Data-intensive computing: Hints and helpful links for project development, by Bina Ramamurthy

1. **Data Aggregation**
   Understand that most data is NOT available as web services. You have to "scrap" them different sources and formats: directly from the web page, using an API, using a URL (link), RSS feed, XML feeds, plain text, marked-up text, and finally standardized WS (SOAP/REST).
   Lesson: Be prepared to access the data using different methods: for FAO data you may use methods as shown below.
   1. Look at this prepared by CSE486 TA Hanifi:
      http://www.cse.buffalo.edu/~bina/cse486/spring2011/xml_sax.pdf
   2. Sample code for accessing XL data:
      http://java.sun.com/developer/technicalArticles/xml/mapping/index.html

2. Persistence (Data Storage)
There are many models to populate that data into the persistence storage. You can use elegant data access models: Data Access Object (design pattern), separate data administration for large databases, to embedded database connector codes: following links provides some examples. Choose the one that is comfortable for you based on your expertise level. You will have to use DAO for accessing Google App Engine.
   1. DAO: http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html
   2. JDBC example: http://download.oracle.com/javase/tutorial/jdbc/
   3. Java Persistence API: http://netbeans.org/kb/docs/javaee/javaee-intro.html
   4. JDBC connection code examples from CSE:
      https://wiki.cse.buffalo.edu/services/content/how-use-jdbc-oracle
      (see the link at the end of this article)

3. Web-services based enterprise application
The application you will develop will use the data stored, analyze it (analytics) and present the data in a suitable format. (We will give a set of drop questions that should be answered by your application: for now just work on the prototype. In response you may have to add fields to your persistence store.) Here are some links on web services.
   1. Oreilly's simple introduction to WS:
      http://onjava.com/pub/a/onjava/2001/08/07/webservices.html
   2. http://netbeans.org/kb/docs/websvc/jax-ws.html
   3. Restful Web services: http://netbeans.org/kb/docs/websvc/intro-ws.html
   4. Restful Web services with database access:
      http://netbeans.org/kb/docs/websvc/rest.html?print=yes

4. User interface
User interface can be constructed out of simple text boxes, drop-down and buttons. It can be simple JSPs. See links below.
   1. www.apache.org See Tomcat page
   2. JSP tutorial: http://www.apl.jhu.edu/~hall/java/Servlet-Tutorial/

5. Application server
You application server can Tomcat, Glassfish, or any equivalent.
6.IDE : Eclipse, Netbeans or command line (anything is acceptable)