Промежуточная аттестация Модуль 3 «Клиент-серверные приложения на Java. Веб-разработка»

Тема проекта (написание приложения) выбирается совместно с наставником. В качестве промежуточной аттестации разрабатывается основная структура итогового приложения.

Перечень примерных тем для проектной работы:

1) Библиотека ресурсов для изучения языка программирования Java.

Реализация сервиса для помощи в изучении новичкам-программистам, его особенностей и характеристик.

2) Интернет-магазин «Шиномонтаж».

Создание веб-ресурса для реализации работы шиномонтажа, с возможностью выбора и оформления онлайн-заказа.

3) Приложение «Пиццерия».

Создание веб-ресурса для работы с покупателями пиццерии, с возможностью выбора и оформления онлайн-заказа.

4) Онлайн-кинотеатр.

Реализация сервиса для ведения аккаунта пользователя онлайн-кинотеатра с возможностью подбора фильмов для простмотра.

5) Онлайн-кинотеатр.

Реализация сервиса для ведения аккаунта администратора кинотеатра с возможностью ведения данных по посетителям.

6) Приложение «Регистратура больницы».

Сервис для больничной регистрации и учёта пациентов. Приложение создается для административного работника больницы с возможностью работать с данными пациента от врача.

7) Приложение «Помощь врачу».

Сервис для заполнения данных по пациенту. Приложение создается для врача поликлиники с возможностью работать с кодификатором болезней.

8) Приложение «Сервис записи на тренировки в спортзал».

Сервис для ведения базы данных тренера спортзалов. Тренер ведет базу спортсменов(игроков, клиентов).

9) Приложение «Проект ЖКХ».

Создать удобный инструмент для учета данных ЖКХ, таких как показатели счетчиков и ежемесячная оплата.

10) Приложение «Статистика мини-игры»

Сервис, в котором пользователь ведет данные по играм, противникам, их характеристикам и результатам сражения.

11) Приложение «Брокерский счет для клиентов банка»

Создать сервис на примере брокерского сервиса для ведения данных счетов брокеров.

12) Другая тема по согласованию с наставником. Тема может быть связана с:

- ведением данных интернет-магазина;
- полезной базы для домашнего использования;
- данных приложения в сфере работы и/или учёбы.

Формулировка задания:

Реализовать Spring Boot приложение по выбранной теме. Зависимости разметить через Maven. Структура папок - шаблоны MVC и Repository.

Для выполнения задания реализовать следующие этапы:

1. Проектирование базы данных для выбранной темы;

2. Создание Spring Boot Maven проекта с загрузкой необходимых библиотек;

3. Разметка структуры приложения. Создание набора папок в соответствии с паттерном MVC, Repository;

4. Реализация слоя работы с данными (репозитория) с применением технологии JPA в соответствии со спроектированной структурой БД. Для работы с моделями данных (классами) допустимо использовать Lombok;

5. Реализация слоя сервисов. Минимально должны быть добавлены

CRUD операции для работы с базой данных по варианту. Дополнительный функционал желательно вынести в папку utils;

6. Реализация веб-контроллера приложения. В веб-контроллере должны быть реализованы 1-3 страницы для работы с приложением. Минимально 1 страница API. Принимать файлы в формате JSON. Удаление сделать через Soft Delete флаг.

7. Дополнительно. Создать слой DTO для работы с приложением. Отделить работу с DTO и с таблицами базы данных;

8. Проверить и дописать настройки приложения в файле настроек Spring Boot;

9. Дополнительно. Настройка генерации базы данных. Создание файла миграций БД и добавление его в resources;

10. Протестировать основной функционал приложения через unit-тесты. Использовать Spring Boot аннотации.

Программа реализуется в отдельной ветке git attestation/attestation03. Допустимо использовать для приложения структуру базы данных из ветки git attestation/attestation02.

При сохранении состояния программы (коммиты) пишется сообщение с описанием хода работы по задаче.

В корне папки с программой должен быть файл .gitignore.

Программа локально сохраняется в коммиты git (каждый шаг желательно сохранять отдельно) и публикуется в репозиторий GitHub на проверку.

Планируемый результат:

1. Ссылка на программу в репозитории gitlab;
2. Отчёт со скриншотами выполнения задач - постановка задачи, код задачи и результат.
3. Отдельно сохранить скриншоты по инструментарию и файлам настроек.

