TWO FACTOR AUTHENTICATION USING MOBILE PHONES

P. 3104

A PROJECT REPORT

Submitted by

MAHESWARAN K **71206104026**

PARTHIBAN PRADEEP P **71206104032**

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2010

Certified that this project report **"TWO FACTOR AUTHENTICATION USING MOBILE PHONES"** is the bonafide work of "**MAHESWARAN K** and **PARTHIBAN PRADEEP P** " who carried out the project under my supervision.

[illegible]

SIGNATURE

Dr. S. Thangasamy

DEAN

Department of Computer Science & Engg

Kumaraguru College of Technology,

Chinnavedampatti Post,

Coimbatore – 641606

[illegible] 15/4/10

SIGNATURE

Ms.P.Malini, L/CSE

SUPERVISOR

Department of Computer Science & Engg

Kumaraguru College of Technology,

Chinnavedampatti Post,

Coimbatore – 641606

The candidates with University Register Nos. **71206104026** and **71206104032** were examined by us in the project viva-voce examination held on

......15-04-2010............

[illegible]

INTERNAL EXAMINER

[illegible]

EXTERNAL EXAMINER

DECLARATION

We hereby declare that the project entitled " **TWO FACTOR AUTHENTICATION USING MOBILE PHONES**" is a record of original work done by us and to the best of our knowledge, a similar work has not been submitted to Anna University or any Institutions, for fulfillment of the requirement of the course study.

The report is submitted in partial fulfillment for the award of the degree of bachelor of computer science and engineering of Anna University, Chennai.

Place: Coimbatore

Date: 15-04-2010

K.Maheswaran

(K.Maheswaran)

[illegible]

(P.Parthiban Pradeep)

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made this possible and whose constant guidance and encouragement crowns all efforts with success.

We are extremely grateful to **Dr. Ramachandran**, Principal, Kumaraguru College of Technology for having given us this opportunity to embark on this project.

We express our sincere and heartfelt thanks to **Dr. S. Thangasamy**, Dean, Department of Computer Science and Engineering, for his kind guidance and support.

We would like to express our sincere thanks to our project coordinator **Mrs. P. Devaki**, for her valuable guidance during the course of the project.

We would also like to thank our class advisor **Mrs. R. Kalaiselvi**. for her constant support and guidance.

We would like to thank our guide **Mrs. P.Malini**, without whose motivation and guidance we would not have been able to embark on a project of this magnitude. We express our sincere thanks for her valuable guidance, benevolent attitude and constant encouragement.

We reciprocate the kindness shown to us by the staff members of our college, people at home and our beloved friends who have contributed in the form of ideas, constructive criticism and encouragements for the successful completion of the project.

ABSTRACT

ABSTRACT

This project describes a method of implementing two factor authentication using mobile phones. The proposed method guarantees that authenticating to services, such as online banking or ATM machines, is done in a very secure manner. The proposed system involves using a mobile phone as a software token for One Time Password generation.

The generated One Time Password is valid for only a short user defined period of time and is generated by factors that are unique to both, the user and the mobile device itself. Additionally, an SMS-based mechanism is implemented as both a backup mechanism for retrieving the password and as a possible mean of synchronization.

Two factor authentications is a mechanism which implements two of the above mentioned factors and is therefore considered stronger and more secure than the traditionally implemented one factor authentication system. Withdrawing money from an ATM machine utilizes two factor authentication; the user must possess the ATM card, i.e. what you have, and must know a unique personal identification number (PIN).

LIST OF ABBREVIATIONS

1.	GSM	Global System for Mobile communication
2.	OTP	One Time Password
3.	ATM	Automatic Teller Machine
4.	SMS	Short Message Service
5.	PIN	Personal Identification Number
6.	IMEI	International Mobile Equipment Identity
7.	IMSI	International Mobile Subscriber Identity
8.	WLAN	Wireless Local Area Network
9.	SPS	Secure processing system

INTRODUCTION

CHAPTER - 1
INTRODUCTION

1.1 Objectives

The title of the project is Two Factor Authentication Using Mobile Phones.

A Secure processing system is a security system that tells online retailers that the user is a genuine cardholder when she/he shops online. It allows user or a customer to use personal password to confirm his identity and protect his/her credit card when the card is used on the Internet, providing greater reassurance and security. It improves the security of Internet payments.

The objective of the proposed system is to make online transaction more efficient to the user who uses the website and shops online. This will have a positive impact on user profitability. To make on-line shopping even simpler and safer, a secure processing system is being introduced.

During the online transaction process, the Merchants payment systems will connect to the secure processing system to carry out security, fraud and validity checks and subsequently authorize and take the payment. It improves the security of Internet payments.To meet the business requirements, the proposed system incorporates the following features:

- Confidentiality of information
- Integrity of data
- Cardholder authentication
- Merchant authentication

GSM MODEM : GLOBAL SYSTEM FOR MOBILE COMMUNICATION

A GSM modem is a wireless modem that works with a GSM wireless network. A wireless modem behaves like a dial-up modem. The main difference between them is that a dial-up modem sends and receives data through a fixed telephone line while a wireless modem sends and receives data through radio waves.

Global system for mobile communication (GSM) is a globally accepted standard for digital cellular communication. GSM is the name of a standardization group established in 1982 to create a common European mobile telephone standard that would formulate specifications for a pan-European mobile cellular radio system operating at 900 MHz.

THE GSM NETWORK

GSM provides recommendations, not requirements. The GSM specifications define the functions and interface requirements in detail but do not address the hardware. The reason for this is to limit the designers as little as possible but still to make it possible for the operators to buy equipment from different suppliers. The GSM network is divided into three major systems: the switching system (SS), the base station system (BSS), and the operation and support system (OSS).

GSM SUBSCRIBER SERVICES

There are two basic types of services offered through GSM: telephony (also referred to as teleservices) and data (also referred to as bearer services). Telephony services are mainly voice services that provide subscribers with the complete capability (including necessary terminal equipment) to communicate with other subscribers. Data services provide the capacity necessary to transmit appropriate data signals between two access points creating an interface to the network. In addition to normal telephony and emergency calling.

The following subscriber services are supported by GSM:

SMS, Voice mail, Fax ,Cell Broadcast etc.

In the proposed system , the user paying through credit card and then a dynamic password will appear as 'sms' in the user's authorized mobile.

The following subscriber service is by GSM:

SMS -- Short Message Services

A convenient facility of the GSM network is the short message service. A message consisting of a maximum of 160 alphanumeric characters can be sent to or from a mobile station. This service can be viewed as an advanced form of alphanumeric paging with a number of advantages. If the subscriber's mobile unit is powered off or has left the coverage area, the message is stored and offered back to the subscriber when the mobile is powered on or has reentered the coverage area of the network. This function ensures that the message will be received.

SMS is an area where the modem can be used to provide features like:

- ✓ Pre-stored SMS transmission.
- ✓ These SMS can be transmitted on certain trigger events in an automation system.
- ✓ SMS can also be used in areas where small text information has to be sent. The transmitter can be an automation system or machines like vending machines, collection machines or applications like positioning systems where the navigator keeps on sending SMS at particular time intervals.
- ✓ SMS can be a solution where GSM data call or GPRS services are not available.

LITERATURE REVIEW

CHAPTER -2

LITERATURE REVIEW

2.1 Identification of need

With growing usage of the Internet, people are utilizing the convenience of online shopping and the ability to place an order for what they want at all hours of the day and night, at the office, home, airport or just about anywhere. Services to the Internet will increase the business potential in many ways. However, e-commerce requires a commitment to securing transaction details, including credit card information from customers.

The following points emphasize need of the proposed system:

- When the user shops online and paying through credit card, the existing system does not contain any additional password security.

- In the existing system anybody can pay through credit card by providing the card pin number. He/She does not provide any additional information for online card payment.

- As e-commerce has grown, so have security threats. When the card is not swapped, it requires more security.

- The SPS provides additional information not visible in the card and facilitates the card holder to register their own password. That makes the user more secure while doing online transaction.

2.2 Preliminary Investigation

Dynamic passcode authentication is one solution that uses the added security of credit cards to offer better protection against online fraud. The primary benefit of this system is the reduction in disputed transactions and the resultant exception handling expense and losses. The substantial proportion of customer complaints could be eliminated with the use of Authenticated Payment. This will have a positive impact on user profitability.To make on-line shopping even simpler and safer, a secure processing system is being introduced.

SPS is a simple password-protected identity-checking service that takes the risk out of online retail customers. The proposed system SPS is a new concept for additional security for all online 'card not present' transaction. In this concept when the card is not swapped , it require more security. Hence the SPS provides additional information not visible in the card and facilitates the card holder to register their own password. That makes the user more secure while doing online transaction.

2.3 Existing System

When the user shops online and paying through credit card, the existing system does not contain any additional password security.

In the existing system anybody can pay through credit card by providing the card pin number. He/She does not provide any additional information for online card payment. There is no assurance that only the right owner of the card paying through his card.

2.4 Proposed System

The proposed system allows user or customer to use a personal password to confirm his identity and protect his/her card when the card is used on the Internet, providing greater reassurance and security. The proposed system is a security system that tells on-line retailers that the user is a genuine cardholder when he shops on-line.

To meet the user requirements, The proposed system incorporates the following features:

- In this proposed system specifically the example of online book shop is taken to represent the online transaction. After the selection of the book the user can select the credit card payment mode and enter the secured processing system(SPS) for secured transaction.

- During the online transaction process, the Merchant payment systems will connect to the sps system to carry out security, fraud and validity checks and subsequently authorize and take the payment. It improves the security of Internet payments by providing an additional password to the user. Using the password the user can successfully make his payment.

- Online card transactions over Internet need enhanced security. Secure processing system facilitates additional security by way of a cardholder-chosen password, which is known only to the cardholder.

- SPS is a new way to add safety when the user buy online. Adding a personal password to the existing Credit Card ensures that only the authorized card holder can use Card online. It's easy to activate the SPS service on your existing Card.

- Whenever the user submits an order at a participating online store, the SPS window will appear. User enter details and the password will appear in the authorized mobile phone, enter the password in the appropriate box and submit, and payment is over.

- SPS receives credit card details from the card issuing bank which is invisible to the customer.
- Cardholders enter their PIN on the key pad to generate a one-time code for secure authentication by SPS.

- Dynamic passcode authentication is one solution that uses the added security of credit cards to offer better protection against online fraud. The primary benefit of this system is the reduction in disputed transactions and the resultant exception handling expense and losses.

- Thus the proposed system is adding an extra layer of security at the point where you enter credit card information online. The service helps to prevent unauthorized online use before it happens by confirming your identity with an additional password.

2.5 Advantages

This e-commerce solution lets you sell products with easy-to-use shopping cart checkout. Shopping cart enable your customers to add items, view their order, continue shopping, and quickly check out. It is helpful for your customers to provide a secured purchasing using SMS using mobile phone.

p-3104

SOFTWARE DESCRIPTION

CHAPTER - 3
SOFTWARE DESCRIPTION

3.1 Java

James Gosling, Patrick Naughton, Chris Warth, Ed Frank and Mike Sheridan convinced Java at Sun Microsystems, Inc. in 1991. It took 18 months to develop the first working version. This language was initially called "Oak" but was renamed "Java" in 1995.

Object oriented programming is the core of Java. All Java Programs are object oriented data. The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple and powerful
- Secure
- Portable
- Object-Oriented
- Robust
- Multithreaded
- Architecture-neutral
- Interpreted
- High performance
- Distributed and Dynamic.

With the emergence of World Wide Web, java was propelled to the forefront of the computer language design, because the web demanded portable programs. Java is programmers language it is logically consistent. Java is not a language with training wheels. It is language for professional programmers.

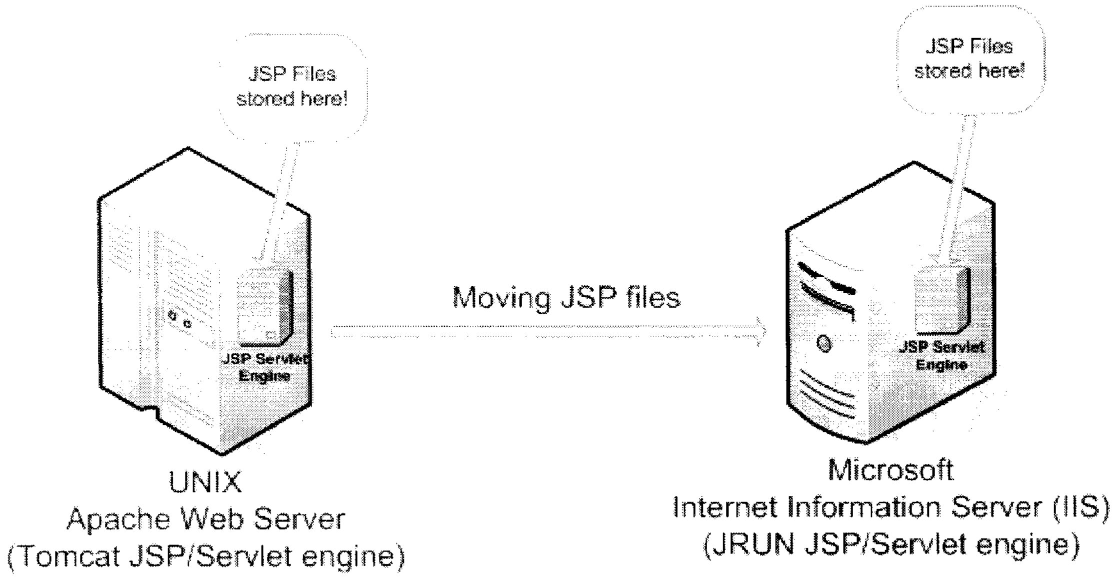
3.2 JSP: Java Server Pages

JSP is a widely used general-purpose scripting language that is especially suited for Web development .In our proposed system JSP is the server side scripting language.

Java Server Pages or JSP for short is Sun's solution for developing dynamic web sites. JSP provide excellent server side scripting support for creating database driven web applications. JSP enable the developers to directly insert java code into jsp file, this makes the development process very simple and its maintenance also becomes very easy. JSP pages are efficient, it loads into the web servers memory on receiving the request very first time and the subsequent calls are served within a very short period of time.

Main reasons to use JSP:

- Multi platform.
- Component reuse by using JavaBeans and EJB.
- We can take one JSP file and move it to another platform, web server or JSP Servlet engine.



In today's environment most web sites servers dynamic pages based on user request. Database is very convenient way to store the data of users and other things. JDBC provide excellent database connectivity in heterogeneous database environment. Using JSP and JDBC its very easy to develop database driven web application.

