

Mark Stevens  July 19, 2020

A Journey through Java, MySQL, Apache, and the MVC Architecture with Eclipse IDE

A Synopsis of learning how these development TOOLS WORK together

**The Beginning**

**07/19/20**

This is my memoirs of developing my full stack web development skills. This journey actually started back in 1986 when I saw my first computer up close and personal. It was during a tour of Pittsburgh Technical Institute before registering for classes in Specialized TechnologyArchitectural Design/Autocad. I was amazed by the huge dual screens, digi-pad(as they were called back in the day) and the whole workstation. Back then you had a table with the pad on it, a mouse, and a menu underneath which needed calibrated from time to time. The leading software packages at that time were called MicroStation and Autocad. I don’t recall the versions, 1.2 or something. I knew right away what my calling was; hooked for life on computers!

When first starting out in my calling, a major engineering firm hired me on. Through the years I dabbled in many different computer technology’s in my free time, including building computers, reformatting hard drives from scratch, building websites with Macromedia products before Adobe bought them, databases, networking, servers, domains, etc,, etc. During that time my interest in learning how everything worked together on the internet(and now in the cloud) became my next passion. When an opportunity arose to go back to college and further my studies I jumped on it! The curriculum path I chose was Associates in Cyber Security. It was during those classes that my interest in databases, Java, Servers, SQL and all things in between peaked again. So I find myself really diving into it again instead of just a hobby, you might say.

I finally have my skills honed enough to attempt a full development cycle of Design, Develop, & Deploy an application to the Cloud. It hasn’t been easy, many frustrating days and nights of learning to pull everything together through online tutorials, books, and classes through the Udemy website among others. Reformatting my laptop several times, getting Maven and Spring installed correctly, where and when to use dependencies, and how to find them. Finally was able to figure out how to properly install

Maven also. Most of the tutorials on the internet only get you so close, you need a descent understanding of how everything comes together between Eclipse, MySql, Apache, Maven, and Spring. Going through all that gave me a good foundation to continue my studies. I found the easiest way to install Maven and Spring is to import the folders into an Eclipse project and run the mvn command files from there. Some guys prefer to use the console window pane in Eclipse and navigate to the directory where the files are and run it that way. To check if everything’s installed correctly just right click on a project and go to the “Run As” pane. You will see a bunch of maven options to choose from. I recommend playing around with those in the console window to get a feel for what it’s doing. Also, when you install Apache, sometimes it sets the HTML/1.1 port to 8080 instead of just 80. I found that out the hard way when I was getting “*Server could not start,* *another application is using the port”* Error. That drove me nuts for a few days then I opened up the server.xml file in Eclipse, and changed the port back to 80 and un-commented out the thread pool settings for future use.

# Bibliography

Gaddis, T. (2016). *Starting Out With Java: From Control Structures Through Objects 6th Edition.* Pearson Education.

Kulkarni, R. (2018). *Java EE 8 Development with Eclipse 3rd Edition.* Packt Publishing.

**Ramash Fadatore**

Java Guides (2020)

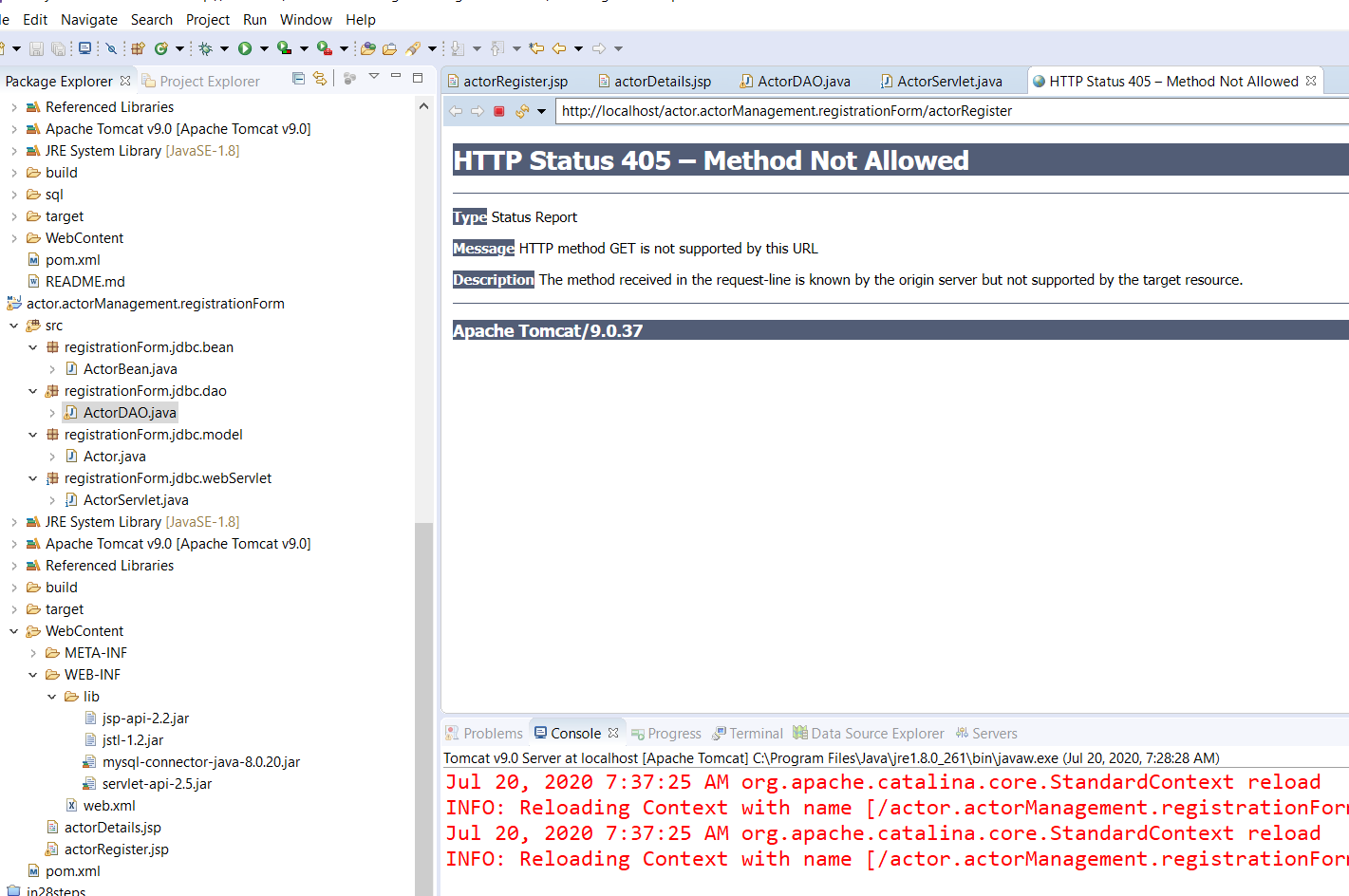


Figure : Error Registering New Actor

I suspect there isn’t a GET method in the actorRegister file, or it’s pointing to the wrong method in a file. The plan is to study Ramesh Fadatare’s Tutorial over at JavaGuides.net. He has an excellent one called *“Registration Form using JSP + Servlet + JDBC + Mysql Example”.* I have done this one already, so I’ll go over it again and figure out why my file is not posting.

