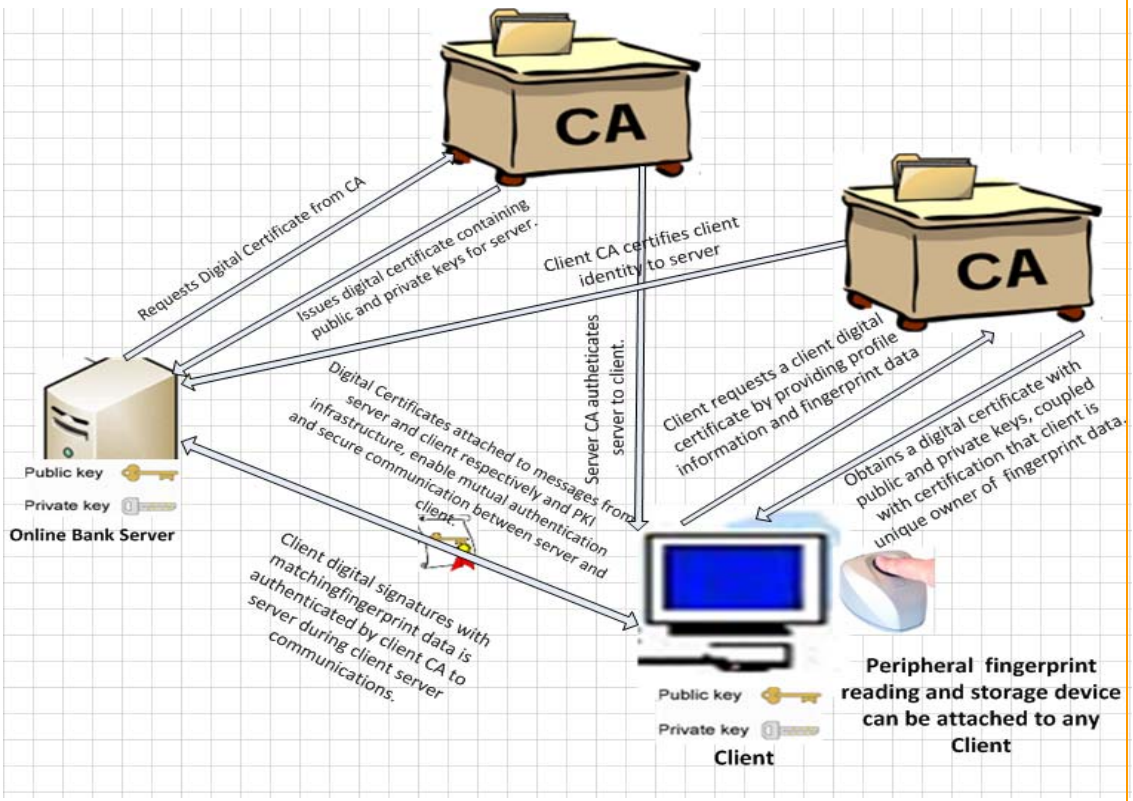
Information Technology University of Copenhagen

Software Development Technology Department

Final Thesis for the award of an MSc. In Computer Science

Spring 2011

Pro-active Architecture and Implementation of a Secure Online Banking System that Uses Fingerprint Data as Part of Client Side Digital Signatures.



By

ANE DIVINE JINOR

(adji@itu.dk)

Supervisor:

Joseph Roland Kiniry

Table of Contents

Acknowledgement

I am very grateful to the Danish Government and the Administration of the Information Technology University in Copenhagen, for allowing me to go through the Master Degree in Computer Science studies on Scholarship, as an international non-European Union student.

I am also grateful to my Supervisor Joseph Roland Kiniry who despite his busy schedule, always found time to read project drafts and to hold project meeting with me, during which many important recommendations were made.

Finally I am grateful to my Wife Colette Ane , my kids Christol Tasi Ane and Janelle Awum Ane who gave me joy and moral support throughout this arduous task.

Dedication

This project is dedicated to my late parents Christol Tasi Ane and Joan Awum Ane, who did not live long enough, to see their son become a Computer Science Engineer.

Abstract

This project focuses on proposing an innovative security mechanism for online banking which uses the unique biometric features of user fingerprints to control access to user accounts and to digitally sign online transactions.

Implementation here is based on the ASP.NET Framework 4.0 Framework, with Visual Studio 2010 as the Integrated Development Environment.

Chapter 1 of these projects based on the Problem Area, identifies weaknesses in contemporary online banking access control and authentication mechanisms.

The use of fingerprints for access control and authentication offers the following advantages not available with other online banking authorization and authentication mechanisms. Fingerprints use is more secure in that they:

- Are unique to each user;
- can neither be scammed, replicated, forgotten nor stolen;
- Are owned by unique users and need not be remembered.

With this in mine, the following Research Question was formulated for this project:

"How will online banking be made more secure by incorporating user fingerprints into client side digital signatures, through a pro-active design and implementation strategy?"

Chapter 2 is based on the methodology to be used for the development process and proposes a hybrid methodology between the Unified Process and the Extreme Programming development methodologies as well as the use of prototyping. This chapter also has the project undertaking on features that must be implemented. It also proposes some features that may be implemented as well as the scope, the limitations and reflections on the project and what has to be delivered at the end of the project.

Chapter 3 is based on the requirement analysis, the risk analysis and the test analysis.

Chapter 4 focuses on the architectural design. Here a hybrid between the Client-Server Pattern and the Layered Architecture Pattern has been used for the architecture of the application. Horizontal decomposition of the server component led to the identification of three subcomponents:

i) An access control component

ii) A component holding the logic for bank transactions and

iii) A component that provides the logic for client and server digital certificates and digital signatures.

Development of the application will be based on separate development and subsequent integration of these components.

The design also proposes the use of mutual authentication between client and server and the Public Key Infrastructure for the encryption process.

The design also proposes an innovative certification regime which should include and certify user fingerprints used to sign digital signatures and to control access to bank accounts.

Chapter 5 is based on the actual implementation process which was carried out according to the components described in the chapter on design.

Here Globally Unique Identifiers (GUIDs) have been used to represent fingerprint data since raw fingerprint data could not be acquired from any third party application.

The implementation has also been delivered in two units due to the impossibility to integrate the signature creation process with the access control unit, without deployment to a Windows Server.

The implementation has two separate digital certificates. One based on the customary self-signed X.509 digital certificates and an innovative fingerprint digital certificate proposed by this author.

Chapter 6 is based on Tests results and the fulfillment of requirements for this project.

Chapter 7 provides a User Guide on how to test the application implemented.

Chapter 8 on reflections presents the experiences this author went through during the development process. It also summarizes the most important features that have been analyzed, designed and successfully implemented. This chapter also identifies the limitations faced by this author and makes proposals on further research in relation to the requirements for the proposed application.

1 Problem Formulation

1.1 The Problem Area

The adoption of online banking is currently experiencing a steady rise.[1]

Users who adopt this modern form of banking are probably enticed by the convenient and the expedient manner in which they can carry out their banking needs.[2]

With regards to banking institutions, online banking offers to them, greater possibilities for cutting costs, offers business efficiency and is not dependent on brick and mortar geographical locations.[3]

Taking into consideration the current growth of online banking, any modern commercial bank is likely to follow this trend, in order to keep in step with competition.

Users with an internet connection can pay bills, do money transfers, trade in stocks, access their bank statements, shop online etc., from anywhere, through online banking.[4]

Contemporary online banking systems make use of varying forms of user authentication:

- In most cases authentication is through a combination of potentially strong passwords and users IDs.[5]
- Most of these systems ensure secure connections between the clients and the servers over SSL and many of them also have server side digital certificates.[6]

---

[1] Source article "Online Banking Growth Outpacing That of Call Centers, Branches, ATMs, Tower Group Report Says" By Katherine Burger published on MAY 21, 2007 on Bank Systems and Technology website available at: http://www.banktech.com/showArticle.jhtml?articleID=199905123 accessed on 30-10-2010

[2] Source "Convenience Driving Online Banking Growth"

Published Monday, July 20, 2009 By Mike Sachoff on Web Pro News web site available at: http://www.webpronews.com/topnews/2009/07/20/convenience-driving-online-banking-growth accessed on 29-10-2010

[3] See article "The Advantages of Online Banking to Banks" available on Ehow Web site at: http://www.ehow.com/facts_5003054_advantages-online-banking-banks.html accessed on 01-11-2010

[4] See Facts on online banking at Danske Bank Web Site : http://www.danskebank.dk/en-dk/Personal/Pages/eBanking.aspx?tab=1#tabanchor accessed on 01-11-2010

[5] See for example : Apple Federal Credit Union's net banking security features available at : https://www.applefcu.org/nbsecurity.asp Accessed on 27-102010 . See also Citi Bank Online Banking Site at: https://online.citibank.com/US/JPS/portal/Index.do accessed on 01-11-2010. See also online banking login page for Danske Bank at: https://netbank.danskebank.dk/html/index.html?site=DBNB&secsystem=DI accessed on 29-10-2010

- Some online banking systems in addition to the above features impose client side digital signatures and certificates.[7]
- In other cases additionally there is the imposition of one time password or regular password modification policies, to users.[8]
- Some systems also limit the number of login attempts as a security measure.[9]

Despite the use of these security features mentioned above, several security vulnerabilities still persist; partially due to design flaws and implementation bugs and in many cases due to the inherent weaknesses generally associated with the use of user IDs and passwords for authentication into systems.

Black hat software experts can exploit these vulnerabilities in order to hack into online banking systems.[10]

Apart from system attacks by software experts, today there is steady rise of a class of social engineers who are not necessarily software engineers. This category of cyber criminals, are increasingly becoming successful in their exploits to illegally get users authentication credentials, through online and other phishing scams.[11]

One strategy to water down such online security vulnerabilities inherent in contemporary online banking systems, is to ensure that users log into such systems using something:

- That is unique to each user;
- which can neither be scammed, replicated, forgotten nor stolen;
- That he owns and not something that he remembers.

---

[6] See for example *Ibidem*

[7] See for example article on My Digital Life Web site about South Africa's NedBank "From Nedbank to Netbank " ; Published Wednesday, 18 June 2008 15:05 Written by Leon Engelbrecht and available at : http://www.mydigitallife.co.za/index.php?option=com_content&view=article&id=1038163&catid=17:digital-connectivity&Itemid=36 , accessed on 24-10-2010

[8] See for example Nordea Bank Denmark's online banking account creation steps; available in Danish at : http://www.nordea.dk/Privat/Internet+og+telefon/R%c3%a5dgivning+om+internet+og+telefon/S%c3%a5dan+skifter+du+til+NemID/1381022.html accessed on 24-10-2010

[9] See for example South Africa Nedbank 's Net Bank Online Security Policy available at : https://www.nedbank.co.za/website/content/promotions/index_detail.asp?PromoID=529 accessed on 01-11-2010

[10] See Michael Howard, David LeBlanc, John Vega , "19 Deadly Sins of Software Security ; Programming Flaws and How to Fix Them" ; Chapter 11: " Use of Weak Pass Word Based Systems ", McGraw-Hill/Osborne 2005

[11] See for example the article "Online Banking Fraud on the Rise" on PC Pro Magazine available at : http://www.pcpro.co.uk/news/356305/online-banking-fraud-on-the-rise By Hani Megerisi Posted on 10 Mar 2010 accessed on 04-11-2010

Biometric web based authentication that makes use of unique human characteristics like iris scans or fingerprints, coupled with a rigorous design and implementation of security requirements, can lead to far more secure banking systems.

Fingerprint scanning essentially provides an identification of a person based on the acquisition and recognition of the unique patterns and ridges in a fingerprint.

The actual fingerprint identification process will change slightly between products and systems.[12]

The basis of identification, however, is nearly the same. Standard systems are comprised of a sensor for scanning a fingerprint and a processor which stores the fingerprint database and software which compares and matches the fingerprint to the predefined database. Within the database, a fingerprint is usually matched to a reference number, or PIN number which is then matched to a person's name or account. In instances of security the match is generally used to allow or disallow access.[13]

As is the case with the use of user IDs and passwords, it is possible to capture finger print data over a network or from its storage location and replay it in order to access the system for which it is required for authentication and authorisation.[14]

In cases where either user IDs and passwords or fingerprints are used for authentication, stringent design and encryption strategies can water down the security vulnerabilities, to a greater extent.

However fingerprint based authentication and authorization, can provide the following distinct and positive security features not available to user ID and password authentication and access control systems:

- Phishing or social engineering cannot be used to acquire finger print data;
- Hacking through brute force or dictionary attacks cannot be used to acquire fingerprint data required for authentication;
- Systems that use fingerprints for authentication do not need complicated password policies.

---

[12] Source: Webopedia website article "How Fingerprint Scanners Work", available at http://www.webopedia.com/DidYouKnow/Computer_Science/2004/fingerprint.asp , Posted: 10-01-2004 , Last Updated: 08-31-2010 accessed on 03-11-2010

[13] Source: Webopedia website article "How Fingerprint Scanners Work", available at http://www.webopedia.com/DidYouKnow/Computer_Science/2004/fingerprint.asp , Posted: 10-01-2004 , Last Updated: 08-31-2010, accessed on 03-11-2010

[14] Source: " Hacking Finger Print Scanners or: Why Microsoft's Fingerprint Reader is not a Security Feature "Available at : http://www.blackhat.com/presentations/bh-europe-06/bh-eu-06-Kiviharju/bh-eu-06-kiviarju.pdf accessed on 04-11-2010

Despite the very limited use of fingerprint authentication for online banking, there is existing research which indicates that users prefer and believe more in fingerprint authentication for online banking, than other forms of authentication.[15]

Recently, a Polish Bank ignored the apathy apparent with European banks to adopt biometric technologies, by incorporating it into their banking systems.[16]

Middle Eastern and Asian banks, unlike their European and American counterparts, are jumping into the ban wagon of fingerprint authentication for online Banking Systems.[17] These facts are indicative of a future trend towards a massive adoption of fingerprint authentication for banking systems, hence the interest of this author to carry out a project in this area.

A well designed and implemented system that uses fingerprints for authentication and access control can ensure valid authentication and access control of users as well as non-repudiation by users.

Mutual authentication between the server and the client through server and client side digital signatures and certificates respectively will also likely reinforce the security of online banking systems.

Client side authentication for such systems can be reinforced further, by incorporating user fingerprints into the client side digital signatures.

---

[15] See for example Prophiserelease , GENERAL PRESS RELEASE OF RESEARCH FINDINGS (Public Release Date: August 21, 2007. See also "Fingerprint recognition technology seen as the most effective underutilized security enhancement to Internet banking", available at: http://www.prophis.com/pr/iBanking5_aug21.pdf , accessed 04-11-2010 . See also MyBank Tracker October 22nd, 2010 "Fingerprint ID Verification Preferred by Most Credit Card Users", available at: http://www.mybanktracker.com/bank-news/2010/10/22/63-consumers-choose-fingerprint-forms-identity-protection/ accessed 04-11-2010.

[16] Source: "First Biometric Fingerprint-Scanning ATM Machine installed in Poland." THE FORENSIC NEWS BLOG, Providing the latest forensic news from across the world of forensic science, published Thursday, 3 June 2010, available at:

http://forensic-news-blog.blogspot.com/2010/06/first-biometric-fingerprint-scanning.html accessed 10-11-2010

[17] Source: "Banque Saudi Fransi Deploys Biometric Authentication" By Anne Rawland Gabriel February 03, 2009 , available at :

http://www.banktech.com/risk-management/showArticle.jhtml?articleID=213001004 accessed 10-11-2010. See also the article " Bank of Central Asia Selects Identix Fingerprint Biometric Authentication Technology for Teller Verification of Electronic Transfers", available at : http://ir.l1id.com/releasedetail.cfm?ReleaseID=208701 accessed 10-11-2010.

The cost of both hardware and software from existing fingerprints authentication providers is a downside to the use of fingerprint authentication on a large scale over the internet.[18]

Coercion of users to login into a system is possible irrespective of the type of authentication system. However, we believe that systems can be designed and implemented in such a way that users of fingerprint authentication for online banking will send secret alerts to competent authorities in such cases.

1.2 The Problem Formulation

"How will online banking be made more secure by incorporating user fingerprints into client side digital signatures, through a pro-active design and implementation strategy?" (Static, dynamic, black box as well as white box testing technique will be used in order to ensure that these requirements are validated.)

In addition to the problem formulation, the formulations of sub-questions have also been carried out. These sub-questions are not necessarily requirement statements. Rather, they are secondary elements of the thesis statement. This author has identified the following sub questions presented below.

1.2.1 Research Sub-questions

- How will the system read user finger prints? (The validation of this requirement will be achieved when either virtual fingerprints or actual fingerprints, are read and actually stored on the client as well as server sides respectively.)

- Will the fingerprint reading application be built from scratch or will a pre-existing finger print reader and software be more appropriate? (After exploring both options the eventual choice will be made for that which will be most affordable and appropriate for this project.)

[18] Source: Webopedia website article "How Fingerprint Scanners Work", available at http://www.webopedia.com/DidYouKnow/Computer_Science/2004/fingerprint.asp , Posted: 10-01-2004 , Last Updated: 08-31-2010, accessed on 03-11-2010

- How will the fingerprint data be stored? (Validation for this sub question will be achieved with the permanent storage or reading of fingerprint data from the client and the transmission of a matching fingerprint template to the server, for eventual comparisons.)

- How will the fingerprints be used for authentication? (Validation for this requirement will be through black boxing testing of the system, for fingerprint authentication to a matching profile.)

- How will users' fingerprints be incorporated into client side digital signatures? (Validation for this requirement will be through black boxing testing of the system for fingerprint authentication to a profile which has matching fingerprint data, incorporated to its client side digital signature.)

- How will the implementation of client side authentication and client side digital signatures be made more convenient for users? (A portable external device, rather than the hard drives of client computers will be used for the storage of client side digital certificates in order to validate this requirement.)

- How will a large scale use of fingerprint authentication for online banking be made cost effective? (A comparative cost effective solution developed and analyzed, will be used to validate this requirement.)

- How will coercion of users to access online banking systems using fingerprint authentication, be avoided? (In order to validate this requirement, we can develop a fake process that mimics actual transactions and at the same time alerts the authorities, in the event of coercion.)

- How will the system guarantee that a given set of fingerprints come from a living person? (For this project, the only means that will be used to validate this requirement will be through the use of existing software and hardware, with such capabilities.)

- How will the mitigation of user privacy concerns related to the use of their biometric data be ensured? (If the system actually guarantees that user finger print data or templates from such data are transmitted and stored using strong encryption, then these concerns can be watered down. Similarly there should be some form of written guarantee to users that the use of their finger print data, will only be for the intended purposes.)

Similar research on a system that uses fingerprint authentication for online banking, according to Sky News, is being carried out by Siemens. This research takes into consideration some of the problems identified above. According to this source[19],

> *"Current online banking systems rely on PIN security for customers to identify themselves. Under the new system, the customer would identify themselves through their online identification, their online banking card and their fingertips. The user can also select a 'panic finger' to use if they are forced to log in under duress, when the bank will only pretend to carry out the transaction."*

There is also research on the incorporation of fingerprints authentication into current Automatic Teller Machine (ATM) platforms.[20]

Seven separate parts make up the methodology for this project.

The first part focuses on development methodology, while the second part focuses on research methodology. On separate paragraphs in this section on methodology, there shall also be the presentation of features of the project that must be implemented, those that may be implemented, presentation of the limitations, the scope of the project, as well as reflections that this author will have at the end of this project.

[19] Source: Fingerprint ID For Online Banking" Wednesday August 06, 2008 available at : http://news.sky.com/skynews/Home/Technology/Revolutionary-fingerprint-identification-for-online-banking-is-tested-by-Siemens/Article/200808115071356

[20] Source: IAMOT conference " ATM & BIOMETRICS: A SOCIO-TECHNICAL BUSINESS MODEL" , By Mario Yanez, Jr. University of Miami, School of Business Administration, CIS Department And Annuar Gomez University of Miami, School of Business Administration, MBA Program , available at : http://www.iamot.org/conference/index.php/ocs/4/paper/viewFile/1104/479 accessed on 10-11-2010

2 Development Methodology

Development methodology for this project will follow an object oriented approach to software development. Object oriented design tools will be used for this project.

The development process will be based on a spiral model of software development that will be a hybrid between the Unified Process (UP) of software development and some elements of an agile method like the Extreme Programming (XP) (For example test first, small releases of business value, code refactoring concepts etc., respectively).

