- The ADS form must contains both in-wallet and out-of wallet data elements
- After the ADS process, the cardholder must continue his purchase as usual
- The ADS must be happen once per cardholder
- the ADS can be performed more than one time , if the cardholder have multiple account (optionnal)
- If the cardholder is already enrolled in 3D-Secure , the ADS must not be performed
- After the ADS, the enrollment server have to send the enrolled data to the Access Control server
- After the ADS, the enrollment server have to inform the cardholder about the activation service
- The enrollment server should set a timeout during the Activation process
- For every activation, the Enrollment server have to log the event for a posterior access

User Interface Requirements

In addition to functionals requirements , we describe the characteristics of interface between the Activation During Shopping precess and its users

- The account validation screen of the service must be in a pop-up window (half screen of the browser)
- A help button have to be on the top left of the pop-up window
- the cardholder should agree the terms and conditions by checking a box field
- After filling the requested data elements, at the same window the cardholder must enter the OTP received on his or her phone
- An exit button is also required on user interfaces
- if the cardholder is connected through a poxy server , the ADS must not be performed
- the cardholder have to accept cookies on his or her browser

Usability

We include specific usability requirements which are affected by cardholders:

- The user documentation for Activation During Shopping and help should be complete

- The help should be context sensitive and explain how to achieve common tasks
- The service have to be easier to use by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.

Performance

We specify in this section static and dynamic numerical requirements placed on the enrollment server in interaction with the ADS service.

Capacity

Capacity requirements ,the number of simultaneous cardholders to be supported, the maximum simultaneous user load, and memory requirements are specified later. **Availability**

Specific and measurable requirements for ADS are:

- Hours of operation
- Level of availability required (percentage)
- scheduled and unscheduled maintenance on uptime
- Maximum permitted number of failures per hour

Latency

We assume the maximum acceptable timeout for Activation During Shopping service up to 3min. (except mistake)

Manageability/Maintainability

Monitoring

- The Enrollment server must detect error on Activation process.
- The Enrollment server should log events.
- A service health monitoring must be performed on Activation process.

Operations

We specify any normal and special operations on activation process by the cardholders:

- Periods of interactive operations
- Periods of unattended operations
- backup and recovery operations
- disaster recovery and business resumption

System Interface/Integration

Other products are required to use in order to integer the ADS on the Enrollment server (e.g., database, operating system), also we specify what devices are to be supported, and protocols used (e.g., signal handshake protocols).

Security

Protection

We specify the factors that will protect the system from malicious or accidental access, modification, disclosure, destruction, or misuse :

- Encrypt all communications with the appropriate ciphers algorithms

- Activity logging, historical data sets
- Restrictions on intermodule communications
- Data integrity checks

Authorization and Authentication

We define the Authorization and Authentication factors:

- Consider using standard tools such as PubCookie.
- All sensitive data must be encrypted on database
- Restrictions on the Enrollment server must be performed
- Include the role strategy on the server
- Encrypt data on cardholders browsers
- According to visa requirements security , cookies must also been encrypted
- The Enrollment server must use certificates to identify himself to third-party.

Data Management

These are requirements for any information that is to be placed into a database:

- types of information used by various functions
- integrity constraints
- frequency of use
- data access rules
- data entities and relationships
- data formats

3.2 Analysis of existing

3.2.1 Functional description of the existing

Activation During Shopping provide issuers the ability to facilitate enrollment cardholders at the time of shopping.

In order to maintain flexibility for issuers implementing ADS, two methods have been developed to facilitate cardholder enrollment:

— Issuer-assigned Password Activation
  Cardholder is pre-enrolled (e.g., with Internet banking password) and activates at time of shopping.

— Cardholder-created Password Activation
  At time of shopping, issuer confirms cardholder identity information, then cardholder selects their password.

Unfortunately neither the first nor the second method were implemented at the host organization.

Our mission was to implement the ADS interface at the first ,in order to provide cardholder enrolling himself at the 3-D Secure during shopping.

3.2.2 Organizational description of the existing

To respect the privacy policy of our internship, we inform the reader that the implementation of the ADS interface, is a need required by a bank that wishes to benefit from the enrollment of its customers using the protocol 3-D Secure.

3.3 Development and modelization methodology

3.3.1 Waterfall model

The **waterfall model** is a popular version of the systems development life cycle model for software engineering. Often considered the classic approach to the systems development life cycle, the waterfall model describes a development method that is linear and sequential. Waterfall development has distinct goals for each phase of development.

The **choice** of waterfall development is that it allows for departmentalization and managerial control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process , and theoretically, be delivered on time. Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order, without any overlapping or iterative steps.

This figure describe the waterfall model used on our project

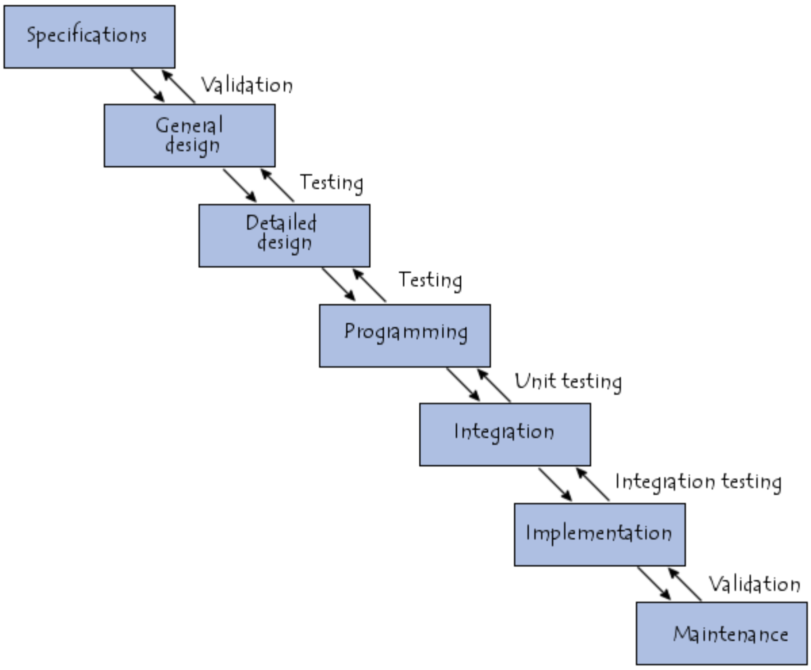


Figure 3.1 – Waterfall model

3.3.2 Modelization of ADS application

Approach used for modelization

Our approach for modeling is using the **Unified Modeling Language**, known as UML. Unified Modeling Language (UML) is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system. It offers a standard way to write a system's blueprints, including conceptual things such as business processes and system functions as well as concrete things such as programming language statements, database schemas, and reusable software components.

Proof of our approach

UML breaks the complex system into discrete pieces that can be understood easily.And also complex system can be understood by the disparate developers who are working on different platforms.UML model is not a system or platform specific. It unifies all disparate developers under one roof.

Use case diagram

We present below the use case diagram that gives an overview on the functional behavior of the ADS process.

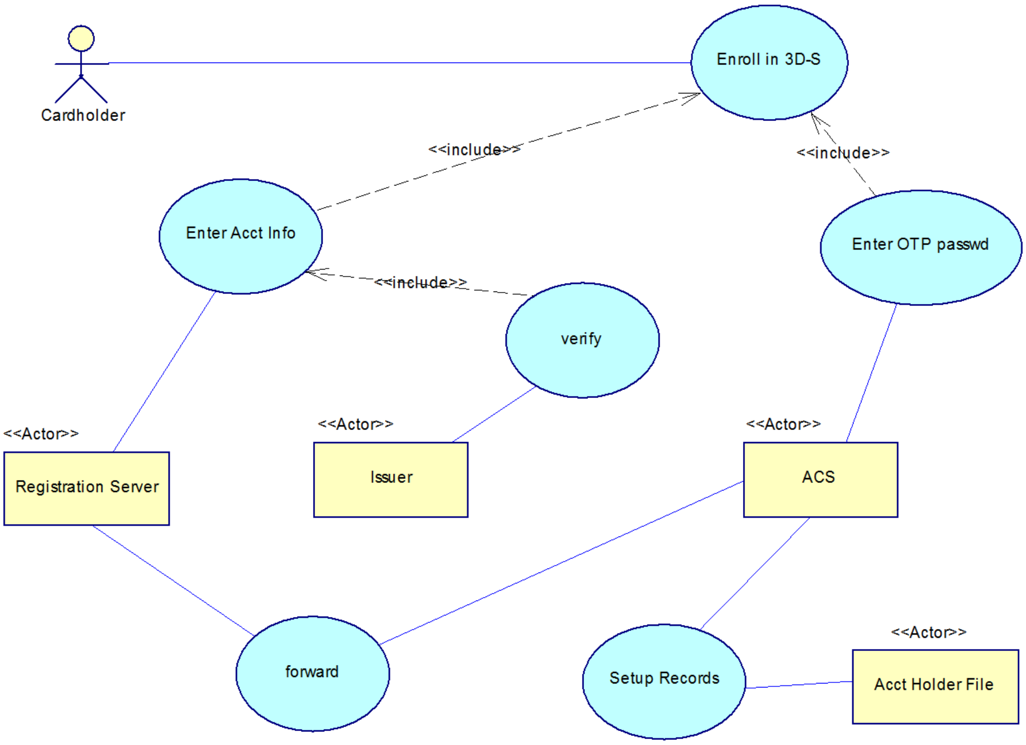


Figure 3.2 – Use Case Diagram

Description

Title: Enroll in 3D-S

Summary: this use case allows the cardholders to participate in 3D-Secure online payment if they want to perform a purchase from a participant website.

Actor: Cardholder (principal)

Preconditions:

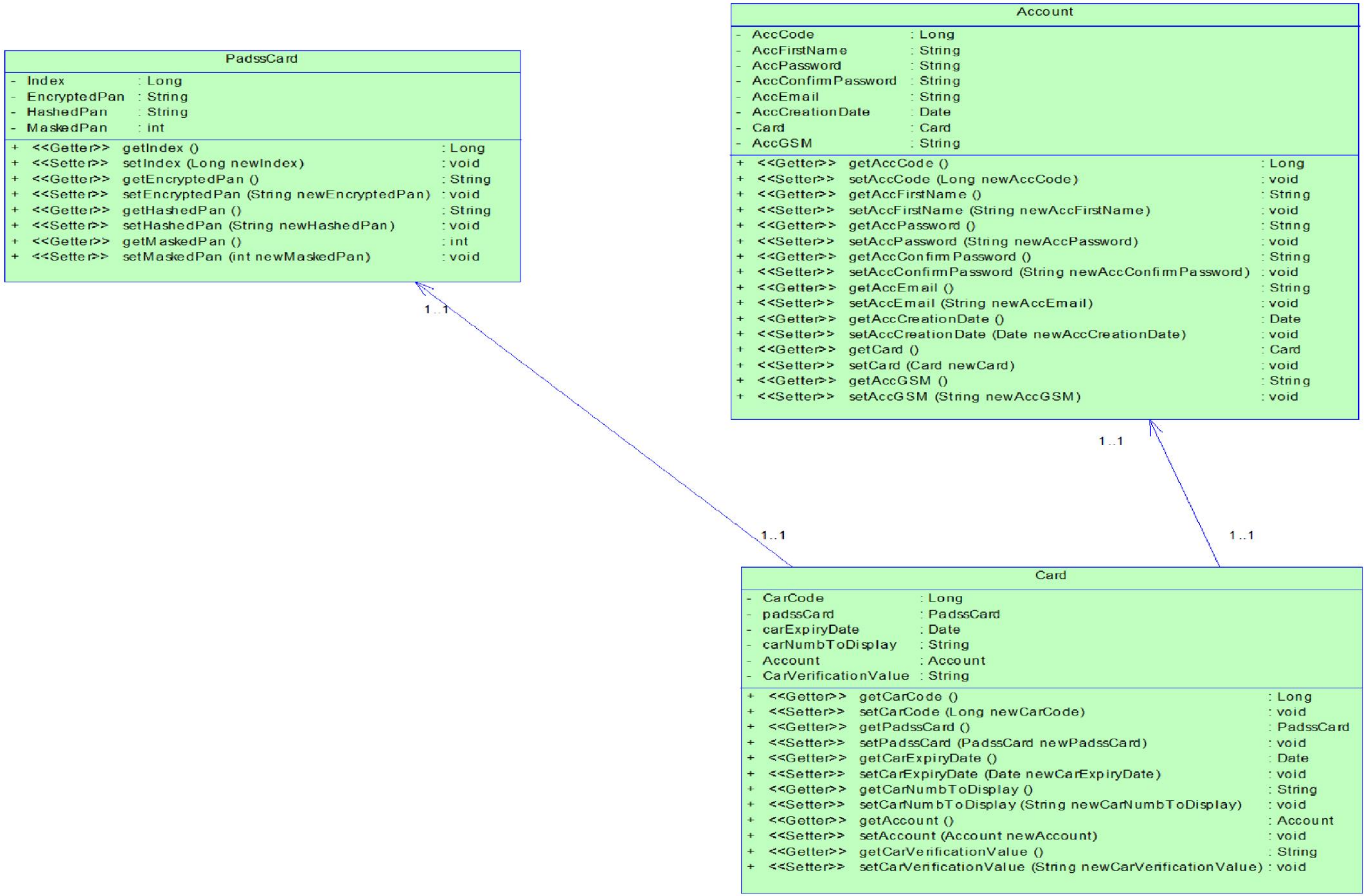
- The website must support the 3D-Secure
- The issuer bank (cardholder's bank) must be a participant in 3D-Secure
- The connection to the website and to the other components must be available and operational
- The cardholder wants to enroll in 3D-S for the first time.

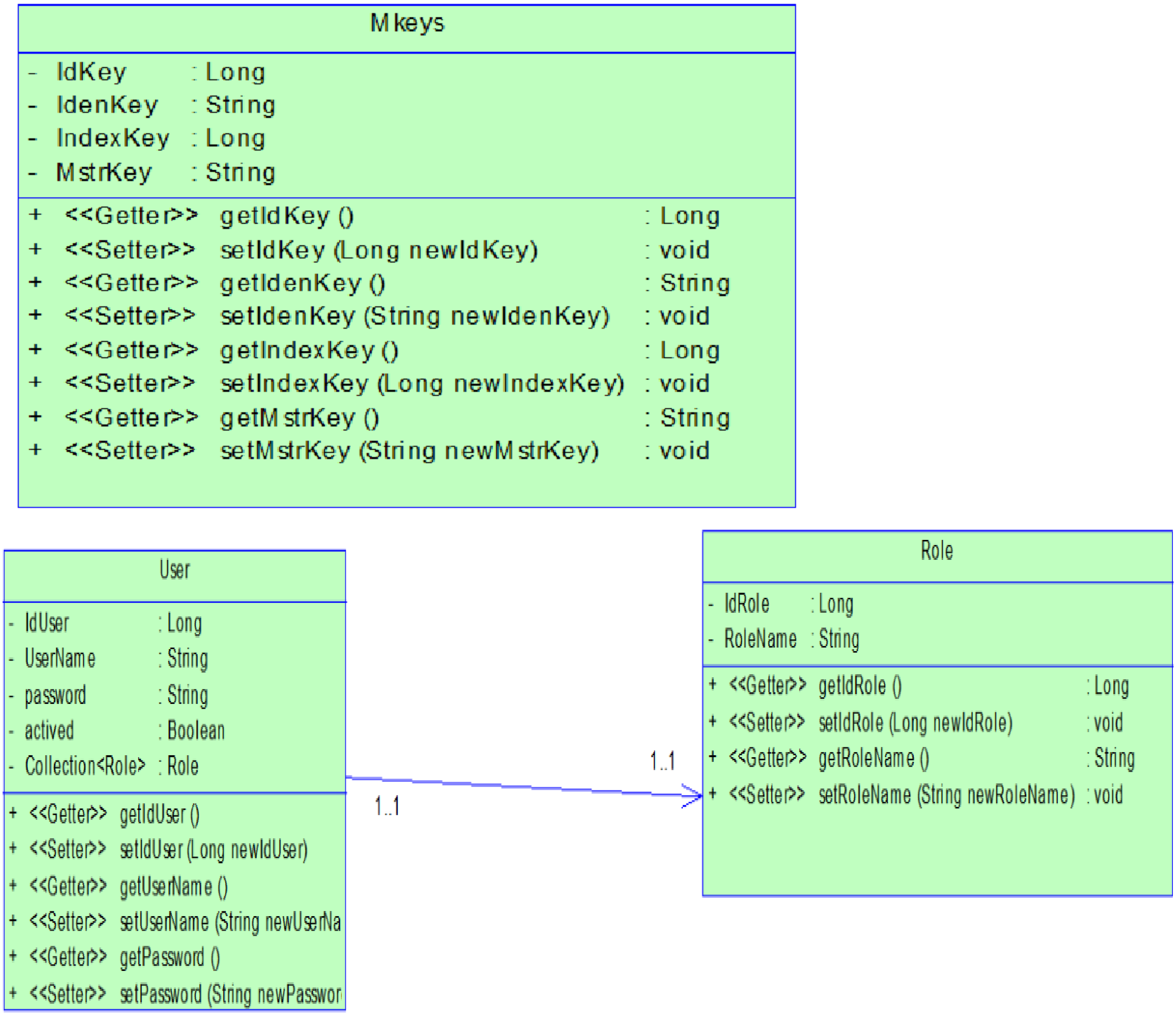
Nominal scenario:

1. The cardholder visits the website and adds items to his shopping cart and finalize purchase.
2. The cardholder is required to enter his account information and submit the data.
3. The data provided by the cardholder will be verified in his issuer which it's recommended to consider this data elements.
4. Upon successful verification by the issuer, the cardholder will be requested to enter an OTP password sent to his phone number and will be verified in the ACS.
5. With the cardholder's account ownership validated, the Registration Server forwards the cardholder information to the issuer Access Control Server (ACS).
6. The ACS setup records in the Account holder file.

Class diagram

The class diagram is a structural diagram for describing the internal structure of classes in terms of attributes and operations, it also allows to represent static assosiations between classes.
This figure represents Class diagram


PadssCard
- Index : Long
- EncryptedPan : String
- HashedPan : String
- MaskedPan : int
+ <<Getter>> getIndex () : Long
+ <<Setter>> setIndex (Long newIndex) : void
+ <<Getter>> getEncryptedPan () : String
+ <<Setter>> setEncryptedPan (String newEncryptedPan) : void
+ <<Getter>> getHashedPan () : String
+ <<Setter>> setHashedPan (String newHashedPan) : void
+ <<Getter>> getMaskedPan () : int
+ <<Setter>> setMaskedPan (int newMaskedPan) : void
Account
- AccCode : Long
- AccFirstName : String
- AccPassword : String
- AccConfirmPassword : String
- AccEmail : String
- AccCreationDate : Date
- Card : Card
- AccGSM : String
+ <<Getter>> getAccCode () : Long
+ <<Setter>> setAccCode (Long newAccCode) : void
+ <<Getter>> getAccFirstName () : String
+ <<Setter>> setAccFirstName (String newAccFirstName) : void
+ <<Getter>> getAccPassword () : String
+ <<Setter>> setAccPassword (String newAccPassword) : void
+ <<Getter>> getAccConfirmPassword () : String
+ <<Setter>> setAccConfirmPassword (String newAccConfirmPassword) : void
+ <<Getter>> getAccEmail () : String
+ <<Setter>> setAccEmail (String newAccEmail) : void
+ <<Getter>> getAccCreationDate () : Date
+ <<Setter>> setAccCreationDate (Date newAccCreationDate) : void
+ <<Getter>> getCard () : Card
+ <<Setter>> setCard (Card newCard) : void
+ <<Getter>> getAccGSM () : String
+ <<Setter>> setAccGSM (String newAccGSM) : void
1..1
1..1
1..1
1..1
Card
- CarCode : Long
- padssCard : PadssCard
- carExpiryDate : Date
- carNumbToDisplay : String
- Account : Account
- CarVerificationValue : String
+ <<Getter>> getCarCode () : Long
+ <<Setter>> setCarCode (Long newCarCode) : void
+ <<Getter>> getPadssCard () : PadssCard
+ <<Setter>> setPadssCard (PadssCard newPadssCard) : void
+ <<Getter>> getCarExpiryDate () : Date
+ <<Setter>> setCarExpiryDate (Date newCarExpiryDate) : void
+ <<Getter>> getCarNumbToDisplay () : String
+ <<Setter>> setCarNumbToDisplay (String newCarNumbToDisplay) : void
+ <<Getter>> getAccount () : Account
+ <<Setter>> setAccount (Account newAccount) : void
+ <<Getter>> getCarVerificationValue () : String
+ <<Setter>> setCarVerificationValue (String newCarVerificationValue) : void


Mkeys
- IdKey : Long
- IdenKey : String
- IndexKey : Long
- MstrKey : String
+ <<Getter>> getIdKey () : Long
+ <<Setter>> setIdKey (Long newIdKey) : void
+ <<Getter>> getIdenKey () : String
+ <<Setter>> setIdenKey (String newIdenKey) : void
+ <<Getter>> getIndexKey () : Long
+ <<Setter>> setIndexKey (Long newIndexKey) : void
+ <<Getter>> getMstrKey () : String
+ <<Setter>> setMstrKey (String newMstrKey) : void
User
- IdUser : Long
- UserName : String
- password : String
- actived : Boolean
- Collection<Role> : Role
+ <<Getter>> getIdUser ()
+ <<Setter>> setIdUser (Long newIdUser)
+ <<Getter>> getUserName ()
+ <<Setter>> setUserName (String newUserNa
+ <<Getter>> getPassword ()
+ <<Setter>> setPassword (String newPasswor
1..1
1..1
Role
- IdRole : Long
- RoleName : String
+ <<Getter>> getIdRole () : Long
+ <<Setter>> setIdRole (Long newIdRole) : void
+ <<Getter>> getRoleName () : String
+ <<Setter>> setRoleName (String newRoleName) : void

3.4 Implementation of Activation During Shopping interface

This section presents the different steps at the implementation of this application. First we will present the languages and tools that helped us in the development of this application. The second section presents the effective implementation of the application explaining its technical and application architecture.

