GENERATING ONE-TIME PASSWORD BY WEB-BASED TECHNOLOGY-NEW APPROACH FOR AUTHENTICATION ON WEB-BASED SERVICE

ANUSORN KANYAPRASANKID

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR
THE DEGREE OF MASTER OF SCIENCE
(TECHNOLOGY OF INFORMATION SYSTEM MANAGEMENT)
FACULTY OF GRADUATE STUDIES
MAHIDOL UNIVERSITY
2005

ISBN 974-04-6092-5

Thesis

Entitled

GENERATING ONE-TIME PASSWORD BY WEB-BASED TECHNOLOGY-NEW APPROACH FOR AUTHENTICATION ON WEB-BASED SERVICE

……

Mr.Anusorn Kanyaprasankid

Candidate

……

Mr.Worawit Israngkul,

M.S.(Technical Management)

Major-Advisor

……

Mr.Chumchok Namsrisakulrat,

M.Eng.(Electrical Engineering)

Co-Advisor

……

Assoc.Prof.Rassmidara Hoonsawat, Ph.D.

Dean

Faculty of Graduate Studies

……

Assoc.Prof.Piya Rattanasuwan, M.Eng.

Chair

Master of Science Programme in Technology of Information System Management

Faculty of Engineering

Thesis
Entitled

GENERATING ONE-TIME PASSWORD BY WEB-BASED TECHNOLOGY-NEW APPROACH FOR AUTHENTICATION ON WEB-BASED SERVICE

Was submitted to the Faculty of Graduate Studies, Mahidol University
For the degree of Master of Science
(Technology of information Management)
on
9 May, 2005

……
Mr.Anusorn Kanyaprasankid
Candidate

……
Mr.Worawit Israngkul,
M.S.(Technical Management)
Chair

……
Mr.Surachai Jonjerdsin,
M.S.(Computer Information System)
Member

……
Mr.Chumchok Namsrisakulrat,
M.Eng.(Electrical Engineering)
Member

……
Assoc.Prof.Rassmidara Hoonsawat, Ph.D.
Dean
Faculty of Graduate Studies
Mahidol University

……
Assoc.Prof.Piya Rattanasuwan, M.Eng
Dean
Faculty of Engineering
Mahidol University

ACKNOWLEDGEMENT

The success of this thesis can been attributed to the extensive support and assistance from my major advisor, Mr. Worawit Israngkul M.S. (Technical Management) and my co-advisor, Mr. Chumchok Namsrisakulrat M.Eng. (Electrical Engineering) I deeply thank them for their valuable advice and guidance in this study.

I have a number of people to thank for their sincere help, their criticisms and suggestion concerning various parts of my dissertation. Among them are all of my colleagues, whose intelligent and valuable information including the first draft of literatures and some devices, which can be used in this study. I would like to extend my thanks to junior staffs in my company and many persons who help me accomplish product's evaluation.

I wish to thank my best friend for her sincere help and concentration in my progress. A part of my intention came from the attention of this person. I would like to share her the successful of my final examination.

Finally, I am grateful to my parent and my family for their support, entirely care, and their endurable for my graduation. I understood them since I was young that they are waiting to see my successful. I would like to dedicate this thesis to them and hopefully this thesis is the one of my successful, which can be the great prize for our family.

Anusorn Kanyaprasankid

GENERATING ONE-TIME PASSWORD BY WEB-BASED TECHNOLOGY-NEW APPROACH FOR AUTHENTICATION ON WEB-BASED SERVICE

ANUSORN KANYAPRASANKID 4337181 EGTI/M

M.Sc.(TECHNOLOGY OF INFORMATION SYSTEM MANAGEMENT)

THESIS ADVISORS WORAWIT ISRANGKUL, M S TECHNICAL MANAGEMENT , CHUMCHOK NAMSRISAKULRAT, M Eng ELECTRICAL ENGINEERING

ABSTRACT

The purpose of this study is to analysis feasibility to apply technology of generating dynamic password to the special authentication process on web-based service. Main concept of this study refers to the concept of token device, device-based technology of generating one-time password, which has the crosscheck process with the server side application. Some applied sub processes in this approach refer to challenge-response system.

Results revealed the possibility of applied processes to use one-time password instead constant password in the specific transaction of web-based service. Although the alternative authentication approach was related to the increasing conviction in beneficial transaction on public web-based service, the system still needed the constant password in case of authentication for beginning transaction. The complication of overall processes related to the confusion of users, who never knew about challenge-response concept, to understand the specific type of service.

These findings suggest that there are some processes in this approach that have to be improved for more security. The alternative authentication by generating one-time password should be considered to improve the algorithm of encoding data in case of using by real public web service. In addition, there are many technologies that can be considered to apply to this alternative approach such as apply handheld device to be one-time password generator or using wireless equipment instead email channel to inform message to client users.

KEY WORDS: ONE-TIME PASSWORD / DYNAMIC PASSWORD / WEB-BASED AUTHENTICATION / TOKEN

153 P. ISBN 974-04-6092-5

แนวคิดใหม่ในการตรวจสอบผู้เข้าใช้บริการผ่านเว็บ โดยใช้เทคโนโลยีในการกำหนดรหัสผ่านเพื่อเข้าใช้ระบบในแต่ละครั้ง (GENERATING ONE-TIME PASSWORD BY WEB-BASED TECHNOLOGY-NEW APPROACH FOR AUTHENTICATION ON WEB-BASED SERVICE)

อนุสรณ์ กันยาประสานกิจ 4337181 EGTI/M

วท.ม. (เทคโนโลยีการจัดการระบบสารสนเทศ)

คณะกรรมการควบคุมวิทยานิพนธ์ : วรวิทย์ อิศรางกูร, M.S. (Technology Management), ชุมโชค นำศรีสกุลรัตน์, M.Eng. (Electrical Engineering)

บทคัดย่อ

การศึกษาครั้งนี้เป็นการศึกษาเพื่อวิเคราะห์ความเป็นไปได้ในการพัฒนาเทคโนโลยีการกำหนดรหัสผ่านแบบใช้ครั้งเดียว โดยประยุกต์เข้ากับกระบวนการการตรวจสอบผู้ใช้ระบบแบบพิเศษบนระบบการให้บริการผ่านเว็บ โดยแนวคิดหลักในการศึกษาจะพัฒนาจากแนวความคิดในการใช้อุปกรณ์เครื่องมือเพื่อการกำหนดรหัสผ่านแบบใช้ครั้งเดียว ซึ่งระบบตรวจสอบก็จะมีวิธีการตรวจสอบรหัสอย่างสอดคล้อง นอกจากนี้แนวความคิดของระบบตรวจสอบแบบ Challenge-Response ถูกนำมาใช้เป็นแนวทางสำหรับการศึกษานี้ด้วย

ผลการศึกษาได้แสดงบทวิเคราะห์เพื่อการประยุกต์ใช้เทคโนโลยีการกำหนดรหัสผ่านแบบใช้ครั้งเดียวทดแทนการใช้รหัสผ่านแบบคงตัวในการเข้าใช้บริการชนิดพิเศษบนระบบการให้บริการผ่านเว็บ ถึงแม้ว่าแนวความคิดใหม่นี้จะมีส่วนช่วยเพิ่มความเชื่อมั่นที่จะเลือกใช้ระบบบริการสาธารณที่เกี่ยวเนื่องกับธุรกิจผ่านทางเว็บ แต่ผลการวิเคราะห์แสดงให้เห็นว่ายังคงมีความต้องการใช้ระบบการตรวจสอบโดยใช้รหัสผ่านแบบคงตัวในกรณีของการเริ่มต้นเข้าสู่ระบบก่อนเข้าใช้ระบบการตรวจสอบผู้ใช้บริการแบบพิเศษ อีกทั้งความยุ่งยากของกระบวนที่เพิ่มมากขึ้นทำให้เกิดความสับสนแก่ผู้ใช้ระบบ โดยเฉพาะผู้ใช้ที่ไม่คุ้นเคยกับระบบ Challenge-Response

จากการศึกษาครั้งนี้มีข้อเสนอแนะว่าควรมีการพัฒนากระบวนการต่างๆเพื่อเพิ่มความปลอดภัยให้กับการตรวจสอบผู้ใช้ระบบถ้าหากมีการนำไปใช้จริงในระบบการให้บริการผ่านเว็บ โดยเฉพาะการพัฒนาวิธีการเข้ารหัสข้อมูล และยังมีเทคโนโลยีอีกหลายประเภทที่สามารถนำมาประยุกต์เข้ากับแนวความคิดนี้ อย่างเช่น การใช้อุปกรณ์มือถือต่างๆเป็นตัวกำหนดและแสดงรหัสผ่านแบบใช้ครั้งเดียว หรือ การใช้ช่องทางการสื่อสารแบบไร้สายทดแทนการส่งข้อมูลผ่านอีเมล์

153 หน้า ISBN 974-04-6092-5

CONTENTS

Page

CONTENTS (CONT.)

LIST OF FIGURES

Page

LIST OF FIGURES (CONT.)

Page

CHAPTER I
INTRODUCTION TO THE STUDY FOR ALTERNATIVE APPROACH OF AUTHENTICATION ON WEB BASED SERVICE

1.1 Introduction

It is interesting to note that some of commercial functions are not implemented on web base application in Thailand. Some of users could not be certain by security of business web site that occur only entering acclimatized User Id and Password into 2 input controls on logon web page. They cannot ensure that anyone else could not catch their User Id and Password then accessed to their information or snatching their authorization to do manipulations that can harm their business. The businessmen, who need to transfer number of their deposition to another one via banking web base service, are not deeply confined that their static User Id and Password can be always hidden from anyone and this constant authentication remain identify to themselves accuracy.

There are many interested approaches that can wipe out lax authentication on public networks – IP source filtering, Proxies (for authentication), cryptographic certificate, Smart card solution (such as USB Minikey, Token, Security Smart card and Smart card reader), Application Security, etc.

Methodology to generate dynamic password is aimed for this study. Combination of the token equipments was selected to be a prototype of new approach for generating dynamic Public-Private key by browser-based protection. Whenever users need to access their information via web browser, they should receive strange key before login on.

1.2 Background

Prominence of token technology, enable to generate dynamic passwords that are event-synchronous with a trusted institutional server. This one-time password[1] can be used once to ensuring secure access, originate computing software based token, which has same function as token[2] (convenient key fob package). This kind of application needs installation process to settle down itself on personal computer (or lap top).

It's important to note that almost users have same behavior about making use of web-based service – they should have to use web browser anywhere to access to their information. So completeness of generating dynamic password on web browser (for users to using it to perform their authentication via preceding web browser) is a main subject for this study.

Cracking the token key fop, two-factor authentication[3] is a factor that user has to enter 4- digit personal identification number (PIN) to provide a one time dynamic password (6-digit). Then they can enter the password (maybe unite this password with static PIN) into the Web or network application to perform authentication. If token was stolen, the thief cannot get password by stolen property without PIN. Procedure of providing new password, processing within token function, must have particularly algorithm that synchronize with server algorithm for generating randomize password. Both of coincident codes were ever changed every minute (or appropriate interval) so, 6-digit password would be regenerated every minute on the both of generator. But

[1] More complex password, "stronger" passwords consisting of arcane and often lengthy combinations of characters that are changed at regular intervals frustrate users, encourage them to write down their passwords and fuel increased support calls that drive up operational costs. (RSA Security: Open Specifications Integrate One-Time Passwords with Enterprise Applications) 3.

[2] A hardware authentication device, or a software authenticator that runs on another platform, such as a laptop, PDA or phone. (RSA Security: Open Specifications Integrate One-Time Passwords with Enterprise Applications) 9.

[3] A strong means of proving identity in which a user enters something he knows (a PIN) and something he-or-she has (the OTP displayed or generated by his-or-her token). (RSA Security: Open Specifications Integrate One-Time Passwords with Enterprise Applications) 9.

nevertheless entering 4- digit PIN was just an operation to trigger off token to display digits of latest code.

1.3 Objective

Essential of this study is the transforming of authentication device concept to additional processing on client side in aspect of ability to generate one time password that synchronous with verification procedure on server side.

Matters of this study are the following:

- Exhausted study in existing technology of authentication device and their elemental factors that cooperate to construct the methodology of one time password generating and its consistent verification.
- Study for appropriate proceeding to combine the concept of generating one-time password by Token and ability of web-based technology to extend functionality.

1.4 Expected Result

This study needs to propose alternative approach to avoid habitual static password. (When the users access a site and they usually see 2 input boxes that ask for User ID and Password) Not only study the concept of existing solution – Token device, but also apply this concept to develop web-based technology to complicate authentication with generating dynamic password.

So, expected results of this study are the following:

- Concept of one-time password and token technology that can be applied to web-based technology.
    - Transform digits of password by appropriated algorithm, which has particularly factor depend on timing. (Ex. generate new password every minute.)
    - These passwords can be used to perform personal authentication at one time. (Receiving latest password must accurately synchronize with latest information that was stored on server side)

  - Trigger off machine to display current password by entering PIN code.
- Simulated case study – one of web service is forced to authenticate their members who access the site with new generated password every time they logon.
  - All of members must generate their dynamic password by their client application before enter Username and latest Password to the site
  - For new users or old members who want to change their machine, this web site should have easy way for them to modify their local machine wherever to encourage this authentication.

1.5 Scope of work

- Study for the probability that can set up authentication process on web service by the concept of one-time password.
  - For this study, new authentication approach is just only concerned on process how to generate one-time password by client users, process of accessing with one-time password through special service of web-based service, process of verification on receiver host.
  - All of gathered consideration in this study were not included the unexpected problems that was motivated by various reason:
    - Human error on authentication progress or human feelings that new approach may be refused because of more complicated access.
    - Communication problem on Internet traffic.
    - Unexpected operation to steal user information or cracking source code of both side of application-web application on server side and local token application on client side.
    - Incompatible platform of each client machine (Web server: simulated web server by The Tomcat 4.1 Servlet/JSP

Container on Windows 2000, Client web browse: Internet Explorer 5.5 on Windows 2000)

- Study for general specifications of authentication device that are corresponded for one-time password generation on web-based technology
  - Qualifications that can assist human to compute the complex hash function so that generate one-time password in the pattern for challenge-response system or time-based algorithm.
  - Not include:
    - Complex specifications those are not necessary for the authentication on web-base service.
- Using web programming and browser plug in technology as tools for created one-time password and verification on server side.
  - Web programming: Java, JSP, JavaScript and HTML
  - Data base: relational database-MS Access 2000
  - Client application: executed application, which was compiled by java programming and was build to be installer by InstallAnywhere 5.5[4].
- Study for appropriate hash function that support for different level of security.
  - Studies for the simplest algorithm for encryption-decryption, Caesar's cipher[5] text algorithm, to be manuscript for initiate the simplest algorithm, which has the time value as the factor for generating the key of encode.
  - Initiate the simplest encryption-decryption algorithm, which has the dynamic key of encode-decode, for simulate web-based service case study.

[4] InstallAnywhere 5.5 offers a host of powerful, flexible, and intuitive new features that cut development times and ease complex software deployment across the enterprise (http://www.zerog.com/newsItems/news_031905_ia55release.html, 2005)

[5] Charles P. Pfleeger (Security in computing: Challenge-Response Systems, 1989) 26.

- Using Object-Oriented Analysis & Design (OOAD) to represent the results of each step in this methodology.
  - UML: Unified Model Language
  - Scope of each model just represent for domain obligation that involve to authentication process of specific web-based service.

CHAPTER II
LITERATURE REVIEW

2.1 One-Time Password (OTP)

A basic authentication scheme based on static password, which can be intercepted and replayed by the others, is vulnerable technique, even major people were intimately acquainted it by simple components as two text boxes that request their recollected login ID and password.

Principal concept of One-Time Password (OTP) is process of generating series of password that are used to access to the system. Once one of password is used, it cannot be used again. The system always expect new password at the next logon. So post password was be eliminated. It would become worthless digit even someone could intercept it.

Clients have to create the series of passwords by complicated algorithm that usually put in the password generator. (Software or Hardware token) Then at the sever site must have the corresponding decryption algorithm for cracking a one-time password that client user sent it over the wire and verify that isn't invalid then authenticate user.

The type of One-Time Password generating fall into two categories: a challenge-response system or time-based algorithm.

2.2 Challenge-Response system

A challenge-response system[6] is essentially cryptosystem in which the server sends the message as a challenge-encryption message to the clients. Then clients can decrypt that challenge with their tool. The users will be ensured that this legal communication was performed by a really host. After that they have to input their

[6] Charles P. Pfleeger (Security in computing: Challenge-Response Systems, 1989) 388.

secret password and encrypt it with a challenge. They will perform the response as a new password at the time.

For generating OTP, the challenges, which were sent by the host, must unique. The following are two sample challenges.

- "Host name is A, what's your secret code?"

  When a client combine his secret code "ABCD" before client encrypt it. This message maybe become "Host name is A, what's your secret code?ABCD". With this scheme after this logon user will be need to get the challenge message for new access. If new challenge is not different from the previous one the response is remain as "Host name is A, what's your secret code?ABCD" that can be intercept and replayed by the others.

  The following is a better challenge.

- "Host name is A, Time is 01/08/2003 10:38:22 AM, what's your secret code?"

  This challenge includes the time stamp that can identify when it was sent. Then user unites his password then encrypts and sends it to the server. He can wait new challenge without doubtfulness. A new secure challenge will be sent to him, as he needs it for new accessing. The sample next challenge maybe is generated, as "Host name is A, Time is 16/09/2003 07:15:30 PM, what's your secret code?"

  There is no use for anyone who intercept the old response "Host name is A, Time is 01/08/2003 10:38:22 AM, what's your secret code?ABCD" (a response which interceptor get is represented in form of encryption message) . The digits, was stolen, would be the waste. Because the security process can track that this response was used once at the past.

For verification on the server side, server received a one-time password from generator (response which contain challenge message concatenate with client's secret code) then verify it by computing the secure hash function once and comparing the result with the previous accepted one-time password that was stored as each client data on the server. If the result of this operation matches the stored previous OTP, the

authentication is successful and the accepted one-time password is stored for future use.

2.3 Time-based Algorithm

For this kind of system, client users have to carry a token device that displays the correct response of the moment. The response on screen of device always change every 60 seconds. (or initiated different ratio) But some kind of token requires a PIN (Personal Identification Number) to be entered to activate the correct response in a device. User has to enter his PIN that was known by only him/her before algorithm inside would be triggered to compute for correct response at a moment.

There is an advantage of time-base system that user has not to handle a challenge from server. Every time that the token is activated (without challenge message), device owner can get a response that can be used for authentic himself or herself.

But disadvantage of this system is that it may be vulnerable to replay attacks for a short time period unless properly implemented. By the way this system also is limited interoperability for user to enter a response to system at a moment.

However, both types of OTP generating (challenge-response system and time-based algorithm) have a union interested concept that client users need to use a tool such as token hardware or software to compute for a different password (response). Because of the necessary hash function or encryption/decryption algorithm (such as MD4, MD5, DES, RSA, etc.) drive OTP generating process is too complex for human ability.

2.4 Authentication Device

For OTP (One-Time Password) System such as challenge-response system, the series of passwords is created by the client, which combines a seed value with a secret password that only the client knows. This combination is then run through either the MD4 or MD5 hash functions repeatedly to generate the sequence of

passwords. While OTP generating as time-based algorithm, the series password is created by compute an encrypted form of a number unique of device (token serial number) and the current date and time.

With reasons above, complicated of hash function made the deep confusion for human to compute for their OTP creating or decrypt necessary information. The duty of authentication device such as smart card, token or response generator is a necessary factor of the system.

Token is the name of the object that can identify its processor. The token must be unforgeable and unique.

Smart card is one kind of device that has similar obligation with token. It has a microprocessor embedded that can retain the necessary information of its owner such as secret code, hash algorithm or etc. With potential of microprocessor, it can also actually perform computation such as computing for the response message of challenge-response system.

When someone wants to initiate a logon from terminal to the network, he will receive the prompt for the password (challenge). After a smart card is putted into a slot, he can type his password (secret code). Instead of message (challenge and response) being transmit in the clear, the message is encrypted by the smart card. In this way, the smart card owner can carry on a secure session with a computing network from anywhere.

SecurID (token that represent in form of key-fob) has the other way to generate OTP. It generate new password every 60 seconds. The inside processor may compute an encrypted form of unique number of token (serial number that adhere with token from manufacturing process) unite by current date and time, *E(ID+Time)*. With unique serial number, this token can guarantee the authenticity of its owner. With time value, is can guard against interceptor trying to replay a previous password. The new version of SecurID was increased more complicated function by include entering PIN code that only owner know. Before activate this device to generate OTP, user has to enter his PIN to verify that he is the genuine token owner.

One kind of specific device is most corresponding to challenge-response system. It has shape as a calculator pocket. User get a challenge message from a host then key the challenge into the pocket. This digital device undertakes the task to

compute for the response. And user read its result on display screen then enters it to the computer keyboard.

It's important to note that the generator and server must use the same algorithm in order to interoperate.

2.5 Two-Factors authentication

Authentication methods, which base on static password, are not secure enough for protecting secret business information. This is important reason drive the reduction of end-users' reliability on web-base service system.

Several systems need the following two important factors for investigating before a user will gain access to a system

- Something user has: This factor includes Keys, Token and so on. Something user has is a tool that can identify who the user is. This tool can be stolen or lost.
- Something user knows: This factor includes password or PIN code that the owners always remember it. But this factor also can be stolen or sniffed by the others.

With token authentication these two factors were included in its security approach. For something user has, the each member user of token smart system usually has token device. The token device, password generator reveals a unique password to its owner each time it is used, can identify accurately that the owner who carry on this device could activate it and got encryption code to enter to the system at time. For something user know, before user can activate token to generate OTP the user have to enter their PIN code (Personal Identification Number) to the token. This PIN code was setting initially by user who owns the device so this PIN is something user knows. By the way, before they access to system (network or web-site) they must get the actual one-time password from their own device. Before accessing OTP which user get is also something user know.

Although each of factors can be stolen or intercepted by the others, the good point of two-factors is the interceptors have to stolen the both of factors so that they can stealthily grant their access.

Secure web-based service system should be considered on this Two-Factor concept.

CHAPTER III
METHODOLOGY TO ANALYSIS AND DESIGN FOR ALTERNATIVE AUTHENTICATION APPROACH

For the study, this project would be started with research for concept and features of exist methodology – token device, which is current widespread technology that involve directly with this principle objective – authentication with generating one time password. Expected results of this pilot study would be analyzed and applied to the essential productions of this project – client application for generating one time password and procedure of authentication on web service site.

The matter of this study is looking for the inheritable functions of two authentication approaches, which appended by generating one time password – token device and installed application on client machine. So, Modeling Language would be selected to present the system functionalities as diagrams. The main objective of all diagrams is the demonstration for adaptation of authentication processes. All diagrams were represented the concept of alternative approach. Although the domain of this study was imitated from the process of authentication with the password generator device, domain of adaptation ideas were represented by the diagrams in term of Object-Oriented Analysis and Design and concept of design would be scoped by united problem domain.

This research methodology would be started with investigation for token authenticated functions. This beginning part of research intent on “how do users work with this authenticated technology?”

3.1 Step 1: Study on functions of authentication by token device

This step focuses on behavior of existing users who have ever used token device (password generator – key fob shape).

- Gather information that involve with all of the cases that users would be driven to use this device for gain access to web service.
- Analyze information to find what is domain of interest.
- This step returns featured result as list of use cases (sub systems that were covered by problem domain), which can be used to analyze for the domain of using OTP on web-based authentication.

Next step, system processes of this authenticate method will be cracked. This part intent on “how the system does?”

3.2 Step 2: Investigate on token authentication system

Study on this authentication technology exhaustively.

- Gather information from public medias (journal, website, etc.) or request for additional information from token device provider (RSA Security Inc.).
- Gather additional information from system administrators of the company, which have ever been customer of token technology.
- Analyze information to find what are system functionalities that were covered by problem domain.
- This step returns the answers about how to generate one time password (by token device) and how can system authenticate user who login by this password. The concept of this step will be considered in the next part of study to finding out the necessary components.

Next step, this research would be continued by start to study for integration of web service authentication and token authenticate method. The following part intent on “What are the differential requirements, when all operations turn to base on web based technology? ”

3.3 Step 3: Study to integrate functional requirements

This step was interested on functional requirements when the one-time password generating processes were transformed from device base method to web base method and this step also focus on changing authentication procedure.

Study to analyze what're the matter authentication requirements for both of client side and server side users.

- Analyze processes that involve with all of the cases that users would be driven to use this web base one-time password authentication in accordance with same domain of interest, which was from first step of this study.
- Analyze processes that involve with the password validation and accurately authenticate user who logon with one-time password.
- Analyze the lists of sub systems that were gathered from first step compare with list of use cases for new approach, then modify use case diagrams according to web base one-time password authentication
- This step returns featured result by list of use cases and use case diagrams that can be represent the domain of this approach.

Next after, following step intent on "How to apply the system for new limitation of requirements? " Necessary of this step are system adaptations that can be taken to support for web base authentication.

3.4 Step 4: Adaptation of token authentication system

Study to analyze on existing system on functionality in an aspect of ability to adapt the system from device base system to web base authentication system.

- Analyze token system functionalities that were covered by same problem domain but has the limitation by web base architecture. (Analyze on both of one-time password generating procedure and validation/authentication procedure in term of web base service)

- Finding out the appropriated component on web based authentication by OTP then represent all of components and their relationship by Class diagrams and Sequence diagrams.
- This step returns featured result by various system diagrams that participate to present the functionalities of new authentication approach base on web architecture.

Finally, with analyzed system that become from previous method. The produce of study should be tested by simulation case.

3.5 Step 5: Simulate case study and implement demonstrative product

One site of web service would be simulated for testing the new authentication approach. And this site has some simultaneous users who use new auxiliary function with web browser to generate one-time password for accessing to their information on this site.

- Provide necessary information for testing ability to authenticate each user who logon to the site such as login id, user profiles, information contents for each user and etc.
- Provide web service site, which has specific authentication system with according to the result of analyzed system
- Testing for authenticate functionality of this simultaneous site.
- Finding out the usefulness and weak point of this study case.

This case study also includes the product evaluation by verbal interview with volunteers for the purpose of discussion to find out the details of problem in term of both business processes and technology.

Not only volunteers' evaluation will be included to a part of conclusion to prove the feasibility for using One-Time Password as a part of authentication on web-based service but also this study need the idea of users to improve this kind of technology for the future opportunity.

CHAPTER IV
WEB-BASED AUTHENTICATION APPROACH USING ONE-TIME PASSWORD

General analysis for finding out the appropriated business solution can be considered base on the various approaches. Object-Oriented Analysis and Design, esteemed analysis approach, is consented consideration for this study to conduct the various output diagrams that can be clearly represented the results of analysis while some kinds of diagram can be transferred to the step of development or Object-Oriented Programming.

In this section, outputs of analysis for finding out the alternative approach are the expectation of this study. Use Case diagram can be considered to represent the domain of web-based authentication approach while Class diagrams and Sequence diagrams were representing the logical view of system functional.

4.1 Use Case diagram: represent problem domain of alternative approach

This study was underlining on conceivable of solution to apply the One-Time password to be auxiliary mechanism of Web-based authentication. Background of this study is the concept of Token, the device generate randomize numeric (6-8 digits) in every constant interval (every 60 seconds). The domain of existing system, which holds on token device as procedure of authentication, would be considered for this study. These following are the considered subjects, which came from the domain of requirement.

- User registration
- Activate password generator
- Protection from larcenous
- Password crosscheck.
- Verification and grant access

- Accessing anytime and anywhere

All above subjects are the overall material, which came from the conclusion of studying on exist system by concentrate product information (token device of RSA SecurID) and conversational gathering information from exist users and exist administrator. By consideration of alternative web-based approach, the following Use Case diagram can be represented the overall of adapted system.

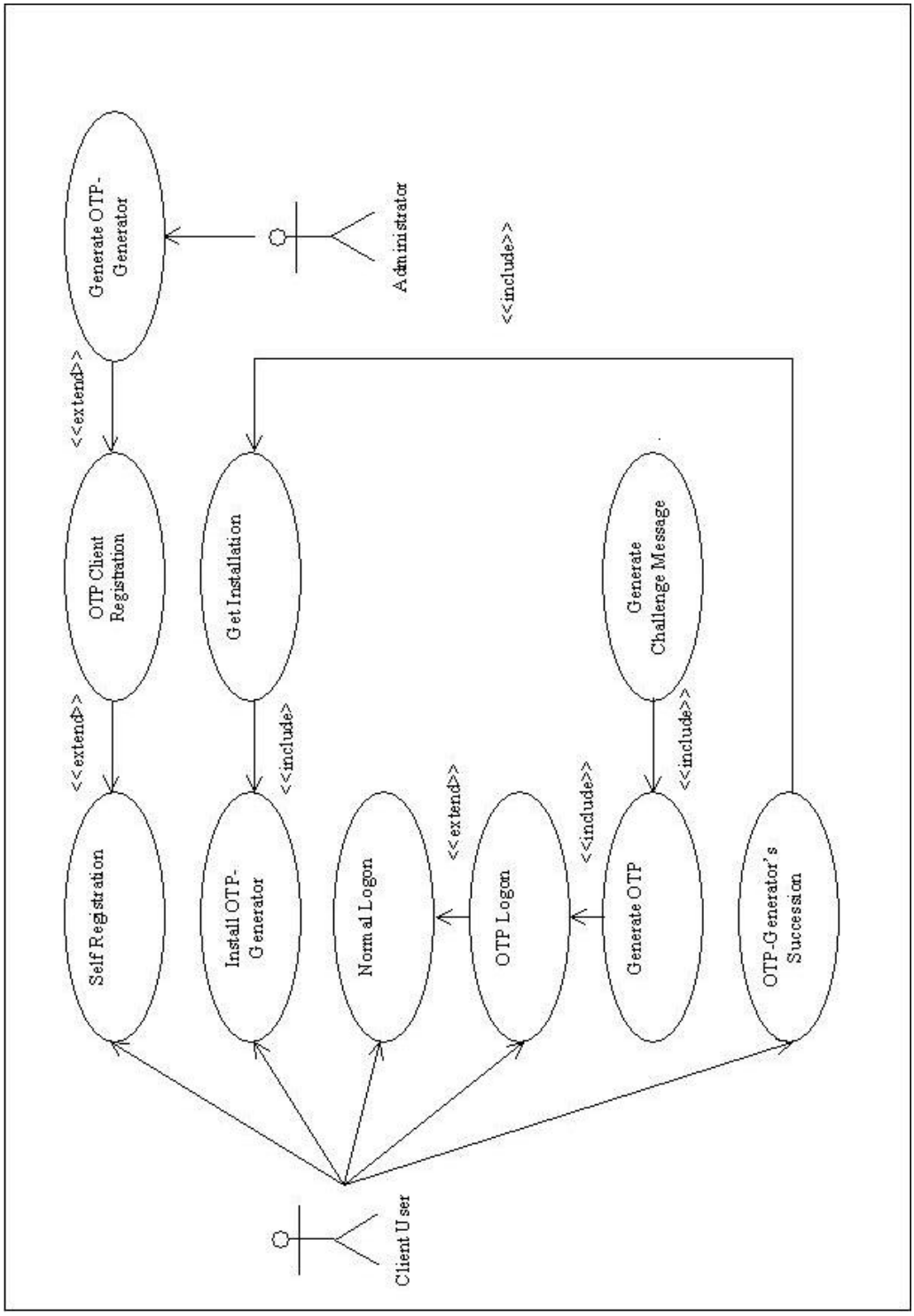


Figure 4-1: Use Cases diagram (problem domain)

- **Case: Self Registration**

  This is one of cooperative process for member registration. Main purpose of this process is gathering for user's information into system database via ProfileForm, the component can be used for entering information and validate the format of entered values. User information was already included login id and static password.

- **Case: OTP Client Registration**

  This case is represented for additional process of member who would like to get preferential accessibility. Essence of this process is how to submit notification to the system and predicament of active normal member, applicant who already got normal level accessibility, would be changed to prospective OTP user. After that, they must wait for privilege accessibility to special domain by using OTP procedure. The important thing which member had to submit to the system is User PIN.

- **Case: Generate OTP-Generator**

  Web master or web administrator of this web service would be roused by record of prospective OTP user. Incoming record was shown in admin web pages, which has accessibility for administrator only. Main obligation of administrator is initiation the program source code, which relate to information of prospective OTP user. Then compile that source code to executed application. Each prospect OTP user record can be gotten 3 related installation, which be stored in appropriated server directory then email will be sent to applicant with content describe how to download and install OTP-generator application in their client machine.

- **Case: Install OTP-Generator and Case: Get Installation**

  Install OTP-Generator is process, which include Get Installation as sub process. After receiving correspondence from system administration, client members must perform their business by using static password to access to personal web page and get one of token installations, which related to personal profile. Then OTP

profile would be updated by setting serial number of downloaded token application and OTP status would be automatically updated to active. Software installation would be distributed to member's client machine and Install OTP-Generator was subsequently conducted. Then stored token application can be triggered whenever owner, who is active OTP client user, would like to conduct OTP Logon process.

- **Case: Normal Logon**

  This cooperative process has same details as authentication of general web based application. Normal level of transaction in this web-based service require normal level of authentication, validation of entered user login id and static password by compare with user profile information (registered profile record in system database). The objective of Normal Logon is not only authentication to allow user access to normal level of transaction, but also there are the purposes like the first checkpoint to identify incoming visitor before access to special transaction.

- **Case: OTP Logon**

  This is essential process of this authentication approach, which require concept of encryption and decryption algorithm. Challenge-Response system is main concept of this process while generating encrypted message by OTP generator application is coming from token device concept. Whenever visitors can get authorization from normal logon process, they can conduct their business not only with normal transaction but also they can join the special transaction by starting with OTP Logon process. For generating encrypted message (OTP), this process requires 2 sub processes of getting challenge message from sever and including server message in encryption algorithm then all of necessary individual information, that were already collected in token application, would be converted to encryption. Generated OTP will be sent to system via OTPLogingOnTicket, the component can be used for entering encrypted message with login id. For verification, OTP would be decoded and separated to each part of information then plain texts would be compared with data that have been collected in system relational database.

- **Case: Generate OTP**

  This process was included to apart of OTP Logon process. After getting challenge message that was included timestamp from server, client user had to activate the token application by key in user PIN to identify the right owner. Then application form would request for valid message and include it to encryption algorithm. With server message and secret personal information that were preserved inside program source code, encryption process was performed to return the expected result – one time password.

- **Case: Generate Challenge Message**

  This process was included to apart of OTP Logon process. To conduct authentication for special transaction, client user, who already had normal level accessibility, had to go to specific web page to get challenge message from server. There are two parts of message, first is current timestamp and another is randomized server string. Timestamp would be displayed by appropriated format of current time when user request for challenge message. (Timestamp is one of most important factor for encryption algorithm of this study) Randomized server string is the one information that will be converted to cipher text and sent to authentication system. Server string can be used to identify incoming message that was transformed by server information or not. Additional, last challenge message would be collected to system database relate to user profile and it can be used to limit the timing of authentication process since message was generating until OTP was verified.

- **Case: OTP-Generator's Succession**

  Weakness of software-based generator is installation was required to conduct on local machine. So there is conflict with the purpose of web-based application for using anytime, anywhere. While there is comfortable for device-based generator, which ability to bring its keyfob shape to anywhere. To remove this weakness Succession of OTP-Generator cannot be ignored. Whenever OTP members would like to access to special transaction by another computer, they had to download another version of token application in their quota by access with static password

to the same page that was used in Get Installation process. Whenever token next version was downloaded, current active token serial number in OTP user profile will be replaced by new serial number. In this case current used token application, which was installed in used machine, cannot be used anymore. Active token application will be install subsequently in new machine, so just only active machine, whose the active token, can be used to access to special zone of this web-based service.

4.2 Class diagram: represent details of objects and their relationship

This section describe about representation of logical concept of the components for this alternative approach. Main purpose of class diagrams in this section is explanation for details of attributes and method of each component.

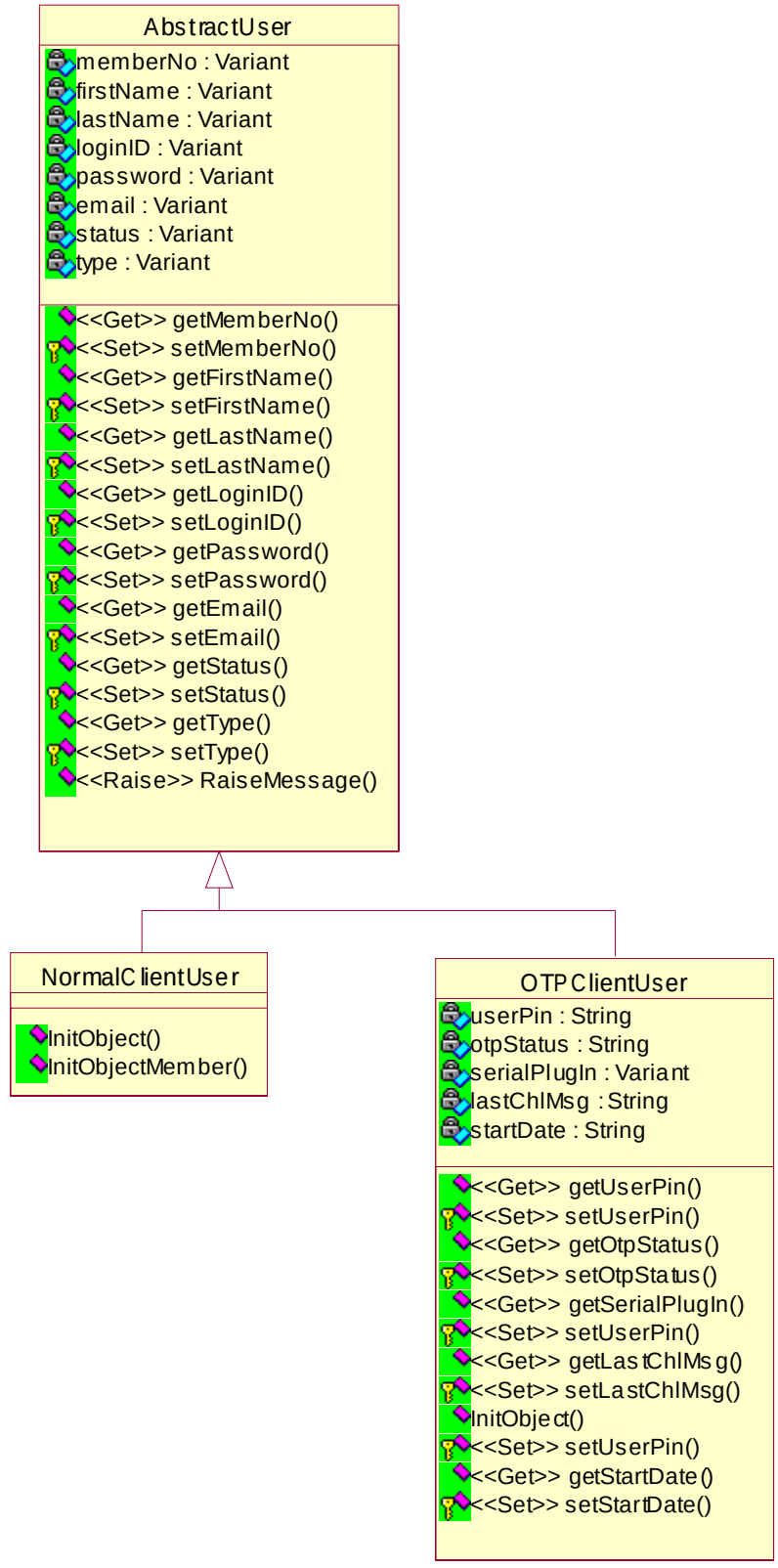


Figure 4-2: Class diagram (user's profile)

Class diagram (user's profile): description

AbstractUser has attributes and methods that represent user profile information and this class is parent class of two main classes that represent each user object in various activities in the system.

NormalClientUser class inherits from AbstractUser. This class has no specific attribute but there are specific methods.

OTPClientUser class also inherits from AbstractUser. And there are some specific attributes and some methods to collect additional profile for identify OTP qualification.

Attribute description

Class **AbstractUser**

- memberNo: automatically generate by system (unique)
- firstName: member's information
- lastName: member's information
- loginID: account id, initialized by visitor who provided registration process
- password: static password, initialized by visitor who provided registration process
- email: member's information
- status: member's status
- type: default value is "customer" for all client users, "webadmin" for web administrator

Class **OTPClientUser**

- userPIN: user PIN, initialized by normal member who provided OTP registration process
- otpStatus: specific status for user who registered for OTP authorization.
- serialPlugin: serial number of token application, which belong to OTP user with active status
- lastChlMsg: the challenge message, user accepted while the OTP logon process was conducting.
- startDate: Date-Time show that when user register to the system for OTP authorization.

