Software Design and Development (SDD) Coursework specification

Important Information

This coursework is an individual assignment. You must complete the work yourself, without assistance from other students. Do not ask other students to help you, and do not share code or reports with other students. All parts of your submission will be checked for similarity to other student submissions. If you try and work together, or share your code or report, this will be detected and you will be punished to the fullest extent permitted under University regulations.

Introduction

The coursework assignment for this module requires you to work with a software project provided by the module staff and to complete a series of tasks outlined in this document. The tasks will ask you to complete several pieces of functionality to complete a Graphical User Interface (GUI) based crime data search engine. You will also write and submit a short report on your submission, describing how your code works for each of the tasks.

You must use what you have learned during the module to write the code to complete these tasks. As you complete each task you should find the program grows more functional.

Task specifications

To complete this coursework you will work through the tasks specified below. **It is strongly recommended that you complete the tasks in the order suggested**, as they build on one another and later tasks may not make sense if the preceding tasks are not completed. You may like to return to a task after you have completed it to add more features, or to increase the complexity of your solution/algorithm; more complete and challenging solutions may be awarded more marks.

It is strongly recommended that you Export your project after you complete every task so that you have a working copy ready to submit at every stage. You should submit the last working version of your solution – **your code must compile and run when it is tested** for marking. Submitted solutions with compiler errors will have a heavy penalty to the marks awarded. Details on how to export and submit your solutions are provided below.

You are going to develop a GUI based mini search engine for the crime data available from the https://data.police.uk/ and will carry out associated tasks. This website provides open data about crime and policing in England, Wales and Northern Ireland. You are provided a data dump of some of the data from this website that you need to use to build this search engine. You are required to complete the following tasks that tests your ability to design and develop a java based fully functional application and will cover various aspects of the SDD module including unit testing, database access from Java, Regex and string processing, GUI development and drawing UML diagrams.

Task 1 – Data Import

Import the crimedata.sql file into your mysql server database using the PHPMyAdmin application on the LAMP server. Make sure that the import creates a `crimedata` table and has the following columns: crime_ID, month, reported_by, falls_within, longitude, atitude, location, LSOA_code, LSOA_name, crime_type. The explanation of each column is provided in the appendix A.

If imported correctly, your table will have *66,683* records.

Task 2– Unit Testing

Write Unit Tests for all your classes and methods you implement in tasks 3,5, and 6.

Task3 – Database Handling

Create a class called `DatabaseHandler`. This class will have a method called `handleDbConnection`. It has no input parameters and returns a `java.sql.Connection` object. You should use your mysql credentials here to make a connection to the database that contains the `crimedata` table. Other methods and classes implemented as part of tasks below will call the `handleDbConnection` method to use the connection object and query the database.

Note: Do not store your credentials in your final submission, instead you should use double quotes (“”).

Task 4 – GUI building

Create a class called `GUIHandler`. This class will be a `JFrame` with appropriate GUI (java.awt or javax.swing) components that allow searching for various crime records by:

- looking up using longitude and/or latitude (user will be able to enter any possible longitude and latitude) and displaying first ten matching results,
- looking up using LSOA name (user will be allowed to enter any LSOA name including any regular expression for searching LSOA name) and displaying first ten results, and
- searching by crime types (for example, user can use an existing crime type from GUI component for their search) and displaying first ten results.

There will be an appropriate GUI component for displaying results in all these cases.

For implementing the GUI, give appropriate consideration to the aesthetics and also use the most appropriate components for a rich GUI and a seamless user interaction.

Task 5 – GUI functionality implementation

To fulfil this task, you need to make your GUI that you developed in the Task 4 functional. Write a class `GUIImplementation`. This class will contain methods and implementation that allows GUI in `GUIHandler` class to be functional. This includes using `DatabaseHandler` class and supporting functionalities such as:

- when someone attempts to search using longitude and/or latitude, it retrieves all the data matching the search input and displays first ten results if the matching data exists,
- when someone attempts to search using LSOA name it retrieves all the data matching the search input and displays first ten results if the matching data exists, and

- If someone attempts to search using a crime type/category then it retrieves all the data matching the search input and displays first ten results if the matching data exists.

Task 6– Data Quality Check

Write a class called `DataQualityCheck`. This class will have appropriate implementation and will use the `DatabaseHandler` class to:

- Find records within database that have no crime_ID, and export all these records in a file called `nocrimeid.txt`.
- Find records that have duplicate crime_ID and export these records in a file called `duplicatecrimeid.txt`.

