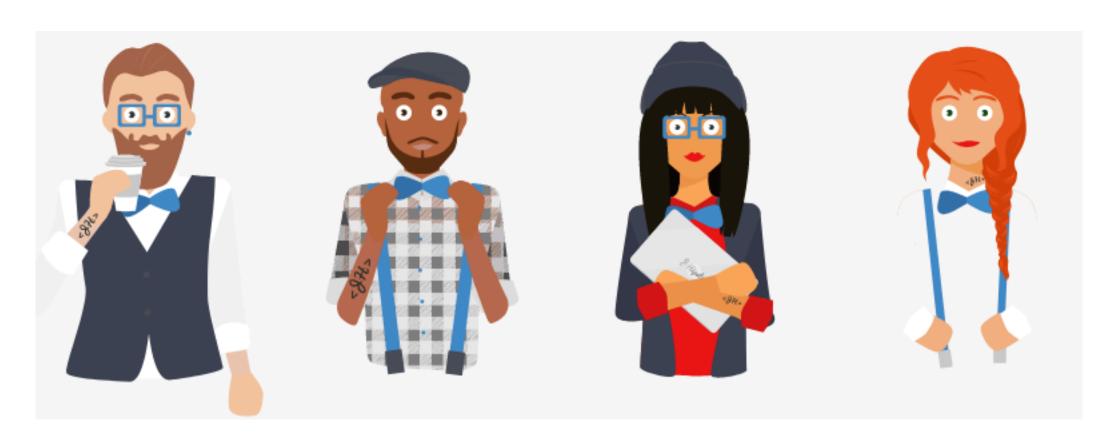
# Agile Team Project

Week 4 Part 4: Introduction to JHipster

From: https://www.jhipster.tech/

### What is 37tipster



- Web application generator
- Modern, production ready, infrastructure
- Combination of industry standard tools
- code generator: JDL
- jhipster command line interface

