**2.1. Анализ используемых данных, их хранение и составление модели данных.(3с)**

Основной целью данной главы понять, что такое тип данных , какие типы данных используются в различных проектах.

Сначала нам надо определиться какие типы данные бывают.

**Тип данных** — понятие, используемое в языках программирования для характеристики данных (переменных, массивов и др.) в программах. Указание конкретного типа определяет множество значений , которые могут принимать данные этого типа, и набор операция которые можно к ним применять. Основная единица данных - переменная. Каждая переменная, используемая в программе, должна иметь тип. Способы указания типа определяются правилами (синтаксисом) языка.

Выпишем наиболее распространённые типы:

**Логический тип** - логические, или булевы значения (по фамилии их изобретателя — Буля), могут иметь лишь одно из двух состояний — «истина» или «ложь».

**Целочисленные типы** - целочисленные типы содержат в себе значения, интерпретируемые как числа (знаковые и беззнаковые).

**Числа с плавающей точкой** - используются для представления вещественных (не обязательно целых) чисел.

**Строковые типы** - Последовательность символов, которая рассматривается как единое целое в контексте переменной.

Как их можно использовать

Для составлении модели нам нужно проанализировать информацию

которая будет хранится в портфолио.

Начнем с личной информации по школьнику его имя и фамилия

- Имя

- Фамилия

- Дата Рождения

- Школа

- Хобби

- Достижения

- Исследования

- Школьная жизнь

- Место рождения

Распишем это в качестве таблицы:

| Имя поля | Тип данных | Java Type |
| --- | --- | --- |
| Имя | строковой | String |
| Фамилия | строковой | String |
| Дата Рождения | целочисленный | Date |
| Место рождения | строковой | String |
| Школа | строковой | String |
| Хобби | строковой | String |
| Достижения | строковой | String |
| Исследования | строковой | String |
| Школьная жизнь | строковой | String |

Структура Профайла

class UserProfile {

String firstName;

String lastName;

Date birthday;

String school;

String hobbies;

String achievements;

String creation;

String studies;

String schoolLife;

String homeland;

};

Хранение профайла

--Какие способы хранения информации существуют

В настоящее время особое значение получило хранение информации в виде последовательностей двоичных символов. Для реализации этих методов используются разнообразные запоминающие устройства. Они являются центральным звеном систем хранения информации.

Для хранения профайлы мы выберем реляционную базу данных с упрощенным использованием.

HSQLDB полностью написана на [Java](https://ru.wikipedia.org/wiki/Java) и отличается небольшим размером (размер около 1100 кБ для версии 2.0). Может использоваться и как отдельный сервер с поддержкой сетевых соединений по [JDBC](https://ru.wikipedia.org/wiki/JDBC), и в виде библиотеки для использования непосредственно в коде программы.

Реляционная база данных – это набор данных с предопределенными связями между ними. Эти данные организованны в виде набора таблиц, состоящих из столбцов и строк. В таблицах хранится информация об объектах, представленных в базе данных. В каждом столбце таблицы хранится определенный тип данных, в каждой ячейке – значение атрибута. Каждая стока таблицы представляет собой набор связанных значений, относящихся к одному объекту или сущности. Каждая строка в таблице может быть помечена уникальным идентификатором, называемым первичным ключом, а строки из нескольких таблиц могут быть связаны с помощью внешних ключей. К этим данным можно получить доступ многими способами, и при этом реорганизовывать таблицы БД не требуется.

Для хранения картинок мы будем использовать файловую систему.

**2.2. Выбор инструментов для создания сайта электронного портфолио, верстка, программирование.**

Страницы сайтов — это файлы с текстом, размеченным на языке HTML. Эти файлы, будучи загруженными на компьютер, используя браузер. Язык HTML позволяет форматировать текст, различать в нём функциональные элементы, создавать гиперссылки и вставлять в отображаемую страницу изображения, звукозаписи и другие мультимедийные элементы с помощью тегов.

