

INTRODUCTION

1.INTRODUCTION

Two-Factor Data Security Protection Mechanism

This system proposed an improve data security protection mechanism using two components. In this system sender sends an encrypted message to a receiver with the help of Server. The sender requires to know identity of receiver but no need of other information such as certificate or public key. To decrypt the cipher text, receiver needs two parts. The first thing is a unique personal security device or some hardware device connected to the computer system. Second one is private key or secrete key stored in the computer. Without having these two things cipher text never decrypted. The important thing is the security device lost or stolen, then cipher text cannot be decrypted and hardware device is revoked or cancelled to decrypt cipher text.

This mechanism provides the following nice features:

1)The system is an IBE (Identity-based encryption) - based mechanism. That is, the sender only needs to know the identity of the receiver in order to send an encrypted data (cipher text) to him/her. No other information of the receiver (e.g., public key, certificate etc.) is required. Then the sender sends the cipher text to the server where the receiver can download it at any time.

2) The system provides two-factor data encryption protection. In order to decrypt the data stored in the server, the user needs to possess two things. First, the user needs to have his/her secret key which is stored in the computer. Second, the user needs to have a unique personal security device which will be used to connect to the computer (e.g., USB, Bluetooth and NFC). It is impossible to decrypt the cipher text without either piece

. 3) More importantly, the system, for the first time, provides security device (one of the factors) revocability. When the security device is stolen/lost, this device is revoked. That is, using this device you can no longer decrypt any cipher text. The server will immediately execute some algorithms to change the existing cipher text to be un-decrypt able by this device. While, the user needs to use his new/replacement device (together with his secret key) to decrypt his/her cipher text; this process is completely transparent to the sender.

MODULE DESCRIPTION

1. Private Key Generator

- A Private Key Generator is a trusted party responsible for issuing the private key for every user.

2. Security Device Issuer (SDI)

- A Security Device Issuer is a trusted party responsible for issuing security device for every user.

3. Sender

- This user is the sender (and the creator) of the cipher text. The sender only knows the identity (e.g., email address) of the receiver but nothing else related to the receiver. After the sender has created the cipher text, he/she sends to the server to let the receiver for download.

4. Receiver

- This user is the receiver of the cipher text and has a unique identity (e.g., email address). The cipher text is stored on server storage while he/she can download it for decryption. The receiver has a private key (stored in his computer) and a security device (that contains some secret information related to his identity). They are given by the PKG. The decryption of cipher text requires both the private key and the security device.

5. Server

- The server is responsible for storing all cipher text (for receiver to download). Once a user has reported loss of his/her security device (and has obtained a new one from the PKG), the server acts as a proxy to re-encrypt all the past and future cipher text corresponding to the new device. That is, the old device is revoked.

SYSTEM REQUIREMENTS

2. SYSTEM REQUIREMENTS

System Requirements Specification is a structured collection of information that incorporates the requirements of a system. This gives an idea about the system specification required to develop and install the project “SUBSCRIPTION PAYMENT MATRIX TREE GENERATION”. The System Requirements Specification is a technical specification of requirements for the software product. The goal of software requirements definition is to completely and consistently specify the technical requirements for the software product.

One of the most difficult task is selecting software, once the system requirements is find out then we have to determine whether a particular software package fits for those system requirements, This selection summarizes the application requirement.

2.1 HARDWARE REQUIREMENTS

- Processor : Intel Pentium Dual Core / above
- Hard Disk Space : 80 GB
- Ram :1 GB
- Display :14.1”Color Monitor(LCD,CRT OR LED)
- Clock Speed :1.67 GHz

2.2 SOFTWARE REQUIREMENTS

- Operating System : Microsoft Windows 7 / above
- Technologies : HTML5, Bootstrap, JQuery
- Platform : PyCharm
- Front End : Python Django Framework
- Back End : MySQL

DBMS Description

A database is a collection of inter related data stored with minimum redundancy to serve many users quickly and efficiently. The general objective of database design is to make the database access easy, inexpensive and flexible to the user. Database design is used to define and then specify the structure of business used in the client/server system . A business object is nothing but information that is visible to the users of the system. The database must be normalized one. Database design is

one of the important parts in developing software. It is a process of developing the conceptual model of data. It minimizes the artificiality embedded in using separate files. It is a definition of the entire information content of the organization and it specifies a relation between the data.

The primary objectives are fast response time to enquiries, more information at low cost, control of redundancy, clarity and ease-of-use and program independence, accuracy and integrity of the system, fast recovery, privacy and security of information and availability of powerful and user languages. For designing a table, the analyst must decide the fields of the tables, types of the fields, field length, default values etc. For this firstly the entity and relationship must be identified. Secondly, their attributes must be specified. This method of organizing the data table is known as normalization.

The data structure can be later redefined through a normalization process that groups data in the simplest way possible so that later changes can be made with ease.

Normalization is designed to simplify relationship and establish logical links between files without losing information. An inherit problem is data redundancy and the inefficiency it generates. In other words, normalization implies splitting the tables into two or more tables with fewer columns. Most designing techniques try to reach and a few 4NF, but many reach 5NF.

The six normalization rules are:

- * 1NF – each row or column must have a single value with no repeating values.
- * 2NF – each non-key column must depend on the primary key column.
- * 3NF – no non-key column can depend on another non-key column.
- * BCNF – no attribute of a composite key depend on the attribute of another composite key.
- * 4NF – an entity cannot have a 1:1 relation between key column and non-key column.
- * 5NF – if and only if every non-trivial join dependency in it is implied by the candidate key.

It is also known as project join normal form.

Features of Operating System

This project work is done in Windows 10, which is the operating system. An operating system is a set of software tools designed to make it easy for people or programmers to make optimum use of the computer. People can be separated into two groups, users and programmers. The user wants a convenient set of commands to manage files of data or programs, copy and run application packages while a programmer uses a set of tools that can be held together and debug programs. No matter where you are working, your computer will easier to use and manage, because Microsoft Windows 10 is more compatible and powerful than any workstation you have used.

The main features of Windows 10 are:

1. Easier to use
2. Easier to manage
3. More compatible
4. More powerful

Easier to use:

With Windows 10, you can have faster access to information, and you are able to accomplish tasks more quickly and easily.

Windows 10 makes it easier to:

- Work with files
- Find information.
- Personalize computing environment.
- Work remotely
- Work taking place the web

Easier to manage:

You and your network administrators can work more efficiently now, because many of the most common management tasks are streamlined with Windows 10.

With Windows 10 your workstation will be easier to:

- Setup
- Administrate
- Support

More compatible:

Windows 10 offers increased compatibility. With different types of network and with wide array of hardware and software.

Windows 10 also provides:

- Improved driver support
- Increased support for new generation hardware multimedia technologies.

More powerful:

For all your computing needs Windows 10 provides:

- Industrial-strength reliability.
- The highest level of security
- Powerful performance.

Kernel Features

The kernel is considered to be the heart of the operating system that provides services to the programs running on the computer. It takes care of the hardware, software, network resources, file systems and the remaining services such as,

- * Security
- * System fault tolerance
- * Multitasking
- * Multiprocessing
- * Platform independence
- * File system reliability
- * File system security
- * Flexible protocol support
- * Support multi-client operating system
- * Enhanced scalability
- * Multi-user environment
- * Communication.

