POLITECNICO DI MILANO
Facoltà di Ingegneria dell'Informazione

POLO REGIONALE DI COMO

Master of Science in
Computer Engineering
(For the Communication)

Web Interface to data streaming of digital verification Object

Supervisor: Prof. Sara Comai

Master Graduation Thesis by: Md. Fakhrul Islam
Student Id. number 722621

Academic Year 2009/2010

POLITECNICO DI MILANO
Facoltà di Ingegneria dell'Informazione

POLO REGIONALE DI COMO

Corso di Laurea Specialistica in
Ingegneria Informatica

Web Interface to data streaming of digital verification Object

Supervisor: Prof. Sara Comai

Tesina di laurea di: Md. Fakhrul Islam
Matricola: 722621

Academic Year 2009/2010

I dedicate this work
To my parents
To my brother and sister
And to all of my friends
………the never missing support to my studies

Abstract

The web interface to digital verification objects with database aims to facilitate user to analyze verification object and also to allow making some other manipulations such as creating new packets, modifying the data of the packets in the database. It also allows representing objects from a database to a web interface in a human readable format. Users, for example verification engineer, control and modify stored objects from the verification environment in a simple way using a web browser.

Users can access this application from anywhere using simply a browser. Using this application, users can perform tasks such as creating new packet, modifying packets, creating packet with different ranges of random values, and displaying all packets with their elements of the fields. In order to make user interactions such as taking requests and responding accordingly for performing the tasks, MVC framework has been employed. To accomplish the user request, I used some java technologies and Tomcat application server in action. When users request for an action, the application performs the user request with some of java tools such as Servlet, Jsp etc. Then, it communicates with the database to retrieve or store data and then send back the appropriate result to the user in web browser.

Contents

Acknowledgement

I would like thank all people who were helped and inspired me during this project study. It could not have been possible without the kind assistance of some people to whom I would like to express my gratitude.

I especially want to thank my advisor, Prof. Sara Comai, for his guidance start to end of the project. Her perpetual energy and enthusiasm in research had motivated all his advisees, including me. In addition, she was always accessible and willing to help me in this project. As a result, I have finished my thesis successfully.

Special thanks to Alberto Allara, CCI Datastorage Division, STMicroelectronics spa, for good advice and suggestions. He also helped to get some real-time data packets and contributed valuable points of view and suggestions regularly. Thanking you very much.

My deepest gratitude goes to my family for their unflagging love and support throughout my life; this dissertation is simply impossible without them.

I would like to thanks all of my friends at Politecnico di Milano for their support in validating and discussing the ideas presented in this work: Khondoker Zahid Hossain, Jalal Uddin Ahmed. Thanks to all.

Last but not least, thanks to God for my life through all tests in the past two years. You have made my life more bountiful. May your name be exalted, honored, and glorified.

July 2010

Md. Fakhrul Islam

1 Introduction

This thesis project is proposed by “STMicroelectronics s.p.a” Data storage Division with the Title of “Web interface to data streaming of digital verification objects”. The crux of the implementation of this project is to build a web interface to visualize and to manipulate digital verification objects from Relational database.

Nowadays, different modern technologies and components of verification environment for testing devices exchange information in term of transactions. Verification engineer needs to analyze those transaction packets or meta-information of the packets to anticipate about the performed tests.

At present, STMicroelectronics Company is using OVM (Open Verification Methodology), which is the first open, interoperable, and SystemVerilog verification methodology in the industry, for verification and testing digital devices. With that OVM verification environment, they added a concept of open relation database. The integration of database with the verification environment satisfies interoperability with any simulator and reusing stored data from database.

Now STMicroelectronics propose to build a simple web interface that will allow the user to manipulate or to control the stored packets of the transactions by using their Meta information from the Mysql database. User now can access database packets by a simple web browser, and then he /she can create new packets, Modify/Update packet on database and also represent all packets with all fields of those packets in readable format to the user browser.

To test this project implementation, I used some sample verification transaction packet in database and it has executed perfectly. Some java J2EE technologies (Servlet, Jsp), JavaScript, Struts Framework, Tomcat Application server and other essential tools used to develop this web application.
The overviews of the remaining chapter of this project are followings:

Chapter-2 presents the high level of development technologies those are used for the project implementation and also those are the related to developed business logic of this application.

Chapter-3 describes a short summary or overall description of the whole system that implemented.

Chapter-4 represents a whole system design structure by block diagram. It also represents how user performs the allowed tasks and execution sequences by using some sequence diagrams.

Chapter-5 presents requirements of project implementations and a details description of the whole implementation process. The real time execution of the project is also shown by some snapshots or image.

Finally, the conclusion shows the final results and future works of the thesis in terms of benefits of the organization, Research action will focus on the ongoing enhancement of the features of the approach.

2 State of the Art

This section presents the level of new technological aspects or software/hardware tools those are frequently used for the implementation of this project. Some of advanced tools have been used to enhancement or achievement of perfect throughput of this project. Some of them are discussed shortly in the rest of parts of this section.

2.1 Framework

A framework is a collection of classes and applications, libraries of SDKs and APIs to help the different components all work together .In the development of this web application, I utilize a powerful framework which apply to build any complex web application. The whole application is developed under Struts 2.0 framework.

2.1.1 Struts 2.0

The Jakarta Struts project, an open source project sponsored by the Apache Software Foundation, is a server side Java implementation of the Model–View–Controller (MVC) design pattern. The Struts project was originally created by Craig McClanahan in May 2000, but since that time it has been taken over by the open source community. The Struts project was designed with the intention of providing an open source framework for creating Web applications that easily separate the presentation layer and allow it to be abstracted from the transaction/data layers. Since its inception, Struts has received quite a bit of developer support, and is quickly becoming a dominant factor in the open source community.

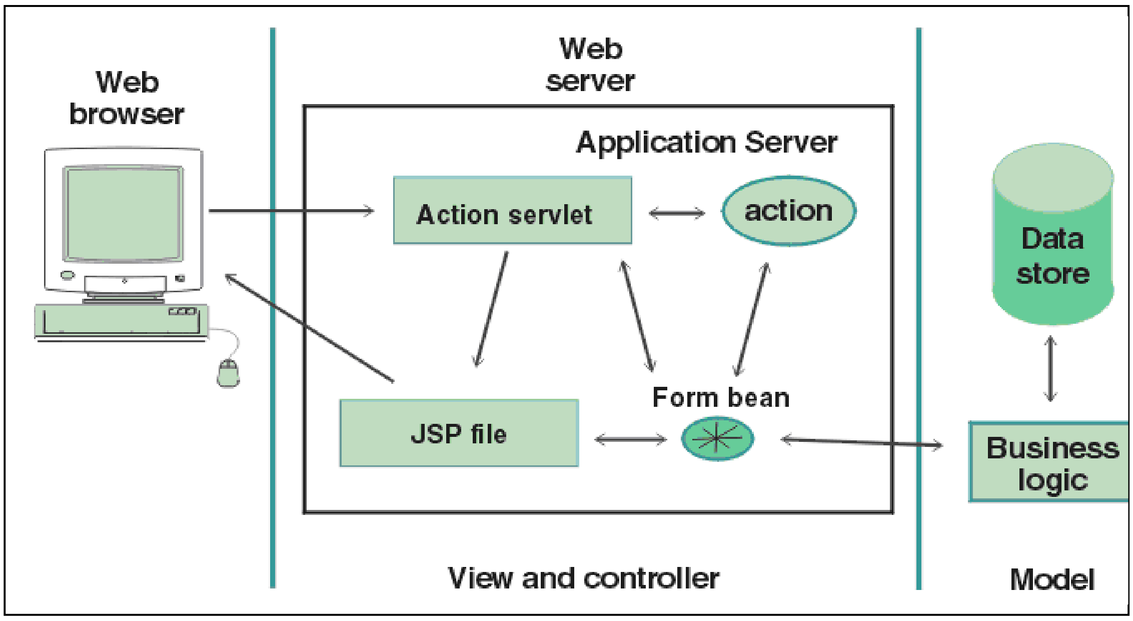


Figure-1: Development framework to build web interface with database

2.1.1.1 Model

A model represents an application's data and contains the logic for accessing and manipulating that data. Any data that is part of the persistent state of the application should reside in the model objects. The business objects update the application state. ActionForm bean represents the Model state at a session or request level, and not at a persistent level. Model services are accessed by the controller for either querying or effecting a change in the model state. The model notifies the view when a state change occurs in the model. The JSP file reads information from the ActionForm bean using JSP tags.

2.1.1.2 Controller

The Struts action Servlet handles runtime events in accordance with a set of rules that are provided at deployment time. Those rules are contained in a Struts configuration file and specify how the Servlet responds to every outcome received from the business logic. Changes to the flow of control require changes only to the configuration fie.

Struts also provide the Java class org.apache.struts.action.Action, which a java developer subclasses to create “Action class”. At run time, the action servlet is said to “execute actions,” which means that the servlet invokes the execute method of each of the instantiated action classes. The object returned from the execute method directs the action Servlet as to what action or JSP file to access next.

To facilitate reuse, invoke business logic from the action class rather than including business logic in that class.

2.1.1.3 View

Java class org.apache.struts.action.ActionForm, which uses subclass to create a form bean that, is use in two ways at run time:

i. When a JSP page prepares the related HTML form for display, the JSP page accesses the bean, which hold values to be placed into the form. Those values are provided from business logic or from previous user input.

ii. When user input is returned from a Web browser, the bean validates and hold that input either for use by business logic or for subsequent redisplay.

2.1.2 Benefits of using Struts

The application of the model-view-controller division to the development of dynamic Web applications has several benefits:

i. A developer can distribute development effort to some extent, so that implementation changes in one part of the Web application do not require changes to another. The developer responsible for the flow of control, and Web-page designers can work independently of the developers.

ii. Developer can more easily prototype his work. It can do as follows, for instance:

   a. By creating a prototype Web application that accesses several workstation-based programs

   b. Change the application in response to user feedback.

   c. By implementing the production-level programs on the same or other platforms.

iii. A developer can maintain an environment that comprises different technologies across different locations.

iv. The MVC design has an organizational structure that better supports scalability(building bigger applications) and ease of modification and maintenance(due to the cleaner separation of tasks)

2.2 Verification

The transactional data those are collected from the Verification Environment are stored in Mysql database. Verification is the process used to demonstrate the functional correctness of a design prior to its fabrication. Typically, a high-level description of the design is simulated using multiple test stimulus patterns over an extended period of time. The specific test cases used to verify the various features (configurable options) in the design are described in a document called the “test plan.” Two categories of verification components are required to verify the functionality of hardware devices that conform to the standard: those that are not directly related to the standard but are useful for the creation and checking of data “packets,” and those that are required to emulate the behavior of hardware devices. The first category includes components such as packet data generators, packet drivers, FIFOs (First-In-First-Out) and payload checkers. The second category includes components such as Ethernet framers, Ethernet defamers, CRC (Cyclic Redundancy Check) generators, payload scramblers and descramblers. The components are designed to be modular and parameterizable so that they maybe reused in applications even outside the IEEE Ten Gigabit Ethernet standard. For example, the “raw packet generator” implemented in this work is protocol-free in the sense that the data packets generated do not conform to any protocol such as Ethernet or HDLC. As such, it can be used in any environment where a data source is required.

2.3 Open Verification Methodology (OVM)

The Open Verification Methodology (OVM), a joint development initiative between Mentor Graphics® and Cadence® Design Systems, provides the first open, interoperable, SystemVerilog verification methodology in the industry. The OVM provides a library of base classes that allow users to create modular, reusable verification environments in which components talk to each other via standard transaction-level modeling (TLM) interfaces. It also enables intra- and inter-company reuse through a common methodology with classes for developing stimulus sequences and block-to-system reuse. Supported on multiple verification platforms, the OVM is the de-facto standard methodology, ideally suited to speed verification for both novice and expert verification engineers. Built on the success of the Advanced Verification Methodology (AVM) from Mentor Graphics and the Universal Reuse Methodology (URM) from Cadence, the OVM brings the combined power of these two leading companies together to deliver on the promise of System Verilog. The OVM offers established interoperability mechanisms for verification IP (VIP), transaction-level and RTL models, and integration with other languages commonly used in production flows.

2.3.1 Why it is called Open Methodology?

In response, Cadence and Mentor Graphics developed a common methodology and a class library that runs on simulators from both companies. Announced on August 16, 2007, the OVM spans the class library and methodology layers of the SystemVerilog verification hierarchy, as shown in Figure-2.

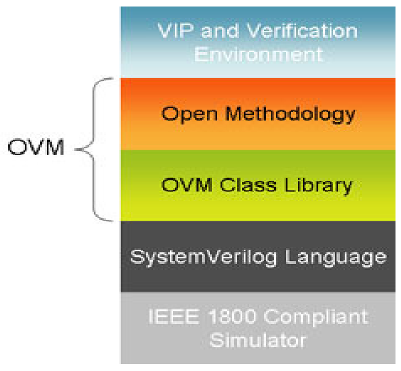

VIP and Verification Environment
OVM
Open Methodology
OVM Class Library
SystemVerilog Language
IEEE 1800 Compliant Simulator

Figure-2 Verification Hierarchy

The class library is supported by both the Mentor Graphics Questa® and Cadence Incisive® verification platforms. Thus, any VIP or test benches built using this library will run on either platform with no conversions, translations, or extra effort required. In fact, since the OVM is delivered in open-source format, the code will run on any simulator that supports the SystemVerilog standard.

2.3.2 Ovm special features

i. Advanced verifications capabilities.

ii. Higher Level of IP integrations

iii. Robust class library

iv. Supporting multiple language and simulator.

2.3.3 Why methodology

Here are the few reasons to use a standard methodology in designing a verification component. It provides consistency and informality in the way verification components are developed, used and connected. So a new engineer joins the team and he knows the methodology the ramp up time will be reduced significantly. Code reuse - more you reuse less will be the bugs Use the best know and proven practices.

2.3.4 Benefits of Using OVM Methodology:

i. Improves productivity

ii. Increases and ensures reusability

iii. Maintenance of the verification components will be much easier because the components are standardized.

iv. Independence between components. A test case writer does not need to understand the full flow of the architecture.

2.3.5 Description of OVM components:

2.3.5.1 Driver

Driver's role is to drive the transaction into the DUT interface. The driver obtains the data item from the sequencer for the execution. So all the drivers are extending from the ovm_driver class either directly or indirectly. The driver has run method which specifies its task and TLM port for communication with the sequencer.

2.3.5.2 Sequencer

Sequencer generates stimulus data and passes it to the driver. All sequencers should be derived from the ovm_sequence base class directly or indirectly. Sequencers have a TLM port which is connected to the driver. Ovm_sequence base call is parameterized for request and response item types.