Java is known for its characteristic of "write once, run anywhere." JSP pages are platform independent. Your port your .jsp pages to any platform.

The most significant of the many good reasons is that it is amazingly easy to develop sophisticated Web sites with JSPs. Through a mechanism called JavaBeans, JSPs have made it possible for large teams or individuals working on complex projects to divide the work in such a way as to make each piece simple and manageable, without sacrificing any power. JSPs also provide a great deal of flexibility when generating HTML, through the ability to create HTML-like custom tags.

JSPs are compiled into Java Servlets by a JSP compiler. A JSP compiler may generate a servlet in Java code that is then compiled by the Java compiler, or it may generate byte code for the servlet directly. "Java Server Pages" is a technology released by Sun.

In addition to this fundamental ease of development, high-quality JSP tools are readily available and easy to use. There is no need to buy expensive software or commit to a particular operating system in order to use JSPs.

JSP provides a Tag based approach to develop the server side executable application which is used to generate dynamic content.

There are five main tags:

1. Declaration tag
2. Expression tag
3. Directive tag
4. Scriptlet tag
5. Action tag

Declaration tag (<%! %>)

This tag allows the developer to declare variables or methods.

Before the declaration it must have <%!

At the end of the declaration, the developer must have %>

Code placed in this tag must end in a semicolon (;)

Expression tag (<%= %>)

This tag allows the developer to embed any java expression and is short for out.println() .

A semicolon(;) does not appear at the end of the code inside the tag.

Directive tag (<%@ directive ... %>)

A JSP directive gives special information about the page to the JSP Engine. There are three main types of directives:

1) page - processing information for this page.
2) Include - files to be included.
3) Tag library - tag library to be used in this page.

Action tag

There are three main roles of action tags :

1) enable the use of server side Javabeans
2) transfer control between pages
3) browser independent support for applets.

Scriptlet tag (<% ... %>)

Between <% and %> tags,any valid Java code is called a Scriptlet. This code can access any variable or bean declared.

Problems with Servlet – If we want to generate a page, which consists of some dynamic content, then we have to include the entire static content into the Servlet (i.e, out.println method) along with the dynamic content. i.e. when a huge view has to be generated then there was a problem with the application development and modifications(view/application).
While developing web applications using JSP technology we can separate the business logic from the presentation logic. Separation of presentation logic from business logic helps in reducing the time required to manage the project in future.

Thus the advantage of JSP is that it provides a better mechanism to develop and maintain huge view compare to servlets.

3.3 J2EE :

Short for Java 2 Platform Enterprise Edition. J2EE is a platform-independent, Java-centric environment from Sun for developing, building and deploying Web-based enterprise applications online. The J2EE platform consists of a set of services, APIs, and protocols that provide the functionality for developing multitiered, Web-based applications.

The Java 2 Platform, Enterprise Edition (J2EE) is a set of coordinated specifications and practices that together enable solutions for developing, deploying , and managing multi-tier server-centric applications.

Some of the key features and services of J2EE:

- At the client tier, J2EE supports pure HTML, as well as Java applets or applications. It relies on Java Server Pages and servlet code to create HTML or other formatted data for the client.
- Java Database Connectivity (JDBC), which is the Java equivalent to ODBC, is the standard interface for Java databases.
- The Java servlet API enhances consistency for developers without requiring a graphical user interface.

3.4 Servlets :

A servlet is a Java technology based web component, managed by a container, that generates dynamic content. Like other Java-based components, servlets are platform independent Java classes that are compiled to platform neutral bytecode that can be loaded dynamically into and run by a Java enabled web server.

Servlets are generic extensions to the Java enabled. A Servlets dynamically loaded module that services requests from a web server. It runs entirely inside the Java Virtual Machine. There are two types of Servlets:

1. Generic Servlets
2. HTTP Servlets

The HTTP Servlets class extends from the Generic Servlets class. The javax.servlet.http contains the generic interfaces and classes that are implemented and extended by all the Servlets, the javax.servlets.http contains classes that are executed when creating HTTP specific Servlets.

A servlet is managed through a well defined life cycle that defines how it is loaded, instantiated and initialized, handles requests from clients, and how it is taken out of service. This life cycle is expressed in the API by the init, service, and destroy methods of the javax.servlet.Servlet interface.

It has 3 important methods

- init()
- service()
- destroy()

Java Servlets do not have a main() method; hence all Servlets must implements the javax.servlet.Servlet interface. Every time a server receives those points to a servelt, it calls the Servlets.service() method.

Two objects that the service method receives are ServletRequest and ServletResponse. The ServletRequest objects holds the information that is being on the servlet whereas the ServeltResponse object is where you place the data you want to send back to the client. When a HTTP Servlet.service() method is invoked, it reads the method type stored in the request and determines which method to invoke on it value.

3.5 Web Server :

The primary function of a web server is to deliver web pages and associated content (e.g. images, style sheets, JavaScripts) to clients. A client, commonly a web browser or web crawler, makes a request for a specific resource using HTTP and the server responds with the content of that resource. The resource is typically a real file on the server's secondary memory, but this is not necessarily the case and depends on how the web server is implemented.

While the primary function is to serve content, a full implementation of HTTP also includes a way of receiving content from clients. This feature is used for submitting web forms, including uploading of files.Many generic web servers also support server-side scripting e.g. Apache HTTP Server.

Some of the main features are:

- A Web Server is a piece of software that enables a website to be viewed using HTTP. HTTP (HyperText Transfer Protocol) is the key protocol for the transfer of data on the web. It dishes out web pages in response to requests from a user sitting at a web browser.
- Web servers aren't limited to serving up static HTML pages; they can also run programs in response to user requests and return the dynamic results to the user's browser.

- Apache Web Server - The Apache HTTP Server Project is a collaborative software development effort aimed at creating a robust, commercial-grade and freely-available source code implementation of an HTTP (Web) server.

3.6 Apache Tomcat :

Apache Tomcat is an open source software implementation of the Java Servlet and JavaServer Pages technologies.

Apache Tomcat is a Java servlet engine and Java Server Pages processor which can run standalone or integrated with the Apache Web server to serve specific virtual paths. This is an aspect of the web that Apache's Tomcat is very good at because Tomcat provides both Java servlet and Java Server Pages (JSP) technologies (in addition to traditional static pages and external CGI programming). The result is that Tomcat is a good choice for use as a web server for many applications.

Apache Tomcat software requirements :--

1. Java 2 Software Development Kit version 1.2.2 or higher from Sun Microsystems, formerly known as the Java Development Kit or JDK.
2. The Apache Tomcat servlet container and JSP environment from the Jakarta Apache site.
3. The latest version of the Apache Web server for Windows binary distribution of the Apache server for versions 1.3.17.

Apache Tomcat is a Web Server implements Servlet and Java Server Pages specification from Java Software and includes many additional features that make it a useful platform for developing and deploying web applications and web services.

Tomcat will operate under any Java Development Kit (JDK) environment that provides a JDK 1.5 (also known as Java2 Standard Edition or J2SE) or later platform. The developer will need a Java Development Kit; as opposed to a Java Runtime Environment so that Servlet other classes and JSP pages can be complied. Tomcat 4 has been extensively tested with JDK 1.5, which is recommended Tomcat to set up a development environment, organize the source code, and then build and test the application.

IMPLEMENTATION DETAILS

CHAPTER - 4

IMPLEMENTATION DETAILS

4.1 MODULES

1. User Interaction Design
2. Server Validation and Process
3. Database Interaction
4. Generating OTP Algorithm
5. GSM Modem Implementation

4.2 MODULE DESCRIPTION

4.2.1 User Interaction Design

Developing the Graphical User Interface of View Layer in Java Server Pages for customers to login and select the items to purchase in web with Java Script Technology. JSP is a widely used general-purpose scripting language that is especially suited for Web development .In our proposed system JSP is the server side scripting language.

Java Server Pages is solution for developing dynamic web sites. JSP provide excellent server side scripting support for creating database driven web applications. JSP enable the developers to directly insert java code into jsp file, this makes the development process very simple and its maintenance also becomes very easy. JSP pages are efficient, it loads into the web servers memory on receiving the request very first time and the subsequent calls are served within a very short period of time. The user can create his account to access the online Shopping . The username and password provided in user interaction design are static . The user can start his shopping by logging with his own username and password.

4.2.2 Server Validation and Process

A server is implemented to generate the OTP on the organization's side. The server consists of a database as described and is connected to a GSM modem for SMS messages exchange. The server application is multithreaded. The first thread is responsible for initializing the database and SMS modem, and listening on the modem for client requests. The second thread is responsible for verifying the SMS information, and generating and sending the OTP. A third thread is used to compare the OTP to the one retrieved using the connection-less method. In order to setup the database, the client must register in person at the organization. The client's mobile phone/SIM card identification factors, e.g. IMEI/IMSI, are retrieved and stored in the database, in addition to the username and PIN. The OTP generating software is installed on the mobile phone. The software is configured to connect to the server's GSM modem in case the SMS option is used. A unique symmetric key is also generated and installed on both the mobile phone and server. Both parties are ready to generate the OTP at that point. Database Interaction Connecting the Database Driver used with Control Layer to validate the Request received from the Tomcat server and responding the further process and is implemented Globalized such that the Database can be changed easily in code.

4.2.3 Database Interaction

A database is needed on the server side to store the client's identification information such as the first name, last name, username, pin, password, mobile IMEI number, IMSI number, unique symmetric key, and the mobile telephone number for each user. The password field will store the hash of the 10 minute password. It will not store the password itself. Should the database be compromised the hashes cannot be reversed in order to get the passwords used to generate those hashes. Hence, the OTP algorithm will not be traced.

4.2.4 Generating OTP Algorithm

In order to secure the system, the generated OTP must be hard to guess, retrieve, or trace by hackers. Therefore, it is very important to develop a secure OTP generating algorithm. Several factors can be used by the OTP algorithm to generate a difficult-to-guess password. Users seem to be willing to use simple factors such as their mobile number and a PIN for services such as authorizing mobile micropayments . Note that these factors must exist on both the mobile phone and server in order for both sides to generate the same password.

In the proposed design, the following factors were chosen:

• **IMEI number:** The term stands for International Mobile Equipment Identity which is unique to each mobile phone allowing each user to be identified by his device. This is accessible on the mobile phone and will be stored in the server's database for each client.

• **IMSI number:** The term stands for International Mobile Subscriber Identity which is a unique number associated with all GSM and Universal Mobile Telecommunications System (UMTS) network mobile phone users. It is stored in the Subscriber Identity Module (SIM) card in the mobile phone. This number will also be stored in the server's database for each client.

• **Username:** Although no longer required because the IMEI will uniquely identify the user anyway. This is used together with the PIN to protect the user in case the mobile phone is stolen.

• **PIN:** This is required to verify that no one other than the user is using the phone to generate the user's OTP. The PIN together with the username is data that only the user knows so even if the mobile phone is stolen the OTP cannot be generated correctly without knowing the user's PIN. Note that the username and the PIN are never stored in the mobile's memory. They are just used to generate the OTP and discarded immediately after that. In order for the PIN to be hard to guess or brute-forced by the hacker, a minimum of
8-characters long PIN is requested with a mixture of upper- and lower-case characters, digits, and symbols.

• **Hour:** This allows the OTP generated each hour to be unique.

• **Minute:** This would make the OTP generated each minute to be unique; hence the OTP would be valid for one minute only and might be inconvenient to the user. An alternative solution is to only use the first
digit of the minute which will make the password valid for ten minutes and will be more convenient for the users, since some users need more than a minute to read and enter the OTP. Note that the software can modified to allow the administrators to select their preferred OTP validity interval.

• **Day:** Makes the OTP set unique to each day of the week.

• **Year/Month/Date:** Using the last two digits of the year and the date and month makes the OTP unique for that particular date.

4.2.5 GSM Modem Implementation

GSM modem is a specialized type of modem which accepts a SIM card, and operates over a subscription to a mobile operator, just like a mobile phone. From the mobile operator perspective, a GSM modem looks just like a mobile phone. Importing the communication Driver and connecting the Modem to the PC with serial port. The mobile phone can request the one time password directly from the server without the need to generate the OTP locally on the mobile phone. In order for the server to verify the identity of the user, the mobile phone sends to the server, via an SMS message, information unique to the user. The server checks the SMS content and if correct, returns a randomly generated OTP to the mobile phone. The user will then have a given amount of time to use the OTP before it expires. Note that this method will require both the client and server to pay for the telecommunication charges of sending the SMS message.