LANGUAGE OVERVIEW

PYTHON

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding,

make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

HYPER TEXT MARKUP LANGUAGE (HTML)

To public information for global distribution, one needs a universally understood language, a kind of publishing mother tongue that all computers may potentially understand. The publishing language used by the World Wide Web is HTML. HTML is short for hypertext markup language. It defines the structure and layout of a web document by using a variety of tags and attributes. HTML pages contain a set of markup symbols or codes intended for display on a browser. The markup tells the browser how to display a webpage's words and images for the user. Each individual markup code is referred to as an element or tag. Some elements come in pairs (container tags) that indicate when some display effect is to begin and when it is to end. Some tags enable the document to display formatted text, color, a variety of fonts, graphic images, special effects, hypertext jumps to other Internet locations and information forms. HTML 5.0 is an advancement over the standard specification of HTML. It extends HTML with mechanisms for style sheets, scripting, frames, embedding objects, improved support for right to left and mixed direction text, richer tables and enhancements to forms offering improved accessibility for people with disabilities. HTML 5.0 also takes great strides towards the internationalization of documents, with the goal of making the Web truly World Wide. Early versions of HTML were defined with loose syntactical rules, which helped its adoption by those unfamiliar with web publishing. Over time, the trend has been to create increasingly strict language syntax. HTML 4.01 is the current version of HTML specification. This project makes use of HTML 4.01 specification.

JAVASCRIPT

JavaScript is a scripting language from Netscape that is only marginally related to java.java and JavaScript is not the same thing. JavaScript was designed to resemble java, which in turn looks like c and C++.the difference is that java was built as a general purpose object language, while JavaScript is intended to provide a quick and simpler language for enhancing web pages and servers. JavaScript is embedded as a small program in web page that is interpreted and executed by mouse functions, buttons or other actions from the user. JavaScript can be used to fully control Netscape and Microsoft web browsers including all the familiar browser attributes.

Without any network transmission an HTML page with embedded JavaScript can interpret the entered text and alter the user with a message.

SUBLIME

Sublime is a professional IDE (integrated development environment) for designing, coding and developing websites, web pages and web applications. The visual editing features in sublime help in creation of web pages quickly without writing a line of code. All the site elements or assets can be dragged from an easy to use panel directly into a document and the changes can be viewed instantly. Sublime also provides a full featured coding environment that includes code editing tools (such as coloring and tag completion) and reference material on HTML, cascading style sheet (CSS), java script etc. Sublime is completely customizable. The developer can create his own objects and commands, modify keyboard shortcuts, and even write JavaScript code to extend Sublime capabilities with new behaviors, property in specters and site reports.

MySQL

SQL which stands for Structured Query Language, is a standard for data manipulation. MySQL is first established by ANSI in 1982.SQL instructions are used to control relational database packages, which is freely available and implements a subset of SQL. MySQL is a true multi-user, multithreaded SQL database server. It is the preferred solution for those holding accounts at the web master or high level, which need database to drive the websites. MySQL is the world's most popular open source database, recognized for its speed and reliability. MySQL AB the company founded by the creators of the MySQL database, provide MySQL software development and related support and services. This project uses MySQL 8.0.21.

Advantages of MySQL

- Freely available, open source, actively maintained, powerful and efficient.
- More sophisticated ALTER TABLE statement.
- Support a compressed server/client protocol.
- Improved performance over slow links.
- Internal support for text search.
- All MySQL table types are implemented as files, which makes it really easy to backup, move or delete databases and tables when the server is down.

Django

Django is a high-level web framework written in Python that allows developers to rapidly build web applications. It follows the model-view-controller (MVC) architectural pattern, which separates the application logic, data model, and user interface into three interconnected components.

Some of the key features of Django include:

- Object-Relational Mapping (ORM)
- Admin Interface
- URL Routing
- Template System
- Form Handling
- Security
- Scalability

Django is a powerful web framework that allows developers to build complex web applications quickly and easily. It has a large and active community, with many third-party packages available for extending its functionality.

SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Drawbacks of existing system

- Time consuming.
- Less accuracy.
- Need of putting notice or email to everyone.
- Submission and execution are conducted physically.
- Need more manpower.

To avoid all these limitations and make the working more accurately the system needs to be computerized.

3.2 PROPOSED SYSTEM

This system proposed an improve data security protection mechanism using two components. In this system sender sends an encrypted message to a receiver with the help of Server. The sender requires to know identity of receiver but no need of other information such as certificate or public key. To decrypt the cipher text, receiver needs two parts. The first thing is a unique personal security device or some hardware device connected to the computer system. Second one is private key or secrete key stored in the computer. Without having these two things cipher text never decrypted. The important thing is the security device lost or stolen, then cipher text cannot be decrypted and hardware device is revoked or cancelled to decrypt cipher text.

Advantages of proposed system

1)The system is an IBE (Identity-based encryption) - based mechanism. That is, the sender only needs to know the identity of the receiver in order to send an encrypted data (cipher text) to him/her. No other information of the receiver (e.g., public key, certificate etc.) is required. Then the sender sends the cipher text to the server where the receiver can download it at any time.

2) The system provides two-factor data encryption protection. In order to decrypt the data stored in the server, the user needs to possess two things. First, the user needs to have his/her secret key which is stored in the computer. Second, the user needs to have a unique personal security device which will be used to connect to the computer (e.g., USB, Bluetooth and NFC). It is impossible to decrypt the cipher text without either piece

. 3) More importantly, the system, for the first time, provides security device (one of the factors) revocability. When the security device is stolen/lost, this device is revoked. That is, using this device you can no longer decrypt any cipher text. The server will immediately execute some algorithms to change the existing cipher text to be un-decrypt able by this device. While, the user needs to use his

new/replacement device (together with his secret key) to decrypt his/her cipher text; this process is completely transparent to the sender.

Modules

- Primary Key Generator
- Security Device Issuer
- Sender Module
- Receiver Module
- Server Module

The Administrator module includes..

1. Private Key Generator

- A Private Key Generator is a trusted party responsible for issuing the private key for every user.

2. Security Device Issuer (SDI)

- A Security Device Issuer is a trusted party responsible for issuing security device for every user.

3. Sender

- This user is the sender (and the creator) of the cipher text. The sender only knows the identity (e.g., email address) of the receiver but nothing else related to the receiver. After the sender has created the cipher text, he/she sends to the server to let the receiver for download.

4. Receiver

- This user is the receiver of the cipher text and has a unique identity (e.g., email address). The cipher text is stored on server storage while he/she can download it for decryption. The receiver has a private key (stored in his computer) and a security device (that contains some secret information related to his identity). They are given by the PKG. The decryption of cipher text requires both the private key and the security device.

5. Server

- The server is responsible for storing all cipher text (for receiver to download). Once a user has reported loss of his/her security device (and has obtained a new one from the PKG), the server acts as a proxy to re-encrypt all the past and future cipher text corresponding to the new device. That is, the old device is revoked.

3.3 FEASIBILITY STUDY

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spend on it. Feasibility study lets the developer foresee the future of the project and the usefulness. Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus when a new project is proposed, it normally goes through a feasibility study before it's approved for development.

The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as technical, economical and behavioral feasibility. Investigating the existing system in the area under investigation does, to test the technical, social and economic feasibility of a system and generating ideas about the new system. There are three aspects in the feasibility study portion of the preliminary investigation.

