MOBILE E-COMMERCE BUSINESS APPLICATION USING J2ME

P.3120

A PROJECT REPORT

Submitted by

VIGNESH R.S **71206104056**

VIGHNESH S.L.R **71206104057**

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

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"MOBILE E-COMMERCE BUSINESS APPLICATION USING J2ME"** is the bonafide work of **"VIGNESH R.S"** and **"VIGHNESH S.L.R"** who carried out the project under my supervision.

SIGNATURE

Dr. S.Thangasamy

DEAN

Department of Computer Science

Kumaraguru College of Technology,

Chinnavedampatti Post,

Coimbatore – 641606

SIGNATURE

Mr. M.Nageswara Guptha

SUPERVISOR

Department of Computer Science

Kumaraguru College of Technology,

Chinnavedampatti Post,

Coimbatore – 641606

The candidates with University Register Nos. **71206104056** and **71206104057** were examined by us in the project viva-voce examination held on

16/4/2010

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION

We

VIGNESH R.S	**71206104056**
VIGHNESH S.L.R	**71206104057**

hereby declare that the projects entitled "**MOBILE E-COMMERCE BUSINESS APPLICATION USING J2ME**", 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 other Institution, for fulfillment of the requirement of the course study.

The report is submitted in partial fulfillment of the requirement for the award of the Degree of Bachelor of Computer Science and Engineering of Anna University, Chennai.

Place: Coimbatore

Date: 15-April-2010.

(Vignesh R.S)

(Vighnesh S.L.R)

ACKNOWLEDGEMENT

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 **Mr. M. Nageswara Guptha,** 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

Wireless communication is the transfer of information over a distance without the use of enhanced electrical conductors or wires. The distances involved may be short or long. The project here deals with Providing a way to access Online Applications by Wireless Communications thereby eliminating wired networks.

Wireless communication eliminates the limitations of location and time. Mobile devices, such as, Personal Digital Assistants (PDAs) and cell phones, plus the wireless networks have provided a solid foundation for the ubiquitous computing. Today, the number of people accessing the Web through wireless devices has been larger than the number of people who access the Web through desktop computers.

Online applications based on the HTTP protocol are shifting from wired networks, known as Web applications and Web services, to wireless networks, known as wireless mobile online applications, due to the rapid growth of mobile devices, such as Personal Digital Assistants and cell phones. Among the enabling technologies, J2ME is the dominant and the most potential one for building up these wireless mobile online applications. This paper presents a framework that makes the modeling, implementation, and maintenance of wireless mobile online applications intuitive and easy, especially for students and beginners.

LIST OF FIGURES PAGENO

INTRODUCTION

CHAPTER 1: INTRODUCTION

1.1 WIRELESS COMMUNICATION

Wireless communication is the transfer of information over a distance without the use of enhanced electrical conductors or wires. The distances involved may be short (a few meters as in television remote control) or long (thousands or millions of kilometers for radio communications).

It encompasses various types of fixed, mobile, and portable two-way radios, cellular telephones, personal digital assistants (PDAs), and wireless networking. Other examples of wireless technology include GPS units, garage door openers and or garage doors, wireless computer mice, keyboards and headsets, satellite television and cordless telephones.

Wireless communication eliminates the limitations of location and time. Mobile devices, such as, Personal Digital Assistants(PDAs) and cell phones, plus the wireless networks have provided a solid foundation for the ubiquitous computing. Today, the number of people accessing the Web through wireless devices has been larger than the number of people who access the Web through desktop computers.

Cell phones and wireless communication have been part of our daily life. Definitely, they are not only available for voice talks but also are the most suitable platform for accessing online services such as browsing Internet, doing business through mobile e-business (m-business), accessing scientific data, and so on. Wireless mobile online applications are similar with wired Web applications in

terms of three-tier architecture, server side programming, and system-level supports.

The difference between them mainly lays on the client side programming. Due to the limited resources of mobile devices, especially cell phones, some mobile devices do not have Web browser installed. Therefore, the communication between the client and server takes the form of end-to end, that is, a specific client program should be implemented and deployed to an individual cell phone. There are different enabling technologies, such as i-mode, Wireless Application Protocol (WAP),and J2ME (Java 2 Micro Edition) . Among them, the J2ME platform is getting popular due to the fact that the majority of mobile devices support J2ME and the majority of IT people (professionals and students) have been trained with Java programming.

Any software development goes through modeling, implementation, and maintenance processes. This paper proposes a framework for making the modeling, implementation, and maintenance of mobile online applications i.e. E-commerce or shopping through mobile based on J2ME intuitive and easy.

1.2 J2ME

J2ME uses smaller virtual machines and leaner APIs. The J2ME supports programming in mobile devices, such as PDAs and cell phones. The architecture of J2ME consists of two parts: configuration and profile. Due to the fact that there are many different types of hardware in mobile devices, the Connected Device Configuration (CDC) and Connected, Limited Device Configuration (CLDC) can only provide common part of APIs and leave device specific needs (features) to so-called Mobile Information Device Profile (MIDP). In other words, the

configuration defines the capability of J2ME while the profile adds additional functionalities for handling the new breed of mobile devices.