2.3.5.3 Monitor

The monitor is responsible for extracting signal information from the bus and translating it into events, structs, and status information. This information is made available to other components using the TLM interface. Monitors should be configurable mean they can be enabled or disabled. The monitor collects bus information through a virtual interface. The collected data is used in coverage collection and checking. The collected data is exported on an analysis port (item_collected_port).

2.3.5.4 Sequences

A sequence is a set of transactions that accomplish a defined complex task for the DUT.
In OVM test cases are typically described at the transaction level which makes them easier to maintain and write for example like we have a packet and instructions etc. And in many of the scenarios a single transaction can not capture the high level intention of the design so generally a ordered stream of transactions is required to accomplish a task.
In OVM some sequences are provided as part of a reusable component.

2.3.5.5 OVM Test

In the ovm mythology all tests are basically classes. And all test classes are extended by ovm_test class. An OVM test has following properties:

i. Unique to OVM Derived from ovm_test base class

ii. Tests are top level components that create the dynamic test bench

iii. They have all simulation phases start_of_simulation(), run() are used to specify the run time parameters of the environment

iv. Using OVM test cases multiple tests are compiled together and a test case can be chosen at the run time using command line option. The constraint random test cases in the ovm are much shorter than the directed test cases. With ovm_test class we don't need to recompile the environment again to run a new test case which saves huge time for big verification environments.

2.3.5.6 Data Items

Data items Represent the main transaction input to the DUT (Device Under Test) Adhering to a protocol, consistent values are generated and sent Examples include packets, transactions, instructions, and so on Test environment randomizes data items (transactions) A default distribution should be generated Tests further steer generation by layering constraints or selecting from pre-defined scenarios This makes test short, readable, easier to write and maintain.

2.3.5.7 Scoreboard

The task of using a scoreboard can be described as storing and then comparing data. The typical use model is as follows:

i. Store the data items sent to the DUT (input).

ii. For every data item output by DUT, check whether the data it contains is correct, based on the stored data.

Although all scoreboards are not identical, substantial framework and facilities can be shared across all scoreboards. The OVM Scoreboard provides such a scoreboard framework, applicable to many kinds of systems and checking requirements. The OVM scoreboard is open source, part of the OVM utilities library. The OVM scoreboard provides:

i. Easy usage for simple point-to-point devices.

ii. Flexible scoreboard services that is applicable for the majority of devices, including complex devices.

iii. Built-in infrastructure for system events (reset, power-down, and so on)

With the OVM scoreboard, you can define a scoreboard unit, connect it to the environment monitors, and customize its matching criteria and checking. You can also use the wide set of built-in queries forgetting information on data items stored in the scoreboard. Using the OVM Scoreboard as the framework for all your scoreboard implementations also enhances reuse and interoperability in your verification environment.

3 Overview of the system

This section presents the summary or overall view of the whole system. In the recent days advanced verification environment setup for testing of digital devices. Generally this verification environment based on some technologies those are consist of several components, interchanging information as a transaction that is consists of data and control or Meta information.

A transaction can represent a bus transaction or a packet transaction or an entire test. Using transactions meant that you needed to consider many aspects of the transaction. Among the considerations – what is being modeled, when is it valid – when does it begin and end, what is contained in the transaction and how this transaction is related to other transactions. Transaction represents packets both from that environment provide to the DUT (Device Under Test) and also response from the DUT (Digital Under Test).

These transaction's data are not a few amount of data instead it contains a huge number of data and information for the verification engineer. Verification engineer utilize or analyze those data to measure of the quality of the simulation and also may previse the coverage status of the created tests.

There already exist some EDA tools use to represent transaction data graphically to facilitate the debug of the device and filter and aggregate them in appropriate functional coverage. But EDA tool uses proprietary data formats for storing the data. So it is not possible to exchange data across different simulators. The integration of an external open database with a verification environment so would satisfy some important issue such as interoperability, transparency, reuse etc. The relational database can be accessed through a set of DPI functions representing the SystemVerilog of the set of procedural interfaces in C provided by the database.

In fact, the main objective of this project is to develop a web interface to streaming object which are persisted in a relational database. It allows user to analyze, create new packet and other testing information from the database. Verification transaction packets and their control information and also the Meta information of the test environment are stored in the database.

The web interface provides an efficient and a simple web interface so that user can easily manipulate or access data from database. Transaction data are stored a specific format that BLOB (binary format).User can easily access database via internet by using a web browser. When user send a request of operation to the server, Application server handle

These request and execute that request with handling some business logic and different java classes and Servlet. After collecting or saving data in database Servlet send response to the user which is running in an application server.

This web interface allow user to some task which are as followings:

i. Create new packet in the database

ii. Modify/Change packet and save in the database

iii. Create new packet with random value using different ranges.

iv. User can see all the packet information with a human readable format.

4 Design

4.1 System architecture

We divide the whole system in a two logical partition one is digital verification environment and second one is web interface to open database where all transactions packet and meta-information is stored. Here user is the verification engineer who will analyze and interpret with via database with verification environment.

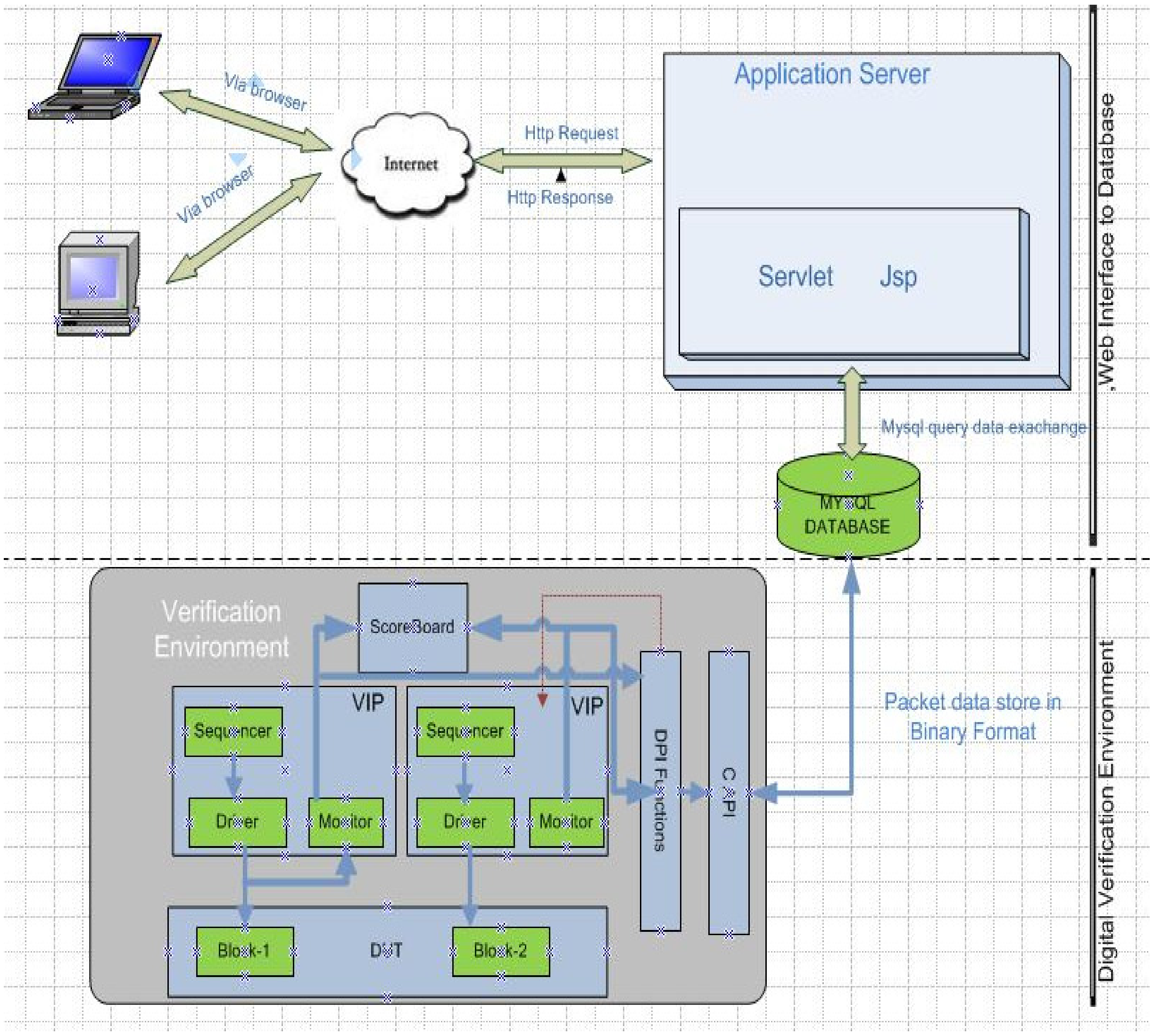


Figure-3: Design Structure of whole System

Transactional data and meta-information are collected from verification environment and then DPI function and C API, object packets are store in a Mysql database. User now can use that's stored packets data by a simple graphical user interface. Using a common web browser it accesses the home page and it sends an http request to the application server. After receiving request from user, Application server handle these request by Servlet and jsp. Then it communicates with the database for any query of search data, update and stored data in the database table. Finally Application server send back the response to user web browser and represent data in human readable format.

4.1.1 Entity relationship diagram and their relationships

A simple Entity relationship diagram (ERD) has depicted to design and represents the relationships among different entities in our system database. Total eight tables have been used to store transactions packets and Meta information's of verification environment. There are some relationships of each table with other tables by referencing primary key, foreign key etc. Now a short specification of those schemas is describing bellow:

- DS_PACKET: This table contains only two columns one is primary called Id and another one is DsBlob. Id is an atomic integer number which is used to search a packet .DsBlob field used to store the packet value.

- DS_FIELD_DESC: It consist of near seven columns .Among them three columns are primary key (Type_pack, Id, ID_ord_field). All these fields or columns used to store the control information's of a packet.For instance, FieldName is storing the each field name of the packet, and FieldLength defines the length of a field value. The ID_ord_field store sequence of number increasing order of a packet fields. These tables information's are use to show packets fields and also for every operation on packet value.

- TEST: test table contains two columns which define test id and test name of each test.

- TYPE_PACK: This schema is consists of single columns that are Type_pack.This field information is also used in TRANSACTION and DS_FIELD_DESC tables. This column information is important to find the specific type of packet from a large number of packets.

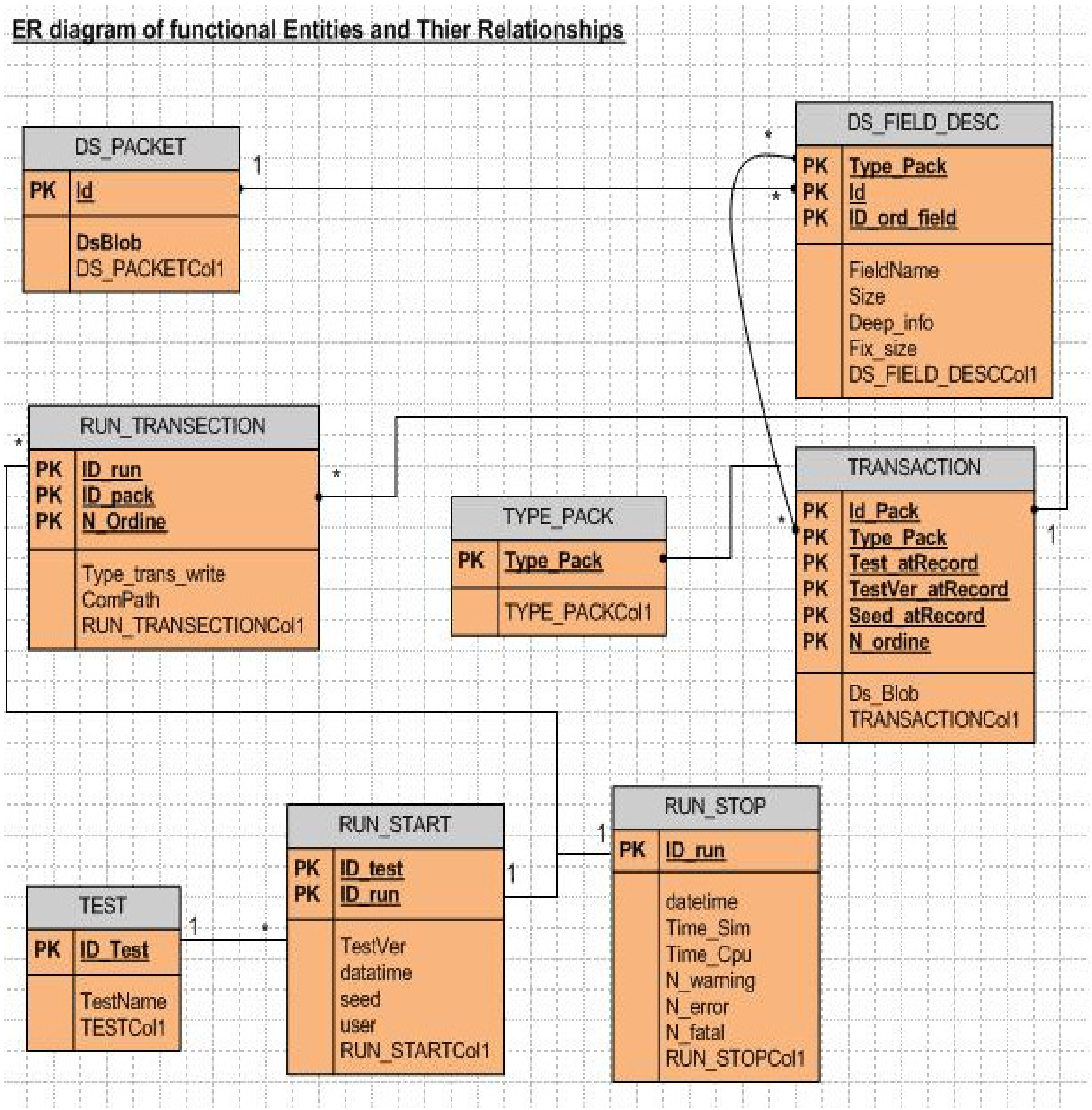


Figure-9: An ER diagram with relationship among entities

- RUN_START: This table used for collecting and providing test starting information such as time & date when it start, seed, user etc.

- RUN_STOP: It stored fields with time when test stops, sim time, cpu time,error, warning of transaction failure etc.
- TRANSACTION: Transaction table consists of six primary key fields and DsBlob field which has relationships with DsBlob field.

4.1.2 Web Interface to Database

This section describes the user interaction by using a simple DFD (data flow diagram) with the system in different phases in the application.

