**Criando Um Rest api com Scala.**

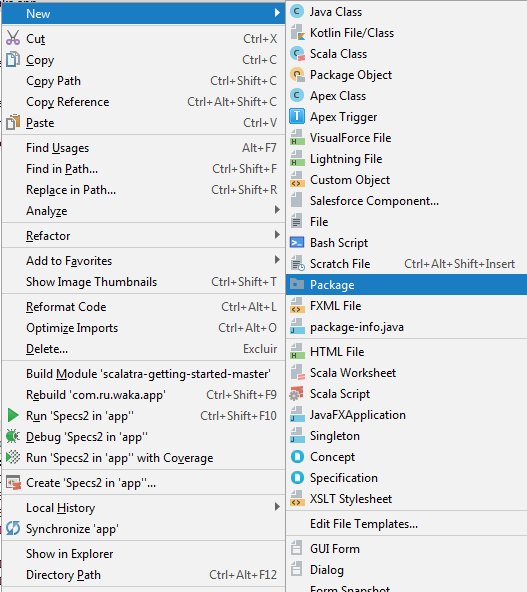
Para iniciar faça o download do projeto base.

<https://github.com/ara-ta3/scalatra-getting-started>

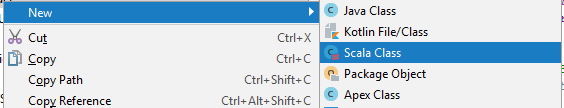
Este projeto já possui um esqueleto para uma aplicação rest em scala.

**Criando um exemplo**

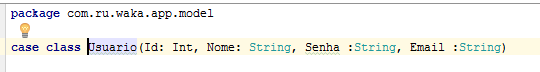
Vamos criar três pastas model, rest e service. Para isso navege até a pasta scala/com.ru.waka.app selecione e click com o botão direito



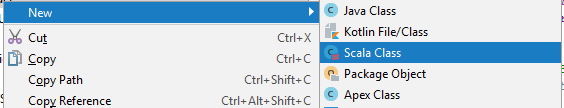
Feito isso vamos adicionar uma model para usuários



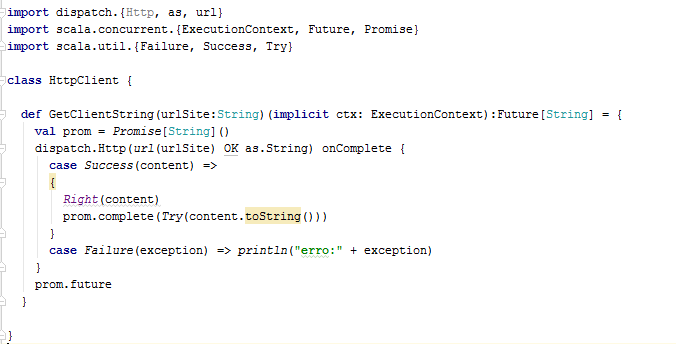
Logo após inclua o seguinte trecho de código



Feito isso vamos adicionar uma class HttpClient na pasta rest . Essa class será responsável pelo gerenciamento das requisições.

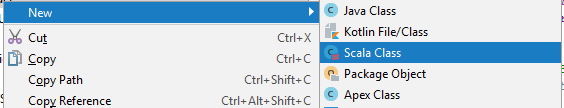


Logo após inclua o seguinte trecho de código

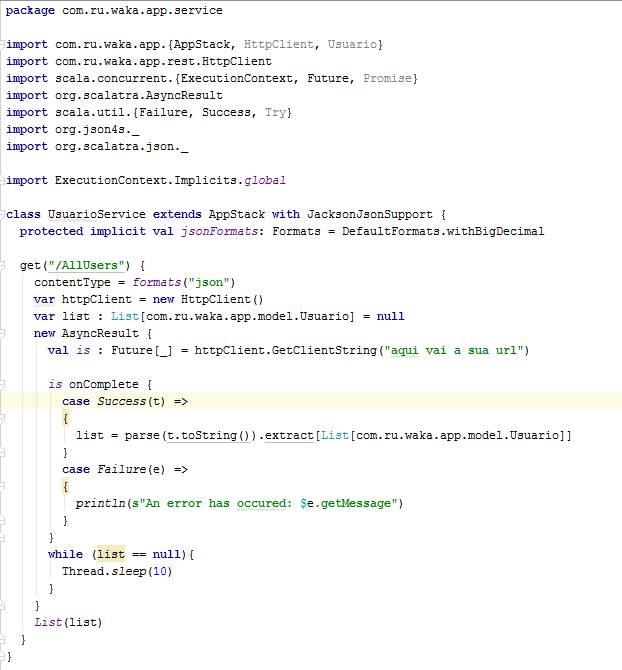


Aqui utilizamos a biblioteca Dispatch exemplo : <http://lollyrock.com/articles/scala-http-requests/>

Feito isso vamos adicionar uma class UsuarioService na pasta service . Essa class será responsável pelos de controller.



