Abdul Muhaymin Abdul Hafiz

COMP0004 Coursework Report

Summary of features

The Java web application I have developed implements all 10 of the requirements listed in the coursework specification. It is a Maven project, which is used here to help manage the project dependencies. The web application first loads in the patient records from a CSV file into a custom DataFrame data structure. The user can then update, add, or remove records using the website interface. The CSV file which the patient records were initially loaded from gets updated to reflect the changes as they are made. The user can also search through the patient records by choosing from a list of fields to search and then entering their search string. Additionally, there is an option to generate a JSON file containing all of the patient records. Furthermore, a variety of statistics and graphs based on the patient data can be viewed, such as the oldest patient, the age distribution of the patients, the most common city etc.

Design description

The web application follows the Model-View-Controller design pattern. The controller in this case is made up of several web servlet classes. These web servlets communicate with the same singular model object, and also communicate with the various JSP files which comprise the view - the servlets act as an intermediary between the model and JSP files. There are 6 servlets in total which are the HomeServlet, PatientListServlet, PatientServlet, PatientAddServlet, SearchServlet, and StatisticsServlet, each of which are mapped to the different pages that the user can go to. There is also a corresponding JSP file for each one of these servlets.

The Model class itself calls on the other classes within the Model package to perform various tasks. For example it uses the DataFrame class to handle the storage and updating of data, it uses the CSVWriter and JSONWriter classes (which implement the DataWriter interface) to handle file writing, and it creates and returns lists of Patient objects which represent a patient record. By doing this, tasks can be split up, and the code is made more readable and maintainable. In the web application, the servlets only directly call the methods in the Model class rather than any of the methods in the other classes in the Model package, thus creating a sense of abstraction around the model. This prevents any added complexities and keeps things cleaner.

With regards to the view, each of the JSP files import and build upon the same ‘base tag’. This base tag contains the common HTML content present across all of the web pages, which each JSP file adds their own unique content body to. This means that rather than copy and pasting the HTML for the header and footer for example into every JSP file, it only needs to be written and maintained in the base tag which is then imported wherever needed. This makes it easier to make changes across the whole website, and ensures consistency. JSTL is also used within the JSP files instead of JSP code snippets as this is more readable and said to be better practice.

Evaluation

Overall, I believe the quality of my web application is of a good standard. It satisfies all of the requirements listed in the coursework specification to a good level in my opinion. Furthermore, effort was put into styling the website using Bootstrap to make it look nicer, and the website is designed in such a way to make it user friendly. The web application also continues to perform well even with a large number of patient records in the CSV file, which indicates that my model implementation is at least somewhat efficient. Each of the methods in the model class are also short and are easy to understand without requiring a lot of additional comments, if any. The directories and structure of the project is also designed in such a way to make it easy to add new pages and features, making the web application maintainable.

Critically speaking, my web application is by no means perfect. Some of the main caveats with the website are that the home page is somewhat lacking, and has a very plain design. The forms that the user submits when updating or adding patient records should have more validation checks to ensure that the data in the different fields is of the correct respective format. And, it would have been nice to have the option to upload and load in a CSV file through the website interface. But given the particular coursework requirements, and the expected scope of the project, I would say that my web application was developed to a good standard.