- Technical Feasibility
- Economic Feasibility
- Behavioral Feasibility

Technical Feasibility

A technical feasibility center's on the existing computer system (hardware, software etc.) and to what extent it can support the proposed web application. The hardware and software requirements of the system are industry standards. Here no extra expenditure is expected to incur. The consideration that are normally associated with technical feasibility include,

- Development Risk
- Resource Availability
- Technology

Economic feasibility

Economic analysis is the most frequently used method for evaluating the effectiveness of the candidate system. Most commonly known as cost/benefit analysis, the procedure is to determine the benefits and savings that are expected from a candidate system; otherwise further alternations will have to be made, if it is to have a chance of approved. Second are the fee details which functions are same as in the passport generation process. Complaint registration form is used for complaint registration. All the process is carried out with the help of computer's so it gives more accuracy and security. The proposed system is cost effective because of its experimental and user-friendly interface. The user can directly view and change the record.

The analysis raises financial or economical question during the preliminary investigation to estimate the following.

- The cost to conduct a full systems investigation.
- The benefits in the form of reduced costs or fewer costly errors.
- The cost if nothing changes.
- The cost of hardware and software for the class application of the project being considered.

Behavioral Feasibility

It centers on the reaction of the users. Since the system is user-friendly, user training is an easy manner. Any one, with the basic knowledge of computer can operate the system. The user need not have prior knowledge of Visual Basic.

PROJECT DESIGN

4. PROJECT DESIGN

4.1 DATAFLOW DIAGRAM

The data flow diagram (DFD) is one of the most important tools used by system analysts. A DFD is also known as “Bubble Chart” has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design phase. So it is the starting point of the design phase that functionally decomposes the requirement specifications down to the lowest level of detail.

Data flow diagrams are made up of a number of symbols, which represent system components. Most data flow modeling methods use four kinds of symbols. These symbols are used to represent four kinds of the system components. Processes, data stores, data flows and external entities. Circles in DFD represent processes, Data flow is represented by a thin line in the DFD and each data store has a unique name and square or rectangle represents external entities.

Constructing DFD

Several rules of thumb are used in drawing a DFD. Process should be named and numbered for easy reference. Each name should be a representative of the process. The direction of flow is from top to bottom and from left to right.

When a process is exploded into lower-level details, they are numbered. The names of data stores, sources and destinations are written in capital letters. Process and data flow names have the first letter of each word capitalized.

To construct a Data Flow Diagram, we use,

- Arrows
- Circles
- Open ended boxes
- Rectangles

Five rules for constructing a Data Flow Diagram:

- Arrows should not cross each other.
- Squares, circles and files must bear names.

- Decomposed data flow squares and circles can have same names.
- Choose meaningful names for data flow.
- Draw all data flows around the outside of the diagram.

DFD Symbols

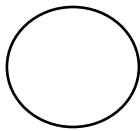
- A Rectangle defines the source or destination of system data.



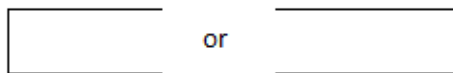
- An Arrows identifies flow of data in motion. It is a pipeline through which information sflows.



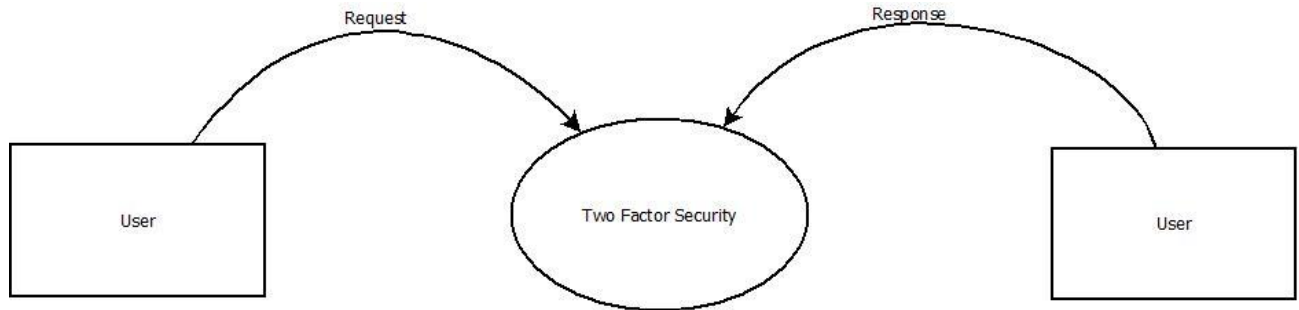
- A circle or bubble represents a process that transforms incoming data flow into outgoing data flows.



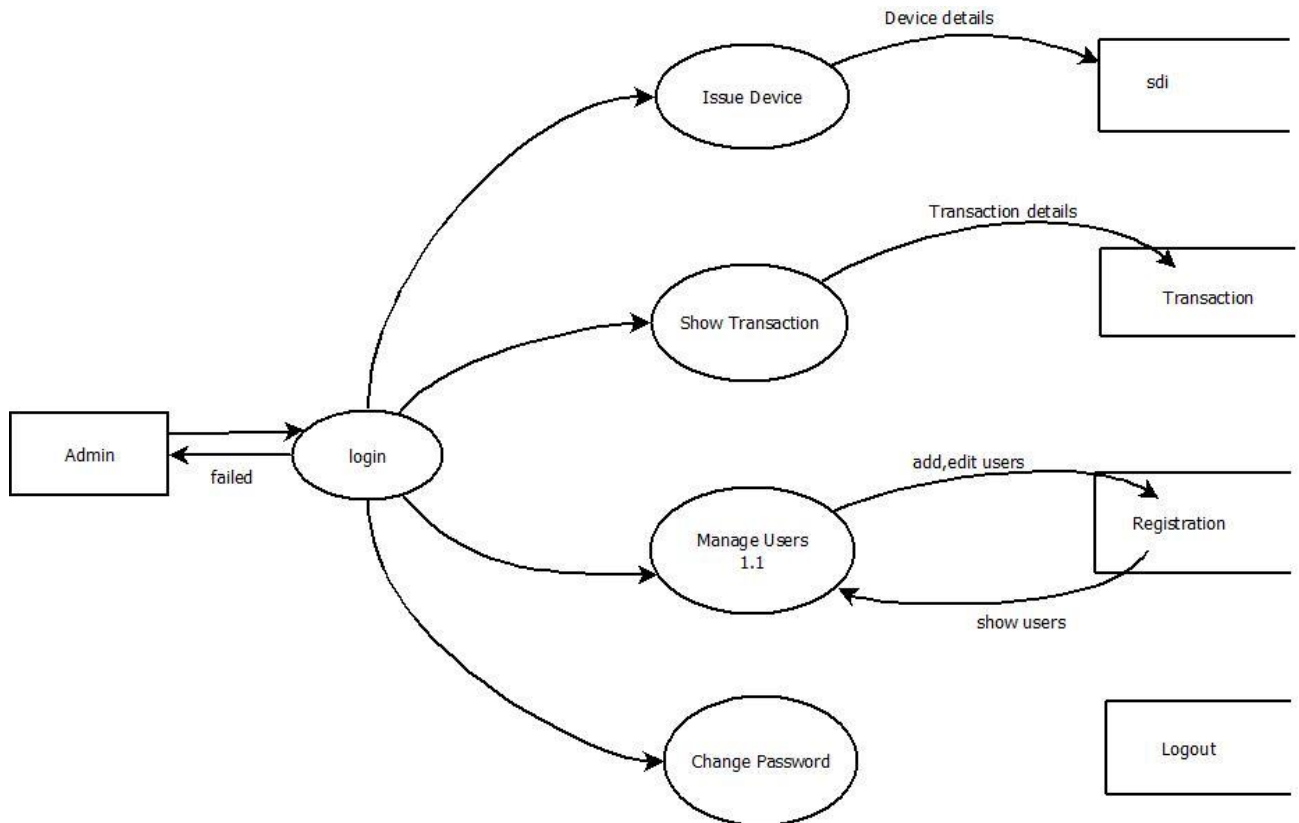
- An open rectangle is a data store.



