Blue Search in Kansas River Database

By
HARITHA REDDY SAMA

A REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas
2008

Approved By:

Major Professor
Dr. Daniel Andresen

Abstract

Students who are Research Assistants under Dr. Craig Paukert, Division of Biology Kansas State University have problems storing/retrieving their field data directly from database. There data is generally related to fishes in Kansas River. All this data is entered in the data sheets during the research in field.

They enter fish related data like tag numbers (each fish caught is given a tag and after noting the data related to the fish it is left back in the river) and other information like size of the fish, place the fish was caught (stations) and information about the shore habitats, water temperature, depth etc. They also need data to recognize the aging in a recaptured fish (fish caught with a tag). The data from these data sheets is manually entered into the database. Each student has their information on the database entered (wherein they actually have to work on a single database). There is inconsistency of data. Also the database doesn't show the concentration of the fishes or location of the fishes right away. For which they require lot of analysis on the data.

The main objective of this project is to provide a solution for this problem. The solution is an interface to store data and retrieve data and show the data on Maps for easy analysis and also have centralized data. The interface is user friendly which removes the hassle from entering the data manually into the database.

Table of Contents

Acknowledgements

I would like to express my deepest gratitude to Dr Craig Paukert and Joanna Whittier, Division of Biology, for their support, assistance and all other facilities that were required during the development of this application.

Heartiest thanks to Dr Daniel Andresen for his encouragement, guidance and his valuable advices during the course of my work.

I would like to thank Dr Gurdip Singh and Dr Mitchell Neilsen for serving in my committee and for their valuable cooperation during this project

1. Introduction

1.1 Goal

The goal of this application is to provide a web interface which can allow users to store/retrieve and also analyze data. The data is generally related to fishes in Kansas River. All this data is entered in the data sheets during the research in field. This data is used to understand the fish population, habitat, and aging in fishes. This information can be used to improve the existing conditions for the fishes in Kansas River. The application provides the following facilities:

- A convenient and easy way to store and retrieve data about the fishes using the web interface.
- To find the information about the recaptured fish (fish caught with a tag).
- Find aging in fishes. This is required to know the growth in fishes.
- Query the database to know the species caught in a given time.
- Calculate Cost per Unit Effort (CPUE) for particular species based on different criteria like river mile, start and end date etc.
- Exporting the CPUE tables to Excel to save it on the end user's machine.
- A web map application which allows the user to locate the different species on Kansas River.
- Allow the user to located endangered species using maps.

1.2 Scope

The application can be used for storing data from a number of fields (Kansas River banks) and help understand the details of fishes. The data collected from the fields can prove very beneficial for the research involved in this field as it helps the researchers in this field to understand the aging in fish and also about endangered species. This data can be used to get an idea about the concentration of a particular species in a particular river mile. Maps generated by the web application can be either used by the researchers from our department or by researchers from other departments/schools to minimize the risk for endangered species.

1.3 Need of the application

There is research data available in the area of Fisheries Science, River Fisheries and Geographic Information System (GIS). The need of the application arises in maintaining this data. This data is mainly used to know about recaptured fish, aging in fishes, query data about different species, and locate different species on maps.

Problem: The data entry forms from the field are entered into the database manually. Each student working on different projects has different data entry form. Each student has different database (wherein they actually have to work on same database). There is inconsistency of data. Also the database doesn't show the concentration of the fishes or location of the fishes right away. For which they require lot of analysis on the data.

Solution: Have a web application, which is user friendly which removes the hassle from entering the data manually into the database. Also have a single database to ensure that all the students are working on the same database and in turn ensuring that each student has the updated data. Also have a map set up for them to view the data on the maps, like showing the concentration of a particular fishes on a river mile and so on.

1.4 Related Work and Problems

These data entry forms can be designed as Access forms in Access 2003. They are easy to develop. But the problems are that the database is not centralized and hence the data is inconsistent as students have to exchange database files. Centralizing the data is important. And creating the application as web application allows the user to enter data from any web access point. The interface is not very user-friendly with Access forms. The end users of our application are also not very technical. Hence, designing the application as web application with centralized data solves the associated problems.

1.5 Platform Specifications – Deployment

1.5.1 Hardware Specification

Processor P III

RAM 128 MB

Minimum Space Required 10 MB

Display 16 bit color

1.5.2 Software Specification

Operating Environment Win 2000/XP

Platform .Net Framework & IIS Visual Studio 2008

Database SQL Server 2005

2. System Requirement Analysis

2.1 Information Gathering

For the development of this project I referred to the professors and students involved in research related to fisheries including Dr Craig Paukert, Associate Professor at Division of Biology. As the application should be convenient for the graduate research assistants I was provided with the details regarding important terminologies to be used in the application. The working of the application is made convenient and easy to use for the end user. Dr Paukert is involved in the research of Kansas fisheries and helped me to improve the user interface to a great extent as he is familiar with the view with which GRAs will look at the application. Dr Andresen, Associate Professor, CIS provided regular feedback on the project.

Other than this, I did a lot of research on various other methods of building this application which and was able to incorporate a few stronger features into the application. The tools and controls used in the application are recommended ASP.NET controls by Microsoft which improves the navigation and map generation to a great extent.

2.2 System Feasibility

The system feasibility can be divided into the following sections:

2.2.1 Economic Feasibility

The project is economically feasible as the only cost involved is having a computer with the minimum requirements mentioned earlier. For the users to access the application, the only cost involved will be in getting access to the Internet.

2.2.2 Technical Feasibility

To deploy the application, the only technical aspects needed are mentioned below:

Operating Environment Win 2000/XP

Platform .Net Framework & IIS

Database SQL Server 2005

Other Software: Arc GIS

For Users:

Internet Browser

Internet Connection

2.2.3 Behavioral Feasibility

The application requires no special technical guidance and all the views available in the application are self explanatory. The forms are designed to emulate the data entry forms in the old system.

3. System Analysis

Working closely with the Dr. Craig Paukert and Joanna Whittier and analyzing the requirements and functionality of the web application, I had three important diagrams by the end of the analysis phase. The ER diagram, data flow diagram and the state transition diagram which were a basis for finding out entities and relationships between them, the flow of information and the various states the application can have.

