Spring 2005

Development of Database and Web Site for D3Multisport

Jay Garrison
Regis University

Regis University
School for Professional Studies Graduate Programs
Final Project/Thesis

Disclaimer

REGIS UNIVERSITY

SCHOOL FOR PROFESSIONAL STUDIES

MASTER OF SCIENCE
IN
COMPUTER INFORMATION SYSTEMS

Development of Database and Web Site for D3Multisport

Facilitator: Mike Nims

PROFESSIONAL PROJECT THESIS

Jay Garrison

September 27, 2005

Acknowledgements

I would like to thank my wife Janice for believing in me and sacrificing many weekends while we stayed home so I could study on all those gorgeous sunny Saturday afternoons.

Abstract

Before this project, D3Multisport.com did not have a software application to track and monitor personal and progress information on athletes training for triathlon events. The previous process was manual, and although it helped provide a training schedule for an athlete it required the data be updated by hand. The process of developing a training program on an excel spreadsheet provided a good basis for the athlete to start the program, but was not suited to make dynamic adjustments if not optimally suited for the athlete.

The application created in this project was designed to keep a coach updated on the progress of an athlete. If necessary, the coach can make modifications to the athletes training program in order to work on any weakness the athlete may be showing. It is important that both coach and athlete have access to updated information related to the training program. The previous process did not provide this requirement because the coach and athlete were not present with each other at all times. Another problem with the previous process was that both coach and athlete might wish to add training criteria to the training program. The application now provides this functionality.

The initial prototype used Microsoft's Access technology with Data Access Pages (DAP). This ensured the web site had the necessary work flow the users were expecting and allowed them to update information to a database. Once the prototype was approved, Microsoft .NET ASP technologies were used to create the web site. The Access database was reverse engineered into a Microsoft MS SQL Server database. Both the coach and athlete are now able to access the

same database via a web browser. A central computer acts as the web server providing access to the central database and downloads web pages. This enables the coach to keep abreast of the athlete and provide updated real-time progress information. Providing a web interface to a central database allows the athlete to record current data on a daily basis and give the coach the ability to analyze this data and make any changes necessary, keeping the athlete on track.

Table of Contents

List of Figures and Tables

Chapter 1: Introduction and Background

Introduction

D3Multisport.com was looking to take advantage of the knowledge base they have for coaching endurance athletes and turning that knowledge into a system that can be leveraged for the purpose of creating additional revenue streams. This system needed to provide software to help both coach and athlete manage and monitor progress within the training program.

The system is specifically geared towards the industry of training triathlon athletes and allows a coach to monitor and consult multiple athletes at the same time. With the ability to log onto remote web sites from almost anywhere these days, including cell phones, the athlete can give immediate results of a training cycle. The results are analyzed and the coach can determine whether the training program needs to be continued as designed, or modified to correct athletes' weaknesses and exploit his strengths.

The technologies available today allow D3Multisport to integrate their knowledge base into a working online system available for all who are interested in being involved with a well developed and monitored training program. This allows a seasoned trainer to help guide an athlete in a real time environment geared to maximize training potential.

This project designed and created a database, then provided a web interface to it so both coach and athlete had access to real time data provided by

the athlete. The coach can make changes to the training program after analysis and can then be carried out by the athlete.

Business Need

It is important that both coach and athlete have access to current information related to the training program. The old process did not provide this requirement because the coach and athlete are not with each other at all times. Another problem with the old process is that a coach may wish to add training criteria to the program. If the athlete has not obtained a copy of the latest training criteria, they will not know of the change.

Athlete and coach relationships have traditionally been either very personal, or very passive. In a traditional training environment, a good coach typically focuses on athletes who have the greatest potential and desire to listen and do as they are told. This, unfortunately, does not allow the less fortunate access to the best coaches available in the sport. In order for a coach to properly train an athlete to their fullest potential, they need to observe and monitor the athlete's progress in real time.

A triathlon event requires competing athletes to run, swim and bicycle various distances in sequence. Although training may initially take place in pools, running tracks and stationary bicycles, the most effective training for a triathlon needs to take place in the real environment. This environment has traditionally been outside, where athletes are swimming in lakes, oceans or rivers, running on streets, over hilly terrain and mountain biking. Obviously, most coaches can not realistically run, swim and bike around with their athletes, but are focused

primarily on results and progress. If the athlete is excelling at running and biking, but is not up to speed in swimming, the coach may change the workout program to include more aggressive swimming workouts. The coach could also incorporate weight training to make the athletes upper body stronger to enable a stronger swim stroke. It is known that all people will progress at different rates determined by initial desire, effort level, and natural abilities. D3Multisport has many years experience at creating training programs for endurance athletes. By analyzing a long list of questions answered by the athlete, experts can put together the best possible initial training program to fit the athlete's personal abilities. With a progress reporting system in place between coach and athlete, the results can be monitored and analyzed to determine whether the program needs to be modified in a way that would benefit the athlete.

The business need for this type of online system is obvious. The more athletes a single coach can monitor and coach at a time, the more money D3Multisport can make per coach. Not only will a coach be able to analyze the results of more athletes at a time, but will also have the means to monitor and analyze each athlete's results in a real time fashion. This was not possible before, even if the coach had only one athlete. Unless, of course, this was a very involved coach that trained alongside the athlete. This system will increase company throughput and quality that will eventually affect the reputation of D3Multisport in a positive manner.

Review of Existing Situation

The old process for creating athlete training programs and schedules was done manually, which was not ideal for ongoing training and development. The athlete was asked to fill out an extensive questionnaire that gave the coach a better idea of the athlete's abilities, confidence and commitment to endurance training. This questionnaire was filled out on a piece of paper that was most likely copied, filed away and dispersed to the proper individuals in order to create the best possible training program. Although these initial athlete credentials remain the same, the athlete continues to grow during the training process by increasing both physical and mental abilities that may not reflect the initial athlete analysis.

Athletes training for triathlons are very active individuals, and their training programs often put them in remote locations. Coaches usually have many athletes in training and cannot always be at every location. The old practice only informed a coach of an athletes progress days, and sometimes weeks, after the athlete had been training. Critical changes to the athletes training program cannot be made until after the coach has analyzed data recorded by the athlete. Providing a web interface to a central database allowed the athlete to record current data on a daily basis and gave the coach the ability to analyze this data and make any changes necessary to keep the athlete on track.

Goals of Project

The system must provide the following primary goals:

- Ability to enter athlete demographics into system by athlete or staff.
- Enable the staff to create and modify specifics of main workouts.

- Effortless functionality for coaches to create and upload training programs
- Accessibility for athletes to view and log workouts from remote locations via Internet.
- Easy for coaches and athletes to make comments about workouts.
- The system must be simple to manage and maintain.
- The system must be staged for growth and additional upgrades after initial development.
- Site will be easy to manage so that it can be kept up to date without complex changes.

Creating a web interface to a database provides these goals and allows the staff, coach and athlete access to the same database via a web browser. A central computer acts as the web server. This provides access to the central database and downloads web pages allowing the coach and athlete to interact. By providing updated real-time information on the progress of the athlete, the coach can keep abreast of what the athlete is expected to do.

Barriers and Issues

The initial barriers apparent in this project were the business logic and how it was portrayed and used in the application. The developer had no Triathlon training experience and relied on knowledge from external multi-level sources. The levels of knowledge transfer were from the client, D3Multisport, to the facilitator, Mike Nims, then to the developer. The geographical locations of each

require that remote communication including phone conversations, emails and documents. The initial documents and talks with the facilitator were efficient to get started, but without interactive suggestions and question/answer sessions, there was the danger the application would not get off on the right track and go astray.

The tools required to create this application was beta Microsoft software. This software was purchased through the Microsoft Developer Network (MSDN) Subscription program. MSDN provides all of Microsoft's software to developers wishing to learn their products. Microsoft changed their subscription process and it affected the date the software was obtained for this project. Many hours were spent on the phone in order to expedite the delivery of this software.

Project Scope

Most ideas for a new system begin as a simple solution to make life easier for everyone; then they start to take on a life of their own. Scope creep can create unnecessary work and distract from the original idea. To combat this, the following guidelines were instituted to insure the project met established goals that had an acceptable return on investment.

- Keep it uncomplicated through the design and build phase. By using Microsoft Access as a prototype development tool the project was kept simple and straightforward. During the Implementation phase, it was decided to use SQL Server and .NET ASP to expand the project into a full production product.

- Although time and money were constraints to the project, cost was a consideration. This was a high priority to insure that the product was meeting acceptable goals.
- Review cycles of the project had to be met in order to continue. These review periods were scheduled at certain points to insure the application was working as designed and requirements were met. This reduced the number of cycles required to go back and correct features and functionality.
- Develop the system in phases to obtain feedback from the client on process workflows and look and feel.
- Define the first stage of completion so both the developer and client knew when the first stage of the project was complete.

Summary

Now that the business needs, goals and scope of the project were understood a proper analysis could start. It was impossible to obtain all project variables and business rules during the first stage. It was, however, important to exhaust the process in order to gain knowledge and understand how the client wished to improve it. This was one of the most critical parts of the project because it defined the processes that were to be designed and developed. Any misunderstanding, or sections left out, affected the design process by having to come back to this phase and define it again. This also affected existing processes of the design that related to the faulty piece of the system.

Chapter 2: Research & Methodology

Research

- **Client Interviews**

  The initial research for the project involved interviews with the client to establish what the application was to provide. The main goal during the interview process was to establish guidelines for the major phases of development. This is where the requirements of the project were established to determine what the application absolutely needed and could not function without. The portions of the application not critical to the initial phases of the system were put aside for a later revision.

- **Literature**

  Since the developer was not an expert in the technology used to create the prototype and the finished application, there was an enormous amount of book reading and research necessary. Please consult the Resources section at the end of the paper for detailed resource information. The main topics were primarily Microsoft based: Access Data Access Pages (DAP), Access database, SQL Server, ASP 2 .NET, Internet Information Server, Network/Router configuration, Web Design, Web and System Security. There was also much research performed in the way of project management techniques to be used in the management of the project.

- **Online Courses**

  Access to online courses was available for download and viewing on the MDSN web site. The demos provided a graphical and audio solution to

understanding and absorbing the vast knowledge necessary to get a start on designing and developing the web site from scratch using Microsoft technology. The courses provided a way to test existing knowledge to know how far along the developer's knowledge was actually coming. The list of online demos and courses is provided below in the Resource section of this paper.

Review of Existing Solutions Available

There were a few available solutions available that D3Multisports was considering. One in particular was Enhance Sports. They provide software that provides a canned solution that D3multisport coaches and athletes could interface through. This, however, does not provide the training curriculums that D3Multisport intended to use and only allows athletes to work from a non-updated training program. The best, most dynamic and scalable solution was to design and create a web site from scratch, using the foundations and ideas laid out in this thesis.

Contribution the project will make to the company

The results of this project provided an interactive web site that integrated the critical business knowledge of the company. This new functionality allowed the business knowledge to be extended to the Internet. thus giving company coaches, athletes, and customer's access to important athlete progress information from anywhere there is access to the Internet. Coaches have the ability to monitor the progress of multiple athletes at a time, which allows the coaches to increase revenue for the company.

Discussion of Technology Considered for Project

Basic Web Design Process

Many technologies were researched to provide a professional foundation and framework for the project web site. A main requirement included the application be scalable, user friendly and usable. In order to accomplish this, the technologies needed to be well understood so they could be applied correctly. The following paragraphs describe the results that were applied to the design of the web site and summarize the philosophy used to create it.

The design of this web site was part art and part science that required insight, experience, good taste and common sense to complete. The science part required the designer to evaluate, measure and select the proper design methodologies to create a usable and extensible web site users could find useful and enjoyable.

The architecture of the web site contributes a great deal to the long term development and evolution. Starting small and defining core functionality enabled the design and development of the site before it became obsolete. Incorporating an incremental development approach from the beginning enabled the development process to evolve throughout the web site life cycle. Early attention to the architecture and framework was important for the initial design and will help designers envision problems and potential weaknesses that might arise in the future.

The beginning of this web site involved complete understanding of the customer’s requirements. Even the best designed web site is rendered useless if

the customer has little or no use for it. A good deal of time was spent up front before the design process began. To gain knowledge of holes that might exist after the customer presented their case, two things were essential: empathetic listening strategies, and asking good questions. Only after the designer and customer agreed on the deliverables could the designer begin to intelligently formulate an architecture and design for the site.

The fundamental architecture design issues considered for this web site project were:

- Performance and Scaling
- Reliability
- Transactions
- Managing state
- Security

Performance and scaling considerations involved both latency and throughput of the overall system. Any piece of the system that was not up to par could slow the whole system down to unacceptable user standards. From a hardware standpoint, the important pieces are CPU speed, RAM size and disk size should all be connected to the highest speed network possible, providing the highest bandwidth to transfer data. The database will eventually require a high volume of transaction processing, so a dedicated system could be justified in this case. This will provide a dedicated system for users requiring process intensive requests, leaving the web server available for users that only require a peek at a document

or fill out an online form. The size of these components was determined by estimated predictions of how many users will access the system and what type of processing requirements they will require within the life time of the site. For this project, all components will start out on the same computer and scale to multiple servers as throughput increases.

Reliability issues consider the danger and severity caused by loss of access or damage to the web site. This site is the central means for tracking athlete progress and dynamically modifying training schedules, so reliability can be severe if it fails. The key to reliability is redundancy. If any critical piece of the system fails, or is compromised, it should have a fail over mechanism that allows the transfer of responsibility. This redundancy involves mirrored RAID arrays with fail over capabilities, database replication on separate systems and even multiple ISP carriers that are on separate hubs. This is a suggestion that hopefully D3Multisport will incorporate along with an adequate backup system.

Transactions are an important design issue in regards to ensuring business processes and transactions are completed. Atomicity is a key concern within a database transaction. If an error occurs, (the system crashes or the network goes down), while a database engine was in the middle of a row insertion; the database has various means to restore the system back to the original database state which can then rollback back to the beginning of the transaction. If the transaction has successfully completed, the database commits the transaction, letting the system know that a full logic cycle is finished. The design of this web

site considered each system component in the same manner. The site functionality determined the level of concern in regards to process atomicity. In the case of this system, atomicity was taken very seriously. The system had to ensure each component within the business process completed successfully before moving on to the next component within the process. For instance, if a coach determined that one of his athletes needed to modify their program, and was in the middle of modifying it when the system crashed, it would be important that the system roll back to the original data, or the last save, rather than have left the data partially modified where the coach was forced off.

Managing the state involved determining where to place the different components of the system. This web site consists of a database server, application server and a web server as the core internal system pieces. Each one of these pieces can house data that provides business information to various components throughout the system. The other area to place information is on the client. Client side information can be stored as XML files, cookies, ini files and various other forms of client side storage that include embedded databases contained in downloaded software. There are pros and cons to placing data and business logic on system components that need to be carefully considered by the web designer. For the sake of this project, the database and web servers were housed on a single computer and all data was stored in a database. XML files may be used in the future to store data on the client side when the athlete feels the need to log information when not possible to get online.

Security was an important component taken into consideration when designing this web site. Aside from the many ways hackers can enter the system via the network; the system was designed to secure important and sensitive data from such an individual in case they are successful at breaking into the site. The degree to which security is implemented depends on the content itself. This web site has users that have different defined levels of security. The design needed to consider what areas of the system each security level allowed a user access to. It is important to enforce the user to logon using the correct userid/password combination in order to gain access to the contents of the web site. Besides userids and passwords, the interaction of system components was very important. The designer needed to know exactly how and what information was being passed from component to component within the business process. Information that was once secure on a previous system component might not be secure on the component it was just passed to. The new ASP .NET framework comes with a complete web site security application that easily plugs into any web site developed in .NET. The administrator logs into a web site to add users and grant privileges to groups and users and is easy to use. Both athlete and coach have access to the system, with the coach having access to multiple athlete profiles. This is accomplished by assigning each user to a coach, administrator, or athlete group.

By carefully considering and implementing these fundamental design issues the following issues were accomplished:

- Enhance the chances of creating and deploying a web site that meets the needs of the user
- Perform and scale according to a natural growth curve,
- Be reliable enough for the user population to tolerate,
- Carry out complete business processes and let each communicating component know if it was successful
- Contain data files and other information in the appropriate system component and keep the system and data safe from unauthorized access and harm.

Now that the general design issues have been discussed, we will now describe the technologies used in the project in greater detail.

Data Access Pages

Data Access Page (DAP) research was necessary for the development of the prototype, which was developed using an Access database and DAP. The following is a bit technical in nature but summarize the research that allowed the prototype to be developed.

DAP allows the creation of data-bound HTML pages within an Access user interface that can be viewed in Microsoft Internet Explorer. DAP has features that allow a user to perform data browsing, data entry, reporting, and analysis from a Web browser. DAP can bind to data from Access or Microsoft SQL Server databases.

DAP is built upon several technologies and components. The most important component is the Microsoft Office Data Source control (MSODSC). The MSODSC is part of the Microsoft Office Web Components suite that allows displaying, editing, and the analyzing of data. The primary functions of this Data Source control are:

- To connect to data sources
- To build and execute commands against data sources
- To retrieve and bind results commands to elements on the page

The Data Source control is an ActiveX control that is not visible at run time. The MSODSC is actually a property of the DataAccessPage object that returns the Component DataSourceControl object (another Office Web Component), which represents an instance of the MSODSC that binds the page to the source data. When a new data access page is created in Access, it inserts an <OBJECT> tag to define the Data Source control in the HEAD element of the data access page. The ID attribute of the <OBJECT> tag, created for a data access page created in Access, is always set to MSODSC.

A data access page involves the interaction of the Data Source control with a set of DIV elements (*used on an HTML page to enclose a block of HTML elements*). These elements act as containers for labels, controls and data-bound elements that make up the page. A Data Source control uses functionality from the Dynamic HTML (DHTML) Object Model, which is implemented by Microsoft's HTML parsing and rendering engine (MSHTML), ActiveX Data Objects (ADO) and an interface to Microsoft Data Access Components (MDAC). The Data

Source control integrates with HTML elements, and contains the HTMLContainer property of the Section object. This enables script to access the collections, properties, and methods of the DHTML Document Object Model. Other common elements that make up a typical data access page are intrinsic HTML controls and data-bound SPAN elements. User interactions with data access pages are events that happen within the context of an HTML document.

There are two main approaches to developing data access pages. The approach used to create the prototype used the VBScript scripting language in the data access page design environment to control the page's behavior at run time. Data access pages are HTML-based, so script code can be embedded within of the HTML and provide runtime functionality for end users; this is the more common of the two approaches.

There are two kinds of data access pages which vary depending on whether the page uses hierarchical grouping. These are Simple pages and Banded pages. A simple page displays the fields from a single record and has no repeated records or hierarchical grouping. Although a simple page has no grouping at run time, the underlying data model sets the property GroupLevels.Count=1. A banded page can display multiple records within different bands that are repeated down the page. A banded page either has no hierarchical grouping, or can display a hierarchy of grouped records at once.

The design of a data access page has up to four kinds of sections:

- **The Caption section** is the first section of the page and is used to display column headings for fields that are displayed in the header section. The

caption section can contain any type of control except data-bound controls.

- **The Header section** is next and is used to display data in the data bound controls and displays any calculated values. A simple page would only have one header section for a result set, whereas a banded page would show multiple header sections for each record in a grouping level. Banded pages that have multiple groups have a nested header for each grouping level.
- **The Footer section** is next and is associated with the header section for that particular grouping. This section is used to display subtotals and totals for the data that is displayed for the associated header section. Controls bound to other fields, or even other controls, can be added here too.
- **The Record Navigation section** is the last section of both simple and banded pages. It is also associated with the same header section of the same group. It contains a Record Navigation Control that is used to scroll between records and to delete, add, sort, save and filter records that are at the same grouping level.

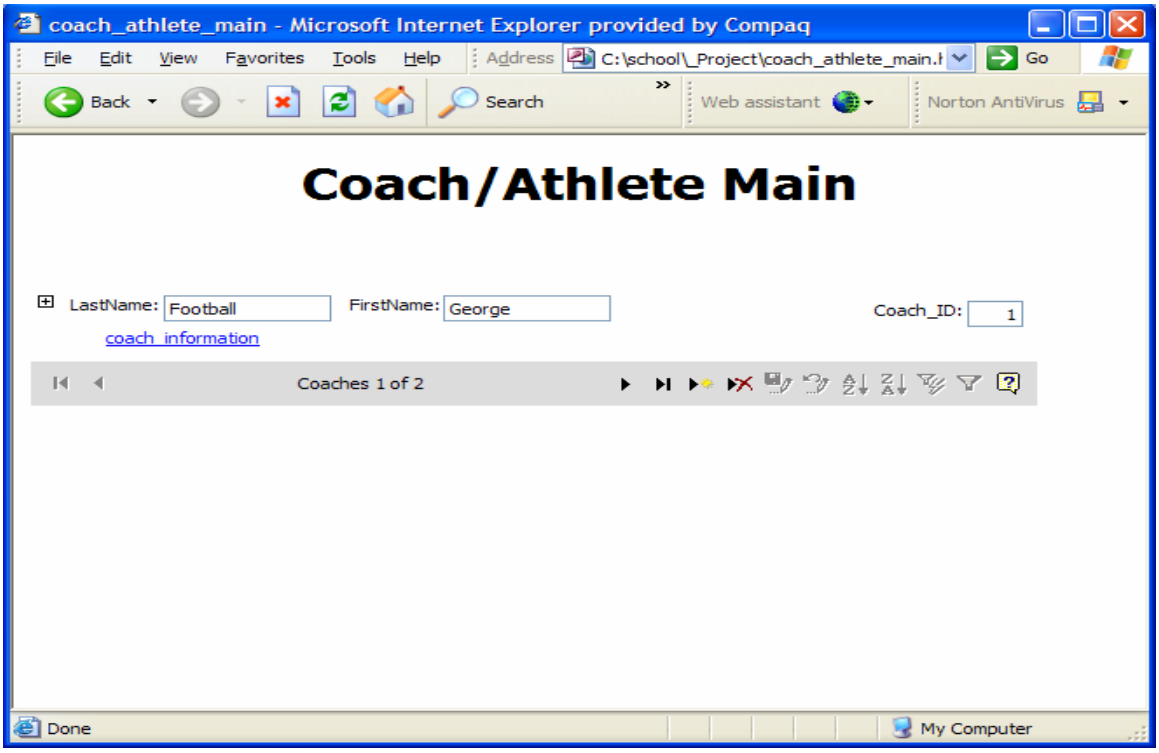


Figure 1. ***Data Access Page showing the four sections.***

When creating a data access page in design view, Access adds a single unbound section that has no type associated to it. A section starts out as unbound because the Data Source control for the page has no Recordset objects defined in its Recordsets collection to bind to. The two ways of binding data to a section are to drag the fields, or tables, from the field list into the section, or by setting the RecordSource property in the sections property sheet.

When the user changes the value of a data bound control, the actual database field is modified. Changes take immediate effect on the data access page and the value is saved back to the database when the record is committed.

The GroupLevel object represents the set of records within the section for the level within the data access page hierarchy. This acts as the bridge between the sections within the grouping level and the set of records which it is bound to.

The DataPage object represents the record set bound to a set of sections sharing the same record navigation control for banded pages. A simple page only has one DataPage object associated with the header section of the page. The top most node in the grouping hierarchy has a single DataPage object, while each node within a grouping hierarchy has its own DataPage object. The DataPage object contains methods such as MoveFirst, MoveLast, MoveNext and MovePrev to allow a user to move between records within a grouping level (See diagram 1).

There are various ways of deploying data access pages along with the database. The chosen method for this project copied the pages and database to a Web server running IIS. The page could then be opened using the HTTP URL pointing to the page. This has the common network and Web server security issues, but is the most versatile and common method for deploying data access pages for multiple users. By placing the database and DAP pages in the virtual web directory, it was possible to access the URL's under a secure web environment.

Microsoft Data Access Page technology allows a developer with basic skills to design and create a graphical web interface to data stored on a database. With built in functionality to scroll, add, delete and update this data, the developer has the ability to give the user community access to a central database, by merely

having access to a browser connected to the Internet. The design view gave a visual representation of how the page will look and gave dragging and dropping functionality of buttons and other common controls, allowing the user to view and manipulate data stored on a central database.

.NET Framework

The .Net framework was researched because it is the foundation of the technology used to develop this project and needed to be understood on a level that ensured the technology relying upon it would work properly. The following is a summary of the research collected while collecting this knowledge.

The .NET Framework is a component of the MS Windows operating system. .NET is included in Windows Server 2003, and the .NET runtime is available as a redistributable package for earlier versions of Windows. The Framework presents a common, unified foundation that includes the “Common Language Runtime” and “Base Class Library” as core elements.

The .NET Framework supports many development languages such as Visual C#, Visual Basic, Visual C++, and Visual J#. The primary language used for this project was Visual Basic, but a few ASP pages were done in C#. This allows extending existing applications to the .NET Framework, regardless of language. All .NET applications need the Common Language Runtime (CLR) as a foundation.

The CLR is a common set of data types, deployment configurations, runtime models and common facilities, such as memory management and thread

handling. This helps enable re-use of software components so application developers do not need to worry about allocating and de-allocating memory for data, the CLR performs this automatically.

