DEPARTMENT OF INFORMATION TECHNOLOGY

Intelligent Information System
for Intelligence's department

A

Dissertation Work

Submitted as Major Project in Partial fulfillment for the award of
Bachelor of Engineering in Information Technology.

Submitted By--

Himanshi Yadav (0101AU131018)

Himanshu Ajmera (0101AU131019)

Neeraj Nigam (0101AU131032)

Tarun Saxena (0101IT131056)

Under the Guidance of

Dr. Mahesh Pawar

UNIVERSITY INSTITUTE OF TECHNOLGY

RAJIV GANDHI PROUDYOGIKI VISHWAVIDYALAYA
BHOPAL (M.P)
2017

DEPARTMENT OF INFORMATION TECHNOLGY
UNIVERSITY INSTITUTE OF TECHNOLGY
RAJIV GANDHI PROUDYOGIKI VISHWAVIDYALAYA
BHOPAL (M.P)

CERTIFICATE

This is to certify that the project entitled **"Intelligent Information System for Intelligence's Departments"** being submitted by **Himanshi Yadav, Himanshu Ajmera, Neeraj Nigam, Tarun Saxena** students of 8th Semester, Department of Information Technology have done their work as MAJOR PROJECT for Partial fulfillment of Bachelor of Engineering in Information Technology from RGPV, Bhopal (M.P.) is a record of bonafide work carried out by his/her under my supervision.

Guided by:

Dr. Mahesh Pawar
Assistant Professor
Dept. Of Information Technology
UIT, RGPV

Forwarded by:

Dr. Roopam Gupta
(H.O.D)
Dept. of **Information** Technology
UIT, RGPV

DEPARTMENT OF INFORMATION TECHNOLGY
UNIVERSITY INSTITUTE OF TECHNOLGY
RAJIV GANDHI PROUDYOGIKI VISHWAVIDYALAYA
BHOPAL (M.P)

DECLARATION

We declare that project is entitled **"Intelligent Information System for Intelligence's Departments"** is our own work conducted under the supervision of "Dr. Mahesh Pawar" Department of Information Technology, University Institute of Technology, RGPV Bhopal (M.P.).

I further declare that, to the best of my knowledge the project does not contain the work which have been submitted for the award of the degree either in the University or in any other University/Deemed University without proper citations.

**Himanshi Yadav,
Himanshu Ajmera,
Neeraj Nigam,
Tarun Saxena
B.E. 8th Semester (I.T.)
UIT-RGPV, BHOPAL (M.P.)**

ACKNOWLEDGEMENT

We take the opportunity to express my cordial gratitude and deep sense of indebtedness to our guide Dr. Mahesh Pawar for the valuable guidance and inspiration throughout the project duration. We feel Thankful to him for his innovative ideas, which led to successful completion of this project work. We feel proud and fortunate to work under such an outstanding mentor in the field of "Intelligent Information System for Intelligence's Departments He has always welcomed our problem and helped us to clear our doubt. We will always be grateful to him for providing us moral support and sufficient time.

We owe sincere thanks to **Dr. Roopam Gupta,** HOD, I.T. who helped us duly in time during my project work in the Department.

At the same time, we would like to thank all other faculty members and all non-teaching staff in Information Technology Department for their valuable co-operation.

Himanshi Yadav,
Himanshu Ajmera,
Neeraj Nigam,
Tarun Saxena
B.E. 8th Semester (I.T.)
UIT-RGPV, BHOPAL (M.P.)

ABSTRACT

Since crime is making the situation worse in our country day by day and there isn't enough data about all the criminals at all the police stations to put a hold on it, we have designed a new system which would help in connecting all the police stations safely and which would enable them to share the confidential data about criminals in a secure way.

This system is more effective than the others in league as it takes only one detail of the accused and brings all the other available ones to the node where the query has been made.

Each police station will act as a node having its own database and all of them will be connected with each other through secure protocols like tcp, and using security measures like AES, blowfish, two fish algorithms.

Table of Contents

Note: Figures should be clear and self-made. Attach CD of Code with Essential Software Required to run the project program with each copy of project report.

Chapter-1

INTRODUCTION

In our project, we have work towards enhancing the technology used at intelligence bureaus in India by developing an information system. Developed Information system has distributed property i.e.; it does not work on centrally based server rather each node act as server and will have its own database. The uniqueness of this Information system is that being independent a node can modify its database without central dependency. For communication and data sharing between every node encryption algorithms are used. This will ensure security between network and will prohibit unauthenticated access to the data.

To develop such system, we have used SQ-lite which is database engine, Socket programming which is used as mechanism to establish communication between two computers using TCP, Tkinter which is python's library used for GUI designing and AES, blow-fish, two-fish which are encryption algorithms are also used in this project keeping in mind the minimization of vulnerability of data. These encryption algorithms provide peer-to-peer encryption.

1.1 PROJECT PERSPECTIVE

The application of such information system is at police headquarters around states, in defence officials to keep data updated about terrorists, in Intelligence departments of country like CID, CBI, RAW, DIA etc.
this project can also be implemented on all over the country's offices to keep record of employees and their activities. Further development can also be done in the direction of security improvement. Since currently system is more vulnerable and so is its data and can easily be manipulated or used by unauthorized user.

1.2 SCOPE OF WORK

Our main aim is to build a distributed database of the criminal records at each state level so that retrieval of information becomes easy for the intelligence and police department and hence make easy to find any criminal's record from any place of the country.
We propose an intelligent information retrieval (IIR) approach that is designed to integrate two disparate methods, namely information retrieving with distributed information record. Simple IR

could be time consuming and may not be achievable without manual interventions for data sets that involve different media such as video, audio, images and documents.

Our Intelligent IR takes into account the meaning of the words used in the query, their relationships such as the order of words in the query, and thereby establishes the relevance. It is also designed to adapt the query based on the user's direct and indirect profile contexts and relevance feedback. Our model of IIR makes use of information utility and relevance. Though utility and relevance are important for all IR operations, measuring them and using them intelligently is important. Utility might be measured in monetary terms: “How much is it worth to the user to have found this document?” “How much did we save by finding this software?” In the literature, the term “relevance” is used imprecisely; it can mean utility or topical relevance or pertinence. Many IR systems focus on finding topically relevant documents, leaving further selection to the user.

Using various tools like cryptography, SEQL, AES and various algorithms and creating data servers at different locations of the country would make it possible and let the police and intelligence department access the database in more efficient and fast way. It will integrate the database of each police station as a single database superficially but it will remain in distributed form.

1.3 ORGANIZATION OF PROJECT REPORT

This section introduces about the report organization and their contents overview in a brief manner. The remaining document is organized as follows:

- The introduction to the thesis is given in Chapter 1. This section describes the general introduction, the need and theory.
- Chapter 2 reviews different existing and emerging technologies that are related to the work presented in this thesis under the title literature survey
- In the next section, Chapter 3 Analysis of the whole research work is done by explaining the detailed work performed in the research work.
- In chapter 4 we give the implementation of the solution and the suggested approach of solving it.
- Chapter 5 describes the various results that are obtained after the complete implementation of the project.
- Chapter 6 provides conclusions from the work described in previous chapters and discusses possibilities for future development.

Chapter-2
Literature Survey

Evaluation studies commonly use recall and precision or a combination. With low precision, the user must look at several irrelevant documents for every relevant document found. More sophisticated measures consider the gain from a relevant document and the expense incurred by having to examine an irrelevant document.

Intelligence System and police department nowadays following traditional file approach to retrieve the information of the criminals and information is distributed and not having in information record with each other. This makes retrieval of the information slow and inefficient which results in the late action and identification from the intelligence and police department.

For example, many relevant documents that merely duplicate the same information just waste the user's time, so retrieving fewer relevant documents would be better. In this paper, we propose an intelligent IR approach considering three main utility attributes, relevant, pertinent and novel, for retrieving a document from a large data base. A document is topically relevant for a particular situation, context, query, or task if it contains information that either directly answers the query or can be used, possibly in combination with other information, to derive an answer or perform the task. It is pertinent with respect to a user with a given purpose if, in addition, it gives just the information needed; is compatible with the user's background and cognitive style so s/he can apply the information gained and is authoritative. It is novel if it adds to the user's knowledge i.e. finding unknown things which is the part of data mining. In this paper, we propose an intelligent IR approach to construct a user model through gaining relevant user feedback, which can significantly arrive at a smaller set of ranked documents that are relevant to the user's interests or search intent.

An Intelligent Image Retrieval Component This component consists of three main steps as described below. Each step includes system verification and validation procedures to maintain the integrity of the data stored in the system. Uploading of images – This step uploads the images of all answer booklets consisting of First part (Student information) and Second part (Exam information). The images of First part have basic details , identification mark, Bar code and other information of Criminal. While images of Second part have question wise and uncommon traits given by Intelligence or in some cases details are also given by any other user; identification code, traits of the criminals, physical appearance of the criminals . Linking of images – Images of First part and Second part are linked using the Cryptography.

Also, they are linked by examination code and seat number so that retrieval can be fast. For example, a query to retrieve the image based on seat number or subject code (examination code) would be much faster with such associations established. Retrieving of images – This step interfaces with the front-end of the system which communicates with the query module to retrieve the images. From these images the information about bundle number and answer booklet number can be used to retrieve the exact location in the storage for faster physical retrieval of the answer booklet. However, not all queries have the necessary inputs, in which case intelligent search and filtering of data is required. Hence, in such cases, it is necessary to interface with the recommender system based on data mining of relevance and user profile to have fast retrieval of required information. This component is useful to many departments involved in facilitating the query process, such as the Physical Storage department, Photocopy Services department, Revaluation department, Student Services department, etc.

Intelligent IR Component The intelligent IR component facilitates the recommender systems to focus on the user context-based recommender paradigm by using keywords and relevance criteria to arrive at the correct answer booklet from the database. This is achieved using the following processes: User interface to manage interaction with the user for query input and document output. Relevance feedback and visualization of results Keywords such as part of student name or subject name with relevance to some known data are intelligently processed using data mining approaches to form index words (tokens). Indexing constructs an inverted index of words resulting in document pointers. Searching retrieves documents that contain a given query token from the inverted index.

Information integration and extraction to arrive at relevance. Relevance is a subjective judgment and includes the context, timeliness, authoritative and satisfaction level of a query. Associate ranking scores to all retrieved documents according to relevance metric. The metric used here is based on data mining of item similarity (such as keywords) and user-item interaction (such as previous queries or user context). Similar approaches are found in literature.

Query operations to transform the query in order to improve retrieval. We adopt query expansion using a thesaurus and query transformation using relevance feedback. While relevance-based probabilistic model for retrieval are adopted in some studies.

