CSE

CAR SEARCH ENGINE

ACKNOWLEDGEMENT

First of all we thank our renowned institution for giving us an opportunity in doing the project. First and foremost we wish to place on records our ardent and earnest gratitude to the project guide Mrs.Nisa, Professor, Department of Computer Science and Engineering

We are extremely happy to mention our gratitude to Prof. Anasamma John, Head of the Department, Computer Science and Engineering for providing us with all the facilities for the completion of this work.

Finally, yet importantly we would like to express our gratitude to our lab in charge for their valuable assistance provided during the course of the project.

We would also extend our gratefulness to all the staff members in the department. We also thank all our friends and well-wishers who greatly helped in this endeavor.

5054,VISHNU PRASAD K.V

5048,VINAYAK M.P

5063,ASWIN M.S

5019,JITHIN THOMAS

5032,RAJESH C.R

5062,ZAYAN K

INDEX

ABSTRACT

Our mission at CAR SEARCH ENGINE is to be your ultimate solution for information on certified cars. Our engine is designed to give you more efficient consulting process and make finding a vehicle easier than ever before.Whether you're for a new certified our engine offers you these benefits.The largest selection of vehicle inventory from dealers and private sellersThe most comprehensive selection The most complete research, including specifications, photos, and moreVehicle information, including manufacturer details .CAR SEARCH ENGINE contains more than 7000+ vehicle listings. We provide the largest selection of vehicles.

INTRODUCTION

In the contermporary life computers have a great significance ineach and every field.As a result of the technological advancement in the field of electronics,computers have brought about a revelution in the field of computing,industries,scientific research and even in the sphere of artslike music,painting etc.The debugging and time consumptions in handling large quantities of data can be reduced by the use of computers.Here we develop such an advanced and efficient package fetures for the proper functioning of bank management system.

We use java as front end and oracle SQL as back end for developing our project. Java is a powerful language that support graphical interface for better user communication.Platform independent and security are the major importance of java.Oracle is a powerful relational database application with which a user can efficiently create and manipulate database systems.

SYSTEM STUDY

System analysis is a detailed study of various operations performed by the system and their relationships within and outside the system. The study of a system starts with analyzing the existing system.

A system can be developed based on existing data, past experience, innovative (implementable), investigation of what current system is lagging and so on. System analysis can be categorized as:

- System planning and initial investigation
- Information gathering
- Applying analysis tools for structured analysis

SYSTEM ANALYSIS:

The previous system that present in the car search engine were mainly manually organized. So that for data insertion and manipulation, lots of record was to be maintained. For insertion of a new entry various records were to be updated.

As technology advanced, storage devices developed, new software sprung up to efficiently store data, database concepts were introduced. Now the data can be maintained in a single place or can be linked as needed, hence reduced the need for keeping lots of records. Insertion, modification, deletion of records became easy. Error is reduced, efficient access of database, user friendly environment etc. make the system highly efficient.

The car search engine developed here is like prototype of the new system adopted around the world. Expansion of the system can be made as required. The feasibility of the system developed is satisfying with the present conditions. The resources needed are efficient to support current requirements and it os economic. As the software developed from existing data, there is not much economy involved. But for the development of the real project, a lot of analysis and development of software is needed.

FEASIBILITY STUDY

Whatever we think may not be feasible. It is wise to think about the feasibility of any problem we undertake. Feasibility is the study of impact, which happens in the organization by the development of the system. The impact can be either positive or negative. When the positives nominate the negatives, then system is considered feasible. Here the feasibility study can be performed in two ways such as Technical feasibility and Economic feasibility.

❖ TECHINCAL FEASIBILITY

We can strong say that it is technically feasible, since there will not be much be difficulty in getting required resources for the development and maintaining the system as well. All the resources for the development of the software as well as the maintenance of the same is available in the organization. Here we are utilizing the resources which are available already.

❖ ECONOMIC FEASIBILITY

Development of the application is highly feasible. The organization needed not spend much money for development of the system that is already available. The only thing is to be done is making an environment for the development with an effective supervision. If we are doing so, we attain the maximum usability of the corresponding resources. Even after the development, the organization will not be in a condition to invest more in the organization's share. Therefore the system is economically feasible.

SYSTEM REQUIREMENTS

Software Requirements:

Requirement	Value
System Architecture	Intel(x86),AMD64 and Intel EM64T
Front End	Java
Developing tool	JDK 7.0
Dtatabase	Oracle 11g

