Brandon Tenorio

Fall 2021

CMSC204

Gary C. Thai

Project 4: Course Database

While working on this project, I was able to solidify my understanding of hash tables and how to manipulate them and use them more effectively, as well as how to create a database. Some of the extra additions that my project consists of include a set of 5 Runtime exceptions, a constructor that validates input data, and a method in the manager class that validates the data being read from it. For this project I was finally able to use a lambda function, which I’ve been dying to use since I first learned about them back in August of this year. Using online resources such as baeldung.com and geeksforgeeks.com, I was able to confidently use a lambda function effectively and efficiently. The design of this project followed from the Word document that was released. I made sure to validate every input regardless of where it comes from. Whether the data is being inputted from a file being read by the program, or it is being manually inputted, I made sure that both input methods check that the data is properly formatted to avoid adding bad elements to the database. As previously mentioned, I created 5 runtime exceptions: InvalidCourseIDException, InvalidCRNException, InvalidCreditAmountException, InvalidRoomNumberException, and InvalidCourseFormatException. Each of these exceptions are meant to be thrown whenever an attempt to input bad data to the database occurs. The method readFile(File input) in the CourseDBManager class only throws InvalidCourseFormatException; the parameterized constructor that isn’t meant for testing in CourseDBElement throws the remaining exceptions.

The hardest part of this project for me was the readFile method in the CourseDBManager class. It was hard because it took me a long time to come up with a good approach to reading and storing the data from a file in a buffer and then validating all of it. Thankfully I was able to come up with an algorithm that ensures that no bad data is inputted into the database without throwing a runtime exception.

One important thing to note is that if one were to input a non-numeric character as the CRN for a course or as the credits for a course, a NumberFormatException is thrown. For instance, inputting “comp sci” in the box meant for a course’s CRN raises a NumberFormatException. This is something that I didn’t spend time trying to fix since rationally one is meant to input a series of 5 digits for the CRN and credit amount, and not input any non-numeric characters. Since Java already throws what seems to be an appropriate exception, namely NumberFormatException, I didn’t think it necessary to account for when the user tries to input a non-numeric character into a data field that only accepts integers.

