This project has five objectives:

First, you are introduced to GlassFish. GlassFish is an open source application server that implements the latest JEE specification. This tool is used throughout the course. The NetBeans integrated development environment is used to build source code and interact with GlassFish.

Second, you build your first set of distributed systems. These are four small web applications using Servlets and Java Server Pages.

Third, you are introduced to simple mobile device awareness and adapting content to be suitable for either desktop or mobile devices.

Fourth, you are introduced to the MVC pattern if you have not used it before.

And finally, as in all projects this semester, you should reflect on the functional and non-functional characteristics (e.g. security, scalability, failure handling, interoperability) of your solutions. There will be questions on the midterm and final exam concerning these characteristics. You should be able to demonstrate a nuanced comprehension of course content and be able to explain the technical aspects in relation to potential real-world applications.

For each project task, software documentation is required. The software that you write (HTML files, Java files and so on) must contain comments that describe what each significant piece of code is intended to accomplish. Points will be deducted if code is not well documented. Read the documentation-related links provided on the course schedule (for class #1) to understand what is expected.

Be sure to consult the rubric linked from the course schedule for details on grading.

For each task below, you must submit screenshots that demonstrate your programs running. These screenshots will aid the grader in evaluating your project.
Write an index.jsp page that asks the user to enter a string of text data, and to make a choice of two hash functions using radio buttons. The hash function choices should be MD5 and SHA-1, with MD5 being the default. When the submit button is pressed a servlet is executed. The servlet must be named ComputeHashes.java. The servlet will compute the appropriate cryptographic hash value from the text transmitted by the browser. You will need to employ the Java crypto API to compute the MD5 or SHA-1 hash of the text. The original text will be echoed back to the browser along with the name of the hash, and the hash value. The hash values sent back to the browser should be displayed in two forms: as hexadecimal text and as base 64 notation. We will discuss the use of such hash values later in the course.

To compute the MD5 and SHA-1 hashes, use these standard java packages:

```java
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
```

To compute the Base64 encoding, use the following package:

```java
import sun.misc.BASE64Encoder;
```

The BASE64Encoder class is an internal non-documented class. BASE64Encoder objects have a method with the signature String encode(byte[]). It returns a base 64 string encoding of an array of bytes.

To compute the hexadecimal representation of a byte array, use the following code:

```java
// From the web site "Real's How To"
public String getHexString(byte[] b) throws Exception {
    String result = "";
    for (int i=0; i < b.length; i++) {
        result += Integer.toHexString((b[i] & 0xff) + 0x100, 16).substring( 1 );
    }
    return result;
}
```

Be sure to provide a user friendly and attractive user interface.

If you are unfamiliar with HTML forms, a simple explanation can be found at: http://www.w3schools.com/html/html_forms.asp.

So that you may test your program, here are example hashes.
Hashes of the string "Hello":

- SHA-1 (Hex): F7FF9E8B7BB2E09B70935A5D785E0CC5D9D0ABF0
- SHA-1 (Base 64): 9/+ei3uy4Jtwk1pdeF4MxdnQq/A=
- MD5 (Hex): 8B1A9953C4611296A827ABF8C47804D7
- MD5 (Base 64): ixqZU8RhEpaoJ6v4xHgE1w==

**Task 2**

Use the NetBeans Project Name: Project1Task2

You do not need to use an MVC framework for this project.

Later in the semester when we are studying the RSA algorithm, it will be useful to be able to do math operations on some arbitrarily large integers. Therefore Task 2 is to create a useful calculator app for this purpose.

Write a simple web application that allows a user to perform one of six operations on two, possibly very large, integers (x, y). The operations will include

1. Addition (x+y)
2. Multiplication (x*y)
3. An operation to determine if x and y are relatively prime
4. Modulo (x mod y)
5. A modular inverse (x⁻¹ mod y)
6. Raise x to the power of y (i.e. x^y)

A JSP page will present three input fields to the user. The first two will be used to collect the two integers, x and y. The third will be used to collect the operation type. The operations supported will be "add", "multiply", "relativelyPrime", "mod", "modInverse", and "power". Use drop down boxes in XHTML. A submit button will be provided and when it is hit a servlet will be visited. The servlet will be named BigCalc.java and will use the BigInteger class to perform the conversions from strings and the appropriate computation. The servlet will return the result to the browser marked up in HTML. You need to validate both integers and the operation. In the case of invalid input return an error message to the browser - but don't crash the server.

The BigInteger class has multiply, add, modInverse, mod, and pow methods to use. For the operation that determines if the two integers are relatively prime use the gcd() method of the BigInteger class. If the greatest common divisor of the two integers is one then the two integers are relatively prime. And finally, note that the exponent for the pow method takes an integer, not a BigInteger.

Be sure to provide a user friendly and attractive user interface.
Task 3

Use the NetBeans Project Name: Project1Task3
You do not need to use an MVC framework for this project.

Write another web application using NetBeans. This application will determine if a string entered into a browser is a palindrome. A string is a palindrome if it is empty, has a single character, or reads the same when reading from left to right or from right to left. Name your servlet Palin.java. Use an appropriate doctype for an Android mobile.

If you don't already have it, download and install the Google Chrome browser.

- Browse to your web application in Chrome
- Toggle device mode to mobile and choose the Google Nexus 4 device (https://developers.google.com/web/tools/setup/device-testing/devtools-emulator?hl=en)
- Reload the page.
- Produce a screen shot showing the emulator working on your web application. (See example screen shot in device mode for Task 4 below.)