3.4.1 Technologies and tools of development

MVC Model

Model–view–controller (MVC) is a software architectural pattern for implementing user interfaces on computers. It divides a given software application into three interconnected parts, so as to separate internal representations of information from the ways that information is presented to or accepted from the user. Traditionally used for desktop graphical user interfaces (GUIs), this architecture has become popular for designing web applications.

Components:
This figure describe the MVC Model

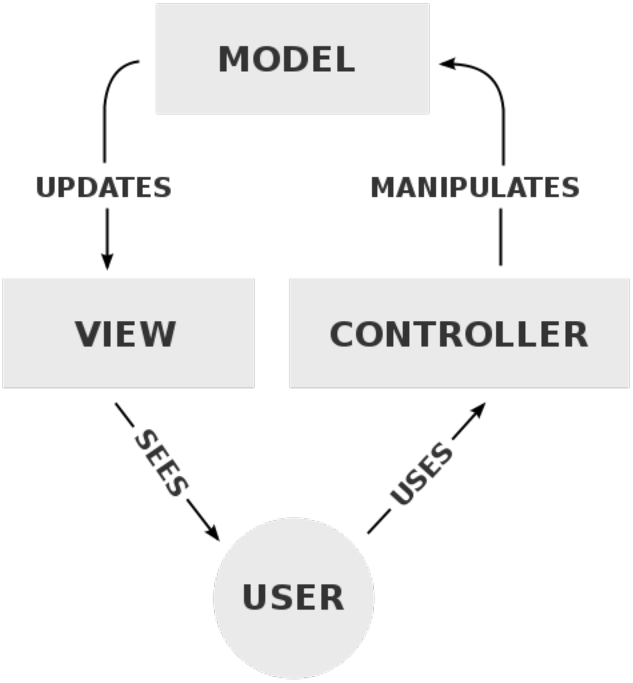


Figure 3.3 – A typical collaboration of the MVC components

The central component of MVC, the model, captures the behavior of the application in terms of its problem domain, independent of the user interface.

- The model directly manages the data, logic and rules of the application.
- A view can be any output representation of information, such as a chart or a diagram. Multiple views of the same information are possible, such as a bar chart for management and a tabular view for accountants.
- The third part, the controller, accepts input and converts it to commands for the model or view.

Interactions:

In addition to dividing the application into three kinds of components, the model–view–controller design defines the interactions between them.

- A model stores data that is retrieved according to commands from the controller and displayed in the view.
- A view generates new output to the user based on changes in the model.
- A controller can send commands to the model to update the model's state (e.g. editing a document). It can also send commands to its associated view to change the view's presentation of the model (e.g. by scrolling through a document).

Java Platform Enterprise Edition

Java Platform, Enterprise Edition or Java EE is a widely used computing platform for enterprise software. The platform provides an API and runtime environment for developing and running enterprise software, including network and web services, and other large-scale, multi-tiered, scalable, reliable, and secure network applications. Java EE extends the Java Platform, Standard Edition (Java SE), providing an API for object-relational mapping, distributed and multi-tier architectures, and web services. The platform incorporates a design based largely on modular components running on an application server. Software for Java EE is primarily developed in the Java programming language. The platform emphasizes convention over configuration and annotations for configuration. Optionally XML can be used to override annotations or to deviate from the platform defaults. Java EE is developed under the Java Community Process.

Multi layered architecture

A multi layered software architecture is a software architecture that uses many layers for allocating the different responsibilities of a software product.
In a logical multi layered architecture for an information system with an object-oriented design, the following four are the most common:
Common layers:

- Presentation layer (a.k.a. UI layer, view layer, presentation tier in multitier architecture).
- Application layer (a.k.a. service layer or GRASP Controller Layer)
- Business layer (a.k.a. business logic layer (BLL), domain layer),the part of the program that encodes the real-world business rules that determine how data can be created, displayed, stored, and changed.

- Data access layer (a.k.a. persistence layer, logging, networking, and other services which are required to support a particular business layer)

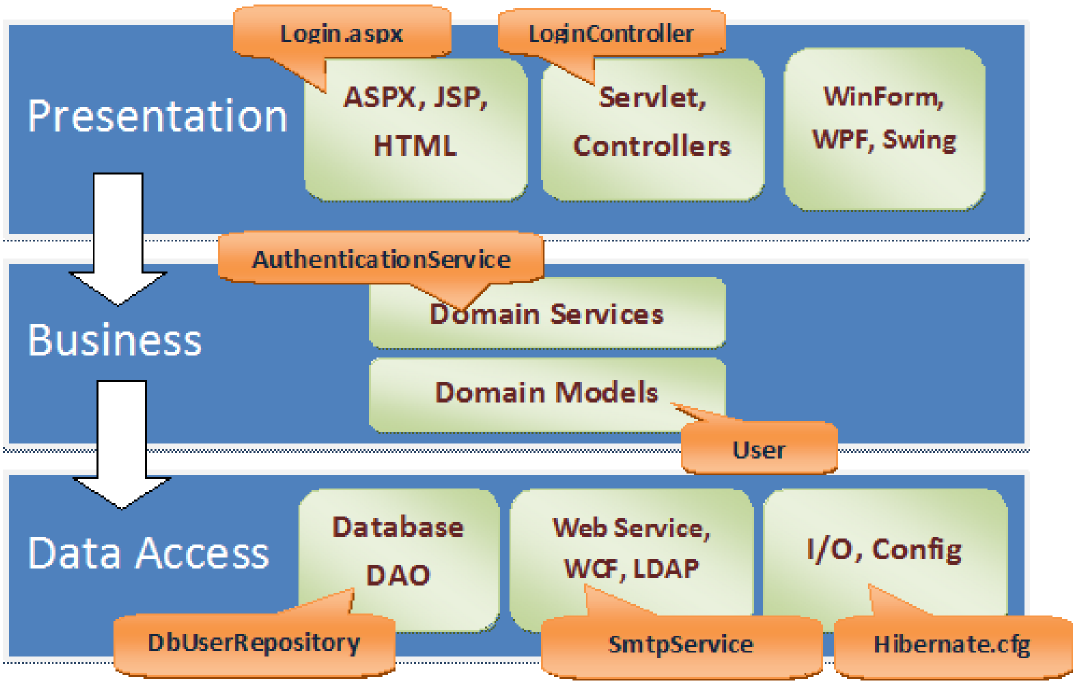


Figure 3.4 – Multi layered architecture

Servers

We use Apache Tomcat which is a project from Jakarta on a servlet container used as a formal implementation of JSP and servlet technologies Java.It has an internal HTTP server. It represents an effective means deployment of web applications.

Tomcat is a servlet, that is to say a program running on a server (Apache), which manages the other servlets to generate dynamic web pages. It can be used alone or coupled with another server. Tomcat Sun meets the specifications for servlets and their managers.
It supports as such all the classes defined by Sun in the "Java Servlet API Specifications". Apache Tomcat is completely free and usually used in coupling with an Apache server. Written in Java, it requires power to operate, the presence of a Java virtual machine, and more specifically the Sun SDK Development Kit complete. This implies that it is completely portable and can be implemented on radically different systems, such as Linux or Windows

Web MVC Frameworks

Spring MVC:
The Spring Framework is an application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE platform. Although the framework does not impose any specific programming model, it has become popular in the Java

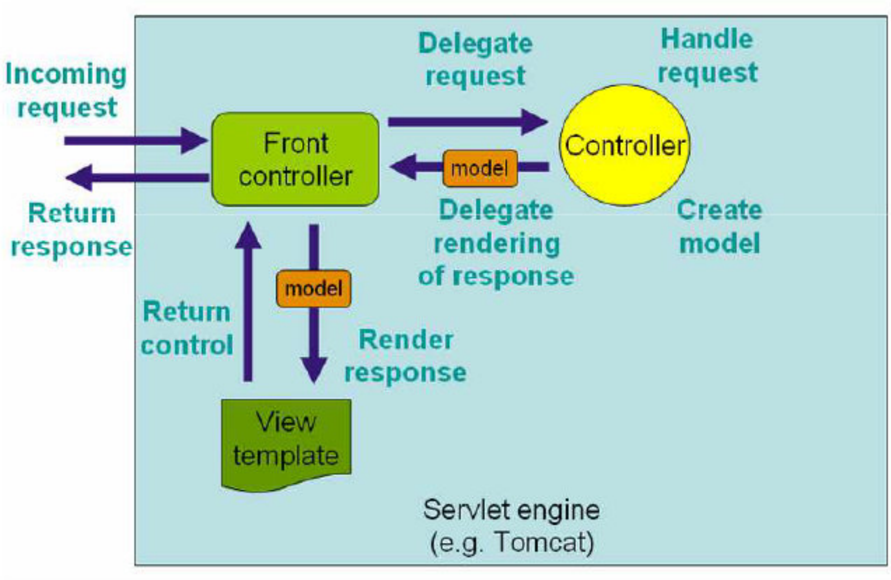


Figure 3.5 – Spring MVC architecture

community as an alternative to, replacement for, or even addition to the Enterprise JavaBeans (EJB) model. The Spring Framework is open source.

JPA:

Java Persistence API is a Java API for persistent mapping Object / Relational, it ensures a higher level of abstraction than the JDBC.

Hibernate:

Hibernate ORM (Hibernate in short) is an object-relational mapping framework for the Java language. It provides a framework for mapping an object-oriented domain model to a relational database. Hibernate solves object-relational impedance mismatch problems by replacing direct, persistent database accesses with high-level object handling functions. Hibernate is free software that is distributed under the GNU Lesser General Public License 2.1. Hibernate's primary feature is mapping from Java classes to database tables; and mapping from Java data types to SQL data types. Hibernate also provides data query and retrieval facilities. It generates SQL calls and relieves the developer from manual handling and object conversion of the result set.

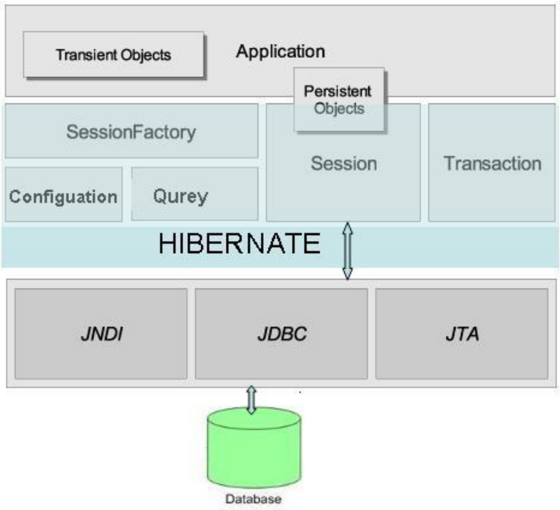


Figure 3.6 – Hibernate architecture

Spring Security:
Spring Security is a Java/Java EE framework that provides authentication, authorization and other security features for enterprise applications.
One of many advantages of Spring Security is filter chain strategies used when intercepting URL requests :

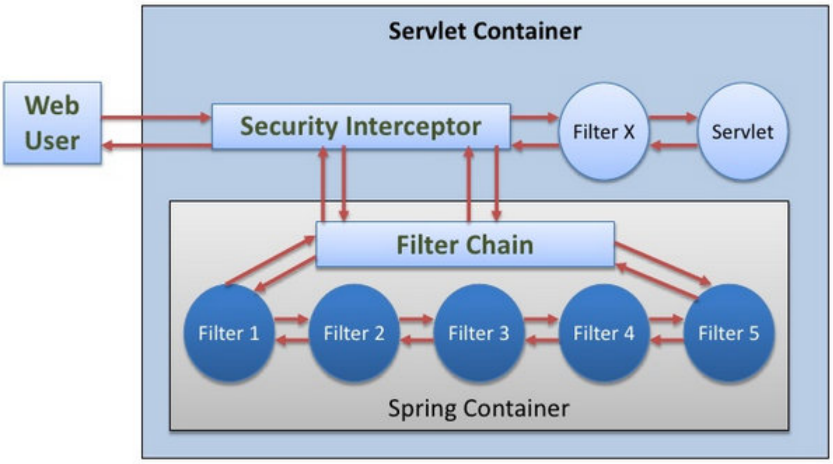


Figure 3.7 – Filter chain Spring Security

Apache maven

Maven is a build automation tool used primarily for Java projects. Maven addresses two aspects of building software: first, it describes how software is built, and second, it describes its dependencies. Contrary to preceding tools like Apache Ant, it uses conventions for the build procedure, and only exceptions need to be written down. An XML file describes the software project being built, its dependencies on other external modules and components, the build order, directories, and required plug-ins.

3.4.2 Cryptographic algorithms

At this part we describe our algorithm used to encrypt Personnal Account Number (PAN) and also Hashes algorithm used for the ADS interface.

PAN Encryption Algorithm

We encrypt PAN with AES algorithm , but the key used for encrypting is divided into two part:
-The first part of master key is obtained by decrypting the keyFile (Key.properties) with the FixedKey (InitMkeyJDBC.java)
-The second part of master key is obtained by decrypting the appropriate MasterKey in the database with the FixedKey (InitMKeyJDBC.java)
To hash PAN we use SHA-256 Algorithm Standard with Java Cryptography Library.
For masking PAN we change the ten middle numbers of the card number by asterix (*).

3.4.3 ADS Implementation

Tables created on database

Tables created on database are: Account, Card, and Mkeys tables.

First of all , we assume that the cardholder is not enrolled in 3-D Secure and he finish shopping at the E-commerce website, so we redirect him to our page in order to enroll him in 3-D Secure:

At the first form the cardholder must enter his Card Number, Expiry date and his Card Verification Value;
We have make control a client-side and server-side to protect our interface against malicious attack and hackers.

Iif somehow, the cardholder did not enter his data or has invalid data, an error is triggered and showed up on the page.

Remark:
During our internship at the host organization, we didn't have access to the real database which contains real data of cardholders; so our external supervisor advises us to work on the Test Database and to show a message of enrolled cardholder if the cardholder is enrolled before on 3-D Secure.

When we check validity of entered data we redirect the cardholder on the regitration form,The registration form contains : Fisrt name, Last name, Password(the password depends on the issuer bank),e-mail and phone number.

With the same way , we take control of the data in the registration form.

All this in wallet and out of wallet data elements are verified before sending a one time password to the cardholder, in order to complete his purchase.

If the data elements are correct we display the congratulation message to the cardholder, and he can continue his purchase as usual, we also inform him that he must validate his purchase with a one time password or with his password created before(it depends at the issuer bank.)

Conclusion

At the end of our first part,we designed and developed the ADS enrollement process and we were able to enroll cardholder after his authentication in 3-D Secure during his shopping, which allows the issuer bank and merchants to reduce fraud and adds extra security for payments. But we need also to ensure the scalability of our interface by developing the management interface at the back office, in order to give more accessebility and performance.

Chapter 4

Registration Using ACS Back Office

This chapter complete our first part of ADS interface, in order to improve online enrollment by cardholder;Our objective is to add a module for supervision and management of enrollment in ACS Back Office, and whose purpose is to manage enrolled cardholder, checking, monitoring the development of these activities, the generation of statistics and summary statements.

4.1 ACS Back Office Overview

The implementation of a 3-D Secure solution for issuing institution (Issuer Domain), amounts to implement the **Access Control Server (ACS)** for this institution, and an enrollment server.

4.1.1 Problematic and context

At the host organization the enrollment server and the ACS are combined into one server.

The ACS is a fundamental element in the transaction to the side 3-D Secure, it is configured according to an architecture that respects the universal standards of electronic banking defined by international institutions (Visa, MasterCard, ...). This architecture is composed of several modules such as **ACS-Front Office** which is responsible for receiving, processing, and manage all data about the 3-D Secure transaction and also other modules.

The Front Office contains projects to cover all treatments that ACS should perform and includes:

- Check enrollment of cardholder.
- Interract with the interoperability domain and the acquirer domain.
- Exchange with the strong authentication module.
- Secure data backup.
- Logging requests for authentication.

4.1.2 Missions and objectives

Our mission was to add a management module interface, to facilitate issuers to check enrolled cardholder, and monitoring banking activities. Our objectives is to manage cards by:

- Enroll cards
- Find cards by Bank indentification Number (BIN)
- check list cards
- Check enrollment history

Existing features at ACS Back Office

The Access Control Server Back Office is able to:

- Setup and configuration of ACS
- Manage Transactions
- Manage BIN
- Manage institutions
- Manage Fraud
- Manage SMS
- Manage Profiles
- Manage Administrators
- Traceability

4.2 Requirements Specification

This section will serve the collection of the module needs to achieve. To clarify the needs of users of our module we will present the functional requirements and non functional requirements.

Functional Requirements

Functional requirements are the services offered by our module. They capture tasks or activities that must be made by the application to different types of users. We specify cards Management (Consultation, modification): Allowing the display of the list of cards with the ability to apply filters or search on the desired field (institution, card number) and also card management gives hand the administrator to change information that exist on the credit card

Non-Functional Requirements

In this part, we discuss the non-functional requirements, this step is very important and complete functional requirements. Also, non-functional requirements often offer many constraints that may influence future choices. The incorporation of non-functional requirements such as time constraints, safety and effectiveness of the operation are extremely delicate. We are interested at identifying non-functional requirements of our module.

Indeed, we must define the main technical objectives of the future architecture, that is, to carefully consider all the forces that will be exerted on the future system as well as the feature of each of they.

The following table provides more details on the technical requirements of the future Module:

Technical requirements	**Description**
Security	The application must be highly secure, the information should not be accessible to everyone, that is to say that the module is accessible by a username and password assigned to an individual course according to rights of the user.
reusability	Include the ability to use the module in another application.
maintainability	The various features of the module must be legible and understandable to maintain and update quickly and easily.
Performance	The operating system speed should be at a higher level to ensure good quality service
Ergonomy	Provide ergonomic interface to facilitate use to users and respond as much as possible the needs of customers.
Database	Work with any DBMS.
Ease of use	The application must be simple, easy to understand by any user

Table 4.1 – Technical Requirements of the future Module

Identification of Stakeholders

An actor is a role played by an external entity that interacts directly with the system studied. The responsibilities of the actors differ depending on their position and according to the hierarchy which they are part. The main stakeholders of our module are:

- **Super Administrator:** the person performing the control, monitoring, management, banking guarantee the smooth transaction and administration module.
- **Administrator:** The person who has the right to consult only the transactions, generate reports and management institutions and Bins.