The main purpose of the common services of the CLR is to make programming easier, regardless of the programming language, and easier to connect these software components together. The .NET Framework includes a unified class library that can be used by all .NET Framework applications regardless of language. Some of the .NET Base Class Library (BCL) services include:

- Application Configuration
- File and Network I/O
- Application Security and Cryptography
- Data Access and Manipulation
- XML Parsing and Processing
- Web Services

Visual Studio 2005 and ASP.NET 2 (Active Server Pages)

Visual Studio 2005 was the primary development environment used to create the business objects binding to the database and ASP pages. There are many new features designed for web developers to overcome the limitations apparent in prior versions. The new Visual Studio IDE enabled the complete management, design and development of the project by integrating the necessary tools into one package. From within the IDE the database, middle tier

business objects, datasets, web pages, and project tracking were all created and maintained.

ASP.NET is part of the .NET Framework and provides a standard runtime host for the .NET Framework applications. ASP.NET is the container that starts and runs the Web applications written in the .NET Framework. ASP.NET is a complete enterprise architecture for developing Web sites.

During the course of this project, Visual Studio 2005 and ASP.NET 2 were still in the beta phase of Microsoft development. Since the developer was an MSDN subscriber, and the project was on a budget, this project used the latest versions of Visual Studio 2005 and ASP.NET 2.0 to create the web pages.

As with any beta products, one runs the risk of running into unexpected bugs, but there are also many advantages to using state of the art software. The following is a list of some of these major advantages.

- The option to utilize a local web server was used in the development and testing environment, making the installation and configuration of IIS on the developer machine optional. This was very useful during development. Before this version, developers had to make code changes, compile and install it on a configured IIS virtual directory for testing. It is now possible to merely right click and test the site dynamically with all the latest code changes in place.
- Does not compile everything in the site into an assembly, as did prior versions. It now uses the ASP.NET dynamic compilation engine. This ensures that changes in all files within the site are captured and

dynamically compiled for the code-behind classes. ASP.NET is compiled code that runs on the server and takes advantage of early binding, just-in-time (JIT) compilation, native optimizations, and caching services. As mentioned above, this was a very nice feature to have in place.

- The Copy Web site feature allowed the web site to be copied to another local or remote location by simply selecting a menu item. When it was time to deploy the web site to IIS, this made the process a trivial task.
- A complete user security application that can be plugged in and used with any Visual Studio application. Users can be assigned to groups and privileges can be assigned to allow/disallow access to certain areas of the web site, or application. This feature uses an embedded Access database to store and maintain user data. This is a very handy feature because almost every interactive application and web site uses this type of functionality and is usually written from scratch each time and specifically designed for the application at hand. The data model originally included user and group tables. But, when this feature was discovered, the tables where abandoned and replaced with this feature.
- Microsoft Active Server Pages (ASP) is the server-side execution environment that enables ActiveX scripts and ActiveX server components to run on the server. By combining scripts and components, dynamic content and powerful Web-based applications can be created easily. This makes deployment trivial. As with traditional software upgrades, customers must download and install the new version to gain access to

the new features. With server side technology, only the software on the server (the web site) needs to be upgraded for all to take advantage of.

- With ASP.NET, application and session variables can be defined that can carry across multiple pages in a Web site. This can be as simple as remembering a user's name and it is necessary in applications, such as a Triathlon training application. This was a useful feature, and made persistence of session data possible while hopping from page to page. This was also used for security purposes by limiting access to items on certain pages to members of a particular user group.
- ASP.NET uses Windows NT Security. ASP files can be easily restricted to just certain users through secure Windows NT authentication, basic Web authentication, or client-side certificates. For additional security, all client-to-server communications can be secured with Secure Sockets Layer (SSL). This would be nice to have in a future version of the application, but was not attempted in the first phase.

Databases

Microsoft Access

Microsoft Access was used to store the data for the prototype application. Although Access is a common database, research was still necessary to properly integrate it into the project.

Microsoft Access is a simple, but very powerful, relational database that has many user friendly features. Access is stored as a file within the file

system with a file type .mdb extension. Although most commercial databases are technically, in a physical sense, just big files that are assigned space on a hard drive, they are not stored in this manner. Databases, such as Oracle and Sybase ASE, are configured onto a system and a variety of files and indexes are created to store and retrieve the data that will be stored in them. The Access database allows designers to create tables, queries, Forms, Reports and Data Access Pages.

Access allows the database designer to graphically create the tables and any relations or constraints they may have with each other. Tables can be created from scratch, or by using a Create Table wizard if the designer is new to database design, or new to Access.

Users gain access to an Access database via the Open Database Connectivity (ODBC) driver. The ODBC driver allows users to add, delete and update the data using the Structured Query Language (SQL). The new .NET2 environment also has an Access ADO object to connect business objects and ASP pages to Access databases.

SQL Server 2005

Microsoft SQL Server 2005 was the main database used to store data in this project. It is the latest version of database technology put out by Microsoft and was still in beta while developing the D3Mulitisport project. Many of the new enhancements are directed towards optimizing search speed, data storage optimization, and accessing information in various formats. The

following is a list of some of the more important features of MS SQL Server 2005:

- Programmers can now take advantage of the Microsoft .NET Framework class library and modern programming languages to implement functionality within the server.
- Using the common language runtime (CLR) integration, developers can code stored procedures, functions, and triggers in the .NET Framework language of choice
- Database development is now integrated into the Microsoft Visual Studio 2005 IDE; so the same tools for developing and debugging database objects and scripts can be used to write middle tier and client tier .NET Framework components and services. This was a nice feature allowing middle tier business objects to attach directly to the database table. These business objects were then called from ASP pages.
- Ability to use managed code. This is more efficient than transact SQL at processing numbers and managing complex execution logic; and provides rich support for string handling and regular expressions.
- HTTP can be used to access SQL Server directly by utilizing a lightweight Web server, HTTPSYS, that is integrated into the Windows operating system without using a middle tier listener such as IIS.
- Integration of ADO.NET 2.0, which has new data access options and supports new data types to managed code languages.

- Highly secure in a networked environment, with role-based security and file and network encryption make intrusions into the system more complicated.
- SQL Server 2005 also adds native data encryption. Using the certificate and key management system built into SQL Server and by utilizing some new system functions enables the encryption and decryption of data on the server. This is a feature to be implemented in a future release of the D3mulitsport system.

Microsoft SQL Server 2005 provides the tools needed to build new classes of database applications. By removing the barriers to code execution and storage location and integrating standards, such as XML, SQL Server 2005 opens up a world of possibilities to the database developer.

Internet Information Services (IIS)

IIS is Microsoft's web server software that comes bundled with its server operating system software. IIS is tightly integrated with the latest Windows operating system software and comes with a graphical user interface for administrators to configure IIS the way they want. IIS works well with Microsoft's other software products by providing filter, interfaces and security protocols that can communicate and limit access to them as well. Although, IIS comes bundled in the operating system, it is not usually installed and configured on the system. It is up to the administrator to add users and configure the virtual directories

needed to hold the web site. The security configurations allow for both user and object level security that integrates directly into the Windows directory service.

Security

The technology used in this project can be developed to run on earlier versions of Windows operating systems, but is intended to run on an MS Windows Server 2003 Operating system environment. As a result, the Windows Server 2003 security model was used to protect and safeguard the system from unauthorized access.

The Windows Server 2003 security model integrates the principles of identification, authentication, authorization and access control, confidentiality, integrity, non-repudiation, trusts and audit entries to protect data and resources on a system and networks they communicate with. The following is a brief discussion of these principles.

Identification involves the creation of information that identifies and validates both users and computers that have access to system resources within the network or domain.

Authentication involves the principles of authenticating data, which is usually in the form of a username and password combination for a user, or other resource, requesting access to a specific set of data. The authentication data is stored and maintained in a system database which stores information on the user, or computer, pertaining to the access rights and privileges the account has. This information creates an access token that is associated with the account for

the duration of the current session. This feature comes in the form of a .NET plug-in and is complete with a web interface and an embedded Access database that allows the addition of users and groups.

Authorization and access control involve the use of the access token, discussed above, and an access control list (ACL). The privileges of the access token are compared to the resource's access control list to determine if the account has access to the system resource or data.

Confidentiality involves protecting the contents of data and how that data is manipulated. It is important to ensure this confidentiality of data while it is retained in a database, or file system, and while traveling from one resource to another. Confidentiality can be maintained via privileged access and encryption algorithms.

Integrity involves ensuring that the data is not modified in any way while it makes its way across the network to its destination.

Nonrepudiation involves a combination of authentication and integrity, discussed above. Nonrepudiation discloses the source of a message, message delivery status and whether a message has been received.

Trusts involve logical relationships between domains that share common users and resources. This trust is accomplished via a pass through authentication that allows a user, or process, to share the same privileges and access rights they have on their home domain, to carry over to the visiting domain. This is very common within companies, where different groups are

contained within a domain, but often certain users of a group need access to resources and data located on other domains.

Audit entries involve the logging of information that is related to the accessing of system resources and data. The information commonly includes data about system operations and various security related events that have happened. It can also reveal information on what data user accounts, or processes, are accessing and where they accessed it from.

The Windows Server 2003 security model uses the above security principles in the following components and technologies:

- Logon and authentication technologies
- Authorization and access control technologies
- Data security technologies
- Group policy technologies
- Trust technologies
- Public key infrastructure (PKI) technologies

The logon and authentication technologies use cryptology algorithms to validate the identity of a user, or process, requesting access to a system domain or network resource. Once access has been granted, the server side decrypts the signed data with a cryptographic key that determines the privileges and access rights the user has to various data and resources within the domain, or network. Cryptographic keys are stored in the Local Security Authority (LSA) of a computer, or the Active Directory of a network. This is a safe central location where these keys can be secure from unauthorized access. Kerberos v5

authentication, NTLM, secure channel, Passport and Digest are authentication protocols used by the Windows Server 2003 operating system that allow users, computers and various other processes access to system and network resources and data.

Authorization and access control technologies allow administrators to limit and control the access rights of certain users and groups to system and network resources that are necessary for them to perform their duties. User rights and permissions are based on two models: An ACL based impersonation model and a roles-based protected subsystem model.

The ACL based impersonation model allows administrators the ability to place users into logical groups of users that give all members of the group the same access rights and system privileges. This model also allows object based ACL which defines the level of access that certain users, groups, or processes can have to itself. For a file, the access properties might be read, update and delete. Some users would have zero to all of these permissions on the file. When the user and object ACL's conflict, the windows Object Manager decides what permissions to give the user for the object.

The roles based model uses an Authorization manager to assign roles to specific objects and users that relate to performing certain tasks within an organization.

The Data Security technologies concentrate on the protection of data at all levels on the system and network. System level data security uses the Encrypting File System to encrypt data stored on the computer hard drive by use

of public key encryption. The system key utility incorporates encryption techniques to store user passwords in the Security Accounts Manager database.

Network level data security uses the Internet Protocol Security (IPSec). This a set of cryptography security protocols used to secure data traveling across the local area network (LAN). IPSec checks packet headers to ensure system level access along with data encryption protection. Windows also uses a service called Routing and remote access that offers routing services which control access to system resources from the LAN level to the Internet. The Internet Authentication Service uses 802.1 level standards to authenticate and grant access to dial-in users.

The Group Policy technologies allow administrators to package groups, users and computers into a Group Policy object and link them to a domain or logical business unit. The Group Policy contains the following extensions: Administrative templates, software installation, security settings, software restrictions, IPSec policy, scripts, wireless network policies, folder redirection, Internet Explorer maintenance, disk quotas, quality of service and remote installation services. The security setting within a group manage the authentication of a user to a system or network, access rights to a resource, the recording of resource access into the event log and if a user is a member of a group. Administrators can also use software restriction policies to limit system or network access to certain types of software.

The main purpose of Trust technologies is to create a trusting relationship between two domains. When a trust relationship exists between domains, each

domain trusts the authentication data that has originated from a trusted domain. This is a powerful feature that allows administrators to grant access rights to entire groups of users across a set of domains without having to set up user and group permissions on each domain within a network.

Public key infrastructure (PKI) is a trust that uses digital certificates, certification authorities (CA) and registration authorities (RA) to verify the authenticity of both parties of an electronic transfer by using public key cryptography. Both parties of the transaction establish a trust in a CA certificate. This trust is established by copying the root certificate and a valid path that contains all valid certificates. Qualified subordination allows trusts between separate CAs to be put into trust hierarchies, also called cross certification. Qualified subordination allows the use of logical groups of CA's to access common resources and data within an organization or separate business partners.

This web site uses only the .NET2 user administration plug-in for security, but technologies like PKI were seriously considered. The nature of the business just did not justify the overhead of using encryption technologies but would have made logging in easier since certificates do not need userids and passwords once established. Maybe if this sport becomes more competitive, security will become more of an issue and this technology can be implemented in a future release. If the business really grows and requires separate domains to support multiple business units, then Trust technologies should be incorporated into the architecture.

Summary

A good understanding and knowledge of the technologies needed to complete the web site were important to discover any constraints and weaknesses that needed to be considered during the development of the system. Now that the system requirements are defined and an understanding of the technologies has been considered, the designer was able to ascertain the feasibility of the system.

Chapter 3: Project Analysis & Design

Business Requirements

Users must have the ability to log onto the site under various account groups allowing access to the pages granted to the group. There are three groups within the system:

- Administrator – Allows the user to create other users within the site and create workouts within the various workout categories. Allows ability to assign a coach to an athlete.
- Coach – Allows user to generate workout programs for athletes from existing workout types and communicate with them during training.
- Athlete – Allows user access to enter demographic information and access to their workout program. Allows access to communicate with their coach during the course of the workout program.

The site must have the ability to create workout categories consisting of one or more categories of workouts. These categories are running, swimming, biking and weights. Each category has similar characteristics but each varies slightly. This results in each category being represented as separate entities within the database and business object models.

Each category must have unique workouts that can be created by D3Multisport staff and used by the coach to assign to athlete workout programs. The staff must have the ability to add, delete and modify workout specifics to each category. Both the coach and athlete must have access to these workouts

to use in their training program schedule. The workout categories must have the following details associated:

- Bike – Bike code, description, purpose, time and detailed comments
- Run – running id, description, purpose, time and detailed comments
- Swim – swim code, description, purpose, time, distance and detailed comments
- Weight – weight code, period, sets, reps, speed recovery and miss days. There are also specifics to each weight workout each containing a weight code, sequence and details description.

Each coach and athlete must have the ability to use any combination of workouts within the categories into their routines. Triathlon athletes need to excel at running, swimming and biking in order to stay competitive. Daily workout programs must combine combinations of all three along with weight lifting routines to condition their bodies. Therefore, both coach and athlete must have the ability to modify the daily workout schedule for various reasons.

The ability to communicate interactively online is necessary in order to create a dynamic relationship between coach and athlete. This allows each to obtain advice that may only be relevant at the time of need.

The ability to record progress online is necessary for analysis by the coach, and for the athlete to monitor progress in a visual manner.

Acceptance of Business Requirements

Once the requirements and planning of this project were completed, the requirements document was presented to the stakeholders for review. Upon document approval the development of the system commenced.

Application Requirements

The application must consist of a dynamic web site accessible to D3multisport staff, their coach's and the athletes who have paid for access to the site. The web site provides the ability to interact with the system, each other and provide data persistence via a relational database. The system should reflect and enable functionality of the stated business requirements.

Planning

Once the requirements were properly defined and agreed upon, it was time to start planning the components of the system. The designer of the system not only needs to understand the business requirements, but needs to know which of them (if not all) have been agreed upon and which ones will be integrated at a later time. This was accomplished by understanding the scope of the project at the various stages of development. Once the scope of the project was defined, a work breakdown structure was created to organize and define the components necessary to create the system from inception to maintenance. The following is a brief description of the documents and diagrams that were used to establish this communication between staff and development.

- **Scope Statement**

  This project had a very short time frame in which a working system had to be in place for the customer. It was important to establish what was absolutely needed to have a useful application. The scope statement outlined the scope of this first phase and defined the deliverables.

- **Work Breakdown Structure (WBS)**

  Organizing a list of detailed deliverables into a WBS allowed the designer to ensure all pieces of the project were considered and determine whether the time frame for the first draft of the application was reasonable.

- **Gantt Chart**

  The content of the WBS were placed into a Gantt chart for visual view of the project tasks and allowed the system planner to work within the time frames and have a schedule to work from. The Gantt chart allowed the schedule to be modified and tasks were removed and added during the course of the project as the requirements changed.

- **Create a Network Diagram**

  The network diagram established the details of how all pieces of the system were connected and communicate with each other. This is a somewhat complex system with a database server, web server, and ASP 2 system under the hood of the operating system. All of the sub components needed to be configured properly to communicate with each other.

Functional Requirements

This project put forth an effort to design and develop an online website that provides a secure, real-time IT solution for athletes to enter their training schedules and progress information. It was now time to turn the results of the approved business requirements and apply them in a technical manner. In order to accomplish this, logical models of the system were created that both business and technical personnel could focus on to communicate ideas and intensions of the project. These logical models depicted what the system is and must do; not how it was implemented. By expressing the business requirements as logical models, the design team could separate the business concerns from the technical solutions. This increased the chance of meeting the needs of the customer, rather than trying to design the needs of the business around the constraints of a specific technology solution.

Technical Design Specifications

The software and database technology involved a prototype of a scaled downed version of the database and web interface. The intentions were to create a prototype using an Access database and Data Access Pages to interface with the database. Once the client approved the content, layout and look and feel of the application, the technology would be updated. The predicted data entering this application is between one, to one and a half gigabytes, per month. Access will not handle either the traffic and will max out at an undesired time frame. Also,

Data Access Pages is limited in its functionality and the web interface will need to be extensible. Because of the limitations of the prototype technologies, the database was forward engineered into the MS SQL Server database format. This was a trivial task since the data model was designed in Sybase PowerDesigner, which forwarded the data model into a SQL Server database. The web interface was created in the Microsoft .NET2 environment, using ASP pages and other Microsoft web technologies. The proposed hardware solution involved using a Compaq desktop personal computer with a configurable router attached. The web site was developed using the latest Microsoft technologies which included MS Server 2005 and the Microsoft Visual Studio Interactive Development Environment (IDE) using Visual Basic as the object oriented language.

System Architecture

The system architecture consists of a detailed list of components for the system and how they interact with each other. The major components on the server end can run on one computer for cost and resource efficiency. The main components are a web server, a database server and an operating system.

The components involved while constructing the system architecture are more physical in nature. Therefore this process was more concerned with the technical and implementation aspects of the system and how they responded and communicated with each other. Not only were the inputs and outputs of the system defined, but data, processes and interfaces were defined with an outline

explaining how these components communicated and interacted across the network.

Hardware Requirements

A high performance PC was needed to handle the necessary load to sustain the system components installed and configured on the site. Once the site is up and running for a while and resources are more exhausted, the need for separate servers will be required. The first logical separation would be to host the web site on one server and the database on another server with a RAID array in order to displace the database across multiple drives for faster access to the data and provide redundancy, in the case of system failure.

A high speed router was needed to handle the transfer of data between the web site and the users interacting with the system from outside the internal network. The router was installed and configured to maintain a firewall and provide Internet capabilities for running a web site. A port on the router was configured to provide port forward processing to the web server.

Software Architecture

Design of the software architecture involved a real world analysis of how the system was developed. There was extensive research into technologies that would work together and produce a system the customer approved. The prototype and finished project were actually two different Microsoft technologies that worked together. The Data Access Pages performed the first part of the

project fine, but was not powerful enough to continue with the rest of the project, where ASP .NET was more appropriate. It was decided to create both anyways, putting the timeline at risk, because of the development effort needed to do both. For maintenance reasons, it was determined that the Access/DAP piece eventually be carried over to the MS SQL Server/ASP model at a future date. The two pieces remain separate but both are functional. The DAP/Access model serves as the initial interface for the athlete to enter personal demographic and other data that is critical for the coach to create a program for the athlete. The ASP/SQL Server model handles the more complex parts of the system that involve creating the workouts, athlete programs and progress reports. A future release could eventually merge the two technologies by creating an ASP interface to the Access database via an AccessADO interface object, but was not essential for this first phase of the architecture. The two applications serve different purposes, so could be left separate if D3Multisport does not have a dire need to merge them.

Life Cycle Models to be Followed

After planning and analysis were finished, the life cycle methodology was selected. A somewhat modified version of the Software Development Life Cycle (SDLC) was considered and is how the project development was initiated. This was considered a modified version because the prototype application, that handled input of athlete profiles via DAP and an Access database, had to be finished first. An incremental method was attempted for the rest of development

due to the systems complexity and incomplete business requirements up front. This enabled development to get underway and add enhancements in an incremental fashion.

The project was initially implemented using a Software Development Life Cycle (SDLC) process known as Scrum. The project spanned a period of 4 months with an expected completion date of December 30, 2005.

Scrum Agile Process Development Cycle

The Scrum process can be used to manage and control software development using iterative and incremental practices. By wrapping existing software development practices, Scrum can increase productivity and reduce the time needed to complete a continuous project. Scrum is not a silver bullet that makes development issues and problems go away, instead it is a practice used to highlight and discover these issues so they can be known and addressed. Scrum helps identify triage and solve issues that can cause problems throughout the development process of the project. The team still needs to figure out how to solve these problems.

The Scrum process describes three types of product development techniques broken down into phases. Each cycle represents the analysis, design, implementation and testing phases. The first technique breaks each item into individual cycles, which could be broken into tasks. Each task runs through each cycle before the next task can begin. The second technique has each phase within the cycle slightly overlapping each other. The third technique, which

Takeuchi and Nonaka envisioned as Scrum, has all phases of the product development overlapping. In an ideal Scrum development environment, once analysis ended on the first cycle, the design would commence and the analysis of the second task would begin simultaneously. This process would continue for each phase within the task for each subsequent task.

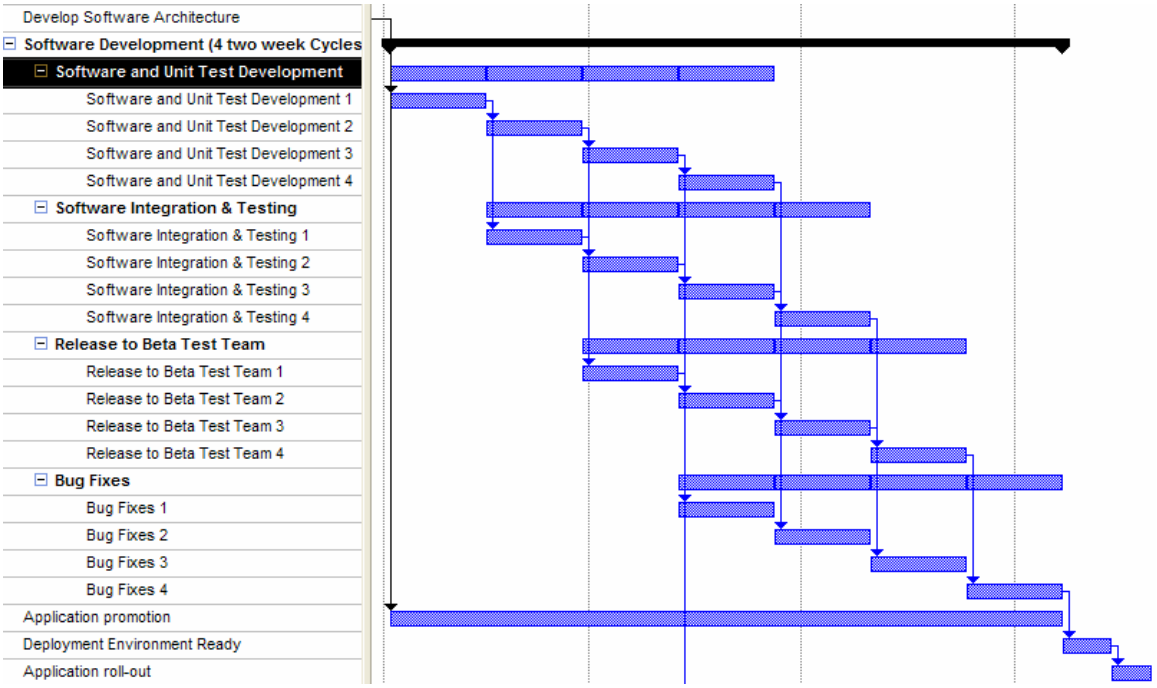


Figure 2. *Gantt Chart Detailing Scrum Process Development Cycle*

There was only one developer on the development team and only one remote analyst and application expert to system test the application as it was developed. Therefore the process was like the second Scrum technique, where the phases slightly overlapped, but not enough to flow the project management after it. Any overlap was on the analysis of requirements planned for a future

release. The Scrum development process was attempted and followed for the initial part of this project.

Phase I – Analysis Phase

Phase 1 consisted of gathering information to determine what functionality needed to be implemented. This included meetings with the client to define business rules and application requirements. This phase was divided into two parts.

- Problem Analysis – Defined the reasons why the existing system does not work and why this new system will. This is where the true deficiencies of the existing system were defined. A good understanding of the issues helped define what the new system needed to accomplish.
- Requirements Analysis – Defined the business requirements for the new system. Establishing what the system needed to provide was the primary objective of the analysis phase. This was one of the most important phases, since it defined what the customer was asking for and what they wanted the system to do. It was critical to identify and define as many requirements as possible before making decisions on which requirements to proceed with and start development of the system.