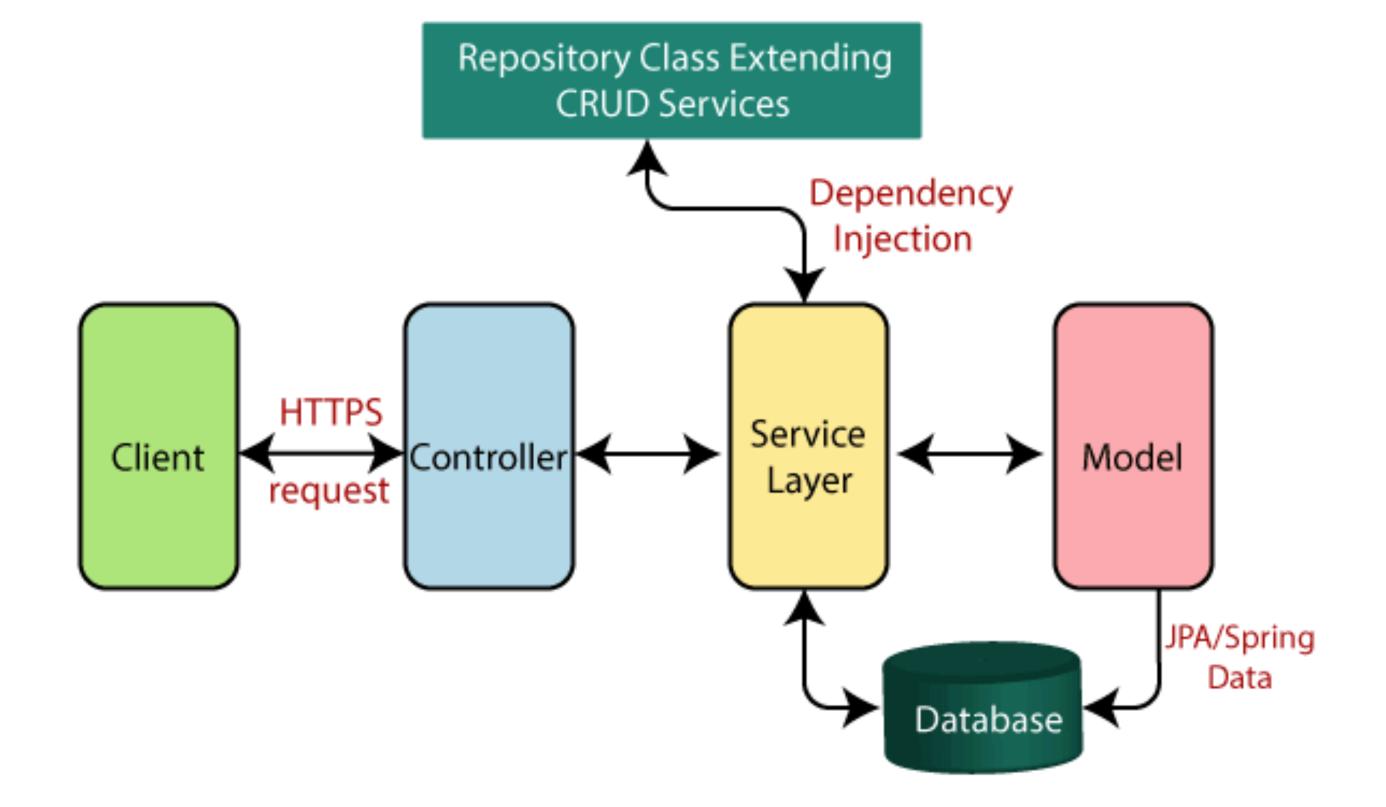


#### JHipster is a tech-stack generator

- Backend: Springboot (others possible)
- Frontends: Angular or React or Vue
- Databases: PostgreSQL, MySQL, noSQL
- Questions asked when generating an application : <a href="https://www.jhipster.tech/creating-an-app/#2">https://www.jhipster.tech/creating-an-app/#2</a>
- .yo-rc.json:contains options used to generate app

# n.b. you can re-generate the template app or use a completely different tech-stack





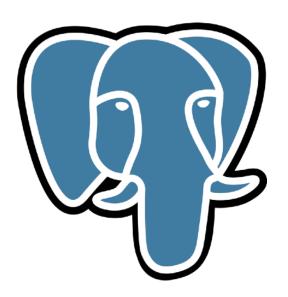
- Boot: "opinionated" framework for building Spring framework apps
  - Makes decisions, generates project files
- Java based: lots of library, tool support
- n.b. Generated by JHipster

### Backend

- Backend:
  - business logic: programs, classes
  - persistence layer: ORM
  - Respond to user requests based on State
  - REST APIs
- Result: dynamic web applications

