

## CST 363: Databases – Project Database Design and Implementation Part 2

This is a continuation of the work done in project 1 where your consulting team designed a database for a pharmacy chain. In part 2 you will

- Revise and resubmit your design report from part 1.
- Implement functional requirements using Java and JDBC and Spring-Boot for the web application.

### Perform the following steps:

Make any needed revisions to your database design from project 1. During the implementation you may continue to revise the database design as needed to support the code for your implementation.

Review the drugs.sql script that creates and inserts 99 rows into the drug table. Make any changes necessary for the script to work with your drug table and then execute this file.

Install Eclipse IDE for Java Developers and the Spring Tool Suite.

### Eclipse and Spring Tool Suite installation

- If you do not have Eclipse on your computer, then download and install Eclipse IDE for Java Developers at

<https://www.eclipse.org/downloads/packages/>

- After installing Eclipse, install Spring Tool Suite
  - Starting from the Eclipse menu Help->Eclipse Marketplace
  - search for “Spring Tools”

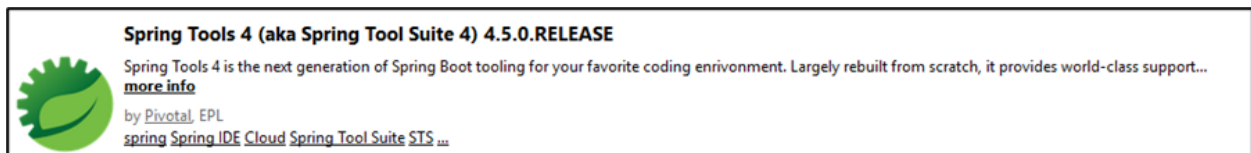


Figure 1 Spring Tools plugin for Eclipse

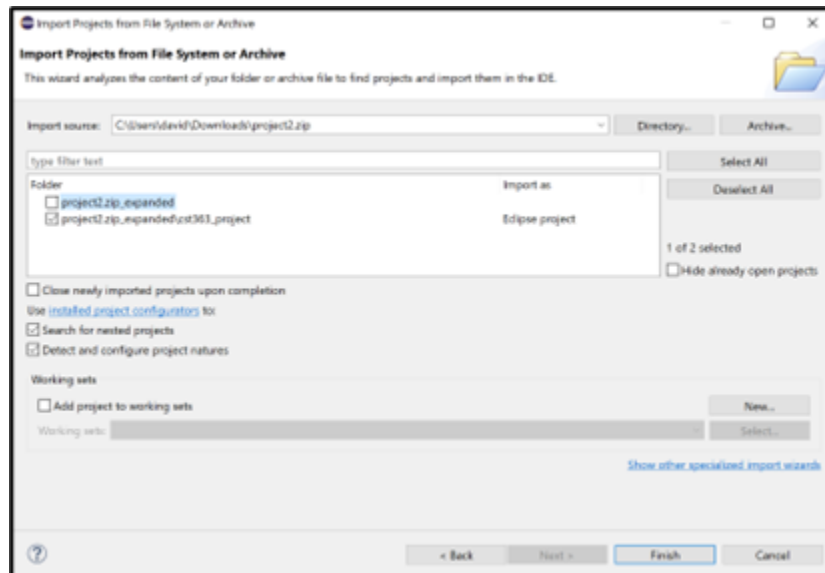
- install the plugin. During install it will restart eclipse.

### Importing an Eclipse project

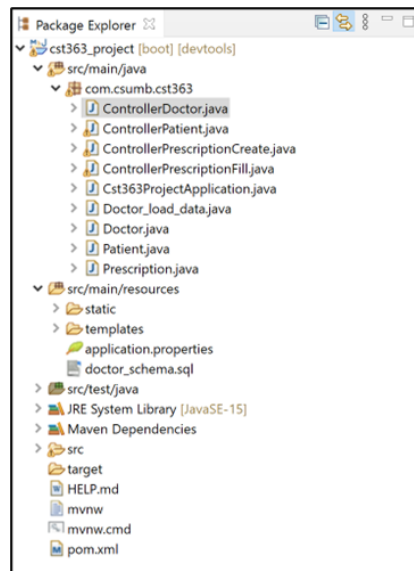
1. Download project2.zip from this assignment.
2. From the eclipse menu bar: File → import
3. Expand the “General” category and select “Projects from folder or archive”, then NEXT

## CST 363: Databases – Project Database Design and Implementation Part 2

4. Import Source. Click the “Archive” button and navigate to and select project2.zip which you downloaded in step 1.



5. Select the folder that reads “project2.zip\_expanded\cst363\_project
6. Make sure that “Search for nested projects” and “Detect and configure project natures” are both checked. (Should be checked by default)
7. FINISH
8. The project’s folders and files should be as shown below in the Package Explorer panel.