**07/20/20 4:00PM** – Well I spent most of the day messing around with it. So close!! I’m getting a file not found error, something’s wrong with a path somewhere! LOL Taking break for awhile and regroup, in the mean time I really need to study this file closely, then modify it to suit my project needs:

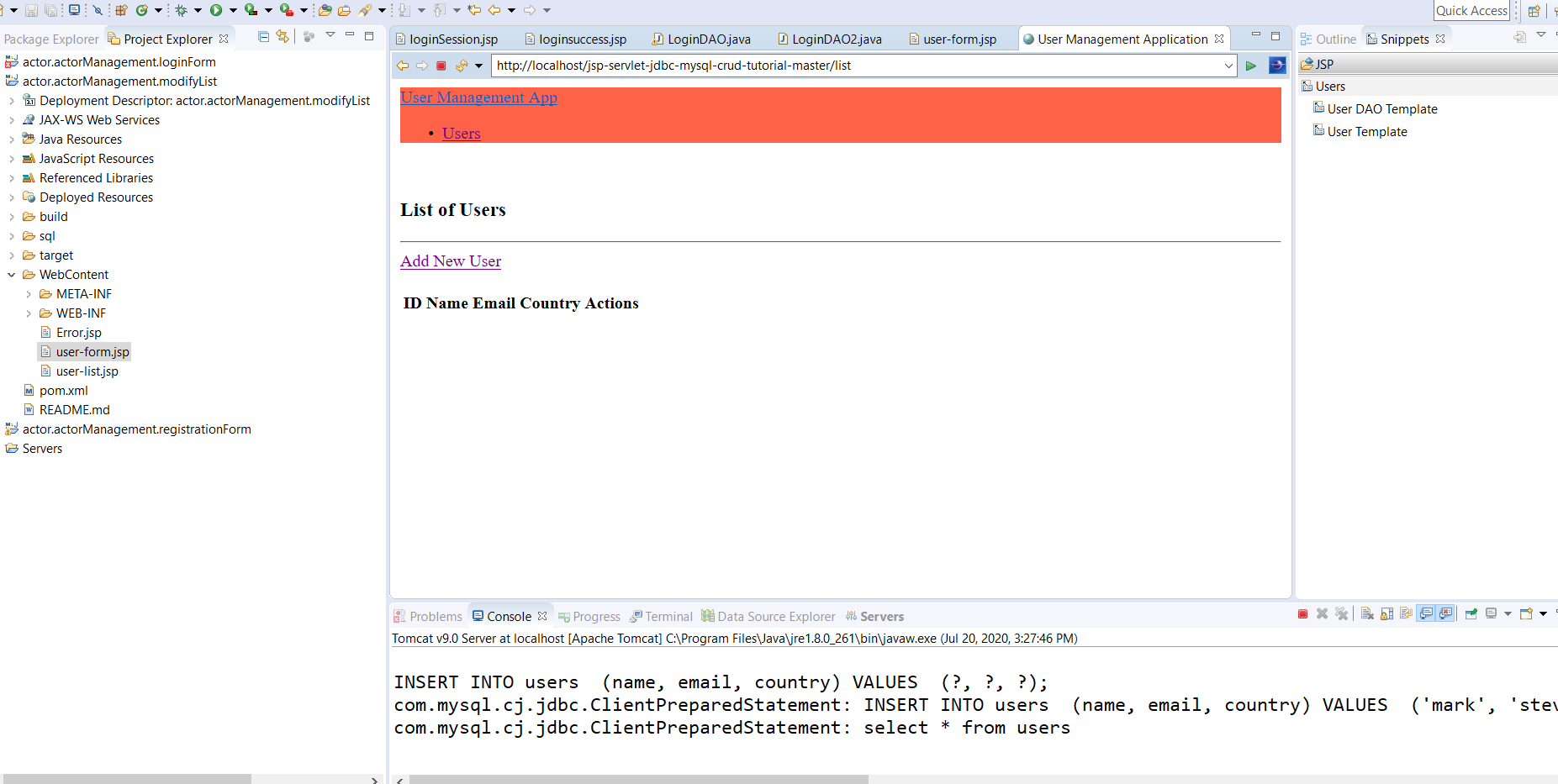


Figure : A working Insert file.

I’m not liking how the address bar path is pointing to a different project. Oh well I’ll figure it out! This could be my new starting base for the project.

**07/21/20 – 10:00AM** I decided to go another route. I was having a hard time getting that setup to work correctly, so I deleted everything in the project folder and started over. The other tutorial I’m using for the User Management files has the hibernate jar files referenced in it, along with a persistence file. So I’m going to try it again along with the Java EE 8 Development book, so my current directories are like this:

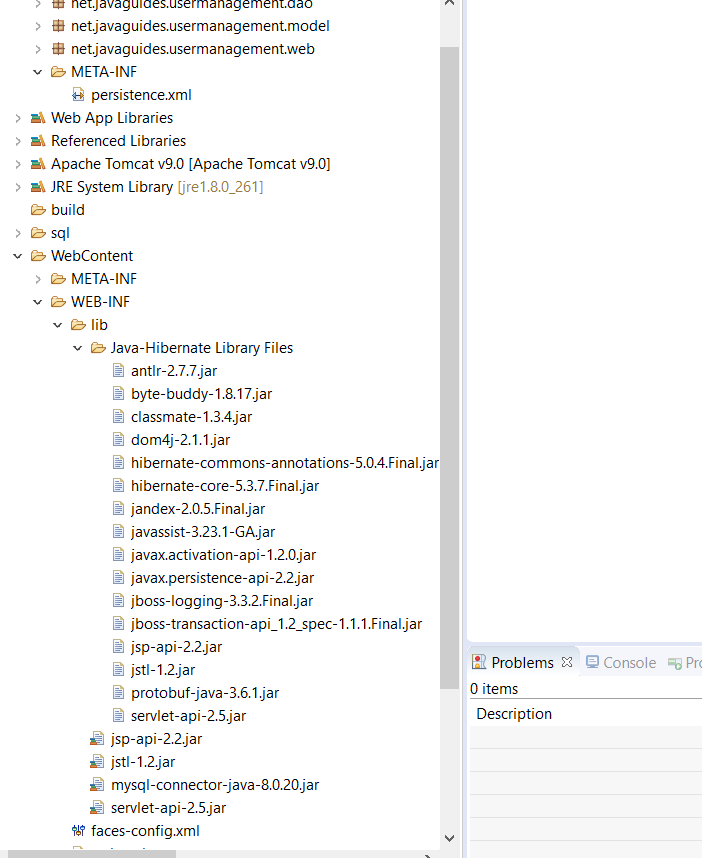


Figure : User Management Project directory structure

Notice the persistence.xml file, we’ll see how this goes….I added the jar file in the xml documents:

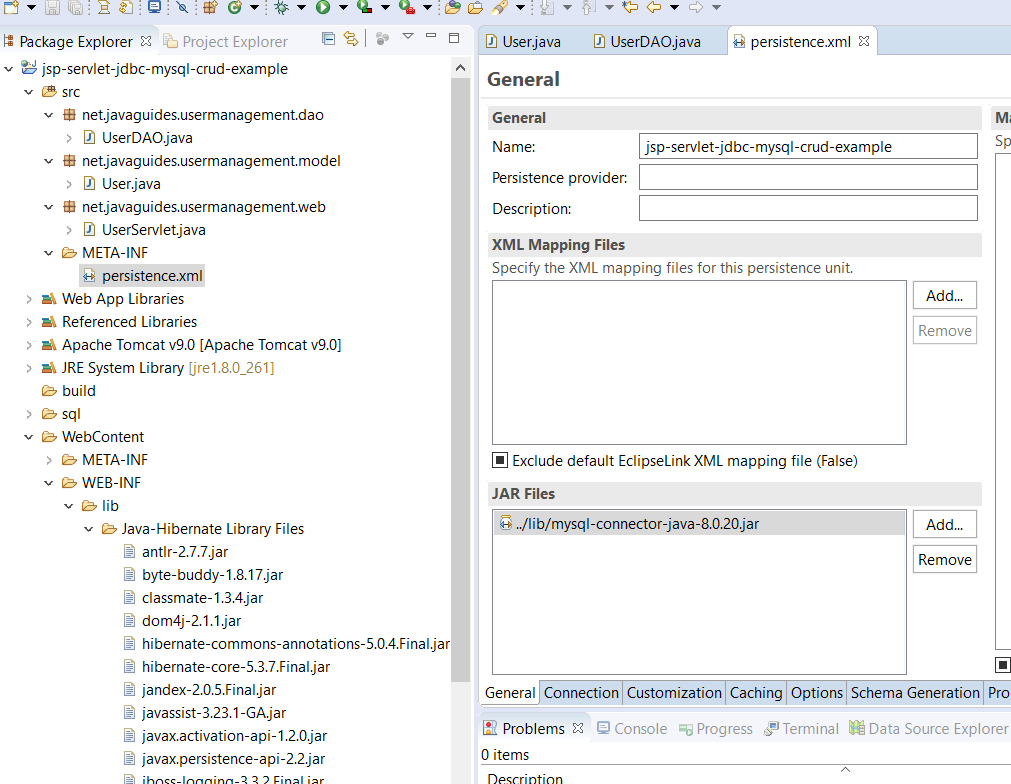


Figure : 8.0.20 Jar file added into persistence.xml file

The ping to the demo DB is successful, also I used the default JTA option and JDBC batch writing, with native SQL box checked. I’m getting a jar file error saying it can’t find it when it’s right there:

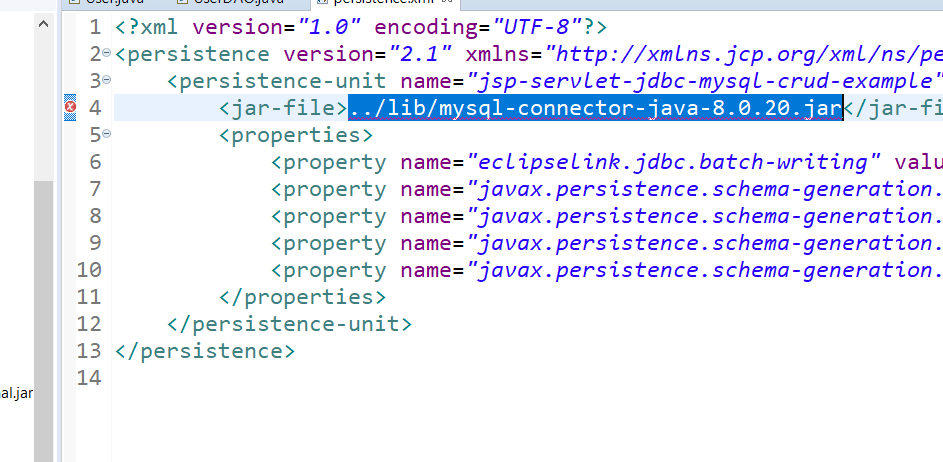


Figure : Jar file error

I went ahead and removed the jar file, and it still pings, so,,,maybe because it’s already mapped in the reference library and WEB-INF library?? Hmm… Oh well we’ll find out soon enough! The web xml file looks interesting:

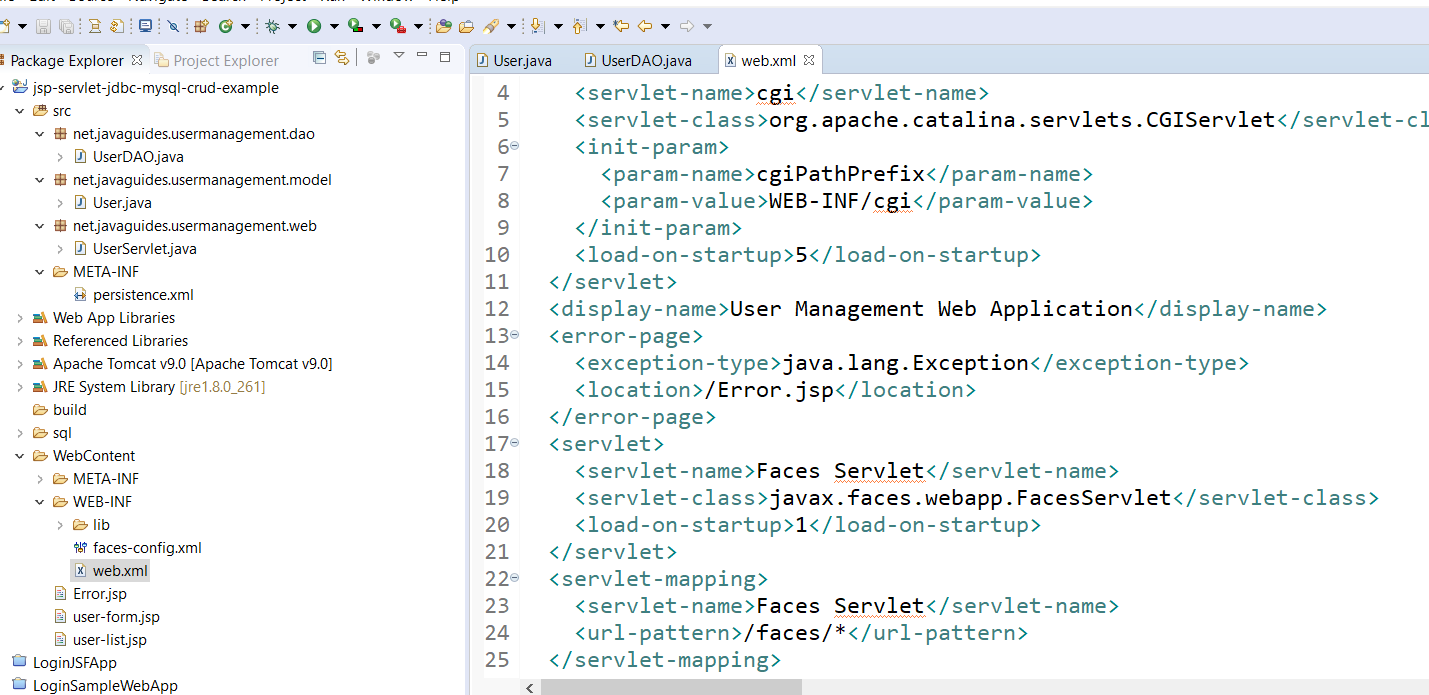


Figure : Web Xml file configuration

I’m not sure why he uses a book reference for a new user here:





Figure : Book new user reference

Right click on the project in Package Explorer🡪 Run As 🡪Run on Server:

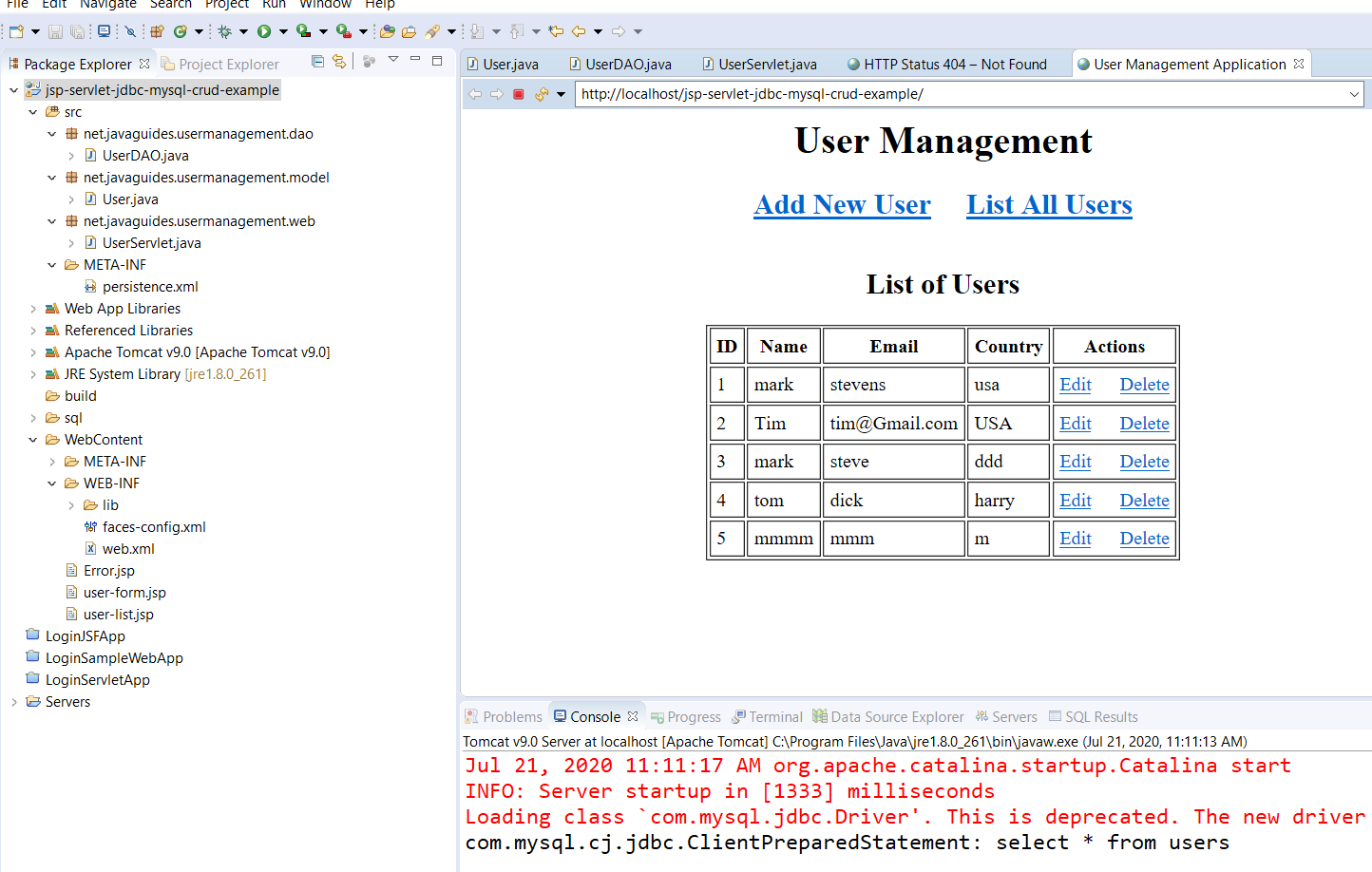


Figure : Runs correctly on server

I should change the Driver class to reflect the com.mysql.cj.jdbc reference! Here’s another little thing:

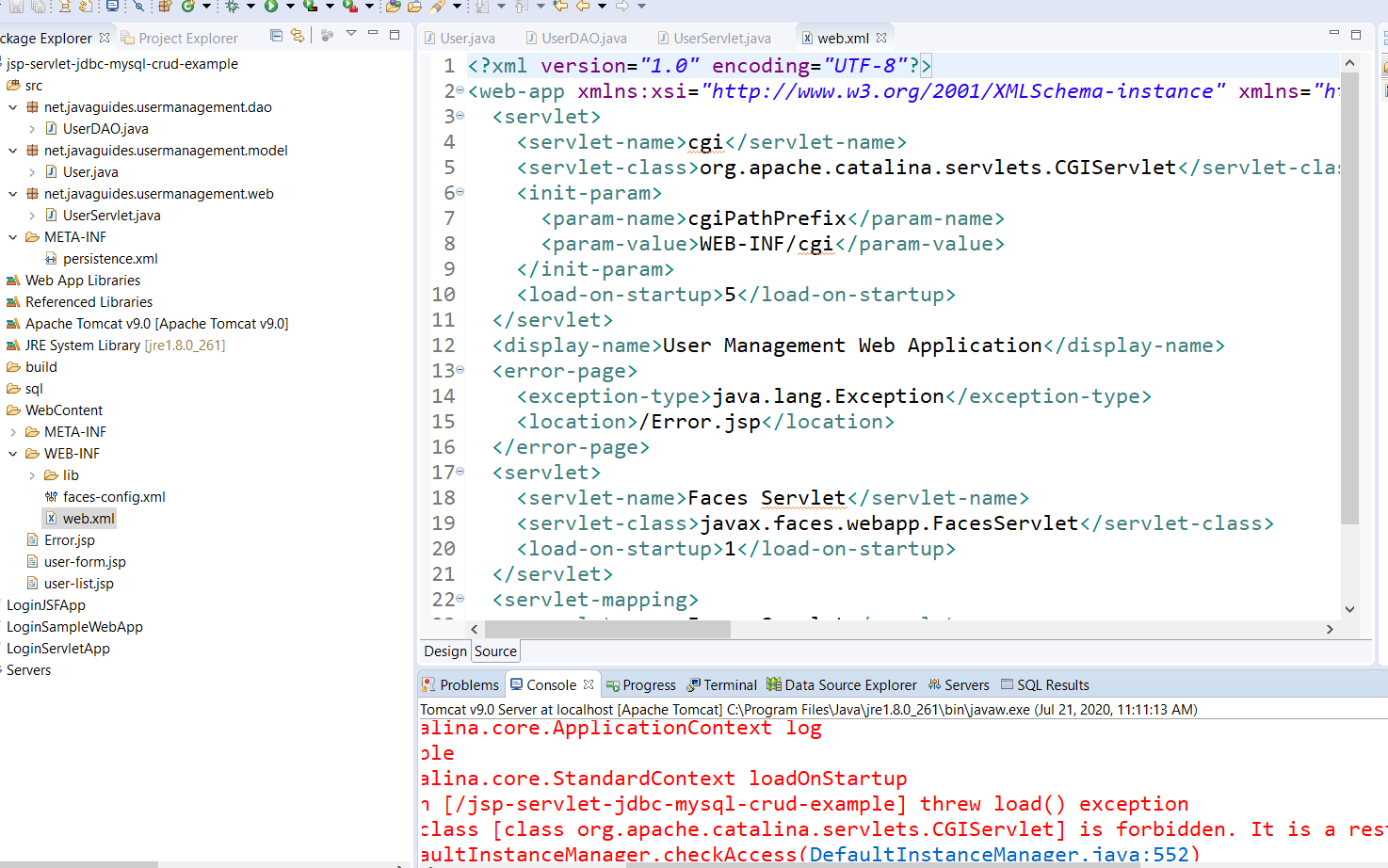




Figure : CGI forbidden here error

Mapping the User class as the managed Class in the persistence unit:

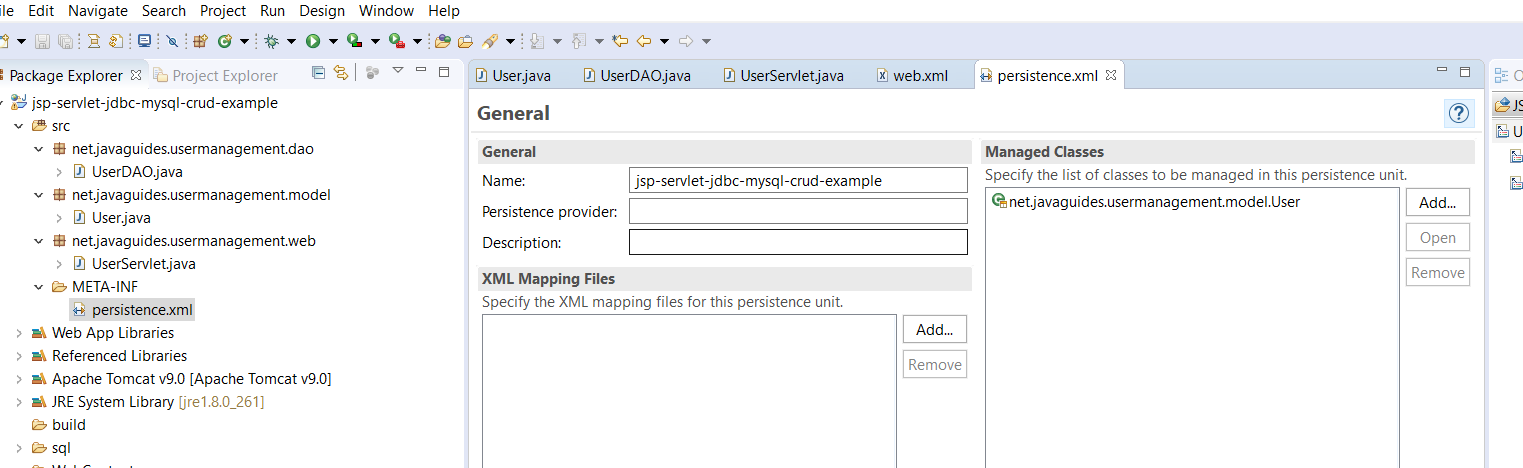


Figure :Adding User class in the persistence unit as the managed class

Open the persistence xml file and add User.