Перечень инструментов, необходимых для реализации проектной деятельности:

Персональный компьютер, JDK 17 (либо OpenJDK 17), Intellij Idea для разработки на Java, GIT, Tortoise GIT, JUnit, Apache Maven, PostgreSQL, Docker, Spring Boot 2 и выше, Swagger UI, Postman и др.

Показатели и критерии оценивания:

1. База данных по теме спроектирована и содержит не менее 2-4 таблиц.
2. Создан Spring Boot Maven проект с загрузкой необходимых библиотек;
3. Структура проекта создана в соответствии с паттернами MVC и Repository;
4. Применена технология JPA (Hibernate) на слое работы с данными (repository).
5. В проекте использована дополнительная библиотека Lombok.
6. Слой сервисов содержит CRUD операции для работы с базой данных по варианту.
7. Дополнительный функционал вынесен в папку utils.
8. В веб-контроллере приложения реализованы 1-3 страницы для работы с приложением. Минимально 1 страница API.
9. Приложение принимает файлы в формате JSON.
10. Удаление данных реализовано через Soft Delete флаг.
11. Создан слой DTO для работы с приложением.
12. Настройки приложения все прописаны в файле настроек Spring Boot;
13. Создан файл миграций БД и добавлен в resources проекта;
14. Добавлена автоматическая генерация миграций БД;
15. Проект покрыт unit-тестами, учитывается процент покрытия и наличие Spring Boot аннотаций.

Шкала оценивания:

Оценка 3 / Удовлетворительно / 0.5 - 0.75:

- Реализован проект на основе структуры БД из Промежуточной аттестации 2.

- Создан Spring Boot Maven проект с загрузкой необходимых библиотек. Maven может не содержать настроек для тестов, ресурсов и упаковки проекта.

- Структура проекта создана в соответствии с требованием Maven. Есть неточности по паттерну MVC, Repository.

- Реализована работа с данными. Можно без JPA (Hibernate).

- Слой сервисов содержит минимальное число CRUD операций для работы с базой данных Промежуточной аттестации 2.

- В веб-контроллере приложения реализована 1 страница для работы с приложением.

- Настройки приложения прописаны в файле настроек Spring Boot.

- Проект минимально покрыт unit-тестами (40-50%), учитывается процент покрытия и наличие Spring Boot аннотаций.

- Данные сохранить (commit) в git репозиторий.

Оценка 4 / Хорошо / 0.75 - 0.9:

- Реализованы требования на оценку 3.

- База данных по теме спроектирована и содержит не менее 2-4 таблиц. Это может быть БД из Промежуточной аттестации 2.

- Создан Spring Boot Maven проект с загрузкой необходимых библиотек. Maven может не содержать настроек для тестов, ресурсов проекта.

- Реализована работа с данными. Можно без JPA (Hibernate) или минимально.

- Слой сервисов содержит минимальное число CRUD операции для работы с базой данных по варианту. БД может быть из промежуточной аттестации 2.

- В веб-контроллере приложения реализованы 2-3 страницы для работы с

приложением. Минимально 2 страницы API.

- Приложение принимает файлы в формате JSON.

- Настройки приложения все прописаны в файле настроек Spring Boot;

- Создан файл миграций БД и добавлен в resources проекта;

- Проект минимально покрыт unit-тестами (40-60%), учитывается процент покрытия и наличие Spring Boot аннотаций.

- Данные сохранить (commit) в git репозиторий в ветке git attestation/attestation03.

- Создан pull request в ветку main (master).

Оценка 5 / Отлично / 0.9 - 1.0:

- Реализованы требования на оценку 4.

- Структура проекта создана в соответствии с паттернами MVC и Repository;

- Применена технология JPA (Hibernate) на слое работы с данными.

- В проекте использована дополнительная библиотека Lombok.

- Слой сервисов содержит полный набор CRUD операции для работы с таблицами базы данных по варианту.

- Дополнительный функционал вынесен в папку utils.

- В веб-контроллере приложения реализованы 2-3 страницы для работы с приложением.

- Приложение принимает файлы в формате JSON.

- Удаление данных реализовано через Soft Delete флаг.

- Создан слой DTO для работы с приложением.

- Добавлена автоматическая генерация миграций БД;

- Проект покрыт unit-тестами (60 и более %), учитывается процент покрытия и наличие Spring Boot аннотаций.

- Данные сохранить (commit) в git репозиторий в ветке git attestation/attestation03.

- Создан pull request в ветку main (master).