

ACKNOWLEDGMENT

We consider this as a privilege to express a few words of gratitude to all those who guided and inspired us for the successful completion of our project work.

We offer our humble pranams at the lotus feet of His Holiness, **Dr. Shree Shree Shivakumara Swamigalu**, Revered President, Shree Siddaganga Education Society, Shree Siddaganga Math for bestowing upon his blessings.

We deem it as a privilege to thank **Dr. Shivakumaraiah**, Principal, S.I.T Tumakuru, for fostering an excellent academic environment in this institution, which made this endeavor fruitful.

We would like to express our sincere gratitude to **Dr. N.R. Sunitha**, Professor and Head, Department of CSE, SIT, Tumakuru for her encouragement and valuable suggestions.

With profound sense of gratitude, we acknowledge the guidance and support extended by **Dr. R Aparna**, Associate Professor, Department of Computer Science & Engineering, S.I.T. Tumakuru. Her incessant encouragement and invaluable technical support have been of immense help in realizing this project. Her guidance gave us the environment to enhance our knowledge, skills and to reach the pinnacle with sheer determination, dedication and hard work.

Above all, we would like to express thanks to our **PARENTS** for their support all along.

With regards,

RASHMI A C
SANDHYA M
VIJAY S YALASANGIMATH
VINAY KUMAR B M

May 2015

ABSTRACT

E-Cash is a concept of electronic cash which would allow users to carry money in the form of digital coins. Transaction can be done both offline and online in the absence of a third party/financial institution. This project proposes an offline model which supports multiple usage of transferable E-Coin. The protocol is based on RSA and line equation on 2D plane. The RSA cryptosystem is used to avoid counterfeiting of E-Coins. Each coin is appended with user history in the form of a point on line i.e., (x, y) which is used to find fraud user. Each user will have unique 'm' value and also each coin will have unique 'c' value, which is received from trusted third party only at the time of withdrawing the coin from E-Cash Company. Line equation $y = mx + c$ is formed by each user with 'm' and 'c' values. At the time of spending the coin, user appends a point on his line, to the coin as user history. This user history part of the coin prevents double spending and preserves user anonymity.

CONTENTS

ABSTRACT.....	2
1. INTRODUCTION	8
1.1 Introduction to Digital Cash.....	8
1.2 Problem statement	9
1.3 Objectives of the project	9
1.4 Scope of the project.....	11
1.5 Summary	11
2. LITERATURE SURVEY	12
2.1 General structure of E-Cash transactions.....	13
2.2 Key elements of a private E-Cash system.....	14
2.3 RSA.....	16
2.4 Principles of RSA	17
2.5 Security of RSA	17
2.6 Attacks against RSA	18
2.7 RSA and Standards	18
2.8 RSA and Patents	19
2.9 Summary	19
3. SYSTEM REQUIREMENTS SPECIFICATION	20
3.1 Requirements overview	20
3.2 Software requirements and hardware requirements.....	21
3.3 Summary	22
4. SYSTEM ANALYSIS	23
4.1 Existing system	23
4.2 Proposed system.....	26
4.3 Software Process model employed	29

4.4 Summary	30
5. E-CASH ALGORITHMS	31
5.1 User account creation.....	31
5.2 Description of coin.....	31
5.3 Verification of coin	32
5.4 User history	33
5.5 Withdrawal protocol	33
5.6 Spending protocol	34
5.7 After n – transactions	34
5.8 Deposit protocol.....	35
5.9 Double spending	35
5.10 Anonymity of user	36
5.11 Summary	36
6. DETAILED DESIGN	37
6.1 Class diagrams	37
6.2 Activity Diagrams	38
7. SYSTEM IMPLEMENTATION	43
7.1 Implementation platform	43
7.2 Implementation language.....	44
7.3 Actual implementation.....	44
7.4 Summary	45
8. TESTING.....	46
8.1 Testing objectives	46
8.2 Test cases	46
8.3 Unit testing.....	47
8.4 Integration testing	49
8.5 System testing	49

8.6 Acceptance testing	51
8.7 Summary	51
9. SNAPSHOTS.....	52
9.1 E-cash Server Side	52
9.2 E-cash application snapshots	56
9.3 Summary	58
10. CONCLUSION AND FUTURE ENHANCEMENTS	59
10.1 Conclusion	59
10.2 Future enhancement	60

LIST OF FIGURES

Figure 2.1	Life-cycle of electronic coins	13
Figure 2.2	Life-cycle of a transferable coi.....	15
Figure 4.1	NetCash scheme	25
Figure 4.2	Line equation.....	27
Figure 4.3	Normal users equations	28
Figure 4.4	Fraud users equations	28
Figure 4.3	Prototyping Model.....	29
Figure 6.1:	BankAccount class	37
Figure 6.2:	Class diagram	38
Figure 6.3:	Activity diagram of user account creation	39
Figure 6.4:	Activity diagram showing the workflow of the withdraw activity	40
Figure 6.5:	Activity diagram presenting the spend activity	41
Figure 6.6:	Activity diagram of the process of depositing of E-cash by user to the E-cash company.....	42
Figure 7.1:	System architecture of E-Cash System.	45
Figure 9.1:	Snap shot of home page of E-cash Company.....	52
Figure 9.2:	Snap shot of admin login page of E-cash Company	52
Figure 9.3:	Snap shot of E-cash registration form page for registering user to obtain E-cash facility	53
Figure 9.4:	Snap shot of the coins that have been withdrawn from E-cash Company but not yet deposited	53
Figure 9.5:	Snap shot of the coins that have been deposited to the E-cash Company by user	54
Figure 9.6:	Snap shot showing the list of registered users of E-cash Company	54
Figure 9.7:	Snap shot detailed user list who are double spent the coin	55
Figure 9.8:	Snapshot of customer login page in E-wallet application for doing transactions by providing customer login id and password provided by E-cash Company	56

Figure 9.9: Snapshot of list of options provided in E-wallet application for user to do transactions. It includes <i>withdrawal</i> of money form E-cash Company, <i>spending</i> of e-coin, <i>transaction history</i> of customer, <i>depositing</i> of e-coin	56
Figure 9.10: Snapshot of available coins they can used for transaction and they are provided by E-cash Company on withdrawal of money.	57
Figure 9.11: Snapshot of <i>deposition phase</i> of e-coin. Here we need to choose the number of available coins for deposition.	57
Figure 9.12: Snapshot of <i>spending phase</i> of e-coin. Here it will display the number of available coins for spending in terms of denominations. We can also intimate the receiver of the coin through sending an SMS by entering (or browsing from contact list of phone) mobile number of receiver.	58
Figure 9.13: Snapshot of history of transactions details done by the customer.....	58

LIST OF TABLES

Table 8.1: Test case description showing unit testing of E-cash	47
Table 8.2: Test case description showing Integration testing of E-cash.....	49
Table 8.3: Test case description showing System testing of E-cash.....	50

1. INTRODUCTION

1.1 Introduction to Digital Cash

Digital cash is a way to implement anonymous electronic payments in an environment of mutual mistrust between the bank and users. The present day payment systems fall into two huge categories: account-based systems and token-based systems. Token-based systems such as paper cash, pre-paid phone cards or mail stamps, do not identify its users. A pre-paid phone card, for example, does not distinguish one caller from the other. Account-based systems such as checks, credit cards or bank accounts need to identify the system users and their transactions, by design.

People would like to use paper cash because it is easy to carry around, they can make a payment with the received cash and they don't need to ask a third party like a bank to perform their payments. Paper cash can, however, be stolen or lost and no one compensates for the lost or stolen money.

Credit cards secure people from the risk of losing their cash, but by using electronic money people are in the risk of losing their privacy. Annually, credit card companies and banks lose large sums of money since they need to compensate for lost cards and the costs associated with the fraud and human error. In light of the hotheaded increase of electronic services such as Internet, the need for more efficient mode of electronic payments has become crucial.

Since anonymity of payments is usually associated with anonymity of paper cash, an anonymous token-based electronic payment system is referred to as digital cash (also known as electronic cash, e-cash, D-cash). Electronic cash (E-Cash) offers a solution to the problems of paper cash and today's credit cards; it is secure and protects people's privacy. The customer can use digital cash to pay over the Internet without the involvement of a bank during their payments.

Ideal properties of an E-Cash system: Security, Anonymity, Portability, Transferability, Divisibility and Off-line Payment.

1.2 Problem statement

E-Cash aims to mimic the functionality of paper cash by providing properties such as anonymity and transferability of payment. It is intended to be implemented as data which can be copied, stored or given as payment (for example: attached to an email, SMS or via a USB stick, Bluetooth, etc.). Just like paper currency and coins, E-Cash is envisioned to represent a value because it is backed by a trusted third party (namely, the government and the banking industry). Much money is already cast-off in the electronic form; for example, by credit, debit card, direct transfer between accounts or by on-line services such as PayPal. This kind of electronic money is not E-Cash, because it doesn't have the properties of cash (namely, anonymity and off-line transferability between holders).

1.3 Objectives of the project

Following are the objectives of the E-cash system:

- A: Security: The transaction protocol must ensure that a high-level security is maintained through sophisticated encryption techniques. For instance, Alice should be able to pass E-Cash to Bob without either of them, or others, able to alter or reproduce the electronic token.
- B: Anonymity: Anonymity assures the privacy of a transaction on multiple levels. Alice should be able to pay Bob without revealing her identity, and without Bob revealing his identity. Moreover, the Bank should not be able to identify the details of the payer or the payee.
- C: Portability: The security and use of the E-Cash is not dependent on any physical location. The cash can be transferred through computer networks and off the computer network into other storage devices. Alice and Bob should be able to walk away with their E-Cash and transport it for use within alternative delivery systems, including non-computer-network delivery channels. Digital wealth should not be restricted to a unique, proprietary computer network.
- D: Two-way: The E-Cash can be transferred to other users. Essentially, peer-to-peer payments are possible without either party required to attain registered merchant status as with today's card-based systems. Alice, Bob, Carol, and David share an elaborate dinner together at a trendy restaurant and Alice pays the bill in full. Bob,

Carol, and David each should then be able to transfer one-fourth of the total amount in E-Cash to Alice.

- E: Off-line capability: The protocol between the two exchanging parties is executed off-line, meaning that neither is required to be host-connected in order to process. Availability must be unrestricted. Alice can freely pass value to Bob at any time of day without requiring third-party authentication.
- F: Divisibility: An E-Cash token in a given amount can be subdivided into smaller pieces of cash in smaller amounts. The cash must be fungible so that reasonable portions of change can be made. Alice and Bob should be able to approach a provider or exchange house and request E-Cash breakdowns into the smallest possible units. The smaller the breakdowns are the better to enable high quantities of small-value transactions.
- G: Infinite duration: The E-Cash does not expire. It maintains value until lost or destroyed provided that the issuer has not debased the unit to nothing or gone out of business. Alice should be able to store a token somewhere safe for ten or twenty years and then retrieve it for use.
- H: Wide acceptability: The E-Cash is well-known and accepted in a large commercial zone. Primarily a brand issue, this feature implies recognition of and trust in the issuer. With several E-Cash providers displaying wide acceptability, Alice should be able to use her preferred unit in more than just a restricted local setting.
- I: User-friendliness: The E-Cash should be simple to use from both the spending perspective and the receiving perspective. Simplicity leads to mass use and mass use leads to wide acceptability. Alice and Bob should not require an advanced degree in cryptography as the protocol machinations should be transparent to the immediate user.
- J: Unit-of-value or monetary freedom: Another important need is that the E-Cash is denominated in market-determined, non-political monetary units. Alice and Bob should be able to issue non-political E-Cash denominated in any defined unit which competes with governmental-unit E-Cash.

1.4 Scope of the project

E-Cash brings with it clear advantages over traditional money. For the user, electronic money is precise, efficient, green (paperless), cost effective (no wear & tear replacement and cash logistics cost) and convenient. It can take advantage of reward program for small purchases and accrue shopping savings over the year. Its existence opens up new business opportunities, especially for small businesses in e-commerce space. For banks, it could mean the elimination of thousands of paper transactions and, in turn, the reduction in user fees. For corporations, it could mean the ability to circumvent banks entirely to create direct company to company transfers.

We can transfer funds, purchase stocks and offer a variety of other services without having to handle physical cash or cheques as long as bank is providing such services online. The significant effect is, we do not have to queue in lines, thus saving our time. Consumers will have greater privacy when shopping on the Internet using electronic money instead of ordinary credit cards.

1.5 Summary

This chapter provides a brief description about the ground rules of the E-Cash system, a transitory overview of the existing systems and the driving force toward our proposed system. In the next chapter, we describe the Literature survey for the project.

2. LITERATURE SURVEY

There have been predictions of the elimination of physical cash as a transaction medium and the substitution of one form or another of an electronic payments system for more than a decade now.

Every time a person makes a purchase with his credit card, every time he makes a telephone call, or pay his taxes that information goes into a data base somewhere. Furthermore, all these records can be linked so that they constitute in effect a single evidence on his life not only your medical and financial history but also what he buys, where he travels and whom he communicates with. It is almost impossible to learn the full extent of the files that various organizations keep about a person, much less to assure their accuracy or to control who can gain access to them. Often, organizations link records from different sources for their own protection. As we all know, banks look at these records to grant loans, or issue credit cards to people. This information lets a financial institution know of a person's financial history, his ability to pay off the loans. That same information in the wrong hands, however, provides neither protection for businesses nor better service for consumers. Thieves routinely use a stolen credit card number to trade on their victims' good payment records; murderers have tracked down their targets by consulting government-maintained address records. The more information that organizations have (whether the intent is to protect them from fraud or simply to target marketing efforts), the less privacy and control people retain.