The deliverables for this phase included:

- Business Requirements Document
- Use Case Diagram *(See appendix C)*
- Analysis Phase Sign Offs

Phase II – Design Phase

During this phase, the database schema and data-model were designed along with the storyboarding of a GUI front-end web interface. The deliverables required for this project were best suited for a model driven design approach, due to the visual nature of the diagrams created to illustrate each deliverable. With the requirements in place, a model driven information engineering process sensitive approach was used to create the data model. With a data model in place, the web site storyboard and work flow diagrams visually represented the site pages and helped ensure the required pieces of the data-model were in place. A prototype was created by following this work flow and used the database on the backend. The prototype was important because it got something in front of the client that they could reply to. With user participation, it was easier to get a fast iterative approach and get the application on the right track. This gave users a chance to modify and add requirements that made the system more useful to the client. As mentioned in the Scrum research, the design phase intermingled with the surrounding analysis and construction phases. So, as requirements and database schemas changed and new pages were added to the site; so did the project plan and Gantt Chart change to reflect the changes the project evolved into.

The deliverables for this phase included:
(Diagrams and Flowcharts can be viewed in Appendix C)

- Scope Statement
- Data Model diagram

- Web Site Storyboard Flow Chart
- Work Flow Diagram
- Network Diagram
- A Prototype Web Site
- Project Plan for entire project
- Gantt Chart for entire project

Phase III – Construction Phase

During the construction phase, the database was enhanced according to the new design and web pages were created to interact with the database. Both MS Access and SQL Server databases were created and enhanced during each increment of this phase. The web site had new pages added as the data model and business requirements increased in number and complexity. Other items completed during this phase were hardware and network installation and configuration. The database and web site needed an operating system, database server, web server, network access, routers and a computer to install and configure everything on. As the system was being created, a list of critical areas needed to be kept up to date, listing what and how the system should be tested. Not only the work flow of the business requirements needed to be tested, but a full logical matrix was constructed for each work flow item to ensure the client did not break the system while being creative in the field.

The deliverables for this phase included:

- Final Design Document
- Functional web site
- An enhanced normalized database
- Strategies for testing and implementation

Phase IV – Testing Phase

This phase provided verification that the requested functionality had been properly implemented and ensured the product met system development requirements. The prototype underwent testing by both developer and client. This phase involved the creation and implementation of a formalized test plan to ensure business requirements were met and proper regression testing was performed. The test plan reduced unnecessary bugs introduced by development during each cycle of the development phase.

The deliverables for this phase included:

- A Test Plan Document
- Client Approval

Phase V – Implementation Phase

During this phase, the final product was deployed to the production environment. All hardware, network and system software was installed and configured. Once the physical system was set up, the application components of the system were installed and configured. The client was properly trained on

using the application and was prepared to beta the system with real customers and data. The beta phase of the project required close monitoring of the client business processes and data integrity issues. If a bug was discovered, it was critical to find a reasonable work round and a solution to resolve the problem as soon as possible. It is common for this phase to churn off another cycle of development if it is something all parties have not considered and is critical for system success.

Application Promotion

Once the application was complete and the system components were configured and communicating with each other, the promotion of the application commenced from the D3Multisport end. This involved real world interaction with an athlete and coach using the system to interact with each other while in the process of training.

Project Deliverables

The deliverables for this project consisted of a working web site that had the ability to save detailed information regarding the training program and progress of an athlete. A database was designed and created to store the information generated by D3multisport, their coaches and the athletes. The athlete has the ability to enter demographics, training level and other personal information beneficial for the coach to create and recommend a training program for the athlete. The coach then creates the training program for the athlete using

predefined workout programs that have been created by D3multisport staff. The athlete is then able to access this program online and begin the training process. The athlete then records the results of the training program, such as times, distances, weight progress online and saves this information to the database. The coach now has the ability to analyze this information to determine whether the athlete is making the proper level of progress. Once performing the analysis of the information, the coach has the option to alter the program, or simply record a message for the athlete to follow.

This project included the following deliverables:

- Business Requirements Document
- Use Case Diagram
- Analysis Phase Sign Offs
- Scope Statement
- Data Model diagram
- Network diagram
- Web site Storyboard Flow Chart
- Project Plan for entire project
- Final Design Document
- Functional web site
- An enhanced normalized database
- Strategies for testing and implementation
- Test Plan Document

- Client Approval
- A Final Product
- Analysis of software development methodology used to build the system
- Analysis of the information technologies used to build this project
- A final presentation of the project

Review of Deliverables

The final presentation to the client included:

- Description of the business problem and technical solution used to provide a solution.
- Overview of the different methodologies, tools, and techniques used to develop the project.
- Overview of the software architecture's strengths and weaknesses.
- Demonstration of the final product.
- Overview of lessons learned during the development of the project.

Discussion of Diagrams

Use Case Diagram - established a clear understanding of the entities interfacing with the web site. The entities were in the form of actors, PC's and the web site itself. This diagram focused on graphically depicting the interactions between the system and all other external systems and users. This diagram provided a high level view of what the external systems and users expected from interaction with the system.

Data Flow - illustrates the flow of data through the entire system and how it is maintained and processed. By abstracting from the physical design of the database and illustrating the technical and business design decisions, that were agreed upon during analysis, this diagram served as a blue print for the development of the system. During database design, it was used to establish which tables and constraints were needed. Programming used it to design and create application interfaces to the database. Network design used it to know what hardware, network components and system software was needed to maintain the system.

Entity Relationship – The Entity Relationship Diagram (ERD) was created in the form of an Object model to show the relationships within the database schema and web pages. This helped find holes in the database in regards to initial business rules. This diagram was data centered in nature because it focused on the raw data of the system stored in the database and how it transpired into web pages. This diagram needed to be drawn before the dataflow diagram since it involved the creation of some of the entities that were used in its creation. The object model was not concerned with table structure or data types within the database, but was concerned with how the entities (tables, views, stored procedures, business objects and web pages) affected each other and how the data was entered, stored and managed within the system.

Data Model - represents how data is stored and relates to each other within the database. Although the data model can integrate external data stored in another database, or even XML files, this model was limited to a single database used to store and access data for this project. The data model is where database tables were designed along with the relationships they have to each other. The relationships are expressed as one-to-one, one-to-many or even many-to-many. Primary and foreign keys were designed on each table so data maintained integrity and meaning.

GUI Storyboard for Web Site -The web site storyboard diagram was actually accomplished via a combination of the data flow and object model diagrams. These two diagrams together provided a structured process flow from a user perspective. The diagrams were important to the designer and client to gain knowledge of how the web pages are related and flow together. This high level view of the connecting relationships between the web pages, and data, provided the user a means to know what page each link would take them to in order to perform certain business functions within the web site.

Network - System components are defined as all the actors defined in the use case diagram (users, client systems, server systems, networks and databases) and how they interact to transfer the data flowing between each other. This diagram is a high level diagram that illustrates the flow of data

between the system components and is concerned with physical locations of the hardware and users of the system. The final decisions made about the nature of the systems purchased, such as model names and system properties like processor speed, hard disk space, band width and software brands were documented in this diagram.

Discussion of Reports

The various reports provided a narrative means to communicate what the project entailed. These reports detailed the projects purpose, requirements, scope, plans for monitoring progress and how the system was tested. Narrative provided the basis for the diagrams that gave the project visual perspective and provided a foundation to add too during each phase of the software development life cycle process.

Goals

The main goal of design and development was to finalize the requirements and design and develop the system. The various reports and diagrams provided both client and architect the necessary means to visualize, analyze and determined the feasibility of the project. Once established, the first phase of development for the project commenced and produced a completed first phase working web site with a database back end.

Summary

The completion of this phase of the project produced a working system with the necessary diagrams, documents, reports and systems in place. Both developer and client realize the system had been finished and the first phase of the system was completed and ready for use. The diagrams and communication established a working relationship that scrutinized the progress and flow of the project so it did not veer off course. The Visual Studio 2005 IDE provided a means to create business objects to interface between the database and ASP pages. As a result, an object model was designed in place of separate ERD and web flow diagrams since it accomplished the desired affects of both. The next chapter will go into more detail on the matter of how the project was managed and provides project history. It deserves mention that due to unforeseen events, the iterative nature of this project broke down about half way and strayed into a sort of remote developer situation. It was the initial efforts and the creation of these diagrams that helped save the project and allowed it to progress into a usable project.

Chapter 4: Project Management and History

Project Beginnings

The beginnings of this project started with a client of the facilitator, Mike Nims, and his lifelong commitment to health and physical training. As a triathlon athlete himself, he and his trainer at D3Multisports have been putting training programs together for years. As the years progressed, weaknesses within the business process pertaining to the efficiency of getting relevant real-time data back and forth between athlete and coach were discovered. As an IT expert of many years at Oracle, Mike and D3Multisports got together and proposed some ideas that would make this process easier to implement, maintain and access. Right around this period, one of Mike's Masters Students was taking his last class with Mike and had mentioned in the forum that he was still searching for a master's thesis and was wondering how to obtain an advisor. Mike had proposed to the class that he had an interesting opportunity that would be a perfect opportunity to use as a masters thesis and gain valuable experience at designing and developing an interactive web site. It was a primary goal of the student to accomplish the design and development of an interactive web site before the completion of a Masters degree.

This was perfect timing and presented an opportunity to work on a real-world project which would truly be useful to others. It was now possible to think about the initial requirements required for the project and get an idea of how large it would be. The last class was finished and focus was placed entirely on

the project. The first phase of the project was considered completed in December 2005.

Project Management

This project initially considered two project management disciplines: the Zachman Framework and Scrum. The Zachman framework was considered because of its thorough nature and the Scrum was also considered because it coincides with the Scrum development life cycle intended for this project. The following is an analysis of the two methodologies and how they were used together to provide a complete project management solution.

Zachman Framework Methodology

The Zachman Framework is a logical structure for organizing and classifying the descriptive representations significant to managing and developing an Enterprise system. The framework's roots were laid from traditional disciplines relating architecture to construction and engineering to manufacturing. This is accomplished by classifying and organizing the artifacts and components that make up and produce a complex system, such as athletes, coaches or workouts and such.

The Zachman framework classifies a complex system into abstract components of "What", "How", "Where", "Who", "When" and "Why". The framework designer can then use these generic components to describe the complex system in a more direct manner that relates to the system being

designed. Breaking down a complex system into these individual components, a framework designer can focus on selected aspects of the system without losing the contextual perspective of the overall system. Once the individual components have been ascertained and scrutinized, their relationships can be connected to see if each components context still holds true in relationship to each other.

Once the system has been broken down into components, the Zachman framework takes a look at the system in reference to the entire enterprise and how the system fits into the architecture of the whole company. John Zachman used the players within the company to base the foundation for how a system will fit into an organization by conceiving each player's point of view of the system as it fits into existing business needs. These players are broken down by roles within the company. The players and roles are: (1) Scope, someone who has decided to plan and undertake the responsibility to create a business in a specific industry. This person could be a planner, or project manager, who defines the scope in a contextual manner of the project, (2) the owner who looks at the system from a business perspective and relates the components from a conceptual business model approach, (3) the designer, who relates the components and translates them into a logical system model in order to represent the business in a more formal manner and has the ability to apply specific technologies to business needs, (4) the system analyst, who designs the specific components on a more detailed level such as database schemas and design of system architecture, (5) the builder of the system, who actually performs the detailed work such as coding of software, designing and creation of

web pages, etc…, (6) the system itself which has it's own row in the Zachman Framework matrix. Examples of this, for a software project that is to be integrated into an enterprise system for a company would include: What (Data), How (Function), Where (Network), Who (People), When (Time) and Why (Motivation).

A generic Zachman Framework would look like this:

	What	How	Where	Who	When	Why
Scope						
Business Model						
System Model						
Technology Model						
Detailed Representations						
Functioning Enterprise						

Table 1: *Zachman Framework*

As table 1 indicates, each player looks at the same category of information, or component. If a framework architect can perceive how each player ascertains and determines the What, How, Where, Who, When and Why's of their piece of the system and how it relates to their piece of the business, then the system has a much better chance of fitting in and being useful to all players and parts of the business. The result is each cell within the matrix represents the various modeling techniques used in the information processing industry today and how they fit into the overall framework of the company.

Each player would ascertain each component from their own views. The following are examples of how each player would perceive the general nature of the six components:

- **Scope** – Views system from a ballpark perspective by defining the direction and purpose of the system within the enterprise. The “What” component might consist of a list of things important to the business and the “Who” component might contain a list of the people who are important to the business.
- **Business Model** – Views system from an owner’s view and is concerned with the nature of the business such as structure and function within the system. The “How” component might be a Business Process Model that illustrates the high level functions of the known, and planned, processes within the company.
- **System Model** – Views system from an architectural perspective and is concerned with defining the business model the owner has in mind. The “Data” component might be a logical data model of the database that fit’s the system into the enterprise from an abstract perspective.
- **Technology Model** – Views the system from a designer perspective. The “Data” component would be a physical data model that actually defines the system at the entity level. Players at this level are concerned with detailed design of each component within the enterprise.
- **Detailed Representations** – Views the system from a builder’s perspective. The “Data” component represents a database designer determining a column within a table needs to be a varchar(10), or a decimal. These players are the actual creators of the system that create the databases, do the programming and network the systems together.

- **Functioning Enterprise** – Views the component from the system level and how it fits into the entire enterprise.

The Zachman Framework is not intended to be a sure fire method to integrate existing systems within the enterprise into a useful cohesive framework, but is intended to be used as a tool to help enterprise framework architects come closer to defining and creating systems that work together within the organization by forcing them to think about how each component of the system relates and affects each part of the business unit and the players within.

By following the principles defined within the Zachman Framework, the definition and design of the D3multisport web site was well established early on in the project. Each component of the matrix defined above was considered and applied to the project and allowed the first phase of the project to produce an almost complete system where all parts of the system were in place and ready to be extended.

Scrum Methodology

Project management of a Scrum project requires that all other project documents follow suite. The project manager (PM) can no longer place all the items from all tasks into an all inclusive document that describes all the risks, times and scopes for the entire project all lumped together. Each item within each task must be broken out into their respective cycle. Not to say this information cannot be contained within the same document, but it must be

broken out into logical units. One advantage is that the PM doesn't have to finish each required document for the whole project, only the first cycle or two. This gets the project off to a fast start and allows room for requirements gathering and analysis, while the developers are working on the tasks within cycle 1. Another valuable side affect, is that as the project progresses, the PM becomes more knowledgeable on the amount of time and resources needed to complete a cycle and can predict the remaining cycles more accurately for management.

By observing a project within the Scrum process, the normal items such as analysis, design, coding and testing are not the actual tasks driving the project, but are instead the cycle. The Scrum process optimizes project management by getting each phase of the project off to a fast start without compromising quality. It also helps the PM process by breaking away from the normal constraints and allows the PM to think outside the box by itemizing the process instead of the items, or tasks.

Combining the Scrum process with the Zachman framework allowed the best of both processes to evolve the design and development of the D3multisport web site project. Although each piece of the design process was analyzed and considered, once there was enough information about a system component, the next phase of the development process could commence. This allowed the project to get off to a good start and maintain completeness throughout the course of the project.

Significant Events and Milestones

Due to the nature of this project, there were only five significant milestones:

- Completion and approval of requirements document – This was significant because the design of the system could commence. Requirements for both the prototype and main web site were completed and approved together. It was important to have the core requirements known in order to design the system properly.
- Completion of prototype – The prototype proved useful in that it provided the customer an idea of how the site will function and gave the developer insight on the shortcomings of the prototype technology.
- Completion of web site design – Once the web site design was in place, the functional details could be storyboarded. The design provided critical information for the completion of the database tables and the ASP pages that needed to be created to interface with the database.
- Completion of the database – With the database completed, in conjunction with the corresponding data model, the completion of the data entities confirmed the web pages had all relevant information needed to hold the business data. The web site and database design evolved simultaneously as holes and weaknesses were discovered.
- Completion of web site – The completion of the web site was the final deliverable and milestone of the project and provided the client with the software to install and configure on a system of their own.

Plan Changes

Remote Development

Although the effort to follow the Scrum SDLC and the proposed project management methodologies was attempted, the reality of project hindered the ability to adhere to these goals. Since the developer was thousands of miles away, from both the sponsor and client, communication relied on phone conversations and emails. The initial intended iterative nature of the project broke down when it was discovered there was no easy way for them to analyze the work.

The prototype was easy to implement because of the technology. It is easy to email an Access database and Data Access Pages and have the client run them on any PC with a browser. The second phase of the project involved more complicated Microsoft components that could not be as easily ported to a new environment without an expert to set it up on the other side. MS SQL Server needed to be installed and configured, the .NET2 framework needed to be installed under the operating system hood in order to run the ASP pages, and MS Internet Information Services (IIS) needed to be loaded and configured. Without the ability to show progress of the project while being developed, and without the ability to gain feedback on the progress of the project, it felt more like being an offshore (remote) developer, and believe this represents the SDLC and project management methodologies used to develop and finish this project to the extent it was. Offshore development has the same issues that were encountered in this project. There was no way for D3Multisport to know if a realistic grasp of

the web site was in place that reflected the initial business requirements agreed upon in the beginning. There was no real way of tracking progress, discovering bugs, modifying or enhancing business requirements as issues arose and no way of knowing that progress was in line with what the customer really wanted.

Evaluation of Goals

The initial goals of the project were completed. The creation of the basic functionality of the web site functioned as originally communicated to the developer, but was not completed in a timely manner; nor was it detailed enough to use as a professional web site to make money off of. In order for this project to have met the true goals that both the client and developer intended, more time was needed to allow for the many set backs that happened. The deadline of the project was an issue from the beginning, but was a challenge to try and accomplish anyways.

What Went Right

Even though the project was not finished in time for the client to use in the event they wished to promote this web site; it still remains that a good framework exists for them to build from. The web site has the ability to integrate into the main business functions of the company and has a great potential of being useful in promoting the training services they offer athletes. The initial requirements were detailed enough to complete a full working web site that can be accessed by anyone with a browser connected to the Internet.

The requirements were well defined and Mike and the developer maintained excellent communication throughout the project, so requirement refinement was possible. The design of the database and system models went well also. The developers professional experience along with the design methodologies and principles learned during schooling at Regis made the design of the overall system more complete and thorough in the beginning and resulted in reduced re-design of the system when development was in progress.

The decision to use new Microsoft technology allowed easier and faster learning and development time to create the system.

What Went Wrong

- **New Beta technology**

  Since this project was on a budget, the client was not able to procure the necessary hardware and software needed to implement this project on their end. The developer was fortunate enough to have most of the hardware, software and knowledge to configure and use it all. The technology used was all Microsoft based. The developer had some older versions that did not have all the functionality needed to implement this project in the manner that was needed to be implemented in. The decision was made to become a Microsoft Developer Network (MSDN) subscriber. An MSDN subscriber has access to all of Microsoft's latest technology and software. Since the old MS software was somewhat outdated, and expired, it was decided to use the new software provided in the MSDN subscription, of which much of it was still in beta.

Although this new technology was considered easier to learn and use, it did have a few bugs in the IDE and protocols. This used up a lot of development time in order to discover there were indeed bugs in the software, and not the developer's technological shortcomings. The bugs were eventually fixed in later beta and release versions, but required the complete uninstall of existing beta versions before installing the new. This also called for many hours of backup, re-installing, re-configuration and a re-write of the ASP pages using the new version.

- **Delay of Development Software**

  The week of purchasing the MSDN subscription, Microsoft decided to change the way they handle and manage subscriptions. Microsoft decided to do away with the MSDN Universal subscription for a more specific membership depending on whether the subscriber was an architect, developer or tester. The developer managed to get in before this, so was able to get the benefits of both the Universal and Developer Team packages. Somewhere along the way, the subscription was changed to a download only, which meant that there was no large software shipment on the way. The developer called to change this after this discovery and waited about a week and a half for its arrival. The software was available via downloads over the Internet, so a large number of CD's were purchased and the software needed to start the project was downloaded. The software was then installed and configured, but not before a good couple weeks had passed.

- **Change of Jobs by Developer**

  A personal issue had come up in the middle of development that provided a good career opportunity. The initial desire was to stay at the present job until completing the Masters degree, it simply was not satisfying enough to stay there, and the better opportunity was chosen. This created an issue with the interest to develop this project in an iterative nature utilizing the Scrum process. This happened immediately during the end of the first phase of the SDLC during the implementing and testing phases. Since the developer was spending week days in hotels and weekends at home and had no static IP address to install the site on. Mike and D3multisport personnel had no way to look at and give feedback on it. As a result, the developer could only keep developing and add to the system with existing information using common sense on what was thought the system should do and behave. The controlled iterative principle of the Scrum Development process had broken down into more of an offshore, remote development process where a set of business requirements is presented to developers and a system is created from this document.

- **Inability to Present Iterative Development Efforts to Client**

  Had the ability existed to show the evolving system to the client, they might have invested the time and money to purchase the hardware and software needed to install the web site for review. If D3multisport had a

system in place, the developer could have sent them copies of the updated software at each stage and cycle for analysis. Without a system to send the initial software to, and no way to present the web site remotely, made both feel uneasy. The developer was frustrated with the inability to come up with a solution, especially since there was so much to show and was proud of the web site. It is certain the client was frustrated in not being able to scrutinize and see any progress that was performed on the web site. As a work around, screen shots of the site were zipped and emailed to the facilitator to review and then show to the client.

Project Variables and Their Impact

As discussed above, there were things that went right and things that went wrong. Both sets of variables impacted the project to varying degrees. The professional manner in which the requirements were collected, and then evolved into business rules, impacted the project by getting it off to a good start. With the business rules in place, the design of all aspects of the system seemed to go smoothly. The knowledge of how system components were to be designed was in place and understood allowing the system design models to reflect the true nature of the requirements. The requirements and design phase, along with the creation of the prototype, went as planned and went faster than expected. The main variables impacting the project in a negative manner was the new beta software and the delay in its arrival.

Another variable, that all were aware of, was the inexperience in the development language used to create the web site. Not only was this beta software, but the developer had never developed, to any real extent, with the MS .NET ASP framework using MS Visual Studio. This was considered a challenge that was overcome by many years experience as a database designer/developer and an object-oriented software engineer using various other languages. This variable did add development time, but progress was made as new techniques where discovered in implementing the development of this web site.

Testing and Implementation

Testing and implementation were performed almost exclusively by the developer. The testing was performed in an iterative and concise manner using a matrix logic approach that took all possible ways a user can enter and manipulate data in the system. The business requirements, established early on, helped provide the testing of data flow through the business rules of the system.

Summary

Although the management of this project was not allowed to follow the original project management processes intended, it still benefited from them. By considering the techniques followed within the Zachman and Scrum project management technique, enabled all pieces of design and development of the project to be considered in the beginning. The spirit of the principles got the project off to a good and complete cycle of development. The only missing piece

within the principles of these two techniques, was the lack of the incremental process, and the feedback needed from the client. By implementing these formal principles in the beginning, allowed the design and development of the project to get on the right track and allowed for re-direction as barriers and issues arose. There were problems throughout the design, development, management and implementation of this project, but by realizing the flexibility inherent within these project management principles enabled the project to proceed in a more complete and natural manner had these principles not been considered from the beginning.

Chapter 5: Lessons Learned

What was learned from the project experience?

The main lesson learned from this project was that preparation is a critical key to a successful project. Although the requirements and design of the project were understood from the beginning, the development environment was not in place. As a result, the development of the system was delayed considerably at the cost of gaining a better knowledge of the technology. It was expected to spend more time learning the technology and applying good software development practices using this technology. But, instead had to scramble and complete what could be done with what little was known of this new Microsoft technology.

There was however, a great deal learned about taking an idea and creating a solution to create a working system with all the components involved in creating and implementing a web site from inception. This new knowledge will definitely prove useful in future endeavors.

What could have been done differently?

Considering the problems incurred trying to acquire the tools and technologies needed to develop the web site, the next attempt would consider using more established and mature technologies already available, such as an earlier version of Visual Studio,.NET ASP v1.1 and SQL Server 2000. Although research indicated these older versions were more cumbersome and would have required a steeper learning curve, it would have at least made earlier progress

possible and efforts could have been carried forward when the newer technology became available.

Did the project meet initial expectations?

Aside from missing the project deadline date, the project met initial expectations to an extent. Certain pieces were left incomplete due to the fact that the customer did not need it for awhile and could be completed at a later date. All the main requirements were complete and functional for the customer to implement and extend as they deem necessary. The following is a list of the initial requirements that were established with a brief analysis of its completion status:

- Ability to enter athlete demographics into system by athlete or staff - **Yes**
- Ability for staff to create and modify specifics of main workouts - **Yes**
- Easy functionality for coaches to create and load up their training programs - **Yes**
- Easy functionality for athletes to view and log workouts from remote locations via Internet – **Not yet, but minimal effort to include.**
- Easy for coaches and athletes to make comments about workouts - **Partially**
- The system must also be easy to manage and maintain - **Yes**
- The system must also be staged for growth and additional upgrades after initial development – **Yes. The client must find someone who knows**

how to develop in .NET ASP, but the system should be easy to extend.

- Site will be easy to manage so that it can be kept up to date without complex changes – **It is very easy to manage both users and data. Administrators can easily add workout categories and types and coachs can easily create workout programs from the various workout types.**

Next stage of evolution for the project if continued

This is a list of important functionality that still needs to be added.

- **Athlete functionality** – Same as coach, but only has access to their own workout. Use same page that coach uses to drill down to each athlete, but disable athlete drop down.
- **Workout details page needs to be re-evaluated** - A very complex SQL statement brings back the results and still needs the rest of the details along with some formatting (within the SQL).
- **Tighter control over the security** - Currently if a user has an account, they can get into the admin, coach or athlete pages, even though the group type will re-direct the user to a different page.
- **The ability to communicate back and forth between coach and athlete** - This is merely a matter of adding a text (blob) column to the workout table. Since the coach and athlete use the same ASP page, they can communicate on this page.