9. Locate the file application.properties located in the src/main/resources folder. Edit it so that the db\_name, password and username are correct for your system.

## CST 363: Databases – Project Database Design and Implementation Part 2

```
spring.datasource.url = jdbc:mysql://localhost:3306/db_name
spring.datasource.username = root
spring.datasource.password = secret
```

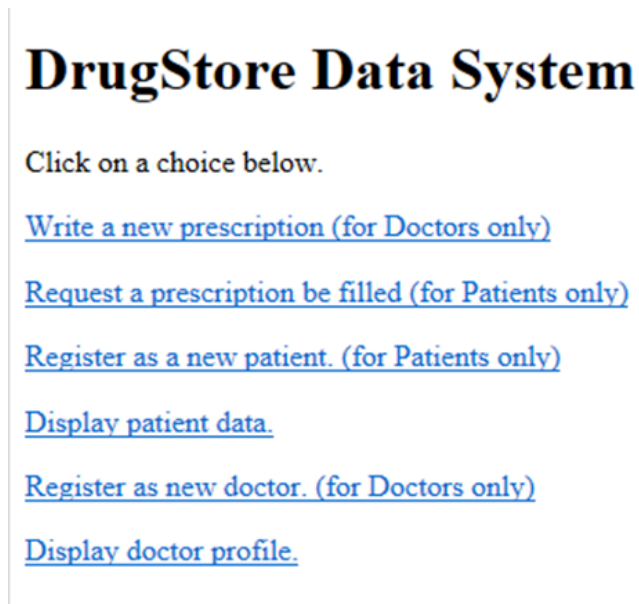
- **Starting and stopping the Spring Server**


Select the Application.java class (this was the class generated by spring tools when you created the project). → Run As → Spring Boot App

Start up messages from the server appear in the Eclipse console panel. You should not see any ERROR type messages and the startup messages end with

```
Tomcat started on port(s): 8080 (http) with context path ''
Started TestprojectApplication in 1.751 seconds (JVM
running for 2.683)
```

From your browser you enter **http://localhost:8080** and you should see the app main menu



To **shutdown the server**, click the RED terminate button  in the console panel.

**NOTE:** *If you make changes to java code, the server in most cases will auto-restart. Otherwise, shutdown and restart the server.*

*If you make any changes to the html files, application.properties or other files, you usually have to manually shutdown and restart the server.*

## CST 363: Databases – Project Database Design and Implementation Part 2

Each member of the team must code two programs. One from the application list and one from the web app list

### Application functional requirements

- Your team consists of 2 or 3 members. I will name these member1, member2 and member 3.
- Member 1 does the following Java JDBC program.
  - A program to generate 100 random patients, 10 random doctors and 100 random prescriptions.
  - Tips on generating random data at the end of this document. This program is a Java application with a "main" method and uses JDBC statements to obtain a connection and execute SQL statements. Name the java class DataGenerate and use the com.csumb.cst 363 package.
- Member 2 does the following Java JDBC program.
  - A pharmacy manager requests a report of the quantity of drugs that have been used to fill prescriptions by the pharmacy. The report will contain the names of drugs used and the quantity of each drug used. Input is pharmacy id and a start and end date range.
- If you have a member 3, then member does the following Java JDBC report program
  - An FDA government official is looking for the quantity of drugs that each doctor has prescribed. The report shows the doctor's name and quantity prescribed. Input is drug name (may be partial name) and a start and end date range.

### Web app functional requirements

Study the code in ControllerDoctor.java that is covered in the online lecture. It is an example of a controller that has functions for registering a new doctor, displaying and updating a doctor's profile.

Team member 1 does the following.

- A doctor creates a new prescription and the patient gets the prescription filled at a pharmacy. Complete the methods marked with //TODO in PrescriptionCreate and PrescriptionFill classes.

Team member 2 does the following

- A patient registers themselves to the system and chooses a primary care doctor. The patient enters their name, social security number, address, birthdate and chooses a primary care physician. The program validates the information for reasonableness. For

## CST 363: Databases – Project Database Design and Implementation Part 2

example a primary care physician should be a doctor who specializes in Family Medicine, Internal Medicine or, if the patient is a child, Pediatrics and not surgery or oncology. Complete the newPatient method marked with // TODO in PatientController class.