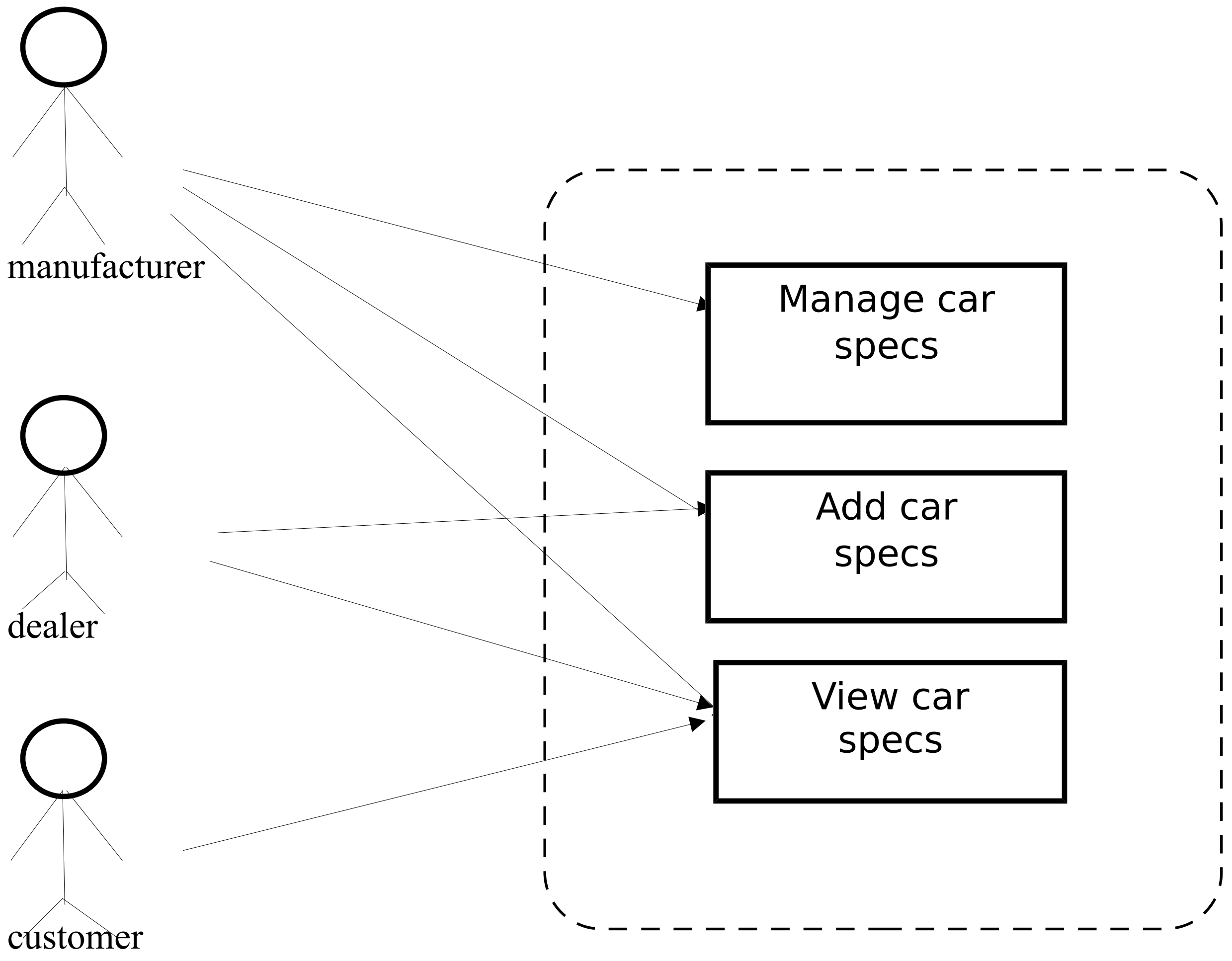
Hardware Requirements

Requirement	Value
Processors	2.0 GHz or Higher
RAM	256 MB or Higher
Hard Disk	20 GB or Higher
Operating System	Windows 8/10