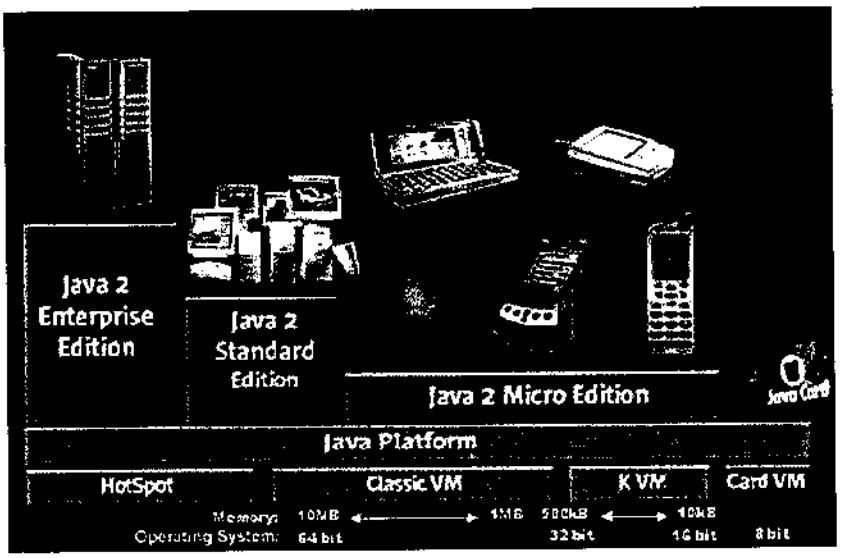


FIG1: J2ME a part of the Java 2 Platform

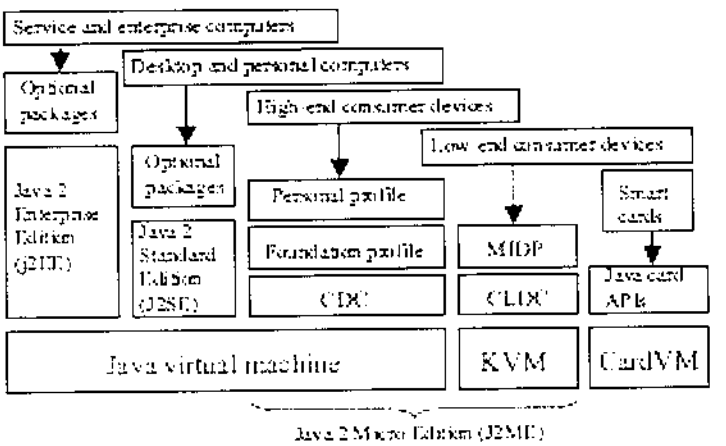


FIG 2 THREE EDITIONS OF JAVA LANGUAGE

The APIs provided by J2ME forms a class hierarchy as shown in Figure3. They are categorized as high-level and low level User Interfaces. The high-level APIs takes the Screen class as its root, which has four sub-classes, namely List, TextBox, Form, and Alert. Among them, the subclass Form is a container that may contain any objects instantiated from all subclass of the Item class, such as DateField, TextField, ImageItem, etc. These high-level APIs support a nice screen display with multiple functionalities. However, they are not flexible enough for creating more customized look-and-feel. The low-level APIs comes in handy. The major class inlow-level APIs is the Canvas, which shares the same superclass Displayable with the Screen class.

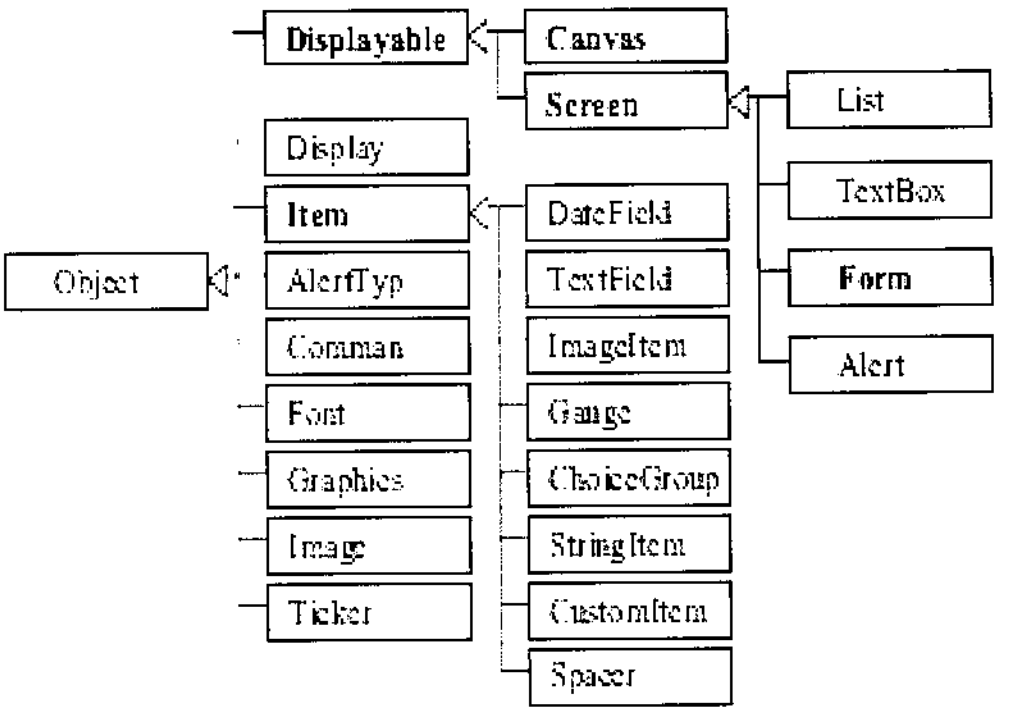


FIG3 :CLASS HIERARCHY OF USER INTERFACES

A wireless mobile online application is based on the three-tier architecture as shown in Figure3. The HTTP is the communication protocol to bridge the client and the server. On the server side, the servlets accept the user's requests, dispatch the requests to other components for

validation, manipulation, confirmation, and finally generate the corresponding responses.The software components on the client side are based on the class MIDlet that is the superclass for supporting Application Manager of J2ME programs. The MIDlet implements the codes that will be deployed onto cell phones.The key components of J2ME is

- Configurations
- Profiles

Configurations

- The configurations define the basic J2ME environment.
- The key point is each connection is geared up to specific family of devices with similar capabilities.

The two Configurations of J2ME is

- CLDC (Connected Limited Device Configuration)
- CDC (Connected Device Configuration)

CLDC:

This is limited for small devices like Mobiles, PDA, and Pagers.

As a group, these devices have important power, memory, and network bandwidth restrictions that directly affect the kind of Java applications that they can support.

- specifies the Java environment for mobile phone, pager and wireless devices
- CLDC devices are usually wireless

- typically has limited power or battery operated
- network connectivity, often wireless, intermittent, low-bandwidth (9600bps or less)

CDC:

The CDC is targeted to the devices that are less restricted like Set Top Boxes (devices that are designed for n/w based computing process), Car Navigation System. The CDC is the superset of CLDC it includes all the classes of CLDC. And also the CDC includes many more classes of core J2SE and this makes the java programmers more comfortable to use

J2ME Connected Device Configuration (CDC)

- Describes the Java environment for digital television set-top boxes, high end wireless devices and automotive telematics systems.
- Device is powered by a 32-bit processor
- 2MB or more of total memory available for Java
- Network connectivity, often wireless, intermittent, low-bandwidth

These two configurations differ only in their respective memory and display capabilities.

Profiles

A profile extends Configuration it adds some domain specific classes to core set of classes. Profiles are the double edged sword of J2ME While they provide important and necessary functionality, not every device will support every profile.

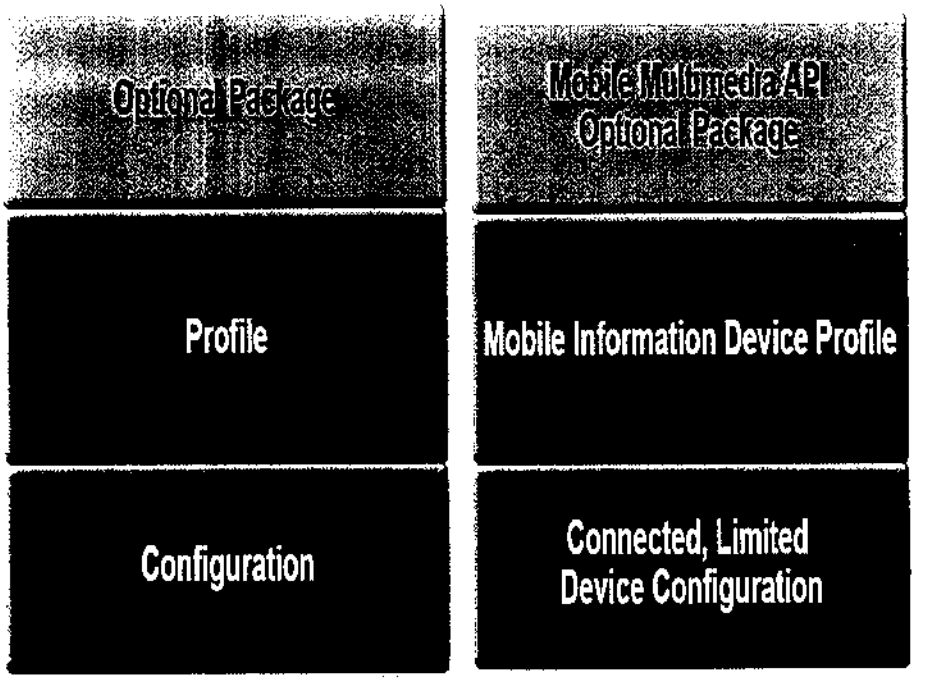


Fig4 :ELEMENTS OF J2ME ARCHITECTURE

J2ME can be divided into three parts, as shown in Figure4 a configuration, a profile, and optional packages. A configuration contains the JVM (not the traditional JVM, but the cut-down version) and some class libraries.