If you have a member 3, they do the following

- A patient updates their profile. Complete the methods getPatientForm and updatePatientForm marked with //TODO in PatientController class.

### All web app programs must also do the following.

- All user input from HTML forms must be sanitized. Read below about XSS attacks and check input for suspicious characters. Parameterized SQL statements must always be used. Never take user input and concatenate into an SQL statement string.

- An message (success or failure) can be returned to the client by the following code

```
model.addAttribute("message", "your error message");
```

All of the templates in the project have a placeholder for a message at the top of the web page.

- Perform data validation checks such as
  - Prescription quantity cannot be zero or negative. Very large numbers are also an error.
  - Names for people, cities, states cannot be a blank line or and must consist of alphabetic a-z or A-Z characters.
  - Zip Codes must be 5 or 5+4 digits.
  - Social security numbers must be 9 digits. Social security numbers never start with a 0 or a 9. The middle 2 digits are 01-99 (never 00). And the last 4 digits are 0001-9999 (never 0000).
  - Year must be 4 digits in range 1900-2022.
  - Other dates are of the format yyyy-mm-dd where mm is in the range 1-12 and dd in the range 1-31.
  - It is possible to validate addresses against a database of all valid addresses in the US. However that is beyond the scope of this project.  
<https://www.smarty.com/articles/usps-api>

## **CST 363: Databases – Project Database Design and Implementation Part 2**

### **Database Connections in a Spring Server Runtime**

- In a Spring server web application, database connections are created by the server. The Controller classes should not use the JDBC DriverManager to obtain a connection. Use the “getConnection” method provided for you in the controller class.

### **Revised report**

Make any changes you feel necessary to your ER model, schema and SQL queries from part 1.

### **Add to your revised report**

- For web applications, show screenshots showing
  - form input
  - the resulting output page
- For JDBC applications, show the console output from running the program.
- Label and explain each screen shot.
- Include screenshots showing invalid input and resulting error messages to demonstrate that you are checking for invalid input.
- Your report should be professional in quality as if you were a real consulting team.

**SQL file.** Create an SQL script file of your database.

- Use the Data Export tool in the workbench to export your database
- Menu → Server → Data Export → select the schema
- Export to a self contained file named “teamXX.sql” (XX is your team number).
- Check the box for “Include create schema”
- This will generate a file that contains create schema, create table and insert statements. You should inspect this file and verify it by using the File → Open SQL Script to load and run the script file.

### **ZIP file of Eclipse project**

Merge all the code changes from the team members into one Eclipse project. Then export to a zip file.

1. Select the project
2. Menu File → Export → General → Archive File
3. Take all the default options
4. Specify a file name of “teamXX.zip” for the archive file.

**What to submit.** three files:

## CST 363: Databases – Project Database Design and Implementation Part 2

- teamXX.pdf Revised PDF report with revision from part 1 and screen shots.
- teamXX.sql Workbench export of your database.
- teamXX.zip Eclipse export of your project code.

### Grading Rubric

140 points total for part 2

Item	Points Possible
Technical correctness. Did your work follow the methods and procedures you studied in this course? <ul style="list-style-type: none"><li>• Do the entities, attributes and relationships reflect the requirements of the project?</li><li>• Does the ER diagram use the correct symbols for entities and 1:1, 1:Many and Many: Many relationships?</li><li>• Is the ER model translated correctly in the SQL schema?</li><li>• Did you document how you checked for normalization and your decision?</li><li>• Are the SQL select statements correct?</li></ul>	10
<ul style="list-style-type: none"><li>• Did you document how each of the requirements is addressed by your design?</li><li>• Did you document any design issues or additional assumptions you made ? Are your assumptions reasonable?</li></ul>	10
Did you use good technical judgment and creativity in selecting names for tables and columns and choosing column data types and keys?	10
Clean work. Is the document well-organized? Is the writing clear and logically sequenced? Is it easy to find information in your report? Basically, is your report of professional quality?	10
Script file <ul style="list-style-type: none"><li>• Submitted sql script executes without errors</li></ul>	25