Class **NormalClientUser**

NormalClientUser inherited all attributes from its parent class, AbstractUser. This class has just specific methods but has no specific attributes

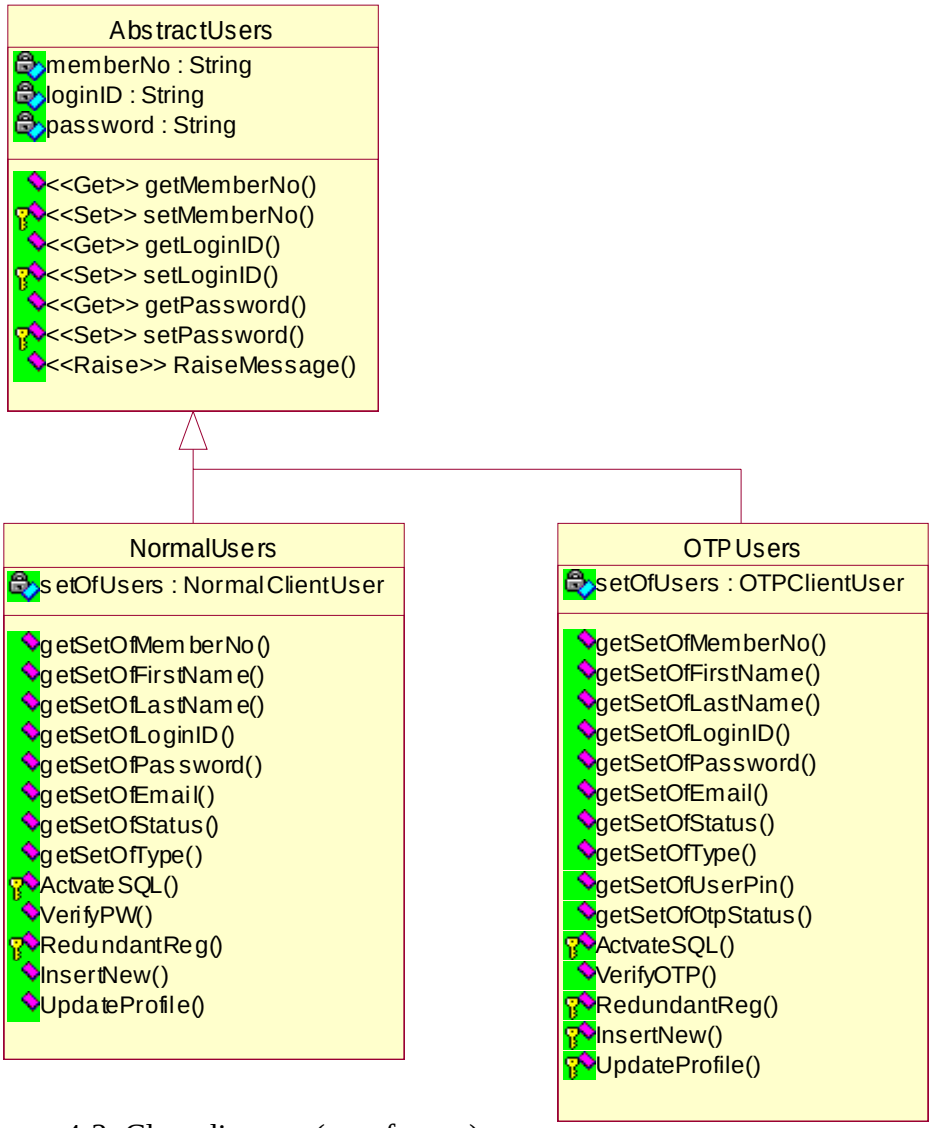


Figure 4-3: Class diagram (set of users)

Class diagram (set of users): description

AbstractUsers has a few attributes and methods. Objective of this class is pattern of both classes that were used for the purpose of verification by login id and password (both static password and One-time password)

NormalUsers class inherits from AbstractUsers. This class not only has purpose for authenticate incoming object by verify static password, but also has the purpose for represent set of normal level member objects which can be updated or added new one.

OTPUsers class also inherits from AbstractUsers. This class not only has purpose for authenticate incoming object by verify OTP, but also has the purpose for represent set of OTP member objects which can be updated or added new one.

Note: set of member objects usually was used in the process of administration, which often maintain users' profile.

Attribute description

Class **AbstractUsers**

- memberNo: collect incoming member number of user who request for authentication
- loginID: collect incoming account id of user who request for authentication
- password: collect incoming OTP message of user who request for authentication

Class **NormalUsers**

- setOfUsers: collect instances of NormalClientUser, using for list the records of normal level users

Class **OTPUsers**

- setOfUsers: collect instances of OTPClientUser, using for list the records of special level users

Figure 4-4: Class diagram (OTPusers and decryption)

Class diagram (OTPusers and decryption): description

Aggregation of OTPClientUser and OTPUsers: OTPUsers, inheritance from AbstractUsers, can be used for represent set of OTP user objects, instance of OTPClientUser, so relation of both classes can be explained by above diagram. In case of OTP registration scenario, object of incoming user would be initialized by OTPClientUser class and was sent to insert to set of OTP registered user, which base on OTPUsers class. For scenario of updating OTP user profile by administrator, each profile would be initialized by OTPClientUser class and was included as set of users in instance of OTPUsers. Then administrator would find the appropriated record by method in OTPUsers and update that profile by appropriated method.

Relationship of OTPUsers and DecryptOTP: For OTP verification, login id and OTP message would be sent to OTPUsers then by its method cipher message was sent to instance of DecryptOTP class. Because of methods of DecryptOTP, decoding process was conducted and the result of plain text, user information, would be returned to OTPUsers again then information would be compared with the set of OTP users.

Attribute description

Class **DecryptOTP**

- userPin: get the result of decoding, collect cracked user PIN from cipher message
- challengeMsg: get the result of decoding, collect cracked challenge message from cipher message
- serialNumber: get the result of decoding, collect cracked token’s serial number from cipher message
- sDateDecode: get the result of decoding, collect cracked date-time header from cipher message
- sFWHeader: collect a part of cipher message, encrypt of date-time
- iBWKey: get the result of decoding, collect backward key, which was calculated from date-time header
- cypherText: collect cipher message, One-Time Password

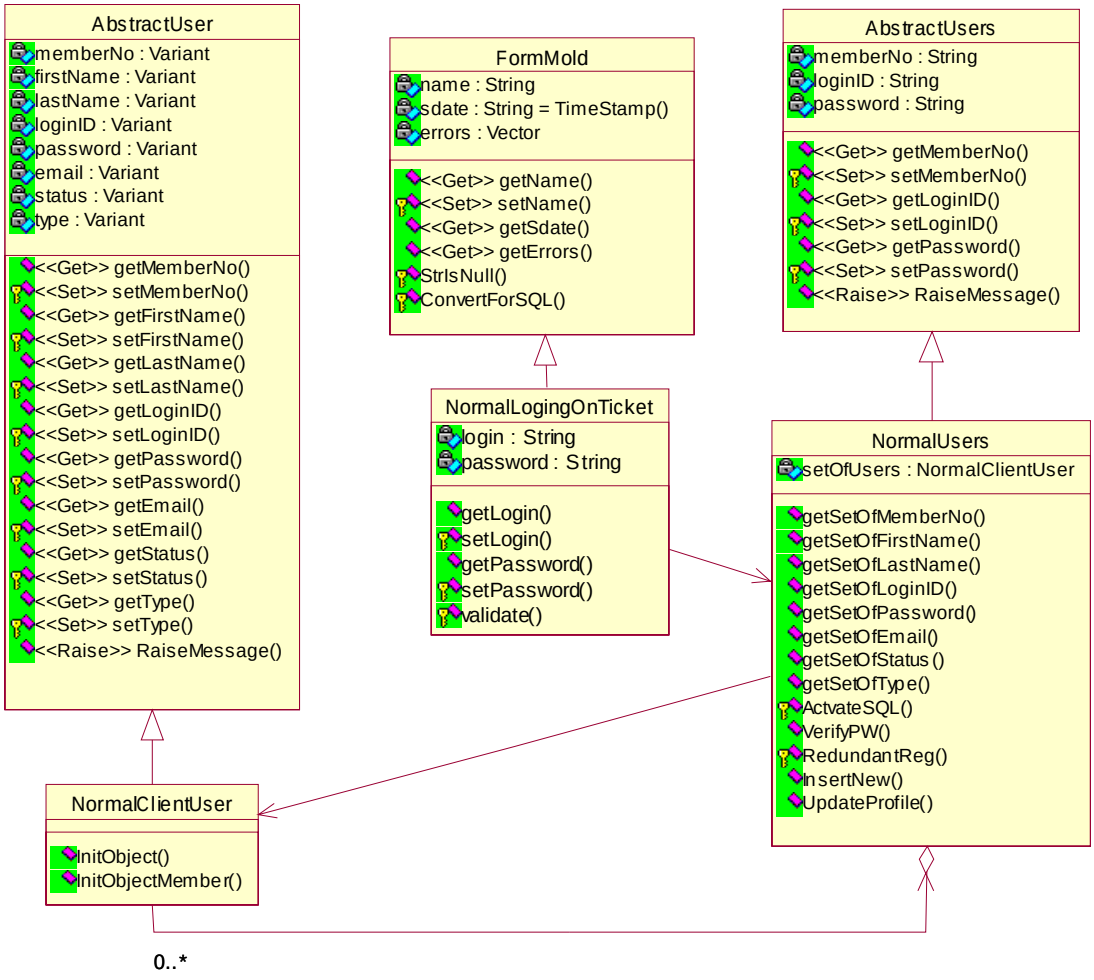


Figure 4-5: Class diagram (form for normal logon)

Class diagram (form for normal logon): description

FormMold class has attributes and methods that represent qualification of form component. All of forms were used in this approach base on this class.

NormalLogingOnTicket class inherits from FormMold. This class has specific attribute to collect login id and static password. Also there is specific method to validate format of content before submitted them to the system.

This diagram can explain the scenario of normal level logon process. Login id and password would be entered into attributes of NormalLogingOnTicket. After all of entered value were validated the format by method of form instance, form was submitted and its attributes would be passed to attributes of instance of

NormalUsers. After that verification method of NormalUsers would be triggered then values of its attributes, login id and password, were compared with information of set of users. NormalClientUser initiated each record in the set of users. Finally, instance of NormalClientUser for authenticated user profile would be set as the session object in this normal level service.

Attribute description

Class **FormMold**

- name: Name of form
- sDate: collect timestamp when instance of from was used.
- error: collect the series of error message

Class **NormalLogingOnTicket**

- login: collect entered account id
- password: collect entered constant password

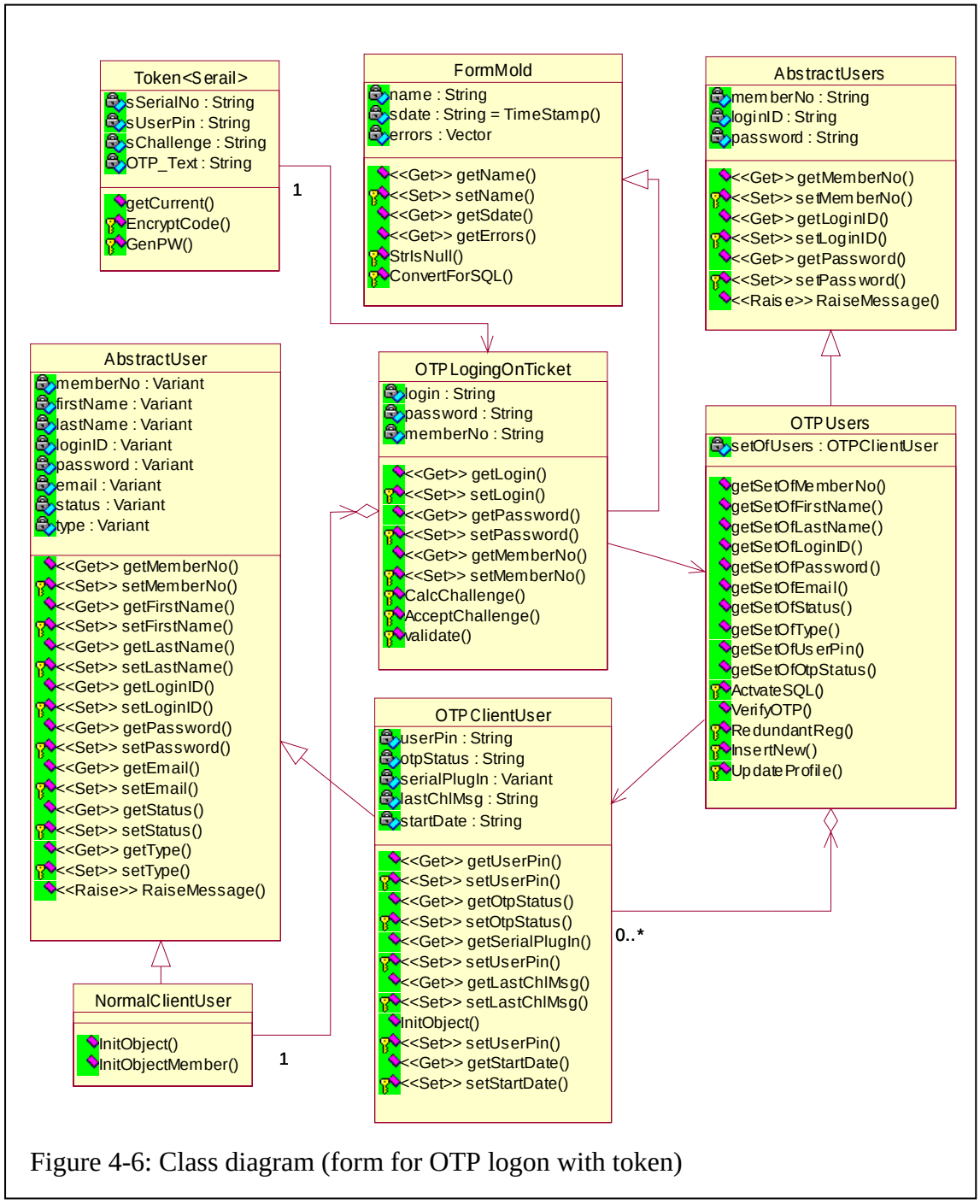


Figure 4-6: Class diagram (form for OTP logon with token)

Class diagram (form for OTP logon with token): description

OTPLogingOnTicket class inherits from FormMold. This class has specific attribute to collect login id and static password. Also there is specific method to validate format of content before submitted them to the system.

This diagram can explain the scenario of special logon process. Login id and OTP password would be entered into attributes of OTPLogingOnTicket. Due to member must have authentication for normal level before access to this form so login id attribute could be pass from previous object of member profile – NormalClientUser. However after all of entered value were validated the format by method of form instance, form was submitted and its attributes would be passed to attributes of instance of OTPUsers. After that verification method of OTPUsers would be triggered then values of its attributes, login id and all apart plain text of OTP password, were compared with information of set of users. OTPClientUser initiated each record in the set of users. Finally, instance of OTPClientUser for authenticated user profile would be set as the session object in this special transaction.

Additional, before above scenario instance of token<serial> must return OTP password by its appropriated method. More explanation for this specific process can be followed by next diagram. (Class diagram 06)

Attribute description

Class **Token<serial>**

- sSerialNo: collect serial number of token application, which was included in source code
- sUserPin: collect user PIN of OTP user, which was included in source code
- sChallenge: collect challenge message, which user entered while the OTP logon process was conducting.
- OTP_Text: get the result of encryption.

Class **NormalLogingOnTicket**

- login: collect entered account id
- password: collect entered OTP message

- memberNo: collect entered member number of normal level user who would like to get authentication by OTP logon process.

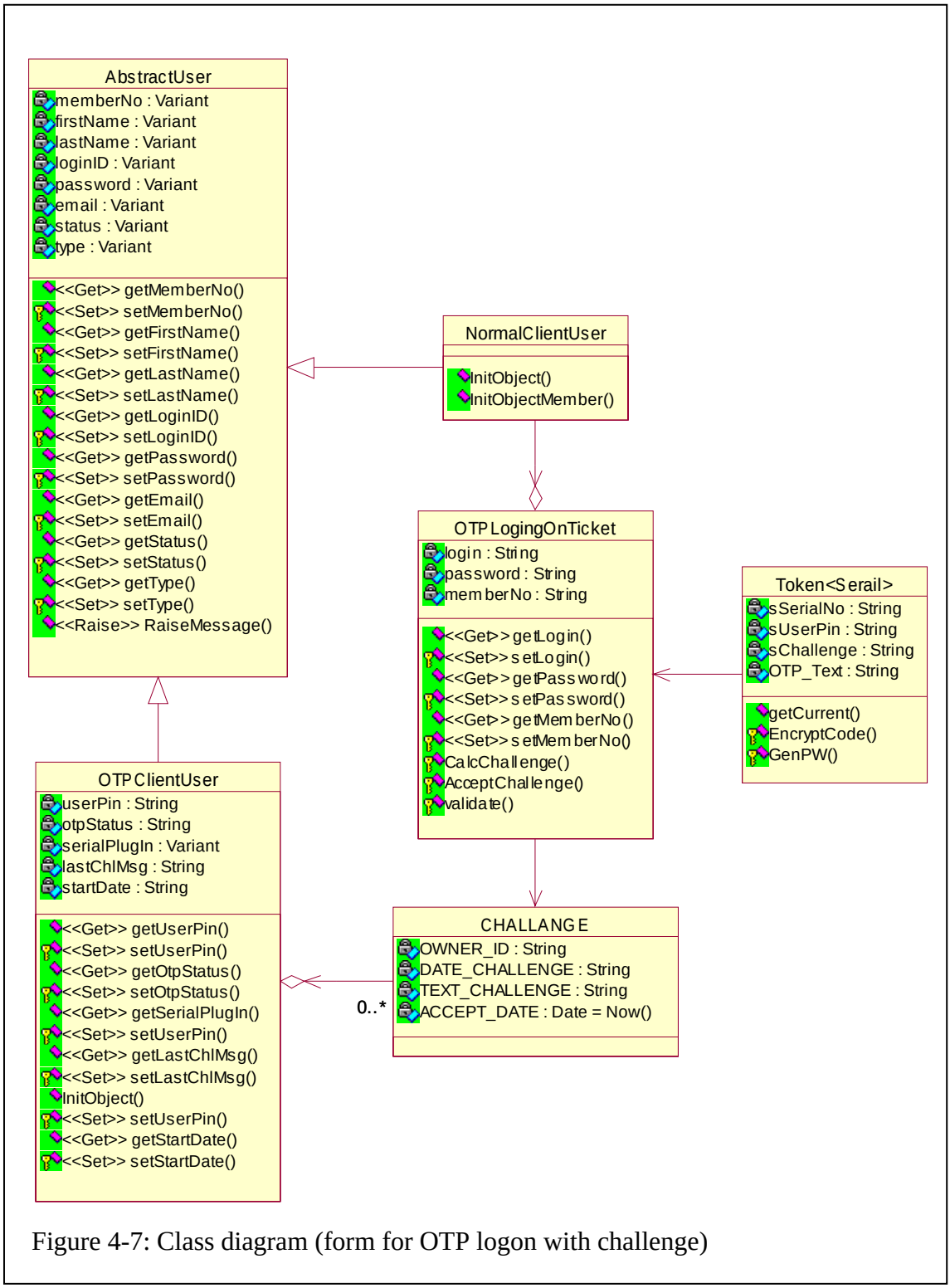


Figure 4-7: Class diagram (form for OTP logon with challenge)

Class diagram (form for OTP logon with challenge): description

OTPLogingOnTicket class has method to generate challenge message-instance of CHALLENGE class, which has the forced format (current timestamp and randomized server message) and this message can be collected as apart of user profile. (The last challenge message of user profile) Also this message was shown on the monitor so user can enter this message into token application-instance of Token<serial>. With preferred challenge message from server and its attributes-inside the program source code, token application can return the result as cipher text of OTP password by its specific methods. Subsequently the cipher message would be put into instance of OTPLogingOnTicket to provide the OTP Logon process.

Attribute description

Class **CHALLANGE**

- OWNER_ID: member number of user who got challenge message
- DATE_CHALLENGE: a part of challenge message, which collect the timestamp when challenge message was generated
- TEXT_CHALLENGE: a part of challenge message, which collect the randomized string
- ACCEPT_DATE: date-time when user got challenge message, will be used for verification of time limitation in OTP logon process

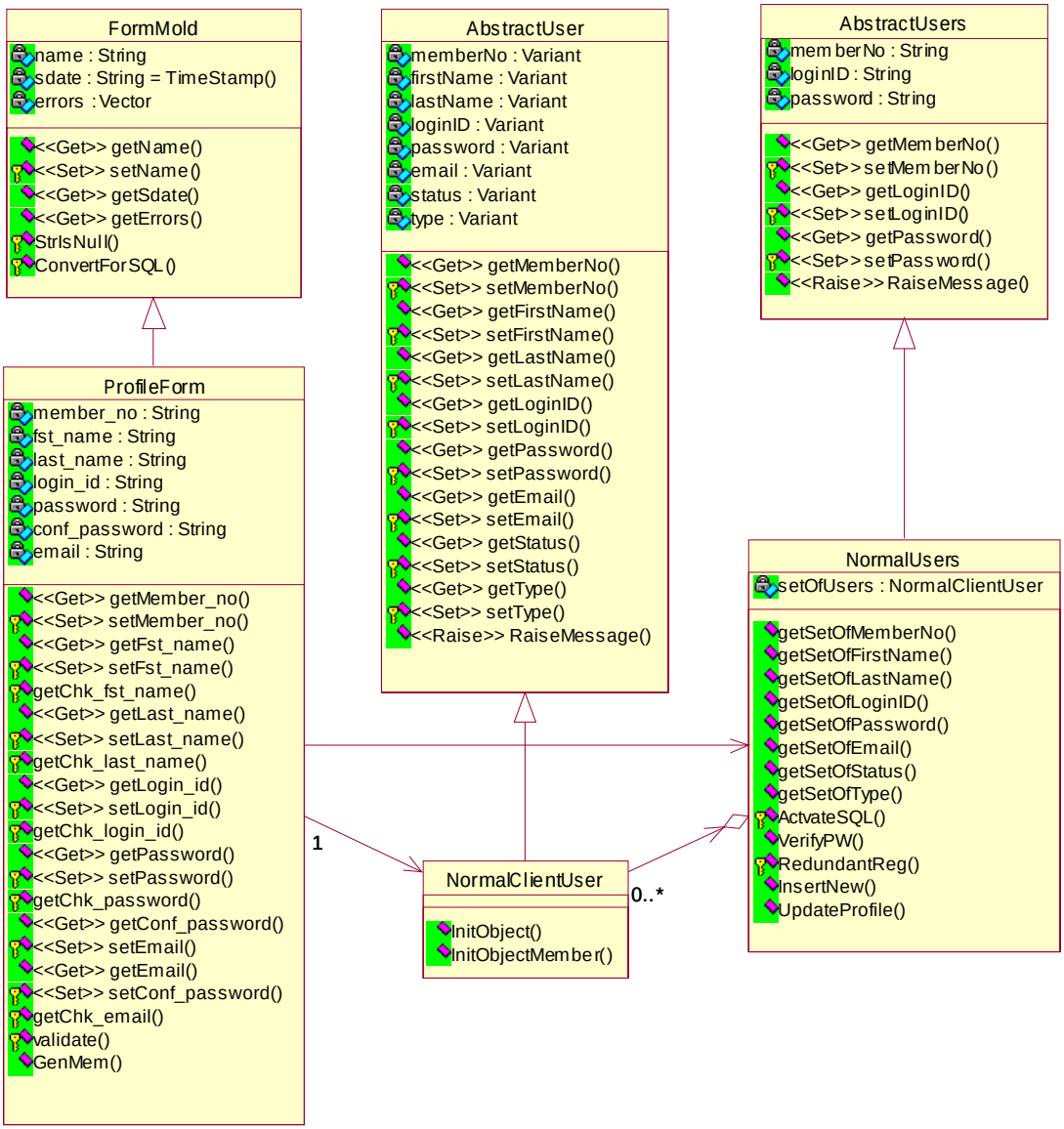


Figure 4-8: Class diagram (form for insert and update user's profile)

Class diagram (form for insert and update user's profile): description

ProfileForm class is inherited from FormMold. This is one of from component in this study. It was used for activities, which invoked user profile. Both business scenarios of visitor (for self-registration) and administrator (for updating user profile) have to trigger this component. Its attributes can be represented for user information. Besides it has specific method can be used for validate format of entered value. Whenever this form was submitted, this from can trigger on method

of NormalUsers for not only update on matched record, but also new record can be inserted to the set of users. And instance of NormalClientUser can be initiated temporary for method of verification in case of new registration.

Attribute description

Class **ProfileForm**

- member_no: display text for member number of user's profile
- fst_name: display text for first name of user's profile
- last_name: display text for last name of user's profile
- login_id: display text for account id of user's profile
- password: display text for constant password of user's profile
- conf_password: display text for reentered password
- email: display text for email address of user's profile

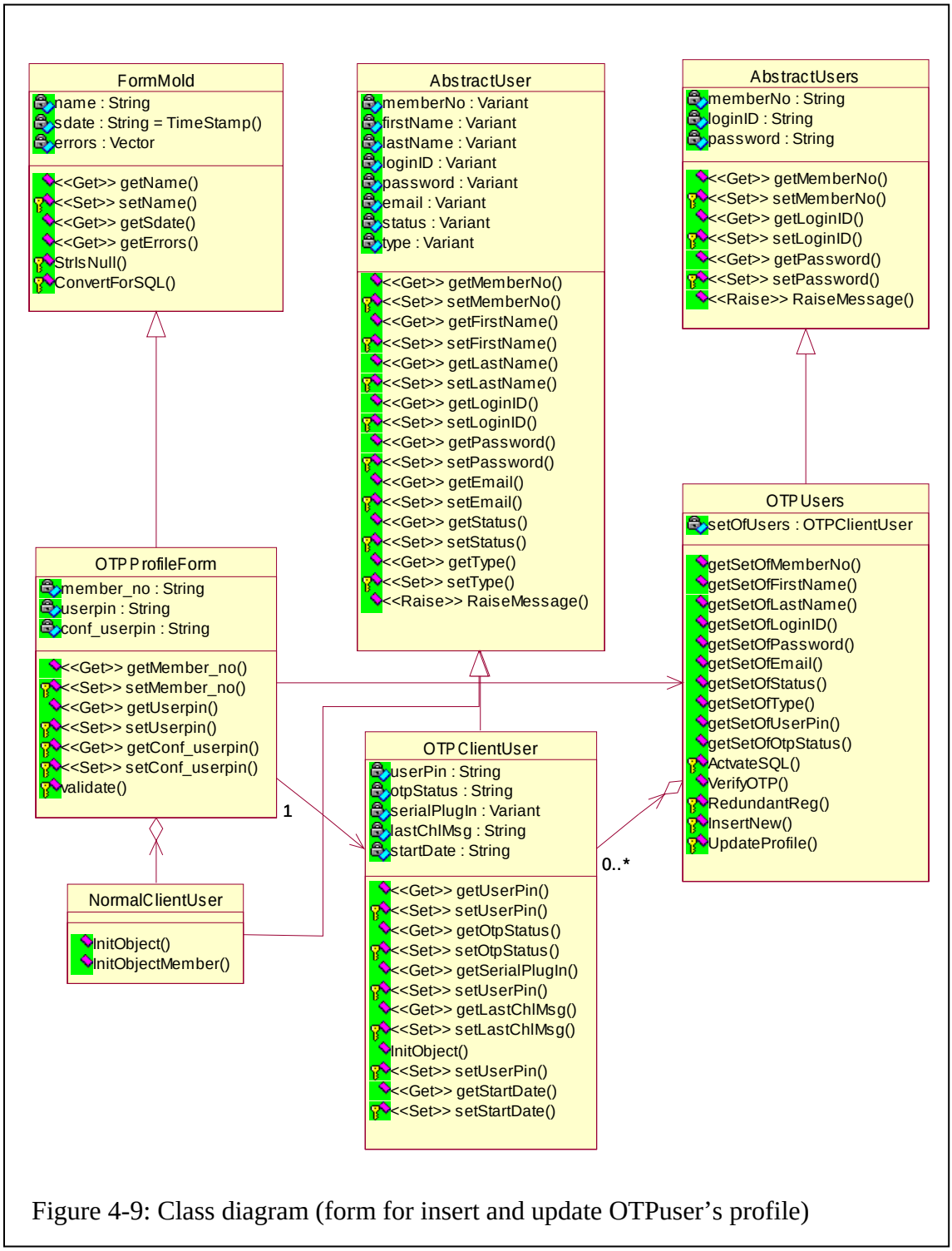


Figure 4-9: Class diagram (form for insert and update OTPuser's profile)

Class diagram (form for insert and update OTPuser's profile): description OTPProfileForm class is inherited from FormMold. It was used for activities, which invoked additional user profile. Both business scenarios of visitor (for OTP

registration) and administrator (for updating OTP user profile) have to trigger this component. Its attributes can be represented for additional of user information. This kind of form was included information from session instance of NomalClientUser, which already had normal level authentication. Besides it has specific method can be used for validate format of entered value. Whenever this form was submitted, this from can trigger on method of OTPUsers for not only update on matched record, but also new record can be inserted to the set of OTP users. And instance of OTPClientUser can be initiated temporary for method of verification in case of new registration.

Attribute description

Class **OTPProfileForm**

- member_no: display text for member number of user's profile
- userpin: display text for user PIN of user's profile
- conf_userpin: display text for reentered user PIN

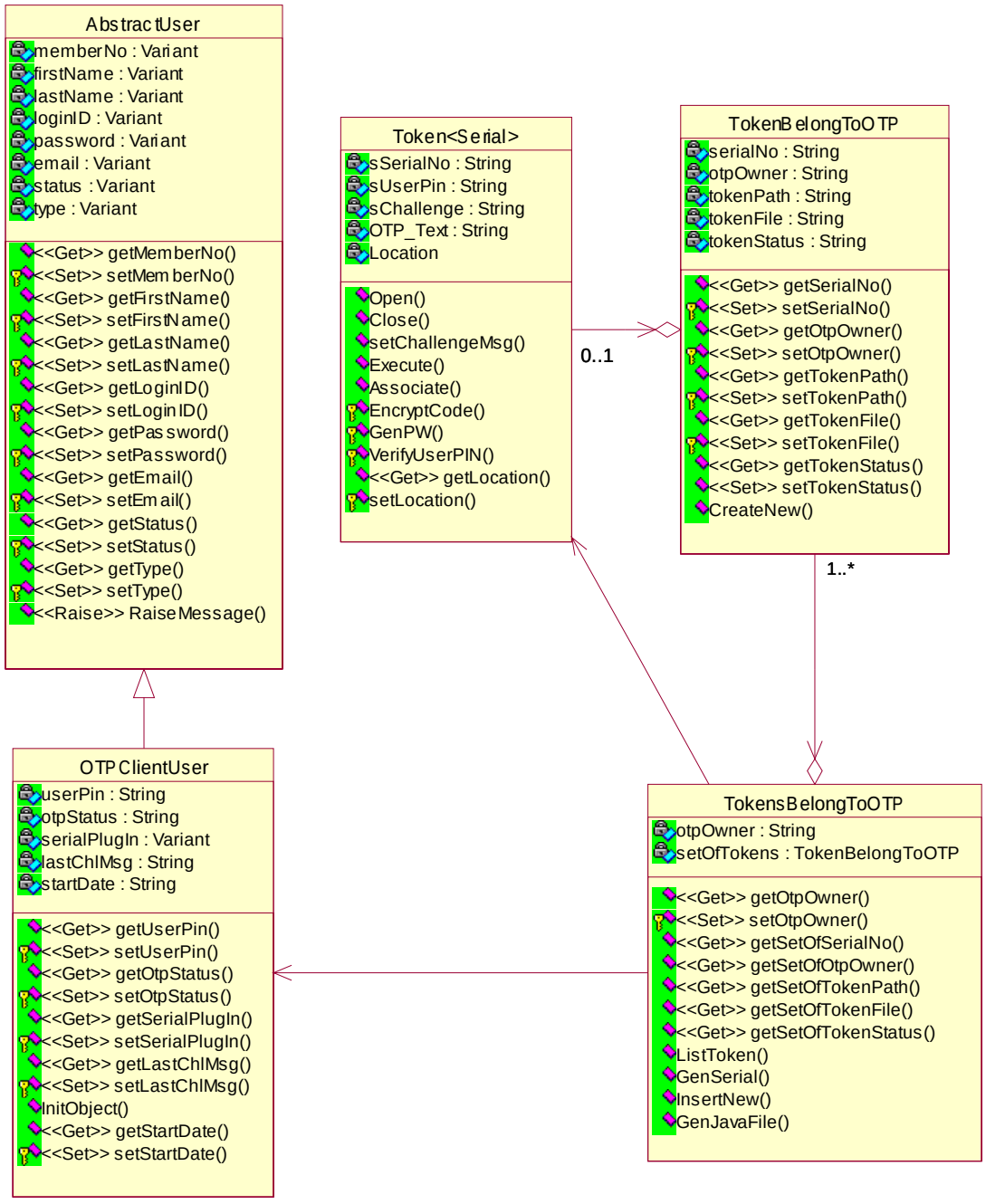


Figure 4-10: Class diagram (token components belong to OTPuser)

Class diagram (token components belong to OTPuser): description

This class diagram can be represented not only the relationship of Token quota records and its owner-OTP user, but also it can show how to relate the physical files, installations of token application were prepared by administrator, with each token record.

TokenBelongToOTP is the class that has necessary attributes for token records. The token record is the information in database, which collects the profile of physical file of token installation. Unique serialNo is the identification number for each token application, which was collected in personal directory on server. Attribute name tokenPath and tokenFile can be referred to that OTP user's personal directory and installation files inside. Just only one token record under ownership of OTP user could have status 'Active'.

TokensBelongToOTP was used for represent the set of token records. This class has attributes for represent all of instance of TokenBelongToOTP that belong to specific owner-OTP Owner. So administrator can use instance of this class in case of token quota management and owner can access to specific web page to use instance of this class in case of download installation or token succession process.

Token<Serial> is the class represent for physical installation file. Actually instance of this class has the different concept for origination. For the approach of this study, this strange instance cannot be initiated by programming. But it was generated semiautomatic whenever web administrator can find the request of incoming OTP user for new installation then administrator has to trigger the command to generate program source code, which was included individual information of OTP user profile. After that, process for execute and building installation was taken over by manually operation. Finally installation file would be moved to OTP user's personal directory on the server and the attributes of token record would be set by refer to that directory.

Attribute description

Class **TokenBelongToOTP**

- serailNo: serial number of token profile
- otpOwner: member number of token's owner
- tokenPath: path of directory on the server where store the token installer
- tokenFile: token installer's file name
- tokenStatus: status of token profile

Class **TokensBelongToOTP**

- otpOwner: member number of tokens' owner

- setOfTokens: collect instances of TokenBelongToOTP, using for list the records of token profile that were associated to OTP user profile

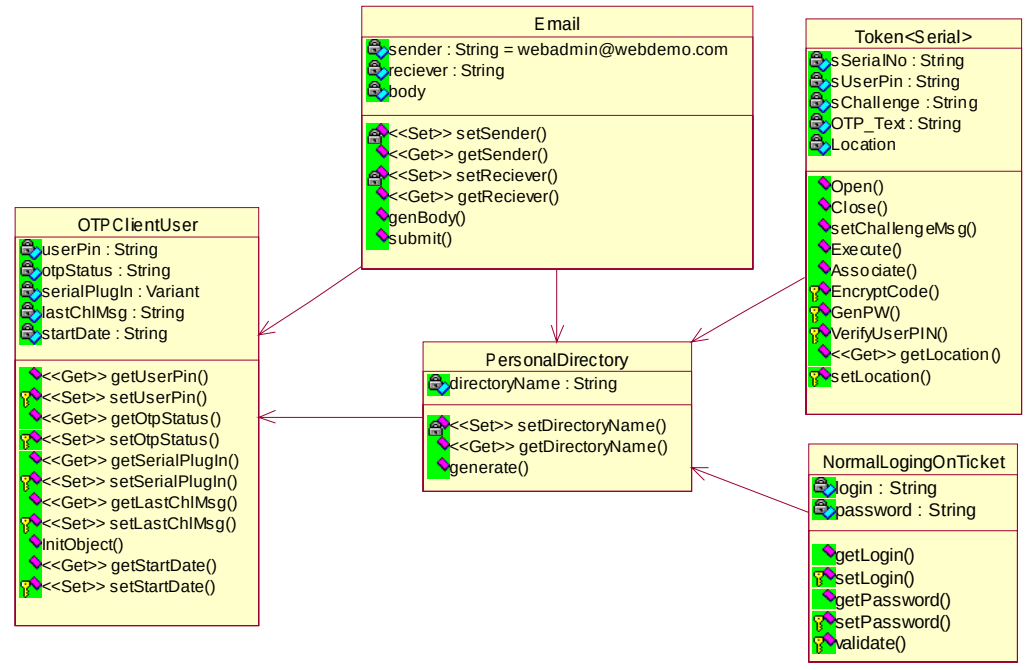


Figure 4-11: Class diagram (Email, Personal directory and token installation)

Class diagram (Email, Personal directory and token installation): description

This Class diagram describe for the relationship of OTP user profile and its assets on the system. Not only that, it can describe for the specific channel-email to inform OTP client users to update their status.

PersonalDirectory is the class that represents the generated directory on the server disk space. All of OTP user must has personal directory and would be informed via email form web administration about how to access to their directory. In this study, personal directory collected installations and personal login page. Personal directory would be automatic generated by specific command, which was triggered by administrator.

Instance of Token<Serial> is the physical file of application installer. This physical file was generated and moved to personal directory by administration process.

One instance of NormalLogingOnTicket also was generated automatically in this directory by administration side. Specific form component would be used for

individual accessing of OTP user who received email, which contain information about specific URL of this form.

Email class was used for represent automatic generated email. Administrator would also trigger this automatic generating. Receiver attribute would be set by user’s email address, one attribute of user profile, while content was generating by specific command in administration side. Also administrator would submit this email to destination address.

4.3 Sequence diagrams: represent combination of objects and process scenario

For elucidation of this approach, sequence diagram was chosen to explain all of business processes for this alternative authentication. Essence of this section is how to know the details of processes that include the roles of objects. There are 7 main scenarios (refer to cases of Use Case diagram) were represented by following 7 diagrams.

- **Self-Registration**: Registration for normal level member.
- **Normal Logon**: access to system with normal level of permission.
- **OTP Client Registration**: Registration for get special permission.
- **Generate OTP Generator**: Generate token application for OTP member.
- **Install OTP-Generator**: How to download and install token application
- **Generate password by OTP Generator**: How to generate OPT by token application.
- **OTP Logon**: special accessing to system for special transaction.

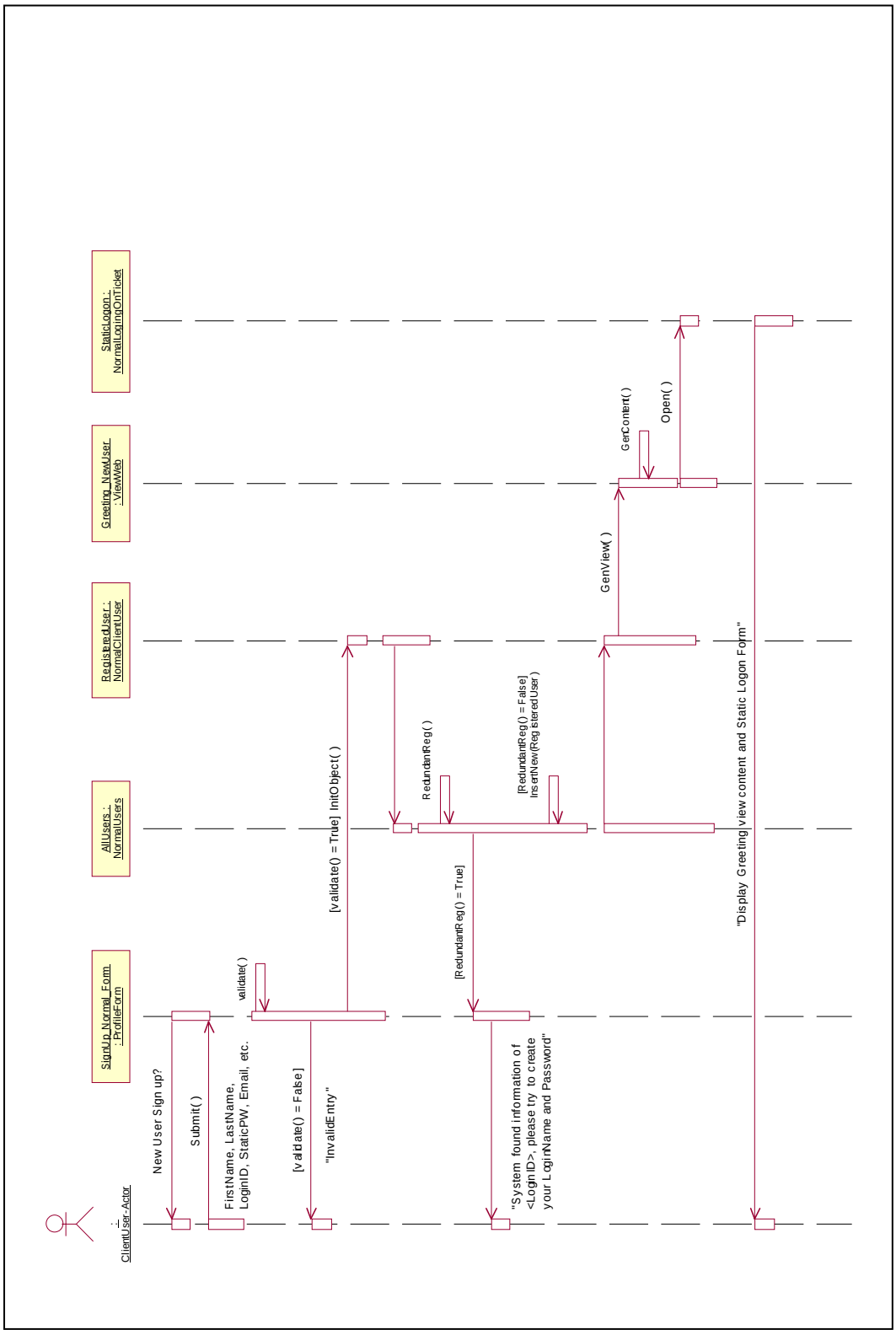


Figure 4-12: Sequence diagram (self-registration)

Sequence diagram (self-registration): description

Self-Registration start by client user visited web site on Signup_Normal_Form. User had to key in his/her information in this form and entered values were validated format by method of this signup form. Then user submited this form, which was included valid format of user information. Information would be collected in format of NormalClientUser. (RegisteredUser object) and its login id and password were verified in set of users (AllUsers) to finding redundant record. If has no redundant record in the system, new user object would be collected as new record in database by insert method of set of users while Greeting_NewUser content was generating and showing to incoming user in web page that included NormalLoginOnTicket (StaticLogon object). When finished this process, user can used his/her login id and static password to request normal level authentication.