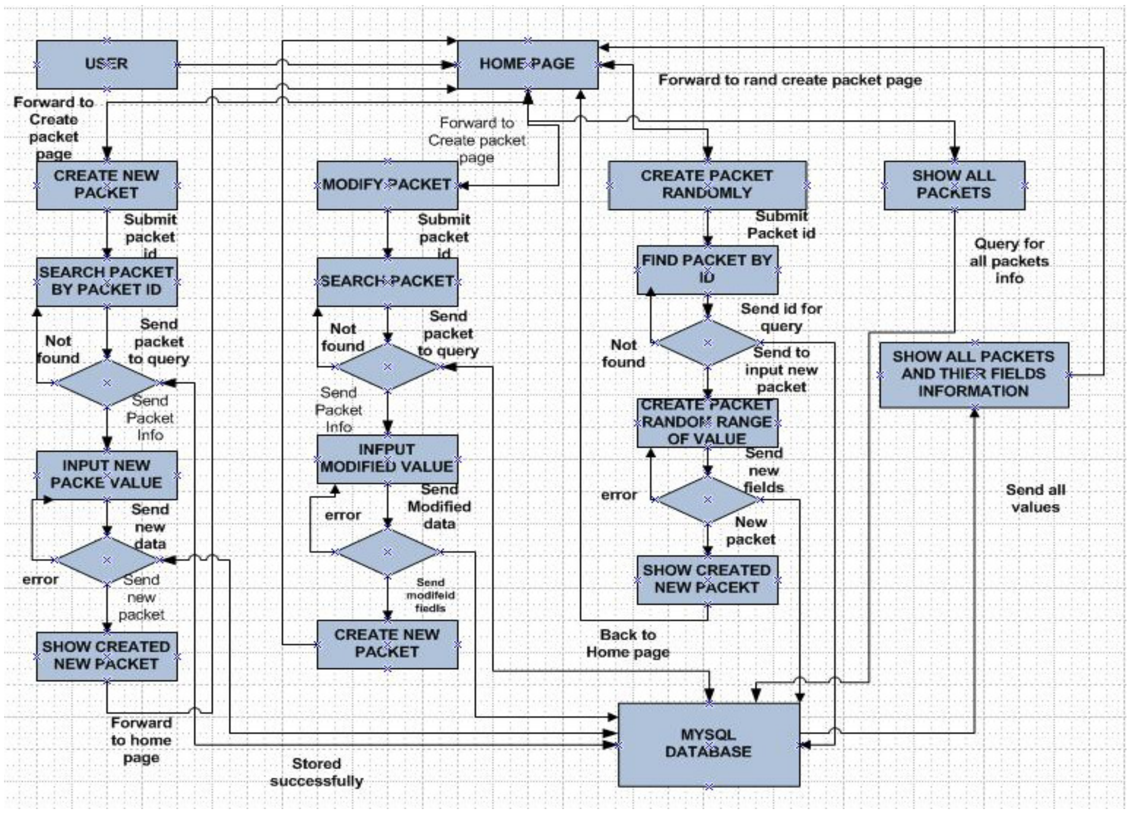


Figure-4: Dataflow diagram of web interface with database

User can interact with using a simple web browser by accessing the home page of the application. In fact, at this time user can perform four basic tasks here those are create new blob, modify/update packet, create packet with randomly and final task is to show all packets from database. After finishing each task, user can back to the home page and check the new or modified packet from database.

4.1.2.1 Create New Packet

This sequence diagram represents sequence of objects interaction when user is going to create new packet in the database. Here is a short description in step by step of all objects which are participating in this execution task.

i. At first user request to forward for performing “Create New Packet” in newblob. Jsp page that has a link from Home page.

ii. In this page user have to input request packet id and then submit it.

iii. ForwardToNewFormAction is a action class which job is to collect this packet id from user and search a query from database table by using user requested packet id.

iv. After getting requested packet fields and their meta-information, this action class accomplishes to back all data to newblob.jsp page.

v. Now user can create new packet using existing field of packet fields and submit it to the server.

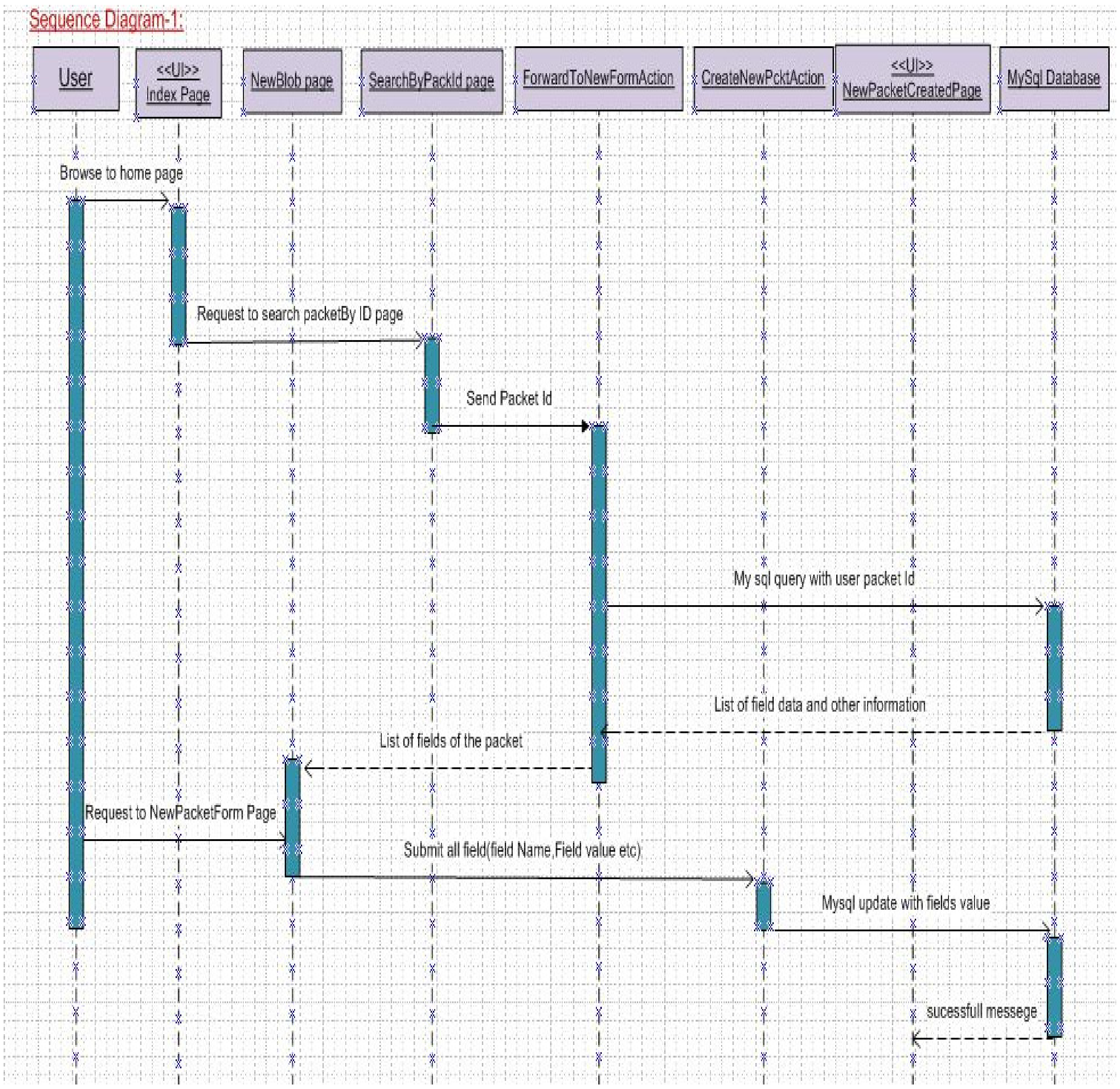

Sequence Diagram-1:
User
<<UI>> Index Page
NewBlob page
SearchByPackId page
ForwardToNewFormAction
CreateNewPcktAction
<<UI>> NewPacketCreatedPage
MySql Database
Browse to home page
Request to search packetBy ID page
Send Packet Id
My sql query with user packet Id
List of field data and other information
List of fields of the packet
Request to NewPacketForm Page
Submit all field(field Name,Field value etc)
Mysql update with fields value
sucessfull messege

Figure-5: Sequence diagram of interaction to create new Packet

vi. At this time, CreateNewPcktAction class collect all the new fields info and call a createPacket () to build new packet and send those fields information of new packet to database.

vii. After storing packet in database successfully, this class shows a successfully success message to the newpacketcreated.jsp page.

4.1.2.2 Modify Packet

In this simple sequence diagram presents sequence of system interactions and actions when user are going to modify packet from database. To describe this sequence diagram By using some steps those has stated bellow:

i. Users have to go through the home page and select link of "Modify Packet" page that is findById.jsp page.

ii. Here users have to enter packet id which packet will be modify. Then this packet id will be process by an action class which is ForwardToModifyForm class. After that it searches a query from database by using this user requested packet.

iii. If this packet does exist in the database, Action class will collect those lists of fields and field's information of that packet and return those lists of value to FrdModifyForm.jsp page.

iv. Now user can modify or change the packet value field and submit it to the server.

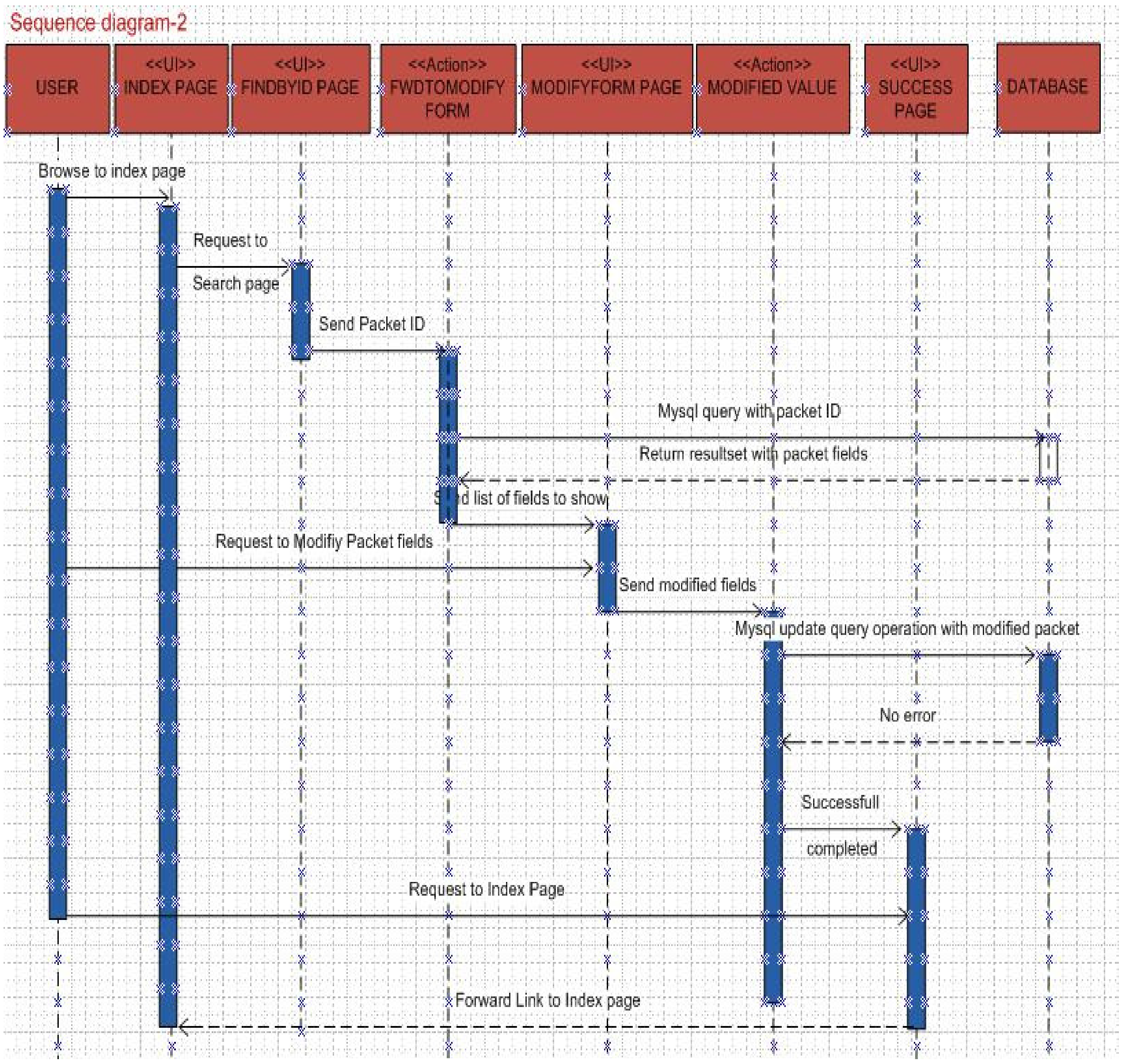

Sequence diagram-2
USER
<<UI>> INDEX PAGE
<<UI>> FINDBYID PAGE
<<Action>> FWDTOMODIFY FORM
<<UI>> MODIFYFORM PAGE
<<Action>> MODIFIED VALUE
<<UI>> SUCCESS PAGE
DATABASE
Browse to index page
Request to Search page
Send Packet ID
Mysql query with packet ID
Return resultset with packet fields
Send list of fields to show
Request to Modifiy Packet fields
Send modified fields
Mysql update query operation with modified packet
No error
Successfull completed
Request to Index Page
Forward Link to Index page

Figure-6: A sequence diagram to modify packet.

v. The ModifiedValue action class handles these form value and it call a function to create new packet with new modified fields. Then it executes an update query to store new packet data in database.

vi. Then the action class sends back new modified packet and a successfully stored data message to the user on the success.jsp page.

vii. Finally, user can go return to home page to see the entire packet with their field elements in readable format.

4.1.2.3 Create new packet with random value

In this sequence diagram represents sequence of object interaction when user are going to create new packet with the random value. In this part of the system allow a user to select some different range and system automatically create field value according to that input ranges. Here is a short description in step by step of all objects which are participating in this execution task.

i. At first user request to forward a link for performing “Create New Packet Randomly” by using index.jsp page which is the Home page of web interface.

ii. Then users have to input a packet id which packet schema will use to create a new packet randomly.

iii. Now ForwardToNewAction class perform the collecting these packet id and execute a query with this packet id to collect that packet fields information from database.

iv. If these packet is exist in the database, ForwardToNewAction action class retrieve the all fields value and other information’s and forward control to other action class that is called CreateNewPacketRandom .