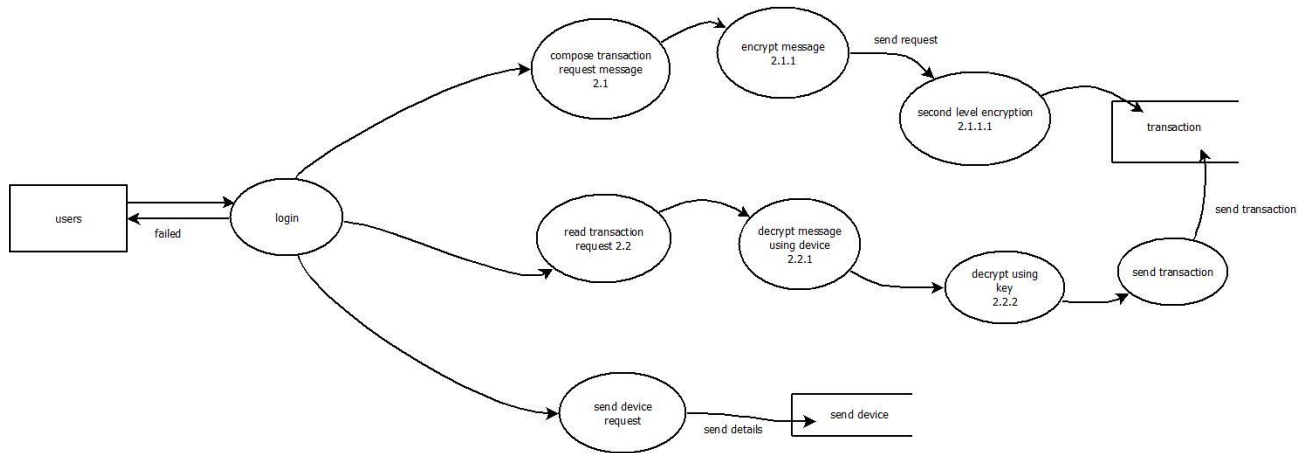
LEVEL 0 (CONTEXT LEVEL)



LEVEL 1(ADMIN)



LEVEL 1(USER)



4.2 USE CASE DIAGRAM

A use case diagram is a type of UML (Unified Modelling Language) diagram that is used to represent the interactions between a system or application and its users or external systems. Use case diagrams are a visual representation of the functional requirements of a system, and they illustrate the different ways in which users or external systems interact with the system.

A use case diagram consists of use cases, actors, and relationships between them. Use cases are represented by ovals or ellipses, and actors are represented by stick figures or rectangles. The relationships between them are represented by arrows.

The use cases represent the specific tasks or actions that a user or system can perform within the system. Actors are the users or external systems that interact with the system. The relationships between the use cases and actors illustrate the different ways in which they interact with each other.

Use case diagrams are useful for software developers, project managers, and other stakeholders in understanding the functional requirements of a system and its interactions with external entities.

They provide a high-level view of the system and help to identify potential issues and areas for improvement.

Constructing Use Case diagram

1. Identify the actors: Identify the actors who will interact with the system. An actor can be a user or an external system.
2. Identify the use cases: Identify the tasks or actions that the system will perform to achieve its objectives.
3. Define the relationships: Define the relationships between the actors and the use cases. An actor can be associated with one or more use cases, and a use case can involve one or more actors.
4. Create the use case diagram: Using a UML modelling tool, create a use case diagram that shows the actors, use cases, and their relationships.
5. Refine the use case diagram: Refine the use case diagram by adding more details, such as preconditions, postconditions, and alternative scenarios. You can also add notes or descriptions to clarify the use cases and actors.
6. Validate the use case diagram: Validate the use case diagram by reviewing it with stakeholders and verifying that it accurately represents the system's functionality.

Rules for constructing a Use Case Diagram:

There are several rules that should be followed when constructing a use case diagram:

1. Actors should be external to the system: Actors should be entities outside the system being modelled. They should not be a part of the system.
2. Use cases should be focused on the system's goals: Use cases should represent the system's functionality and its goals. Use cases should not represent technical details or implementation-specific details.

3. Use cases should be named with verbs: Use cases should be named with an action verb that describes the task or activity the system performs. For example, "Register a new user", "Update user profile", etc.

4. Actors should be related to use cases: Actors should be related to the use cases they interact with.

This relationship should be represented by an association between the actor and the use case.

5. Use cases should be related to other use cases: Use cases should be related to other use cases

that they depend on or that depend on them. This relationship should be represented by include

or extend relationship between the use cases.