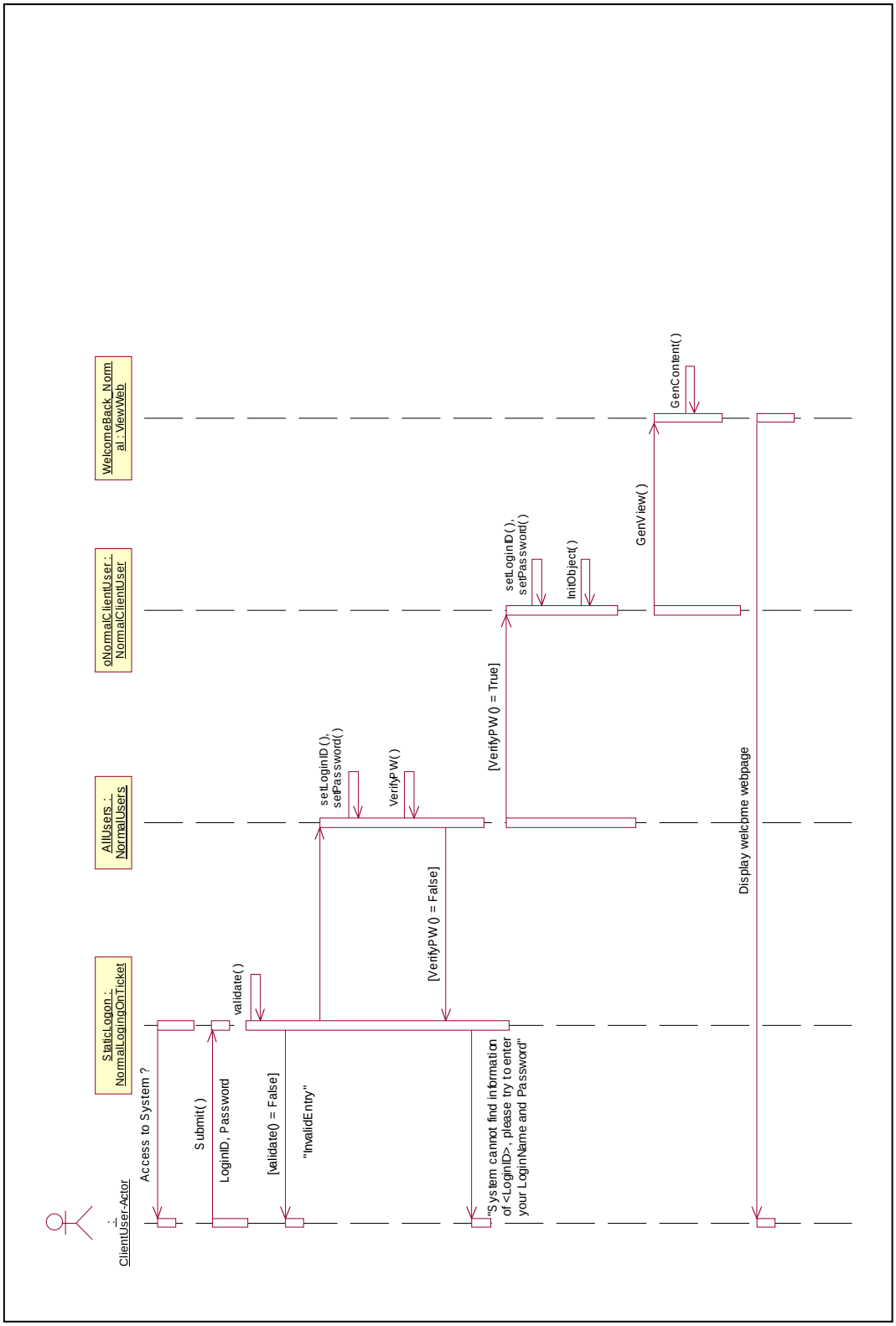


Figure 4-13: Sequence diagram (normal logon)

Sequence diagram (normal logon): description

Normal Logon start by client user visited web site on StaticLogon form. User had to submit his/her login id and password that was validated for appropriated format by form's method. Both of information was passed to AllUsers and verification method would be triggered to search the right user record in the system database. In case of verification return false, client user would be navigated to the StaticLogon form again and system would display suggest message to user for trying to login again. On another hand, whenever password verification return true user record would be initiated as oNormalClientUser (object instance of NormalClientUser) and this object had the session property mean that profile of this object can be used in every pages of normal transaction in this web service. Finally normal client member was navigated by system to WelcomeBack_Normal, web page was included welcome sentence for member.

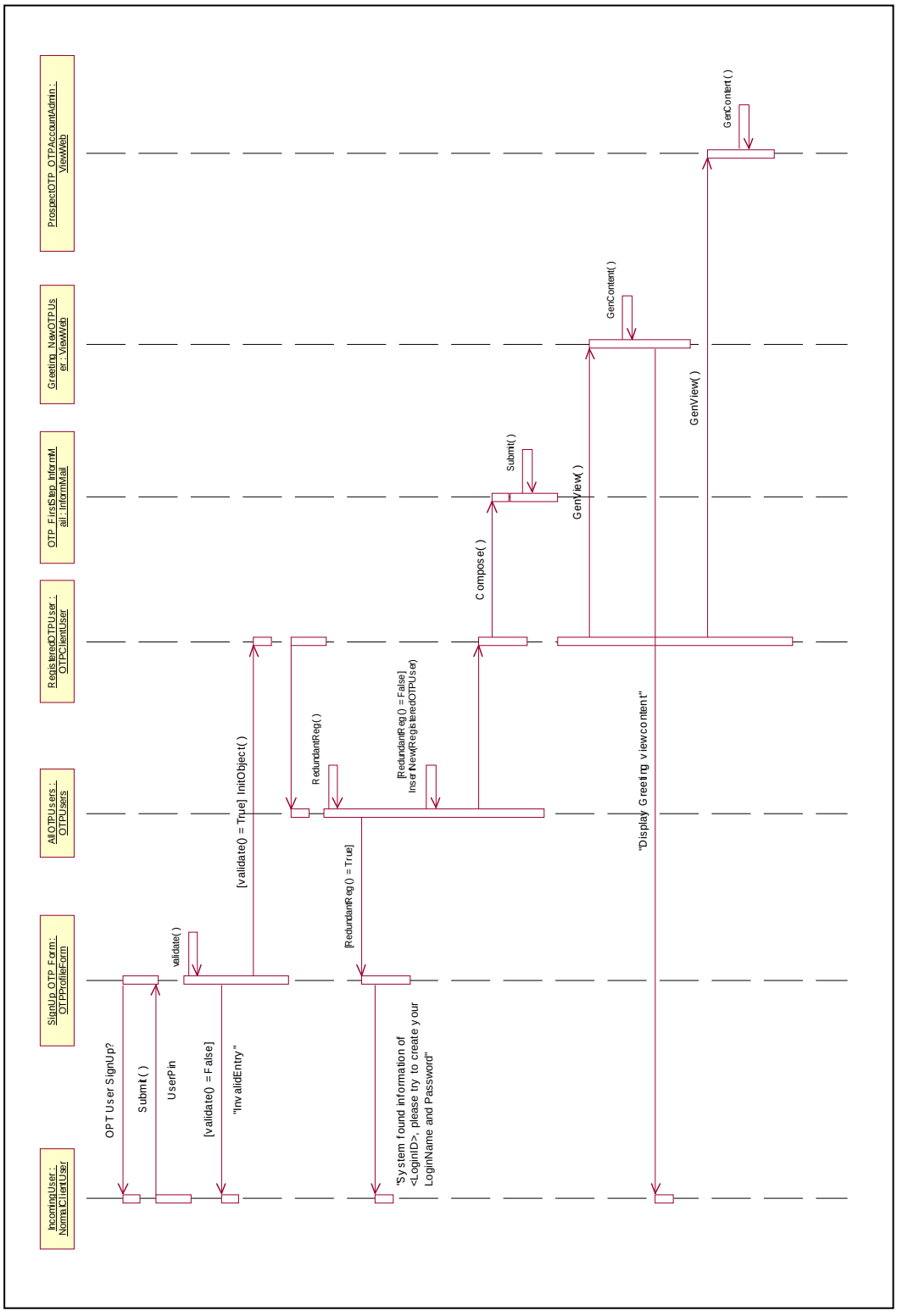


Figure: 4-14: Sequence diagram (OTP client registration)

Sequence diagram (OTP client registration): description

OTP Client Registration would be started whenever normal user, who already had normal level authentication but has no permission for special transaction, would like to conduct on high-level service.

At first normal client user (IncomingUser object) has to open Signup_OTP_Form and entered necessary information to this form. In this case profile of user was already initialized so this form required just additional profile to completed the OTP qualification- UserPIN is mandatory information for OTP user profile. UserPIN was considered and entered into form by user. After validated format of values by form's method, all of profile would be collected in format of RegisteredOTPUser. (Instance object of OTPClentUser) Then this instance was passed to the set of OTP users (AllOTPUsers object) to finding redundant record. If has no redundant record in the system, new OTP user object would be collected as new record with prospective status in database by insert method of the set of OTP users. While user was navigated to Greeting_NewOTPUser, view with greeting sentence for user. And OTP_FirstStep_InformMail (instance of Email) would be composed by get receiver email address from user profile and sent to inform client user that system already get his/her request. Client has to wait for next response from system administration. Not only that, new record of OTP user with status prospective would be shown in administration web page so administrator can know about next step of process.

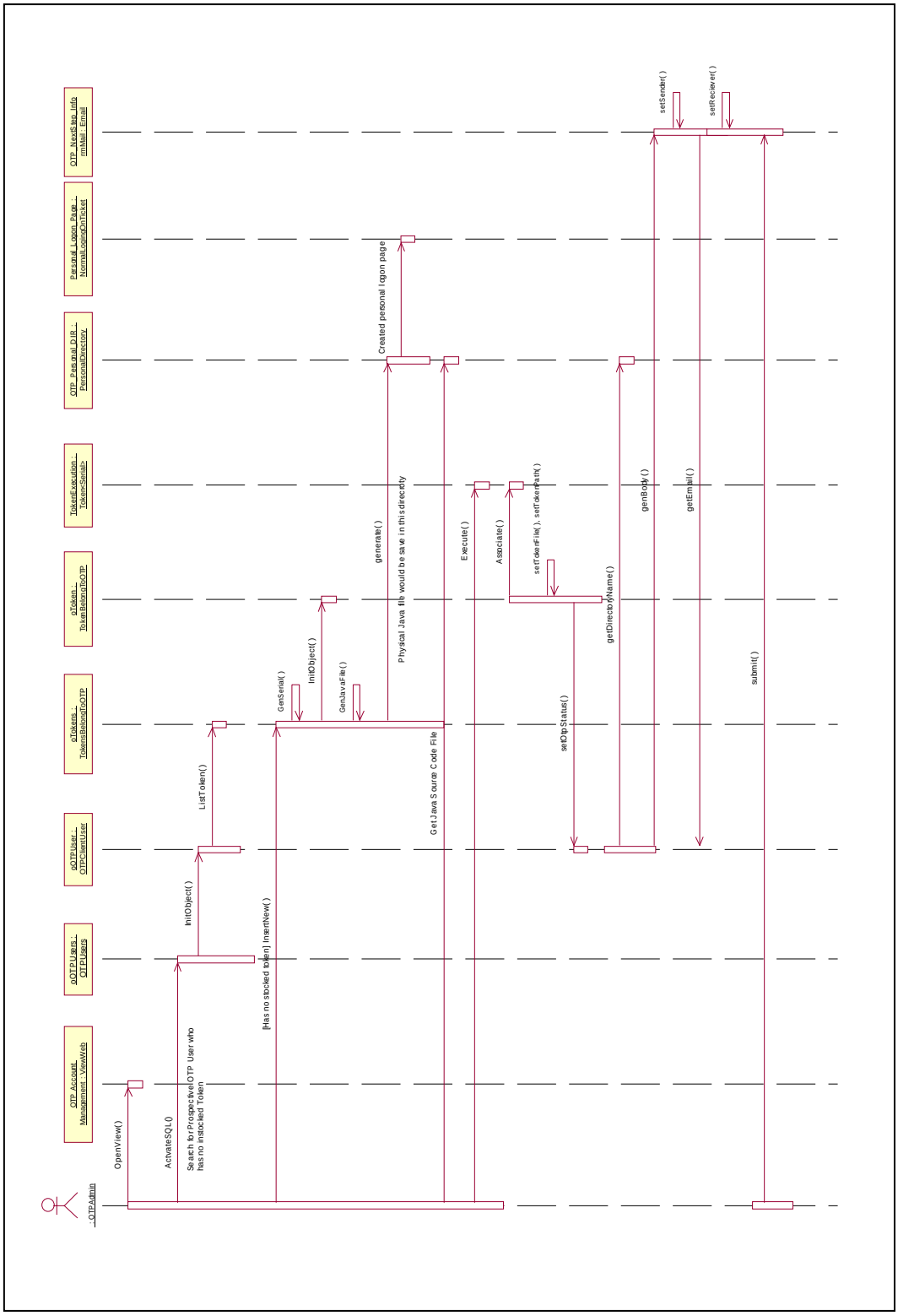


Figure 4-15: Sequence diagram (generate OTP generator)

Sequence diagram (generate OTP generator): description

Generate OTP Generator would be started by obligation of OTPAdmin. On administration web page, prospective OTP user profile would be found by querying. Whenever prospect OTP user was shown, next command of this scenario would be triggered to list all of tokens that owned by the prospect OTP record (oTokens object). If has no token (or not enough), Administrator triggered insert command then set of tokens would generate new serial number and initiated new object with this serial. While incoming token record was set in system database, parallel process was generating physical java file. Program source code was included serial number and profile of owner. This generated java file would be collected in personal directory of this OTP user. Personal directory would be generated if system could find that OTP user never had personal directory in server disk space (Naming of directory must correspond with user information) and specific web page for personal access to get installation was automatic created and collected in this directory. After physical source code file was generated, administrator had to go to pick that file and provided execution process then token installation was originated and also kept in personal directory. Installation file would be associated with token record that file name and path of directory would be collected as profile of token installation record (this path and file would be used for download hyperlink in next process). After that personal logon page in directory would be automatic generated in email body as URL for allow email receiver link to the personal web page to access to data in their personal directory. Email also collected necessary information such as receiver email address, first name and last name from OTP client user object. Then administrator manually submitted this email to user.

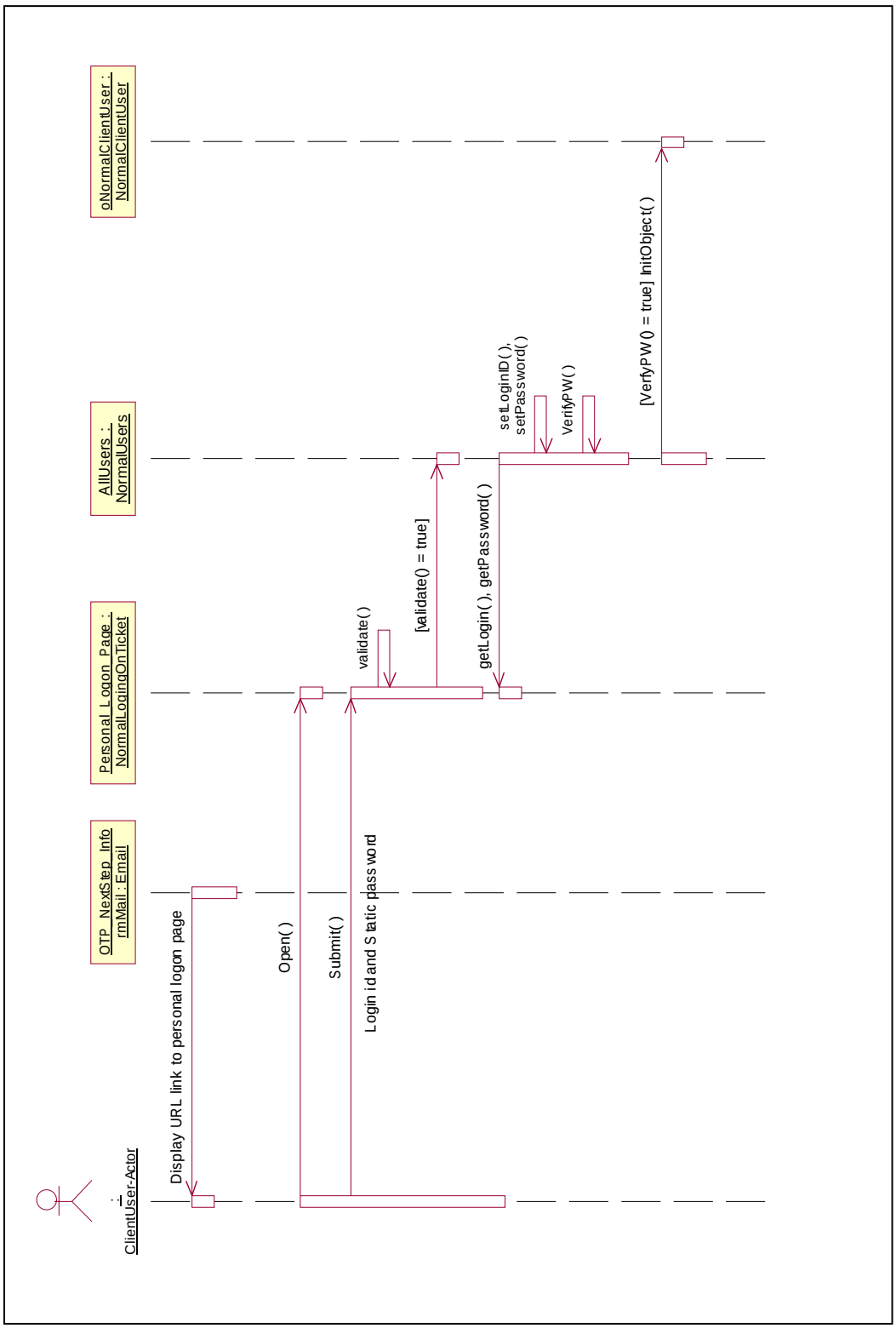


Figure 4-16: Sequence diagram (install OTP generator - 1)

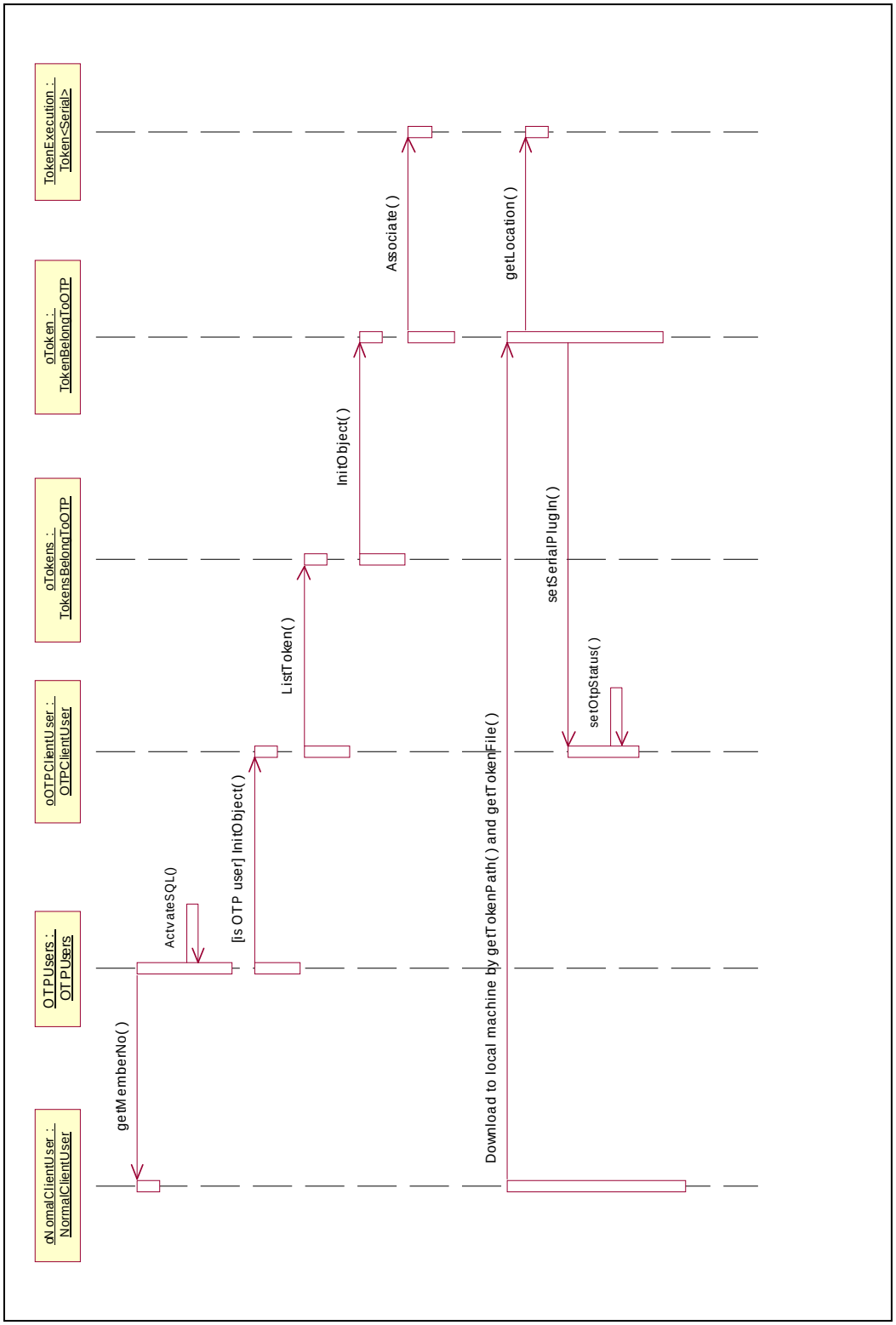


Figure 4-17: Sequence diagram (install OTP generator - 2)

Sequence diagram (install OTP generator – 1 and 2): description

Install OTP-Generator would be started after user could receive email, which contained information of personal logon page. URL in email content represented the location of specific form component, which was generated and contained in personal directory of user who had ever requested for OTP authorization. Normal level of authentication would be used to verify user, so login id and static password of member were passed to system again via personal logon component (Personal_Logon_Page object). Authentication would be verified with same logic as Normal Logon process. Finally user would have the authentication and normal user object (oNormalClientUser object) would be created to continue the process.

Normal user object was brought to verify again on member id attribute that was match with the one in set of OTP user or not. If system can identify OTP user by this member id, OTP user object would be initialized automatically (oOTPClientUser object). Then the set of tokens that belong to this OTP member were displayed on web page with their information. Location of installers were display in format of hyperlink that user could download one of token installation via this link. Whenever one of installer was downloaded, its serial number would be set to attribute of OTP user profile mean that this token serial was active for this member and prospective, current value member status, would be changed to active. After downloading, token application was installed subsequently on client machine.

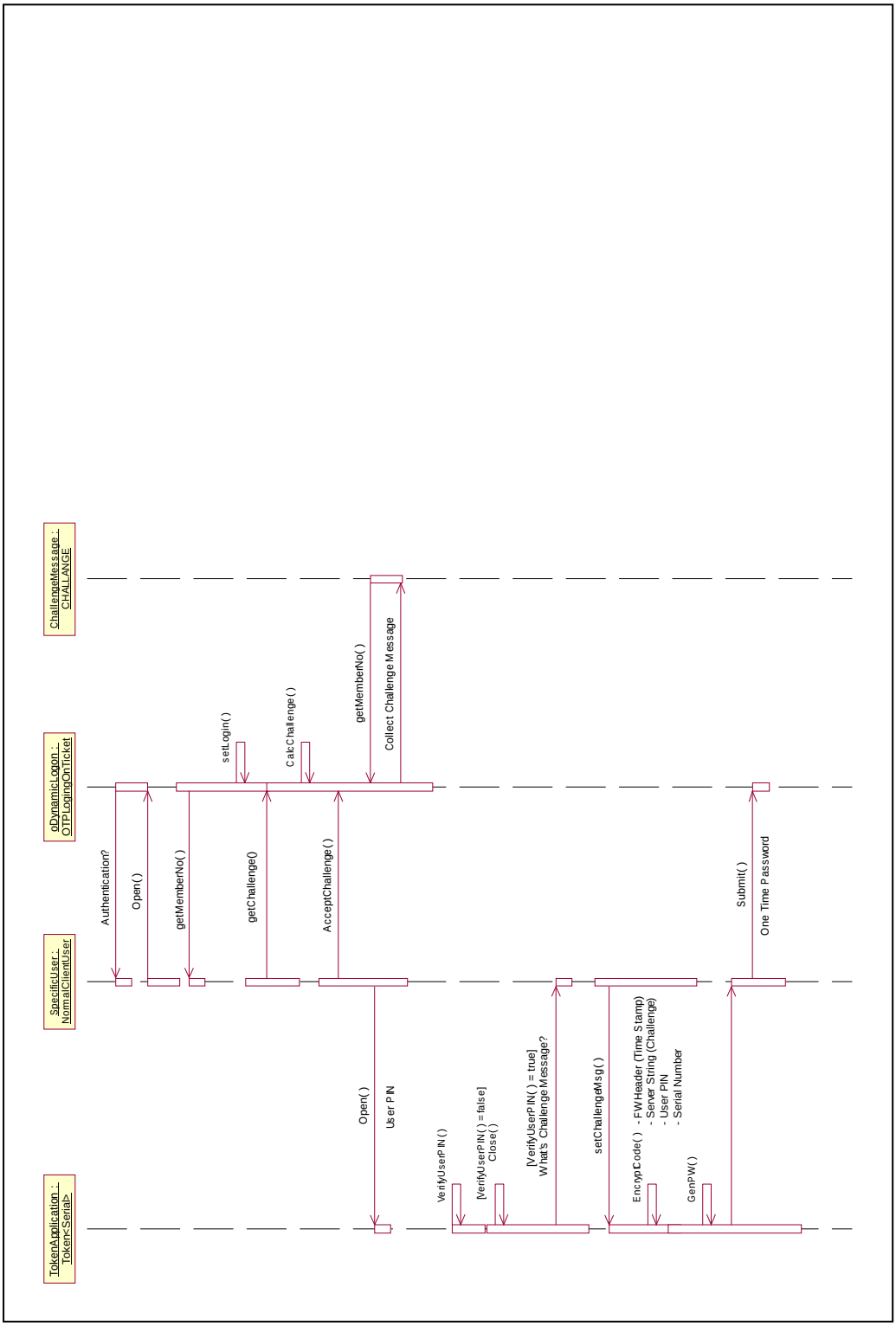


Figure 4-18: Sequence diagram (generate password by OTP generator)

Sequence diagram (generate password by OTP generator): description

Generate password by OTP Generator would be started when normal user who already has normal level authentication would like conduct his/her business on web service with special transaction. This kind of transaction required special authentication. It required One Time Password, which must be generated by specific application. Specific form component was used for this purpose. It could collect profile of normal user by include session object, which was generated at the end of normal logon process. So login id was not required, just only OTP message that user had to enter to the form. Not only required special password, but also it could return the challenge message, calculated message, which was combination of current timestamp and server string.

User (SpecificUser object) had to get challenge message from the logon form (oDynamicLogon object). After form return the message, user has to accept that message and that message would be collected as apart of user information (ChallengeMessage object). Then user had to open token application (TokenApplication object) on his/her local machine. To activate it user had to fill in User PIN into the application. Without the right User PIN application cannot be opened. After open token application user could bring challenge message entered to the application and algorithm inside would be activated. Encryption process inside the application would combine the plain information of Timestamp, Server string, User PIN and Token’s Serial number. Finally the cipher text, result of encryption, would be returned. That returned information was OTP-One Time Password which user could bring to submit to the system.

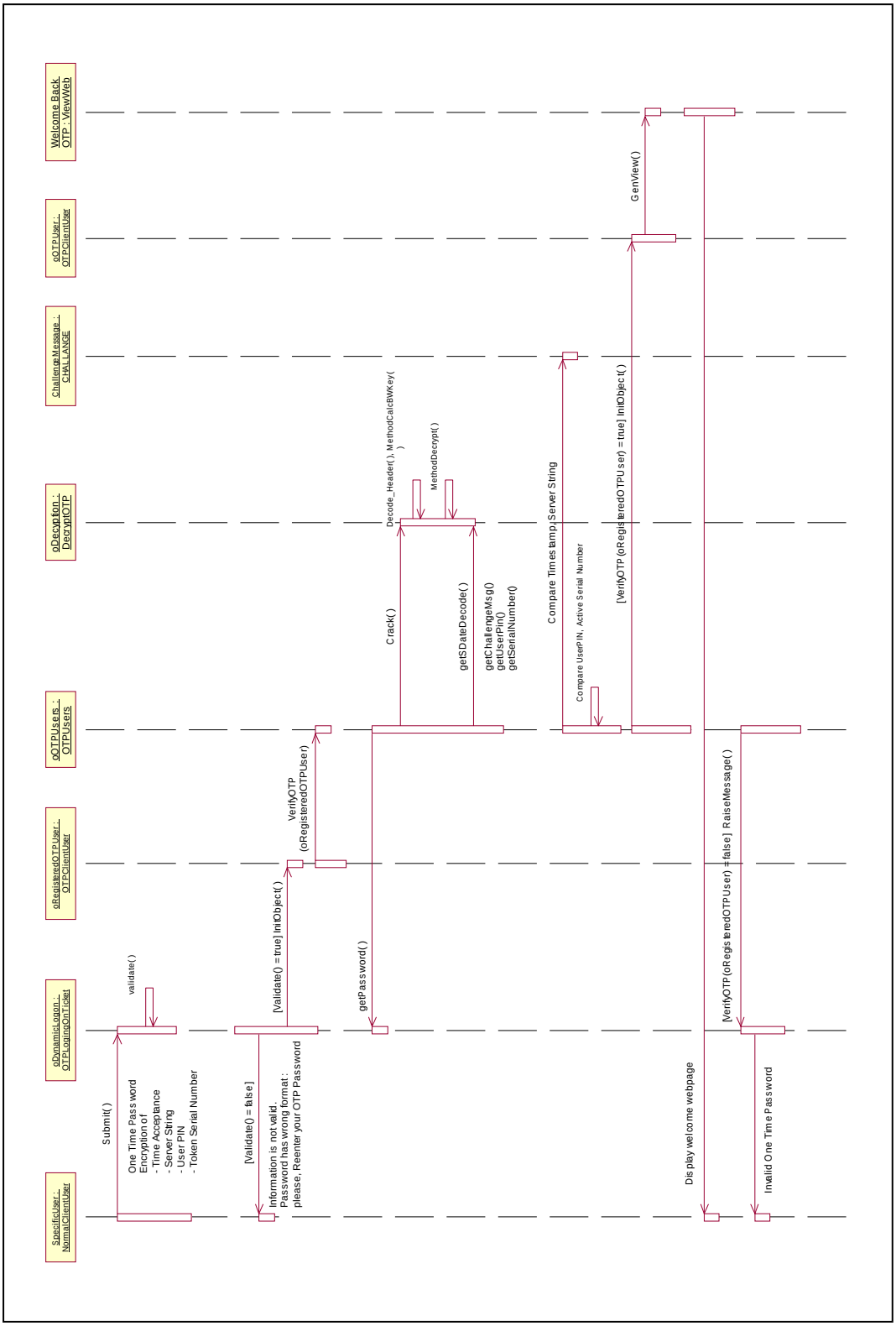


Figure 4-19: Sequence diagram (OTP logon)

Sequence diagram (OTP logon): description

OTP Logon would be started after One Time Password was retuned by local token application. Normal user (SpecificUser object) submitted the OTP massage to the specific logon form (oDynamicLogon object). Then method in form would be triggered to validate the OTP format. Appropriated format included the cipher text of Timestamp, Server string, User PIN and Token's Serial number. Valid message would initiate temporary object of OTP user profile (oRegisteredOTPUser object), which included OTP massage as password attribute. This object was passed to object of the set of OTP users (oOTP Users object) that had the method can verify coming OTP object. The process of decryption was starting with cracking the constructed message. Timestamp would be used as a main factor of calculation for the key of decryption. With the key, Server string was decoded and compared with the last challenge message of user. User PIN was decoded and compared with attribute of OTP user object. And Serial Number was decoded and compared with Serial Num of current active token, which belong to OTP user. If all of information could be matched with the records that related together in relational database, mean that verification of OTP was returned true. Whenever verification was return true, OTP user would be initialized as session object (oOTPUser object) and incoming user could get authentication for special transaction of this web service.

4.4 Demonstrative algorithm for Encryption and Decryption

Alternative authentication in this study is an approach, which was demonstrated for the concept to using generating One Time Password for specific authentication on web-based service. Assumption of study based on the idea that whenever client users would like to conduct the high level transaction, they have to get the special authentication also.

Correspond with the purpose of this study. Due to demonstration idea, encryption algorithm in this study was adapted by most simple algorithm-Caesar's cipher. Although alternative algorithm was adapted in the part of selected appropriated key of encode correspond with time value of activating, the security for this algorithm was still expecting just secure equal to Caesar' cipher algorithm.

This section describes how to encode and decode the necessary information with alternative algorithm.

Monoalphabatic Ciphers (Substitutions)

Caesar's cipher, the simplest algorithm of monoalphabatic ciphers (or simple substitution), was referred in this study.

```
REF =  "ABCDEFGHIJKLMNOPQRSTUVWXYZ+-
*/_?<>!#$%^&():[];{},=.'0123456789abcdefghijklmnopqrstuvwxyz" + " "
```

String REF would be using as reference while the both process of encryption and decryption was activated. String can be converted to array of character, which has sizing accord with the length of String.

Array of Character:

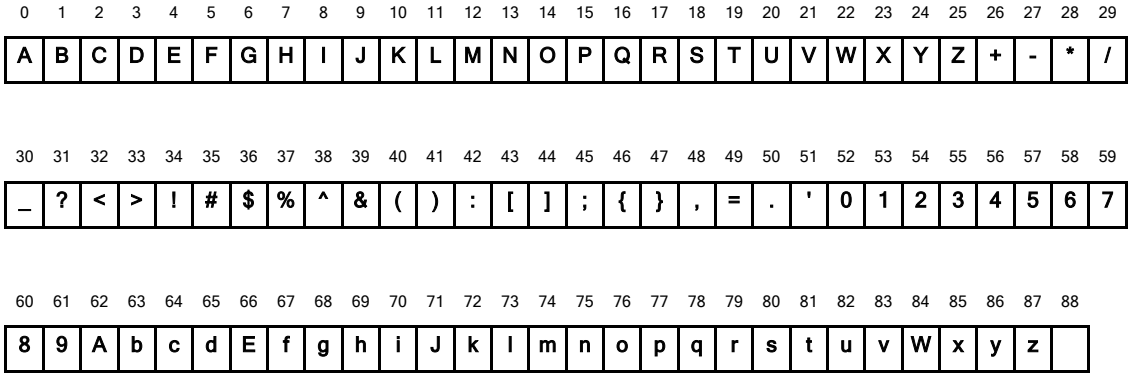


Figure 4-20: Array of character

Likewise, the following relation can also represent this sequence of array.

r_{REF} = {(x,n) | there is n, which $x_n \in$ character set of REF and n is index of array}

or

r_{REF} = {(A,0), (B,1), (C,2), (D,3), (E,4), (F,5), (G,6), (H,7), (I,8), (J,9), (K,10), (L,11), (M,12), (N,13), (O,14), (P,15), (Q,16), (R,17), (S,18), (T,19), (U,20), (V,21), (W,22), (X,23), (Y,24), (Z,25), (+,26), (-,27), (*,28), (/,29), (_,30), (?,31), (<,32), (>,33), (!,34), (#,35), ($,36), (%,37), (^,38), (&,39), ((,40), (),41), (:,42), ([,43), (],44), (;,45), ({,46), (},47), (,,48), (=,49), (.,50), (',51), (0,52), (1,53), (2,54), (3,55), (4,56), (5,57), (6,58), (7,59), (8,60), (9,61), (a,62), (b,63), (c,64), (d,65), (e,66), (f,67), (g,68), (h,69), (i,70), (j,71), (k,72), (l,73), (m,74), (n,75), (o,76), (p,77), (q,78), (r,79), (s,80), (t,81), (u,82), (v,83), (w,84), (x,85), (y,86), (z,87), (<space>,88)}

With this reference relation, system can compare the incoming characters with array values so that get the index of that character to proceed on the continued encrypt or decrypt process.

Let REF is the universe of this study.

This below function would be used for monoalphabet, for all characters can get value of index by compare it with character set of REF

$$f_{Index}(x) = y$$

Such as for character ‘R’, the value of index in REF can be found by $f_{Index}(R) = 17$

And next function, for all integers in range of index of array can be converted to character

$$f_{Char}(y) = x$$

For index of character equal to 17, the right character in REF can be found by $f_{Char}(17) = R$

With the ordered pair which represent array of REF string and the both of basic functions, not only get the index of specific character in REF, but also can identify the right character in REF whenever index of array can be specified. So this concept can be considered for the simple encryption and decryption process.

From ordered pair of REF and both of basic functions above, so

$$f_{Char}(f_{Index}(x)) = x$$

If forward key equal to 3, this forward key can be added to index of specific character to decode that character to be cipher message (Caesar Cipher Text), so

$$f_{Encode}(x) = f_{Char}((f_{Index}(x) + 3) \text{ mod by } 89)$$

Such as character ‘R’ index of cipher character of ‘R’ base on REF is, (f_{Index}(R) + 3) mod by 89 equal to (17+3) mod by 89 = 20

Then cipher character of ‘R’ in term of Caesar Cipher Text, which base on REF is, f_{Encode}(R) = f_{Char}(20) so, the answer is ‘U’

On another hand, 3 can be set to backward key in order to decode cipher character to be plain character. This concept can be considered for decrypting function.

$$f_{Decode}(x') = f_{Char}(((f_{Index}(x') + 89) -3) \text{ mod by } 89)$$

Such as character ‘U’ index of plain character base on REF is, (f_{Index}(U) + 89) – 3) mod by 89 equal to ((20+89) -3) mod by 89 = 17

Then plain character of ‘U’ in term of Caesar Cipher Text, which base on REF is, f_{Decode}(U) = f_{Char}(17) so, the answer is ‘R’

Example

Let incoming message is
Plain Text = “Anusorn”

The appropriated value of index in REF can be found by f_{Index}(x)

f_{Index}(A) = 0 so (f_{Index}(A) + 3) mod by 89 = 3

f_{Index}(n) = 75 so (f_{Index}(n) + 3) mod by 89 = 78

f_{Index}(u) = 82 so (f_{Index}(u) + 3) mod by 89 = 85

f_{Index}(s) = 80 so (f_{Index}(s) + 3) mod by 89 = 83

$f_{Index}(o) = 76$ so $(f_{Index}(o) + 3)$ mod by 89 = 79

$f_{Index}(r) = 79$ so $(f_{Index}(r) + 3)$ mod by 89 = 82

$f_{Index}(n) = 75$ so $(f_{Index}(n) + 3)$ mod by 89 = 78

And can identify each encoded character by, $f_{Encode}(x) = f_{Char}((f_{Index}(x) + 3)$ mod by 89)

$f_{Encode}(A), f_{Char}(3) =$ ‘D’

$f_{Encode}(n), f_{Char}(78) =$ ‘q’

$f_{Encode}(u), f_{Char}(85) =$ ‘x’

$f_{Encode}(s), f_{Char}(83) =$ ‘v’

$f_{Encode}(o), f_{Char}(79) =$ ‘r’

$f_{Encode}(r), f_{Char}(82) =$ ‘u’

$f_{Encode}(n), f_{Char}(78) =$ ‘q’

Then Cipher text is consolidation of these monoalphabet functions
So, encode of Plain Text = “Anusorn” is Cipher Text = “Dqxvruq”

These below JAVA source code are a part of method for encoding message.

To proof these source code as Caesar’s cipher algorithm
Let iFWKey = 3, sWord = “Anusorn”

```
public static int getIFWKey(){
  return iFWKey;
}

public static String getSWord(){
  return sWord;
}

public static void MethodEncrypt(){
  int giFWKey = getIFWKey();
  String gsWord = getSWord();
  int i=0, j=0, iaREF, iaWord;
  String sREF =  "ABCDEFGHIJKLMNOPQRSTUVWXYZ+-
  */_?<>!#$%^&():[];{},=.'0123456789abcdefghijklmnopqrstuvwxyz" + " ";
  String sResult="";
  char aREF[];
  aREF = sREF.toCharArray();
  iaREF = sREF.length();

  if(giFWKey>iaREF)
        giFWKey = giFWKey%iaREF;

  char aWord[];
  aWord = gsWord.toCharArray();
  iaWord = gsWord.length();
  for(i=0; i<iaWord; i++){
    for(j=0; j<iaREF; j++){
        if(String.valueOf(aWord[i]).equals(String.valueOf(aREF[j]))){
            if((j + giFWKey)<iaREF)
                sResult += String.valueOf(aREF[(j + giFWKey)]);
```

```
                else
                    sResult += String.valueOf(aREF[(j + giFWKey)-iaREF]);
            }
          }
        }
        setSEncrypt(sResult);
      }

      public static void setSEncrypt(String pasEncrypt){
        sEncrypt = pasEncrypt;
      }
```

Finally, attribute name sEncrypt would be set value ‘Dqxvruq’

On another hand, Cipher Text = “Dqxvruq” can be decoded

The appropriated value of index in REF can be found by $f_{Index}(x)$

$f_{Index}(D) = 3$ so $((f_{Index}(D) + 89) -3)$ mod by $89 = 0$

$f_{Index}(q) = 78$ so $((f_{Index}(q) + 89) -3)$ mod by $89 = 75$

$f_{Index}(x) = 85$ so $((f_{Index}(x) + 89) -3)$ mod by $89 = 82$

$f_{Index}(v) = 83$ so $((f_{Index}(v) + 89) -3)$ mod by $89 = 80$

$f_{Index}(r) = 79$ so $((f_{Index}(r) + 89) -3)$ mod by $89 = 76$

$f_{Index}(u) = 82$ so $((f_{Index}(u) + 89) -3)$ mod by $89 = 79$

$f_{Index}(q) = 78$ so $((f_{Index}(q) + 89) -3)$ mod by 89 =75

And can identify each decoded character by, $f_{Decode}(x') = f_{Char}(((f_{Index}(x') + 89) -3)$ mod by 89)

$f_{Decode}(D), f_{Char}(0) =$ ‘A’

$f_{Decode}(q), f_{Char}(75) =$ ‘n’

$f_{Decode}(x), f_{Char}(82) =$ ‘u’

$f_{Decode}(v), f_{Char}(80) =$ ‘s’

$f_{Decode}(r), f_{Char}(76) =$ ‘o’

$f_{Decode}(u), f_{Char}(79) =$ ‘r’

$f_{Decode}(q), f_{Char}(75) =$ ‘n’

Then Plain text is consolidation of these monoalphabet functions
So, decode of Cipher Text = “Dqxvruq” is Plain Text = “Anusorn”

These below JAVA source code are a part of method for decoding cipher message.