4.2.6 Data Flow Diagram

Level 0 DFD

User Login : Level 0

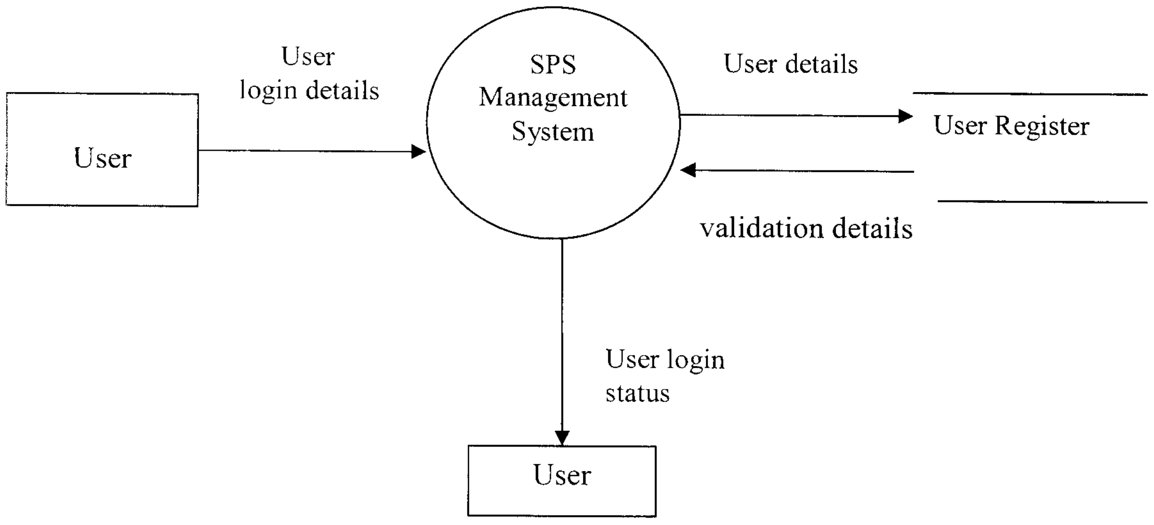


Figure 1 – Level 0 DFD for User Login

Transaction process : Level 0

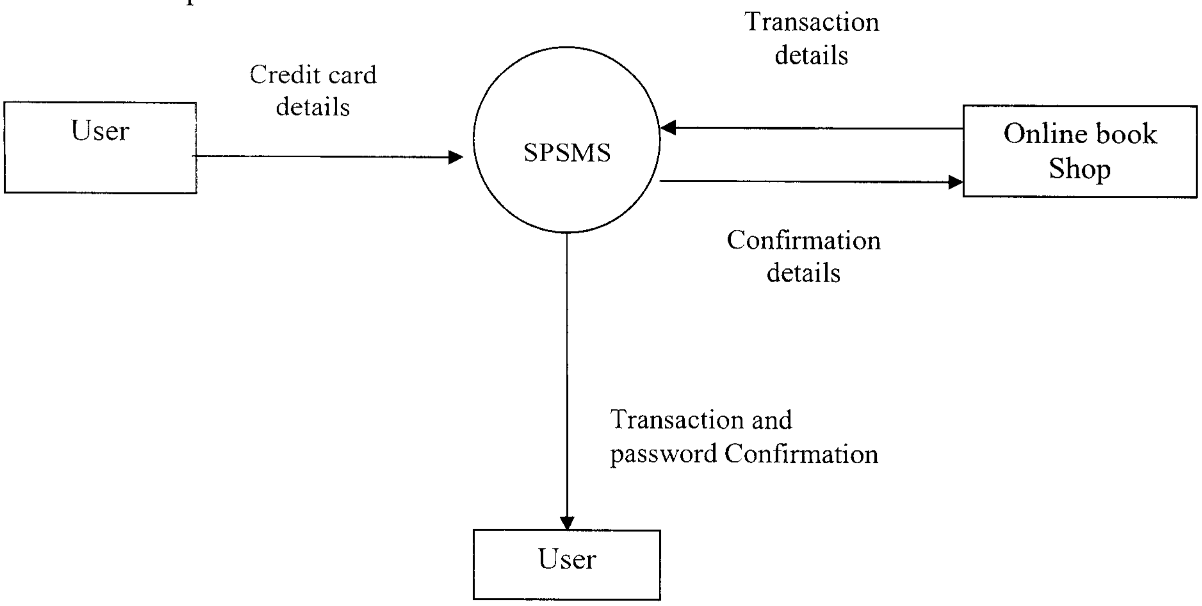
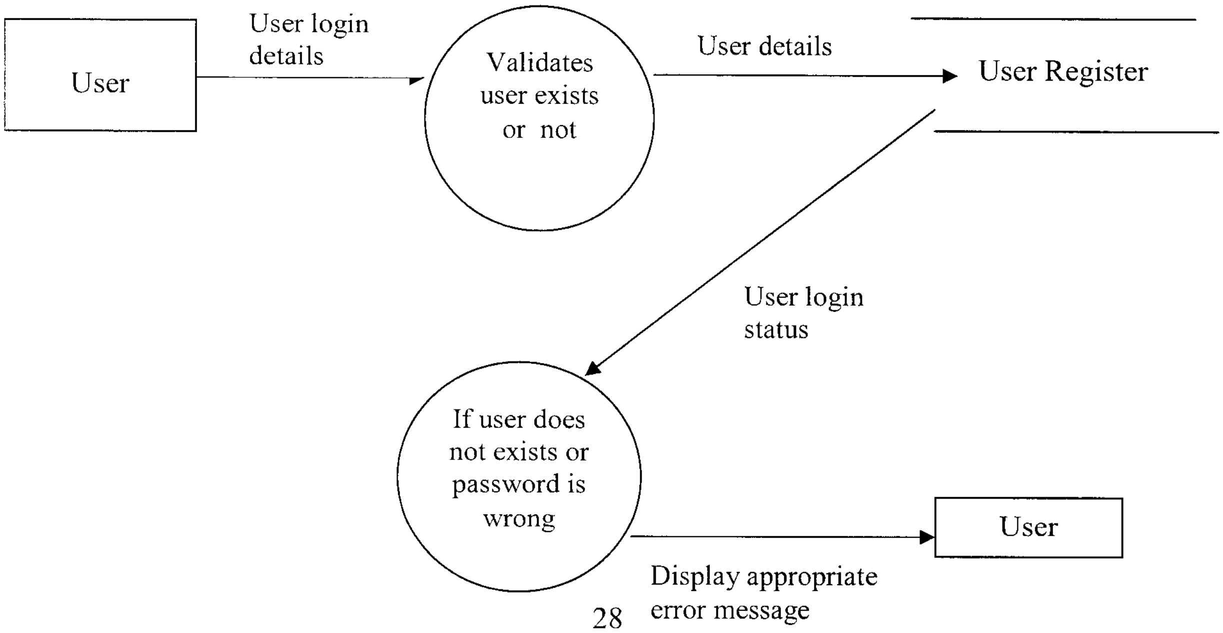


Figure 2: Level 0 DFD for Transaction process

Level 1 DFD

User Login : Level1



User
User login details
Validates user exists or not
User details
User Register
User login status
If user does not exists or password is wrong
User
Display appropriate error message

Figure 3: Level 1 DFD for User Login

User Registration : Level1

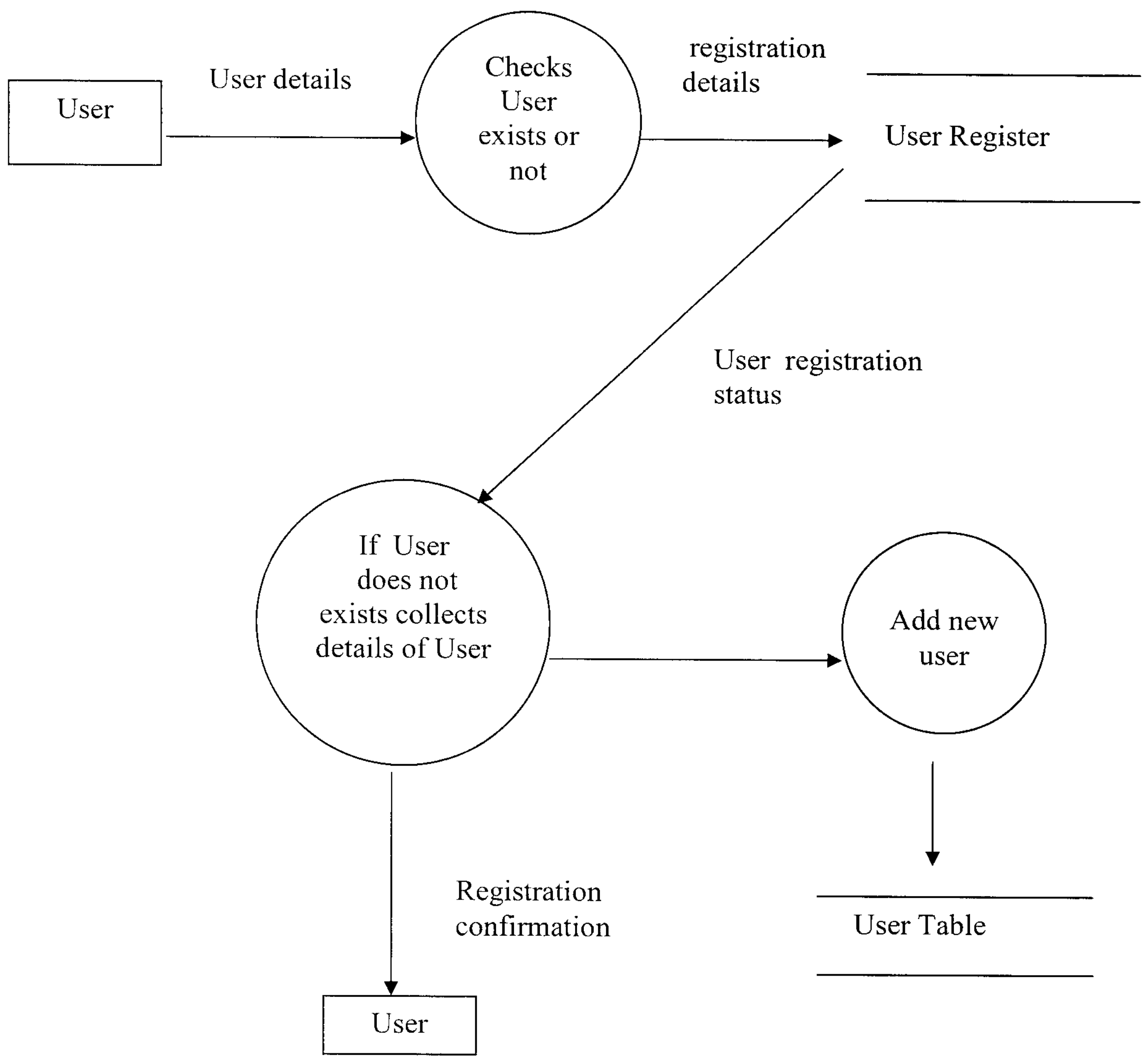


Figure 4: Level 1 DFD for User Registration

Transaction process : Level1

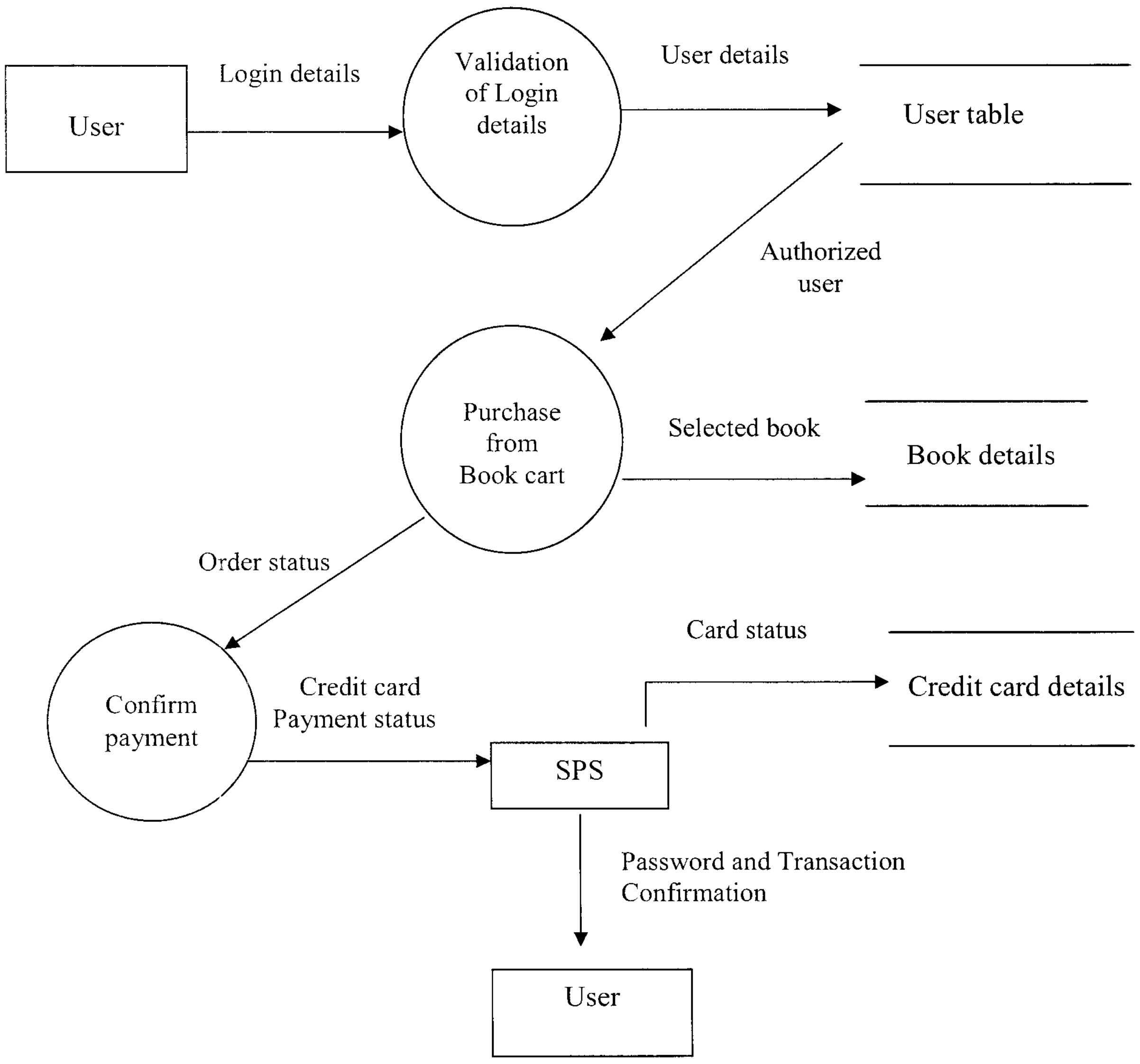
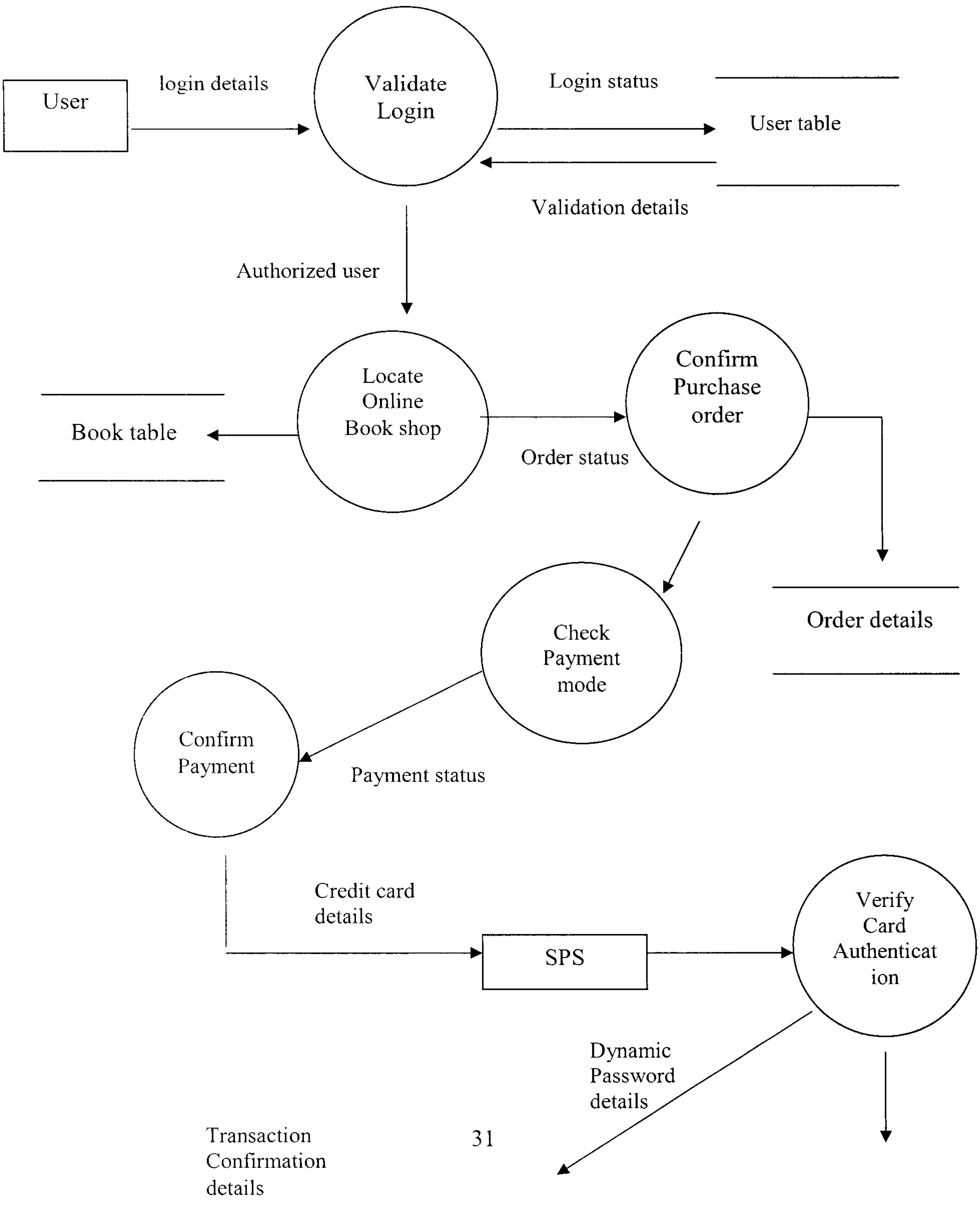
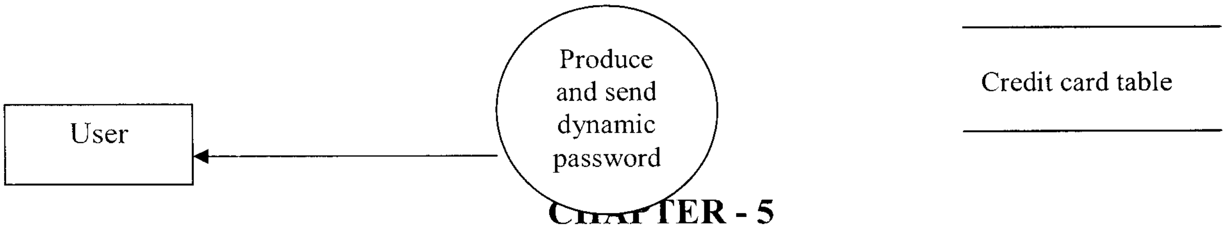