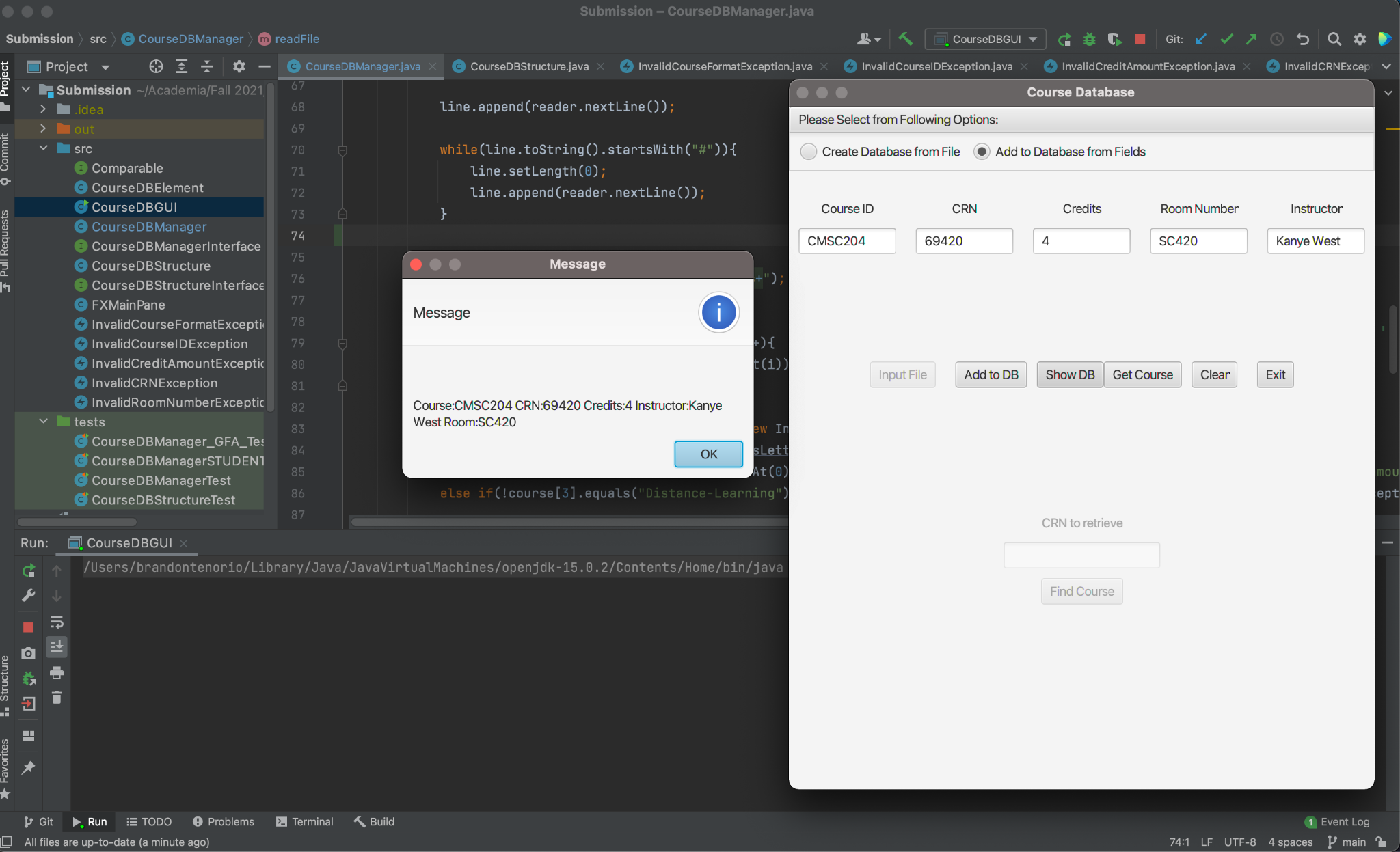
The algorithm for my readFile method is simple. It reads an entire text file line by line, skipping any and all lines that begin with a hashtag, and saving those that don’t into a StringBuilder object called line. Using Regex, I then split the line on every space, taking into account that there may possibly be more than one space between any two sequences of characters, and I save every sequence of characters in a String array called course, whose size is greater than or equal to 4. The idea behind this is that, sinc every line in the file contains a course in the format of courseID, CRN, credits, room number, and instructor name, index locations 0 to 3 (inclusive) in the array will still the courseID, the CRN, the credit amount, and the room number respectively, and the instructor’s name will be saved in a StringBuilder object called name. Moving forward, the algorithm proceeds to perform input validation on the existing data that was saved to the String array courses, throwing an exception if the data is not valid, and continuing to add the courses to the database if they are valid.

One minor problem that I encountered arrived when trying to retrieve a CRN from the database that does not exist. Since the method that performs this action, namely CourseDBStructure.get(int CRN), is meant to return a CDBElement object, I couldn’t find a way to return something more descriptive than null when this method fails to find the element corresponding to the CRN passed in. I would’ve liked to have thrown a NoSuchElementException or something of that sort, but doing so would raise an error in my IDE saying that there was a NoSuchElementException with no catch block, thus I left it as return null.

