Why use Angular?

- Single page applications – better user experience
  - Improved speed and performance
- Developer advantages
  - Re-use code (for example, re-using a Nav component)
  - Utilize OOP concepts (Components themselves, .ts file is a class w/ properties, functions, etc)
    - Every component has properties and behaviors
  - Databinding **abstracts** the process of modifying what is displayed on the screen
- Modern technologies are mainly about abstracting away “what is under the hood”

Abstraction examples (not to be confused with 4 pillars of OOP)

- Angular: abstracts away event listeners, AJAX (http requests), DOM Manipulation
- Java: abstracts away having to write lower level code such as assembly or even pure binary (0 and 1s)
- Hibernate: abstracts away JDBC (Java Database Connectivity)
  - ORM (object relational mapper): maps objects to rows in a database table
  - Ex. User objects can be added to the corresponding user table in the database without having to worry about using the INSERT keyword syntax

The objective for today

- Establish the fundamentals of Hibernate
- Establish the fundamentals for Spring
  - A framework made of many different modules
  - Normally in the past, we would have had to put together all of these modules ourselves and have extensive configuration for them
- Once we have established the fundamentals of Hibernate and Spring…
  - Spring Boot
    - What is known as a “Spring project”
    - Heavily used in Java development
    - A template for Java-based projects utilizing the idea of “convention over configuration”
    - We stick with a certain convention that requires less configuration than if we were to put together everything independently
  - Once we get into Spring boot, we’ll be using Spring web (a module)
    - Allows us to map endpoints
  - Spring Data (another module)
    - Comes with Hibernate