Figure 5: Level 1 DFD for Transaction process

Level 2 DFD





CHAPTER - 5

DESIGN AND DEVELOPMENT PHASE

5.1 DESIGN AND DEVELOPMENT

The design phase is a multi step process which focuses on system creation with the help of user specifications and information gathered in the above phases. It is the phase where the system requirements are translated to operational details.

5.2 SYSTEM DESIGN

System Design is the process of making the newly designed system fully operational and consistent in performance. The following steps have been followed in the implementation of the system.

- Implementation in planning
- User Training

As the part of implementation, the system is taken the site and Loaded on to client's computer. Some of the user's level, exposure to computer etc. A detailed documentation is prepared for the employees and they trained to access the software. These users are trained first and they can run the system for a month.

After installation of software, the hardware specifications are checked. If hardware specifications are satisfactory, then the software is loaded for pilot run.

User training starts at this time itself. Users will be given a user manual, which documents how to use the system and all the exception handling procedures.

5.3 INPUT DESIGN

Input design is the part of overall system design which requires very careful attention. Often the collection of input data is the most expensive part of the system, in terms of both the equipment used and the number of people involved; it is the point of most contact for the users with the computer system; and it is prone to error. If data going into the system are incorrect, then the processing and output will magnify these error.

In this system inputs are given in two ways, the Existing users can directly enter into the system using login form, and new users have to register all their details in the registration form provided.

Input design is the very important part in the project and should be concentrated well as it is prone to error. The data that are to be inserted are to be inserted with care as this plays a very important role. In order to get the meaningful output and to achieve good accuracy the input should be acceptable and understandable by the user.

5.4 OUTPUT DESIGN

Output design plays a very important role in a system. Getting a correct output is a task that has to be concentrated, as a system is validated as a correct one only if it gives the correct output according to the input.Here in this project in all the three days of inductions if the employee has completed all his/her input, then the output shows the status as completed or his status will be pending.

5.5 DATABASE DESIGN

A well-designed database is essential for the performance of the system. Several tables are manipulated for varying purposes.

Table Name	: Registration
Purpose	: Store the user details
Primary Key	: password

1. Registration table

Sl No	Field Name	Type	Width	Constraint	Description
1	Credit Card number	varchar	50	Not null	Unique credit card number
2	Username	varchar	50	Not null	Unique name give by the cardholder
3	Password	varchar	50	Primary key	Unique password
4	Confirm password	varchar	50	Not null	Password is same confirm password.
5	Firstname	varchar	100	Not null	First name of the user
6	Lastname	varchar	200	Not null	Last name of user
7	Pan number	varchar	10	Not null	Unique pan numb

8	Eamil Id	varchar	100	Not null	Unique email id.
9	Mobile number	Int	12	Not null	Unique mobile no.

Table 1 - Registration table

Name of the table : Credit card database as provided by bank.

Purpose : Contains user credit card details.

2. Credit card table

Sl No	Field Name	Type	Width	Constraint	Description
1	Credit card number	Varchar	20	Not null	The unique number of the card
2	User name	Varchar	100	Not null	Name of the cardholder
3	Password	Varchar	10	Primary key	Unique password
4	Retype password	Varchar	10	Not null	Same as that of unique password.
4	Address	Varchar	300	null	Address of the user
5	Pan number	Varchar	10	Not null	Unique pan number
6	Mobile number	Int	20	Not null	User mobile number
7	Pin code	Int	20	Not null	Unique pin number
8	Valid period	Date		Date	Show month year

Table 2- Credit card table

In the following database introducing a online book shop from where the user will purchase the book and start its credit card payment in further steps.

Name of the table : Cart table

Purpose : To show the online transaction : First introduce a online book shop from where the user can check to purchase a book .

3. Book Cart Table

Sl No	Field Name	Type	Width	Constraint	Description
1	BookId	Varchar	30	Primary key	The unique id of the book
2	BookName	Varchar	100	Not null	Name of the book
3	Price	Double		null	Price of the book
4	Quantity	Int	11	null	Number of books
5	SessionId	Varchar	200	null	The session id.

Table 3- Book Cart Table

Name of the table: Categories table

Purpose: The details of the categories offered by the book shop are stored in this table.

4. Categories Tables

Sl No	Field Name	Type	Width	Constraint	Description

1	Category_Id	Int	10	Not null	Unique id
2	Category_Name	Varchar	200	null	Name of category

Table 4 - Categories Tables

Name of the table: Book
Purpose: Individual book details are stored in this table.

5. Books Table

Sl No	Field Name	Type	Width	Constraint	Description
1	isbn	varchar	100	Primary key	Unique code
2	title	varchar	300	null	Title of the book
3	subtitle	varchar	300	null	Subtitle of the book
4	author	varchar	300	null	The name of author
5	publisheddate	varchar	200	null	Date of publication
6	pages	Int	300	null	Number of pages
7	category	Int	10	null	Category of the book
8	newrelease	Int	10	null	Any new release
9	description	varchar	500	null	Description of the book
10	detail	varchar	200	null	Other details
11	price	double		null	The cost of the book

Table 5 - Books Table

Name of the table: Customers Tables

Purpose: The details of the customer is stored in this table.

6. Customers Tables

Sl No	Field Name	Type	Width	Constraint	Description
1	username	varchar	(50)	Primary key	Unique name given by the user
2	password	varchar	(50)	not null	Password of the user
3	firstname	varchar	(50)	null	User first name
4	lastname	varchar	(50)	null	User last name
5	address	varchar	(100)	null	The address of user
6	address1	varchar	(100)	null	Other address of the user
7	city	varchar	(100)	null	The name of the city
8	state	varchar	(100)	null	Name of the state
9	pincode	varchar	(20)	null	The code detail
10	telephone	varchar	(20)	null	The contact number

Table 6 - Customers Tables

Name of the table: Order Items Table

Purpose: All details of the item with its order such as order quantity, orderid are stored in this table.

7. Order Items Table

Sl No	FieldName	Type	Width	Constraint	Description
1	orderid	Int	12	null	The unique orderid
2	bookid	varchar	50	null	The bookid of the ordered book
3	quantity	int	12	null	Number of books ordered.
4	price	double		null	The total price.
5	subtotal	double		null	Subtotal

Table 7 - Order Items Table

Name of the table : Orders Table

Purpose : The details of order from customer such as order date , amount are stored in this table.

8. Orders Table

Sl No	FieldName	Type	Width	Constraint	Description
1	orderid	int	12	Primary key	The unique orderic
2	amount	double		Null	The amount (if any
3	orderdate	varchar	100	Null	Date of the order placed

4	customername	varchar	100	null	The name of the customer

Table 8 - Orders Table

5.6 SYSTEM TESTING & TEST PLANS

Testing techniques and Testing strategies :

There are four testing strategies that are mainly used. These are,

- Unit Testing
- Integration Testing
- Validation Testing
- System Testing

This system was tested using Unit Testing and Integration Testing strategies because these were the most relevant approaches for this project.

Software Testing Techniques:

Specification-based testing:

Specification-based testing aims to test the functionality of software according to the applicable requirements. Thus, the tester inputs data into, and only sees the output from, the test object. This level of testing usually requires thorough test cases to be provided to the tester, who then can simply verify that for a given input, the output value (or behaviour), either "is" or "is not" the same as the expected value specified in the test case. Specification-based testing is necessary, but it is insufficient to guard against certain risks.

A software engineering product can be tested in one of the two ways:

- ➢ Black Box Testing
- ➢ White Box Testing

5.6.1 Black Box Testing:

Knowing a specified function that a product has been designed to perform, determine whether each function is fully operational. Black box testing treats the software as a "black box"—without any knowledge of internal implementation. Black box testing methods include: equivalence partitioning, boundary value analysis, all-pairs testing, fuzz testing, model-based testing, traceability matrix, exploratory testing and specification-based testing. Tests are based on requirements and functionality.

5.6.2 White Box Testing:

Knowing the internal workings of a software product determine whether the internal operations implementing the functions performs according to the specification, and all the internal components have been adequately exercised.

The following types of white box testing exist:

- API testing (application programming interface) - Testing of the application using Public and Private APIs
- Code coverage - creating tests to satisfy some criteria of code coverage (e.g., the test designer can create tests to cause all statements in the program to be executed at least once)
- Fault injection methods - improving the coverage of a test by introducing

faults to test code paths

- Mutation testing methods
- Static testing - White box testing includes all static testing

5.6.3 Unit testing:

We adopt white box testing when using this testing technique. This testing was carried out on individual components of the software that were designed. Each individual module was tested using this technique during the coding phase. Every component was checked to make sure that they adhere strictly to the specifications spelt out in the Data Flow Diagram and ensure that they perform the purpose intended for them.

All the names of the variables are scrutinized to make sure that they truly reflect the element they represent. All the looping mechanisms were verified to ensure that they were as decided. Besides these, we trace through the code manually to capture syntax errors and logical errors.

5.6.4 Integration Testing:

After finishing the Unit Testing, next is the integration testing process. In this testing process we put our focus on identifying the interfaces between components and their functionality as dictated in the Data Flow Diagram. The Bottom Up incremental approach was adopted during these testing. Low-level modules are integrated and combined as a cluster before testing.

The Black Box testing technique was employed here. The interfaces between the components were tested first. This allowed identifying any wrong linkages or parameters passing early in the development process as it can be just passed in a set of data and checked if the result returned is an accepted one.

5.6.5 Validation Testing:

Software testing and validation is achieved through a series of black box test cases. A test procedure defines specific test cases that demonstrate conformity with the requirements. Both, the plan the procedure are designed to ensure that all functional requirements are achieved, documentation is correct and other requirements are met. After each validation test case has been conducted, one of the two possible conditions exists. They are,

- The function or performance characteristics confirm to specification and are accepted.
- A deviation from the specification is uncovered and a deficiency list is created.

The deviation or error discovered at this stage in project can rarely be corrected prior to scheduled completion. It is necessary to negotiate with the customer to establish a method of resolving deficiencies.

5.6.6 System Testing

System testing is a series of different tests whose primary purpose is to fully exercise the computer-based system. Although each test has a different purpose, all the work should verify that all system elements have been properly integrated and perform integrated functions.

System testing also ensures that the project works well in the environment. It traps the errors and allows convenient processing of errors without coming out of the program abruptly.

Recovery testing is done in such a way that failure is forced to a software system and checked whether the recovery is proper and accurate. The performance of system is highly effective.

5.6.7 Web Testing

Our proposed project is of 3 tier applications (developed for Internet). Here we will be having Browser, web server and DB server. The applications accessible in browser would be developed in HTML, JavaScript etc. We can monitor through these applications for the web server would be developed in Java, JSP, JavaScript, (All the manipulations are done on the web server with the help of these programs developed).The DBserver would be having oracle, sql server, sybase, mysql etc. (All data is stored in the database available on the DB server).

The tests performed on these types of applications would be

- User interface testing
- Functionality testing
- Security testing
- Browser compatibility testing
- Load / stress testing
- Storage and data volume testing

A web-application is a three-tier application.
This has a browser (monitors data) [monitoring is done using html-> webserver (manipulates data) [manipulations are done using programming languages or

scripts like adv java,jsp, javascript,] -> database server (stores data) [data storage and retrieval is done using databases like mysql].