Chapter-3
Proposed Solution

System administrators can distribute collections of data (e.g. in a database) across multiple physical locations. A distributed database can reside on organized network servers or decentralized independent computers on the Internet, on corporate intranets or extranets, or on other organization networks. Because Distributed databases store data across multiple computers, distributed databases may improve performance at end-user worksites by allowing transactions to be processed on many machines, instead of being limited to one.

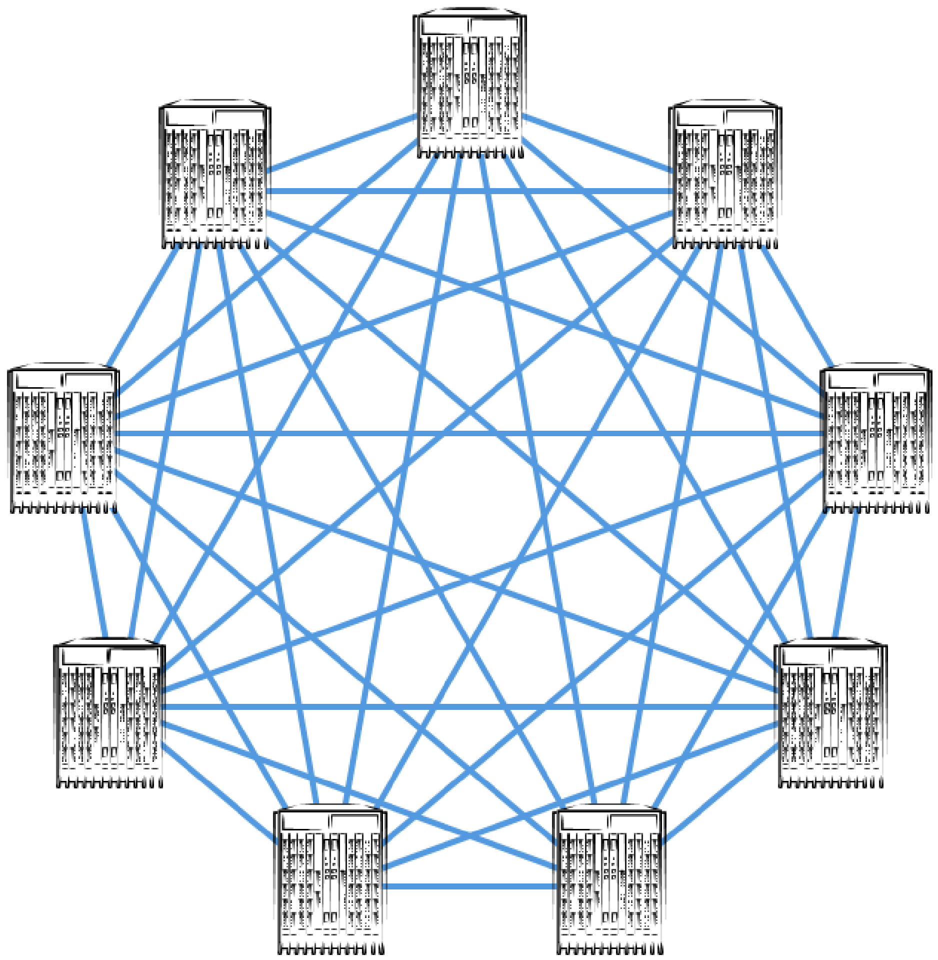
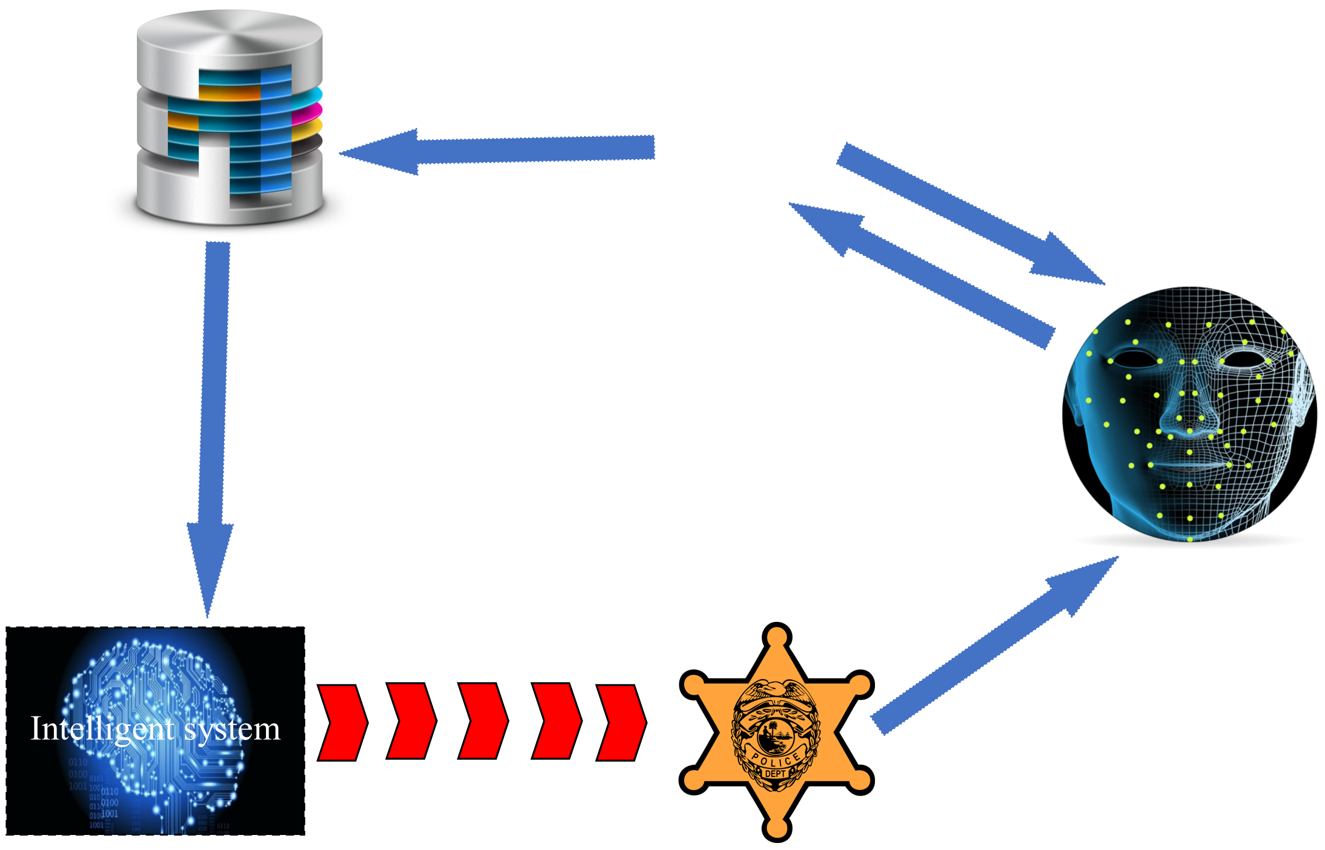


Figure 1.1 Shows a distributed Database network

So, if a criminal goes anywhere in the country, since details about every individual criminal is shared over the distributed database. His/her activity can be traced and he can be easily located by looking to his details.

Some premium feature like if the image processing and rapid notification techniques using IoT is also included the tracking and catching the criminal would become easier and this can help in reducing the crime rates.

Figure 1.2 Shows proper and future implementation



Such solution can be implemented in future if proper technology is used. Since, the evolution is taking place in technology at such a pace that it's not so far to create something like god's eye in future. By mapping exact coordinates of the person's location.

Chapter-4
Implementation

4.1 Tools and Techniques

The various tools used in this project are given below.

4.1.1 NetBeans and concept of Maven

NetBeans is a software development platform written in Java. The NetBeans Platform allows applications to be developed from a set of modular software components called *modules*. Applications based on the NetBeans Platform, including the NetBeans integrated development environment (IDE), can be extended by third party developers.

The NetBeans IDE is primarily intended for development in Java, but also supports other languages, in particular PHP, C/C++ and HTML5.

NetBeans is cross-platform and runs on Microsoft Windows, Mac OS X, Linux, Solaris and other platforms supporting a compatible JVM.

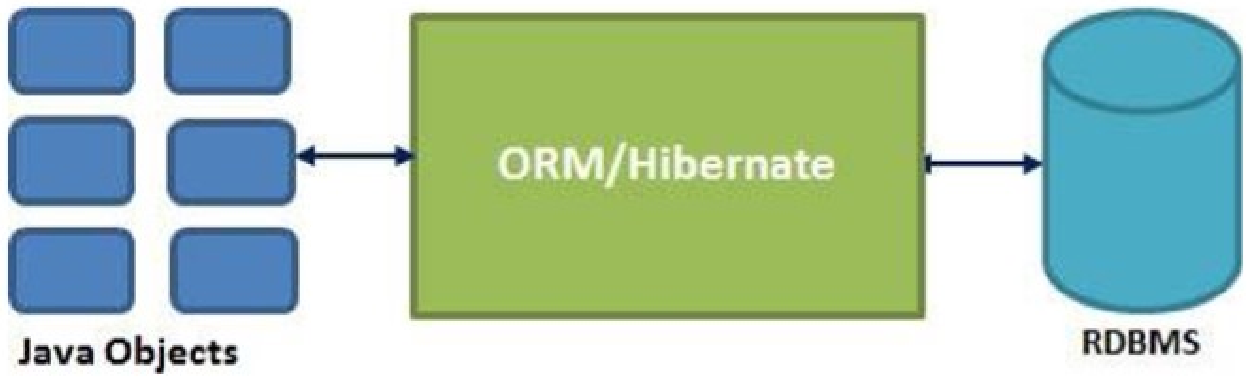
The NetBeans Team actively supports the product and seeks feature suggestions from the wider community. Every release is preceded by a time for Community testing and feedback. The NetBeans Platform is a framework for simplifying the development of Java Swing desktop applications. The NetBeans IDE bundle for Java SE contains what is needed to start developing NetBeans plugins and NetBeans Platform based applications; no additional SDK is required.

Applications can install modules dynamically. Any application can include the Update Center module to allow users of the application to download digitally signed upgrades and new features directly into the running application. Reinstalling an upgrade or a new release does not force users to download the entire application again.

Maven is a build automation tool for Java project management. You can easily open and work with Maven projects in the IDE. In NetBeans IDE 6.7 and newer, Maven support is included in the IDE. The IDE enables you to create Maven projects from archetypes using the New Project wizard. The IDE also includes a Maven Repository browser that enables you to view your local repository and registered external Maven repositories.