DESIGN

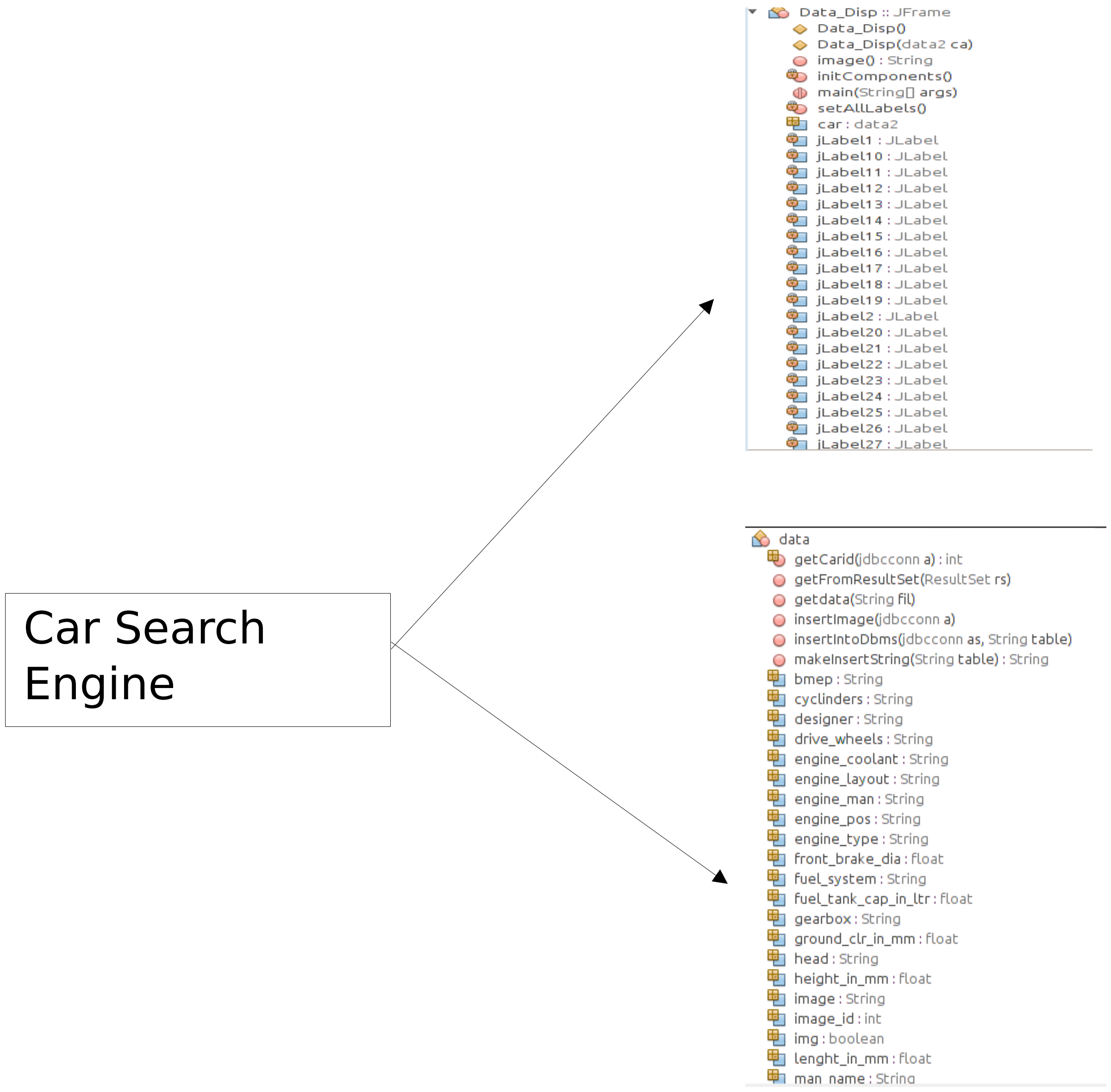
USE CASE DIAGRAM

A use case diagram depicts actors, use cases and the relationship among them. The use case concept was introduced by Ivar Jacobson. Use case is software engineering term that is equivalent to a social scientist's notation of task. The actor can be a human or external system. With some symbol extensions, use case diagram can be used to represent a usability professional's task model.



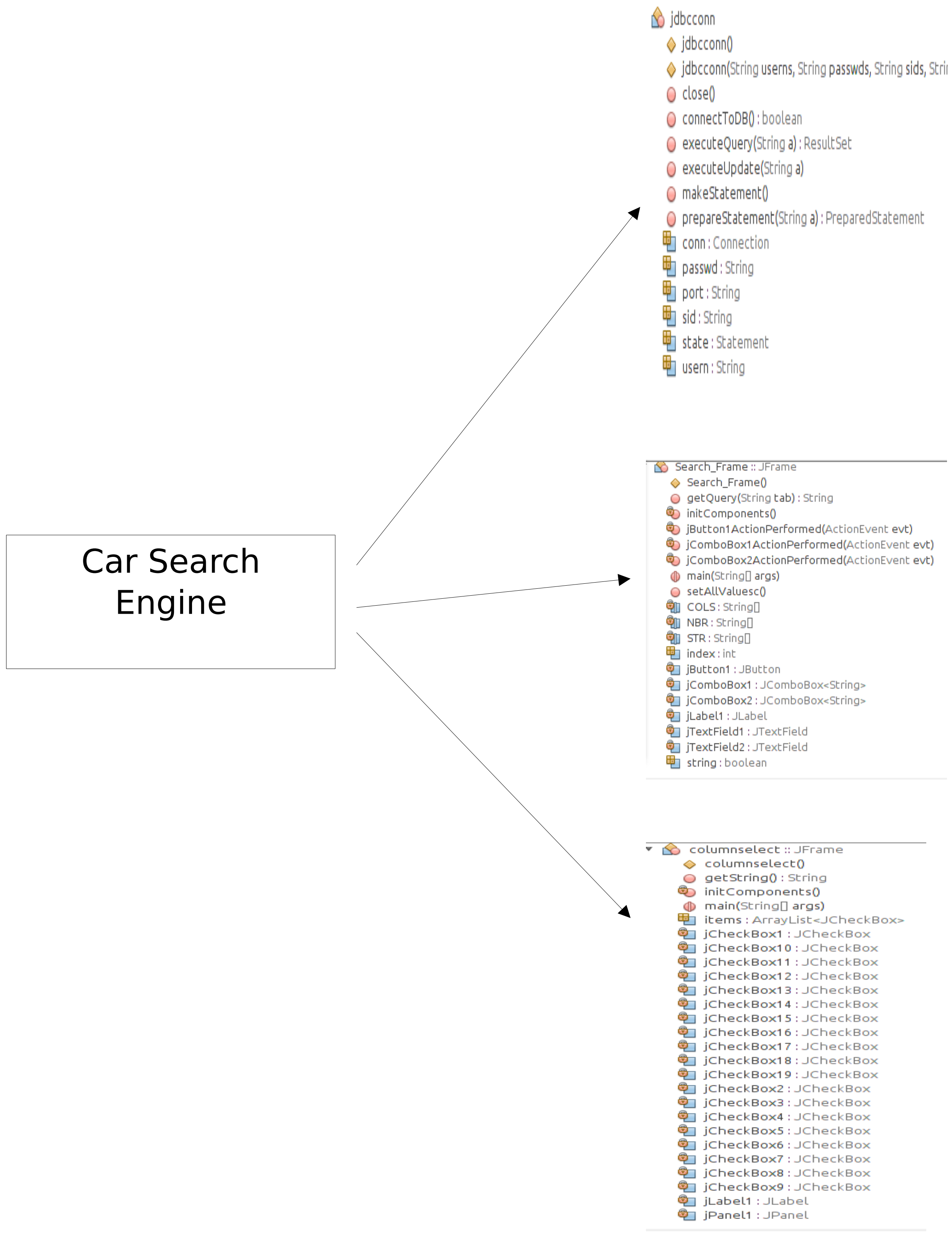
CLASS DIAGRAM

Class diagrams are the backbone of almost every object oriented method including UML. They describe the static structure of a system. Classes represents an abstraction of entities with common characteristics. Associations represent the relationship between classes. The various class diagram features are active class, visibility, association, multiplicity, constraints, composition, aggregation and generalization.



```
Data_Disp :: JFrame
    Data_Disp()
    Data_Disp(data2 ca)
    image() : String
    initComponents()
    main(String[] args)
    setAllLabels()
    car : data2
    jLabel1 : JLabel
    jLabel10 : JLabel
    jLabel11 : JLabel
    jLabel12 : JLabel
    jLabel13 : JLabel
    jLabel14 : JLabel
    jLabel15 : JLabel
    jLabel16 : JLabel
    jLabel17 : JLabel
    jLabel18 : JLabel
    jLabel19 : JLabel
    jLabel2 : JLabel
    jLabel20 : JLabel
    jLabel21 : JLabel
    jLabel22 : JLabel
    jLabel23 : JLabel
    jLabel24 : JLabel
    jLabel25 : JLabel
    jLabel26 : JLabel
    jLabel27 : JLabel
```

```
data
    getCarid(jdbcconn a) : int
    getFromResultSet(ResultSet rs)
    getdata(String fil)
    insertImage(jdbcconn a)
    insertIntoDbms(jdbcconn as, String table)
    makeInsertString(String table) : String
    bmep : String
    cyclinders : String
    designer : String
    drive_wheels : String
    engine_coolant : String
    engine_layout : String
    engine_man : String
    engine_pos : String
    engine_type : String
    front_brake_dia : float
    fuel_system : String
    fuel_tank_cap_in_ltr : float
    gearbox : String
    ground_clr_in_mm : float
    head : String
    height_in_mm : float
    image : String
    image_id : int
    img : boolean
    lenght_in_mm : float
    man_name : String
```