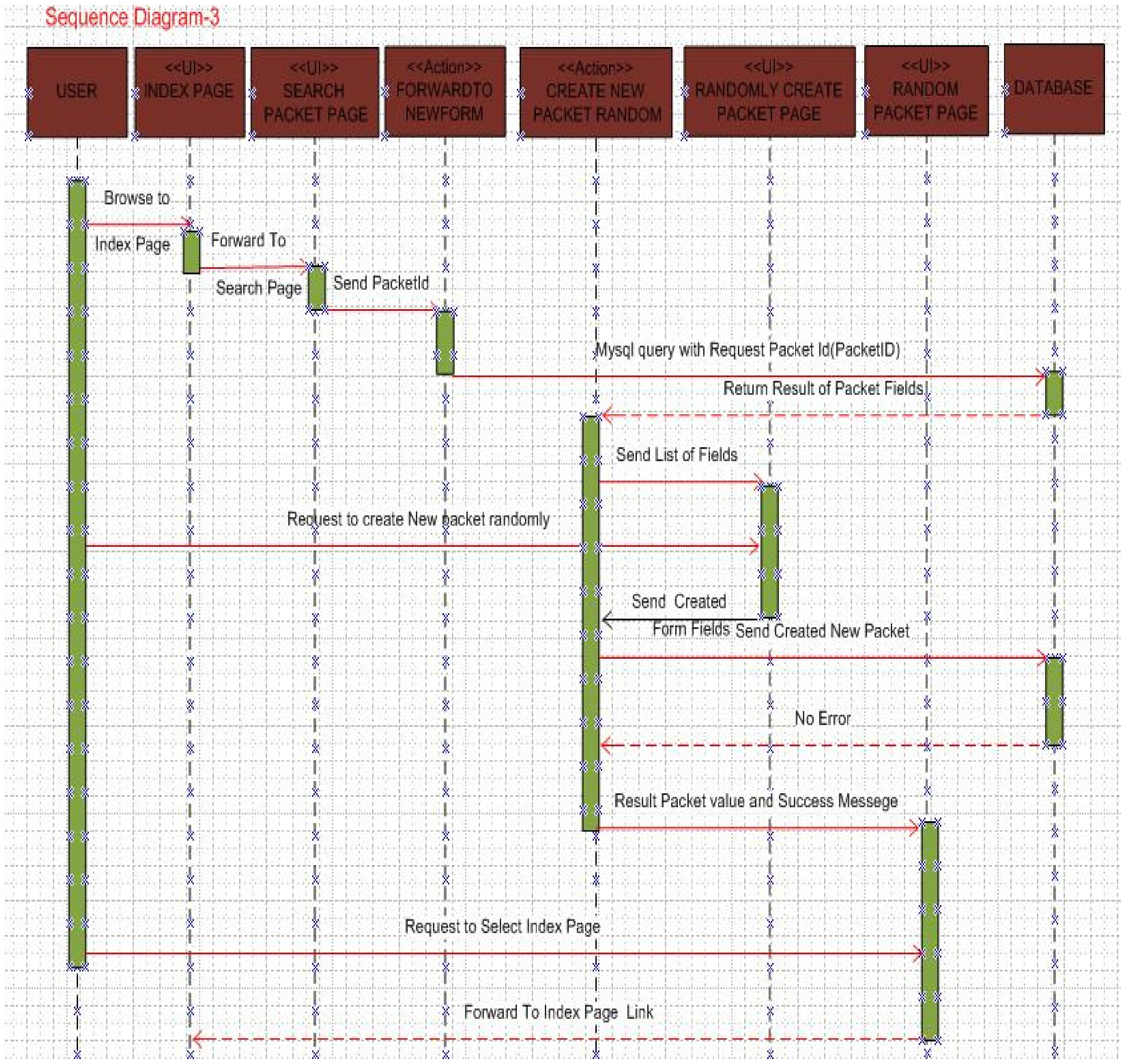

Sequence Diagram-3
USER
<<UI>> INDEX PAGE
<<UI>> SEARCH PACKET PAGE
<<Action>> FORWARDTO NEWFORM
<<Action>> CREATE NEW PACKET RANDOM
<<UI>> RANDOMLY CREATE PACKET PAGE
<<UI>> RANDOM PACKET PAGE
DATABASE
Browse to Index Page
Forward To Search Page
Send PacketId
Mysql query with Request Packet Id(PacketID)
Return Result of Packet Fields
Send List of Fields
Request to create New packet randomly
Send Created Form Fields
Send Created New Packet
No Error
Result Packet value and Success Messege
Request to Select Index Page
Forward To Index Page Link

Figure-7: A sequence diagram to show how to create new packet randomly

v. These class forward list of field’s information to randomly createpackt.jsp for user and show that in a form.

vi. Now user allow to input one or more ranges to create each fields value and fields value automatically generated in field.

vii. After creating all fields it will submit by user and then CreateNewPacketRandom class call a function to perform update query in database.

viii. When all new fields information and packet value stored successfully in the database, the action class forward that new packet value and a successful transaction message to the user in randompacket.jsp page.

ix. Finally, user can go back to home page by using a forward link in radompacket.jsp page.

4.1.2.4 Show packets from database

After updating or creating or modifying data packet in the database, user may need to control whether packet modified/create correctly or not. Now user can see all packet value and their all fields’ information with separating all the elements of the fields. Here is a short description of these sequence diagram bellow:

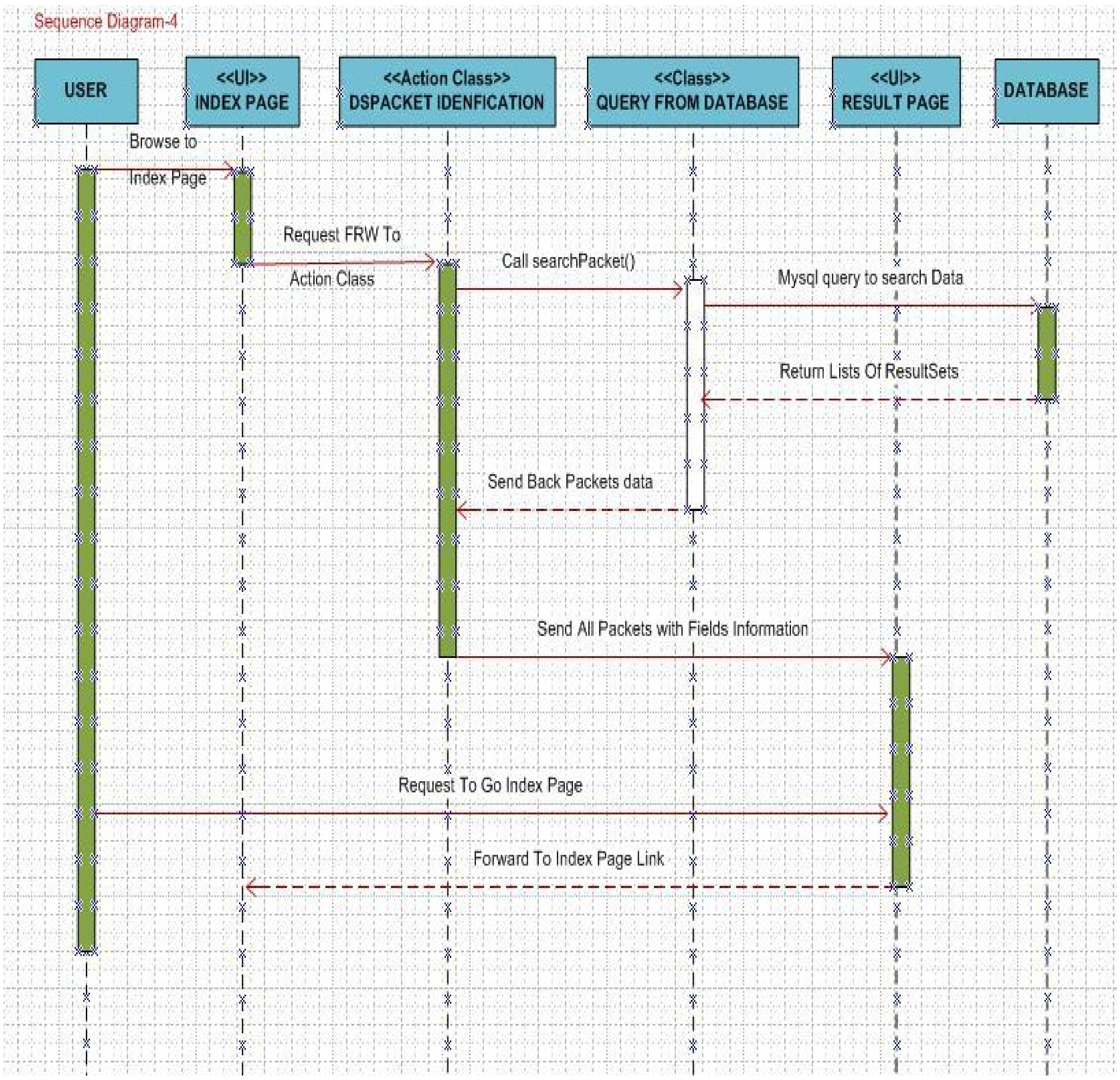

Sequence Diagram-4
USER
<<UI>> INDEX PAGE
<<Action Class>> DSPACKET IDENFICATION
<<Class>> QUERY FROM DATABASE
<<UI>> RESULT PAGE
DATABASE
Browse to Index Page
Request FRW To Action Class
Call searchPacket()
Mysql query to search Data
Return Lists Of ResultSets
Send Back Packets data
Send All Packets with Fields Information
Request To Go Index Page
Forward To Index Page Link

Figure-8: A sequence diagram to show all packets from database

I. Using home page or index page, user can find a button named "Show All Packet "to get all the packets information's.

II. The action class DsPacketIdentification class call a searchPacket () function to query all packets from database. But it uses another helper class to call that function. That class is called QueryFromDatbase.It work is to execute query from database and send control to DspacketIdentification action class.

III. After getting lists packets and their fields information's, Action class send all the values to the Result.jsp page and it shows in a human readable format.

IV. Finally, user can back to the home page again or abort from the application.

4.2 Algorithm to field generation with random rage values

```
randomValueGeneration(){   // Main function to generate random value

        Val data ranges: = get Input ();
        Val Range: = Split (data ranges," ,")
        Val I: =0, Max_ Sum =0
        Array Max, Min, PRBT, Rand Values
        While (Range. length){
                Val Index: =0
                 Index: = Range.getInex ("-");
                  Max_ Sum: = Max[i]:= Range (0, Index) +Max_ Sum
                  Min[i]:= Range (Index+1)
                  Rand Values [I]:= getEachRangeRandValue (Max, Min, I)
                  I++

                }
        For (Each Min Value Iterate K) {
                PRBT [K] =get Probability (Max_Sum, Min [], K,)

        }
        Val    Rand Value
        Rand Value: = GetRandValue (PRBTK, RandValues)
        Return Rand Value
}



GetEachRangeRandValue (Max, Min, I) {  // generate a random value of specific range

        Val Rand No: = Math. Rand ()
        Val Value: = Math. Floor (Math. rand ()*(Max-Min+1) +Min)
Return Value
}

Get Probability (Max_Sum, Min [], K,) {  // calculate probability for each of range value

        Val Prb
        Prb: = (Max[k]-Min[k] +1)/ Max_ Sum
        Return Prb
}
```

4.2.1 Some examples of random generation of values

i. If user inputs 2 ranges for a field as example 0-20,100-210. Then random generator automatically finds out random value for each range. For this example, random generator may create a value 12 for first range with probability (21/(21+111))=0.15 and for second range it get value 120 with probability of (210-100+1)/(21+111)=0.75. Using their probabilities, system will take a value for field among 12 or 120. This process will perform for each of field ranges.

ii. Here user can input any range of value and unlimited number of ranges to create fields. Suppose user input 3 ranges for a field and those are 30-100,400-1000, 1000-65500. Suppose system random values are generated 3 values 40,110 and 34000 respectively with their probabilities 0.0012, 0.0032, 0.99. After than system will choose one value among them .For this example it can choose 34000 as field value.

4.3 Algorithm to blob conversion

```
Result Set rs1:= result all result sets from Ds_packet table

Result Set rs2:= set all result set from Ds_field_desc table

 While [rs1.next] {
  Result bean: = new Result Bean ()
  Result bean: =setBlobId (rs1.id)
  Result bean: =setBlobId (rs1.DsBlob)
        While [r2! =null] {
                Val db2.length, db2.size, limit: =0, index: =0
                 While [r2.next] {
                 db2.size=get Size (r2.size)
                 db2.length=get Length (r2.length)

      If [db2_length=1]{
                Limit: =limit+db2_size
                fieldValue1:= index of (index, limit) for db1_blob
                Field: = new Ds_fieldDataStoreBean ()
                Field: = setfieldvalue (fieldvalue1)
                resultbean.setfield (field)
                        }
     Else if [db2_length>1]
               Limit: =index+db2_size
                fieldValue2:= index of (index, limit) for db1_blob
                Index: =index+db2_size
                Field: = new Ds_fieldDataStoreBean ()
                Field: = setfieldvalue (fieldvalue2)
                resultbean.setfield (field)
                }

        }
}
```

4.3.1 Example of Blob conversion

Verification transaction packet’s data are store in a database on blob format i.e. Binary data. Now this packet data need to show for a user in a readable format. For example a packet value is:”11101111000001111000” and its Meta information provide 2 fields. First field is size 2 and length 5. Second field is size 5 and length 1.Now conversion will show First field with two elements such [11101] and [11100]. Second field with elements such as [00], [01], [11], [10], [00].

5 Implementation

The main objective of this project is to implement a web user friendly web interface for digital verification streaming objects. So I wanted to develop a well-organized web interface for users. Digital verification transaction data are persisting in simple Mysql database which are comprise of different tables with relation.

5.1 Graphical User Interface

5.1.1 Home Page

This is the starting page for the user whose are going to execute operation on the stored packet from database. Home page is a very simple graphical user interface with four buttons. With those button's user can perform respective operations such i.e Create new Packets, Modify Packets etc. Here is a snapshot of the Home page when the application was executing in an application server.