4.1.2 Hibernate ORM

Hibernate is an Object-Relational Mapping ORM solution for JAVA and it raised as an open source persistent framework created by Gavin King in 2001. It is a powerful, high performance Object Relational Persistence and Query service for any Java Application. Hibernate maps Java classes to database tables and from Java data types to SQL data types and relieve the developer from 95% of common data persistence related programming tasks. Hibernate sits between traditional Java objects and database server to handle all the work in persisting those objects based on the appropriate O/R mechanisms and patterns.



In addition to its own "native" API, hibernate is also an implementation of the Java Persistence API (JPA) specification. As such, it can be easily used in any environment supporting JPA including Java SE applications, Java EE application servers, Enterprise OSGi containers, etc.
Hibernate supports lazy initialization, numerous fetching strategies and optimistic locking with automatic versioning and time stamping. Hibernate requires no special database tables or fields and generates much of the SQL at system initialization time instead of at runtime.

Hibernate setup:

Hibernate is framework that provides tools for object relational mapping (ORM). Following steps demonstrates the support for the Hibernate framework included in the IDE and how to use wizards to create the necessary Hibernate files. After creating the Java objects and configuring the application to use Hibernate.

Creating database:

Step 1: Start Apache and MySql server from xampp control panel

Step 2: Open address in your browser http://localhost/phpmyadmin

Step 3: Now on left side select “new” in database section

Step 4: name your database we are naming our DB “project”

Step 5: go to permission tab and add new user

username: neeraj

password: “neeraj”

Step 6: Add tables with appropriate details.

Creating Test application / our project setup

In this step you create a simple Java Swing application project called SocketDB.

1. Choose File > New Project (Ctrl-Shift-N). Select Java Application from the Java category and click Next.
2. Type **SocketDB** for the project name and set the project location.
3. Deselect the Use Dedicated Folder option, if selected. For this tutorial there is little reason to copy project libraries to a dedicated folder because you will not need to share libraries with other users.
4. Deselect Create Main Class. Click Finish.

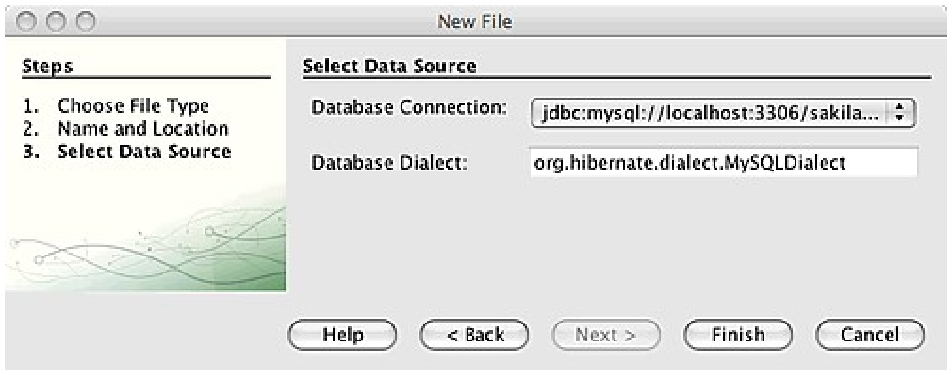
When you click Finish, the IDE creates the Java application project. The project does not have a main class. You will create a form and then set the form as the main class.

To add support for Hibernate to a J2SE project you need to add the Hibernate library to the project. The Hibernate library is included with the IDE and can be added to any project by right-clicking the 'Libraries' node in the Projects window, selecting 'Add Library' and then selecting the Hibernate library in the Add Library dialog box.
The IDE includes wizards to help you create the Hibernate files you may need in your project. You can use the wizards in the IDE to create a Hibernate configuration file and a utility helper class. If you create the Hibernate configuration file using a wizard the IDE automatically adds the Hibernate libraries to the project.

Creating Hibernate Support to project

The Hibernate configuration file (hibernate.cfg.xml) contains information about the database connection, resource mappings, and other connection properties. When you create a Hibernate configuration file using a wizard you specify the database connection by choosing from a list of database connection registered with the IDE. When generating the configuration file the IDE automatically adds the connection details and dialect information based on the selected database connection. The IDE also automatically adds the Hibernate library to the project classpath. After you create the configuration file you can edit the file using the multi-view editor, or edit the XML directly in the XML editor.

1. Right-click the Source Packages node in the Projects window and choose New > Other to open the New File wizard.
2. Select Hibernate Configuration Wizard from the Hibernate category. Click Next.
3. Keep the default settings in the Name and Location pane (you want to create the file in the src directory). Click Next.
4. Select the sakila connection in the Database Connection drop down list. Click Finish.



Database Connection: jdbc:mysql://localhost:3306/sakila...

Database Dialect: org.hibernate.dialect.MySQLDialect

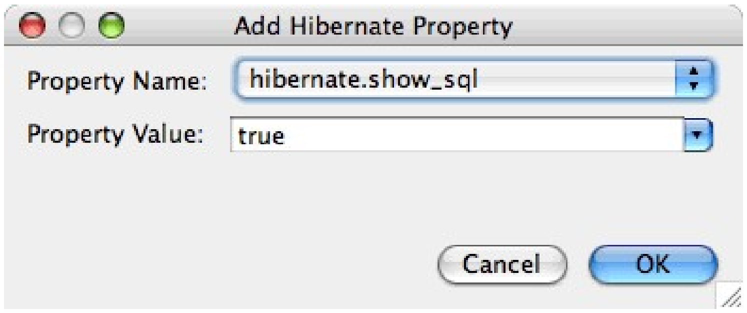
When you click Finish the IDE opens hibernate.cfg.xml in the source editor. The IDE creates the configuration file at the root of the context classpath of the application (in the Files window, WEB-INF/classes). In the Projects window the file is located in the <default package> source package. The configuration file contains information about a single database. If you plan to connect to multiple databases, you can create multiple configuration files in the project, one for each database servers, but by default the helper utility class will use the hibernate.cfg.xml file located in the root location.
If you expand the Libraries node in the Projects window you can see that the IDE added the required Hibernate JAR files and the MySQL connector JAR.

Modifying hibernate configuration file

In this exercise, you will edit the default properties specified in hibernate.cfg.xml to enable debug logging for SQL statements.

1. Open hibernate.cfg.xml in the Design tab. You can open the file by expanding the Configuration Files node in the Projects window and double-clicking hibernate.cfg.xml.
2. Expand the Configuration Properties node under Optional Properties.
3. Click Add to open the Add Hibernate Property dialog box.
4. In the dialog box, select the hibernate.show_sql property and set the value to true. Click OK. This enables the debug logging of the SQL statements.

5.



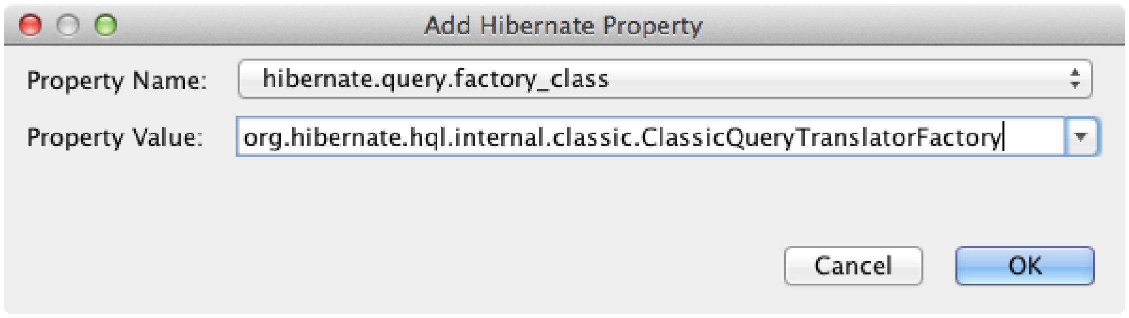
Property Name: hibernate.show_sql
Property Value: true

6. Click Add under the Miscellaneous Properties node and select hibernate.query.factory_class in the Property Name dropdown list.

7. Type **org.hibernate.hql.internal.classic.ClassicQueryTranslatorFactory** as the Property Value.

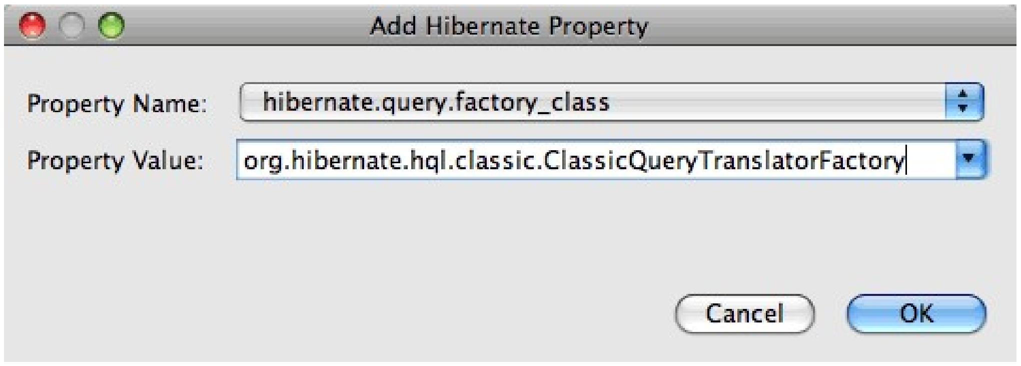
   This is the translator factory class that is used in Hibernate 4 that is bundled with the IDE.

   Click OK.



Property Name: hibernate.query.factory_class
Property Value: org.hibernate.hql.internal.classic.ClassicQueryTranslatorFactory

If you are using NetBeans IDE 7.4 or earlier you should select **org.hibernate.hql.classic.ClassicQueryTranslatorFactory** as the Property Value in the dialog box. NetBeans IDE 7.4 and earlier bundled Hibernate 3.