No arg-constructor is required so Hibernate can create the instance of the Persistance class by *newInstance()* method like this: *public user() { }* Initializes the method.

**A helper class should be created to help Bootstrap Hibernate. Should only be used once during initialization to be used by all code.**

**See HibernateUtil class.**

The UserServlet file is the page controller for the application, handling all requests.

This was a perfect project to go through, it has a lot of techniques to use as a foundation. You can find this and others at javaguides.net.

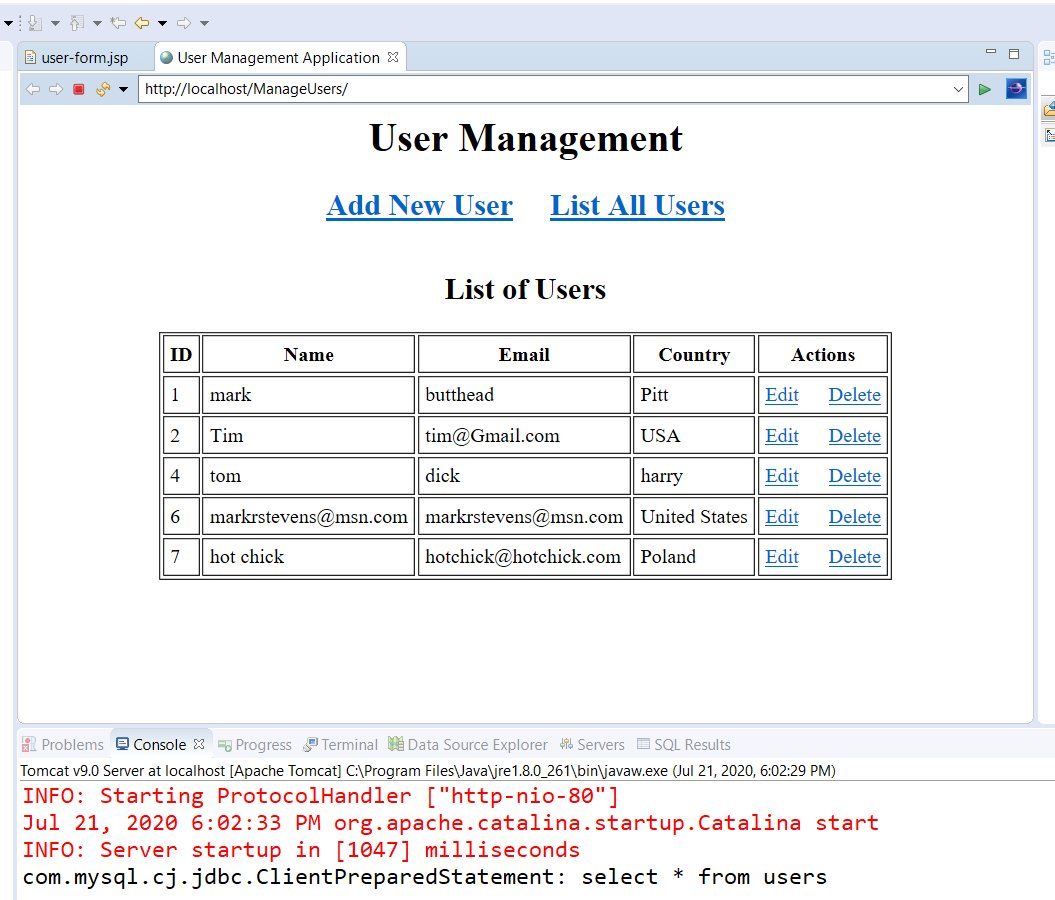


Figure : The finished project

Everything works as intended, Edit/Delete/Add/List from the UserServlet file. I love how it prints out the SQL statement in the console, good for troubleshooting. Before I go back to the Java EE 8 with Eclipse book and take another look at the Course Management Project, I’m doing one more tutorial, a 3-part To-Do list. It should go a lot smoother now that I have a pretty good grip on the process. I’m sure I’ll be revisiting the UserManagement files again to see how he did that! 😊

**Part I To Do List**

07/22/20 8:00AM – The to-do list project is in three parts, the Model, Data Access Object, and Controller. First part is done with no problems:

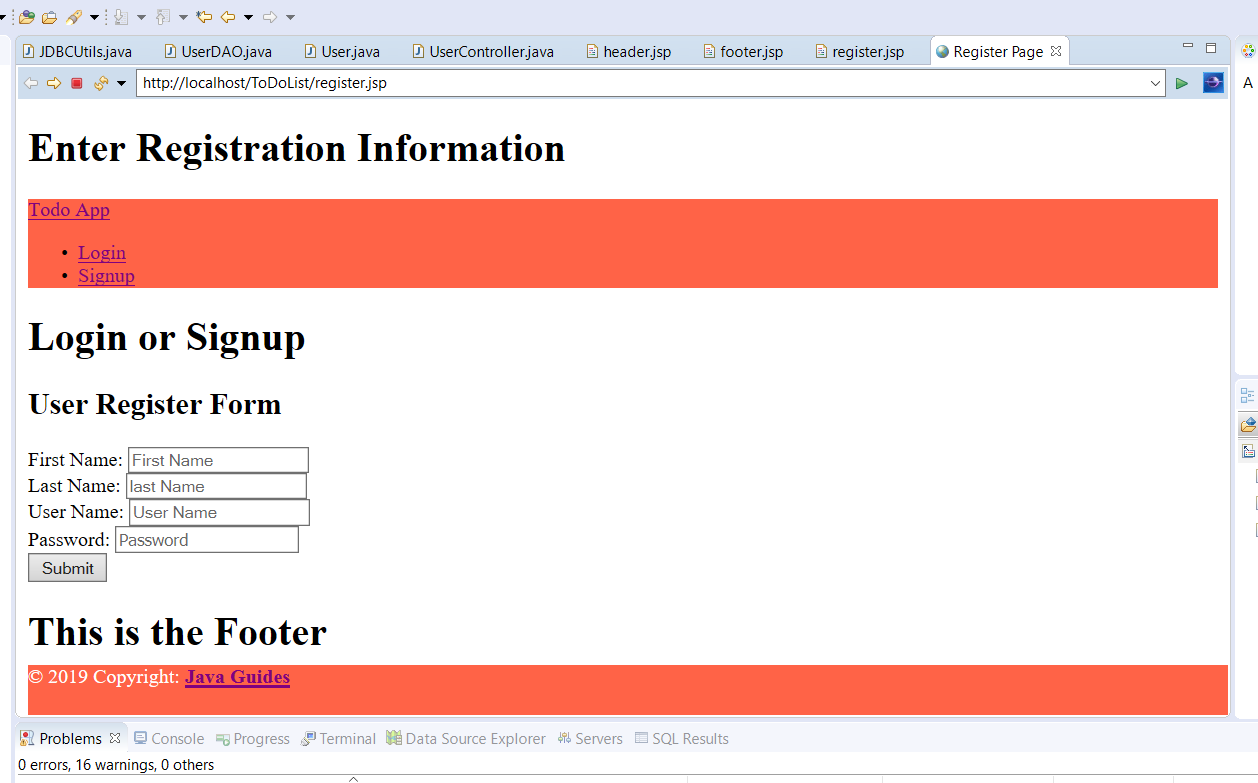


Figure :Phase 1 ToDo Tutorial

One thing about this tutorial, the file names don’t match. For example it calls for a LoginBean file, when mine is LogInDao. The references to the files in the tutorial don’t match as you progress through it. For example it talks about a LoginBean file, but the file is actually LogInDao.I had to go back through the code and figure out what was wrong. I think this is right:



Figure : Verify user credentials

User is the file that does verification, we will see if I’m right! Although I think it should be UserController?



Figure : LoginDao validates User credentials

I think it is LogInDao that validates. Yes I was correct. Another annoying thing, Eclipse likes to put the login file under WEB-INF even though I said WEB-CONTENT, so just copy it over.

**Part II**

This part addresses management features such as Add, Update, List, and Delete. First is to create a Model Class that represents a ToDo Entity called,, well, ToDo!

Next we create a ToDo Dao interface object file in the dao folder.

Next is a DAO class tat provides CRUD operations for the table todo’s in the database. This code is very complex, it iterates through all the scenario’s too, depending on what method is called upon. TodoDaoImpl file. This file is key to making everything work as intended. I suspect I’ll spend a lot of time here! LOL!

I was wrong! The TodoController file is the Page Controller for the application. It handles all page requests, Process all the Http request handlers, and direct to the appropriate JSP page after data is stored in database. I’ll definitely have to go over this page many times.

07/23/20 8:30AM – Going over Part II before I move on to Part III. There’s a lot going on here! If things go sideways(which they always do when testing a new project) that’s the first place I’ll look! I have to todo-list and error page files to do yet. The todo list gather’s up all the todo’s the user selected.

**Part III**

This part is just configuring Tomcat to display the pages. Register page works, database was updated with new user info, although it does not redirect you to a todo list, I’ll figure that out later. I think I’ll try the User Management Project again using Hibernate as the JPA implementation and see what the differences are after the XML Hibernate Configuration

**Hibernate XML Configuration Example**

**(Fadatare, n.d.)**

**3:30PM**

Well I got sidetracked again! LOL 😊 But.I managed to fix up the User Management files. Apparently my dumbass mixed up demo, user, and users databases! HAHA. I figured it out though, and even have Hibernate going on it:

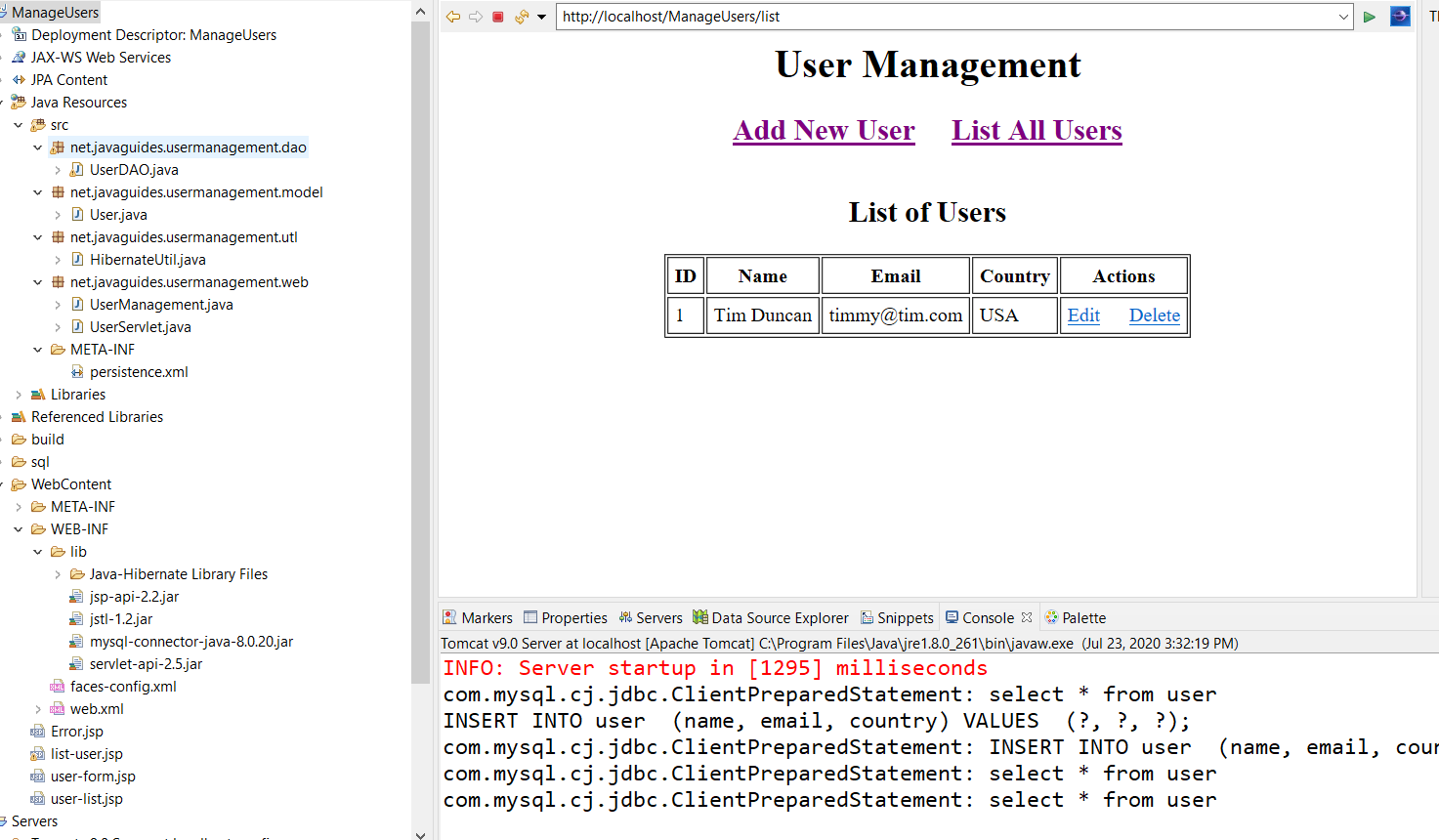


Figure : User Management converted to Hibernate

Sweet! Let the fun begin! Now to check the ToDoList files.

