Reddit Reader

Web Project

Assignment 01

CST8288 2020 Spring

Table of Contents

Important Notes

SQL

Do **Not** use the old SQL script from prelab.
There are some minor changes.

Skeleton

1) Do **Not** use the old skeleton from prelab.
2) Read all the comments in the skeleton, I have explained many things in the comments.
3) Do not use any methods that are @Deprecated.
4) Project is made for NetBeans 11 however, if you desire you can use other IDE's but you have to handle the conversion.

Output

Simply run your project and you can see all example tables. The pages that say 404 are to be completed by you. The output of the "Images" page is up to you. You can use the HTML code for image grid plus the CSS provided to use my suggestion of how the page will look like.

Bright Space

Keep an eye out for updates that will be posted on bright space assignment page and announcements.

Due Date

June 28th Midnight however, you can submit till **July 5th Midnight** without penalty.

Penalty

10% off for every day late, zero on 5th day.

Do not Touch (unless needed for common issues)

1) Do not touch **persistence.xml** unless you know you can fix it yourself. However, be aware of its content.
2) Do not touch **context.xml** unless you know you can fix it yourself. However, be aware of its content. You can change the URL of your webpage through **context.xml**.
3) **GenericDAL** and **GenericLogic** can be expanded for more functionality but do not reduce or change the existing code.
4) Do not recreate your Entity classes. Use the ones provided in skeleton.

Attachments

1) You need to download the SQL script attached and run it in your workbench. Do it again even if you have done it in prelab.
2) Download the skeleton code, unzip it and open it using NetBeans 11 LTS.
3) Class and Sequence diagrams.
4) ERR diagram for DB.
5) "missing users.m2.repository.org.glassfish.jaxb.jaxb-runtime.2.3.1 jar files.zip"
   a) When you do clean and build or Junit test you might get few warning exceptions about missing jar files. These warnings have no effect on your code.
   b) This zip file has the missing jar files. copy the content of the zip file to address below: "C:\Users\[your user]\.m2\repository\org\glassfish\jaxb\jaxb-runtime\2.3.1\"

Submission

Submit one zip file of your project.

[firstName]-[lastName]-[labSection#].zip

ex: shawn-emami-11.zip

There is no demo for this assignment. Just submit to Bright Space.

Export Instruction Slides

If you don't know how to export look at:

"BrightSpace/Content/Lecture Material/Runnable Jar, Archive and war files".

Common Issues

1) Make sure you have JDK 1.8 in c:\program files\java, not JRE 1.8.
2) You need MySQL server 8 to run the SQL script correctly it will not finish correctly if ran in 5.6 version. 5.7 sometimes works but you will need to remove all the “VISIBLE” keywords from the SQL script. You can use MariaDB in xampp as well, but it is up to you to fixed it. Using MySQL 8 is much easier.
3) If you start tomcat in NetBeans and the password screen ask for username and password with **XDB** message, it means you have a conflict with your oracle services. Search for Services in start menu and run it. Scroll down till you find oracle services. There should be 2 of them running, can be more. For each running oracle service, right click and go to properties. Then from the drop-down menu and choose **disabled**, then click stop. When done click apply and finish. Go back to NetBeans and start tomcat server again. Now instead of XDB it should say tomcat manager.
4) Make sure you have your output tab open to see the logs. If you don't see it click on window in toolbar and choose output. When you run your application for the first time it should open two new windows in output called tomcat and tomcat log. If they are not there click on window again and choose services. In the opened tab expand servers and right click your tomcat server. Choose view server log and view server output or just restart. They should both popup.
5) The content of outputs are logs. Meaning they record everything that has happened. So, to find your error check the timestamp as you might have old errors that are irrelevant. Scroll all the way down.
6) If in your log you are getting an error with access denied in it, it means you do not have the user: cst8288 and pass: 8288 in DB. Open workbench, double click connection, choose file and then run SQL. Navigate to the location which you downloaded the newest SQL script and run it.
7) If you are getting DB not found it means you did not load the SQL script attached to the assignment. Open workbench, double click connection, choose file and then run SQL. Navigate to the location which you downloaded the newest SQL script and run it.
8) If you are getting an error saying "zeroDateTimeBehavior must be one of the", navigate to context.xml in your netbeans. Look for zeroDateTimeBehavior=convertToNull and change it to zeroDateTimeBehavior=CONVERT_TO_NULL.
9) If you are getting an error regarding your time zone especially if your base windows language is not English, navigate to context.xml. Look for “?zeroDateTimeBehavior=convertToNull” and change it to: ?zeroDateTimeBehavior=convertToNull&useUnicode=true&useJDBCCompliantTimezoneShift=true& useLegacyDatetimeCode=false& serverTimezone=UTC
   All in one line.
10) If you try to run your code and nothing happens with no errors, try changing your default browser from NetBeans. Go to tools/options/general then change web browser to chrome or Firefox. If you have already opened your project also right click on project and go to properties/build/run and change browser to chrome or Firefox.
11) If you are on mac and you run windows on parallel, when you open a project it might suddenly disappear from project list. With no errors or a NullPointerException. To fix this move your project to drive c:\ and try opening it again. If it opens but you cannot save you will need to change permissions on your project directory for write permission. If you cannot do it come and see me.