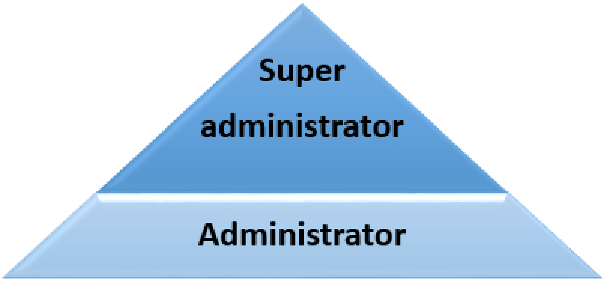


Figure 4.1 – identification of stakeholders

4.3 Modelization of Registration Module

In this section we present the needs of the registration module. That is to say using the use case diagram of the UML modeling language. The use case diagram aims to describe the behavior of the registartion Back Office module. It also helps to structure the needs of users and module objectives.

4.3.1 Use case diagram

The use case diagram below show cases, that inherit other use cases.In this diagram use cases are abstract if they do not actually exist, they are there to simplify the presentation of the chart to make it more readable, something to mark is that all use cases included the case **must authenticate**, so we can't run any use case without **authentication**

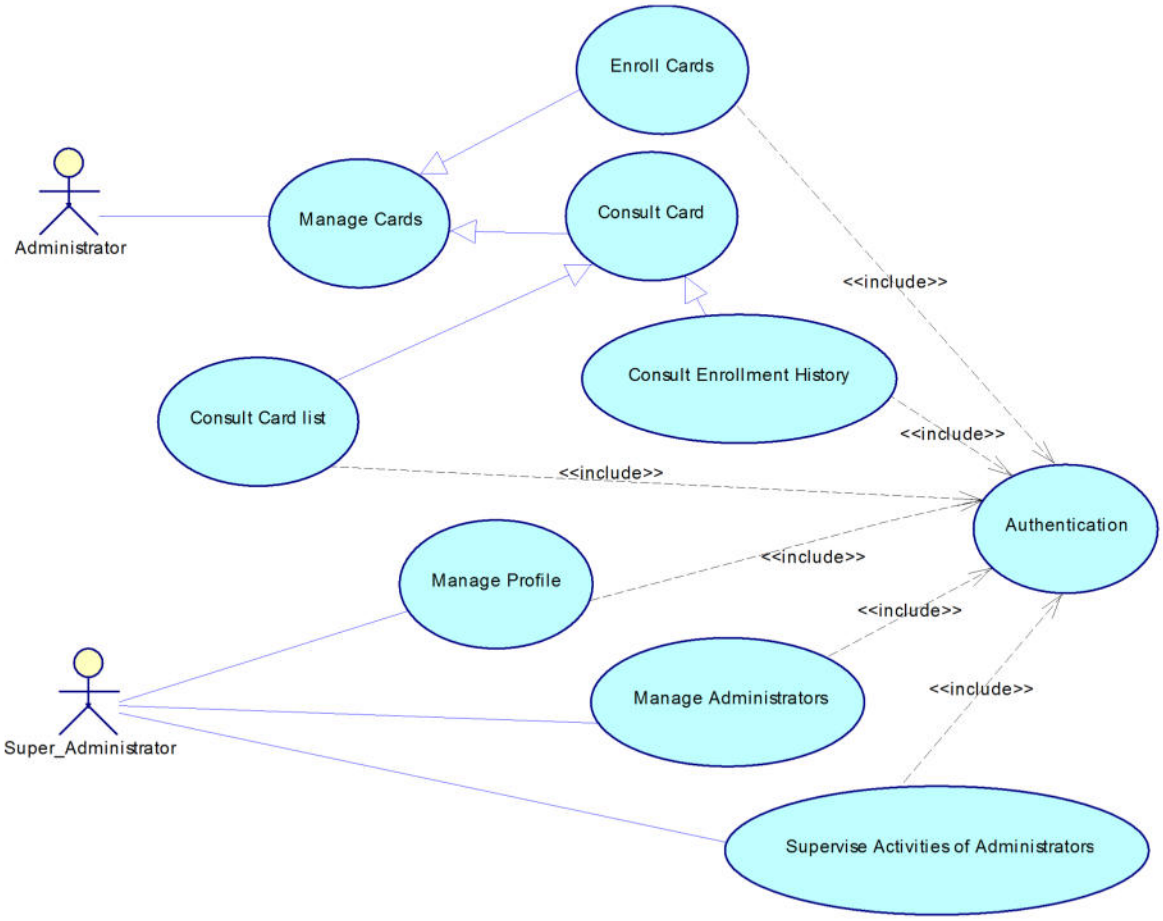


Figure 4.2 – Use case diagram

4.3.2 Sequence diagram

These sequence diagrams are called black box because we still don't know how the system will act. Here we specify the interaction between users of the system actors and the system itself. A sequence diagram "system = black box" can be enriched by internal actions (often checks) of reply to alternative links.

Sequence Diagram "Authentication"

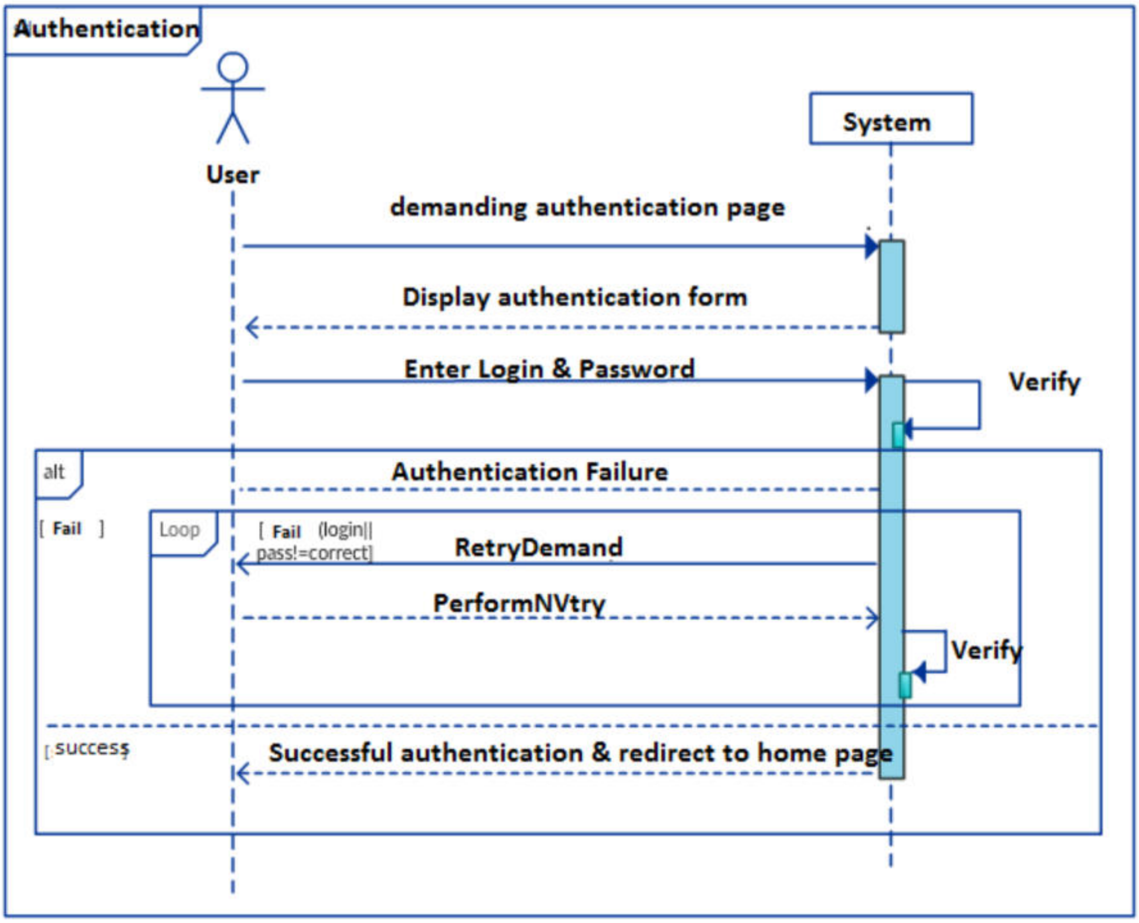

Authentication
User
System
demanding authentication page
Display authentication form
Enter Login & Password
Verify
alt
Authentication Failure
[Fail]
Loop
[Fail (login|| pass!=correct]
RetryDemand
PerformNVtry
Verify
[success
Successful authentication & redirect to home page

Figure 4.3 – Sequence diagram of authentication

Remark:
We use **Spring Security** Framework for authentication and also we have protected our ADS interface with a Login/password.

Sequence Diagram "List cards"

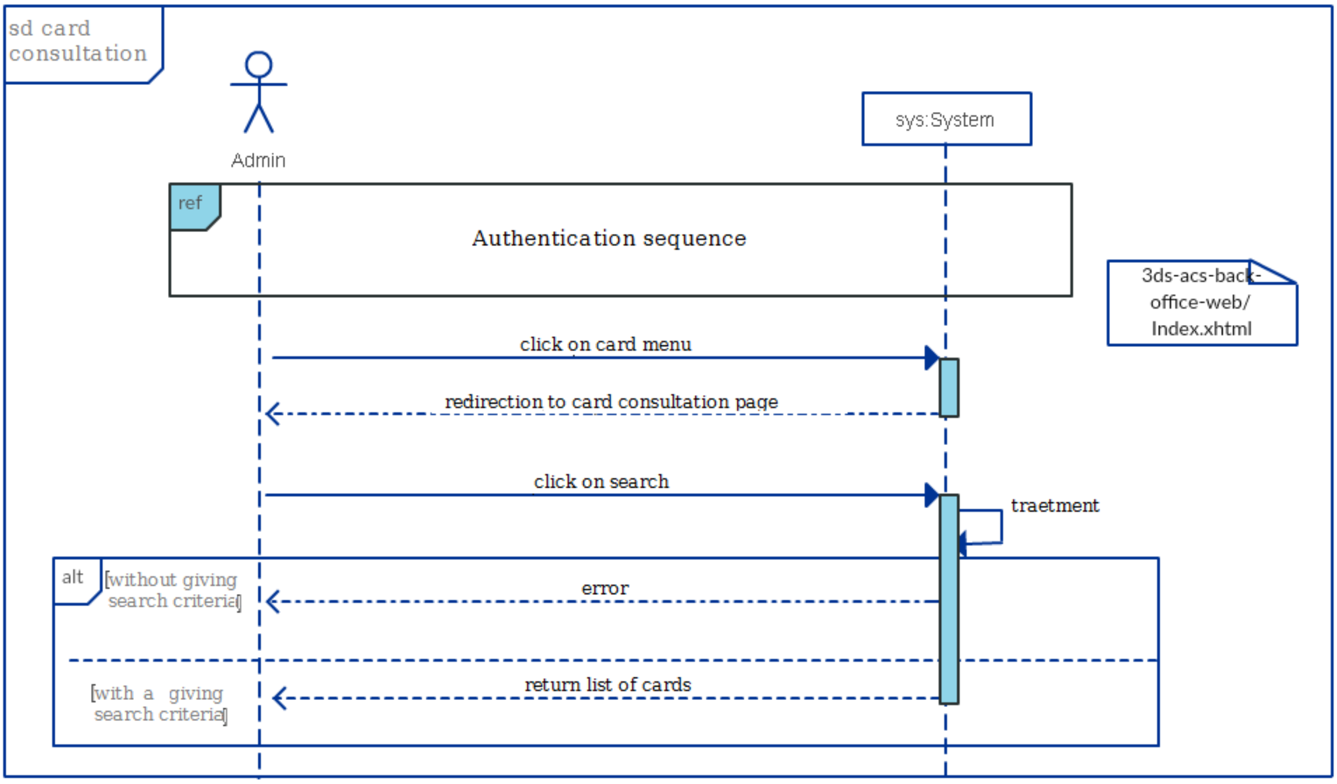

sd card consultation
Admin
sys:System
ref
Authentication sequence
3ds-acs-back-office-web/ Index.xhtml
click on card menu
redirection to card consultation page
click on search
traetment
alt
[without giving search criteria]
error
[with a giving search criteria]
return list of cards

Figure 4.4 – Sequence diagram of listing cards

Sequence Diagram "Edit cards"

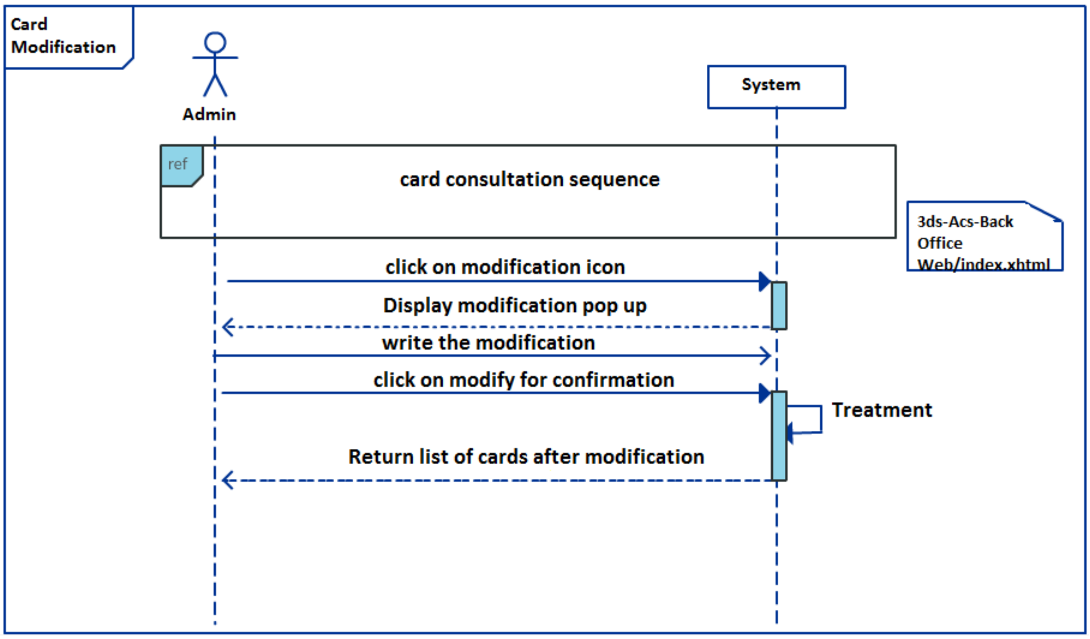

Card Modification
Admin
System
ref
card consultation sequence
3ds-Acs-Back Office Web/index.xhtml
click on modification icon
Display modification pop up
write the modification
click on modify for confirmation
Treatment
Return list of cards after modification

Figure 4.5 – Sequence diagram of editing cards

Sequence Diagram "Enroll cards"

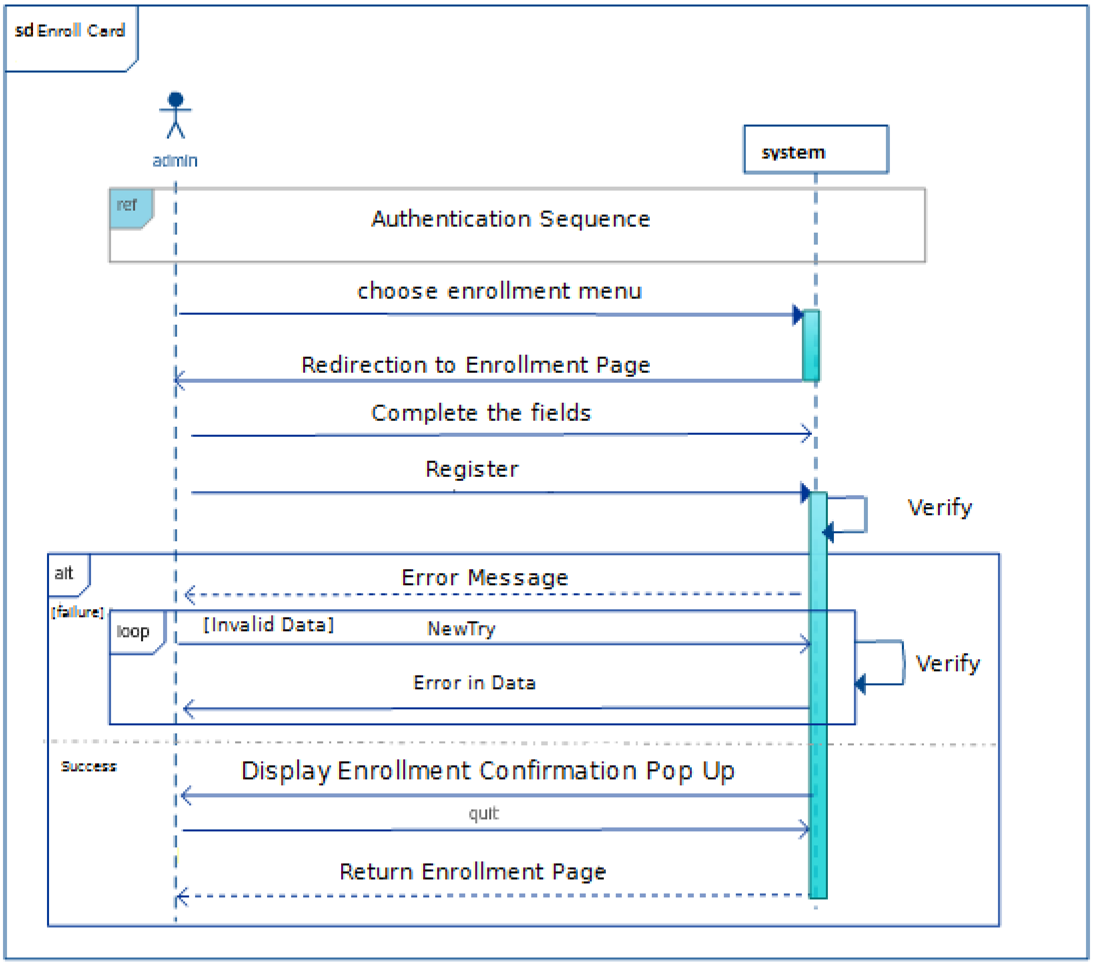


Figure 4.6 – Sequence diagram of enrolling cards

4.3.3 Activity diagram

Activity diagrams are close to the flow chart representation; the description of a use case by an activity diagram corresponds to its algorithmic translation. The activity diagram gives a vision of the sequences of activities operation or a use case. It is attached to a class category and describes the activities in this category. This activity is called "flow control". It indicates the part of each object in the performance of work. It will be enriched by the sequence of conditions.

The authentication activity diagram allows us to see the internal behavior of the system, when opening the back-office unit by the administrator, the system shows him the authentication form after login and password passes are typed, the system checks the validity of these and displays the home page otherwise it displays an error message.

```
Data: Login and password
Result: Access Session to Registration module of Back Office
Launch the Browser;
Enter the URL of Back Office module;
Display the authentication form;
Enter Login and password;
while Invalid Data do
    Enter Login and password
    if Valid Data then
        Open Session to Registration module;
    else
        Valid data and time out greater than 10 min
    end
    Log out
    Display the authentication form;
end
```

Algorithm 1: Algorithm for authentication to registration module of Back Office

4.3.4 Package diagram

Package diagram is an UML diagram that provides a graphical representation of high-level organization of the application, it identifies generalization links and dependencies between packages.