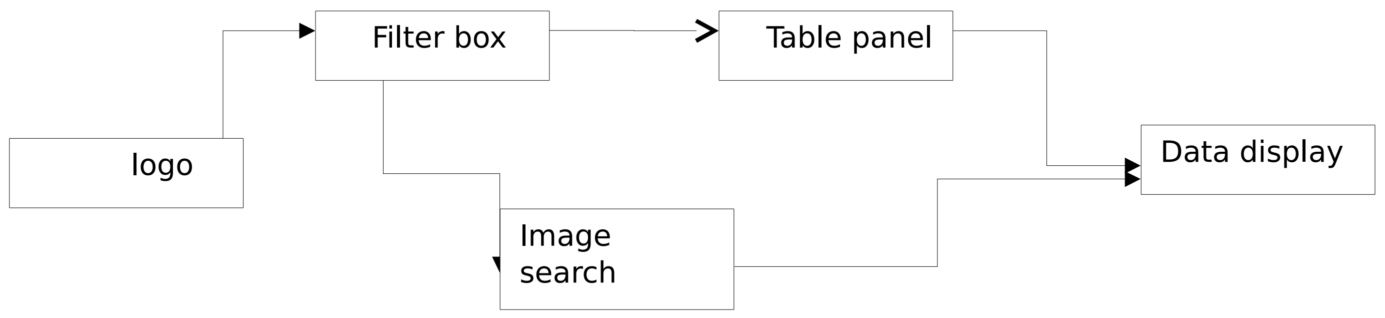
```
jdbcconn
    jdbcconn()
    jdbcconn(String userns, String passwds, String sids, Stri
    close()
    connectToDB() : boolean
    executeQuery(String a) : ResultSet
    executeUpdate(String a)
    makeStatement()
    prepareStatement(String a) : PreparedStatement
    conn : Connection
    passwd : String
    port : String
    sid : String
    state : Statement
    usern : String
```

```
Search_Frame :: JFrame
    Search_Frame()
    getQuery(String tab) : String
    initComponents()
    jButton1ActionPerformed(ActionEvent evt)
    jComboBox1ActionPerformed(ActionEvent evt)
    jComboBox2ActionPerformed(ActionEvent evt)
    main(String[] args)
    setAllValuesc()
    COLS : String[]
    NBR : String[]
    STR : String[]
    index : int
    jButton1 : JButton
    jComboBox1 : JComboBox<String>
    jComboBox2 : JComboBox<String>
    jLabel1 : JLabel
    jTextField1 : JTextField
    jTextField2 : JTextField
    string : boolean
```

```
columnselect :: JFrame
    columnselect()
    getString() : String
    initComponents()
    main(String[] args)
    items : ArrayList<JCheckBox>
    jCheckBox1 : JCheckBox
    jCheckBox10 : JCheckBox
    jCheckBox11 : JCheckBox
    jCheckBox12 : JCheckBox
    jCheckBox13 : JCheckBox
    jCheckBox14 : JCheckBox
    jCheckBox15 : JCheckBox
    jCheckBox16 : JCheckBox
    jCheckBox17 : JCheckBox
    jCheckBox18 : JCheckBox
    jCheckBox19 : JCheckBox
    jCheckBox2 : JCheckBox
    jCheckBox3 : JCheckBox
    jCheckBox4 : JCheckBox
    jCheckBox5 : JCheckBox
    jCheckBox6 : JCheckBox
    jCheckBox7 : JCheckBox
    jCheckBox8 : JCheckBox
    jCheckBox9 : JCheckBox
    jLabel1 : JLabel
    jPanel1 : JPanel
```

DATA FLOW DIAGRAM

A data flow diagram (DFD) is a graphical representation of the “flow” of data through an information system, modelling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFD’s can also be used for the visualization of the data processing (structured deisgn).
A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel.



IMPLEMENTATION

Swing Package Basics

The Java foundation class(JFC) is used to develop the GUI applications which provides user friendly environment for the end-users. This package introduces a platform-independent view of GUI and thus removes the drawback of awt package. The drawbacks of awt class have been removed by the introduction of JFC. Most swing components are pure java components- they are written, manipulated and displayed in java. Most swing components are not tied to actual GUI components supported by the under lying platform on which an application executes. Such GUI components are known as light weight components. The visual display component and the action components are together called LOOK and FEEL component. The LOOK and FEEL component of a swing can either be system dependent or independent depending upon the choice of the programmer , this is an advantage of swing over the awt . several swing GUI components from package javax.swing are given below that are used to build java GUIs.

Components	**Description**
jLabel	Display uneditable text or icon
jTextField	Enable users to enter data from the keyboard. Can also be used to display editable or uneditable text
jButton	Triggers and event when clicked with the mouse
jComboBox	Provides a drop down list of items from which user can make a selection by clicking an item or possible by typing into the box.
jList	Provide a list of items for which the user can make a selection by clicking on any item in the list
jPanel	Provide an area in which components can be placed and organized. Can also be used as a drawing area for graphics.

jCheckBox	Specifies an option that can be selected or not selected.