3.1 ER Diagram

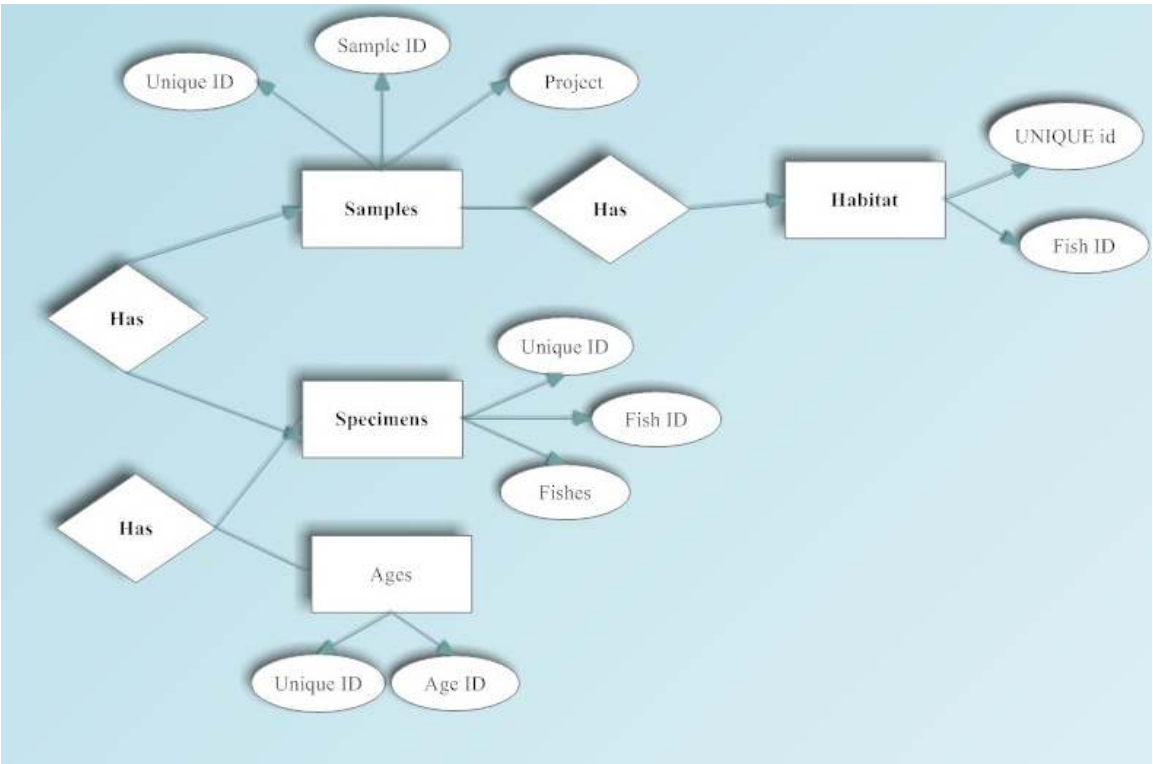

Figure 3.1 Entity Relation Ship Diagram

3.2 Data Flow Diagram

Figure 3.2 Context Level Data Flow Diagram

3.3 State Transition Diagram

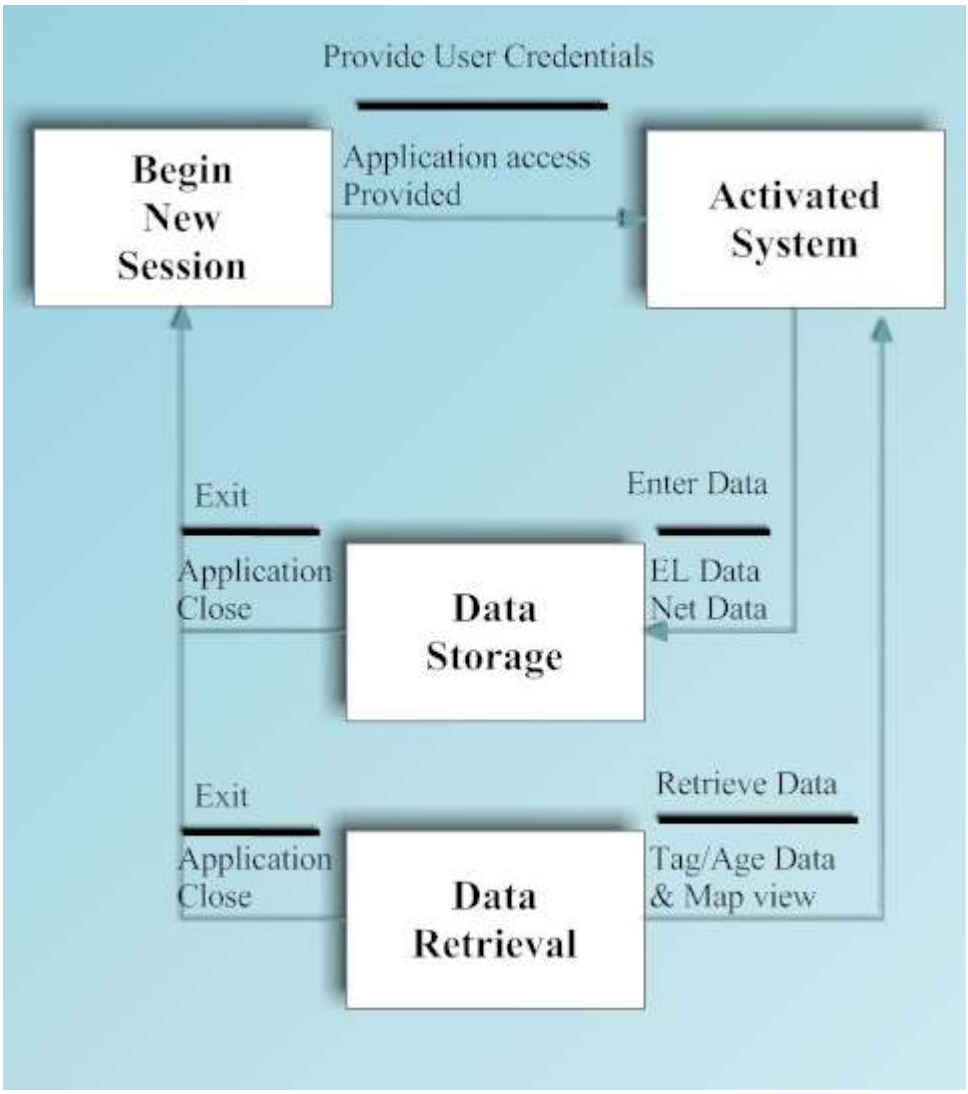

Figure 3.3 State Transition Diagram

4. Design

4.1 Design Goals

- The design of the web application involves the design of the data entry forms, query forms and web maps.
- Design of a structure which helps the user to navigate from one form to the other at any point of time.
- Design of interactive maps which should facilitate printing maps too.

4.2Architectural Design

4.2.1Architectural Context Diagram

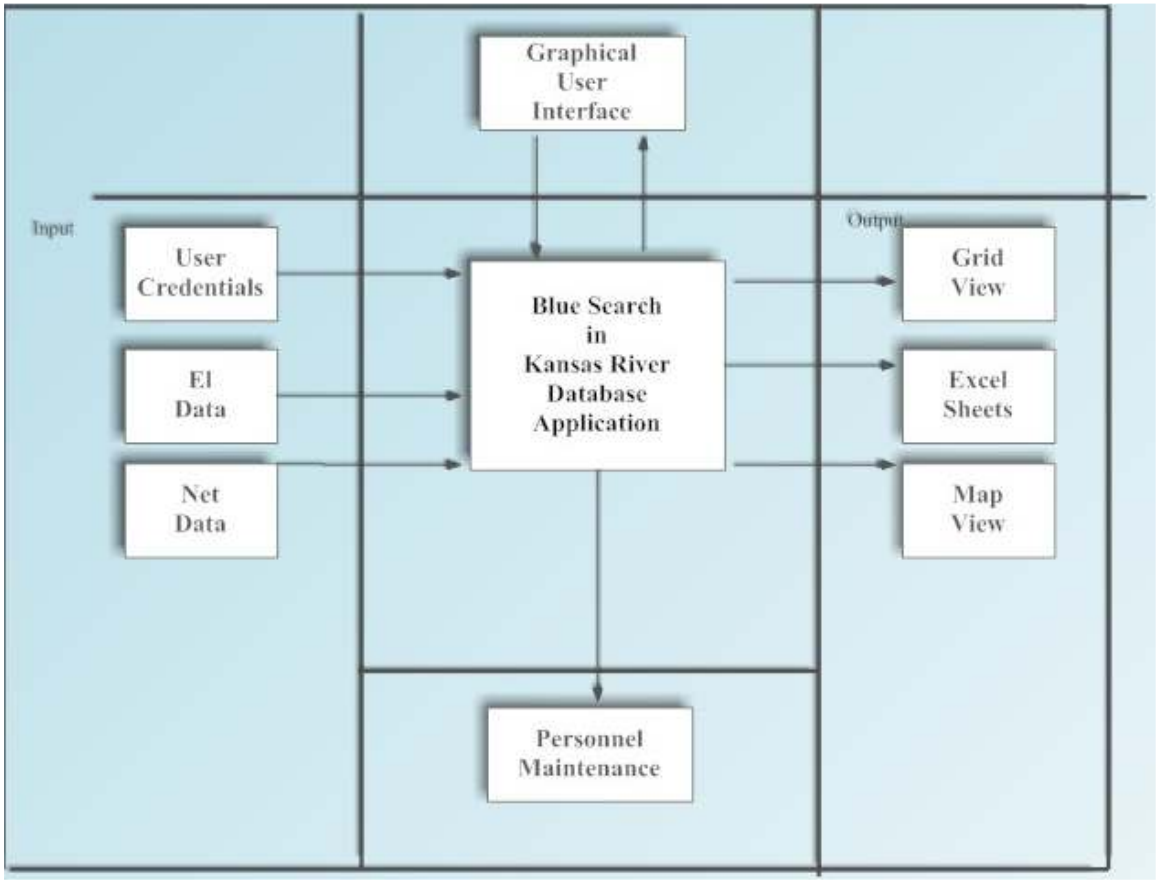

Figure 4.1 Architectural Context Diagram

4.2.2 Description of Architectural Design

In this context diagram, the information provided to and received from the 'Blue Search in Kansas River Database Application' is identified. The arrows represent the information received or generated by the application. The closed boxes represent the set of sources and sinks of information.

