Загальні вимоги до реалізації

1. На основі сутностей предметної області створити класи, які їм відповідають.
2. Класи і методи повинні мати назви, що відображають їх функціональність, і повинні бути рознесені по пакетам.
3. Оформлення коду має відповідати Java Code Convention.
4. Інформацію щодо предметної області зберігати у реляційній базі даних (в якості СУБД рекомендується використовувати MySQL або PostgreSQL).
5. Застосунок має підтримувати роботу з кирилицею (бути багатомовним), в тому числі при зберіганні інформації в базі даних:
    a. повинна бути можливість перемикання мови інтерфейсу;
    b. повинна бути підтримка введення, виведення і зберігання інформації (в базі даних), записаної на різних мовах;
    c. в якості мов обрати мінімум дві: одна на основі кирилиці (українська або російська), інша на основі латиниці (англійська);
    d. дати повинні бути реалізовані через DataTime бібліотеку (Java8).
6. Реалізувати захист від повторної відправки даних на сервер при оновленні сторінки (реалізувати PRG).
7. У застосунку повинні бути реалізовані аутентифікація і авторизація, розмежування прав доступу користувачів системи до компонентів програми. Шифрування паролів заохочується.
8. Впровадити у проект журнал подій із використанням бібліотеки log4j.
9. Код повинен містити коментарі документації (всі класи верхнього рівня, нетривіальні методи і конструктори).
10. Застосунок має бути покритим модульними тестами (мінімальний відсоток покриття 40%). Написання інтеграційних тестів заохочуються. Використання Mockito заохочується.
11. Реалізувати механізм пагінації сторінок з даними.
12. Всі поля введення повинні бути із валідацією даних.
13. Застосунок має коректно реагувати на помилки та виключні ситуації різного роду (кінцевий користувач не повинен бачити stack trace на стороні клієнта).
14. Самостійне розширення постановки задачі по функціональності заохочується! (додавання капчі, формування звітів у різних форматах, тощо)
15. Використання HTML, CSS, JS фреймворків для інтерфейсу користувача (Bootstrap, Materialize, ін.) заохочується!
16. Розробка проектів за допомогою Git заохочується.

До сервлетного проекту:

17. Для доступу до даних використовувати JDBC API із застосуванням готового або ж розробленого самостійно пулу з'єднань.
    НЕ допускається використання ORM фреймворків
18. Архітектура застосунка повинна відповідати шаблону MVC.
    НЕ допускається використання MVC-фреймворків
19. При реалізації бізнес-логіки необхідно використовувати шаблони проектування: Команда, Стратегія, Фабрика, Будівельник, Сінглтон, Фронт-контролер, Спостерігач, Адаптер та ін.
    Використання шаблонів повинно бути обґрунтованим

20. Використовуючи сервлети і JSP, реалізувати функціональність, наведену в постановці завдання.
21. Використовувати Apache Tomcat у якості контейнера сервлетів.
22. На сторінках JSP застосовувати теги з бібліотеки JSTL та розроблені власні теги (мінімум: один тег custom tag library і один тег tag file).
23. При розробці використовувати сесії, фільтри, слухачі.

До спрингового проекту:

24. Застосунок повинен буду структурованим за архітектурою MVC та Spring Boot.
    Дозволено використання Project Lombok.
25. Використання Spring Resource Bundle заохочується.
26. Повинно бути застосована Spring Authorization.
27. Для доступу до даних використовувати JPA (Spring Data та/або Hibernate).
28. Обробка виключних ситуацій за допомогою Exception Handling with Spring for REST API заохочується.
29. Використання ThymeLeaf заохочується.
30. Використання додаткових фреймворків (Swager та інші) заохочується.