Всего тегов более сотни, но действительно важных и постоянно нужных не так много. Самые главные теги:

html — это глобальный тег-контейнер, в который помещается все содержимое веб-страницы. Он является парным, впрочем, как и большинство описываемых далее тегов. Если и будут попадаться одинарные, то я об этом скажу.

head — содержит разную информацию, которая никак не будет выводиться на странице, но от нее непосредственно зависит внешний вид информации. Например, в head прописывается кодировка, подключается таблица стилей и файлы javascript. То есть здесь указывается все, что нужно для корректной работы странички. Следующие четыре тега прописываются именно в head.

link — с помощью этого одинарного тега подключаются внешние файлы. Например, иконка или таблица стилей

script — подключает файлы с расширением js (javascript).

title — задает название страницы, ее заголовок, который будет виден в поисковиках и в браузере.

meta — одинарный тег, в котором прописывается служебная информация, такая, как кодировка или ключевые слова.

body — тело страницы. Контейнер для всего содержимого, которое будет непосредственно выведено на веб-страничку.

Теги для текста, списки

p — создает параграф или абзац, который отделяется от других элементов небольшими вертикальными отступами. Основной элемент для хранения текстовой информации.

blockquote — большая цитата, которая можете содержать в себе другие теги для ее оформления. Может состоять, например, из нескольких абзацев.

h1 — h6 — теги для выделения заголовков на веб-странице.

q — маленькая цитата.

b, strong — выделяет текст жирным. Второй тег также придает словам особую важность.

em, i — выделение курсивом.

sub, sup — выводит текст в нижнем/верхнем индексе, уменьшая немного размер шрифта.

ol — в html это тег нумерованного списка, пунктам которого будут присваиваться порядковые номера.

ul — маркированный список. Его пункты просто отмечаются маркером, внешний вид которого можно изменить. Маркированный список делают тогда, когда нет нужно в строгом перечислении каких-то пунктов.

li — парный тег, в который помещается один пункт маркированного или нумерованного списка. То есть ul или ol выступает лишь в роли контейнера.

Картинки

Картинки в html вставляются с помощью одинарного тега . Обязательным атрибутом является src (source), то есть путь к самой картинке. Остальные параметры (размер, позиционирование, эффекты) лучше настраивать через css.

Таблицы

table — тег-контейнер, в который помещается вся таблица.

tr — ряд таблицы, в который вписываются ее ячейки. Сам ряд никак нельзя выделить, он выступает лишь в роли контейнера.

td — ячейка таблицы. Вот к ней как раз и можно применить какие-то стили, размер, в нее можно вписать информацию, вставить картинку или видео.

Пустые блочные и строчные контейнеры

div — пустой блочный контейнер, не выполняющий больше никаких функций. До сих пор с его помощью, например, верстают боковую колонку на сайте.

span — аналогично с дивов это пустой контейнер, но не блочный, а строчный. Чаще всего применяется для выделения отдельных кусков текста и применения к ним каких-то интересных стилей.

Мультимедиа

В список тегов html можно добавить и более современные. Например, тег video, в атрибутах которого можно задать адрес до видеоролика, а также кодеки для его проигрывания на веб-странице. Сюда же входит audio. С помощью этого парного тега можно вставить аудиозапись.

Формы

**form** — в этот тег помещается вся форма (обычно для регистрации, заказа чего-то). Имеет атрибуты, связывающие ее с сервером, иначе она не будет работать.

**input** — основной одинарный тег, который позволяет вставить в форму различные поля (для ввода текста, телефона, выбора даты рождения).