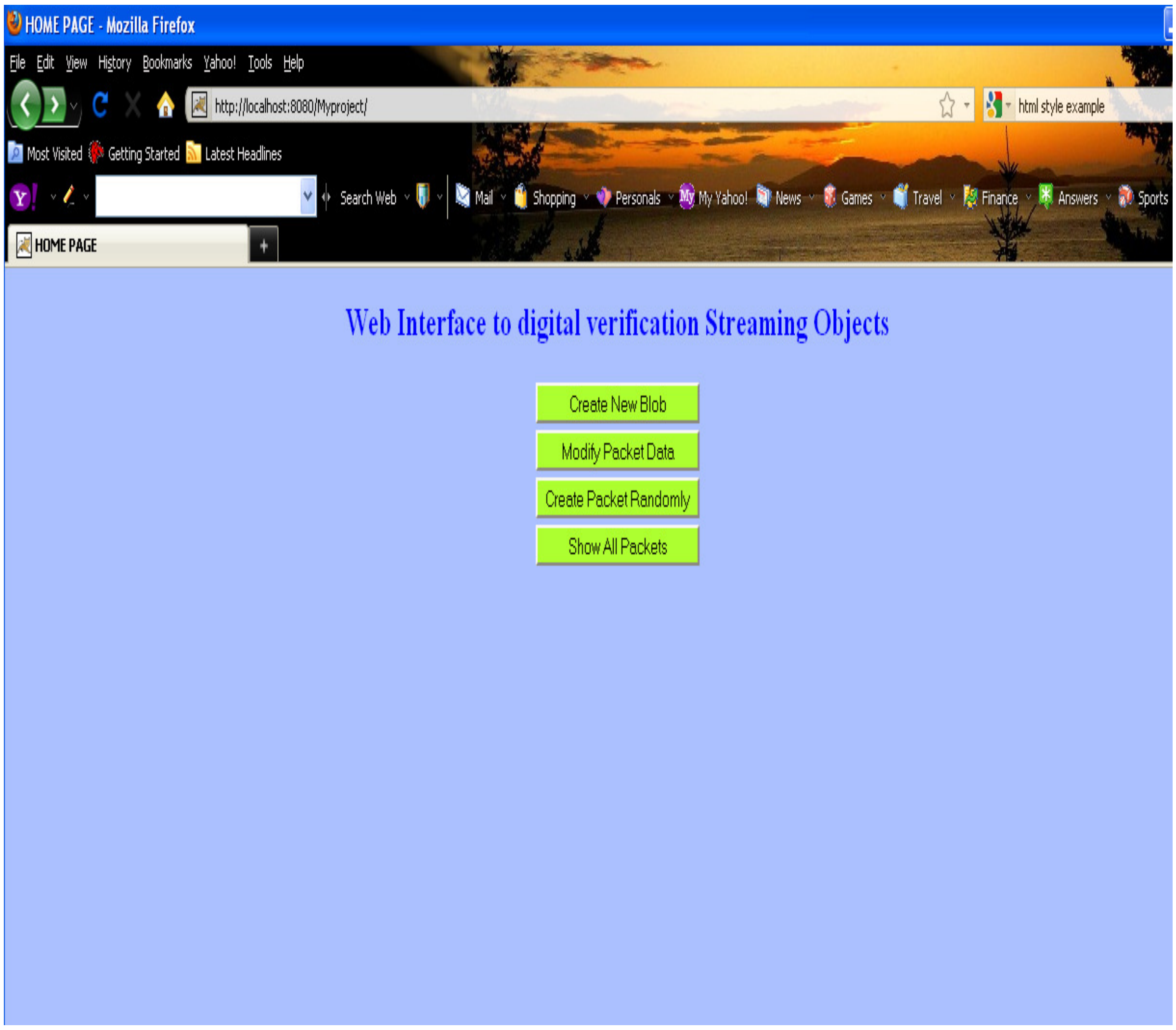

HOME PAGE - Mozilla Firefox
File Edit View History Bookmarks Yahoo! Tools Help
http://localhost:8080/Myproject/
html style example
Most Visited Getting Started Latest Headlines
Search Web Mail Shopping Personals My Yahoo! News Games Travel Finance Answers Sports
HOME PAGE
Web Interface to digital verification Streaming Objects
Create New Blob
Modify Packet Data
Create Packet Randomly
Show All Packets

Figure-10: Home page of the web interface.

5.1.2 Create New Packet

Sometimes users may need to create a new packet for testing purpose. To create new packet, users can access specific type of packet from database and after changing that packet it can be persist in the database perfectly. Now the user interaction with the system to create a new packet is shown bellow in step by step:

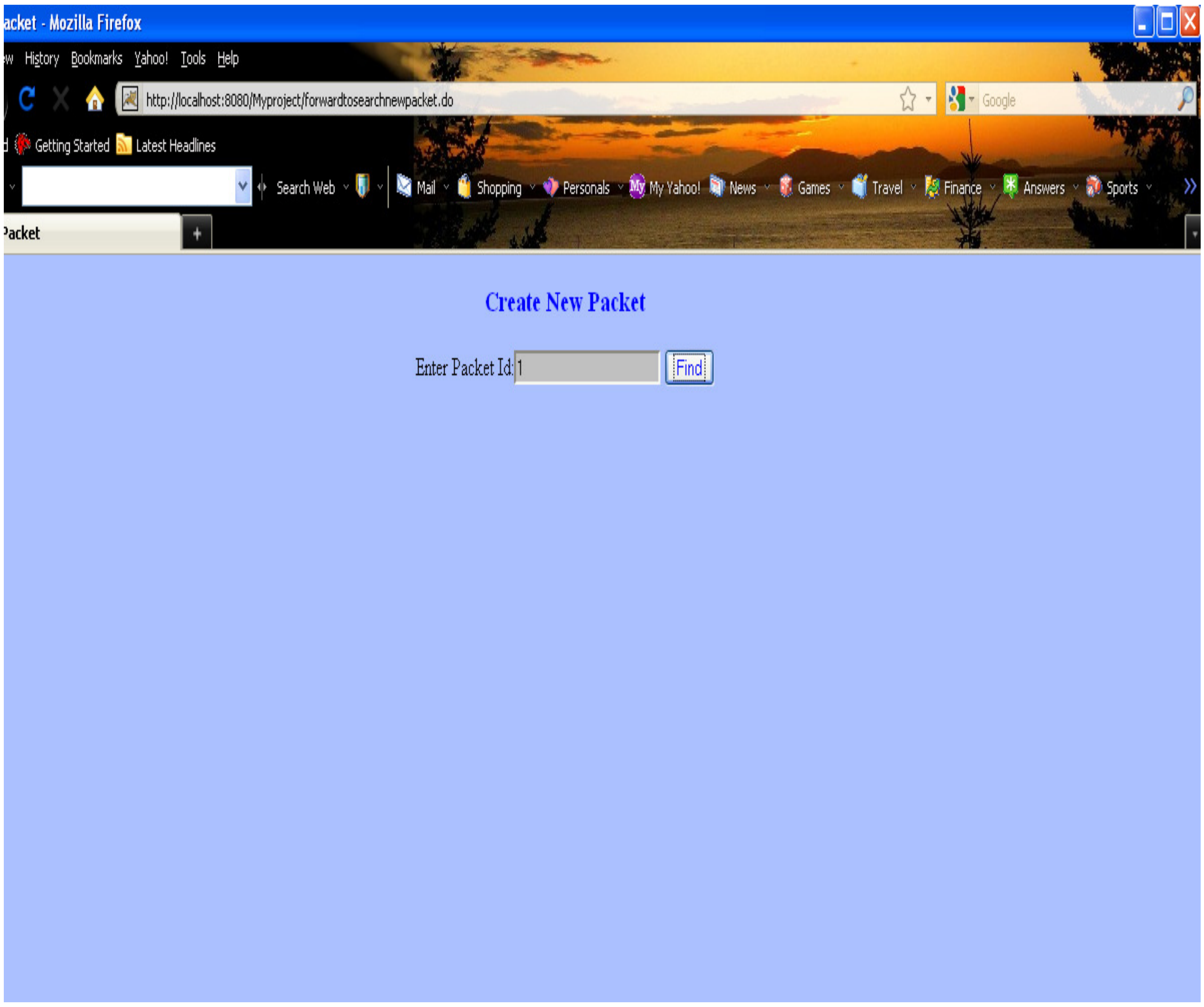
I. Users will search a packet by using Packet Id

II. If that packet does not exist in the database, it response with a error a message "The packet is not exist" and try with other packet. Otherwise in success, the servers will response with a new packet form with every field and all control information's (field value, field size, field length etc) of request packet.

III. Without keeping any change of schema of the packet, Users can input new packet value and submit it to the server to create a new packet on that specific packet schema in database.

IV. If all information successfully stored in the Ds_Packet and Ds_Field_Desc tables, then server will send a Successful creation of new packet message to the user.

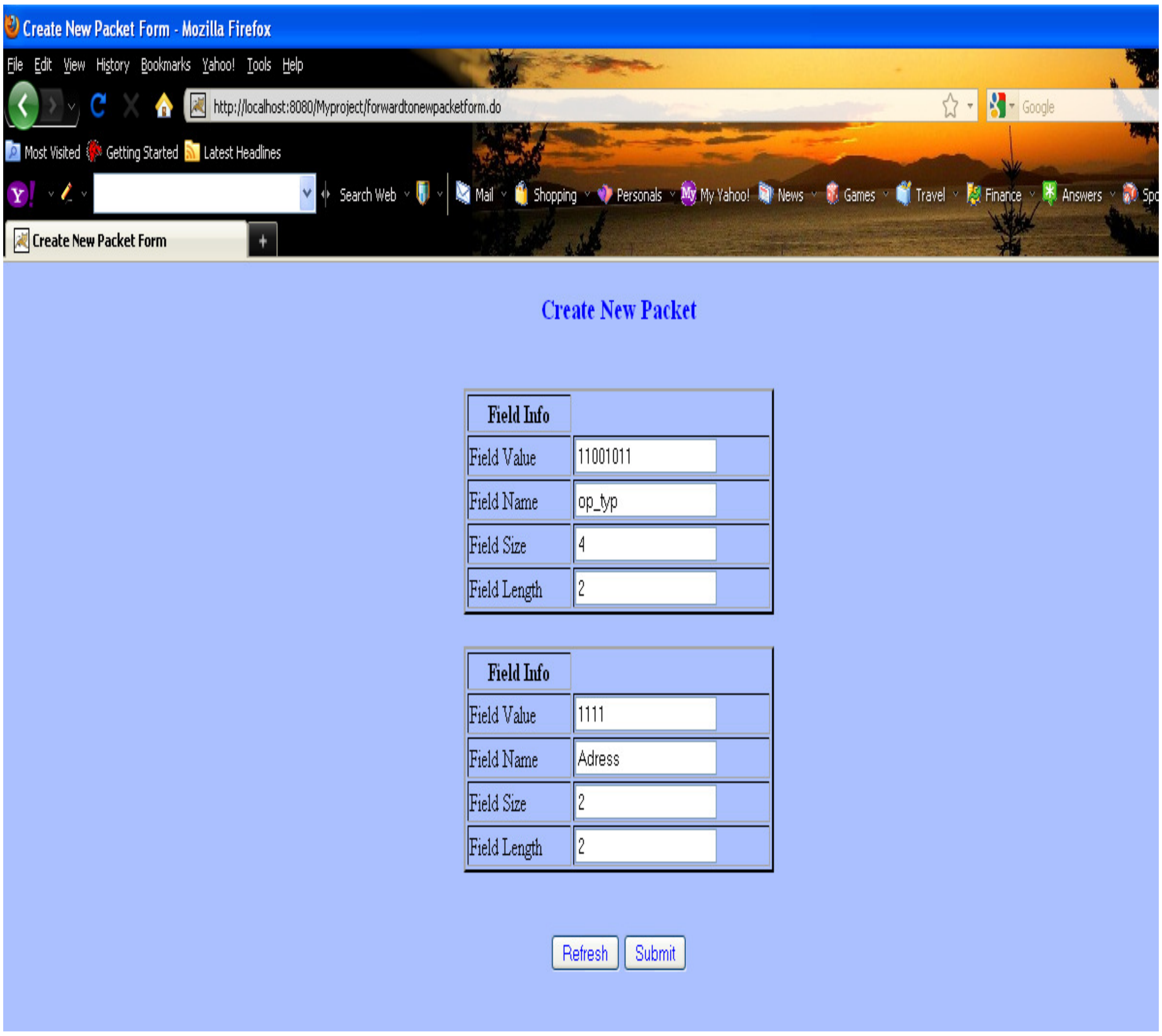
V. After that users can check whether the new created packet by using show all value button or users can back to the home page.


acket - Mozilla Firefox
w History Bookmarks Yahoo! Tools Help
http://localhost:8080/Myproject/forwardtosearchnewpacket.do
Google
Getting Started Latest Headlines
Search Web
Mail Shopping Personals My Yahoo! News Games Travel Finance Answers Sports
Packet

Create New Packet

Enter Packet Id: 1 Find

Figure-11: Searching Packet from database

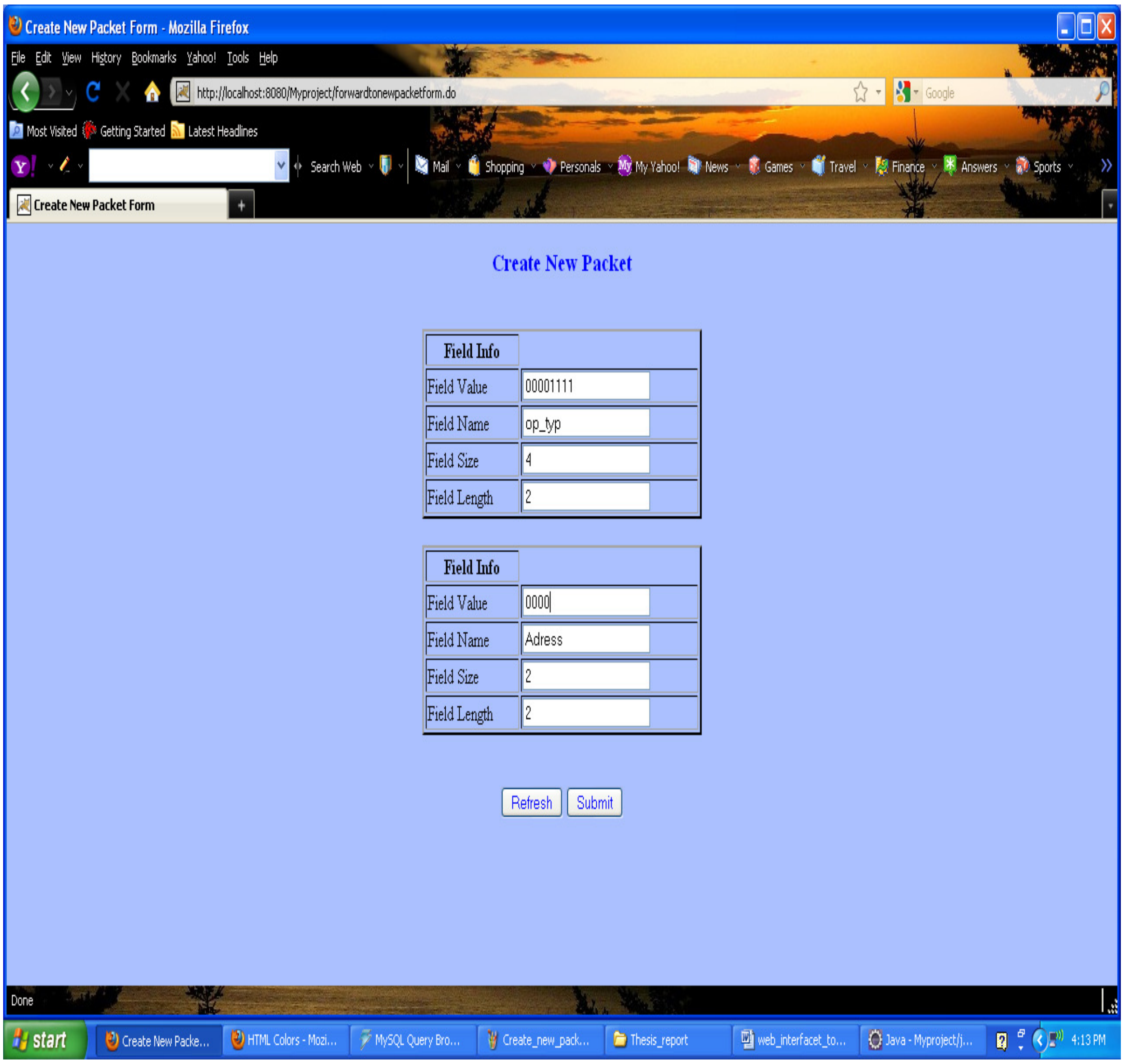

Create New Packet Form - Mozilla Firefox
File Edit View History Bookmarks Yahoo! Tools Help
http://localhost:8080/Myproject/forwardtonewpacketform.do
Refresh
Submit