Because of these concerns, analysts have shown interest in developing mechanisms that provide electronic privacy. This can be made possible if customers can pay for their purchases with untraceable E-Cash or present digital credentials that serve the function of a banking passbook, driver's license or voter registration card without revealing their identity.

Chih-Hung Wang and Wei-Ming Chiang [1], they discussed e-cash is a kind of network payment; however, although various types of demands in e-cash namely anonymity, un-traceability, reusability so on, have been conceived of, an e-cash scheme with the fairness property has not yet been proposed. The e-cash scheme discussed here properly combines the payment protocol with fair exchange procedure, so that customer and the

merchant can fairly exchange their money and goods, with the aid of an off-line trusted third party (off-line TTP).

E-Cash was introduced by Chaum [3]. Chaum used a blind signature to provide “anonymity” and “un-traceability” for E-Cash.

Brands in 1993 provide an untraceable E-Cash scheme in wallet with observers [11]. In his paper, he proposed that a coin can be traced only if double-spending occurs.

Song and Korba [8] describe how to construct an E-Cash scheme which has both features of “un-traceability” and “non-repudiation”. This means it can be ensured that a legal user would never be traced; however, once a dispute happens, a user cannot deny that he had spent the E-Cash.

2.1 General structure of E-Cash transactions

There are three different types of transactions during an E-Cash procedure and the transactions are as shown in Figure 2.1.

- a) *Withdrawal*, in which Alice transfers some of her money from her bank account to her wallet (it could be a smart card or a personal computer).
- b) *Payment*, in which Alice transfers money from her wallet to Bob's.
- c) *Deposit*, in which Bob transfers the money he has received to his bank account.

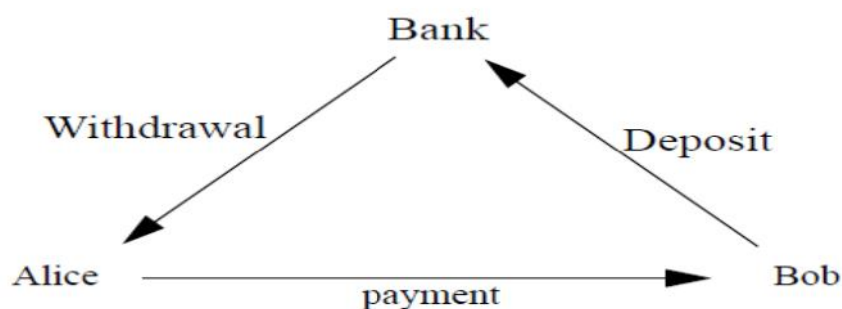


Figure 2.1 Life-cycle of electronic coins

In an E-Cash system, we have three kinds of actors:

- A financial network (The bank).

- A payer or consumer (Alice).
- A payee or a shop (Bob).

2.2 Key elements of a private E-Cash system

As would-be currency providers should note, there are ten key elements to a successful, private E-Cash system. This section compares and contrasts true E-Cash to paper cash as we know it today. An E-Cash token is assumed to be used in the system in any transaction.

Security: The transaction protocol must ensure that a high-level security is maintained through sophisticated encryption techniques. For instance, Alice should be able to pass E-Cash to Bob without either of them, or others, able to alter or reproduce the electronic token.

Anonymity: Anonymity assures the privacy of a transaction on multiple levels. Beyond encryption, this optional un-traceability feature of E-Cash promises to be one of the major points of competition as well as controversy between the various providers. Transactional privacy will also be at the heart of the government's attack on E-Cash because it is that feature which will most likely render current legal tender irrelevant. Both Alice and Bob should have the option to remain anonymous in relation to the payment. Furthermore, at the second level, they should have the option to remain completely invisible to the mere existence of a payment on their behalf.

Portability: The security and use of the E-Cash is not dependent on any physical location. The cash can be transferred through computer networks and off the computer network into other storage devices. Alice and Bob should be able to walk away with their E-Cash and transport it for use within alternative delivery systems, including non-computer-network delivery channels. Digital wealth should not be restricted to a unique, proprietary computer network.

Two-way payments: The E-Cash can be transferred to other users. Essentially, peer-to-peer payments are possible without either party required to attain registered merchant status as with today's card-based systems. Alice, Bob, Carol, and David share an elaborate dinner together at a trendy restaurant and Alice pays the bill in full. Bob, Carol, and David each should then be able to transfer one-fourth of the total amount in E-Cash to Alice.

Off-line capability: The protocol between the two exchanging parties is executed off-line, meaning that neither is required to be host-connected in order to process. Availability must be unrestricted. Alice can freely pass value to Bob at any time of day without requiring third-party authentication.

Transferability: Transferability (fig 2.2) allows a user to spend a coin that he has received in a payment without having to contact the bank. A payment is transfer if the payee can use the received coin in a payment. A payment system is transferable if it allows at least one transfer per coin. We have to notice that the ability to transfer paper cash is very important in our daily life. The life-cycle of an electronic coin in a transferable system looks like:

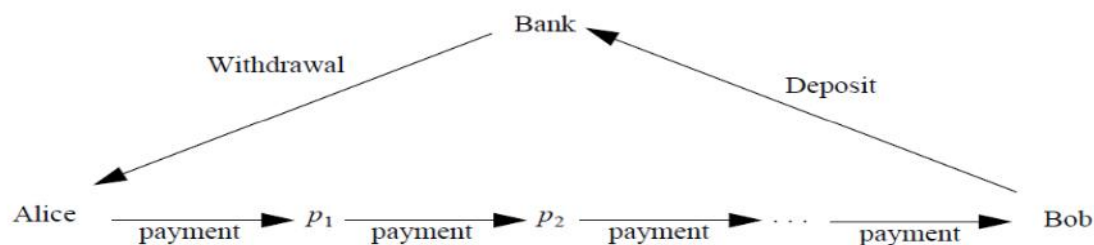


Figure 2.2 Life-cycle of a transferable coin

Divisibility: An E-Cash token in a given amount can be subdivided into smaller pieces of cash in smaller amounts. The cash must be fungible so that reasonable portions of change can be made. Alice and Bob should be able to approach a provider or exchange house and request E-Cash breakdowns into the smallest possible units. The smaller the breakdowns are the better to enable high quantities of small-value transactions.

Infinite duration: The E-Cash does not expire. It maintains value until lost or destroyed provided that the issuer has not lowered the unit to nothing or gone out of business. Alice should be able to store a token somewhere safe for ten or twenty years and then retrieve it for use.

Wide acceptability: The E-Cash is well-known and accepted in a large commercial zone. Primarily a brand issue, this feature implies recognition of and trust in the issuer. With several E-Cash providers displaying wide acceptability, Alice should be able to use her preferred unit in more than just a restricted local setting.

User-friendliness: The E-Cash should be simple to use from both the spending perspective and the receiving perspective. Simplicity leads to mass use and mass use leads to wide acceptability. Alice and Bob should not require an advanced degree in cryptography as the protocol machinations should be transparent to the immediate user.

There are two types of E-Cash schemes:

- On-line: Validity of the transaction is checked while it is occurring. The coin is sent back to the bank or similar authority during the transaction to verify authenticity of coin and that it was not spent before. The advantage is that the bank can check and prevent illegal operations as they are happening unlike the case in off-line systems.
- Off-line: validity of the transaction is checked after the transaction has occurred. The merchant or bank can conduct a series of calculation to reveal the customer's identity when a security breach has occurred.

In general off-line schemes are more efficient than on-line ones. The two fundamental issues with any off-line E-Cash scheme have been the detection of double spending and provision of anonymity.

One of the first techniques that was introduced to address the issue of double spending in an offline scheme was the Cut-and-Choose technology. In this technique, a prover answers a series of random challenges that establish with high portability that the prover is honestly following the defined protocol. However, it is not very efficient. Subsequently, other techniques have been proposed to address both the problems without the Cut-and-Choose method.

2.3 RSA

In a public-key algorithm, the keys are formed in a pair of an encryption and a decryption key and it is infeasible to generate one key from the other. The algorithm keeps one key secret and sends the other to the partner over an open canal. To date many public-key cryptography algorithms have been proposed. Many of them are impractical others insecure. Only a few algorithms are both secure and practical and of these algorithms.

A digital signature is a mathematical technique used to validate the authenticity and integrity of a message, software, or a digital document. Just few of the existing algorithms work for both encryption and digital signatures. RSA is by far the easiest Public-Key algorithm to understand and implement. It was introduced in 1978 by Rivest, Shamir and Adleman and it works for encryption as well as for digital signatures [10].

2.4 Principles of RSA

RSA relies upon modulo arithmetic. Both encryption and decryption are completed by raising numbers to a power modulo, a number which is product of two large primes (at least 100 to 200 digits). To encode a message using RSA, a user needs a pair of keys. The key generation scheme in RSA works as shown below.

p, q : Two large prime numbers. For maximum security, these should be of equal length.

n : The public key, obtained as the product of p and q .

e : The public key, a randomly chosen number which is less than n and relatively prime to $(p - 1)(q - 1)$. e has no factors in common with $(p - 1)$ and $(q - 1)$.

d : The private key, obtained as the inverse of $(e \bmod (p - 1)(q - 1))$ such that $(ed = 1 \bmod (p - 1)(q - 1))$.

The two large prime factors p and q must be kept secret or destroyed, because if anyone could factor n into p and q , the private key e , could be obtained. Encryption is simple. A message m , is divided into numerical blocks, smaller than n . The encrypted message c , will be made up of similarly sized message blocks of about the same length.

Encryption formula is: $c = m^e \bmod n$

Decryption formula is: $m = c^d \bmod n$

2.5 Security of RSA

RSA gains its security from the difficulty of factoring large prime numbers. Recovering plaintext from the corresponding cipher text, given the public-keys is equivalent to problem of factoring, depends on length. A 40 bit RSA key could probably be broken with no calculator. That is only digits and factoring such a small number is not difficult (tables of

all primes up to million exist). The best size for an RSA key depends on one's security needs. Larger the key is, greater would be the security, but slower the RSA operations would become. One should choose a key length upon considerations. First, of one's security needs, such as the value of the protected data and how long it needs to be protected and second, of how powerful one's potential enemies are. It is believed that 512-bit keys no longer provide sufficient security with the advent of new factoring algorithms and distributed computing. Such keys should not be used after 1997 or 1998. The RSA Laboratories recommended key sizes are now 768 bits for personal use, 1024 bits for corporate use, and 2048 bits for extremely valuable keys.

2.6 Attacks against RSA

There are a few possible interpretations of "breaking RSA". The most damaging would be for an attacker to discover the private key corresponding to a given public key, this would enable the attacker both to read all messages encrypted with the public key and to forge signatures. The obvious way to do this attack is to factor the public key, n , into its two prime factors, p and q . From p , q , and e , the public exponent, the attacker can easily get d , the private exponent. The hard part is factoring n , the security of RSA depends on factoring being difficult. In fact, an easy factoring method would 'break' RSA.

It is also possible to attack RSA by *guessing* the value of d . This attack is not easier than factoring.

Another possibility for an attacker would be to try every possible until the correct one is found. This *brute-force* attack is even less efficient than trying to factor.

2.7 RSA and Standards

RSA has become a part of many standards around the world. For instance, ISO 9796, ITU (was called CCITT) X.509 digital certification standard. It is included in France's ETEBAC 5 standard and Australia key management standard, AS2805.6.3, and the ANSI X9.31 draft standard for U.S. banking industry and RSA is a part of the Society for Worldwide Interbank Financial Telecommunications (SWIFT) standard. RSA is a "de facto" standard in the financial community. The existence of a "de facto" standard is very important to the development of internet commerce. If one public-key system is available everywhere,

then signed digital documents can be exchanged among users in many different nations, around the world, using different software on different platforms. If there is an accepted standard for digital signatures, it is possible to have passports, checks, wills, leases, etc. exists in electronic form and a paper version will be nothing more than a copy of the original electronic document.

2.8 RSA and Patents

RSA is patented under U.S. Patent 4405829, licensed by Public Key Partners (PKP), issued September 29, 1983 and held by RSA Data Security, Inc. of Redwood City, California, the patent expires 17 years after issue, on September 20, 2000. RSA Data Security usually allows free non-commercial use of RSA, with written permission, for academic or university research purposes.

2.9 Summary

In this chapter, description about Literature survey of E-Cash system is discussed. The E-Cash system uses cryptographic tools, namely, asymmetric cryptography to provide security in transactions, encryptions to protect data and digital signature to authenticate the notes. The next chapter will discuss “System Requirements Specification”.

3. SYSTEM REQUIREMENTS SPECIFICATION

3.1 Requirements overview

Here we present a brief outline of functional and non-functional requirements of E-cash system.

3.1.1 Functional Requirements

Sign in: This activity authenticate the user to provide access to the E-Cash Wallet, in return it returns the parameter like 'C' which is required to carry out further transactions.

Withdraw: Activity collects the card details along with the denomination of the E-Cash to be generated. The server will receive the request and generates the requested E-Cash and also place them into database for further processing.

Deposit: This is the counter part of withdraw. This activity submits the E-Cash of the users to the E-Cash Company. The server will validate the E-Cash received along with it also check for double spending of that particular E-Cash received.

Spend: This activity enables the user to transfer the coin to another party by SMS.

3.1.2 Non-Functional Requirements

These are requirements that are not functional in nature, that is, these are constraints within which the system must work.