A profile builds on top of these base class libraries by providing a useful set of APIs; and optional packages, are well, an optional set of APIs that you may or may not use when creating your applications.

Optional packages are traditionally not packaged by the device manufacturers, and you have to package and distribute them with your application. The configuration and profile is supplied by the device manufacturers and they embedded them in the devices.

The most popular profile and configuration that Sun provides are the Mobile Information Device Profile (MIDP) and Connected Limited Device Configuration (CLDC)

MIDlets

Every program in j2me is called MIDlet and execution starts from MIDlet.Its like an Applet.All applications from the Mid profile must be derived from a special class MIDlet.The MIdlet class manages the life cycle of the application.It is located in the package javax.microedition.midlet.MIDlet has three different methods

- startApp():The application is started by the startApp().

Now the MIDlet will be in the active stage until we call the pauseApp() or destroyApp().

- pauseApp(): This will stop the application and release the resources that are needed for the program.
- destroyApp():This takes the Boolean as parameter.If it is false then the MIDlet is allowed to refuse its termination by throwing a MIdletStateChangeException.

ADVANTAGES OF J2ME

- Reduced number support
  - float, double primitives
  - Double, Float classes
- Extendable UI components
  - Final High-level components
  - New components subclass Canvas
- Lacks:
  - JNI
  - Sound API
  - Serialization

The J2ME APPLICATION CYCLE

Contrary to the web browser model, which requires continuous connectivity and offers a limited user interface and security experiences, J2ME allows applications to be dynamically downloaded to a mobile device in a secure fashion. J2ME applications can be posted on a Web server, allowing end users to initiate the download of an application they select through a micro browser or other application locator interface. Wireless operators, content providers, and ISVs can also push a set of J2ME applications and manage them remotely. The Java provisioning model puts the responsibility of checking the compatibility of the applications (such as version of the J2ME specification used, memory available on the handset) on the handset itself, allowing the end user to ignore the intricacies associated with typical desktop systems.

Once a J2ME application is deployed on a mobile device, it stays there until the user decides to upgrade or remove it. The application can be operated in disconnected mode (such as standalone game, data entry application) and store data locally, providing a level of convenience that is not available on current browser-based solutions. Because the application resides locally, the user doesn't experience any latency issues, and the application can offer a user interface (drop-down menus, check boxes, animated icons) that is only matched by native C applications. The level of convenience is increased because the user can control when the application initiates a data exchange over the wireless network. This allows for big cost savings on circuit switched networks, where wireless users are billed per minute, and allows a more efficient exchange of data, since many applications can use a store and forward mechanism to minimize network latency.

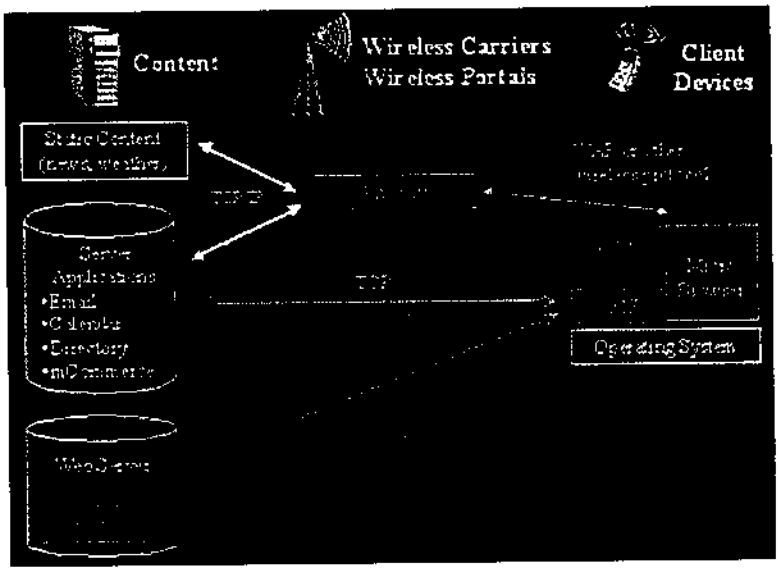


FIG5 :Cycle Of J2ME

Additionally, J2ME applications can leverage any wireless network infrastructure, taking advantage of a WAP network stack on current circuit-switched networks (GSM, CDMA, and TDMA). The same applications are ready to be used on packet-based networks, allowing the use of standard Internet protocols, such as HTTPS over SSL (data encryption), IMAP (email), LDAP (directories), between the J2ME enabled client application and the backend infrastructure.

J2ME Benefits on Wireless Devices

When Java technology is added to this environment, it brings additional benefits that translate into an enhanced user experience. Instead of plain text applications and latency associated to a browser-based interface, the user is presented with rich animated graphics, a fast interaction, the capability to use an application off-line and maybe most interestingly, the capability to dynamically download new applications to the device.

For application developers, this means that you can use your favorite programming language and your favorite development tools, rather than learning a new programming environment. There are over 2.5 million developers who have already developed applications using the Java programming language, primarily on the server side. Once these developers become familiar with the small set of J2ME APIs, it becomes relatively easy to develop small client modules that can exchange data with server applications over the wireless network.

The challenges that remain the same for Java, WAP, or native APIs is that small screens and limited input interfaces require developers to put some effort into the development of the application user interface. In other worlds, small devices force developers to abandon bad or lazy programming techniques.

1.3 WAP

Wireless Access Protocol (WAP) is an open international standard for application-layer network communications in a wireless-communication environment. Most use of WAP involves accessing the mobile web from a mobile phone or from a PDA.

A WAP browser provides all of the basic services of a computer-based web browser but simplified to operate within the restrictions of a mobile phone, such as its smaller view screen

WAP (Wireless Application Protocol) is a specification for a set of communication protocols to standardize the way that wireless devices, such as cellular telephones and radio transceivers, can be used for Internet access, including e-mail, the World Wide Web, newsgroups, and instant messaging. The following diagram shows the standardization of internet access for wireless devices through WAP.