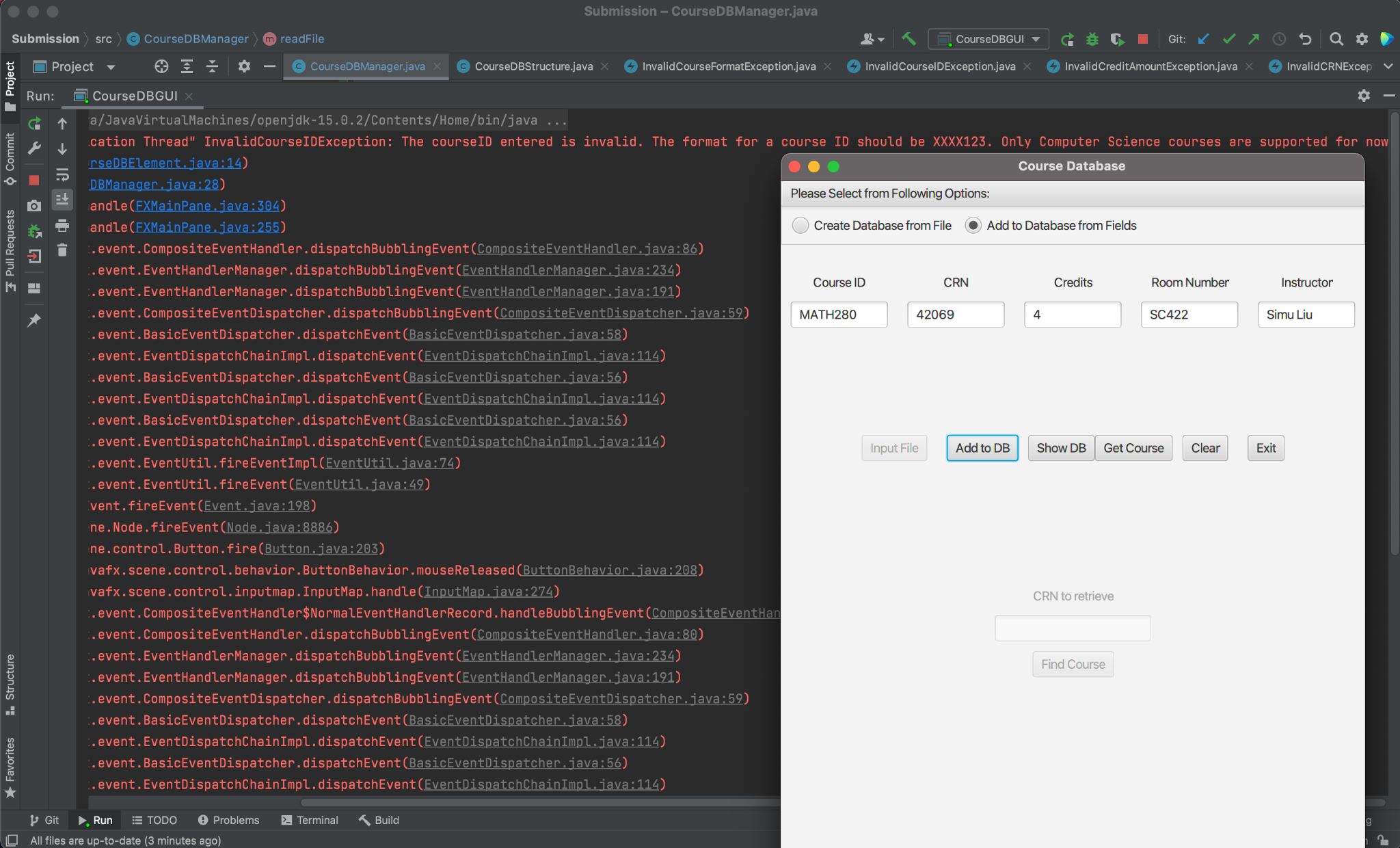
Tests

| Input | Expected output | Actual output | Pass? (y/n) |
| --- | --- | --- | --- |
| **courseID:**  CMSC204  **CRN:**  69420  **Credits:**  4  **Room number:** SC420  **Instructor name:**  Kanye West | **courseID:**  CMSC204  **CRN:**  69420  **Credits:**  4  **Room number:** SC420  **Instructor name:**  Kanye West | **courseID:**  CMSC204  **CRN:**  69420  **Credits:**  4  **Room number:** SC420  **Instructor name:**  Kanye West | Yes |
| **courseID:**  MATH204  **CRN:**  42069  **Credits:**  4  **Room number:** SC422  **Instructor name:**  Simu Liu | InvalidCourseIDException (only CMSC classes are supported for now) | Exception in thread "JavaFX Application Thread" InvalidCourseIDException: The courseID entered is invalid. The format for a course ID should be XXXX123. Only Computer Science courses are supported for now | Yes |
| **courseID:**  CMSC204  **CRN:**  2  **Credits:**  4  **Room number:** SC420  **Instructor name:**  Kim Kardashian | InvalidCRNException(CRN must be 5 nonnegative digits) | Exception in thread "JavaFX Application Thread" InvalidCRNException: The CRN for this course must be a unique positive number of 5 digits | Yes |
| **courseID:**  CMSC204  **CRN:**  12345  **Credits:**  69  **Room number:** SC420  **Instructor name:**  Barack Obama | InvalidCreditAmountException (amount should be a nonnegative integer between 1 and 4 inclusive) | Exception in thread "JavaFX Application Thread" InvalidCreditAmountException: The amount of credits for this course must be between 1 and 4 (inclusive). Please re-enter the information with the proper amount of credits | Yes |
| **courseID:**  CMSC204  **CRN:**  54321  **Credits:**  4  **Room number:** lolXD  **Instructor name:**  Michelle Obama | InvalidRoomNumberException (room number must be in the form XX123) | Exception in thread "JavaFX Application Thread" InvalidRoomNumberException: The room number you entered is invalid. Expected format is XX123 for in-person courses, and Distance-Learning for online courses | Yes |