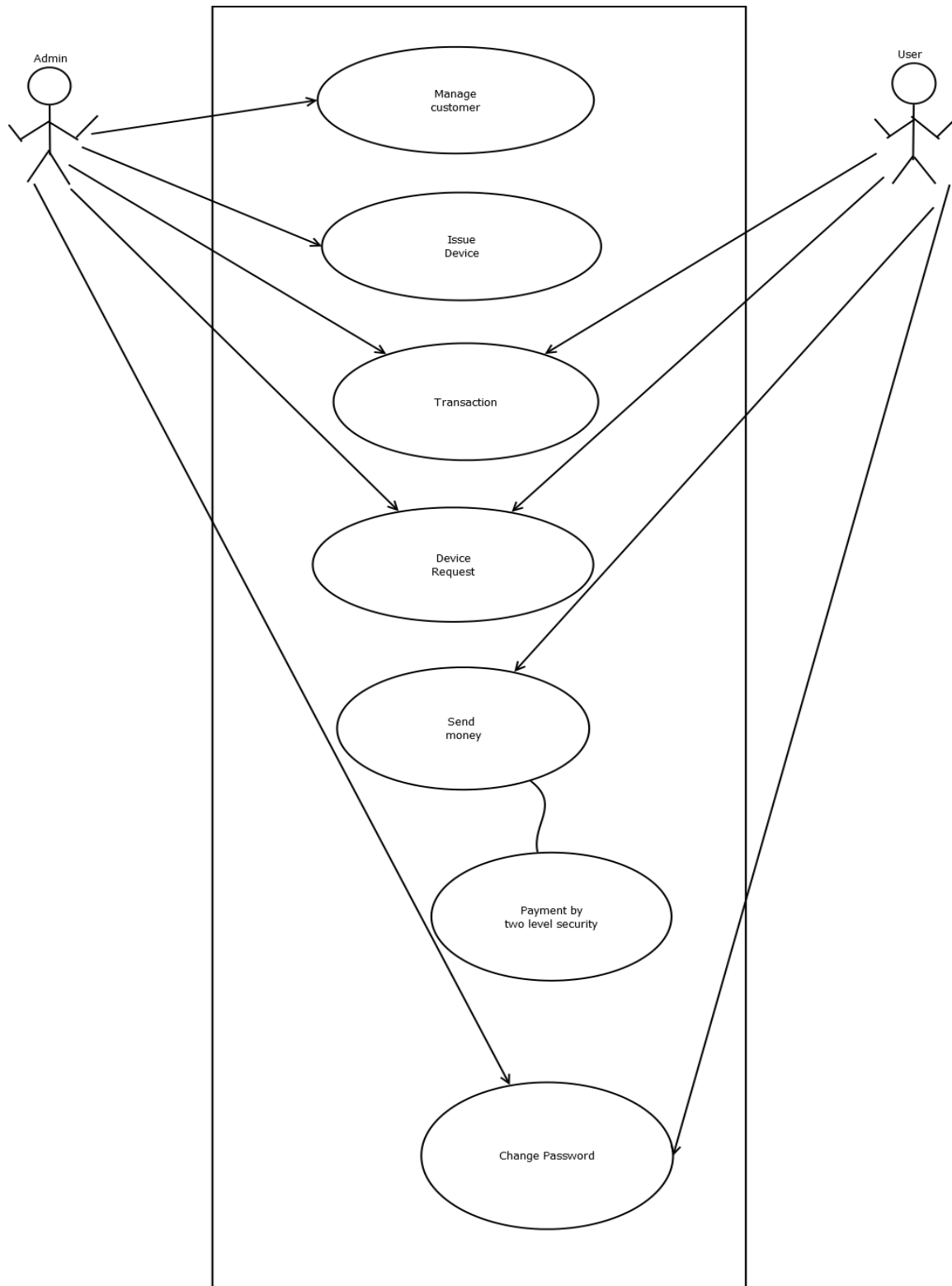
6. The diagram should be readable and understandable: The diagram should be organized and

easy to read. The use cases and actors should be arranged in a logical and meaningful way.

Labels, icons, and colors can be used to improve the readability of the diagram.

The diagram should be validated: The use case diagram should be validated to ensure that it accurately represents the system's functionality and requirements. Validation can be done by reviewing the diagram with stakeholders or using a UML modelling tool to validate the diagram.

TWO FACTOR SECURITY



4.2 DATABASE DESIGN

To design an application it is necessary to design the database. A database is a collection of interrelated data with minimum redundancy to serve the user quickly and efficiently. Database Management System is the process of organizing the tables in the database. The most important step in system design is the database design. The database management system allows the data to be protected and organized separately from other resources. Efficient database management system is software which provides service for accessing, storing and manipulating database.

The objectives of database are:

- Data Integration
- Data Security
- Data Independence

Direct access technique used to permit efficient and flexible linking although sequential organization can be used in database. Data integrity process gives more consistent information and updating being sufficient to achieve a new record status for all application that uses it. This leaves data redundancy.

4.2.1 TABLES

TABLE DESIGN – TWO FACTOR SECURITY

BLOCKDEVICE

Column	Type	Null	Default	Key
customer_dev_id (<i>Primary</i>)	int(200)	No		
email	varchar(200)	No		
date	date	No		
reason	varchar(300)	No		
status	varchar(200)	No		

CUSTOMER

Column	Type	Null	Default	Key
customerid (<i>Primary</i>)	int(100)	No		
accountno	varchar(100)	No		
aadharno	bigint(200)	No		
customername	varchar(200)	No		
gender	char(30)	No		
ph_no	bigint(100)	No		
branchname	varchar(200)	No		
ifsccode	int(30)	No		
email	varchar(200)	No		
address	text	No		
dob	date	No		
image	text	No		
private_key	text	Yes	NULL	
status	varchar(50)	No	inactive	

LOGIN

Column	Type	Null	Default	Key
username (<i>Primary</i>)	varchar(200)	No		
password	varchar(100)	No		
utype	varchar(200)	No		

SDI

Column	Type	Null	Default	Key
customer_dev_id (<i>Primary</i>)	int(200)	No		
email	varchar(200)	No		
devicecode	text	No		
status	varchar(200)	No		

SENDDEVICE

Column	Type	Null	Default	Key
req_id (<i>Primary</i>)	int(11)	No		
email	varchar(200)	No		
accountno	int(200)	No		
date	date	No		
status	varchar(200)	No		
message	text	No		

TRANSACTIONS

Column	Type	Null	Default	Key
transid (<i>Primary</i>)	int(11)	No		
name	varchar(200)	No		
accountno	int(200)	No		
branchname	varchar(200)	No		
ifsccode	text	No		
amount	int(200)	No		
taccountno	int(200)	No		
tifsccode	text	No		
otp	text	No		
status	varchar(200)	No		

PROJECT IMPLEMENTATION

5. PROJECT IMPLEMENTATION

The primary goal of the product implementation is the development of source code that is easy to read and understand. Source code debugging, testing, and modification of a software product consume a large portion of most software budgets. We observed that most of the difficulties encountered during implementation are caused by inadequate analysis and design. Given adequate design documentation, implementation of a software product should be a straightforward, low stress, highly efficient process.

The basic trend of structured coding is use of single entry, single exit constructs. When a program is written using only single entry, single exit constructs, the dynamic flow of execution will match the static structure of the source text. This allows one to understand the program behavior by reading the code from the start to end, as written. Strict adherence to single entry, single exit programs will require repeated code segments or repeated code segments or repeated subroutine calls. Strict adherence to single entry, single exit would prevent restore loop exits and branching to exception handling code.

Our philosophy of structured coding is to adhere to single entry; single exit constructs in a majority of situations, but to violate single entry, single exit as common sense dictates in particular, forward transfer of control to local region of the program do not intend to encourage poor coding style, but to acknowledge the realities of implementation. This view should not be taken as licenses' to substitute go to statements for careful thought and redesign.

Adherence to implementation standards and guidelines by all programmers on a project results in a product of uniform quality standards were defined as those concerns that can be checked by an automated tool, while determining adherence to the guideline enquires human interpretation. Several conditions must be observed to obtain voluntary adherence to standards and guidelines. These conditions were discussed, and it was observed that the psychological atmosphere established by the project leader and the senior programmer is crucial in obtaining voluntary adherence to standard and guidelines.

Supporting documents for the implementation phase include all baseline work products of the analyzer and the design phases and the program unit notebooks. A program unit is the unit of work assigned to an individual programmer. Finally, guideline for the documentation prologues in individual routine and compilation units and internal commenting conventions were discussed.

The major milestone for product implementation is successful integration of source code components into a functioning system. There are however, several intermediate milestones that typically occur prior to integration. Product integration typically occurs in carefully planned stages, with successful completion of each stage providing an intermediate milestone. The ultimate milestone for product implementation is successful demonstration of product capabilities on the customer's acceptance tests.