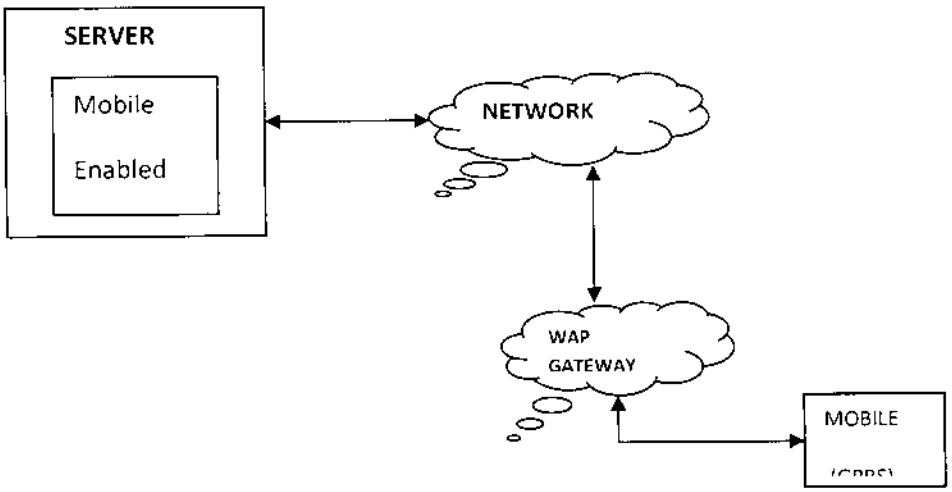


FIG 6 : FLOW DIAGRAM OF WAP GATEWAY

PROBLEM DEFINITION

CHAPTER 2 :PROBLEM DEFINITION

Mobile business refers to executing electronic commerce activities through combining mobile communication equipment, such as the handset, the palm and the beeper, to the Internet. The main feature of mobile business is nimble, simple and convenient. It can be customized according to consumer's specifications, equipment choice as well as the way of providing the service and the information is free for consumers. Through the mobile business, the users may gain the service, the application, the information and the entertainment at anytime and anywhere. Consumers may use intelligence telephone or PDA to search, choose and buy commodity and services when they are convenient. The purchase may complete immediately, and the business decision may also be implemented. The payment consists of many kinds of ways, such as the bank, the user call bill and real-time in the special-purpose prepay account the debit. It meets the different need. Carries on the reliable electronic transaction through individual mobile equipment ability regards as moves the Internet service an important aspect. Because the Web service technology has solved the problems of cross-language and cross-platform which are brought by enterprise integration, so it has received more and introduction of Web service's brings the new generation of ecommerce.

REQUIREMENTS

CHAPTER 3 REQUIREMENTS

Software Requiremnts

- JAVA
- MYSQL SERVER 5.0
- APACHE TOMCAT 5.5
- J2ME WIRELESS TOOLKIT 2.2
- SQLyog

JAVA :

In the Java programming language, all source code is first written in plain text files ending with the .java extension. Those source files are then compiled into .class files by the javac compiler. A .class file does not contain code that is native to your processor; it instead contains bytecodes — the machine language of the Java Virtual Machine[1] (Java VM). The java launcher tool then runs your application with an instance of the Java Virtual Machine.

MYSQL :

MySQL is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases.Many web applications use MySQL as the database component of a LAMP software stack. Its popularity for use with web applications is closely tied to the popularity of PHP, which is often combined with MySQL. Several high-traffic web sites (including Flickr,Facebook,Wikipedia,Google (though not for searches), Nokia and YouTube) use MySQL for data storage and logging of user data.

APACHE TOMCAT:

Apache Tomcat (or Jakarta Tomcat or simply Tomcat) is an open source servlet container developed by the Apache Software Foundation (ASF). Tomcat implements the Java Servlet and the JavaServer Pages (JSP) specifications from Sun Microsystems, and provides a "pure Java" HTTP web server environment for Java code to run.

This is the top-level entry point of the documentation bundle for the Apache Tomcat Servlet/JSP container. Apache Tomcat version 5.5 implements the Servlet 2.4 and JavaServer Pages 2.0 specifications from the Java Community Process, and includes many additional features that make it a useful platform for developing and deploying web applications and web services.

J2ME WIRELESS TOOLKIT:

The Sun Java Wireless Toolkit (formerly known as Java 2 Platform, Micro Edition (J2ME) Wireless Toolkit) is a state-of-the-art toolbox for developing wireless applications that are based on J2ME's Connected Limited Device Configuration (CLDC) and Mobile Information Device Profile (MIDP), and designed to run on cell phones, mainstream personal digital assistants, and other small mobile devices. The toolkit includes the emulation environments, performance optimization and tuning features, documentation, and examples that developers need to bring efficient and successful wireless applications to market quickly

The J2ME Wireless Toolkit is a comprehensive set of tools for building MIDP applications. The toolkit can be used standalone, or incorporated into many popular integrated development environments

SQLyog:

SQLyog is a GUI tool for the RDBMS MySQL. The Prominent features of SQLyog are:

- Visual Schema Designer
- Visual Query Builder
- Smart Autocomplete
- Intelligent Code Completion

Hardware Requirements

- Mobile(with GPRS)
- Pentium III / IV processor
- 512 gb RAM
- 2.80GHZ Operating frequency
- Windows XP or above.

DESIGN

CHAPTER 4 :DESIGN

4.1 MODEL

Based on the MVC (Model-View- Controller) model, we have proposed a graphical languagenamed MVC diagram for modeling, implementing, and maintaining Web applications . The MVC diagram has been successfully applied to model wired Web applications. The MVC diagram can also be applied for modeling wireless mobile online applications. Figure 8 shows a MVC diagram for a J2ME application, named mathTest (Mathematics test). As the figure shows, a MVC diagram uses five symbols. A circle represents a screen display, such as "Welcome", "Login",, and so on. A pair of parallel lines refers to adatabase table, such as "login" table, "questions" table, etc. Arrows with labels exist between circles are known as event arrows, such as the arrow with the label "URL", "Submit", and so on. An arrow points to a circle from a pair of parallel lines indicates a query on the database table that is which points to a pair of parallel lines from a circle implies an update or insert operation on the database table. A small solid circle is the starting or the ending point.

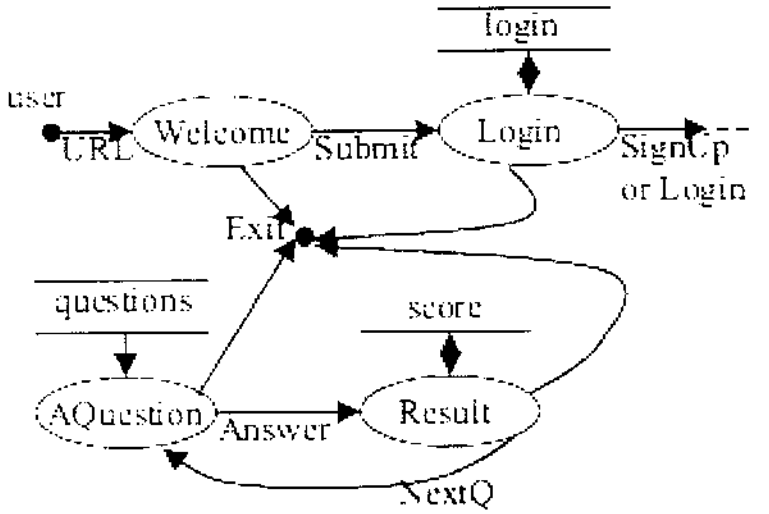


FIG 7: MVC DIAGRAM

The semantics of the diagram can be briefly described as follows. Besides displaying welcome information to the user, the "Welcome" screen also provides a choice as either sign-up for new users or login for registered users. When the user makes a choice and clicks the "Submit" command the choice will be sent along the arrow "Submit" to the "Login" screen. Depending on the user's choice, the "Login" screen either allows a first-time user to sign-up his/her username and password or allows a registered user to login by providing his/her username and password. For the first-time user, the sign-up information will be accepted and sent to the server side. The data arrow points toward the database table "login" above the "Login" screen represents a JavaBeans component that inserts the username and password into the table "login". If the user expressed to login, his/her username and password will be sent to the server side to be verified. That is, the arrow points to the circle "Login" from the database table "login" serves a query that accesses the database table "login" and brings the information from the database to the corresponding JavaBeans component for the verification