In the system, we can observe that the user interacts with the application through a graphical user interface. The inputs to the system are the user info information provided by the user and EL Data and Net Data forms filled by the user. Also, the output is in the form of grid views which are based on the parameters values entered by the user. There are some static outputs which do not change unless the project is reloaded; they are data which is shown in the maps.

Other than this, the maintenance of the system can be done by the administrator which can be addition, editing or deletion of few controls in the data entry forms.

4.2.3 Control Hierarchy

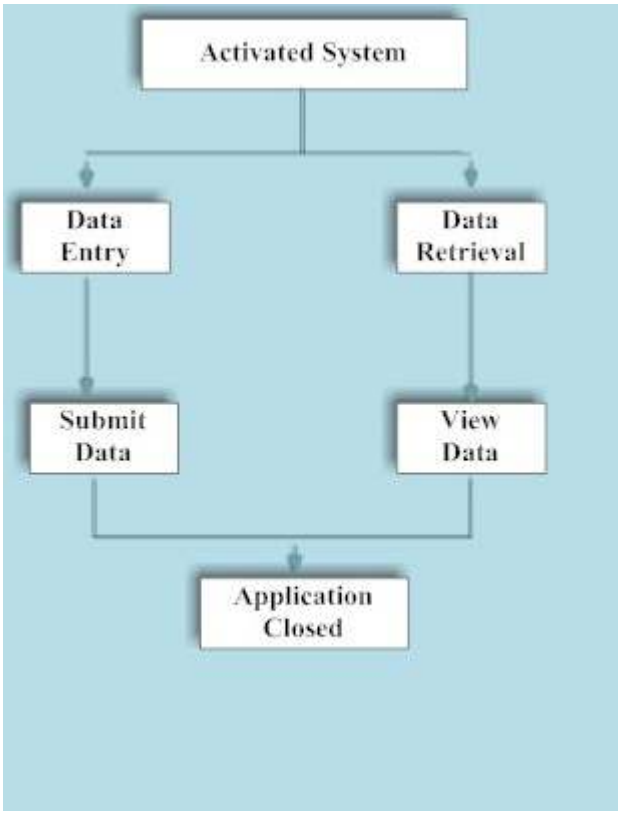

Figure 4.2 Control Hierarchy Diagram

4.3 Procedural/Modular Approach

Following are all the modules designed for the Blue Search in Kansas River Database application.

4.3.1 Initial Information Submission Module

This module starts with the generation of a session for the user and the session of the user is maintained until the end of the application. The session begins when the user enters valid credentials.

4.3.2 EL Data/Net Data Submission Module

The UniqueID is an automatically generated primary key by concatenating Project, Sample ID and Start Date values in the database and if any updates are to be made to the initial information they are to be updated using the same ID in the database but manually not through the interface.

4.3.3 Query (Species Searcher\CPUE) Module

The database is queried for information about recaptured fishes, aged fishes and also about the species and CPUE. Information about the recaptured fish is used when a fish is caught with the tag. The fish may be caught by any person. As the tag also contains a contact number, the person may contact the researches. The tag number and tag type are entered to retrieve the data about that particular fish.

4.3.4 View data on Maps Module

The database is also queried about the location of various species in Kansas River. Each species data is queried from the database and shape files are created from these queries. All these shape files are configured and viewed on the Kansas River map. The boundaries of the map can be changed to view the other river boundaries. Initially all the species are left unselected to speed up the drawing of the map. The user has to select the species to view their locations. The user can also zoom in/zoom out, pan, print the map and also user a magnifier to view the a particular part map closely.

5. Implementation

The design of the database was similar to the analysis phase. The database has been developed using SQL Server 2005.

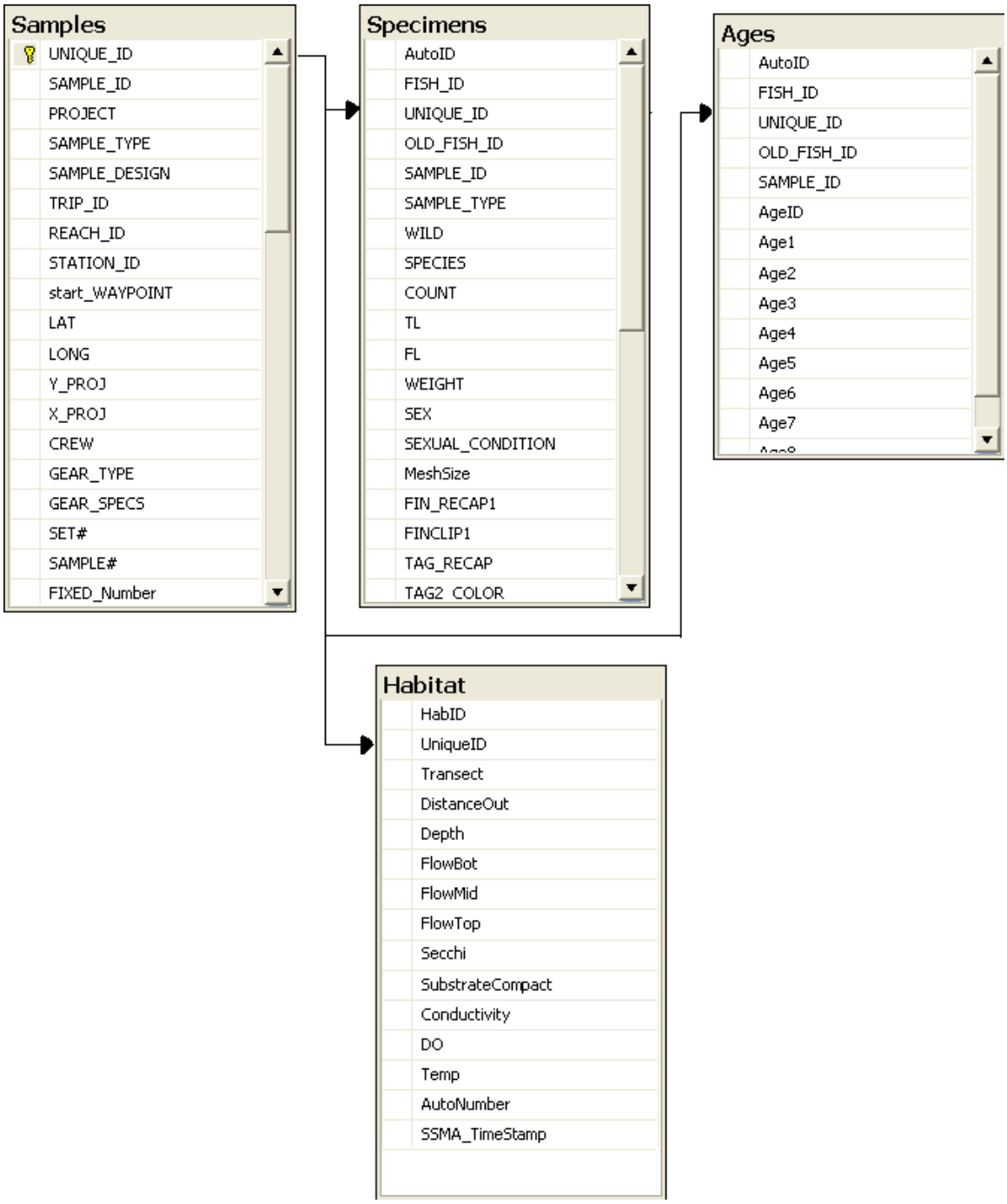

Figure 5.1 Database Implementation using SQL Server 2005

These are the main tables in the application and others are lookup and query tables.