**textarea** — большое текстовое поле. Обычно в него можно вписать отзыв или комментарий.

**Среда разработки**

В качестве среды разработки можно использовать текстовый редактор, но мы выберем

**IntelliJ IDEA** — интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java, JavaScript, Python

**Языки разработки**

**JavaScript** -  мультипарадигменный [язык программирования](https://ru.wikipedia.org/wiki/%25D0%25AF%25D0%25B7%25D1%258B%25D0%25BA_%25D0%25BF%25D1%2580%25D0%25BE%25D0%25B3%25D1%2580%25D0%25B0%25D0%25BC%25D0%25BC%25D0%25B8%25D1%2580%25D0%25BE%25D0%25B2%25D0%25B0%25D0%25BD%25D0%25B8%25D1%258F). Поддерживает [объектно-ориентированный](https://ru.wikipedia.org/wiki/%25D0%259E%25D0%25B1%25D1%258A%25D0%25B5%25D0%25BA%25D1%2582%25D0%25BD%25D0%25BE-%25D0%25BE%25D1%2580%25D0%25B8%25D0%25B5%25D0%25BD%25D1%2582%25D0%25B8%25D1%2580%25D0%25BE%25D0%25B2%25D0%25B0%25D0%25BD%25D0%25BD%25D0%25BE%25D0%25B5_%25D0%25BF%25D1%2580%25D0%25BE%25D0%25B3%25D1%2580%25D0%25B0%25D0%25BC%25D0%25BC%25D0%25B8%25D1%2580%25D0%25BE%25D0%25B2%25D0%25B0%25D0%25BD%25D0%25B8%25D0%25B5), императивный и [функциональный](https://ru.wikipedia.org/wiki/%25D0%25A4%25D1%2583%25D0%25BD%25D0%25BA%25D1%2586%25D0%25B8%25D0%25BE%25D0%25BD%25D0%25B0%25D0%25BB%25D1%258C%25D0%25BD%25D0%25BE%25D0%25B5_%25D0%25BF%25D1%2580%25D0%25BE%25D0%25B3%25D1%2580%25D0%25B0%25D0%25BC%25D0%25BC%25D0%25B8%25D1%2580%25D0%25BE%25D0%25B2%25D0%25B0%25D0%25BD%25D0%25B8%25D0%25B5) стили

**Java** -  [строго](https://ru.wikipedia.org/wiki/%25D0%25A1%25D0%25B8%25D0%25BB%25D1%258C%25D0%25BD%25D0%25B0%25D1%258F_%25D0%25B8_%25D1%2581%25D0%25BB%25D0%25B0%25D0%25B1%25D0%25B0%25D1%258F_%25D1%2582%25D0%25B8%25D0%25BF%25D0%25B8%25D0%25B7%25D0%25B0%25D1%2586%25D0%25B8%25D1%258F) типизированный [объектно-ориентированный язык программирования](https://ru.wikipedia.org/wiki/%25D0%259E%25D0%25B1%25D1%258A%25D0%25B5%25D0%25BA%25D1%2582%25D0%25BD%25D0%25BE-%25D0%25BE%25D1%2580%25D0%25B8%25D0%25B5%25D0%25BD%25D1%2582%25D0%25B8%25D1%2580%25D0%25BE%25D0%25B2%25D0%25B0%25D0%25BD%25D0%25BD%25D1%258B%25D0%25B9_%25D1%258F%25D0%25B7%25D1%258B%25D0%25BA_%25D0%25BF%25D1%2580%25D0%25BE%25D0%25B3%25D1%2580%25D0%25B0%25D0%25BC%25D0%25BC%25D0%25B8%25D1%2580%25D0%25BE%25D0%25B2%25D0%25B0%25D0%25BD%25D0%25B8%25D1%258F) общего назначения

**Фраймворки**

Для создания сайта электронного портфолио были выбраны популярные фраймворки (frameworks)

**Framework**— [программная платформа](https://ru.wikipedia.org/wiki/%25D0%259A%25D0%25BE%25D0%25BC%25D0%25BF%25D1%258C%25D1%258E%25D1%2582%25D0%25B5%25D1%2580%25D0%25BD%25D0%25B0%25D1%258F_%25D0%25BF%25D0%25BB%25D0%25B0%25D1%2582%25D1%2584%25D0%25BE%25D1%2580%25D0%25BC%25D0%25B0), определяющая структуру программной системы; [программное обеспечение](https://ru.wikipedia.org/wiki/%25D0%259F%25D1%2580%25D0%25BE%25D0%25B3%25D1%2580%25D0%25B0%25D0%25BC%25D0%25BC%25D0%25BD%25D0%25BE%25D0%25B5_%25D0%25BE%25D0%25B1%25D0%25B5%25D1%2581%25D0%25BF%25D0%25B5%25D1%2587%25D0%25B5%25D0%25BD%25D0%25B8%25D0%25B5), облегчающее разработку и объединение разных компонентов большого программного проекта.

