Презентационный проект использования платформы CUBA.

Проект демонстрирует два способа реализации двух возможных архитектур при использовании платформы:

1. Традиционная 3-х уровневая архитектура с «богатым» наполнение среднего слоя бизнес‑логики и развитой системой бизнес-сервисов;
2. Также часто используемая 2-х уровневая архитектура, часто реализуемая как 1.5, а иногда как 2.5-х уровневая архитектура с «прозрачным» уровнем бизнес‑логики, когда значительную часть её берёт на себя верхний Web-уровень.

# Реализация 3-х уровневой архитектуры

## Web-уровень

На Web-уровне в данном проекте реализованы возможности обращения к следующим ресурсам:

* удалённым сервисам по HTTP GET-запросу,
* локальным ресурсам с префиксами file: classpath: system:

## Сервисы бизнес‑логики

Описание RESTful сервисов, их названия параметры и их тип приведены в файле

<root>\modules\web\src\ru\zrv\tazacom\rest-services.xml

Само название этого файла и расположение его можно задать в качестве свойства

cuba.rest.servicesConfig

в файле свойств модуля в файле:

<root>\modules\web\src\ru\zrv\tazacom\web-app.properties

В качестве примера реализован RESTful‑сервис импорта списка книг из внешнего источника, где URL‑адрес передаётся в качестве параметра.

Реализацию сервиса см. в классе

ru.zrv.tazacom.service.ImportServiceBean.java

Файл класса находится в файле

C:\Projects\CUBA\Tazacom\modules\core\src\ru\zrv\tazacom\service\ImportServiceBean.java

В качестве тестового адреса сервиса можно использовать в окне браузера адрес:

http://localhost:8080/app/rest/v2/services/tazacom\_ImportService/doImportRest?url=file:BookList.Example.json

Сервис берёт на себя функцию запроса данных из внешнего источника, парсинг данных и запись импортированных данных в базу данных.

## Уровень базы данных

Для работы с базой данных используется паттерн DAO, предоставляющий функции CRUD, а также три дополнительных метода:

1. Проверка уникальности строки в таблице. Например, для таблицы книг проверяется уникальность сочетания
   * «Фамилия, имя и отчество автора»
   * Название книги, редакция и год издания
2. Подсчёт количества строк в таблице по определённым критериям;
3. Вставка новой строки, если таковой ещё нет, либо изменение существующей.

Для презентационных целей продемонстрировано использование хорошо знакомого EntityManager’a.

Другая, более интересная, возможность работы с базой данных – внедрение и сервиса DataService. Пример его использования смотри в реализации 2-х уровневой архитектуры.

# Реализация 1.5‑ уровневой архитектуры

Такое название архитектуре дано потому, что формально, средний уровень присутствует, но он очень тонкий, почти «прозрачный».

## Сервисы бизнес‑логики

В данном проекте сервис импорта данных реализован как примитивный посредник, который:

* Получает в качестве параметра URL‑адрес,
* читает JSON‑данные по указанному адресу,
* проверяет, нет ли ошибок при передаче запрашиваемых данных, формирует код возврата и возвращает его вместе с самими данными на Web‑уровень.

## Web-уровень

При такой архитектуре этот уровень берёт на себя львиную долю задач:

* Это и контроллеры переднего плана/паттерн Facade, которые берут на себя функцию координации взаимодействия с пользователем,
* Большую часть бизнес логики и анализа входящей и подготовки исходящей информации,
* Взаимодействие с базой данных.

Реализацию такого контроллера см. в классе

ru.zrv.tazacom.web.importing. ImportingUrlDialog.java

в файле модуля Web

<root>\modules\web\src\ru\zrv\tazacom\web\importing\ImportingUrlDialog.java

### Взаимодействие с базой данных

Специально для такой архитектуры предусмотрен сервис DataService, который можно внедрить в класс аннотациями:

@Inject

ImportService importService;