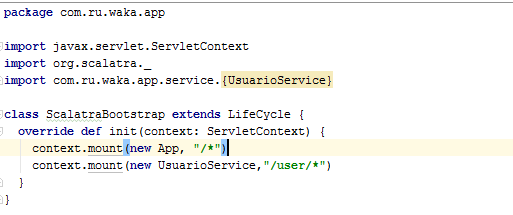
Logo após inclua o seguinte trecho de código



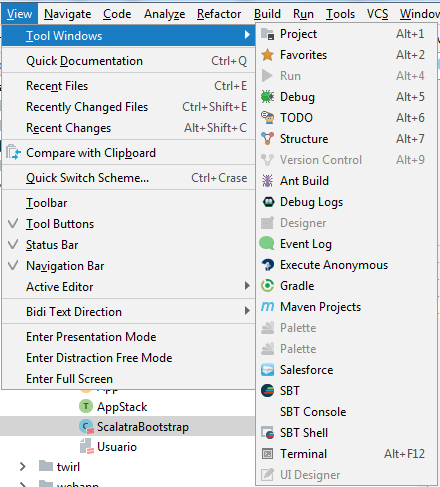
Aqui utilizamos a biblioteca Futures exemplo : <https://doc.akka.io/docs/akka/2.5.4/scala/futures.html>

Criadas as devidas class´s vamos configurar a rota do serviço relacionado ao Usuário.

Navegue até a class ScalatraBootstrap e inclua o seguinte código



Agora estamos prontos para iniciar a aplicação. Navegue conforme imagem abaixo para habilitar o SBT Shell



No SBT Shell rode os seguinte comandos

compile

jetty:start

http://localhost:8080/user/AllUsers

Importar projeto para o Intellij

No diretório



**Adicionando Um Get no Rest api conecta Scala credenciado-medico.**

No arquivo scala Inclua o seu método do tipo get/post

**get("/credenciado/:codigoCredenciado/teste/:marcaOticaCpf") {**

**new AsyncResult {**

**val codigoCredenciado = tokenSession.codigoCredenciadoAmil**

**val is = { Future.successful(BadRequest(ResultMessage("elegibilidade.dado.pesquisa.invalido").json))}**

**}**

**}**

O trecho **("/credenciado/:codigoCredenciado/teste/:marcaOticaCpf")** corresponde a rota de seu método.

No arquivo ts Inclua o seu método do tipo get/post ao serviço relacionado.