Performance Requirements

- The Overall performance should be reliable and should enable the users to work efficiently.
- User interface created will have a direct impact on performance, a simple and an easy to use interface will result in a better performance.
- User must be able to access the transactions by a quick and efficient manner. Also database server access should be quick enough for update and retrieval processes.

Safety Requirements

- Since the database is accessed by all the registered users, the access to that should be secure. We can maintain a back-up of the database so that in case of any crash, the back-up copy can be used.

Security Requirements

- Only authenticated users are allowed to login.
- There will be proper security regarding the access of data.
- Database security is the most important security requirement. Any unauthorized access to the database may corrupt the database and may lead to irrevocable loss of data.

Software Quality Attributes

Capacity, scalability and availability represents the quality attributes to be achieved.

- The system shall achieve high availability at all times.
- The system shall be scalable to support additional clients.

3.1.3 User Requirements

- The users should ensure proper registration process by sending the account number and password to their email.
- The database access for withdrawal and depositing of coin should be instantaneous for the registered users.

3.2 Software requirements and hardware requirements

Following are the software and hardware requirements for our E-cash project:

3.2.1 Software Requirements

- Software: Eclipse with ADT, Php designer.
- Operating System: Windows and Android.

3.2.2 Hardware Requirements

- Hard disk: 40GB HDD.
- RAM: 1GB RAM.
- Processor: Core 2 duo processor.

3.3 Summary

In this chapter, description about software requirements specification is discussed. The next chapter discusses about the system analysis of our work on E-cash.

4. SYSTEM ANALYSIS

4.1 Existing system

Here we provide few examples of the present-day scenario of e-cash on the internet. Some of them are InternetCash, DigiCash, NetCash, CyberCash, NetBill, First Virtual and PayPal.

4.1.1 DigiCash

DigiCash was founded in Amsterdam by David Chaum in 1990. One of DigiCash products is E-Cash; it is an online payment system over email or internet based on Chaum's E-Cash system using blind signatures.

To use E-Cash, every user opens an account with a digital bank on the internet which issues the coins for them. The E-Cash software (cyber wallet) issues an asymmetric key for each user based on RSA.

For Alice to withdraw cash, she needs to state how much money she needs. The software generates a random serial number, which is usually of 100 digits, for the coin, a blinding factor and sends it to the bank.

The bank verifies the message to make sure that it was signed by Alice, signs it and debits Alice account.

Alice unbinds the coins and stores them on her PC. When Alice wants to buy something from Bob, she sends him the coins. Bob sends the coins to the bank to verify the authenticity of them and that they have not been spent before.

DigiCash's advantages are anonymity for customers and the possibility of recovering lost coins by giving the bank their serial numbers.

Disadvantages of this system are that merchants must reveal their identity to the bank for cashing the coins and that both of them and their customers must open accounts at the same bank. Also maintaining a database for spent coins is a major problem because it can become very large and unmanageable. Possible attacks are man in the middle attack and

interception attack, since the bank sends account numbers and passwords to users via unencrypted email messages.

4.1.2 NetCash

NetCash was developed at the Information Sciences Institute of the University of Southern California. It uses identified online e-cash.

The system (fig 4.1) consists of buyers, merchants, and currency servers. The currency server issues the coins; each coin is signed by server's private key and consists of:

- Currency Server Name
- Currency Server Network Address
- Expiry Date
- Serial Number
- Coin Value

The currency servers do not keep records of coin holders. Coin holders can exchange coins between different currency servers. The currency servers prevent double spending by keeping a record of only valid and unspent coins.

When Alice wants to buy something from Bob, she sends him the coins, identifier of the merchandise, a new secret key, and her public key all encrypted by Bob's public key. Bob verifies the coins by sending them to the issuing currency server along with a new secret key and type of transaction encrypted by the server's public key. The currency server checks that the coins are valid and are in its database. Then it exchanges them for new coins and sends the new coins to Bob encrypted with the secret key sent by Bob. Bob then sends a receipt to Alice signed with his private key and encrypted with their secret key.

This scheme does not protect Alice from fraud. Bob can spend the coins without sending Alice any receipt. Extensions to the protocol solve this problem and provide additional features such as anonymity and an offline scheme.

Net Cash's advantages are that it is secure and scalable, but the lack of anonymity and extensive use of session keys which slows down its working are its disadvantages.

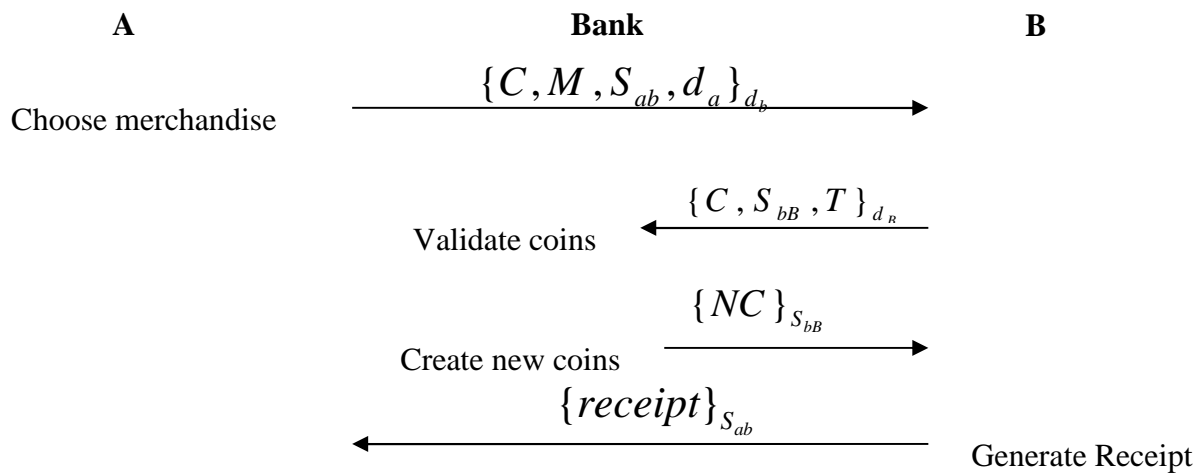


Figure 4.1 NetCash scheme

4.1.3 InternetCash

InternetCash gives customers a chance to pay for their shopping on the internet with cash instead of credit cards. It uses digital signatures and RSA.

Customers buy prepaid cards from any store and go online to activate the card by entering a 20 digit number provided on the back of the card and create a PIN for themselves. After the customer finishes shopping, a secure browser window is opened to request for his PIN. The merchant sends the PIN to the InternetCash server to validate the card along with the payment request. After the InternetCash server validates the card, it deducts the amount from the card and credits to the merchant.

InternetCash cards are comprised because of the following:

- *The Card ID (CID)* : public nine alphanumeric (base 32) digits.
- *The Card Secret Code (CSC)*: public eleven alphanumeric (base 32) digits. The *CSC* is a keyed hash function of the truncated *CID* based on SHA-1 and InternetCash secret key.
- *A secret PIN*: used for additional security in case the *CID* and *CSC* are compromised.

The concatenation of *CID* and *CSC* is called the “InternetCash card number” and it is twenty alphanumeric digits long.

Issuing Protocol

Following explains the methodology used in Internet cash system to issue E-cash to its user:

1. Alice is given an InternetCash card number over an encrypted channel with only the InternetCash server being authenticated or by buying a card from any retail store (over SSL or TLS).
2. Alice chooses a *PIN* over an encrypted channel with only the InternetCash server being authenticated.

Payment Protocol

Consists of a secret key digital signature of the payment information based on *CSC* and *PIN*. The generated signature is called the Payment Authentication Number (*PAN*). It uses a keyed hash function based on SHA-1. The user's *CID* and the *PAN* are sent to Bob over encrypted channel to eliminate eavesdropping.

Clearing Protocol

Bob forwards the payment data (amount, time/date, etc.), the *CID* and the *PAN* to InternetCash over a secure and authenticated channel. InternetCash recreates the *PAN* from the payment data and the *CID*. It is then compared with the received *PAN* and debits Alice's account and credits Bob's account. InternetCash is anonymous and secure, but maintaining a huge database for the cards acts as a disadvantage.

4.2 Proposed system

In our system we address the issues of E-cash system with the help of line equation on 2D plane.

Prerequisites:

Line equation (fig 4.2) on 2D plane will be in the form of $y = mx + b$.

Where,

m : Slope of the line with respect to x-axis

b:Y-axis intercept of line.

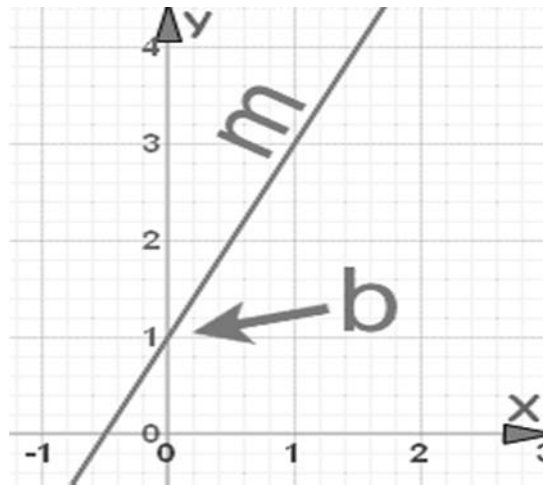


Figure 4.2 Line equation.

m (slope) of line can also be represented in \tan of angle made by line with x-axis. We know that range of $\tan(\theta) \in \llbracket -\infty, +\infty \rrbracket$.

E-Cash system:

In the proposed system, each user will be given a unique number which is 'm'. Coin is described with the following details: Coin Id (A_k), Face value (F_k) and user history ($\{X_{ij}, Y_{ij}\}$). And every coin will be having a unique 'b' value that is received from TTP and is kept hidden from the E-Cash Company.

User History:

While spending a coin, user calculates a point on the line which is formed by $y = mx + b$ where 'm' will be unique to user and b will be unique to coin.

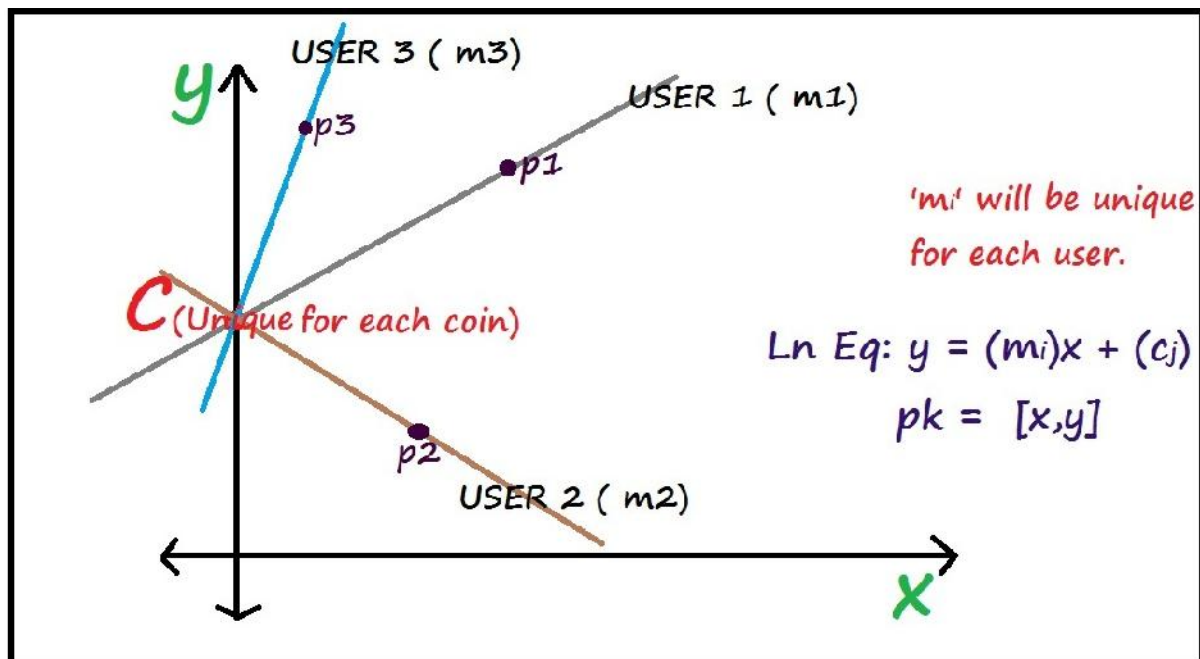


Figure 4.3 Normal users equations

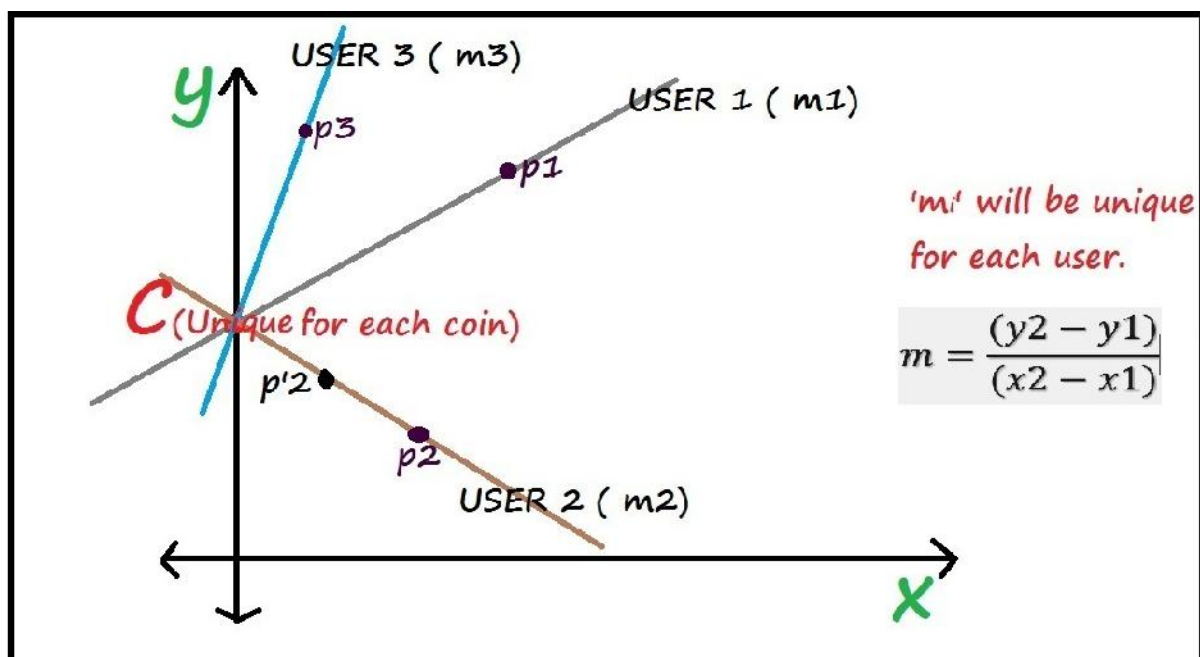


Figure 4.4 Fraud users equations

Handling double spending:

If the user spends same coin again, the new point calculated by user will fall on same line, then E-cash Company will be having two points (fig 4.4) of same on line then company can calculate the slope from those two point to find fraud user with 'm' value.

$$m = -\frac{(y1 - y2)}{(x1 - x2)}$$

Anonymity of user:

If the user does not spend same coin again then company can't find the identity of the user since we cannot find the slope with single point (fig 4.3).