To proof these source code as Caesar’s cipher algorithm
Let iBWKey = 3, sEncrypt = “Dqxvruq”

```
public static int getIBWKey(){
  return iFWKey;
}
```

```
public static String getSEncrypt(){
  return sEncrypt;
}

public void MethodDecrypt(){
  int giBWKey = getIBWKey();
  String gsEncrypt = getSEncrypt();
  int i=0, j=0, iaREF, iaWord;
  String sREF =  "ABCDEFGHIJKLMNOPQRSTUVWXYZ+-
  */_?<>!#$%^&():[];{},=.'0123456789abcdefghijklmnopqrstuvwxyz" + " ";
  String sResult="";
  char aREF[];
  aREF = sREF.toCharArray();
  iaREF = sREF.length();
  if(giBWKey>iaREF)
    giBWKey = giBWKey%iaREF;
  char aWord[];
  aWord = gsEncrypt.toCharArray();
  iaWord = gsEncrypt.length();
  for(i=0; i<iaWord; i++){
    for(j=0; j<iaREF; j++){
        if(String.valueOf(aWord[i]).equals(String.valueOf(aREF[j]))){
            if((j - giBWKey)<iaREF && (j - giBWKey)>=0)
              sResult += String.valueOf(aREF[(j - giBWKey)]);
            else
              sResult += String.valueOf(aREF[(j - giBWKey)+iaREF]);
        }
    }
  }
  setSDecrypt(sResult);
```

```
    }

    public static void setSDecrypt(String pasDecrypt){
      sDecrypt = pasDecrypt;
    }
```

Finally, attribute name sDecrypt would be set value ‘Anusorn’

Caesar’s cipher with time stamp forward key, adapted algorithm for this study

As previous provable algorithm, forward key ‘3’ is essence of encode and decode logic, whenever encoder and decoder even know exactly about REF string. This section has the purpose to demonstrate the adaptable algorithm, which was applied from previous logic. Demonstrated algorithm, Caesar’s cipher algorithm with has the variant forward key, has dynamic forward key which was not fixed the value as 3, but it was depended on the time stamp of beginning of encode process. In every second forward key will be changed by the logic that calculated from each number of date-time values. (By the format of dd/mm/yy hh:nn:ss such as 25/11/04 10:12:08 has 6 values are 25, 11, 04, 10, 12, 8)

To disguise the time stamp, as the header of cipher message, process of encoding and decoding must has reference string for each integer values of date-time to pretend interrupters by represent themselves by reference characters.

```
HDREF =
“ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz012345
67”
```

For this study, only 60 characters were selected to be the reference characters. It meant the most possible value of date-time in this study can be between 0 – 59.

(These are possible exception values of second's integer. However the value of short year's integer can be larger than 59 such as in 2060 integer of this year will be 60. This demonstrated algorithm would be limited within current year 2004 to 2059)

Array of Character:

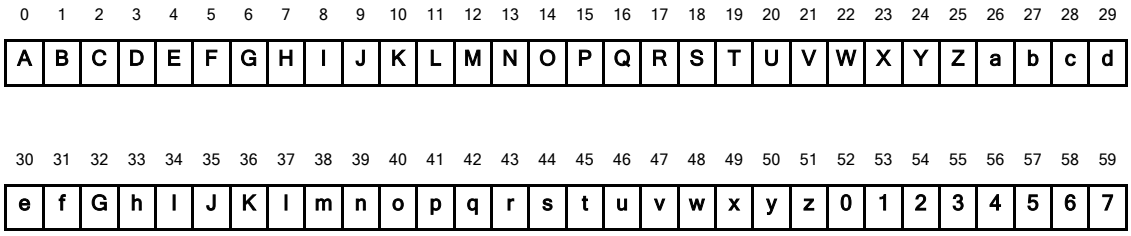


Figure 4-21: Array of character for date-time encryption

Likewise, the following relation can also represent this sequence of array.

r_{HD} = {(x,n) | there is n, which $x_n \in$ character set of REF and n is index of array}

or

r_{HD} = {(A,0), (B,1), (C,2), (D,3), (E,4), (F,5), (G,6), (H,7), (I,8), (J,9), (K,10), (L,11), (M,12), (N,13), (O,14), (P,15), (Q,16), (R,17), (S,18), (T,19), (U,20), (V,21), (W,22), (X,23), (Y,24), (Z,25), (a,26), (b,27), (c,28), (d,29), (e,30), (f,31), (g,32), (h,33), (i,34), (j,35), (k,36), (l,37), (m,38), (n,39), (o,40), (p,41), (q,42), (r,43), (s,44), (t,45), (u,46), (v,47), (w,48), (x,49), (y,50), (z,51), (0,52), (1,53), (2,54), (3,55), (4,56), (5,57), (6,58), (7,59)}

With this reference relation, system can compare the incoming characters, the representative of cipher message header, with array values in order to get the index of that character to proceed on the continued encrypt or decrypt process.

How to encode and decode Time Stamp (Cipher text from date-time values)

Let HD is the universe of this study.

This below function would be used for monoalphabet, for all characters can get value of index by compare it with character set of HD

$$f_{IndexHD}(x) = y$$

Such as for character ‘Z’, the value of index in REF can be found by $f_{Index}(Z) = 25$

And next function, for all integers in range of index of array can be converted to character

$$f_{CharHD}(y) = x$$

For index of character equal to 25, the right character in REF can be found by $f_{Char}(25) = Z$

For example

- Let Time Stamp equal to 23/09/04 13:22:55

  Date-time value can be separated to 6 element values. Then can consider on each value, refer to index of character set of reference HDREF string.

  - DD is equal to 23, so $f_{CharHD}(23) = X$
  - MM is equal to 9, so $f_{CharHD}(9) = J$
  - YY is equal to 4, so $f_{CharHD}(4) = E$
  - HH is equal to 13, so $f_{CharHD}(13) = N$
  - NN is equal to 22, so $f_{CharHD}(22) = W$
  - SS is equal to 55, so $f_{CharHD}(55) = 3$

  Then Cipher text is consolidation of these monoalphabet functions

  So, encode of Time Stamp = “23/09/04 13:22:55” is Cipher Text of Time Stamp = “XJENW3”

These below JAVA source code are a part of method about generating header of cipher message, which encode from values of current date-time.

To proof these source code

Let getSDate() = “23/09/04 13:22:55”, getIDayCrack() = 23, getIMonthCrack() = 9, getIYearCrack() = 4, getIHourCrack() = 13, getIMinuteCrack() = 22, getISecondCrack() = 55

```
public static void Gen_sFWHeader(){
    String gDate = getSDate();
    String sREF =
    "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz01
    23456789";
    String sDay, sMonth, sYear, sHour, sMinute, sSecond;
    String sResult ="";
    int iaREF, iDIV;
    char aREF[];
    aREF = sREF.toCharArray();
    iaREF = sREF.length();

    iDIV = getIDayCrack() / iaREF + 1;
    if(getIDayCrack()<iaREF)
            sDay = String.valueOf(aREF[getIDayCrack()]);
              else
       sDay = iDIV + String.valueOf(aREF[getIDayCrack()%iaREF]);

    iDIV = getIMonthCrack() / iaREF + 1;
    if(getIMonthCrack()<iaREF)
      sMonth = String.valueOf(aREF[getIMonthCrack()]);
    else
```

```
        sMonth = iDIV + String.valueOf(aREF[getIMonthCrack()%iaREF]);
      iDIV = getIYearCrack() / iaREF + 1;
      if(getIYearCrack()<iaREF)
        sYear = String.valueOf(aREF[getIYearCrack()]);
      else
        sYear = iDIV + String.valueOf(aREF[getIYearCrack()%iaREF]);

      iDIV = getIHourCrack() / iaREF + 1;
      if(getIHourCrack()<iaREF)
        sHour = String.valueOf(aREF[getIHourCrack()]);
      else
        sHour = iDIV + String.valueOf(aREF[getIHourCrack()%iaREF]);

      iDIV = getIMinuteCrack() / iaREF + 1;
      if(getIMinuteCrack()<iaREF)
        sMinute = String.valueOf(aREF[getIMinuteCrack()]);
      else
        sMinute = iDIV + String.valueOf(aREF[getIMinuteCrack()%iaREF]);

      iDIV = getISecondCrack() / iaREF + 1;
      if(getISecondCrack()<iaREF)
        sSecond = String.valueOf(aREF[getISecondCrack()]);
      else
        sSecond = iDIV + String.valueOf(aREF[getISecondCrack()%iaREF]);

      sResult = sDay + sMonth + sYear + sHour + sMinute + sSecond;
      setSFWHeader(sResult);
    }

    public static void setSFWHeader(String pasFWHeader){
```

```
        sFWHeader = pasFWHeader;
    }
```

Finally, attribute name sFWHeader would be set value 'XJENW3'

- On another hand, Let Cipher Text of Time Stamp is "XJENW3" Each cipher character can be considered to refer to character set of HDREF string.
  - "X" represent for DD, so DD is $f_{IndexHD}(X) = 23$
  - "J" represent for MM, so MM is $f_{IndexHD}(J) = 9$
  - "E" represent for YY, so YY is $f_{IndexHD}(E) = 4$
  - "N" represent for HH, so HH is $f_{IndexHD}(N) = 13$
  - "W" represent for NN, so NN is $f_{IndexHD}(W) = 22$
  - "3" represent for SS, so SS is $f_{IndexHD}(3) = 55$

Then Plain text of Time Stamp is consolidation of these monoalphabet functions

So, decode of Cipher Text of Time Stamp = "XJENW3" is Plain text of Time Stamp = "23/09/04 13:22:55"

These below JAVA source code are a part of method about decoding header, which was sent with secret information as cipher message.

To proof these source code

Let sFWHeader = "XJENW3"

```
public String getSFWHeader(){
  return(sFWHeader);
}
```

```
public void Decode_Header(){
  String gHeader = getSFWHeader();
  String sResult="", sMatch="";
  String sREF =
  "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz012
  3456789";
  int iaREF;
  char aREF[];
  aREF = sREF.toCharArray();
  iaREF = sREF.length();

  int iaHeader;
  char aHeader[];
  aHeader = gHeader.toCharArray();
  iaHeader = gHeader.length();

  int i=0, j=0;
  for(i=0; i<iaHeader; i++){
        for(j=0; j<iaREF; j++){
              if(aHeader[i]==aREF[j])
                    if(i==2){
                          sMatch = String.valueOf(j+2000);
                    }else{
                          if(j<10)
                                sMatch = "0" + String.valueOf(j);
                          else
                                sMatch = String.valueOf(j);
                    }
        }
        switch(i){
```

```
                    case 0:
                        sResult = sMatch + "/";
                        break;
                    case 1:
                        sResult += sMatch + "/";
                        break;
                    case 2:
                        sResult += sMatch + " ";
                        break;
                    case 3:
                        sResult += sMatch + ":";
                        break;
                    case 4:
                        sResult += sMatch + ":";
                        break;
                    case 5:
                        sResult += sMatch;
                        break;
                    }
          }
          setSDateDecode(sResult);
        }

        public void setSDateDecode(String paSDateDecode){
          sDateDecode = paSDateDecode;
        }
```

Finally, attribute name sDateDecode would be set value “23/09/04 13:22:55”

How to calculate forward key

One most important concept of this demonstrated approach of One – Time Password authentication is a formula of dynamic forward key calculation. While calling a formula, a Date-Time value of that moment would be converted to milliseconds such as “23/09/2004 13:22:08” was a value of a time stamp, so milliseconds of a time stamp is “1095920528000” and on the next second time stamp is “23/09/2004 13:22:09” its millisecond’s value is “1095920529000”.

There is the function in Java Platform, which return long number can find out the value of milliseconds since midnight on January 1, 1970.

The appropriate number could divide these millisecond values to find out the modulus. There are so many conditions have to be considered in calculation and there are some chances, which have the divider as 2 or any odd numbers that can divide 1000 with modulus equal to 0. Even the definition of One-Time is the value always is changed second by second but conversion of time stamp was represented in term of volume of 1000 so, the modulus will be 0 whenever the divider equal to 2 or any odd numbers that can divide 1000 with modulus equal to 0.

So, it is the congenial to convert Date-Time value to the number of seconds since midnight on January 1, 1970. (Millisecond’s value / 1000)

There’s more than one conditions must be considered for calculate a forward key value. One of these below formulas will be chosen in case of calculate forward key from date-time values

- **FWkey =** modulus of **(date-time’s milliseconds)/1000** divide by its **second value** whenever second value is not equal to 0 and modulus is not equal to 0

- **FWkey =** modulus of **(date-time’s milliseconds)/1000** divide by its **minute value** whenever (second value is equal to 0 or modulus by second is equal to 0) and minute value is not equal to 0 and modulus is not equal to 0

- **FWkey =** modulus of **(date-time's milliseconds)/1000** divide by its **hour value** whenever (second value is equal to 0 or modulus by second is equal to 0) and (minute value is equal to 0 or modulus by minute is equal to 0) and hour value is not equal to 0 and modulus is not equal to 0

- **FWkey =** modulus of **(date-time's milliseconds)/1000** divide by its **day value** whenever (second value is equal to 0 or modulus by second is equal to 0) and (minute value is equal to 0 or modulus by minute is equal to 0) and (hour value is equal to 0 or modulus by hour is equal to 0) and day value is not equal to 0 and modulus is not equal to 0

- **FWkey =** modulus of **(date-time's milliseconds)/1000** divide by its **month value** whenever (second value is equal to 0 or modulus by second is equal to 0) and (minute value is equal to 0 or modulus by minute is equal to 0) and (hour value is equal to 0 or modulus by hour is equal to 0) and (day value is equal to 0 or modulus by day is equal to 0) and month value is not equal to 0 and modulus is not equal to 0

- **FWkey =** modulus of **(date-time's milliseconds)/1000** divide by its **year value** whenever (second value is equal to 0 or modulus by second is equal to 0) and (minute value is equal to 0 or modulus by minute is equal to 0) and (hour value is equal to 0 or modulus by hour is equal to 0) and (day value is equal to 0 or modulus by day is equal to 0) and (month value is equal to 0 or modulus by month is equal to 0) and year value is not equal to 0 and modulus is not equal to 0

- **FWkey = 3** whenever

modulus of **(date-time's milliseconds)/1000** divide by its **second value** is equal to 0 and
modulus of **(date-time's milliseconds)/1000** divide by its **minute value** is equal to 0 and
modulus of **(date-time's milliseconds)/1000** divide by its **hour value** is equal to 0 and
modulus of **(date-time's milliseconds)/1000** divide by its **day value** is equal to 0 and
modulus of **(date-time's milliseconds)/1000** divide by its **month value** is equal to 0 and
modulus of **(date-time's milliseconds)/1000** divide by its **year value** is equal to 0

For Example

- Let Time Stamp equal to "06/02/2005 01:18:59"
  With accurate formula,
  modulus of **(date-time's milliseconds)/1000** divide by its **second value**, so forward key can be found that modulus of 1107627539 divide by 59 equal to 7

- Let Time Stamp equal to "06/02/2005 01:17:57"
  Because of modulus of **(date-time's milliseconds)/1000** divide by its **second value**, 1107627477 divide by 57 equal to 0
  With accurate formula
  modulus of **(date-time's milliseconds)/1000** divide by its **minute value**, so forward key can be found that modulus of 1107627477 divide by 17 equal to 8

- Let Time Stamp equal to "06/02/2005 01:32:32"
  Because of modulus of **(date-time's milliseconds)/1000** divide by its **second value**, 1107628352 divide by 32 equal to 0

and modulus of **(date-time's milliseconds)/1000** divide by its **minute value**, 1107628352 divide by 32 equal to 0

and modulus of **(date-time's milliseconds)/1000** divide by its **hour value**, 1107628352 divide by 1 equal to 0

With accurate formula

modulus of **(date-time's milliseconds)/1000** divide by its **day value**, so forward key can be found that modulus of 1107628352 divide by 6 equal to 2

These below JAVA source code are a part of method about finding forward key, which calculated from date-time values.

To proof these source code

Let String sDate = "06/02/2005 01:17:57".

```
public String getSDate(){
  return sDate;
}

public void MethodCalcFWKey(){
  java.util.Date oDate = new java.util.Date();
  try{
        java.text.SimpleDateFormat formatter = new
        java.text.SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
        formatter.setLenient(false);
        oDate = formatter.parse(getSDate());
  }catch(java.text.ParseException e){
        System.out.println(e.getMessage());
  }
  GregorianCalendar calendar = new GregorianCalendar();
```

```
calendar.setTime(oDate);
int iSecond = calendar.get(Calendar.SECOND);
int iMinute = calendar.get(Calendar.MINUTE);
int iHour = calendar.get(Calendar.HOUR_OF_DAY);
int iDay = calendar.get(Calendar.DAY_OF_MONTH);
int iMonth = calendar.get(Calendar.MONTH) + 1;
int iYear = calendar.get(Calendar.YEAR);
        if(iYear >= 2000)
                iYear = iYear - 2000;

long lMilliSec = oDate.getTime();
lMilliSec = lMilliSec / 1000;
Long lResult = null;
if( iSecond != 0 && (lMilliSec%iSecond) != 0 )
        lResult = new Long(lMilliSec%iSecond);
else if ( iMinute != 0 && (lMilliSec%iMinute) != 0 )
        lResult = new Long(lMilliSec%iMinute);
else if ( iHour != 0 && (lMilliSec%iHour) != 0 )
        lResult = new Long(lMilliSec%iHour);
else if ( iDay != 0 && (lMilliSec%iDay) != 0 )
        lResult = new Long(lMilliSec%iDay);
else if ( iMonth != 0 && (lMilliSec%iMonth) != 0 )
        lResult = new Long(lMilliSec%iMonth);
else if ( iYear != 0 && (lMilliSec%iYear) != 0 )
        lResult = new Long(lMilliSec%iYear);
else
        lResult = new Long(3);

int iFWKey = lResult.intValue();
setIFWKey(iFWKey);
```

```
    }

    public static void setIFWKey(int paiFWKey){
      iFWKey = paiFWKey;
    }
```

Finally, attribute name iFWKey would be set value ‘8’

Collecting and encoding individual information (Generate One Time Password)

Circumspect authentication usually needs the necessary information, which can be used to identify owner accurately. The approach of authentication in this study also needs the specific member information. Some information would be stored in system database by user registration process while operating member’s log on process force other information were immediately generated and be distributed to both of member and system.

In process of user registration, member profile with appropriated format would be stored in system database such as member first name, last name, email address etc. Especially for OTP authentication, **User PIN** is the additional information, which member must generate and enter it to the system.

After OTP registration process, system administrators would be noticed to operate their job about generating the set of token application plug in relate to OTP member profile. (Initialize 3 tokens for each new coming OTP member) Unique serial number would be included in token source code. Then registered user will be informed to download one of productive token and automatically set its serial to be **Active Token Serial** for that member.

According with Challenge-Response system, while authentication process was performing user has to request challenge message from authentication system. The challenge message was included information to identify the source of message or specific information, which has the purpose to displaying to destination receiver for encoding then, sends it back to source. In this study challenge message would be

created by system whenever user request and it would be stored as meantime relation of user profile. So a part of challenge message can be included **Time Stamp** and randomized **Server String**. The objective of Time Stamp is basis of generating forward key of encryption. And Server String was the identical information for source of message generator.

Whenever encryption process was triggered, user had to put User PIN to activate his/her own token application, which was already included information of **User PIN** and **Active Token Serial**. Then he/she had to continue access to authentication system to get challenge message, which has necessary information about **Time Stamp** and **Server String** that related with OTP user profile. Finally all of individual information would be combined and encode by particular algorithm. (Caesar's Cipher with time stamp forward key) Result of encryption is the particular format of string which separates information by preserve character '@'. The format of cipher text is 'encode of **Time Stamp**@encode of **Server String**@encode of **User PIN**@encode of **Active Token Serial**'.

For example

- Let defined string, using for date-time encoding is "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz01234567"

- Let Time Stamp equal to 06/02/2005 01:58:22 So, the encode of this time stamp is "GBFB6W" and value of forward key is 16

- Let defined string, using for content encoding is "ABCDEFGHIJKLMNOPQRSTUVWXYZ+-*/_?<>!#$%^&():[];{},=.'0123456789abcdefghijklmnopqrstuvwxyz" + " "

- Let Server String is "Monoalphabatic" So, the encode of this string is "*DCDqAExqrqIys"
- Let User PIN is "demo" So, the encode of this string is "tuBD"

- Let Active Token Serial is “1-93675” So, the encode of this string is “h[pjmnl”

Then Cipher text is consolidation of these encoded information, which was separated by special character “@”. So, cipher text is

“GBFB6W@*DCDqAExqrqIys@tuBD@h[pjmnl”

Decoding and verifying individual information (Verify One Time Password)

Authentication system usually gets the request from visitors who need to access for conducting their business. OTP message was sent with login id to request for special grant access to deeper transaction while the owner of login id was assuming the normal grant access. Static password was used at first for access to web service, mean that the owner of login id could prove himself/herself to the system and can conduct his/her business in the normal level of service. Whenever licensee would like to access to conduct special business. He/She had to request for challenge message and bring it to precede a One Time Password as the explanation in previous section (Collecting and encoding individual information)

This section is underlining for mechanism of cracking, decoding OTP message and include separating information to verify with user profile, which were recorded in system database.

After receiving OTP, system suddenly verified the format of message. Message must has the correct format, which has 4 parts of information and each part was separated by specific character “@”. So, first matter is limitation of message format must has 3 places of specific character “@”. Whenever system can find separator more than or less than 3 places in OTP message, mean that the wrong one was sending from trespassers.

The next convention of both sender and receiver is the sequence of information, which was separated by special character. First part of message, 6 characters, must represent the **cipher text of challenge message’s Date-Time**, which can be brought to calculate for backward key. Cipher text of Date-Time must be compared with header reference string. (String, which can be converted to array of 60 characters).

The second part of message must represent the **cipher text of Server String**, which was united with Date-Time as challenge message. The next part is information represent the **cipher text of User PIN**, which had ever retrieved from database and was buried in source code of OTP generator. The last part of message represents **the cipher text of Serial Number** of OTP generator, which was also included in application's source code and can be identified on relational database that which number is the activated serial number of user.

After that, the matter of calculation for backward key would be cracked. First 6 alphabets of cipher message represent value of day, month, year, hour, minute and second. Each alphabet would be compared with header reference, array of characters that was transformed by string, and get the index of array. 6 values would be converted in term of Date-Time and converted again to be millisecond since midnight on January 1, 1970. Then with the same algorithm as encryption process the final result of algorithm would be **Backward Key** of decryption. Such as "GBFB6W", which can be converted to 06/02/2005 01:58:22, is the main factor of result of calculation, 16.

Backward key is the main point of next step. Reference string, same string as reference in encryption process (sREF), would be used for transforming cipher message. Each alphabet of cipher text of Server String would be compared with reference string. Whenever cipher alphabet has same value as one of character in array, system would get the array index of that character then consider on previous index by reverse counting with a backward key value. So, system could identify the character on the right indication.

Such as "*DCDqAExqrqIys" is a cipher text of Server String, if backward key is 16, 28 is an index of first alphabet '*' base on reference string sREF then a result of reverse counting from 28 with 16 is 12. Index of 'M' is 12 (base on sREF) so first plain alphabet is 'M'. With the same algorithm, the consolidation of plain characters is "Monoalphabatic".

Other cipher information, User PIN and Serial Number, could be used with same algorithm to get the plain information. Such as "tuBD" is the cipher text of user PIN while "h[pjmnl" is the cipher text of Token Serial Number. With the accurate

algorithm, the consolidations of result are “demo” for user PIN and “1-93675” for Serial Number.

After all of plain message could be searched by system, profile information, User PIN and Active Serial Number, would be compared with user’s data, which were completely recorded in relational database since user registration process was successful. While Date-Time and Server String were be comparing with last challenge message of conducting user. In case of one of information form OTP mismatch with the information on database, the access was denied.

One Time Password (will be expired within appropriated interval)

Whenever visitor accepted challenge message, its allotted span of life must be counted down simultaneously. This conceptual was considered to setting up in this approach. The authentication approach of this study was originated base on functional of authentication device. The one of beneficial qualifications is any password could not be used twice to accessing to the system.

An OTP (one-time password) system generates a series of passwords that are used to log on to a specific system. Once one of the passwords is used, it cannot be used again. The logon system will always expect a new one-time password at the next logon. This is done by expiration of challenge message. Therefore, the possibility of replay attacks is eliminated.

In fact, system can investigate OTP message by cracking the Server String. Server String, apart of challenge message, was generating uniquely by server site whenever visitor request for challenge message. It was combined with time stamp and was sent to visitor. This kind of message would be included in One-Time cipher message (Date-time, Server String, User PIN and Active Serial Number). In point of view of database, challenge message record was inserted relate with profile of visitor whenever early generated challenge message was accepted by visitor. Then status of early-accepted challenge message would be set to ‘Active’. While other records must have status as ‘Used’ for completely used challenge message and ‘Expired’ for uncompleted challenge message (visitor cannot response this message within appropriated interval). In case of old Server String, with status as ‘Used’ or ‘Expired’,

was sent again to request for grant access. System could investigate that used OTP wasn’t enable for allowing to access to conduct special service.

Additional, not only password which was generated by authentication device cannot be used again, but also generator device can be set the configuration for timing to change dynamic message. 60 seconds might be set as a value of interval that frequently change consequential message. To apply this conceptual to appropriated authentication approach, active challenge message must be set expiration. Actually interval would not be set directly on challenge message record. The current date-time, when system was triggered for OTP verification, was the one factor of validation for adapted interval. Subtraction of millisecond of current time and millisecond of date-time header, apart of OTP was adapted from timestamp when challenge message was generating, brought to compare with the appropriated interval, which was buried in decryption algorithm. Such as 60 seconds is the right interval for operation since generating challenge message to entering OTP to access to system. Whenever decryption process was triggered result of current time subtract with the time of generating challenge message must not more than 60 seconds, if not this accessing was denied.

CHAPTER V
DISCUSSION

This section describe about consideration on alternative authentication in this study, the objective of this approach is the specific purpose for granting connection to user who submit one-time password to authentication system. This process will be used for grant access in role of special member who can conduct the extra service on web site.

Gathering resolution by the assessable testing, the volunteers could explain their satisfaction in the alternated security approach on web-based service in term of more reliability was increased while the doubtfulness in process of self-registration, token installation and log on by OTP also was considering after simply evaluation by volunteers.

Conclusion by the volunteers can be advocated the following summary about advantage and disadvantage of the alternative approach in this study.

There are some advantages of this authentication concept.

"Wipe out replay attack"

Due to the accustomed authentication by static password, it seem an anxious about password interception had been remained in user's mind. Whenever invader can know user's login id and its matched password, this constant information could be used stealthily for illegal accessing to the system all the time. The concept of One Time Password can remove this kind of anxiety. Interceptor cannot use the constant password to get authentication. Although they can find the way to get the latest OTP, the used OTP is just the garbage information of this approach.

“More concise authentication for special transaction”

In case of demonstrative web service that represented the approach of this study, the extra level of observable service could be separated ultimately from the normal level service. The distinguishable feature is the way to request for authority to access to system for that service. For normal level of service users can be authentic by simple login id and constant password, while more concise process will be require whenever user would like to access to extra level of service. By this significant notification, whoever want to conduct the extra service can be ensured that the specific service already had more complicated of authentication to identify the right person.

“Two factors authentication concept was used in this approach”

Due to the one conspicuous concept in device based authentication- two factors authentication, such as authentication on ATM, technology of banking business, accessing needs both PIN number as well as the magnetic strip card. Even if someone attains the PIN number, the card is also needed for access. In addition, if the card is lost or stolen it cannot be used without the PIN.

Conclusion of the tester volunteers, it showed that all users could understand in two substantial factors of this concept.

- Something user has - OTP members must install the matched token application in their client machine.

- Something user knows - User PIN, which was memorized by owner, must be input to activate the token client application and also this PIN would be included as apart of One Time Password.

“Communication between the right persons”

The OTP members, who gain extra access authority by using OTP approach, can be ensured that they were communicating with the right destination host while the authentication process could make sure in the satisfying level that only extra member, who owned the client application, can submit the set of individual information as One Time Password to the system.

In term of users, Challenge-Response process is the indication of reliable communication with the right host. Only host was implicated with specific authentication process could generate the right format of challenge message, which would be validated by client process of generating OTP.

While the server side can be ensured in the satisfying level that the right member submit the appropriated OTP from client side. Because of OTP is the message includes the individual information, which must match with only one registered user profile (This profile information also was included the last owned challenge message).

There are some limitations have to concern in this authentication concept.

Although there are so many security concepts support together in this approach for the main purpose of increasing the reliability of authentication on web-based service, a little hole of any concept could expand the overall weakness of the authentication system.

“Breakable demonstrative encode&decode algorithm”

Monoalphabetic cipher or simple substitution is the simple way of encrypting text. Especially cryptanalysis of the Caesar Cipher can explain the weakness that this kind of algorithm is so easy to crack. The following sample can represent the weakness, which due to frequency distribution.

In English, some letters are used more frequently than others.

For example ‘THIS IS A CAT’

Letter ‘A’ occurs frequently in English while the letters ‘I’ and ‘S’ also occur as a companion frequently in English sentence.

The message has actually been enciphered with a 27-symbol alphabet. (A through Z and space for separator between words) So, encryption by Caesar Cipher is ‘WKLV LV D FDW’

By the simple clue, in English there are relatively few small words, which combine 2 characters, frequency used. Such as ‘am’, ‘is’, ‘to’, ‘be’, ‘he’, ‘we’ and so on. In this cipher message, it is so easy to substitute ‘LV’ with ‘IS’ so invader can know ‘WK**IS IS** D FDW’. Even if they ‘re still not ensured to find the forward key,

they can continue guess by simple clue. After word ‘is’, there’re so many times that article ‘A’, ‘AN’ or ‘THE’ is follow intransitive verb. So invader can know ‘WK**IS IS A** F**A**W’. Whenever the attacks can guess the forward key, they can crack this cipher message accurately.

Although encrypting algorithm in this study was improved to be more complex than Caesar Cipher by adding the transmutable shift key, this algorithm was still categorized to Monoalphabetic cipher.

“Still require constant password”

With demonstrative processes in this study, there are some constitutions that still need the normal log on by login id and constant password before operating that process. For example:

- Require accessing to system as normal user before conducting process to get challenge massage to put into token application.
- Process of distributing password generator to the member. System has to identify the right user by static password before granting access to download token application

“The chance to intercept application installer”

Even using User PIN to activate client application support the concept of Two Factors Authentication, the approach of this study still has the overlooked chance for trespassers to steal members’ installation of token application.

The basic authentication by constant password may be enough for verify the right member before let him/her get the installation from the private directory. But what is the affectation, if he/she remained that installer in his/her machine after installation was finished. Somebody, who can open that machine, also can access to installer file then copy it to another machine. Then what is the violence, if unexpected fellow already knew his/her User PIN and login id.

In case once interceptor can attain the email, which inform about special URL for access to download individual installation from his/her quota. The damage would

be occurred if invader could know the login id and constant password. And there is most violence in case of trespasser already knew the User PIN.

"Inconvenience of token succession process"

Succession of token in quota of member seem form the usefulness with concept of anytime-anywhere while the inconvenience of this process was evoking the trouble to user who would like to return to the used machine. Due to process of token succession, only latest version of token can be used for this authentication. Whenever user already downloaded new token to new machine the previous machine cannot be used until it keep the active version of token.

Because of anytime-anywhere is the one interested concept of web-based service, so the productive authentication should support this concept also. But there are confusions in this demonstrative concept of succession. It seem that users had to be reminded all the time when they tried to use the client token. They had to control the client application on their current machine to always have the correct version, which matched with the profile information on the server side.

Whenever use moved to another machine, new version of token would be activated and it evolved inactive status for the previous version on the old machine. This is the normally concept of succession in this demonstrative. The confusion would be occurred after that. In case of user would like to return to the previous machine (anytime-anywhere), user had to control more than one version of token. User had the option to remain the latest version of downloaded token by download it again to previous machine then uninstall the local version and starting install the new one. Alternative option, user could reactivate the previous version by access to the private web page and changed status of the local version to be active. It means the last version on new machine could not be used. Another option, user downloaded the unused version to a previous machine and uninstall the local token then install the new one. This option conducts the result that both previous token versions on both machines could not be used.

“Inconvenience of generating token quota by administrator”

Complexity of the administration process is the one concern of this supposed web-based service. Disadvantage was shown in this demonstration that participation of administration process frequently depend on the manual operation of web administrator. Especially the processes of generating quota of token belong to each member.

Administrative web pages were provided with the friendly interface to support administrator about managing the account profiles that include the token quota generating. Although system in this study would be generated java source code (filename.java) automatically whenever new record of token was generated, administrator still manually compile that source code to be the class files and use the specific tool for collecting them as installer.

Specific tool (InstallAnywhere 5.5) was chosen for transforming all of involved class files, with class extension (filename.class), to be the executed file, with exe extension (filename.exe).

Since compiling java source code to be class files until executed file (.exe) was produced in specific tool and associating this physical execution to member profile, every step were operated manually by web administrator.

CHAPTER VI
CONCLUSION AND RECOMMENDATION

6.1 Conclusion

This section describe about conclusion of this study. Inspiration of this study is the interested functions of authentication system by device-based dynamic password generating. The main purpose of this study is analysis for feasibility to combine the process of authentication by One-Time Password into the web-based service, which prepared extra processes for the specific members.

By gathering information and setting demonstrative case study, consequence of the study showed the possibility in the satisfaction level to apply this authentication concept to web-based service. Although authentication by using constant password on web-based service cannot be avoided, One-Time Password can increase the reliability for customer to using it in specific authentication for more beneficial transaction.

However this study just approach the idea to find out the solution for using One-Time Password in web-based authentication. So many parts of this concept have to be improved for set up producible authentication.

6.2 Recommendation

There were some recommendations about future study that involve the opportunity of applying One-Time Password technology to the authentication system of web-based service that was shown below.

- Using more complicated encode&decode algorithm
- Technology for generating token client application
- Message should be distributed via appropriated channel
- Continue study for tendency to apply on handheld devices

Using more complicated encode&decode algorithm

Caesar's cipher algorithm was chosen to modify the experimental algorithm in this study. However there are so many algorithms that has more complex concept than this demonstrative algorithm. Other kinds of Monoalphabetib Substitution algorithm may be able to considered in order to replace Caesar's cipher such as permutation-reordering the elements of series.

If "a_1, a_2, a_3, … a_k" are the letters of the plaintext alphabet and π is permutation of the numbers 1,2,3… ,k in a monoalphabetic substitution each $Ci=a_{\pi(P_i)}$

For example:

If reference string in format of "a_1,a_2,a_3, … a_k" is "A,B,C,D,E,F,G,H,I,J"

And permutation is π = 1,3,5,7,9,2,4,6,8,10 and $\pi(a_4) = 7$

Such as plain text is 'F', $p_{i=} a_6$, so $\pi(a_6) = 2$

So cipher text is a_2 = 'B'

There are many interested algorithms that can be candidate to choose for modify its algorithm relate to this authentication concept. (Such as RSA algorithm) Finding out the appropriated algorithm then apply to OTP concept is the one more suggestion to extend this productivity.

Technology for generating token client application

Although many components of web service site were created with java technology, it does not mean that distributed client application must be limited by generating with java only.

In another word, the java technology was selected to provide the decryption process on this demonstrative server site. But there is no limitation to build the encryption process on client application by java only.

It is interested that appropriated algorithm should be included in source code as native method, which can be registered on client side (such as DLL file) and this technology can be considered to develop as the plug in which can be added on the web browser. In addition, it is possible to generate Visual Basic source code that already include necessary information and decryption algorithm then execute the token application in case of can be ensured that client side using Windows technology as operation system.

Message should be distributed via appropriated channel

There are some limitation in the communication process, which has apart of distributing information to users. Such as the response emails to inform user about the successful of OTP self-registration or information of URL, which can be represented the location of privacy login page, in the process of accessing to get the installation file.

Due to the transmission of personal information must be controlled by more secure process. While email channel may not secure enough for the real service, The appropriated communication process should be considered to improve the approach of this study. There is possibility to integrate the web-based service with the wireless communication. Whatever message will be able to distribute to the client users via short text message on mobile phone.

However to improve the communication process, the self-registration process should be modified so that the details of user profile can be included the necessary information for the appropriated communication channel

Continue study for tendency to apply on handheld devices

Although this study intends on web-based authentication that using computer-based technology, the authentication device is manuscript of generating dynamic password that was referred by the main concept of this approach.

Retracing to considering on original concept may be the right way to push up the alternative approach of this study. Considering on device concept what is interested to motivate this study. Due to after registration, the series of message on device screen always synchronize with information of that device on server so the

message can be used as the dynamic password in the authentication process. The specific device had the algorithm inside so that encodes the new message in every minute. And that wonder device is compact in order to can be carried easily. There are interested ideas to put that algorithm into the different device, which can be brought to anywhere and anytime.

Before submitting OTP to get authentication on web service, client users had to access to website with normal authority and get challenge message, which was generated from server side, and they had to activate the special application on their mobile or pocket PC by entering user PIN then put that challenge message into handheld device. Finally they can receive the time-based message, which was generated by algorithm inside mobile phone or pocket PC. The special authentication can be continuing. One time password was produced already.

REFERENCES

1.Charles P. Pfleeger. Security in computing.PP.226-236, 252-254, 387-392.

2.One-Time Password. http://www.linktionary.com/o/one_time_password.html (accessed 20 May 2003).

3.One-Time Password. http://www.swcp.com/~mccurley/cs.sandia.gov/health/node10.html (accessed 21 May 2003).

4.N. Haller (Bellcore), C. Metz (Kaman Sciences Corporation), P. Nesser (Nesser & Nesser Consulting), M. Straw (Bellcore).A One-Time Password System. February 1998. http://www.faqs.org/ftp/rfc/pdf/rfc2289.txt.pdf (accessed 14 June 2003).

5.A Two-Factor Universal Platform. http://www.darwinmag.com/read/whitepapers/110101_fipass/ authentication.html (accessed 25 June 2003).

6.The ABC's of Two-Factor Authentication. http://www.itsecurity.com/papers/rainbow2.htm (accessed 25 June 2003).

7.RSA SecurID. A ClearTrust Web Access Management Overview http://rsasecurity.com (accessed 6 July 2003).

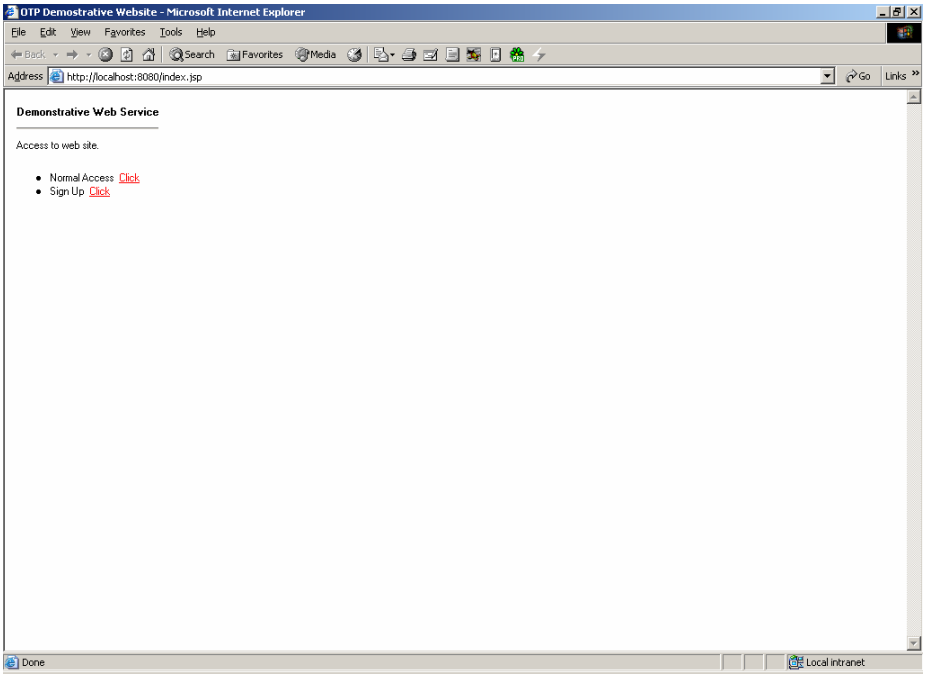
APPENDIX

APPENDIX

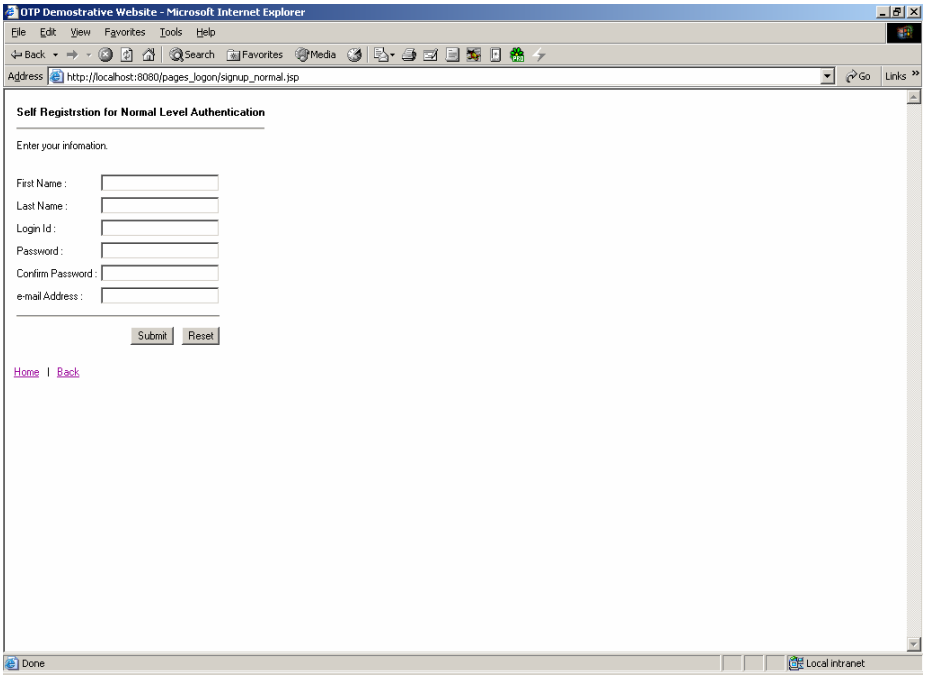
This section describe about details of how to use the demonstrative web site for every processes.