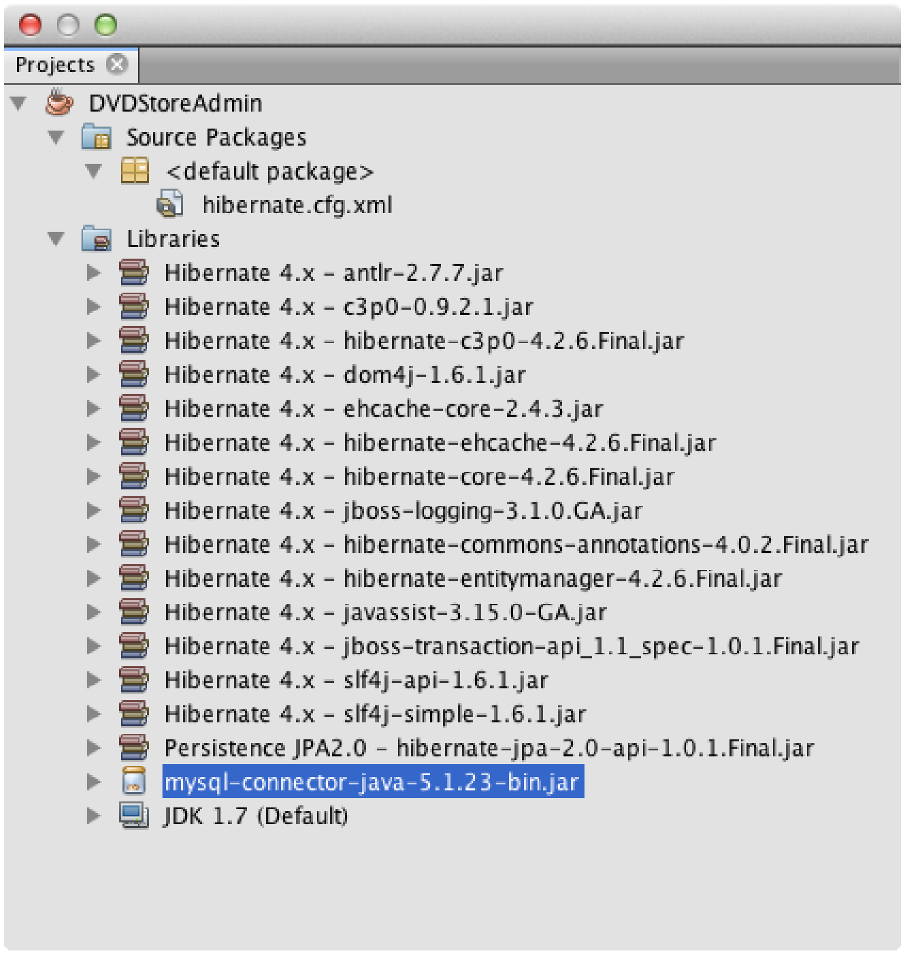
Property Name: hibernate.query.factory_class
Property Value: org.hibernate.hql.classic.ClassicQueryTranslatorFactory

If you click the XML tab in the editor you can see the file in XML view. Your file should look like the following:

```
<hibernate-configuration>
  <session-factory name="session1">
    <property
name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
    <property
name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
    <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/sakila</property>
    <property name="hibernate.connection.username">root</property>
    <property name="hibernate.connection.password">######</property>
    <property name="hibernate.show_sql">true</property>
    <property
name="hibernate.query.factory_class">org.hibernate.hql.internal.classic.ClassicQuer
yTranslatorFactory</property>
  </session-factory>
</hibernate-configuration>
```

8. Save your changes to the file.

After you create the form and set it as the main class you will be able to see the SQL query printed in the IDE's Output window when you run the project.



```
Projects
▼ DVDStoreAdmin
  ▼ Source Packages
    ▼ <default package>
        hibernate.cfg.xml
  ▼ Libraries
    ▶ Hibernate 4.x - antlr-2.7.7.jar
    ▶ Hibernate 4.x - c3p0-0.9.2.1.jar
    ▶ Hibernate 4.x - hibernate-c3p0-4.2.6.Final.jar
    ▶ Hibernate 4.x - dom4j-1.6.1.jar
    ▶ Hibernate 4.x - ehcache-core-2.4.3.jar
    ▶ Hibernate 4.x - hibernate-ehcache-4.2.6.Final.jar
    ▶ Hibernate 4.x - hibernate-core-4.2.6.Final.jar
    ▶ Hibernate 4.x - jboss-logging-3.1.0.GA.jar
    ▶ Hibernate 4.x - hibernate-commons-annotations-4.0.2.Final.jar
    ▶ Hibernate 4.x - hibernate-entitymanager-4.2.6.Final.jar
    ▶ Hibernate 4.x - javassist-3.15.0-GA.jar
    ▶ Hibernate 4.x - jboss-transaction-api_1.1_spec-1.0.1.Final.jar
    ▶ Hibernate 4.x - slf4j-api-1.6.1.jar
    ▶ Hibernate 4.x - slf4j-simple-1.6.1.jar
    ▶ Persistence JPA2.0 - hibernate-jpa-2.0-api-1.0.1.Final.jar
    ▶ mysql-connector-java-5.1.23-bin.jar
    ▶ JDK 1.7 (Default)
```

Note. NetBeans IDE 8.0 bundles the Hibernate 4 libraries. Older versions of the IDE bundled Hibernate 3.

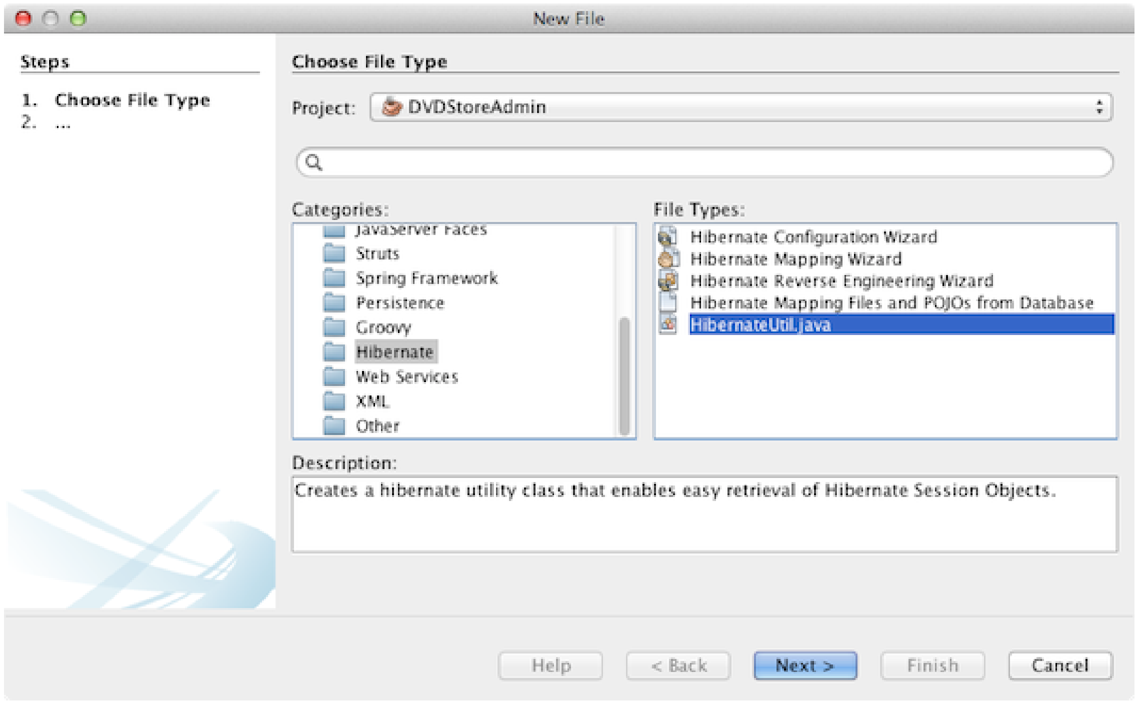
Creating the HibernateUtil.java Helper File

To use Hibernate you need to create a helper class that handles startup and that accesses Hibernate's SessionFactory to obtain a Session object. The class calls Hibernate's configure() method, loads the hibernate.cfg.xml configuration file and then builds the SessionFactory to obtain the Session object.

In this section, you use the New File wizard to create the helper class HibernateUtil.java.

1. Right-click the Source Packages node and select New > Other to open the New File wizard.

2. Select Hibernate from the Categories list and HibernateUtil.java from the File Types list. Click Next.



Creates a hibernate utility class that enables easy retrieval of Hibernate Session Objects.

3. Type **HibernateUtil** for the class name and **sakila.util** as the package name. Click Finish.

When you click Finish, HibernateUtil.java opens in the editor. You can close the file because you do not need to edit the file.

Generating Hibernate Mapping Files and Java Classes

In this tutorial, you use a plain old Java object (POJO), Actor.java, to represent the data in the table ACTOR in the database. The class specifies the fields for the columns in the tables and uses simple setters and getters to retrieve and write the data. To map Actor.java to the ACTOR table you can use a Hibernate mapping file or use annotations in the class.

You can use the Reverse Engineering wizard and the Hibernate Mapping Files and POJOs from a Database wizard to create multiple POJOs and mapping files based on database tables that you select. Alternatively, you can use wizards in the IDE to help you create individual POJOs and mapping files from scratch.

Notes.

- When you want to create files for multiple tables you will most likely want to use the wizards. In this tutorial, you only need to create one POJO and one mapping file so it is fairly easy to create the files individually. You can see the steps for creating the POJOs and mapping files individually at the end of this tutorial.

Creating the Reverse Engineering File

The reverse engineering file (hibernate.reveng.xml) is an XML file that can be used to modify the default settings used when generating Hibernate files from the metadata of the database specified in hibernate.cfg.xml. The wizard generates the file with basic default settings. You can modify the file to explicitly specify the database schema that is used, to filter out tables that should not be used and to specify how JDBC types are mapped to Hibernate types.

1. Right-click the Source Packages node and select New > Other to open the New File wizard.
2. Select Hibernate from the Categories list and Hibernate Reverse Engineering Wizard from the File Types list. Click Next.
3. Type **hibernate.reveng** for the file name.
4. Keep the default **src** as the Location. Click Next.
5. Select **actor** in the Available Tables pane and click Add. Click Finish.