4.3 Software Process model employed

Prototype model:

The basic idea here is that instead of freezing the requirements before a design or coding can proceed, a prototype is built to understand the requirements. It is developed based on the currently known requirements. By using this prototype(fig 4.3), the client can get an “actual feel” of the system, since the interactions with prototype can enable the client to better understand the requirements of the desired system. Prototyping is an attractive idea for complicated and large systems for which there is no manual process or existing system to help determining the requirements. The prototype is usually not a complete system and many of the details are not built in it. The goal is to provide a system which describes overall functionality.

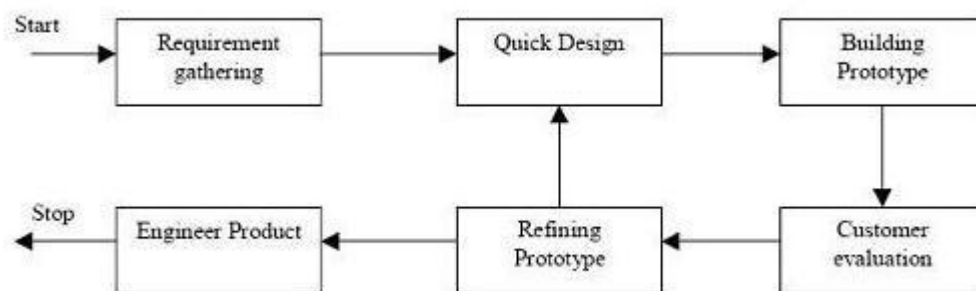


Figure 4.3 Prototyping Model

Cycle 1:

Requirement gathering: Wamp server, Eclipse with ADT, php, java, xml.

Quick Design: AsyncTask is used at the client side to create http client objects to communicate with the server.

Building Prototype: A simple prototype is built in which client sends request to the server. In return, the server responds with an echo message, and that is processed in the client.

Refining Prototype: In the previous prototype, we used http GET method for client and in this, step we shifted to http POST method.

Cycle 2:

Requirement gathering: RSA algorithm, AES (Advanced Encryption Standard) algorithm, MD5 algorithm, Reading the encrypted text from server and decrypting in android client using RSA crypto system, Exchange of RSA keys.

Quick design: We used built-in packages in both the client and the server side to decrypt and encrypt data respectively.

Building Prototype: The previous prototype is extended with classes which used to implement RSA crypto system.

Cycle 3:

Requirement gathering: Mathematical models to be used in finding double spend and user anonymity.

Quick design: To generate unique numbers, we are storing the maximum number used, in the data base. And to generate the new number, we are popping the maximum number used, and that will be used as reference to generate new unique number.

Building Prototype: We used the unique key generation module to implement the mathematical modules.

4.4 Summary

In this chapter, we discussed about the system analysis of the E-cash system. In the next chapter, we describe the algorithms used for the implementation of E-cash system.

5. E-CASH ALGORITHMS

Following section will explain the algorithms used in our system.

5.1 User account creation

Firstly, the user needs to create an account to use the E-cash application. Following algorithm explains the steps it requires:

- Every user creates an account in the bank. During account creation every i^{th} user gets a unique identity/account number I_i .
- The Bank generates a unique value m (slope) for every user's account. The bank sends a copy of this value to the user.
- This value is used for generating user history in order to indicate coin ownership chain during transactions.
- Also, the bank uses m value to obtain the user identity from the user history in case of double spending.

5.2 Description of coin

Following algorithm narrates the description of E-coin and process involved in coin generation:

- In this model every coin has its own identity number. This identity number is used to differentiate between every coin.
- Each coin has a particular face value which represents its denomination. Each k^{th} coin has a unique identity number A_k and a face value F_k .
- Each coin is represented by a unique y-intercept value C , which is generated by a Trusted Third Party (TTP).
- Bank generates a pair of RSA keys as its signature keys. It provides digital signature [5] on the coin using its private key (e, n) . The public key (d, n) is public to users verify the bank's signature and hence the validity of the coin.
- Once the bank generates the coin, this part of the coin becomes immutable. Users can only verify the coin using the public key and signatures.

- Another part of the coin is the user history part. It contains the transaction details of the user.

$$\text{sign}(x) = x^e \bmod n$$

$$[A_k, F_k, \text{sign}(A_k), \text{sign}(A_k \wedge F_k)] \{ \text{User History} \}$$

5.3 Verification of coin

Following are the steps involved in the verification of coin during transactions:

- During transaction the coin is verified against counterfeiting and denomination change.
- The digital signatures, provided by the bank on the coin's identity number and the face value, are checked using the public key of the coin.
 - $((A_k)^e \bmod n)^d \bmod n = A_k$ [From RSA] -----(1)
 - $((A_k \text{ XOR } F_k)^e \bmod n)^d \bmod n = (A_k \text{ XOR } F_k)$ [From RSA]-----(2)
 - $(A_k \text{ XOR } F_k) \text{ XOR } A_k = F_k$ [Verification of denomination] -----(3)

5.3.1 Counterfeiting

If A_k is generated by the counterfeiter, it cannot receive the digital signature $(A_k)^e$ as he does not know the bank private key of the coin and he cannot generate it from the public key of the other coins. The counterfeit coin gets discarded during the verification step (1) and it cannot be generated, provided the bank's private key for the coin is not compromised.

5.3.2 Changing denominations

The counterfeiter can try to change the denomination of the coin by changing F_k . But the scheme prevents such denomination change by using $(A_k \text{ XOR } F_k)^e$ as the mode of verification for F_k value. Let the changed denomination value be F_k' value. The F_k' gets discarded as it cannot pass the verification steps (2 and 3). So the denomination of the coin cannot be changed.

5.4 User history

User history details need to be added to the coin to indicate the chain of transactions and spending ownerships of the coin. Following explains how it is done:

- In this model, the transaction details are stored in the coin. In bit coin, the user transaction details are transmitted as a chain of digitally-signed transactions [11] in a peer-to-peer network for verification of chain of ownerships.
- Here, a similar idea has been proposed, where the user transactions details are added to the coin. The user history consists of two numbers, ' x ' and ' y ' which represent x co-ordinate and y co-ordinate values as a point specifying the spending transaction.
- X and Y are generated by means of the user's unique value ' m ' and coin's unique value ' c '
- The X and Y values that are added into the user history part of the coin helps to identify the fraud user for future verification of double spending of the coin.
- Once stored in the coin, the user details cannot be modified. It can be just viewed by the bank, for verification purposes. $\{x, y\}$

The user uses these values add his details into the user history part of the coin, when he successfully spends the coin. The existing user history cannot be modified by the users. Only new transaction details can be added in it.

5.5 Withdrawal protocol

For a user to make use of the E-cash application to transfer money, he needs to withdraw the required amount from E-cash Company, it is done through following steps:

- The bank generates the coin - (A_k, F_k) . A_k is the coin identity number and F_k is the denomination of the coin.
- It uses the private key (e, n) of the RSA pair to provide digital signature on the coin's contents and provides the public key (d, n) for the verification of the coin.
- The bank signs as follows:
 - $\text{sign}(A_k) = (A_k)^e \bmod n$
 - $\text{sign}(F_k) = (A_k \text{ XOR } F_k)^e \bmod n$
- The bank sends the coin to the user as :

$$[A_k, F_k, \text{sign}(A_k), \text{sign}(A_k \wedge F_k)]$$

- The TTP then sends the coin's unique y-intercept value C to the user, which is later used to generate user history when the user spends the coin.

5.6 Spending protocol

Following explains the mode that is followed by a user in order to spend his e-cash to another party:

- The user spends the coin to a merchant. The merchant verifies the bank's signatures in the coin using the public keys.
- After successful verification, the user generates a random number, 'x' which is used to encrypt the user account details into the coin.
- The user computes $y = mx + c$ and incorporates them in the coin and the transaction takes place. This incorporation denotes the successful spending of the coin by the user.
- These values are used for calculation of m_i in case of double spending.
- The contents of the coin, passed on to the merchant:

$$[A_k, F_k, \text{sign}(A_k), \text{sign}(A_k \wedge F_k)]\{x, y\}$$

5.7 After n – transactions

Following explains the structure of a coin after n transactions of E-coin:

- The coin can undergo any number of transactions before being deposited in the bank. Let us assume the coin underwent n -transactions before being deposited in the bank.
- The coin underwent n receipts and n-1 spending protocols. The contents of the coin after m -transactions :

$$[A_k, F_k, \text{sign}(A_k), \text{sign}(A_k \wedge F_k)]\{x_1, y_1\} \dots \dots \dots \{x_i, y_i\}$$

$$i = 1, 2, 3, \dots$$

5.8 Deposit protocol

Following steps explain the mode of deposit of a coin by the user to the E-cash company to his account:

The n^{th} -user deposits the coin in the bank. He incorporates his details in the coin using the values X_{ij} , Y_{ij} as follows: $\{ X_{nj}, Y_{nj} \}$

- The bank receives the coin and verifies its signature in the coin. It checks for double spending by using the A_k value of the coin.
- The bank checks in its record whether any other coin with the same identity number has been deposited or not.
- If a coin with same identity number has been deposited earlier, then double spending has occurred and the coin is sent for double spending verification protocol.
- If double spending has not been detected then the coin is deposited in the bank and the user history and coin identity number is stored by the bank for double spending verifications against future deposits of coins.

5.9 Double spending

There can be fraud cases where a user spends same E-coin to two or more different parties. Thus, here we explain the way we detect such deceitful users.

- The bank checks in its record for any deposited coin with the same A_k value as this. If it does not exist then double spending has not occurred.
- If another coin with the same A_k value has been deposited earlier, then double spending has occurred.
- If double spending has occurred, then there must be at least one user who has received the coin, copied it and spent it twice.
- That user will have same different X and Y values in the coins. Both the coins will have same coin identity number, denomination and signatures. Only the user history details will vary.
- Let the first coin be:

$$[A_k, F_k, \text{sign}(A_k), \text{sign}(A_k \wedge F_k)] \{x1, y1\} \text{-----} (1)$$

- Let the second coin be:

$$[A_k, F_k, \text{sign}(A_k), \text{sign}(A_k \wedge F_k)]\{x_2, y_2\} \text{-----} (2)$$

- Both the A_k , F_k values for the coins will be same as the coin is received once by the user and spent twice or more. Only the $\{X, Y\}$ values will differ.
- The bank needs to obtain m using $\{X_1, Y_1\}$ and $\{X_2, Y_2\}$ from (4) and (5). It computes m using slope equation as follows:

$$m = \frac{(y_2 - y_1)}{(x_2 - x_1)} \text{-----} (3)$$

- The bank obtains account number I of the double spender by mapping the m value acquired from (3).

5.10 Anonymity of user

E-cash can be synonymous with the paper cash in terms that E-cash also doesn't reveal the confidential credentials of the spender or the receiver.

- The bank cannot obtain the identity number of any user unless that user has double spent a coin.
- The bank cannot find any user details from one record (point) and it cannot identify more than one record of any user.

5.11 Summary

This chapter describes the E-Cash algorithm. The next chapter will discuss the design and implementation details.

6. DETAILED DESIGN

The introduction of structured programming in 1960's and 70's brought with it the concept of Structured Flow Charts. In addition to standard set of symbols, structured flow charts specify conventions for linking the symbols together into a complete flow chart.

The structured programming paradigm evolved from the mathematically proven concept that all problems can be solved using only three types of control structures.