5.2 User Interface Design and Implementation

The user interface of the application has been designed using Microsoft Visual Studio 2008 and Arc GIS. The main controls used in the design are Grid View control, Menu Control and these controls are provided with ASP.NET 2008. Following are the screenshots of the user interface.

Figure 5.2 Login Page

The user will login using this page. There are no roles in this application. And every user has access to all the forms. If the user is a valid user then the user is directed to Home page (Figure 5.3). The user is provided with the switch board in the home page where they can either select to enter the data collected in the field or query the data about a recaptured fish or about aged fishes. They can also choose to view the location of the fishes (Species) on map.

Figure 5.3 Home Page

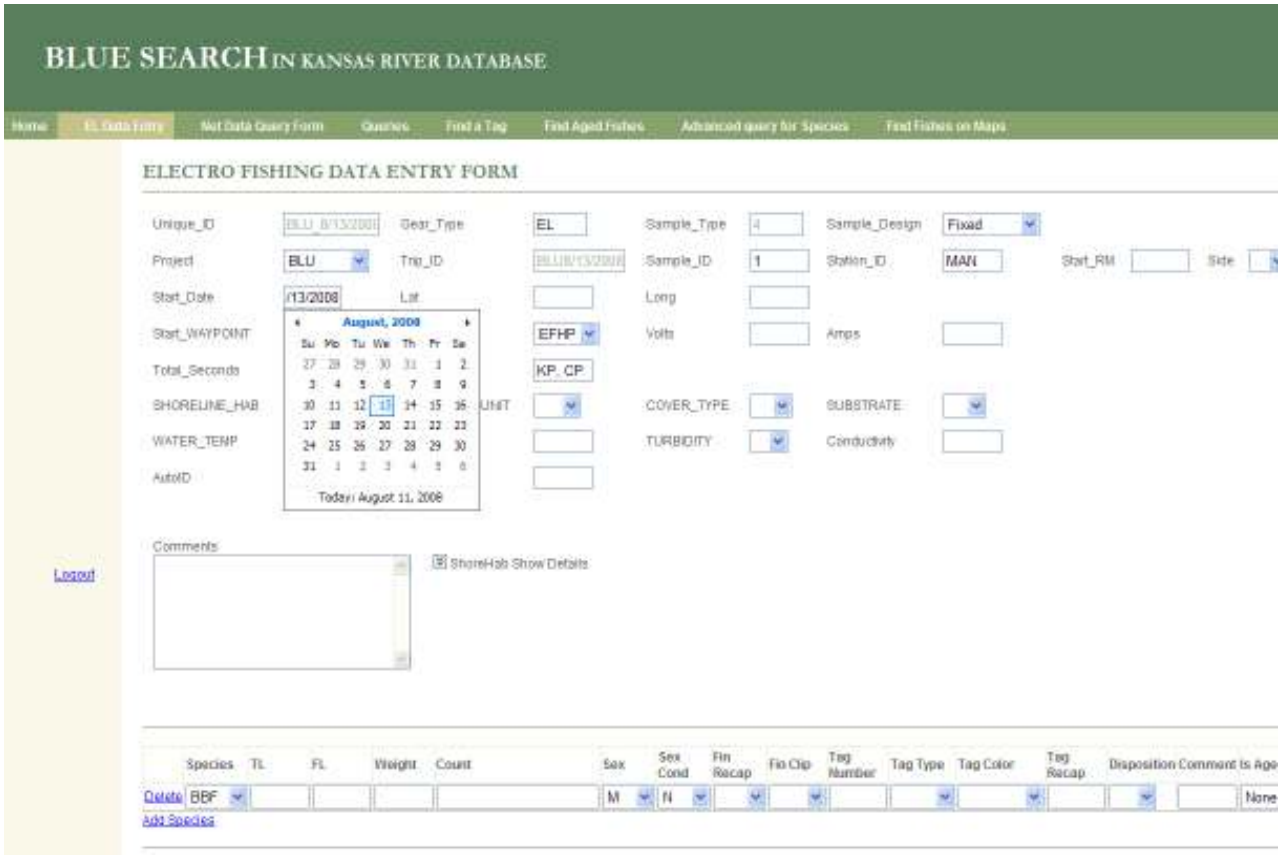

Figure 5.4 EL Data Entry Form

The EL Data Entry form is used to fill the data about fishes caught by Electro Fishing method.

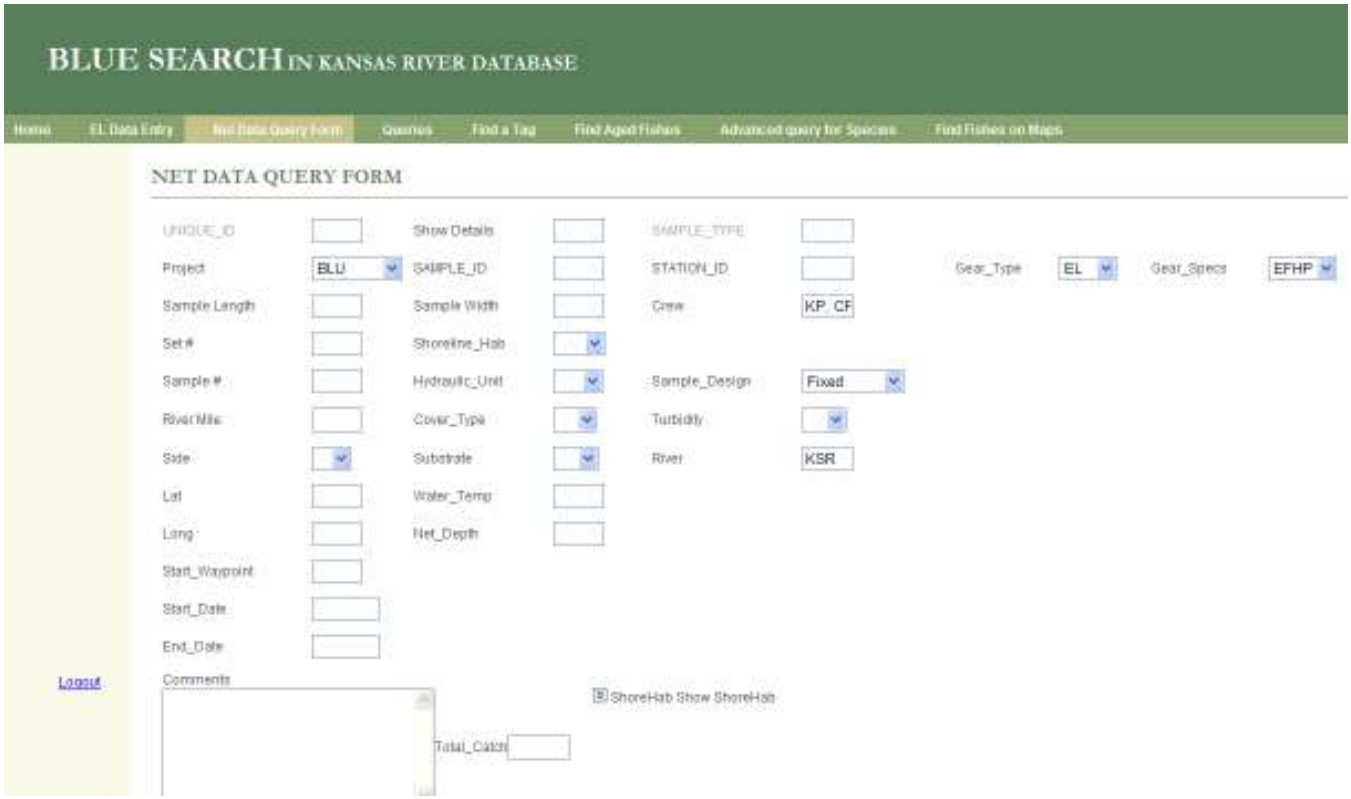

Figure 5.5 Net Data Entry Form

The Net Data Entry form is used to enter data about fishes caught by Gill Net. This is the second data entry form. These are the only two data entry forms in the application. The next forms are forms which query the database. The Species Searcher Form for CPUE table is used to calculate the Cost per Unit Effort. CPUE is a variable which is calculated by dividing the total number of fishes caught by total number of seconds.