## CST 363: Databases – Project Database Design and Implementation Part 2

<p>Screenshots</p> <ul style="list-style-type: none"><li>• entering a new prescription</li><li>• entering an invalid prescription (i.e. invalid SSN, name or drug)</li><li>• Successful register of new patient</li><li>• Patient requesting a prescription be filled</li><li>• Failure when patient requests prescription be filled</li><li>• Registration failure for invalid data</li><li>• FDA or Pharmacy manager report example output</li></ul>	25
<p>Java code</p> <p>Inspect the Java code for the 4 methods in ControllerPrescription.java</p> <ul style="list-style-type: none"><li>• addPrescription<ul style="list-style-type: none"><li>o validate the doctor SSN and name, patient SSN and name and drug</li><li>o perform SQL insert to prescription table</li></ul></li><li>• processFillForm<ul style="list-style-type: none"><li>o validate pharmacy name and address</li><li>o SQL select for rxid (prescription key)</li><li>o SQL update of prescription row</li></ul></li><li>• pharmacyReport<ul style="list-style-type: none"><li>o SQL select over prescription table for SUM(quantity) for date range and pharmacyID</li><li>o May be for single drug or multiple drugs</li></ul></li><li>• fdaReport<ul style="list-style-type: none"><li>o SQL select over prescription for SUM(quantity) by doctor for date range and drug name</li></ul></li><li>• Method should have logic to get connection, close connection, try-catch for exceptions</li></ul> <p>Use PreparedStatement with SQL parameters (?). Do not embed user data into the SQL statement string!</p>	50

### Reference and Background information

- Spring uses Model-View-Controller pattern



## CST 363: Databases – Project Database Design and Implementation Part 2

Spring uses a Model-View-Controller pattern for web applications. The model is the MySQL database and the classes Patient, Doctor and Prescription which are used to transfer data to/from the view templates. The views are parameterized html files called “templates” that contain placeholders for values determined at runtime. The controller classes contain the application logic.

When the Spring server receives a request message it routes the message to a controller method based on a url pattern. The `@GetMapping` or `@PostMapping` annotations in the controller class define these patterns.

Included in the server message is an HTTP VERB which is just the word “GET” or “POST”. The “GET” verb is used when you enter a URL into the browser’s entry field or click on a hyperlink. A “POST” verb is used when you fill out an HTML FORM. Both the URL string and the HTTP VERB are used to route the incoming request to a controller method. If the server cannot find a match for the URL and HTTP VERB, a status code -404 (NOT FOUND) is returned to the client.

The controller method reads or writes to the database and returns

- the name of a template file (the filename without the .html suffix).
- values to be substituted for placeholders in the template file. The values are attributes that the controller adds to the spring Model object.

The ThymeLeaf template engine that is part of the Spring server processes the template replacing the placeholders with Model values.

The templates have been given to you for this project and should not require any changes. But if you make changes and there are syntax errors in the template or are missing Model values, then the template engine will write error messages to the Eclipse console.

Here is the list of application URLs and Verbs

http verb	url path	action
GET	/index.html or /	display menu page
GET	/prescription/new	display blank form for a new prescription. Used by a doctor.
POST	/prescription/new	process the new prescription completed form.

## CST 363: Databases – Project Database Design and Implementation Part 2

GET	/prescription/fill	display a blank form to request a prescription be filled by a pharmacy. Used by the patient.
POST	/prescription/fill	process the fill request.
GET	/patient/register	display blank form to register a new patient
POST	/patient/register	process the new patient registration
GET	/patient/edit	display blank form to search for patient record.
GET	/patient/edit/nnnnn	display patient record for patient id nnnnn
POST	/patient/edit	process a patient record update.
GET	/doctor/register	display blank form to register new doctor
POST	/doctor/register	process doctor registration
GET	/doctor/edit/nnnnn	display profile for doctor id nnnnn
GET	/doctor/edit	display form to to update doctor's profile
POST	/doctor/edit	process doctor profile update

Table 1

### Tips on generating random data in Java

- Java has a random number generator. The `nextInt(a)` method will generate a uniform random integer in the range  $[0, a-1]$ . To generate a random integer in the range  $[A, B]$

```
import java.util.Random;

Random gen = new Random();
int x = A + gen.nextInt(B-A+1);
int y = A + gen.nextInt(B-A+1);
```

- Using `new Random()` will generate a new sequence of pseudo random numbers each time you run the program. If you want a predictable sequence of pseudo random

## CST 363: Databases – Project Database Design and Implementation Part 2

numbers you can use a “seed” when you create the Random generator. A seed is an integer chosen by you.

```
Random gen = new Random(2222);
int    x =    A + gen.nextInt(B-A+1);
int    y =    A + gen.nextInt(B-A+1);
```