- Self-registration process for normal level member
- Logon process for normal level member
- Self-registration process for extra level member, OTP user
- Logon process for administrator
- Account management for normal level member by administrator
- Account management for extra level member - OTP user by administrator
- Process for generating new token related to OTP user by administrator
- Generate installer by InstallAnywhere 5.5
- Associate installer file to token profile record by administrator
- Activate and Download token to client machine
- Install token application
- Launch local token application
- Logon process for extra level member, OTP user

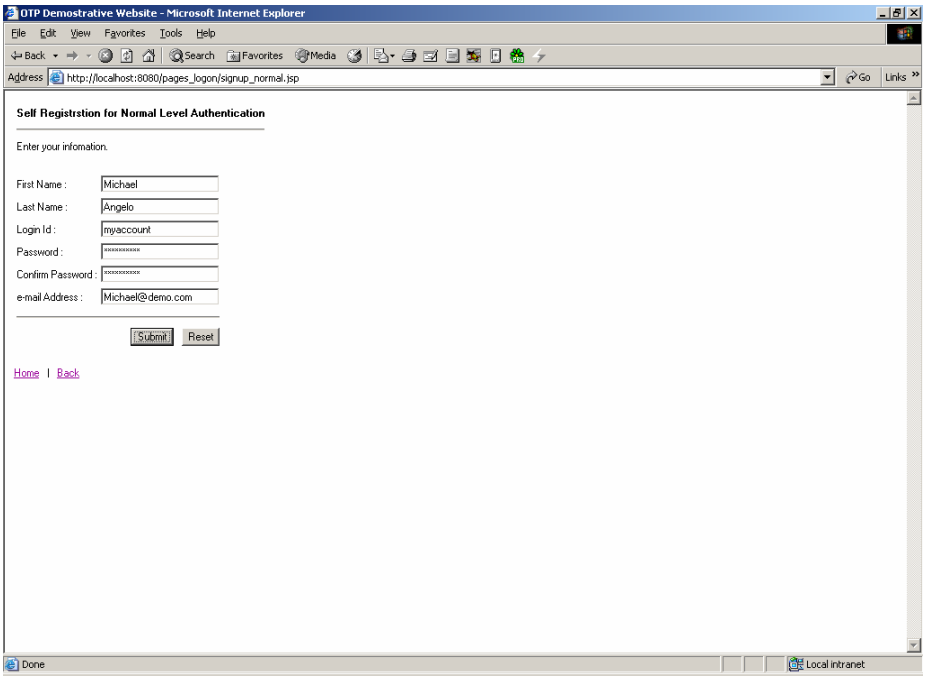
Self-registration process for normal level member



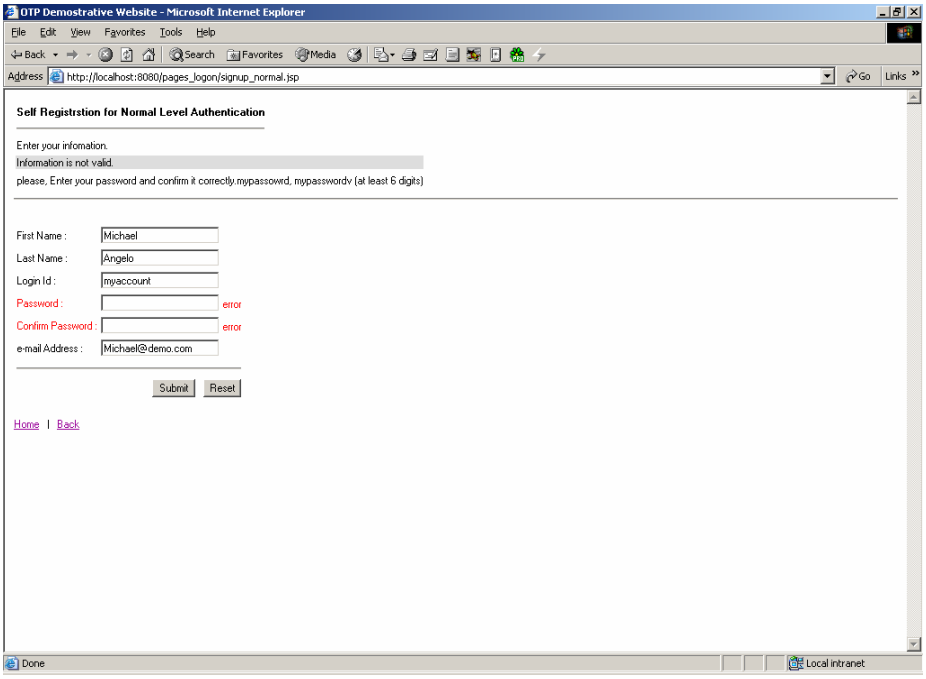
- Open main page of demonstrative web site by enter “http://localhost:8080/index.jsp” in Address bar of web browser
- Click on link “Click” of signup menu item



- http://localhost:8080/pages_logon/signup_normal.jsp is opened

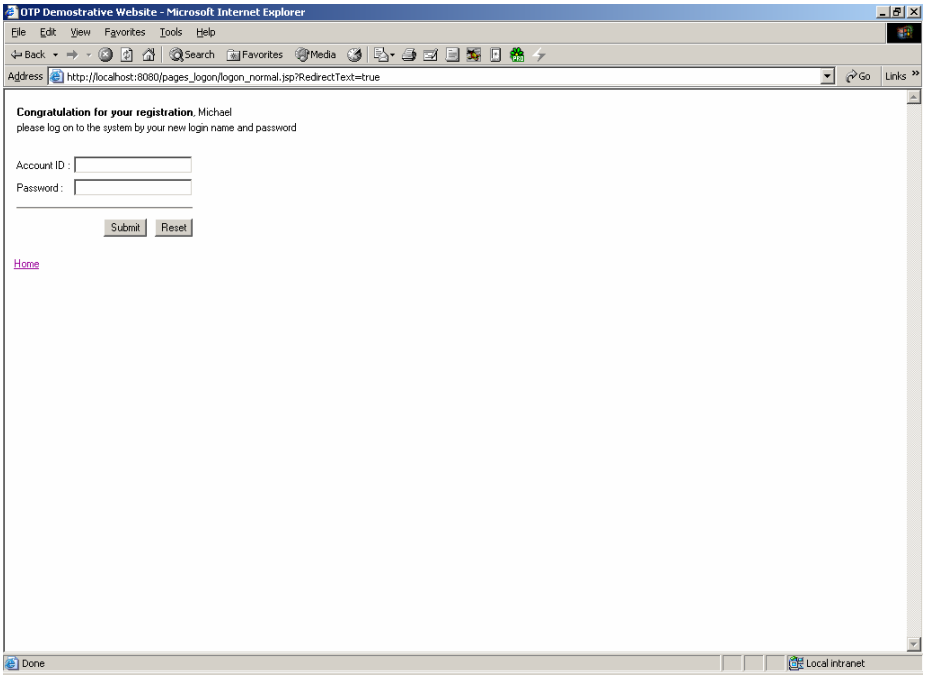


- Enter client user's information on Signup page
- The necessary information are First Name, Last Name, Login id, Password, Confirm Password and e-mail Address then submit information to the system by pressing Submit button



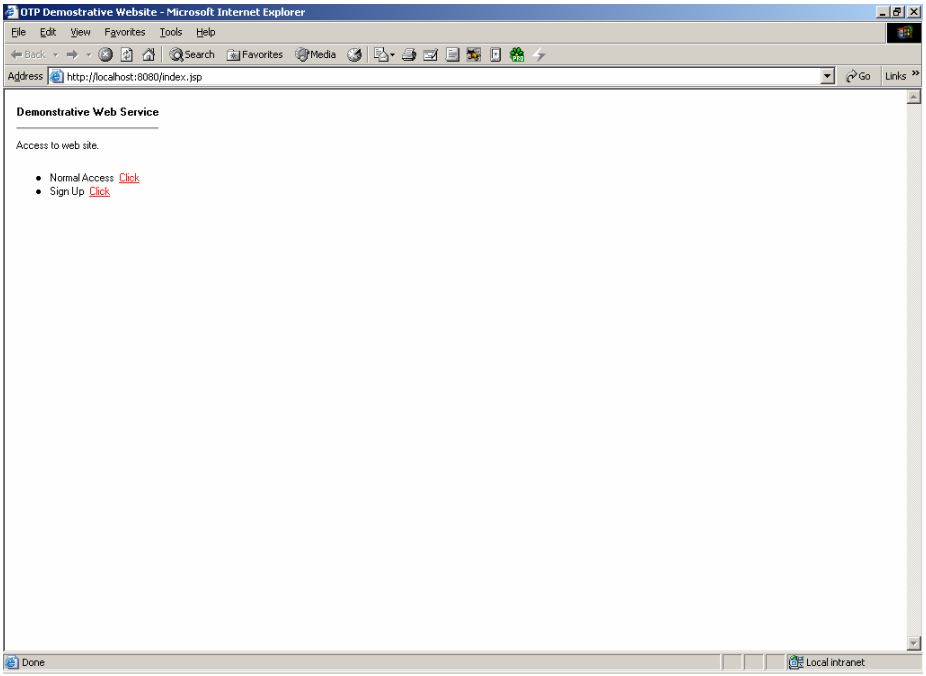
- In Case entered data is invalid the signup page will be reloaded with alert messages and waiting for valid information
- Client user has to reenter the right information and submit to system by

pressing on Submit button

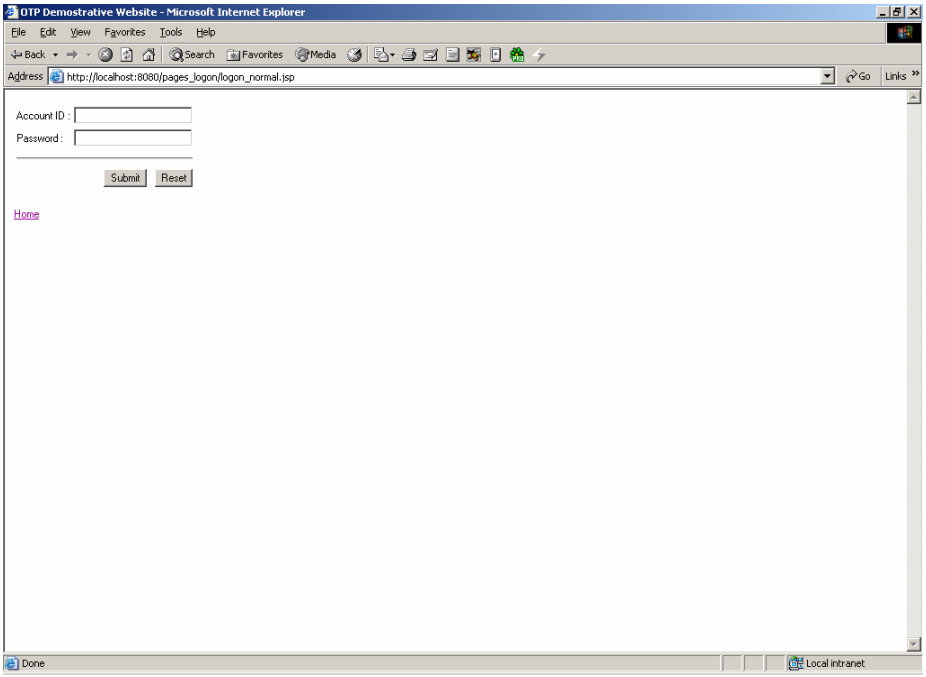


- Finally “http://localhost:8080/pages_logon/logon_normal.jsp” is opened to allow user access to the system.

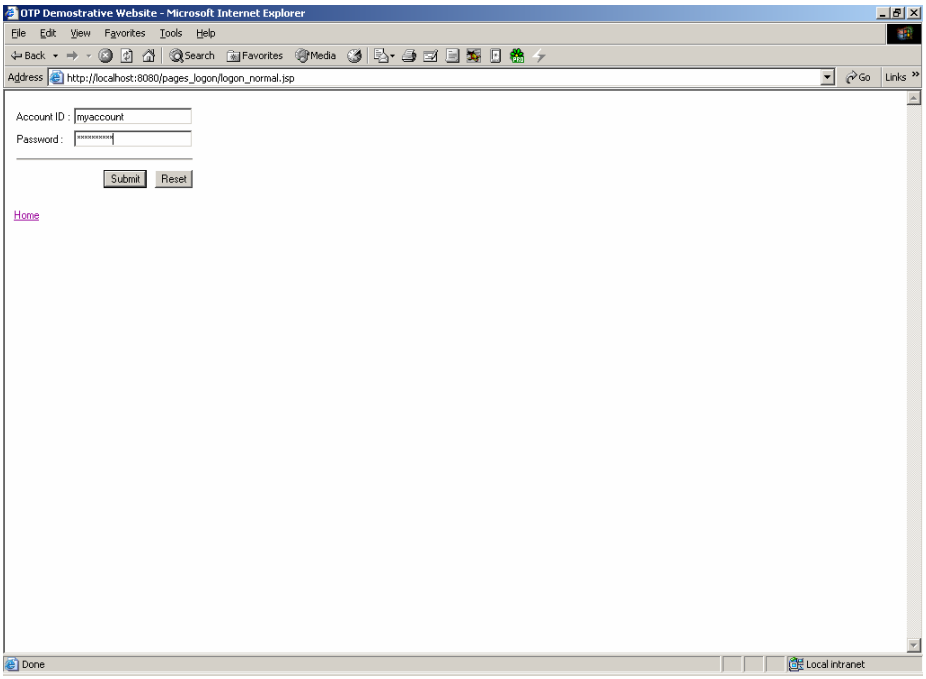
Logon process for normal member



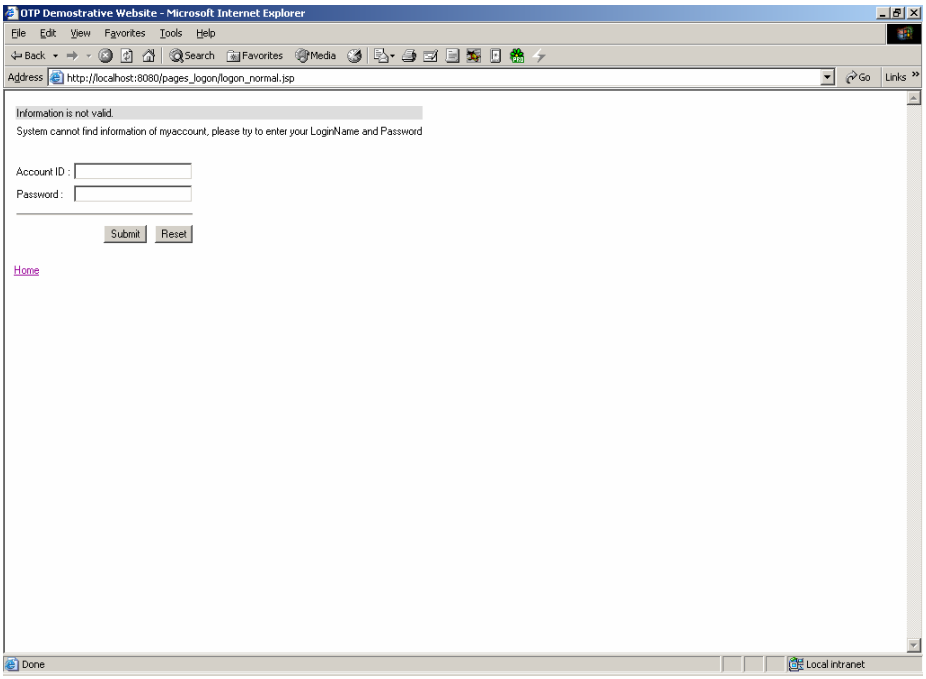
- Open main page of demonstrative web site by enter "http://localhost:8080/index.jsp" in Address bar of web browser
- Click on link "Click" of normal access menu item



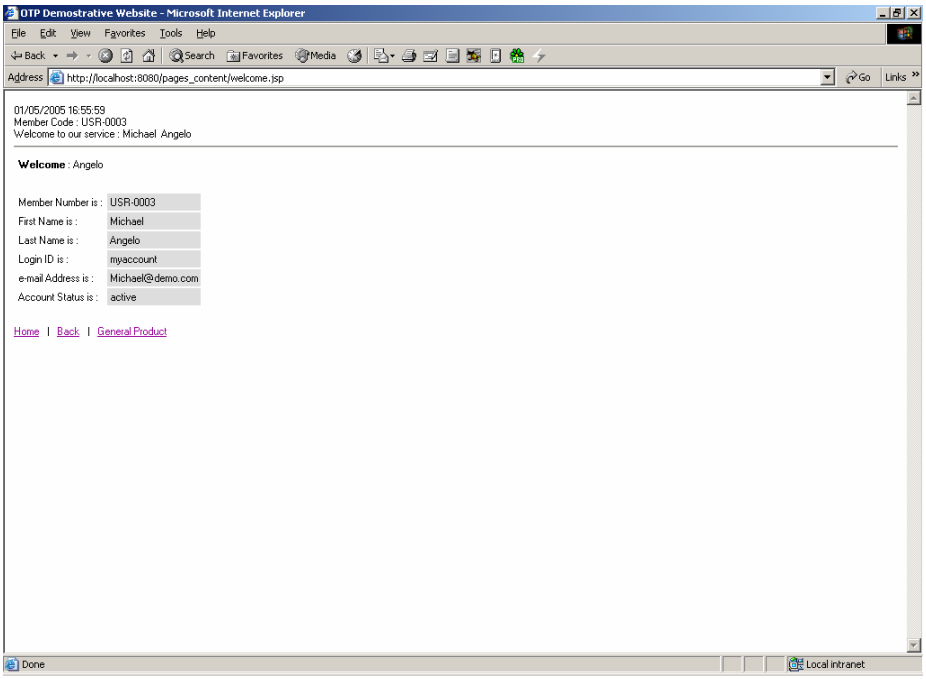
- http://localhost:8080/pages_logon/normal_logon.jsp is opened



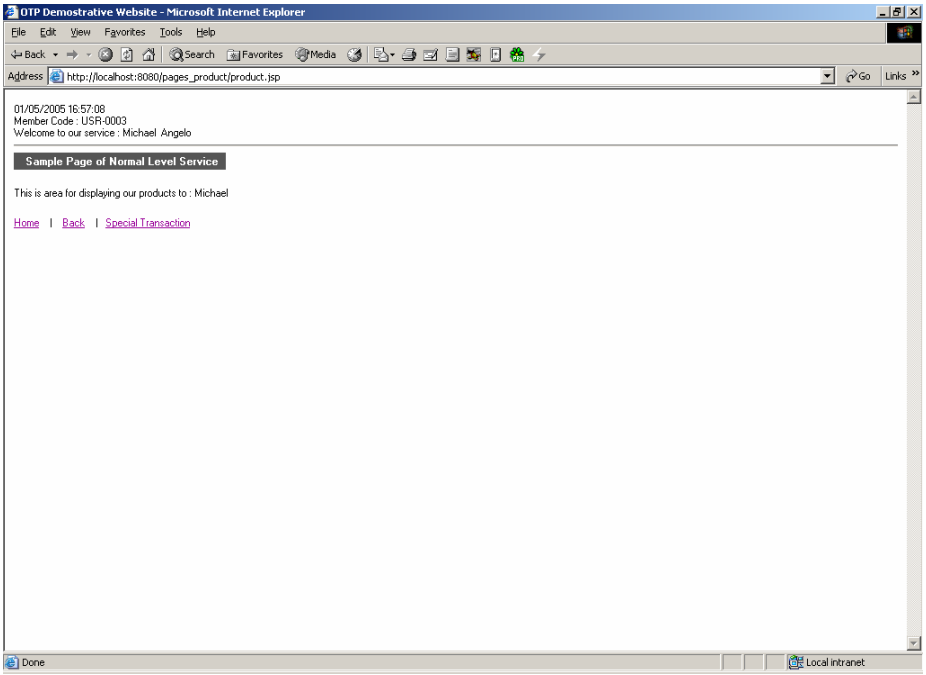
- Submit client user's Account ID and Password through normal logon page



- In Case entered data is invalid the logon page will be reloaded with alert messages and waiting for valid information
- Client user has to reenter the right information and submit to system by pressing on Submit button

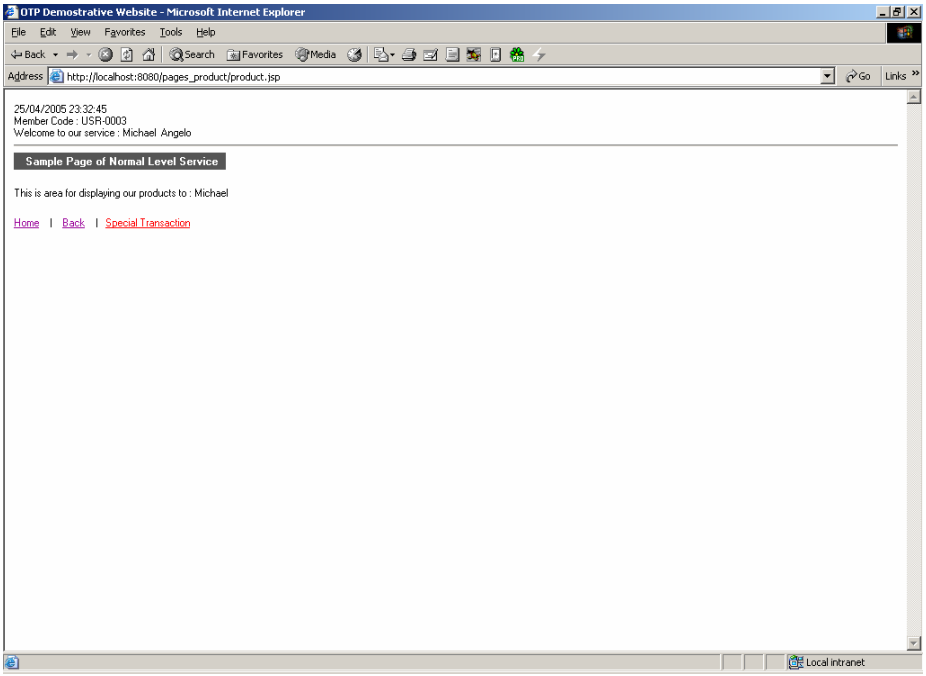


- These information, Account ID and Password, were verified before system allow client user access to introduction page, “http://locclhost:8080/pages_content/welcom.jsp”
- Then user can access to next page for conduct the service by click link “General Product”

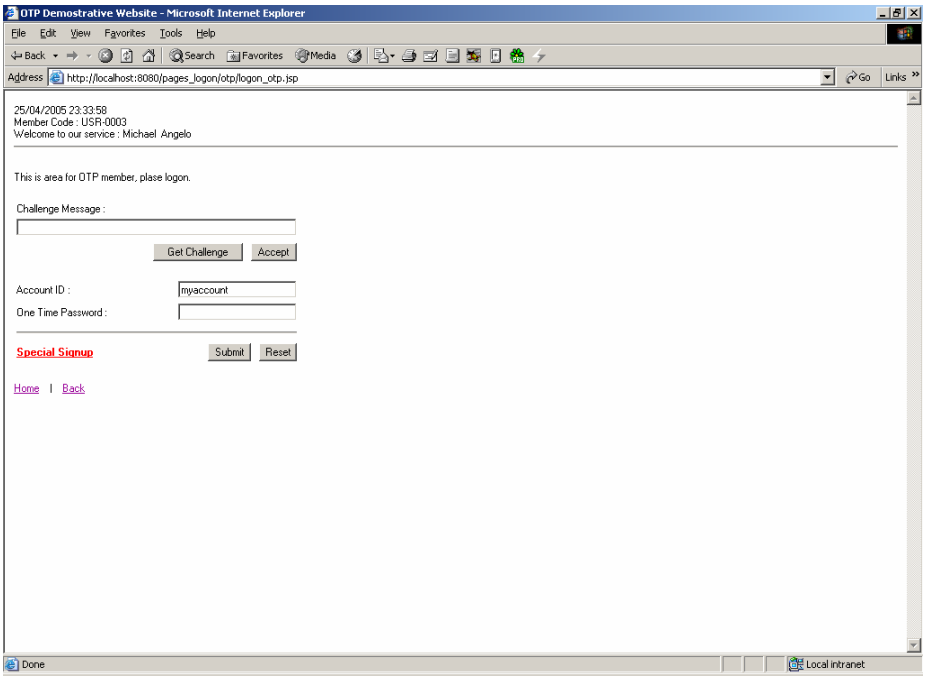


- The sample page of normal level service, http://localhost:8080/pages_product/product.jsp is opened

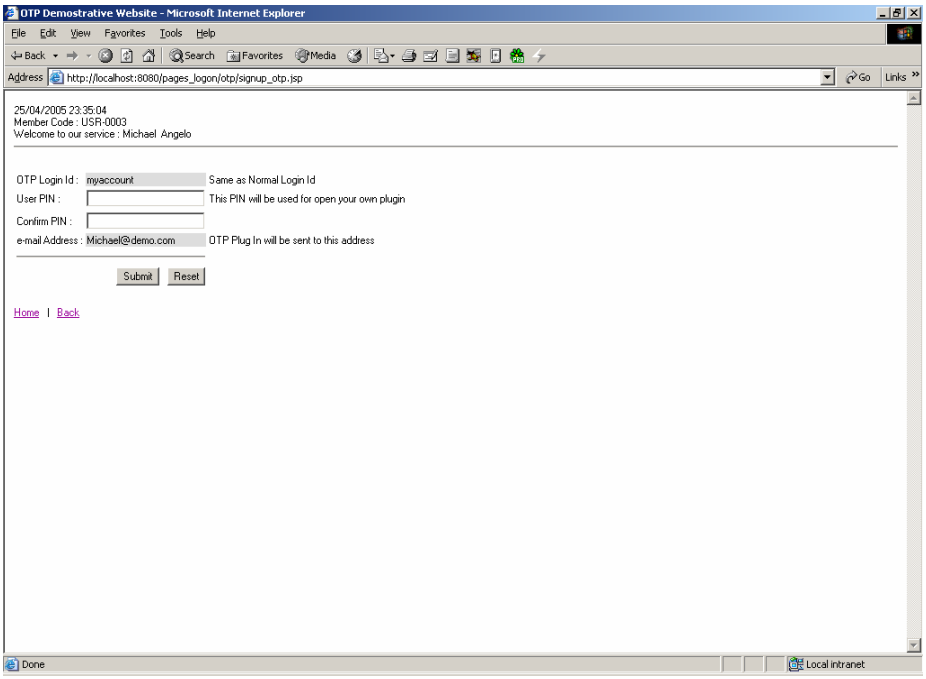
Self-registration process for extra level member, OTP user



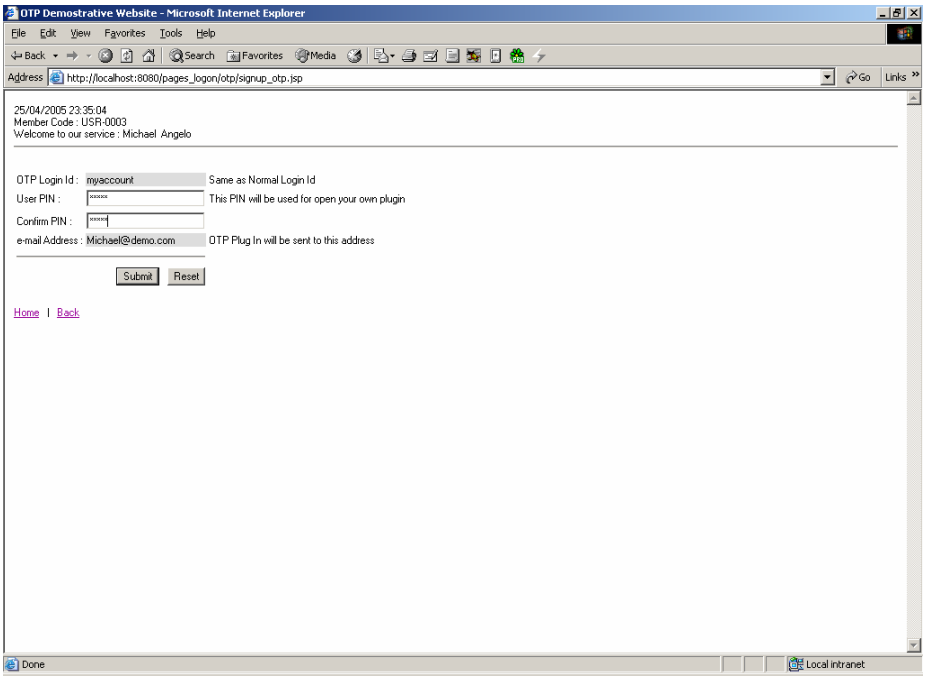
- User can access to special page for conduct the extra service by click on link "Special Transaction" on normal level service page



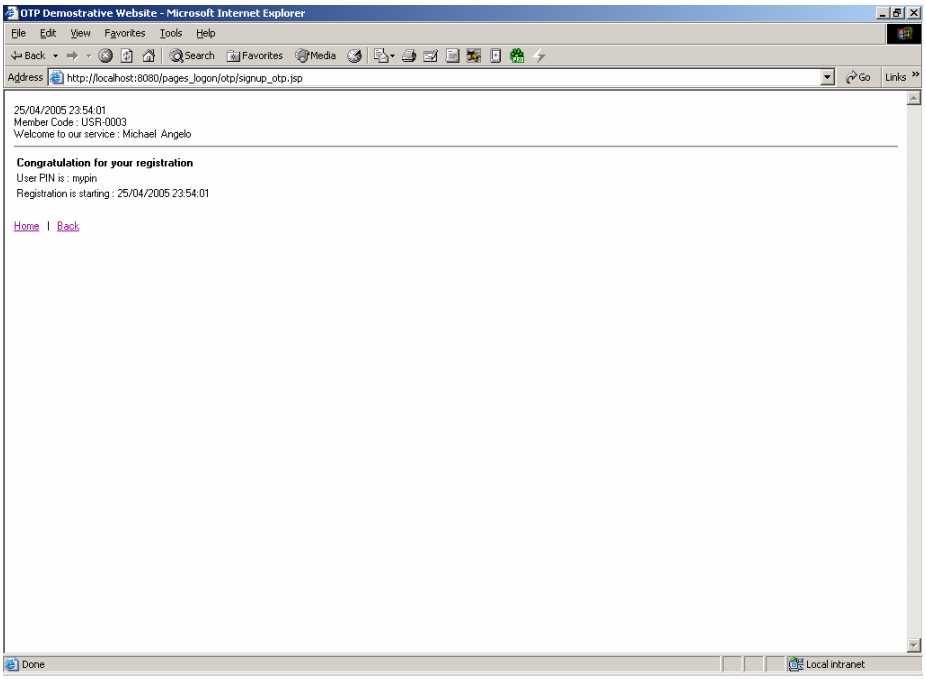
- The first page of extra level service, http://localhost:8080/pages_logon/otp/logon_otp.jsp is opened
- Client user, who has never registered to system for special authentication, has to request for OTP self-register page by click on link "Special Signup"



- http://localhost:8080/pages_logon/otp/signup_otp.jsp is opened

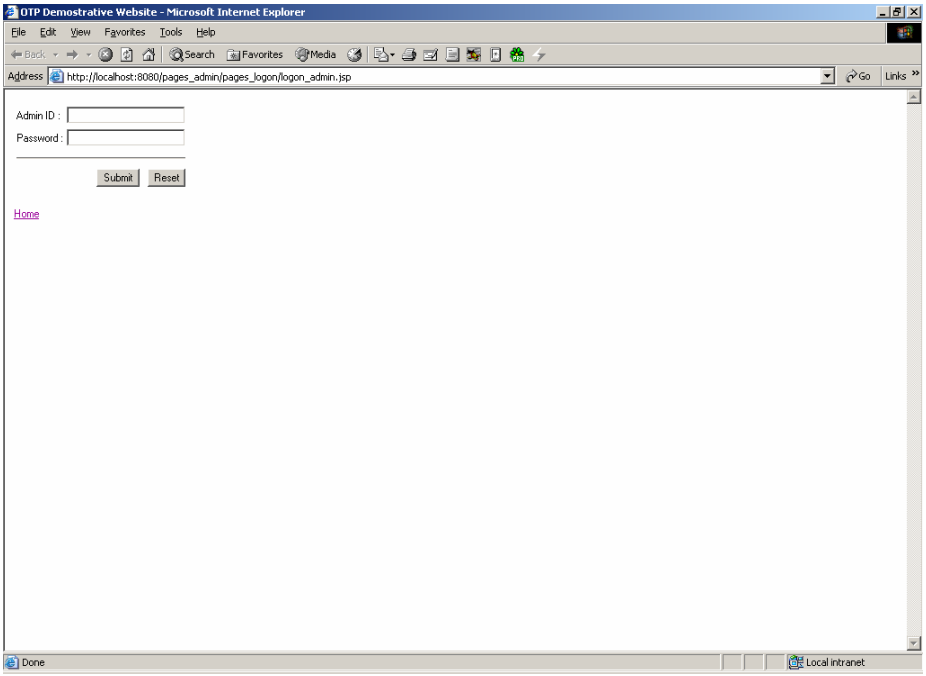


- Enter client user's information on OTP Signup page
- The necessary information is just User PIN and Confirm PIN then submit information to the system by pressing on Submit button

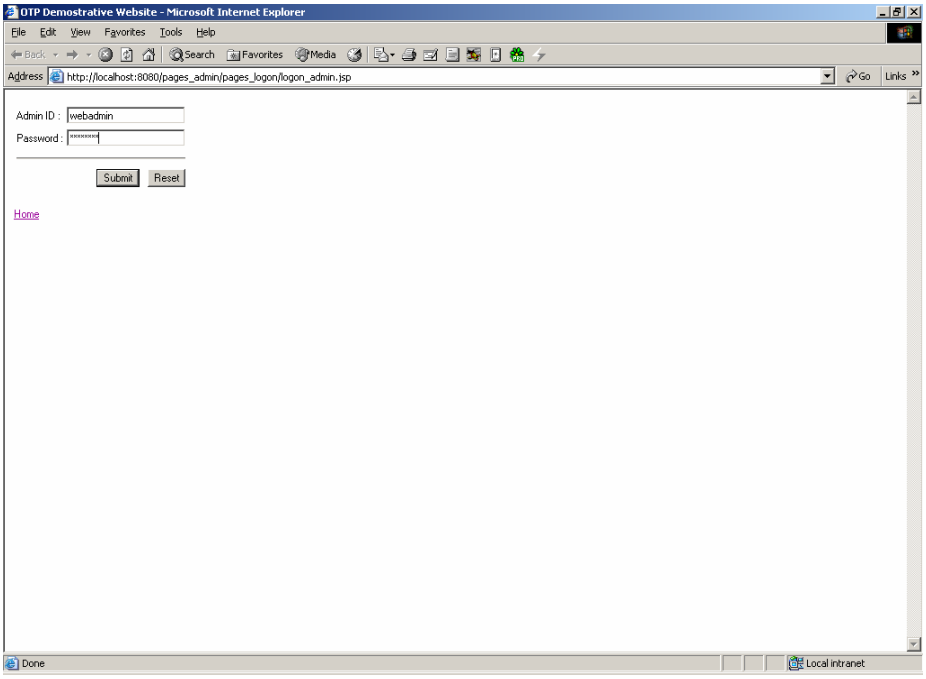


- Finally “http://localhost:8080/pages_logon/otp/signup_otp.jsp” is reloaded to display greeting sentence for new coming registered user and inform user about the time of the registration.

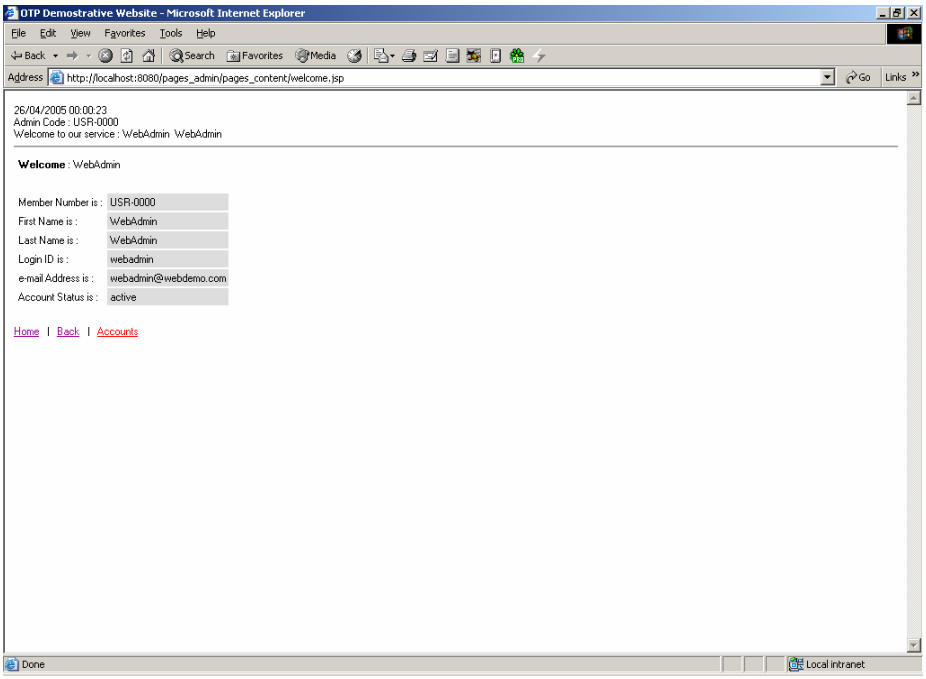
Logon process for Administrator



- Open main page of demonstrative web site by enter “http://localhost:8080/pages_admin/pages_logon/logon_admin.jsp” in Address bar of web browser

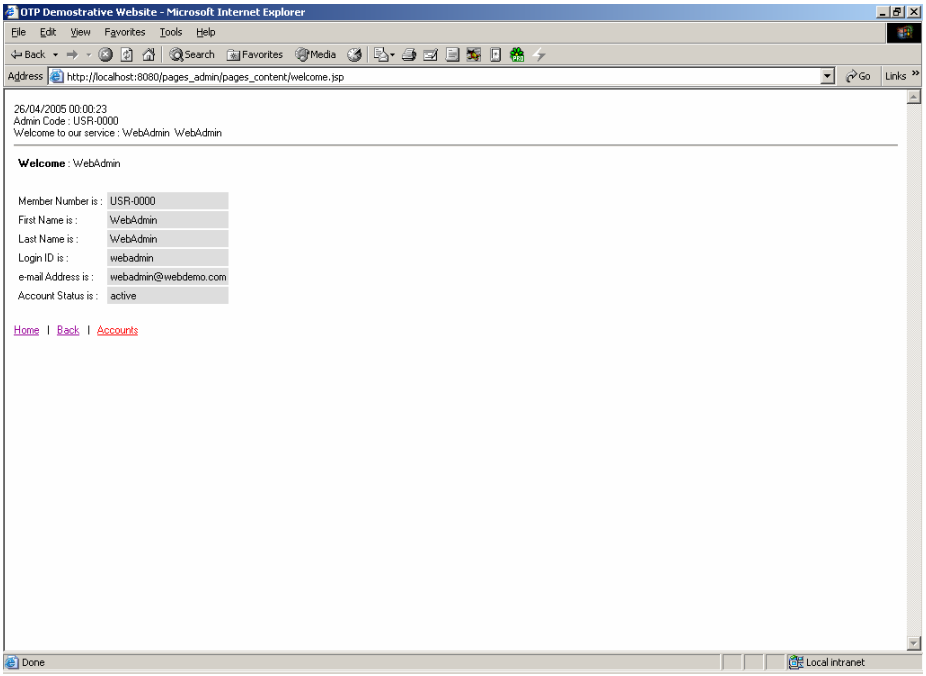


- Enter “webadmin” and “webadmin” order by Admin ID and Password then submit information to the system by pressing on Submit button

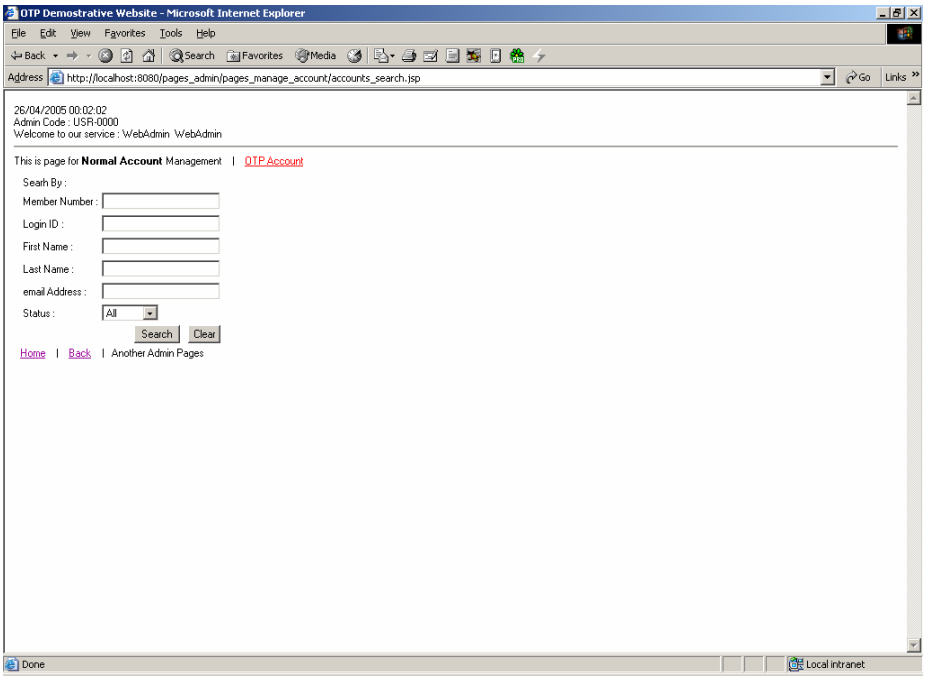


- These information, Admin ID and Password, were verified before system allow administrator access to introduction page, “http://locclhost:8080/pages_admin/pages_content/welcome.jsp”

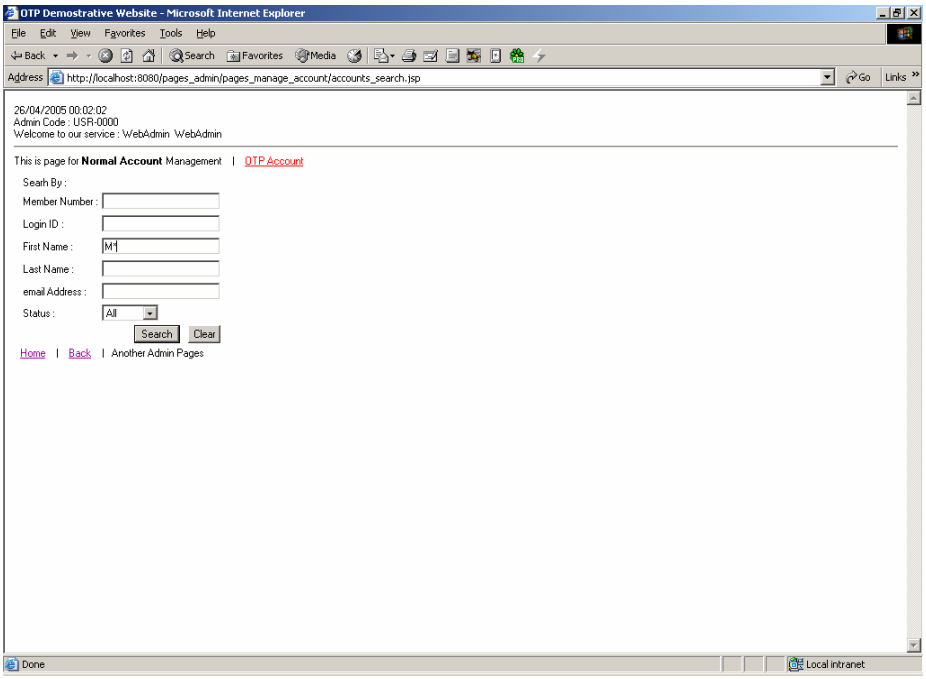
Account management for normal level member by administrator