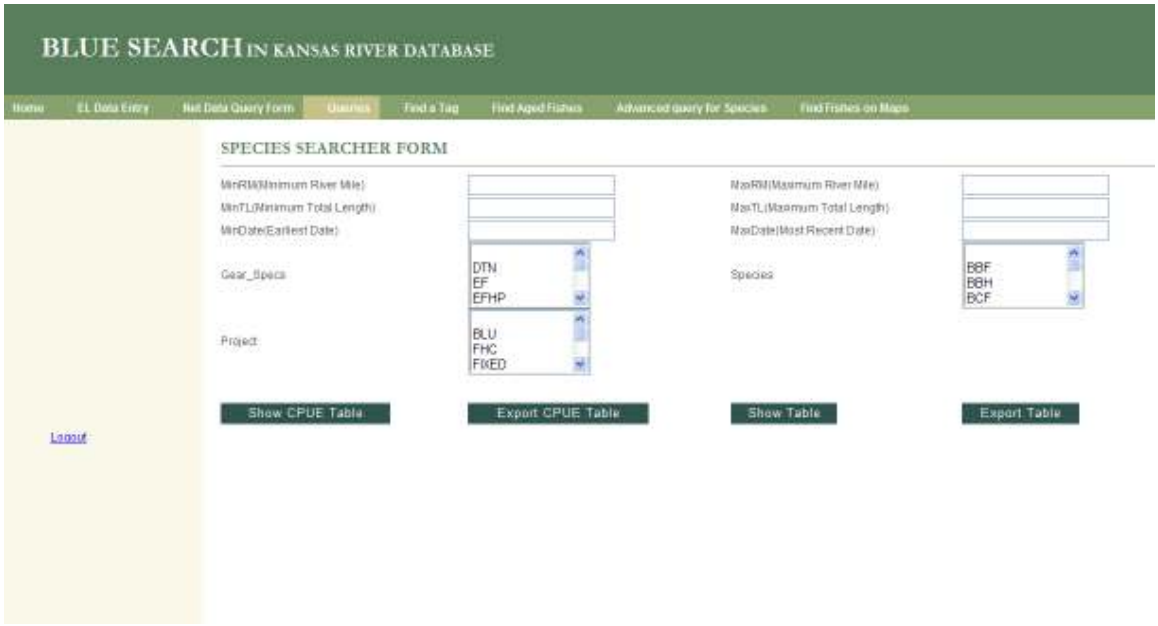

Figure 5.6 Species Searcher Form for CPUE calculator

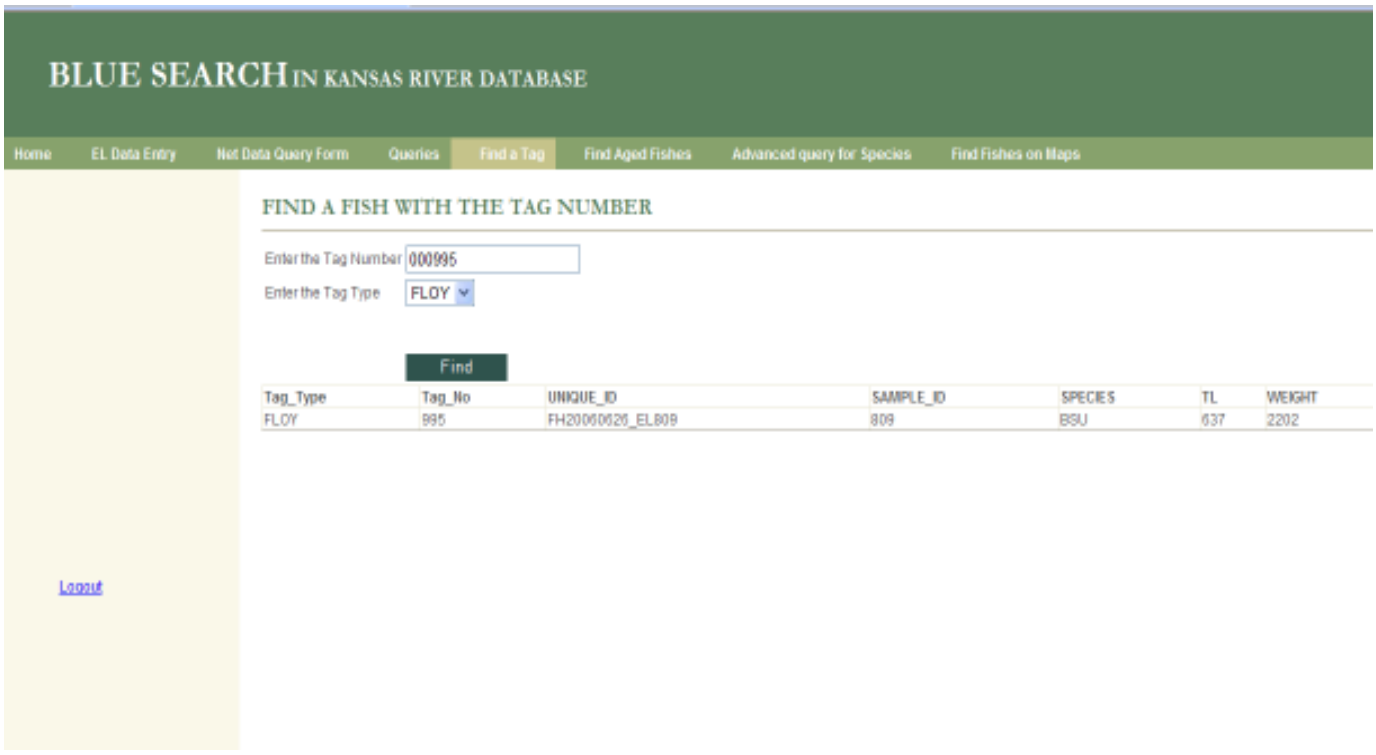

Figure 5.7 Tag Finder is used to get the details about a recaptured fish (fish caught with a tag).

The Tag Finder page is used to obtain data about a fish which is caught with a tag. The tag number and tag type are the parameters given to obtain the details about the fish. Tag number is the number written found the tag and tag type is one the two types, FLOY and FHC.

Figure 5.8 Aged Fish Finder

The above form is used to know the aging in fishes. The aging in fishes is calculated by reading the scales on the fish. The researchers use this data to know the growth in the fish.

The species searcher form is used to get the details of a particular species found between two dates.