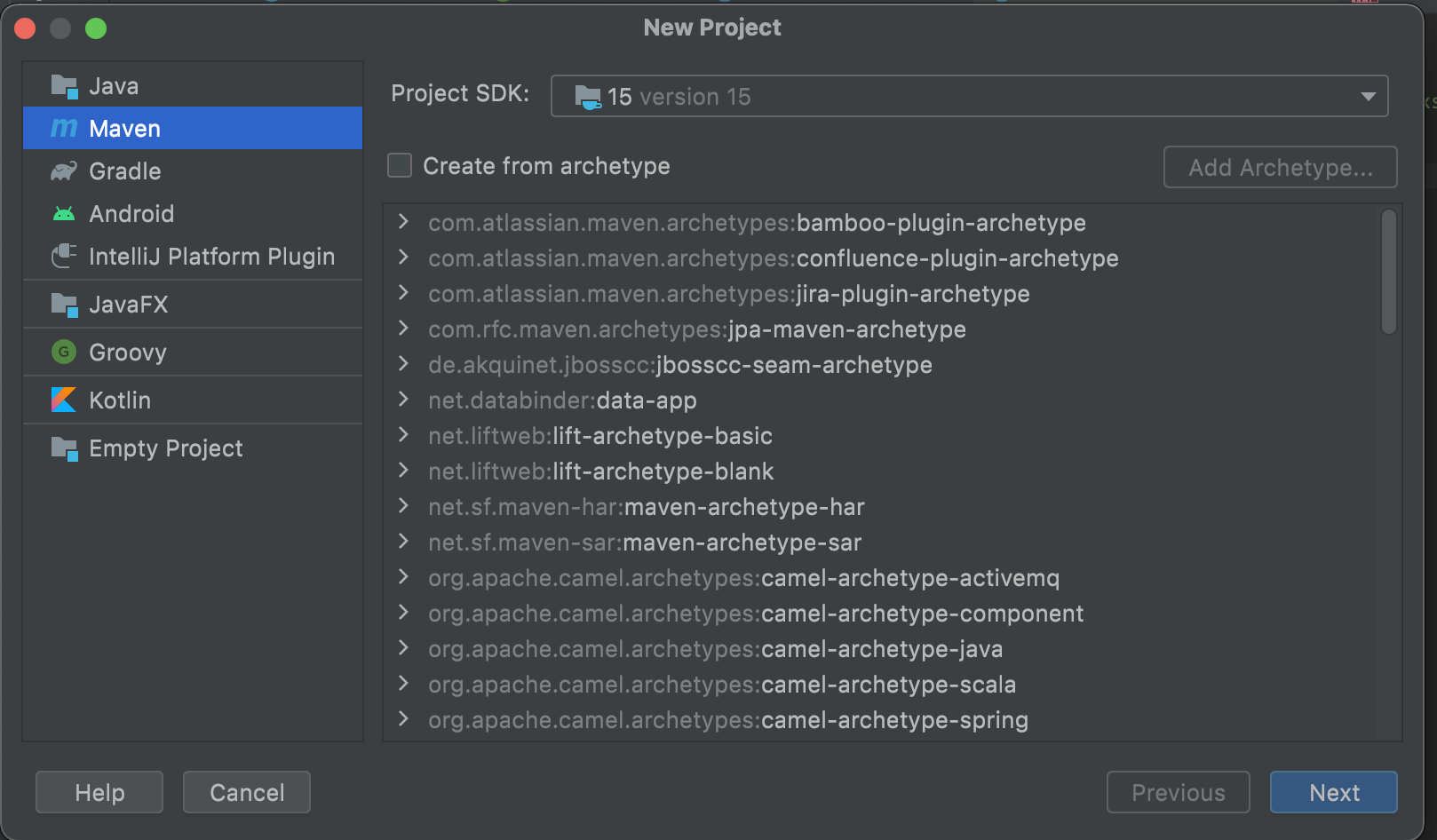
**Bootstrap** - свободный набор инструментов для создания [сайтов](https://ru.wikipedia.org/wiki/%25D0%25A1%25D0%25B0%25D0%25B9%25D1%2582) и [веб-приложений](https://ru.wikipedia.org/wiki/%25D0%2592%25D0%25B5%25D0%25B1-%25D0%25BF%25D1%2580%25D0%25B8%25D0%25BB%25D0%25BE%25D0%25B6%25D0%25B5%25D0%25BD%25D0%25B8%25D0%25B5). Включает в себя [HTML](https://ru.wikipedia.org/wiki/HTML)- и [CSS](https://ru.wikipedia.org/wiki/CSS)-шаблоны оформления для [типографики](https://ru.wikipedia.org/wiki/%25D0%25A2%25D0%25B8%25D0%25BF%25D0%25BE%25D0%25B3%25D1%2580%25D0%25B0%25D1%2584%25D0%25B8%25D0%25BA%25D0%25B0), веб-форм, кнопок, меток, блоков навигации и прочих компонентов веб-интерфейса, включая [JavaScript](https://ru.wikipedia.org/wiki/JavaScript)-расширения.