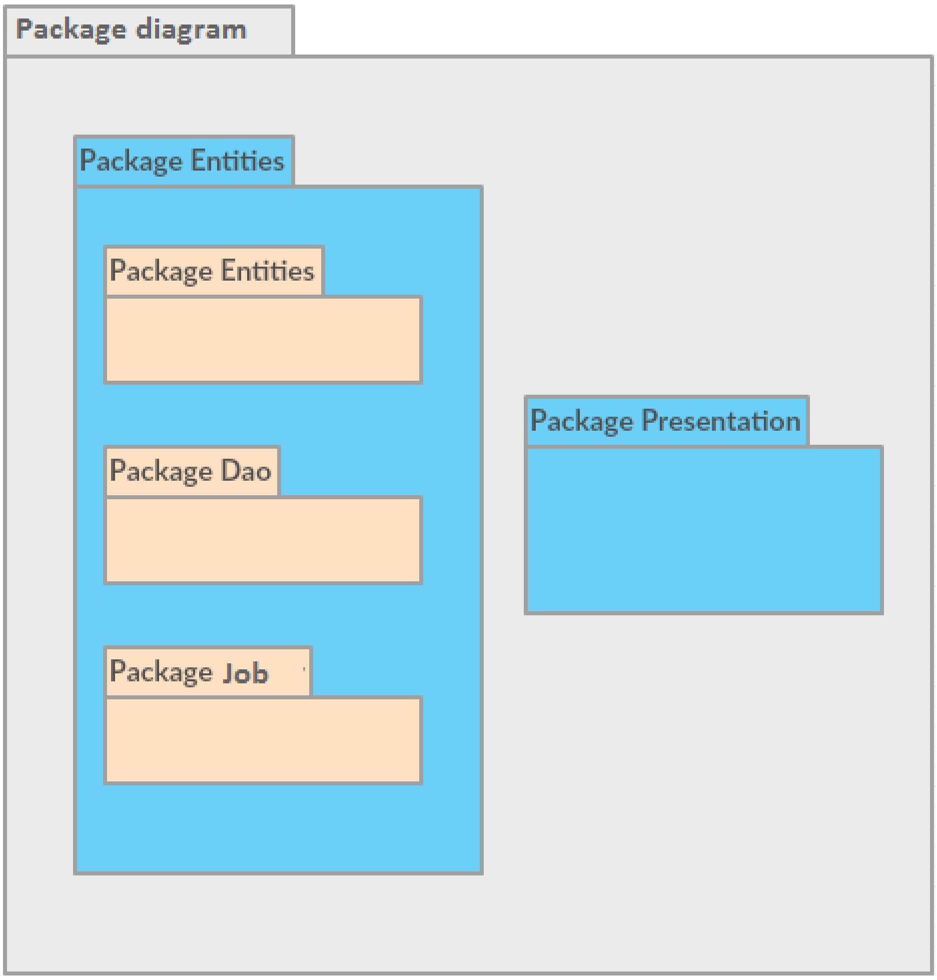


Figure 4.7 – Package diagram

— **Entities Package:** This package allows us to expose a reusable business entity.
  - Entities Package: This package contains persistent classes with the attributes, their getters and setters.
  - Dao Package: In this package will be defined for each entity class **DAO class** serving as liaison between the entity and its table in the database.
  - Business Package: This package contains the functionality of the system translated into Java methods. Those were tested to ensure their execution.

— **Presentation Package:** This package consists of the GUI application (views) and the controllers (Managed Beans).

4.4 Implementation of Registration module

After presenting the functional design part, comes the technical study and development. This section aims to provide the software architecture of the module, the various frameworks and technologies. The features of the application that have been developed, tested.

4.4.1 Tools and technical environment

The technical study aims to define the architecture of the solution as well as the relations intercomponent. With this sense, this part presents the software architecture adopted, and finally presents the main screens made with a description of their contents.

Software architecture

Before beginning on the development of any computer system, it is important to prepare the software architecture, because the software architecture expresses a fundamental structural organization schema for software systems. It is essential for an understanding, management and optimization of any system.
To structure our module, we opted for a cut in layers. this is also justified by the adopted Java Enterprise Edition architecture. In addition, a layered architecture promotes the maintainability of the application and re-adopting the solutions to problems with a similar operation.
In this section we focus on the software architecture of our module.

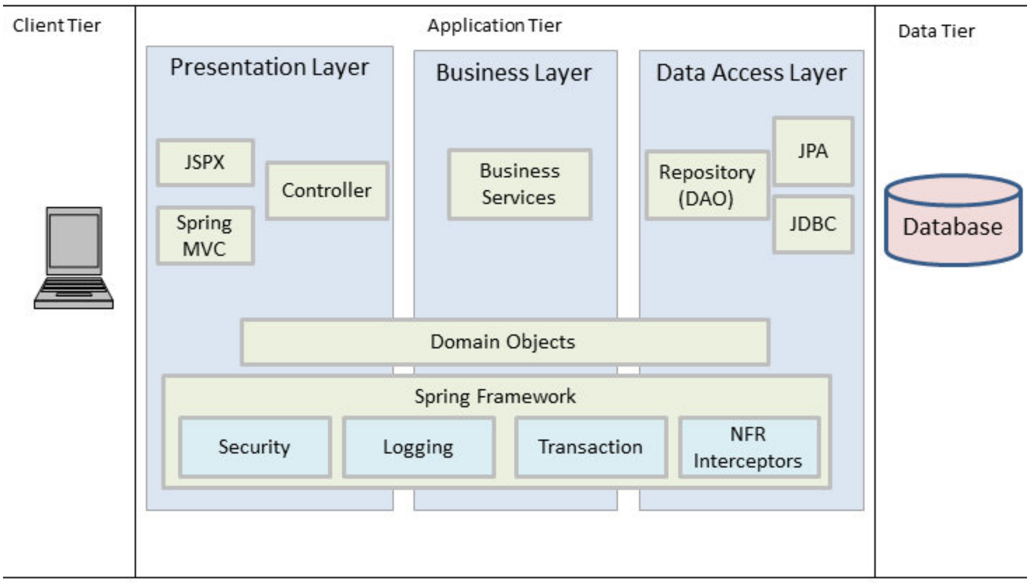


Figure 4.8 – Software architecture

- **Security layer:**The security layer is used to ensure that a user has the right to perform the operation he is doing. Here we used the Spring security framework that provides a complete security solution for Java JEE applications.
- **Web layer:**This layer corresponds to the portion of the visible and interactive application with users.Human Machine interface interaction. In most cases, it is a rich client

interface or a web interface. This layer is implemented by the Framework JSF with RichFaces library.

- **Business layer:**This layer contains the Java classes required for business logic, and therefore has the services offered by the module to the user. These services are non implementation scenarios identified during the design phase.When calling a service, it uses the business objects needed to fulfill the request of the user. This layer is performed using Spring.
- **DAO layer:**This layer is responsible for relationship management data sources. It's at this level that we find the basic functionality for creating,finding and removing business entities in respect of transactional properties. It's also in this layer that the conversion mechanisms Object / Relational can take place.

Hardware architecture

On our project, the three-tier architecture is most suitable case to our needs. It aims to model the application as a stack of three software layers whose role is clearly defined:

— Data presentation: corresponding to the display, the return on the workstation, the dialogue with the user.

— Web server: Since the data will be shared between two heterogeneous environments, the main role of the web server is to manage the communication between the client and the database server.

— The database server provides data to the web server.

In this approach, layers communicate with each other through an "exchange model", and each one provides a set of services. The services of a layer are available to the upper layer. Each layer communicates only with its immediate neighbors. The hardware architecture provided for the module is as follows:

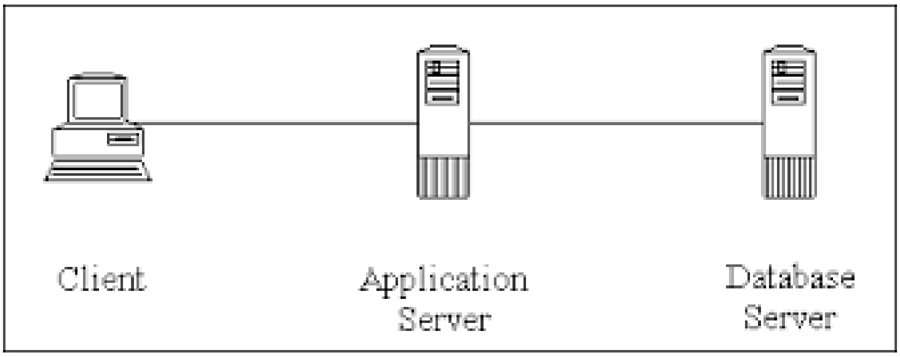


Figure 4.9 – Hardware architecture

Web Services:

A Web service is a service offered by an electronic device to another electronic device, communicating with each other via the World Wide Web. In a Web service, Web technology such as HTTP, originally designed for human-to-machine communication, is utilized for machine-to-machine communication, more specifically for transferring machine readable file formats such as XML and JSON.
Types of web services:

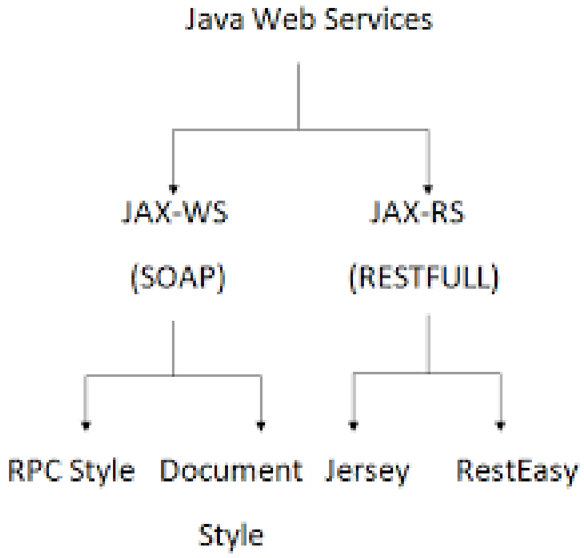


Figure 4.10 – Types of web services

On our poject we use SOAP web service because it's highly extensible and also we only use the parts that we need to implement our Application.
Also one of the most important SOAP features is built-in error handling. If there's a problem with our request, the response contains error information that we can use to fix the problem.This particular feature is extremely important; otherwise we would be left guessing as to why things didn't work. The error reporting even provides standardized codes so that it's possible to automate some error handling tasks in our code.

This figure describe the difference between SOAP and REST:

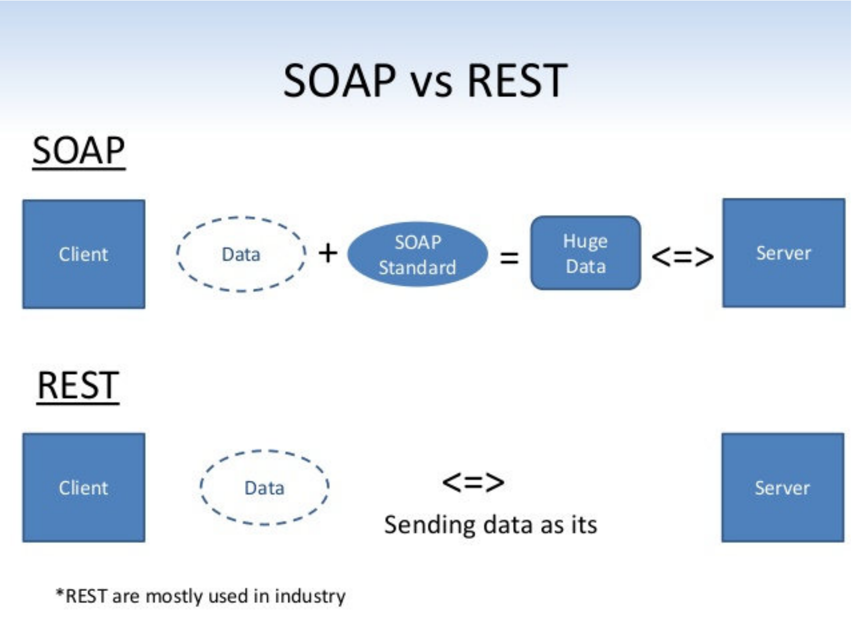


Figure 4.11 – SOAP vs REST

SOAP Security Extensions:

As a container for XML-based messages, SOAP 1.1 has responsibilities to support the use of XML-based security technologies.

Digital Credentials Extensions to SOAP

To achieve end-to-end application security (encryption, authorization, and authentication), an exchange of digital credentials is required. Digital credentials come in different forms. The most commonly used credential is a digital certificate that conforms to a standard called X.509. Microsoft has recently announced plans to base its efforts on another type of credential called Kerberos tickets. In either case, these credentials hold information about the Holder, including information about the encryption methods being used and the Holder's digital signature. Microsoft and IBM have proposed extending the SOAP 1.1 specification to include a security-specific credentials header, which would standardize the use of multiple types of credentials within a SOAP message. The motivation for the extensions is to give SOAP-based services the ability to sign portions of the SOAP envelope.

Digital Signature Extensions to SOAP

To use XML Digital Signatures (or any digital signature) effectively in SOAP messages, we need a standardized way to incorporate them into the message. To address this need, IBM and Microsoft have proposed a set of SOAP 1.1 header extensions that standardize the use of digital signatures. The goal is to enable SOAP envelopes to contain a digital signature that can be used to sign one or more elements contained within the envelope.

Tools:

Eclipse

Eclipse is an integrated development environment free scalable, universal and polyvalent, to create development projects implementing any programming language. Eclipse IDE is mainly written in Java (using the graphical library SWT, IBM), and this language using specific libraries is also used to write extensions. The specificity of Eclipse IDE is because of its architecture fully developed around the concept of plug in. Several commercial software are based on the free software, such as IBM Lotus Notes 8, IBM Symphony or WebSphere Studio Application Developer.

Oracle

Oracle Database is a database management system (DBMS) provided by Oracle Corporation. This is one of the most popular DBMS in the world. It runs on a relational system and even on a relational model object since version 8.

4.4.2 Registration module integration at Back Office

Authentication

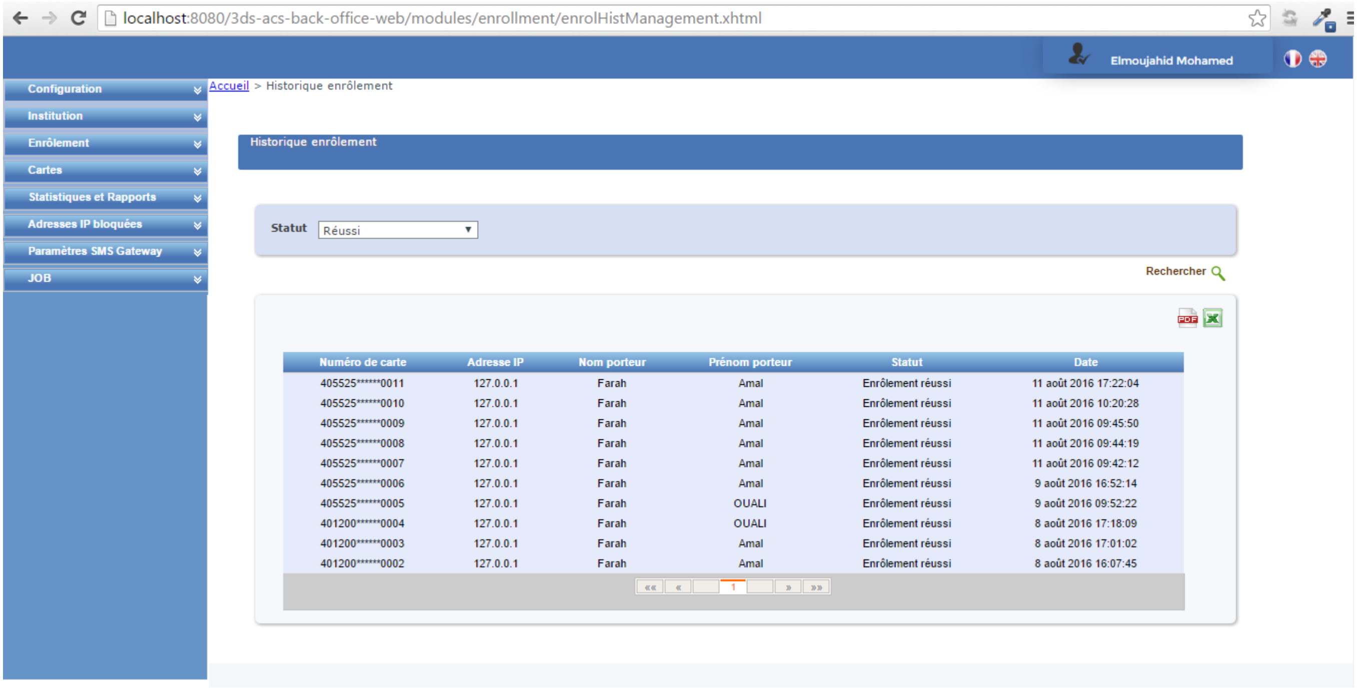
The Authentication page prompts the user to bring their username and password. Each user will have a different page depending on the roles assigned to their account.

Bank Identification Number

The page of Bank Identification Number allows the consultation of the list of Bins,we just have to select the institution (mandatory fields) and we click on search.

Enrollment

The page of enrollment allows an administrator to enroll a card so that it can be integrated to 3-D Secure Service.The administrator enter FirstName,LastName,PAN,Expiry Date,CVV on the fields in order to enroll cardholder.



Statut: Réussi

Numéro de carte	Adresse IP	Nom porteur	Prénom porteur	Statut	Date
405525******0011	127.0.0.1	Farah	Amal	Enrôlement réussi	11 août 2016 17:22:04
405525******0010	127.0.0.1	Farah	Amal	Enrôlement réussi	11 août 2016 10:20:28
405525******0009	127.0.0.1	Farah	Amal	Enrôlement réussi	11 août 2016 09:45:50
405525******0008	127.0.0.1	Farah	Amal	Enrôlement réussi	11 août 2016 09:44:19
405525******0007	127.0.0.1	Farah	Amal	Enrôlement réussi	11 août 2016 09:42:12
405525******0006	127.0.0.1	Farah	Amal	Enrôlement réussi	9 août 2016 16:52:14
405525******0005	127.0.0.1	Farah	OUALI	Enrôlement réussi	9 août 2016 09:52:22
401200******0004	127.0.0.1	Farah	OUALI	Enrôlement réussi	8 août 2016 17:18:09
401200******0003	127.0.0.1	Farah	Amal	Enrôlement réussi	8 août 2016 17:01:02
401200******0002	127.0.0.1	Farah	Amal	Enrôlement réussi	8 août 2016 16:07:45

Figure 4.12 – Module of back office

Cards

The screen of Cards lets us view the list of cards, we select the institution and then we click on search. We can customize the search by entering the card number. The PAN display depends on the user's authority defines "user admin", if the user has the right to see the PAN, it appears clear else, the PAN mask appears.

Enrollment via Batch

Enrollment process can be applied via a batch file, which means that the issuer bank send to the administrator of ACS a batch file (contains cardholders data), this file is executed and therefore all the cardholders are enrolled.

Enrollment history

We can also display hitory of enrollment in order to check whether the enrollment process has been succeed or not.

4.5 Conclusion

This chapter has been devoted to implementation of enrollment process at the Back Office of ACS.We have justified the choice of architecture Java Enterprise Edition 3-tier, we described the architecture in software layers and hardware architecture of the module. We also mentioned the various tools with which we implemented the application. A functional study, with all its elements, technical design and module design allow to define the project and to have an overall idea on its structure, which can facilitate its development.

Conclusion

This report is the result of six months of work and study, a period during which we realized a project that assume an important role generally in the electronic bunking field and especially for S2M company and Outsourcing services in S2M Transactions.

We gave a global overview about 3-D Secure protocol and its three domains, the architecture, and the various entities and messages then we did the necessary studies about specifications and standards to be respected in order to build the enrollment module with the correct way and make it functional and secure and finally we performed the tests with the supervision of the team of Edition and Integration departments within S2M, then we realized that the project is ready to be integrated and exploited in the outsourcing service that gives the opportunity to the clients to use its functionalities.

Indeed we learn new concepts during this experience whether about electronic commerce, S2M Company business, the professional way to manage and direct projects, engineering qualities, team spirit and collaboration, the efficiency when we are facing security issues and trying to make approaches between the business process and security measures or about tools, technologies, frameworks and all the technical and professional skills.

Finally we achieved the objectives of our internship after providing all the efforts to shape what we have already learned during our formation within the faculty of sciences in Rabat and make it real within the company.

In perspective we envision the study and development of other features to enrich 3DS software include: The traceability of actions performed by administrators, and emails management as well as we gave another proposal consisting on conforming all the modules of 3DS project including the enrollment process to the mobile.

Bibliography

[1] Deepak Alur, Dan Malks, John Crupi, Grady Booch, and Martin Fowler. *Core J2EE Patterns (Core Design Series): Best Practices and Design Strategies*. Sun Microsystems, Inc., 2013.

[2] John Arthur and Shiva Azadegan. Spring framework for rapid open source j2ee web application development: a case study. In *Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Network*, pages 90–95. IEEE, 2008.

[3] Christian Bauer and Gavin King. Hibernate in action,manning greenwich ct. 2015.