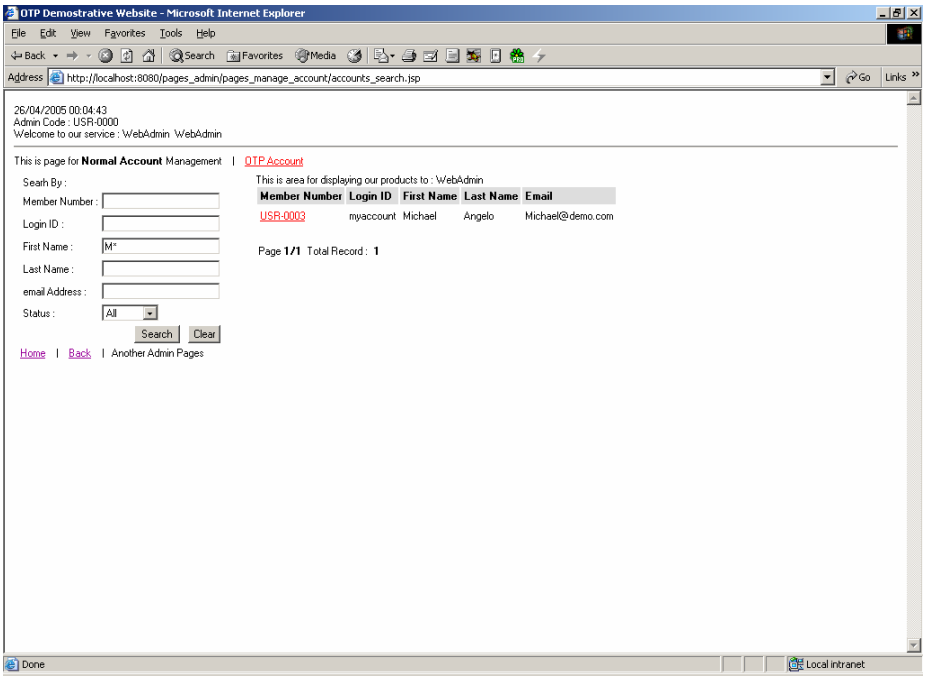
- Administrator can access to the page for perform account management by click on link "Account" on introduction page



- The administration page searching appropriated records of normal level members, "http://localhost:8080/pages_admin/pages_manage_account/accounts_search.jsp" is opened

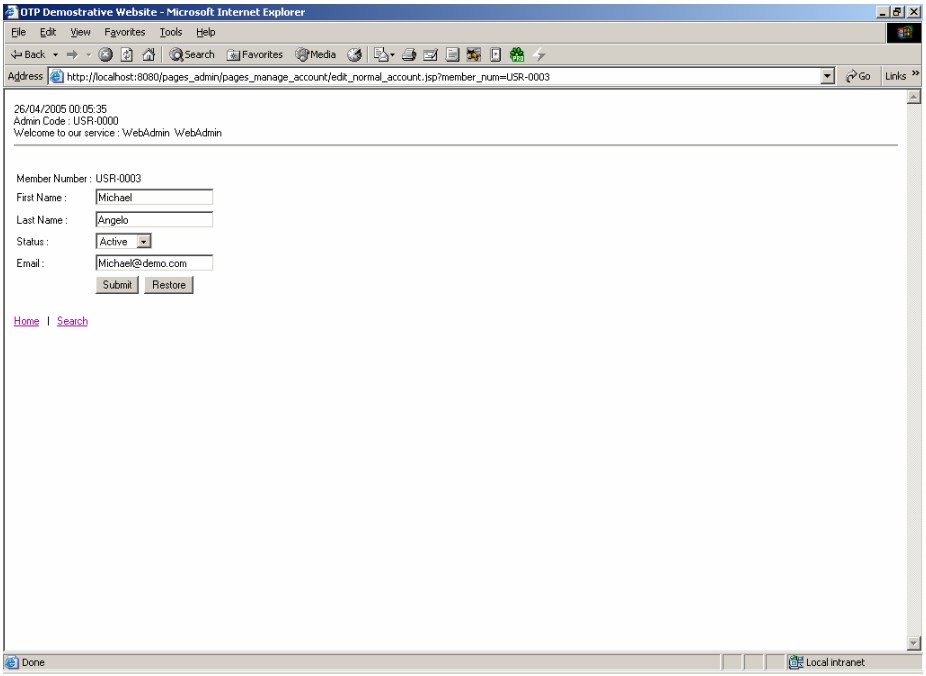


- Enter client user's information in order to search the appropriated records
- Such as "M*" and "All" order by First Name and Status then click on Search button



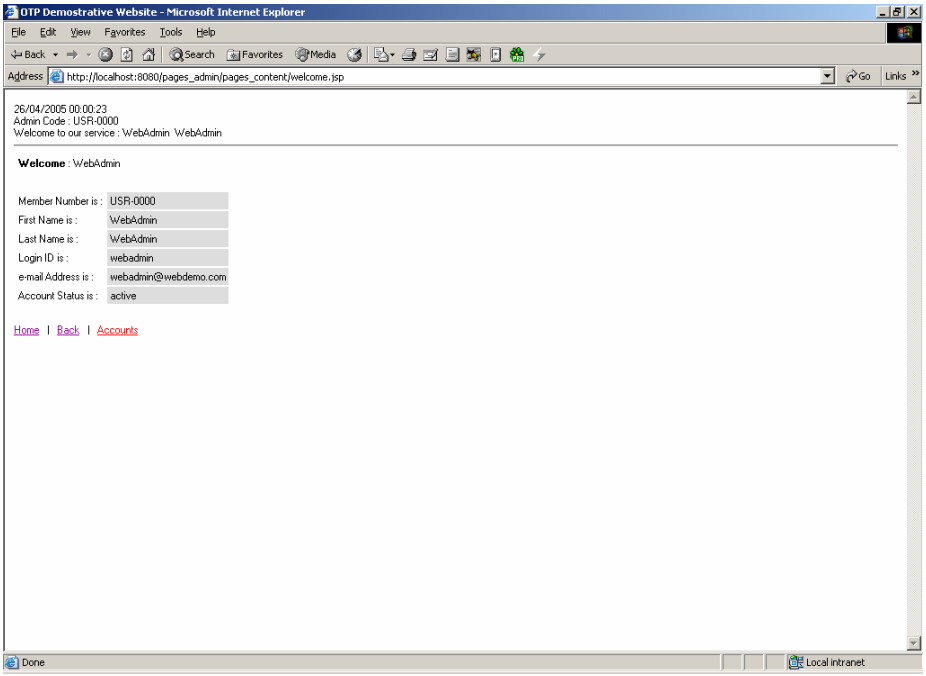
- This search page, "http://localhost:8080/pages_admin/pages_manage_account/accounts_search.jsp" was reloading then the search result is shown on the right side of page
- Administrator can click on link, which is displayed as member id, so that the

page for manage profile of this member will be loaded.

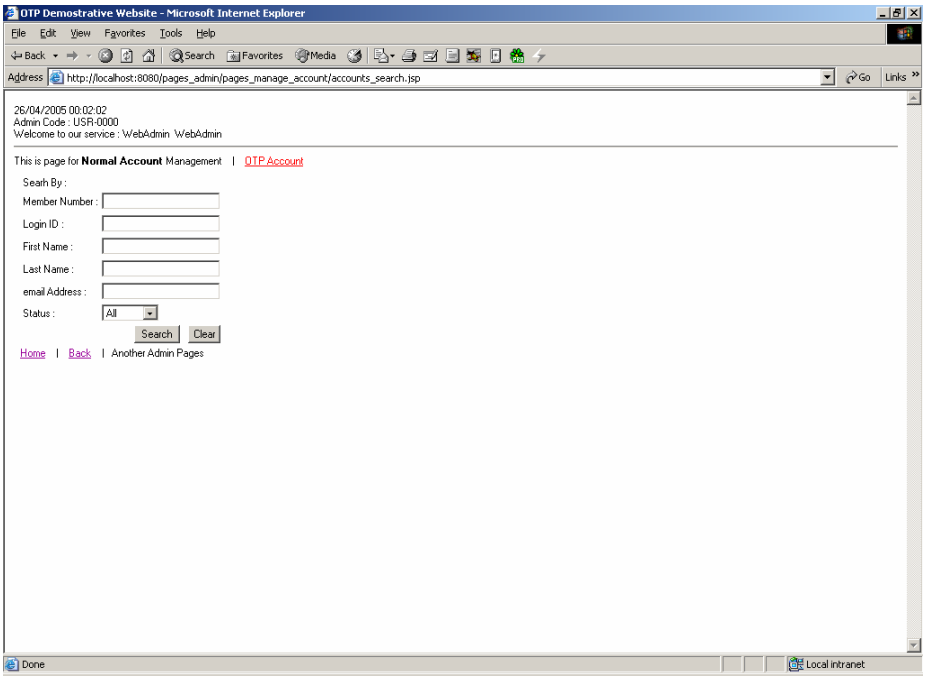


- The page for manage member's profile "http://localhost:8080/pages_admin/pages_manage_account/edit_normal_account.jsp" is opened
- Administrator can modify member's information (First Name, Last Name, Email Address and Status) then pressing on submit button to save this modification.

Account management for extra level member - OTP user by administrator

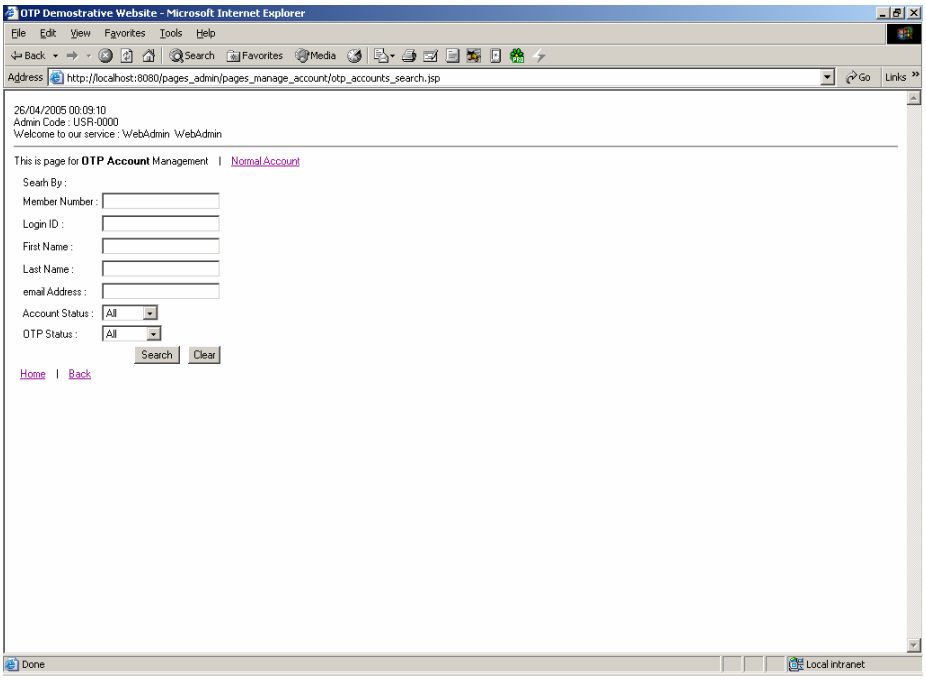


- Administrator can access to the page for perform account management by click on link “Account” on introduction page

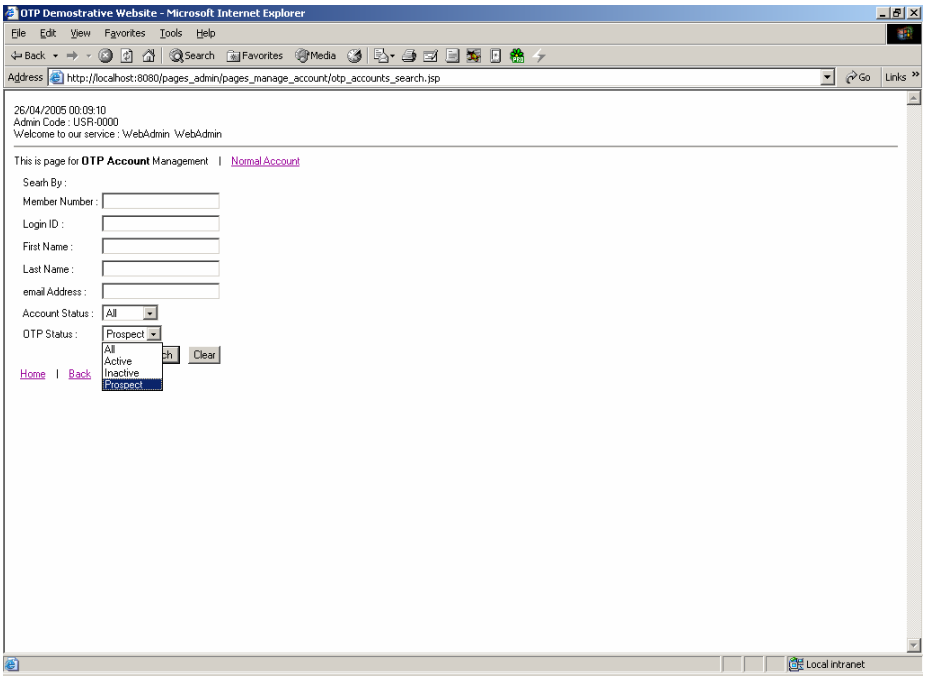


- The administration page searching appropriated records of normal level members, “http://localhost:8080/pages_admin/pages_manage_account/accounts_search.jsp” is opened

- Administrator can swap the page of normal account management to the page of OTP account management by click on link “OTP Account”

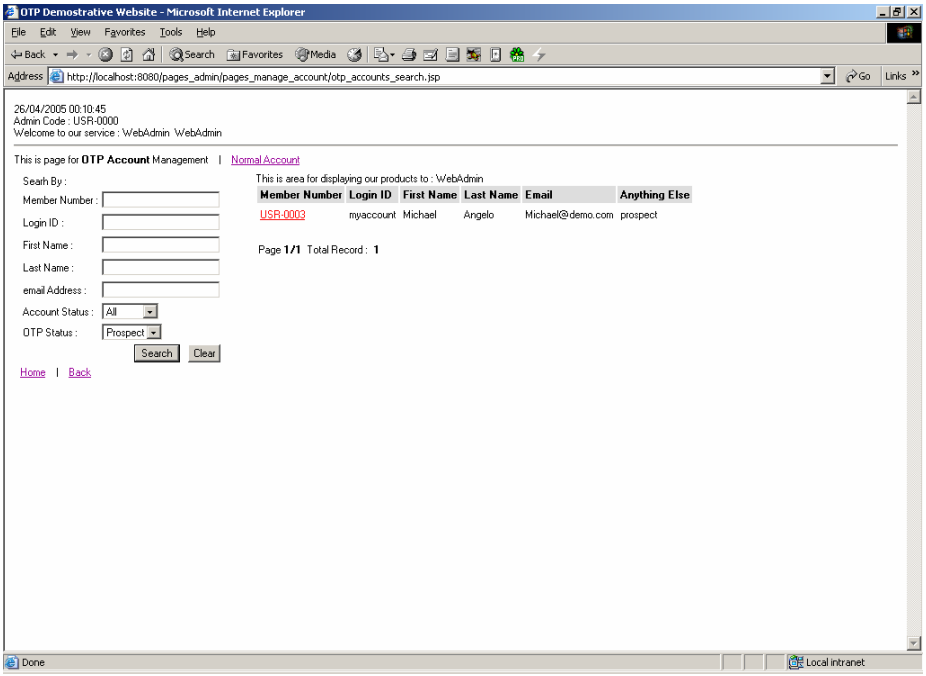


- The administration page searching appropriated records of normal level members, “http://localhost:8080/pages_admin/pages_manage_account/otp_accounts_search.jsp” is opened

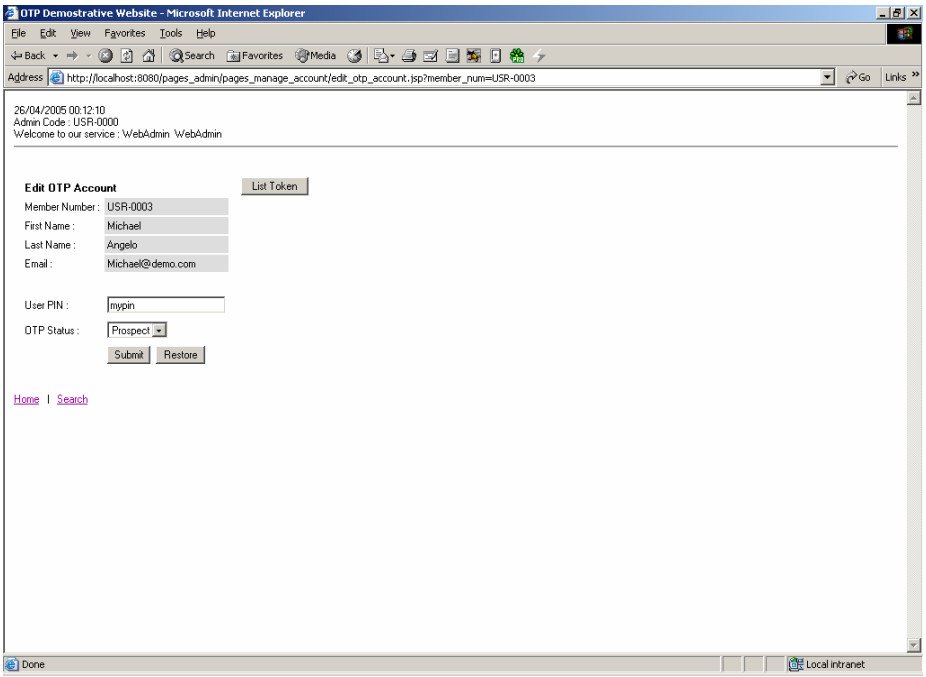


- Enter OTP client user’s information in order to search the appropriated records

- Such as “Prospect” for OTP Status then click on Search button



- This search page, “http://localhost:8080/pages_admin/pages_manage_account/otp_accounts_search.jsp” was reloading then the search result is shown on the right side of page
- Administrator can click on link, which is displayed as member id, so that the page for manage profile of this member will be loaded.

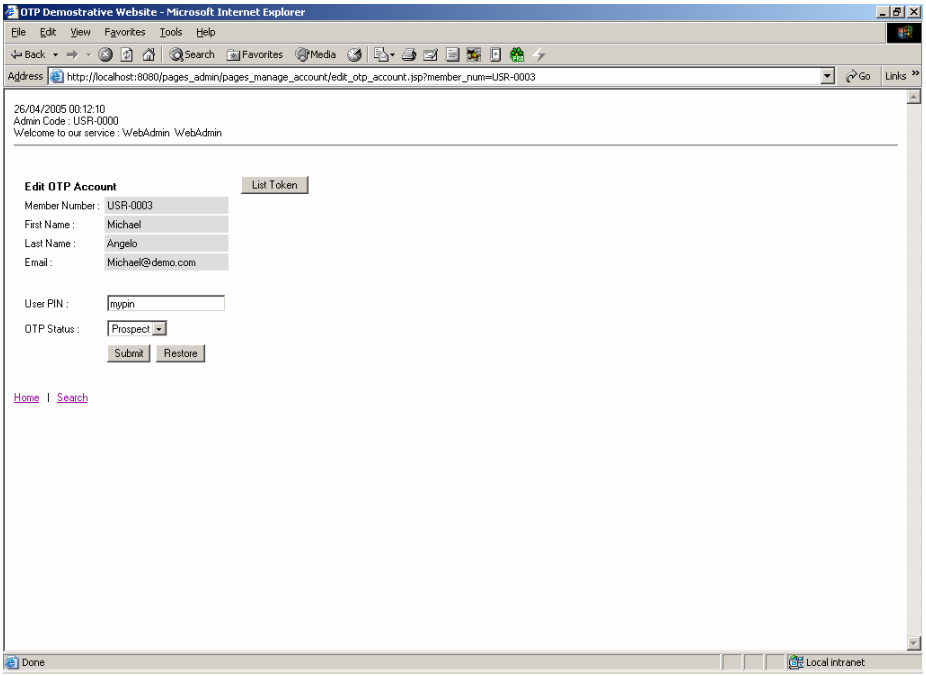


- The page for manage member’s profile

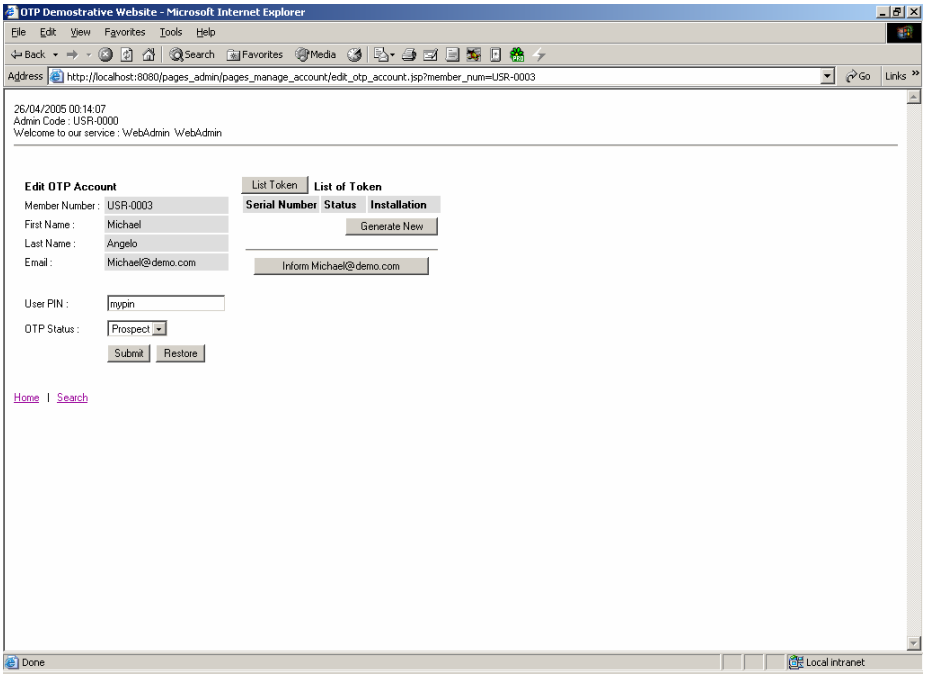
"http://localhost:8080/pages_admin/pages_manage_account/edit_otp_account.jsp" is opened

- Administrator can modify member's information (just User PIN and OTP Status) then pressing on submit button to save this modification.

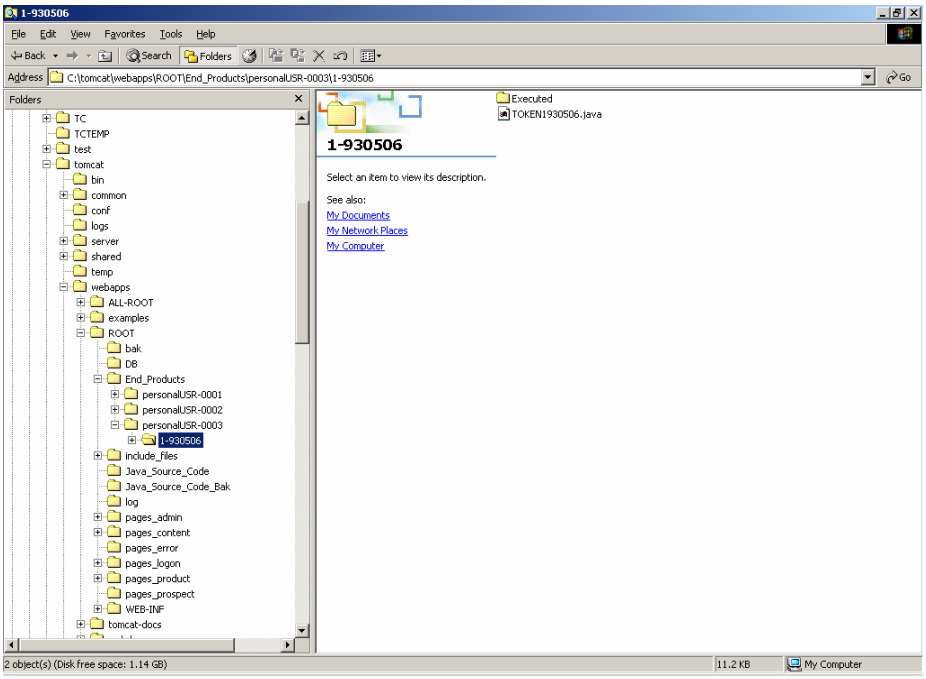
Process for generating new token related to OTP user by administrator



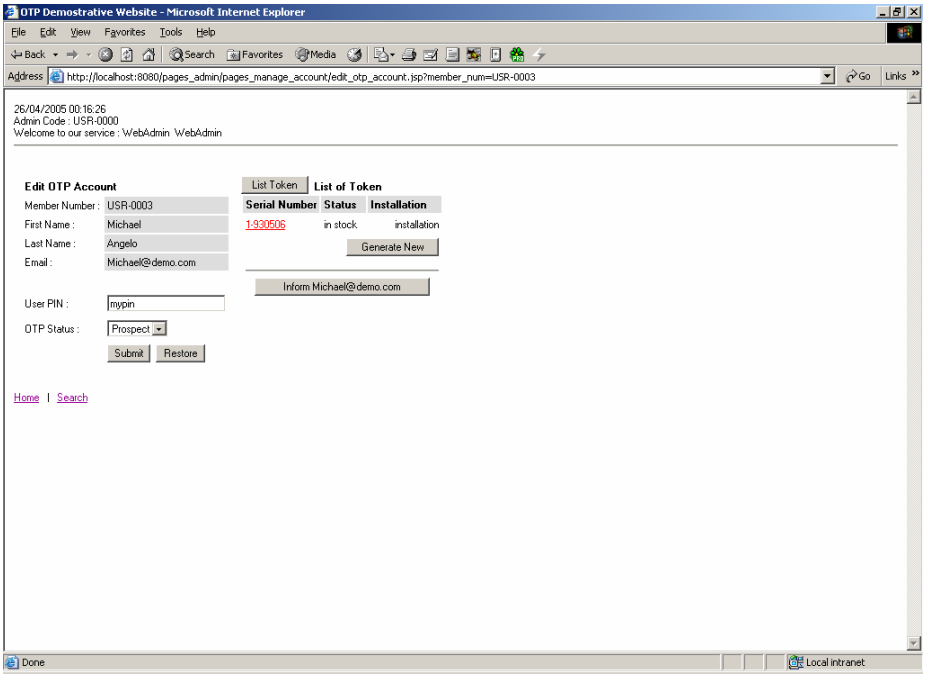
- On the page for manage member's profile "http://localhost:8080/pages_admin/pages_manage_account/edit_otp_account.jsp" administrator can list all of related token by pressing on "List Token" button



- In order to generate new token and relate to this OTP user, administrator can press on "Generate New" button.



- Personal directory of active OTP user would be created automatically on server disk space. Suddenly the java source code file also was created and stored in that directory automatically. (Administrator has to compile this java file and store the product of this compilation, class files, to the appropriated directory)

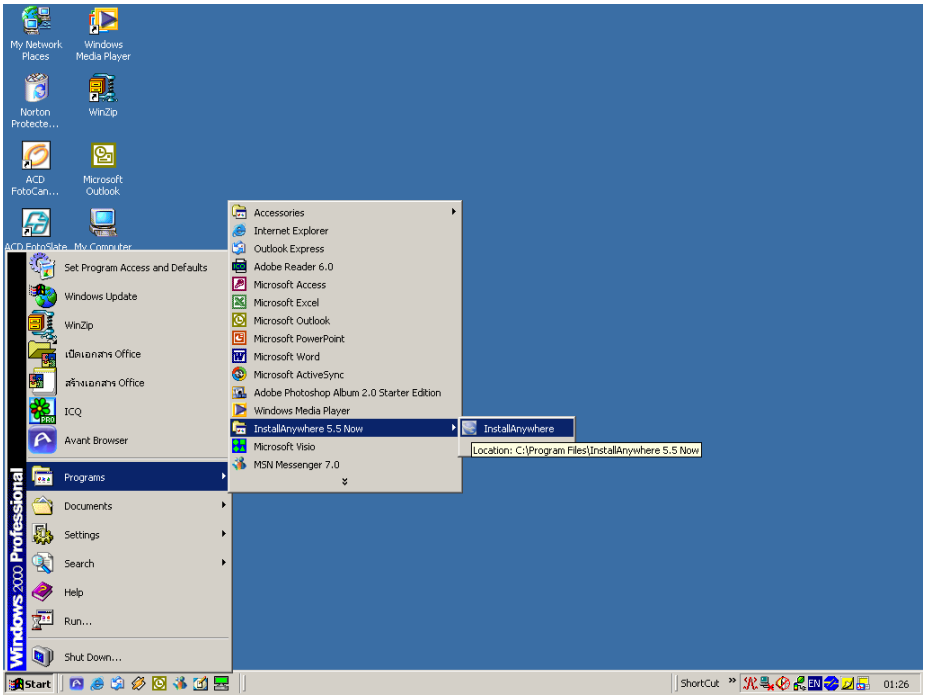


- This OTP account management page, “http://localhost:8080/pages_admin/pages_manage_account/edit_otp_account.

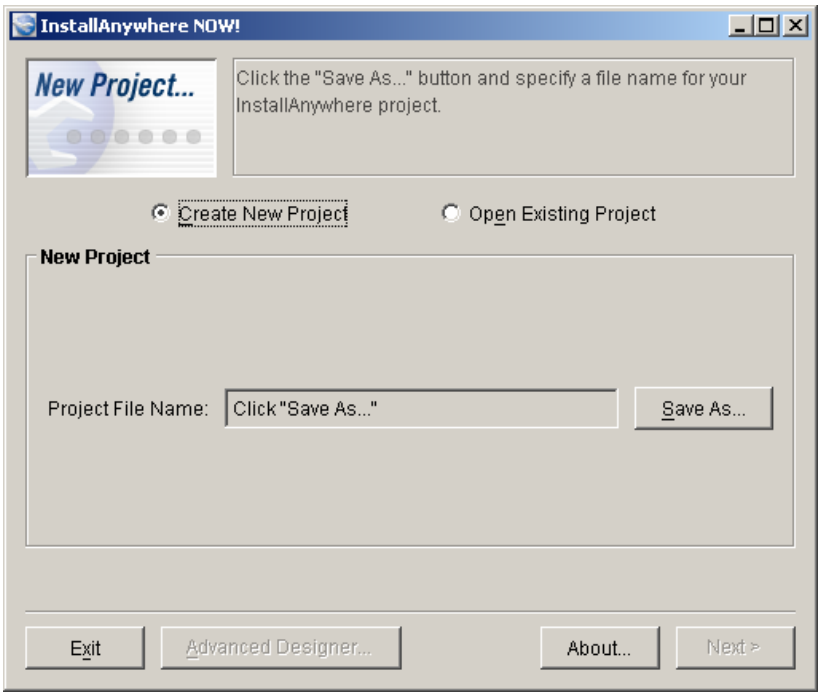
jsp” was reloading then the search result of new token is shown on the right side of page

- Administrator can click on link, which is displayed as token’s serial number, so that the page for manage profile of related token will be loaded.

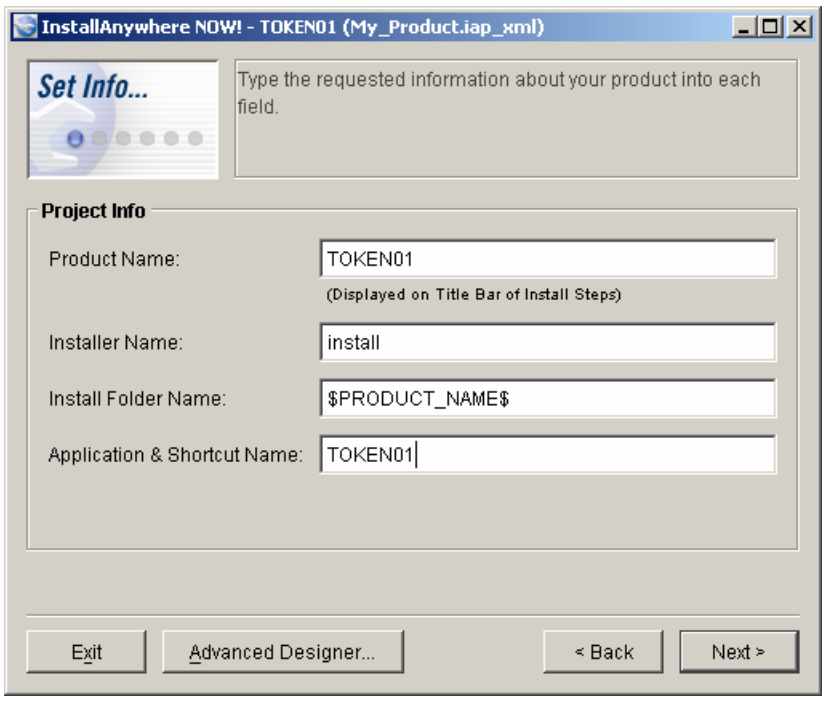
Generate installer by InstallAnywhere 5.5



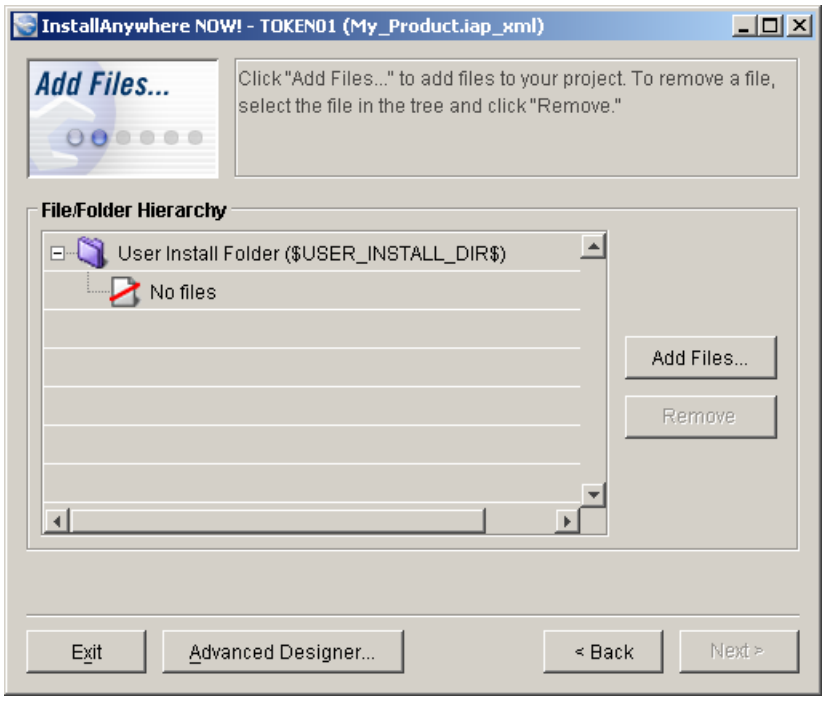
- Activate InstallAnywhere 5.5 (for window), specific application for create installer from java class files, by using command Start button on task bar / Programs / InstallAnywhere 5.5 Now / InstallAnywhere



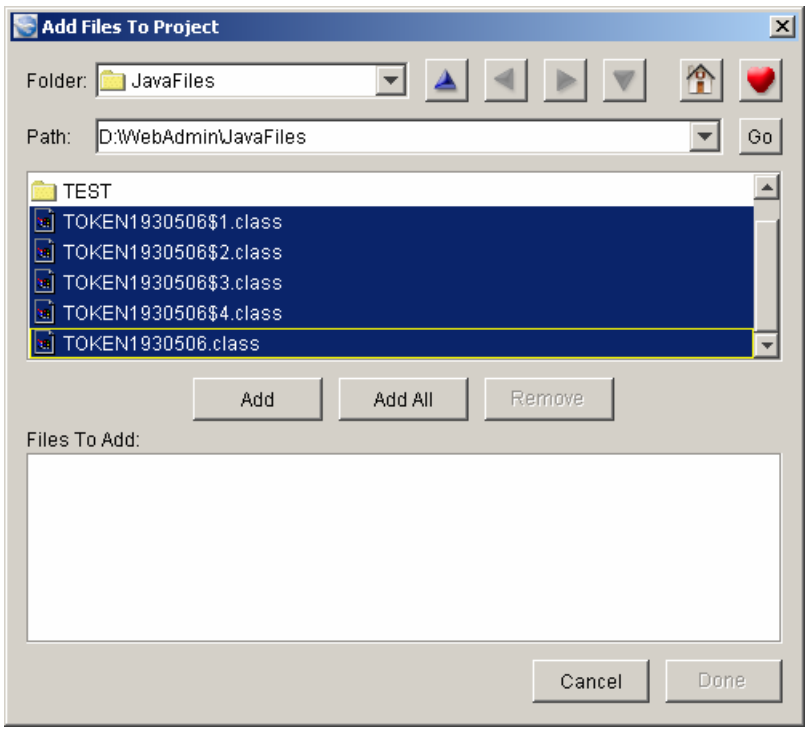
- Popup window of starting the project for generate installer
- Administrator has to press on Save As… button



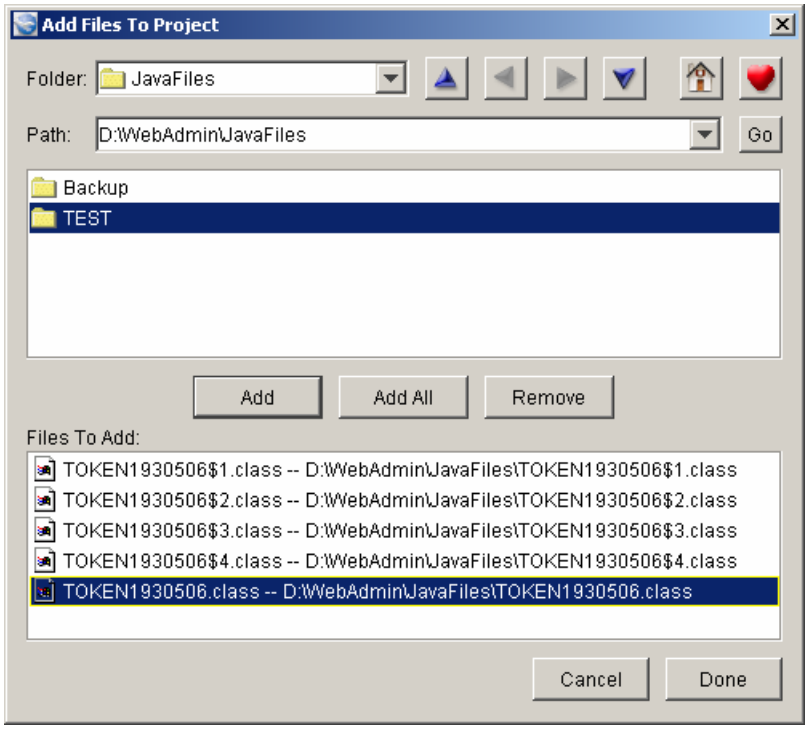
- Enter the Product Name, Installer Name, Install Folder Name and Application & Shortcut Name
- Then press on Next > button



- Popup window of adding java class files to installer
- Administrator has to press on Add File… button to select appropriated class file to this project.