The types of tests, which can be applied on this type of applications, are:

1. User interface testing for validation & user friendliness
2. Functionality testing to validate behaviors, i/p, error handling, o/p, manipulations, services levels, order of functionality, links, content of web page & backend coverage's
3. Security testing
4. Browser compatibility
5. Load / stress testing
6. Interoperability testing
7. Storage & data volume testing

CONCLUSION

CONCLUSION

Today single factor authentication is no longer considered secure in the internet and banking world. Two factor authentication has recently been introduced to meet the demand of organization for providing stronger authentication options to its users. In most cases, a hardware token is given to each user for each account. The increasing number of carried tokens and the cost of manufacturing and maintaining is burden. Since many clients carry a mobile phone today at all times. An alternative is to install all the software tokens on the mobile phones. This will help reduce the manufacturing cost and numbers of devices carry by the client. The proposed system has SMS based methodology of running. This method is successfully implemented and tested and shown to be robust and secure. The system has several factors that make it difficult to hack.

CHAPTER - 7
FUTURE ENHANCEMENTS

In order to utilise this application without using GSM Modem ,web services provides an interoperable solution to perform the operations done by the GSM to the mobile user. Our next goal is to enhance the current implementation by providing web service for GSM operations without using GSM modem to the users and hence the benefits of web services can be realized.

APPENDIX

CHAPTER - 8

APPENDIX

SAMPLE CODE

```
//BOOK DATA
package bookstore;
import javax.sql.DataSource;
import java.util.ArrayList;
import java.sql.Connection;
import java.sql.Statement;
import java.sql.ResultSet;
import java.sql.SQLException;
public class BookData {
public static ArrayList getNewReleaseList() throws Exception
{
ArrayList books = new ArrayList();
Connection conn = null;
Statement stmt = null;
ResultSet rs = null;
Book book = null;
Database db = null;
try{
db = new Database();
}catch(SQLException e2){
}
catch(ClassNotFoundException e3){
}


final String query ="select * from books,categories where
books.category=categories.category_id and books.newrelease= true";
rs = db.executeQuery(query);
while (rs.next())
{
book = new Book();
book.setIsbn(rs.getString("isbn"));
book.setTitle(rs.getString("title"));
book.setAuthor(rs.getString("author"));
book.setDescription(rs.getString("description"));
book.setPrice(new Double(rs.getDouble("price")));
book.setThumbnail(rs.getString("thumbnail"));
books.add(book);
}
return books;
}
public static ArrayList getCategoryList() throws Exception
{
ArrayList categories = new ArrayList();
Connection conn = null;
Statement stmt = null;
ResultSet rs = null;
Category category = null;
Database db = null;
try{
db = new Database();
}catch(SQLException e2){
}
catch(ClassNotFoundException e3){
}
final String query ="select * from categories";
rs = db.executeQuery(query);
while (rs.next()) {
category = new Category();
category.setCategoryId(rs.getInt("category_id"));
category.setCategoryName(rs.getString("category_name"));
categories.add(category);
```

```
}
return categories;
}
public static ArrayList getCategoryBooks(String categoryId) throws Exception
{
ArrayList books = new ArrayList();
Connection conn = null;
Statement stmt = null;
ResultSet rs = null;
Book book = null;
Database db = null;
try{
db = new Database();
}catch(SQLException e2){
}
catch(ClassNotFoundException e3){
}
final String query ="select * from books where category="+ categoryId ;
rs = db.executeQuery(query);
while (rs.next()) {
book = new Book();
book.setIsbn(rs.getString("isbn"));
book.setTitle(rs.getString("title"));
book.setAuthor(rs.getString("author"));
book.setDescription(rs.getString("description"));
book.setPrice(new Double(rs.getDouble("price")));
book.setThumbnail(rs.getString("thumbnail"));
books.add(book);
}
return books;
}
public static String getCategoryName(String categoryId) throws Exception
{
Connection conn = null;
Statement stmt = null;
ResultSet rs = null;
String categoryName = null;
Database db = null;
try{
db = new Database();
```

```
}catch(SQLException e2){
}
catch(ClassNotFoundException e3){
}
final String query ="select category_name from categories where
category_id="+categoryId;
rs = db.executeQuery(query);
while (rs.next()) {
categoryName = rs.getString(1);
}
return categoryName;
}
public static Book getBook(String isbn) throws Exception
{
Connection conn = null;
Statement stmt = null;
ResultSet rs = null;
Book book = null;
Database db = null;
try{
db = new Database();
}catch(SQLException e2){
}
catch(ClassNotFoundException e3){
}
final String query ="select * from books where isbn="+isbn;
rs = db.executeQuery(query);
while (rs.next()) {
book = new Book();
book.setIsbn(rs.getString("isbn"));
book.setTitle(rs.getString("title"));
book.setSubTitle(rs.getString("subtitle"));
book.setAuthor(rs.getString("author"));
book.setDetail(rs.getString("detail"));
book.setPrice(new Double(rs.getDouble("price")));
book.setImage(rs.getString("image"));
book.setPages(rs.getInt("pages"));
book.setPublishedDate(rs.getString("publisheddate"));
}
return book;
```

```
}
}
```

```
//CARD DATA
package bookstore;
import java.io.*;
import java.util.*;
import java.sql.*;
import java.sql.DriverManager.*;
import java.sql.Connection.*;
import java.sql.ResultSet.*;
public class CardData {
static Connection con;
static Statement st;
static ResultSet rs;
public static boolean checkAvailablity(String cardno, String pinno,String
month,String year) {
try
{
Class.forName("com.mysql.jdbc.Driver");
con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/login","root","passwo
rd");
st = con.createStatement();
rs = st.executeQuery("select * from log where cardnum='"+cardno+"'&&
pinnum='"+pinno+"'&& month='"+month+"'&& year='"+year+"'");
if(rs.next()) {
System.out.println("CardData : true");
return true;
}else {
System.out.println("CardData : false");
return false;
}
}catch (Exception s)  {
System.out.println("SQL statement is not executed!"+s);
return false;
}
}
}
//CART DATA
package bookstore;
```

```
import javax.sql.DataSource;
import java.util.ArrayList;
import java.sql.Connection;
import java.sql.Statement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.*;
import java.util.*;
public class CartData {
public static int addToCart(CartItem cartItem,String sesId) throws Exception
{
Connection conn = null;
Statement stmt = null;
ResultSet rs = null;
Book book = null;
int result = 0;
String bookId = null;
String bookName = null;
Double price = 0.0;
Double subtotal = 0.0;
int quantity = 0;
Database db = null;
try{
db = new Database();
}catch(SQLException e2){
}
catch(ClassNotFoundException e3){
}
bookId = cartItem.getBookId();
bookName = cartItem.getBookName();
price = cartItem.getUnitPrice();
subtotal = cartItem.getSubTotal();
quantity = cartItem.getQuantity();
System.out.println("q1 = "+quantity);
System.out.println("s1 = "+subtotal);
final String query1 = "select quantity from cart where sessionid='"+sesId+"' and
bookid='"+bookId+"'";
rs = db.executeQuery(query1);
if(rs.next()) {
System.out.println("rsq "+rs.getInt(quantity));
```

```
quantity = rs.getInt(quantity) + 1;
System.out.println(quantity);
subtotal = price * quantity;
System.out.println(subtotal);
final String query2 = "update cart set quantity="+quantity+" ,
subtotal="+subtotal+" where sessionid='"+sesId+"' and bookid='"+bookId+"'";
System.out.println(query2);
result = db.executeUpdate(query2);
System.out.println("5");
}else {
System.out.println("6");
final String query ="insert into cart values('"+bookId
+"','"+bookName+"',"+price+","+quantity+","+subtotal+",'"+sesId+"')";
result = db.executeUpdate(query);
System.out.println("7");
}
System.out.println(result);
return result;
}
public static ArrayList getCart(String sesId) throws Exception
{
Connection conn = null;
Statement stmt = null;
ResultSet rs = null;
CartItem cartItem = null;
ArrayList cart = new ArrayList();
Database db = null;
try{
db = new Database();
}catch(SQLException e2){
}
catch(ClassNotFoundException e3){
}
final String query = "select * from cart where sessionid='"+sesId+"'";
rs = db.executeQuery(query);
if(rs.next()) {
do{
cartItem = new CartItem();
cartItem.setBookId(rs.getString("bookid"));
cartItem.setBookName(rs.getString("bookname"));
cartItem.setQuantity(rs.getInt("quantity"));
cartItem.setUnitPrice(rs.getDouble("price"));
cartItem.setSubTotal(rs.getDouble("subtotal"));
cart.add(cartItem);
}while(rs.next());
} else {
cart = null;
}
return cart;
}
public static void removeFromCart(String isbn, String sesId) throws Exception
{
Connection conn = null;
Statement stmt = null;
Database db = null;
try{
db = new Database();
}catch(SQLException e2){
}
catch(ClassNotFoundException e3){
}
final String query = "delete from cart where sessionid='"+sesId+"'and
bookid='"+isbn+"'";
db.executeUpdate(query);
}
public static void updateCart(String isbn,int quantity, String sesId) throws
Exception
{
Connection conn = null;
Statement stmt = null;
ResultSet rs = null;
int result = 0;
Database db = null;
try{
db = new Database();
}catch(SQLException e2){
}
catch(ClassNotFoundException e3){
}
```

```
final String query = "select price from cart where sessionid='"+sesId+"'and
bookid='"+isbn+"'";
rs = db.executeQuery(query);
if(rs.next()){
Double unitPrice = rs.getDouble("price");
Double newSubTotal = quantity * unitPrice;
final String query2 = "update cart set quantity="+quantity+" ,
subtotal="+newSubTotal+" where sessionid='"+sesId+"' and bookid='"+isbn+"'";
result = db.executeUpdate(query2);
}
}
public static int getOrderId() throws Exception
{
Connection conn = null;
Statement stmt = null;
ResultSet rs = null;
int orderId = 0;
Database db = null;
try{
db = new Database();
}catch(SQLException e2){
}
catch(ClassNotFoundException e3){
}
final String query1 = "insert into orders
(amount,orderdate,customername)values(0.0,'','')";
db.executeUpdate(query1);
final String query = "select max(orderid) from orders";
rs = db.executeQuery(query);
if(rs.next()) {
orderId = rs.getInt(1);
}
return orderId;
}
public static void completeOrder(Order order,ArrayList cart,String userName)
throws Exception
{
Connection conn = null;
Statement stmt = null;
ResultSet rs = null;
```

```
Double total  = 0.0;
Database db = null;
try{
db = new Database();
}catch(SQLException e2){
}
catch(ClassNotFoundException e3){
}
SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
final String query = "update orders set amount ="+order.getAmount()+",orderdate
= '"+sdf.format(new Date())+"',customername ='"+userName+"' where
orderid="+order.getOrderId();
int i = db.executeUpdate(query);
if(i>0) {
for (Iterator iter = cart.iterator(); iter.hasNext();) {
CartItem cartItem = (CartItem) iter.next();
db.executeUpdate("insert into orderitems
values("+order.getOrderId()+",'"+cartItem.getBookId()+"',"+cartItem.getQuantity(
)+","+cartItem.getUnitPrice()+","+cartItem.getSubTotal()+")");
}
}
}
public static void removeCart(String sesId) throws Exception
{
Connection conn = null;
Statement stmt = null;
Database db = null;
try{
db = new Database();
}catch(SQLException e2){
}
catch(ClassNotFoundException e3){
}
final String query = "delete from cart where sessionid='"+sesId+"'";
db.executeUpdate(query);
}
}
```

```
//CART ITEM
package bookstore;
```

```
public class CartItem {
private int quantity = 0;
private double unitPrice = 0.0;
private String bookId = null;
private String bookName = null;
private double subTotal = 0.0;
public void setQuantity(int n){
quantity = n;
}
public int getQuantity(){
return quantity;
}
public void setUnitPrice(double p) {
unitPrice = p;
}
public double getUnitPrice() {
return unitPrice;
}
public void setBookId(String i){
bookId = i;
}
public String getBookId() {
return bookId;
}
public void setBookName(String i){
bookName = i;
}
public String getBookName() {
return bookName;
}
public void setSubTotal(double t) {
subTotal = t;
}
public double getSubTotal() {
return subTotal;
}
}
```

```
//CATEGORY
package bookstore;
public class Category
```

```
{
private int categoryId;
private String categoryName;
public Category()
{
}
/** category name */
public String getCategoryName()
{
return categoryName;
}
public void setCategoryName(String categoryName)
{
this.categoryName = categoryName;
}
/** category */
public int getCategoryId()
{
return categoryId;
}
public void setCategoryId(int categoryId)
{
this.categoryId = categoryId;
}
}
```

```
//CATEGORY BOOKS
package bookstore;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.*;
import javax.sql.*;
import javax.naming.*;
public class CategoryBooks extends HttpServlet {
public void doGet (HttpServletRequest request, HttpServletResponse response) {
doPost(request, response);
}
public void doPost (HttpServletRequest request, HttpServletResponse response) {
ArrayList categoryBooks;
String categoryName;
String categoryId = request.getParameter("c");
try {
categoryBooks  = BookData.getCategoryBooks(categoryId);
categoryName = BookData.getCategoryName(categoryId);
request.setAttribute("categoryName",categoryName);
request.setAttribute("categoryBooks",categoryBooks);

System.out.println("Error category 1111");
getServletConfig().getServletContext().getRequestDispatcher("/category.jsp").forw
ard(request, response);
}
catch ( Exception e ) {
System.out.println("Error category"+e);
}
}
//CHECKOUT FORM
package bookstore;
import java.util.*;
public class CheckoutForm
{
private String cardNo;
private String pinNo;
private String expMonth;
private String expYear;
private Hashtable errors;
public CheckoutForm()
{
cardNo = "";
expMonth = "";
expYear ="";
pinNo ="";
errors = new Hashtable();
}
public void setCardNo(String cardNo)
{
this.cardNo = cardNo;
}
public String getCardNo()
{
return cardNo;
```

```
}
public void setPinNo(String pinNo)
{
this.pinNo = pinNo;
}
public String getPinNo()
{
return pinNo;
}
public void setExpMonth(String expMonth)
{
this.expMonth = expMonth;
}
public String getExpMonth()
{
return expMonth;
}
public void setExpYear(String expYear)
{
this.expYear = expYear;
}
public String getExpYear()
{
return expYear;
}
public void setErrors(String key, String msg)
{
errors.put(key,msg);
}
public boolean validate()
{
boolean allOk=true;
if (cardNo.equals(""))
{
errors.put("cardNo","Enter your Card No");
cardNo="";
allOk=false;
}
else
{
```

```
try
{
Long i = Long.parseLong(cardNo);
if(cardNo.length() != 16)
{
errors.put("cardNo","Card No must be 16 Digits");
cardNo="";
allOk=false;
}
}
catch (NumberFormatException e)
{
errors.put("cardNo","Card No must be Numeric");
cardNo="";
allOk=false;
}
}
if (pinNo.equals(""))
{
errors.put("pinNo","Enter your Pin No");
pinNo="";
allOk=false;
}
else
{
try
{
Long j=Long.parseLong(pinNo);
if(pinNo.length() != 4)
{
errors.put("pinNo","pin No must be 4 Digits");
pinNo="";
allOk=false;
}
} catch (NumberFormatException e)
{
errors.put("pinNo","pin No must be Numeric");
pinNo="";
allOk=false;
}
```

```
}
f (expMonth.equals(""))
{
rrors.put("expMonth","Enter your Expiary Month");
inNo="";
llOk=false;
}else
{
}
if (expYear.equals(""))
{
errors.put("expYear","Enter your Expiary year");
pinNo="";
allOk=false;
}else
{
}
return allOk;
public String getErrorMsg(String s)
{
String errorMsg =(String)errors.get(s.trim());
return (errorMsg == null) ? "":errorMsg;
}
}
```

```
//COMLETE ORDER
package bookstore;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.*;
import javax.sql.*;
import javax.naming.*;
import java.text.*;


public class CompleteOrder extends HttpServlet
{
HttpSession session;
public void doGet (HttpServletRequest request, HttpServletResponse response) {
doPost(request, response);
}
```

```
public void doPost (HttpServletRequest request, HttpServletResponse response) {
CheckoutForm checkoutForm;
Order order;
ArrayList cart;
Customer customer;
session = request.getSession();
try
{
checkoutForm = new CheckoutForm();
checkoutForm.setCardNo(request.getParameter("cardNo"));
checkoutForm.setPinNo(request.getParameter("pinNo"));
checkoutForm.setExpMonth(request.getParameter("expMonth"));
checkoutForm.setExpYear(request.getParameter("expYear"));
session.setAttribute("checkoutForm",checkoutForm);
if(checkoutForm.validate())
{
boolean isValidate = CardData.checkAvailablity(checkoutForm.getCardNo(),
checkoutForm.getPinNo(),
checkoutForm.getExpMonth(),
checkoutForm.getExpYear());
HttpSession session = request.getSession();
order = (Order) session.getAttribute("order");
cart = (ArrayList) session.getAttribute("cart");
customer =(Customer) session.getAttribute("customer");
CartData.completeOrder(order,cart,customer.getUserName());
CartData.removeCart(session.getId());
session.removeAttribute("order");
session.removeAttribute("cart");
if(isValidate)
{
getServletConfig().getServletContext().getRequestDispatcher("/manif").forward(re
quest, response);
}else
{
getServletConfig().getServletContext().getRequestDispatcher("/invalid.jsp").forwa
rd(request, response);
}
} else {
getServletConfig().getServletContext().getRequestDispatcher("/checkout.jsp").for
ward(request, response);
}
}catch ( Exception e ) {
System.out.println("oc " +e);
}
}
}
```

```
//CUSTOMER
package bookstore;
public class Customer
{
protected String userName;
protected String password;
protected String firstName;
protected String lastName;
protected String address;
protected String address1;
protected String city;
protected String state;
protected String zipCode;
protected String telephone;
public void setUserName(String userName)
{
this.userName = userName;
}
public String getUserName()
{
return userName;
}
public void setPassword(String password)
{
this.password = password;
}
public String getPassword()
{
return password;
}
public void setFirstName(String firstName)
{
```

```
this.firstName = firstName;
}
public String getFirstName()
{
return firstName;
}
public void setLastName(String lastName)
{
this.lastName = lastName;
}
public String getLastName()
{
return lastName;
}
public void setAddress(String address)
{
this.address = address;
}
public String getAddress()
{
return address;
}
public void setAddress1(String address1)
{
this.address1 = address1;
}
public String getAddress1()
{
return address1;
}
public void setCity(String city)
{
this.city = city;
}
public String getCity()
{
return city;
}
public void setState(String state)
{
```

```
this.state = state;
}
public String getState()
{
return state;
}
public void setZipCode(String zipCode)
{
this.zipCode = zipCode;
}
public String getZipCode()
{
return zipCode;
}
public void setTelephone(String telephone)
{
this.telephone = telephone;
}
public String getTelephone()
{
return telephone;
}
}
```

```
//CUSTOMER DATA
package bookstore;
import java.util.*;
import javax.sql.DataSource;
import java.util.ArrayList;
import java.sql.Connection;
import java.sql.Statement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.*;
public class CustomerData {
public static Customer loginValidation(String username,String password) throws
Exception
{
Connection conn = null;
Statement stmt = null;
```

```
ResultSet rs = null;
ResultSet rs1 = null;
Customer customer = null;
Database db = null;
try{
db = new Database();
}catch(SQLException e2){
}
catch(ClassNotFoundException e3){
}
final String query ="select * from customers where username='" + username + "'
and password='" + password + "'";
rs = db.executeQuery(query);
if ( rs.next() ) {
customer = new Customer();
customer.setUserName(rs.getString("username"));
customer.setFirstName(rs.getString("firstname"));
}
else {
throw new Exception("Invalid Username and Password!");
}
return customer;
}
public static void registerCustomer(Customer customer) throws Exception
{
Connection conn = null;
Statement stmt = null;
ResultSet rs = null;
Database db = null;
try{
db = new Database();
}catch(SQLException e2){
}
catch(ClassNotFoundException e3){
}
final String query1 = "select * from customers where
username='"+customer.getUserName()+"'";
rs = db.executeQuery(query1);
if(rs.next()) {
throw new Exception("Username already exists.");
} else {
StringBuffer sqlString =
new StringBuffer("insert into customers ");
sqlString.append("values ('"
+ customer.getUserName() + "', ");
sqlString.append("'" +
customer.getPassword() + "', ");
sqlString.append("'"
+ customer.getFirstName() + "', ");
sqlString.append("'"
+ customer.getLastName() + "', ");
sqlString.append("'"
+ customer.getAddress() + "', ");
sqlString.append("'"
+ customer.getAddress1() + "', ");
sqlString.append("'"
+ customer.getCity() + "', ");
sqlString.append("'"
+ customer.getState() + "', ");
sqlString.append("'"
+ customer.getZipCode() + "', ");
sqlString.append("'"
+ customer.getTelephone() + "')");
final String query =sqlString.toString();
int result = db.executeUpdate(query);
}
}
}

//CUSTOMER FORM
package bookstore;
import java.util.*;
public class CustomerForm {
private String userName;
private String password;
private String firstName;
private String lastName;
private String address;
private String address1;
private String city;
private String state;
private String zipCode;
private String telephone;
private Hashtable errors;
public CustomerForm() {
userName = "";
password = "";
firstName = "";
lastName = "";
address = "";
address1 = "";
city = "";
state = "";
zipCode = "";
telephone = "";
errors = new Hashtable();
}
public void setUserName(String userName)
{
this.userName = userName;
}
public String getUserName()
{
return userName;
}
public void setPassword(String password)
{
this.password = password;
}
public String getPassword()
{
return password;
}
public void setFirstName(String firstName)
{
this.firstName = firstName;
}
public String getFirstName()
{
return firstName;
```

```
}
public void setLastName(String lastName)
{
this.lastName = lastName;
}
public String getLastName()
{
return lastName;
}
public void setAddress(String address)
{
this.address = address;
}
public String getAddress()
{
return address;
}
public void setAddress1(String address1)
{
this.address1 = address1;
}
public String getAddress1()
{
return address1;
}
public void setCity(String city)
{
this.city = city;
}
public String getCity()
{
return city;
}
public void setState(String state)
{
this.state = state;
}
public String getState()
{
return state;
```