of the user's login information. The verification will return a true or a false. The true means the user is authorized to access the application, the application enters the next circle marked as "AQuestion". This screen displays a question with a set of multiple choices. The question comes from the database table "questions". When the user makes a selection and issues the command "Answer", the MIDlet sends the user's selection to the server side. The components in the server validate the answer, calculate the score, update the table "score", and send the related information to be displayed on the screen "Result" for the user. The user can use the "NextQ" (next question) command to continue the test and loop back to the screen "AQuestion". The remaining part of the diagram expresses that the user can exit from the test from any screen by clicking the command "Exit".

The MVC diagram models the entire application. It decomposes the application into the Model (database tables represented by the pairs of parallel lines; JavaBeans that access the database tables represented by the data arrows), the View (screen displays represented by circles), and the Controller (user actions represented by the event arrows with labels). The diagram clearly indicates all required components including all database tables, commands, screens, and their dynamic behaviors, as well as the enabling technologies, such as servlets, JavaBeans, etc. By using the MVC diagram, we can easily model a complicated mobile online application, visualize the interrelations among all components, and count all resources needed for the application. All these together definitely benefit the project management and teamwork a lot.

4.2 MODULES

- Client Side Design Module.
- Server Side Module.
- Server Connecting Module.

4.2.1 CLIENT SIDE DESIGN MODULE

- The client-side system is a MIDlet application which serves as an interface to feed in the contents and control instructions which is interpreted on the server and the appropriate action is taken. The MIDlet has the task of creating textual news contents, creating media contents as well textual news contents. The news creation task is done through a data entry interface which contains various sections to be filled. Once done the data is uploaded to the server and stored in the database server.
- The media news capture is the most important section of the MIDlet application. It has options to capture videos for the devices that support it. These media can then be uploaded to the server and stored in a particular directory structure.

4.2.2 SERVER SIDE DESIGN MODULE

- The server-side system comprises of a web-server, i.e., apache. The scripts are JSP-based while the backend database server is MySQL. The news contents/control instructions sent by the MIDlet client is received at the server end and processed by the respective JSP page.
- The Servlet that handle the MIDlet interaction perform various database based queries.
- The user-interface is a simple webinterface which displays the news contents by fetching them from the server depending on the criteria. The admin-interface is basically for administering the news contents from the desktop

4.2.3 SERVER CONNECTING MODULE

This server connecting module is used for connecting the server to the mobile client using Stream Socket Connection. For the connection the remote desktop IP address is given in the mobile and then connected. Jar file is created using J2ME wireless toolkit & installed into the cellular phone through the USB port

IMPLEMENTATION

SAMPLE CODE

1.CallServletGET.java

```
import javax.microedition.midlet.*;

import javax.microedition.lcdui.*;

import javax.microedition.io.*;

import java.io.*;


public class CallServletGET extends MIDlet implements CommandListener,
Runnable

{

  private Display display;

  private Form fmMain,fmMain1;

  private TextField tfname,tfaddr,tftel,tfmtel,tfuname,tfmail;      // Get account
number

  private TextField tfPwd,tip;

  private StringItem str;

  private Command cmCall;

  private Command cmExit;
```

```
private List menu = null;


String na;


public CallServletGET()

{

  display = Display.getDisplay(this);

  tip = new TextField("IpAddress:", "", 25, TextField.ANY);

  tfname = new TextField("Name:", "", 10, TextField.ANY);

  tfaddr = new TextField("Address:", "", 10, TextField.ANY);

  tftel = new TextField("Telephone:", "", 10, TextField.NUMERIC);

  tfmtel = new TextField("Mobile Telephone:", "", 10, TextField.NUMERIC);

  tfuname = new TextField("UserName:", "", 10, TextField.ANY);

  tfPwd = new TextField("Password:", "", 10, TextField.ANY |
TextField.PASSWORD);

  tfmail = new TextField("Email:", "", 25, TextField.ANY);


  str=new StringItem("","New User Created Successfully");
```

```
cmCall = new Command("Ok", Command.SCREEN, 2);

cmExit = new Command("Exit", Command.EXIT, 1);

//cmExit1 = new Command("Exit", Command.EXIT, 1);



fmMain = new Form("Data from servlet");

fmMain1 = new Form("Data from servlet");

fmMain.append(tip);

fmMain.append(tfname);

fmMain.append(tfaddr);

fmMain.append(tftel);

fmMain.append(tfmtel);

fmMain.append(tfuname);

fmMain.append(tfPwd);

fmMain.append(tfmail);

fmMain.addCommand(cmExit);

fmMain.addCommand(cmCall);
```

```
 //fmMain1.append(str);
 //fmMain1.addCommand(cmExit1);
 fmMain.setCommandListener(this);
}

public void startApp()
{
 display.setCurrent(fmMain);
}

public void pauseApp()
{ }

public void destroyApp(boolean unconditional)
{ }

private void callServlet() throws IOException
{
```

```
new Thread(this).start();

}

public void run() {



HttpConnection http = null;

InputStream iStrm = null;



String url =
"http://"+tip.getString()+"/mobileshopping/LoginMobile"+"?"+"username="+tfuna
me.getString()+"&"+"password=" +
tfPwd.getString()+"&"+"name="+tfname.getString()+"&"+"addr=" +
tfaddr.getString()+"&"+"tel=" + tftel.getString()+"&"+"mtel=" +
tfmtel.getString()+"&"+"mail=" + tfmail.getString();

try

{

 http = (HttpConnection) Connector.open(url);

  System.out.println("http"+url);
```

```
http.setRequestMethod(HttpConnection.GET);
System.out.println("method"+url);



if (http.getResponseCode() == HttpConnection.HTTP_OK)
{
System.out.println("if"+url);
 iStrm = http.openInputStream();


 int length = (int) http.getLength();
 System.out.println(length);
 if (length >= 0)
 {
  byte servletData[] = new byte[length];
  iStrm.read(servletData);
  na = new String(servletData);
  fmMain.append("New User Created Succcessfully");
 }
```

```
    else
      fmMain.append("Unable to read data");
   } else {
     System.out.println("not OK");
   }
}
catch (Exception e)
{
  fmMain.append("Network error");
  System.out.println(e.toString());
}
finally
{
  // Clean up
  if (iStrm != null) {
  try
  {
    iStrm.close();
  }
```

```
catch (Exception e)
{
 System.out.println(e.toString());
}
  }
if (http != null)  {
  try
{
 http.close();
}
catch (Exception e)
{
 System.out.println(e.toString());
}

  }

}
}
```

```
public void commandAction(Command c, Displayable s)
{
 if (c == cmCall)
 {
  try
  {
   callServlet();


  }
  catch (Exception e)
  {
   System.out.println(e.toString());
  }
 }
 else if (c == cmExit)
 {
  destroyApp(false);
```

```
    notifyDestroyed();
  }

 }
}
```