Requirements

1) Finish the DAO design pattern. Review the Class Diagrams Posted. You can add extra methods if needed.
2) Create a Junit for ImageLogic. It must be in the test folder.
3) Complete the LogicFactory using the sequence diagram provided.
4) Complete ImageDelivery using the sequence diagram provided.
5) Make Servlets for view and create for each table (Image does not need create), and ImageView servlets in view for displaying images from reddit.
6) Follow proper coding and naming conventions.
7) Provided proper documentation on methods that need it, use your judgment. You don't need to use any fancy JavaDoc keywords. You simply need to provide meaningful definition and purpose.
8) Create a Sequence diagrams for ImageView doGet.

Bonuses

1) Add update
2) Add delete
3) Add functional search for all tables, search should be able to at least search two columns.
4) Actively change the Board to download images on ImageView.
5) Create Input servlet for Image, allow image upload.
6) Add user verification for login. to do so you need:
   a) Sessions, cookies or URL editing. Session is the best way.
   b) @WebFilter which will allow you to filter incoming URL requests. It also allows redirecting to other page, convenient for login page.
7) Beautify the web pages as you see fit. Creativity effects bonus award.
8) Add page number or infinite scrolling to ImageView.
9) Use JSP, **don't** use scriptlets (<% %>). Use JSP in combination of Servlet using Expression Language.

Instructions

Reddit Information

If you have not done so already, finish the reddit steps of the prelab. In short you must do the following (detailed instructions are in the prelab):

1) Create a reddit account
2) Create a reddit webapp from setting/privacy/dev section.
3) Copy the information generated by reddit.com to the your netbeans code. Inside of the reddit package class Reddit.java.
4) You need to copy information regarding CLIENT_ID, CLIENT_SECRET, REDDIT_USER.
5) Finally look at APPID value and replace **yourname** with your name (no spaces).

Finish DAO design pattern

Review the Class Diagrams Posted. Class diagrams for all packages are attached. You can add extra methods if needed. Take advantage of Account code that is already given to you.

1) Create the rest of logic classes. Take advantage of Class diagrams attached.
   a) Pay close attention to accessors on Class diagrams for logic.
   b) Logic should do proper error checking for input data, such as string is not too large, what can and cannot be null. This information can be found in ERR diagram.
   c) Use current date inside of createEntity of logic if date parsing has failed. Look at common.ComponentExamples for examples.
   d) Inside of createEntity do **not** create logics, dals, and/or entities of other types. Look at attached sequence diagram for example BoardLogic::createEntity.
   e) Inside of updateEntity which is part of update bonus, you can create other logics and entites.
2) Create the rest of dal classes using the GenericDAL provided. Take advantage of Class diagrams attached. When using findResult and/or findResults you need to use the name of @NamesQuery of appropriate Entity. Look at Account and AccountDAL for examples.
3) You are required to provide get for all columns of every table.

Servlets

Servlets to view each Table

Every table should be displayed in a web page. You can put the code for it in one servlet which updates the table based on the URL or you can create 3 different servlets for each table. A servlet for each table is a bit more coding but the easiest way.

1) Look at the sample servlets provided for Account. JPS is bonus.
2) When creating the html output use getColumnNames, getColumnCodes and extractDataAsList. do not hard code or type the data one at a time.