[4] D Elliott Bell and Leonard J La Padula. Secure computer system: Unified exposition and multics interpretation. Technical report, DTIC Document, 1976.

[5] Stephanie Bodoff. *The J2EE tutorial*. Addison-Wesley Professional, 2012.

[6] Paolo Giorgini, Fabio Massacci, and John Mylopoulos. Requirement engineering meets security: A case study on modelling secure electronic transactions by visa and mastercard. In *International Conference on Conceptual Modeling*, pages 263–276. Springer, 2013.

[7] Pita Jarupunphol and Chris J Mitchell. Measuring 3-d secure and 3d set against e-commerce end-user requirements. In *Proceedings of the 8th Collaborative Electronic Commerce Technology and Research Conference*, pages 51–64. Citeseer, 2003.

[8] Rod Johnson. J2ee development frameworks. *Computer*, 38(1):107–110, 2009.

[9] WANG Mei-qin. Research of the integration of ssh lightweight j2ee-based framework [j]. *Computer Knowledge and Technology*, 21:056, 2009.

[10] Steven J Murdoch and Ross Anderson. Verified by visa and mastercard securecode: or, how not to design authentication. In *International Conference on Financial Cryptography and Data Security*, pages 336–342. Springer, 2010.

[11] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Analysis of recommendation algorithms for e-commerce. In *Proceedings of the 2nd ACM conference on Electronic commerce*, pages 158–167. ACM, 2000.

[12] Inderjeet Singh and Mark Johnson. *Designing enterprise applications with the J2EE platform*. Addison-Wesley Professional, 2012.

[13] Efraim Turban, David King, Jae Lee, and Dennis Viehland. Electronic commerce: A managerial perspective 2002. *Prentice Hall: ISBN 0*, 13(975285):4, 2002.

[14] VISA. 3-d secure: Protocol specification – core functions (confidential only for visa clients). 2007.

[15] VISA. Verified by visa issuer implementation guide (confidential only for visa clients). 2010.

Appendix A

Benchmarking analysis between 3-D Secure and SET protocol

A.1 Introduction

The threat of credit card fraud is arguably the most serious issue of concern to e-commerce participants, including consumers and merchants. SSL/TLS and SET are two widely discussed means of securing online credit card payments. Because of implementation issues, SET has not really been adopted by e-commerce participants, whereas, despite the fact that it does not address all security issues, SSL/TLS is commonly used for Internet e-commerce security. The three-domain (3D) security schemes, including 3-D Secure and 3D SET, have recently been proposed as ways of improving ecommerce transaction security. These schemes can provide the participants in an e-commerce transaction with a greater range of security services than provided by naive use of SSL/TLS, without introducing significant implementation complexity for merchant or consumer. However, in ecommerce, not only security requirements but also implementation requirements must be considered.

A.2 ADVANTAGES AND DISADVANTAGES OF SSL/TLS AND SET

Before considering the 3-D Secure protocol, we start by considering the main advantages and disadvantages of the SET and SSL/TLS protocols when used for e-commerce security.

A.2.1 Brief analysis of SSL/TLS

The main advantages of SSL/TLS, when used to protect e-commerce transactions, are as follows.

- Ease of use for e-commerce end-users. The cardholder can use SSL/TLS completely transparently because it is already built into commonly used web browsers, and merchants can also implement SSL/TLS without changing their payment model in any way.
- The system is not complex, resulting in minimal impact on transaction speed.

The main disadvantages of SSL/TLS for e-commerce are as follows.

- The merchant cannot reliably identify the cardholder. In cases where consumers use a stolen credit card to initiate e-commerce transactions, merchants are responsible for 'card not present' transaction charge backs.
- Since SSL/TLS only protects the communications link between consumer and merchant, it does nothing to protect sensitive cardholder information whilst it is stored at the merchant server. Merchants therefore need to implement additional security measures to protect the secrecy of this information.
- SSL-based e-commerce permits the merchant to see consumer payment information,potentially causing security concerns to cardholders.

A.2.2 Brief analysis of SET

The main advantages of SET are as follows.

- SET ensures the confidentiality of payment information at all stages of transaction processing, including data transmission and data storage.
- SET prevents the merchant from seeing consumer payment information, since the payment information is forwarded to the acquirer in encrypted form (encrypted using the acquirer' s public key).
- To ensure merchant privacy, SET prevents the acquirer from seeing consumer order information stored at the merchant web server.

The main disadvantages of SET are as follows.

- Implementing SET is more costly than SSL/TLS for both consumers and merchants.
- Using SET is much more complicated than using SSL.
- SET does not permit the cardholder to place an order from PCs other than the cardholder's SET-initialised PC because the cardholder' s private key required to conduct a SET tra nsaction is stored in this PC.
- SET employs complex cryptographic mechanisms that may result in an unacceptable transaction speed.

A.3 3-D SECURE AND SET

We now give an overview of the 3-D Secure and SET payment systems.

A.3.1 3-D Secure

The 3-D Secure payment system can be regarded as the integration of SSL with the 3D model. When used simply to protect the cardholder-merchant link, SSL/TLS does not provide verification of the cardholder, which can result in credit card fraud at the consumer side. Integration of the 3D architecture with SSL can help address this issue. 3-D Secure, originally knownas 3-D SSL, was developed by Visa. In 3-D Secure, the payment gateway, which provides an interface between the merchant/acquirer' s payment system and the Visa proprietary payment network VisaNet, must be implemented in the acquirer domain. Merchants are responsible for installing an SSL/TLS Merchant Plug-In (MPI) at their servers, as would normally be the case if they wish to implement SSL/TLS for consumer-merchant communication protection. For 3-D Secure, this MPI is required to have additional functions to handle communication with a centralized Visa directory.

A.3.2 SET

SET (Server-based SET) is another 3-D payment scheme that predates 3-D Secure. SET was developed by a number of SET software vendors system into the 3D model. As a replacement for the traditional SET digital wallet that must be stored at a consumer PC, SET uses a SET Wallet Server in the issuer domain. The cardholder' s certificate is also securely stored at the issuer' s secure server. Within the acquirer domain, there is no need for the merchant to have a certificate installed at the merchant server. As in the issuer domain, the acquirer stores the merchant' s certificate and implements the payment gateway at the acquirer secure server. Figure 2 shows how 3D SET operates. The 3D SET transaction procedure involves the following main steps, as indicated in Figure 1. In this explanation, and as previously, C, M, I and A denote the Cardholder, Merchant, Issuer and Acquirer respectively.

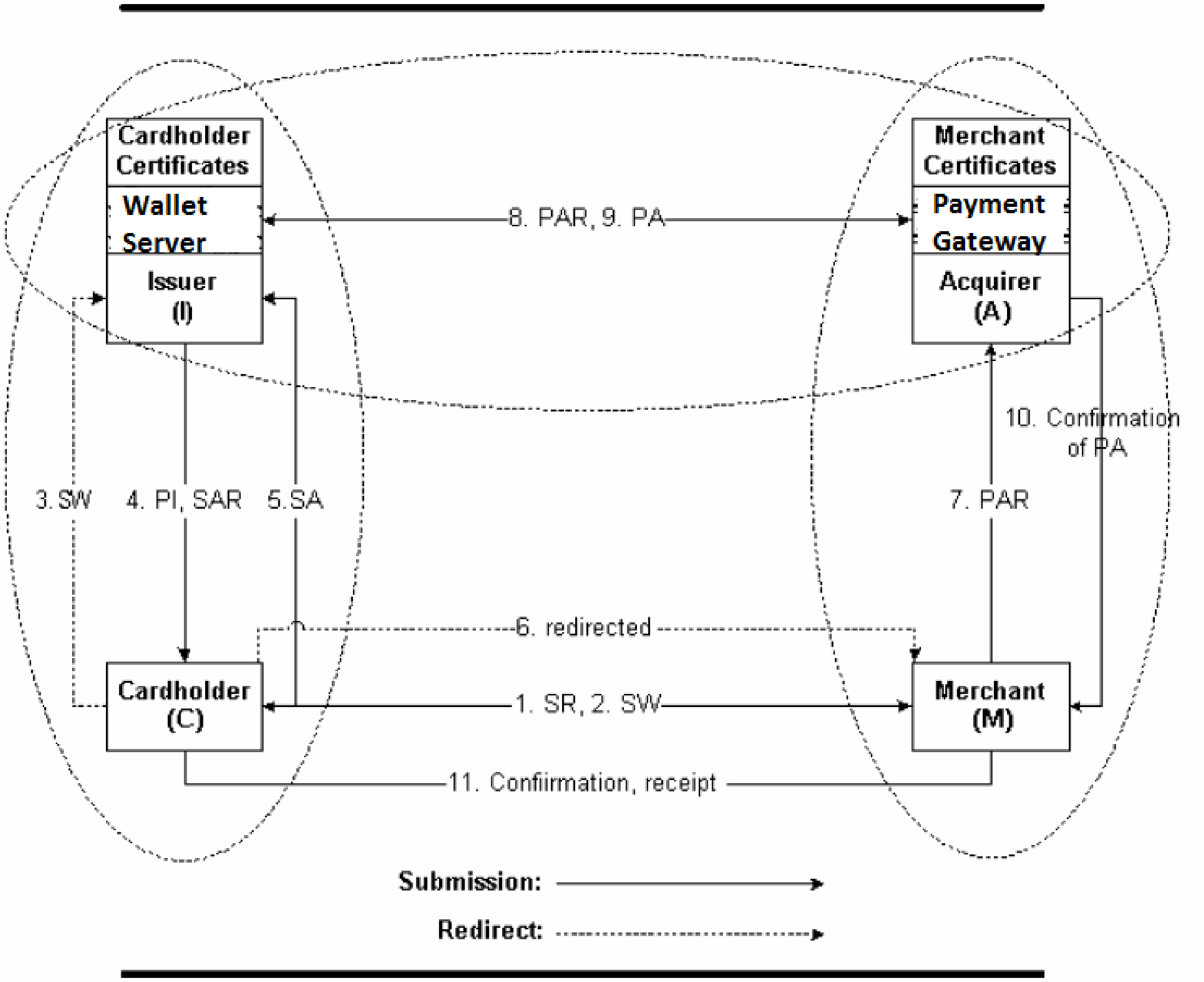


Figure A.1 – The SET Transaction procedure

C → **M** The cardholder passes a SET request (SR) message to the merchant.
M:C The merchant sends a SET Wake-up (SW) message to the cardholder.
C → **I** The cardholder's browser redirects the (SW) message to the SET Wallet Server at the issuer
I → **C** The issuer displays a window to the cardholder containing the payment information (PI), and requests entry of secret authentication information. **C** → **I** The cardholder enters the secret authentication (SA) information.If the verification is successfuk, the issuer will perform the SET transaction.
C → **M** The cardholder browser is redirected back to the merchant after the cardholder authentication process is completed.
M → **A** The merchantsends the payment authorization request (PAR) to the acquirer.
A → **I** The acquirer forwards the PAR to the issuer. **I** → **A** The issuer responds with a payment authorization (PA) to the acquirer .
A → **M** The acquirer confirms authorzation of the transaction to the merchant.
M → **C** The merchant confirms the transaction and issues a receipt to the cardholder.

A.4 3-D SECURE AND SET AGAINST THE END-USER REQUIREMENTS

A.4.1 Security requirements

Confidentiality :Both 3-D Secure and SET provide for the encryption of payment information using state of the art cryptographic techniques. Note, however, that in 3-D Secure the merchant has access to all the consumer's payment information, just as would be the case in today's typical environment, where SSL/TLS is used to protect the customer-merchant Internet link.

Integrity :Although traditional SSL alone cannot provide payment integrity for stored data, 3-D Secure can address this problem, since the payment information (PI) must be authorised and signed by the issuer prior to passing to the merchant. In **SET**, the integrity provisions supported by SET still apply, although the process is performed via the interoperability domain where the issuer holds the cardholder's certificate and the acquirer holds the merchant' s certificate.

End entity verification: Both 3-D Secure and SET provide a measure of mutual authentication between merchant and consumer. In 3-D Secure, authentication of merchant to cardholder is supported by the use of SSL, whereas authentication of cardholder to merchant is performed indirectly through the use of the ACS (that is, the ACS vouches to the merchant that it has authenticated the cardholder). In **SET**, the issuer authenticates the cardholder, and the acquirer is responsible for ensuring that it is communicating with the correct merchant, and hence mutual authentication is therefore performed via the interoperability domain.

Non-repudiation is provided because of the end entity verification mechanisms provided by both 3-D Secure and SET, consumers and merchants cannot deny having participated in a completed transaction. The consumer cannot deny ordering products or services from the merchant, and the merchant also cannot deny having received the consumer order. Hence both 3-D Secure and SET effectively meet consumer and merchant security requirements.

A.4.2 Implementation requirements

Usability 3-D Secure has the major advantage for the merchant that it preserves the payment model used for existing SSL/TLS-protected e-commerce transactions. Initialisation is also simple for both merchant and cardholder; the merchant simply needs to install a special plug-in on his/her server, and the cardholder needs no special software and must simply follow an on-line enrolment process with the card issuer, using a 'standard' web browser. SET is also simple to initialise, since the cardholder does not need to generate his/her own key pair and obtain a certificate – all this is taken care of by the card issuer. Similarly, the merchant initialisation is also simple, since the acquirer takes care of the key management and certification for the merchant. However, unlike 3-D Secure, the payment model for SET is now different to the current mode of operation, and more significant changes will be necessary to the payment application running on the merchant server.

Flexibility Both 3-D Secure and SET have the desirable property that they can be used from any PC, as is currently the case for e-commerce transactions relying simply on SSL/TLS for cardholdermerchant communications security. This is achieved since neither of these 3D schemes require special software or keying material to be installed on the e-commerce PC.

Affordability In 3D SET, merchants are still required to have a point-of-sale (POS) application to send a SET Wake-up message in reply to a SET request message from the cardholder. In addition, the POS application is also used in order to communicate with the payment gateway installed at the acquirer' s server. Although it is not clear whether consumers need to pay for their 3D SET certificate, we assume here that there is more investment in using 3D SET than using 3-D Secure because of the requirement for the POS application at the merchant web server.

Reliability The 3-D Secure Merchant Plug-In and the merchant software for 3D SET must also perform their functions correctly. Although implementation failures of these 3D protocols have not yet been reported, this is not surprising because of their recent emergence. Nevertheless, it can reasonably be assumed that the likelihood of system failures is low, since these two 3D protocols are supported by large reputable organisations. Of course, whilst the presence of incorrect functionality in security critical elements of implementations of the 3D schemes is unlikely, there is still a significant possibility that accidental vulnerabilities will be present in implementations of the schemes. Past experience indicates that it is very difficult to produce software which does not possess vulnerabilities (e.g. arising through buffer overflows) exploitable by malicious software.

Availability One of the major issues with SET is the problem of availability. Consumers can perform all the work of installing SET on their PC, but they cannot use it unless merchants also install SET at their servers. Consumers will certainly not wish to go to the trouble of performing the installation unless they are convinced that SET will be of immediate practical benefit to them. In exactly the same way, merchants will not wish to invest in a costly SET implementation unless they are convinced that a significant number of consumers will have the necessary SET installation to use their SET transaction service.

This issue is to a large extent avoided by 3-D Secure. Of course, as with any such system, 3-D Secure requires card issuers and acquirers to implement the system before anyone else – however, there are a relatively small number of such entities. Once the acquirer and issuer support is in place, merchants can install 3-D Secure in the knowledge that consumers will be immediately capable of using the system, since consumers do not need to install any new software on their PC (they simply need to carry out a simple registration process which can be totally web based). Equally, consumers will be relatively happy to perform a simple web registration process, since the time required will be minimal, and there will be no software to install or letters to write. Thus availability should not be an issue for 3-D Secure.

Similar arguments apply to SET. **Customers can be enrolled using a simple process**, and it will be much simpler to convince merchants that the (smaller) investment necessary to use 3D SET will have a speedy return. However, it is also true that, as discussed under 'Usability' above, since Merchants will have to adopt a somewhat different payment model to use 3D SET, there are greater availability issues with this scheme than with 3-D Secure.

Speed of transaction 3-D Secure primarily employs SSL/TLS to meet security requirements. Apart from this, in 3-D Secure there are other features that may affect the transaction performance, including using the Visa Directory and the Issuer ACS to verify the cardholder's identity. By contrast,SET uses complex cryptographic mechanisms to secure entire e-commerce transactions, e.g. certification among participants, protection mechanisms for consumer and merchant sensitive information, etc. It is difficult to decide which scheme is more effective with regard to transaction speed, for the following reasons.

— It is possible for Issuer and Acquirer servers to perform SET operations very quickly, as long as appropriate hardware and software are used.
— In both schemes the central servers may prove to be a bottleneck.
— Apart from software/hardware requirements, high-speed networking is required to enable the various necessary interactions to be performed quickly.

Interoperability How well 3-D Secure andSET meet the interoperability requirement remains unproven, since the two systems have not yet been widely deployed. However, since neither system relies on special software being installed on the consumer PC, and instead makes use of 'standard' browser features, **interoperability issues are less likely to arise**.

In **3-D Secure**, the only remaining problems would appear to be merchant – Visa Directory interactions. This link is protected using 'standard' means (i.e. SSL/TLS), and also there is only one Visa Directory – thus again interoperability should not be a major problem.

In **SET**, interoperability between merchant server and acquirer server should not be an issue, since we assume that the merchant software is supplied by the acquirer. This only leaves interactions between issuer and acquirer servers. Whilst interoperability problems could arise here if cryptographic and other SET functionality is provided by different vendors, the numbers of parties involved should be sufficiently small that such problems can be overcome quickly. In summary, both the 3D schemes would appear to have fewer potential interoperability problems than SET. However, 3-D Secure would appear to offer a slight advantage over SET, given that the **complex cryptographic functionality** in SET is likely to be one possible cause of interoperability issues.

A.5 3-D SECURE VS SET

We now summarise and compare how well the two schemes meet the identified end-user requirements. This table gives a comparison between the two protocols with respect to e-commerce end-user requirements.

Requirements	**E-commerce end-users**				**Effectiveness against end-users**	**Comments**
	Consumer		**Merchant**			
Security	**SET**	**3-D Secure**	**SET**	**3-D Secure**		
Confidentiality	Yes	Yes	Yes	Yes	3D SET (marginally)	The merchant has access to all the consumer's paymentinformation
Integrity	Yes	Yes	Yes	Yes	Equally effective	Both SET and 3-D Secure meet the requirements
Verification	Yes	Yes	Yes	Yes	Equally effective	Both,SET and 3-D Secure meet the requirements
Non-repudiation	Yes	Yes	Yes	Yes	Equally effective	Both,SET and 3-D Secure meet the requirements
Implementation	**SET**	**3-D Secure**	**SET**	**3-D Secure**		
Usability	Yes	Yes	Yes	Yes	3-D Secure	The payment model for SET is different to the current mode of operation
Flexibility	Yes	Yes	Yes	Yes	Equally effective	Both SET and 3-D Secure meet the requirements
Affordability	Yes	Yes	Yes	Yes	3-D Secure	More investment in using SET than using 3-D Secure
Reliability	Yes	Yes	Yes	Yes	Equally effective	Both SET and 3-D Secure meet the requirements
Availability	Yes	Yes	Yes	Yes	3-D Secure	Usability issues
Speed of transaction	N/A	N/A	N/A	N/A	Unclear	Appropriate hardware and software, highspeed networking
Interoperability	Yes	Yes	Yes	Yes	3-D Secure (marginally)	Cryptographic and other SET functionality provided bydifferent vendors

Table A.1 – 3-D Secure and SET versus e-commerce consumer requirements

Appendix B

Comparative study between multiple frameworks with J2EE

B.1 Introduction

With the large number of Frameworks for the development of the presentation layer with Java / J2EE, we must choose a solution based on the main features and needs and resources (quality, interoperablility, **security**, scalablility models and the production time of the graphic and presentation of attractive interfaces). At this section we have done a comparative study between multiple frameworks with J2EE in order to choose the convenient framework.

At first, we conducted a literature study to provide answers to a number of questions identified in cooperation with our key stakeholder. The primary contributions of this study are manifold. At first, we explained frameworks according to a meta-model, which shows an abstract overview of the common aspects between different frameworks. Secondly, we classified frameworks into three categories showing that J2EE is a middleware infrastructure framework. Then the different J2EE frameworks itself were categorized and we concluded this appendix interested in J2EE meta-frameworks. These frameworks address development needs in all tiers of the J2EE reference architecture. Our literature study ended with an overview of software quality analysis methods, which provided the foundation for our approach.