- **Add more columns to the athlete and coach tables** - This was held off since there was more concern with the functionality and flow of the web site first. This was considered an easy but time consuming addition to the site since it only requires columns in the database and a simple ASP page accessing these columns.

There are many other technological additions that would be nice to incorporate into the site, but as for functionality, this is all that is left, other than access to the database and the site via the web. But this is a technology issue that involved buggy new MS technology and not having access to a static IP address. The bugs are resolved in the new release of Visual Studio 2005 Team Edition just released.

Conclusions & Recommendations

This project had some issues in implementation aspects, in that there were problems leading to development and project management areas. The delay of the software, new technology in regards to learning and bug issues made problems that caused the deadline to be missed and the proper management of the project to breakdown. This system would have been more complete had there been more time spent establishing a means for feedback on the system during the phases of development.

As it stands, the web site provides a good framework to work from for future development. It is recommended the web site be evaluated by the client

and have feedback provided to the development staff for future re-design and enhancements.

Summary

Overall, this project was a success in that all parties came out ahead with either knowledge or a functional web site that can grow into a central business tool for D3multisport to use in everyday business practices. With full implementation of development of the suggested items mentioned above, this site would be robust enough to take into the field for paying clients to use and benefit from.

References and Online Course Material Used for Project

Cornell Paul. “Working with Data Access Pages” July 2001. MSDN Library. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnofftalk/html/office07052001.asp

Roberts, Mark. “Scripting the Data Source Control in Data Access Pages”. April 2002 MSDN Library. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnacc2k2/html/ODC_acc10_DSCOM.asp

Roberts, Mark. “Deploying Data Access Pages on the Internet or Your Intranet”. September 2001. Access 2002 VBA Handbook. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnacc2k2/html/deploydap.asp

Khor, Siew-Moi. “Beginning Web Development in Access 2002”. December 2001. MSDN Library. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnacc2k2/html/deploydap.asp

Microsoft Technet. “Windows Server 2003 Technical Reference”. March 2003. http://www.microsoft.com/resources/documentation/WindowsServ/2003/all/techref/en-us/Default.asp?url=/Resources/Documentation/windowsserv/2003/all/techref/en-us/w2k3tr_sec_over.asp

“Scrum It’s About Common Sense.” Online. Internet. Available: http://www.controlchaos.com/

Sutherland, Jeff. “Scrum Evolution: Type A, B, and C Sprints.” Online. Internet. March 11 2005. Available: http://jeffsutherland.com/scrum/2005/03/scrum-evolution-type-b-and-c-sprints.html

Sutherland, Jeff. “Voices From the Past: Uncle Bob on Project Management” April 15 2004. Online. Internet. Available: http://jeffsutherland.com/scrum/2004/04/voices-from-past-uncle-bob-on-project.html

Niederst, Jennifer. September 2001. “Web Design in a Nutshell”. 2nd Edition. O’Reilly & Associates, Inc.

Treese, Winfield. 2000. “Designing Systems for Internet Commerce”. Addison-Wesley.

Garfinkel, Simpson. 2002. “Web Security Privacy & Commerce”. 2nd Edition O’Reilly & Associates, Inc.

Champ Heather, “Web Design Basics”. Online. Available: http://www.microsoft.com/mind/1297/webdesign.asp

“The Zachman Institute for Framework Advancement”. 2005. Online. Internet. 2005. Available: http://www.zifa.com/?OVRAW=zachman%20framework&OVKEY=zachman%20framework&OVMTC=standard

The Systems and Software Consortium. “Architecture Frameworks”. Online. Internet. Available: http://www.software.org/pub/architecture/zachman.asp

“Zachman Framework”. Online. Internet. 7 Sep 2004. Available: http://www.mywiseowl.com/articles/Zachman_framework

Hay, David. “Essential Strategies. The Zachman Framework: An Introduction”. Online. Internet. Available: http://www.tdan.com/i001fe01.htm

Zachman, John. “Concepts of the Framework for Enterprise Architecture”. Online. Internet. 2004. Available: http://members.ozemail.com.au/~visible/papers/zachman3.htm

Whitten Jeffrey. 2001. “System Analysis and Design Methods”. 5th Edition McGraw-Hill Companies, Inc.

Booch, Jacobson & Rumbaugh. 2000. “Using UML, Software Engineering with Objects and Components”. Addison-Wesley.

Schwalbe, Kathy. 2006. “Information Technology Project Management”. 4th Edition Thomson Course Technology.

MacDonald, Matthew. 2005. “Pro ASP.NET 2.0 in C# 2005”. Apress.

Esposito, Dino. 2005. “Introducing Microsoft ASP.NET 2.0”. Microsoft Press.

Larman, Craig. 2002. “Applying UML and Patterns” 2nd Edition Prentice Hall.

Rizzo, Thomas. 2005. “Pro SQL Server 2005”. Apress.

Course 2310: Developing Microsoft® ASP.NET Web Applications Using Visual Studio® .NET

Microsoft ASP.NET Programming with Microsoft Visual Basic .NET Version 2003 Step By Step https://www.microsoftelearning.com/eLearning/referenceDetail.aspx?productId=40305

Building Web Solutions with ASP.NET and ADO.NET
https://www.microsoftelearning.com/eLearning/referenceDetail.aspx?productId=40749

Appendix A (Definition of Terms)

Triathlon – A sporting event in which athletes compete in swimming, running and bicycling against each other.

Workout category – A specific type of workout such as running, swimming, biking and weight lifting.

Workout – A specific workout within a category created for a specific purpose. For example, running workouts can be short and fast to gain speed, or long and slower to gain endurance. Each workout has a purpose and is integrated into a workout program for a specific reason.

Workout Program– A combination of workout categories carried over a specified period of time established by the coach to achieve a goal.

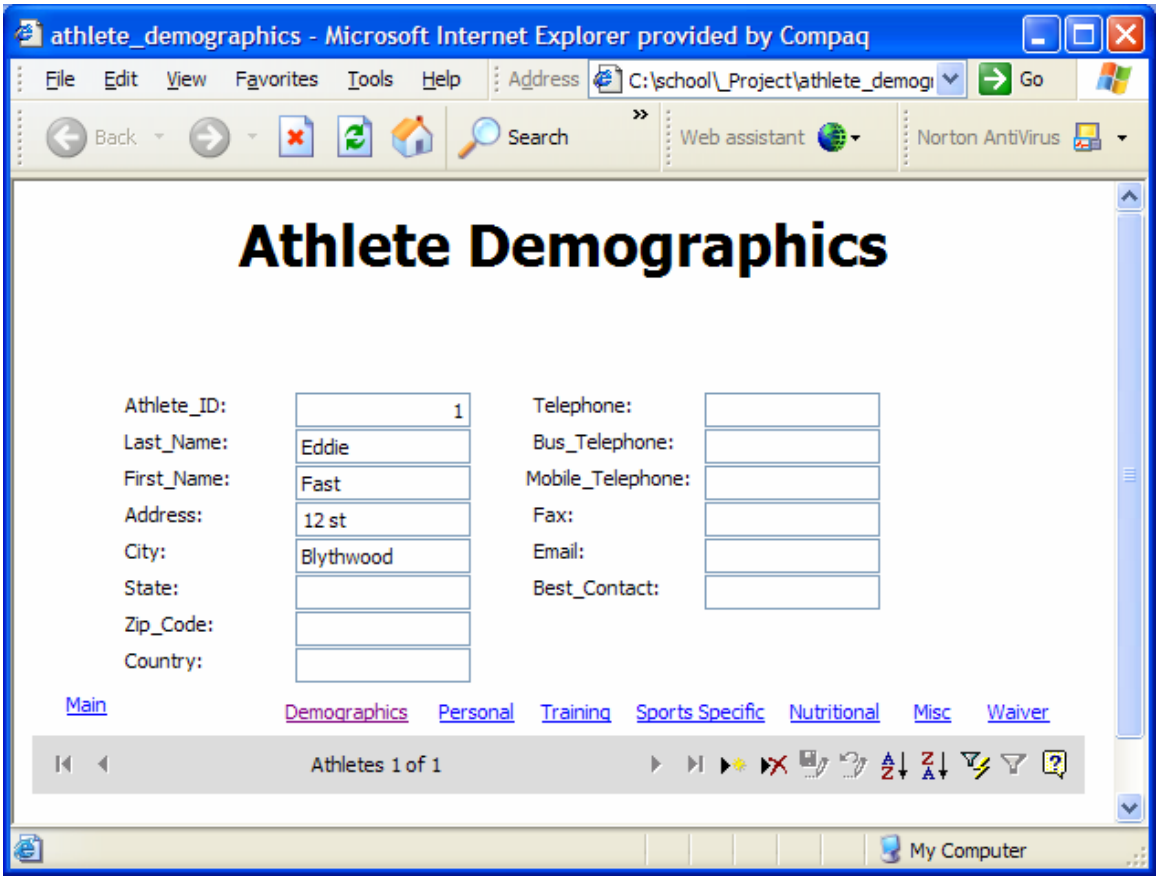
Athlete – The individual performing the exercise and training.

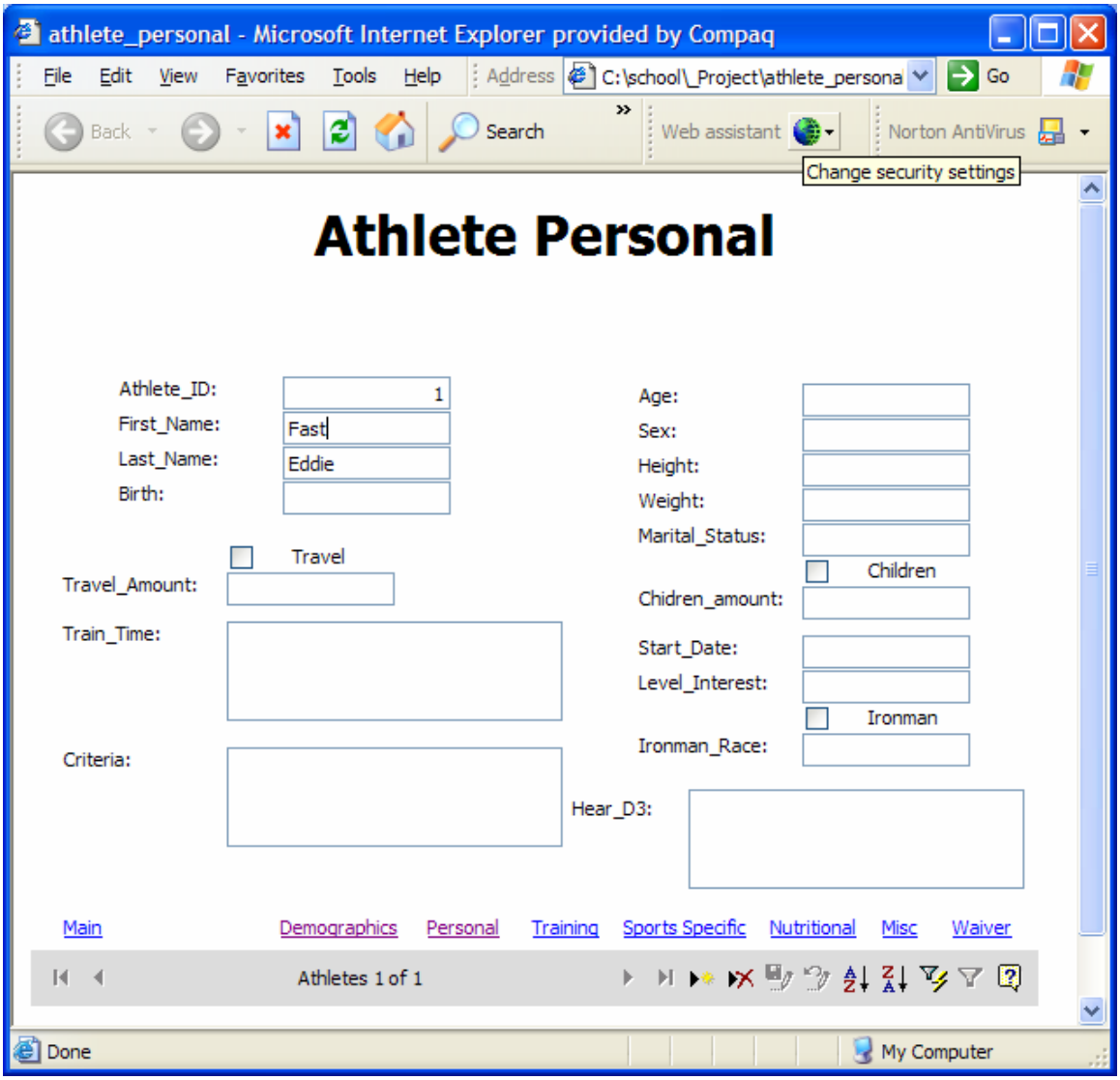
Staff – Employees of D3multisport that enter athlete and coach specifics and create the base workouts.

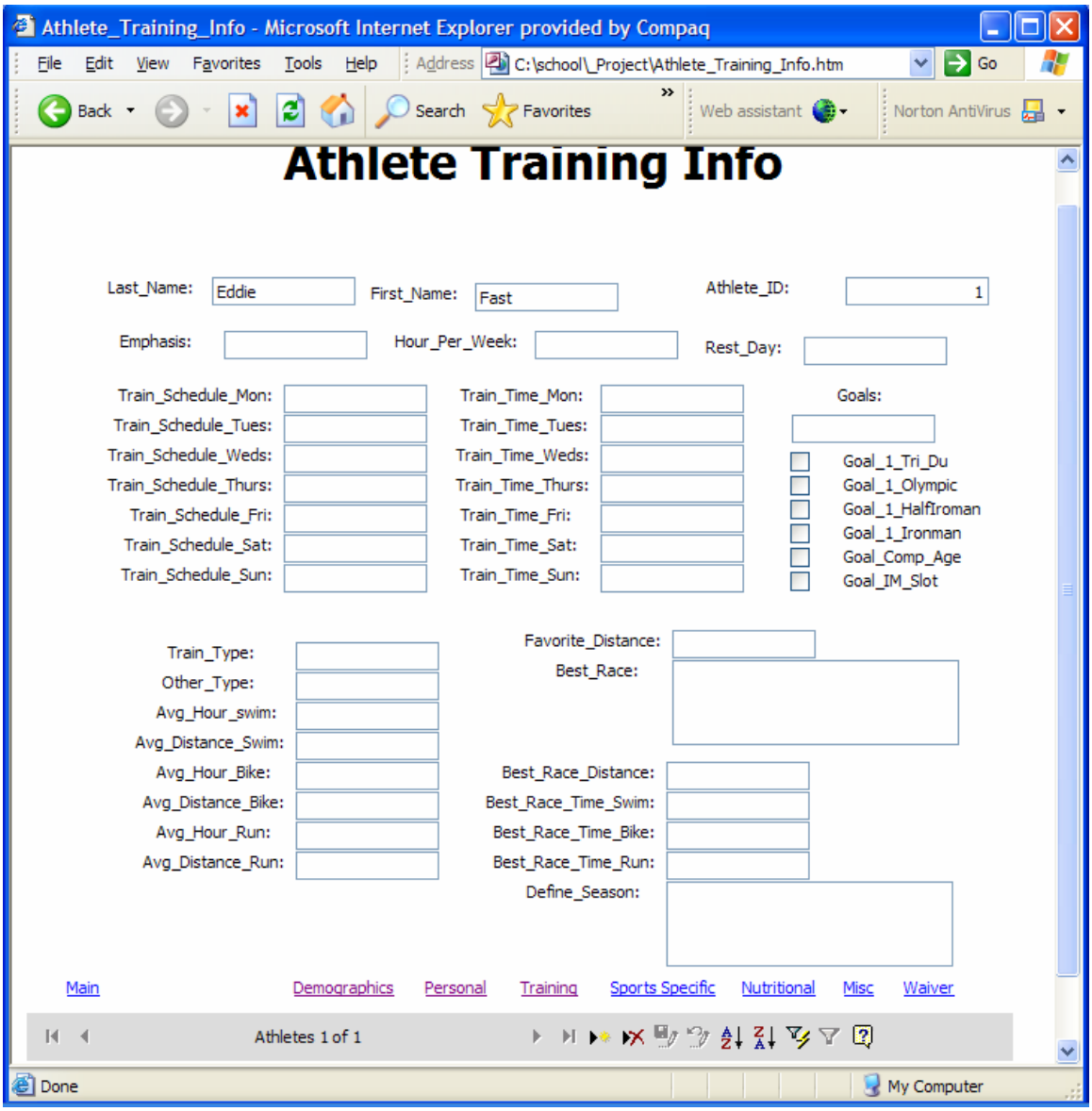
Coach – Employed by D3multisport to create training programs for athletes and to monitor and consult them during the duration of the training program.

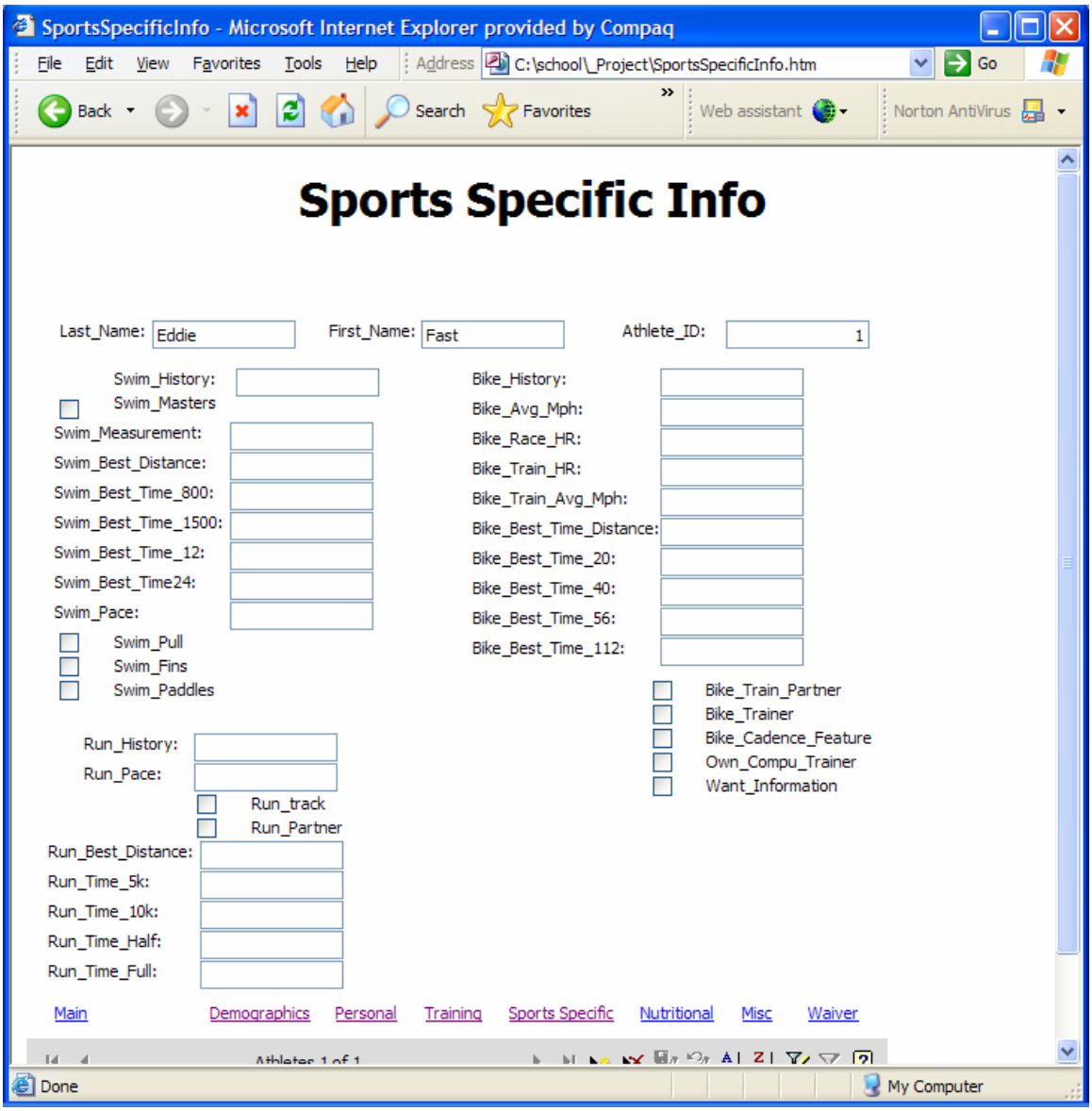
Appendix B (Screen Shots)

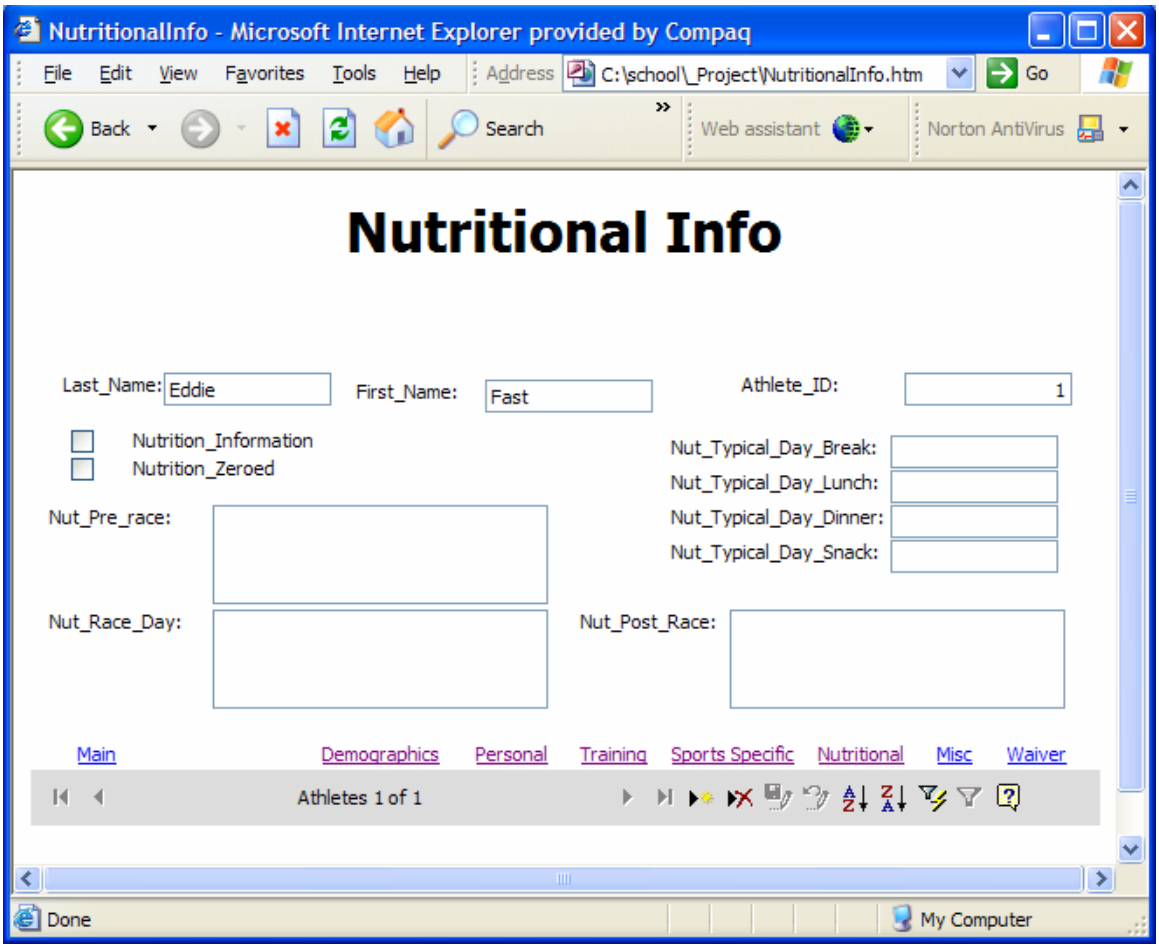
Screen Shots of Prototype Data Access Pages