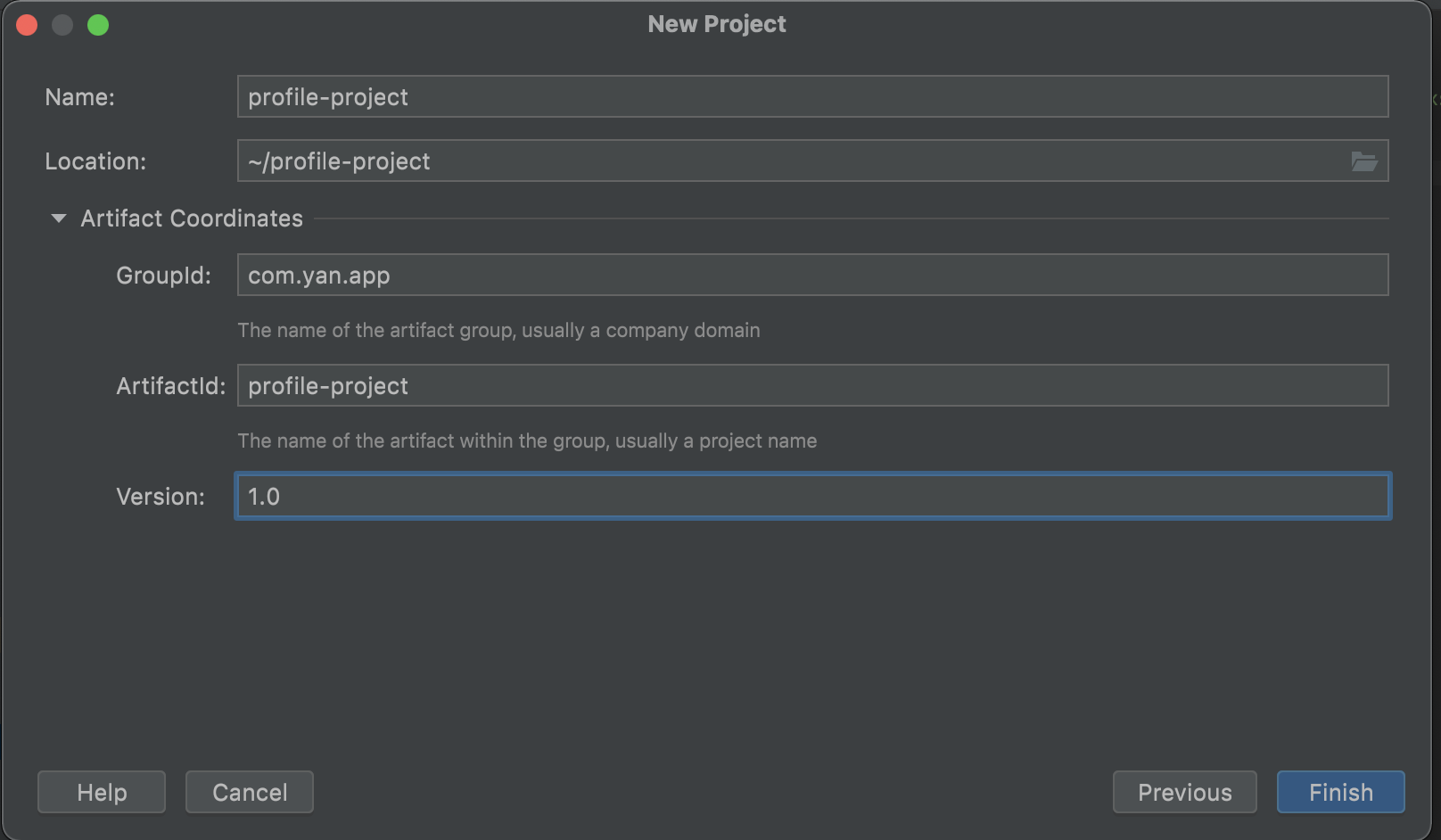
**Spring Boot** - это проект, построенный по принципу convention-over-configuration (соглашения по конфигурации), призванный упросить создание приложений на базе [Spring Framework](https://ru.bmstu.wiki/Spring_Framework). Решение предварительно настроено командой разработчиков с учётом использования платформы Spring и сторонних библиотек.