Frameworks come in many different forms. The participants in a framework fulfil a particular role. In J2EE terms an example of such a role could be the data access technology. These roles often share a relation with another role. One could imagine that a transaction model is used, together with the data access component to accomplish writing data in an atomic fashion. A role in a framework is described in terms of an interface that the participant must support. The interface consists of a set of attributes and method signatures which participants fulfilling that role must implement. Data access methods for example are mostly described by methods that are able to write or read data to and from the database.

B.2 Frameworks strength and weakness

Framework	Advantage	Disadvantage
JSF	Architecture structured around the components. Clear separation of the business layer. Supports HTML5.Simplification of the configuration file.structured framework.	Framework relatively recent .little rich documentation. server-side validation only.
Struts 2	The use of taglibs..The code is simpler structured Framework.	Changing quickly for version. Complex for simple applications
Spring MVC	Mature Framework, flexible. The dependency injection.	Complex for simple applications. No built-in support for Ajax
Wicket	Separation of HTML and Java code. Separation between the client side and the server side. No configuration file Easy hand on with AJAX exclusively uses the XHTML pages. Explicit errors	Storing information in session Manual integration with JavaScript is not obvious.
GWT	Using Java APIs (server side and client) technical JUnit Integration. different code for each browser	Very long to learn and take in hand Depending only on Google Adapted for Java developers, not the Javascript / HTML developers. AJAX security issues
Tapestry	Fast development complete separation of HTML presentation with Java code	Immaturity (released in December 2008).

Table B.1 – Advantages and disadvantages of frameworks

B.3 Frameworks comparative analysis

Having reviewed the characteristics, advantages and disadvantages of each Framework. It would be wise to establish a comparison that summarize the strengths and weaknesses of each framework and to facilitate the task of choosing the most advantageous Framework with the least possible restrictions.

Framework	Creation Date	Maturity	MVC	Components or query	Template	Compiler from java to javascript	Ease of handling	Ajax
Java Server Faces	2004	Mature	MVC2	Components	Facelets JSP pages	No	Easy	Yes
Struts	2000	Mature	MVC2	Query	JSP pages	No	No, Servlets technologie to master	Yes
Spring MVC	2002	Mature	MVC2	query	JSP pages	No	No, Servlets technologie to master	Yes
Wicket	2004	Mature	No	Components	Specefic to Wicket	Yes	Yes, WYSIWYG HTML	Yes
GWT	2006	Mature	No	Neither query nor components	N/A	Yes	Yes,only with JAVA	Yes
Tapestry	2002	Mature	MVC2	Components	Specefic to Tapestry	No,but all can be written with JAVA	Yes	Yes
Vaadin	2009	Little Mature	No	Neither query nor components	N/A	Yes	Yes,only with JAVA	Yes

Table B.2 – Comparison table of Frameworks

B.4 Choice and justification

After presenting the results of the comparative study, to the project manager,the Framework Wicket was eliminated systematically since the manual integration with JavaScript is not obvious, in fact, the development of a solution that is easy to use, does not require much time and effort that developers can get used to. He then asked us to make the choice between Spring MVC, JSF, GWT and Vaadin because they have similar characteristics, the only criterion for differentiation was ease of use and compatibility with the business layer.

For our project we have decided to find the best choice to develop with the most appropriate technology, this choice is fixed on Spring MVC Framework for the following reasons:

- It's a lightweight applications container that provides dependency injection between the layers of the application.
- It's a Framework that affects all layers of the application and can be integrated with other specialized frameworks in specific layers.
- Spring MVC web tiers are typically easier to test than Struts web tiers, due to the avoidance of forced concrete inheritance and explicit dependence of controllers on the dispatcher servlet.
- Spring Controllers are configured via IoC like any other objects. This makes them easy to test, and beautifully integrated with other objects managed by Spring.

Framework	**Prototyping**	**Facility**	**Documentation**	**Scalability**	**Templating**	**Testability**	**Total**
Spring MVC	2	3	5	4	3	4	21
Struts		2	3	4	3	3	4
Java server Faces		3	3,5	3,5	3,5	4	2,5
GWT		5	4	4	4	1	2 ,5
Wicket		3	2	3	3	2	4
Vaadin	5	4	4	4,5	1	3	21,5

Table B.2 – Final results for each framework

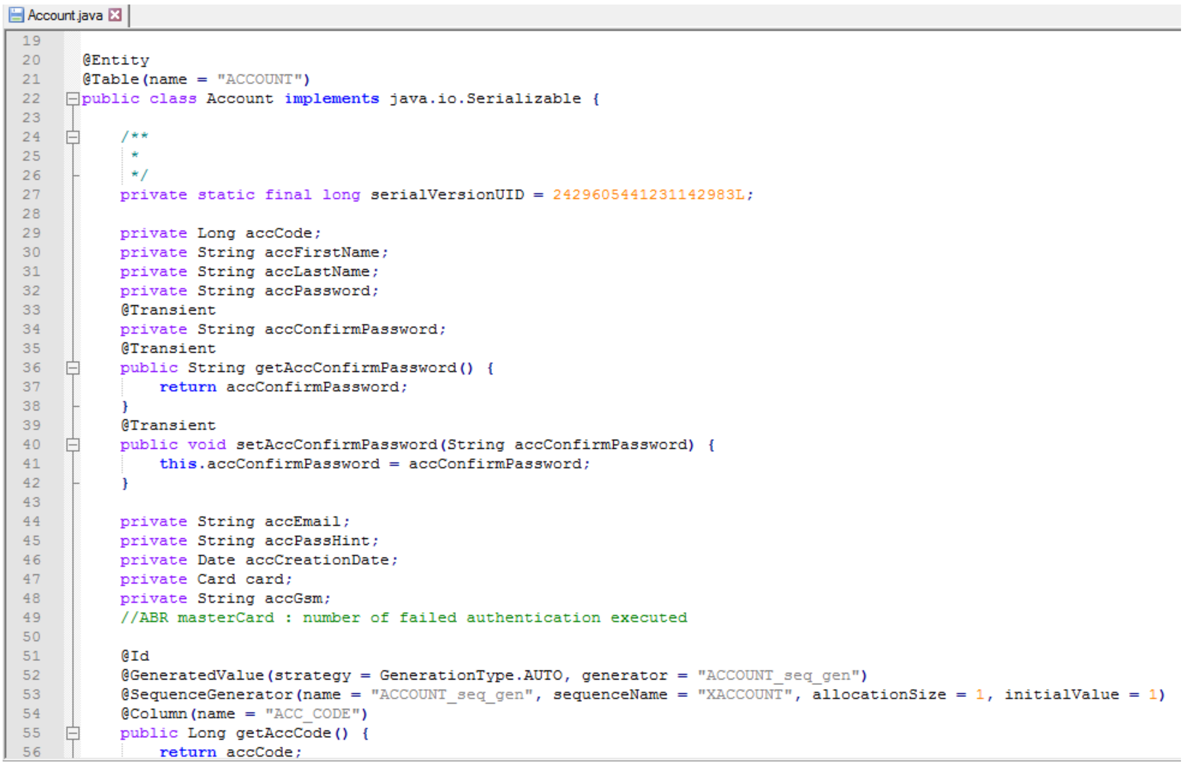
Appendix C

Source code used for 3-D Secure

In order to respect the privacy and the confidentiality of the company,the source code in this appendix is **NOT COMPLETE**.We just attempt to shed light on some classes that we use on our project.

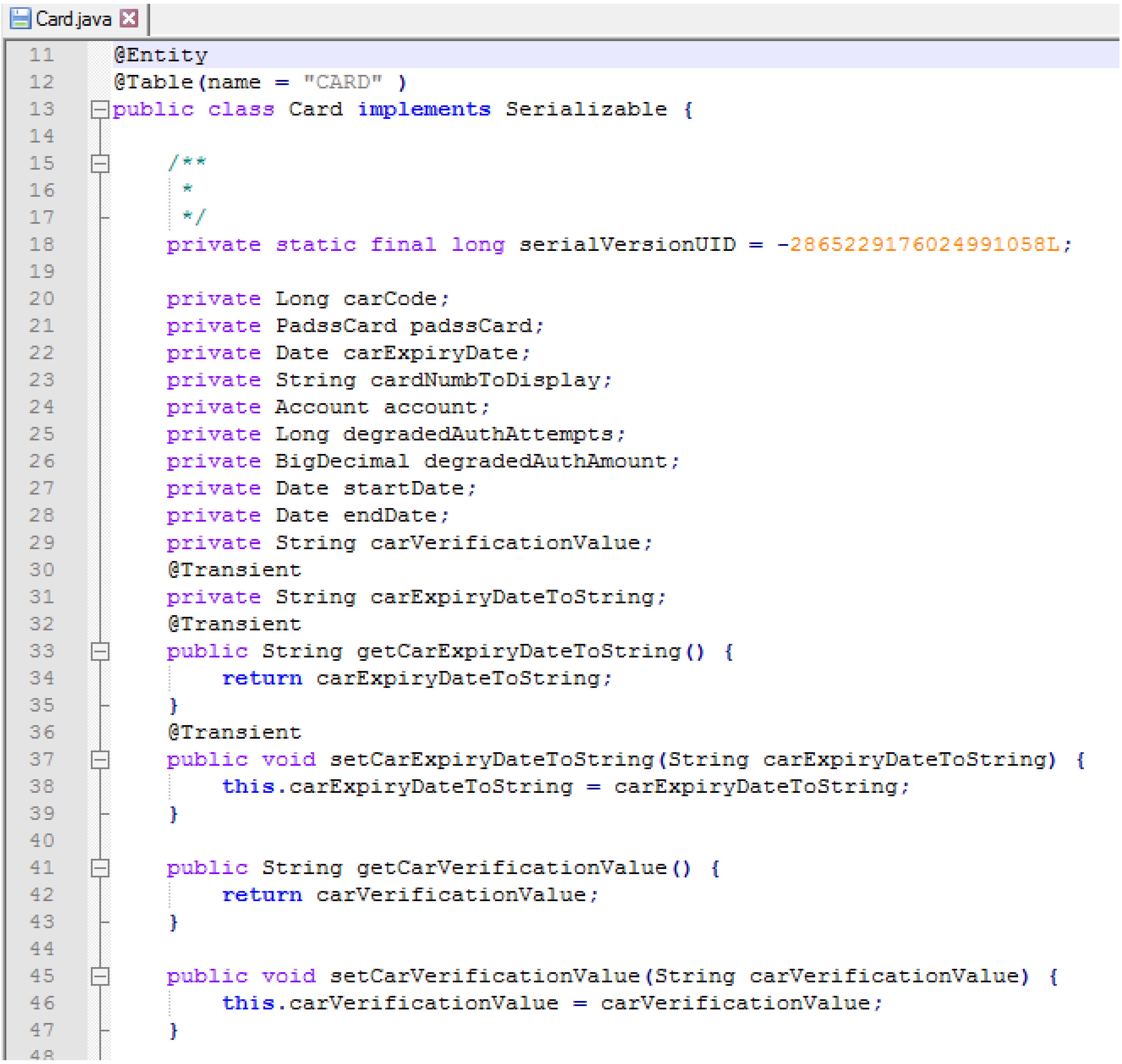
C.1 Domain model

The domain model describe entities used which are actually the tables on database:



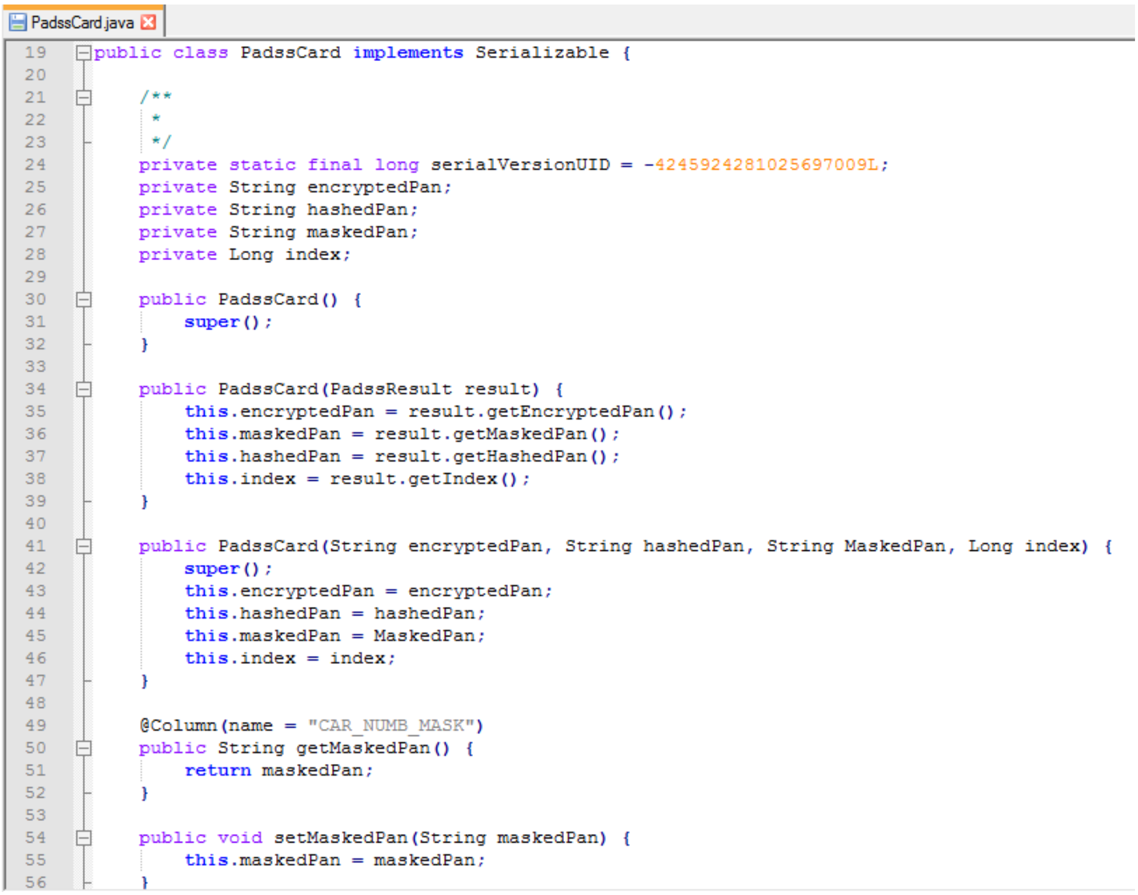
```java

@Entity
@Table(name = "ACCOUNT")
public class Account implements java.io.Serializable {

    /**
     *
     */
    private static final long serialVersionUID = 2429605441231142983L;

    private Long accCode;
    private String accFirstName;
    private String accLastName;
    private String accPassword;
    @Transient
    private String accConfirmPassword;
    @Transient
    public String getAccConfirmPassword() {
        return accConfirmPassword;
    }
    @Transient
    public void setAccConfirmPassword(String accConfirmPassword) {
        this.accConfirmPassword = accConfirmPassword;
    }

    private String accEmail;
    private String accPassHint;
    private Date accCreationDate;
    private Card card;
    private String accGsm;
    //ABR masterCard : number of failed authentication executed

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO, generator = "ACCOUNT_seq_gen")
    @SequenceGenerator(name = "ACCOUNT_seq_gen", sequenceName = "XACCOUNT", allocationSize = 1, initialValue = 1)
    @Column(name = "ACC_CODE")
    public Long getAccCode() {
        return accCode;
```

Figure C.1 – Account.java



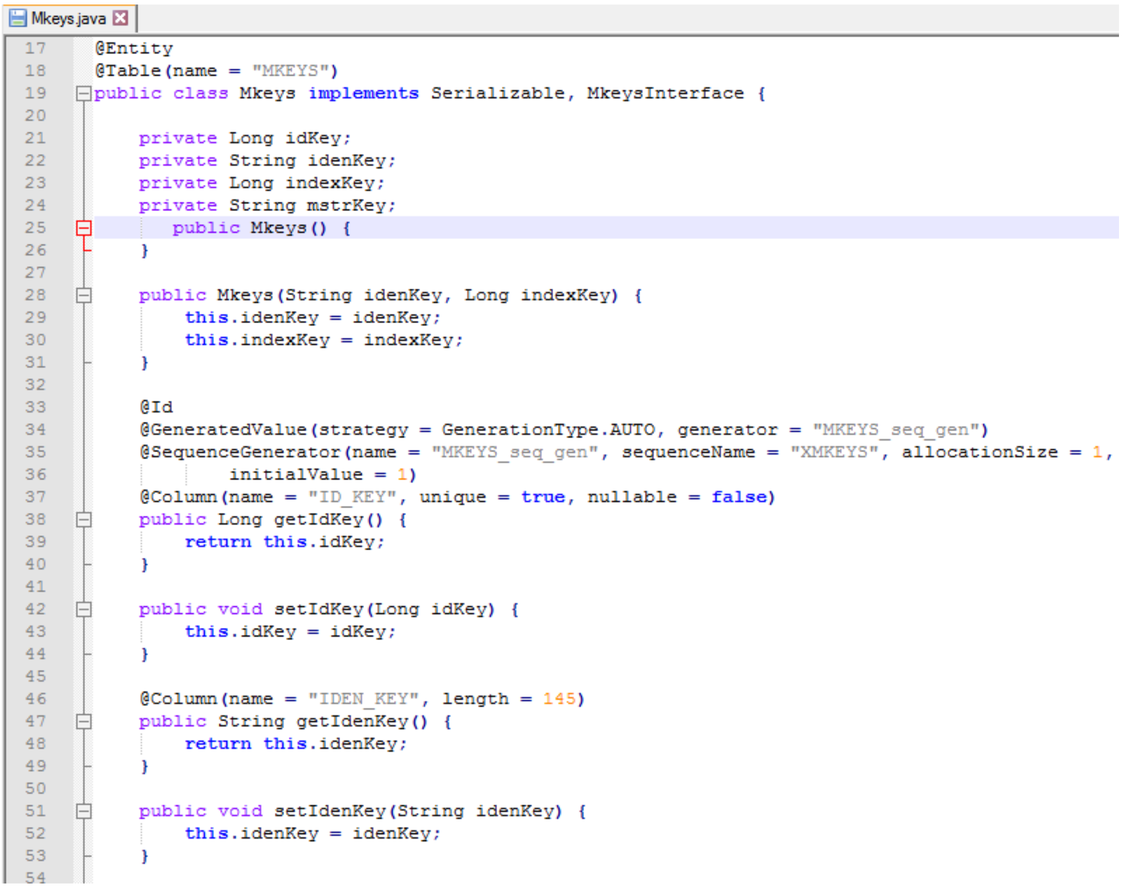
```java
@Entity
@Table(name = "CARD" )
public class Card implements Serializable {

    /**
     *
     */
    private static final long serialVersionUID = -2865229176024991058L;

    private Long carCode;
    private PadssCard padssCard;
    private Date carExpiryDate;
    private String cardNumbToDisplay;
    private Account account;
    private Long degradedAuthAttempts;
    private BigDecimal degradedAuthAmount;
    private Date startDate;
    private Date endDate;
    private String carVerificationValue;
    @Transient
    private String carExpiryDateToString;
    @Transient
    public String getCarExpiryDateToString() {
        return carExpiryDateToString;
    }
    @Transient
    public void setCarExpiryDateToString(String carExpiryDateToString) {
        this.carExpiryDateToString = carExpiryDateToString;
    }

    public String getCarVerificationValue() {
        return carVerificationValue;
    }

    public void setCarVerificationValue(String carVerificationValue) {
        this.carVerificationValue = carVerificationValue;
    }
```