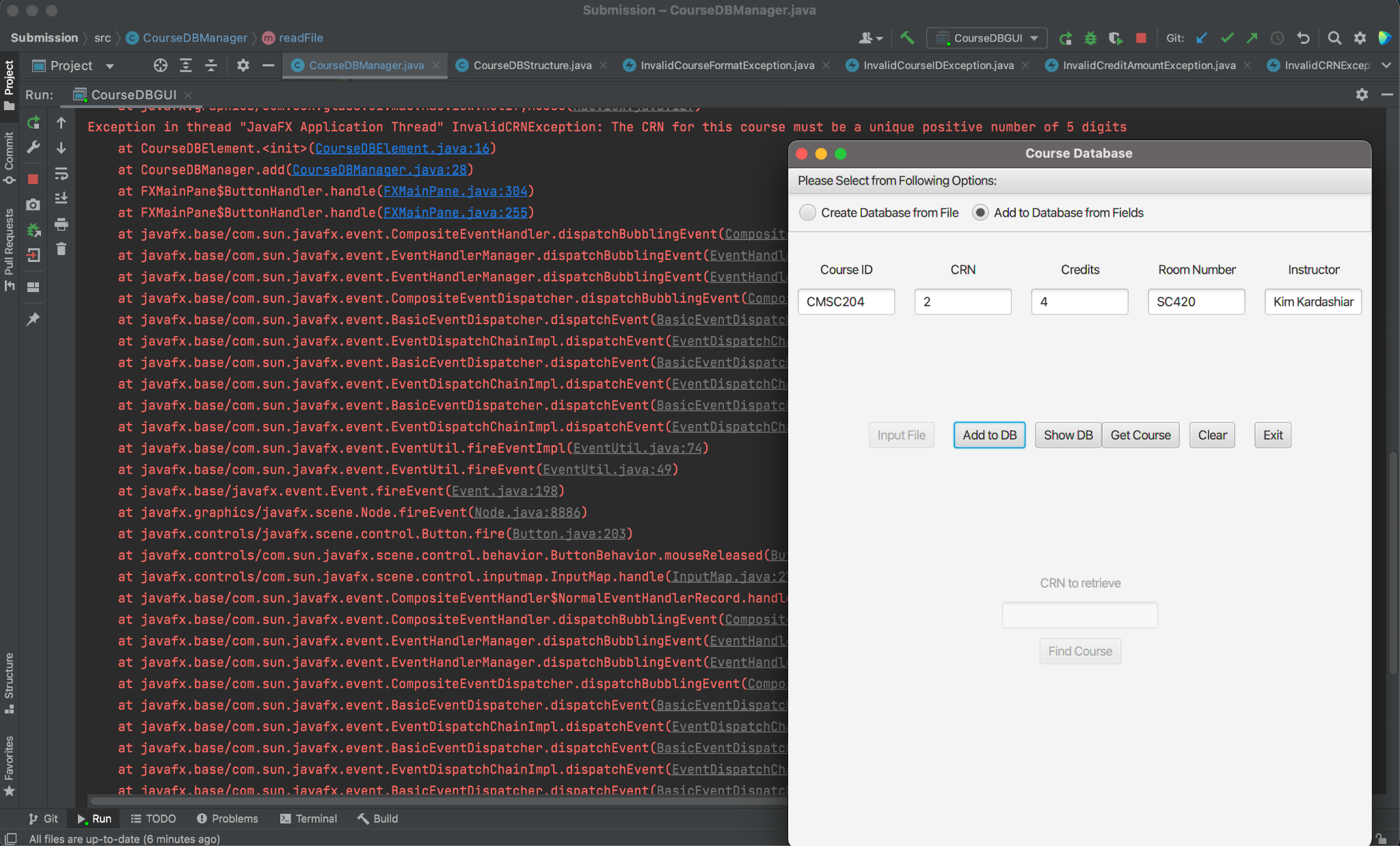
Test 1