The following iterative and incremental steps will be carried out during the software development process:

- Analysis – this phase will cover the analysis of the problem domain, in order to formulate the requirements for the system. The focus in the analysis will be on security requirements. The analysis will also take into account the cost and the complexity for developing and deploying such a system.
- Design – The overall architecture for the online banking system using fingerprint authentication will incorporate security requirements as well as test designs that take into to consideration the problem formulation and the sub questions. Object oriented design tools will be used for the design process. The design shall also make use of relevant software design patterns. The design shall basically follow a component based development approach.
- Implementation – the implementation of the system shall be based on the Microsoft .Net 4.0 Framework using the C# 3.5 programming language within the framework. The integrated Development Environment (IDE) for the implementation shall be the Microsoft Visual Studio 2010 and the Relational Database Management System (RDMS) to be used shall be Microsoft SQL Server Express 2008 that is inbuilt in the IDE. Throughout the project, a version control tool will be used to track the implementation and the documentation.
- Testing – Unit, component, integration, and system as well as acceptance tests respectively will be analyzed, designed and implemented as much as possible. These tests shall be carried out using appropriate testing tools. These tests will also have to ensure that the security requirements of the system are met. After each of the iterative test phases, mitigation proposals shall made against

vulnerabilities and further design and implementation will be carried out in order to mitigate such vulnerabilities.

2.1 Research Methodology

In addition to architecture and implementation of the online banking system, the research related area shall be to explore and implement the possibilities of incorporating user fingerprints into client side digital signatures and to create a new certification mechanism where a certification authority explicitly certifies the identity of the owner of fingerprint used for digital signatures.[21] The research will also include the possibilities of implementing a single fingerprint reading process for login and for digital signatures.

In parallel with the development Methodology, this author shall use qualitative research methods to identify, analyze , criticize and reflect on the problem area in order to identify the specific issues for the case study identified for the research;

During this research process, qualitative analysis of contemporary online banking systems will be explored. The focus of the qualitative analysis will be on security vulnerabilities.

Data sources for the research process will be from the internet as well as from Libraries. However, this author will not carry out any field work with any bank; rather he shall make use of his personal banking experience with some Commercial Banks in Denmark.

The problem formulation or research question and related sub questions, will be formulated based on the findings from the problem area.

The use of the word “system” in this report refers to: all hardware, software and networks with which the user interacts using his PC.

[21] Similar Research Work: You Lina, , Xu Maozhib and Zheng Zhimingc "Digital signature systems based on smart card and fingerprint feature" Journal of Systems Engineering and Electronics Volume 18, Issue 4, December 2007, Pages 825-834; Abstract available at: http://tinyurl.com/25be4tk

2.3 What must be carried out

- The entire: problem area, problem formulation, analysis and design processes described above must be carried out.
- With respect to implementation, this must at least be carried out to allow for secure authentication to a mock online banking system through the use of some form of virtual finger print reader. This authentication should allow for the possibility to carry out some mock online banking transactions.
- Unit, component, integration, system and acceptance tests respectively will be analyzed, designed, implemented and carried out as already described under testing above.
- Deployment basically will be in the form of the DLL of the system in a compressed file format.
- With regards to the research element of the project the implementation of a system that: incorporates virtual finger prints into the client side digital signatures, provides an innovative certification regime that includes fingerprint authentication, and a single fingerprint reading process to allow for login and digital signatures, should be designed and implemented to a reasonable extent.
- The latest Microsoft Technologies in the .NET 4.0 Framework and Visual Studio 2010 as the IDE will be used.

In order to validate these features that must be carried out, this author will use the validation processes already described under the problem formulation and sub questions section above.

2.4 What may be carried out

- The system may be developed further to make use of fingerprint hardware and software in order to read and store actual user finger print data that will be used for authentication and client side digital signatures.
- The entire system may also be eventually deployed to a remote web server production environment.
- Secure web services, may also be created for the application.
- Efficient performance of the system may also have to be ensured.

2. 5 Scope of project

Though through online banking several services can be offered, this project shall be limited to a prototype banking system that will be used only for money transfers and for online payments.

2.6 Limitations

Work in this research area is not very public. For this reason, the work carried out in the problem formulation and in the requirements analysis sections respectively, will be this author's best effort and are not guaranteed to be accurate. The main reason for this is because this project is not carried out with the collaboration of any major banking institution.

2.7 Reflections

At the end of the process this author will present, reflections on the lessons learned, limitations experienced and recommendation for further research.

2.8 What will be handed in

The following items will be handed in:

- Three hard copies of the documentation of the entire project;
- Three CDs containing the digital version of the project documentation, the source code for the system in a compressed file format, and instructions on how to test the system.

3 Requirement Analyses

This section of the report will cover the software research and development plan in terms of the customized methodology, the phases of development and time boxing.
This chapter will also cover the actual process carried out during the analysis phase of the project.

3.1 Methodological approach

The entire process of the development of this system will follow an approach in which a plan for the process will be established. Thereafter, the actual process will be documented and the final phase will be the reflections.
The plan for the process will be presented in this chapter on analysis; while the actual process and reflections will be presented as they are being applied in the course of the project.
In this project also, the author has to play roles both as the customer and developer respectively and may also get feedback from a supervisor in addition.

3.1.1 Methodological Plan

The methodological approach this developer plans to use in this software development process will be a hybrid approach covering the two methodological model paradigms commonly used by software developers: Prototyping and the Spiral Model. The spiral model will be focused on a combination of aspects of Extreme Programming (XP) approach and the Unified Process (UP) approach.
The predominant methodological approach will be to a large extent the UP approach followed by some elements of the XP approach that the developer finds useful. Elements of Prototyping which the developer finds useful will also be part of the development process.
In a nutshell, this developer plans to make use of a methodological approach which takes the best out of all the above mentioned methodological paradigms and even intends to make some methodological improvisations of his own, in the course of the development process if need be.

3.2 Inception or planning game phase

In keeping with the hybrid methodological approach adopted for this report the first activity in the development process is the inception (in UP parlance) which has a nuance with the planning game (in XP).

The analysis in the problem formulation presented in section 1 s*upra*, is indicative of the vision for the system as well as illustrates the fact that there are both business and research requirements respectively, which need to be further analyzed, designed and implemented. The business case has been made and there is a clear vision of the scope for the project.

This section of the report will cover the use- case model which is a description of the functional requirements. During this process, the names of the use cases will be identified and work will start on the detailed analyses of the use cases or user stories in XP parlance.

This section of the report will also present the supplementary specification, which is a description of non-functional requirements that will have a major impact on the architecture and implementation of the system.

This section will also have the test analysis that will include the risk list and risk management plan. This plan will describe the business, technical, resource and schedule risks respectively and will make proposal for their mitigation.

In this section, prototypes and proof of concepts will also be carried out in order to clarify the vision and validate key technical ideas.

The concepts described above under the inception phase are akin to the exploratory phase in the planning game of the XP approach.

The inception phase under UP in a similar manner to the commitment phase under XP requires that the process be risk driven, architecture centric and use case driven. This implies that the choice of the parts of the system that have to be built first, need to be chosen on a balance between business value, technical risks and the core parts of the system. For this reason use cases and security requirements will have to be ranked in order of priority.

In order to carry out the analyses tasks outlined above, it will be instructive to present the system definition.

3.2.1 System definition

System definition is a concise description of a computerized system, expressed in natural language.[22]

The approach to be used for the formulation of the system definition here is based on the FACTOR criterion.[23] Each of the six letters refers to key elements in the system definition.

F represents Functionality. Functionality here describes the system functions that support the application domain tasks.

A represents the application domain. These are the parts of an organization that administer, monitor or control a problem domain[24].

C represents Conditions. This describes the conditions under which the system will be developed and used.

T represents Technology. Here a description is presented on the technology used to develop the system as well as the technology on which the system will run.

O represents Objects: These are the main objects in the problem domain.

R represents Responsibility: This covers the systems overall responsibility, in relation to its context.

The following FACTOR criterion, have been identified for the system to be developed in this project:

- Functionality: A web based system that uses fingerprint authentication for online banking.
- Application domain: automated virtual banking services for customers.
- Conditions: The context for development is a school project with one student and one supervisor. The system developed and its documentation will be used for the evaluation of a final master thesis.
- Technology: A PC with contemporary features and in addition, more technical features like the .NET 4.0 Framework and Visual Studio 2010 IDE.

[22] Mathiasesen Lars, Munk-Madsen Andreas, Nielsen Peter Axel and Stage Jan " Object Oriented Analyses and Design " Marko Publishing ApS, Aalborg, Denmark 2000 at pg. 24

[23] See more on FACTOR criterion *ibid.*

[24] The Problem domain is that part of a context that is administered, monitored or controlled by a system. See *ibid* at pg 6.

- Objects: User Profile, account, virtual bank branch, transactions, alert and payments.

3.2.2 Functional requirements

The functional requirements for this system will cover the use cases; the security risk analysis and the test requirements.

3.2.2.1 Use Cases.

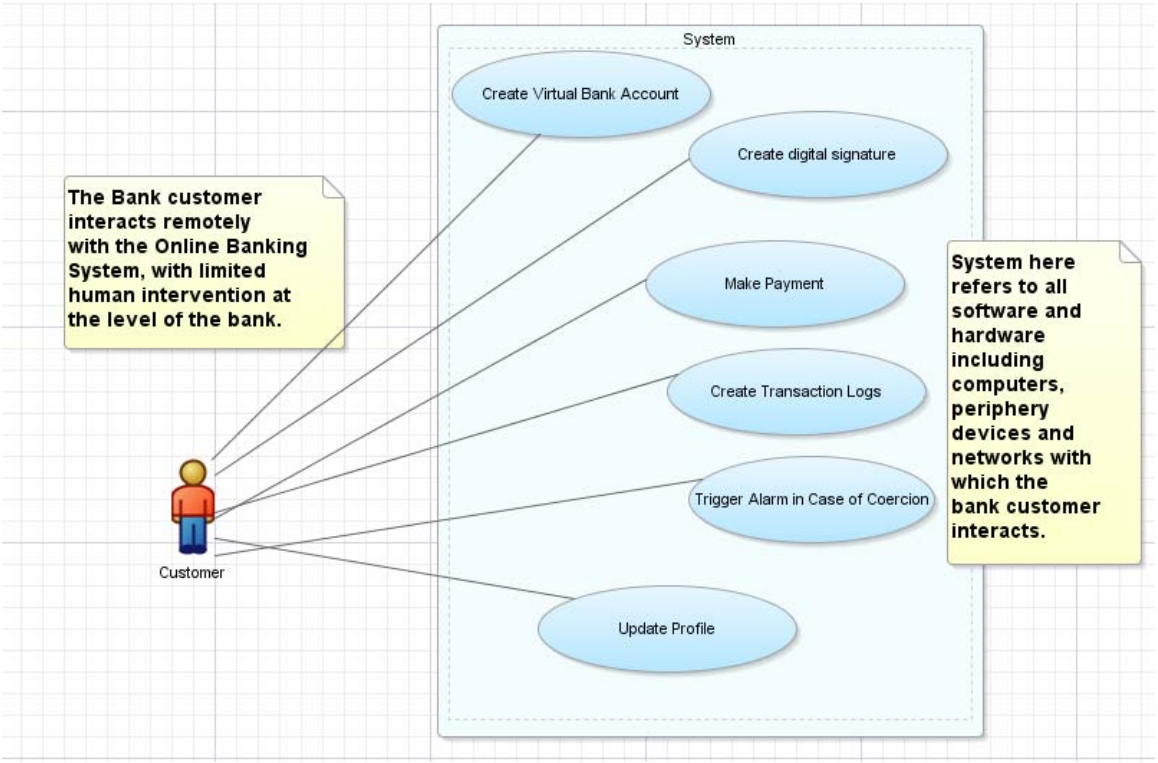
Use Cases are used here to describe the most important sequence of scenarios between the Customer and the System.

Under the Use cases the most important alternative scenarios will be presented for the respective Use Cases.[25]

The following use cases have been identified for this application:

- Virtual Account creation
- Digital signature creation
- User payments
- Transaction log
- Coercion trigger and
- Profile update.

[25] Less important alternative scenarios will be handled in the design and implementation through tests and mitigation as well as error handling.



Use Case Diagram describing the main types of interactions between the customer and the online bank.

Use Case UC0: Digital Signature Creation

Scope: Online banking application

Level: User goal

Primary Actor: Bank customer

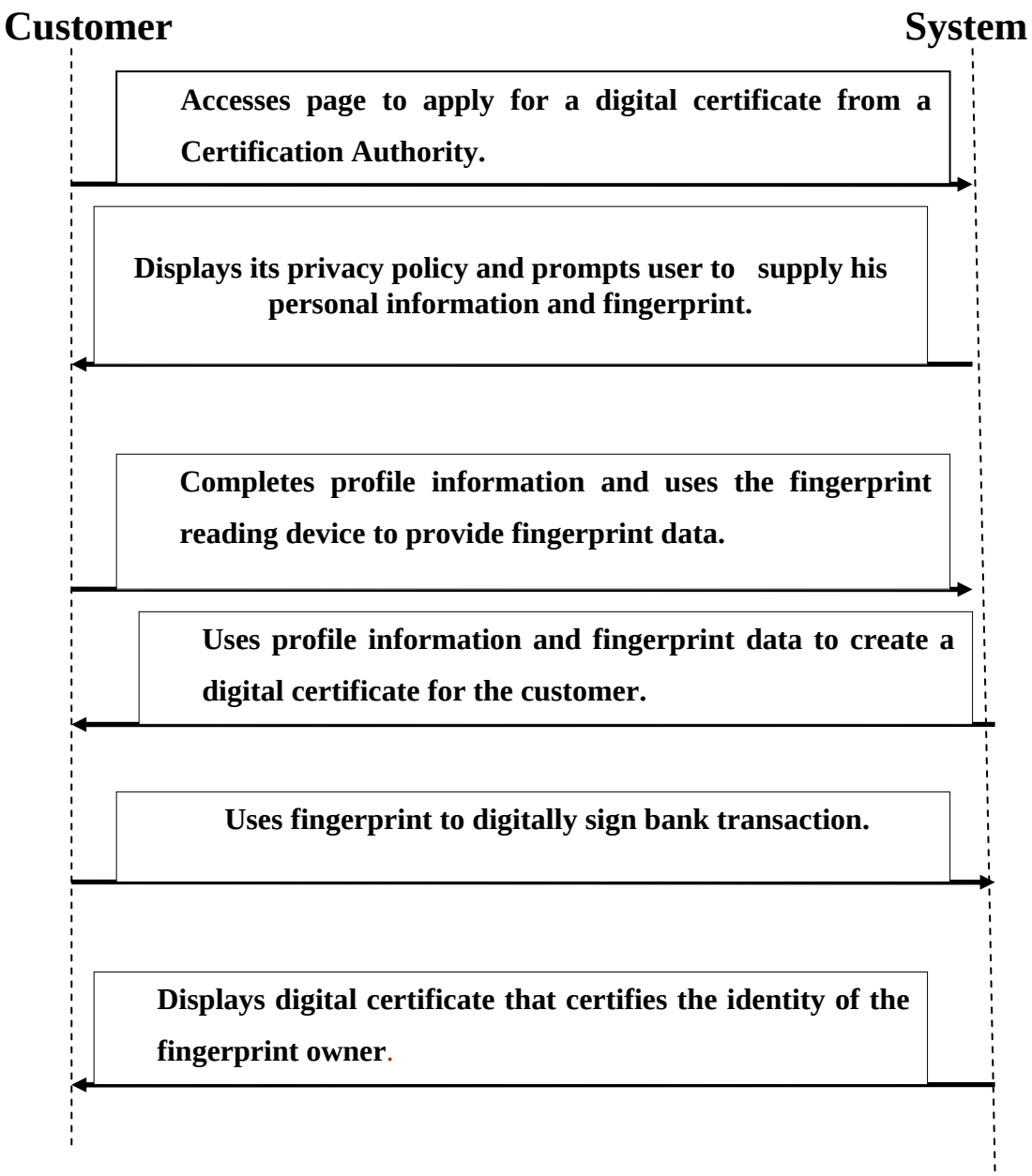
Stake holders and interests:

- Bank customer: wants efficient, secure, non-privacy intrusive and user friendly digital signature virtual bank account creation process.
- Bank: wants a mechanism that guarantees the identity of its online customer's through the use of customer's fingerprints. User privacy protection should also be guaranteed.
- **Preconditions:** In order to create a digital signature using fingerprint data, the customer obtains a USB fingerprint enrollment and reading device as well as relevant software for its functioning, from the bank.
- **Success guarantee (or Post conditions):** Users fingerprint data and other profile information are securely transmitted and stored by system and user obtains a digital signature that incorporates his fingerprint data. A digital

Certificate including fingerprint owner's identity should also be available to certify the signature.

Main Success Scenario:

1) Customer accesses page to apply for a digital certificate from a Certification Authority.
2) System displays its privacy policy and prompts user to supply his personal information and fingerprint.
3) User completes profile information and uses the fingerprint reading device to provide fingerprint data.
4) System uses profile information and fingerprint data to create a digital certificate for the customer.
5) User uses fingerprint to digitally sign bank transaction.
6) System displays digital certificate that certifies the identity of the fingerprint owner.



System Sequence Diagram of Digital Signature Creation Use Case.

Alternative Scenarios for Use Case 0:

- **Scenario 3 fingerprint not read by system.**

  3a) System prompts user to swipe finger once more on scanner.

  3b) Return to scenario 4.

Use Case UC1: Virtual Account Creation

Scope: Online banking application

Level: User goal

Primary Actor: Bank customer

Stake holders and interests:

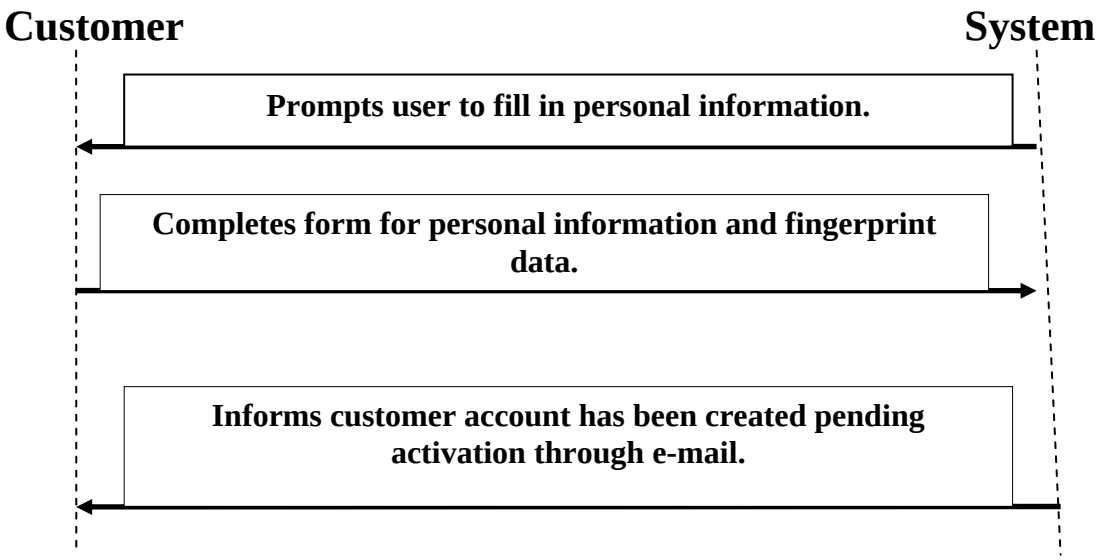
- Bank customer: wants efficient, secure, non-privacy intrusive and user friendly virtual bank account creation process.
- Bank: wants to cut cost by offering automated and secure virtual account creation for customers with user privacy protection guarantees.

Preconditions: In order to create an online bank account, the customer accesses the bank web site which has a digital certificate and is connected over SSL. A priori, the customer must have read the help page on how to create the account.

Success guarantee (or Post conditions): Users fingerprint data and other profile information are securely transmitted and stored by system and user account is created.

Main Success Scenario:

1) The system prompts the customer to enter personal information for account creation.
2) Customer completes form for personal information including the same fingerprint that had been used for digital signatures.
3) The system, informs user that account activation link has been sent via e-mail, confirms that profile has been successfully registered, informs user that the login fingerprint data will be used as part of the user's digital signature as well as for subsequent logins.



System Sequence Diagram of Virtual Account Creation Use Case.

Alternative Scenarios for Use Case 1:

- **Scenario 5 fingerprint not read by system.**

  2 a) System prompts user to swipe finger once more on scanner.