Loginmobile.java (server side)

```
import java.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;
public class LoginMobile extends HttpServlet
  {
   Connection dbcon;

   public void init(ServletConfig sc )throws ServletException{super.init(sc);}

   public void doGet(HttpServletRequest req,HttpServletResponse res)
```

```
throws ServletException,IOException
{
doPost(req,res);
}
public void doPost(HttpServletRequest req,HttpServletResponse res)throws ServletException,IOException
{
System.out.println("servlet called");

try
{

Class.forName("com.mysql.jdbc.Driver");

dbcon=DriverManager.getConnection("jdbc:mysql://localhost:3306/mobileshopping","root","password");
System.out.println("Connection established");
}
catch(Exception e)
{
```

```
            System.out.println("Database driver not found");
            System.out.println(e.toString());
        }
res.setContentType("text/plain");
PrintWriter out=res.getWriter();

String name=req.getParameter("name");
String addr=req.getParameter("addr");
String tel=req.getParameter("tel");
String mtel=req.getParameter("mtel");
String uname=req.getParameter("username");
String password=req.getParameter("password");
String mail=req.getParameter("mail");

        System.out.println(" Attribute  value " + uname);
        System.out.println(" Password  value " + password);
        System.out.println(" NAme  value " + name);
        System.out.println(" Address  value " + addr);
```

```
try
{
PreparedStatement st=dbcon.prepareStatement("select * from customer where username='"+uname+"' and password='"+password+"'");

ResultSet r=st.executeQuery();

if(r.next())
{
int id=r.getInt(1);

out.println(" Already registered value " + id);
out.print(name);

}
else
{
```

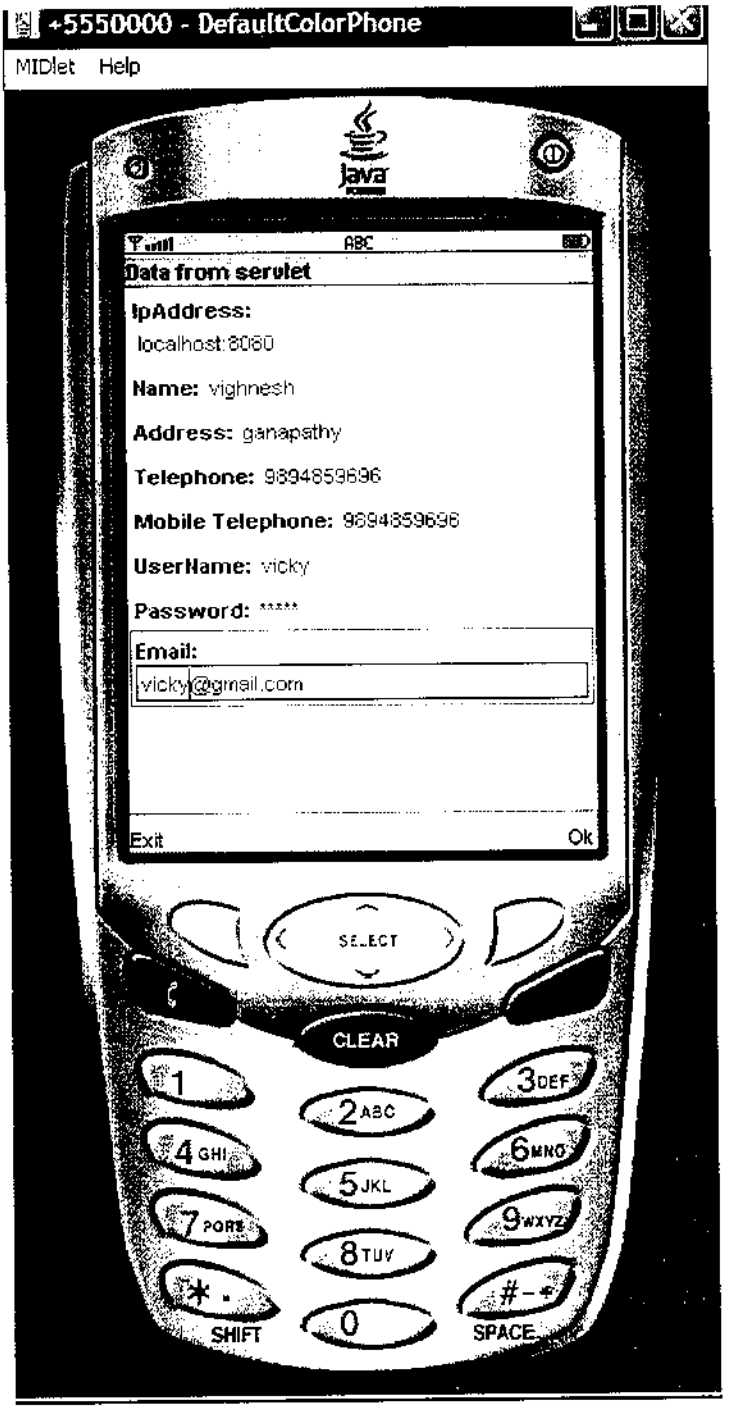
```
PreparedStatement st1=dbcon.prepareStatement("insert into
customer(name,address,tel,mobiletel,username,password,email)
values('"+name+"','"+addr+"','"+tel+"','"+mtel+"','"+uname+"','"+password+"','"+m
ail+"')");
        int i=st1.executeUpdate();
        System.out.println(i);
        if(i>0)
         {
          System.out.println("inserted");
         }
         else
         {
          out.println("While inserted error");
    }
        }       }
        catch(Exception e)
        {
        System.out.println("something wrong");
        System.out.println(e.toString());
        }}}
```