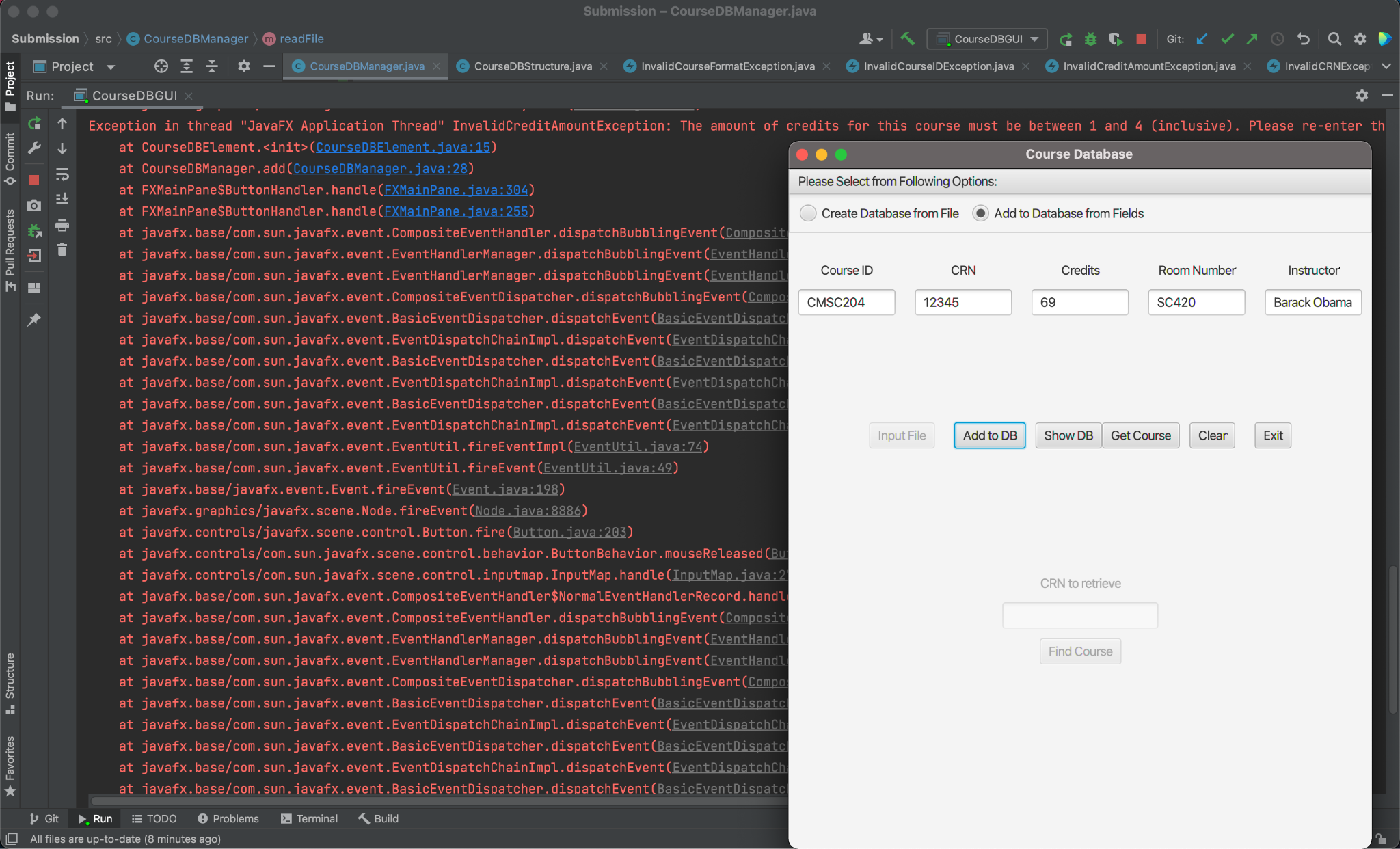
Test 2



Test 3



Test 4



Test 5