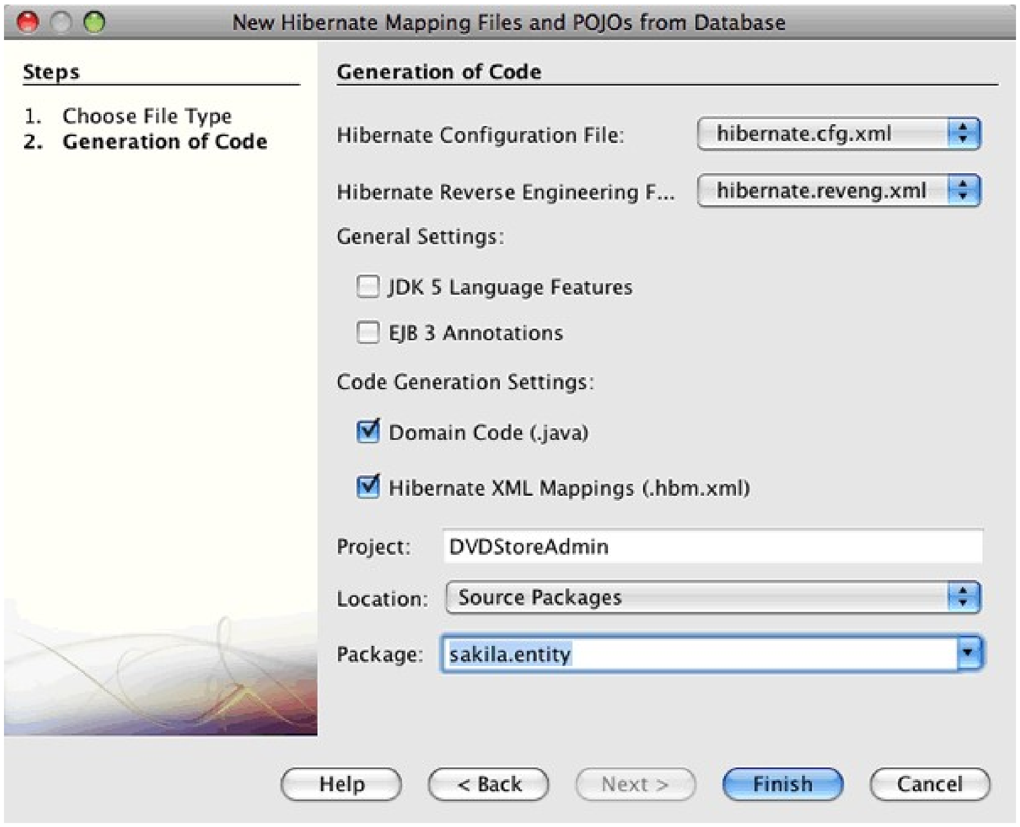
The wizard generates a hibernate.reveng.xml reverse engineering file. You can close the reverse engineering file because you will not need to edit the file.

Creating Hibernate Mapping Files and POJOs From a Database

The Hibernate Mapping Files and POJOs from a Database wizard generates files based on tables in a database. When you use the wizard, the IDE generates POJOs and mapping files for you based on the database tables specified in hibernate.reveng.xml and then adds the mapping entries to hibernate.cfg.xml. When you use the wizard you can choose the files that you want the IDE to generate (only the POJOs, for example) and select code generation options (generate code that uses EJB 3 annotations, for example).

1. Right-click the Source Packages node in the Projects window and choose New > Other to open the New File wizard.

2. Select Hibernate Mapping Files and POJOs from a Database in the Hibernate category. Click Next.
3. Select hibernate.cfg.xml from the Hibernate Configuration File dropdown list, if not selected.
4. Select hibernate.reveng.xml from the Hibernate Reverse Engineering File dropdown list, if not selected.
5. Ensure that the **Domain Code** and **Hibernate XML Mappings** options are selected.
6. Type **sakila.entity** for the Package name. Click Finish.


New Hibernate Mapping Files and POJOs from Database

Steps
1. Choose File Type
2. Generation of Code

Generation of Code

Hibernate Configuration File: hibernate.cfg.xml

Hibernate Reverse Engineering F... hibernate.reveng.xml

General Settings:

- [ ] JDK 5 Language Features
- [ ] EJB 3 Annotations

Code Generation Settings:

- [x] Domain Code (.java)
- [x] Hibernate XML Mappings (.hbm.xml)

Project: DVDStoreAdmin

Location: Source Packages

Package: sakila.entity

When you click Finish, the IDE generates the POJO Actor.java with all the required fields and generates a Hibernate mapping file and adds the mapping entry to hibernate.cfg.xml.

Now that you have the POJO and necessary Hibernate-related files you can create a simple Java GUI front end for the application. You will also create and then add an HQL query that queries the database to retrieve the data. In this process, we also use the HQL editor to build and test the query.

4.1.3 XAMPP

XAMPP is a free and open source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. XAMPP stands for Cross-Platform (X), Apache (A), MariaDB (M), PHP (P) and Perl (P). It is a simple, lightweight Apache distribution that makes it extremely easy for developers to create a local web server for testing and deployment purposes. Everything needed to set up a web server – server application (Apache), database (MariaDB), and scripting language (PHP) – is included in an extractable file. XAMPP is also cross-platform, which means it works equally well on Linux, Mac and Windows. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server extremely easy as well.

XAMPP is regularly updated to the latest releases of Apache, MariaDB, PHP and Perl. It also comes with a number of other modules including OpenSSL, phpMyAdmin, MediaWiki, Joomla, WordPress and more. Self-contained, multiple instances of XAMPP can exist on a single computer, and any given instance can be copied from one computer to another. XAMPP is offered in both a full and a standard version (Smaller version).

Open source tools is a phrase used to mean a program -- or tool -- that performs a very specific task, in which the source code is openly published for use and/or modification from its original design, free of charge. Open source tools are typically created as a collaborative effort in which programmers improve upon the code and share the changes within the community, and is usually available at no charge under a license defined by the Open Source Initiative.

Open source tools may be viable alternatives to popular closed-source applications and some open source tools offers features or performance benefits that surpass their commercial counterparts. The phrase open source tools are synonymous with open source utility and similar to open source applications.

Installation of XAMPP

XAMPP for Windows 7 version provides an easy to install Apache-MySQL-PHP-PERL-PEAR framework. XAMPP saves time and effort and provides the software support for web frameworks like Drupal, Joomla, Moodle, or Wikimedia on any Windows PC.

- In your web browser, go to https://www.apachefriends.org/index.html
- Click on the download link for XAMPP.
- When prompted for the download, click "Save" and wait for your download to finish.
- Open CD or DVD drive from My Computer. Install the program, and click on "Run."
- Accept the default settings. A command will open and offer an initial installation prompt. Just hit the Enter key, and accept the default settings. To simplify installation, just hit ENTER when prompted on the command line. You can always change settings, by editing the configuration files later.
- When your installation is complete, exit the command window by typing x on the command line.
- Start the XAMPP Control Panel.
- Start the Apache and MySQL components. You can also start the other components, if you plan to use them.
- Verify the Apache install, by clicking on the Apache administrative link in the Control Panel.
- Verify the MySQL installation, by clicking on the MySQL administrative link in the XAMPP Control Panel.
- If the verification steps are successful, XAMPP should be successfully installed on your PC. Open a browser and enter "localhost" on your address bar. You will be redirected to a page telling you that you've successfully installed XAMPP on your system.

4.2 Implementation Code

4.2.1 Code for Hibernate.cfg file, this file tells hibernate about which db. to use and other stuff

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
  <session-factory>
    <property name="hibernate.dialect">org.hibernate.dialect.DerbyDialect</property>
    <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
    <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/project</property>
    <property name="hibernate.connection.username">neeraj</property>
    <property name="hibernate.connection.password">neeraj</property>
    <mapping class="" file="" jar="" package="" resource="hibernate.hbm.xml"/>
  </session-factory>
</hibernate-configuration>
```

4.2.2 Hibernate Mapping file:

This file maps database tables into Java classes

FILENAME: hibernate.hbn.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
  <class name="tables.User" table="users">
      <meta attribute="class-description">
          this class contains users details
      </meta>
      <id column="id" name="id" type="int">
          <generator class="native" />
      </id>
      <property column="username" name="username" type="string" />
      <property column="password" name="password" type="string" />
  </class>
  <class name="tables.Criminal" table="criminals">
      <meta attribute="class-description">
          this class contains criminal details
      </meta>
      <id column="id" name="id" type="int">
          <generator class="native" />
      </id>
      <property column="name" name="name" type="string" />
      <property column="age" name="age" type="int" />
      <property column="height" name="height" type="int" />
      <property column="skin_color" name="skin_color" type="string" />
      <property column="others" name="others" type="string" />
  </class>
  <class name="tables.Site" table="sites">
      <meta attribute="class-description">
          this class contains site details
      </meta>
      <id column="id" name="id" type="int">
          <generator class="native" />
      </id>
      <property column="ip" name="ip" type="string" />
      <property column="port" name="port" type="int" />

  </class>
  <class name="tables.Inventory" table="inventory">
      <meta attribute="class-description">
          this class contains inventory details
      </meta>
      <id column="id" name="id" type="int" >
          <generator class="native" />
      </id>
      <property column="name" name="name" type="string" />
      <property column="value" name="value" type="int" />
  </class>

</hibernate-mapping>
```

4.2.3 This Java code Help Initialize hibernate
This is auto generated code

FILENAME: HibernateUtil.java

```
package socketdb;

import org.hibernate.cfg.AnnotationConfiguration;
import org.hibernate.SessionFactory;

/**
 * Hibernate Utility class with a convenient method to get Session Factory
 * object.
 *
 * @author neeraj
 */
public class HibernateUtil {

    private static final SessionFactory sessionFactory;

    static {
        try {
            // Create the SessionFactory from standard (hibernate.cfg.xml)
            // config file.
            sessionFactory = new AnnotationConfiguration().configure().buildSessionFactory();
        } catch (Throwable ex) {
            // Log the exception.
            System.err.println("Initial SessionFactory creation failed." + ex);
            throw new ExceptionInInitializerError(ex);
        }
    }

    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }
}
```

4.2.4 This code Help us to communicate (our software) with Hibernate framework

FILENAME: SocketDb.java

```
package socketdb;

import frames.MainFrame;
import java.awt.Component;
import javax.swing.JOptionPane;
import org.hibernate.HibernateException;
import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.Transaction;

/**
 *
 * @author neeraj
 */
public class SocketDb {

    /**
     * @param args the command line arguments
     */




      public static void main(String[] args) {
       // TODO code application logic here
       java.awt.EventQueue.invokeLater(new Runnable() {
           public void run() {
               new MainFrame().setVisible(true);
           }
       });
    }
    public static void insert(Component root, Object entry , String success, String failure){
       Session session = HibernateUtil.getSessionFactory().openSession();;
       Transaction tx = null;
       try{
           tx = session.beginTransaction();
           session.saveOrUpdate(entry);
           tx.commit();
           JOptionPane.showMessageDialog(root,            success            ,            "success"            ,
JOptionPane.INFORMATION_MESSAGE);
       }catch(HibernateException e){
           if(tx != null) tx.rollback();
           e.printStackTrace();
           JOptionPane.showMessageDialog(root,                        failure,                        "Error",
JOptionPane.WARNING_MESSAGE);
       }finally{
           session.close();
       }
    }

    public static Query createQuery(String hql){
       Session session = HibernateUtil.getSessionFactory().openSession();;
       Query q = session.createQuery(hql);
       session.close();
       return q;
    }
}
```