CODING

6.1 CODING

```
from django.shortcuts import render
from django.http import HttpResponseRedirect
from mypro import models

from django.http import HttpResponseRedirect
from django.shortcuts import render,redirect
from django.views.decorators.csrf import csrf_exempt
from django.template import loader
from django.contrib import messages
from mypro import models
from django.core.files.storage import FileSystemStorage

#def index(request):
#    # return HttpResponseRedirect("frontpage.html")
#def dashboard(request):
#    # return render(request,'admin/newusers.html')
def customer(request):
    customers=models.Customer.objects.all();
    context={'customers':customers};#{key:value}
    return render(request,'admin/customer.html')

def usersection(request):
    return render(request,'admin/usersection.html')

def newcustomer(request):
    return render(request,'admin/newcustomer.html')
```



```
def adduser(request):
    return render(request,'admin/adduser.html')

def sdi(request,rid):
    req=models.senddevice.objects.get(req_id=rid);
    context={'req_msg':req}; #{key:value}

    return render(request,'admin/sdi.html',context)

def blockdevice(request):
    return render(request,'admin/blockdevice.html')

def issuedevice(request):
    return render(request,'admin/issuesdevice.html')

def saveblockdevice(request):
    return render(request,'admin/savesdevice.html')

def showissuedevice(request):
    issues=models.Sdi.objects.all();
    context={'issues' :issues}; #{key:value}
    return render(request,'admin/showissuedevice.html',context)

def showblockdevice(request):
    blocks=models.Blockdevice.objects.all();
    context={'blocks' :blocks}; #{key:value}
    return render(request,'admin/showblockdevice.html',context)
```

```
@csrf_exempt
def login(request):
    template=loader.get_template('login.html')
    return HttpResponse(template.render())

@csrf_exempt
def customer(request):
    customers= models.Customer.objects.all();
    context={"customers":customers };
    return render(request,'admin/customer.html',context)

@csrf_exempt
def dashboard(request):
    template=loader.get_template('admin/dashboard.html')
    return render(request,'admin/dashboard.html')

@csrf_exempt
def login(request):
    username=request.POST["username"]
    password = request.POST["password"]
    usertype = request.POST["usertype"]
    obj1 = models.Login(username=username,password=password,)
    obj1.save()
    #obj2 = models.user(userid=userid,fullname=fullname,address=address,phno=phno,email=email)
    #obj2.save()
    messages.success(request, 'Your registration is successfull!')
    return redirect("../customer")

@csrf_exempt
def savecustomer(request):
    #customerid = request.POST["customerid"]
    accountno = request.POST["accountno"]
    aadharno = request.POST["aadharno"]
```

```
customername = request.POST["customername"]
```

```
gender = request.POST["gender"]
```

```
ph_no = request.POST["ph_no"]
```

```
branchname = request.POST["branchname"]
```

```
ifsccode = request.POST["ifsccode"]
```

```
email = request.POST["email"]
```

```
address = request.POST["address"]
```

```
dob = request.POST["dob"]
```

```
private_key = "
```

```
#nvalue = request.POST["nvalue"]
```

```
status = 'active'
```

```
utype='user'
```

```
adddocument = request.FILES['image']
```

```
fs = FileSystemStorage()
```

```
image = fs.save(adddocument.name, adddocument)
```

```
confirm_password =request.POST["password"]
```

```
obj2 = models.Customer(accountno= accountno,aadharno= aadharno,customername=  
customername,gender= gender,ph_no= ph_no,branchname= branchname,ifsccode= ifsccode,email=  
email,address= address,dob= dob,image= image,private_key= private_key,status= status)  
obj2.save()
```

```
obj3 = models.Login(username= email,password= confirm_password,utype=utype)
```

```
obj3.save()
```

```
#obj4 =  
models.adduser(name=name,address=address,phno=phoneno,email=email,password=password,conf  
irm_password=conirm_password)  
#obj4.save()  
messages.success(request, 'Your registration is successfull!')  
return redirect("../customer")
```

```
@csrf_exempt  
def savesdi(request):  
    email = request.POST["email"]  
    devicecode = request.POST["devicecode"]  
    reqid = request.POST["reqid"]  
    status='issued'  
    issues=models.Sdi.objects.filter(email=email,status=status);  
    req=models.senddevice.objects.filter(req_id=reqid);  
  
    l= len(issues)  
    if l==0:  
        obj3 = models.Sdi(email=email,devicecode=devicecode,status=status)  
        obj3.save()  
        req.update(status='issued')  
        messages.success(request, 'Device Created successfull!')  
    else:  
        messages.error(request, 'A Device already exist in this account')  
  
    return redirect("showissuedevice")  
@csrf_exempt  
def saveblockdevice(request,cid):  
  
    obj4 = models.Blockdevice.objects.get(customer_dev_id=cid)
```

```
email= obj4.email;
issues=models.Sdi.objects.filter(email=email);
issues.update(status='blocked')
bd = models.Blockdevice.objects.filter(customer_dev_id=cid)
bd.update(status='blocked')
messages.success(request, 'blocked successfull!')
return redirect("showblockdevice")

@csrf_exempt
def add_blockdevice(request):
    #customer_dev_id=request.POST["name"]
    email=request.POST["email"]
    date=request.POST["date"]
    reason=request.POST["reason"]

    obj1 =
models.blockdevice(user_id=email,user_type='freelancer',status='inactive',password=password,)
    obj1.save()
    messages.success(request, 'Your registration is successfull!')
    return redirect("../blockdevice")

def newrequestdevice(request):

    req=models.senddevice.objects.filter(status='request');

    context={'req_msg':req}; #{key:value}

    return render(request,'admin/requestdevice.html',context)
```

TESTING

7. TESTING

7.1 SYSTEM TESTING

Software testing is the processes of executing software in a controlled manner, in order to answer the question-Does the software behave as specified? Software testing is often used in association with the terms verification and validation. Validation is the checking or testing of items, includes software, for conformance and consistency with an associated specification.

Software testing is just one kind of verification, which also uses techniques such as reviews, analysis, inspections, and walkthroughs. Validation is the process of checking that what has been specified is what the user actually wanted.

Validation : Are we doing the right job?

Verification: Are we doing the job right?

Software testing should not be confused with debugging. Debugging is the process of analyzing and localizing bugs when software does not behave as expected. Although the identification of some bugs will be obvious from playing with the software, a methodical approach to software testing is a much more through means for identifying bugs. Debugging is therefore an activity which supports testing, but cannot replace testing.

Other activities which are often associated with software testing are static analysis and dynamic analysis. Static analysis investigates the source code of software, looking for problems and gathering metrics without actually executing the code. Dynamic analysis looks at the behavior of software while it is executing, to provide information such as execution traces, limiting profiles, and test coverage information.

Testing is a set of activity that can be planned in advance and conducted systematically. Testing begins at the module level and work towards the integration of entire computer based system. Nothing is complete without testing, as it vital success of the system testing objectives, There are several rules that can serve as testing objectives. They are:

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

If a testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software also testing demonstrate that the software function appears to be working according to the specification, that performance requirement appear to have been met.

There are three ways to test program

- Testing For correctness
- For implementation efficiency
- For computational complexity

Test for correctness are supposed to verify that a program does exactly what it was designed to do. This is much more difficult that it may at first appear, especially for large program.

Test Plan:

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods the test plan acts as a blue print for the action that is to be followed. The software engineers create a computer program, its documentation and related data structures. The software developers is always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder to test the thing that has been built. The specific objectives of testing should be stated in measurable terms. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test plan .