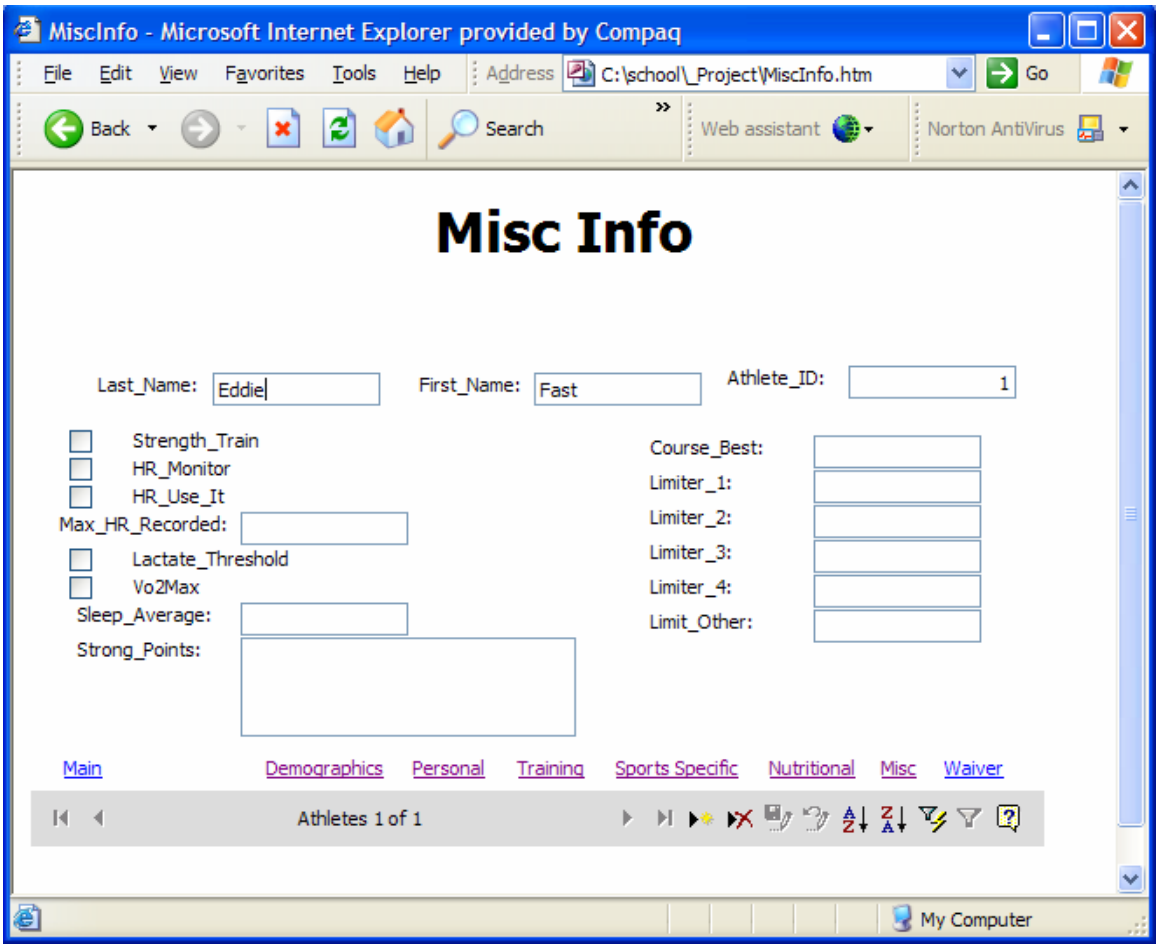



athlete_personal - Microsoft Internet Explorer provided by Compaq
File Edit View Favorites Tools Help
Address C:\school_Project\athlete_persona
Go
Back
Search
Web assistant
Norton AntiVirus
Change security settings
Athlete Personal
Athlete_ID:
1
First_Name:
Fast
Last_Name:
Eddie
Birth:
Age:
Sex:
Height:
Weight:
Marital_Status:
Travel
Travel_Amount:
Children
Chidren_amount:
Train_Time:
Start_Date:
Level_Interest:
Ironman
Ironman_Race:
Criteria:
Hear_D3:
Main
Demographics
Personal
Training
Sports Specific
Nutritional
Misc
Waiver
Athletes 1 of 1
Done
My Computer


Athlete_Training_Info - Microsoft Internet Explorer provided by Compaq
File Edit View Favorites Tools Help
Address C:\school_Project\Athlete_Training_Info.htm
Go
Back
Search
Favorites
Web assistant
Norton AntiVirus
Athlete Training Info
Last_Name: Eddie
First_Name: Fast
Athlete_ID: 1
Emphasis:
Hour_Per_Week:
Rest_Day:
Train_Schedule_Mon:
Train_Schedule_Tues:
Train_Schedule_Weds:
Train_Schedule_Thurs:
Train_Schedule_Fri:
Train_Schedule_Sat:
Train_Schedule_Sun:
Train_Time_Mon:
Train_Time_Tues:
Train_Time_Weds:
Train_Time_Thurs:
Train_Time_Fri:
Train_Time_Sat:
Train_Time_Sun:
Goals:
Goal_1_Tri_Du
Goal_1_Olympic
Goal_1_HalfIroman
Goal_1_Ironman
Goal_Comp_Age
Goal_IM_Slot
Train_Type:
Other_Type:
Avg_Hour_swim:
Avg_Distance_Swim:
Avg_Hour_Bike:
Avg_Distance_Bike:
Avg_Hour_Run:
Avg_Distance_Run:
Favorite_Distance:
Best_Race:
Best_Race_Distance:
Best_Race_Time_Swim:
Best_Race_Time_Bike:
Best_Race_Time_Run:
Define_Season:
Main
Demographics
Personal
Training
Sports Specific
Nutritional
Misc
Waiver
Athletes 1 of 1
Done
My Computer

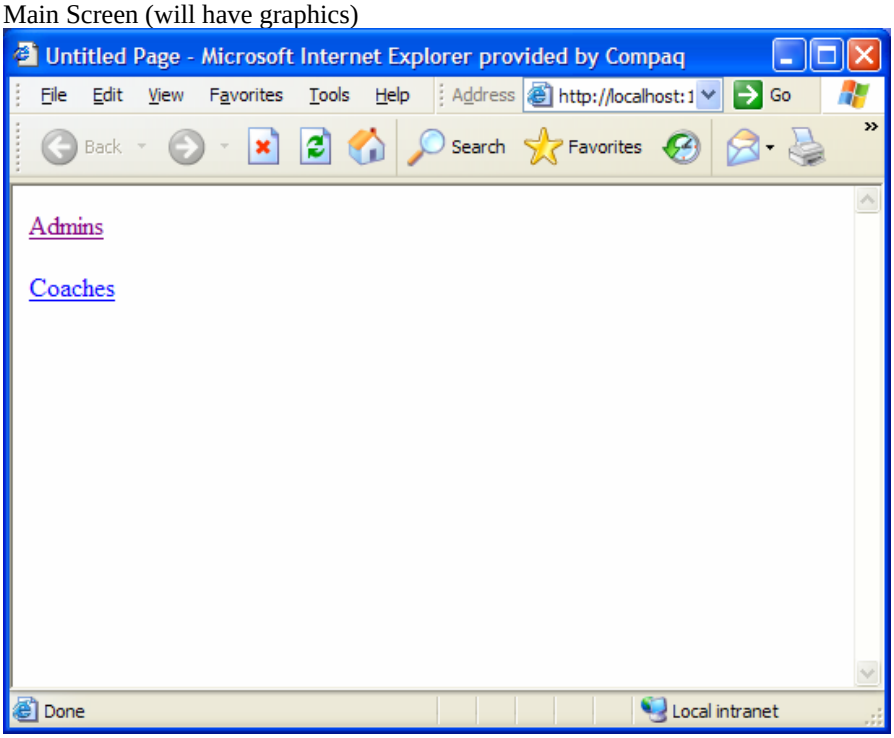

SportsSpecificInfo - Microsoft Internet Explorer provided by Compaq
File Edit View Favorites Tools Help
Address C:\school_Project\SportsSpecificInfo.htm
Go
Back
Search
Favorites
Web assistant
Norton AntiVirus
Sports Specific Info
Last_Name: Eddie
First_Name: Fast
Athlete_ID: 1
Swim_History:
Swim_Masters
Swim_Measurement:
Swim_Best_Distance:
Swim_Best_Time_800:
Swim_Best_Time_1500:
Swim_Best_Time_12:
Swim_Best_Time24:
Swim_Pace:
Swim_Pull
Swim_Fins
Swim_Paddles
Bike_History:
Bike_Avg_Mph:
Bike_Race_HR:
Bike_Train_HR:
Bike_Train_Avg_Mph:
Bike_Best_Time_Distance:
Bike_Best_Time_20:
Bike_Best_Time_40:
Bike_Best_Time_56:
Bike_Best_Time_112:
Bike_Train_Partner
Bike_Trainer
Bike_Cadence_Feature
Own_Compu_Trainer
Want_Information
Run_History:
Run_Pace:
Run_track
Run_Partner
Run_Best_Distance:
Run_Time_5k:
Run_Time_10k:
Run_Time_Half:
Run_Time_Full:
Main
Demographics
Personal
Training
Sports Specific
Nutritional
Misc
Waiver
Done
My Computer


NutritionalInfo - Microsoft Internet Explorer provided by Compaq
File Edit View Favorites Tools Help
Address C:\school_Project\NutritionalInfo.htm
Go
Back
Search
Web assistant
Norton AntiVirus
Nutritional Info
Last_Name: Eddie
First_Name: Fast
Athlete_ID: 1
Nutrition_Information
Nutrition_Zeroed
Nut_Typical_Day_Break:
Nut_Typical_Day_Lunch:
Nut_Typical_Day_Dinner:
Nut_Typical_Day_Snack:
Nut_Pre_race:
Nut_Race_Day:
Nut_Post_Race:
Main
Demographics
Personal
Training
Sports Specific
Nutritional
Misc
Waiver
Athletes 1 of 1
Done
My Computer

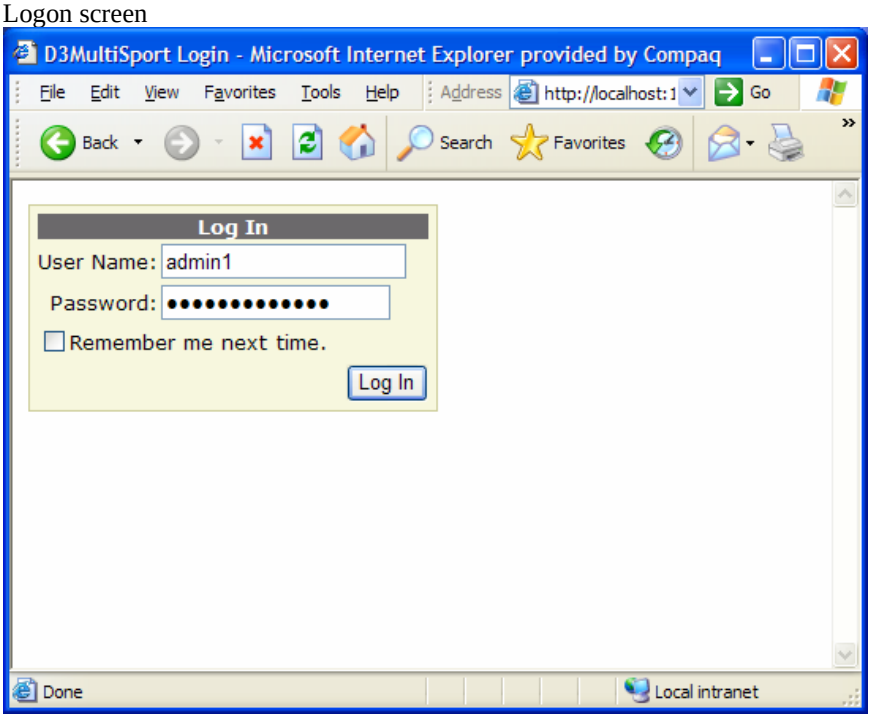

MiscInfo - Microsoft Internet Explorer provided by Compaq
File Edit View Favorites Tools Help
Address C:\school_Project\MiscInfo.htm
Go
Back
Search
Web assistant
Norton AntiVirus
Misc Info
Last_Name: Eddie
First_Name: Fast
Athlete_ID: 1
Strength_Train
HR_Monitor
HR_Use_It
Max_HR_Recorded:
Lactate_Threshold
Vo2Max
Sleep_Average:
Strong_Points:
Course_Best:
Limiter_1:
Limiter_2:
Limiter_3:
Limiter_4:
Limit_Other:
Main
Demographics
Personal
Training
Sports Specific
Nutritional
Misc
Waiver
Athletes 1 of 1
My Computer

Web Site Screen Shots (MS SQL Server and ASP .NET2

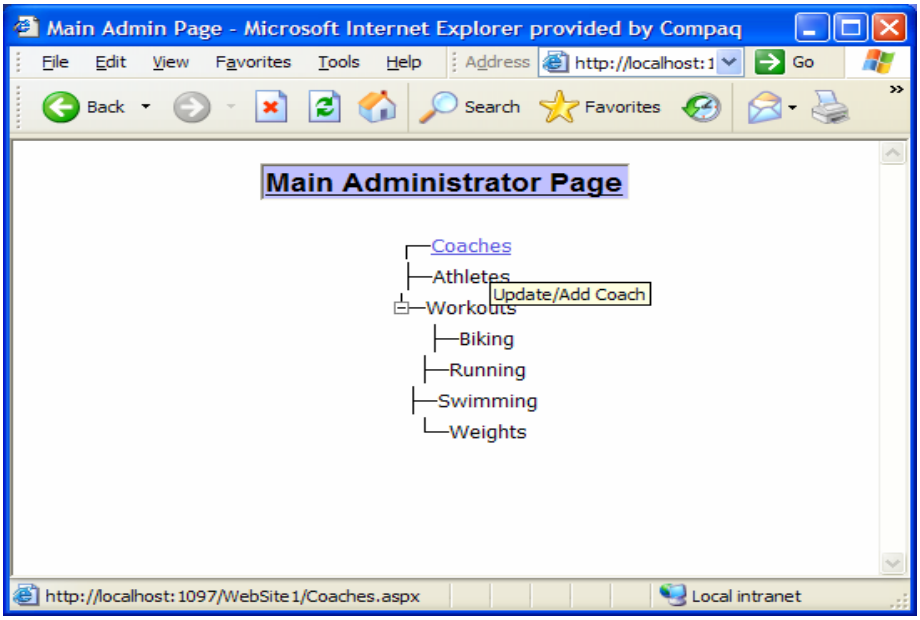
Main Screen (will have graphics)