- Sequence
- Decision (Selection)
- Iterative (Looping)

6.1 Class diagrams

The class diagram (fig 6.1) is the main building block in object oriented modelling. It is used both for general conceptual modelling of a systematic application, and for detailed modelling translating the models into programming code. The classes in the class diagram represent both the main objects and or interactions in the application and the objects to be programmed. In the class diagrams, these classes are represented with boxes which contain three parts. For example,

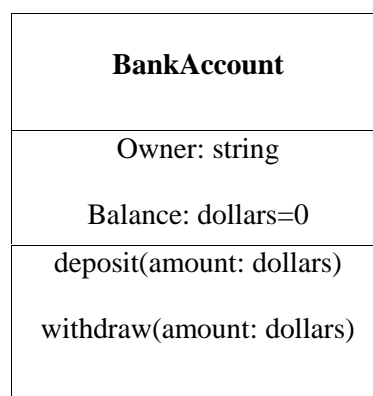


Figure 6.1: BankAccount class

- The upper part holds the name of the class.
- The middle part contains the attributes of the class.
- The bottom part gives the methods or operations the class can undertake.

The classes involved in our application on E-cash (fig 6.2), 'Mobile E-cash Wallet' are included in the package named *company.example.E-Cash*.

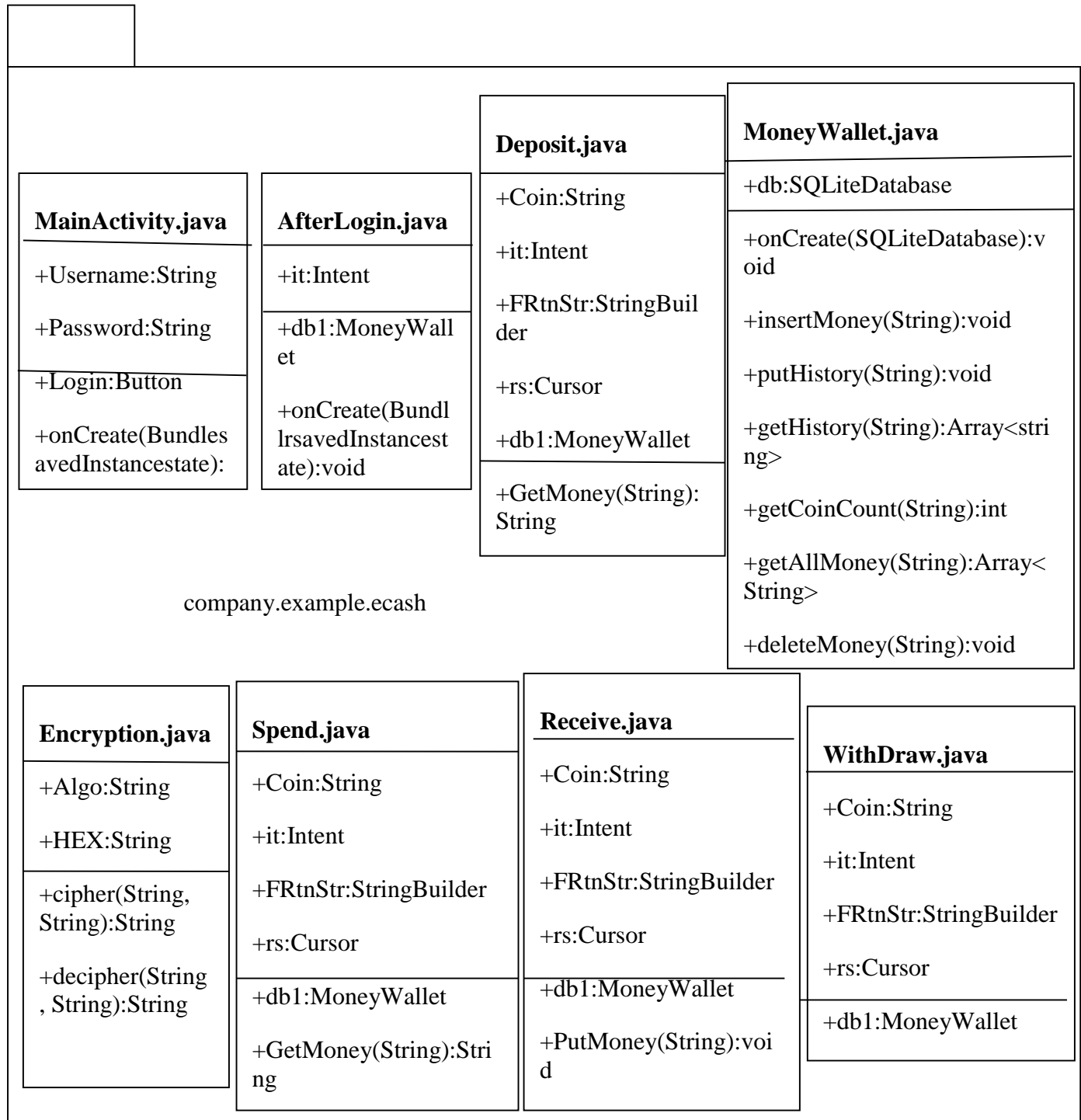


Figure 6.2: Class diagram

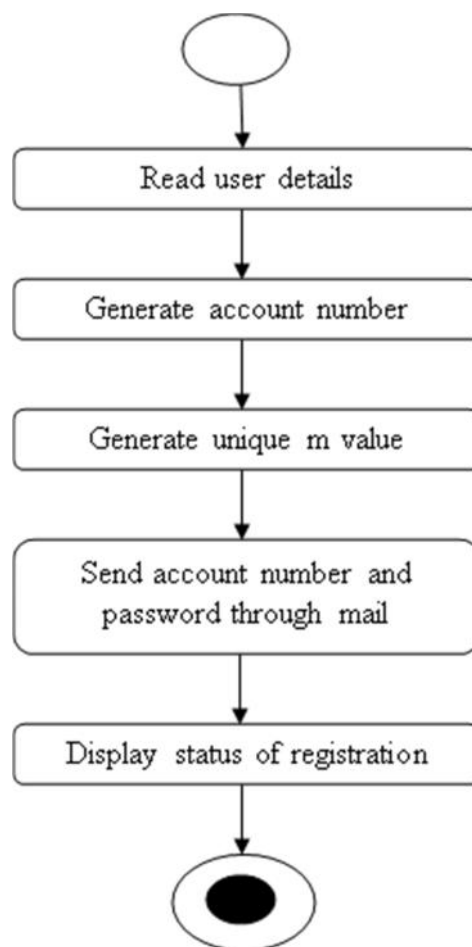


Figure 6.3: Activity diagram of user account creation

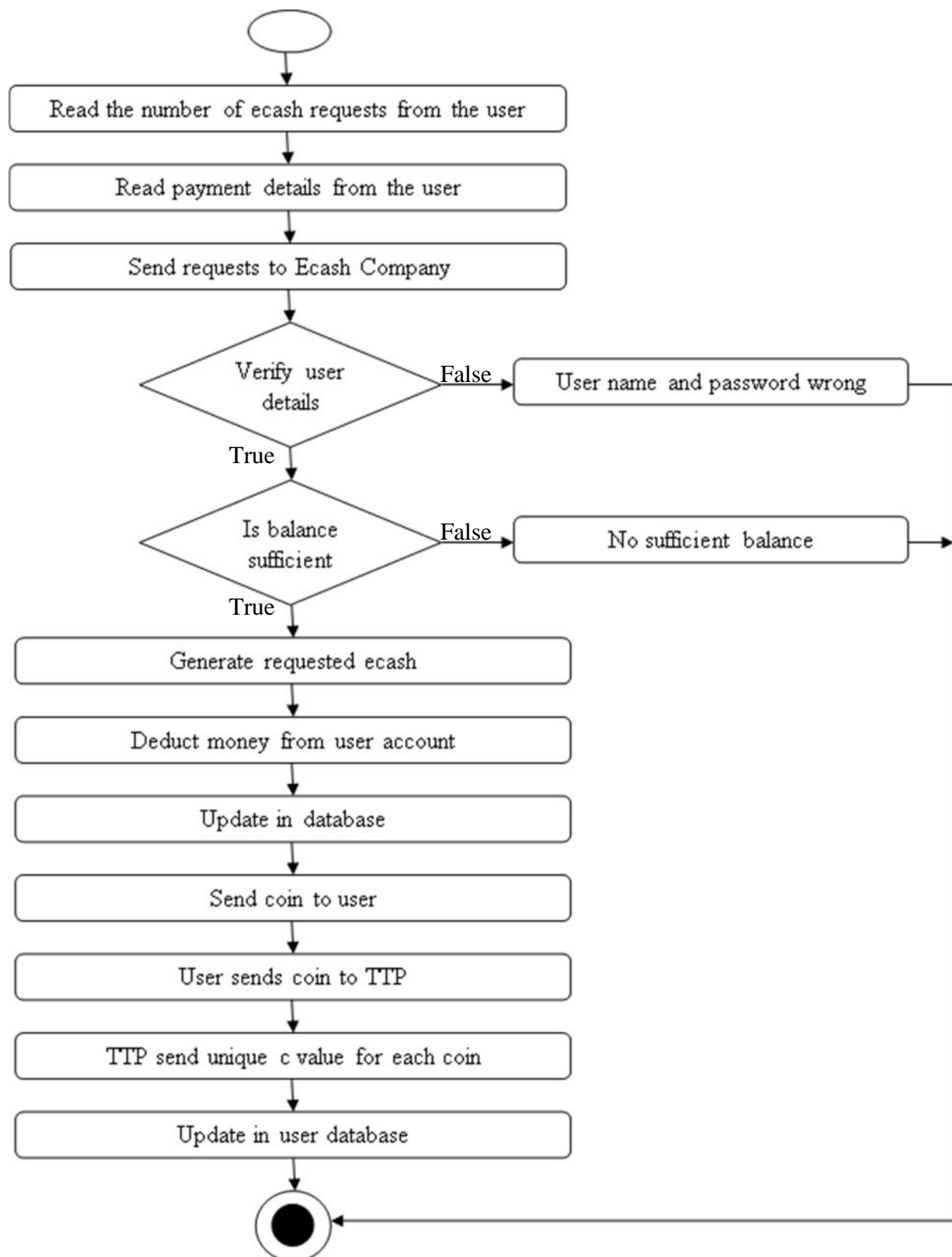


Figure 6.4: Activity diagram showing the workflow of the withdraw activity

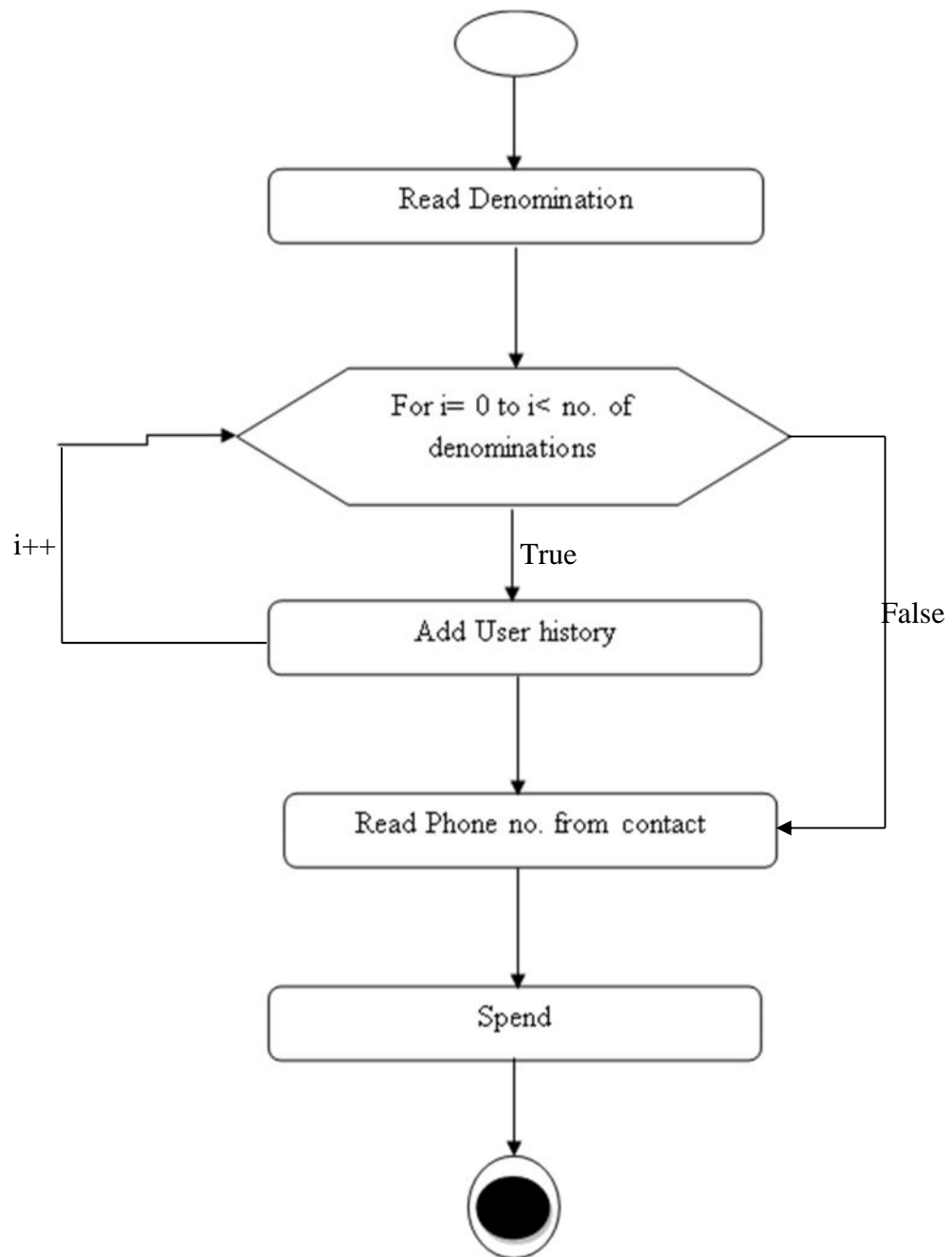


Figure 6.5: Activity diagram presenting the spend activity

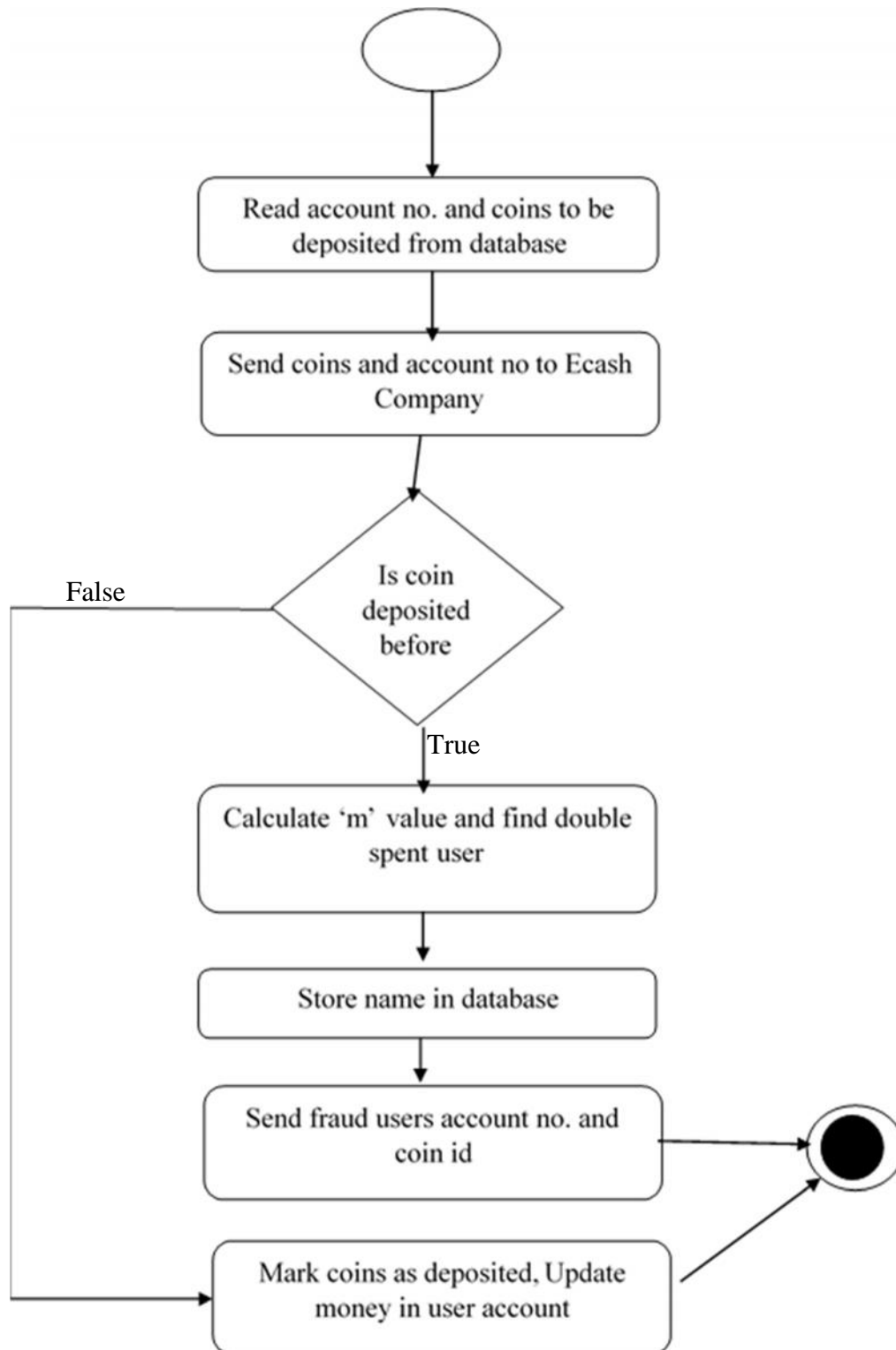


Figure 6.6: Activity diagram of the process of depositing of E-cash by user to the E-cash company

7. SYSTEM IMPLEMENTATION

7.1 Implementation platform

Client: At the client side, we have developed the android application for E-cash.

Android:

Android is an open source and Linux-based Operating System for mobile devices such as smartphones and tablet computers. Android was developed by the *Open Handset Alliance*, led by Google, and other companies.

Android offers a unified approach to application development for mobile devices which means developers need only develop for Android, and their applications should be able to run on different devices powered by Android.

The first beta version of the Android Software Development Kit (SDK) was released by Google in 2007 where as the first commercial version, Android 1.0, was released in September 2008.

On June 27, 2012, at the Google I/O conference, Google announced the next Android version, 4.1 Jelly Bean. Jelly Bean is an incremental update, with the primary aim of improving the user interface, both in terms of functionality and performance.

The source code for Android is available under free and open source software licenses. Google publishes most of the code under the Apache License version 2.0 and the rest, Linux kernel changes, under the GNU General Public License version 2

Wamp Server:

The acronym WAMP refers to a set of free (open source) applications, combined with Microsoft Windows, which are commonly used in Web server environments. The WAMP stack provides developers with the four key elements of a Web server: an operating system, database, Web server and Web scripting software. The combined usage of these programs is called a server stack. In this stack, Microsoft Windows is the operating system (OS), Apache is the Web server, and MySQL handles the database components, while PHP, Python, or PERL represents the dynamic scripting languages.

7.2 Implementation language

PHP:

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language. As of January 2013, PHP was installed on more than 240 million websites (39% of those sampled) and 2.1 million web servers. The reference implementation of PHP is now produced by The PHP Group. While PHP originally stood for Personal Home Page, it now stands for PHP: Hypertext Pre-processor, a recursive acronym.

PHP code is usually interpreted by a web server with a PHP processor module, which generates the resulting web page; it can also be embedded directly into a HTML source document rather than calling an external file to process data. PHP has also evolved to include a command-line interface capability and can be used in standalone graphical applications.

PHP is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge

7.3 Actual implementation

In our system, we have three entities namely, E-Cash Company, Users and TTP server. The following diagram (fig 7.1) depicts the schematic representation of the system architecture that we have used.

The following diagram also explains the life cycle of the E-Cash.

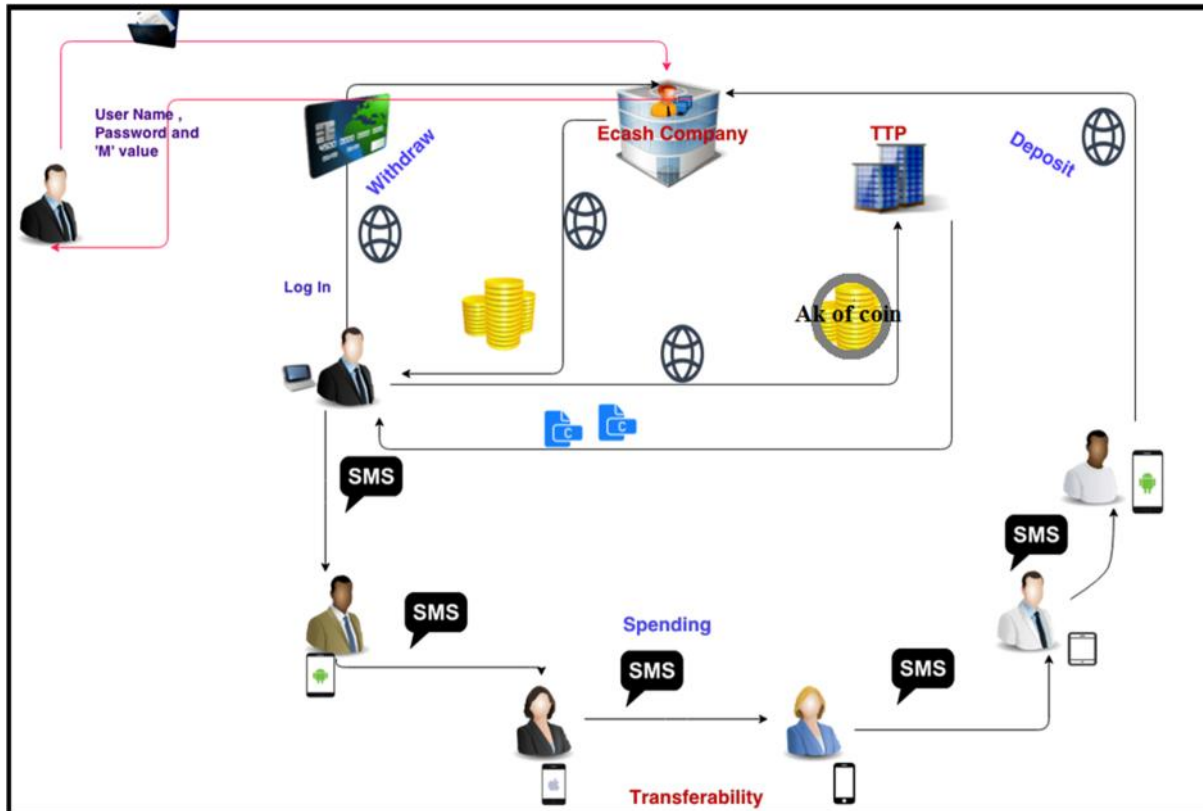


Figure 7.1: System architecture of E-Cash System.