Create New Packet

Field Info	
Field Value	11001011
Field Name	op_typ
Field Size	4
Field Length	2

Field Info	
Field Value	1111
Field Name	Adress
Field Size	2
Field Length	2

Figure-11: Feedback of request packet information from database


Create New Packet Form - Mozilla Firefox
http://localhost:8080/Myproject/forwardtonewpacketform.do
Done

Create New Packet

Field Info	
Field Value	00001111
Field Name	op_typ
Field Size	4
Field Length	2

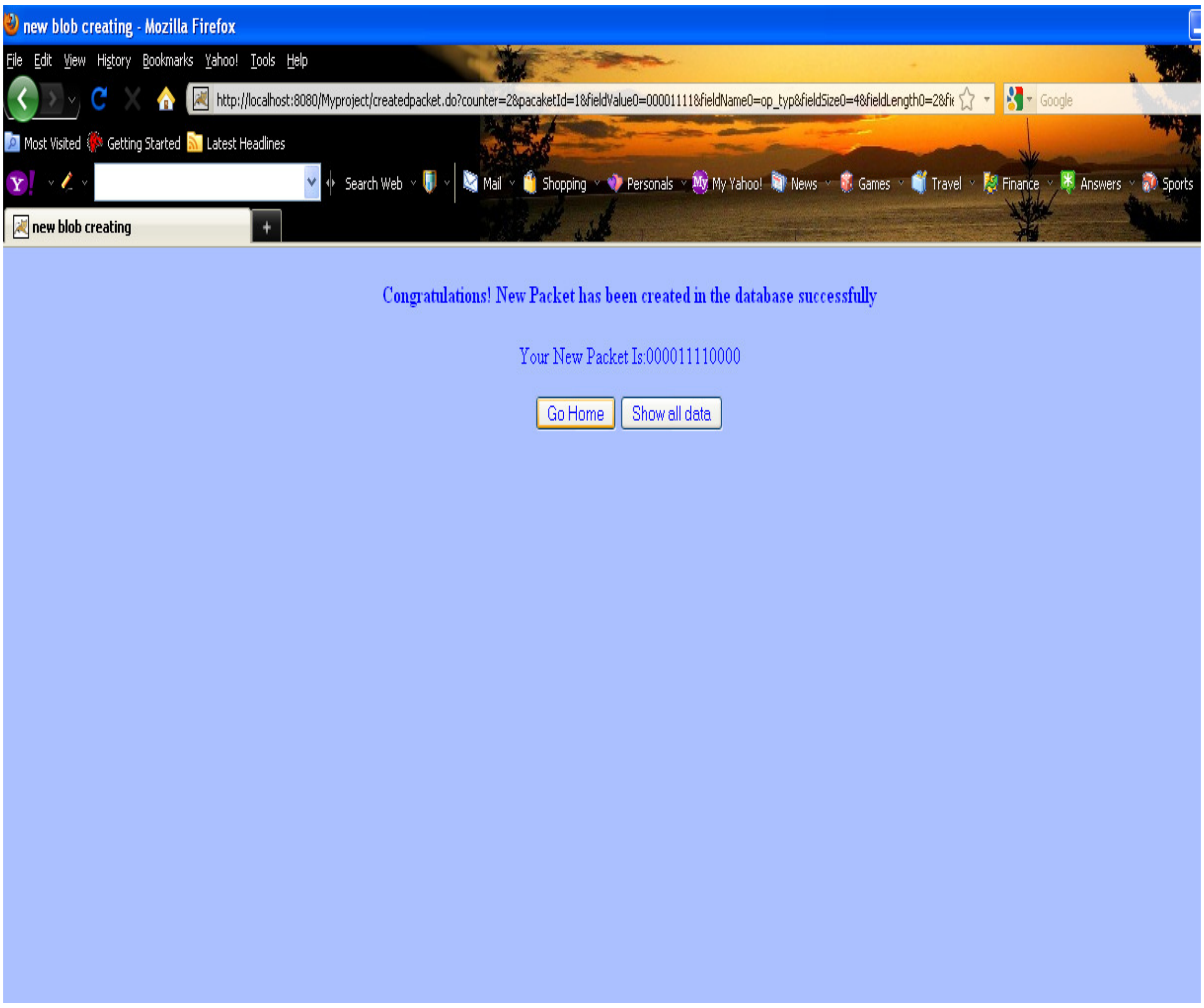
Field Info	
Field Value	0000
Field Name	Adress
Field Size	2
Field Length	2

Figure-12: User is entering new packet value in this form

5.1.3 Modify Packet Fields

To reconstruct of transaction data or packet, it may need to modify or change in the stored packet from database. But this task can be done very easily with this web application.

After searching the target packet from the database, users can modify or change operation in any packets. The process is describes bellow:


new blob creating - Mozilla Firefox
File Edit View History Bookmarks Yahoo! Tools Help
Most Visited Getting Started Latest Headlines
Search Web Mail Shopping Personals My Yahoo! News Games Travel Finance Answers Sports
new blob creating
Go Home
Show all data

http://localhost:8080/Myproject/createdpacket.do?counter=2&pacaketId=1&fieldValue0=0001111&fieldName0=op_typ&fieldSize0=4&fieldLength0=2&fie

Congratulations! New Packet has been created in the database successfully

Your New Packet Is:000011110000

Figure-13: Final throughput of created new packet is showing in browser

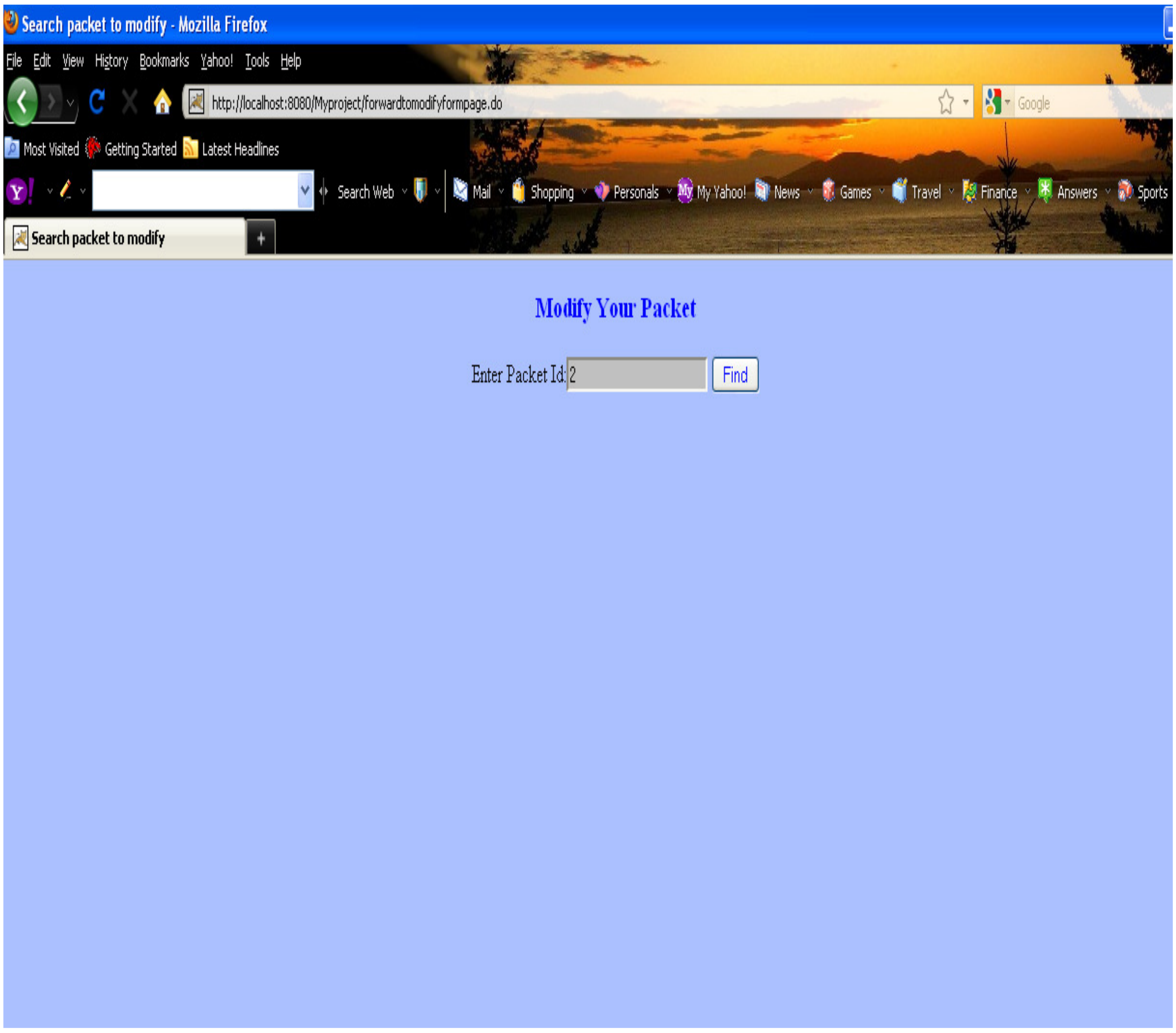
i. Like “Create new packet” process users can select or search a packet with Packet Id.

ii. If server find that packet in the Ds_Packet table, it collect that packet value and all control information/fields information (field name, field size, field length ,field element order etc) of that packet from Ds_Field_Desc table and Send back all data as response to user with creating a from of field information’s dynamically .

iii. Now users can modify or change packet with constraint of unchanged schema of that packet. Users can change the packet values and then will send it to server to store packet in the database.

iv. If the new modified packet store successfully in database, server will send a response with successful modification of packet to users. Otherwise it will send an error message of re-entering packet.

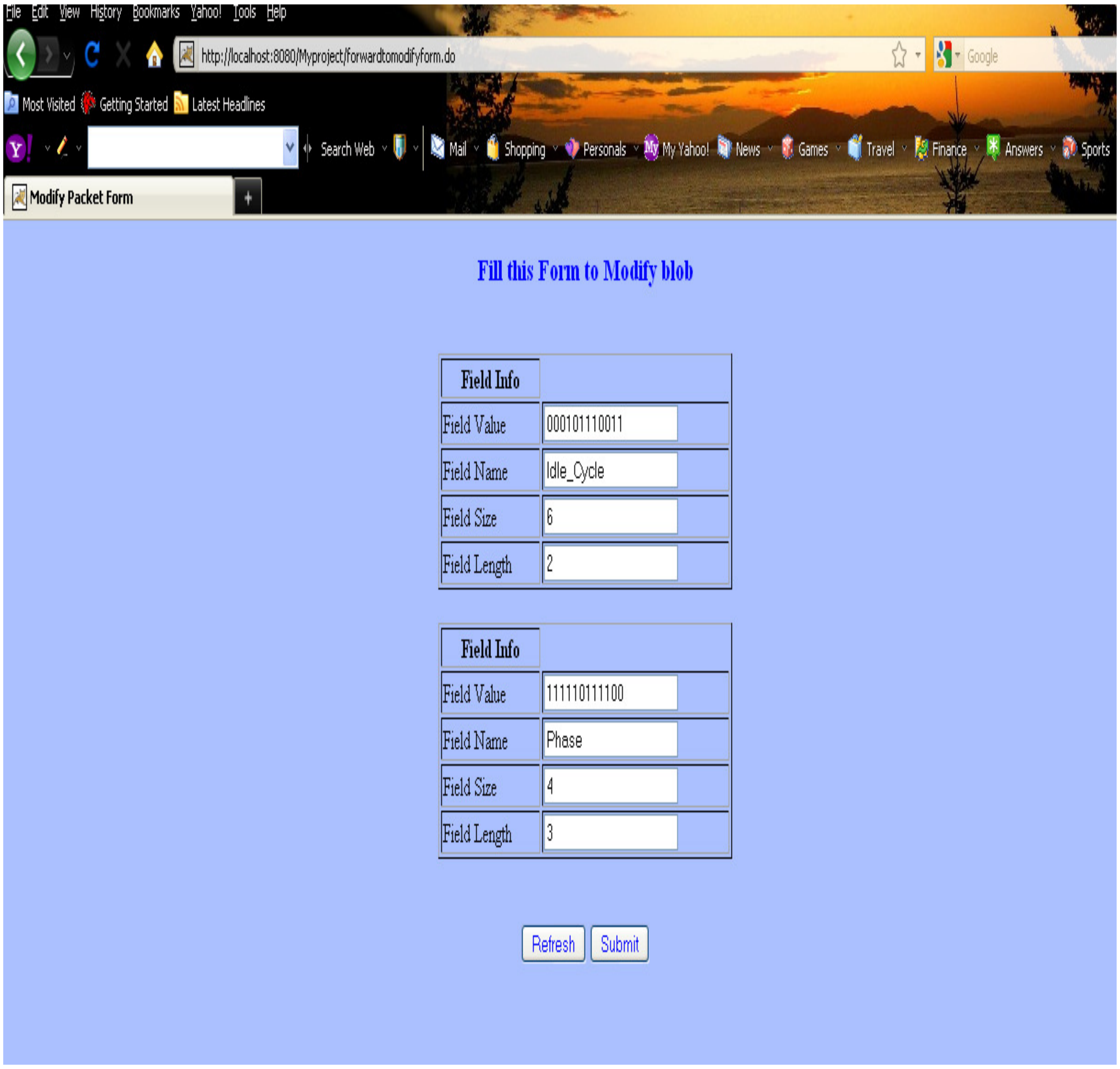
v. Now user can check the packet modified correctly or not by using show all packet data from database.