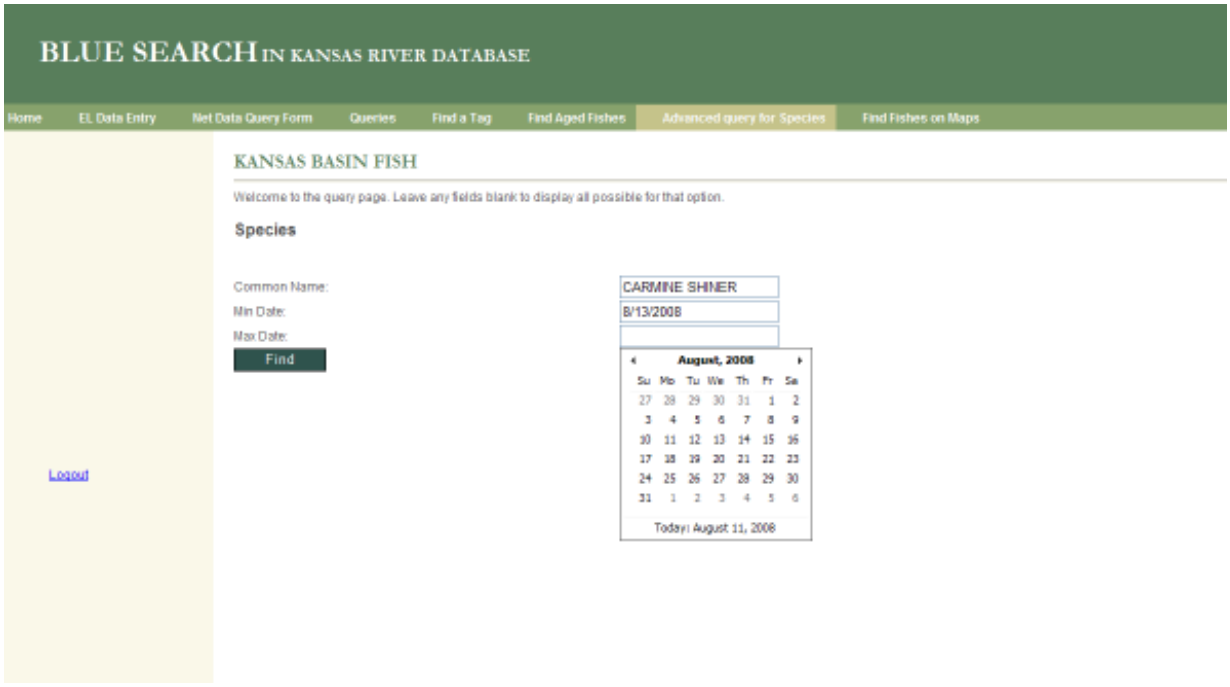

Figure 5.9 Species searcher

The Map for fishes in Kansas River page is used to analyze the data. This page provides you information about location of different species of fishes on Kansas River. This page is designed using ArcGIS software. The page allows you to view location of each species independently. It allows you to magnify the streams. It also allows us to change the borders of the map. One can also print the map. The results can be saved too. The data is stored in the form of shape files. These shape files are used to show data on the maps.

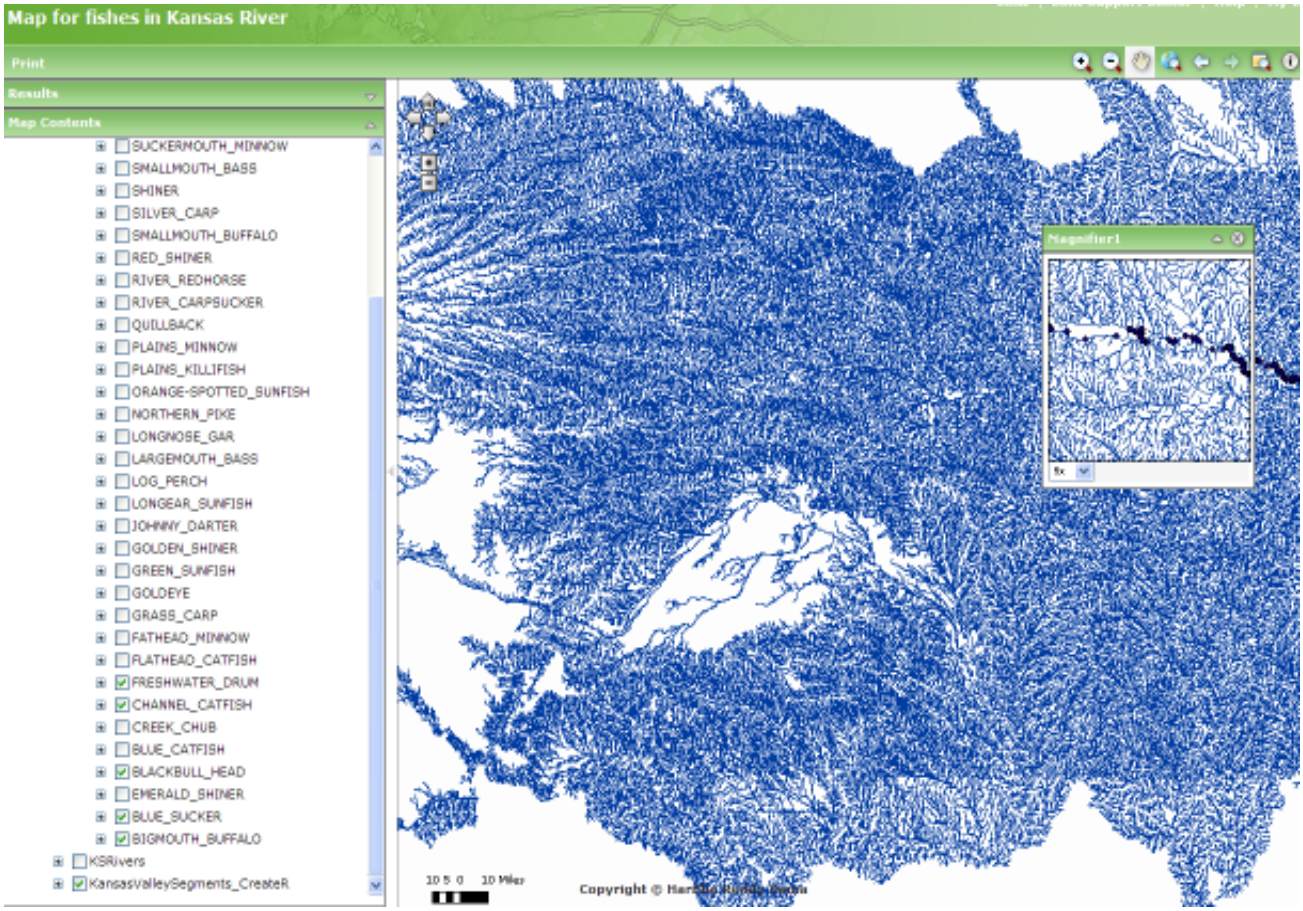

Figure 5.10 Map for fishes in Kansas River

5.3 Technical Discussions

The implementation of the database in application has a table named ‘Samples’. This database table consists of all the data collected in the form of samples in the field. But the user interface implementation still consists of few drop drown lists which are static and are not retrieved from the database. The reason for this redundant implementation is to improve the navigation between the different pages. If data is retrieved from the database, the speed of the navigation is affected and the navigation from one part of the application to the other becomes comparatively slower. As the

application consists of forms which consists of controls which use different tables, the slower navigation affects the performance of the application and increases the data entry time for the user.

Also, as per the specifications of the application from the client, those drop list values will not be changed in the near future. So keeping few dropdown lists static on the user interface was a satisfactory implementation to improve the performance of the navigation of the application. If in the near future, there are a huge data is added to those dropdown lists a better approach would be to retrieve the data from the database dynamically.

6. Testing

Software testing is a process of running with intent of finding errors in software. Software testing assures the quality of software and represents final review of other phases of software like specification, design, code generation etc.

6.1 Unit Testing

Unit testing emphasizes the verification effort on the smallest unit of software design i.e.; a software component or module. Unit testing is a dynamic method for verification, where program is actually compiled and executed. Unit testing is performed in parallel with the coding phase. Unit testing tests units or modules not the whole software.

I have tested each view of the application individually. As the modules were built up testing was carried out simultaneously, tracking out each and every kind of input and checking the corresponding output until module is working correctly.

The functionality of the modules was also tested as separate units. As we have mentioned, the user navigation module works along with the other modules, it was also tested independently without considering the other modules. The next button and the back button functionality was an important way to test as they fall under the separate cases in the logic. Each individual case was tested independently which made sure that each view of the application was working as a separate unit.

Also, as the user data submission module submits all the information gathered from the entry forms into the database, responses from each view were tested independently. As the entry forms were being implemented, it was made sure that data in the all the fields are recorded into the database. An individual submission with each question has been tested for the response capture.