SNAPSHOTS

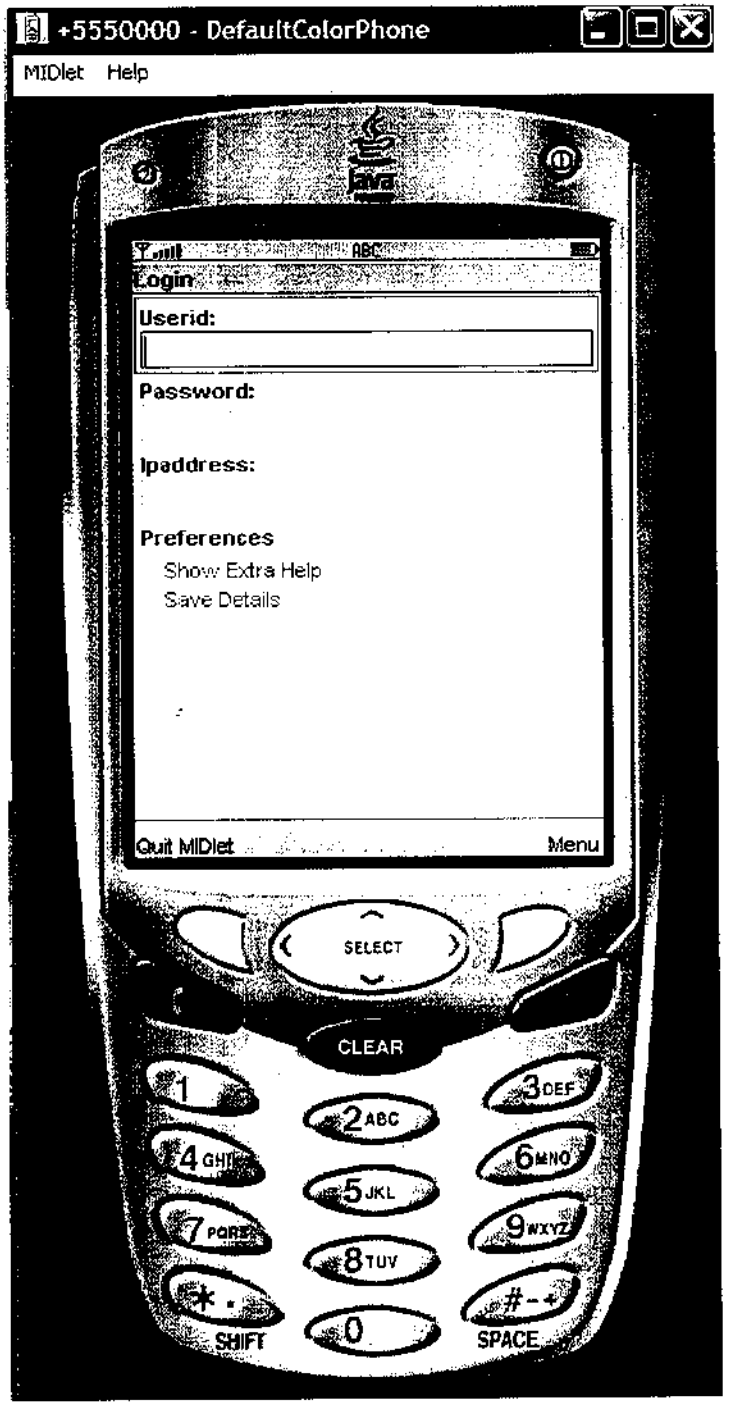
CHAPTER 6 :SNAPSHOTS

Snapshot 1 : Application launch screen

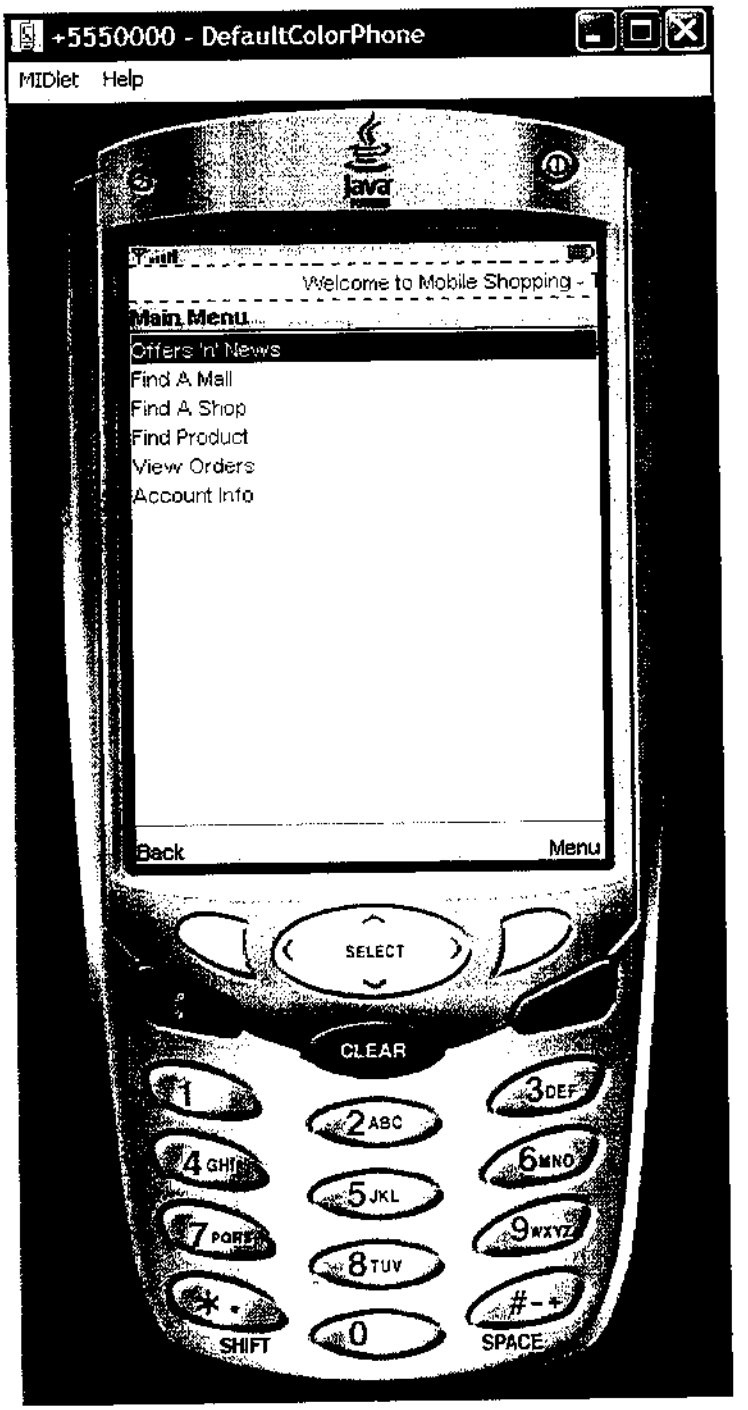
Snapshot 2 : New user registration



Snapshot 3 : User login Form

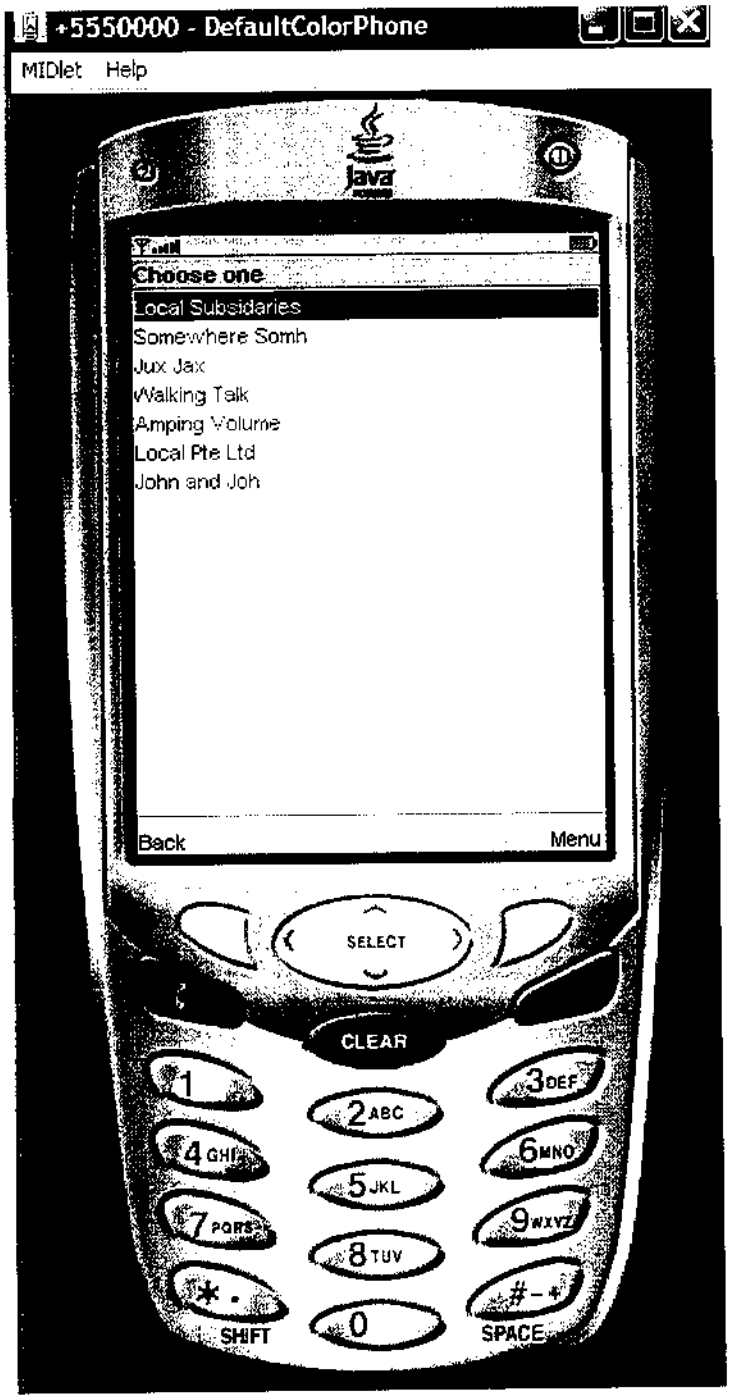


Snapshot 4 : Main menu

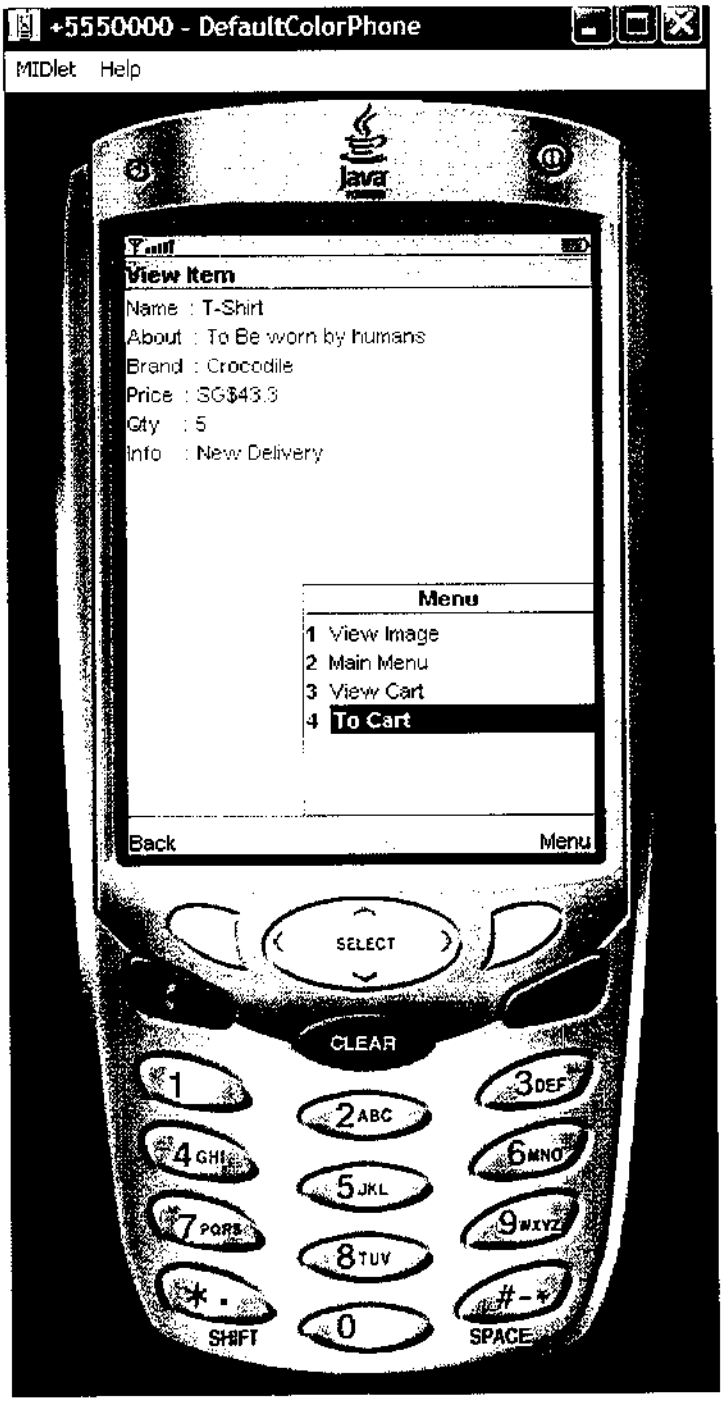


Snapshot 5 : Searching for shopping mall

Snapshot 6 : Selecting a shop



Snapshot 7 : Adding item to cart



Snapshot 8 : Viewing cart

CONCLUSION

CHAPTER 7 :CONCLUSION

The MVC diagram models a wireless mobile online application. They together provide a graphical modeling language for the specification of an application, guide the entire modeling, implementation, and maintenance processes, and support reusable components. This framework decomposes a complex online application into modules. Each module is a plug-and-play unit. The components in the libraries (the method library and the JavaBeans library) can be directly called and used. The MVC diagram provides the software with a strong adaptability and a high degree of flexibility. All these make a special sense to students and beginners.

FUTURE SCOPE

We will further apply the framework to more practices, improve its structure, and enrich the libraries. Several more complicated applications, for example, an online bookstore, an online ticket office, an online IT terminology, an online conference submission systems are being developed for the goal.

REFERENCE

REFERENCE

Peter Coad, Mark Mayfield, an Jon Kern, "Java Design – Building Better Apps and Applets", Prentice-Hall, E1999.

Deitel and Deitel, "Wireless Internet and Mobile Business—How to Program", Prentice-Hall, 2002.

T. Husted, C. Dumoulin, G. Franciscus, and D. Winterfeldt, "Struts in Action", Manning, 2003.

C.A. Horstmann and G. Cornell, "Core Java 2 Volume I – Fundamentals", 7/e, Sun Microsystems Press, 2005.

Robert Martin, "Design Principles and Design Patterns", www.objectmentor.com, 2000.

Bertrand Meyer, "Object Oriented software Construction", Prentice-Hall, 1988.

Martine J. Wells, "J2ME Game Programming", Thomson Course Technology, 2004.