Search packet to modify - Mozilla Firefox
File Edit View History Bookmarks Yahoo! Tools Help
http://localhost:8080/Myproject/forwardtomodifyformpage.do
Search packet to modify
Find

Modify Your Packet

Enter Packet Id: 2

Figure-15: Search packet page from database

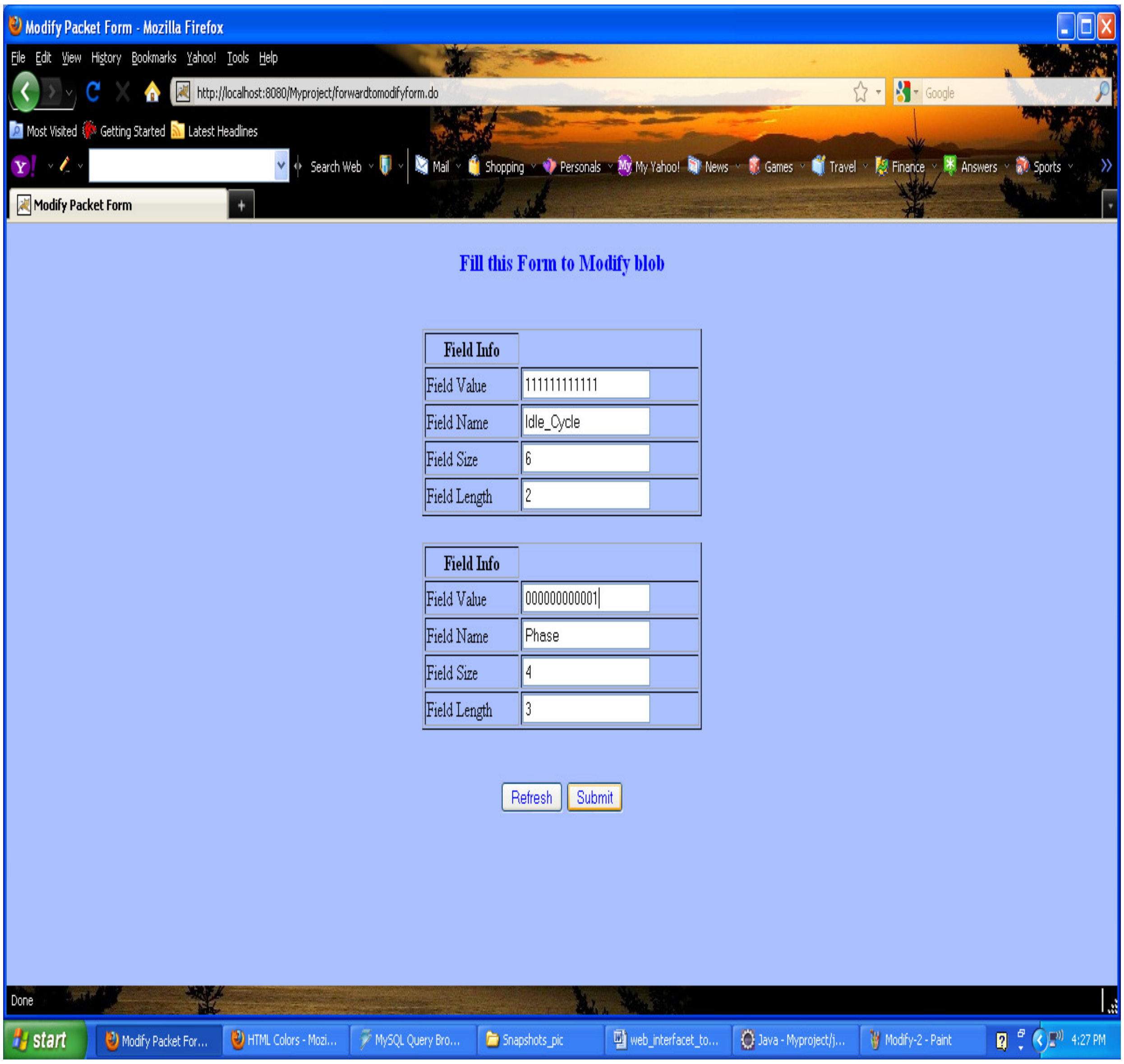

File Edit View History Bookmarks Yahoo! Tools Help
http://localhost:8080/Myproject/forwardtomodifyform.do
Modify Packet Form
Refresh
Submit

Fill this Form to Modify blob

Field Info	
Field Value	000101110011
Field Name	Idle_Cycle
Field Size	6
Field Length	2

Field Info	
Field Value	111110111100
Field Name	Phase
Field Size	4
Field Length	3

Figure-16: Showing retrieved Fields information from database

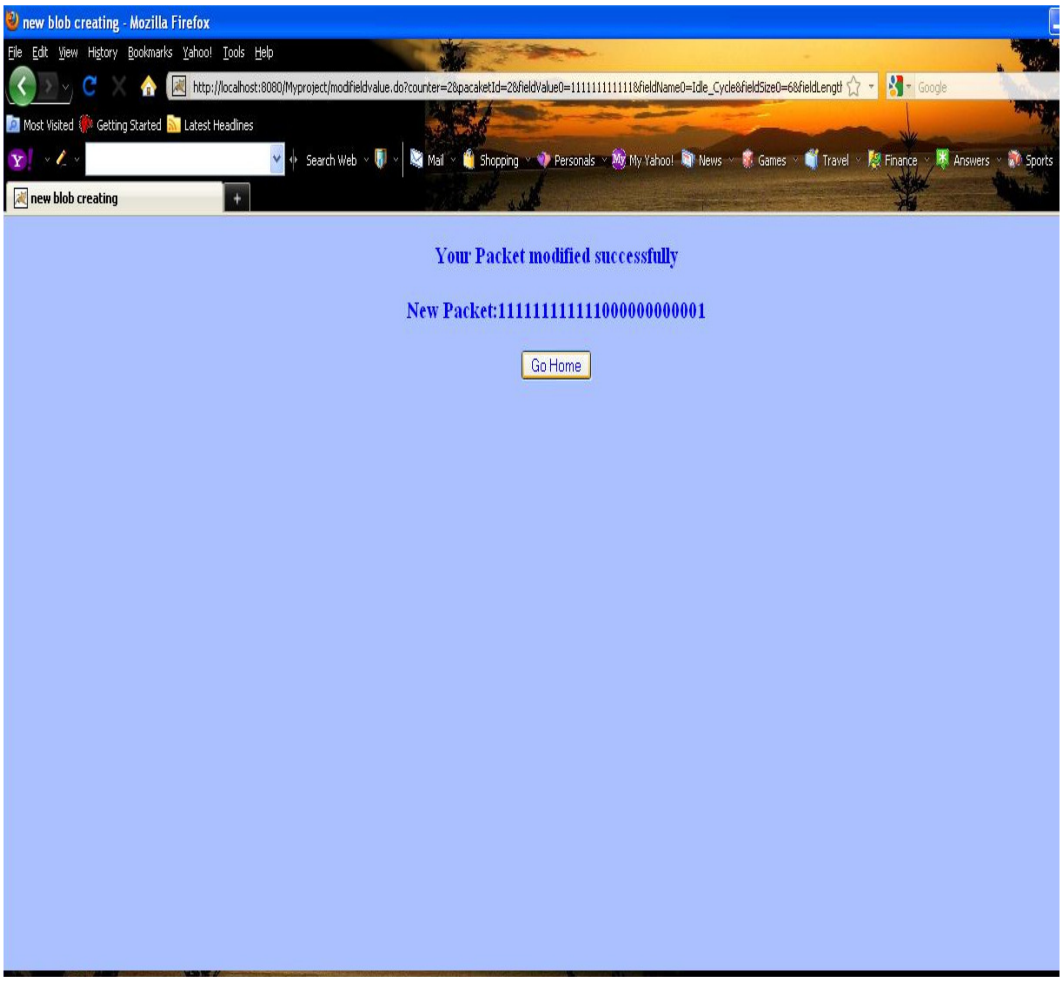

Modify Packet Form - Mozilla Firefox
http://localhost:8080/Myproject/forwardtomodifyform.do
Modify Packet Form
Refresh
Submit
Done

Fill this Form to Modify blob

Field Info	
Field Value	111111111111
Field Name	Idle_Cycle
Field Size	6
Field Length	2

Field Info	
Field Value	000000000001
Field Name	Phase
Field Size	4
Field Length	3

Figure-17: User input page to modify packet


new blob creating - Mozilla Firefox
File Edit View History Bookmarks Yahoo! Tools Help
Most Visited Getting Started Latest Headlines
Search Web Mail Shopping Personals My Yahoo! News Games Travel Finance Answers Sports
new blob creating
Go Home

http://localhost:8080/Myproject/modifieldvalue.do?counter=2&pacaketId=2&fieldValue0=111111111111&fieldName0=Idle_Cycle&fieldSize0=6&fieldLengt

Your Packet modified successfully

New Packet:1111111111111000000000001

Figure-18: Page is showing new modified packet

5.1.4 Create New Packet Randomly

It provides facility to users for creating a new packet which field values are created from various range of value. Users can choose one or more ranges for fields and then field value automatically created randomly according to users input ranges. Any length of bits range can be acceptable such as 0-255,30-65,500, etc. The execution process or system interactions with users are describing bellow:

i. First users need to search the packet on which he/she will create new packet with different ranges random value.

ii. If It can find the users requested packet in the database then it will dynamically collect that packet's field's and their all other control information i.e. field

Name, field length, Field size, field order etc. and send back a form with all that information to users.

iii. After that Users can put desire ranges for each field and dynamically server will response with creating a field value which value is consists of random value of input ranges.

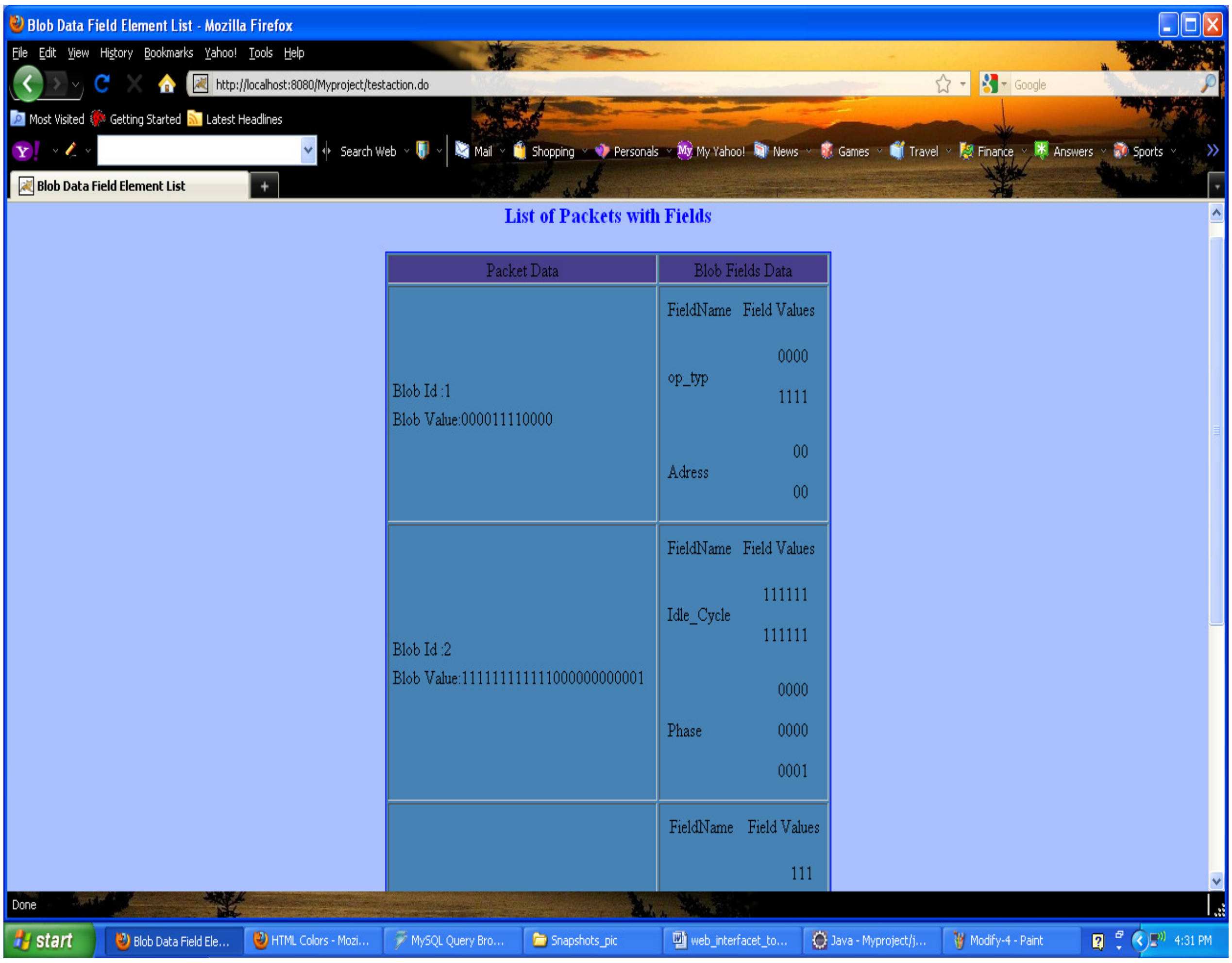
iv. Then users submit the entire field's information to the server.

v. After generating a new packet from submitted fields, it store the new packet in the database table and show a message to users that new packet stored in the database successfully.

vi. At this point, Users can check the new packet using show all packets or can be leave from this part to main page.

5.1.5 Show all Packets

After completion of a action or operation users may need to check packet does stored accurately in the database or not. To observe the packets in database, users simply have to click a button and server will respond with the entire packet's information from database with a human readable graphical representation on web page.


Blob Data Field Element List - Mozilla Firefox
http://localhost:8080/Myproject/testaction.do
Blob Data Field Element List
Done
start
4:31 PM

List of Packets with Fields

Packet Data	Blob Fields Data
Blob Id :1 Blob Value:000011110000	FieldName Field Values op_typ 0000 1111 Adress 00 00
Blob Id :2 Blob Value:1111111111111000000000001	FieldName Field Values Idle_Cycle 111111 111111 Phase 0000 0000 0001
	FieldName Field Values 111

Figure-20: All packets and their field's element showing separately

5.2 Implementation Requirements

This section represents the implementation choice of particular requirement in terms of used scripting language, java development tools, and class libraries as well as solving techniques. It also describes development strategy.