Note:

- For judging a palindrome, only consider letters. Disregard case, punctuation and white space. For example "Madam I'm Adam" is a palindrome.

Task 4

Use NetBeans Project Name: Project1Task4
You **MUST** use an MVC framework for this project.

For task 4, you will build an application that searches for pictures of birds in the Bird Photo Gallery of the Smithsonian Migratory Bird Center (http://nationalzoo.si.edu/scbi/migratorybirds/featured_photo/).

The Smithsonian Migratory Bird Center does not have a public API, so we will therefore have to use screen scraping to get the information we need. Screen scraping means having a program (instead of a web browser) fetch HTML from a web site and then process it to extract useful data. You will write the code that makes requests to the Bird Photo Gallery and processes the HTML that is returned.
If the Smithsonian Migratory Bird Center had a public API that we could use, we could get structured data from it in a more straightforward and reliable way. Screen scraping is a handy technique to use with older and legacy systems from which we can’t get structured data, or newer sites that just don’t provide a separate API. Screen scraping is also called data scraping and web scraping and you can learn more about it at: https://en.wikipedia.org/wiki/Web_scraping.

Your app should start with a welcome page that invites the user to select from a list of birds the bird they would like a photo of. The welcome page on the right has a background image of a Northern Cardinal. You do NOT have to have a background image.

You can find the list of birds that the Bird Photo Gallery includes at http://nationalzoo.si.edu/scbi/migratorybirds/featured_photo/. If you examine the HTML for that page, you will see a <form> element that contains a <select> element, and it contains a large number of <option> elements. These can be used in your prompt and result JSP pages. When copying code from elsewhere, it is always good to add a comment citing the source.

Once the user selects a bird name and clicks submit, your servlet should search for a suitable image in the Bird Photo Gallery and return it to the browser to be displayed. For example, the screen shot on the left shows a picture of a Bald Eagle.

On this page, be sure to include:

1. The name of the bird in the response (E.g. Here is a picture of a Bald Eagle).
2. A link to the Smithsonian Migratory Bird Center Bird Photo Gallery.
3. The copyright notice of the photographer who took the picture. This information can also be screen scraped from the Bird Photo Gallery site. (E.g. on the above image, © Jack Moskovit).

4. The menu and button to allow another request.

For each bird, there are typically several images available. Therefore, your program should randomly choose an image each time the user searches for a bird name. In other words, your web app should not return the same picture of a Bald Eagle each time, but should randomly choose which Bald Eagle picture to display. For some birds there are only one image, for most there are many.

Your web app should adapt for desktop and mobile access by using different size images for each:

- Desktop: 500 pixels wide images.
- Mobile: 250 pixel wide images

The URL of an appropriately sized image should be returned to browser. It is not adequate to return the image of a too-large image, and use HTML or CSS to change its displayed size.

SCREEN SCRAPING HINTS:

Screen scraping is a craft that requires figuring out how the pages are organized and experimenting with searching and selecting the pieces of information you need. Go to the site (http://nationalzoo.si.edu/scbi/migratorybirds/featured_photo/), select bird names to view their galleries, and then examine the resulting HTML. Try doing searches in the code to find the URLs and photographer names that you will need.

Just like InterestingPicture makes a request to Flickr, you can craft a URL to make a request to the Bird Photo Gallery based on what you experiment with in the browser. Once you get the results page, choose one of the images randomly, and then extract the picture URL and photographer information about that image.

The image size that the Bird Photo Gallery displays is not necessarily the size that you need for desktop and mobile. Therefore you should experiment and find out how the site scales the images that it presents.
Notice that the Smithsonian Institution (which the Migratory Bird Center is a part of) has its own service for resizing images. These URLs start with https://ids.si.edu/ids/deliveryService, and take a size and a URL of the original image as parameters. Experiment with how to set the maximum WIDTH (not height) of the image to 500px (pixels) for desktop or 250px for mobile as you need for this project.

Make sure you have an appropriate DOCTYPE string (for desktop/laptop or mobile) defined as the first element of your HTML replies.

Exceptions:

• If a bad name of a bird is given, a photo is not found, or the site is down, then your app should fail gracefully by saying no images are available. It should NOT result in a 500 Internal Server Error.

You should submit screenshots demonstrating your application working.

• Include screenshots showing the welcome page and results pages with 3 different bird names.
• Include screenshots showing 2 different bird photos from same bird name request.
• Include screenshots showing the welcome page and 2 result pages on mobile

You are allowed to and encouraged to build your solution based on the InterestingPicture code you have been given. You MUST refactor it, however, to have variable and class names that make sense for your application. For example, you will lose points if your class is still named InterestingPictureServlet.

Questions:

Questions should be posted to the Blackboard Discussion Board, under the Project 1 Discussion Forum.

Summary:

There should be four projects in NetBeans.

The NetBeans projects will be named as follows:

• Project1Task1
• Project1Task2
• Project1Task3
• Project1Task4

You should also have four screenshots folders:

• Project1Task1 Screenshots
• Project1Task2 Screenshots
• Project1Task3 Screenshots
- Project1Task4 Screenshots

Be sure to review the Rubric linked on the course schedule for the first day.

For each NetBeans project, File->Export Project->To Zip… each. You must export in this way and NOT just zip the NetBeans project folders. Finally, zip all the NetBeans export zips and the screenshot folders into a folder named with your andrew id.

Zip that folder, and submit it to Blackboard.

The submission should be a single zip file.