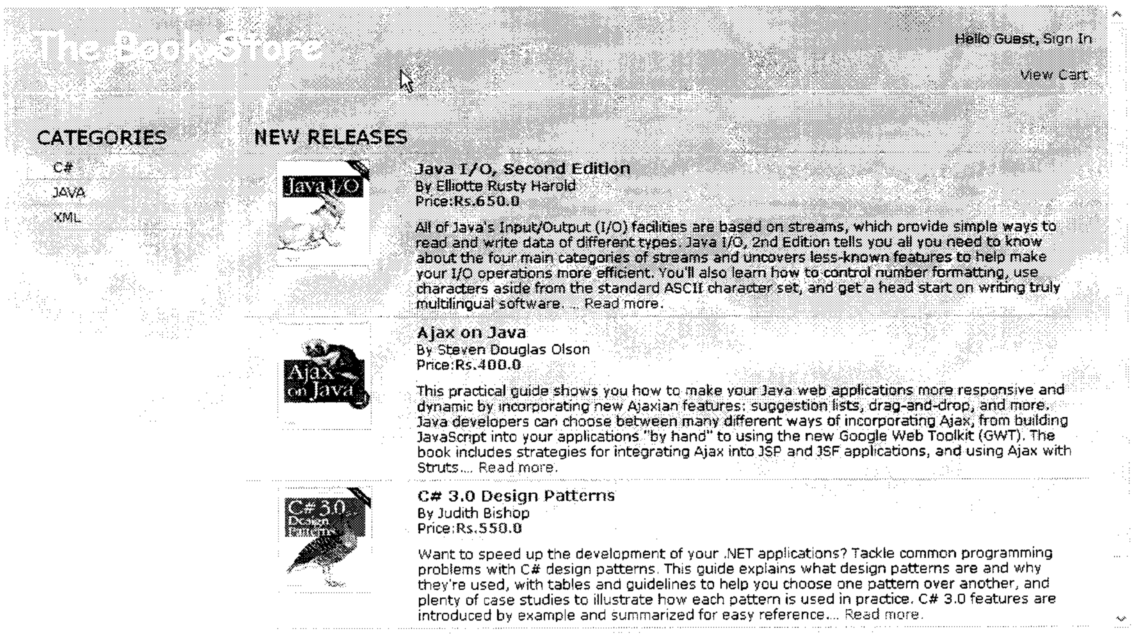
```
}
public void setZipCode(String zipCode)
{
this.zipCode = zipCode;
}
public String getZipCode()
{
return zipCode;
}
public void setTelephone(String telephone)
{
this.telephone = telephone;
}
public String getTelephone()
{
return telephone;
}
public void setErrors(String key, String msg) {
errors.put(key,msg);
}
public boolean validate() {
boolean allOk=true;
if (userName.equals("")) {
errors.put("userName","User name required");
userName="";
allOk=false;
}
if (password.equals("")) {
errors.put("password","Password required");
password="";
allOk=false;
}
if (firstName.equals("")) {
errors.put("firstName","First name required");
firstName="";
allOk=false;
}
if (lastName.equals("")) {
errors.put("lastName","Last name required");
lastName="";
allOk=false;
}
if (address.equals("")) {
errors.put("address","Address required");
address="";
allOk=false;
}
if (address1.equals("")) {
errors.put("address1","Address1 required");
address1="";
allOk=false;
}
if (city.equals("")) {
errors.put("city","City name required");
city="";
allOk=false;
}
if (state.equals("")) {
errors.put("state","State required");
state="";
allOk=false;
}
if (zipCode.equals("")) {
errors.put("zipCode","Postal Code required");
zipCode="";
allOk=false;
}
if (telephone.equals("")) {
errors.put("telephone","Telephone No required");
telephone="";
allOk=false;
}else {
try {
int i = Integer.parseInt(telephone);
} catch (NumberFormatException e) {
errors.put("telephone","Telephone No must be Numeric");
telephone="";
allOk=false;
}
}
```

```
return allOk;
}
public String getErrorMsg(String s) {
String errorMsg =(String)errors.get(s.trim());
return (errorMsg == null) ? "":errorMsg;
}
```

SCREENSHOTS

SCREENSHOTS

Home page:



The Book Store

Hello Guest, Sign In

View Cart

CATEGORIES

C#

JAVA

XML

NEW RELEASES

Java I/O, Second Edition
By Elliotte Rusty Harold
Price:Rs.650.0

All of Java's Input/Output (I/O) facilities are based on streams, which provide simple ways to read and write data of different types. Java I/O, 2nd Edition tells you all you need to know about the four main categories of streams and uncovers less-known features to help make your I/O operations more efficient. You'll also learn how to control number formatting, use characters aside from the standard ASCII character set, and get a head start on writing truly multilingual software. ... Read more.

Ajax on Java
By Steven Douglas Olson
Price:Rs.400.0

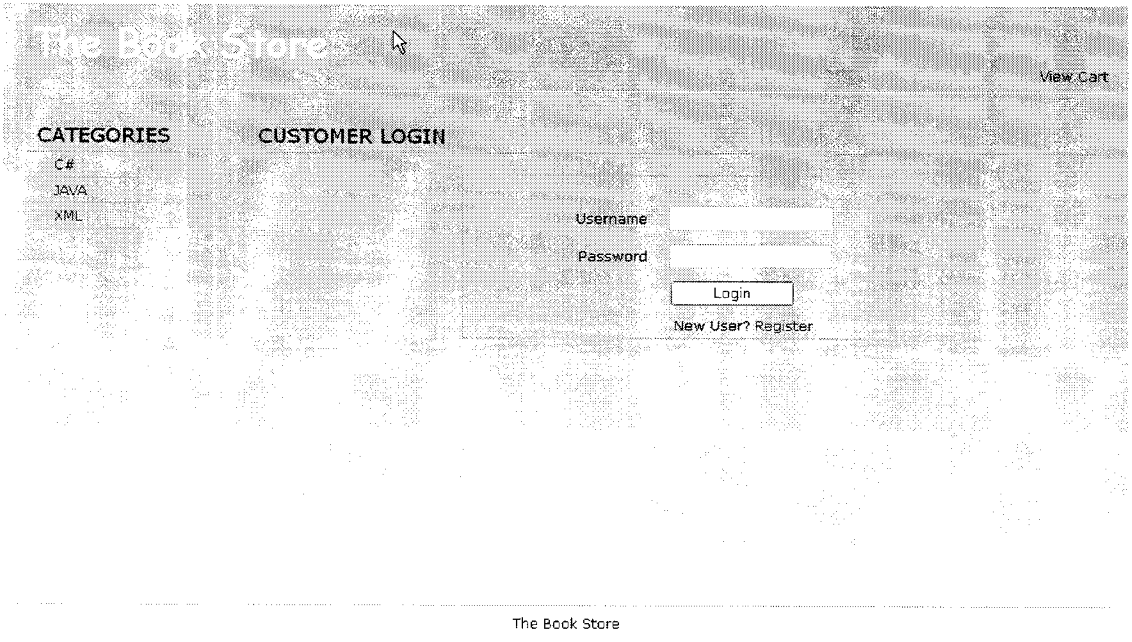
This practical guide shows you how to make your Java web applications more responsive and dynamic by incorporating new Ajaxian features: suggestion lists, drag-and-drop, and more. Java developers can choose between many different ways of incorporating Ajax, from building JavaScript into your applications "by hand" to using the new Google Web Toolkit (GWT). The book includes strategies for integrating Ajax into JSP and JSF applications, and using Ajax with Struts.... Read more.

