**Spring Boot Первое приложение**

Для создания приложения будем использовать сервис который называется **Spring initializr**.

Это удобный сервис для создания пустого **Spring Boot** приложения.

**1.** Нужно перейти по ссылки <https://start.spring.io/>.

**2.** Выбираем настройки:

**1.** проект(maven или gradle)

**2.** язык,

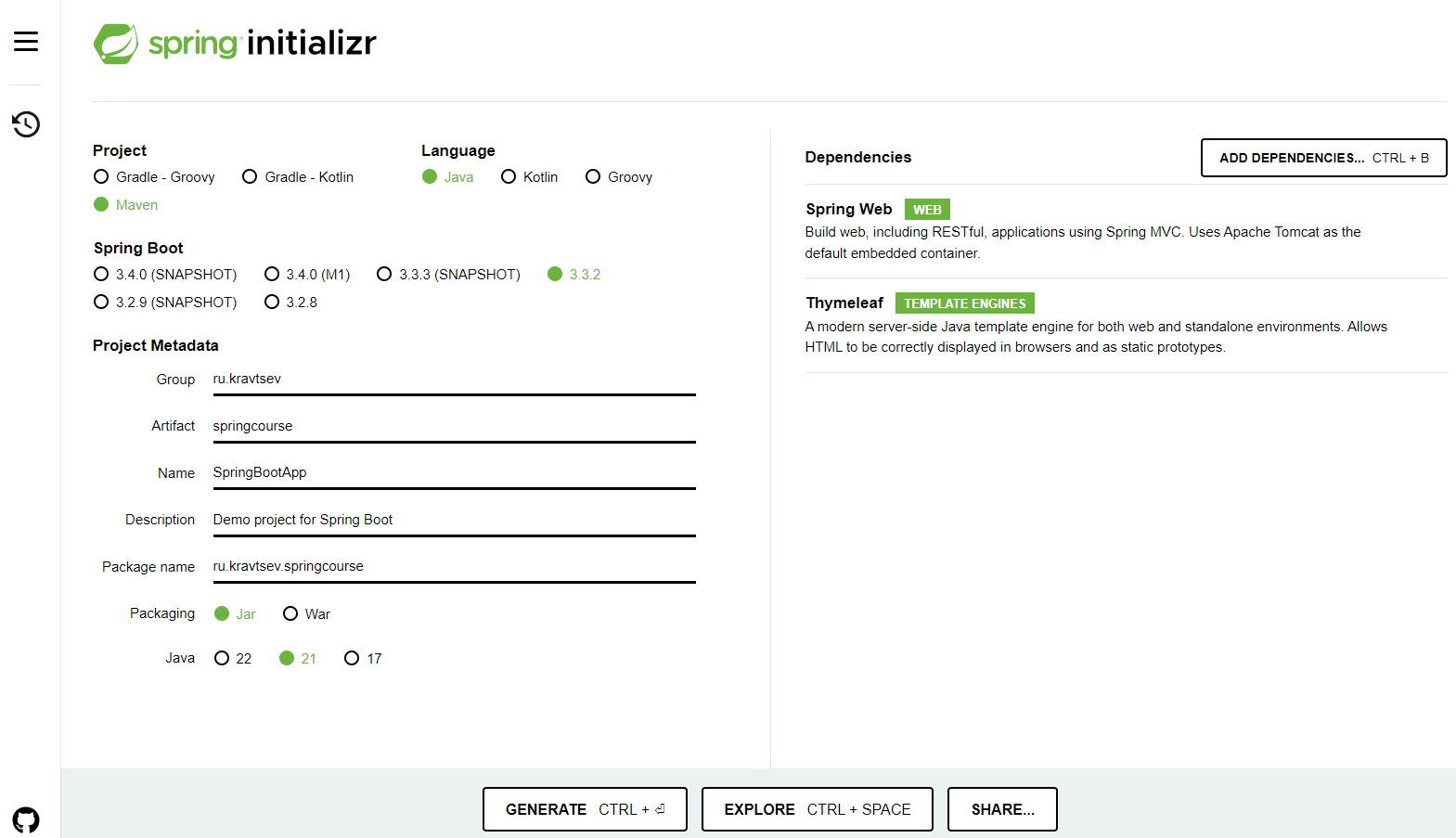
**3.** версию спринга

**4.** режим сборки (jar или war). Для использования встроенного сервера Spring Boot нужно выбрать jar.

**5.** версию джавы.

**6.** добавляем зависимости (стартеры). Spring Web, Thymeleaf.

После настройки можно нажать на кнопку **GENERATE** и проект который мы настроили скачается.



**3.** Открываем созданный проект.

Теперь посмотрим самый главный класс (**SpringBootAppApplication**), единственный класс который пока есть в нашей папке springcourse. Этот класс запускает наше **Spring** приложение. Здесь мы видим, что есть аннотация **@SpringBootApplication –** это главная аннотация в Spring Boot, которая в себе содержит все остальные аннотации, тость одна аннотация для настройки всего грубо говоря. Эта аннотация проводит автоконфигурацию приложения, настраивает сканирование компонентов и т. д. Эта аннотация содержит в себе такие аннотаци такие как **@ComponentScan**, **@Configuration** и т. д.