Figure C.2 – Card.java



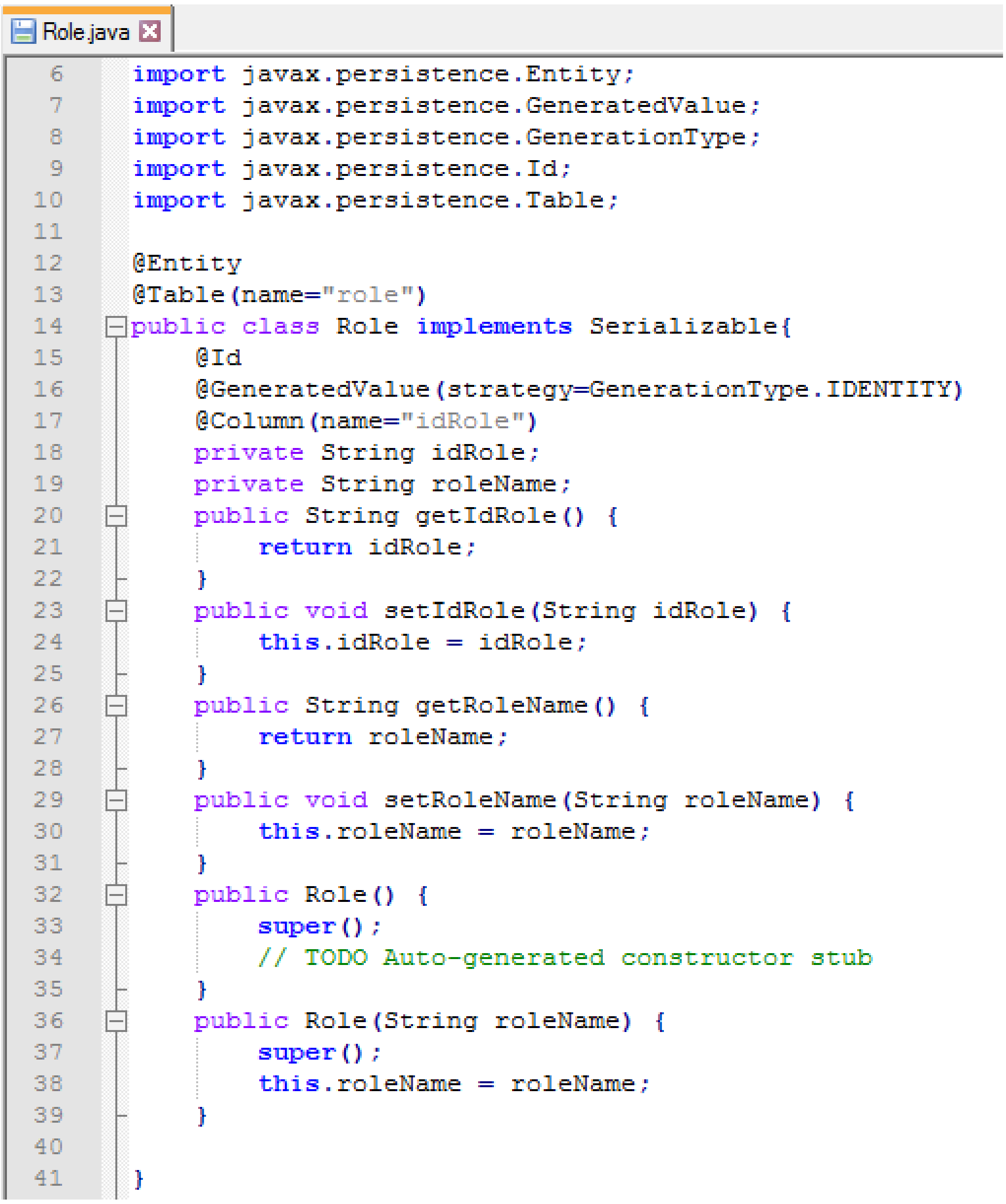
```java
public class PadssCard implements Serializable {

    /**
     *
     */
    private static final long serialVersionUID = -4245924281025697009L;
    private String encryptedPan;
    private String hashedPan;
    private String maskedPan;
    private Long index;

    public PadssCard() {
        super();
    }

    public PadssCard(PadssResult result) {
        this.encryptedPan = result.getEncryptedPan();
        this.maskedPan = result.getMaskedPan();
        this.hashedPan = result.getHashedPan();
        this.index = result.getIndex();
    }

    public PadssCard(String encryptedPan, String hashedPan, String MaskedPan, Long index) {
        super();
        this.encryptedPan = encryptedPan;
        this.hashedPan = hashedPan;
        this.maskedPan = MaskedPan;
        this.index = index;
    }

    @Column(name = "CAR_NUMB_MASK")
    public String getMaskedPan() {
        return maskedPan;
    }

    public void setMaskedPan(String maskedPan) {
        this.maskedPan = maskedPan;
    }
```

Figure C.3 – PadssCard.java



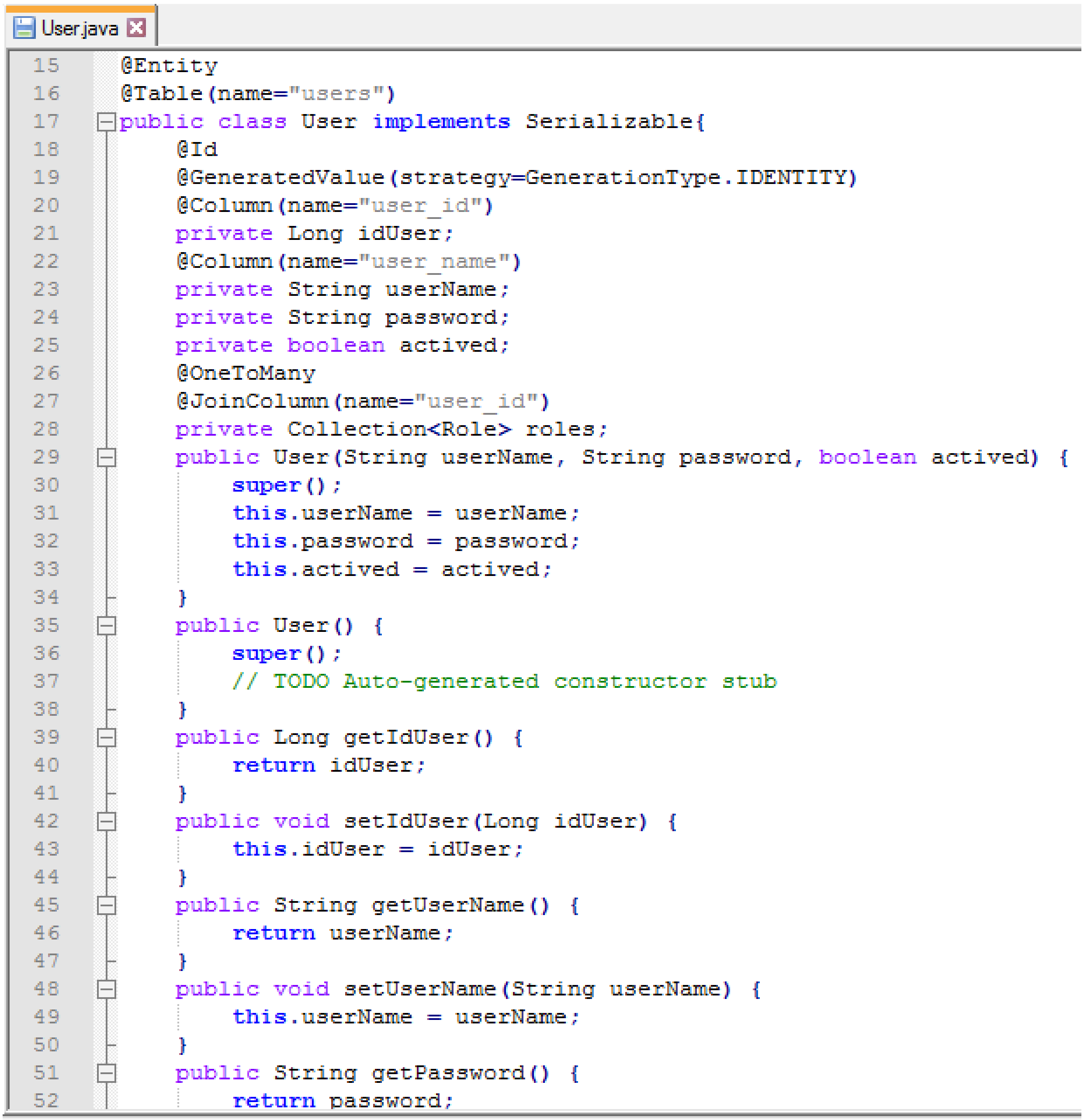
```java
@Entity
@Table(name = "MKEYS")
public class Mkeys implements Serializable, MkeysInterface {

    private Long idKey;
    private String idenKey;
    private Long indexKey;
    private String mstrKey;
       public Mkeys() {
    }

    public Mkeys(String idenKey, Long indexKey) {
        this.idenKey = idenKey;
        this.indexKey = indexKey;
    }

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO, generator = "MKEYS_seq_gen")
    @SequenceGenerator(name = "MKEYS_seq_gen", sequenceName = "XMKEYS", allocationSize = 1,
            initialValue = 1)
    @Column(name = "ID_KEY", unique = true, nullable = false)
    public Long getIdKey() {
        return this.idKey;
    }

    public void setIdKey(Long idKey) {
        this.idKey = idKey;
    }

    @Column(name = "IDEN_KEY", length = 145)
    public String getIdenKey() {
        return this.idenKey;
    }

    public void setIdenKey(String idenKey) {
        this.idenKey = idenKey;
    }

```

Figure C.4 – Mkeys.java



```
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name="role")
public class Role implements Serializable{
    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    @Column(name="idRole")
    private String idRole;
    private String roleName;
    public String getIdRole() {
        return idRole;
    }
    public void setIdRole(String idRole) {
        this.idRole = idRole;
    }
    public String getRoleName() {
        return roleName;
    }
    public void setRoleName(String roleName) {
        this.roleName = roleName;
    }
    public Role() {
        super();
        // TODO Auto-generated constructor stub
    }
    public Role(String roleName) {
        super();
        this.roleName = roleName;
    }

}
```

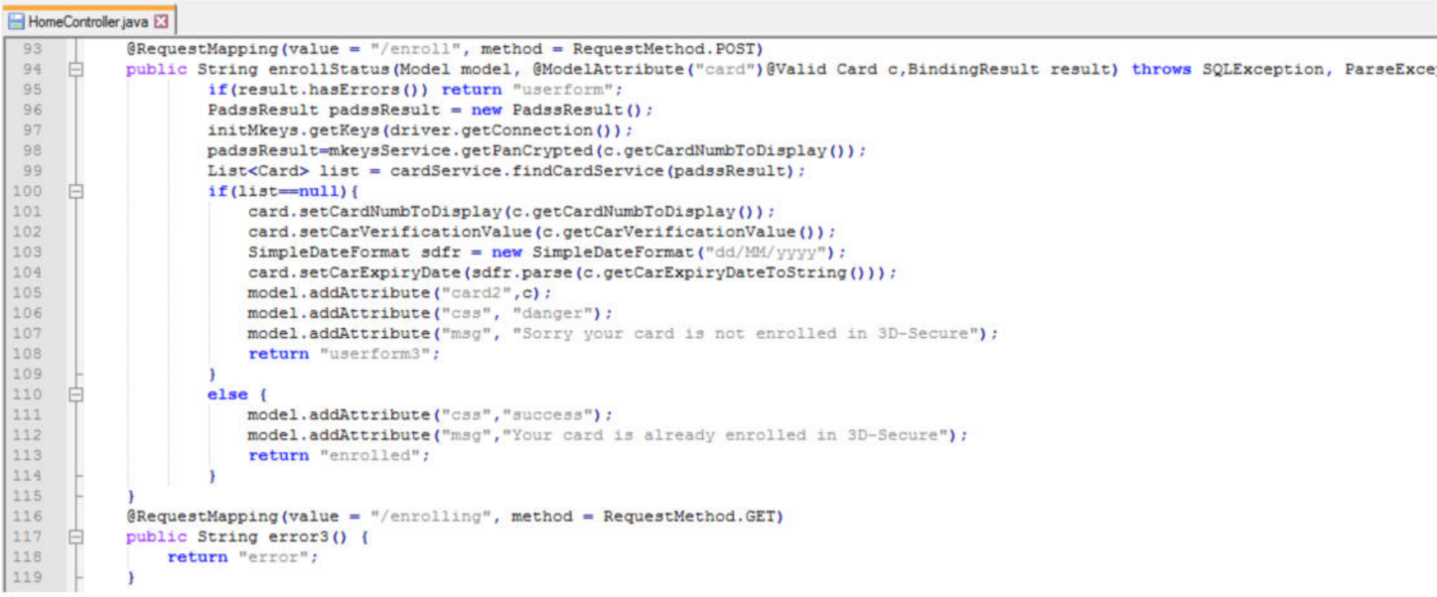
Figure C.5 – Role.java



```java
@Entity
@Table(name="users")
public class User implements Serializable{
    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    @Column(name="user_id")
    private Long idUser;
    @Column(name="user_name")
    private String userName;
    private String password;
    private boolean actived;
    @OneToMany
    @JoinColumn(name="user_id")
    private Collection<Role> roles;
    public User(String userName, String password, boolean actived) {
        super();
        this.userName = userName;
        this.password = password;
        this.actived = actived;
    }
    public User() {
        super();
        // TODO Auto-generated constructor stub
    }
    public Long getIdUser() {
        return idUser;
    }
    public void setIdUser(Long idUser) {
        this.idUser = idUser;
    }
    public String getUserName() {
        return userName;
    }
    public void setUserName(String userName) {
        this.userName = userName;
    }
    public String getPassword() {
        return password;
```

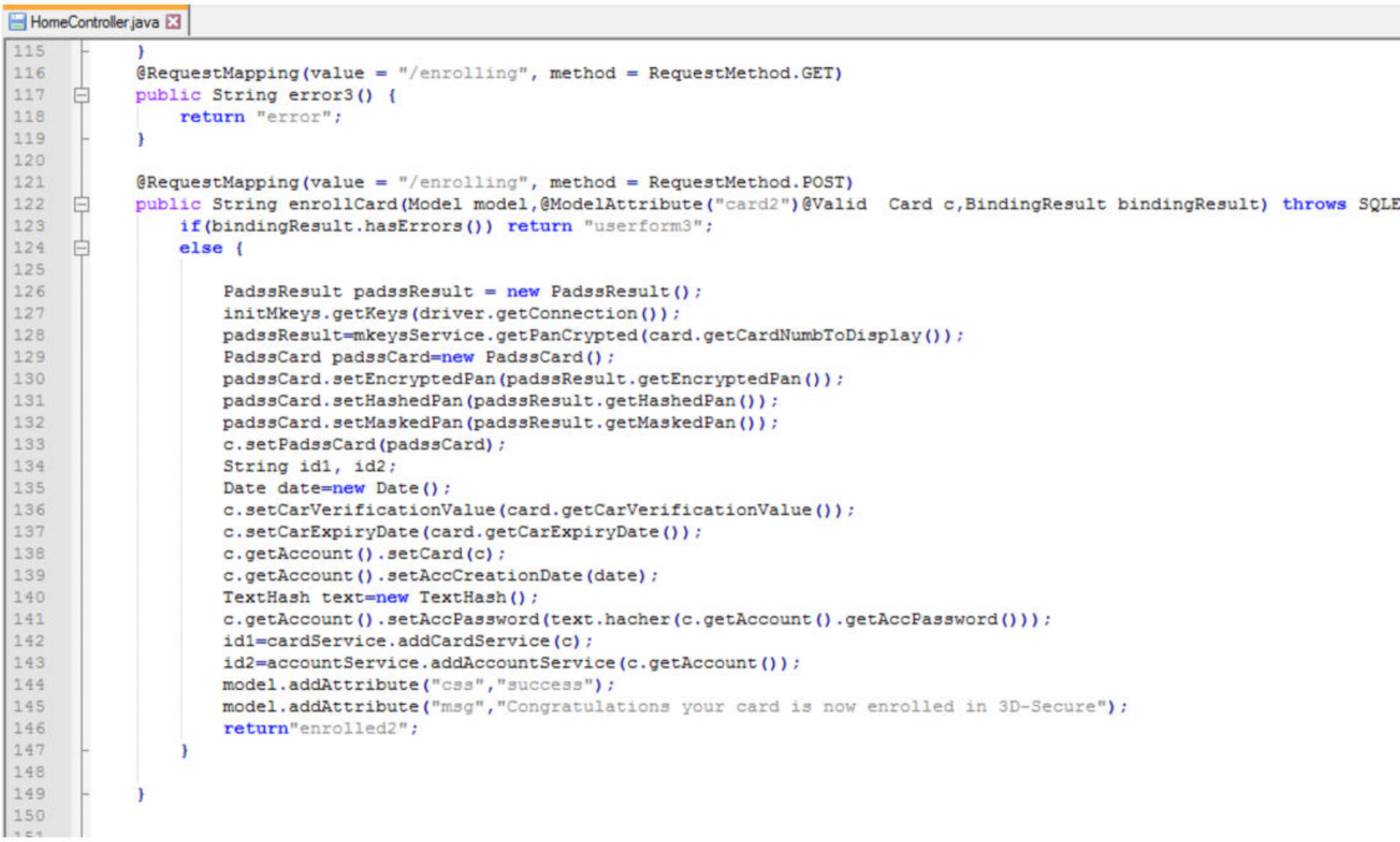
Figure C.6 – User.java

C.2 Enrollment process



```java
@RequestMapping(value = "/enroll", method = RequestMethod.POST)
public String enrollStatus(Model model, @ModelAttribute("card")@Valid Card c,BindingResult result) throws SQLException, ParseExce
        if(result.hasErrors()) return "userform";
        PadssResult padssResult = new PadssResult();
        initMkeys.getKeys(driver.getConnection());
        padssResult=mkeysService.getPanCrypted(c.getCardNumbToDisplay());
        List<Card> list = cardService.findCardService(padssResult);
        if(list==null){
            card.setCardNumbToDisplay(c.getCardNumbToDisplay());
            card.setCarVerificationValue(c.getCarVerificationValue());
            SimpleDateFormat sdfr = new SimpleDateFormat("dd/MM/yyyy");
            card.setCarExpiryDate(sdfr.parse(c.getCarExpiryDateToString()));
            model.addAttribute("card2",c);
            model.addAttribute("css", "danger");
            model.addAttribute("msg", "Sorry your card is not enrolled in 3D-Secure");
            return "userform3";
        }
        else {
            model.addAttribute("css","success");
            model.addAttribute("msg","Your card is already enrolled in 3D-Secure");
            return "enrolled";
        }
}
@RequestMapping(value = "/enrolling", method = RequestMethod.GET)
public String error3() {
    return "error";
}
```

Figure C.7 – Enrollement process (part 1)