  2 b) Return to scenario 3.

Use Case UC2: User Payments

Scope: Online banking application

Level: User goal

Primary Actor: Bank customer

Stake holders and interests:

- Bank customer: wants efficient, secure and user friendly online banking services.

- Bank: wants to cut cost by offering automated and secure online banking services to customers.
- Business: wants to carry out banking transactions with customers, through the customer's online account.
- Individuals: want to carry out online banking transactions with customers.

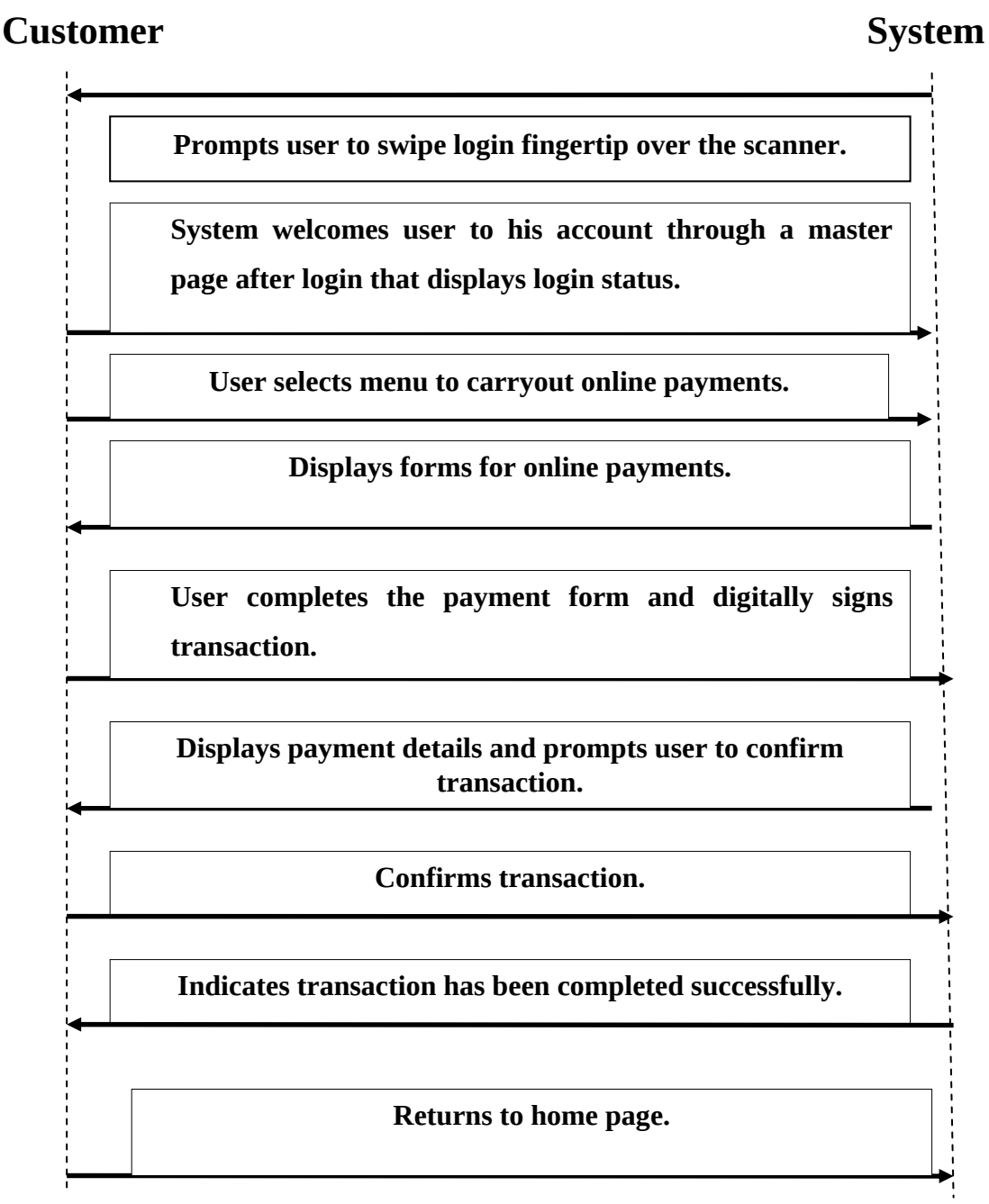
Preconditions: The user has activated the account through the link sent by e-mail, to the site that has a digital certificate and is connected over SSL. The user must have also credited the account with some money.

Success guarantee (or Post conditions): User is securely and uniquely authenticated into his account by swiping the login fingertip to the system. The transaction should be over SSL. The Bank's digital certificate should be available to the client and the client's digital certificate should be available to the bank server.

User carries out banking transactions securely and successfully and gets transaction confirmation message.

Main Success Scenarios:

1) The system prompts the user to swipe login fingertip over the fingerprint reader
2) System welcomes user to his account through a master page after login that displays login status.
3) User selects menu to carryout online payments.
4) The system displays the form for online payments.
5) User completes the payment form and digitally signs transaction.
6) System displays payment details and prompt's user to confirm transaction.
7) User confirms transaction.
8) System indicates transaction has been successfully carried out and prompts user to return to home page.
9) User returns to home page.



System Sequence Diagram of the Payments Use Case.

Alternative Scenarios for Use Case 2:

- **Scenario 1: login attempt with fingerprint unsuccessful.**

  1 a) System gives user ten attempts to swipe login finger again if user is sure he has credentials to log into account.[26]

  1 b) User successfully logs into account.

[26] If this fails he will get an error message that he is not authorized to log into the account, otherwise, he should contact the system administrator.

1 c) Return to scenario 2.

Use Case 3 UC3: Transaction log

Scope: Online banking application

Level: User goal, business goal.

Primary actor: Bank customer

Stake holders and interests:

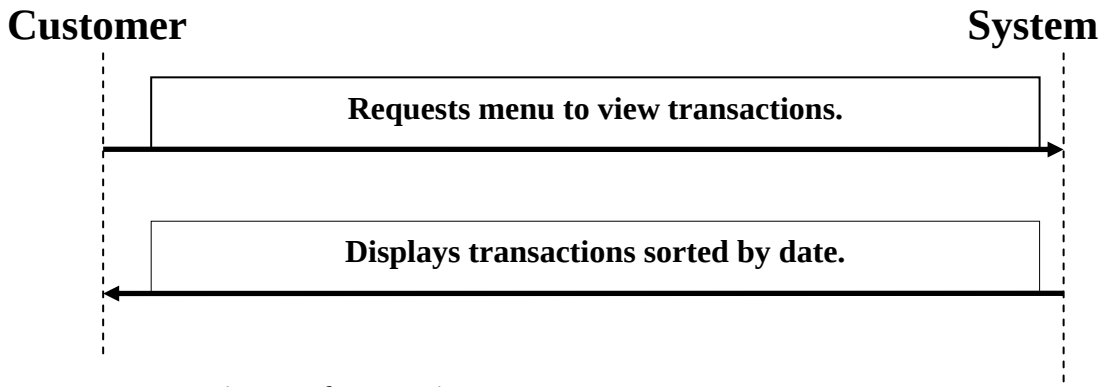
- Bank: intends to ensure that all transactions are atomic, consistent, isolated and durable [27]
- Bank customer: wishes to see transaction information at the end of every transaction as well as the history of transactions.

Preconditions: The user must have completed a transaction.

Success guarantee (or post conditions): The system displays an accurate version fulfilling the ACID properties for every transaction.

Main Success Scenario:

1) The user, requests menu to view transactions
2) The system, displays a list of all transactions related to the account, sorted by date.



System Sequence Diagram of Transaction Log Use Case.

Use Case 4 UC4: Coercion trigger

Scope: Online banking application

Level: User goal

Primary Actor: Bank customer

Stake holders and interests:

[27] So called ACID properties of the database are maintained.

- Bank customer needs to be assured that in the event of coercion to log into his account, there will be the possibility to trigger an alert to competent authorities, while at the same time a mock transaction will be taking place.
- The Bank intends to assure customers that the system has security mechanisms against coercion.

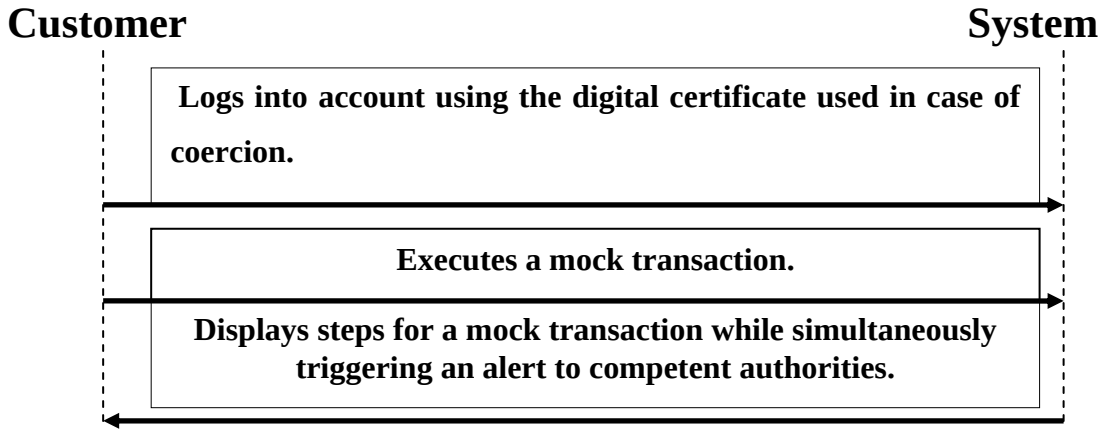
Pre-conditions: The user must have created another digital certificate to use in the event of coercion.

A situation occurs in which the user is coerced to log into his account.

Success guarantee (or post conditions): When the coercion digital certificate is used for login, an alert is sent to competent authorities while at the same time a mock transaction will be completed and thereafter the account is blocked.

Main success scenarios:

1) The bank customer logs into his account, using the digital certificate used in case of coercion.
2) The bank customer executes a transaction.
3) The system displays steps for a mock transaction while simultaneously triggering an alert to competent authorities.



System Sequence Diagram of Coercion Trigger Use Case.

Alternative scenarios to Use Case 4

- **Scenario 1: login attempt with coercion alert fingerprint unsuccessful.**

  1 a) System gives user ten attempts to swipe login finger again if user is sure he has credentials to log into account.[28]

  1 b) User successfully logs into account.

  1 c) Return to scenario 2.

Use Case 5 UC5: Profile Update

Scope: Online banking application

Level: User goal

Primary Actor: Bank customer

Stake holders and interests:

- Bank customer needs the possibility to update his profile
- The Bank intends to offer the possibility for customers to remotely update their profiles.

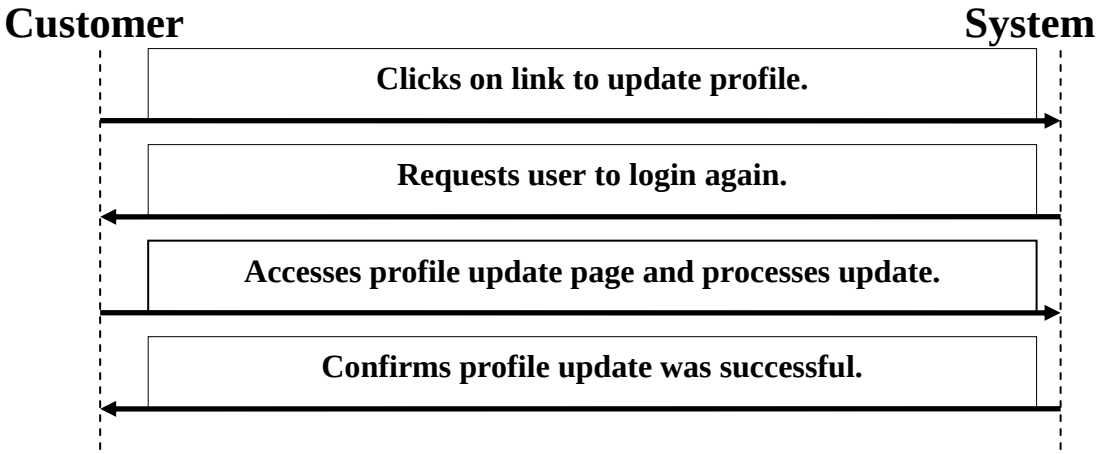
Pre-conditions: The bank customer is logged into his account.

Success guarantee (or post conditions): The bank customer successfully updates his profile and the update is persisted in the system.

Main success scenarios:

- User clicks on link to update profile
- Before any profile update, the system requests the user to login again using his login credentials.
- The user accesses the profile update page and processes the update.
- The system confirms that the profile update was successful.

[28] If this fails he will get an error message that he is not authorized to log into the account, otherwise, he should contact the system administrator.



System Sequence Diagram of Profile Update Use Case.

3.2.2.2 Risk Analysis

The strategy for risk management in this project will follow a risk management framework.

A risk management framework (RMF) is a high-level approach to iterative risk management that is deeply integrated throughout the software development lifecycle (SDLC) and unfolds over time. The basic idea is simple: identify, rank, track, and understand software security risk as it changes over time.[29]

The RMF consists of the five fundamental activity stages as follows[30]:

- Understand the business context,
- Identify the business and technical risks,
- Synthesize and prioritize the risks, producing a ranked set,
- Define the risk mitigation strategy,
- Carry out required fixes and validate that they are correct

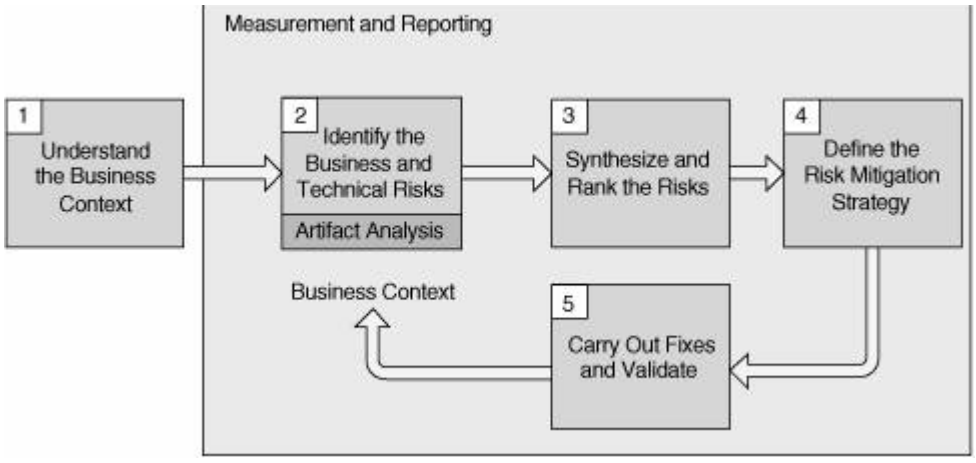


Diagram describing the Risk Management Framework[31]

[29] Source: McGraw, Gary *Software Security - Building Security in*, Addison Wesley Professional January 23, 2006 at Chapter 2

[30] *Ibidem*

Software best practices on how to work security engineering into requirements, architecture, design, coding, testing, validation, measurement, and maintenance can be applied together with the RMF.[32]

This model of best practices will be applied for this project in an iterative manner, which means that the best practices will be cycled through more than once as the software development evolves.

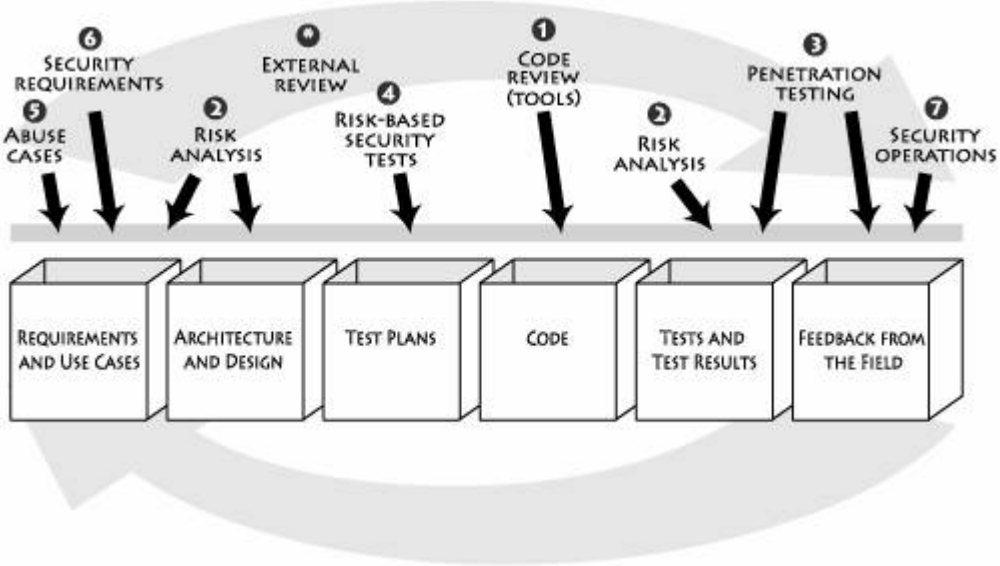


Diagram describing the elements of software security best practices[33]

The risk analysis for this project focuses on the business goals[34]. These business goals have been used to: identify technical risks associated to business risks and the level of the risk impact and likelihood in terms of: high (H), medium (M) and low (L).[35]

For this project, abuse cases are considered as technical risks.

The table below describes the business risks, the technical risks[36] , the mitigation proposals for these risks as well as the ranking of the respective risks in terms of *High, Medium or Low*.

A rank of *high* may cause heavy losses or great damage to the reputation of the company. A rank of *medium* may indicate heavy loss of tangible assets or resources

---

[31] Copied from McGraw, Gary *Software Security - Building Security in,* Addison Wesley Professional January 23, 2006 at Chapter 2.
[32] *Ibidem Chapter 3*
[33] Copied from McGraw, Gary *Software Security - Building Security in,* Addison Wesley Professional January 23, 2006 at Chapter 3
[34] In this case are predominantly the use cases.
[35] Source: McGraw, Gary *Software Security - Building Security in,* Addison Wesley Professional January 23, 2006 at Chapter 2.
[36] Risk IDs have been created for each type of technical risk

and may harm the reputation of a company, while a *low* rank may cause some financial losses or losses to some tangible assets or resources of the company and may as well have some effect on the company's reputation. [37]

A rank of *high* may be critical to the existence of project and the company and therefore must be implemented immediately. [38]

Rank of *medium* may be very important for the company and may have negative effects on high ranked business goals. Medium ranked risks should be implemented within reasonable time.[39] Implementation of medium ranked risks can be carried out in subsequent releases of the application.[40]

A rank of *low* affects only a small portion of company's revenue and may have some impact on the company's reputation. A risk ranked as low may or may not be implemented, depending on the business choice.[41]

[37] Source: McGraw, Gary *Software Security - Building Security in,* Addison Wesley Professional January 23, 2006 at Chapter 2.
[38] *Ibidem*
[39] *Ibidem*
[40] This author's idea.
[41] Source: McGraw, Gary *Software Security - Building Security in,* Addison Wesley Professional January 23, 2006 at Chapter 2.

Business Risk	Technical risk	Mitigation Proposal	Risk Level
I) Difficulties to Sign up or Login	**-rid1:** problems with fingerprint data registration and storage. **-rid2:** problems to use fingerprints for authentication.	**For rid1:** - Ensure that fingerprint data is securely: transmitted and stored - Ensure that each unique fingerprint is mapped to a unique fingerprint data used for authentication. **For rid2:** -Ensure fingerprint reading software and hardware function properly.	H
II) Invasion of users' privacy and sensitive data.	**-rid3:** user login credentials can be captured and replayed. **-rid4:** Users' worried about abuse of their fingerprint data as well as the possibility of false acceptance[42].	**For rid3:** -Ensure that fingerprint data is encrypted during transmission and storage. **For rid4:** - Actual fingerprints should never be stored on the servers rather each fingerprint used should have a one to one mapping with a related data stored on the server. System should ensure usage of live fingerprints.	H
III) Errors during usage	**-rid5:** Stack traces printed for exceptions may be used to hack system. **rid6:** Cascading failures	**For rid5:** - Display error pages and log errors to secure pages on server side. **For rid 6:** - Compartmentalize development.	H
IV) Release delays	**-rid7:** Non compatible developed modules. **-rid8:** Several implementation bugs. **-rid9:** Some modules not ready.	**For rid7:** - Regular integration tests between modules. **For rid8:** - Avoid complex code. **For rid9**: - Compartmentalize development.	H
V) Brief system interruptions	**-rid10:** System temporarily shut down during updates.	**For rid10:** -Possibility of concurrent updates without system interruptions.	L[43]

Table describing the risk analysis for this project.