The web map module has been tested as an independent unit. A user is provided with many selections options in order to view the reports. While configuring the map, each individual species was independently tested for its functionality. It was made sure that the desired data related to a particular species is retrieved from the database.

6.2 Integration Testing

In integration testing a system consisting of different modules is tested for problems arising from component interaction. Integration testing should be developed from the system specification. Firstly, a minimum configuration must be integrated and tested.

In my project I have done integration testing in a bottom up fashion i.e. in this project I have started construction and testing with atomic modules. After unit testing the modules are integrated one by one and then tested the system for problems arising from component interaction.

6.3 Validation Testing

It provides final assurances that software needs all functional, behavioral & performance requirement. Black box testing techniques are used.

There are three main components

- Validation test criteria (no. in place of no. & char in place of char)

- Configuration review (to ensure the completeness of s/w configuration.)

- Alpha & Beta testing-Alpha testing is done at developer's site i.e. at home & Beta testing once it is deployed.

Test Cases- I have used a number of test cases for testing the product. There were different cases for which different inputs were used to check whether desired output is produced or not.

1. Proper run time generation of the Unique ID.

2. Because same form is inserting data into more than one table in the database atomicity of the transaction is tested.

3. Duplication of data is not allowed.

4. Each entry form data is reflected to the correct Unique ID in the database.

5. Page is available by correct login only.

6. Back and Next button navigates the user to the correct and desired page.

7. Once submitted user should not be able to make changes.

6.4 White Box Testing

In white box testing knowing the internal working of the product, tests can be conducted to ensure that internal operations are performed according to specification and all internal components have been adequately exercised. In white box testing logical path through the software are tested by providing test cases that exercise specific sets of conditions and loops.

Using white-box testing software developer can derive test case that

• Guarantee that all independent paths within a module have been exercised at least once.

• Exercise all logical decisions on their true and false side.

• Exercise all loops at their boundaries and within their operational bound.

• Exercise internal data structure to ensure their validity.

At every stage of project development I have tested the logics of the program by supplying the invalid inputs and generating the respective error messages. All the loops and conditional statements are tested to the boundary conditions and validated properly.

6.5 Stress and Performance Testing

Performance testing is an essential element in successfully deploying a Web application. It's important to understand how the application and the Web server would behave as more and more users visit the Web site. In order to simulate that type of usage for a Web application, we would either need to coordinate with hundreds or even thousands of real users to access our Web site within a designated period of time or work with a testing tool that can reproduce such user loads.

Many Web performance testing tools are available to help. Basically, these tools allow us to use a minimal number of client computers to simulate a large number of virtual users, concurrently requesting predefined pages of the Web site. Each of these virtual users emulates the exact communication protocols between a real Web browser and the Web server.The tool that has been used in this application is called the Web Application Stress Tool - WAS from Microsoft. The test script was created which completes the data storage/retrieval and the settings 20 were changed to simulate 100 users hitting the site for 30 minutes and performance was tested with allow dial up bandwidth. The steps and the results have been demonstrated as follows:The concept behind WAS is simple: We can create a test script by capturing a browser session using Internet Explorer basically walking through our application, as a typical user would do. As we do this WAS captures the content of all these Web requests. WAS captures everything: Hyperlink clicks, Form submissions, Redirect links and everything needed to capture the user's session through our site. We can use the Browser Recorder (Figure 6.1) to capture a browser session and have WAS generate a test script from captured links. As you can see in Figure 6.2 there are a few options we can choose for the Browser Recorder: Capture delay, and record cookies and host headers. The delay between requests will result a more realistic test in terms of how people are actually navigating a site, giving us a more accurate picture of how users on a site map to connections on the Web server. When we click Next|Finish on the browser recorder we are whisked into IE and ready to capture requests in our browser.

Once we are done, we can switch back to the running WAS application in the background and click on the Stop Recording button (Figure 6.3).

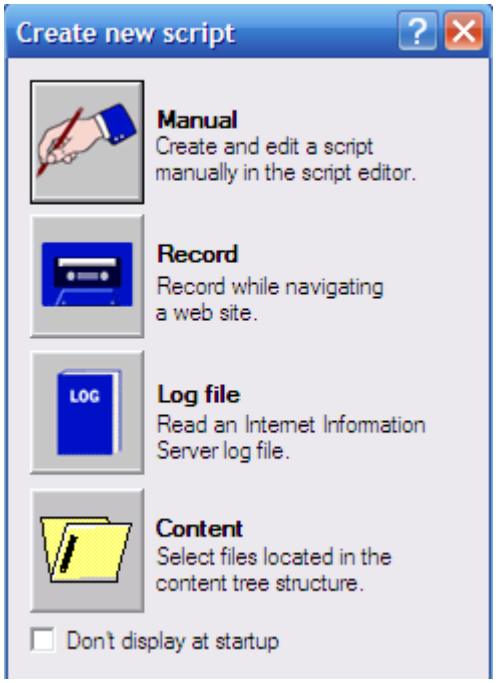

Figure 6.1 The Browser Recorder lets you use your browser to create scripts.

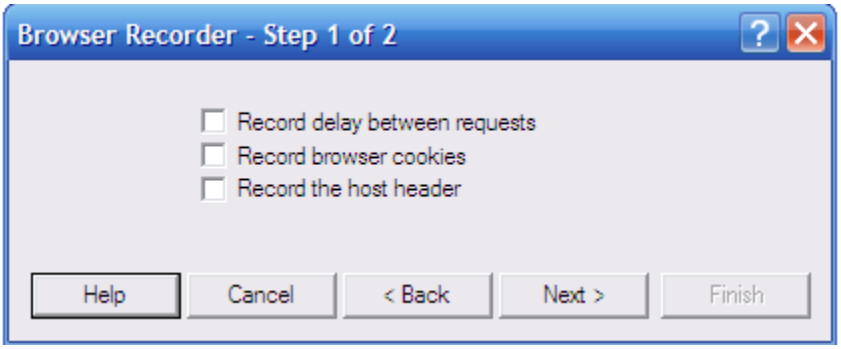

Figure 6.2 The recorder captures all links, HTML form submissions and even Cookies, Redirects and Authentication information.

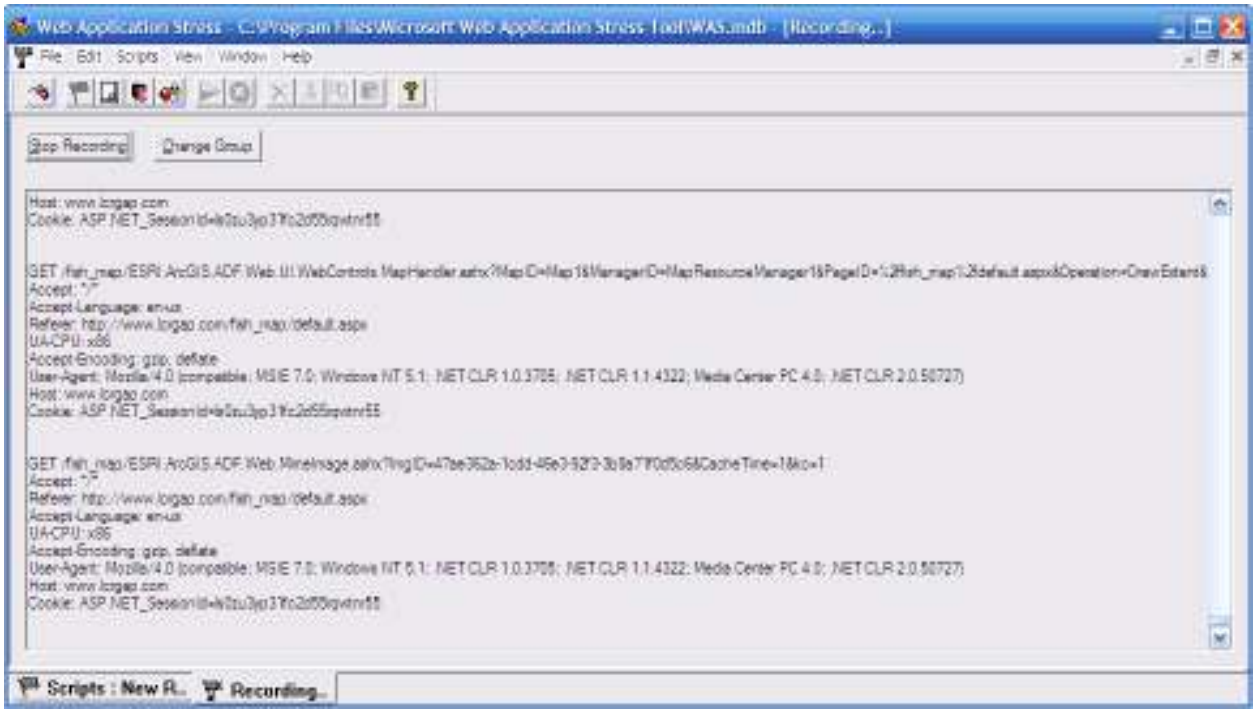

Figure 6.3 While recording a script WAS monitors IE to capture all incoming data and stores it into a database file.