- To choose a random string value from a list of strings.

```
String[] specialties = { "Internal Medicine",
    "Family Medicine", "Pediatrics", "Orthopedics",
    "Dermatology",    "Cardiology", "Gynecology",
    "Gastroenterology", "Psychiatry", "Oncology" };

String random_specialty =
    specialties[gen.nextInt(specialties.length)];
```

- To generate a random date in the format YYYY-MM-DD. The following code generates a random day in the year 2021.

```
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Date;
import java.util.Random;

SimpleDateFormat simpleDateFormat =
    new SimpleDateFormat("YYYY-MM-dd");
Calendar c = Calendar.getInstance();
c.set(Calendar.YEAR, 2021);
c.set(Calendar.DAY_OF_YEAR, 1);
c.add(Calendar.DAY_OF_YEAR, gen.nextInt(365));
Date dt = new Date(c.getTimeInMillis());
String random_date = simpleDateFormat.format(dt);
```

- You might want to check this out to assist in generation of test data.
  - <https://github.com/datafaker-net/datafaker> (code download)
  - <https://java-faker.herokuapp.com/> (demonstration site)

### About URLs

a URL is composed of parts:

## CST 363: Databases – Project Database Design and Implementation Part 2

`http://hostName:portNumber/path?query_parameters`

The “http:” part is called the protocol. Other common protocols are “jdbc:”, “file:”, “ftp:” It specifies the type of communication used between the client and the server.

The portNumber defaults to 80 and often does not appear. But the Spring server uses port 8080 so we must explicitly enter it.

The path is everything after the “/” until a “?” or the end of the URL. The path looks like a directory and file name but the server can interpret it to be a file or a method name or any way the server wants. The path may contain program parameters such as

`http://localhost:8080/patient/edit/12345`

If there is no path, as in the URL `http://localhost:8080` the server will use a default path “index.html”

Following the “?” are a list of name,value pairs.

`http://localhost:8080/patient/edit/12345?lastname=Johnson&firstname=Jim`

### Cross Site System attacks (XSS)

To prevent XSS attacks (this is where a malicious user enters Javascript code into an entry field on an HTML form. If this string data is stored into a database and later redisplayed it can direct the browser to a malicious website.

To prevent this attack, all data from entry fields MUST be sanitized. One simple way to do this is to check that none of the characters `< > & ' ' ( ) # & ; + -` appear in the data. If the input data contains any of these characters, do not accept the input.

### SQL Injection Attacks

When coding a web application it is important to use coding techniques that prevent your application from being vulnerable to SQL injection attacks. An SQL injection attack is when a malicious user enters a fragment of SQL or malicious code into an html form as part of some entry string field such as a name or an address. If this malicious code becomes concatenated as part of an SQL statement and executed, it may cause serious problems.

Never use string concatenation with user data to an SQL statement string.

```
String sql =
```

```
"select name, ssn, zipcode from patient where id = "+p.getId();
```

where `p.getId()` refers to a string value entered by the user on an html form page.

## CST 363: Databases – Project Database Design and Implementation Part 2

The proper way to do this is using parameterized SQL statements.

```
String sql = "select name, ssn, zipcode from patient where id = ?";
```

```
PreparedStatement ps = con.prepareStatement(sql);
```

```
ps.setInt(ps.getId());
```

Using parameterized SQL (statements with ? for user values) avoids this problem by keeping the user data separated from the SQL statement text.

### References

<b>HTML</b>	<ul style="list-style-type: none"><li>• <a href="https://www.w3.org/MarkUp/Guide/">https://www.w3.org/MarkUp/Guide/</a></li><li>• <a href="https://www.w3.org/MarkUp/Guide/Advanced.html">https://www.w3.org/MarkUp/Guide/Advanced.html</a></li></ul>
<b>JDBC</b>	<ul style="list-style-type: none"><li>• <a href="https://www.tutorialspoint.com/jdbc/index.htm">https://www.tutorialspoint.com/jdbc/index.htm</a></li><li>• My own guide to JDBC <a href="https://docs.google.com/document/d/1PUoIUJILVQifyb1Fqloaqk837wvM6joh/edit?usp=sharing&amp;ouid=101749804057693144715&amp;rtpof=true&amp;sd=true">https://docs.google.com/document/d/1PUoIUJILVQifyb1Fqloaqk837wvM6joh/edit?usp=sharing&amp;ouid=101749804057693144715&amp;rtpof=true&amp;sd=true</a></li></ul>
<b>Spring</b>	<ul style="list-style-type: none"><li>• <a href="https://spring.io/guides/gs/relational-data-access/">https://spring.io/guides/gs/relational-data-access/</a></li></ul>