Week 2 – Assignment

Core Java + Features JAVA 8, 9, 10, 11 + Inbuilt Packages	**1. Build a Date-Time Calculator, with menu options; Inputs to be taken from the console and output to be displayed to the console;** **- Add, Subtract between two dates and express the output in days, dates, weeks, months; (Involves String to Dates conversion);** **- Add, Subtract 'n' Days, Months, Weeks to the given date and derive the final date** **- Determine the Day of the Week for a given date** **- Determine the Week number for a given a date** **- Translate natural language phrases like "Today, Tomorrow, Day-after-tomorrow, yesterday, Day-before-yesterday, Last week, Previous week, Next week, Next month, Next Year, Last month, Last year, Month after, Month before, n weeks from now, n days from now, n months from now, n years from now, n days earlier, n weeks earlier, n months earlier, n years earlier etc."** **2. Extend the Date-Time Calculator to capture and display history of the calculations in the session in Memory** **3. Extend the Date-Time Calculator to support multiple languages (Ex: English, Hindi, French etc.)** **4. Extend the Date-Time Calculator to capture, persist (file storage) and display history for the last 100 sessions (the display needs to be paused & continued) - Using Serialization, Custom Data Structures & text, binary File formats, CSV, XML, YAML, JSON** **5. Extend the Date-Time Calculator to support bulk processing & operations - A file of 100,000 operations;** **6. Convert the OOPS based design to Functional programming using lambdas** **7. Automate the data-creation part of generating the file with 100,000 operations** *** Apply/Practice TDD, BDD while doing the assignment; Outputs should include unit test report;**

	>> Ensure the session storage captures info. Using different data types --> Strings, Dates, and Integers/Longs etc. **>> Extensions and features to be built without changing existing code as much as possible** **>> Those who demonstrate Innovative solutions (ex: not using 'if/switch' conditions for functions), Engg. practices, Optimized code will be awarded brownie points**
Postgresql	- Implement the Date-Time Calculator as db operations, stored-procs & queries - Write queries to process the file with 100,000 operations and store the outputs back in a file - using DB - Build a dashboard from the Session data to indicate: - Overall Statistics across all sessions, per operation - Sessions wise statistics, per operation - Operations wise statistics per Session & across all sessions
Hibernate basics, ORM framework, CRUD examples	- Extend the Date-Time Calculator to capture, persist and on-demand retrieval, search the history of last 100 sessions using PG
JSP, Servlet, JSLT + Hibernate	- Expose the Date-Time Calculator as a Servlet Application with UI created using JSP
Rest & Swagger	- Define the Service contracts using Swagger - defining the inputs & outputs; - Convert all the menu options into Service End-points, including the ones for retrieval of data & statistics
Spring MVC	- Expose the Date-Time Calculator as a Spring Boot service with UI created using RESTful services
Microservices with Spring Boot	- Expose the Date-Time Calculator as a Spring Boot service with UI created using RESTful services