[42] False acceptance is the instance of a security system incorrectly verifying or identifying an unauthorized person.

[43] Brief system interruptions for maintenance purposes, can be tolerable.

Table describing the risk analysis for this project, continued.

Business Risk	Technical Risk	Mitigation Proposal	Risk Level
VI) Difficult to modify system.	**-rid14:** Update difficulties	**For rid14:** - Make system extensible.	H
VII) Portability of system	**-rid15:** application inaccessible from any computer connected to the internet from anywhere.	**For rid15:** - Use external computer periphery device for user fingerprint authentication and client side digital signature.	M[44]
VIII) Complex system for users.	**-rid16:** Bad user interface implementation and complex usage procedures.	**For rid16:** Make simple user interface, and procedures user friendly.	M[45]
IX) Failed Acceptance	- **rid:17:** Failed unit, component, integration and user tests.	**For rid17:** continues testing at all levels of development.	H
X) Users feel insecure.	**-rid18:** User coerced to log into account	**For rid18:** Create an alert procedure and a mock transaction, in the event of coercion	M[46]
XI) System not cost effective	**-rid19:** development abandoned due to high costs.	**For rid19:** Adopt a cost effective development approach.	M[47]

3.2.2.3 Use Case Ranking

The ranking of the Use Cases in order of importance and priority for development, will be in keeping with the hybrid development methodology between UP and XP

[44] In the absence of a portable device to store to store user finger print data and client side digital signatures, both can as well be stored using a secure mechanism on the hard drives of a portable computer device.
[45] Features might be complex but a comprehensive help manual can make the process less complex to users.
[46] The risk is substantial but the likelihood will likely not be high.
[47] It will be difficult to have precise estimates on the development cost at the initial phases of the project.

software development methodologies as well as the Risk Management Framework that have been chosen for this project.

In the preceding sections of this chapter, the business context has been identified as well as the business and technical risks.

Initial mitigation strategies have been proposed under the risk analysis.[48]

In keeping with the hybrid development methodology described above, the Use Cases have been ranked below in terms of their architecture centric nature, their business value and the technical and business risks associated to them. The development process for this project will prioritize the Use Cases in terms of their importance.

With the foregoing in mine, the ranking of the Use Cases have been ranked according to their importance as follows:

1) Virtual Account Creation
2) Digital signature creation
3) User Payments
4) Transaction Log
5) Coercion Trigger
6) Profile Update

3.2.2.4 Test analysis

The test strategy throughout the software development Lifecycle (SDLC) for this application is going to follow an iterative "V" model according to which:

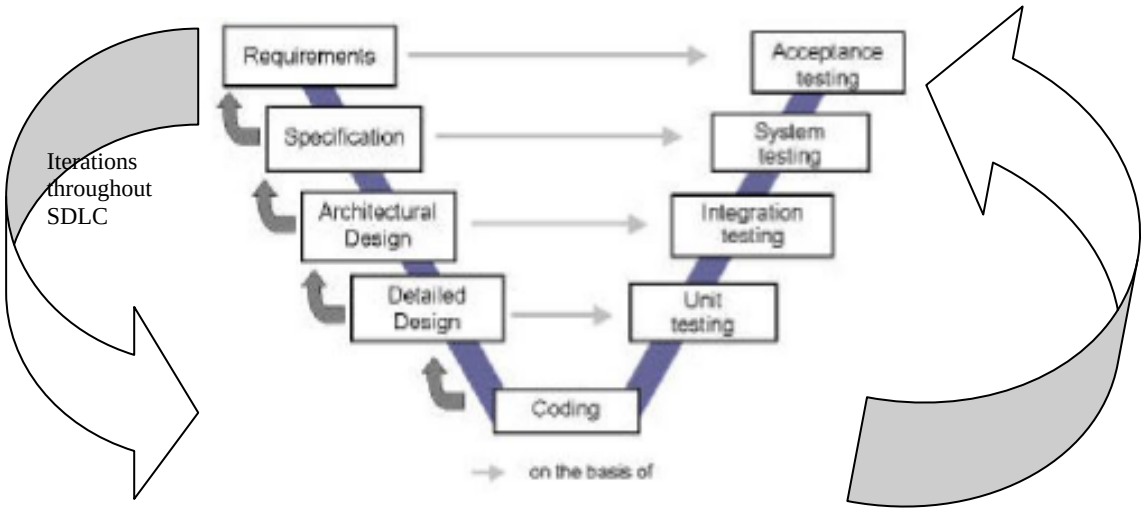
Compliance to user requirements will be satisfied by Acceptance tests;

Compliance to system requirements are going to be satisfied by System testing;

Architectural Design will be satisfied through integration testing and

Detailed Design for the coding will be backed by unit testing for each respective component.

[48] These mitigation strategies will be further developed in the design and implementation phases of the project.



The iterative test model "V" test model is illustrated by the diagram above.

3.2.2.3.1 Acceptance Test Plan

For all these test cases, expected results are not going to come from the code, but should rather come from actual pre-existing results for the test cases.

The acceptance test will be the final test for the application taking into consideration all the specificities of the requirements and overall functionality.

The pre-condition for the acceptance test will be based on the fulfillment of the elements of the system test.

The expected results for the acceptance tests should be derived through a means independent of the application to be tested and should satisfy the business goals for the system.

3.2.2.3.2 System Test Plan

The system test plan will cover tests at a higher level and will be more or less based on the testing of the steps in the Use Case description and risk analysis.

This test is carried out when all the components must have been integrated and tested.

The expected results for each of the steps in the Use Case description will be derived through a means independent of the application to be tested.

The tables below, summarizes test plans for system tests.[49]

[49] Only the most sensitive test plans will be outlined here. Tests will be developed and implemented further during the design and implementation phases.

Test case id Virtual Account Creation.	Scenario.	Information Validity.	Expected results.
TC1	Profile information and fingerprint data entered.	Parts of profile information or fingerprint *Not Valid*	Error message indicating the fields with invalid values.
TC2	Profile information and fingerprint data entered.	*Valid*	A query on any user's profile information should reflect the information entered and user should be able to login with login fingerprint and sign transactions, after account creation.
TC3	User swipes fingertip on scanner to create a digital certificate and login credentials.	Fingerprints from a non-living source or fingerprints wrongfully swiped over scanner	Error message "Non-human fingerprints or fingerprints wrongfully swiped. Please try again."
TC4	User swipes fingertip on scanner.	Live human fingerprints and correct swiping over scanner	Only the owner of a given fingerprint data should be able to log into his account and digitally sign transactions after the account must have been created.

Table summarizing Test Cases for Use Cases 0 and 1(Virtual Account Creation and Digital signature Creation)

Test Case ID	Scenario	Information Validity	Expected Result
TC5	User swipes fingertip on scanner.	Valid	Give access to account connected over SSL and uniquely mapped to the unique fingerprint login credentials.
TC6	User uses coercion digital certificate to sign a transaction.	Valid	Give access to mock account, connected over SSL and uniquely mapped to the login credentials and trigger's alert to competent authorities.
TC7	User Confirms Transaction.	Valid	Transaction completed ,and persisted in system And can be seen in the transactions log page.
TC8	User logs into account 24 hours after a coercion alert.	Valid situation after a coercion trigger.	The balance in the account reflects the balance prior to the coercion situation.

Table of Test cases for Use Case 2 (User Payments) and Use Case 4 (Coercion Trigger)

3.2.2.3.3 Integration testing

Integration testing covers the tests carried out when components of the system that are developed separately, are being systematically integrated to one another to complete the whole system.

Three components have been identified for this application. These are: the *Access Control Component*, the *Digital Signature Creation Component* and the *Bank Transactions Component* .

ii) The *Access Control Component*

It includes:

- all the scenarios in the Virtual Account Creation and Update Use Cases,
- The login process for the User Payments,[50]
- All other scenarios related to access.

A test to validate this component should be to ensure that test cases TC1 to TC6 above are passed.

ii) The Digital Signature Creation Component

[50] Scenario one.

It includes:

- All the scenarios in the Digital signature creation Use Case.
- Scenarios 5 in the Payments Use Case.
- All scenarios related to the alert trigger mechanism.
- The entire mechanism related to the creation of the mutual Public Key Infrastructure (PKI) and the connection over secure network.

A test to validate this component should be to ensure that test cases TC6, TC7 and TC9 above, are validated.

iii) **The *Bank Transactions Component***

These are scenarios not dependent on access control or the digital certificates for their implementation. They include:

- Scenarios 3,4,6,7 and 8 of the User Payments Use Case,
- All the scenarios involved in the Transaction Log Use Case,
- All the scenarios in the User Profile Update Use Case and
- Scenarios related to carrying out mock transactions for the Coercion Trigger Use Case, apart from scenarios related to access or digital signatures.[51]

A test to validate this component should ensure that the requirements for the scenarios above are met.

The integration test should ensure that there is a smooth integration between the three components above.

The precondition for this test should be acceptable unit testing of the codes of each respective component and acceptable test results for each component.

- The integration between the Access Control and Digital Signature components, should be such that the user logs into the system for a particular session just once with his login fingertip, gets unique access into his account and at the same time, can digitally sign transactions using his digital certificate.
- The Access Control component should be integrated with the Bank Transactions component such that only an authenticated user can have access into his account in order to carry out transactions, or to update his profile.
- The Digital Signature component should be integrated with the Bank Transactions component such that only the user with a digital certificate that

[51] Scenario 2 and part of scenario 3, for this Use Case.

contains the same fingerprint profile as that of the logged in user can digitally sign transactions for that account.

3.2.2.3.4 Unit Testing

A unit in software development is the smallest piece of code that can be tested. Unit testing in this case will involve a detailed testing of the units of code that can be tested within each component.

The components are made up of classes and the classes are made up of methods which have decision outcomes. The Unit tests will be based on the decision outcomes of the methods in the classes.

The expected results for each of the steps in the unit tests will be derived through a means independent of the application to be tested. A unit test passes, if all the unit test cases for each unit test pass their respective tests.

The next part of this report will be on the supplementary specifications for this application.

3.2.3 Supplementary Specifications

Supplementary Specification or Nonfunctional Requirements have to do with what a system is supposed to be, in contrast with the behaviour of the system. They have to do more with the qualities of a system. Such qualities include: Security, usability, testability, maintainability and extensibility.

It also has to do with tools, hardware and software that will be used for development and usage.

Because security is the central issue for this project, the main focus on the supplementary specifications for this project is on security.[52]

Other elements of supplementary specification will be taken into consideration during the Logical Design and the implementation phases of this project.

3.2.1.1 Security Specifications

The architectural infrastructure for security chosen for this project is based on the *Public Key Infrastructure* (PKI). In this part of the project, a background will be presented on the general features of PKI.

[52] The most important tools, hardware and software to be used have already been presented under the Development Methodology chapter.

In the chapter on design, a concise choice of requirements for the PKI for this project will be presented.

3.2.1.1.1 Background on PKI

PKI enables users of a basically unsecure public network such as the Internet, to securely and privately exchange data and money through the use of a public and a private cryptographic key pair that is obtained and shared through a trusted authority. The public key infrastructure provides for a digital certificate that can identify an individual or an organization and directory services referred to as Certification Authorities (CAs) that can store and, when necessary, revoke the certificates. [53]

The public key infrastructure assumes the use of public key cryptography, which is the most common method on the Internet for authenticating a message sender or encrypting a message. Traditional cryptography has usually involved the creation and sharing of a secret key for the encryption and decryption of messages. This secret or private key system has the significant flaw that if the key is discovered or intercepted by someone else, messages can easily be decrypted. For this reason, public key cryptography and the public key infrastructure is the preferred approach on the Internet. (The private key system is sometimes known as symmetric cryptography and the public key system as asymmetric cryptography).[54]

PKI implementation of a system can ensure confidentiality, integrity, authentication and non-repudiation.[55]

Digital signature is a related concept to asymmetric encryption.

Digital signatures are created using asymmetric cryptography, the approach on which digital signatures are based. Asymmetric Cryptography is distinguished by having two different keys which are created using algorithms like RSA[56] and AES[57] , a private key to encrypt messages and a public key to decrypt them. The cryptographic private

---

[53] Source: http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci214299,00.html accessed on 14-02-2011 .

[54] Source: *Ibidem*

[55] Source: JoelWeise - "Public Key Infrastructure Overview" SunPSSM Global Security Practice Sun BluePrints™ OnLine - Sunmicrosystems August 2001 available at page 9 http://www.sun.com/blueprints/0801/publickey.pdf accessed on 14-02-2011 accessed on 14-02-2011

[56] RSA is a public-key system designed by Rivest, Shamir, and Adleman . Source: http://www.cryptographyworld.com/algo.htm accessed on 16-02-2011

[57] This is the Advanced Encryption Standard (using the Rijndael block cipher) approved by NIST (National Institute of Standards and Technology in the USA). Source: http://www.cryptographyworld.com/algo.htm accessed on 16-02-2011

key K_0 (a suitable array of bytes) is used with an appropriate algorithm to transform the initial human-readable message into a different message that is encrypted. [58]

A second public cryptographic key K_1, which is related to the private one, is used to change the encrypted message back to its original decrypted form via a second related algorithm.[59]

The process of creating digital signatures starts with the hashing of data to be transmitted. The reason for performing hashing is to ensure data integrity. Hashing is simply a process whereby you calculate a hash code from some data. The generated hash code is mathematically derived and is unique and specific for the data it was derived from. If any byte changes in the data then a completely different hash code is generated. Hash codes are a means to check that data was received/read as it was sent/written, any accidental damage/modification or malicious changes are checkable using hashes.

Some hashing algorithms include MD5, SHA1, and SHA256. [60]

One of the problems with hashing is that it is wide open to man in the middle attacks. Without doubt hashing has its uses but in terms of sending data there is nothing stopping someone from intercepting the data, modifying it, and then resending the new message with a new hash. What the receiver gets is a message where the hash code matches the data, even though the data has been modified. The use of digital signatures can avoid this scenario.

In digital signatures public/private asymmetric keys are used. [61]

The two keys of asymmetric encryption are mathematically related to each other and one key can be used to verify that the encryption was done with the other key. With digital signatures the sender encrypts the hash using their private key while the receiver verifies the digital signature using the sender's public key. Of course since the public key is more freely available then anyone can verify the message's source. [62]

So, for example, Bob wants to send Alice some data and Alice wants to be able to check the data was unchanged and came from Bob. Bob creates the hash and encrypts

---

[58] Source: http://www.simple-talk.com/dotnet/.net-framework/beginning-with-digital-signatures-in-.net-framework/

[59] *Ibidem*

[60] Source: http://dotnetslackers.com/articles/security/Hashing_MACs_and_Digital_Signatures_in_NET.aspx

[61] *Ibidem*

[62] *Ibidem*

it into a digital signature using his private key. He sends the data and the digital signature over to Alice. Alice uses Bob's public key to verify that the digital signature was created using Bob's corresponding private key. If everything checks out then Alice knows the message hasn't been modified and that it came from Bob. [63]

Asymmetric key encryption by itself is not enough because it is necessary to trust the public key received. An attacker can deceive you by signing a message with his private key and send you a digitally confirmed message with its (related) public key, whilst pretending he is someone else.[64]

The public-key infrastructure (PKI) avoids this by utilizing a third-party entity, called Certification Authority (CA) that, under its responsibility, binds a public key to its owner. The binding occurs when the Certification Authority digitally sign a message that contains the public key and the identity of its owner. A digital certificate is obtained.[65]

With this mechanism, the recipient is sure that the message that she/he received is your message, because only you hold the private key that is related to the public, shared, key. This way, you digitally 'sign' your message.[66]

Asymmetric encryption has an overhead and not suitable for large texts. As a result, combination of two methods is employed for secure transmission on the web. HTTPS[67] and SSL[68] are using this combination. It starts with Asymmetric communication between client and server during which key is transferred in asymmetrically encrypted form. Once key is delivered it is then used to decrypt the symmetrically encrypted text.[69]

---

[63] *Ibidem*

[64] http://www.simple-talk.com/dotnet/.net-framework/beginning-with-digital-signatures-in-.net-framework/

[65] *Ibidem*

[66] *Ibidem*

[67] HTTPS (HTTP over SSL or HTTP Secure) is the use of Secure Socket Layer (SSL) or Transport Layer Security (TLS) as a sublayer under regular HTTP application layering. HTTPS encrypts and decrypts user page requests as well as the pages that are returned by the Web server. The use of HTTPS protects against eavesdropping and man-in-the-middle attacks. Source: http://searchsoftwarequality.techtarget.com/definition/HTTPS accessed on 17-02-2011

[68] The Secure Sockets Layer (SSL) is a commonly-used protocol for managing the security of a message transmission on the Internet. SSL has recently been succeeded by Transport Layer Security (TLS), which is based on SSL. SSL uses a program layer located between the Internet's Hypertext Transfer Protocol (HTTP) and Transport Control Protocol (TCP) layers. SSL is included as part of both the Microsoft and Netscape browsers and most Web server products. Source: http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci343029,00.html accessed on 17-02-2011

[69] http://tutorial.visualstudioteamsystem.com/details.aspx?item=132

When dealing with applications offering services like online banking as is the case for this project, authenticating the server as well as the client may become necessary in order to ensure a higher level of security.

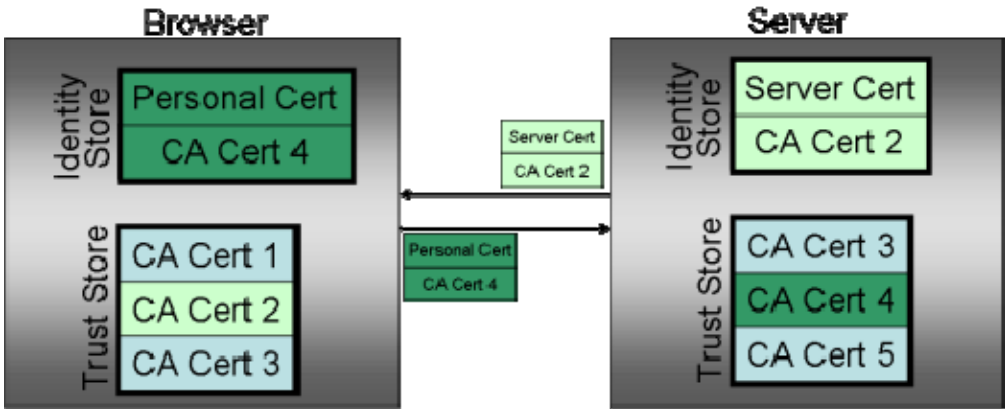
3.2.1.1.1.1 Mutual authentication

One way SSL is what is common in most cases.

One-way SSL simply means that only the server certificate passes from the server to the client but not the other way around. Thus, only the server is authenticated using the mechanism on PKI explained above.[70]

With mutual authentication, both sides pass certificates to each other to establish an SSL link, we have mutual authentication via two-way SSL. Both sides now know the identity of the other from their respective certificates. The diagram below shows what's required for two-way SSL or mutual authentication:[71]

Mutual authentication architecture shall be the approach to follow in the course of the development of security features for this project.



Two-way SSL comprises the one-way scenario along with the browser passing a personal certificate and its CA chain to the server. The server trusts the CA that issued the personal certificate.

[70] See http://monduke.com/2006/06/04/the-fifteen-minute-guide-to-mutual-authentication/ accessed on 17-02-2011

[71] Source: *Ibidem*

4 ARCHITECTURAL DESIGN

The architectural design for this project will cover:

- the choice of architectural pattern,
- the mutual authentication mechanism
- the access control mechanism
- the choice of database entities and classes
- the digital signature mechanism and the
- the risk mitigation design strategy

4.1 Architectural Pattern

The architectural pattern that will be predominantly used for this application, will be the Client-Server architectural pattern, but will also incorporate some elements of the Layered Architecture pattern as well as architectural improvisations from this author.