**Cборка проекта**

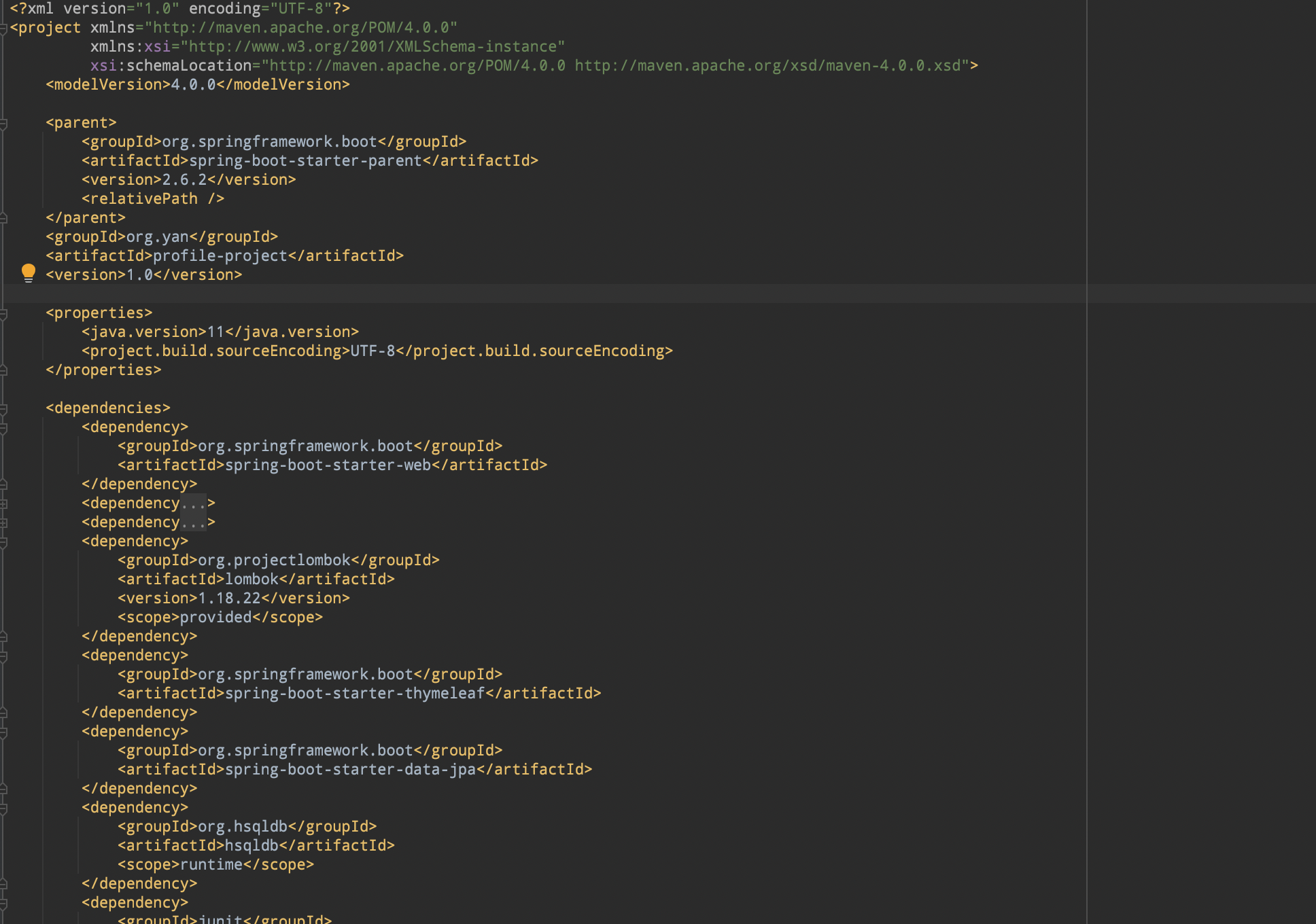
Для сборки проекта мы будем использовать популярный сборщик Apache Maven.  
**Apache Maven** — [фреймворк](https://ru.wikipedia.org/wiki/%25D0%25A4%25D1%2580%25D0%25B5%25D0%25B9%25D0%25BC%25D0%25B2%25D0%25BE%25D1%2580%25D0%25BA) для [автоматизации сборки](https://ru.wikipedia.org/wiki/%25D0%2590%25D0%25B2%25D1%2582%25D0%25BE%25D0%25BC%25D0%25B0%25D1%2582%25D0%25B8%25D0%25B7%25D0%25B0%25D1%2586%25D0%25B8%25D1%258F_%25D1%2581%25D0%25B1%25D0%25BE%25D1%2580%25D0%25BA%25D0%25B8) проектов на основе описания их структуры в файлах на языке [POM](https://ru.wikipedia.org/w/index.php?title=POM_(%25D1%258F%25D0%25B7%25D1%258B%25D0%25BA_%25D0%25BA%25D0%25BE%25D0%25BD%25D1%2584%25D0%25B8%25D0%25B3%25D1%2583%25D1%2580%25D0%25B0%25D1%2586%25D0%25B8%25D0%25B8_%25D0%25BF%25D1%2580%25D0%25BE%25D0%25B5%25D0%25BA%25D1%2582%25D0%25BE%25D0%25B2)&action=edit&redlink=1) ([англ.](https://ru.wikipedia.org/wiki/%25D0%2590%25D0%25BD%25D0%25B3%25D0%25BB%25D0%25B8%25D0%25B9%25D1%2581%25D0%25BA%25D0%25B8%25D0%25B9_%25D1%258F%25D0%25B7%25D1%258B%25D0%25BA) [Project Object Model](https://en.wikipedia.org/wiki/Project_Object_Model)), являющемся подмножеством [XML](https://ru.wikipedia.org/wiki/XML).