Task 7 – Errors and Exceptions Handling

You are required to have appropriate Exceptions and Error handling mechanism throughout your implementation. In addition, write two of your own custom Exception classes and use them as part of your implementation using `throw` and `throws` keywords.

Task 8– Class Diagram

Draw a Class Diagram for describing and documenting all of your implementation using Unified Modelling Language (UML).

Task 9 – Additional Features

You should pick at least one of the sub tasks in this section and complete it. You may wish to complete more than one, and to add your own additional features, both of which can allow you to gain more marks for your submission. Some of these tasks require you to modify the other classes in the software – do so carefully and make sure you have exported a backup; a fully working version of your submission before you change anything in case you break existing functionality.

9.1 Visualisation to compare counties

Show a visual representation in your GUI using a bar chart or similar to compare different counties (using `falls_within` field in `crimedata` table) in terms of total number of crime records in each of them.

9.2 Functional Testing

Write functional test cases by considering requirements in each of the tasks as part of this coursework. Prepare a functional testing report using the given template in appendix B and for each of the test cases provide indication of whether each of them passed/failed.

9.3 Sorting results

Write implementation that builds on the functionality of task 3. In addition to requirements in tasks 4 & 5, this implementation will also allow sorting the results based on different columns of the `crimedata` table.

10.4 Something else…

Design and implement your own feature(s) for the search engine, adding code and/or classes where needed and documenting them appropriately.

The Report

Part of your submission for this coursework is a short written report that describes how your code solution for each task works. The report should be presented professionally and conform to the guidance below.

The report should be structured as follows:

- A single cover page that incudes your name and UB number
- One section for each task, comprised of **at most two paragraphs of text** that concisely describes how your code for each task works. The text should be focussed on explaining how your code works, and there **should be between 100 and 300 words** describing your solution for each task. **300 is a maximum** and some tasks should use less than 300 words if they are simpler to explain.
- A final section describing any tasks that were not completed, and any partial solutions that you developed but did not submit.

The report must be submitted through Blackboard, using the TurnItIn link in the Assessment area. There are two submission links, a draft submission that you should use to check your final report for any accidental similarity to existing sources (correcting them where needed) and **a final submission link where you must submit your final report**. You can submit to each link only once, so you should use them only for your final report version.

Submission and deadlines

You must submit the software through the CWSUBMIT folders for this module. You must submit your final version before the deadline, and **submit a version that works** i.e. that compiles and is testable by the module staff by running the project through Netbeans, demonstrating the completed tasks. You should have a folder set up in the directory:

L:\cwsubmit\2017-18\COS4017-B\

Your folder will be named the same as your e-mail address. You must submit the ZIP file containing your software solution that Netbeans creates when you Export it from the File menu as a ZIP file.

Please export your solution when you complete each task, keeping a backup of every version stored in your home folders. This will avoid any problems with "lost" coursework and allow you to submit the last version that compiles and runs. **It is vital that your final submission runs successfully when module staff test it by running the complete project.**

You must submit the written report for this assignment through the TurnItIn "final" submission link on Blackboard. You are encouraged to check your final report using the draft submission – please bear in mind that **the processing of the draft takes time** and you will not get a report on similarity back in time if you submit very close to the deadline.