Essentially, the components in client server architecture are a server and several clients. The server has a collection of components that it makes available to the clients. This design is such that the server does not presume any knowledge about the client but the clients must know of the server and its operations. The server has the responsibility to provide what is common to the clients and usually this is done over a network. In such a case, it is the duty of the client to provide an interface for its users.[72]

The client server pattern can be viewed as a more general pattern for component architectural design.[73]

The layered architectural design pattern in its simplest form consists of several components designated as layers. The design of each component describes its responsibility as well as it's upwards and downwards interfaces. The downward interface describes which operations the components can access in the layer below and the upward interface describes the operations it makes available to the layer above. In a layered architecture a given component can be decomposed into vertical or horizontal subcomponents. [74]

---

[72] Mathiasesen Lars, Munk-Madsen Andreas, Nielsen Peter Axel and Stage Jan " Object Oriented Analyses and Design " Marko Publishing ApS, Aalborg, Denmark 2000 at pg. 197

[73] *Ibidem* at pg. 197

[74] *Ibidem* at pg.193

The architectural design for this application presumes that the servers of the client and server Certification authorities are lower layers components to the client and server components respectively, and provide them with their respective digital signatures capabilities and digital certificates. For reasons of simplicity, these components will not be included in the architectural design diagram.

The proposed design has two major components: The client and the server components which are generic in nature with the client providing the interface[75] to interact with the functionality provided by the server.

What is of interest here, is rather the components obtained when the server is decomposed horizontally into subcomponents. Such decomposition has led to the identification of three components:

i) An access control component

ii) A component holding the logic for bank transactions and

iii) A component that provides the logic for client and server digital certificates and digital signatures.

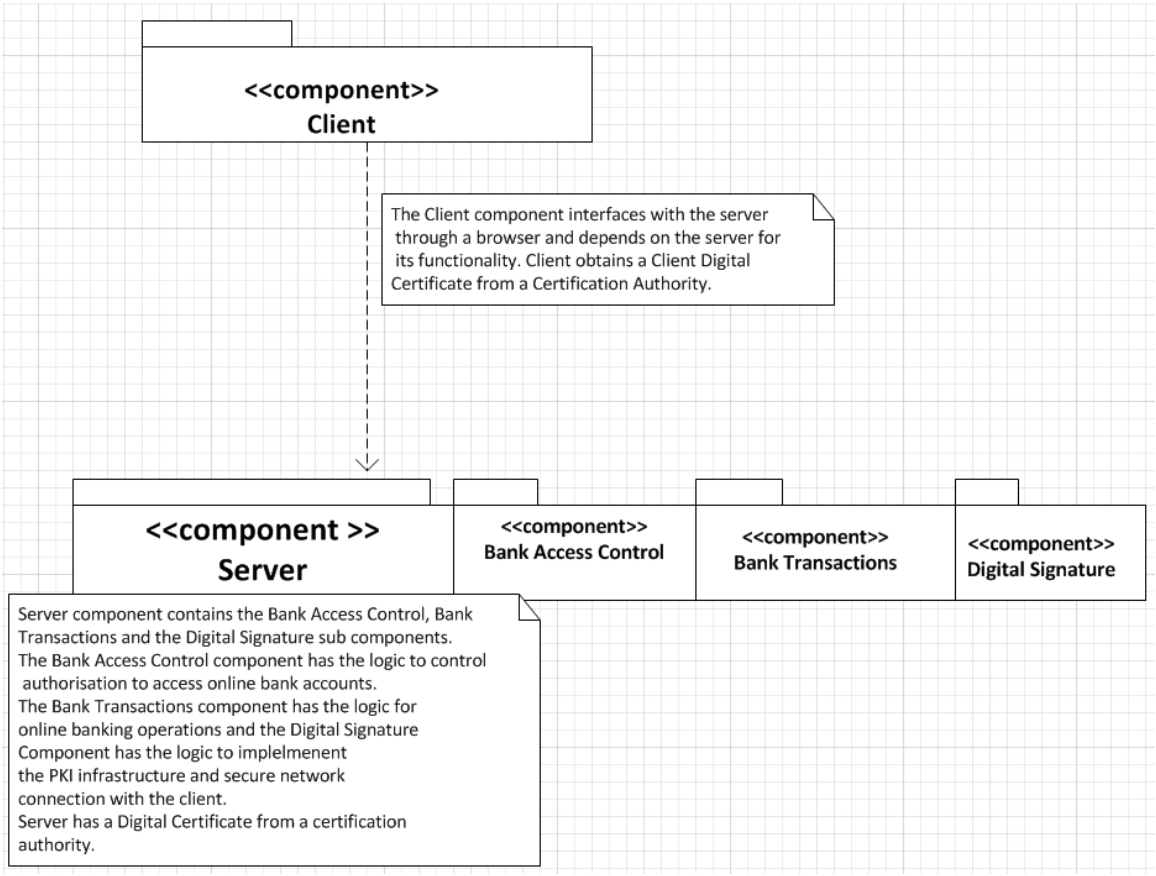


Diagram describing the client server architecture with horizontal decomposition of the server component.

[75] A browser in this case.

4.2 Mutual Authentication Mechanism

The choice for authentication for this project is the PKI with mutual authentication that includes the use of fingerprints.

The tables below describe the main features of the chosen PKI Infrastructure and fingerprint authentication respectively .A diagram also illustrates the main features of the PKI and fingerprint authentication architecture for this project.

Server Certificate	Client Certificate	Public Keys	Private Keys	Data Encryption algorithms	Protocol	Contents of transferred data
Root X.509 Certificates installed in IIS[76] which is stored in a certificate store root.	X.509 certificates from xenos[77] certificate provider.	Derived from the X.509 certificates in both server and clients respectively and are bound to specified certificates .	Created from the respective X.509 certificates and stored in the key containers of the server and client respectively.	- **AES** is used for encryption of the keys to be exchanged. - SHA1 for encryption of data to be transmitted.	-For data transportation Https (http over SSL). - For data encryption, Secure Socket Layer (SSL)	Keys, profile and fingerprint data during account creation, fingerprint data for authorization and digital signing of banking transactions.

Table describes the main features of the chosen PKI Infrastructure.

[76] Internet Information Service on Windows Server 2008

[77] Can be obtained from http://www.xenossoftware.com/freetools/certificategenerator/

Storage of Fingerprint data on client	Fingerprint data on server	Authentication mechanism
Stored on a PC peripheral device and encrypted with SHA1 encryption algorithm	-Transported in hashed format through https and is part of client digital signature. - Not actual fingerprint data, but a representation of it.	-Third party human sensitive fingerprint scanning device. -Matching hash fingerprint data on client and server.

Table describes the main features for fingerprint authentication respectively.

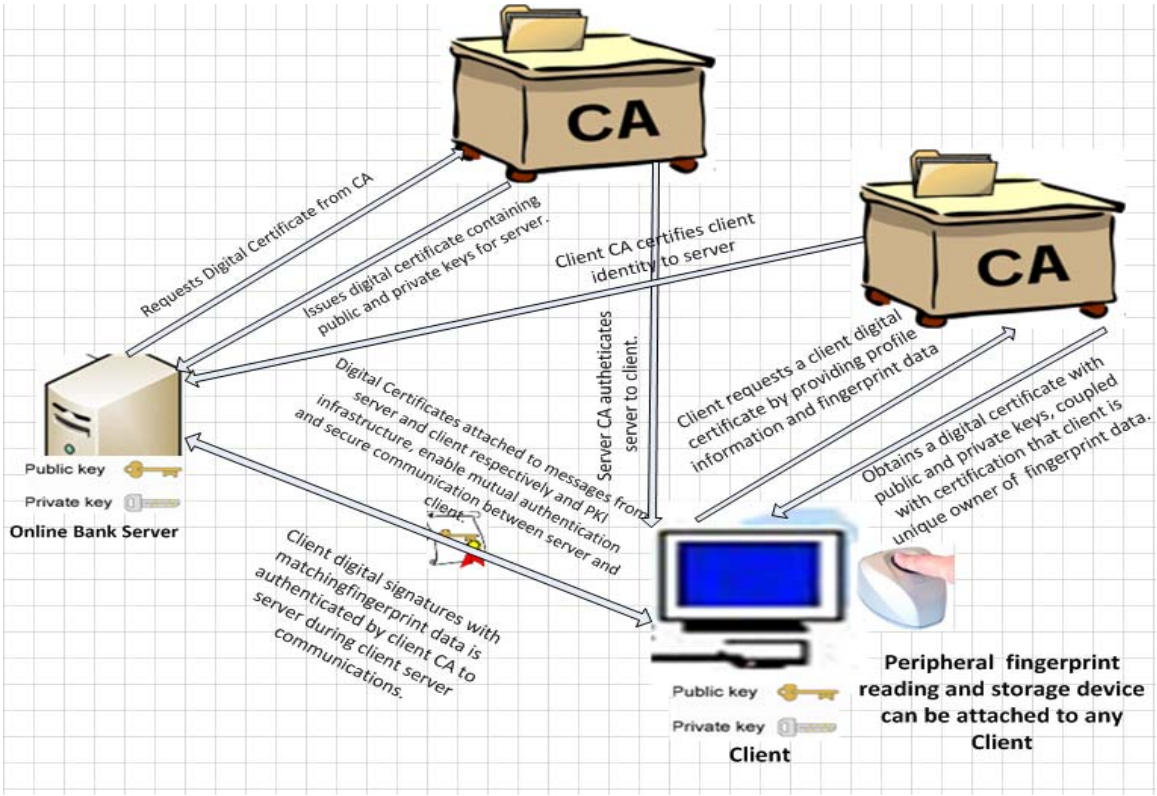


Diagram illustrating the main features of the mutual PKI that uses client fingerprint data for authentication.

The architecture for this mutual authentication mechanism is such that the client and the server both obtain digital certificates from certification authorities and use their respective private and public keys for communication following the PKI.

Authentication of the server to the client is through digital certificates as usual and communication between them uses TSL protocol.

The innovation in this design is that a new regime for client certification is being proposed in which the client in addition to the usual information provided for digital certificates, has to provide fingerprint information that will be included in the client certificate. Enrollment and reading of fingerprints should be through a peripheral computer device. Following this innovative design, the client authentication to the server through a digital certificate, should amongst other things certify that the fingerprint data in the certificate belongs to the client and the client only.

The client uses a peripheral device to store the digital certificate in order to ensure portability. The digital certificate on the client also has to be stored encrypted in order to ensure security.

Fingerprint authentication to the server should be through the use of a peripheral fingerprint reading device to also ensure portability.

Transmission of fingerprint data to the server should always be encrypted and this is achieved through the use of the HTTPS protocol obtained through the use of TSL.

Authentication of fingerprint from client to server is through unique matching data on server that can be achieved through the use of strong hash algorithms.

Privacy concerns of users will be ensured by avoiding the storage of actual fingerprint data on the server. Rather fingerprint data should be stored in an encrypted format on the server.

4.3 Access Control Mechanism

The access control mechanism for this application shall be based on the ASP.NET Provider Model available from ASP.NET 2.0 and above.

This ASP.NET Provider Model was designed with the following goals in mind[78]:

- To make ASP.NET state storage both flexible and extensible
- To insulate application-level code and code in the ASP.NET run-time from the physical storage media where state is stored, and to isolate the changes required to use alternative media types to a single well-defined layer with minimal surface area
- To make writing custom providers as simple as possible by providing a robust and well-documented set of base classes from which developers can derive

[78] Source: http://msdn.microsoft.com/en-us/library/aa478948.aspx

provider classes of their own.The following table documents the features and services that are provider-based and the default providers that service them[79]:

Feature or Service	Default Provider
Membership	System.Web.Security.SqlMembershipProvider
Role management	System.Web.Security.SqlRoleProvider
Site map	System.Web.XmlSiteMapProvider
Profile	System.Web.Profile.SqlProfileProvider
Session state	System.Web.SessionState.InProcSessionStateStore
Web events	N/A (see below)
Web Parts personalization	System.Web.UI.WebControls.WebParts.SqlPersonalizationProvider
Protected configuration	N/A (see below)

SQL providers such as *SqlMembershipProvider* and *SqlProfileProvider* use Microsoft SQL Server or SQL Server Express as their data source. *InProcSessionStateStore* stores session state in memory, while *XmlSiteMapProvider* uses XML files as its data source. N/A (Not Applicable) designates features and services for which providers must be explicitly identified[80].

This application shall make use of the Membership Provider, the Role Provider and the Profile Provider.[81]

[79] *Ibidem*
[80] *Ibidem*
[81] For detailed explanations on the Provider Model visit: http://msdn.microsoft.com/en-us/library/aa478948.aspx

4.4 Database Entities and Classes

The implementation for this application shall be to a large extent based on Microsoft's ADO.NET[82].

ADO.NET provides consistent access to data sources such as SQL Server and XML, and to data sources exposed through OLE DB and ODBC. Data-sharing consumer applications can use ADO.NET to connect to these data sources and retrieve, handle, and update the data that they contain. [83]

ADO.NET separates data access from data manipulation into discrete components that can be used separately or in tandem. ADO.NET includes .NET Framework data providers for connecting to a database, executing commands, and retrieving results. Those results are either processed directly, placed in an ADO.NET DataSet object in order to be exposed to the user in an ad hoc manner, combined with data from multiple sources, or passed between tiers. The DataSet object can also be used independently of a .NET Framework data provider to manage data local to the application or sourced from XML.[84]

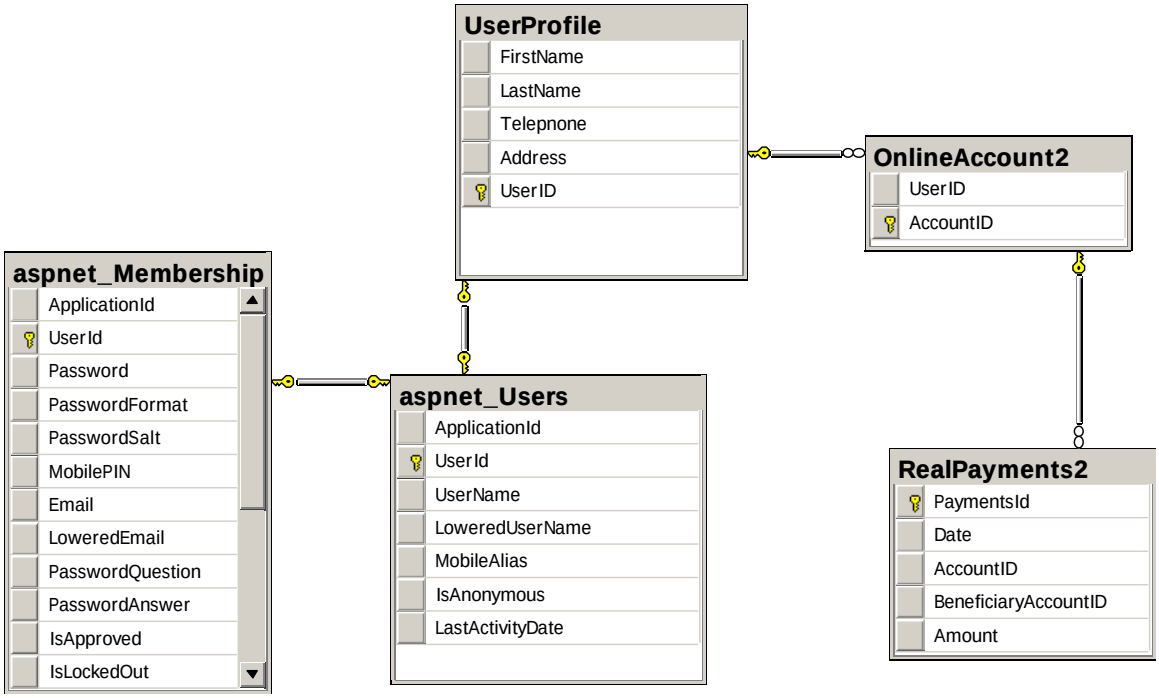
Further extensibility of the application in this project can be achieved through Object Relational Mapping using Microsoft's Language Integrated Query (LINQ) in which the database entities can be mapped into classes, thereby giving the developer more flexibility.[85]

With the use of ADO.NET for this project, the design of the database entities is very important. The bulk of the business logic will be through direct access and manipulation of the data using ADO.NET.

The design here will be that of the database, with the assumption that the database can easily be mapped into classes if need be. The diagram below describes the Database Diagram for this project.[86]

---

[82] ADO stand for ActiveX Data Objects
[83] Source: http://msdn.microsoft.com/en-us/library/h43ks021.aspx
[84] *Ibidem*
[85] However this functionality will not be implemented for this version of the application
[86] Only two entities from the Provider Model have been included in this diagram. This is because they hold information about the UserID and Login credentials.



The database diagram for the online banking application

This diagram has the following relations:

- aspnet_Memebership,
- asp_netUsers,
- UserProfile
- OnlineAccount2 and
- RealPayments2

The aspnet_Memebership and asp_netUsers are relations from the Provider Model, while UserProfile, OnlineAccount2 and RealPayments2 are additional relations created for this application.

4.4.1 aspnet_Memebership relation

The aspnet_Memebership holds amongst other things the login credentials for each user and is associated to the asp_netUsers relation through the UserID foreign key. The Password row for this relation according to this design will hold the hash value of each user's fingerprint data. This same value is what is used as fingerprint data for the client digital certificate.

4.4.2 asp_netUsers relation

The asp_netUsers relation contains each user's name which is unique and this username is what should be displayed on the client digital certificate, together with the hash value of the Password in the aspnet_Memebership relation. This way only the owner of a given fingerprint data can be authenticated through the digital certificate, as the one who digitally signed a transaction for an account belonging to him.

4.4.3 userProfile relation

The UserProfile relation contains rows with personal data for each user. Each user profile can be uniquely linked to the aspnet_Memebership and asp_netUsers relations as well as the rest of the Provider Model Relations, through the UserID foreign key. This way, the User Profile and other relevant data related to the user can be easily retrieved from the Provider Model relations.

4.4.4 OnlineAccount2 relation

This relation contains the information for each users account and is linked to the UserProfile relation, through the UserID foreign key.

4.4.5 Realpayments2

This relation contains information that is used by each user to make online banking payments.

4.5 Digital Signature Mechanism

The digital signature mechanism for this application will make use of third party API software that gives the possibility to create digital signatures from X.509 digital certificates.

The capabilities of this API will be used to create digital signatures that include fingerprint data of the signatory.

This API is Bonnie.NET - Cryptographic API for .NET Framework.

Version 4.0.3.1. Bonnie.NET is a Microsoft® .NET framework API that implements all the instruments needed when developing .NET code with enhanced data protection based on cryptography.[87]

Bonnie.NET includes a complete set of instruments for symmetric and asymmetric encryption, X.509 certificates management, Hash and Keyed Hash (HMAC) generation, hardware based cryptography.[88]

This API provides the following capabilities[89]:

- Digital signature of texts, forms data and files from ASP.NET web pages.
- PKCS#7 signed-data message generation from ASP.NET web pages.
- Multiple digital signature generation by different users.
- PKCS#7 signed-data message generation from desktop applications.
- Browsing of PKCS#7 signed-data message from all sources.
- Full customization of SignerButtons web controls.
- Ajax and SEO enabled SignerButtons web controls.

4.5 Risk Mitigation Design Strategy

The mitigation design strategies for the risks identified in section 3.2.2.2 are presented in the table below.

[87] Source: http://www.we-coffee.com/bonnie.aspx
[88] *Ibidem*
[89] *Ibidem*

Business Risk	Technical risk	Mitigation Design Strategy
I) Difficulties to Sign up or Login	**-rid1:** problems with fingerprint data registration and storage. **-rid2:** problems to use	**For rid1:** - Fingerprint data is securely transmitted using TSL and stored encrypted on the server. - Fingerprint data of a distinct user and this data is used for authentication of the user as well as for user's digital signature. **For rid2:** -Ensure fingerprint reading software and hardware function properly.
II) Invasion of users' privacy and sensitive data.	**-rid3:** user login credentials can be captured and replayed. **-rid4:** Users' worried about abuse of their fingerprint data as well as the possibility of false acceptance[90].	**For rid3:** **F**ingerprint data is encrypted during transmission and storage. **For rid4:** - Actual fingerprints are not stored on the servers rather each fingerprint used has a one to one mapping with a related data stored on the server. System ensures usage of live fingerprints by using appropriate third party hardware.
III) Errors during usage	**-rid5:** Stack traces printed for exceptions may be used to hack system. **rid6:** Cascading failures	**For rid5:** - Displays error pages and log errors to secure pages on server side. **For rid 6:** - Compartmentalize development approach.
IV) Release delays	**-rid7:** Non compatible developed modules. **-rid8:** Several implementation bugs. **-rid9:** Some modules not ready.	**For rid7:** - Regular integration tests between modules. **For rid8:** - Avoidance of complex code[91]. Simple design approach. **For rid9**: - Compartmentalize development.
V) Brief system interruptions	**-rid10:** System temporarily shut down during updates.	**For rid10:** - Concurrent updates without system interruptions or inform users about brief periods of system interruptions during maintenance.
VI) Difficult to modify system.	**-rid14:** Update difficulties	**For rid14:** - Make system extensible.