SQL(Structured Query Language):

A database is an organized collection of data. There are many different strategies of organizing data to facilitate easy access and manipulation. Today's most popular systems are Relational Database. In current DBMS a comprehensive integrated language is used that includes constructs for conceptual schema definition, view definition and data manipulation. That comprehensive database language is the SQL Relational database language which represents a combination of DDL, VDL and DML as well as a statement for constraint specification, schema evolutions and other features.

The most common operation in SQL is the query, which is performed with declarative SELECT statement. SELECT retrieves data from one or more tables.

```
SELECT First_Name,Last_Name

FROM Employee

WHERE Car_Number IS NOT NULL;
```

SELECT in a statement is used to retrieve information from a table. FROM clause gives the table from which the column will select. WHERE clause in a SELECT statement provides the criteria for selecting values.

Database Connectivity:

A database connection is a facility in computer science that allows Client software to communicate with database server software, whether on the same machine or not. A connection is required to send commands and receive answers.

Connections are built by supplying an underlying driver or provider with a connection string, which is a way of addressing specific database or server and instance as well as user authentication credentials. Once a connection has been built it can be opened and closed at will, and properties(such as command time-out length, or transaction, if one exists) can be set. The Connection String is composed of a set of key/value pairs as dictated by data access interface and data provider being used.

JDBC Drivers and its types:

The JDBC API defines the Java interfaces and classes that programmers use to connect to databases and send queries. A JDBC driver implements these interfaces and classes for a particular DBMS vendor.

A Java program that uses the JDBC API loads the specified driver for a particular DBMS before it actually connects to a database.

The JDBC DriverManager class then sends all JDBC API calls to the loaded driver.

The four types of JDBC drivers are :

- **JDBC-ODBC bridge plus ODBC driver, also called TYPE 1.**

  Translates the JDBC API calls into Microsoft open database connectivity(ODBC) calls that are then passed to the ODBC driver. The ODBC binary code must be loaded on every client computer that uses this type of driver.

- **Native-API, partly Java driver, also called TYPE 2**

  Converts JDBC API calls into DBMS-specific client API calls. Like the bridge driver, this type of driver requires that some binary code to be loaded on each client computer.

- **JDBC-Net, pure Java driver, also called TYPE 3**

  Sends JDBC API calls to a middle-tier net server that translates the calls into the DBMS-specific network protocol. The translated calls are then sent to particular DBMS.

- **Native-protocol, pure Java driver, also called TYPE 4**

  Converts JDBC API calls directly into the DBMS-specific n protocol without a middle-tier. This allows the client applications to connect directly to the database server.

Java Database Connectivity Steps:

Before you can create a Java JDBC connection to the database, you must first import the java.sql package.

importjava.sql;

The star (*) indicates that all of the classes in the package java.sql are to be imported.

1. **Loading a database driver:**

   In this step of the JDBC connection process, we load the driver class by calling Class.forName() with the Driver class name as an argument. Once loaded, the Driver class creates an instance of itself. A client can connect to Database Server through JDBC Driver. Since most of the Database servers support ODBC driver therefore JDBC-ODBC Bridge driver is commonly used.

   The return type of the Class.forName(String ClassName) method is “Class”.Classd us a class in java.lang package.

```
try{
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
//Or any other driver
}

Catch(Exception x){
System.out.println("Unable to load the driver class.");
}
```

2. **Creating an Oracle JDBC Connection:**

   JDBC URL Syntax::jdbc:<subprotocol>:<subname>
   JDBC URL Example::jdbc:<subprotocol>:<subname>

- Each driver has its own subprotocol
- Each subprotocol has its own syntax for the source. We're using the jdbcodbcsubprotocol, so the DriverManager knows to use the sun.jdbc.odbc.JdbcOdbcDriver.

```
try{
Connection dbConnection=DriverManager.
getConnection(url,"loginName','Password')
}
Catch(SQLException x)
{
System.out.println("Couldn't get connection!");
}
```

3. **Create a JDBC Statement object:**

   Once a connection is obtained we can interact with the database. Connection interface defines methods for interacting with the database via the established connection. To execute SQL statements, you need to instantiate a Statement object from you connection object by using the createStatement() method.

   Statement statement=dbConnection.createStatement();

   A statement object is used to send and execute SQL statements to database. Three kinds of statements :

   Statement : Execute simple SQL queries without parameters. Statement createStatement() creates an SQL statement object.

   Prepared Statement : Execute precompiled SQL with or without parameters PreparedStatement prepare statement objects are precompiled SQL statements.

Callable Statement : Execute a call to a database stored procedure CallableStatementprepareCall(String sql) returns a new CallableStatement object CallableStatement objects are SQL stored procedure call statements.

4. **Executing a SQL statement with the Statement object and returning a jdbcresultset:**

   Statement interface defines methods that are used to interact with database via the execution of SQL statements. The statement class has three methods for executing statements.

   executeQuery(), executeUpdate() and execute().

