Explicando o que é spring boot:

O Spring Boot é um framework Spring que permite a criação de códigos robustos com uma facilidade maior e sendo menos complexos que “fazer na mao”. Ele tem vários conjuntos de dependências para ajudar em determinados assuntos e é aberto para novas dependências.

O que é lombok?

Lombok é uma biblioteca/dependência que iremos usar para facilitar e reduzir a quantidade de código boiler-plate, que é aquele código chato de fazer e que acaba tornando o código cada vez mais ilegível e aberto a erros.
Usaremos essa biblioteca majoritariamente para criação de getters e setters.
Devemos adicionar a anotação @Getter e @Setter antes de uma classe e ela irá automaticamente fazer os getters e setters de cada atributo da classe.

O que é um DTO:

Data Transfer Object é uma ideia que consiste basicamente em agrupar um conjunto de atributos uma classe simples de forma a otimizar a comunicação.
Pode ser usado para mapear informações obtidas do banco de dados e então fazer o que for necessário com isso, seja criar novos dados a partir de um service ou ler essas informações na sua aplicação.
Como tudo em software temos escolhas que tem seu lado bom e o lado ruim(tradeoffs), Usar isso para mapear o banco de dados acaba sendo uma escolha eficiente quando o banco de dados não pretende ser escalável e pretende trabalhar só com uma gama de itens. Caso isso não aconteça, os dados podem acabar sendo redundantes e não performáticos no projeto, para projetos realmente complexos normalmente se usa o padrão beans, mas não irei abordar isso aqui. Sinta-se livre para pesquisar!

Comandos para criação de um DTO:

Usando o Spring Boot JPA e o Hibernate podemos criar um DTO facilmente com os seguintes comandos:
@Entity - Usado antes da classe para informar que uma classe representa uma entidade e que seus objetos devem ser persistidos no banco de dados

@Table(name = “variavel”) - Isso deve ser usado antes da criação da classe e esse comando serve para criar/usar a tabela no banco de dados.

@Id - Como o nome sugere, é usado antes de criar o atributo ID e declara que isso será um ID.

@GeneratedValue(strategy = “suaEstrategia”) - Aqui se declara qual o tipo de valor ele irá criar para o ID.

@Column(name = “nomeColuna”, columnDefinition = “definicaoColuna”) - Isso vem antes da criação de atributos que serão colunas no banco de dados, coloquei também a definição de colunas, isso pode dar mais controle sobre o tipo de coluna queremos criar, se não colocarmos nada o padrão e criar uma tabela de string.

Um bom exemplo de DTO junto com o lombok:

```java
@Getter
@Setter
@Entity
@Table(name = "user")
// Aconselho sempre nomear as tabelas em minusculo
// Isso faz ocupar menos memoria e gasta menos energia
// É pouco o que salvamos de energia, mas é algo.
public class User {
    @Id
    @GeneratedValue(strategy = "uuid2")
    @Column(name = "id", columnDefinition = "BINARY(16)")
    private UUID id;
    // Preferi escolher a definicao de coluna para o tipo BINARY(16)
    // pois isso minimiza o espaco(menos bytes e sem caracteres especiais)

    @Column(name = "name")
    private String name;

    @Column(name = "email")
    private String email;

    @Column(name = "password")
    private String password;
}
```