The levels of testing include:

- Unit Testing
- Integration Testing
- Data Validation Testing
- Output Testing

Unit Testing:

Unit testing focuses verification effort on the smallest unit of software design – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The relative complexity of tests and uncovered scope established for unit testing. The unit testing is white box oriented, and step can be conducted in parallel for multiple components. The modular interface is tested to ensure that information properly flows into and out of the program unit under testing.

Integration Testing:

The major concerns of Integration testing are developing and incremental strategy that will limit the complexity of the entire actions among components as they are added to the system. Developing a component as they are added to the system, developing and implementation and integration schedules that will make the Modules available when needed, and designing test case that will demonstrate the viability of the evolving system. Though each program works individually, they should also work after linking them together. This is also referred to as interfacing.

Data may be lost across interface and one module can have adverse effect on another. Subroutines are to linking may not do the desired function expected by the main routine. Integration testing is a symmetric technique for constructing programs structure while at the same time conducting tests to uncover errors associated with the interface. In the testing the programs are constructed and tested in small segments.

System Testing:

When a system is developed it is hoped that it performs properly. In practice however some errors always occur. The main purpose of testing an information system is to find the errors and correct them. A successful test is the one, which finds and error. The main objectives of the system testing are:

- To ensure the operation the system in perform as per specification.
- To make sure that system meets user's requirements during operation.
- To verify that controls incorporate in the system function intend.
- To see that when correct inputs are fed to the system the output are correct.
- To make sure that during operation incorrect input and output will be deleted.

The scope of the system test should include both manual operations and computerized operations. System testing is the comprehensive evolution of the programs, manual procedures, computer operations and controls. System testing is the process of checking if the developed system is working according to original objectives and requirements. All testing needs to be conducted in accordance to the test conditions specified earlier.

Acceptance Testing:

An acceptance test has the objective of checking the validity and the reliability of the system. It verifies that the system procedures operate the system specifications and the integrity of vital data is maintained. The system is found to be user-friendly and working effectively.

7.2 TRAINING

A well designed system, if not operated and used properly could fail. Training the user is important, as if not done well it could prevent the successful implementation of an information system. As system becomes more and more complex the need for education and training is important.

The training could cover:

- Familiarization with the system itself.
- Training in using the application.
- Good document is essential.
- There is no substitute for hands on operation of the system while learning its use.

Change over or conversion:

The plan should be formulated in consultation with the users. The conversion plan includes a description of all activities that must occur to implement the new system and put into operation. This includes identification of people responsible and timetable for each activity that is to be carried out.

During the planning of conversion, the analyst should form a list containing all tasks including the following:

- List all files for conversion.
- Identify all data required to build new file conversion
- Identify all controls to be conversion

- Verify conversion schedule

The conversion plan should anticipate possible problems and ways to deal with them.

Training on application software:

After providing the necessary basic training on the computer awareness the user will have to train on the new application software. This will give the underlying philosophy of the use of the new system such as screen flow. Screen design, type of errors while entering data, the corresponding validation check at entry and ways to correct the data entered.

7.3 DOCUMENTATION

After the job of testing and training were completed the whole system was documented and presented in readable manner. This is to ensure that all the future corrections can be made easily with the help of this document which describes each and every module of the project well described and accurately.

CONCLUSION

8. CONCLUSION

The Web-application entitled **Two Security Security** has satisfied all the requirements of the user. It provides easy and powerful method to handle large amount of data and explore through those data. All the requirements specifications was followed as possible and few additional features were added that can make the application friendlier and less complicated. The project was sufficiently completed with the time span allotted. All the modules are tested separately and put together from the main system, finally the system is tested with real data and it worked successfully. Thus the system has fulfilled the entire objective defined. The project **Two Factor Security** has been developed with the proper guidance from the client. A fully fledged user manual for this system is providing to the user for future working references. We hope the **Two Factor Security** fulfils all the needs in possible manner. The system has been developed in an interactive manner; the reports generated by system are clear and legible. The system is flexible, user friendly and has its own full data security and all data recovery facility.

The major advantages are:

- Easy retrieval of data available in data base.
- Quick implementation of results.
- Very user friendly.
- Does not require large amount of memory.
- Very less manual work is needed.
- Very cost effective.

FUTURE ENHANCEMENT

9. FUTURE ENHANCEMENT

Any system which has been in use for a number of years gradually and become less expensive because of change of change in environment to which it has to be adapted. For the time beings it is possible to overcome problems by amendments and minor modifications to acknowledge the need of fundamental changes. The **Two Factor Security** satisfies the requirements of the monitoring a PC. The system is developed in a user friendly manner.

It has one module for monitoring the data base. The application has been enhanced in the future with the needs of the management. The data base and the information can be updated to the latest coming versions these are also possibilities for enhancing and future developing the project with the latest information and needs of the management, since the coding are in procedural block formats, alter in the code is also made easy. All the functions have been carefully and successfully in the software, and if any development is necessary, in future it can be without affecting the design by adding the additional modules to the system. Some of the enhancements that can increase the valued applications are following:

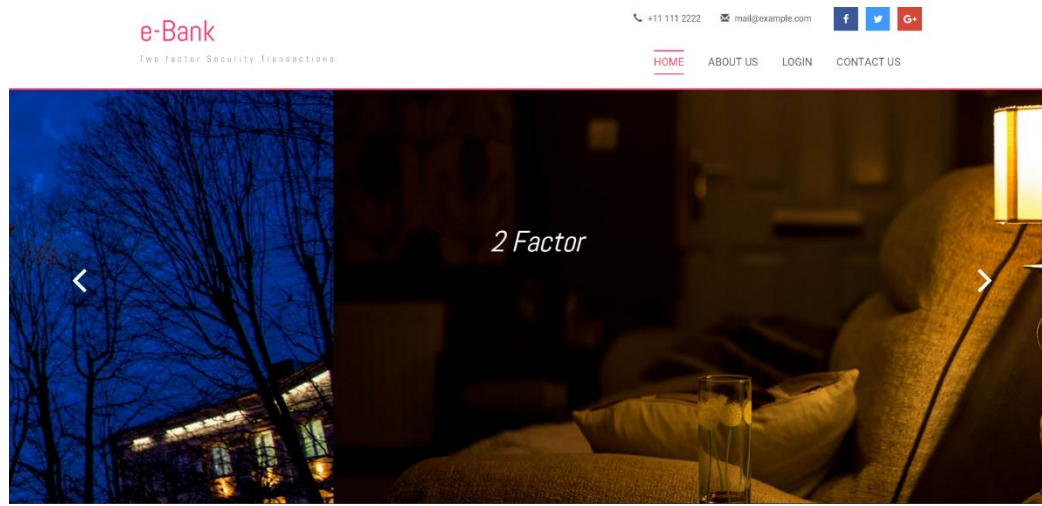
- More number of item details can be inserted into the database.
- It can be integrated with the web for universal access.
- Upgrading the performance.

This project can be later on being used to implement as on online system as well. This system is now implemented at the client machine only but as a feature Enhancement we can modify the system in such a way to make on a client-server network. The system can be even more enhanced by making it an internet based system.