Создание сайта

Cоздадим проект в среде Intellij IDEA тип проекта Maven:

Заполним все поля в форму такие как Name, GroupId 

И так мы создали проект, следующим шагом надо добавить зависимости такие как Spring Boot, Thymeleaf в файл pom.xml



Структура моего проект:

Главным файлом моего приложения будет класс: ProfileApplication, этот класс будет точкой входа в наше приложение! Spring Boot просканирует все пакеты входящие в мой проект и запустит мой сервер!

В качестве основной модели работы приложения я выбрал патерн MVC (Model View Controller) - модель-представление-контроллер.

Это способ организации кода, который подразумевает выделение основных элементов, отвечающих за решение разных задач. Один блок отвечает за данные приложения, другой отвечает за внешний вид, а третий контролирует работу приложения.

Как пример попишем регистрацию ученика используя MVC:

* Модель: пользовательский профайл который хранится в Базе Данных.
* Представление: это HTML-шаблон, который возвращает сервер после обработки запроса. Если запрос корректно обрабатывается, вы получаете веб-страницу со списком портфолио учеников. Если запрос некорректный, вы попадаете на страницу ошибки 404(Not Found).
* Контроллер: обрабатывает входящие запросы. В фреймворке это может заключаться в определении конкретных URL, на которые попадает пользователь при переходе по ссылке или при нажатии кнопки.

**Модель** у нас представляет собой взаимодействие двух классов это UserProfile и UserProfileRepository.

Объект класса UserProfileRepository используется для сохранения, поиска и удалению портфолио ученика.

**Контроллер** - представлен классом по имени ProfileController, и на каждый запрос, контроллер возвращает подготовленную HTML страницу с определённым набором данным - которые прописываются в определенный HTML шаблон.