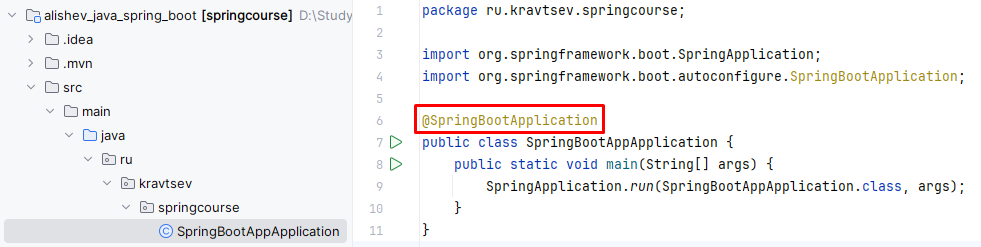
Благодаря этой аннотации если мы создадим класс и пометим его аннотацией **@Component**, то автоматически он будет отсканирован и из этого класса будет создан бин.

В этой аннотации **@SpringBootApplication,** аннотация **@ComponentScan** настроена таким образом, что он сканирует все компоненты в текущей папке, тоесть в папке в которой находится наш класс и во всех подпапках причем рекурсивно.

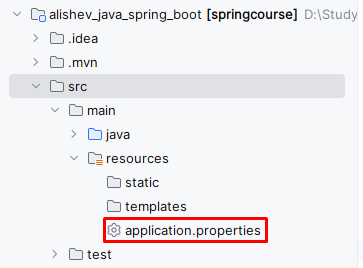
Класс **SpringBootAppApplication** тоесть тот класс с аннотацией **@SpringBootApplication** должен находится в корне нашего проекта, тоесть все остальные классы должны находится на одном уровне с этим классом или на уровень ниже.

**Посмотрм метод *main()***

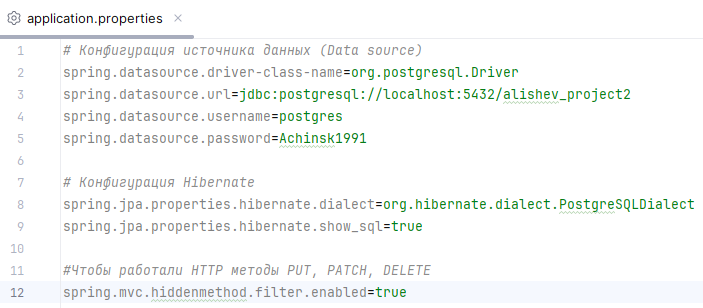
Этот метод который мы будем запускать. После запуска этого метода будет запускаться встроенный веб сервер и уже на нем будет запускаться **Spring Boot** приложение.



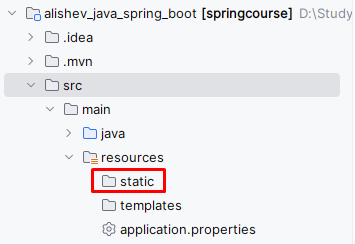
**4.** Следующий важный файл который есть в Spring Boot приложении это файл **application.properties**. Этот файл автоматически цепляется **Spring Boot**, он автоматически смотрит его содержимое и здесь мы помещаем нашу конфигурацию нашего **Spring Boot** приложения. Несмотря на то что **Spring Boot** очень сильно упрощает конфигурацию всё равно часто нам нужно что то конфигурировать в приложении. В этом файле находится вся конфигруация которая есть у нас в приложении, причем находится в формате кюч-значение. В этом файле так же будет находится конфигурация базы данных. Помимо стандартных ключей в этом файле так же можно вводить свои ключи, эти ключи и значения можем использовать в нашей программе для собственных нужд.



Пример свойств для подключения к базе данных и некоторых настроек **hibernate**:

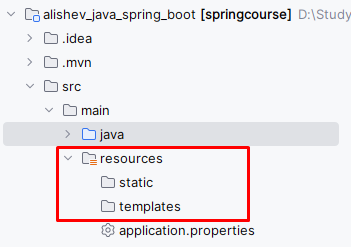


**5.** Папка **static**. В этой папке содержатся какие то статически ресурсы, например, файлы, картинки, css стили или возможно html страницы.



**6.** Папка **templates**. В этой папке лежат наши шаблоны, например, Thymleaf шаблоны.

Без использования **Spring Boot** все эти файлы лежали в папке webapp, но теперь папки webapp нет и все эти файлы будут лежать в папке **resources**. Вообще папка webapp это стандартная папка и эта пака может быть только в том случае если наш проект собирается в .war файл. Если проект собирается в .war файл, то веб сервер будет смотреть на содержимое папки webapp. Но так как у нас сейчас веб сервер встроеный и наше приложение собирается в .jar файл, то мы не должны теперь использовать папку webapp, потому-что для .jar файлов папка webapp не учитывается. Поэтому все наши шаблоны, HTML страницы, JS файлы, CSS файлы — все они будут лежать в папке **resources.**

****

**7.** Папка **test**. Здесь содрежится всё, что относится к unit-тестированию.