Business Risk	Technical Risk	Mitigation Proposal

Table describing the risk mitigation strategy

[90] False acceptance is the instance of a security system incorrectly verifying or identifying an unauthorized person.
[91] Use of Microsoft ADO.NET and the Provider Model

VII) Portability of system	**-rid15:** application inaccessible from any computer connected to the internet from anywhere.	**For rid15:** - USB fingerprint reading device for reading fingerprints for authorization and client side digital signatures.
VIII) Complex system for users.	**-rid16:** Bad user interface implementation and complex usage procedures.	**For rid16:** Simple user interface with help information, and use of tooltips as well as video demos.
IX) Failed Acceptance	**- rid:17:** Failed unit, component, integration and user tests.	**For rid17:** continues testing at all levels of development.
X) Users feel insecure.	**-rid18:** User coerced to log into account	**For rid18:** An alert procedure and a mock transaction, in the event of coercion is triggered, when another client digital certificate created for this purpose is used.
XI) System not cost effective	**-rid19:** development abandoned due to high costs.	**For rid19:** Use of existing USB fingerprint reading device can be cost effective.[92]

Table describing the risk mitigation strategy

[92] See for example transcend jetflash 220 at:
http://www.transcendusa.com/Products/ModDetail.asp?LangNo=0&ModNo=169

5 IMPLEMENTATION

The implementation process for this application shall follow the modular development approach, in which the application will be developed in components before eventual integration of the different components.

This implementation, will present what has actually been implemented and will present reasons for deviation from the actual design, where appropriate.

The components identified for this project include the Access Control Component, the Digital Signature Component and the Bank Transactions Component.

5.1 Access Control Implementation.

According to the design, the user is supposed to use a fingerprint reader to provide a fingerprint tip that should be used for login into his account.

Due to the fact that it has not been possible to find a fingerprint reading device that provides raw fingerprint data that could be manipulated upon, in the implementation, GUIDs[93] have been used to represent each unique fingerprint data.

A GUID is a 128-bit integer number used to identify resources. 128-bits is big enough and the generation algorithm is unique enough that if 1,000,000,000 GUIDs per second were generated for 1 year the probability of a duplicate would be only 50%. Or if every human on Earth generated 600,000,000 GUIDs there would only be a 50% probability of a duplicate. GUIDs are used in software development as database keys, component identifiers, or just about anywhere else a truly unique identifier is required. GUIDs are also used to identify all interfaces and objects in COM programming.[94]

The access control mechanism has been implemented by making use of the ASP.NET Provider Model described in the chapter for design.

The Provider Model used created the ASPNETDB.MDF data store based on the SQL Server Express 2008 and is located in the App_Data folder of the application. This data store contains all the relations created by default when the provider model is used. This data store also contains the other relations created for this application.

[93] Globally Unique Identifiers

[94] Source for paragraph: http://www.guidgenerator.com/online-guid-generator.aspx

However in order to have some elements of fingerprint enrolment and subsequent reading for authentication and authorization, this implementation makes use of a fingerprint reading USB device that enrolls fingerprints and can be used to encrypt data on storage devices. This device called JetFlash 220 has the following functionalities:[95]

- Fully compatible with Hi-Speed USB 2.0
- Advanced Fingerprint Recognition Technology
- Easy Plug and Play installation
- AES256 encryption technology
- Protect Files: Protect files stored on your computer using fingerprint security
- Mobile Favorites: Access your Internet Explorer Favorites on another computer
- Website Auto-Login: Automatically login to websites where you are a registered user (Microsoft Internet Explorer only)
- Repartition Tool: Adjust the size of the Private and Public partition areas for storing data
- USB powered. No external power or battery needed
- LED indicates data transfer activity
- Driverless (Windows 2000/XP/Vista/7).

In this application, this fingerprint reading device is used to encrypt[96] sensitive data like the Login GUID and the Digital Certificate, stored on a portable storage device like a USB flash memory stick. Decryption of the encrypted data is through the reading of the enrolled fingerprint by the reader.

5.1.1 User Account Creation

The creation of an account starts with the use of the CreateUserWizard control from the Visual 2010 toolbox in the design palette. For this application, this create user Wizard has been implemented in the *Register.aspx* web page located in the *Account* folder of the application. This CreateUserWizard has been configured by this developer in the *Membership* section of the web.config xml file of the application such that a password with at least thirty six characters is required and this password

[95] Source: http://www.transcendusa.com/Products/ModDetail.asp?LangNo=0&ModNo=169
[96] Encryption uses enrolled fingerprints.

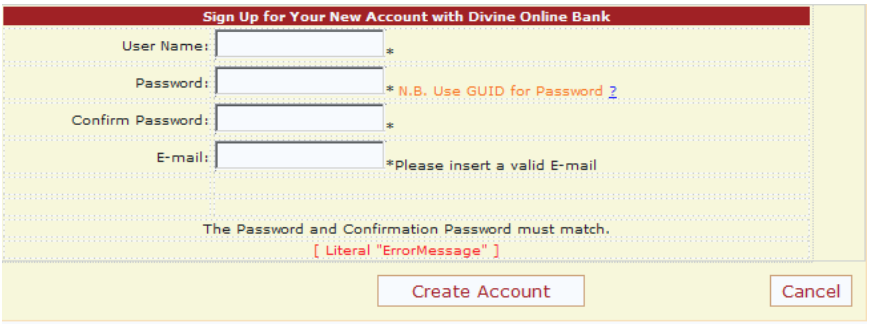
should have at least one non alphanumeric character[97]. This wizard has also been configured to enable password reset[98], a maximum of three password attempts[99] and a maximum of 10 minutes interval between login attempts for a login attempt to be considered as the first login attempt[100]. The password retrieval, the require question and answer and the require unique e-mail features, have been disabled.

The code snippet of this configuration is presented below.

```
membership>
      <providers>
        <clear/>
        <add name="AspNetSqlMembershipProvider"
type="System.Web.Security.SqlMembershipProvider"
connectionStringName="ApplicationServices"
             enablePasswordRetrieval="false" enablePasswordReset="true"
requiresQuestionAndAnswer="false" requiresUniqueEmail="false"
             maxInvalidPasswordAttempts="3" minRequiredPasswordLength="36"
minRequiredNonalphanumericCharacters="1" passwordAttemptWindow="10"
             applicationName="/" />
      </providers>
    </membership>
```

Code snippet of configuration file for the CreateUserWizard

The design of the user interface of the Register.aspx web page is presented below.



User interface design of the Register.aspx web page

The link after the password field with the question sign, takes the user to a new window where he can generate a GUID to use as password. Error handling in this Registration form is ensured by red asterisks in case a field is left empty before

[97] This setting is appropriate for GUIDs because they have at least 36 characters and contain non alpha numeric characters.
[98] This feature can be useful if user intends to change GUID representing fingerprint data.
[99] Good feature to prevent brute force attacks
[100] Also a good feature to prevent brute force attacks

submission. An Error message that makes use of the compare field validator control in the Visual Studio will indicate that Password and Confirm Password must match in case the respective inserted values do not match. Another error message will indicate that Password length must be at least 36 characters and must contain at least one non-alphanumeric character in case values not complying with these requirements are inserted. A required field validator control from Visual Studio toolbox ensures that a valid email is inserted.

When valid account data is submitted part of the data is stored in the aspnet_Membership relation and a UserID primary key is automatically created for the user. The actual GUID used as password is not stored in the database but rather a SHA1 hash algorithm is used to create a keyed-hash of the password that is stored in the database[101]. [102].

After submission, the Username is stored in the aspnet_User relation together with the UserID in the aspnet_Membership relation, as a foreign key.

5.1.2 Login Implementation

According to the design, the user is supposed to log into his account using his fingerprint but because of the difficulty to have actual fingerprint data as explained above, the GUID used for data creation that represents the user fingerprint data, is used to log into the account, together with the user's UserName.

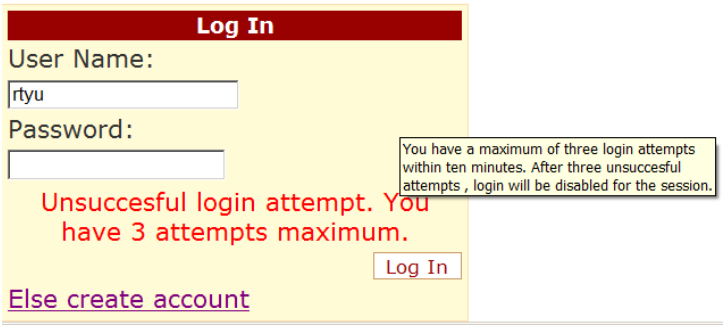
The Login page is located in the Login.aspx web page in the *Account* folder of the application.

The Login is implemented using the Login control in the Visual Studio toolbox. This Login page has been implemented to provide fields for Username and Password, a Login button and redirect button to create new account. Error handling is also ensured through required field validators in case a field is left empty before the login button is pressed. An error message will also be displayed in case wrong login credentials are submitted. Tooltip is also used to inform the user of the limit to the number of Login attempts.

The diagram below presents the login page when viewed in internet explorer after wrong login credentials have been inserted.

[101] See http://msdn.microsoft.com/en-us/library/aa479048.aspx#bucupro_topic11

[102] This feature ensures privacy of fingerprint data as well as security for the same data.



Screen dump shows login page when viewed in internet explorer, after wrong login credentials have been inserted.

The login is achieved through a comparison between the Username and Password inserted with the same fields in the database[103].

5.1.3 User Profile

Once a valid account is created, the user is redirected to a page to insert personal information including first name, last name, telephone number, address and the UserID acquired when account was created[104].

The UserID field serves as a link between the User Profile and the Provider Model relations.

The page to insert the User profile is implemented in the AccountProfile.aspx page located in the *Account* folder of the application.

The implementation is made possible through the use of a *Formview* control from the VS[105] toolbox which is data bound to an *sqldatasource* which contains the fields in the UserProfile relation. The *Formview* control is configured to use the insert item template and set to the insert default mode, in the properties window of VS.

Required field validators have been used to indicate to users that obligatory fields have to be completed.

The SQL insert query used for the sqldatasource in this case is :

INSERT INTO UserProfile(FirstName, LastName, Telepnone, Address, UserID) VALUES (@FirstName, @LastName, @Telepnone, @Address, @UserID)

[103] In the case of the password, this comparison is done between the actual password and its hash value in the database.
[104] In order for the user to be able to complete the profile information this UserID has to be sent to the User through another mechanism not implemented in this version of the project.
[105] Visual Studio 2010

5.1.4 Roles management

Roles are managed by the nature of the authorization given to different users to access specific folders.

The pages in the root folder which is the *DivineOnlineBank* folder is accessible to anonymous users *i.e.* non logged in users as well as logged in users.

The *Account subfolder* has pages for registering new members and for login. This pages are always open to any user but the *AccountProfile.aspx* page in this folder, has been configured to allow for access only to registered users. Within this page registered users can insert their profile information after completing the registration process.

The *CustomerPages* subfolder has been configured to grant access only to registered users. Each subfolder within this subfolder is configured to give access to one registered user only. This user subfolders have pages to make payments, view transactions for a particular account and to update the logged in user's profile.

The *ProtectedPages* subfolder has also been configured to grant access only to logged in users.

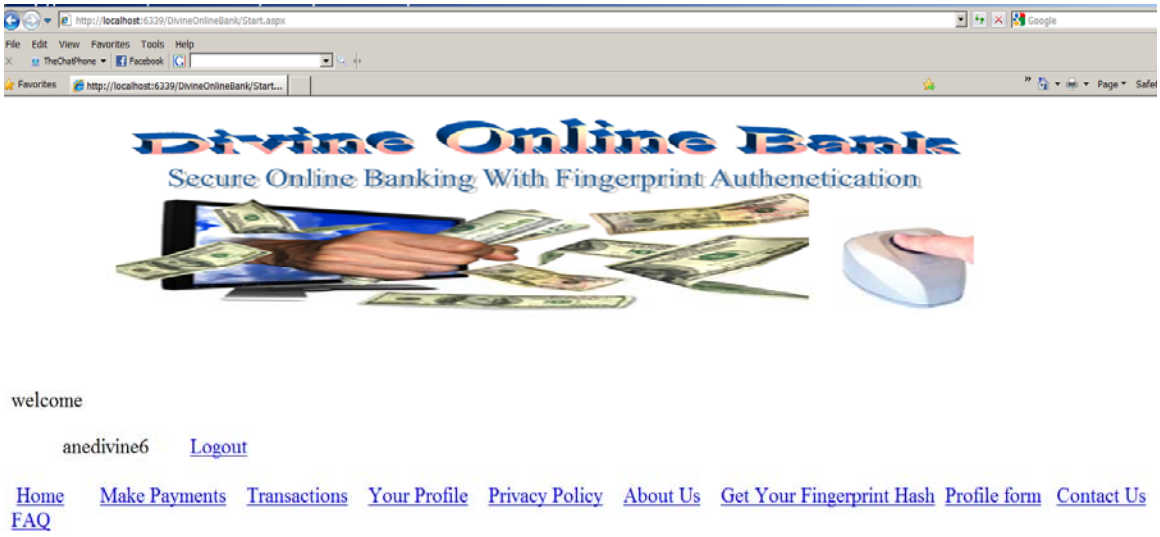
The configuration of the roles is done in the ASP.NET Configuration Manager tool in VS and the roles data is stored in the aspnet_Roles and aspnet_UsersInRole tables created by the Provider Model.

5.1.5 LoginView, LoginStatus and LoginName

The *LoginView* control in VS toolbox is used to create short customized messages for logged in and for anonymous users respectively.

The *LoginStatus* displays information depending on whether the user is logged in. If the user is logged in, he can use the link to logout, or if he is logged out, he can use the same link to login. The *LoginStatus* Control in VS toolbox has been used to implement this feature.

The *LoginName* Control is used to implement a feature, which displays the name of the logged in user.



Screen dump of a window of the application indicating the LoginView, LoginName and LoginStatus of a logged in user

5.1.6 Change Password

The *ChangePassWord* Control in VS toolbox, is used to implement a feature where the user can change his password[106]. This has been implemented in the *UpdateLoginInfo.aspx* page. This requires the user to first insert the old password, before inserting the new password twice.

5.2 Digital Signature implementation

The creation of the digital signature, starts in the *CreateDigitalSignature.aspx* page located in the Protected Pages folder and contains the following information for the user:

> *"In order to create a digital signature, you must apply for a digital certificate. For this application you will use a self signed digital certificate. Download certificate creation software from http://www.xenossoftware.com/freetools/certificategenerator/ .*
>
> *You will need two certificates. One of them is used to digitally sign genuine transactions, and the other is used to sign mock transactions in the event of coercion.*
>
> *Transactions digitally signed as done under coercion will appear real on the client but will not be executed by the server. Rather an alert will be sent to competent authorities and the account will be blocked temporarily.*
>
> *Make sure you store your certificates in another USB flash drive encrypted using the transcend jetflash 220 USB fingerprint reading flash drive sent to you.6*

[106] *I.e.* Change the GUID which in this case will represent the use of another fingerprint

In order to be able to sign your transactions from any computer decrypt and use the certificate on your flash and make sure you encrypt it again after usage. You also need your fingerprint hash[107] for your certificates as well as for the digital signing of your online banking transactions."

The next step of the signature creation process is to get the fingerprint hash that is used for the digital certificate applications as well as for digitally signing transactions. The process to acquire the fingerprint hash is implemented in the *FingerprintHash.aspx* page in the Protected pages subfolder.

If the user has not yet got his fingerprint hash, he has to navigate to the *FPHList.aspx* page in the protected pages folder, which contains a selection of all hashes of the passwords in the database and the respective usernames. The selection is ordered by username and is obtained through a *Formview* Control This control is configured to read only default mode in the properties window and data bound to an *sqldatasource* made up of a selection of the Usernames and password hashes for each user.

The following sql query has been used to create this selection:

SELECT aspnet_Membership.Password, aspnet_Users.UserName FROM aspnet_Membership INNER JOIN aspnet_Users ON aspnet_Membership.UserId = aspnet_Users.UserId ORDER BY aspnet_Users.UserName

When the user gets his fingerprint hash he has to save it so he can get it subsequently when needed.

The mechanism to save the fingerprint hash starts with the creation of a profile for the finger print hash of logged in users in the web.config XML file using the following configuration:

```
<profile>
      <properties>
        <add
          name="FPHash"
          allowAnonymous="false" />
      </properties>

    </profile>
```

The next step is to map this profile to the textbox, save the fingerprint hash and retrieve it later, using the C# code in the *FingerprintHash.aspx.cs* code behind file. The code snippet for this implementation is the following:

```
protected void Page_Load(object sender, EventArgs e)
   {
      // map any text in the textbox to the FPHHash Profile
```

[107] The Fingerprint hash, is the hash value of the GUID password in the database which for this project, represents fingerprint data.

```
    if (!IsPostBack)
    {
        FPtxt.Text = Profile.FPHash;

    }
}
// Save the fingerprint hash in the aspnet_Profile relation and return to start page
protected void FPbtn_Click(object sender, EventArgs e)
{
    Profile.FPHash = FPtxt.Text;
    Response.Redirect("Start.aspx");
}
// Retrieve the fingerprint hash and display it in the label
protected void FPHButton_Click(object sender, EventArgs e)
{
    FPHashLabel.Text = Profile.FPHash;

}
```

The screen dump below shows the above implementation when viewed in a browser

Screen dump of page to get saved Fingerprint hash data when viewed in a browser.

Once the fingerprint hash is obtained, the user has to use the second application unit, in order to digitally sign any transaction.

A second application unit is used for this project because it is not possible to integrate this component and the two other components (Access Control and Bank Transactions), due to the fact that they have not been deployed to a windows server with IIS.

This component is therefore presented separately as a second application unit with the knowledge that both units developed could have can been integrated into a single application, had it been the author, had access to a windows server with IIS 6.0 and above for the deployment of the application.

This second application unit for this project is referred to as *5. Signerbuttons* and is created using the trial version of BONNIE.NET Web Edition API[108] that can be used to sign data from ASP.NET pages.

This API has been used to create a demo application where a fingerprint hash is inserted in a textbox and incorporated into the digital signature[109].

This is unit as implemented, just to illustrate the possibility to use fingerprint data in the digital signature of an ASP.NET web page. This functionality can be extended to directly sign a bank transaction, once the submit button is pressed. However for this implementation the transaction is submitted first and the digital signature is submitted through a second procedure.

For this application to work, the client needs to have a digital certificate installed in his computer. For testing purposes, a self-signed X.509 digital certificate, can be created using the Xenos Certificate Generator.[110]

The Cassandra Framework plugin[111] needs to be installed in the client's computer for this application to work.

Another limitation for this implementation is the fact that this developer has not got access to any code of the implementation of a digital certificate. For this reason, this second application unit has two types of digital certificates implemented.

The first one is the self-signed digital certificate created using the Xenos self-signed digital certificate creator and the second one is an innovative certification regime proposed by this developer, that certifies that the fingerprint hash used to sign a digital signature, actually belongs to a distinct user.

This implementation has been developed with the assumption that further research can be made if a code for implementing digital certificates is available. In such a case this innovative approach can be improved upon to incorporate fingerprint data into mainstream digital certificates and to certify the owner of such data, when he uses the fingerprint, to digitally sign a transaction.

This second application unit has the *default.aspx* page which has information on how to create and sign digital signatures using fingerprint data.

---

[108] This API can be downloaded at http://www.we-coffee.com/bonnie/bonnie-web.aspx

[109] Part of this implementation is an adaptation by this author, of an example provided in the Bonnie.Net API download.