C# 3.0 Design Patterns
By Judith Bishop
Price:Rs.550.0

Want to speed up the development of your .NET applications? Tackle common programming problems with C# design patterns. This guide explains what design patterns are and why they're used, with tables and guidelines to help you choose one pattern over another, and plenty of case studies to illustrate how each pattern is used in practice. C# 3.0 features are introduced by example and summarized for easy reference... Read more.

Fig.1- Home page

Customer Login:



The Book Store

View Cart

CATEGORIES

C#

JAVA

XML

CUSTOMER LOGIN

Username

Password

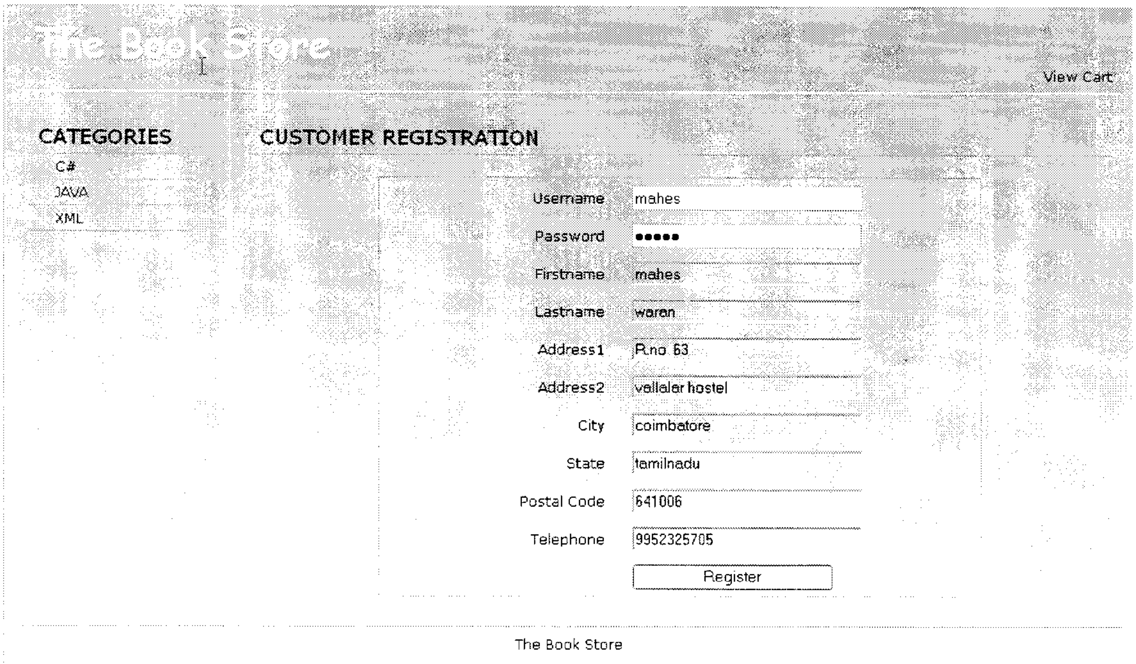
Login

New User? Register

The Book Store

Fig.2- Customer Login

Customer Registration:



The Book Store

View Cart

CATEGORIES

C#

JAVA

XML

CUSTOMER REGISTRATION

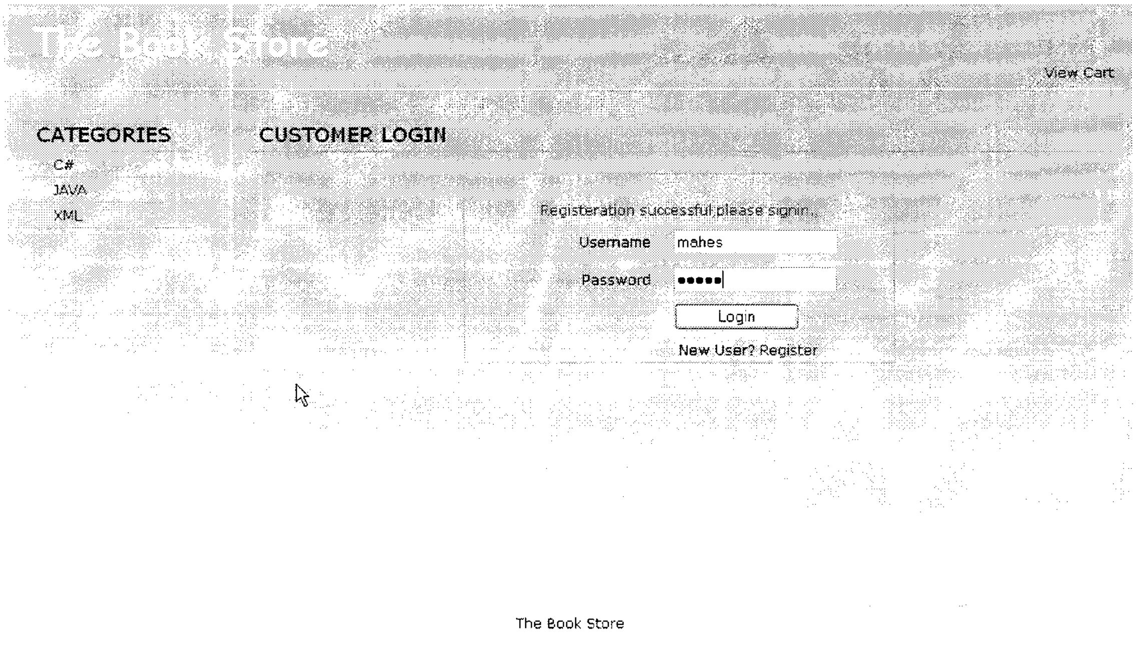
Username	mahes
Password	•••••
Firstname	mahes
Lastname	waran
Address1	R.no 63
Address2	vellalar hostel
City	coimbatore
State	tamilnadu
Postal Code	641006
Telephone	9952325705

Register

The Book Store

Fig.3- Customer Registration

Customer Login:



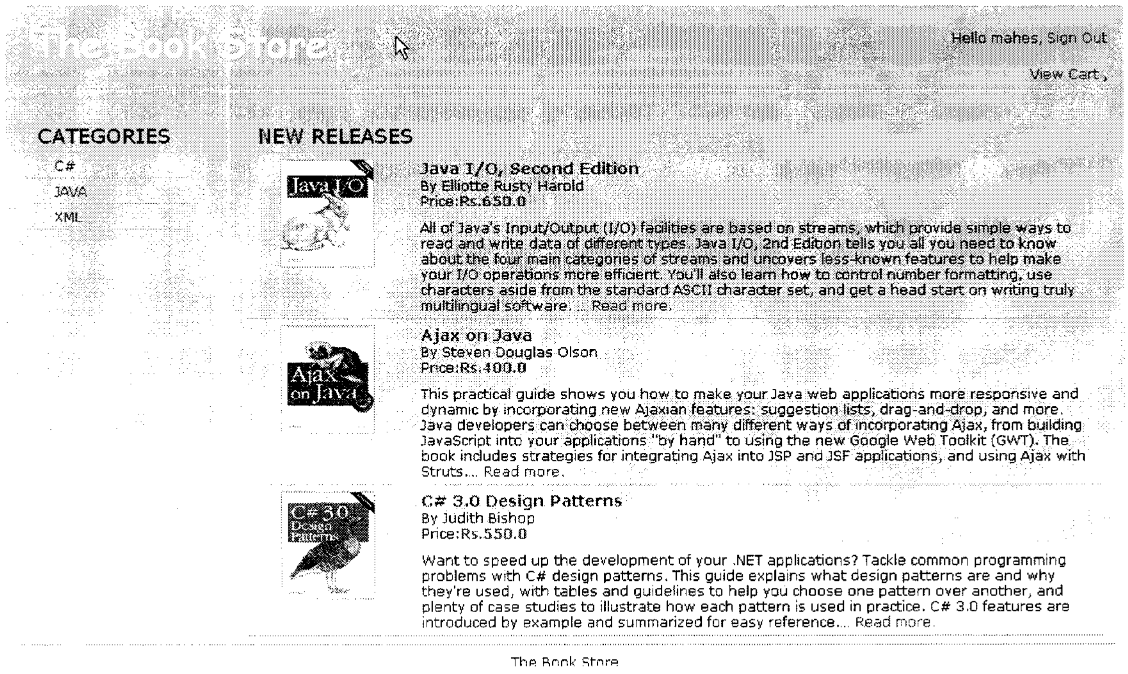
Registeration successful please signin..

Username mahes

Password •••••

Fig.4- Customer Login

Book Details :



Java I/O, Second Edition
By Elliotte Rusty Harold
Price:Rs.650.0

All of Java's Input/Output (I/O) facilities are based on streams, which provide simple ways to read and write data of different types. Java I/O, 2nd Edition tells you all you need to know about the four main categories of streams and uncovers less-known features to help make your I/O operations more efficient. You'll also learn how to control number formatting, use characters aside from the standard ASCII character set, and get a head start on writing truly multilingual software. ... Read more.

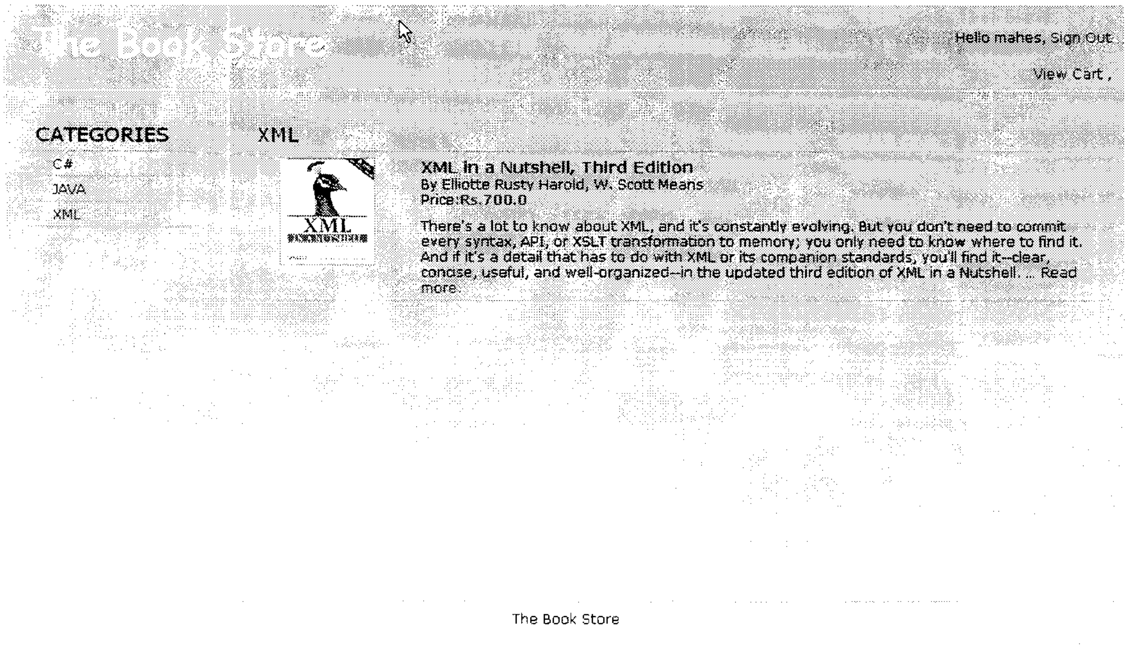
Ajax on Java
By Steven Douglas Olson
Price:Rs.400.0

This practical guide shows you how to make your Java web applications more responsive and dynamic by incorporating new Ajaxian features: suggestion lists, drag-and-drop, and more. Java developers can choose between many different ways of incorporating Ajax, from building JavaScript into your applications "by hand" to using the new Google Web Toolkit (GWT). The book includes strategies for integrating Ajax into JSP and JSF applications, and using Ajax with Struts.... Read more.

C# 3.0 Design Patterns
By Judith Bishop
Price:Rs.550.0

Want to speed up the development of your .NET applications? Tackle common programming problems with C# design patterns. This guide explains what design patterns are and why they're used, with tables and guidelines to help you choose one pattern over another, and plenty of case studies to illustrate how each pattern is used in practice. C# 3.0 features are introduced by example and summarized for easy reference.... Read more.

Fig.5- Book Details



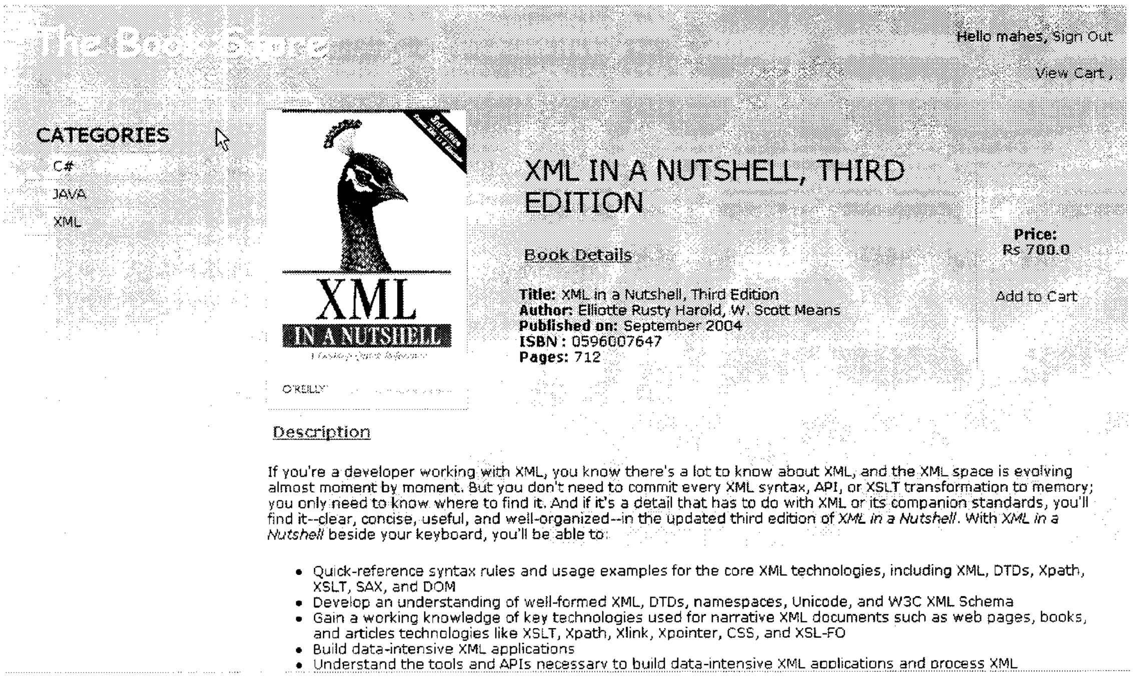
XML in a Nutshell, Third Edition
By Elliotte Rusty Harold, W. Scott Means
Price:Rs.700.0

There's a lot to know about XML, and it's constantly evolving. But you don't need to commit every syntax, API, or XSLT transformation to memory; you only need to know where to find it. And if it's a detail that has to do with XML or its companion standards, you'll find it--clear, concise, useful, and well-organized--in the updated third edition of XML in a Nutshell. .. Read more.