Every user of the E-cash application needs to register himself with the E-cash company by entering his credentials and other login details. Once he is registered, he can make use of the financial transaction facilities provided by the E-cash company. The Trusted Third Party (TTP) is used to generate a unique value C for every E-coin which represents the y-intercept value in our methodology. User is required to enter his login credentials in the E-cash application on his Smartphone to withdraw or deposit the amount in his account. The administrator at the E-cash company will have access to the service which enables him to login as he administrator and keep track of registered users and their transactions of E-cash.

The user spends the E-cash that he has withdrawn, or received from another user through SMS facility, which can be later deposited to the E-cash company for his account.

7.4 Summary

In this chapter, the implementation of the project is discussed. In the next chapter, we will explain the testing of the E-cash application.

8. TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising a software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

8.1 Testing objectives

There are several rules that can serve as testing objectives, they are:

- Testing is a process of executing a program with the intent of finding an error.
- A good test case is one that has a high probability of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered error.

The location proof updating system fulfills all the objectives of testing with minimum or no errors. The system works according to the specification and the performance requirements are also met.

8.2 Test cases

A test case in software engineering is a set of conditions or variables under which a tester will determine whether an application or software system is working correctly. The mechanism for determining whether a software program or system has passed or failed such a test is known as a test oracle. In some settings, an oracle could be a requirement or use case, while in others it could be a heuristic. It may take many test cases to determine that a software program or system is considered sufficiently scrutinized to be released. Test cases are often referred to as test scripts, particularly when written. Written test cases are usually collected into test suites.

The example test cases are:

- The working of User registration both in unique and duplicate cases.
- Authenticated user login.
- Generation of RSA Keys.
- Storing data in server.
- Decryption to verify the E-Cash.

8.3 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produces valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive.

The proposed system passes all the basic tests at component level. Each unique path of a process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Table 8.1: Test case description showing unit testing of E-cash

Test Case:-	UTC-1
Name of Test:-	Check password for user authentication
Item being tested:-	E-cash Company module
Sample Input:-	Account number: 1 , Password: 1234
Expected output:-	Login to be successful
Actual output:-	Login successful
Remarks:-	Success

Test Case:-	UTC-2
Name of Test:-	Generation of RSA keys
Item being tested:-	RSA module
Sample Input:-	RSA algorithm parameters
Expected output:-	A pair of keys to be generated
Actual output:-	A Pair of keys generated
Remarks:-	Success

Test Case:-	UTC-3
Name of Test:-	Create Account
Item being tested:-	Bank module
Sample Input:-	Customer details
Expected output:-	Account creation to be successful
Actual output:-	Account created successfully
Remarks:-	Success

8.4 Integration testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Integration testing for Server Synchronization:

- Testing the IP Address for to communicate with the other Nodes.
- Check the registration request is sent from User to Server.
- Check the handshake between various project modules.
- Check the aggregated data is stored in server.
- Check the data is verified by verifier.

Table 8.2: Test case description showing Integration testing of E-cash

Test Case:-	ITC-1
Name of Test:-	Handshake between various project modules
Item being tested:-	Complete application
Sample Input:-	IP address of server
Expected output:-	Connection to be established between server modules and the E-cash application (Android app)
Actual output:-	Connection established
Remarks:-	Success

8.5 System testing

After the integration testing, the software was completely assembled as a package; interfacing errors have been uncovered and corrected. The final series of software tests, validation tests begins. Validation test succeeds when the application functions in a manner that can be reasonably expected by the customer. Here, the system was tested against system requirements specifications. System testing was actually a series of different tests whose primary purpose was to fully exercise the computer-based system. Although each test has a different purpose, all work to verify that all system elements have been properly integrated and perform allocated functions.