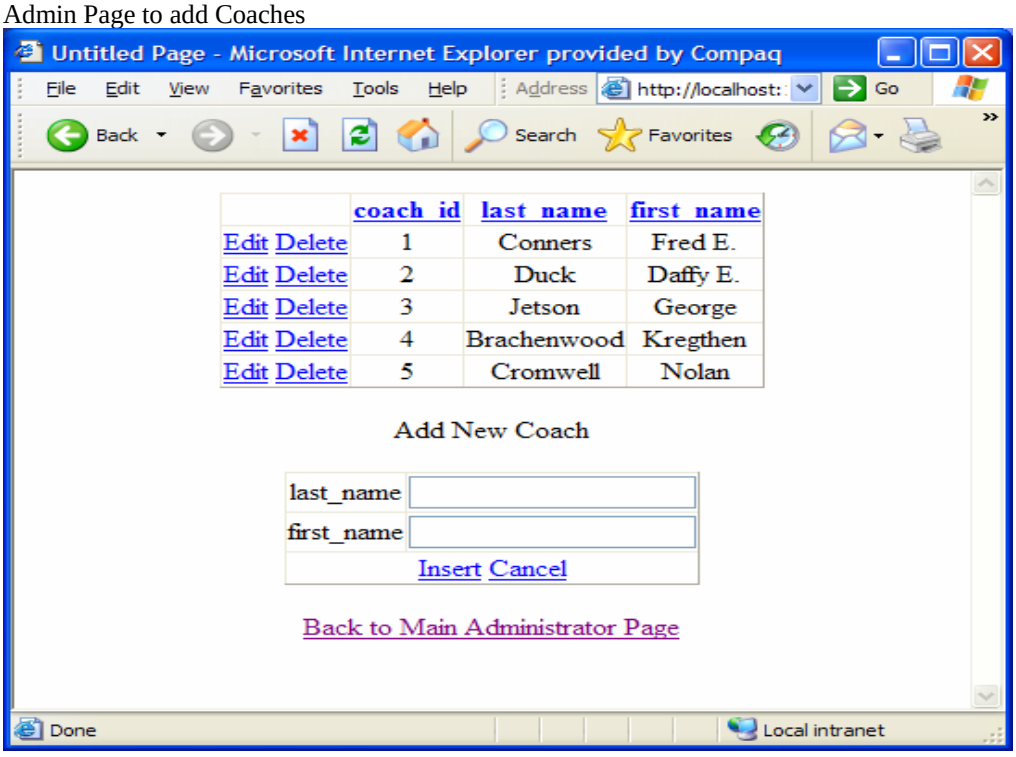
Logon screen



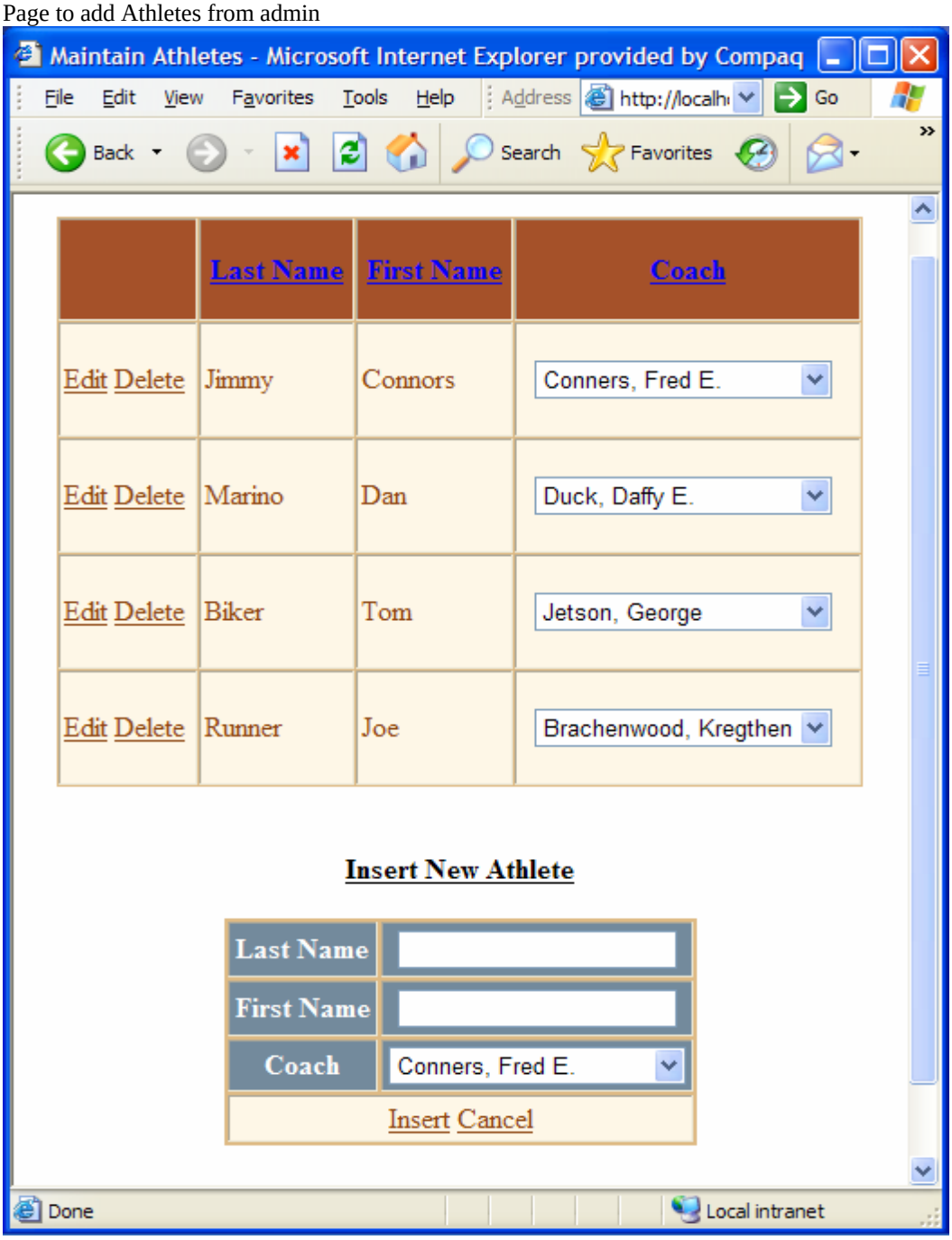
Admin Screen



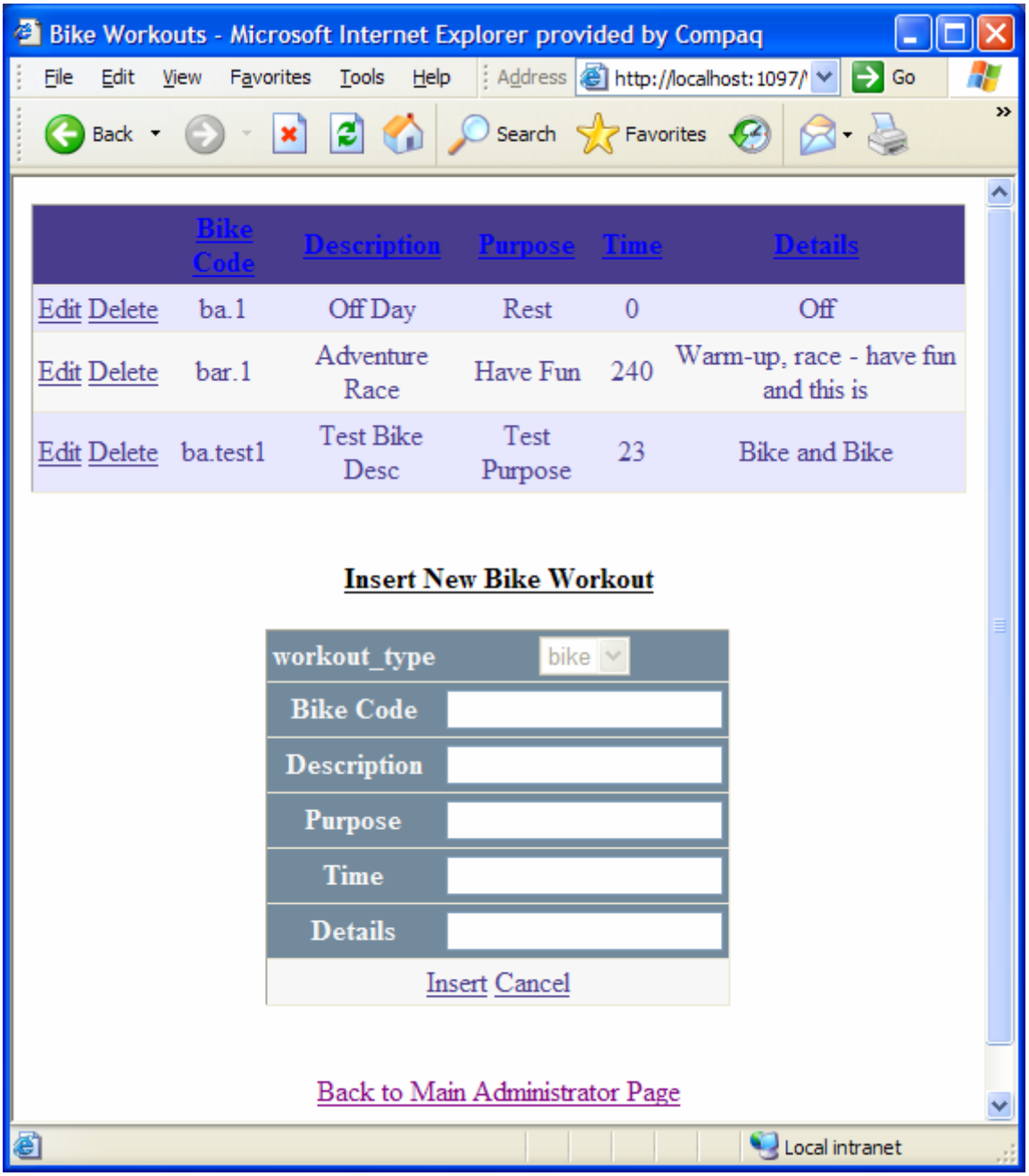
Admin Page to add Coaches



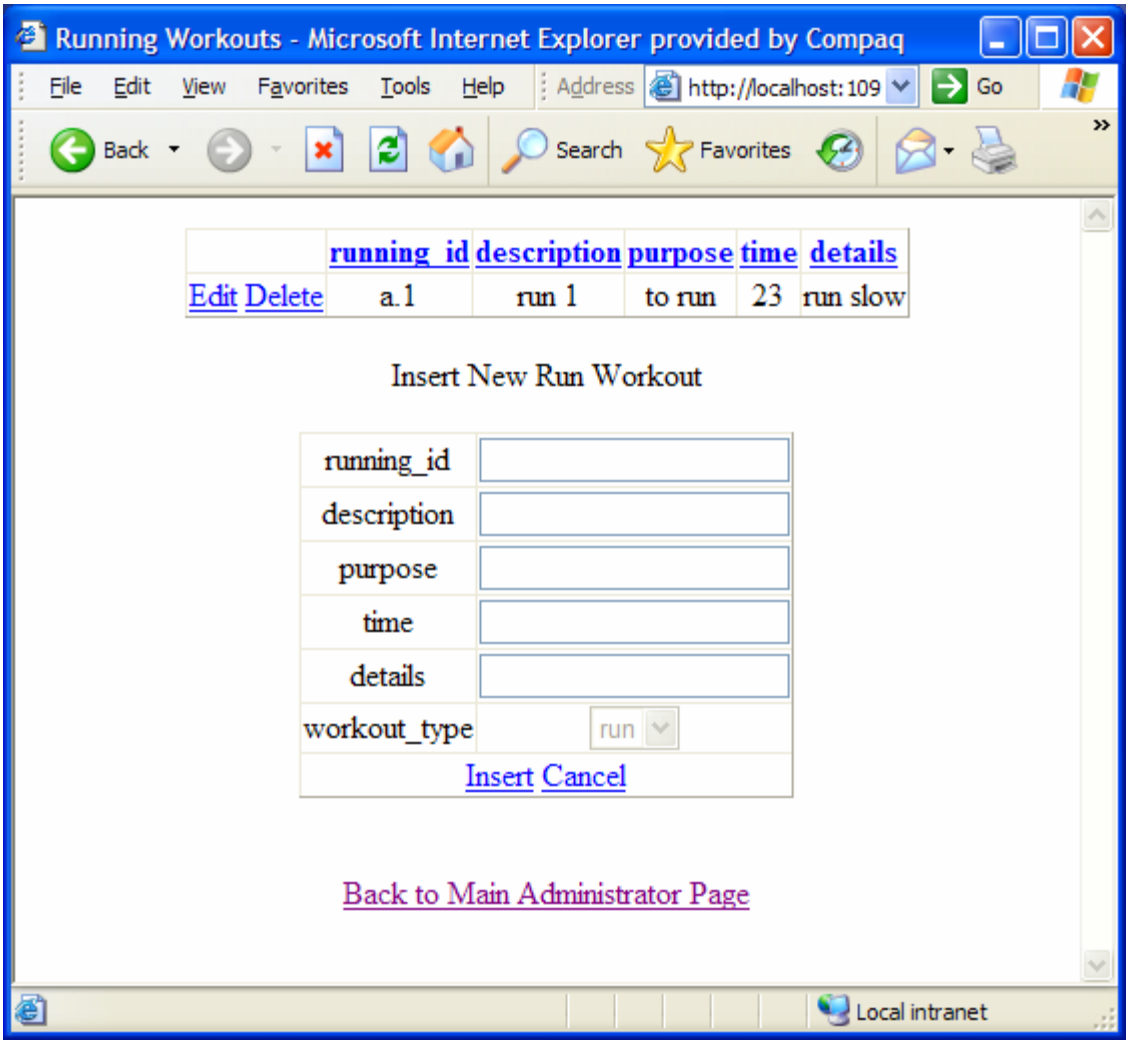
Page to add Athletes from admin



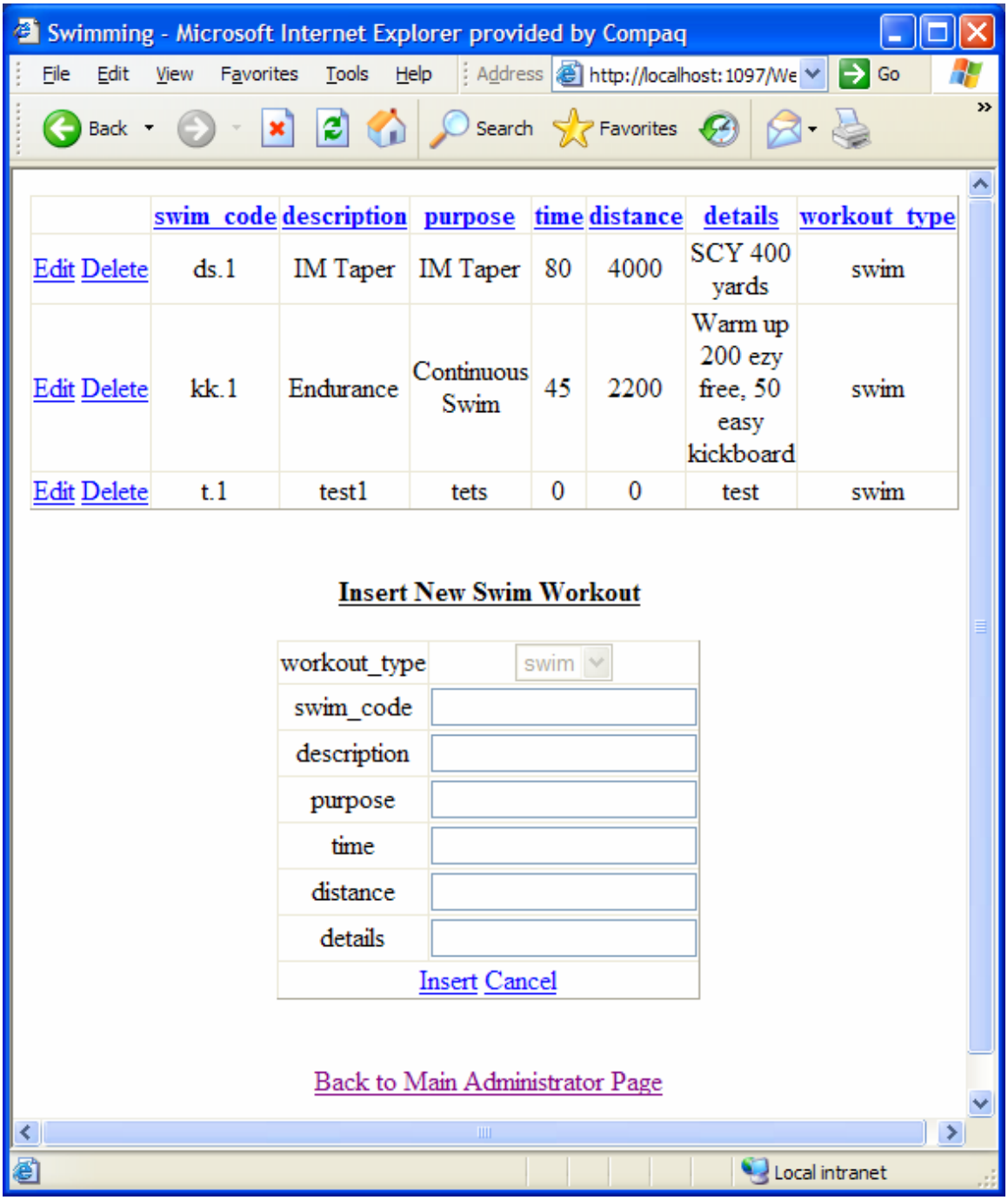
Admin Add bike workouts



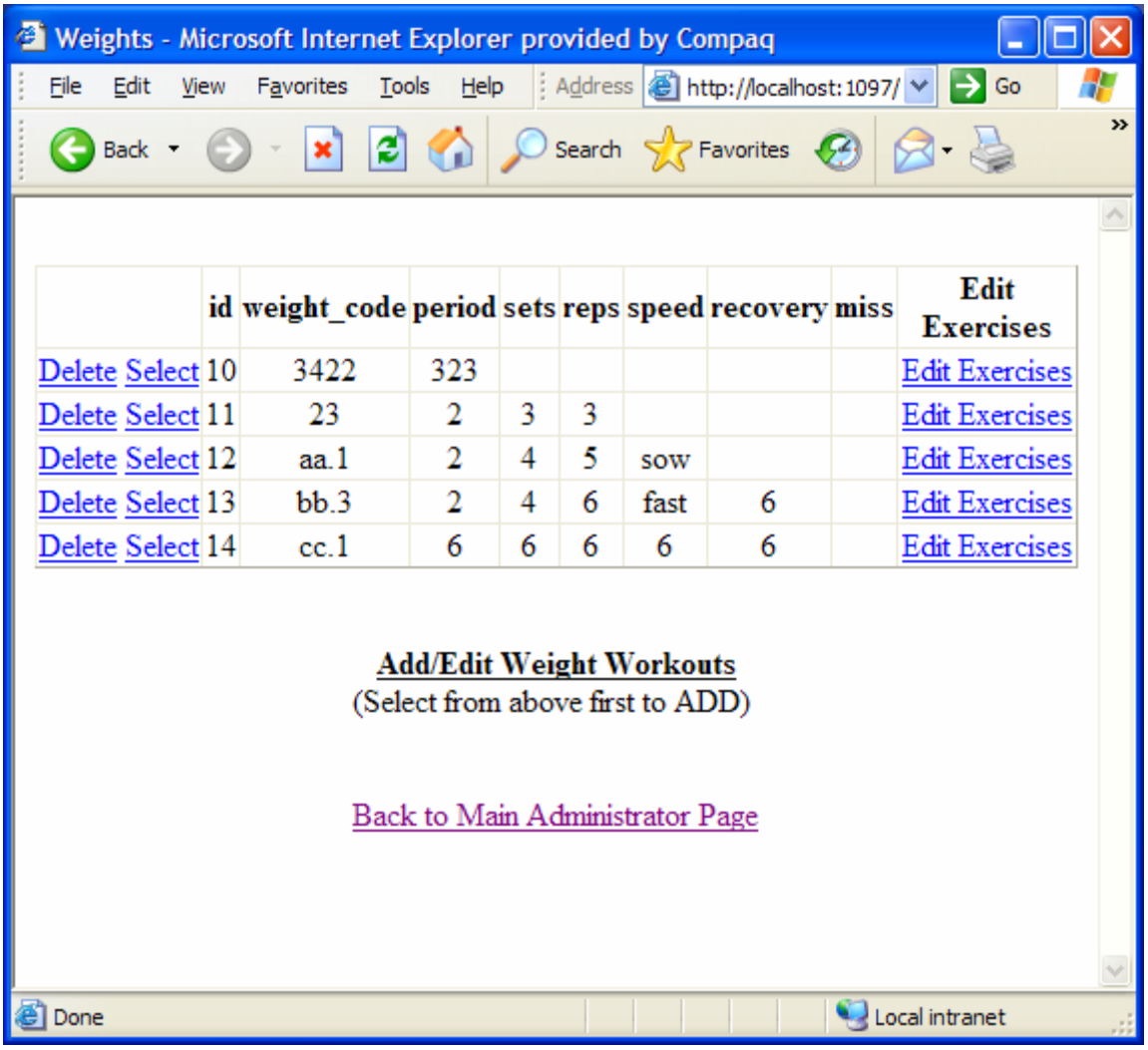
Admin Add Run workouts



Admin Add Swim workouts



Admin add weight workouts

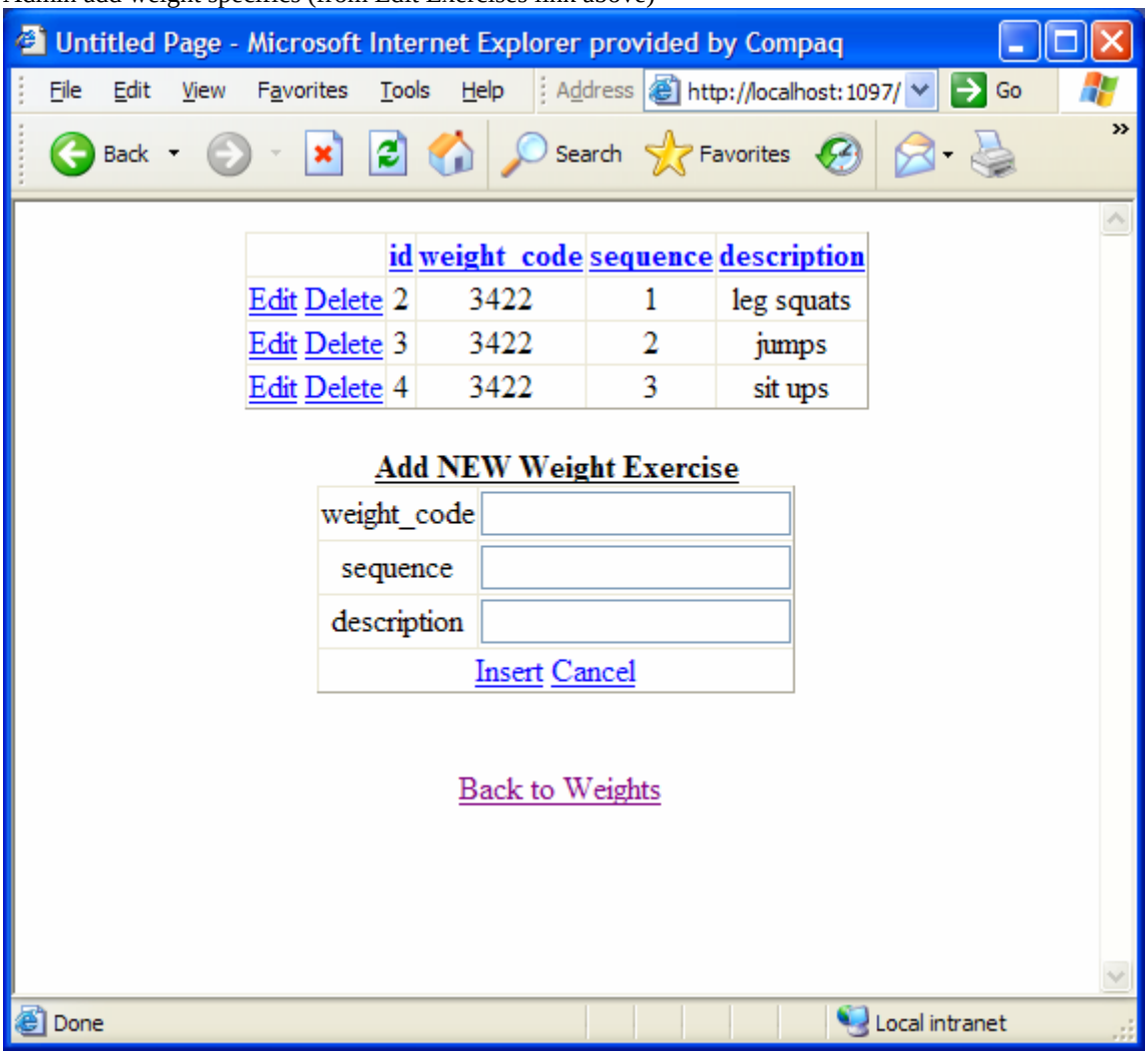


	id	weight_code	period	sets	reps	speed	recovery	miss	Edit Exercises
Delete Select	10	3422	323						Edit Exercises
Delete Select	11	23	2	3	3				Edit Exercises
Delete Select	12	aa.1	2	4	5	sow			Edit Exercises
Delete Select	13	bb.3	2	4	6	fast	6		Edit Exercises
Delete Select	14	cc.1	6	6	6	6	6		Edit Exercises

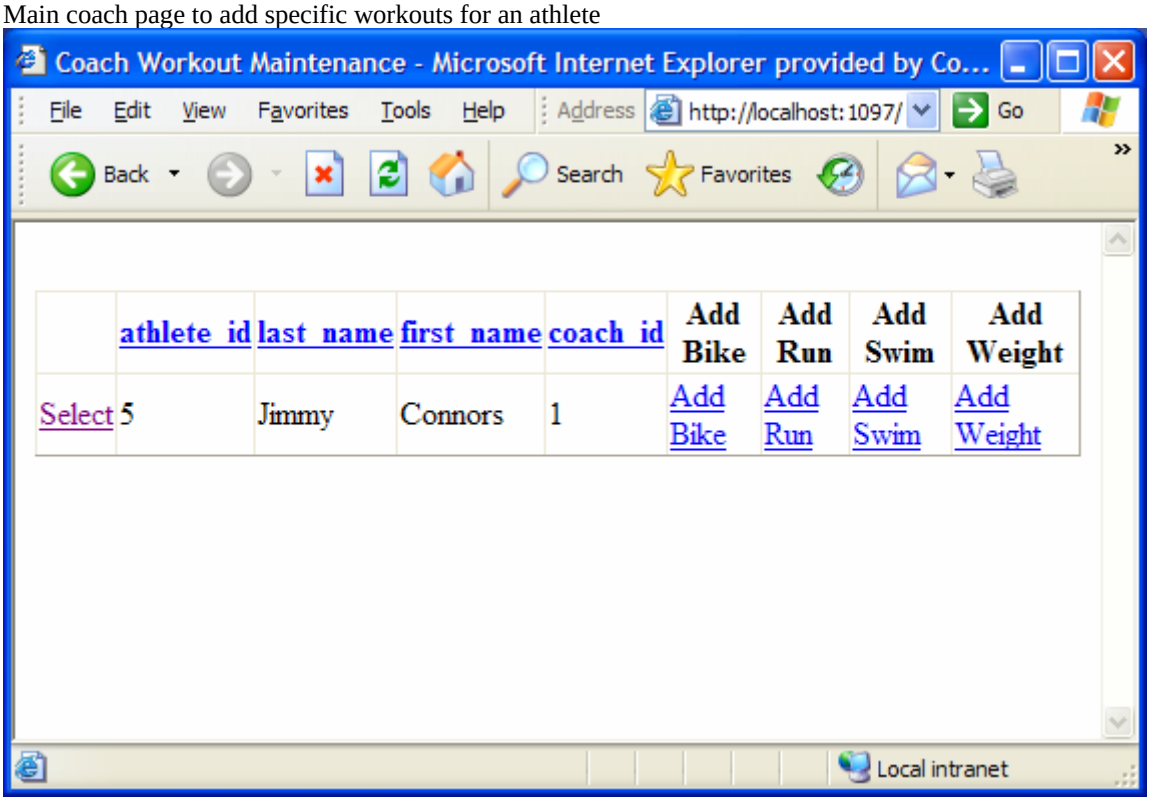
Add/Edit Weight Workouts
(Select from above first to ADD)

Back to Main Administrator Page

Admin add weight specifics (from Edit Exercises link above)

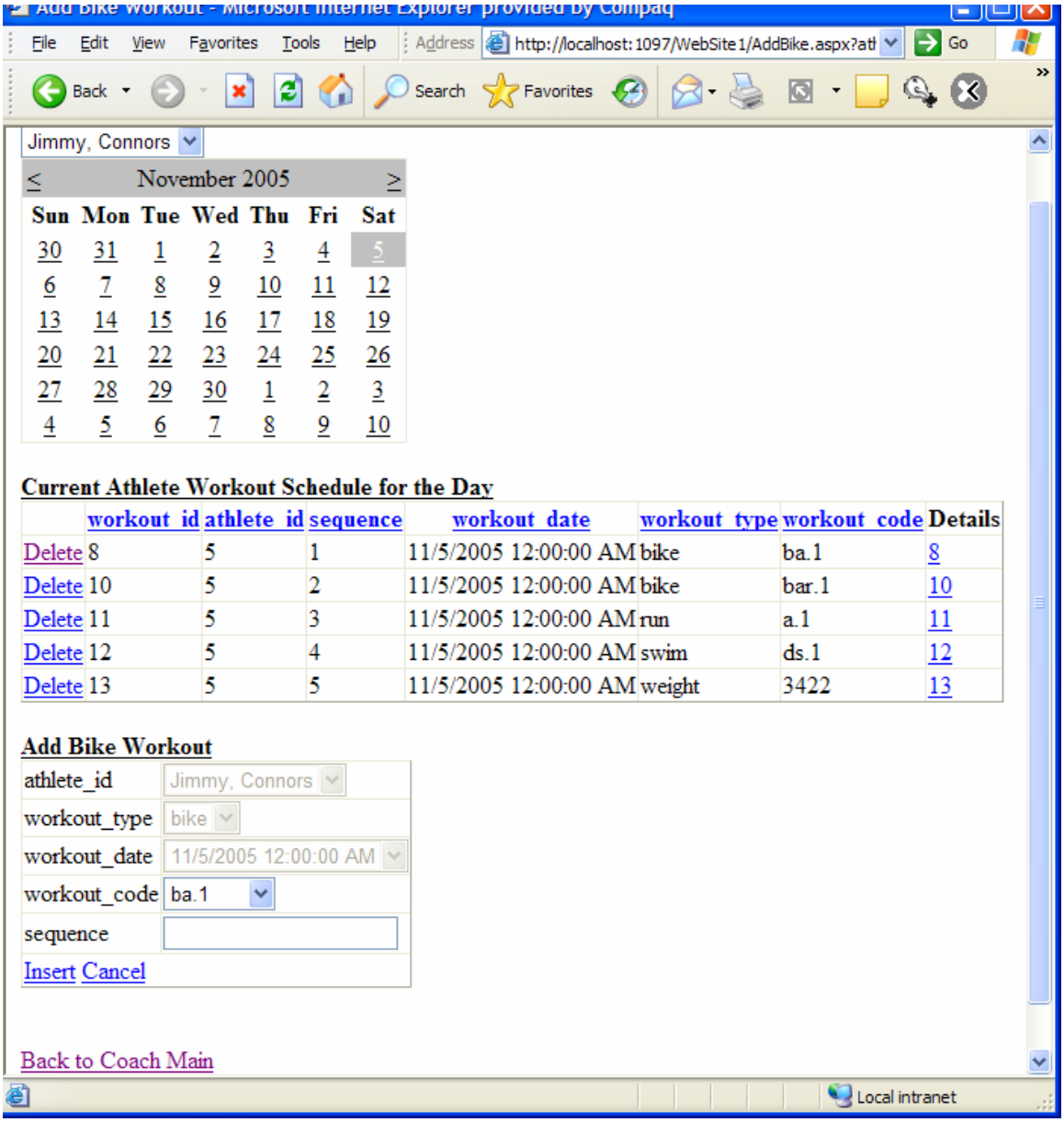


Main coach page to add specific workouts for an athlete

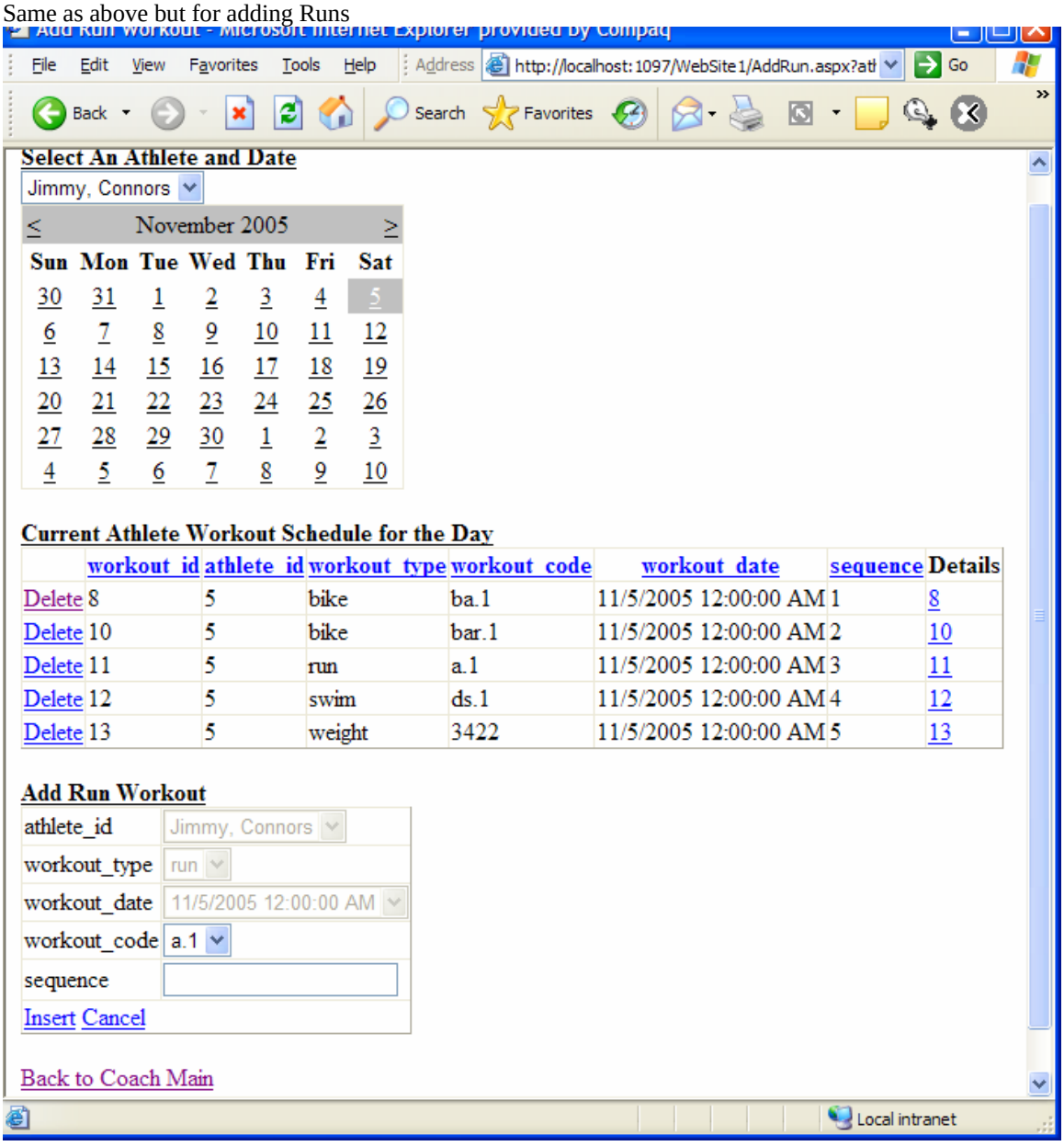


	athlete_id	last_name	first_name	coach_id	Add Bike	Add Run	Add Swim	Add Weight
Select	5	Jimmy	Connors	1	Add Bike	Add Run	Add Swim	Add Weight

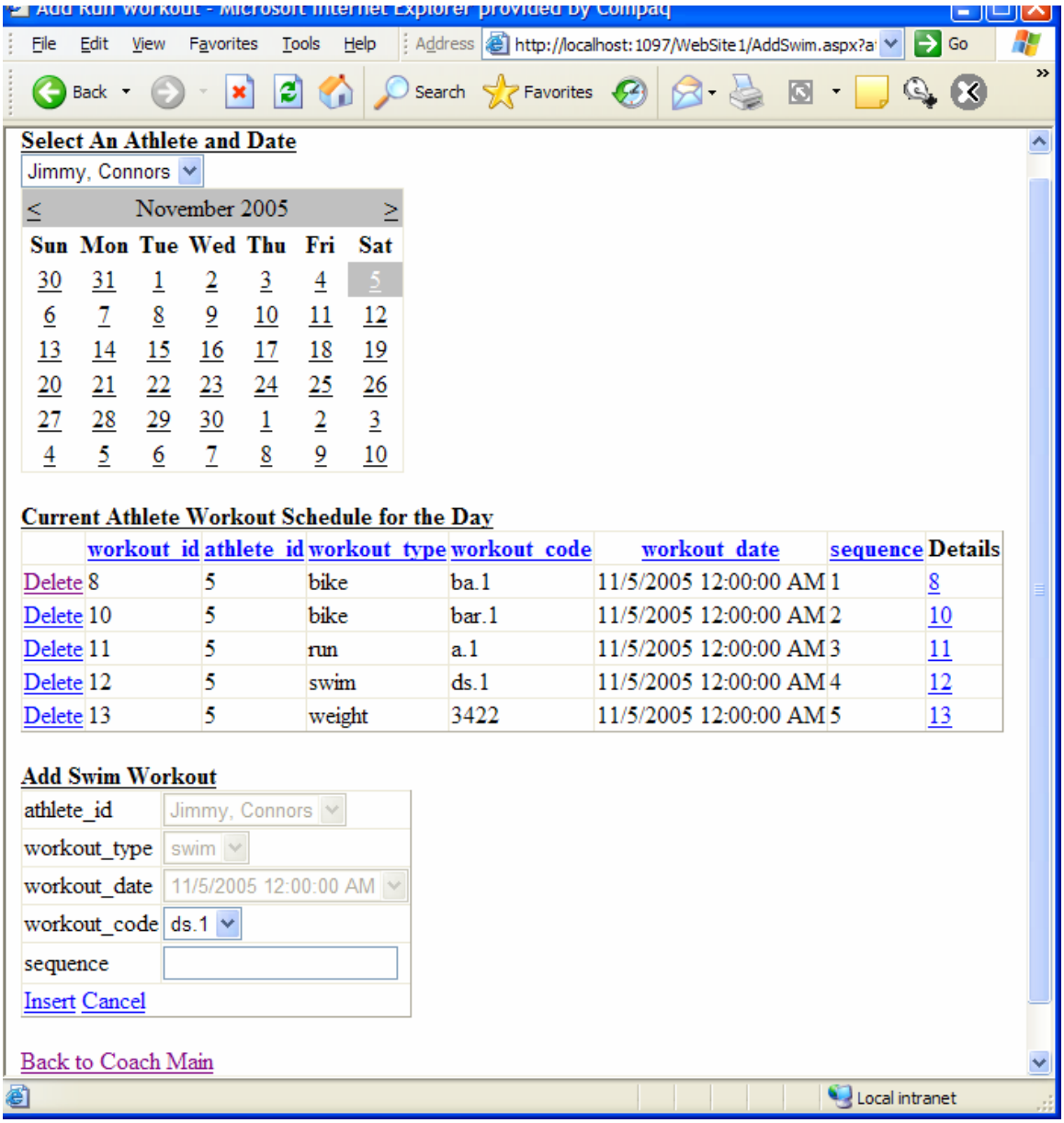
Design / view workout schedule for day for Jimmy Connors. Coach or athlete only needs to choose workout code and sequence to add to the days workout schedule. Allows athlete to view details of workout by selecting details (shown below) This specific view is to add a bike workout.