4.2.5 MAIN PROGRAM

FILENAME: MainFrame.java

```
package frames;

import java.util.ArrayList;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import socketdb.RemoteHelper;
import socketdb.GetCriminals;
import socketdb.LocalServer;
import socketdb.SocketDb;

/**
 *
 * @author neeraj
 */
public class MainFrame extends JFrame {
    /**
     * Creates new form MainFrame
     */
    public MainFrame() {
        initComponents();
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {
        jLabel1 = new javax.swing.JLabel();
        IDCheckBox = new javax.swing.JCheckBox();
        NameCheckBox = new javax.swing.JCheckBox();
        AgeCheckBox = new javax.swing.JCheckBox();
        SkinColorCheckBox = new javax.swing.JCheckBox();
        HeightCheckBox = new javax.swing.JCheckBox();
        OthersCheckBox = new javax.swing.JCheckBox();
        IDTextField = new javax.swing.JTextField();
        NameTextField = new javax.swing.JTextField();
        SkinColorTextField = new javax.swing.JTextField();
        HeightTextField = new javax.swing.JTextField();
        OthersTextField = new javax.swing.JTextField();
        AgeTextField = new javax.swing.JTextField();
        SearchButton = new javax.swing.JButton();
        SearchRemoteButton = new javax.swing.JButton();
        ServerLabel = new javax.swing.JLabel();
        jMenuBar1 = new javax.swing.JMenuBar();
        FileMenu = new javax.swing.JMenu();
```

```
AddCriminalMenuItem = new javax.swing.JMenuItem();
AddUserMenuItem = new javax.swing.JMenuItem();
AddSitesMenuItem = new javax.swing.JMenuItem();
AddInventoryItemMenuItem = new javax.swing.JMenuItem();
ShowAllMenuItem = new javax.swing.JMenuItem();
ServerMenu = new javax.swing.JMenu();
RunServerMenuItem = new javax.swing.JMenuItem();
StopServerMenuItem = new javax.swing.JMenuItem();
HelpMenu = new javax.swing.JMenu();
AboutMenuItem = new javax.swing.JMenuItem();
setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
jLabel1.setText("Query Criminals");
IDCheckBox.setText("ID");
IDCheckBox.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      IDCheckBoxActionPerformed(evt);
   }
});
NameCheckBox.setText("Name");
NameCheckBox.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      NameCheckBoxActionPerformed(evt);
   }
});
AgeCheckBox.setText("Age");
AgeCheckBox.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      AgeCheckBoxActionPerformed(evt);
   }
});
SkinColorCheckBox.setText("Skin Color");
SkinColorCheckBox.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      SkinColorCheckBoxActionPerformed(evt);
   }
});
HeightCheckBox.setText("Height");
HeightCheckBox.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      HeightCheckBoxActionPerformed(evt);
   }
});
OthersCheckBox.setText("Others");
OthersCheckBox.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      OthersCheckBoxActionPerformed(evt);
   }
});
IDTextField.setEnabled(false);
NameTextField.setEnabled(false);
```

```
        SkinColorTextField.setEnabled(false);
        SkinColorTextField.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                SkinColorTextFieldActionPerformed(evt);
            }
        });
        HeightTextField.setEnabled(false);
        OthersTextField.setEnabled(false);
        AgeTextField.setEnabled(false);
        SearchButton.setText("Search");
        SearchButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                SearchButtonActionPerformed(evt);
            }
        });
        SearchRemoteButton.setText("Search remote");
        SearchRemoteButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                SearchRemoteButtonActionPerformed(evt);
            }
        });
        ServerLabel.setText("Server: OFF");
        FileMenu.setText("File");

AddCriminalMenuItem.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEve
nt.VK_N, java.awt.event.InputEvent.CTRL_MASK));
        AddCriminalMenuItem.setText("Add new criminal record");
        AddCriminalMenuItem.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                AddCriminalMenuItemActionPerformed(evt);
            }
        });
        FileMenu.add(AddCriminalMenuItem);
        AddUserMenuItem.setText("Add new user");
        AddUserMenuItem.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                AddUserMenuItemActionPerformed(evt);
            }
        });
        FileMenu.add(AddUserMenuItem);
        AddSitesMenuItem.setText("Add sites");
        AddSitesMenuItem.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                AddSitesMenuItemActionPerformed(evt);
            }
        });
        FileMenu.add(AddSitesMenuItem);
        AddInventoryItemMenuItem.setText("Add inventory item");
        AddInventoryItemMenuItem.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                AddInventoryItemMenuItemActionPerformed(evt);
            }
        });
        FileMenu.add(AddInventoryItemMenuItem);

ShowAllMenuItem.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.V
K_A, java.awt.event.InputEvent.CTRL_MASK));
        ShowAllMenuItem.setText("Show all criminals");
        ShowAllMenuItem.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                ShowAllMenuItemActionPerformed(evt);
            }
        });
        FileMenu.add(ShowAllMenuItem);
        jMenuBar1.add(FileMenu);
        ServerMenu.setText("Server");
        RunServerMenuItem.setText("Run server");
        RunServerMenuItem.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                RunServerMenuItemActionPerformed(evt);
            }
        });
        ServerMenu.add(RunServerMenuItem);

        StopServerMenuItem.setText("Stop server");
        StopServerMenuItem.setEnabled(false);
        StopServerMenuItem.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                StopServerMenuItemActionPerformed(evt);
            }
        });
        ServerMenu.add(StopServerMenuItem);

        jMenuBar1.add(ServerMenu);

        HelpMenu.setText("Help");

        AboutMenuItem.setText("About");
        AboutMenuItem.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                AboutMenuItemActionPerformed(evt);
            }
        });
        HelpMenu.add(AboutMenuItem);

        jMenuBar1.add(HelpMenu);

        setJMenuBar(jMenuBar1);

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(62, 62, 62)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(IDCheckBox)
                    .addComponent(NameCheckBox)
                    .addComponent(SkinColorCheckBox)
                    .addComponent(HeightCheckBox)
                    .addComponent(OthersCheckBox)
                    .addComponent(AgeCheckBox))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(NameTextField,        javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 81, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(IDTextField,          javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 81, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(SkinColorTextField,  javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 81, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(HeightTextField,      javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 81, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(AgeTextField,         javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 81, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(OthersTextField,      javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 81, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(94, 94, 94))
            .addGroup(layout.createSequentialGroup()
                .addGap(163, 163, 163)
                .addComponent(jLabel1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(ServerLabel)
                .addContainerGap())
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                .addContainerGap(67, Short.MAX_VALUE)
                .addComponent(SearchRemoteButton)
                .addGap(87, 87, 87)
                .addComponent(SearchButton)
                .addGap(78, 78, 78))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(jLabel1))
                    .addComponent(ServerLabel))
            .addGap(17, 17, 17)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
               .addComponent(IDCheckBox)
               .addComponent(IDTextField,             javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
               .addComponent(NameCheckBox)
               .addComponent(NameTextField,           javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
               .addComponent(SkinColorCheckBox)
               .addComponent(SkinColorTextField,      javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
               .addComponent(HeightCheckBox)
               .addComponent(HeightTextField,         javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
               .addComponent(AgeCheckBox)
               .addComponent(AgeTextField,            javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
               .addComponent(OthersCheckBox)
               .addComponent(OthersTextField,         javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(18, 18, 18)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
               .addComponent(SearchButton)
               .addComponent(SearchRemoteButton))
            .addContainerGap(20, Short.MAX_VALUE))
      );
   pack();
   }// </editor-fold>
   private void AddCriminalMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
      // TODO add your handling code here:
      java.awt.EventQueue.invokeLater(new Runnable() {
         public void run() {
            new AddCriminalForm().setVisible(true);
         }
      });
   }
   private void IDCheckBoxActionPerformed(java.awt.event.ActionEvent evt) {
      // TODO add your handling code here:
      if(IDTextField.isEnabled()){
         IDTextField.setEnabled(false);
```

```
    }else{
        IDTextField.setEnabled(true);
    }
}
private void AgeCheckBoxActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if(AgeTextField.isEnabled()){
        AgeTextField.setEnabled(false);
    }else{
        AgeTextField.setEnabled(true);
    }
}
private void SkinColorCheckBoxActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if(SkinColorTextField.isEnabled()){
        SkinColorTextField.setEnabled(false);
    }else{
        SkinColorTextField.setEnabled(true);
    }
}
private void HeightCheckBoxActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if(HeightTextField.isEnabled()){
        HeightTextField.setEnabled(false);
    }else{
        HeightTextField.setEnabled(true);
    }
}
private void OthersCheckBoxActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if(OthersTextField.isEnabled()){
        OthersTextField.setEnabled(false);
    }else{
        OthersTextField.setEnabled(true);
    }
}
private void ShowAllMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    GetCriminals.showAllCriminals();
}
private void NameCheckBoxActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if(NameTextField.isEnabled()){
        NameTextField.setEnabled(false);
    }else{
        NameTextField.setEnabled(true);
    }
}
private void SearchButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
        boolean nameCheckbox = NameCheckBox.isSelected(), idCheckbox =
IDCheckBox.isSelected(), ageCheckbox = AgeCheckBox.isSelected(), heightCheckbox =
HeightCheckBox.isSelected() ,othersCheckbox = OthersCheckBox.isSelected() , skincolorCheckbox
= SkinColorCheckBox.isSelected();
        if(!(nameCheckbox || idCheckbox || ageCheckbox || othersCheckbox || skincolorCheckbox ||
heightCheckbox)){
            System.out.println("nothing selected, please select atleast one!");
            return;
        }
        String query = "FROM Criminal C WHERE";
        ArrayList<String> l = new ArrayList<String>();
        if(nameCheckbox){
            //String t = " C.id = " + IDTextField.getText();
            l.add(" C.name = " +"'" + NameTextField.getText() + "'");
        }
        if(idCheckbox){
            l.add(" C.id = " + IDTextField.getText());
        }
        if(ageCheckbox){
            l.add(" C.age = " + AgeTextField.getText());
        }
        if(heightCheckbox){
            l.add(" C.height = " + HeightTextField.getText());
        }
        if(othersCheckbox){
            l.add(" C.others = " + "'" + OthersTextField.getText() + "'");
        }
        if(skincolorCheckbox){
            l.add(" C.skin_color = " + "'" + SkinColorTextField.getText() + "'");
        }
        String joined = String.join("and",l);
        query += joined;
        GetCriminals.showWithQuery(query);
    }
    private void AddInventoryItemMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new AddInventoryItemForm().setVisible(true);
            }
        });
    }
    private void AddUserMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new AddUserForm().setVisible(true);
            }
        });
    }
    private void AddSitesMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new AddSitesForm().setVisible(true);
            }
        });
    }
    private void AboutMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        JOptionPane.showMessageDialog(this, "This software is daveloped by: Neeraj Nigam,
Himanshu, Himanshi, Tarun");
    }
    private void SearchRemoteButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        boolean nameCheckbox = NameCheckBox.isSelected(), idCheckbox =
IDCheckBox.isSelected(), ageCheckbox = AgeCheckBox.isSelected(), heightCheckbox =
HeightCheckBox.isSelected() ,othersCheckbox = OthersCheckBox.isSelected() , skincolorCheckbox
= SkinColorCheckBox.isSelected();
        if(!(nameCheckbox || idCheckbox || ageCheckbox || othersCheckbox || skincolorCheckbox ||
heightCheckbox)){
            System.out.println("nothing selected, please select atleast one!");
            return;
        }
        String query = "FROM Criminal C WHERE";
        ArrayList<String> l = new ArrayList<String>();
        if(nameCheckbox){
            //String t = " C.id = " + IDTextField.getText();
            l.add(" C.name = " +"'" + NameTextField.getText() + "'");
        }
        if(idCheckbox){
            l.add(" C.id = " + IDTextField.getText());
        }
        if(ageCheckbox){
            l.add(" C.age = " + AgeTextField.getText());
        }
        if(heightCheckbox){
            l.add(" C.height = " + HeightTextField.getText());
        }
        if(othersCheckbox){
            l.add(" C.others = " + "'" + OthersTextField.getText() + "'");
        }
        if(skincolorCheckbox){
            l.add(" C.skin_color = " + "'" + SkinColorTextField.getText() + "'");
        }
        String joined = String.join("and",l);
        query += joined;
        RemoteHelper c = new RemoteHelper();
        ArrayList<String> resultList = c.requestQuery(query);
        java.awt.EventQueue.invokeLater(new Runnable() {
```

```
        public void run() {
            new ShowDataForm(resultList).setVisible(true);
        }
    });
}
private void SkinColorTextFieldActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}
private void RunServerMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    SocketDb.getServer().startServer();
    RunServerMenuItem.setEnabled(false);
    StopServerMenuItem.setEnabled(true);
    ServerLabel.setText("Server: ON");

}

private void StopServerMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    SocketDb.getServer().stopServer();
    RunServerMenuItem.setEnabled(true);
    StopServerMenuItem.setEnabled(false);
    ServerLabel.setText("Server: OFF");
}

public JFrame getComponent(){
    return this;
}
/**
 * @param args the command line arguments
 */

// Variables declaration - do not modify
private javax.swing.JMenuItem AboutMenuItem;
private javax.swing.JMenuItem AddCriminalMenuItem;
private javax.swing.JMenuItem AddInventoryItemMenuItem;
private javax.swing.JMenuItem AddSitesMenuItem;
private javax.swing.JMenuItem AddUserMenuItem;
private javax.swing.JCheckBox AgeCheckBox;
private javax.swing.JTextField AgeTextField;
private javax.swing.JMenu FileMenu;
private javax.swing.JCheckBox HeightCheckBox;
private javax.swing.JTextField HeightTextField;
private javax.swing.JMenu HelpMenu;
private javax.swing.JCheckBox IDCheckBox;
private javax.swing.JTextField IDTextField;
private javax.swing.JCheckBox NameCheckBox;
private javax.swing.JTextField NameTextField;
private javax.swing.JCheckBox OthersCheckBox;
```

```
    private javax.swing.JTextField OthersTextField;
    private javax.swing.JMenuItem RunServerMenuItem;
    private javax.swing.JButton SearchButton;
    private javax.swing.JButton SearchRemoteButton;
    private javax.swing.JLabel ServerLabel;
    private javax.swing.JMenu ServerMenu;
    private javax.swing.JMenuItem ShowAllMenuItem;
    private javax.swing.JCheckBox SkinColorCheckBox;
    private javax.swing.JTextField SkinColorTextField;
    private javax.swing.JMenuItem StopServerMenuItem;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JMenuBar jMenuBar1;
    // End of variables declaration

}
```