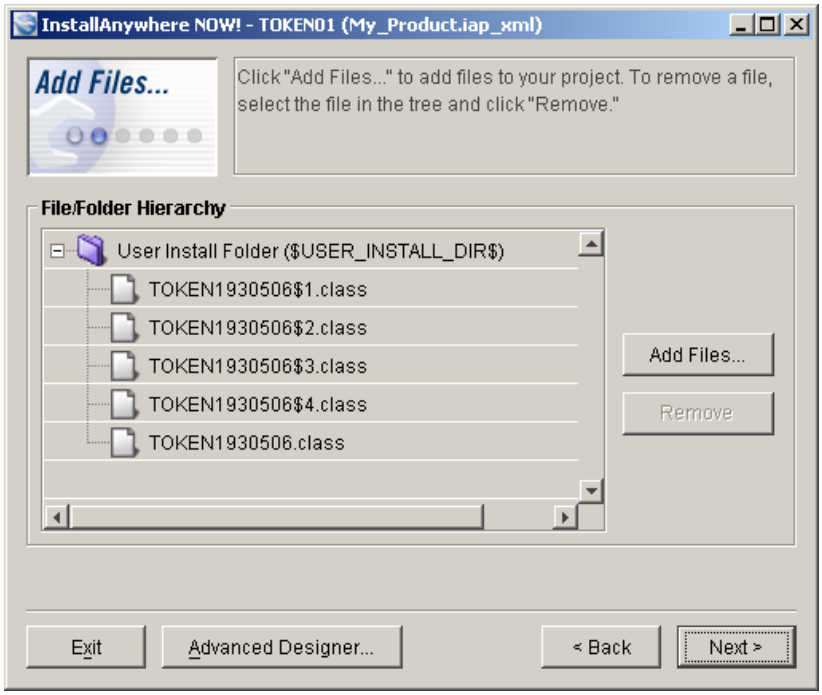


- Popup window of selecting appropriated java class files that were compiled from java source code file.
- Administrator has to select all of related class files then press on Add button

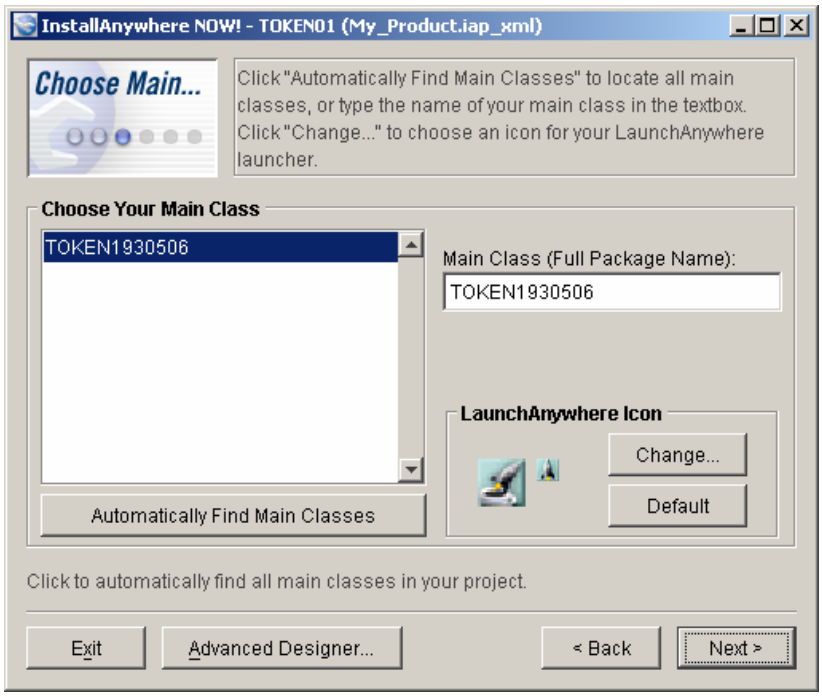


- Popup window of selecting appropriated java class files show the result of selection

- Administrator has to press on Done button to confirm this slection

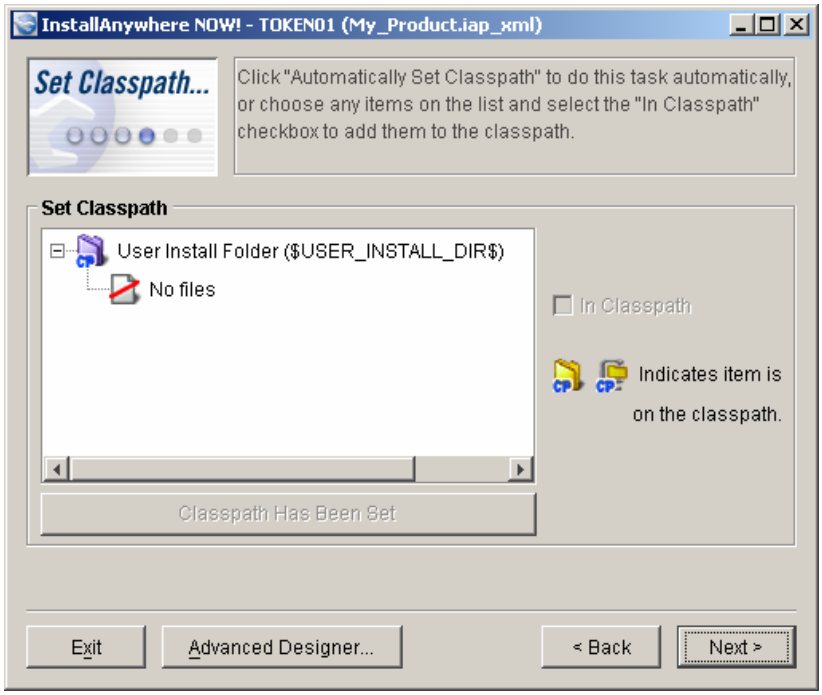


- Turn back to popup window of adding java class files to installer, which display all of included class files.
- Administrator has to press on Next > button

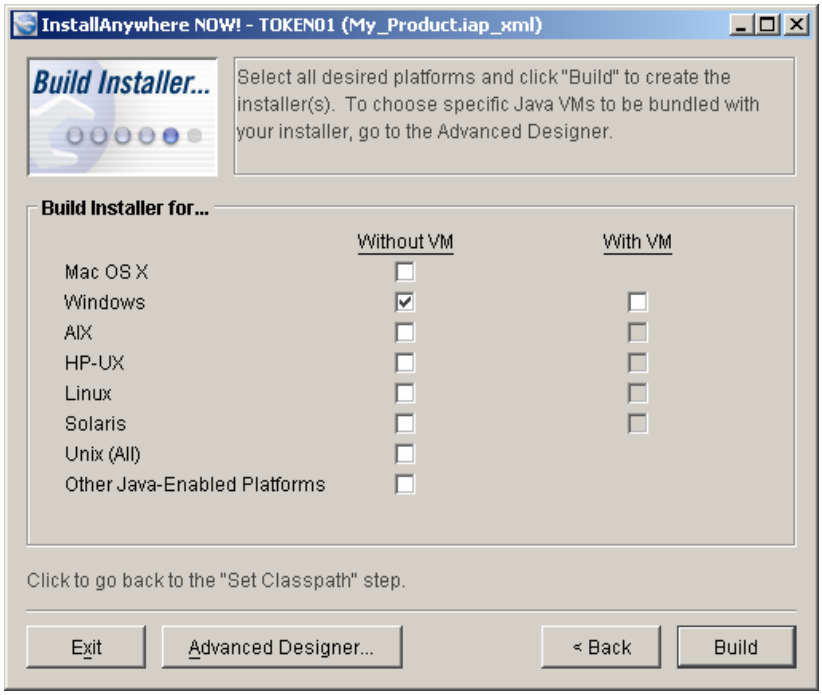


- Popup window of selecting the main class of java application
- Administrator has to press on Automatically Find Main Class button then

press on Next > button

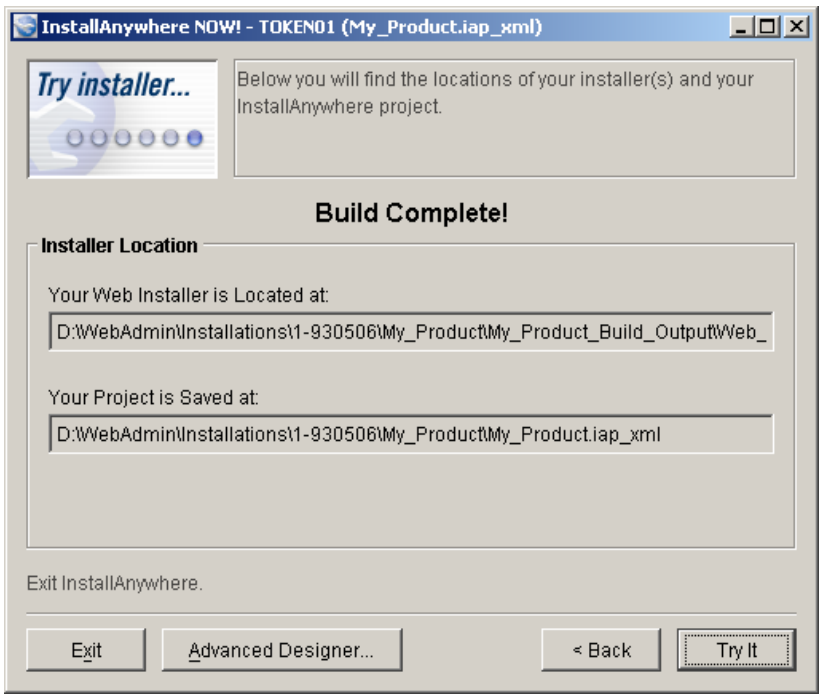


- Popup window of setting the class path
- Administrator can skip this task by press on Next > button

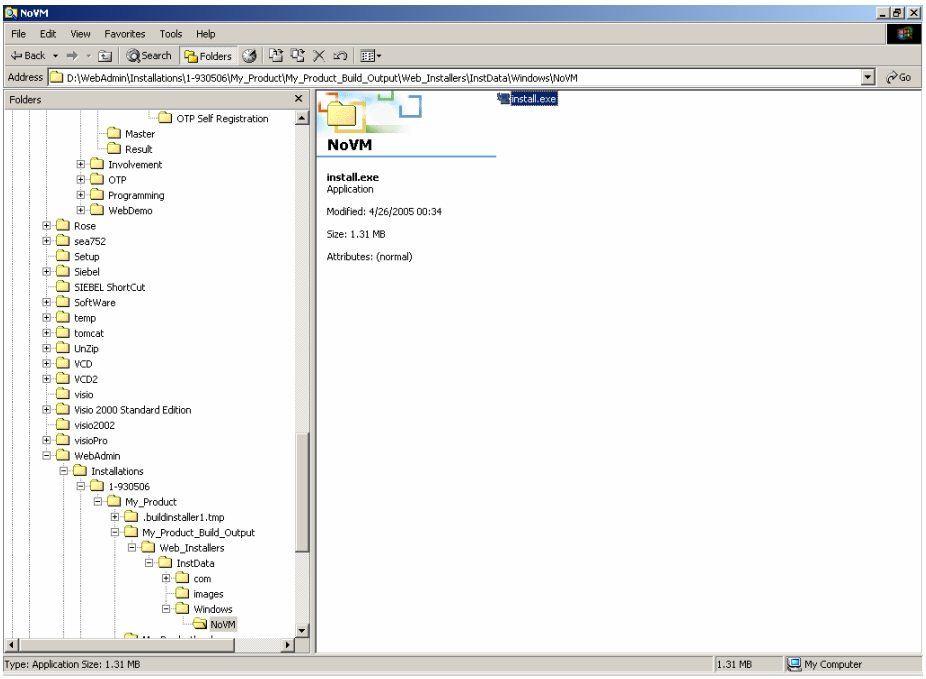


- Popup window of selecting appropriated platform
- Administrator has to select Windows platform without Java Virtual Machine then press on Build button.

- Popup window of indicating that installer is under producing

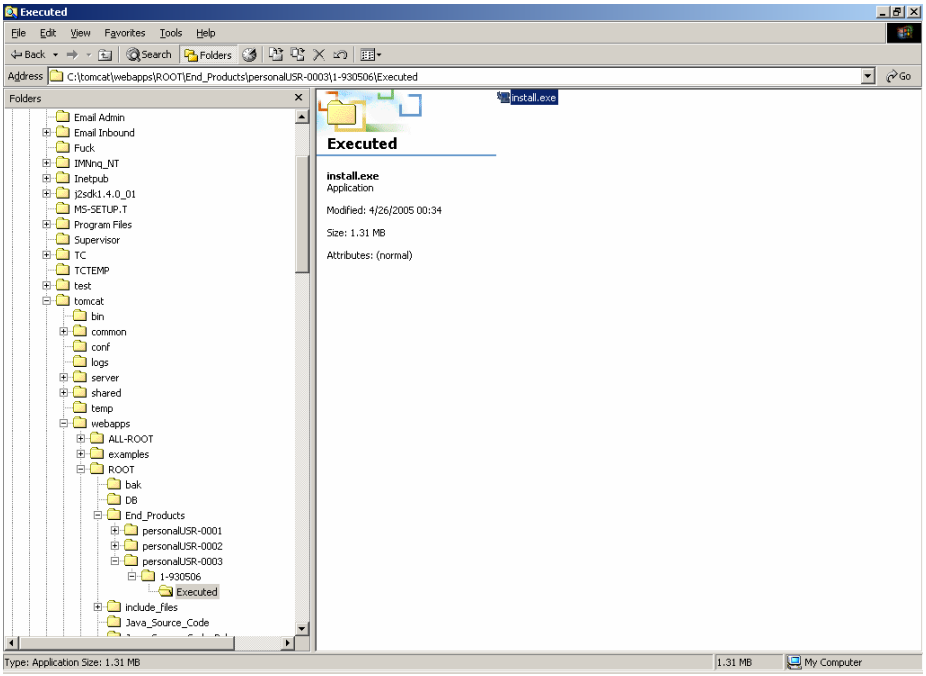


- Popup window of displaying that building installer is completed and show the path of project result.
- Administrator has to press Exit button then go to the directory, which store the expected installer.

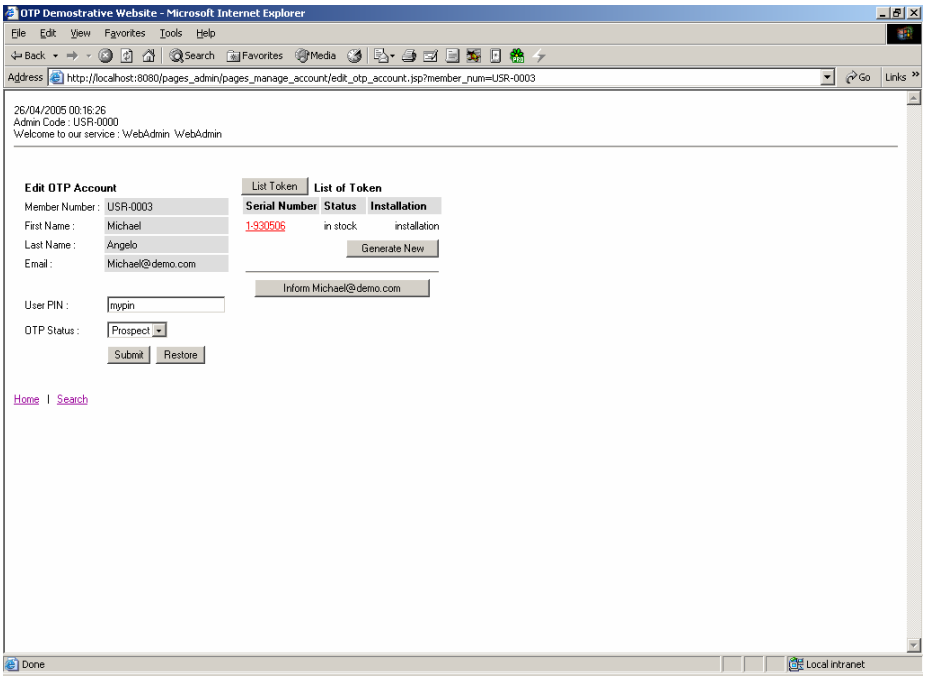


- Window explorer displays the path and file of install.exe, product of this InstallAnywhere project.
- Administrator has to copy or move this install.exe to the personal directory of OTP user who owns the quota of this token.

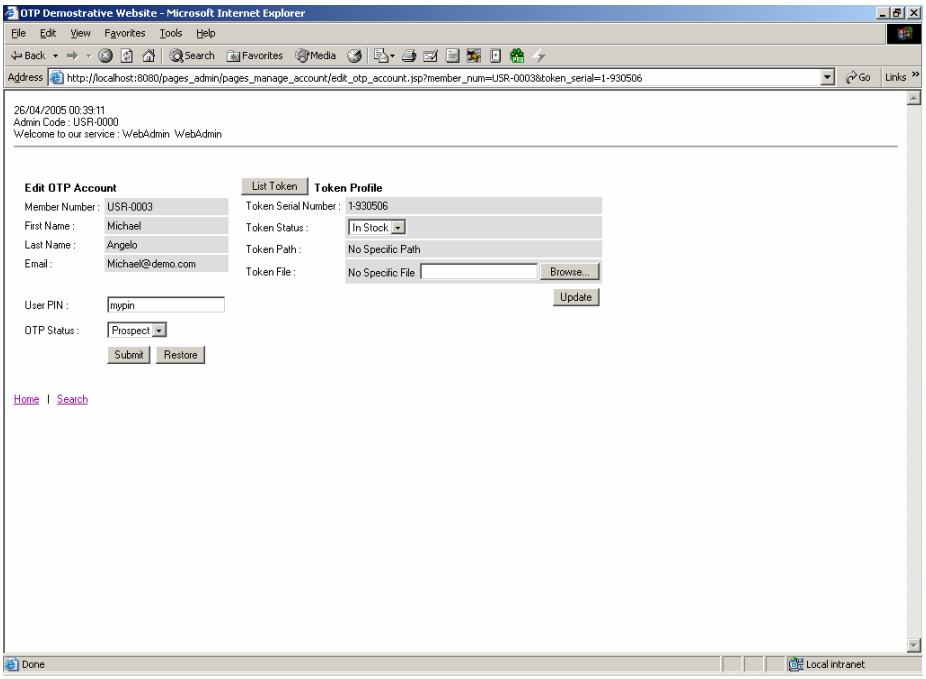
Associate installer file to token profile record by administrator



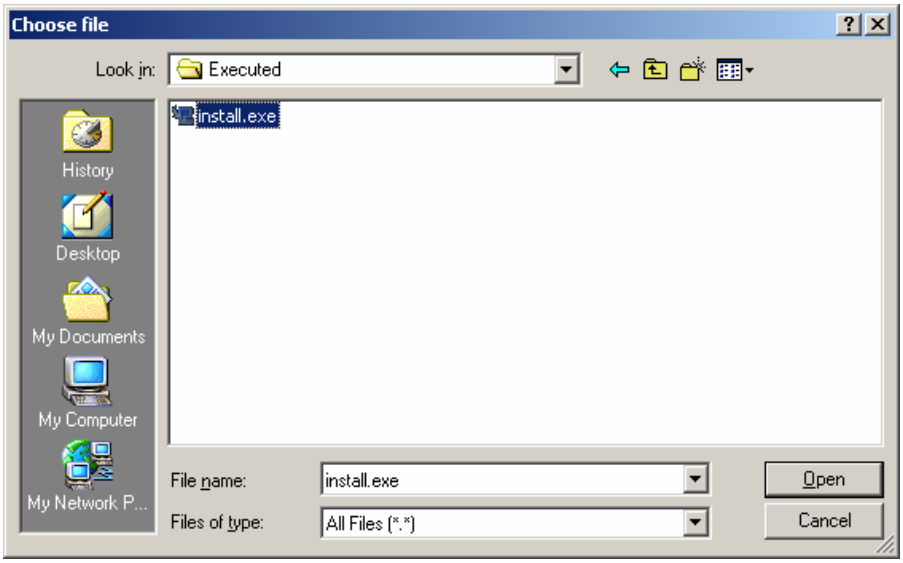
- Copy or move the installer.exe to personal directory of current OTP user (sub directory “Executed” in personal directory is recommended)



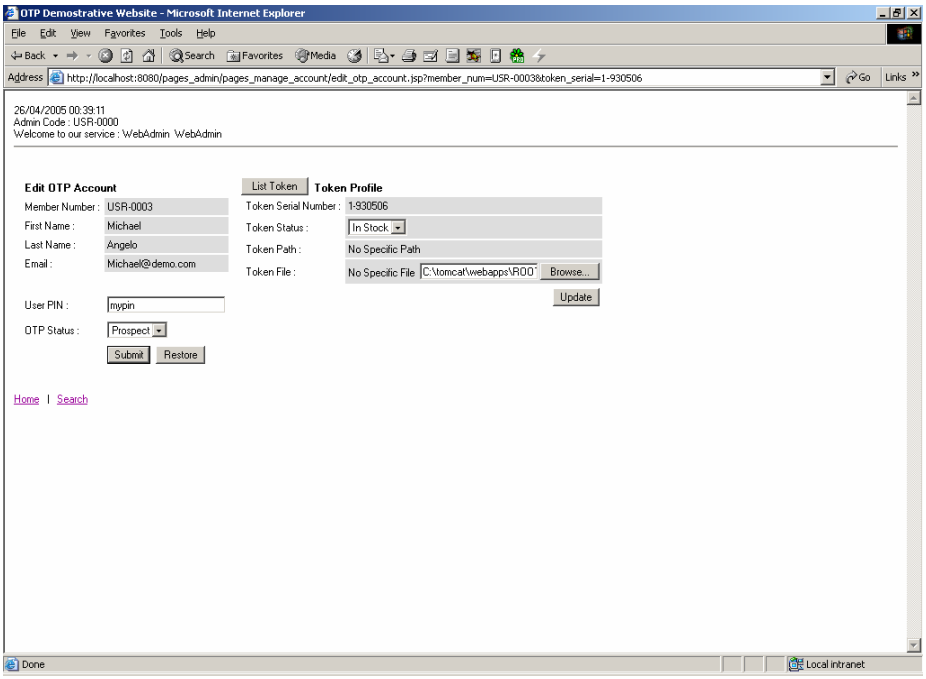
- On OTP account management page, Administrator can click on link, which is displayed as token’s serial number, so that the page for manage profile of related token will be loaded.



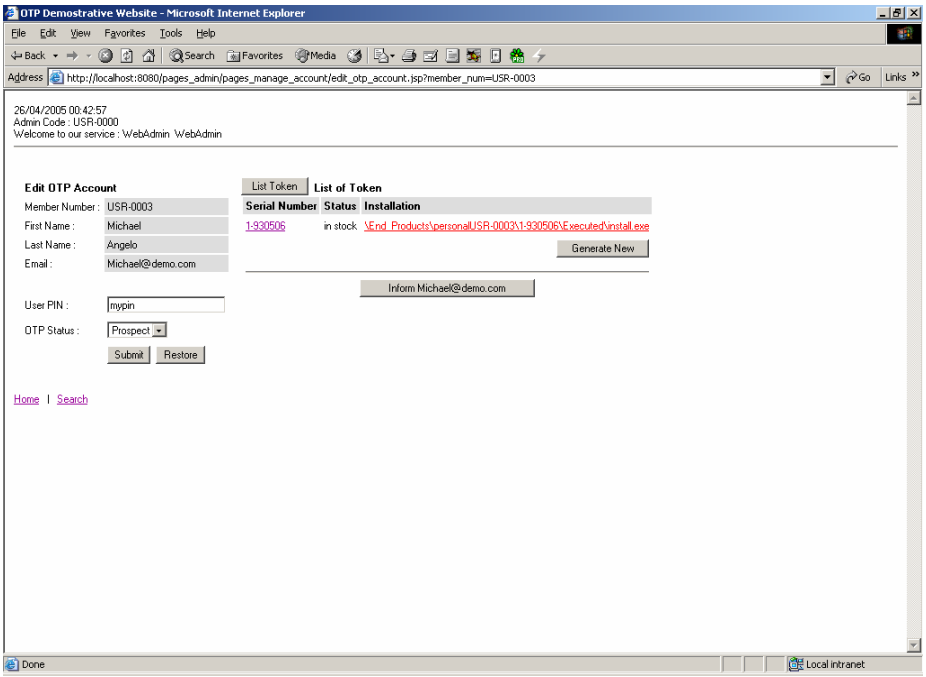
- The page for manages OTP member's profile is reloaded again with the form for modifying the current token profile.
- Administrator can modify token's information by browsing the physical installer file (press on Browse… button).



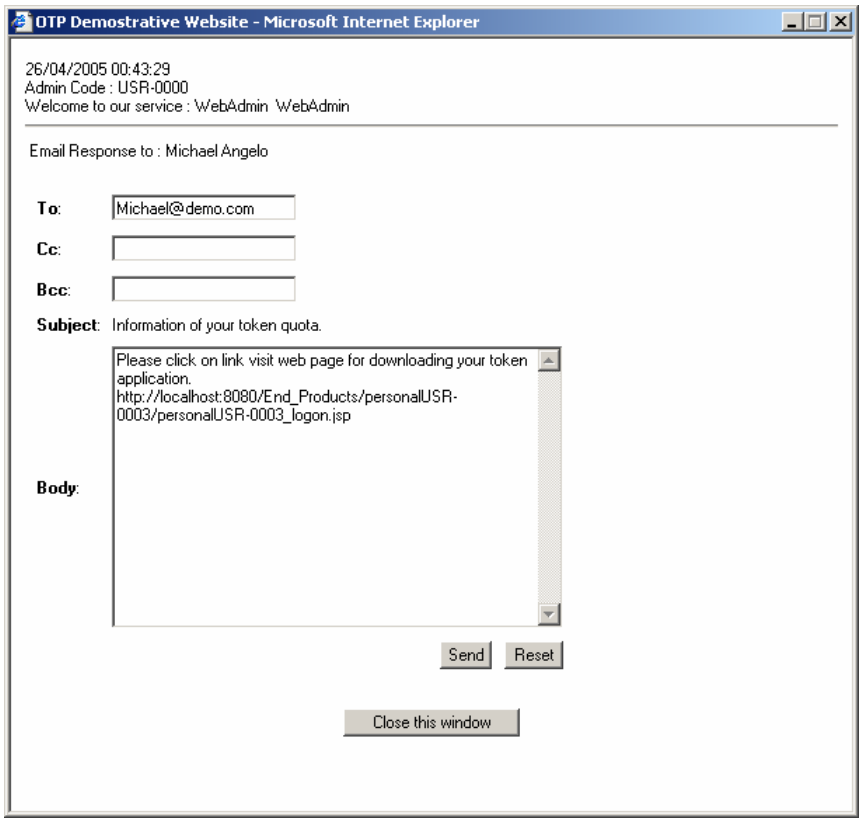
- Select installer from personal directory.



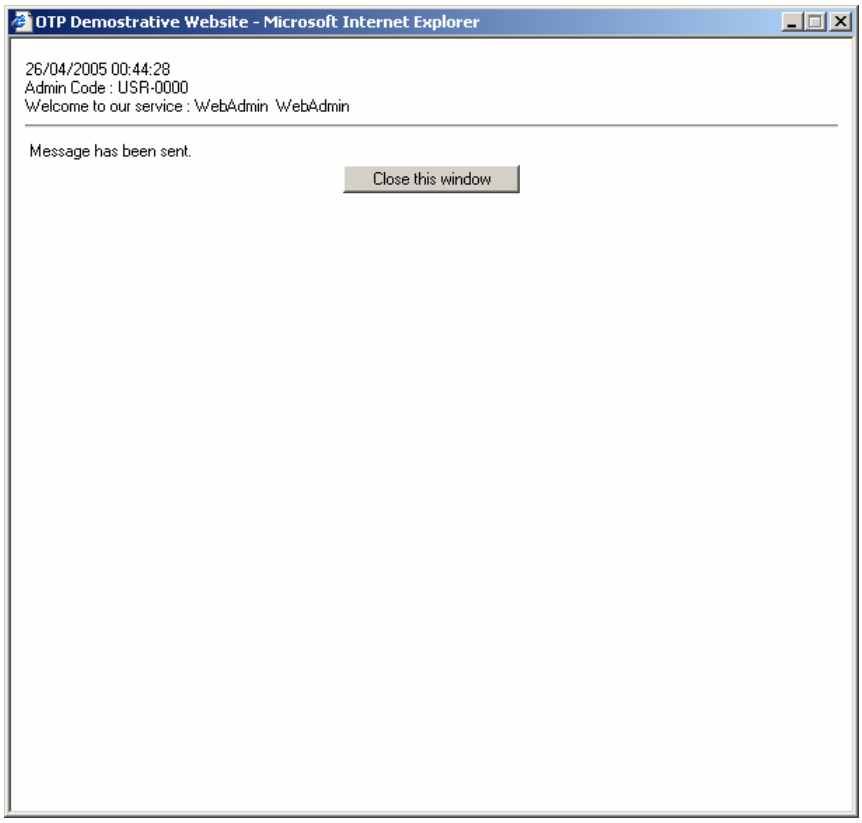
- The page for manage OTP member's profile display details of current token already had token installer.
- Administrator has to press on Update button.



- The page for manages OTP member's profile display list of related token, which was associated by physical installer already.
- Application can support administrator for generating email to inform remote user by press on Inform <user's email address> button

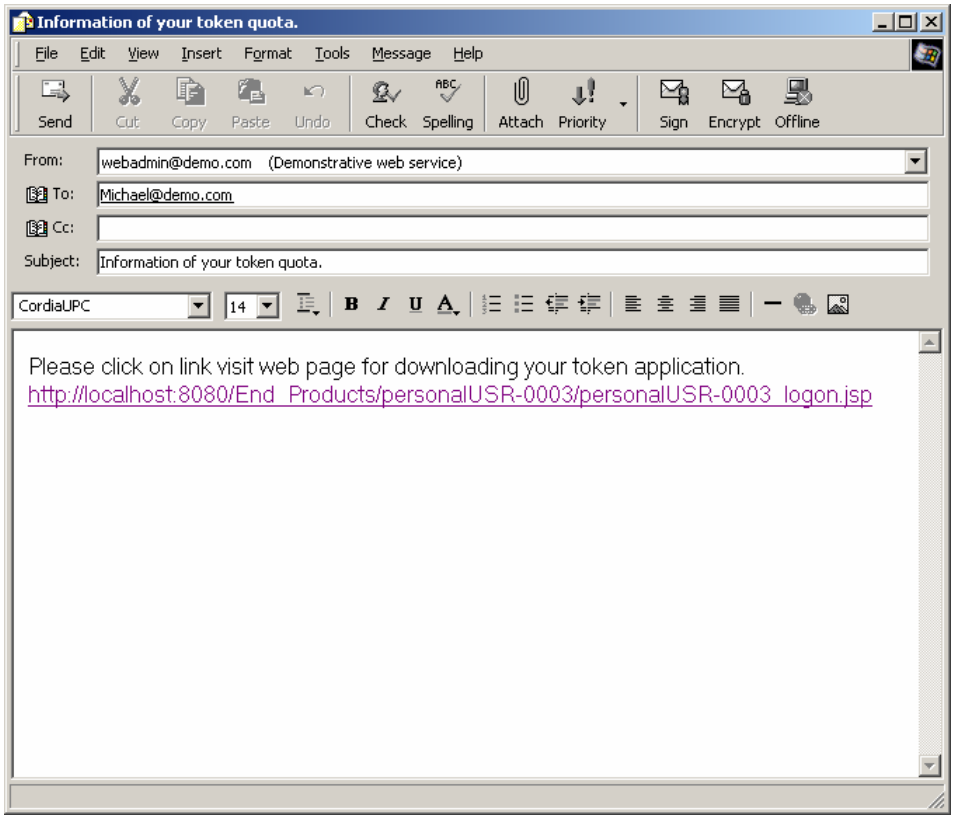


- Popup windows display details of email, which has to be sent to remote user, for informing he/she that the system already was finished about preparing the components for prospect OTP user.
- Administrator has to press on Send button.

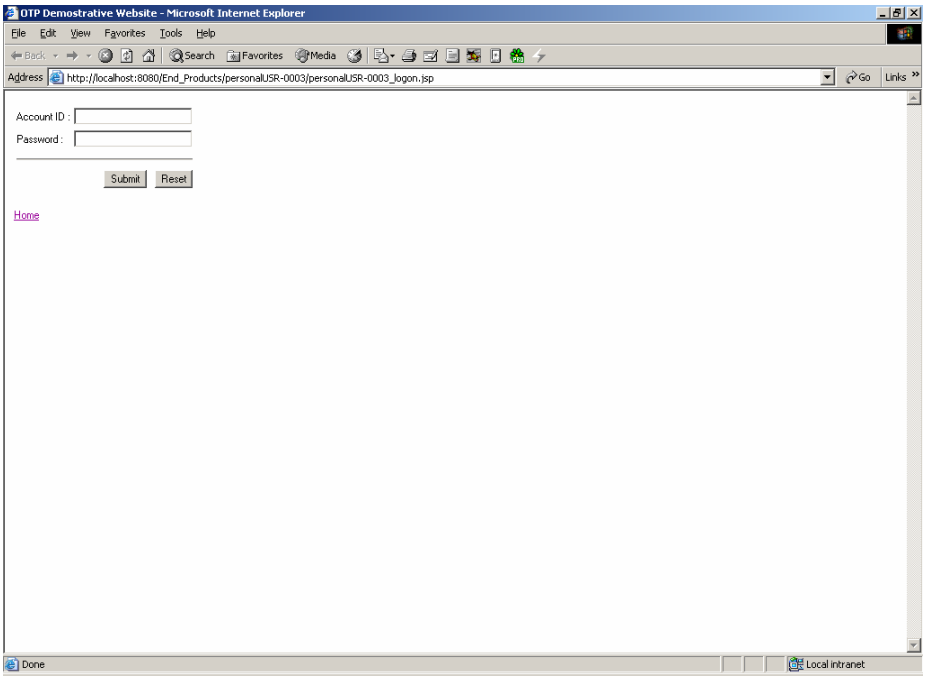


- Inform email was sent already.

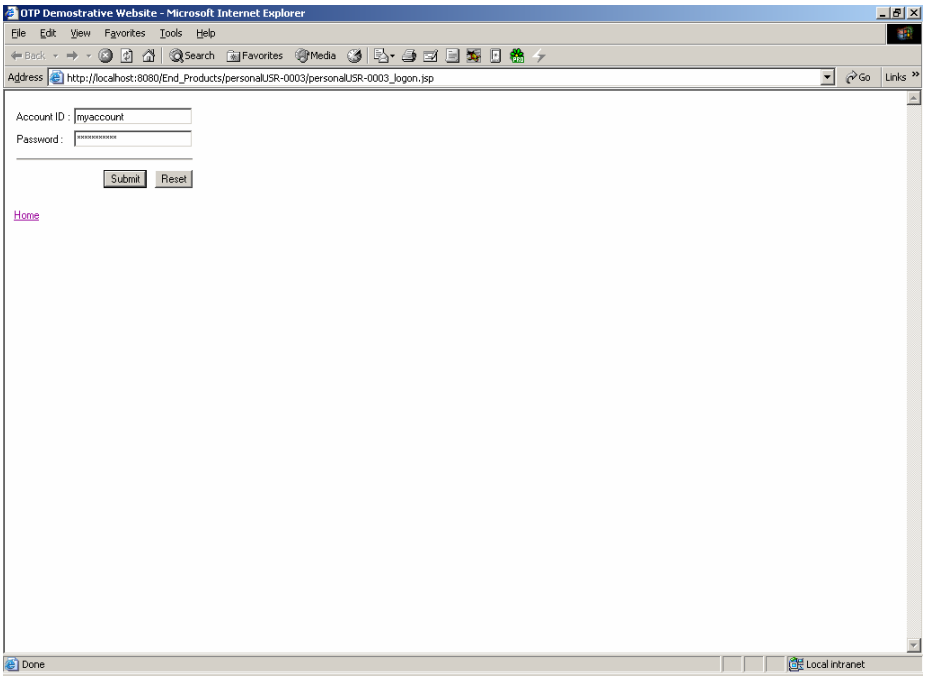
Activate and Download token to client machine



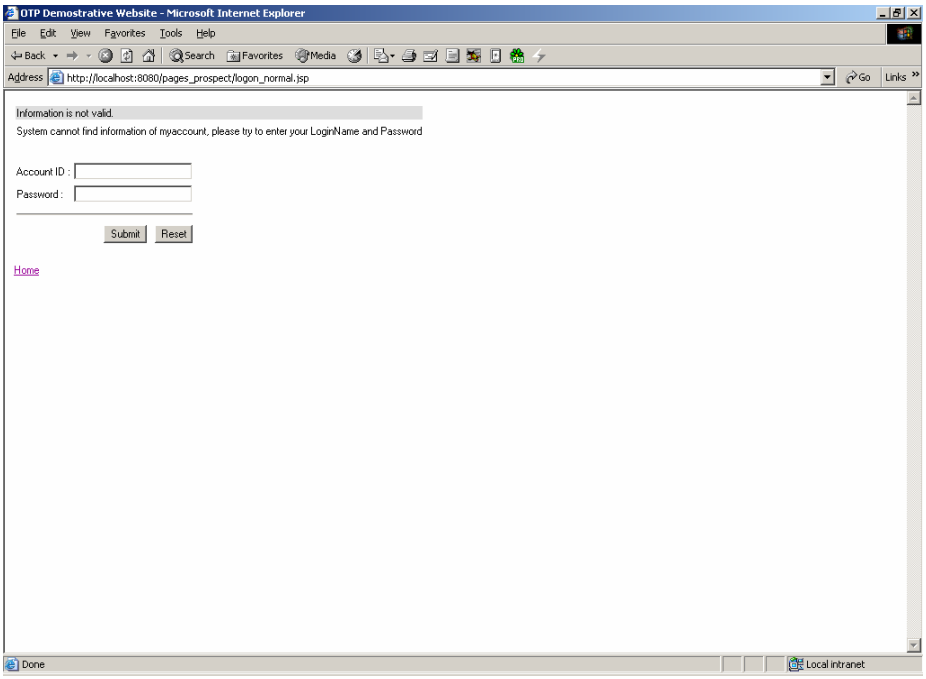
- Email from web administration display the content to inform client user that try to process downloading token application.
- Client user has to access to personal web page for special logon to mange their token quota.



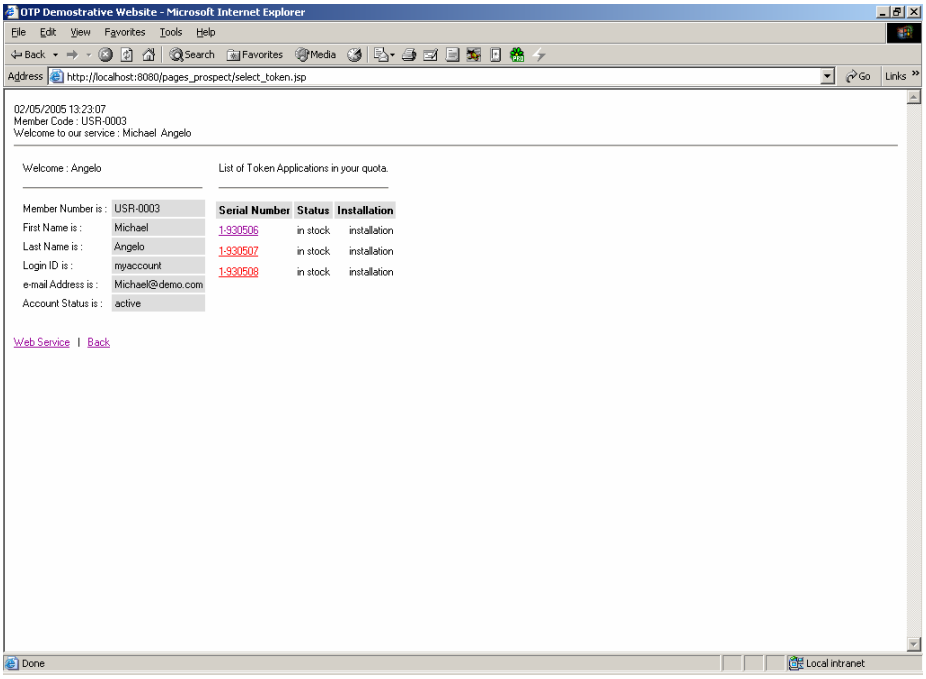
- Personal logon web page, which was initiated in personal directory, is opened



- Submit client user's account id and static password through personal logon page

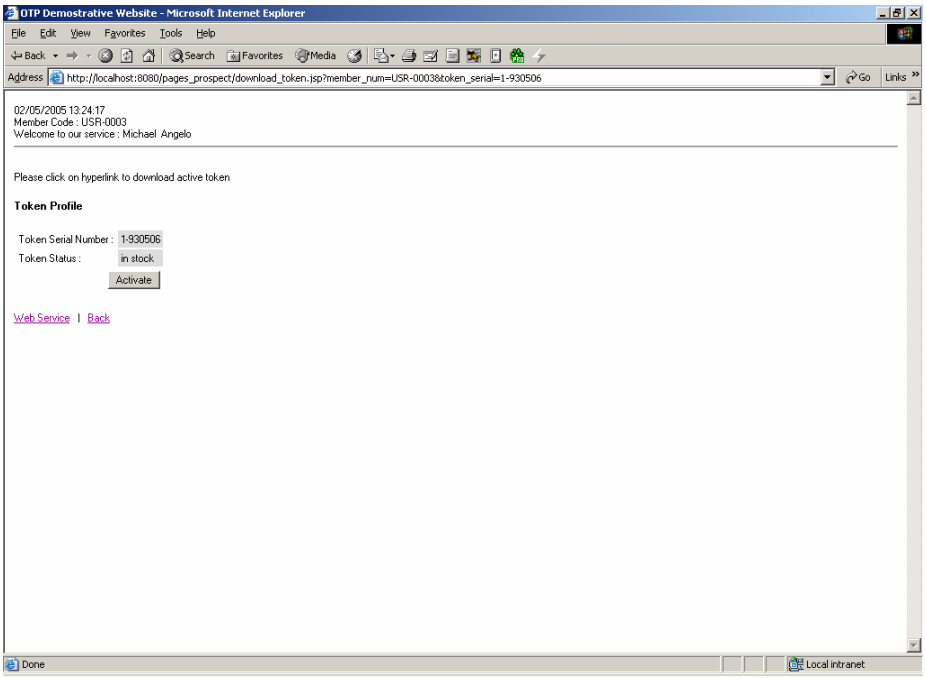


- In Case entered data is invalid the logon page will be reloaded with alert messages and waiting for valid information
- Client user has to reenter the right information and submit to system by pressing on Submit button

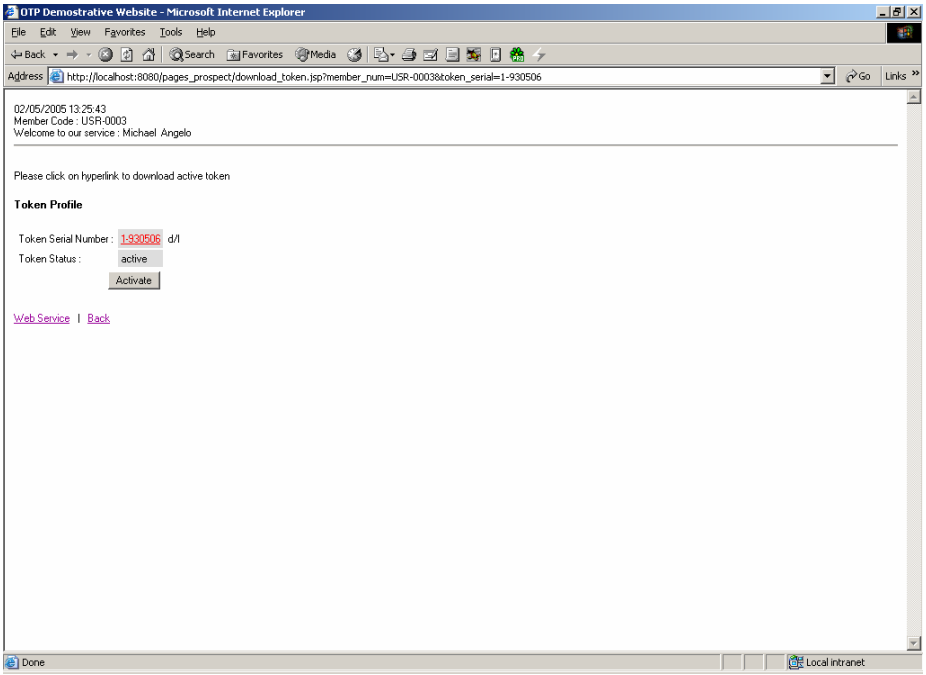


- These information, Account ID and Password, were verified before system allow user access to web page for managing his/her token quota, “http://locclhost:8080/pages_prospect/select_token.jsp”

- Client user can select one in list of tokens to update its status by click on link, which is displayed as token's serial number.

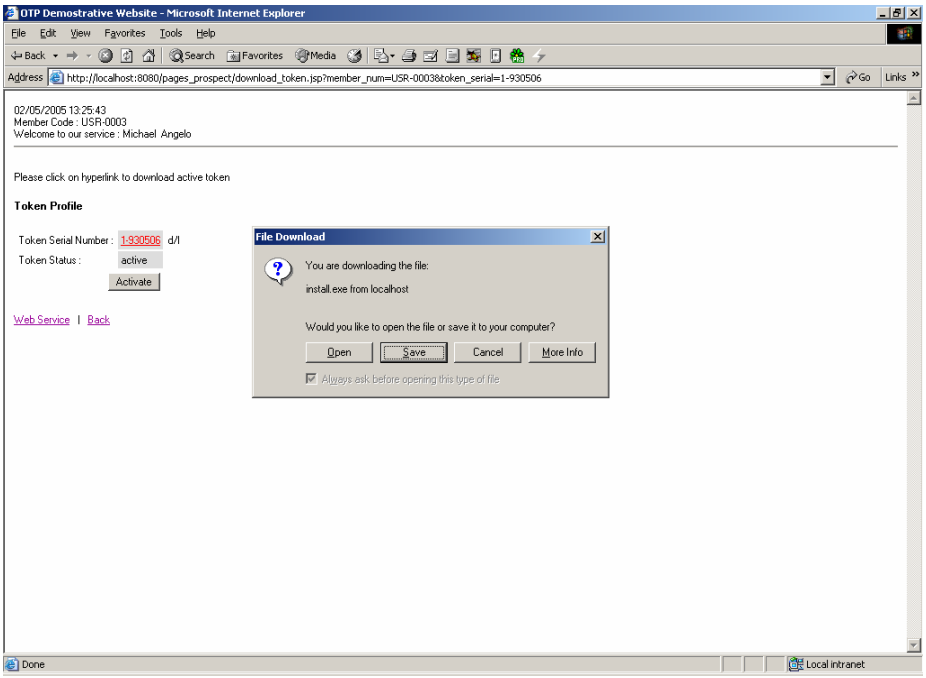


- The web page, using for activate the selected token application http://localhost:8080/pages_prospect/download_token.jsp is opened
- Client user can set token status to be active by pressing on Activate button.