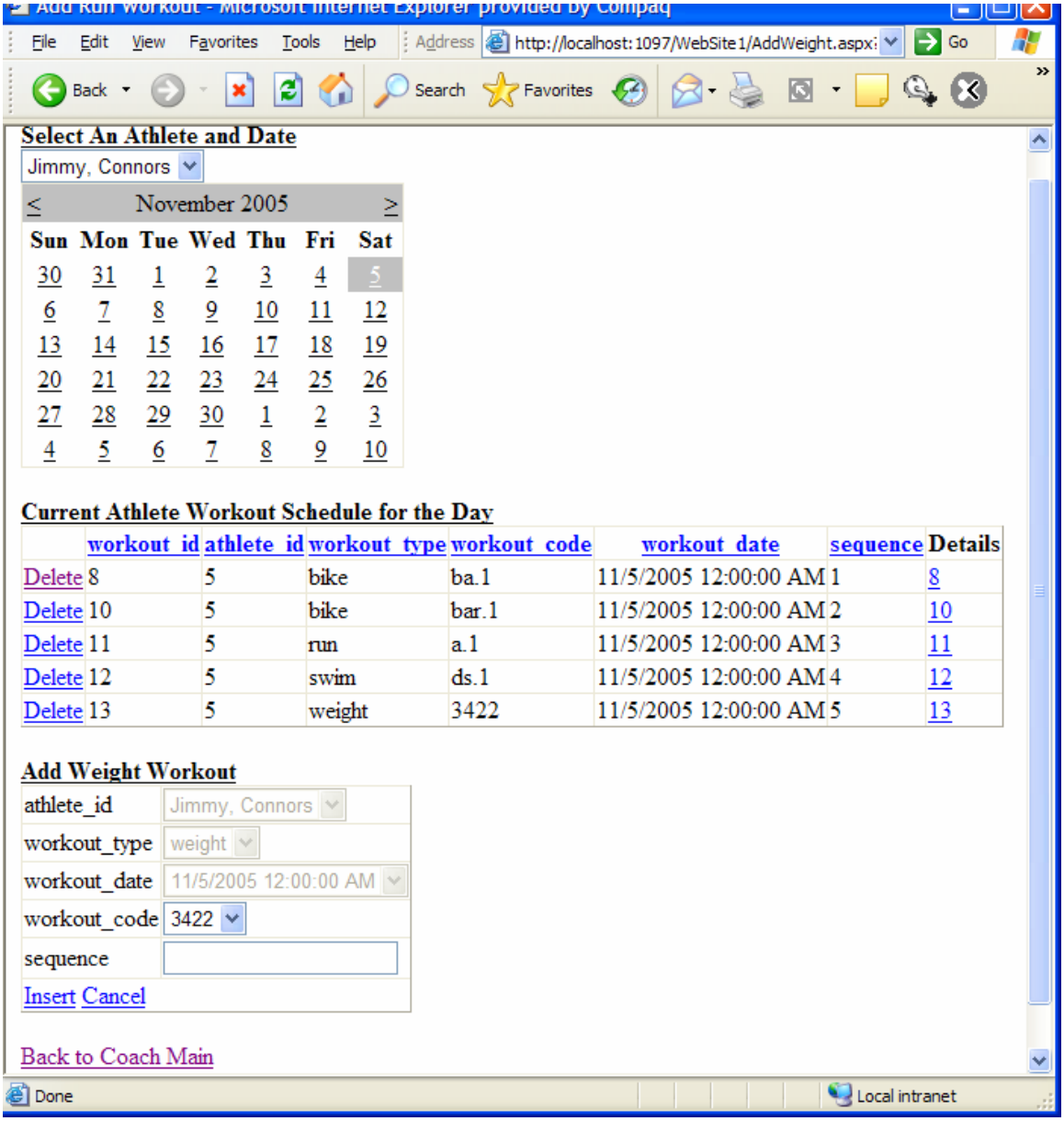
Same as above but for adding Runs



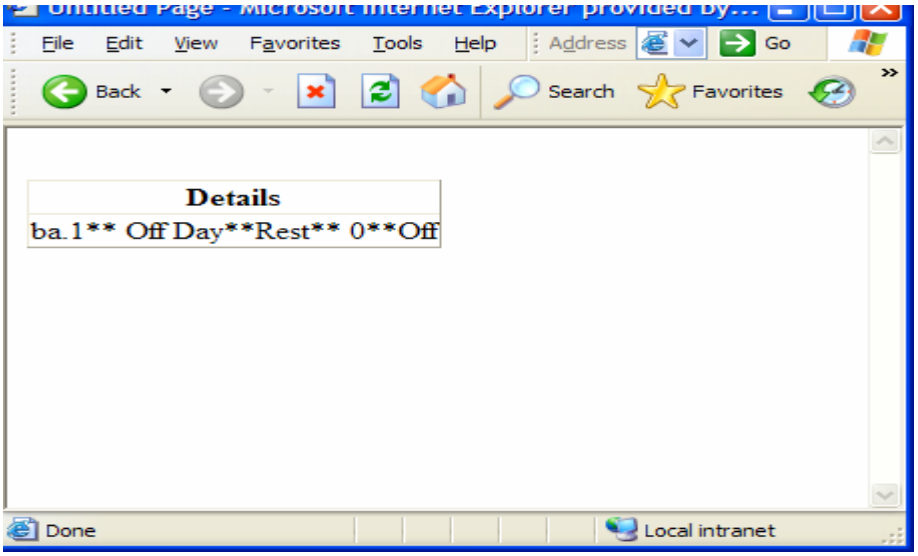
Same as above but for adding Swim



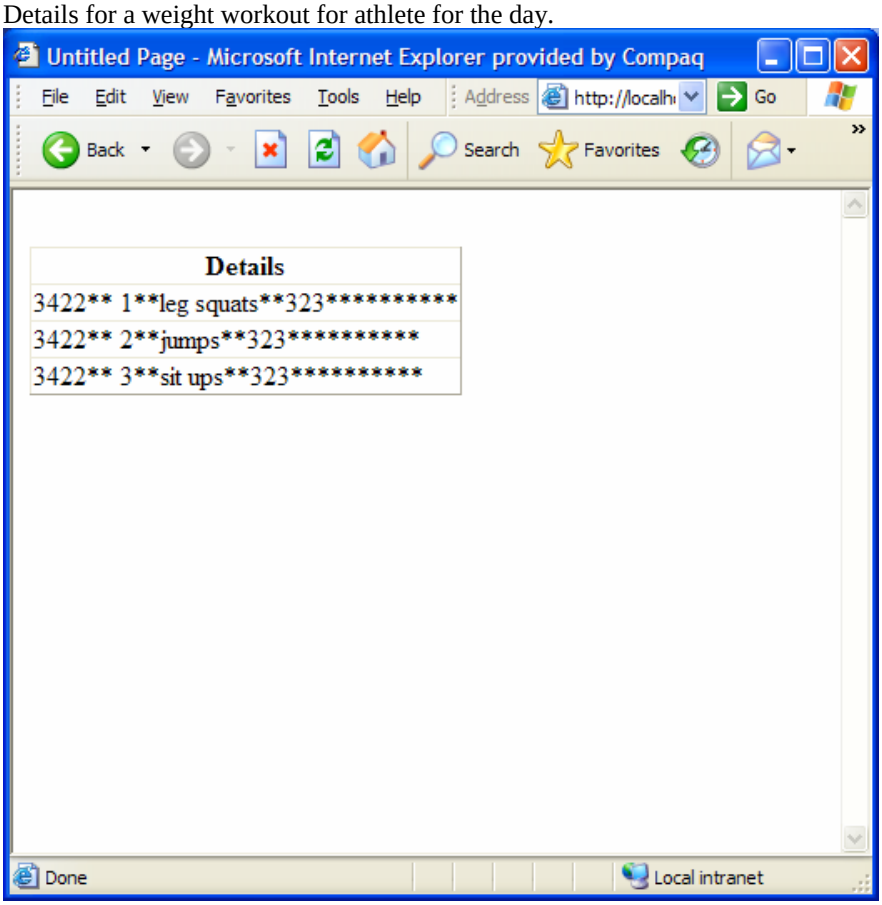
Same as above but for adding Weight



Details for Bike workout. (Needs to be prettied up, but this method does not require database to store specific information in a workout table. All data is extracted from workout tables.

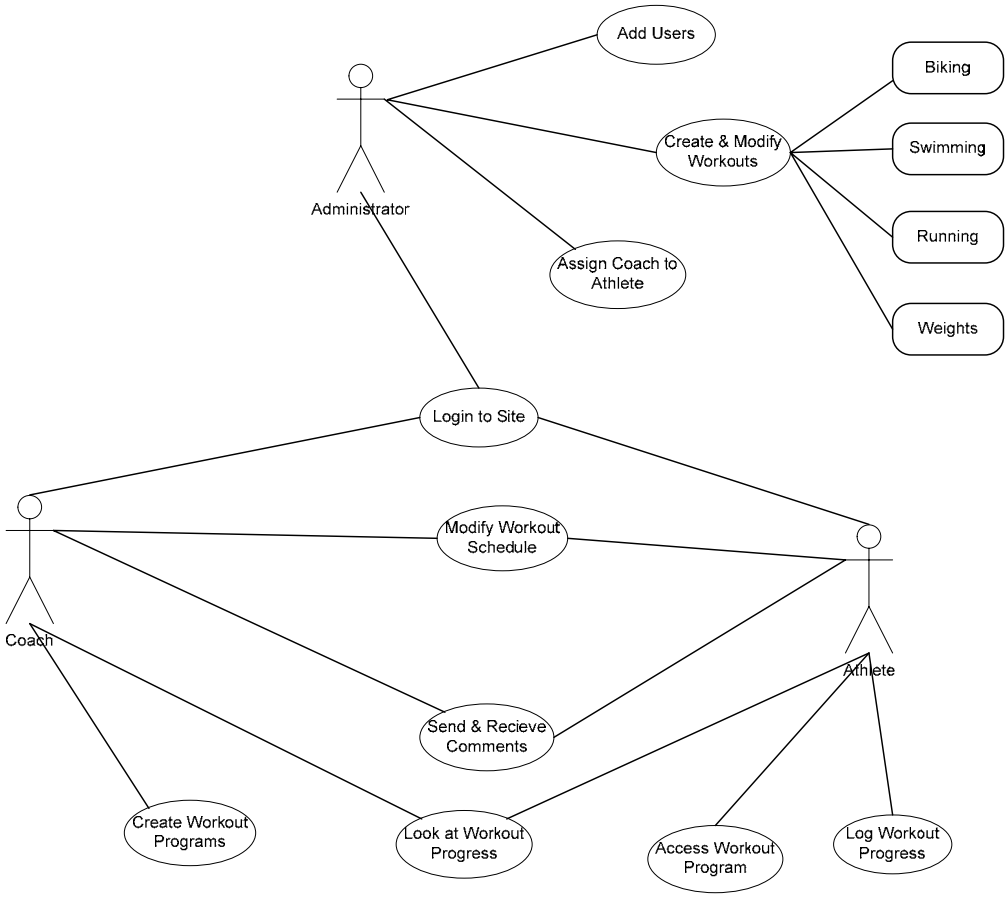


Details for a weight workout for athlete for the day.



Appendix C (Project Charts and Diagrams)

Use Case Diagram

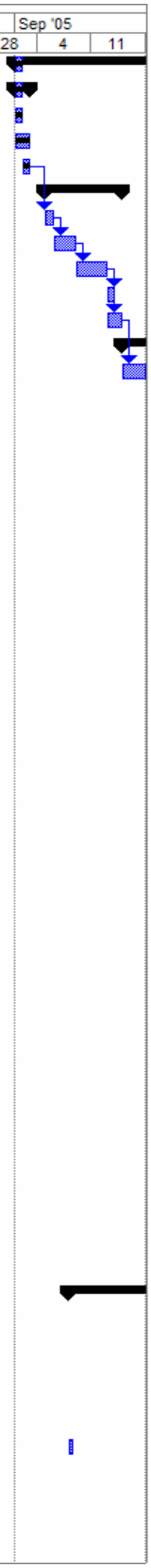


Work Breakdown Structure (WBS)

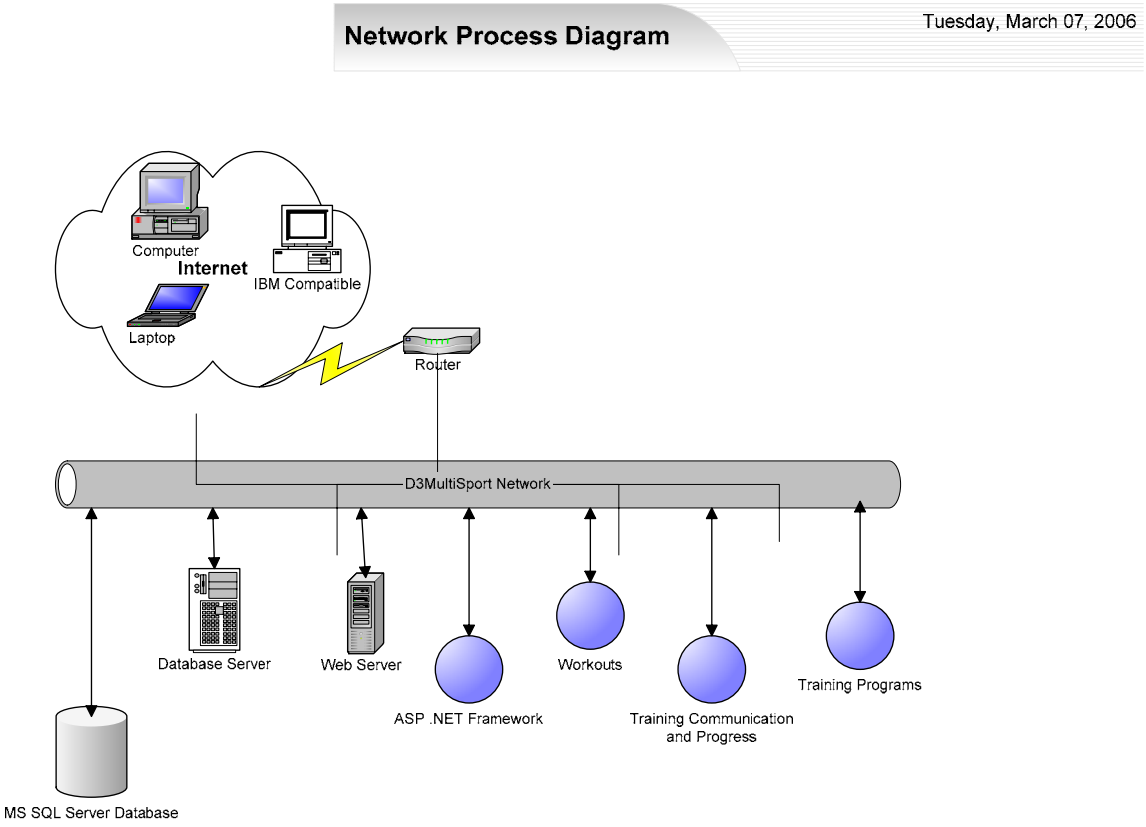
1.0	**D3Multisport Web Site Project**
1.1	Initiating
1.1.1	Assign Roles and responsibilities
1.1.2	Prepare Business Case/ Requirements
1.1.3	Prepare Project Charter
1.2	Planning
1.2.1	Develop a scope statement
1.2.2	Define Goals of Project
1.2.3	Define Barriers and Issues
1.2.4	Develop a work breakdown structure (WBS)
1.2.5	Create Gantt chart
1.2.6	Define Resource Requirements
1.2.7	Create Network diagram
1.3	Executing
1.3.1	Develop Business Requirments
1.3.2	Develop Application Requirements
1.3.3	Develop Functional Requirements
1.3.4	Stakeholder Review of Requirements
1.3.5	Acceptance of Requirements
1.3.6	Stakeholder Review of Specifications
1.3.7	Usability Design and Paper Prototype Testing
1.3.8	Revision of Specification
1.3.9	Acceptance of Specification
1.3.10	Develop System Architecture
1.3.11	Develop Hardware Requirements & Submit Request
1.3.12	Develop Software Architecture
1.3.13	Create Diagrams
1.3.13.1	Data Model
1.3.13.2	Entity Relationship
1.3.13.3	Work Flow
1.3.13.4	Web Site Storyboard
1.3.13.5	Network
1.3.14	Software Development (4 two week Cycles)
1.3.14.1	Software & Unit Test Development
1.3.14.2	Software Integration & Testing
1.3.14.3	Release to Beta Test Team
1.3.14.4	Bug Fixes
1.3.15	Application promotion
1.3.16	Deployment Environment Ready
1.3.17	Application roll-out
1.4	Controlling
1.4.1	Update Gantt Chart
1.4.2	Update scope statement
1.4.3	Status Reports
1.4.4	Progress Reports
1.5	Closing
1.5.1	Prepare final project presentation
1.5.2	Prepare a lessons-learned report for project
1.5.3	Prepare final project report

Gantt Chart

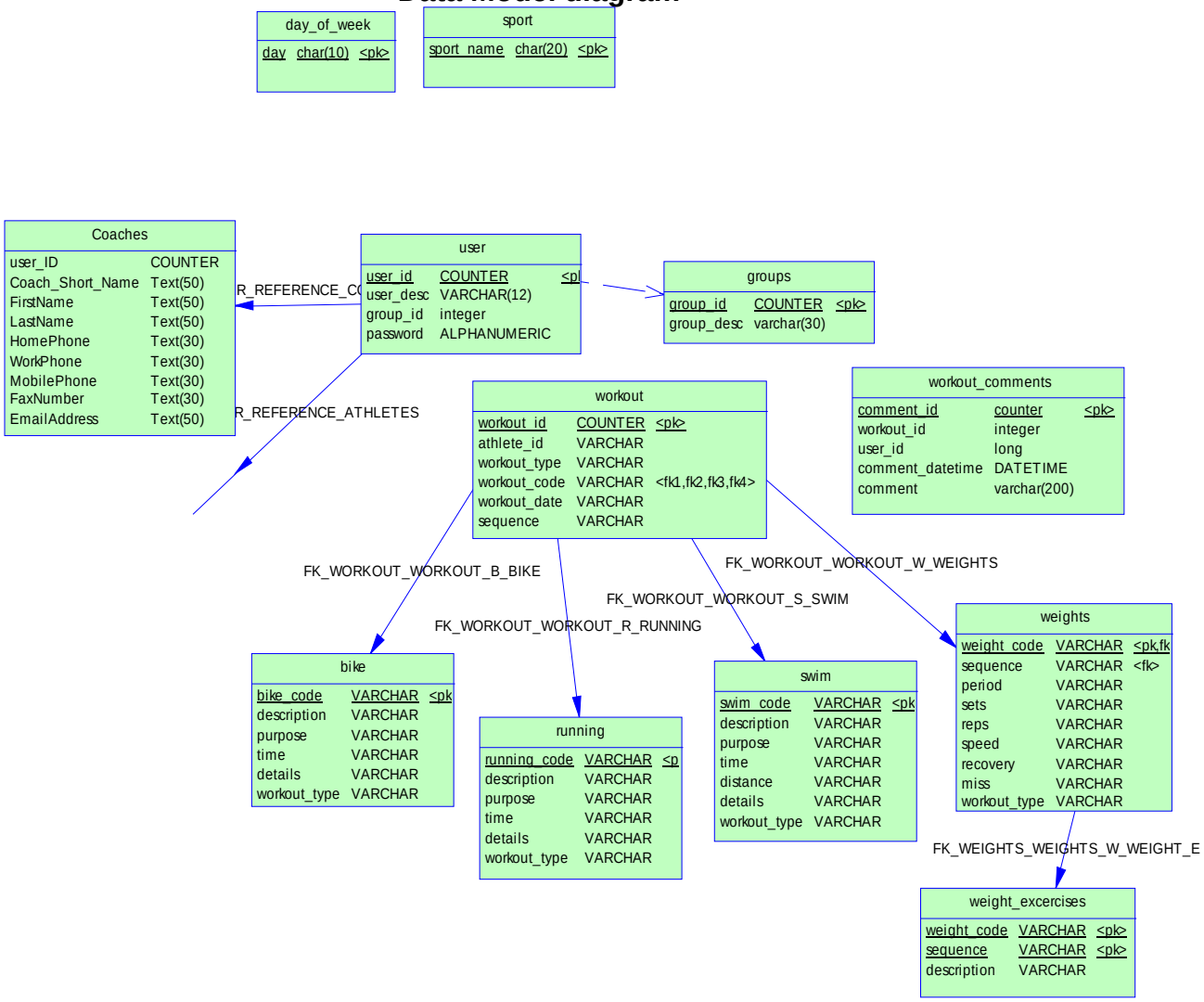
ID	Task Name	Duration	Start	Finish
1	**D3Multisport Internet Project Application**	**125 days**	**Thu 9/1/05**	**Wed 2/22/06**
2	**Initiating**	**2 days**	**Thu 9/1/05**	**Fri 9/2/05**
3	Assign Project Manager	1 day	Thu 9/1/05	Thu 9/1/05
4	Prepare business case	2 days	Thu 9/1/05	Fri 9/2/05
5	Prepare Project Charter	1 day	Fri 9/2/05	Fri 9/2/05
6	**Planning**	**8 days**	**Mon 9/5/05**	**Wed 9/14/05**
7	Develop a scope statement	1 day	Mon 9/5/05	Mon 9/5/05
8	Develop a work breakdown structure (WBS)	3 days	Tue 9/6/05	Thu 9/8/05
9	Create Gantt chart	2 days	Fri 9/9/05	Mon 9/12/05
10	Create Network diagram	1 day	Tue 9/13/05	Tue 9/13/05
11	Create a probability/impact matrix and list of prioritized risks	2 days	Tue 9/13/05	Wed 9/14/05
12	**Executing**	**115 days**	**Thu 9/15/05**	**Wed 2/22/06**
13	Survey Existing Relevant Information	3 days	Thu 9/15/05	Mon 9/19/05
14	Develop Functional Requirements	4 days	Tue 9/20/05	Fri 9/23/05
15	Stakeholder Review of Requirements	1 day	Mon 9/26/05	Mon 9/26/05
16	Acceptance of Requirements	2 days	Tue 9/27/05	Wed 9/28/05
17	Develop Specifications	5 days	Thu 9/29/05	Wed 10/5/05
18	Stakeholder Review of Specifications	2 days	Thu 10/6/05	Fri 10/7/05
19	Design and Develop Prototype for Client to Test	5 days	Mon 10/10/05	Fri 10/14/05
20	Revision of Specification	2 days	Mon 10/17/05	Tue 10/18/05
21	Acceptance of Specification	2 days	Wed 10/19/05	Thu 10/20/05
22	Develop System Architecture	10 days	Fri 10/21/05	Thu 11/3/05
23	Develop Hardware Requirements & Submit Request	10 days	Fri 10/21/05	Thu 11/3/05
24	Develop Software Architecture	10 days	Fri 10/21/05	Thu 11/3/05
25	**Software Development (4 two week Cycles)**	**70 days**	**Fri 11/4/05**	**Thu 2/9/06**
26	**Software and Unit Test Development**	**40 days**	**Fri 11/4/05**	**Thu 12/29/05**
27	Software and Unit Test Development 1	2 wks	Fri 11/4/05	Thu 11/17/05
28	Software and Unit Test Development 2	2 wks	Fri 11/18/05	Thu 12/1/05
29	Software and Unit Test Development 3	2 wks	Fri 12/2/05	Thu 12/15/05
30	Software and Unit Test Development 4	2 wks	Fri 12/16/05	Thu 12/29/05
31	**Software Integration & Testing**	**40 days**	**Fri 11/18/05**	**Thu 1/12/06**
32	Software Integration & Testing 1	2 wks	Fri 11/18/05	Thu 12/1/05
33	Software Integration & Testing 2	2 wks	Fri 12/2/05	Thu 12/15/05
34	Software Integration & Testing 3	2 wks	Fri 12/16/05	Thu 12/29/05
35	Software Integration & Testing 4	2 wks	Fri 12/30/05	Thu 1/12/06
36	**Release to Beta Test Team**	**40 days**	**Fri 12/2/05**	**Thu 1/26/06**
37	Release to Beta Test Team 1	2 wks	Fri 12/2/05	Thu 12/15/05
38	Release to Beta Test Team 2	2 wks	Fri 12/16/05	Thu 12/29/05
39	Release to Beta Test Team 3	2 wks	Fri 12/30/05	Thu 1/12/06
40	Release to Beta Test Team 4	2 wks	Fri 1/13/06	Thu 1/26/06
41	**Bug Fixes**	**40 days**	**Fri 12/16/05**	**Thu 2/9/06**
42	Bug Fixes 1	2 wks	Fri 12/16/05	Thu 12/29/05
43	Bug Fixes 2	2 wks	Fri 12/30/05	Thu 1/12/06
44	Bug Fixes 3	2 wks	Fri 1/13/06	Thu 1/26/06
45	Bug Fixes 4	2 wks	Fri 1/27/06	Thu 2/9/06
46	Application promotion	70 days	Fri 11/4/05	Thu 2/9/06
47	Deployment Environment Ready	5 days	Fri 2/10/06	Thu 2/16/06
48	Application roll-out	4 days	Fri 2/17/06	Wed 2/22/06
49	**Controlling**	**#########**	**Thu 9/8/05**	**Fri 2/10/06**
50	**Update Gantt Chart**	**75.5 days**	**Fri 10/28/05**	**Fri 2/10/06**
60	**Update Project Charter**	**0.5 days**	**Fri 12/16/05**	**Fri 12/16/05**
61	Update Project Charter 1	4 hrs	Fri 12/16/05	Fri 12/16/05
62	**Update Scope Statement**	**11 days**	**Fri 12/2/05**	**Fri 12/16/05**
65	**Status Reports**	**65.5 days**	**Fri 9/30/05**	**Fri 12/30/05**
80	**Progress Reports**	**90.5 days**	**Thu 9/8/05**	**Thu 1/12/06**
88	**Closing**	**5 days**	**Fri 2/10/06**	**Thu 2/16/06**
89	Prepare final project presentation	5 days	Fri 2/10/06	Thu 2/16/06
90	Prepare a lessons-learned report for project	2 days	Fri 2/10/06	Mon 2/13/06
91	Prepare final project report	3 days	Fri 2/10/06	Tue 2/14/06