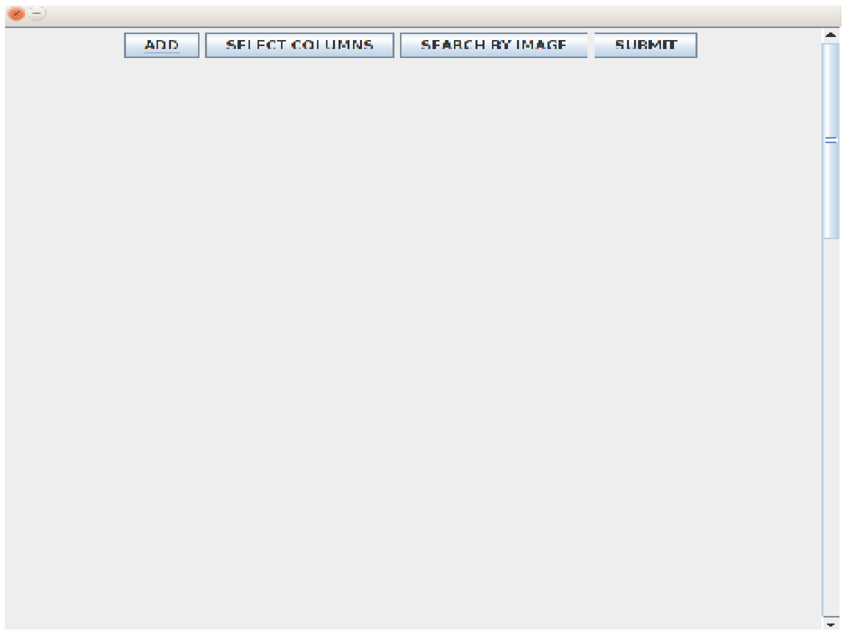
   For a SELECT statement, the method to use this executeQuery(). For statement that create or modify tables, the methods to use is executeUpdate().

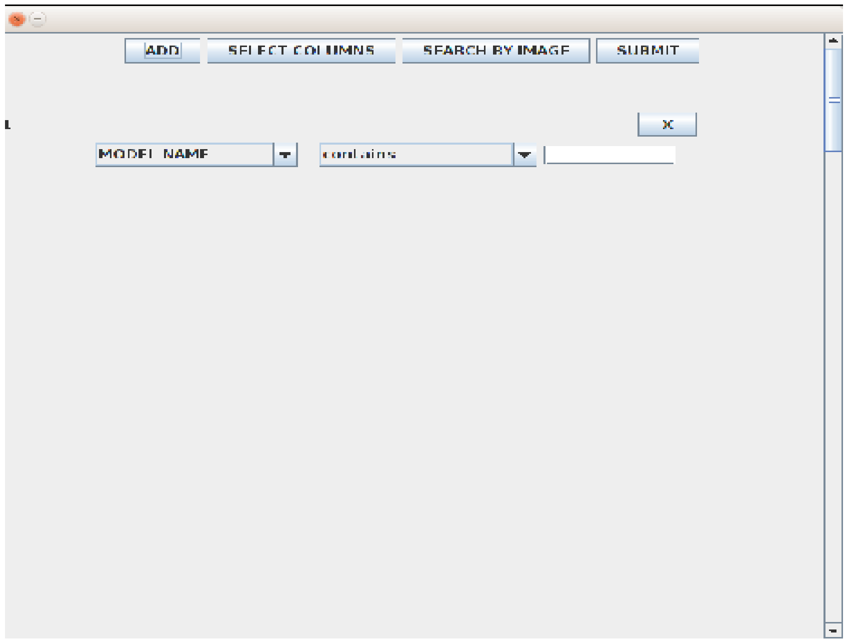
   Note: Statements that create a table, alter a table or drop a table are all examples of DDL statements and are executed with the method executeUpdate(). execute() executes an SQL statement that is written as a String object.

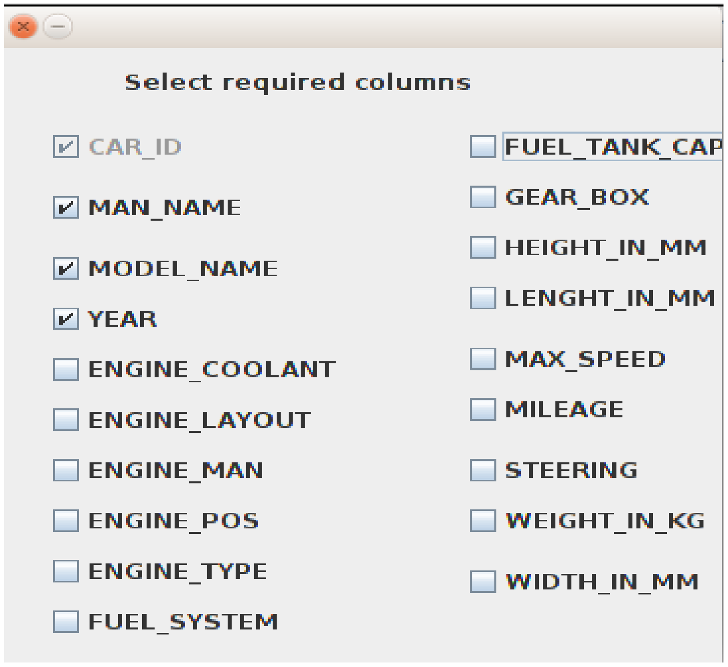
   Resultset provide access to the table or data generated by executing a statement. The table rows are retrieved in sequence. A Resultset maintain a cursor pointing to its current row of data. The next() method is used to successively step through the rows of the tabular results.

ResultsetMetaData interface holds information on the types and properties of the columns in a Resultset. It is constructed from the connection object.