Example System Test cases are:

- User anonymity.
- Transferability of E-cash.
- Double Spending user's detection and notification.

Table 8.3: Test case description showing System testing of E-cash

Test Case:-	STC-1
Name of Test:-	Anonymity Check
Item being tested:-	Complete application
Sample Input:-	Select an application user(U1) let him spend the E-cash withdrawn from the company to another application user(U2)
Expected output:-	Successful transfer of E-cash without U2 being able to know any credentials of U1
Actual output:-	Same as expected output
Remarks:-	Success

Test Case:-	STC-2
Name of Test:-	Fairness check
Item being tested:-	Complete E-cash application

Sample Input:-	Select an application user(U1) let him spend the E-cash withdrawn from the company to another application user(U2)
Expected output:-	U2 successfully receives the same amount of E-cash sent by U1
Actual output:-	Transaction done in fair manner
Remarks:-	Successful

Test Case:-	STC-3
Name of Test:-	Detection of user who tries to Double spend the E-cash
Item being tested:-	E-cash module
Sample Input:-	Select an application user(U1) let him spend the E-cash withdrawn from the company to another application user(U2) and allow him to spend same e-cash to another user(U3)
Expected output:-	Double spending by U1 to be found when either U2 or U3 tries to deposit the E-cash and an alert to be sent through mail
Actual output:-	Same as expected
Remarks:-	Successful

8.6 Acceptance testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

8.7 Summary

In this chapter, various test cases are discussed. In the next chapter we provide the snapshots related to our project.

9. SNAPSHOTS

9.1 E-cash Server Side

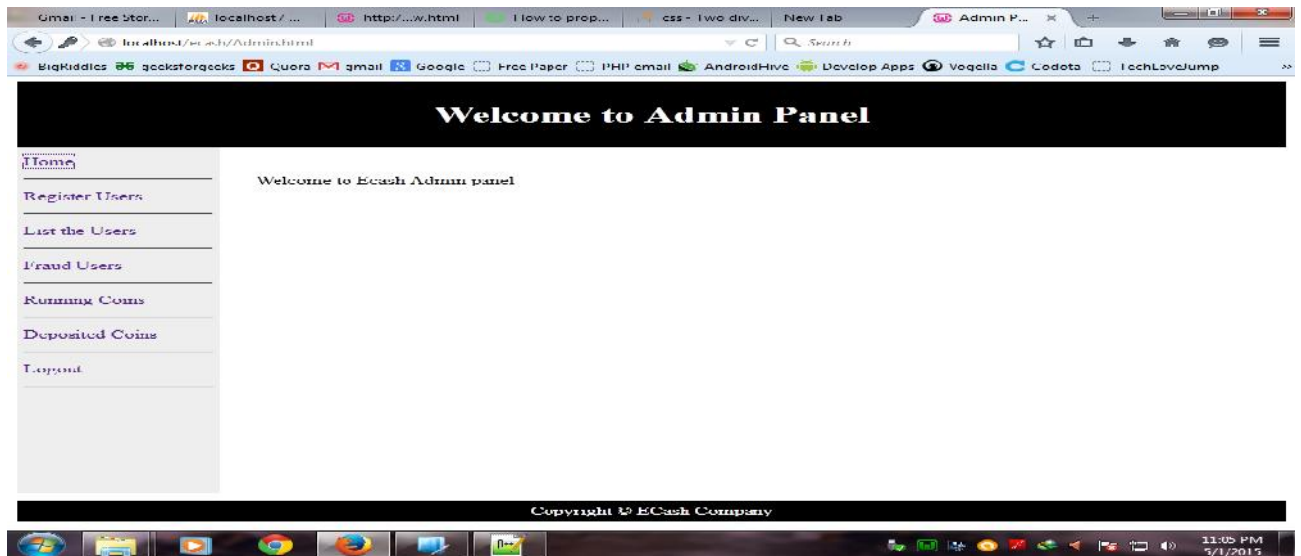


Figure 9.1: Snap shot of home page of E-cash Company

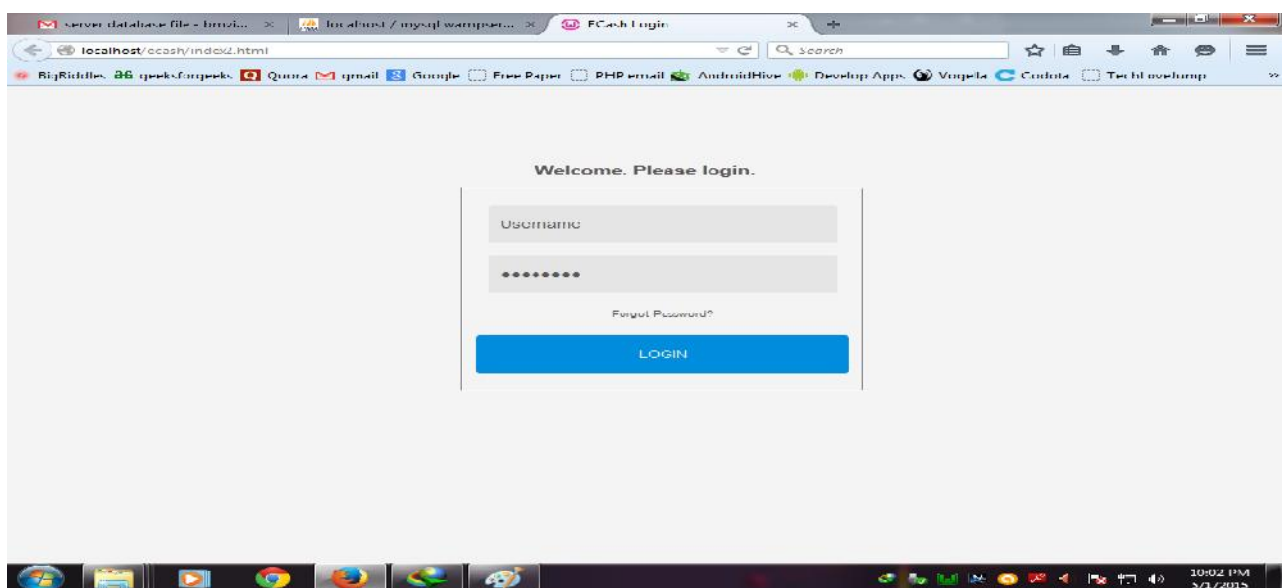


Figure 9.2: Snap shot of admin login page of E-cash Company

Welcome to Admin Panel

Home

Register Users

List the Users

Fraud Users

Running Coins

Deposited Coins

Logout

Ecash Registration Form

Name
First and Last Name

I mail
example@domain.com

Unique ID
Your Unique Id Number

Create a password

Birthday
yyyy-mm-dd

Copyright © ECash Company

Figure 9.3: Snap shot of E-cash registration form page for registering user to obtain E-cash facility

Welcome to Admin Panel

Home

Register Users

List the Users

Fraud Users

Running Coins

Deposited Coins

Logout

Running Coins

NUMBER OF 10'S	NUMBER OF 50'S	NUMBER OF 100'S
6	4	4

Total Amount: 660

Copyright © ECash Company

Figure 9.4: Snap shot of the coins that have been withdrawn from E-cash Company but not yet deposited

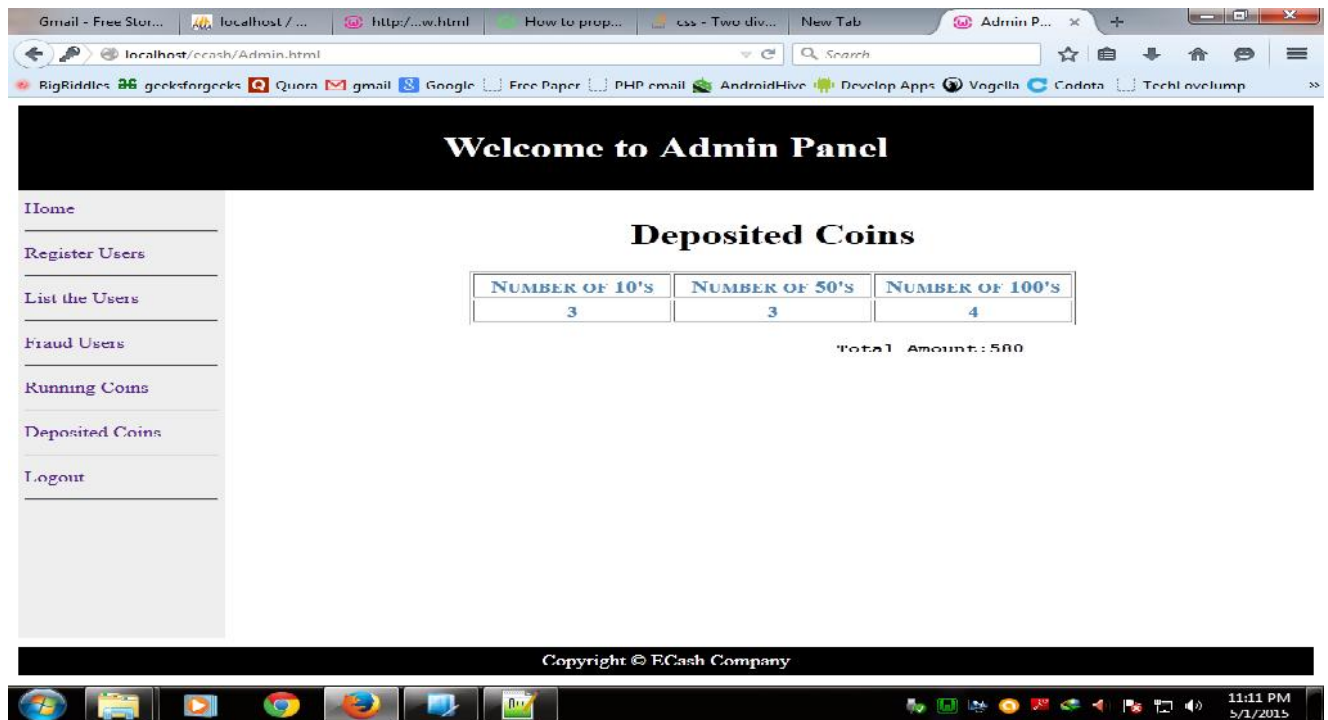


Figure 9.5: Snap shot of the coins that have been deposited to the E-cash Company by user