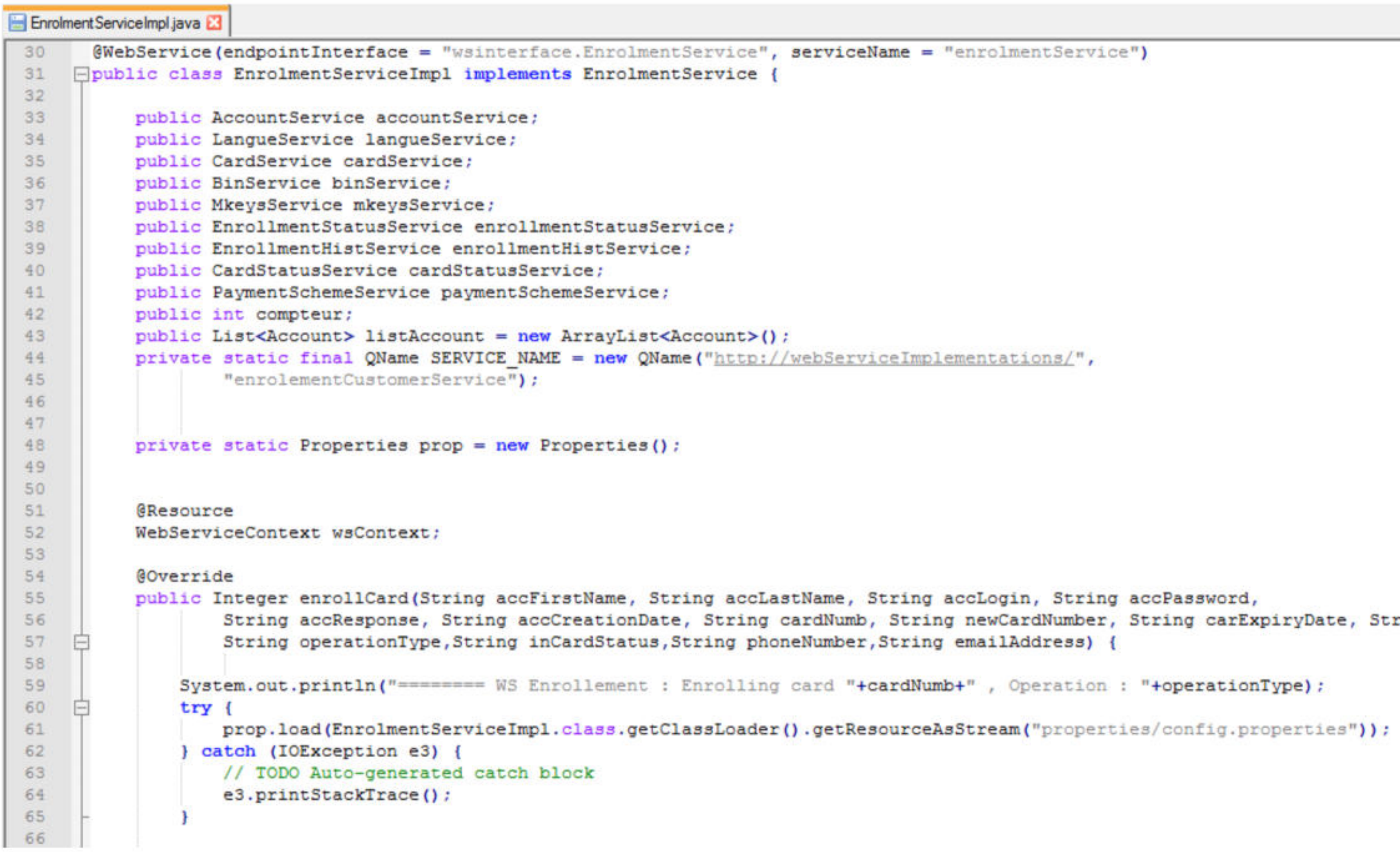


```
    }
    @RequestMapping(value = "/enrolling", method = RequestMethod.GET)
    public String error3() {
        return "error";
    }

    @RequestMapping(value = "/enrolling", method = RequestMethod.POST)
    public String enrollCard(Model model,@ModelAttribute("card2")@Valid  Card c,BindingResult bindingResult) throws SQLE
        if(bindingResult.hasErrors()) return "userform3";
        else {

            PadssResult padssResult = new PadssResult();
            initMkeys.getKeys(driver.getConnection());
            padssResult=mkeysService.getPanCrypted(card.getCardNumbToDisplay());
            PadssCard padssCard=new PadssCard();
            padssCard.setEncryptedPan(padssResult.getEncryptedPan());
            padssCard.setHashedPan(padssResult.getHashedPan());
            padssCard.setMaskedPan(padssResult.getMaskedPan());
            c.setPadssCard(padssCard);
            String id1, id2;
            Date date=new Date();
            c.setCarVerificationValue(card.getCarVerificationValue());
            c.setCarExpiryDate(card.getCarExpiryDate());
            c.getAccount().setCard(c);
            c.getAccount().setAccCreationDate(date);
            TextHash text=new TextHash();
            c.getAccount().setAccPassword(text.hacher(c.getAccount().getAccPassword()));
            id1=cardService.addCardService(c);
            id2=accountService.addAccountService(c.getAccount());
            model.addAttribute("css","success");
            model.addAttribute("msg","Congratulations your card is now enrolled in 3D-Secure");
            return"enrolled2";
        }

    }

```

Figure C.8 – Enrollement process (part 2)

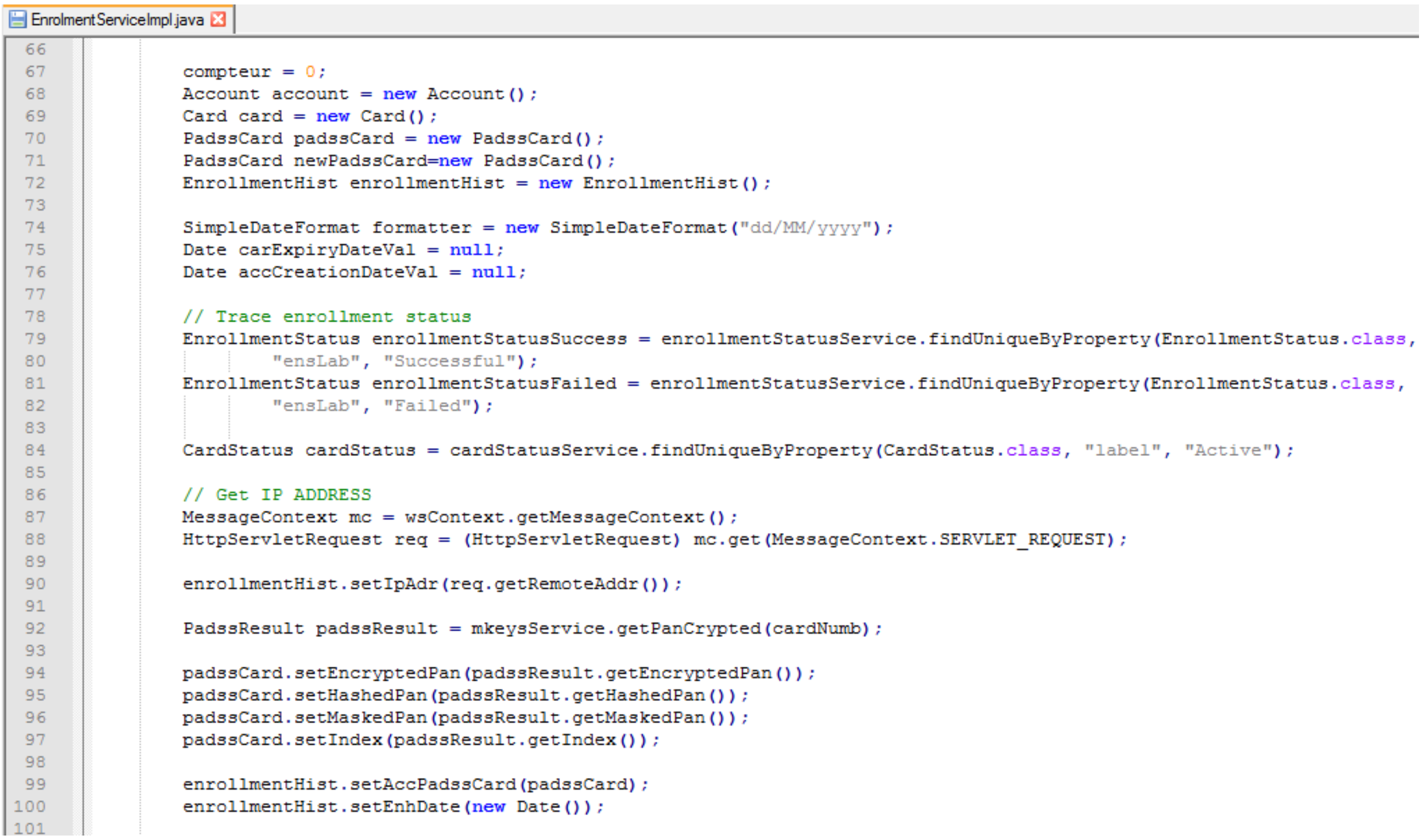
C.3 Enrollment Service Implementation

At this section we present the enrollment service used at the Back Office:



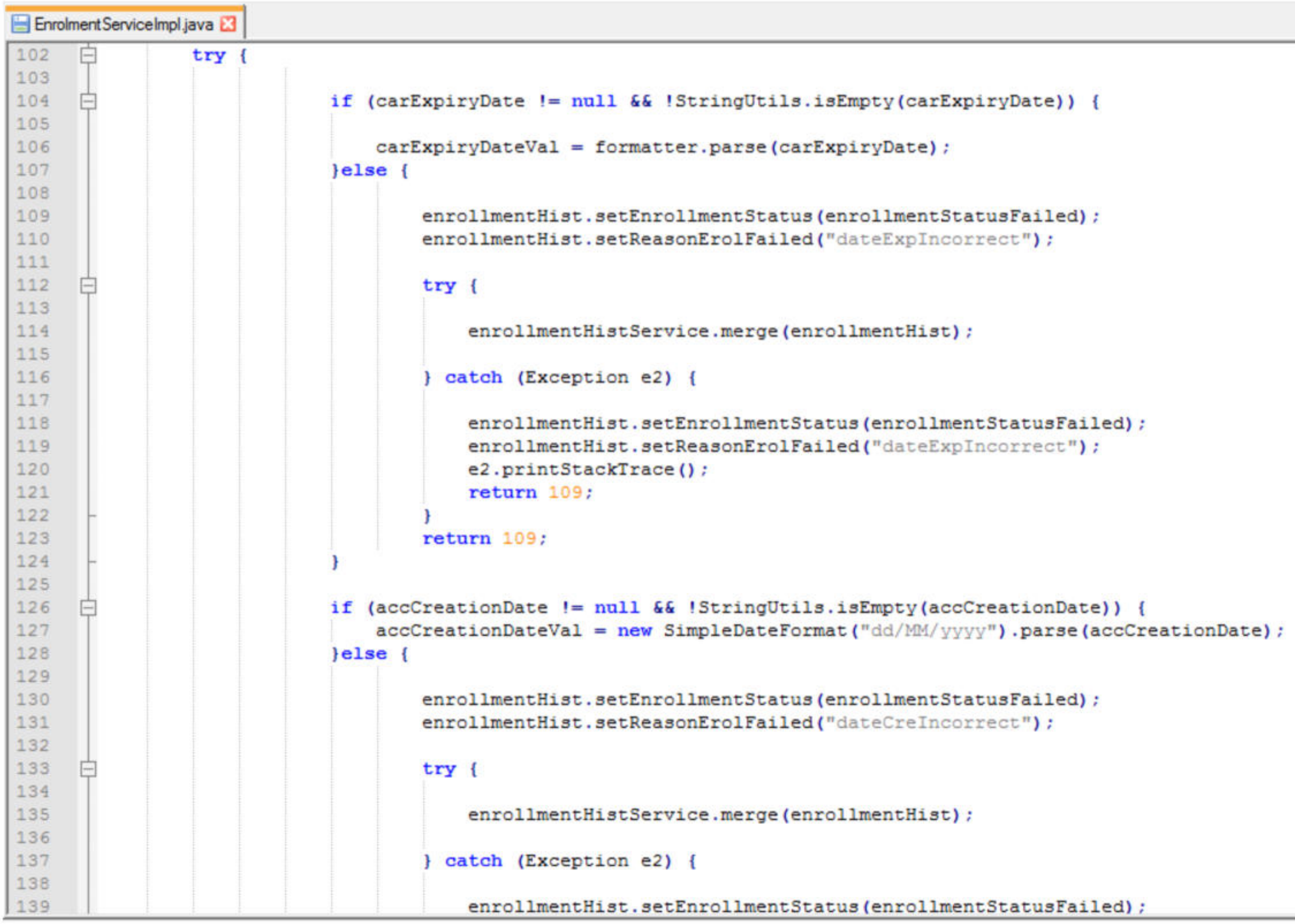
```
@WebService(endpointInterface = "wsinterface.EnrolmentService", serviceName = "enrolmentService")
public class EnrolmentServiceImpl implements EnrolmentService {

    public AccountService accountService;
    public LangueService langueService;
    public CardService cardService;
    public BinService binService;
    public MkeysService mkeysService;
    public EnrollmentStatusService enrollmentStatusService;
    public EnrollmentHistService enrollmentHistService;
    public CardStatusService cardStatusService;
    public PaymentSchemeService paymentSchemeService;
    public int compteur;
    public List<Account> listAccount = new ArrayList<Account>();
    private static final QName SERVICE_NAME = new QName("http://webServiceImplementations/",
            "enrolementCustomerService");


    private static Properties prop = new Properties();


    @Resource
    WebServiceContext wsContext;

    @Override
    public Integer enrollCard(String accFirstName, String accLastName, String accLogin, String accPassword,
            String accResponse, String accCreationDate, String cardNumb, String newCardNumber, String carExpiryDate, Str
            String operationType,String inCardStatus,String phoneNumber,String emailAddress) {

        System.out.println("======== WS Enrollement : Enrolling card "+cardNumb+" , Operation : "+operationType);
        try {
            prop.load(EnrolmentServiceImpl.class.getClassLoader().getResourceAsStream("properties/config.properties"));
        } catch (IOException e3) {
            // TODO Auto-generated catch block
            e3.printStackTrace();
        }

```

Figure C.9 – Enrollment Service code (part 1)



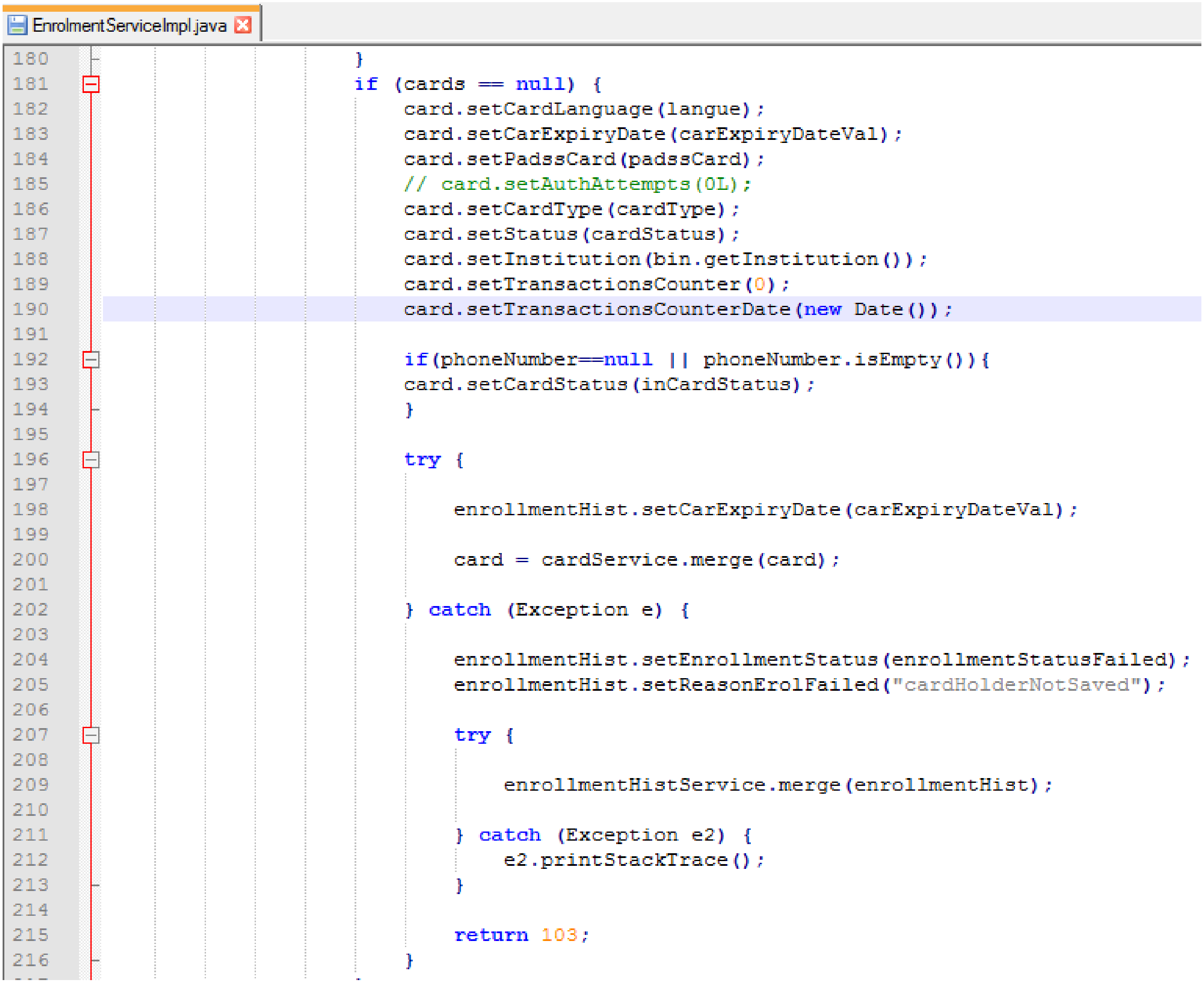
```

        compteur = 0;
        Account account = new Account();
        Card card = new Card();
        PadssCard padssCard = new PadssCard();
        PadssCard newPadssCard=new PadssCard();
        EnrollmentHist enrollmentHist = new EnrollmentHist();

        SimpleDateFormat formatter = new SimpleDateFormat("dd/MM/yyyy");
        Date carExpiryDateVal = null;
        Date accCreationDateVal = null;

        // Trace enrollment status
        EnrollmentStatus enrollmentStatusSuccess = enrollmentStatusService.findUniqueByProperty(EnrollmentStatus.class,
                "ensLab", "Successful");
        EnrollmentStatus enrollmentStatusFailed = enrollmentStatusService.findUniqueByProperty(EnrollmentStatus.class,
                "ensLab", "Failed");

        CardStatus cardStatus = cardStatusService.findUniqueByProperty(CardStatus.class, "label", "Active");

        // Get IP ADDRESS
        MessageContext mc = wsContext.getMessageContext();
        HttpServletRequest req = (HttpServletRequest) mc.get(MessageContext.SERVLET_REQUEST);

        enrollmentHist.setIpAdr(req.getRemoteAddr());

        PadssResult padssResult = mkeysService.getPanCrypted(cardNumb);

        padssCard.setEncryptedPan(padssResult.getEncryptedPan());
        padssCard.setHashedPan(padssResult.getHashedPan());
        padssCard.setMaskedPan(padssResult.getMaskedPan());
        padssCard.setIndex(padssResult.getIndex());

        enrollmentHist.setAccPadssCard(padssCard);
        enrollmentHist.setEnhDate(new Date());

```

Figure C.10 – Enrollment Service code (part 2)



```java
        try {

                    if (carExpiryDate != null && !StringUtils.isEmpty(carExpiryDate)) {

                        carExpiryDateVal = formatter.parse(carExpiryDate);
                    }else {

                            enrollmentHist.setEnrollmentStatus(enrollmentStatusFailed);
                            enrollmentHist.setReasonErolFailed("dateExpIncorrect");

                            try {

                                enrollmentHistService.merge(enrollmentHist);

                            } catch (Exception e2) {

                                enrollmentHist.setEnrollmentStatus(enrollmentStatusFailed);
                                enrollmentHist.setReasonErolFailed("dateExpIncorrect");
                                e2.printStackTrace();
                                return 109;
                            }
                            return 109;
                    }

                    if (accCreationDate != null && !StringUtils.isEmpty(accCreationDate)) {
                        accCreationDateVal = new SimpleDateFormat("dd/MM/yyyy").parse(accCreationDate);
                    }else {

                            enrollmentHist.setEnrollmentStatus(enrollmentStatusFailed);
                            enrollmentHist.setReasonErolFailed("dateCreIncorrect");

                            try {

                                enrollmentHistService.merge(enrollmentHist);

                            } catch (Exception e2) {

                                enrollmentHist.setEnrollmentStatus(enrollmentStatusFailed);
```

Figure C.11 – Enrollment Service code (part 3)



```
}
if (cards == null) {
    card.setCardLanguage(langue);
    card.setCarExpiryDate(carExpiryDateVal);
    card.setPadssCard(padssCard);
    // card.setAuthAttempts(0L);
    card.setCardType(cardType);
    card.setStatus(cardStatus);
    card.setInstitution(bin.getInstitution());
    card.setTransactionsCounter(0);
    card.setTransactionsCounterDate(new Date());

    if(phoneNumber==null || phoneNumber.isEmpty()){
    card.setCardStatus(inCardStatus);
    }

    try {

        enrollmentHist.setCarExpiryDate(carExpiryDateVal);

        card = cardService.merge(card);

    } catch (Exception e) {

        enrollmentHist.setEnrollmentStatus(enrollmentStatusFailed);
        enrollmentHist.setReasonErolFailed("cardHolderNotSaved");

        try {

            enrollmentHistService.merge(enrollmentHist);

        } catch (Exception e2) {
            e2.printStackTrace();
        }

        return 103;
    }
```

Figure C.12 – Enrollment Service code (part 4)