The Book Store

Fig.6- Book Details

Book Description:



XML IN A NUTSHELL, THIRD EDITION

Book Details

Title: XML in a Nutshell, Third Edition
Author: Elliotte Rusty Harold, W. Scott Means
Published on: September 2004
ISBN : 0596007647
Pages: 712

Price:
Rs 700.0

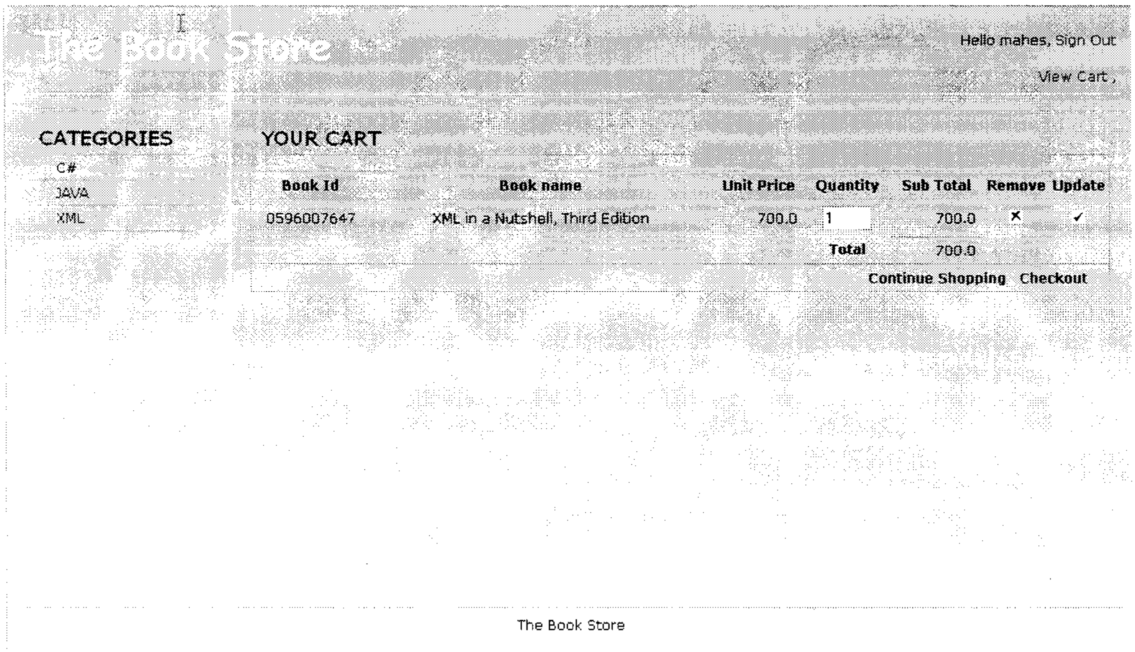
Description

If you're a developer working with XML, you know there's a lot to know about XML, and the XML space is evolving almost moment by moment. But you don't need to commit every XML syntax, API, or XSLT transformation to memory; you only need to know where to find it. And if it's a detail that has to do with XML or its companion standards, you'll find it--clear, concise, useful, and well-organized--in the updated third edition of *XML in a Nutshell*. With *XML in a Nutshell* beside your keyboard, you'll be able to:

- Quick-reference syntax rules and usage examples for the core XML technologies, including XML, DTDs, Xpath, XSLT, SAX, and DOM
- Develop an understanding of well-formed XML, DTDs, namespaces, Unicode, and W3C XML Schema
- Gain a working knowledge of key technologies used for narrative XML documents such as web pages, books, and articles technologies like XSLT, Xpath, Xlink, Xpointer, CSS, and XSL-FO
- Build data-intensive XML applications
- Understand the tools and APIs necessary to build data-intensive XML applications and process XML

Fig.7- Book Description

Cart:


The Book Store

Hello mahes, Sign Out

View Cart ,

CATEGORIES

C#

JAVA

XML

YOUR CART

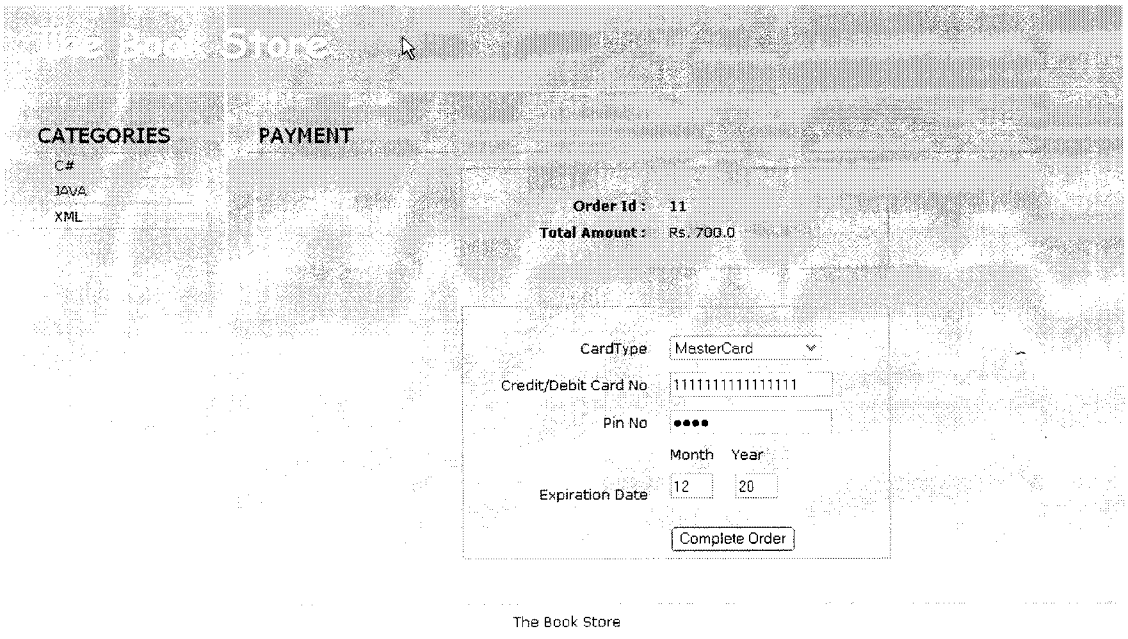
Book Id	Book name	Unit Price	Quantity	Sub Total	Remove	Update
0596007647	XML in a Nutshell, Third Edition	700.0	1	700.0	×	✓
			Total	700.0		

Continue Shopping Checkout

The Book Store

Fig.8- Cart

Order Checkout:


The Book Store

CATEGORIES

C#

JAVA

XML

PAYMENT

Order Id : 11

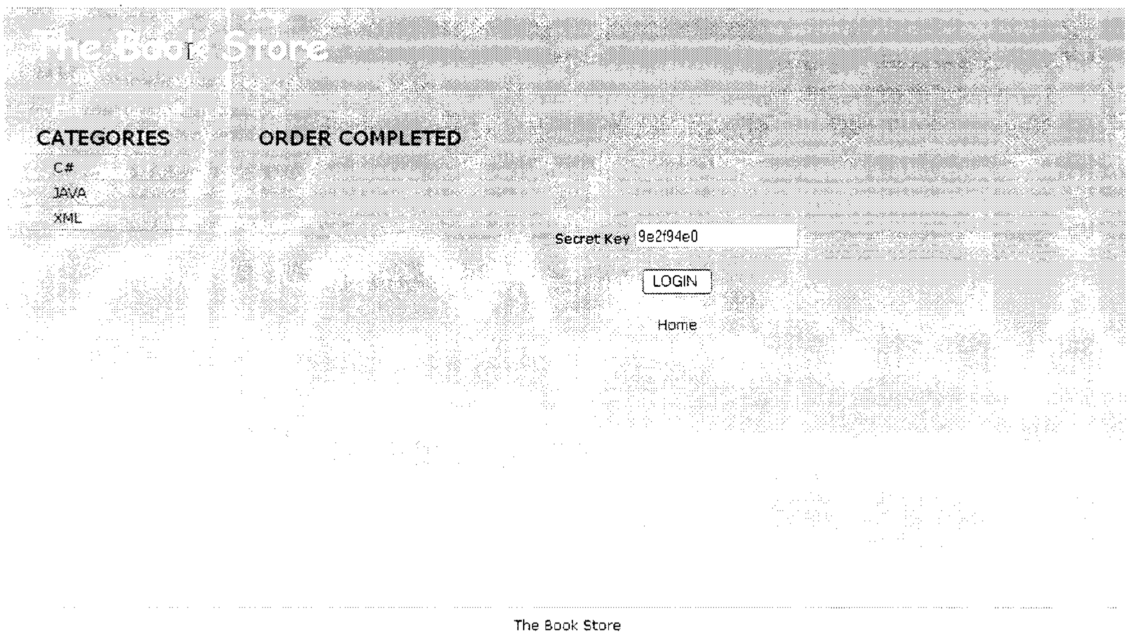
Total Amount : Rs. 700.0

CardType	MasterCard
Credit/Debit Card No	1111111111111111
Pin No	••••
Expiration Date	Month 12 Year 20

The Book Store

Fig.9- Order Checkout

Order Confirm:



The Book Store

CATEGORIES

C#

JAVA

XML

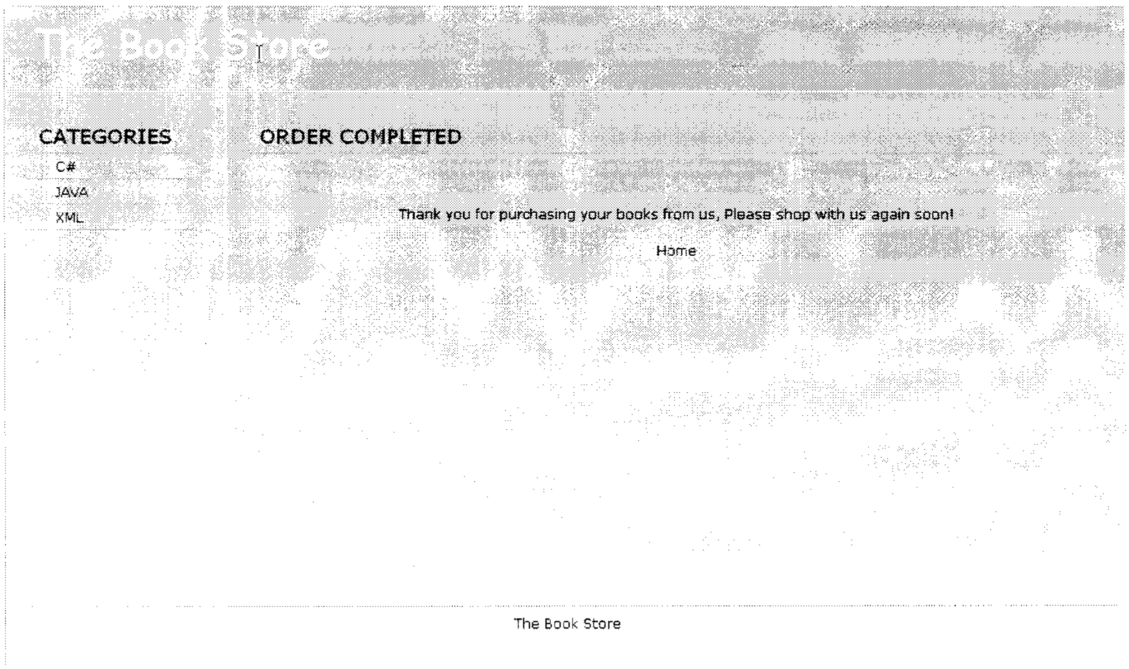
ORDER COMPLETED

Secret Key 9e2f94e0

The Book Store

Fig.10- Order Confirm

Order Complete:



The Book Store

CATEGORIES

C#

JAVA

XML

ORDER COMPLETED

Thank you for purchasing your books from us, Please shop with us again soon!

The Book Store

Fig.11- Order Complete

REFERENCES

Papers Referred:

[1] Jøsang and G. Sanderud, "Security in Mobile Communications: Challenges and Opportunities," in *Proc. of the Australasian information security workshop conference on ACSW frontiers*, 43-48, 2003.

[2] Aladdin Secure SafeWord 2008. Available at *http://www. securecomputing.com/index.cfm?skey=1713*

[3] A. Medrano, "Online Banking Security – Layers of Protection," Available at *http://ezinearticles.com/?Online-Banking-Security---Layers-of-Protection&id=1353184*

[4] B. Schneier, "Two-Factor Authentication: Too Little, Too Late," in *Inside Risks 178*, *Communications of the ACM*, 48(4), April 2005.

[5] D. Ilett, "US Bank Gives Two-Factor Authentication to Millions of Customers," 2005. Available at *http://www.silicon.com/ financialservices/0,3800010322,39153981,00.htm*

[6] D. de Borde, "Two-Factor Authentication," *Siemens Enterprise Communications UK- Security Solutions*, 2008. Available at http://www.insight.co.uk/files/whitepapers/Twofactor%20authentication %20(White%20paper).pdf

[7] A. Herzberg, "Payments and Banking with Mobile Personal Devices," *Communications of the ACM,* 46(5), 53-58, May 2003.

[8] J. Brainard, A. Juels, R. L. Rivest, M. Szydlo and M. Yung, "Fourth-Factor Authentication: Somebody You Know," *ACM CCS*, 168-78.2006.

[9] NBD Online Token. Available at *http://www.nbd.com/NBD/NBD_CDA/CDA_Web_pages/Internet_Bankin g/nbdonline_topbanner*

[10] N. Mallat, M. Rossi, and V. Tuunainen, "Mobile Banking Services," *Communications of the ACM,* 47(8), 42-46, May 2004.