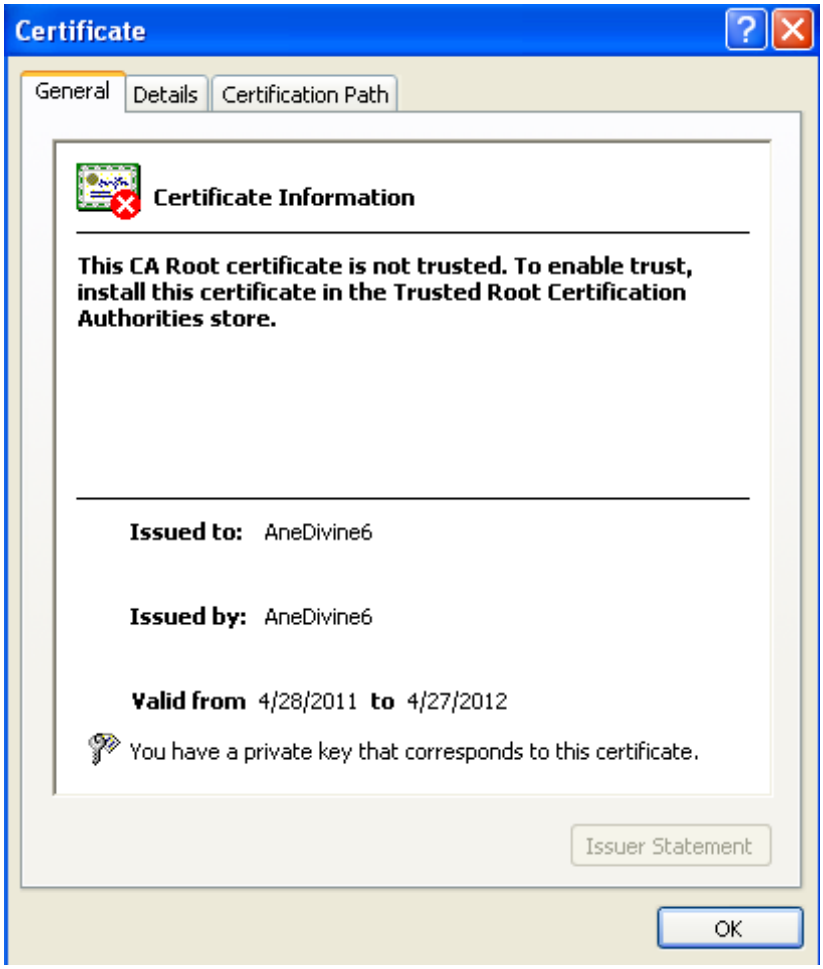
[110] This generatorb is available at : http://www.xenossoftware.com/freetools/certificategenerator/

[111] The plugin can be downloaded at : http://www.we-coffee.com/framework/cassandra-client-framework.aspx

The signature creation page is implemented in the *customSignerButton.aspx* page and its code behind file *customSignerButton.cs*.[112] A third page *FingerPrintCA.aspx* takes the user to the innovative Fingerprint Certification Authority page, proposed by this author.

This Fingerprint Certification is implemented by selecting the user profile information of users including usernames, finger print hashes and UserIDs, from the database implemented with the Provider Model and displaying them sorted in order of names.

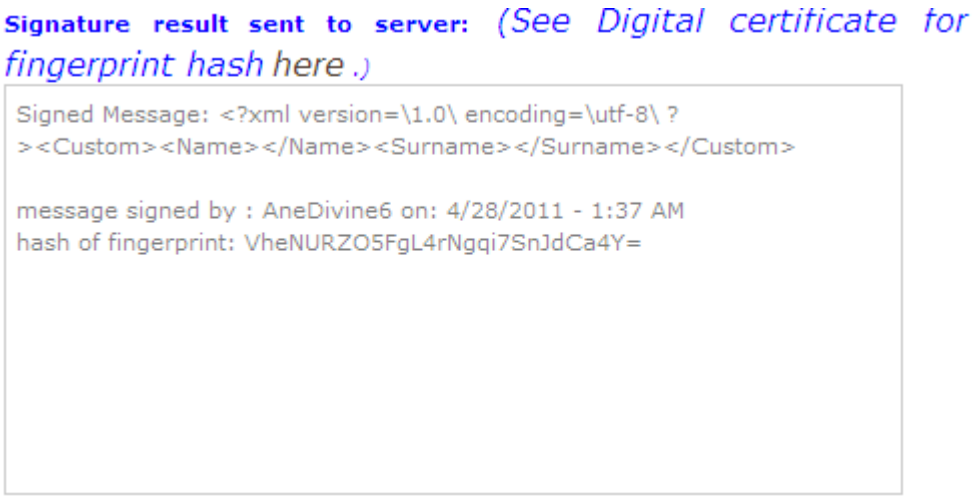
When this application runs in a browser, the user inserts his fingerprint hash in the textbox provided and submits the signature. Digital certificates in the computer account are displayed and the user chooses the appropriate certificate to sign a transaction.



Screen shot of sample self-signed digital certificate as seen in a browser during the digital signature signing process.

[112] This implementation is available in the appendix of this report.

When this is done, a custom JavaScript message containing the Username and the Finger print hash of the signer is displayed on the browser. This message represents the signature that is sent to the server. A screen shot of a sample digital signature as seen on a browser is shown below.



A screen shot of a sample digital signature with finger print data as seen in a browser.

The next step is to navigate to the Fingerprint Certification page that opens in a new window. This page has the user profile information of users and each users profile information, represents their respective fingerprint certificates. Each certificate has a username, finger print hash, UserID and other personal data selected from the database implemented with the Provider Model. This data is displayed sorted in order of names with paging functionalities.

The idea behind this fingerprint certificate is to implement a mechanism in the server which can ensure a match between the finger print hash used by a logged in user to sign a transaction, with the finger print hash and username in the fingerprint digital certificate. However this feature has not been implemented for this version of the application. Rather for illustrative purposes, this comparison can be done visually in order to authenticate the signer of a digital signature using fingerprint data.

The screen shot below shows the fingerprint certificate for the digital signature shown in the previous screen dump, as seen in the browser.

Screen shot of a sample finger print digital certificate as seen in the browser.

5.3 Bank Transactions and User Profile Update

A logged in user, can make payments to other accounts through a page located in a subfolder in the Customer Pages sub folder, to which access is granted solely to him. For example, the AneDivine6 subfolder in the customer pages sub folder can be accessed only by the User with username AneDivine6. This subfolder has the *Anedivine6Account.aspx* page from which this user can make payments to other accounts. Once the payment is made, the user is redirected to the page where he has to digitally sign the transaction.

The *AneDivine6Transactions.aspx* page in the AneDivine6 subfolder has been implemented with a *Formview* control and displays the transactions carried out by this user, starting from the most recent.

This Formview was implemented by data binding it to an SQL data source through the following sql statement :

SELECT PaymentsId, Date, AccountID, BeneficiaryAccountID, Amount FROM RealPayments2 WHERE (AccountID = 102) ORDER BY Date DESC

The screen dump below shows transactions carried out by the user AneDivine6:

Payment ID	Date	Beneficiary's Account ID	Amount Transferred
102	4/14/2011 5:05:00 AM	103	1000.0000
101	4/14/2011 4:36:00 AM	103	0.0000
100	4/14/2011 3:44:00 AM	103	0.0000

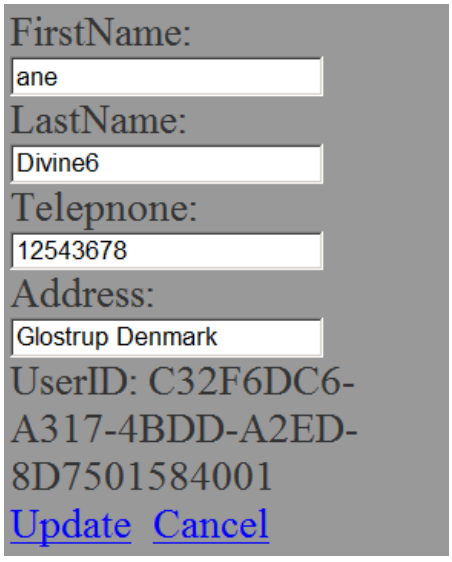
The screen dump from browser, showing transactions carried out by the user AneDivine6

To view or to Update the profile of AneDivine6 a page called *ProfileUpdate.aspx* in the AneDivine6 subfolder has been configured with a *Formview* and *sqldatasource* data bound to each other, to display this users profile and also has the possibility for the profile to be updated.

The SQL data source uses the following sql Update statement:

> ***UPDATE UserProfile SET FirstName = @FirstName, LastName = @LastName, Telepnone = @Telepnone, Address = @Address WHERE (UserID = 'C32F6DC6-A317-4BDD-A2ED-8D7501584001')***

The screen dump below shows this page in a browser.



Screen dump from browser showing the profile of a user, with the possibility to update the profile.

6 Tests and Fulfillment of Requirements

This chapter is based on an evaluation on how the most important requirements in the Thesis Contract as well as the requirements for this project, elaborated in earlier chapters of this report, have been fulfilled.

This chapter shall include the documentation of test results from the execution of the test plan outlined in earlier sections of this report as well as the management of the risks identified in the Requirement Analysis part of this report.

6.1 Tests and Risk Management

The documentation of tests shall be based on component, integration, system and acceptance testing. Unit testing has not been carried out because in keeping with the XP programming methodology of avoiding complex code, the extensive use of C# codes has been limited in this project. Rather in order to strike a balance between simplicity and functionality, Microsoft's ADO.NET has been used extensively, to implement the business logic.

The documentation on risks management shall be based on the risks identified under section 3.2.2.2 of this report.

6.1.1 Tests

6.1.1.1 Component Testing

It focuses on the tests of the functional requirements for each component.

The functionality test of the different components as implemented can be acquired from the explanations on the implementation of these respective components.[113]

6.1.1.2 Integration Testing

It focuses on the integration of the different components.

- The Bank Transaction Component has been integrated with the Access control Component successfully, through the use of the UserID field as a foreign key of the UserProfile relation. This foreign key links the UserProfile, to the rest of the Access Control component relations. The proof of this is the screen shot of a sample finger print digital certificate as seen in the browser under section

[113] Section 5.1 for Access Control, Section 5.2 On digital signatures and section 5.3 on Bank Transactions and User Profile Update.

5.2 above and the screen dump from browser showing the profile of a user, with the possibility to update the profile under section 5.3 above.

- The integration of the Access Control and the Bank Transaction Components with the Digital Signatures Component into one unit has not been possible because of the fact that tests carried out indicated that such integration can only be possible with the deployment of these components as applications on a Windows Server with IIS 6.0 and above[114]. However, the Digital Signatures Component has been implemented as a separate unit. For testing purposes a copy of the same database used in the other components, has been used in the Digital Signature Component implementation, to obtain fingerprint data[115] and the data for the innovative Fingerprint Digital Certificate.

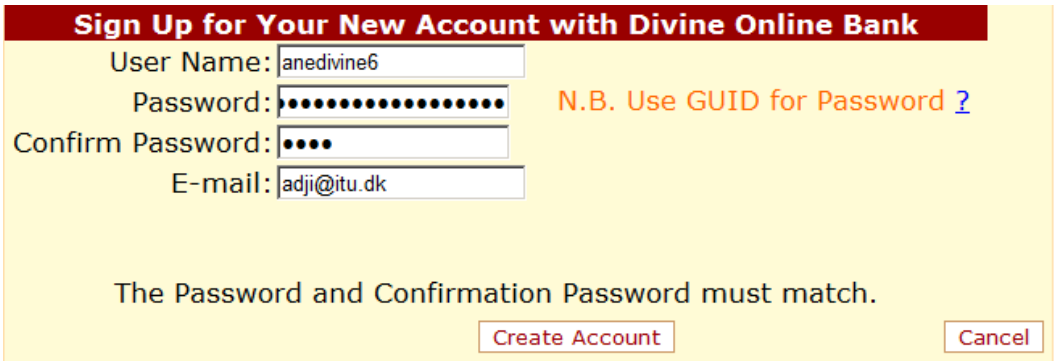
6.1.1.3 System Testing

System Testing focuses on the results of the Test Cases TC1 to TC8 outlined in the System Test Plan in Section3.2.2.3.2 above. Most of these tests have been impliedly documented in the implementation part of this report. Some sample tests results will be presented below.

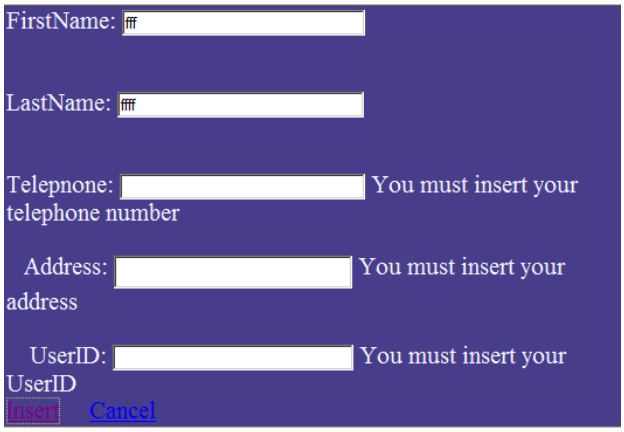
- TC1 and TC2 refer to the validity of profile information entered. The two screen dumps below indicate sample error messages on the browser, during the account creation and the user profile insert processes respectively. In section 5.1.2 a screen dump shows login the page when viewed in internet explorer, after wrong login credentials have been inserted.

  Validated account creation and user profile information, when submitted are persisted in the database.

[114] This developer did not have access to a Window's server so this entire application has been developed and tested using the Localhost provided by VS 2010.
[115] The GUIDs and their hashes.



Screen dump of the account creation process in the browser indicating an error message that password and confirm password fields must match.



Screen dump of the User Profile insertion process in the browser, with an error message indicating fields that must be filled[116].

- TC3 and TC4 and TC5 regarding enrollment and reading of fingerprint in this implementation deals more with the use of GUIDs to represent fingerprint data. In section5.1.1 for example the documentation indicates that account creation process has been configured in such a way that that the password should have at least 36 characters with at least one non alpha numeric character[117]. If a wrong password is inserted, an error message will indicate the requirement indicated in the previous sentence.
- TC6 dealing with test of the use of User coercion digital certificate to sign a transaction has not been automated to trigger an alert and block the account for this version of the application but can only be identified manually.

[116] This message requiring fields to be completed has been implemented in other forms for this application and in some cases indicating stars at the end of fields in the respective forms, that must be filled.
[117] These charateristics are very close to those of a GUID.

- TC7 dealing with test of the User Confirming a Transaction for this implementation, does not simultaneously lead to the digital signing of the transaction using fingerprint data. Rather for this implementation the fingerprint signing is carried out in a second step and the identity of the signer can only be certified using a manual step rather than an automated step. Transaction logs are available after a transaction[118].
- TC8 test plan dealing with the situation after coercion trigger procedure requiring alert and account blockage for 24 hours, has not been implemented.
- The tests on Use Case 0 related to the creation of digital signatures, can be found in section 5.2 describing the implementation of digital signatures.
- The tests on Use Case 1 related to Bank Account Creation, can be found in the documentation of the implementation of the Access Control Component in section 5.1.
- The tests for Use Case 2 and Use Case 3, related to Bank Transactions can be deduced from this report under section 5.3 dealing with the implementation for Bank Transactions.
- The tests for Use Case 4 on coercion trigger as already indicated cannot be carried out because this feature has not been implemented. Rather a manual comparison can be done with the digital signatures used to sign a transaction, in order to identify that coercion had been used for that particular transaction.
- The test on profile update has already been impliedly documented under section 5.3 of this report.

6.1.1.4 Acceptance Testing

Acceptance testing for this application will be documented under the reflections chapter of this report.

6.1.2 Risk Management

The entire risk analysis described in section 3.2.2.2 have been taken into consideration throughout the software development process for this application.

[118] See The screen dump from browser, showing transactions carried out by the user AneDivine6 under section 5.3.

6.2 Fulfillment of Requirements

The requirements for this project include the fulfillment of things that must be carried out as undertaken by the developer in the Thesis Contract, The validation of the Research Question and Sub Questions as well as the validation of Use Cases[119].

6.2.1 Fulfillment of Undertaking

As per the Thesis Contract under section 2.3 of this report, this developer made commitments that had to be met for this project. These commitments have been fulfilled as follows:

- The entire problem area, problem formulation, analysis and design processes have been carried out.
- The implementation has been carried out to allow for secure authorization to a user's bank account, using GUIDs that represent user fingerprints. Authentication of users has been made possible through the implementation of digital signatures incorporating fingerprint hash values which can be certified by a customary self-signed digital certificates, as well as an innovative form of digital certificate proposed by this developer, which authenticates the identity of the owner of fingerprint data, used to sign digital signatures for online transactions. The implementation has created the possibility for the user to make some kind of online transfer of money from his account, to another account.
- In keeping with the XP development methodology that proposes the avoidance of complex code, this developer limited the use of lots of code in the implementation and made extensive use of Microsoft's ADO.NET. For this reason unit testing of code has not been carried out. However, component, integration, system and acceptance tests have been carried out.
- The DLL for the implementation is available for delivery in a compressed file format.
- With regards to the research element of the project, the system design has created a design to incorporate finger prints into the client side digital signatures and in addition has proposed a new regime for digital certificates which indicate the identity of the owner of digital fingerprints used to sign online transactions. This design also proposes a system in which a single

[119] The validation of the use cases has been a part of the system tests described above.

fingerprint login procedure is sufficient to allow for authorization to an online bank account as well as the use of the same fingerprints to digitally sign online transactions.

The implementation fulfilled the design to the extent that GUIDs representing user fingerprints were used to control access to bank accounts in one application unit, and in a second application unit, uses the hash values of the GUID to digitally sign an online transaction. Certification of the identity of the owner of fingerprint data used for signatures is done manually for this implementation. Further research in this area is needed to incorporate the fingerprints directly into mainstream digital certificates and to automatically certify the identity of the person who has the fingerprint used for a digital signature. Further research is also needed to create a onetime fingerprint reading process that controls access to a bank account and at the same time allows for the digital signatures using the same fingerprints.

- The latest Microsoft Technologies in the .NET 4.0 Framework and Visual Studio 2010 as the IDE have been used for this project.
- In addition to obligatory undertakings in the Thesis Contract, this developer optionally implemented a procedure for the application, whereby a USB fingerprint reader is used to store sensitive data like digital certificates and login credentials, which can be encrypted and decrypted using enrolled fingerprints.

6.2.2 Validation of the Research Question

The extent of the validation of the main Problem Formulation has been carried out in the form of the tests and tests results presented above. The validation of the Research Sub Questions also forms part of the validation of the Research Question requirements.

With respect to the Research Sub Questions in section 1.2.1, the following validation procedures have been carried out.

- The design proposes a cost effective USB fingerprint reading device for the reading of fingerprint data and encrypted storage of this data on the client as well as the server respectively. In the implementation, GUIDs are used to represent fingerprint data and are stored encrypted in a portable device

attached to the client and stored as a SHA1 hash values of the GUID on the server.

- An affordable USB fingerprint reading device and its software have been chosen for this project.
- Fingerprint data or the GUID that represents it, is stored on the client using an AES 256 encryption standard, through the use of fingerprint enrolled and read using a third party USB fingerprint reading device .They are stored as a SHA1 hash values of the GUID on the server. The design also proposes the transmission of the fingerprint data over the network in the form of encrypted data through https protocol.
- The fulfillment of use of the fingerprint for authentication has been covered in bullet 11 of section 6.2.1.
- Incorporation of user fingerprints in client side digital signatures has been covered in section bullet 11 of section 6.2.1.
- The use of portable USB fingerprint reading and storage [120]devices will offer more convenience and portability of the application for users.
- The use of the USB fingerprint reader, offers a cost effective solution for users.
- In the event of coercion, the design proposes the use of a second digital certificate to be used for digitally signing transactions in such circumstances. According to the design the use of this certificate will allow for mock online transactions while triggering an alert to competent authorities from the server and a blockage of the account.
- This implementation does not propose a solution to ensure that fingerprints come from a living human being. This kind feature may be available from third party solutions.
- User privacy concerns have been ensured by avoiding the storage of actual fingerprint data on the server, and through an express a commitment to users on the web siite, that fingerprint reading services will not be used for any criminal investigation or other purposes for which it was not intended.

[120] Portable storage device to store digital certificates and fingerprint data in encrypted form.

7 User Guide

As already indicated, this implementation consists of two units, one Called *AneDivine Online* Bank dealing with access control and bank transactions and a second one dealing with digital signatures.

The first unit has instructions on how to use this application in the default start page located in *start.aspx* page.

This unit when run can be tested using the username*:* ***anedivine6*** and the *password*: ***121790a6-db7c-4116-9682-5dc6a758aba4*** to log into a sample account and test the functionalities.

To run both units from Visual 2010, unzip the folder delivered in order to have two separate folders containing the sub folders of both units. In the file menu of Visual Studio, go to *open* then *website* and then select the folder of the unit you want to test. Any of the applications when opened in Visual Studio, can be viewed in a browser by simultaneously pressing Ctrl and F5.