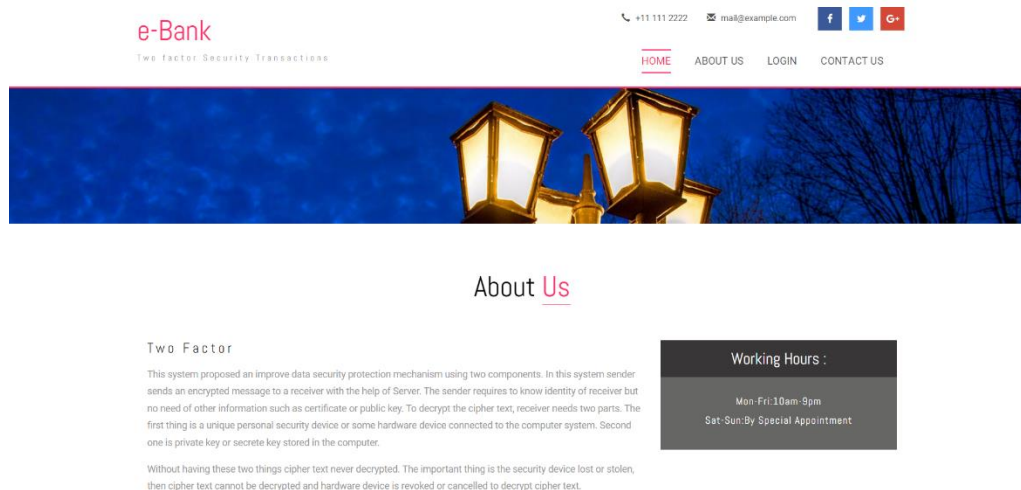
APPENDIX

10. SCREENSHOTS

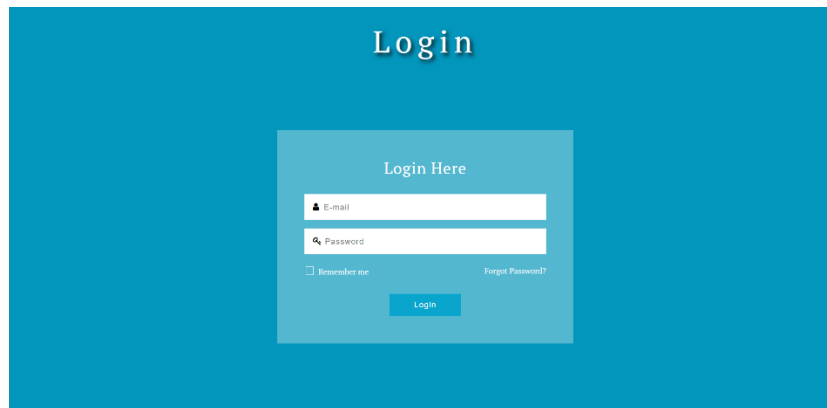
1.HOME



2.ABOUT US

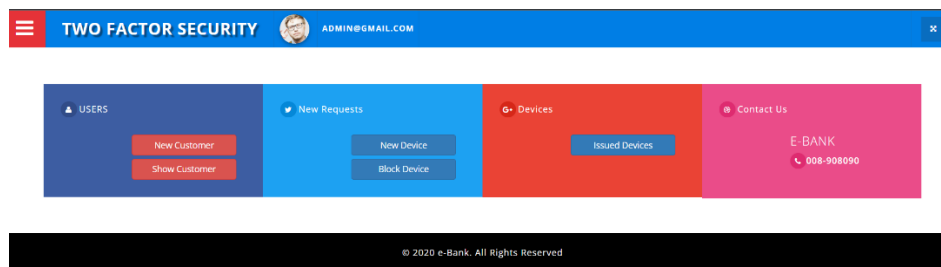


3.LOGIN



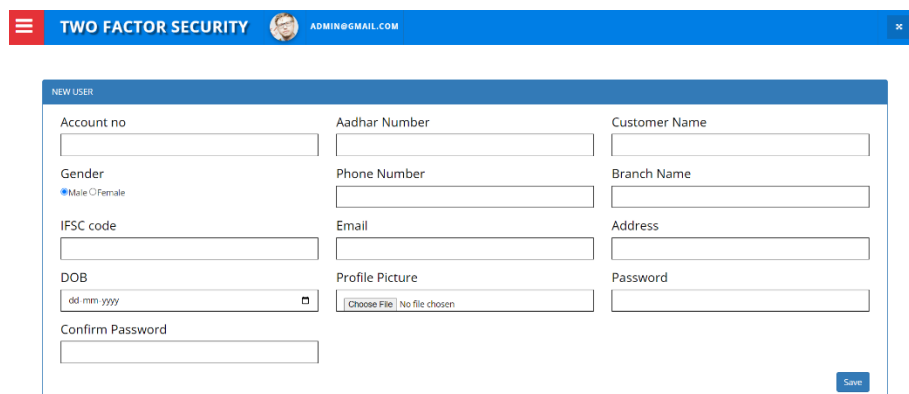
The login page features a blue background with the word "Login" in white at the top center. Below it is a white box titled "Login Here" containing two input fields: "E-mail" and "Password". There is a "Remember me" checkbox and a "Forgot Password?" link. A blue "Login" button is at the bottom of the box.

4.ADMIN






The admin dashboard has a blue header with a hamburger menu, "TWO FACTOR SECURITY", a user profile icon, and "ADMIN@GMAIL.COM". Below the header are four colored cards: "USERS" (blue) with "New Customer" and "Show Customer" buttons; "New Requests" (light blue) with "New Device" and "Block Device" buttons; "Devices" (red) with an "Issued Devices" button; and "Contact Us" (pink) with "E-BANK" and "008-908090". A black footer contains the text "© 2020 e-Bank. All Rights Reserved".


5.USER ENTRY





The "NEW USER" form is a white box with a blue header. It contains several input fields: "Account no", "Aadhar Number", "Customer Name", "Gender" (with radio buttons for Male and Female), "Phone Number", "Branch Name", "IFSC code", "Email", "Address", "DOB" (with a date picker), "Profile Picture" (with a "Choose File" button), "Password", and "Confirm Password". A blue "Save" button is at the bottom right.


6.USER

 TWO FACTOR SECURITY  SREE@GMAIL.COM 

 Devices
[My Device](#)



 New Requests
[New Device](#)
[Block Device](#)

 Payments
[New Payment](#)

 Contact Us
E-BANK
008-908090

© 2020 e-Bank. All Rights Reserved

7.ISSUED

 TWO FACTOR SECURITY  ADMIN@GMAIL.COM 

Email	Devicecode	Status
parvathiunnikrishnan83@gmail.com	90	issued
sree@gmail.com	80	issued

© 2020 e-Bank. All Rights Reserved

BIBLIOGRAPHY

11. BIBLIOGRAPHY

- Elmsri Ramez and Shamkant B Navathe - FUNDAMENTALS OF DATABASE SYSTEMS, Dorling Kindersly , 2013
- Awad M Elias - SYSTEM ANALYSIS AND DESIGN, Galgotia Publications,1996
- Schlossnagle George - ADVANCED PHP PROGRAMMING, Sams,2004
- Welling Luke & Thomson Laura - PHP AND MYSQL WEBDEVELOPMENT Addison-Wesley

Websites:

- <https://www.apache.org/>
- <https://www.adobe.com/>
- <https://www.w3schools.com/>
- www.mysql.com