4.2.6 Criminal form java class

FILENAME: AddCriminalForm.java

```
package frames;

import javax.swing.JOptionPane;
import org.hibernate.HibernateException;
import org.hibernate.Session;
import org.hibernate.Transaction;
import socketdb.HibernateUtil;
import socketdb.SocketDb;
import tables.Criminal;
/**
 *
 * @author neeraj
 */
public class AddCriminalForm extends javax.swing.JFrame {

    /**
     * Creates new form AddCriminalForm
     */
    public AddCriminalForm() {
        initComponents();
    }
    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {
        NameLabel = new javax.swing.JLabel();
        NameTextField = new javax.swing.JTextField();
        SkinColorLabel = new javax.swing.JLabel();
        SkinColorTextField = new javax.swing.JTextField();
        HeightLabel = new javax.swing.JLabel();
        HeightTextField = new javax.swing.JTextField();
        AgeLabel = new javax.swing.JLabel();
        OthersLabel = new javax.swing.JLabel();
        AgeTextField = new javax.swing.JTextField();
        OthersTextField = new javax.swing.JTextField();
        AddButton = new javax.swing.JButton();
        AddNewButton = new javax.swing.JButton();
        jLabel1 = new javax.swing.JLabel();
        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);

        NameLabel.setText("Name");
        SkinColorLabel.setText("Skin Color");
        HeightLabel.setText("Height");
        AgeLabel.setText("Age");
        OthersLabel.setText("Others");
        AddButton.setText("Add");
        AddButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                AddButtonActionPerformed(evt);
            }
        });
        AddNewButton.setText("Add New");
        AddNewButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                AddNewButtonActionPerformed(evt);
            }
        });

        jLabel1.setText("Add criminal record form");
        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(47, 47, 47)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(HeightLabel)
                    .addComponent(NameLabel)
                    .addComponent(SkinColorLabel))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,              34,
Short.MAX_VALUE)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addComponent(SkinColorTextField, javax.swing.GroupLayout.PREFERRED_SIZE,
65, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(NameTextField, javax.swing.GroupLayout.PREFERRED_SIZE, 65,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(HeightTextField, javax.swing.GroupLayout.PREFERRED_SIZE, 65,
javax.swing.GroupLayout.PREFERRED_SIZE))
                  .addComponent(AddButton))
                .addGap(58, 58, 58)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                  .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                      .addComponent(AgeLabel)
                      .addComponent(OthersLabel))
                    .addGap(44, 44, 44)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
                      .addComponent(OthersTextField, javax.swing.GroupLayout.DEFAULT_SIZE, 65,
Short.MAX_VALUE)
                      .addComponent(AgeTextField)))
                  .addComponent(AddNewButton))
                .addGap(62, 62, 62))
              .addGroup(layout.createSequentialGroup()
                .addGap(154, 154, 154)
                .addComponent(jLabel1)
                .addGap(0, 0, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel1)
                .addGap(27, 27, 27)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                  .addComponent(NameLabel)
                  .addComponent(NameTextField,          javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                  .addComponent(AgeLabel)
                  .addComponent(AgeTextField,           javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                  .addComponent(SkinColorLabel)
                  .addComponent(SkinColorTextField,     javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                  .addComponent(OthersLabel)
                  .addComponent(OthersTextField,        javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(HeightLabel)
                .addComponent(HeightTextField,          javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(60, 60, 60)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(AddButton)
                .addComponent(AddNewButton))
            .addContainerGap(69, Short.MAX_VALUE))
    );

    pack();
  }// </editor-fold>
  private void AddNewButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    this.addCriminal();
  }
  private void AddButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    this.addCriminal();
    dispose();
  }
  private void addCriminal(){
    String cname = NameTextField.getText();
    int age=0, height=0;
    if(cname == null || cname.equals("")){
      JOptionPane.showMessageDialog(rootPane,    "Name    Cannot    be    empty!",    "Error",
JOptionPane.WARNING_MESSAGE);
      return;
    }
    String ageStr = AgeTextField.getText(), heightStr = HeightTextField.getText();

    /// Work here //////////////////////
    if(!(ageStr.equals("") || heightStr.equals(""))){
      try{
        age = Integer.parseInt(AgeTextField.getText());
        height = Integer.parseInt(HeightTextField.getText());
      }catch(NumberFormatException e){
        e.printStackTrace();
        JOptionPane.showMessageDialog(rootPane, "Please enter a number", "Not a number",
JOptionPane.WARNING_MESSAGE);
        return;
      }
    }
    Criminal c = new Criminal();
    c.setName(cname);
    c.setAge(age);
    c.setHeight(height);
    c.setSkin_color(SkinColorTextField.getText());
    c.setOthers(OthersTextField.getText());
    String success = "Criminal information added to DB", failure = "something went wrong";
```

```
        SocketDb.insert(rootPane, c, success, failure);
        NameTextField.setText("");
        AgeTextField.setText("");
        SkinColorTextField.setText("");
        HeightTextField.setText("");
        OthersTextField.setText("");
    }
    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for                     (javax.swing.UIManager.LookAndFeelInfo                    info                    :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(AddCriminalForm.class.getName()).log(java.util.logging.Level.
SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(AddCriminalForm.class.getName()).log(java.util.logging.Level.
SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(AddCriminalForm.class.getName()).log(java.util.logging.Level.
SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(AddCriminalForm.class.getName()).log(java.util.logging.Level.
SEVERE, null, ex);
        }
        //</editor-fold>
  /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new AddCriminalForm().setVisible(true);
            }
        });
    }
    // Variables declaration - do not modify
    private javax.swing.JButton AddButton;
    private javax.swing.JButton AddNewButton;
    private javax.swing.JLabel AgeLabel;
    private javax.swing.JTextField AgeTextField;
    private javax.swing.JLabel HeightLabel;
    private javax.swing.JTextField HeightTextField;
    private javax.swing.JLabel NameLabel;
    private javax.swing.JTextField NameTextField;
    private javax.swing.JLabel OthersLabel;
    private javax.swing.JTextField OthersTextField;
    private javax.swing.JLabel SkinColorLabel;
    private javax.swing.JTextField SkinColorTextField;
    private javax.swing.JLabel jLabel1;
    // End of variables declaration
}
```