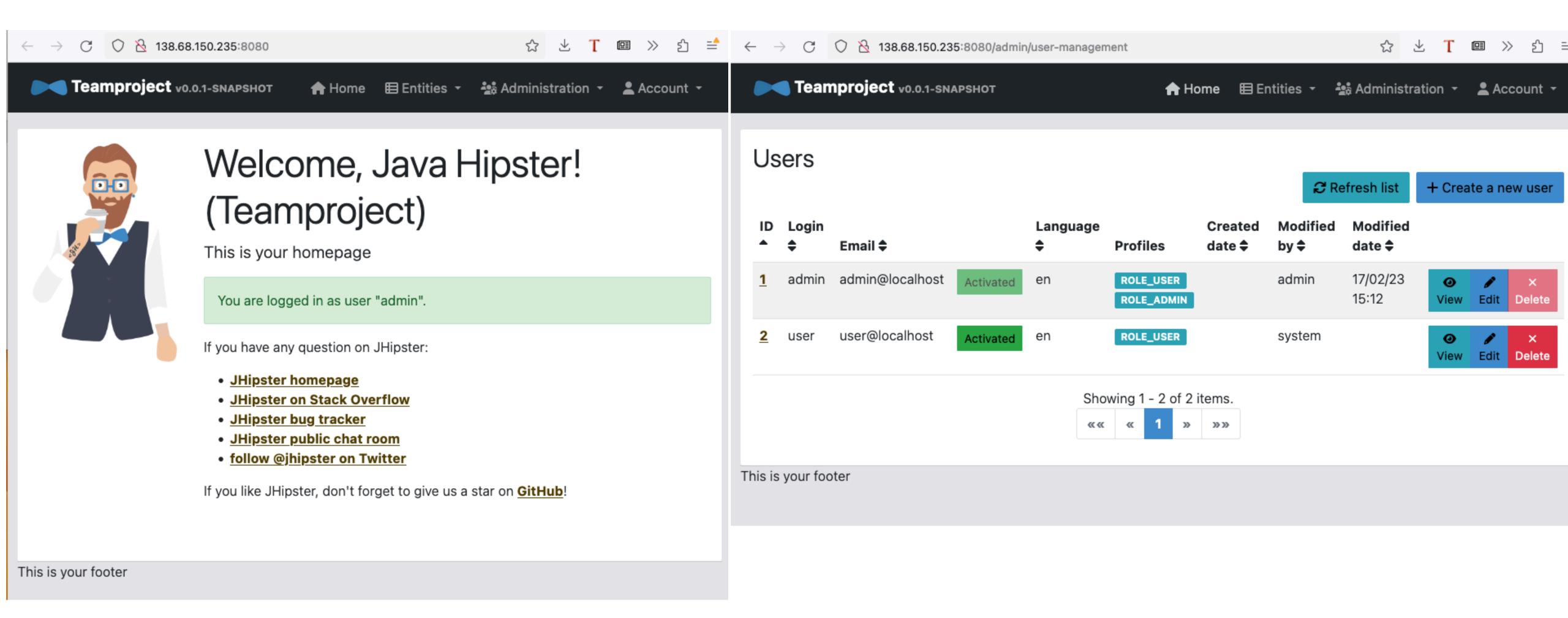
# Supported Databases

- PostgreSQL
- MySQL
- MariaDB
- H2 dev mode





# Admin Interface / User Management



# JHipster Domain Language

- JDL: Domain Specific Language
- Specify Entities and Relationships
- Editor: JDL Studio, IDE plugins
- UML Class-Diagram-like

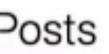
```
JDL-Studio
                                                   <Select template> ~
entity Region {
     regionName String
                                                                    JobHistory
 entity Country {
                                                                 startDate : Instant
     countryName String
                                                                  anguage : Language
 entity Location {
                                               Language
     streetAddress String,
                                                                       Department
                                                <<enum>>
     postalCode String,
                                               FRENCH
ENGLISH
                                                                  departmentName : String
     city String,
                                               SPANISH
     stateProvince String
entity Department {
                                                                                        Employee
                                                                    Location
                                                                                    firstName : String 
lastName : String
     departmentName String required
                                                                 streetAddress : String
                                                                                    email : String
phoneNumber : String
                                                                postalCode : String
                                                                 city: String
                                                                                     hireDate : Instant
 entity Task {
                                                                stateProvince : String
                                                                                    salary: Long
     title String,
     description String
                                                                    Country
entity Employee {
                                                                countryName : String
     firstName String,
                                                                                      maxSalary : Long
     lastName String,
     email String,
                                                                                         Task
                                                                     Region
     phoneNumber String.
                                                                                     title : String
                                                                 regionName : String
                                                                                     description: String
     hireDate Instant,
     commissionPct Long
```

https://www.jhipster.tech/jdl/intro#-jhipster-domain-language-jdl

### CRUD

#### Using JDL we get a CRUD app for our entities

- Create
- Read
- Update
- Delete





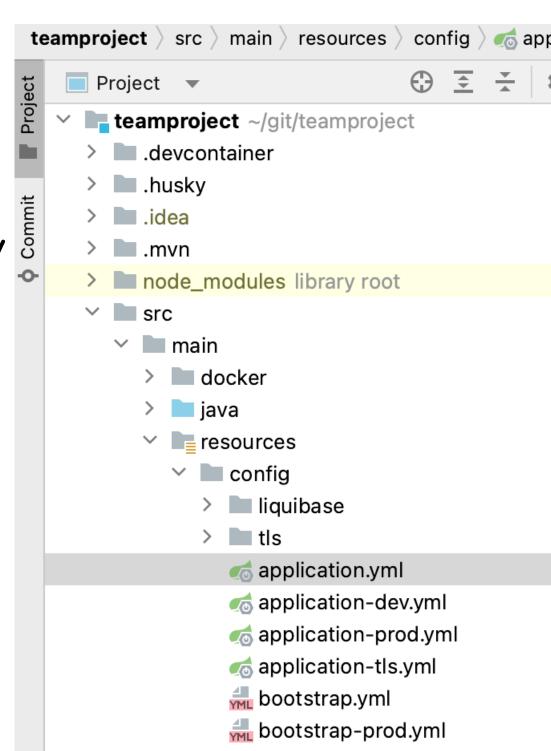
# FakerJS for dev profile

- Automatically generates example entities
- Useful- avoid to manually creating dummy data
- In dev profile only; disable in:

```
C UsernameAlreadyUsedException.java × C UserService.java × 👩 application-dev.yml ×
UserRepository.java × C MailService.java ×
             poolName: Hikari
                                    src/main/resources/config/application-dev.yml
             auto-commit: false
         h2:
                          Line 48 remover faker
           console:
             # disable spring boot built-in h2-console since we start it manually with correct conf
             enabled: false
         jpa:
         liquibase:
           # Remove 'faker' if you do not want the sample data to be loaded automatically
47
           contexts: dev, faker
48
         mail:
49
           host: localhost
50
```