Servlets to create Entities

Create input servlets. Remember auto generated IDs are filled by the DB, no need to ask for them.

1) Use CreateAccount as an example. Also look at sequence diagram for CreateBoard::doPost.
2) You do not need to create an input servlet for Image. Bonus if you do.

Servlets for ImageView

Create a new servlet called ImageView which displays images downloaded from reddit. By default, all images are displayed, filtering images by board is bonus. This class will be mostly up to you to finish. All your work will be in doGet.

doGet

1) You need to make a directory at **System.getProperty("user.home") + "/My Documents/Reddit Images/"** using java. Do this in your doGet or init of ImageView. You can use FileUtility::createDirectory. Do **not** change this path.
2) Using LogicFactory get the logics you need to create and add an Image to DB. Don't forget the dependencies.
3) Get the board you want to use. You will use the name of this board object in reddit buildRedditPageConfig method.
4) Use the example provided in **reddit.TestRunReddit::exampleForReadingNextPage** to see how to use the reddit object.
5) Create your custom lambda to create and Image entity, download it and add it to DB.
    a) Only accept post that are over18 and are images.
    b) The image should not already exist in the db.
    c) You must use createEntity to create entities, do not use the constructor of entities outside of createEntity method.
    d) Download it to your computer using FileUtility::downloadAndSaveFile(String url, String dest).
        1) Destination is the folder you made in step 1.
    e) Extract the data from the Post object which is the argument of your lambda. Add them to a newly created Map using ImageLogic static variables as keys.
    f) Create the Entity image, set the dependency and add it to db.

processRequest

Create a basic html page like other view servlets. However, instead of the table you can use the html code below. This is an example for one image. The associated CSS is in webpages/style/imageview.css. Feel free to modify this if you like.

```
<div align="center">
    <div align="center" class="imageContainer">
        <img class="imageThumb" src="image/[image_name]"/>
    </div>
</div>
```

The value **Image_name** above is the name of the image which can be extracted from an Image Entity using FileUtility::getFileName(url:String).

Servlet for ImageDelivery

Follow the instruction provided in the sequence diagram.

In ImageView html code we set the src for img html tag in format of image/[image_name] which is equivalent to the idea of image/*. ImageDeliver pattern is /image/*, meaning anything that has **image** in its path will come to this servlet. Second half (after /) is the name of the image saved on your computer. Knowing the name and the directly the images are stored at we can open the file and send it to client.

Browser will read the html that was send by ImageView first. While rendering the page the browser will ask the server for any media files to be send to the browser. At this point ImageDelivery will be called to send the images back to browser. The original html send to browser is only text, it includes no media, CSS or JavaScript in it. Assuming the CSS and/or JavaScript are written in a sperate file.

Junit

Create Junit for Image logic like the sample in your assignment. All Methods and their edge, normal, and invalid states must be tested. You can use the example provided for AccountTest as starting point.

Styling and Conventions

Follow proper coding and naming conventions.

- Everything must be properly indented.
- All class names must start with capital letter.
- Package names must all be small letters.
- All instance variables must be private.
- Static variables should be accessed from Class not instance.
- Variable names must make sense and be meaningful.
- If you are unsure about anything ask, do not assume anything.

Documentation

Provided proper documentation on methods that need it, use your own judgment. You don't need to use any fancy JavaDoc keywords, like @link, @see, <Code>, and <P>. You simply need to provide meaningful definition and purpose.

UML Diagrams

Create a Sequence diagrams for ImageView doGet.

Approach

Suggested work flow (somewhat contradicts weekly schedule)