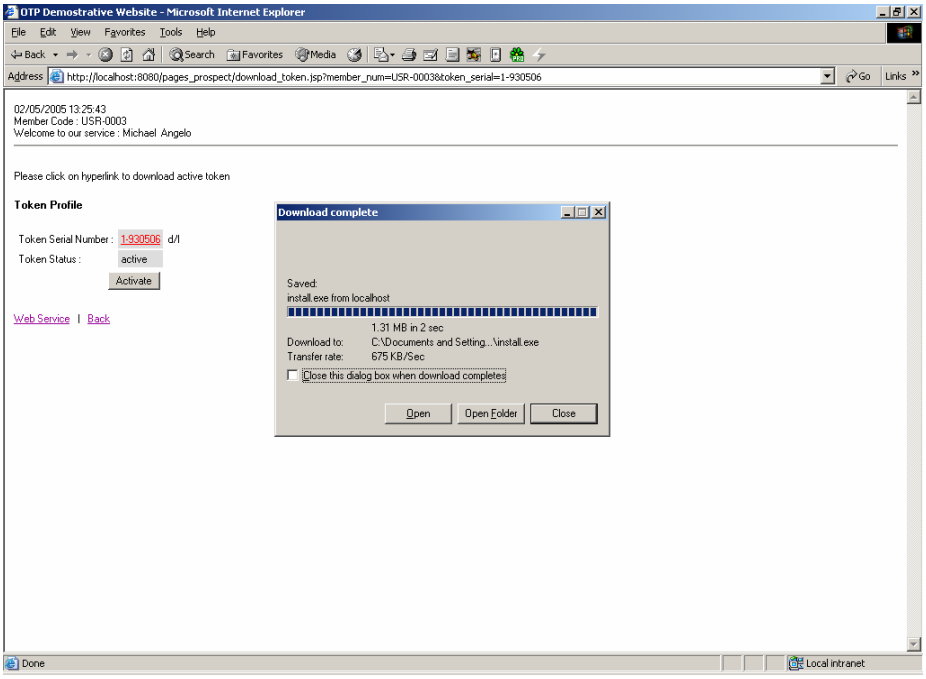


- Status of current token record is changed to "Active" while system can investigate for the others must have no status as active automatically.

- And client user can get the installer file to his/her client machine by click on link, which is displayed as token's serial number.

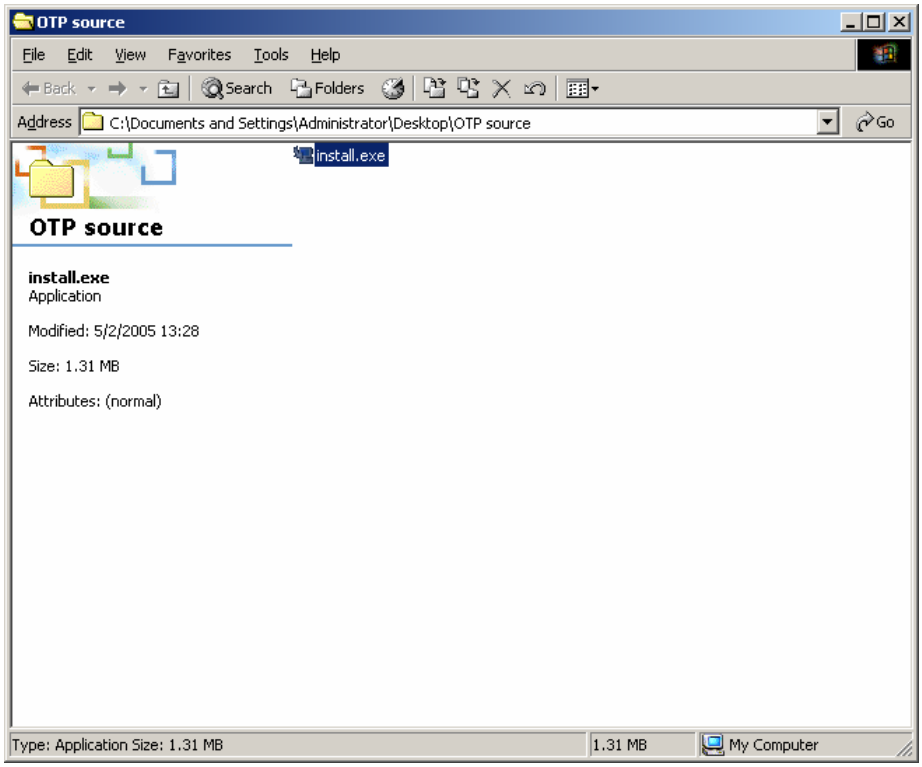


- Installation named "installer.exe" will be downloaded to local machine.
- Client user has to identify the right place to save this installer file.

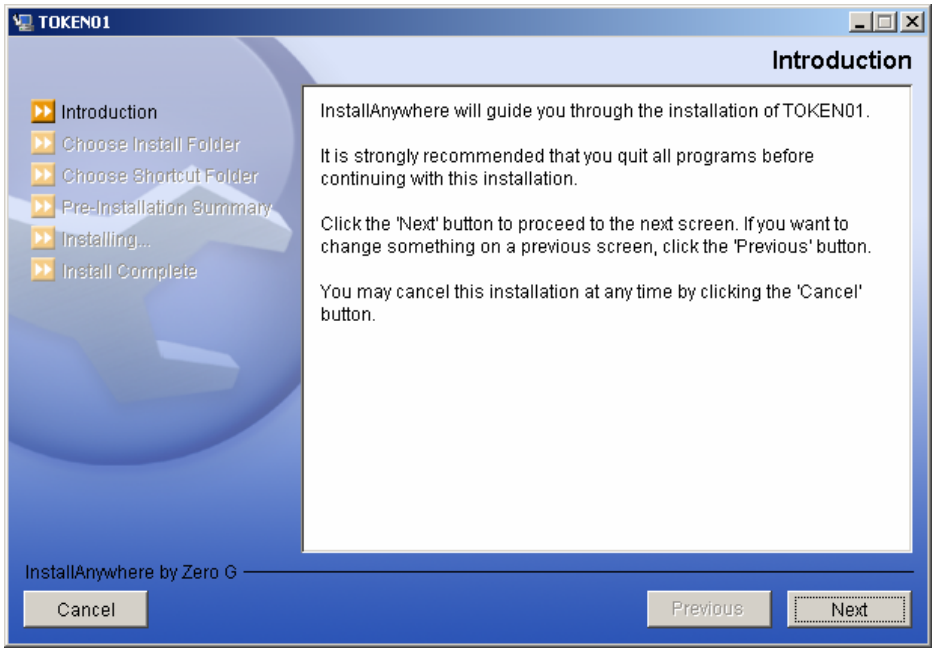


- Installer is downloaded to local machine

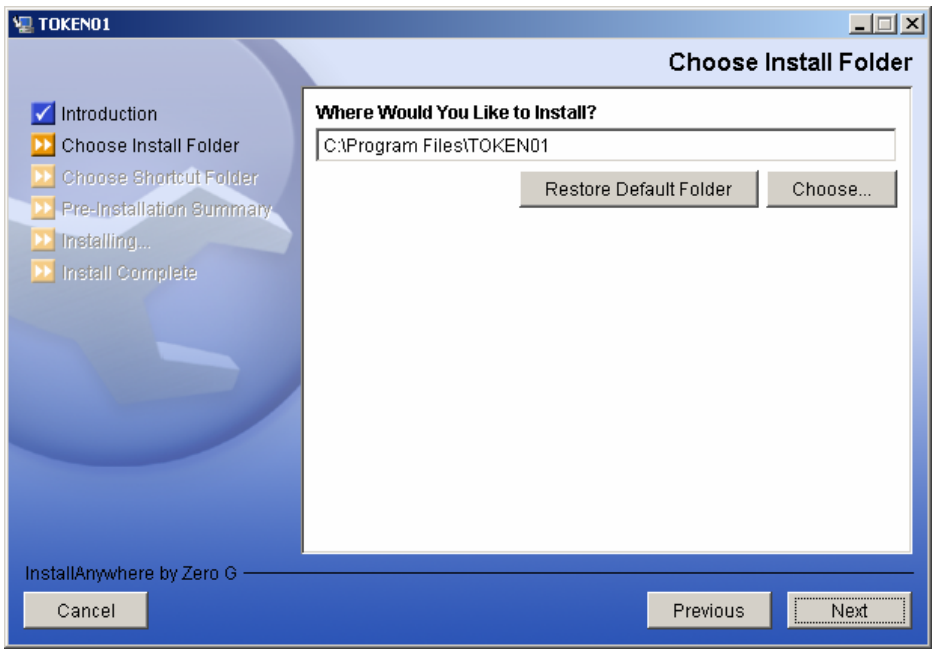
Install token application



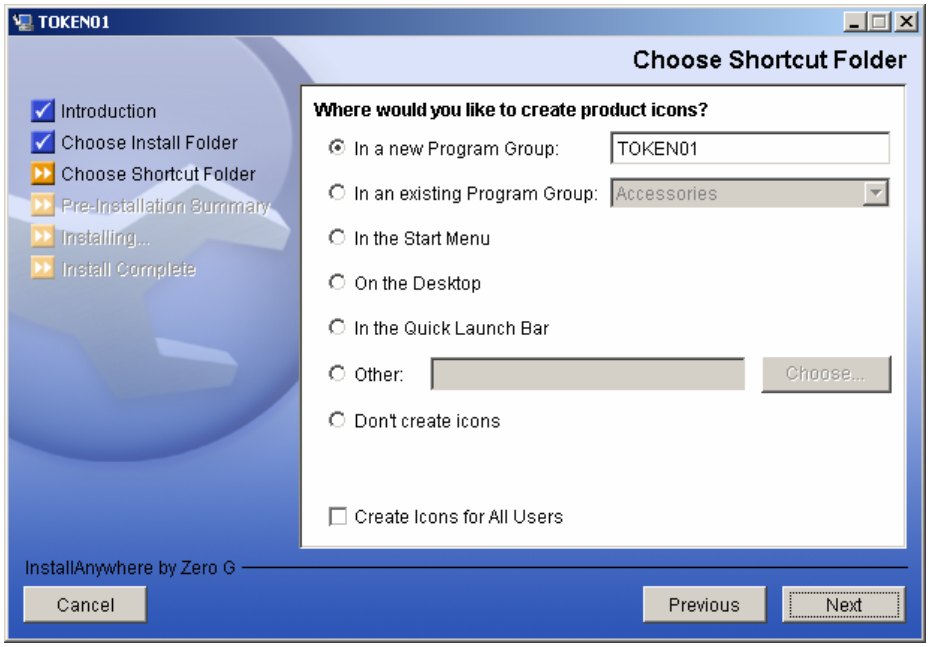
- Open installer by double click on icon of installer.exe



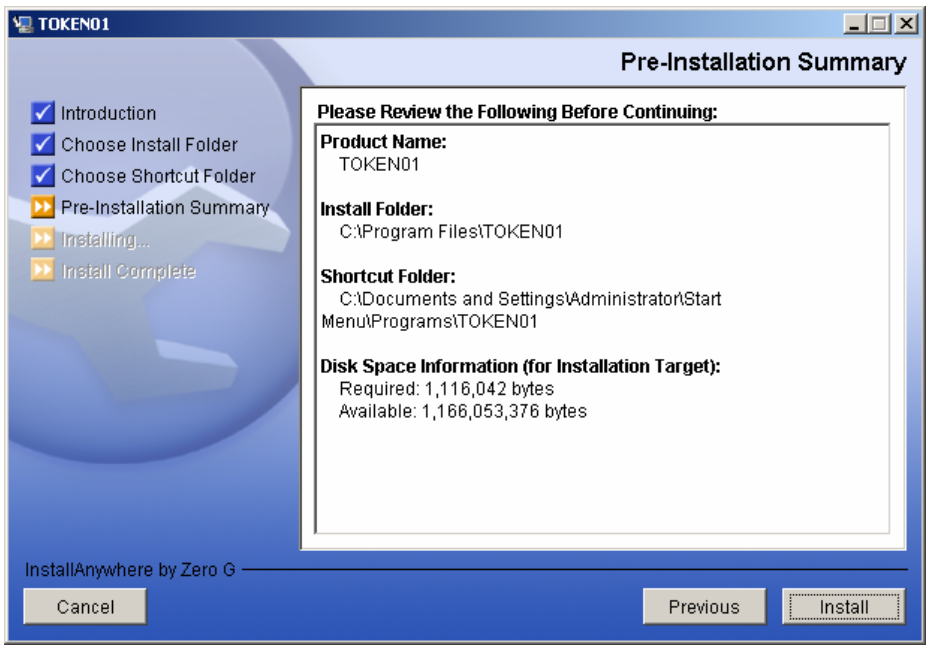
- Popup window display introduction message
- Client user has to press on Next button



- Popup window allow user to select appropriated folder to install.
- User can use default folder or choose others then press on Next button.

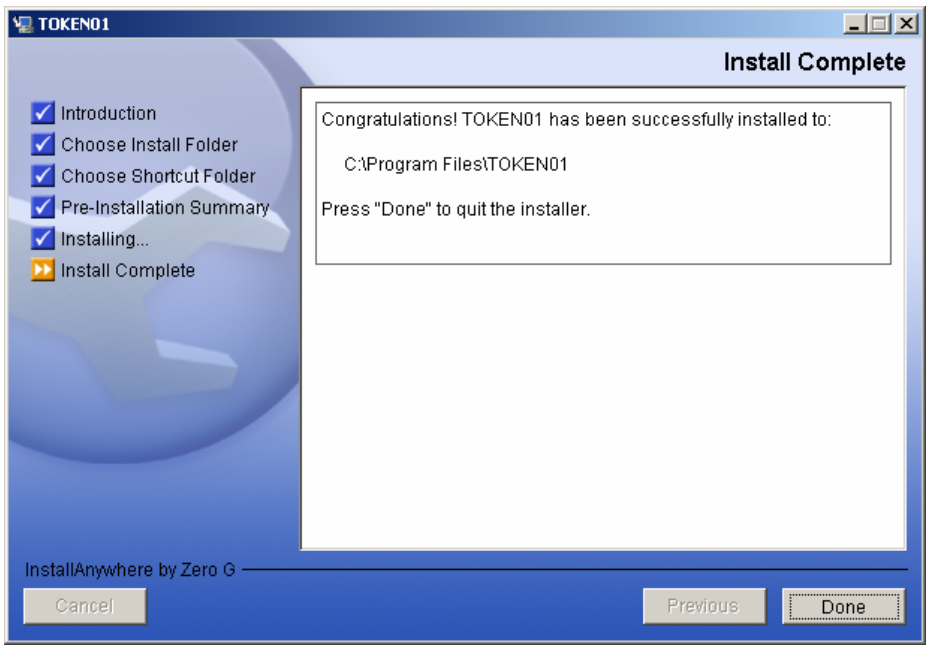


- Popup window allow user to identify the place where store the icon of token application
- After select the appropriated place, user has to press on Next button

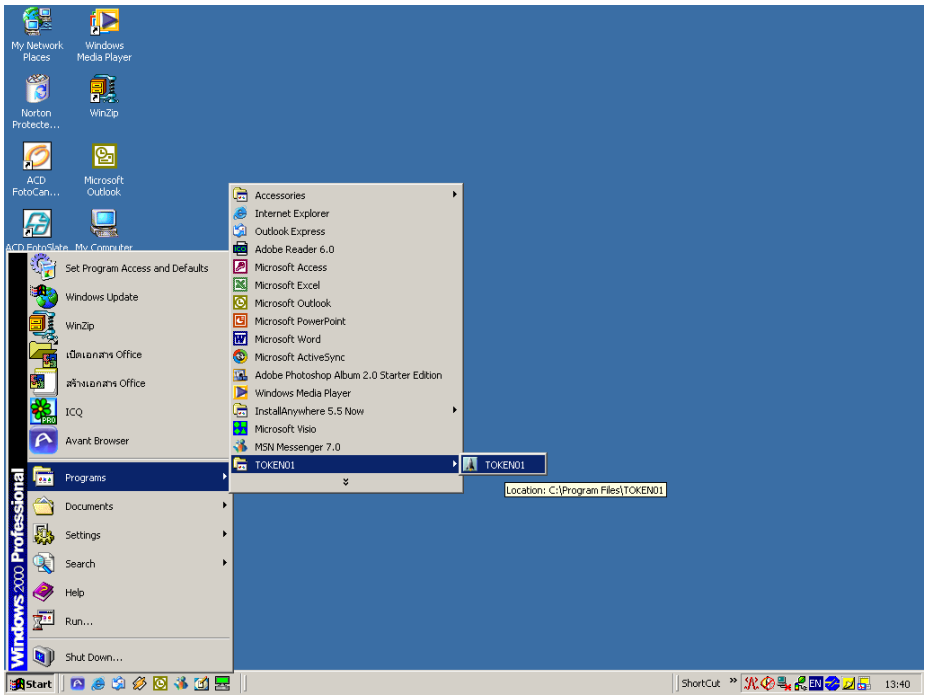


- Popup window show the pre-installation summary message
- After acknowledge, user has to press on Install button

- Popup window indicate the process of installation



- Popup window show inform message that install completion
- After acknowledge, user has to press on Done button to finish installation process



- To open the token application, user can use the shortcut of application, which identified in the step of installation.

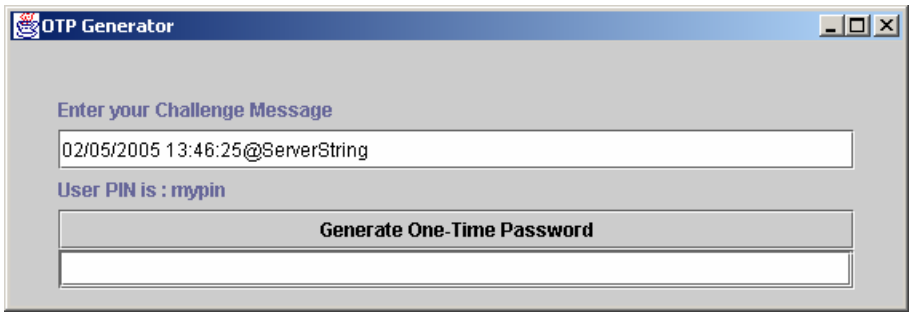
Launch local token application

- The first popup window is asking user for the right user PIN

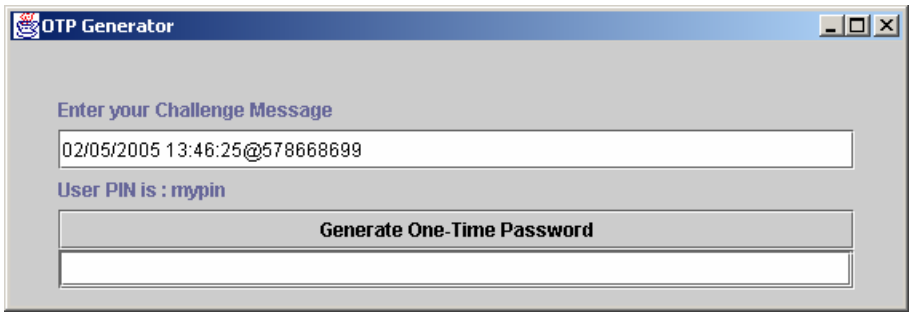
- Enter the user PIN

- In case of entering invalid user PIN, popup window inform user that error.

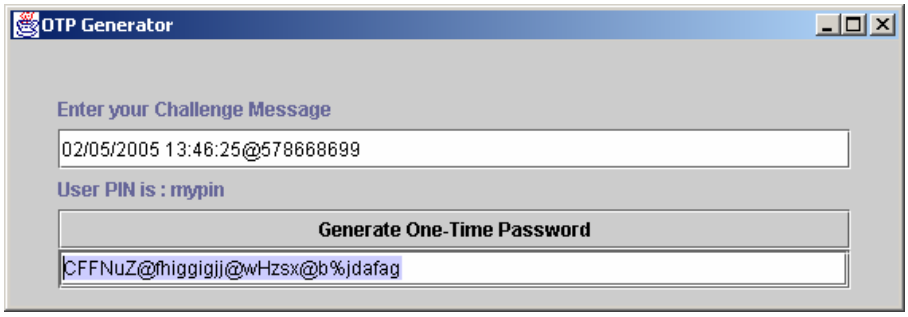
- In case there are 3 times of error, the application will be terminated automatically for that using time.



- After user the right PIN was verified, application will bring the screen for generating One Time Password.
- Client user has to enter the challenge message into above text box before using command to generate OTP

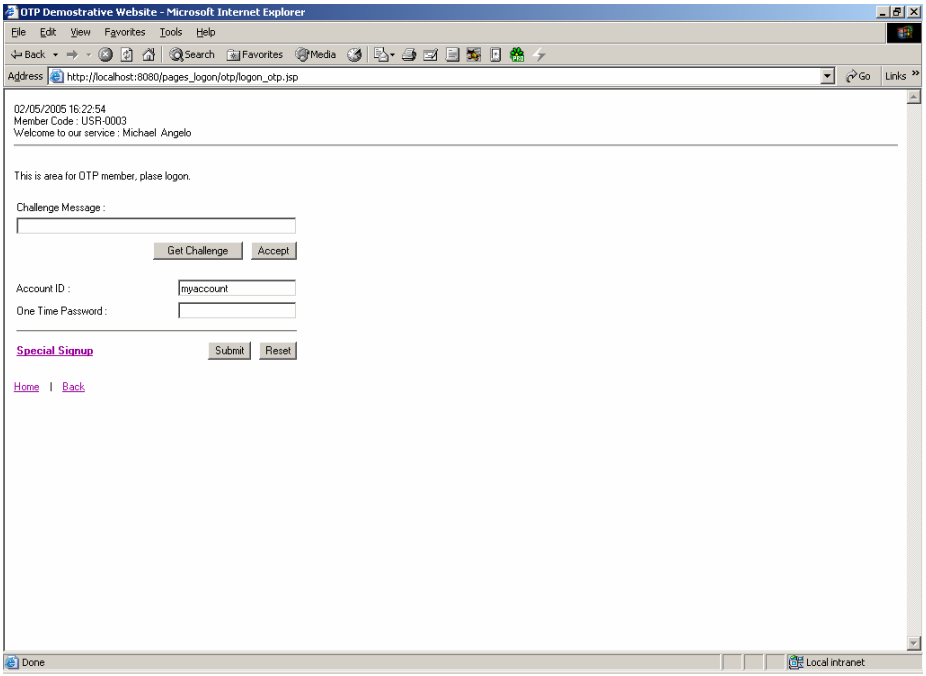


- Put the challenge message into text box.
- Then press on the Generate One-Time Password button

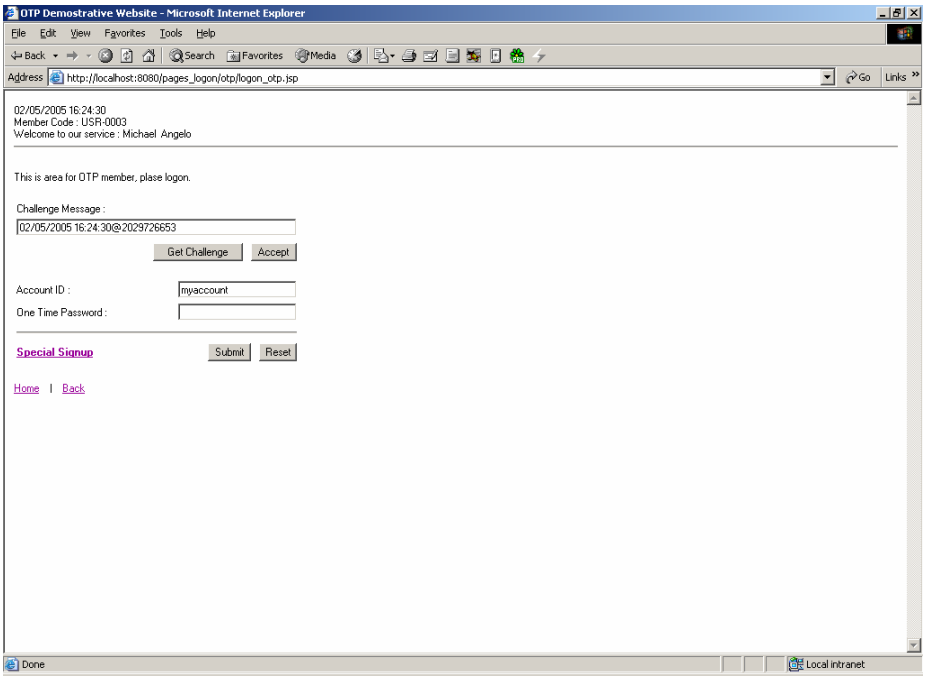


- Cipher message is generated then the expected One-Time Password is shown in the below text box.
- Client user can bring this final message to use as One-Time Password in logon process for extra level member, OTP user.

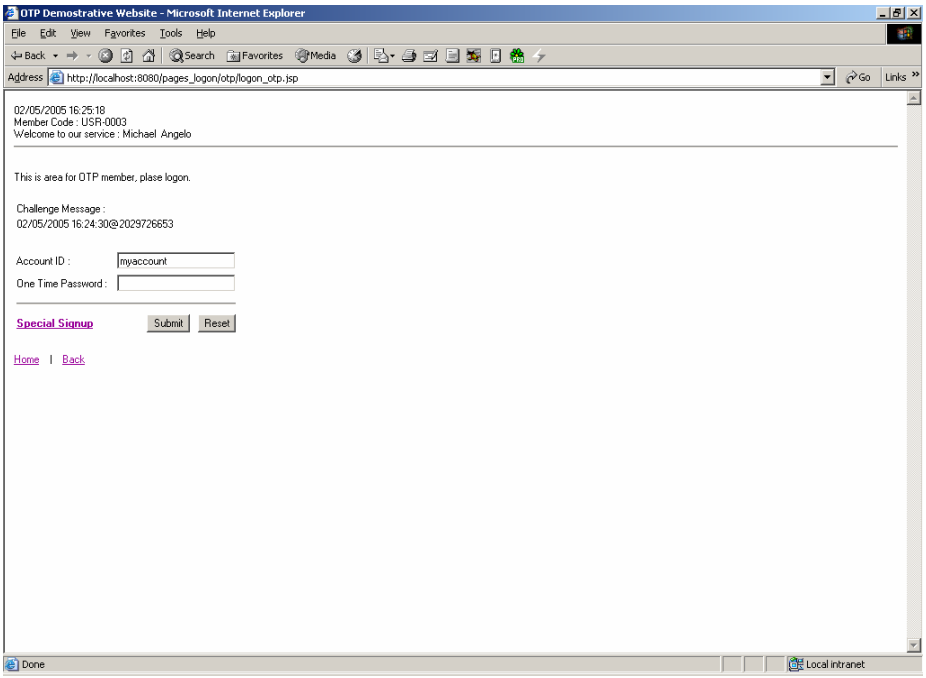
Logon process for extra level member, OTP user



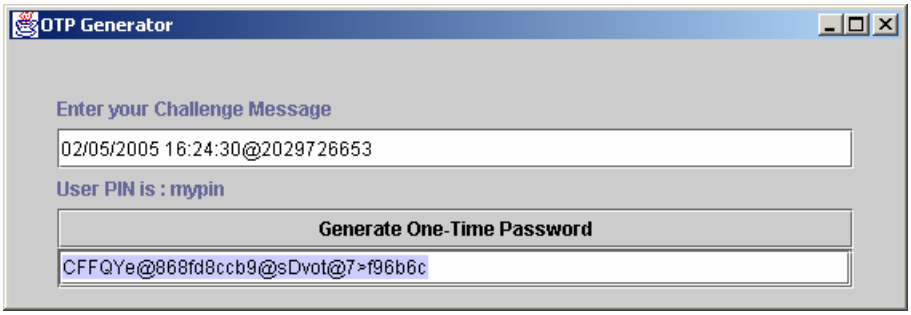
- After get authentication by logon process for normal level user, client user can access to the web page for special authentication to next level of service
- On OTP logon web page, “http://localhost:8080/pages_logon/otp/logon_otp.jsp”, client user has to get challenge message by pressing on Get Challenge button



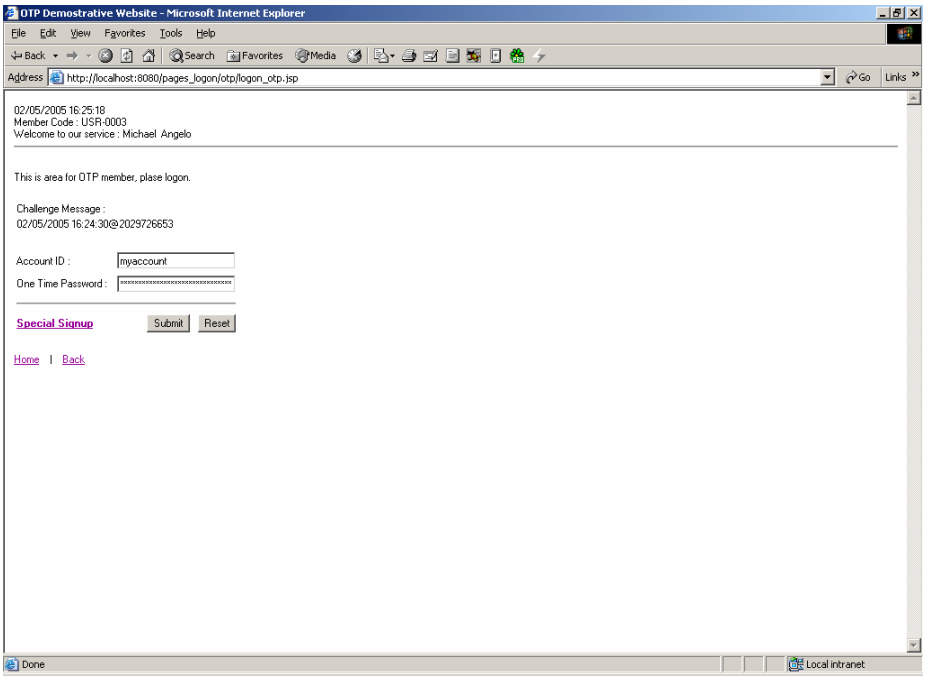
- The immediate challenge message is displayed on the text box. Client user can change to any message by pressing again on Get Challenge. Challenge message can be changed all the time until user accepts it.
- To accept the message user has to press on Accept button.



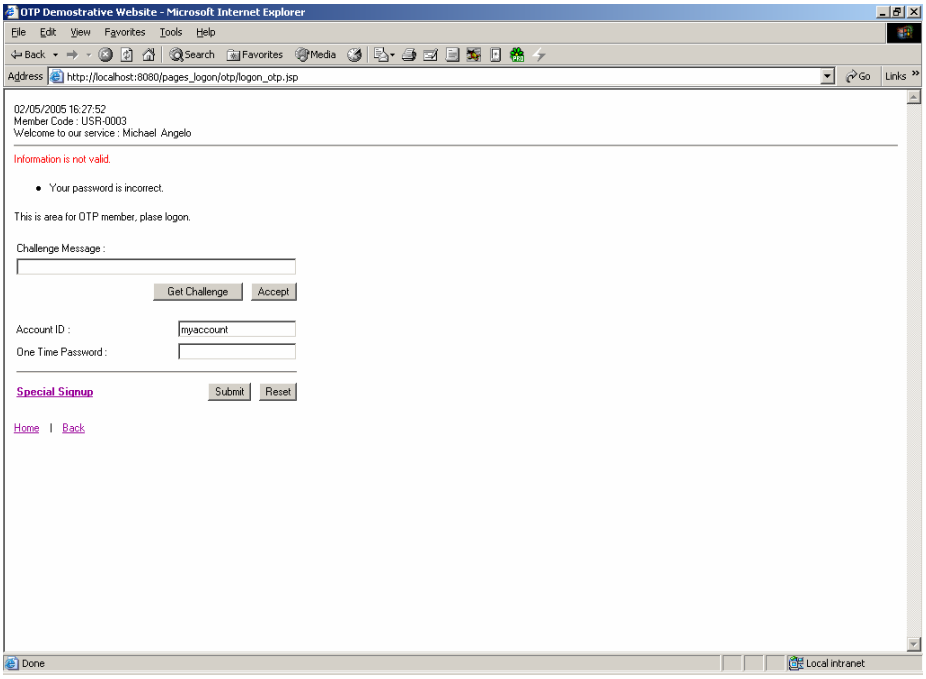
- While challenge message is accepted, the system is counting down for the time that was limited for logon process.
- Client user has to bring this challenge message using in token application



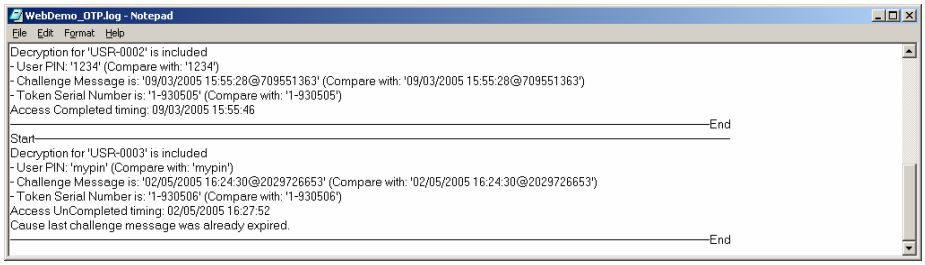
- One-Time Password is generated on token application
- Client user has to bring this OTP using on next step.



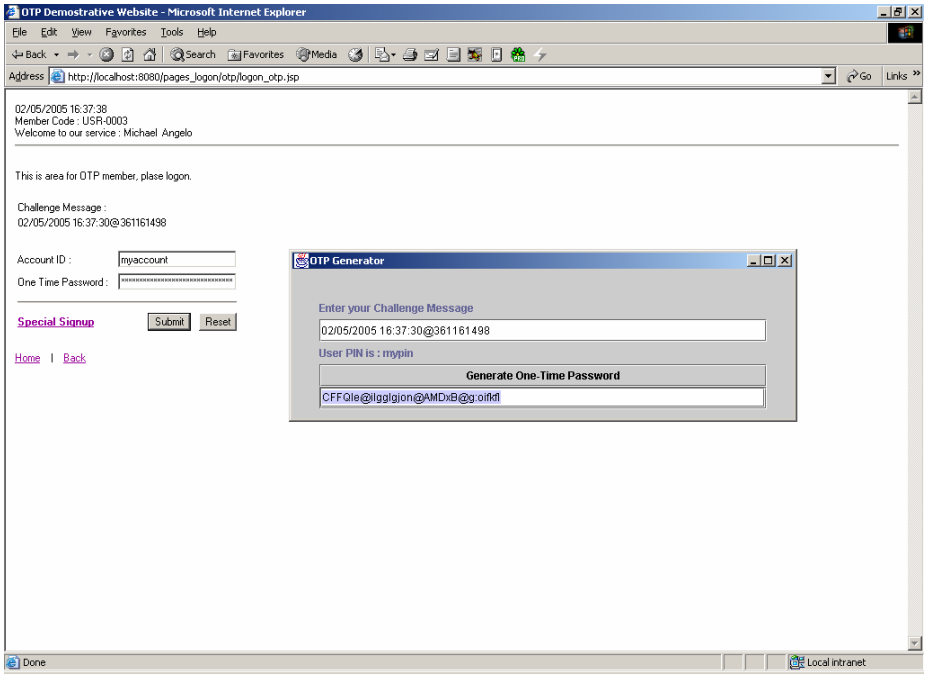
- One-Time Password is entered to the text box and then information is submitted to the authentication system by pressing on Submit button.



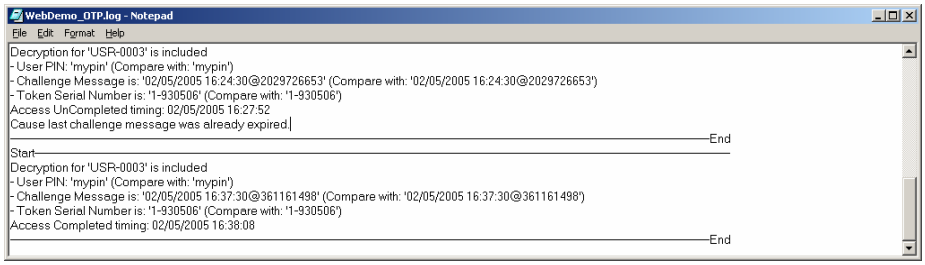
- In Case entered data is invalid the OTP logon page will be reloaded with alert messages and waiting for valid information
- Client user has to reprocess since get new challenge message until resubmit the One-Time Password to the system.



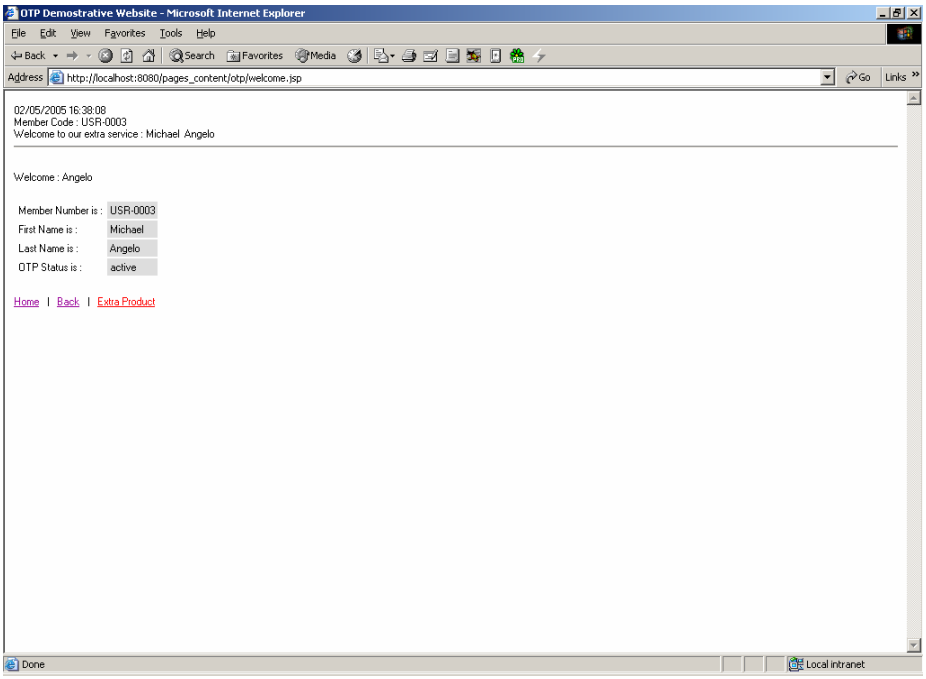
- On server side, administrator can open log file to see the result of authentication process.
- In this sample log file show that this request using the expired challenge message.



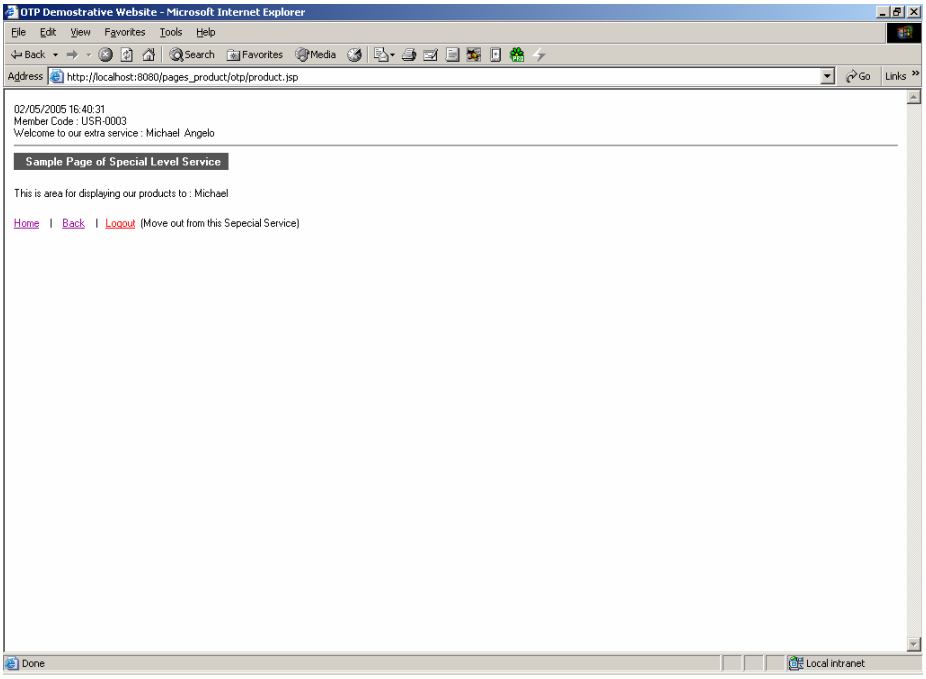
- Reprocess for special authentication by using One-Time Password



- On server side, administrator can open log file to see the result of authentication process.
- In this sample log file show that this authentication is already completed.



- These information, Account ID and One-Time Password, were verified before system allow OTP client user access to special greeting page, “http://locclhost:8080/pages_content/otp/welcom.jsp”
- Access to special page of extra service by click link “Extra Product”



- The sample page of extra level service, http://localhost:8080/pages_product/otp/product.jsp is opened

BIOGRAPHY

NAME Mr.Anusorn Kanyaprasankid

DATE OF BIRTH 10 December 1973

PLACE OF BIRTH Bangkok, Thailand

INSTITUTIONS ATTENDED Thammasat University, 1997:
Bachelor of Psychology
Mahidol University, 2005:
Master of Science (Technology of Information System Management)

POSITION&OFFICE Application Engineer
Locus Telecommunication Inc.,
LTD. Thailand
Tel. +66(0)-2989-3400
E-mail: anusorn@locus.co.th

HOME ADDRESS 1/2 m.1 Tumbol Napa Amphur Maung
Chonburi Thailand
Tel. (038)441-982
Email Address: kanya_mao@yahoo.com