Network Diagram



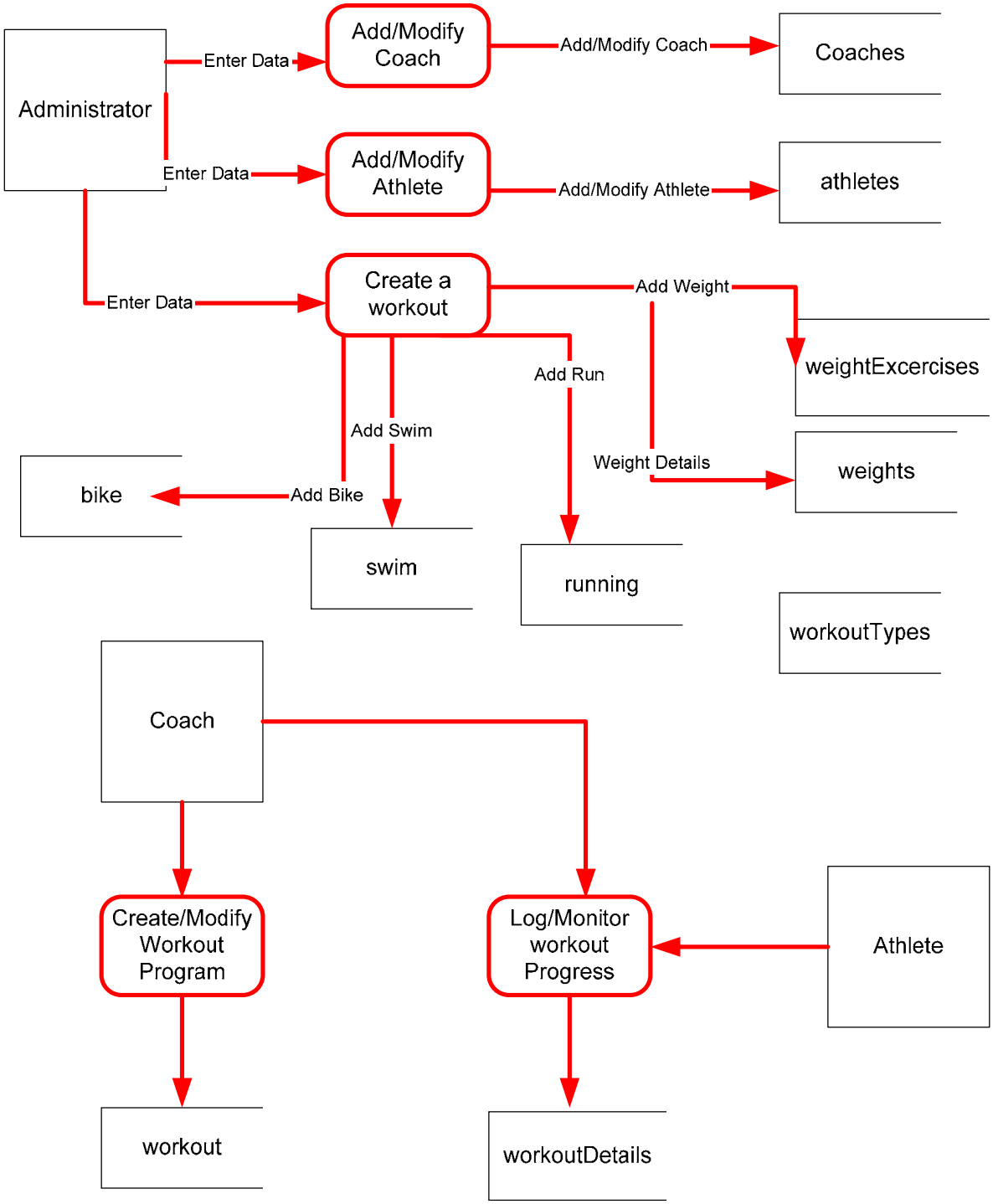
Data Model diagram



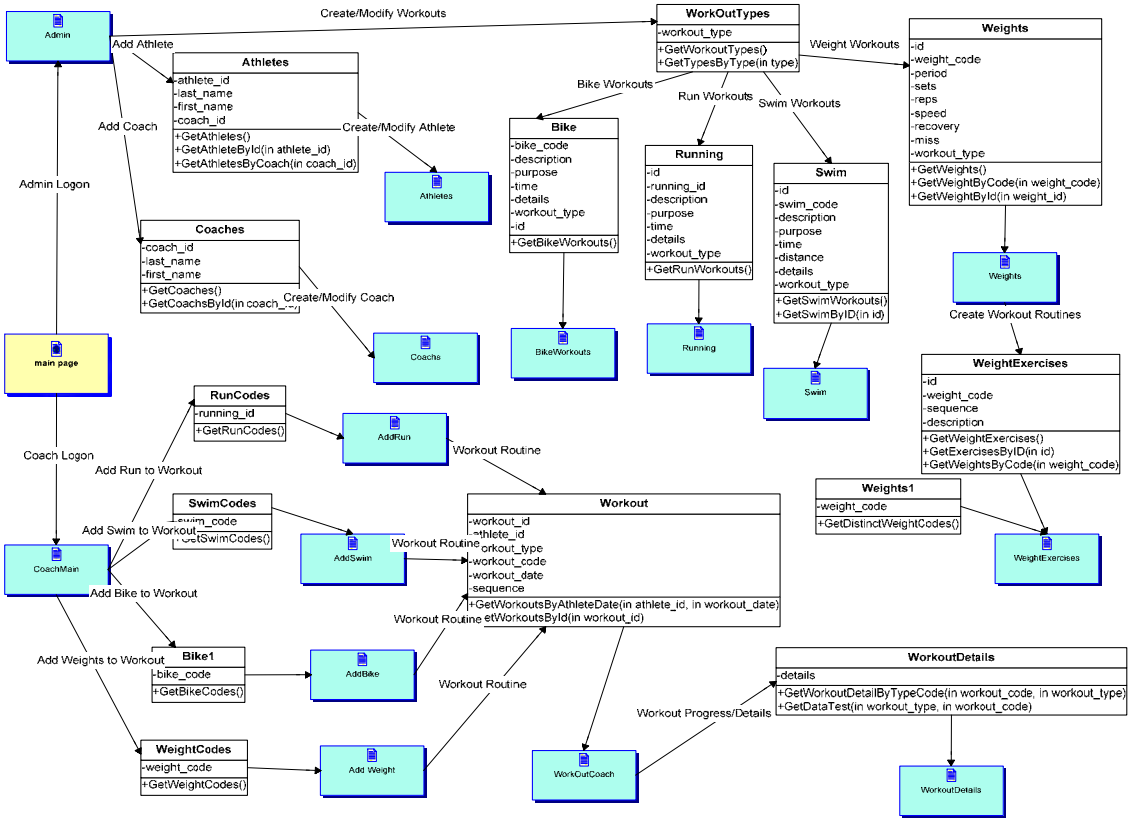
Data Model User Table

Athletes	
user_ID	COUNTER
First_Name	Text(50)
Last_Name	Text(50)
Address	Text(50)
City	Text(50)
State	Text(50)
Zip_Code	Text(50)
Country	Text(50)
Telephone	Text(50)
Bus_Telephone	Text(50)
Mobile_Telephone	Text(50)
Fax	Text(50)
Email	Text(50)
Best_Contact	Text(50)
Birth	Text(50)
Age	Text(50)
Sex	Text(50)
Height	INTEGER
Weight	INTEGER
Marital_Status	Text(50)
Children	YesNo
Chidren_amount	INTEGER
Travel	YesNo
Travel_Amount	INTEGER
Train_Time	Memo
Criteria	Memo
Hear_D3	Memo
Start_Date	DATETIME
Level_Interest	Text(50)
Ironman	YesNo
Ironman_Race	Text(50)
Emphasis	Text(50)
Hour_Per_Week	INTEGER
Rest_Day	Text(50)
Train_Schedule_Mon	Text(50)
Train_Schedule_Tues	Text(50)
Train_Schedule_Weds	Text(50)
Train_Schedule_Thurs	Text(50)
Train_Schedule_Fri	Text(50)
Train_Schedule_Sat	Text(50)
Train_Schedule_Sun	Text(50)
Train_Time_Mon	DATETIME
Train_Time_Tues	DATETIME
Train_Time_Weds	DATETIME
Train_Time_Thurs	DATETIME
Train_Time_Fri	DATETIME
Train_Time_Sat	DATETIME
Train_Time_Sun	DATETIME
Train_Type	Text(50)
Other_Type	Text(50)
Avg_Hour_swim	DATETIME
Avg_Distance_Swim	INTEGER
Avg_Hour_Bike	DATETIME
Avg_Distance_Bike	INTEGER
Avg_Hour_Run	DATETIME
Avg_Distance_Run	INTEGER
Goals	Text(50)
Goal_1_Tri_Du	YesNo
Goal_1_Olympic	YesNo
Goal_1_Halftroman	YesNo
Goal_1_Ironman	YesNo
Goal_Comp_Age	YesNo
Goal_IM_Slot	YesNo
Favorite_Distance	Text(50)
Best_Race	Memo
Best_Race_Distance	Text(50)
Best_Race_Time_Swim	DATETIME
Best_Race_Time_Bike	DATETIME
Best_Race_Time_Run	DATETIME
Define_Season	Memo
Swim_History	Text(50)
Swim_Masters	YesNo
Swim_Measurement	Text(50)
Swim_Best_Distance	Text(50)
Swim_Best_Time_800	DATETIME
Swim_Best_Time_1500	DATETIME
Swim_Best_Time_12	DATETIME
Swim_Best_Time24	DATETIME
Swim_Pace	DATETIME
Swim_Pull	YesNo
Swim_Fins	YesNo
Swim_Paddles	YesNo
Bike_History	Text(50)
Bike_Avg_Mph	INTEGER
Bike_Race_HR	INTEGER
Bike_Train_HR	INTEGER
Bike_Train_Avg_Mph	INTEGER
Bike_Best_Time_Distance	Text(50)
Bike_Best_Time_20	DATETIME
Bike_Best_Time_40	DATETIME
Bike_Best_Time_56	DATETIME
Bike_Best_Time_112	DATETIME
Bike_Train_Partner	YesNo
Bike_Trainer	YesNo
Bike_Cadence_Feature	YesNo
Own_Compu_Trainer	YesNo
Want_Information	YesNo
Run_History	Text(50)
Run_Pace	DATETIME
Run_track	YesNo
Run_Partner	YesNo
Run_Best_Distance	Text(50)
Run_Time_5k	DATETIME
Run_Time_10k	DATETIME
Run_Time_Half	DATETIME
Run_Time_Full	DATETIME
Nutrition_Information	YesNo
Nutrition_Zeroed	YesNo
Nut_Pre_race	Memo
Nut_Race_Day	Memo
Nut_Post_Race	Memo
Nut_Typical_Day_Break	Text(50)
Nut_Typical_Day_Lunch	Text(50)
Nut_Typical_Day_Dinner	Text(50)
Nut_Typical_Day_Snack	Text(50)
Strength_Train	YesNo
HR_Monitor	YesNo
HR_Use_It	YesNo
Max_HR_Recorded	INTEGER
Lactate_Threshold	YesNo
Vo2Max	YesNo
Sleep_Average	INTEGER
Strong_Points	Memo
Course_Best	Text(50)
Limiter_1	Text(50)
Limiter_2	Text(50)
Limiter_3	Text(50)
Limiter_4	Text(50)
Limit_Other	Text(50)
Eighteen_Age	YesNo
Parental_Agreement	YesNo
Signed_Waiver	YesNo
Coach_ID	INTEGER

Data Flow Diagram



Object Model Diagram



Appendix D (Code Examples)

This is the code behind the “Add Run” ASP Page which is displayed in Appendix B. Most of this code was generated by merely dragging and dropping objects onto the ASP page in design view, then manually configured to communicate with the desired business object.

ADD RUN ASP Code:

```
<%@ Page Language="VB" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<script runat="server">

</script>

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Add Run Workout</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <strong><span style="text-decoration: underline">Select An
Athlete and Date</span></strong><br />
        <asp:DropDownList ID="DropDownList1" runat="server"
DataSourceID="ObjectDataSource1"
            DataTextField="last_first" DataValueField="athlete_id">
        </asp:DropDownList><asp:ObjectDataSource ID="ObjectDataSource1"
runat="server" SelectMethod="GetAthleteByAthlete"
            TypeName="coaches_dsTableAdapters.athletes1TableAdapter">
            <SelectParameters>
                <asp:QueryStringParameter Name="athlete_id"
QueryStringField="athlete_id" Type="Int32" />
            </SelectParameters>
        </asp:ObjectDataSource>
        <br />
        <asp:Calendar ID="Calendar1" runat="server" Height="151px"
SelectedDate="2005-11-05"
            Width="253px"></asp:Calendar>

    </div>
        <br />
        <strong><span style="text-decoration: underline">Current
Athlete Workout Schedule for
            the Day</span></strong><br />
        <asp:GridView ID="GridView1" runat="server" AllowPaging="True"
AllowSorting="True"
            AutoGenerateColumns="False" DataKeyNames="workout_id"
DataSourceID="ObjectDataSource2">
            <Columns>
                <asp:CommandField ShowDeleteButton="True" />
```

```
                <asp:BoundField DataField="workout_id"
HeaderText="workout_id" InsertVisible="False"
                    ReadOnly="True" SortExpression="workout_id" />
                <asp:BoundField DataField="athlete_id"
HeaderText="athlete_id" SortExpression="athlete_id" />
                <asp:BoundField DataField="workout_type"
HeaderText="workout_type" SortExpression="workout_type" />
                <asp:BoundField DataField="workout_code"
HeaderText="workout_code" SortExpression="workout_code" />
                <asp:BoundField DataField="workout_date"
HeaderText="workout_date" SortExpression="workout_date" />
                <asp:BoundField DataField="sequence"
HeaderText="sequence" SortExpression="sequence" />
                <asp:HyperLinkField
DataNavigateUrlFields="workout_type,workout_code"
DataNavigateUrlFormatString="WorkoutDetails.aspx?workout_type={0}&amp;w
orkout_code={1}"
                    DataTextField="workout_id" HeaderText="Details"
Text="Details" />
            </Columns>
        </asp:GridView>
        <asp:ObjectDataSource ID="ObjectDataSource2" runat="server"
DeleteMethod="Delete"
            InsertMethod="Insert"
SelectMethod="GetWorkoutsByAthleteDate"
TypeName="coaches_dsTableAdapters.workoutTableAdapter"
            UpdateMethod="Update">
            <DeleteParameters>
                <asp:Parameter Name="Original_workout_id" Type="Int64"
/>
            </DeleteParameters>
            <UpdateParameters>
                <asp:Parameter Name="athlete_id" Type="Int32" />
                <asp:Parameter Name="workout_type" Type="String" />
                <asp:Parameter Name="workout_code" Type="String" />
                <asp:Parameter Name="workout_date" Type="DateTime" />
                <asp:Parameter Name="sequence" Type="Int32" />
                <asp:Parameter Name="Original_workout_id" Type="Int64"
/>
            </UpdateParameters>
            <SelectParameters>
                <asp:ControlParameter ControlID="DropDownList1"
Name="athlete_id" PropertyName="SelectedValue"
                    Type="Int32" />
                <asp:ControlParameter ControlID="Calendar1"
Name="workout_date" PropertyName="SelectedDate"
                    Type="DateTime" />
            </SelectParameters>
            <InsertParameters>
                <asp:Parameter Name="athlete_id" Type="Int32" />
                <asp:Parameter Name="workout_type" Type="String" />
                <asp:Parameter Name="workout_code" Type="String" />
                <asp:Parameter Name="workout_date" Type="DateTime" />
                <asp:Parameter Name="sequence" Type="Int32" />
            </InsertParameters>
        </asp:ObjectDataSource>
        <br />
```

```
        <span style="text-decoration: underline"><strong>Add Run
Workout</strong></span><br />
        <asp:DetailsView ID="DetailsView1" runat="server"
AutoGenerateRows="False" DataKeyNames="workout_id"
            DataSourceID="ObjectDataSource2" DefaultMode="Insert"
Height="50px" Width="125px">
            <Fields>
                <asp:BoundField DataField="workout_id"
HeaderText="workout_id" InsertVisible="False"
                    ReadOnly="True" SortExpression="workout_id" />
                <asp:TemplateField HeaderText="athlete_id"
SortExpression="athlete_id">
                    <ItemTemplate>
                        <asp:Label ID="Label1" runat="server" Text='<%#
Bind("athlete_id") %>'></asp:Label>
                    </ItemTemplate>
                    <EditItemTemplate>
                        <asp:TextBox ID="TextBox1" runat="server"
Text='<%# Bind("athlete_id") %>'></asp:TextBox>
                    </EditItemTemplate>
                    <InsertItemTemplate>
                        <asp:DropDownList ID="DropDownList2"
runat="server" DataSourceID="ObjectDataSourceID"
                            DataTextField="last_first"
DataValueField="athlete_id" SelectedValue='<%# Bind("athlete_id") %>'
Enabled="False">
                        </asp:DropDownList><asp:ObjectDataSource
ID="ObjectDataSourceID" runat="server"
SelectMethod="GetAthleteByAthlete"

TypeName="coaches_dsTableAdapters.athletes1TableAdapter">
                            <SelectParameters>
                                <asp:ControlParameter
ControlID="DropDownList1" Name="athlete_id"
PropertyName="SelectedValue"
                                    Type="Int32" />
                            </SelectParameters>
                        </asp:ObjectDataSource>
                    </InsertItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="workout_type"
SortExpression="workout_type">
                    <ItemTemplate>
                        <asp:Label ID="Label2" runat="server" Text='<%#
Bind("workout_type") %>'></asp:Label>
                    </ItemTemplate>
                    <EditItemTemplate>
                        <asp:TextBox ID="TextBox2" runat="server"
Text='<%# Bind("workout_type") %>'></asp:TextBox>
                    </EditItemTemplate>
                    <InsertItemTemplate>
                        <asp:DropDownList ID="DropDownList3"
runat="server" DataSourceID="ObjectDataSourcetype"
                            DataTextField="workout_type"
DataValueField="workout_type" SelectedValue='<%# Bind("workout_type")
%>' Enabled="False">
```

```
                        </asp:DropDownList><asp:ObjectDataSource
ID="ObjectDataSourcetype" runat="server"
                            InsertMethod="Insert"
SelectMethod="GetTypesByType"
TypeName="coaches_dsTableAdapters.workout_typesTableAdapter">
                            <SelectParameters>
                                <asp:Parameter DefaultValue="run"
Name="type" Type="String" />
                            </SelectParameters>
                            <InsertParameters>
                                <asp:Parameter Name="workout_type"
Type="String" />
                            </InsertParameters>
                        </asp:ObjectDataSource>
                    </InsertItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="workout_date"
SortExpression="workout_date">
                    <ItemTemplate>
                        <asp:Label ID="Label4" runat="server" Text='<%#
Bind("workout_date") %>'></asp:Label>
                    </ItemTemplate>
                    <EditItemTemplate>
                        <asp:TextBox ID="TextBox4" runat="server"
Text='<%# Bind("workout_date") %>'></asp:TextBox>
                    </EditItemTemplate>
                    <InsertItemTemplate>
                        <asp:DropDownList ID="DropDownList4"
runat="server" DataSourceID="SqlDataSourcedate"
                            DataTextField="Column1"
DataValueField="Column1" SelectedValue='<%# Bind("workout_date") %>'
Enabled="False">
                        </asp:DropDownList><asp:SqlDataSource
ID="SqlDataSourcedate" runat="server" ConnectionString="<%$
ConnectionStrings:jaytestConnectionString1 %>"
                            SelectCommand="getDateSent"
SelectCommandType="StoredProcedure">
                            <SelectParameters>
                                <asp:ControlParameter
ControlID="Calendar1" Name="thisdate" PropertyName="SelectedDate"
                                    Type="DateTime" />
                            </SelectParameters>
                        </asp:SqlDataSource>
                    </InsertItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="workout_code"
SortExpression="workout_code">
                    <ItemTemplate>
                        <asp:Label ID="Label3" runat="server" Text='<%#
Bind("workout_code") %>'></asp:Label>
                    </ItemTemplate>
                    <EditItemTemplate>
                        <asp:TextBox ID="TextBox3" runat="server"
Text='<%# Bind("workout_code") %>'></asp:TextBox>
                    </EditItemTemplate>
                    <InsertItemTemplate>
```

```
                        <asp:DropDownList ID="DropDownList5"
runat="server" AutoPostBack="True" DataSourceID="ObjectDataSourcecode"
                            DataTextField="running_id"
DataValueField="running_id" SelectedValue='<%# Bind("workout_code")
%>'>
                        </asp:DropDownList><asp:ObjectDataSource
ID="ObjectDataSourcecode" runat="server"
                            SelectMethod="GetRunCodes"
TypeName="coaches_dsTableAdapters.runCodesTableAdapter">
                        </asp:ObjectDataSource>
                    </InsertItemTemplate>
                </asp:TemplateField>
                <asp:BoundField DataField="sequence"
HeaderText="sequence" SortExpression="sequence" />
                <asp:CommandField ShowInsertButton="True" />
            </Fields>
        </asp:DetailsView>
        <br />
        <asp:HyperLink ID="HyperLink1" runat="server"
NavigateUrl="~/coach_main.aspx">Back to Coach Main</asp:HyperLink>
    </form>
</body>
</html>
```