# SpringBoot: profiles

- configurations for production and development
- databases, DEBUG messages, dummy data etc
- Local DB in /target/h2db
   (local machine only "dummy data")
- Remote DB: ~/volumes/jhipster/teamproject/postgresql/ (VM only-"real data")
  - VM DB config: src/main/docker/app.yml

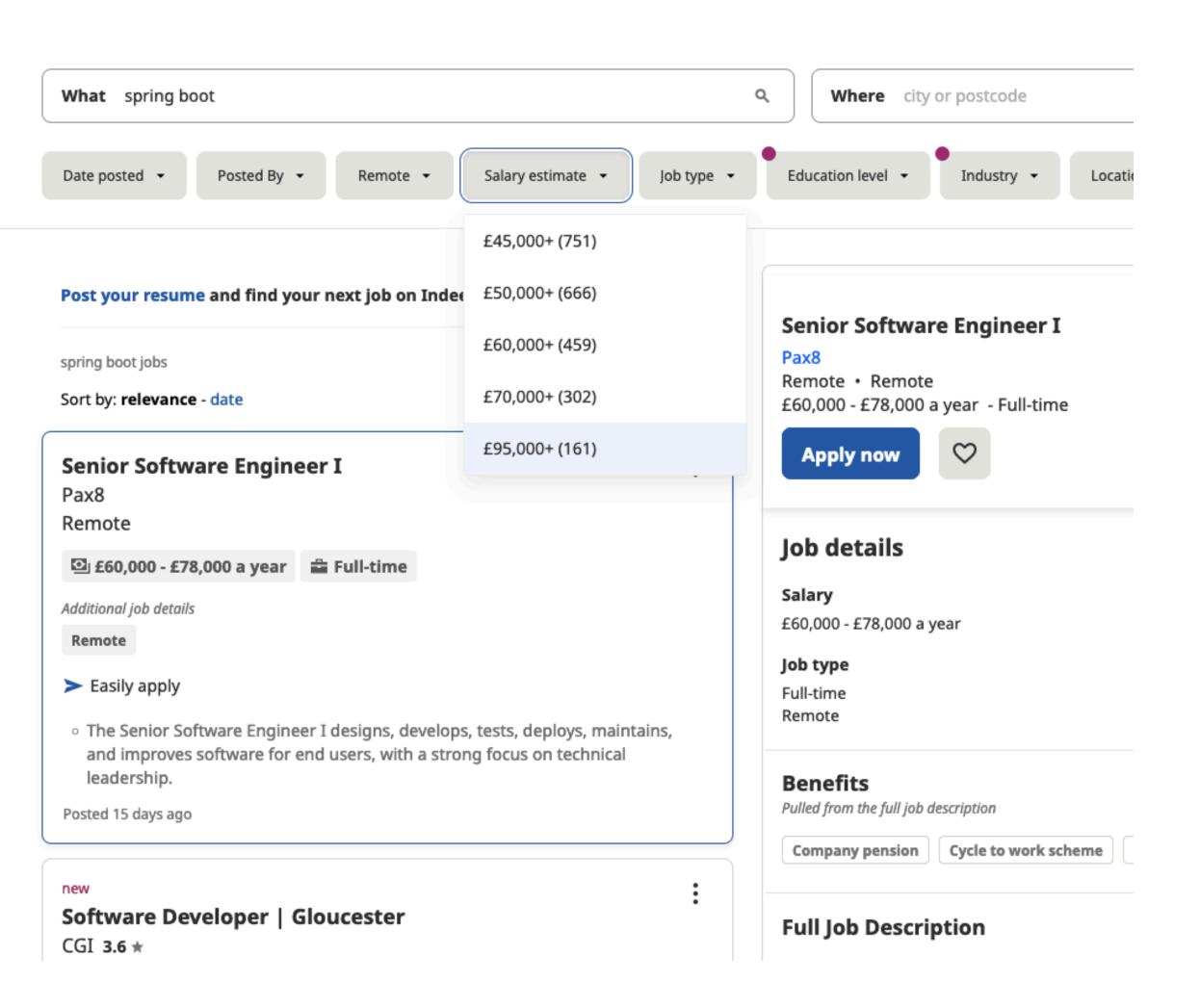


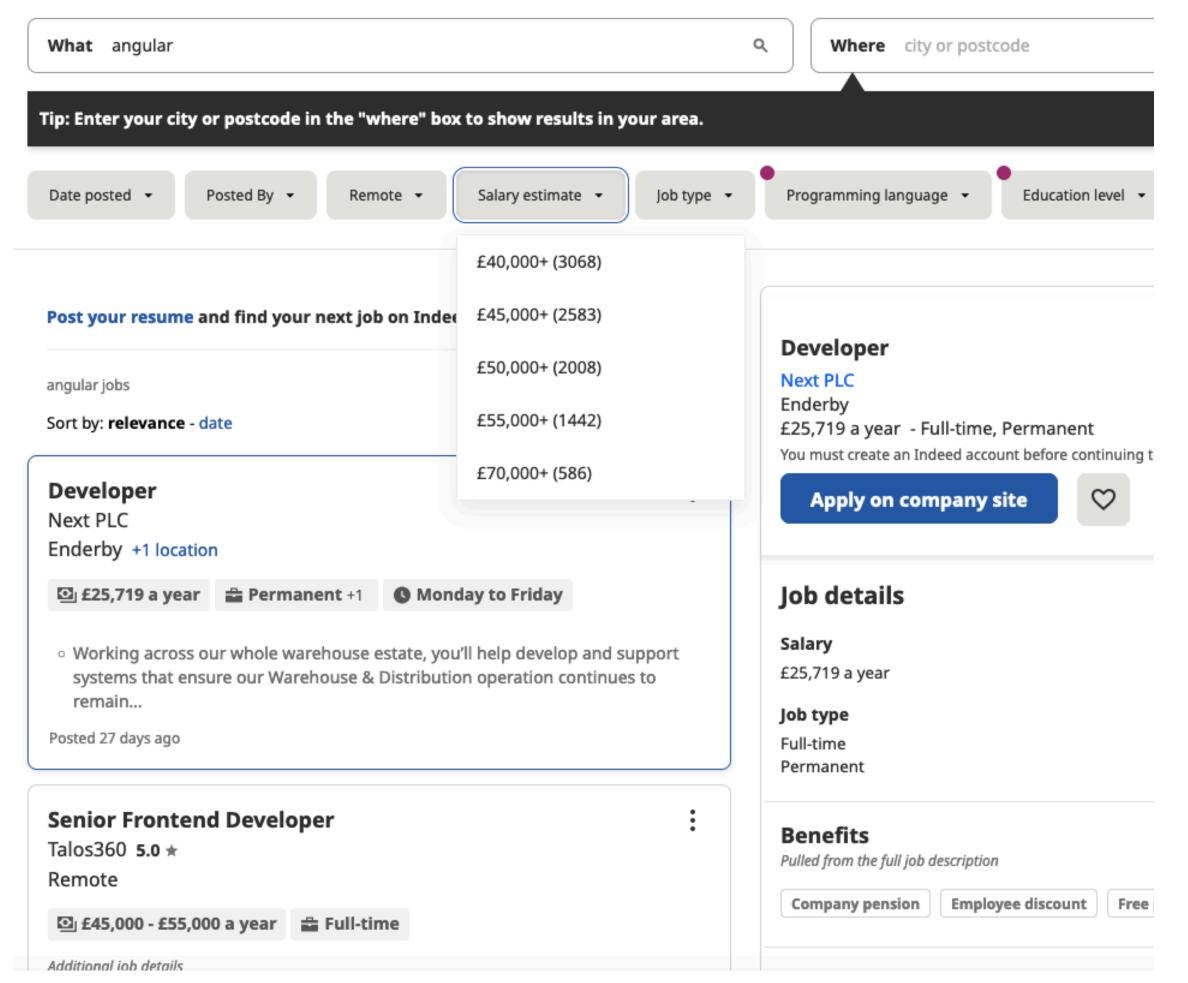
# Angular Front End

- TypeScript
  - Microsoft, type-checked JS
- Angular
  - Google aka angular2
- Modular
- Event/property binding
- npm build environment

```
// src/app/app-routing.module.ts
import { NgModule } from '@angular/core'
import { Routes } from '@angular/router'
import { NativeScriptRouterModule } from '@nativescript/angular'
const routes: Routes = [
  // Update this 👇
  { path: '', redirectTo: '/home', pathMatch: 'full' },
  // Add this 👇
    path: 'home',
    loadChildren: () => import('./features/home/home.module').then(m =>
@NgModule({
  imports: [NativeScriptRouterModule.forRoot(routes)],
  exports: [NativeScriptRouterModule]
```

## JHipster Careers





### Summary

- JHipster: modern web app generator
- SpringBoot: dynamic, Java backend
- Angular Front End