4.2.7 Database mapping files for each table
FILE NAME: Criminal.java

```
package tables;

/**
 *
 * @author neeraj
 */
public class Criminal {
    private int id;
    private String name;
    private int age;
    private int height;
    private String skin_color;
    private String others;

    public Criminal(){}
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
    public int getHeight() {
        return height;
    }
    public void setHeight(int height) {
        this.height = height;
    }
    public String getSkin_color() {
        return skin_color;
    }
    public void setSkin_color(String skin_color) {
        this.skin_color = skin_color;
    }
    public String getOthers() {
        return others;
    }
    public void setOthers(String others) {
        this.others = others;
    }
}
```

4.2.8 Inventory table
FILE NAME: Inventory.java

```
package tables;

/**
 *
 * @author neeraj
 */
public class Inventory {
    private int id;
    private String name;
    private int value;

    public Inventory(){}

    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public int getValue() {
        return value;
    }
    public void setValue(int value) {
        this.value = value;
    }
}
```

4.2.9 Site management

FILE NAME: Site.java

```
package tables;

/**
 *
 * @author neeraj
 */
public class Site {

    public String getIp() {
        return ip;
    }

    public void setIp(String ip) {
        this.ip = ip;
    }

    public int getPort() {
        return port;
    }

    public void setPort(int port) {
        this.port = port;
    }
    public int id;

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }
    private String ip;
    private String name;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }
    private int port;
    public Site(){}
    public Site(String ip, int port, String name){
        this.ip = ip;
        this.port = port;
        this.name =  name;
    }
}
```

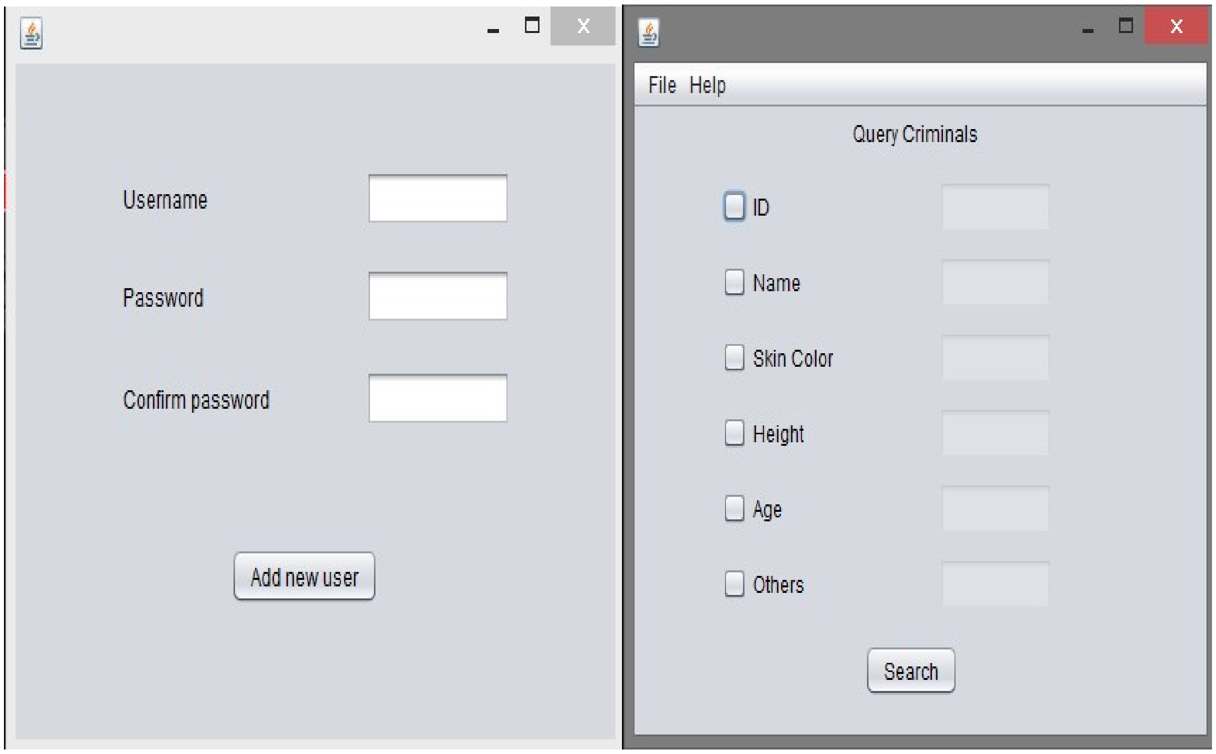
4.3 Chapter Summary

The given chapter provides the details of the tools used in implementation of the distributed system. After that the proposed scenario is represented and the technique used for establishing it is explained. Then, all the programs are shown that compiled together build up the application for static environment.

Chapter 5

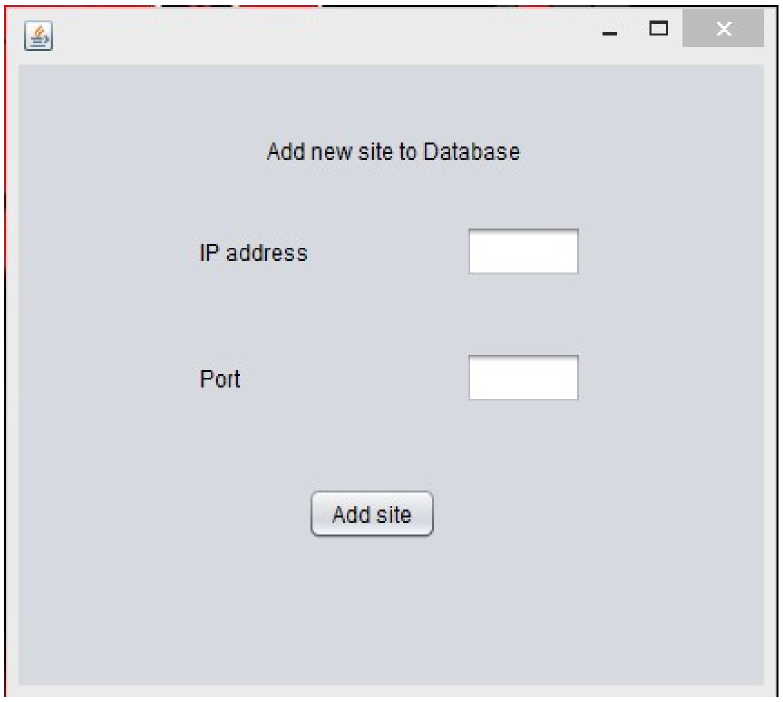
Result Analysis

This chapter provides the evaluation of results and their understanding. In addition of that the performance outcomes by which the results are concluded is also provided in this section.



This window is for authorization purpose and hence only legitimate user can access it. From here the Username and Password as a string will be taken to database and as per the existing record a user can gain access to the database.

While window shown on right side will enable the user to identify the criminal and fetch records as per the matching credentials. Since application include more than five such identification factors which makes the recognition of criminal quite easy and accurate.



This window is added for socket programming purpose. To gain access and fetch data from some other server in the network which is located in some different location this application via this window provide the ease. User has to enter IP address and Port No. and he can then operate some other server on his device.

Chapter 6
CONCLUSION

6.1 CONCLUSION

After development and performing detailed integration testing of the application we analysed that this application can be extended and using deep learning and IoT can thwart the crimes happening around and to identify and catch the thief.

6.2FUTURE WORK

In future, this project can also be implemented on all over the country's offices to keep record of employees and their activities. Further development can also be done in the direction of security improvement. Since currently system is more vulnerable and so is its data and can easily be manipulated or used by unauthorized user.

Our application can be connected to the CCTV cameras around the city and an identification at server end will be continuously checking a CCTV footage. And once a convict is identified from footage will ping the responsible authority.

References

1. O'Brien, J. & Marakas, G.M. (2014) Management Information Systems (pp. 185-189). New York, NY: McGraw-Hill Irwin
2. Ashdown, Lance; Kyte, Tom (September 2011). "Oracle Database Concepts, 11g Release 2 (11.2)". Oracle Corporation. Retrieved 2013-07-17. Distributed SQL synchronously accesses and updates data distributed among multiple databases. [...] Distributed SQL includes distributed queries and distributed transactions.
3. https://data.gov.in/dataset-group-name/crime-statistics
4. http://ncrb.gov.in/
5. Linwood, Jeff; Minter, Dave (May 28, 2016), Beginning Hibernate (Second ed.), Apress, p. 400, ISBN 1-4302-2850-4
6. Bernard, Emmanuel; Griffin, John (December 30, 2014), Hibernate Search in Action (First ed.), Manning Publications, p. 488, ISBN 1-933988-64-9
7. Elliott, James; O'Brien, Tim (April 22, 2008), Harnessing Hibernate (First ed.), O'Reilly Media, p. 380, ISBN 0-596-51772-6
8. "Schism", with its connotations, is a common usage, *e.g.* "the Lemacs/FSFmacs schism" (Jamie Zawinski, 2012), "Behind the KOffice split" (Joe Brockmeier, *Linux Weekly News*, 2012-12-14), "Copyright assignment - once bitten, twice shy" (Richard Hillesley, *H-Online*, 2012-08-06), "Forking is a feature" (Anil Dash, 2014-09-10), "The Great Software Schism" (Glyn Moody, *Linux Journal*, 2016-09-28), "To Fork Or Not To Fork: Lessons From Ubuntu and Debian" (Benjamin Mako Hill, 2015).
9. Finley, Klint. "7 Cloud-Based Database Services". *ReadWriteWeb*. Retrieved 9 November 2015
10. James, William (2015). "What is Emotion". Mind. **9**: 188–205. doi:10.1093/mind/os-IX.34.188. Cited by Tao & Tan 2015.
11. Kahneman, Daniel; Slovic, D.; Tversky, Amos (2014). Judgment under uncertainty: Heuristics and biases. New York: Cambridge University Press. ISBN 0-521-28414-7.
12. Katz, Yarden (1 November 2012). "Noam Chomsky on Where Artificial Intelligence Went Wrong". The Atlantic. Retrieved 26 October 2014.
13. "Kismet". MIT Artificial Intelligence Laboratory, Humanoid Robotics Group. Retrieved 25 October 2014.
14. Koza, John R. (2012). Genetic Programming (On the Programming of Computers by Means of Natural Selection). MIT Press. ISBN 0-262-11170-5.
15. Kleine-Cosack, Christian (October 2016). "Recognition and Simulation of Emotions" (PDF). Archived from the original (PDF) on 28 May 2016.
16. Kolata, G. (2014). "How can computers get common sense?". Science. **217** (4566): 1237–1238. doi:10.1126/science.217.4566.1237. PMID 17837639.