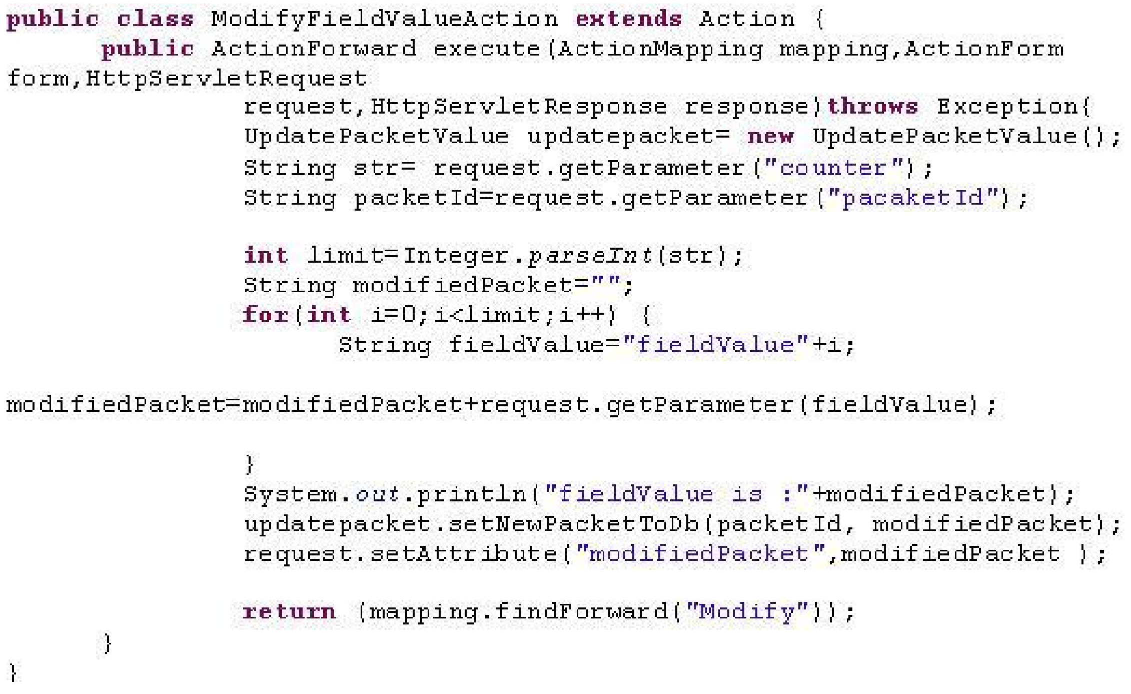
The scripting language JavaScript is in front-end to produce dynamic web pages and for back-end side I used J2EE platform and its web development technology such as Servlet, Jsp. A framework used in the whole application that is struts framework. A lot of Action classes and plain coding java classes are used to execute the back-end application. In the front-end I made some JavaScript functions to handle dynamically various types of data from database and also creation of different dynamic table and form.

5.2.1 Action Classes

In struts framework controller handles all request response by using some different class which is called Action class. Controller selects which Action class is appropriate for current type of request and then forward request to action class. Action class then process that request by using other helper class/java bean and it update the view page for the client.

5.2.1.1 Action class for modification task

This class performs creating modified packet and persist packet information in database. To accomplish this task, it call a function from UpdatePacketValue helper class to store and create new modified packet from user's input fields value's. Finally it forward modified packet to jsp page for the user.



```
public class ModifyFieldValueAction extends Action {
      public ActionForward execute(ActionMapping mapping,ActionForm
form,HttpServletRequest
               request,HttpServletResponse response)throws Exception{
               UpdatePacketValue updatepacket= new UpdatePacketValue();
               String str= request.getParameter("counter");
               String packetId=request.getParameter("pacaketId");

               int limit=Integer.parseInt(str);
               String modifiedPacket="";
               for(int i=0;i<limit;i++) {
                     String fieldValue="fieldValue"+i;

modifiedPacket=modifiedPacket+request.getParameter(fieldValue);

               }
               System.out.println("fieldValue is :"+modifiedPacket);
               updatepacket.setNewPacketToDb(packetId, modifiedPacket);
               request.setAttribute("modifiedPacket",modifiedPacket );

               return (mapping.findForward("Modify"));
      }
}
```

5.2.1.2 Action class to create new packet

CreateNewBlobAction is an action class which first task is to collect user requested form input fields by using ActionForm Bean. ActionForm is a bean class which persists user submitted form values and to reuse by other class. Using that fields information, Action class call a function to create new packet and also some functions to store data in different database tables. It also sends back the new created packet by controller to jsp page.

```
Public class CreateNewBlobAction extends Action {
      Public ActionForward execute (ActionMapping mapping, ActionForm
form, HttpServletRequest
                     Request, HttpServletResponse response) throws
Exception {
                     UpdatePacketValue updatepacket= new
UpdatePacketValue ();
                     String STR= request.getParameter ("counter");
                     String packetId=request.getParameter
("pacaketId");
                     Int packetid=Integer.parseInt (packetId);
```

```
                    Int limit=Integer.parseInt (STR);
                    String newCreatedPacket="";
                    Int id_ord_field=1;
                    for (int i=0;i<limit;i++) {
                          String fieldValue="fieldValue"+i;
                          String fieldName="fieldName"+i;
                          String fieldLength="fieldLength"+i;
                          Int fieldlen;
                          Int fieldsize;
                          String fieldSize="fieldSize"+i;

NewCreatedPacket=newCreatedPacket+request.getParameter (fieldValue);
                          FieldName=request.getParameter (fieldName);
                          FieldSize=request.getParameter (fieldSize);
                          FieldLength=request.getParameter
(fieldLength);
                          Fieldlen=Integer.parseInt (fieldLength);
                          Fieldsize=Integer.parseInt (fieldSize);
                          System.out.println ("fieldSize:"+fieldSize);
                          updatepacket.setRecord_For_Field_Desc_Db
(packetid, fieldName, fieldsize, fieldlen, id_ord_field);
                          Id_ord_field++;

                    }
                    System.out.println ("fieldValue
is:"+newCreatedPacket);
                    updatepacket.setNewPacketToDb (packetId,
newCreatedPacket);
                    request.setAttribute ("newCreatedPacket",
newCreatedPacket);

                    Return (mapping.findForward ("shownewpacket"));
             }
}
```

5.2.2 Struts-config.xml file configuration

Struts is a model-view-controller (MVC) framework and it is use for the web application. This framework uses a configuration file to initialize its own resources. These resources include collecting input from users; ActionMappings is to direct input to server-side Actions and ActionForwards to select output pages. In fact, I configured in three sections of controller initial resource file. First one is <form-Bean>, then <ActiondMapping> and last one is <ActionForwards>. Here Action-Mappings configured section for my web application is shown shortly:

```
<Action-mappings>
<action path="/testaction" type="myproject.webapp.DsPacketIdentAction">
<forward name="success" path="/WEB-INF/results/result.jsp"/>
</action>

<action path="/forwardtohome"
Type="myproject.webapp.ForwardToHomeAction">
<forward name="gohome" path="/form.jsp"/>
</action>

<action path="/forwardtosearchnewpacket"
type="myproject.webapp.ForwardToSearchPacket">
<forward name="suc" path="/findPacketByid.jsp"/>
</action>


<action path="/forwardtonewpacketform"
type="myproject.webapp.ForwardToNewFormAction" >
<forward name="gotonewform" path="/newblob.jsp"/>
</action>

<action path="/createdpacket" type="myproject.webapp.CreateNewBlobAction" >
<forward name="shownewpacket" path="/newblobcreated.jsp"/>
<forward name="failure" path="/WEB-INF/results/errorpage.jsp"/>
</action>


<action path="/forwardtomodifyformpage"
type="myproject.webapp.modify.ForwardToModifyPageAction" >
<forward name="gonow" path="/findById.jsp"/>
</action>

<action path="/forwardtomodifyform"
type="myproject.webapp.modify.ForwardToModifyFormAction" >
<forward name="gotoform" path="/modifyform.jsp"/>
</action>
```

```
<action path="/modifieldvalue"
type="myproject.webapp.modify.ModifyFieldValueAction" >
<forward name="Modify" path="/modifiedsuccess.jsp"/>
<forward name="failure" path="/errorpage.jsp"/> </action>

<action path="/showData" type="myproject.webapp.ShowPacketDataAction"><forward
name="showData" path="/ShowAllValue.jsp"/></action>

<action path="/forwardtorandcreateblob"
type="myproject.webapp.newblobwithconstraint.ForwardToNewFormAction">
<forward name="ok1" path="/forwardtoform.jsp"/></action>

<action path="/forwardtonewform"
type="myproject.webapp.newblobwithconstraint.ForwardToFormAction">
<forward name="ok" path="/rendomcreatblob.jsp"/>
</action>


<action path="/createrandomblob"
type="myproject.webapp.newblobwithconstraint.CreateNewBlobRandAction" >
<forward name="success1" path="/randPacket.jsp"/> </action>

</action-mappings>
```

5.2.3 Web.xml file configurations

A web application uses a deployment descriptor to initialize resources like Servlets and Taglibs. The deployment descriptor is formatted as a XML document and name “web.xml”.The web.xml web application descriptor file represents the core of the Java web application, so it is appropriate that it is also part of the core of the Struts framework. In the web.xml file, Struts defines its FilterDispatcher, the Servlet Filter class that initializes the Struts framework and handles all requests. Here is a simple web.xml configuration that I used for my web application:

Project name defined in the <display-name> tag

```
<display-name>My Final Project</display-name>
```

Action Servlet and strtus-config.xml files and configured as below

```
<Servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
    <init-param>
      <param-name>config</param-name>
      <param-value>/WEB-INF/struts-config.xml</param-value>
     </init-param>
</servlet>
```

This **config** parameter tells the Strut's Action Servlet where to find its central configuration file struts-config.xml

```
<servlet-mapping>
         <servlet-name>action</servlet-name>
         <url-pattern>*.do</url-pattern>
<servlet-mapping>
```

In servlet mapping <url-pattern> tag is saying that ActionServlet should process any requests for pages that's end in .do

Tag Library descriptors will be configured as below

```
<Taglib>
<taglib-uri> /WEB-INF/tlds/struts-bean.tld</taglib-uri>
<taglib-location> /WEB-INF/tlds/struts-bean.tld</taglib-location>
</taglib>
```

The first page of the web application is given in <welcome-file-list> so that when user enter application domain, it will get automatically the home page of the application. All those tags must in between <web-app> and </web-app> tags.

5.2.4 Implemented JavaScript functions

5.2.4.1 Dynamic form creation function

Some JavaScript functions have been created to handle dynamic task on table, form etc. In the modification packet part, when user get the fields information by searching packet id, then a javascript function called createTable () that creates a form. This form shows field information's those are collect from database. Data collections from database are also done dynamically. Another JavaScript function renders all fields information in a Jsp page.

```
function createTable(i){
      var myForm= document.getElementById('myform');

      var fieldValue=document.getElementById('fieldvalues').value;
      var fieldName=document.getElementById('fieldname').value;
      var fieldSize=document.getElementById('fieldsize').value;
      var fieldLength=document.getElementById('fieldlength').value;
     parentElement=document.getElementById("parentId");
     tbl = document.createElement("table");
     tbl.setAttribute('id','table1');
     tbl.setAttribute('width',"320px");

     tblBody = document.createElement("tbody");
    // row = document.createElement("tr");
     cell = document.createElement("th");
       textNode = document.createTextNode("Field Info");

     cell.appendChild(textNode);
   //  row.appendChild(cell);
     tblBody.appendChild(cell);
  var row1 = document.createElement("tr");
       var cell = document.createElement("td");
       var textNode = document.createTextNode("Field Value");

      cell.appendChild(textNode);
       row1.appendChild(cell);
       var cell = document.createElement("td");
            textField=document.createElement("input");
            textField.value=fieldValue;
            textField.id='fieldValue'+i;
            textField.name='fieldValue'+i;
          cell.appendChild(textField);
          row1.appendChild(cell);
          tblBody.appendChild(row1);


      var row2 = document.createElement("tr");
       var cell = document.createElement("td");
       var textNode = document.createTextNode("Field Name");
```

5.2.7 Used Technology Tools

Different types of programming language and software tools has been used for both Front-end or client side and Back-end side .The whole application built under a framework its called Struts framework. Main platform of this project is java. JavaScript used frequently in front-end and Jsp also used in some part of client side.

5.2.7.1 Font-end tools

Language/tools/Software	**Type/Category**
JavaScript 1.6	Scripting language
Html 2.0	Hypertext markup
JSTL libraries	Tag library
CSS	Design/style/color

Table-1: Fron-end Technologies/Tools

5.2.7.2 Back-end tools

Language/tools/Software	**Type/Category**
Jsp 2.0	J2EE
Eclipse	IDE(Integrated Development Environment)
Servlet 2.0	J2EE
Mysql 5.5	RDBMS
ANT	Apache project builder
Struts 2.0	Framework
Apache Tomcat 6.0	Application Server

Table-2: Back-end Technologies/Tools

6 Conclusion and future work

The web interface to verification object in database enhances and enriches the analysis of verification transaction data and meta-information in an efficient and simple way. Users are allowed to perform these tasks such as modification of blob object, creation of new packet or creation of packet randomly etc. So all these user facilities are: to reuse transaction data, to reduce testing phase and data analysis time. Stored data packets can be managed by user with a simple web interface.

To evaluate this web application, I used some real time verification streaming data which are collected from STMicroelectronics Company. The output of the implementation shows that all of the requirements have been implemented and delivered successfully in time.

As a future work, I had a plan to improve the use authentication and user profile part in the database. This is an important issue of unauthorized user can access database and can change packets and other Meta information of created tests. The graphical representation of stored blob objects can be improved more in future. As an extension work, we can represent a computation coverage based on a set of blob elements where a set of functional cover points represent in XML.

References

[1] OVM official website, Retrieved on January 3rd,2010 from
http://www.ovmworld.org/whitepapers/OVM_Whitepaper_12-21-07.pdf

[2] Alessandro Alice, Sara Comai ,” Open database idea with OVM- An Open Database For the Open Verification Methodology.pdf”, Retrieve on January 11th, 2010.

[3] OVM Hierarchy Chart, Retrieved on January 19th, 2010 from
http://www.iqmagazineonline.com/IQ/IQ22/pdfs/IQ22_pgs52-56.pdf

[4] Struts framework, Retrieved on February 10th, 2010 from
http://www.sequencediagrameditor.com/uml/sequence-diagram.htm.

[5] Servlet & Jsp concept, Retrieved on February 15th, 2010 from
http://www.apl.jhu.edu/~hall/java/Servlet-Tutorial/
.

[6] Sequence Diagram, Retrieved on April 14th, 2010 from
http://struts.apache.org/