SCREENSHOTS

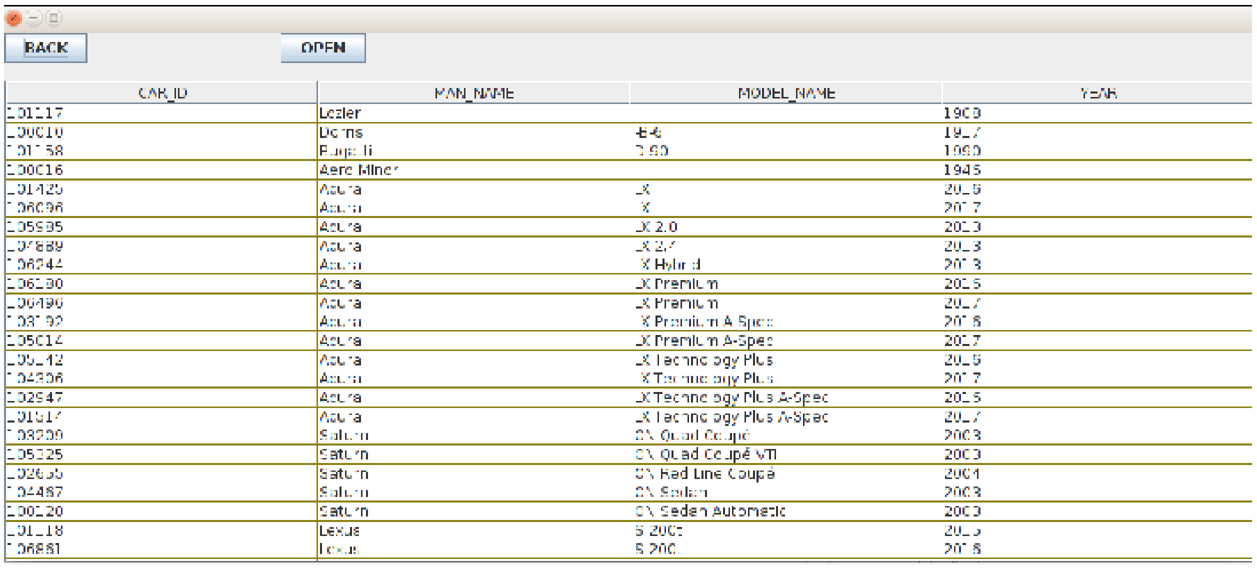

ADD
SELECT COLUMNS
SEARCH BY IMAGE
SUBMIT


ADD
SELECT COLUMNS
SEARCH BY IMAGE
SUBMIT
X
MODEL_NAME
contains

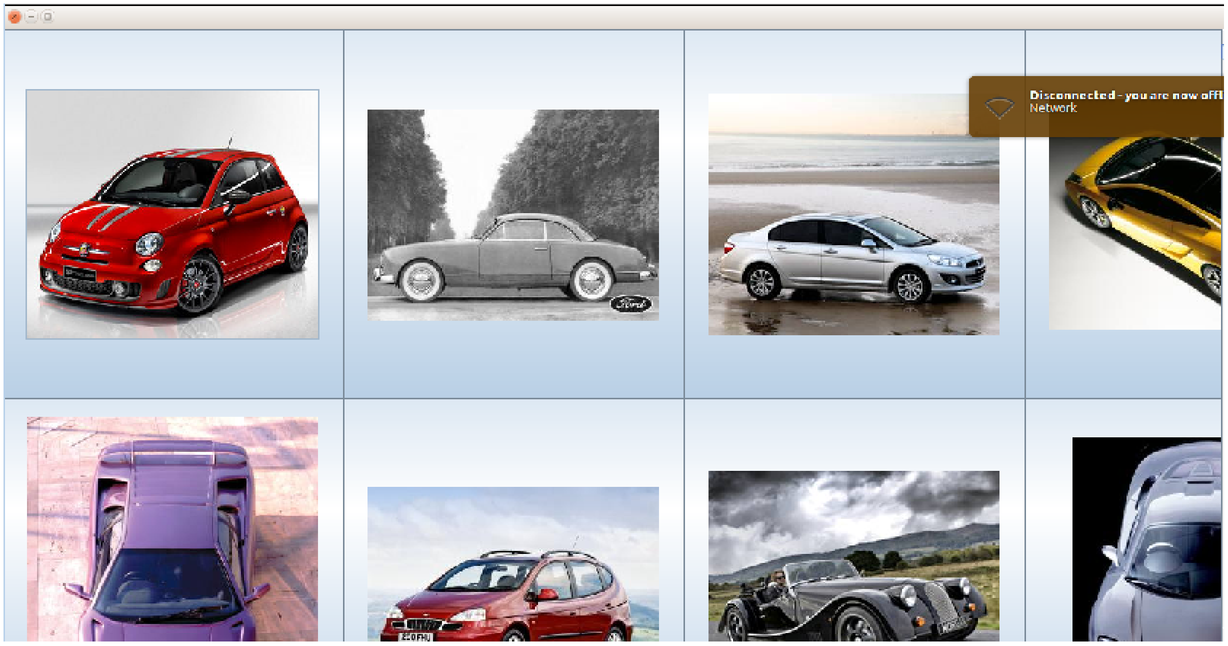


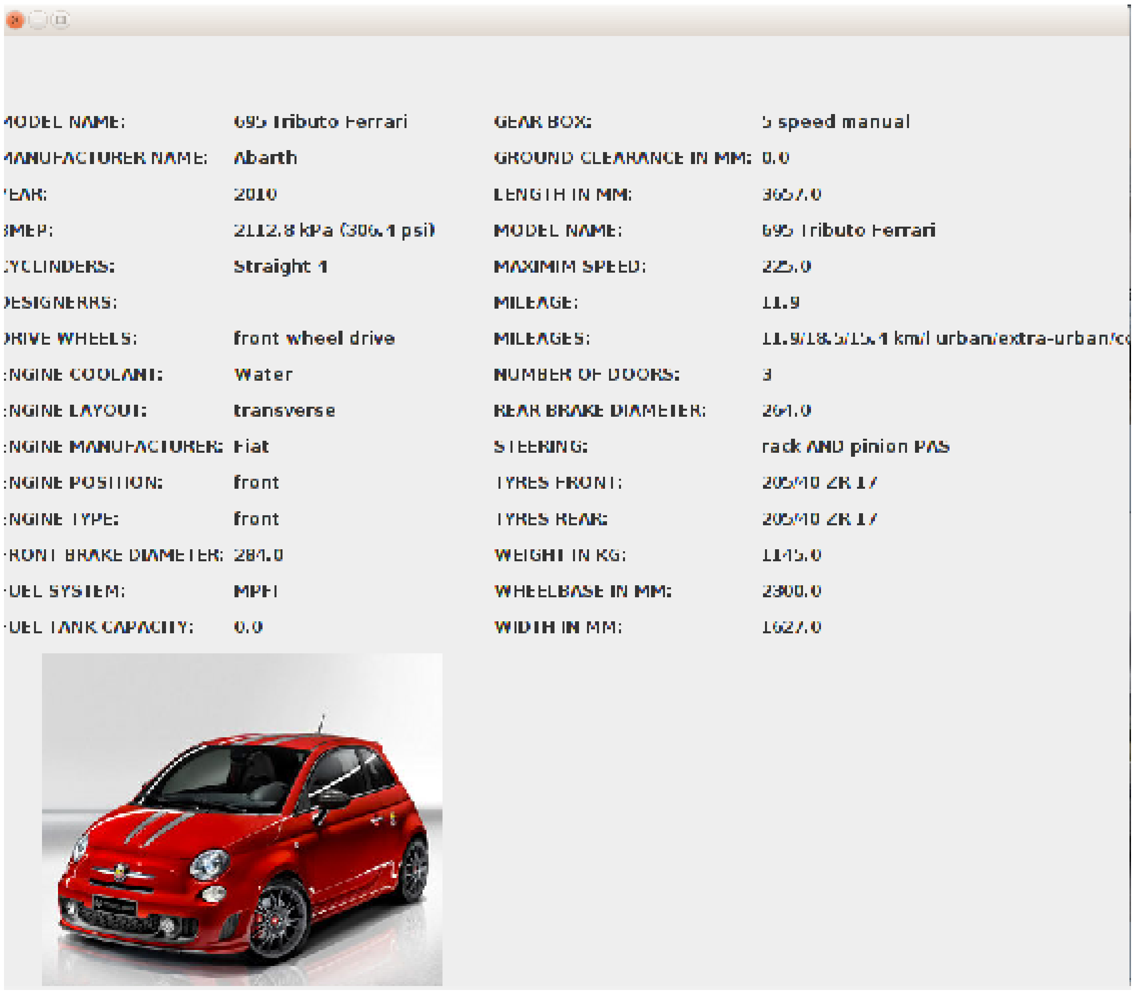
Select required columns

- [x] CAR_ID
- [x] MAN_NAME
- [x] MODEL_NAME
- [x] YEAR
- [ ] ENGINE_COOLANT
- [ ] ENGINE_LAYOUT
- [ ] ENGINE_MAN
- [ ] ENGINE_POS
- [ ] ENGINE_TYPE
- [ ] FUEL_SYSTEM
- [ ] FUEL_TANK_CAP
- [ ] GEAR_BOX
- [ ] HEIGHT_IN_MM
- [ ] LENGHT_IN_MM
- [ ] MAX_SPEED
- [ ] MILEAGE
- [ ] STEERING
- [ ] WEIGHT_IN_KG
- [ ] WIDTH_IN_MM



CAR_ID	MAN_NAME	MODEL_NAME	YEAR
101117	Lozier		1909
100010	Dorris	E-6	1917
101158	Bugatti	[illegible] 90	1990
100016	Aero Minor		1945
101425	Acura	ILX	2016
106098	Acura	ILX	2017
105995	Acura	ILX 2.0	2013
107689	Acura	ILX 2.4	2013
106244	Acura	ILX Hybrid	2013
106190	Acura	ILX Premium	2016
106490	Acura	ILX Premium	2017
103192	Acura	ILX Premium A-Spec	2018
105014	Acura	ILX Premium A-Spec	2017
105142	Acura	ILX Technology Plus	2016
104306	Acura	ILX Technology Plus	2017
102947	Acura	ILX Technology Plus A-Spec	2016
101917	Acura	ILX Technology Plus A-Spec	2017
103200	Saturn	ION Quad Coupé	2003
105325	Saturn	ION Quad Coupé VTi	2003
102655	Saturn	ION Red Line Coupé	2004
104487	Saturn	ION Sedan	2003
100120	Saturn	ION Sedan Automatic	2003
101118	Lexus	IS 200t	2016
106881	Lexus	IS 200t	2018