**import { Injectable } from '@angular/core';**

**import { Headers, Http } from '@angular/http';**

**import { Observable } from 'rxjs/Observable';**

**import { SessionService } from '../session/session.service';**

**import { ConsultaElegibilidade } from '../elegibilidade/elegibilidade.model';**

**@Injectable()**

**export class InfoBeneficiarioService {**

**//Propriedade relacionada a Rota de seu método scala**

**private elegibilidadeTesteUrl = 'credenciadomedico/elegibilidade/credenciado/{credenciado}/teste/{marcaOticaCpf}';**

**private headers = new Headers({**

**'Accept': 'application/vnd.amil.credenciado.medico.elegibilidade.v1+json',**

**'Content-Type': 'application/json'**

**});**

**constructor(private http: Http, private sessionService: SessionService) { }**

**response(result: any) {**

**return Array.isArray(result) ? result : [result];**

**}**

**//Seu método**

**consultarElegibilidadeTeste(marcaOticaCpf: string): Observable <ConsultaElegibilidade> {**

**debugger;**

**return this.http.get(this.getUrlTeste(marcaOticaCpf), { headers: this.headers })**

**.map(res => new ConsultaElegibilidade(res.json()));**

**}**

**//Método responsável pela formatação da url**

**private getUrlTeste(marcaOticaCpf: string) {**

**let query = marcaOticaCpf.replace(/[^0-9]/g, '');**

**return this.elegibilidadeTesteUrl.replace(/\{credenciado\}/, this.sessionService.getCodigoCredenciado())**

**.replace(/\{marcaOticaCpf\}/, query);**

**}**

**}**

Logo após inclua o import referente ao seu serviço e o método abaixo

**this.InfoBeneficiarioService.consultarElegibilidadeTeste(value).subscribe(**

**consultaElegibilidade => {**

**let seuValor = seuRetorno;**

**},**

**err => {**

**console.log(err);**

**},**

**() => {**

**console.log('Erro');**

**});**

Logo após é necessário atualizar o apache e o weblogic . para isso usaremos nos seguintes códigos no terminal da VM.

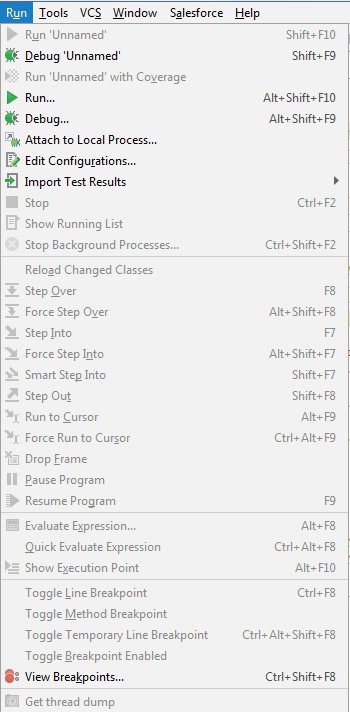
sbt clean package

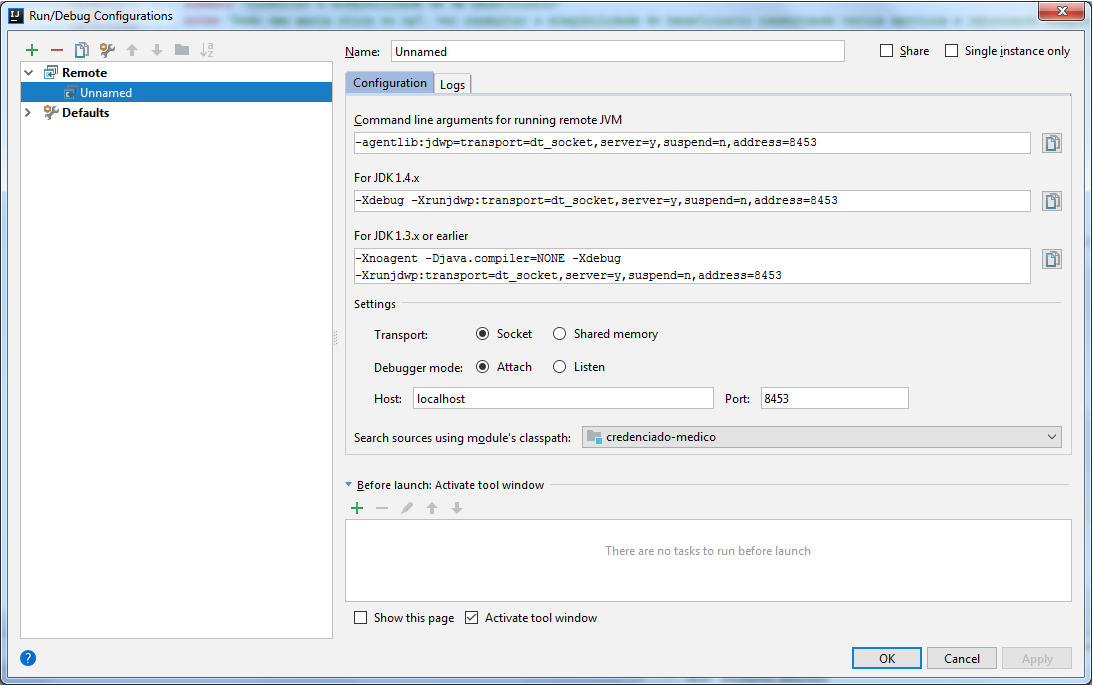
docker restart amil-weblogic amil-apache

docker-npm run build-dev

Para debuggar o seu método no scala é necessário configurar o debugger.

No ide Intellij

 selecione Edit Configurations



Preencha o campo Port com o valor 8453.

No campo “Search sources using module´s classpath” selecione a pasta do credenciado-medico.Click em “Apply”