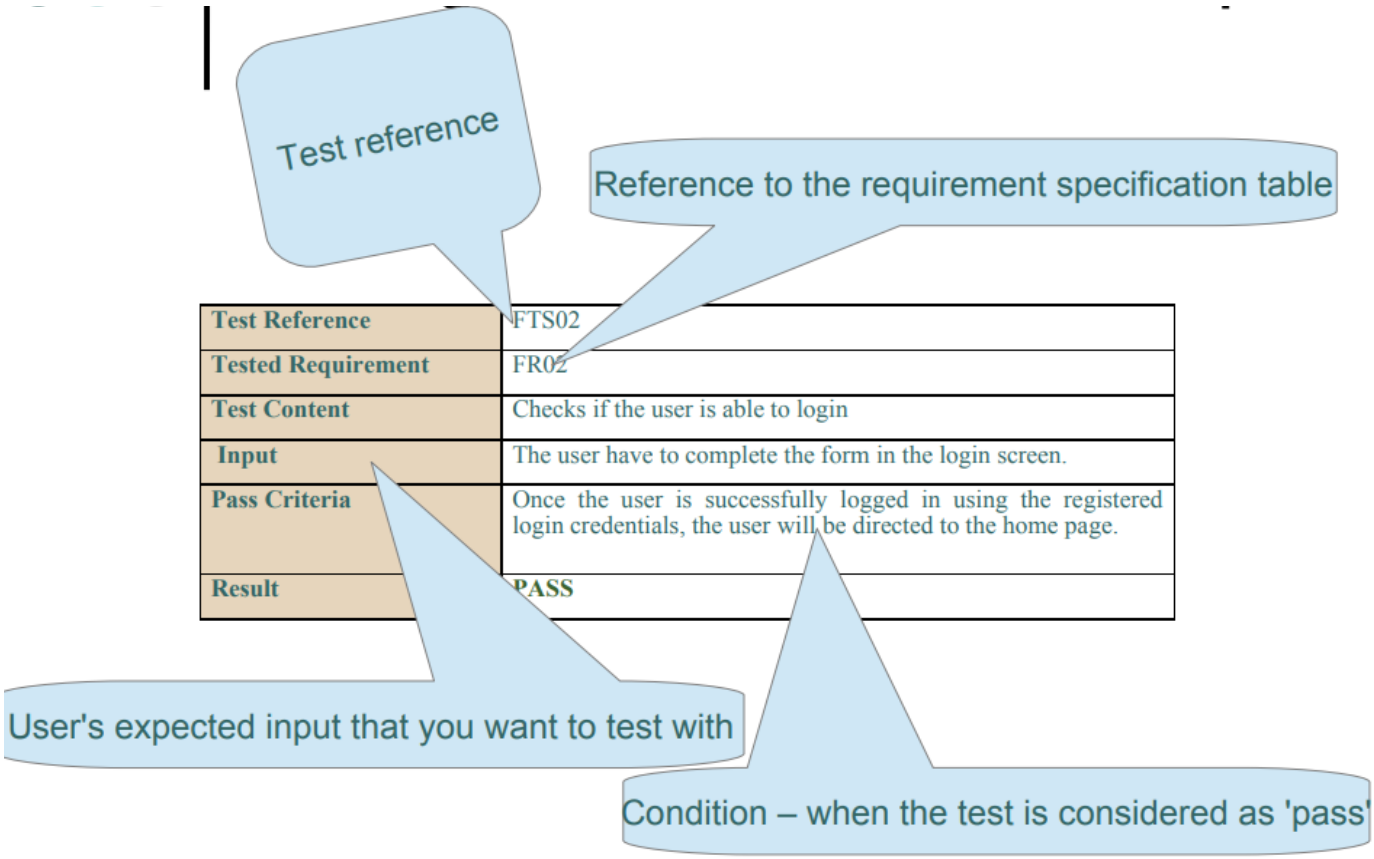
The deadline for submitting your software and report is 3pm Friday 11th May 2018. It will not be possible to submit software after this time, and any reports

submitted after the deadline will result in a mark of zero for your entire coursework.

Appendix A: Crime data table fields explanation

Column	Explanation
Crime_Id	Id of the crime record
Month	The month the crime occurred
Reported_By	The constabulary/force that reported the crime. Note: This can be different from the constabulary that the crime falls under.
Falls_within	The constabulary/force the crime falls within.
Longitude	Anonymised Longitude of the location of the crime scene.
Latitude	Anonymised Latitude of the location of the crime scene.
Location	Some indication of approximate location.
LSOA code & LSOA name	References to the Lower Layer Super Output Area that the anonymised point falls into, according to the LSOA boundaries provided by the Office for National Statistics.
Last outcome category	A reference to whichever of the outcomes associated with the crime occurred most recently. For example, this crime's 'Last outcome category' would be 'Formal action is not in the public interest'.
Context	A field provided for forces to provide additional human-readable data about individual crimes. Currently, for newly added CSVs, this is always empty.
Crime type	One of the crime types listed in the Police.UK FAQ.

Appendix B: Functionality testing template



Test Reference	FTS02
Tested Requirement	FR02
Test Content	Checks if the user is able to login
Input	The user have to complete the form in the login screen.
Pass Criteria	Once the user is successfully logged in using the registered login credentials, the user will be directed to the home page.
Result	**PASS**