ODEL NAME:	695 Tributo Ferrari	GEAR BOX:	5 speed manual
ANUFACTURER NAME:	Abarth	GROUND CLEARANCE IN MM:	0.0
EAR:	2010	LENGTH IN MM:	3657.0
MEP:	2112.8 kPa (306.4 psi)	MODEL NAME:	695 Tributo Ferrari
YLINDERS:	Straight 4	MAXIMIM SPEED:	225.0
ESIGNERRS:		MILEAGE:	11.9
RIVE WHEELS:	front wheel drive	MILEAGES:	11.9/18.5/15.4 km/l urban/extra-urban/c
NGINE COOLANT:	Water	NUMBER OF DOORS:	3
NGINE LAYOUT:	transverse	REAR BRAKE DIAMETER:	264.0
NGINE MANUFACTURER:	Fiat	STEERING:	rack AND pinion PAS
NGINE POSITION:	front	TYRES FRONT:	205/40 ZR 17
NGINE TYPE:	front	TYRES REAR:	205/40 ZR 17
RONT BRAKE DIAMETER:	284.0	WEIGHT IN KG:	1145.0
UEL SYSTEM:	MPFI	WHEELBASE IN MM:	2300.0
UEL TANK CAPACITY:	0.0	WIDTH IN MM:	1627.0

CONCLUSION

We have tried to develop a simple outline of car searching system. There are very many features that can be added to the system. The database memory is limited, so that only a few records compare to the real world can be added here. Everything implemented has been made friendly as possible for the user. Searching of records are made easy. Option are left open, so that if there are if there are any future enhancement requires, it can possibly added to the system. We hope that this project will serve as the base for car searching system and hence fulfill the purpose upon which it was developed.

REFERENCES

- Schildt H., *Java: The Complete Reference*, 8/e, Tata McGraw Hill, 2011.