В контроллере представленные основные HTTP вызовы нашего приложения:

**GET /profile/list** - получить и отобразить весь список профайлов.

**GET /profile/new** - получить файловый профайл для создания и открыть форму для создания профайла.

**GET /profile/view/**{uuid} - посмотреть профайл по его номеру регистрации.

**GET /profile/edit/**{uuid} - редактирование профайла.

**POST /profile/edit/**{uuid} - создание или сохранение профайла.

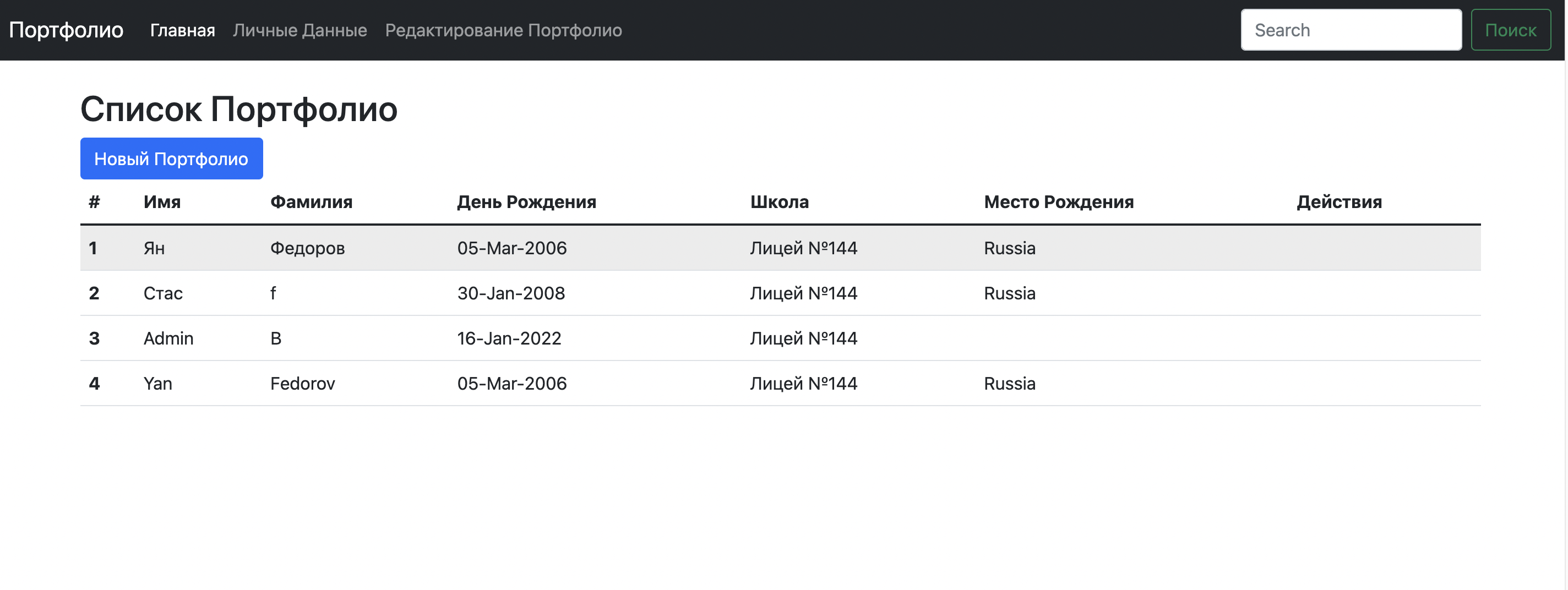
**POST /profile/upload** - загрузка нового изображения для профайла.

**GET /profile/image/**{uuid} - сервис для загрузки изображения профайла из файловой системы.

**Представление** - набор HTML Thymeleaf шаблонов , которые расположены в подразделе проекта **resources** в папке **templates**.

Основной шаблон **main.html** включающий в себя основные JavaScript фреймворки, стили и библиотеки.

Так же шаблон **main.html** содержит элементы навигации по нашему сайту при перехода по которым у нас загружаются страницы сайта.

Первой и основной страницей является список профайлов представленный в качестве таблицы:

Другие страницы сайта так же загружаются через основной шаблон main.html используя механизм фрагментов (thymeleaf fragments ) ,

расположенные в папке templates/fragments.

Следующие шаблоны мы будем использовать:  
**profile\_list.html** - шаблон для отображения списка профайлов;

**profile\_view.html** - шаблон для представления конкретного профайла ученика;

**profile\_edit.html** - шаблон для редактировния конкретного профайла ученика;

**navbar.html** - шаблон для осуществления навигации по моему сайту.

---- TODO: описание методов

Описание