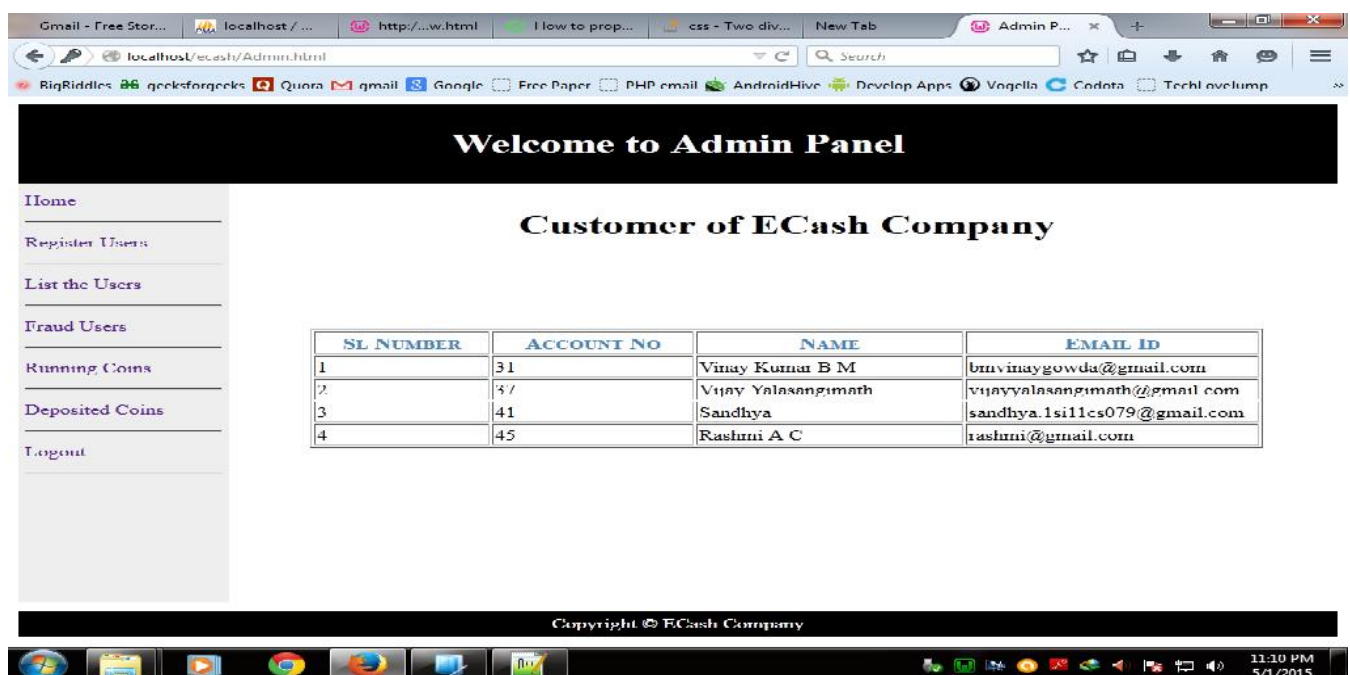
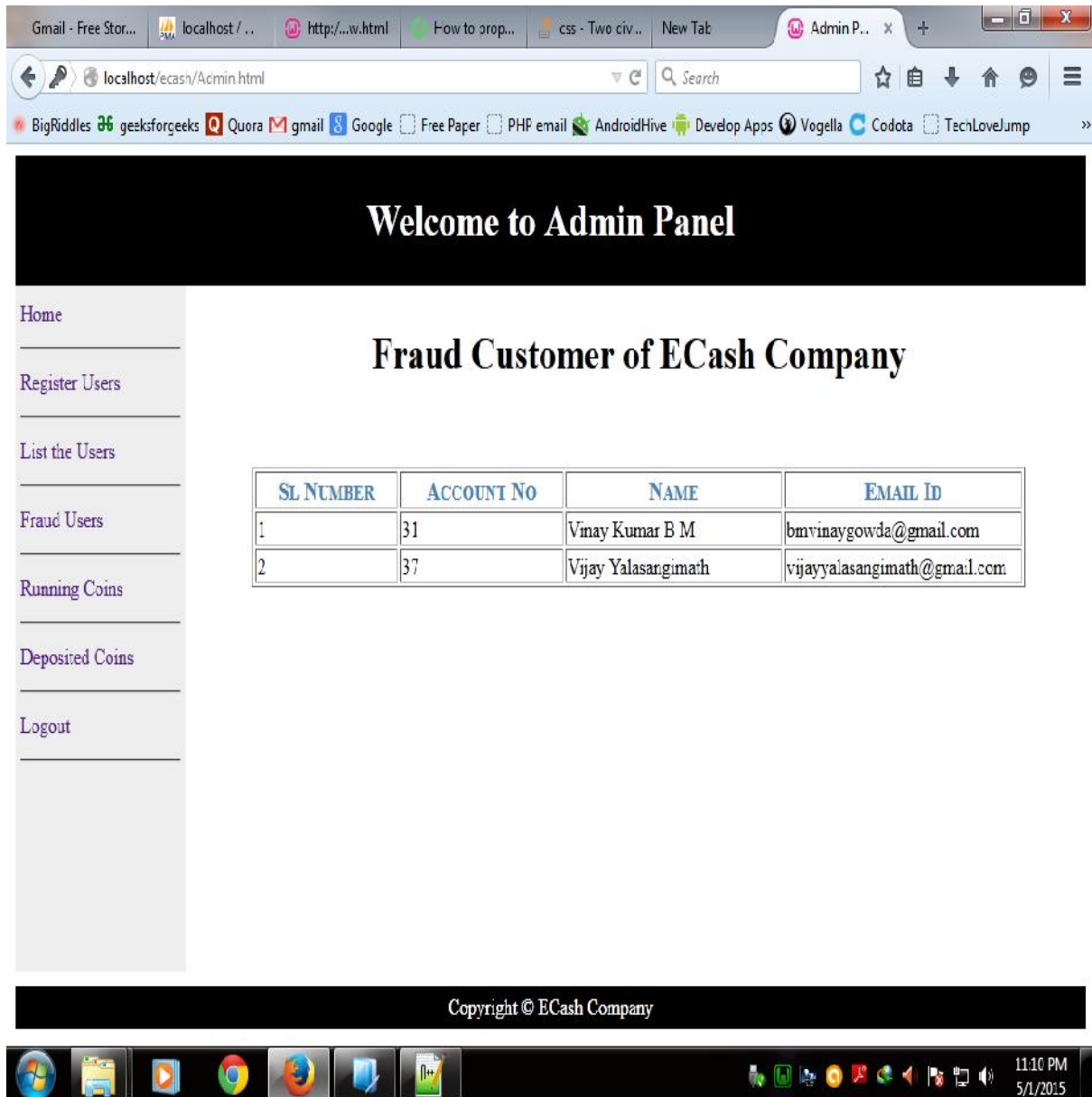


Figure 9.6: Snap shot showing the list of registered users of E-cash Company



The screenshot shows a web browser window with the address bar displaying 'localhost/ecash/Admin.html'. The page has a black header with the text 'Welcome to Admin Panel'. On the left, there is a vertical menu with links: Home, Register Users, List the Users, Fraud Users, Running Coins, Deposited Coins, and Logout. The main content area is titled 'Fraud Customer of ECash Company' and contains a table with the following data:

SL NUMBER	ACCOUNT NO	NAME	EMAIL ID
1	31	Vinay Kumar B M	bmvinaygowda@gmail.com
2	37	Vijay Yalasangimath	vijayyalasangimath@gmail.com

At the bottom of the page, there is a black footer with the text 'Copyright © ECash Company'. The Windows taskbar at the bottom shows the time as 11:10 PM on 5/1/2015.

Figure 9.7: Snap shot detailed user list who are double spent the coin

9.2 E-cash application snapshots

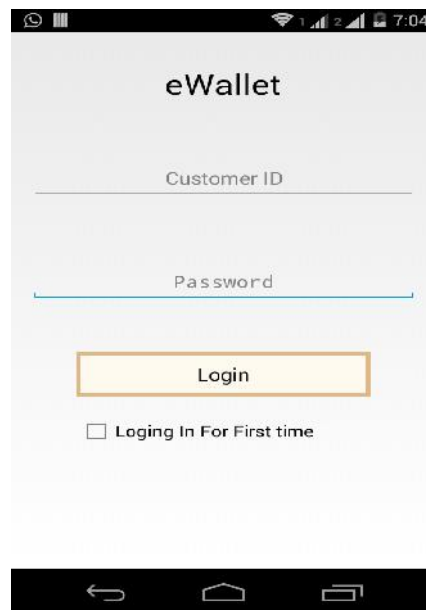


Figure 9.8: Snapshot of customer login page in E-wallet application for doing transactions by providing customer login id and password provided by E-cash Company

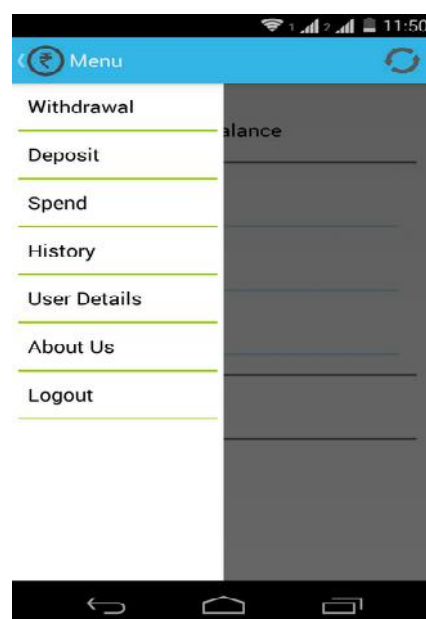


Figure 9.9: Snapshot of list of options provided in E-wallet application for user to do transactions. It includes *withdrawal* of money form E-cash Company, *spending* of e-coin, *transaction history* of customer, *depositing* of e-coin

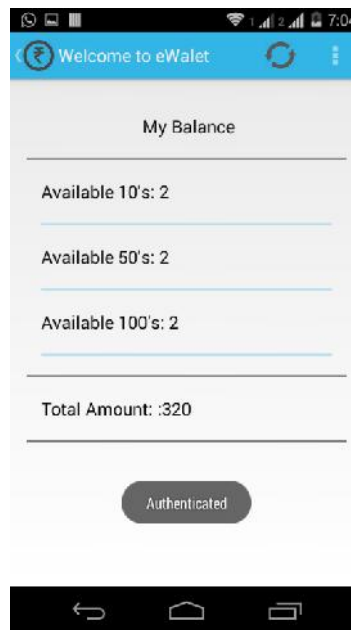


Figure 9.10: Snapshot of available coins they can use for transaction and they are provided by E-cash Company on withdrawal of money.

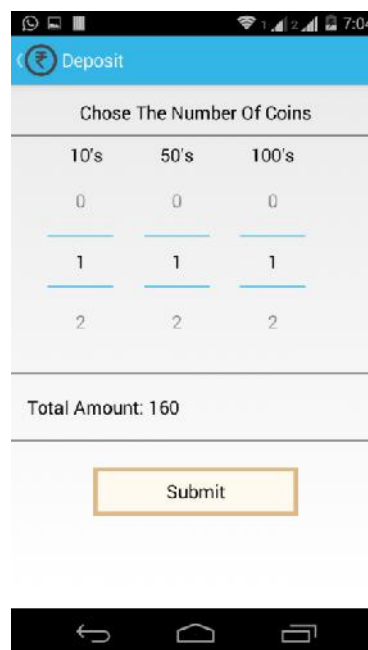


Figure 9.11: Snapshot of *deposition phase* of e-coin. Here we need to choose the number of available coins for deposition.

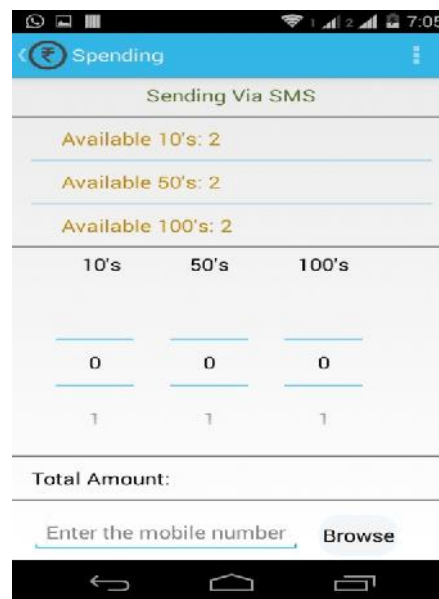


Figure 9.12: Snapshot of *spending phase* of e-coin. Here it will display the number of available coins for spending in terms of denominations. We can also intimate the receiver of the coin through sending an SMS by entering (or browsing from contact list of phone) mobile number of receiver.



Figure 9.13: Snapshot of history of transactions details done by the customer

9.3 Summary

In this chapter, we showed the User Interface of our project through snapshots of the system. In the next chapter, we conclude our project and highlight the possible future enhancements.

10. CONCLUSION AND FUTURE ENHANCEMENTS

10.1 Conclusion

The practical way of making payment requires the transfer of physical cash from the buyer to the seller in lieu of some goods and services. The sense of belonging of cash with the seller gives him the satisfaction of completing transaction. Seller now can act as buyer to next seller, where he pays the same cash to purchase some commodities.

The paper based physical cash system has the feature of belongingness, anonymity and transferability. And a user cannot spend paper cash multiple times; Double-spending concept doesn't arise. Any cash system that doesn't support all these features would be of limited use.

The E-Cash system, E-Cash uses digital data for representation of money and has the feature of belongingness by nature. However, transferability and prevention of double spending is a critical issue. The existing systems are inconvenient as they don't support multiple-transferability of E-cash as a user who has received E-cash from another user needs to refresh the coin with some issuing authority. Also, these kind of systems fail to check the Double-Spending which is possible in fraud cases.

In our work, we have implemented an E-cash application which supports a fair way of E-cash transactions. Any registered user can spend the E-cash he is owning to another registered user, without having required to attend an issuing authority between two spending transactions. Thus, our work supports a fair multiple transferability as well as anonymity.

Also, our E-cash application has the capability of checking fraud transactions .i.e., double-spending can be avoided and the fraud user is detected and his information is alerted through e-mail to the one who has got deceived.

Our scheme also hides the identity of E-coins from the application user i.e., it is not possible to identify the coin by a user who had used or seen it before. In other words, if an E-coin reaches again to the same user, he will not be able to conclude the previous ownership of the coin.

These features provide our application an upper hand over the other existing systems.

10.2 Future enhancement

Our work can be extended further in several ways. The following extension is possible in future.

Multiple-face valued E-coins (Divisibility)

Our solution doesn't address the representation of multiple face values on E-coin. Incorporating this feature will add more value to E-cash protocol.

BIBLIOGRAPHY

- [1] Chih-Hung Wang and wEi-Ming Chiang “The Design of a Novel-E-Cash System with Fairness property and its Implementation in Wireless Communications”, Journal of Computer Vol.18, No.2, July 2007.
- [2] D.Chaum, A. Fiat and M. Naor. “Untraceable E-Cash”, In Advances in Cryptology- Crypto’88, Lecture Notes in Computer Science, pp. 319-327, Springer- Verlag, 1990.
- [3] D. Chaum, “Blind signature systems”, In Advances in Cryptology – Proceedings of Crypto ’83, pp.153-156, 1983.
- [4] “Achieving Electronic privacy”, D. Chaum, (invited) Scientific American, August 1992, pp. 96-101.
- [5] T.Okamoto and K. Ohta, “Universal E-Cash”, In Advances in Cryptology – Proceedings of Crypto ’91, pp.324-337, 1998.
- [6] Indrajit Ray and Indrakshi Ray, Department of Computer Science, Colorado State University, “Fair Exchange in E-commerce”, ACM SIGecom Exchange, Vol.3, NO.2, May 2002, pages 9-17.
- [7] N. Asokan, M. Schunter and M.Waidner, “Optimistic protocol foe Fair Exchange”, Proceedings of \$th ACM Conference on Computer and Communications Security , pp.6-17, 1997.
- [8] R. Song and L. Korba, “How to make E-cash with non-repudiation and anonymity”, Conference on information Technology: Coding and Computing (ITCC ’04), 2004
- [9] Rivest; Ronald L. (Belmont, MA), Shamir; Adi (Cambridge,MA). Adleman; Leonard M. (Arlington, MA), December 14, 1997, U.S. Patent 4,405,829.
- [10] R.L.Rivest, A.Shamir and L.Adleman, “A method for obtaining Digital Signatures and Public-key cryptosystems”, Communications of ACM (1978)

Websites:

- www.wikipedia.org
- www.stackoverflow.com