However in other to test the digital signature unit in addition to the procedure above, you must have installed at least a self-signed .X509 digital certificate on the testing computer[121]. You must also download and install: a trial version of the Bonnie.NET - Cryptographic API for .NET Framework[122] as well as the Cassandra Framework[123]. This certificate must be selected to sign transactions. A fingerprint hash for each user is also available to be used for signing the fingerprint version of the digital signature.

A video demo of this application will subsequently be made available at: http://cameroonoboso.com/camerooniansindk/DKCamNews.aspx

[121] You can create one using Xenos Certificate generator available at : http://www.xenossoftware.com/freetools/certificategenerator/

[122] Available at: http://www.we-coffee.com/bonnie.aspx

[123] Available at : http://www.we-coffee.com/framework/cassandra-client-framework.aspx

8 Reflections

This project was quite a learning process for this author. The iterative development strategy brought about many new ideas that were not foreseen in the beginning of the project. Several new tools, technologies and implementation strategies were learned by this author during the process.

This project has proposed designed and implemented features that propose a new approach to digital certificates that should include user fingerprint data, and an innovative certification regime to attest to the identity of the person who has the fingerprints used for a digital signature.

This project also proposes that a single fingerprint reading procedure should allow for login and digital signing of transactions by the logged in user.

All these features have been geared towards implementing a more secure online banking system that uses fingerprint data for access control and for digitally signing banking transactions.

This project has proposed elaborate requirements analysis and design to achieve the goals for the project.

The implementation has also been quite extensive and illustrates most of the features of the proposed secure system. However due to resource constraints some of the features have been implemented as prototypes, to describe the functionality of the proposed application.

The implementation in this project has been carried out in two units: the first unit deals with access control and the second with digital signatures, all of which use user fingerprints. The reason for the separate implementations was a limitation in the process due to the fact that a Windows Server with IIS 6.0, needed for deployment and integration of this units was not available to this developer. The fact that deployment to a server was not carried out, server digital certificates have not been implemented as well as the mechanism for network encryption enabled through the HTTPS protocol. However no innovation is involved in this process so it has been sufficient to describe it in the design.

Also due to the fact that this author did not have access to any API, or code on the implementation of digital certificates, two separate digital certificates have been implemented. The first one is a customary digital certificate created from existing self-signed certificate generator software and a second one proposed by this author

certifies fingerprint data used to digitally sign transactions. Access to code on digital certificate, could have led to the integration of the old and the proposed certification regimes. However this part of the project is open to further research to integrate both certificates.

Another limitation faced in this project, is the fact that the author could not have access to a fingerprint enrolment and reading device which gave raw fingerprint data to be manipulated upon. For this reason, this author chose to represent fingerprint data with Globally Unique Identifiers (GUIDs).

Despite this limitations which can be watered down by availability of more resources and further research, this author is sure that an acceptable prototype application with lots of functionalities meeting the requirements for the system have been implemented and can be tested in a browser.

Glossary of Terms Used in Use Case Description

References

Books in alphabetical order

Larman Craig ”Applying UML and Patterns; An Introduction to Object Oriented Analysis and Design and Iterative Development” Third Edition Pearson Education, Inc 2005

Mathiasesen Lars, Munk-Madsen Andreas, Nielsen Peter Axel and Stage Jan ” Object Oriented Analyses and Design ” Marko Publishing ApS, Aalborg, Denmark

McGraw, Gary *Software Security - Building Security in*, Addison Wesley Professional January 23, 2006

Michael Howard, David LeBlanc, John Vega , “19 Deadly Sins of Software Security ; Programming Flaws and How to Fix Them” ; Chapter 11: “ Use of Weak Pass Word Based Systems “, McGraw-Hill/Osborne 2005

Web Sites in alphabetical order

Apple Federal Credit Union’s net banking security features available at : https://www.applefcu.org/nbsecurity.asp

“ ATM & BIOMETRICS: A SOCIO-TECHNICAL BUSINESS MODEL” , By Mario Yanez, Jr. University of Miami, School of Business Administration, CIS Department And Annuar Gomez University of Miami, School of Business Administration, MBA Program , available at : http://www.iamot.org/conference/index.php/ocs/4/paper/viewFile/1104/479

“ Bank of Central Asia Selects Identix Fingerprint Biometric Authentication Technology for Teller Verification of Electronic Transfers”, available at : http://ir.l1id.com/releasedetail.cfm?ReleaseID=208701

“Banque Saudi Fransi Deploys Biometric Authentication” By Anne Rawland Gabriel February 03, 2009 , available at : http://www.banktech.com/risk-management/showArticle.jhtml?articleID=213001004

Citi Bank Online Banking Site at: https://online.citibank.com/US/JPS/portal/Index.do

Convenience Driving Online Banking Growth
Published Monday, July 20, 2009 By Mike Sachoff on Web Pro News web site available at: http://www.webpronews.com/topnews/2009/07/20/convenience-driving-online-banking-growth

Facts on online banking at Danske Bank Web Site : http://www.danskebank.dk/en-dk/Personal/Pages/eBanking.aspx?tab=1#tabanchor

Fingerprint ID For Online Banking” Wednesday August 06, 2008 available at : http://news.sky.com/skynews/Home/Technology/Revolutionary-fingerprint-identification-for-online-banking-is-tested-by-Siemens/Article/200808115071356

"Fingerprint recognition technology seen as the most effective underutilized security enhancement to Internet banking", available at: http://www.prophis.com/pr/iBanking5_aug21.pdf

"First Biometric Fingerprint-Scanning ATM Machine installed in Poland." THE FORENSIC NEWS BLOG, Providing the latest forensic news from across the world of forensic science, published Thursday, 3 June 2010, available at:
http://forensic-news-blog.blogspot.com/2010/06/first-biometric-fingerprint-scanning.html

"Hacking Finger Print Scanners or: Why Microsoft's Fingerprint Reader is not a Security Feature "Available at : http://www.blackhat.com/presentations/bh-europe-06/bh-eu-06-Kiviharju/bh-eu-06-kiviarju.pdf

http://www.jera.com/techinfo/xpfaq.html
http://www.jera.com/techinfo/xpfaq.html
http://www.xprogramming.com/xpmag/whatisxp.htm#small
http://www.xprogramming.com/xpmag/whatisxp.htm#small
http://www.xprogramming.com/xpmag/whatisxp.htm#small
http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci214299,00.html
http://www.cryptographyworld.com/algo.htm

http://www.simple-talk.com/dotnet/.net-framework/beginning-with-digital-signatures-in-.net-framework/

http://dotnetslackers.com/articles/security/Hashing_MACs_and_Digital_Signatures_in_NET.aspx

http://www.simple-talk.com/dotnet/.net-framework/beginning-with-digital-signatures-in-.net-framework/

http://searchsoftwarequality.techtarget.com/definition/HTTPS

http://searchsoftwarequality.techtarget.com/definition/HTTPS

http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci343029,00.html

http://tutorial.visualstudioteamsystem.com/details.aspx?item=132

http://monduke.com/2006/06/04/the-fifteen-minute-guide-to-mutual-authentication/

http://www.xenossoftware.com/freetools/certificategenerator/

http://msdn.microsoft.com/en-us/library/aa478948.aspx

http://msdn.microsoft.com/en-us/library/aa478949.aspx

http://msdn.microsoft.com/en-us/library/aa478950.aspx

http://msdn.microsoft.com/en-us/library/aa478953.aspx

http://www.we-coffee.com/bonnie.aspx

http://www.guidgenerator.com/online-guid-generator.aspx

http://www.transcendusa.com/Products/ModDetail.asp?LangNo=0&ModNo=169

http://msdn.microsoft.com/en-us/library/aa479048.aspx#bucupro_topic11

http://www.xenossoftware.com/freetools/certificategenerator/

http://www.we-coffee.com/framework/cassandra-client-framework.aspx

JoelWeise - "Public Key Infrastructure Overview" SunPSSM Global Security Practice
Sun BluePrints™ OnLine - Sunmicrosystems August 2001 available at page 9
http://www.sun.com/blueprints/0801/publickey.pdf accessed on 14-02-2011

MyBank Tracker October 22nd, 2010 "Fingerprint ID Verification Preferred by Most Credit Card Users", available at: http://www.mybanktracker.com/bank-news/2010/10/22/63-consumers-choose-fingerprint-forms-identity-protection/

My Digital Life Web site article about South Africa's NedBank "From Nedbank to Netbank " ;
Published Wednesday, 18 June 2008 15:05 Written by Leon Engelbrecht and available at :
http://www.mydigitallife.co.za/index.php?option=com_content&view=article&id=1038163&catid=17:digital-connectivity&Itemid=36

Nedbank ‘s Net Bank Online Security Policy available at : https://www.nedbank.co.za/website/content/promotions/index_detail.asp?PromoID=529

Nordea Bank Denmark’s online banking account creation steps; available in Danish at : http://www.nordea.dk/Privat/Internet+og+telefon/R%c3%a5dgivning+om+internet+og+telefon/S%c3%a5dan+skifter+du+til+NemID/1381022.html

“Online Banking Growth Outpacing That of Call Centers, Branches, ATMs, Tower Group Report Says” By Katherine Burger published on MAY 21, 2007 on Bank Systems and Technology website available at: http://www.banktech.com/showArticle.jhtml?articleID=199905123

Online banking login page for Danske Bank at: https://netbank.danskebank.dk/html/index.html?site=DBNB&secsystem=DI

“Online Banking Fraud on the Rise” on PC Pro Magazine available at : http://www.pcpro.co.uk/news/356305/online-banking-fraud-on-the-rise By Hani Megerisi Posted on 10 Mar 2010

“The Advantages of Online Banking to Banks” available on Ehow Web site at: http://www.ehow.com/facts_5003054_advantages-online-banking-banks.html

Webopedia website article "How Fingerprint Scanners Work", available at http://www.webopedia.com/DidYouKnow/Computer_Science/2004/fingerprint.asp , Posted: 10-01-2004 , Last Updated: 08-31-2010

You Lina, , Xu Maozhib and Zheng Zhimingc "Digital signature systems based on smart card and fingerprint feature" Journal of Systems Engineering and Electronics Volume 18, Issue 4, December 2007, Pages 825-834; Abstract available at: http://tinyurl.com/25be4tk

Abbreviations in alphabetical order

ACID Atomic, Consistent, Isolated, Durable

ADO ActiveX Data Objects

AES Advanced Encryption Standard

API Application Programming Interface

ASP Active Server Pages

ATM Automatic Teller Machine

CA Certification Authority

DLL Dynamic Link Library

GUID Globally Unique Identifiers

HTTPS HTTP over SSL or HTTP Secure

ID Identity

IDE Integrated Development Environment

IIS Internet Information Service

LINQ Language Integrated Query

PC Personal Computer

PIN Personal Identification Number

PKI Public Key Infrastructure

RDMS Relational Database Management System

RSA Rivest-Shamir-Adleman

RMF Risk Management Framework

SEO Search engine optimization

SHA1 , SHA256 Secure Hash Algorithm

SDLC Software Development Lifecycle

SQL Structured Query Language

SSL Secure Socket Layer

TLS Transport Layer Security

UP Unified Process

XP Extreme Programming

VS Visual Studio 2010

Appendices

Database Table Definitions

1) Table Definition for aspnet_Users relation

Column Name	Data Type	Allow Nulls
ApplicationId	uniqueidentifier	☐
UserId	varchar(50)	☐
UserName	nvarchar(256)	☐
LoweredUserName	nvarchar(256)	☐
MobileAlias	nvarchar(16)	☑
IsAnonymous	bit	☐
LastActivityDate	datetime	☐
		☐

2) Table Definition for aspnet_Membership relation

ApplicationId	uniqueidentifier	☐
UserId	varchar(50)	☐
Password	nvarchar(128)	☐
PasswordFormat	int	☐
PasswordSalt	nvarchar(128)	☐
MobilePIN	nvarchar(16)	☑
Email	nvarchar(256)	☑
LoweredEmail	nvarchar(256)	☑
PasswordQuestion	nvarchar(256)	☑
PasswordAnswer	nvarchar(128)	☑
IsApproved	bit	☐
IsLockedOut	bit	☐
CreateDate	datetime	☐
LastLoginDate	datetime	☐
LastPasswordChanged...	datetime	☐
LastLockoutDate	datetime	☐
FailedPasswordAttemp...	int	☐
FailedPasswordAttemp...	datetime	☐
FailedPasswordAnswe...	int	☐
FailedPasswordAnswe...	datetime	☐
Comment	ntext	☑
		☐

3) Table Definition for OnlineAccount2 relation

Column Name	Data Type	Allow Nulls
UserID	varchar(50)	☐
AccountID	int	☐
		☐

4) Table Definition for RealPayments2 relation

Column Name	Data Type	Allow Nulls
PaymentsId	int	☐
Date	smalldatetime	☐
AccountID	int	☐
BeneficiaryAccountID	int	☐
Amount	money	☐
		☐

5) Table Definition for UserProfile relation

	Column Name	Data Type	Allow Nulls
▶	FirstName	varchar(50)	☐
	LastName	varchar(50)	☐
	Telepnone	varchar(20)	☐
	Address	varchar(100)	☐
🔑	UserID	varchar(50)	☐

Implementation Codes[124]

Web.config file

```
<?xml version="1.0"?>

<!--
  For more information on how to configure your ASP.NET application, please
 visit
  http://go.microsoft.com/fwlink/?LinkId=169433
  -->

<configuration>
  <connectionStrings>
    <add name="ApplicationServices" connectionString="data source=.\SQLEXPR
ESS;Integrated Security=SSPI;AttachDBFilename=|DataDirectory|\aspnetdb.mdf;
User Instance=true"
      providerName="System.Data.SqlClient" />
    <add name="ConnectionString" connectionString="Data Source=.\SQLEXPRESS
;AttachDbFilename=|DataDirectory|\ASPNETDB.MDF;Integrated Security=True;Use
r Instance=True"
      providerName="System.Data.SqlClient" />
  </connectionStrings>

  <system.web>
    <customErrors defaultRedirect="~/ErrorPage.aspx" />
    <compilation debug="false" targetFramework="4.0" />

    <authentication mode="Forms">
      <forms loginUrl="~/Account/Login.aspx" timeout="2880" />
    </authentication>

    <membership>
      <providers>
        <clear/>
        <add name="AspNetSqlMembershipProvider" type="System.Web.Security.S
qlMembershipProvider" connectionStringName="ApplicationServices"
             enablePasswordRetrieval="false" enablePasswordReset="true" req
uiresQuestionAndAnswer="false" requiresUniqueEmail="false"
             maxInvalidPasswordAttempts="3" minRequiredPasswordLength="36"
minRequiredNonalphanumericCharacters="1" passwordAttemptWindow="10"
             applicationName="/" />
      </providers>
```

[124] This shall be limited to the web.config files and the code behind files. Code for the aspx pages are voluminous . They can be seen from the code in the CD that has been handed in.

```
    </membership>

    <profile>
      <providers>
        <clear/>
        <add name="AspNetSqlProfileProvider" type="System.Web.Profile.SqlPr
ofileProvider" connectionStringName="ApplicationServices" applicationName="
/"/>
      </providers>
      <properties>
        <add
          name="FPHash"
          allowAnonymous="false" />
      </properties>

    </profile>

    <roleManager enabled="true">
      <providers>
        <clear />
        <add connectionStringName="ApplicationServices" applicationName="/"
          name="AspNetSqlRoleProvider" type="System.Web.Security.SqlRolePro
vider" />
        <add applicationName="/" name="AspNetWindowsTokenRoleProvider"
          type="System.Web.Security.WindowsTokenRoleProvider" />
      </providers>
    </roleManager>

  </system.web>

  <system.webServer>
     <modules runAllManagedModulesForAllRequests="true"/>
  </system.webServer>
</configuration>
```

DivineOnlineBank/Protected Pages/FingerprintHash.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class Protected_Pages_FingerprintHash : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

        if (!IsPostBack)
        {
            FPtxt.Text = Profile.FPHash;

        }
    }
```

```
    protected void FPbtn_Click(object sender, EventArgs e)
    {
        Profile.FPHash = FPtxt.Text;
        Response.Redirect("Start.aspx");
    }
    protected void FPHButton_Click(object sender, EventArgs e)
    {
        FPHashLabel.Text = Profile.FPHash;

    }
}
+
```

5. SignerButtons/customSignerButton.aspx.cs

```
using System;
using System.Web.UI;
using System.Web.UI.WebControls;
using Cassandra.Framework.Cryptography.Controls;
using Cassandra.Framework.Cryptography.Networking;


namespace Cassandra.Framework.Examples.WebDemo
{

    /// Test Page

    public partial class PageCustomSignerButton : Page
    {

        Label _exceptionControl;
        /// Override

        protected override void OnLoad(EventArgs e)
        {
            base.OnLoad(e);

            if (!IsPostBack)
            {
                /* Disable the co-sign button. */
                btCoSign.Enabled = false;
            }


            //Set the custom function to use
            btSign.CustomScriptFunction = "CreateDocument()";


            //if exceptions are managed on the client, set the control to use
            //to display exceptions messages on it.
            if (!btSign.PostBackExceptions)
            {
                btSign.SetControlForException(lbException);
            }
```

```
            //Set the control that will host the description to add to the
signature.
            btSign.SetControlForDescription(txtNote);

            //if exceptions are managed on the client, set the control to use
            //to display exceptions messages on it during the co-signature.
            if (!btCoSign.PostBackExceptions)
            {
                btCoSign.SetControlForException(lbCoException);
            }

            //Set the control that will host the description to add to the co-
signature.
            btCoSign.SetControlForDescription(txtCoNote);


            //set the event handlers raised after the signatures
            btSign.OnSignatureDone += new
EventHandler<SignatureDoneEventArgs>(btSign_OnSignatureDone);
            btCoSign.OnSignatureDone += new
EventHandler<SignatureDoneEventArgs>(btCoSign_OnSignatureDone);

            //internal use
            _exceptionControl = lbException;

        }



        /// Event handler for the signature

        void btSign_OnSignatureDone(object sender, SignatureDoneEventArgs e)
        {
            ClearExceptionMessages();

            //if the client generate an exception, the event can manage it (if
PostBackExceptions is true)
            if (e.HasException)
            {
                //exceptions are reported as enum value. This permits to manage
them on the server
                ManageError(e.ClientException.Message, _exceptionControl);
                return;
            }

            /*
                The signature computed is returned by the client as string
base64 encoded.
                Itis contained on the e.SignedMessage property. This value must
be save somewhere
                (for example a database)
             */


            /*
                The signature and its attributes can be browsed and managed
with the
                Pkcs7StringSignatureBrowser class.
             */

            Pkcs7StringSignatureBrowser browser = new
Pkcs7StringSignatureBrowser(e.SignedMessage);
```

```
            //verify the signature computed
            if (!browser.VerifySignature())
            {
                lbException.Text = "Signature not valid.";
                return;
            }

            //Show on the web page the message signed and the info about the
signer
            ReportSignatureData(browser);

            //activate the co-Sign button
            btCoSign.Enabled = true;

            //set the data to co-Sign (the previous generate signature that can
be
            //retrieved form a database)
            btCoSign.DoCoSignature(e.SignedMessage);

        }

        /// Event Handler for the CoSignature
        void btCoSign_OnSignatureDone(object sender, SignatureDoneEventArgs e)
        {
            //internal use
            _exceptionControl = lbCoException;
            btSign_OnSignatureDone(sender, e);
        }


        /// Show on the web page the message signed and the info about the
signer

        private void ReportSignatureData(Pkcs7StringSignatureBrowser browser)
        {

            Label l = new Label();
            pCustomBrowser.Controls.Clear();
            pCustomBrowser.Controls.Add(l);

            l.Text = "Signed Message: " + browser.SignedString.Replace("<",
"&#60;").Replace(">", "&#62;") + "<br/><br />";

            foreach (Pkcs7SignatureInfo i in browser.SignatureInfoCollection)
            {
                l.Text += "message signed by : " +
i.SignerIdentityInfo.CommonName;

                DateTime d = i.UTCSignatureDate.ToLocalTime();
                l.Text += " on: " + d.ToShortDateString() + " - " +
d.ToShortTimeString();
                l.Text += "<br /> hash of fingerprint: " +
i.SignatureDescription;
                l.Text+=  "<br /><br />";
            }

        }


        /// Manage the OnSignatureDone event error status
```

```
        private void ManageError(string message,Label labelForException)
        {
            labelForException.Text = message;
        }


        /// Clear the exceptions messages

        private void ClearExceptionMessages()
        {
            lbException.Text = "";
            lbCoException.Text = "";
        }
    }
}
```