If we look closely at the WAS form before clicking the Stop Recording button we can see how WAS is capturing the browsers progress. The data is captured and stored in Access (MDB) database file including any content captured from form variables.

Once we have captured the script we can see that our captured Web links are shown on the right in the data view of the main WAS window. With the links captured our next step is to configure the load options for running the script. We do this using the Settings option in the list and we can see a dialog as shown in Figure 6.4.

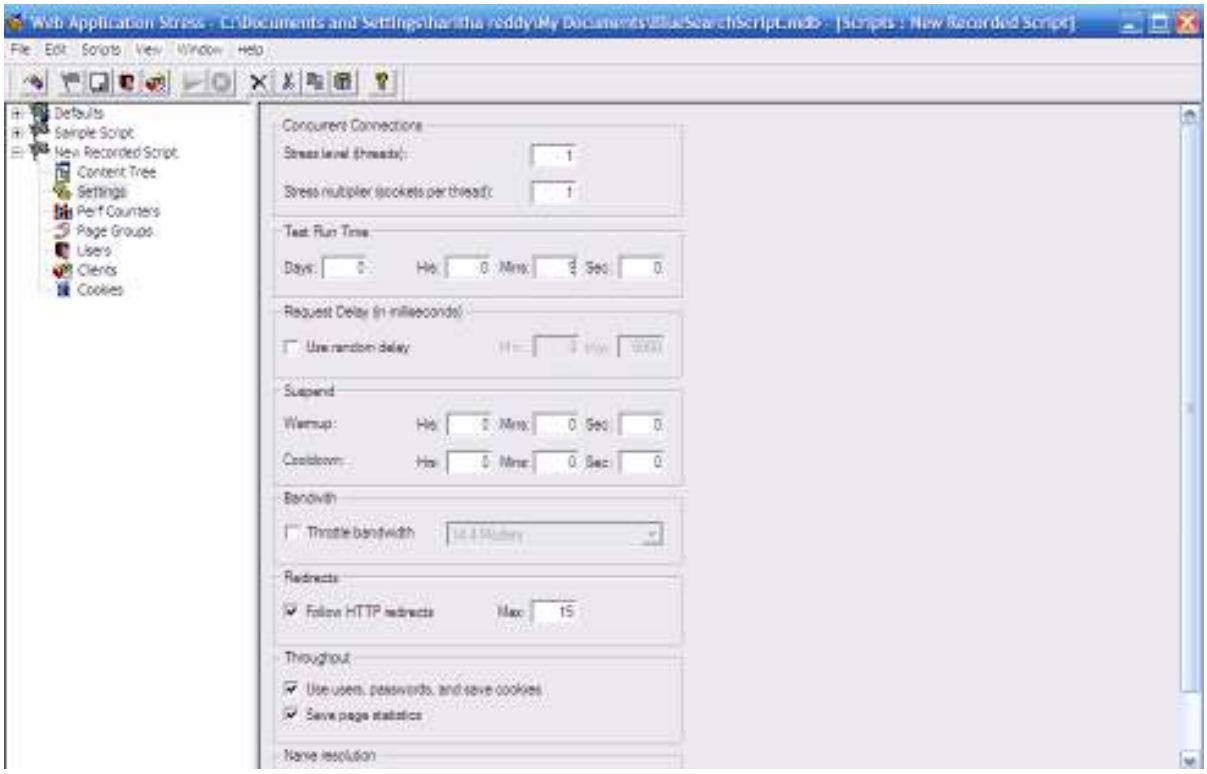

Figure 6.4 The Settings dialog lets us configure how the requests will be run against the server. We can specify the number of simulated clients by setting the number of threads and number of sockets on each thread – in this case 10 threads and 10 sockets yields 100 simulated clients.

The following report was generated by the WAS tool for testing the performance.

Overview

==========

Report name: 8/9/2008 7:35:59 PM

Run on: 8/9/2008 7:35:59 PM

Run length: 00:30:21

Web Application Stress Tool Version: 1.1.293.1

Number of test clients: 1

Number of hits: 22123

Requests per Second: 12.29

Socket Statistics

Socket Connects: 22220

Total Bytes Sent (in KB): 12264.92

Bytes Sent Rate (in KB/s): 6.81

Total Bytes Recv (in KB): 47649.89

Bytes Recv Rate (in KB/s): 26.47

Socket Errors

Connect: 0

Send: 0

Recv: 0

Timeouts: 0

RDS Results

Successful Queries: 3

Script Settings 29

==========

Server: localhost

Number of threads: 100

Test length: 00:30:00

Warmup: 00:00:00

Cooldown: 00:00:00

Use Random Delay: No

Follow Redirects: Yes

Max Redirect Depth: 15

After running the test script with 100 concurrent users and for 30 minutes, there were no performance issues and no socket errors and all essential details are obvious from the above observations generated from the WAS tool.

7. Results & Challenges

The application is being used by researchers at Division of Biology Kansas State University to store\retrieve and also used to analyze data from the maps.

7.1 Challenges

- Understanding the client requirements is the major concern as the terminology which they use is completely new to me. They have different kinds of problems entering the data, like they need different forms for different projects handled by the students.
- Dealing with large database, which has more than 50,000 records.
- Next challenge is to learn new software called ArcGIS and use it with C# for creating maps.
- Creating the forms and databases keeping in view that this application will be used at least for 4 more years and even more depending on the students/projects in future.
- The last but important challenge is to setup the application on the server and get it running. Because I was always involved in coding the application but was never into bringing up the application live.
- Learning technologies with little/no guidance.

The overall idea of doing this project is to get a real time experience. Learn new technologies.

8. Conclusions

The 'Blue Search in Kansas River Database' is designed to provide a platform through which researchers can work on a single database and from any place. I hope that the information stored through this application will be beneficial for a lot of data analysis by the researchers. Also, there can be a significant analysis of data from the maps like the concentration of fishes on a particular river mile or about the endangered species etc.

8.1 Limitations

There are two minor issues in the application which are not resolved yet. One is about the HabID in the Habitat Table and about the query to calculate Cost per Unit Effort (CPUE) from the effort. The need of the Habitat information is not decided yet. The other problem is that the current database does not have total seconds or total catch information for certain samples. This will force division by null which will throw an error.

8.2 Scope for Future Work

The following things can be done in future.

- The application can be modified to resolve the above issues.
- The map module can be designed further to avoid the slow drawing.
- Grouping of the controls can be done to save space on the data entry forms.
- The application can be modified so that it can be used by other students/researches working on other river basins.

9. References

- All about ArcGIS

  http://www.esri.com/software/arcgis/

- All about Microsoft controls in C#

  http://www.msdn.microsoft.com/

- About AJAX Controls

  http://www.asp.net/ajax

- How to use and install Web Application Stress

  http://support.microsoft.com/kb/313559

- Wikipedia for various diagrams & testing methods

  http://www.wikipedia.org/