It appears to be working ok, except for one thing, I don’t have a database set up for it! HAHA! Maybe later I’ll get to it.

**07/24/20 9:00AM**

**Practice Projects available for learning:**

**Java EE 8 Development with Eclipse, 3rd Edition:**

* **Course Management –** A Domain Model based on Courses, Teachers, Classes, & Students. This project starts at the beginning, making an EER diagram, MySql, Tomcat, JDBC, & JPA with Hibernate. I’m going to start here since the book involves Eclipse.

**Java Persistence with Hibernate, 2005:**

* **Caveat Emptor** – A domain model based on an auction site,much like E-Bay.

**Hibernate In Action, 2005:**

* **Caveat Emptor** – Another domain model based on the same domain Model.

**Java Server Pages, 3rd Edition, 2003:**

* A variety of small examples/projects demonstrating JSP 2.0, JSTL, Java Beans, Enterprise Java Beans(EJB), and JDBC. Also good information on Generating Dynamic Content and JSP page creation development cycle.

**JSP- The Complete Reference, 2001:**

* Thorough JSP reference book, with a Customer Support Project

**Pro JSP 2, 2005**

* Another thorough JSP reference book, Chapter 9 has excellent JDBC information and Object Model Mapping

**07/25/20 10:00AM** – The User Management project works great now:

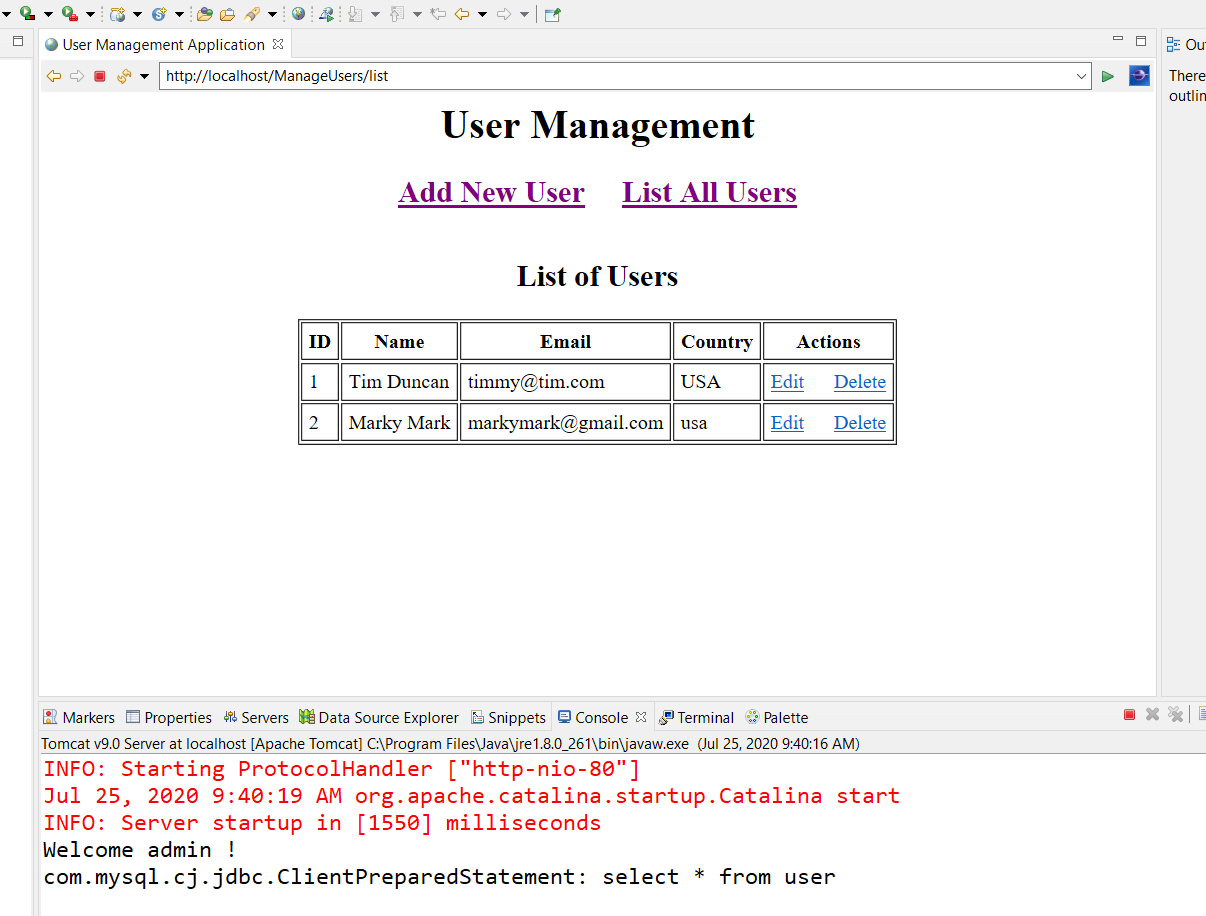


Figure 16: User Management before changing to Course Management

What I’m going to attempt is to take the lessons from this project, and modify it for use with the Course Management Project as the addCourse portion.

**12:00PM** Guess there is no easy way of doing this! As usual,,the coursemanagement import from the master directory wasn’t working out to well,,,It keeps reverting back to JSE-1.5, the Maven Reference Library is messed up,, blah blah..Looks like I will be starting from scratch again after all! Oh well, I need the practice.

4:00PM Well that was pretty brutal, gave me a headache! I deleted the whole CourseManagementJPA project and started a new one. Created the entities Person, Student, Teacher, & Course from scratch, easy enough. Then created the StudentDAO very carefully, now to track down the errors! LOL

**StudentServlet needs work!**

**07/26/20 9:00AM –** I need a new strategy, Nothing seems to be working, I’m missing something! Think I tried to run before I could walk. I may just close the CourseManagement project for now, build another tutorial with a combination of ManageUsers & ToDoList. So,,I’m going to start new documentation also.