1) Read the sequence and class diagrams first, take your time and become familiar with everything.
2) Take a good look at AccountDAL.java then start creating classes in dal package. You do not need to make everything. Start with findAll then later come back and finish the rest. Each dal class must inherit from GenericDAL.
3) With DAL somewhat done start on logic. Again, look at AccountLogic.java first then start making the rest. Each logic class must inherit from GenericLogic. Since you only have finished findAll only finish getAll in logic. You can leave the other methods empty for now.
4) Finish LogicFactory as it is the only way you can make logic classes. Pay attention to access level of constructor in logic, you can see it in class diagram.
5) Start on your first servlet. Like before, look at AccountTableViewNormal.java first. Remake this for every single entity or if you feel comfortable you can make one servlet to display all tables. After you get more comfortable you can try the JSP version for bonus.
6) At this point you should be able to see all tables. Now go back to step 2 and 3 and finish all left over methods.
    a) Important note on create entity method. Do not create different type logics within createEntity.
    b) Finish the create entity for host first as it is simpler than the rest.
7) Create the servlet for create host first which again you can take advantage of createAccount. After you are done go back to step 6 and finish other create entity methods in their appropriate servlets (if you have not done so already). Like step 5 you can have one servlet for each table or one for all.
    a) You do not need to create a servlet for creating image.
8) Look through reddit.Post.java, and reddit.TestRunReddit::exampleForReadingNextPage to get a better understanding of how reddit interface actually works. If you have not done so already, do the same thing you did in prelab for reddit.Reddit.java class and copy your info to it. Look at heading **Reddit Information** in page 8 for more info.
9) Create a new servlet for ImageView. Look at page 9 for more details
10) At this point if you run the project and go to image link it won't show any images. Because you don't have a delivery system for images. So, time to make one. Look at page 10.
11) You are done

Suggested Weekly Schedule

Week 02 – Read the Assignment and Class Diagrams

Make sure to read everything. Quiz 2 and 3 will include questions from assignment writeup.

Week 03 – Complete Logic and DAL

All you need for this is inside of skeleton already provided to you. Look at Different Account classes.

Week 04 – Complete Junit and View Servlets

Use AccountTest as example to complete the junit for ImageLogic. Create the remaining view servlets.

Week 05 – Complete Create Servlets

Finish the servlets that display the tables. Finish the simple ones before you try the bonus.

Week 06 – Complete ImageView and ImageDelivery Servlets

It should be very small amount of work, good time to catch up on any work you are behind.

Midterm Break – Complete the Sequence Diagram

Finish the sequence diagram. Double check your work. Try bonuses or just relax.
Do not forget to submit.

Hibernate Notes

Adding, updating, and/or removing

Hibernate is designed to be OOP, meaning when you are trying to add an entity to a table you need to make sure your object does not have null dependency/s. For example, when creating a new Entity to be added to DB, its dependences must also be set on the Entity object. These are the steps to have in mind, look at doPost method in createAccount.java for an example:

1) Create all needed Logics. Primary logic (logic of entity to be added to DB) and logics for dependency entities.
2) Using methods from primary logic to check for duplicates at this point.
    a) You can use **request.getParameter(key:String):String** to get a specific key.
3) If duplicate does exist
    a) Complete errorMessage and continue.
4) If duplicate does not exist
    a) Call createEntity on primary logic and pass request.getParameterMap():Map<String, String[]> to it. The return is an entity.
        i) If your entity has dependencies call on relevant logics and use their get methods to find dependent entities.
        ii) Set the entities returned from getters to the entity created using createEntity.
            (1) Do not create entities of other types in createEntity.
    b) Finally call add/update on primary logic and pass the entity created using createEntity.

URL Patterns

These patterns already match the “href” in **index.html**. If you do not follow these, you must fix **index.html** and other references.

Viewing Tables

- /AccountTable
- /HostTable
- /BoardTable
- /ImageTable

Creating Entities

- /CreateAccount
- /CreateBoard
- /CreateHost

Image Page

- /ImageView

Image Delivery

- /image/*

DB Details

- All String are by default minimum 1 character long if not null.
- No column can be null.
- Lengths can be found on the EER diagram.

Account

- id is auto generated
- nickname
- username
- password

Image

- id is auto generated
- board_id, set indirectly by setting board object to image entity
- title, taken form Post::getTitle
- date, taken form Post::getDate. You can use the convertDate method in ImageLogic to convert to String.
- local_path, taken from imageDirectory plus FileUtility.getFileName(url)
- url, taken from Post::getUrl

Host

- id is auto generated
- name, name of the website.
- url, address of the website.
- extraction_type, can be “json”, “html”, “xml”

Board

- id is auto generated
- url, address of the webpage, such as subreddit full address.
- host_id, id of host which contains this page.
- name, name of the webpage (subreddit).