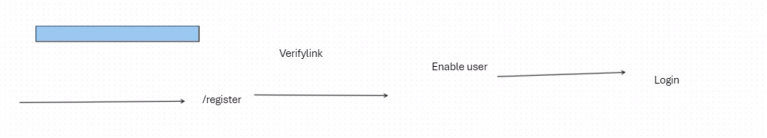


In controller do the changes

In product service and also in product service implementation.

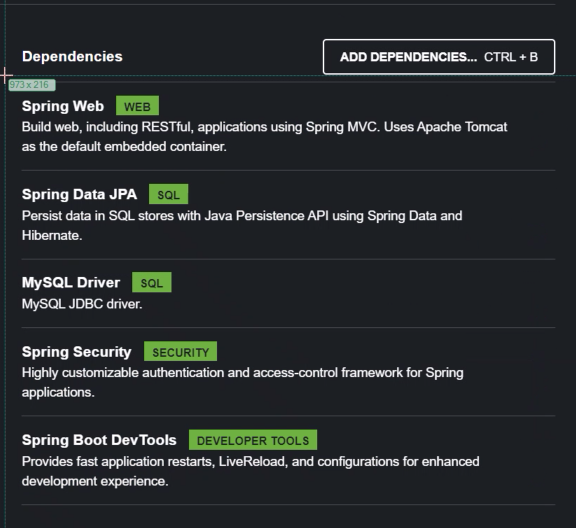
ApiResponse

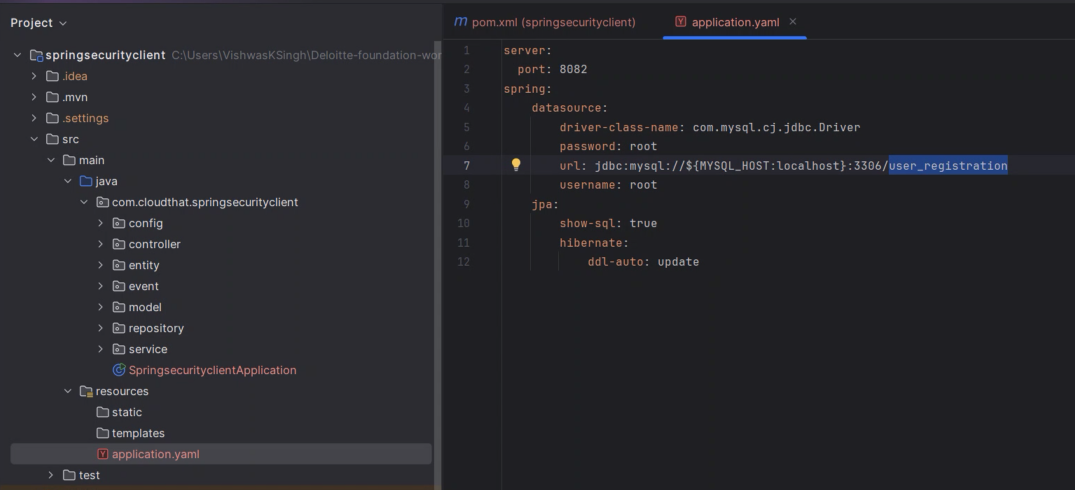
ProductModel – will only have the fields which I want to show to the user for this change product service and product serviceimpl



Dependencies : datajps, web, starter security connectorJ.

Create an spring initializer application add the dependencies

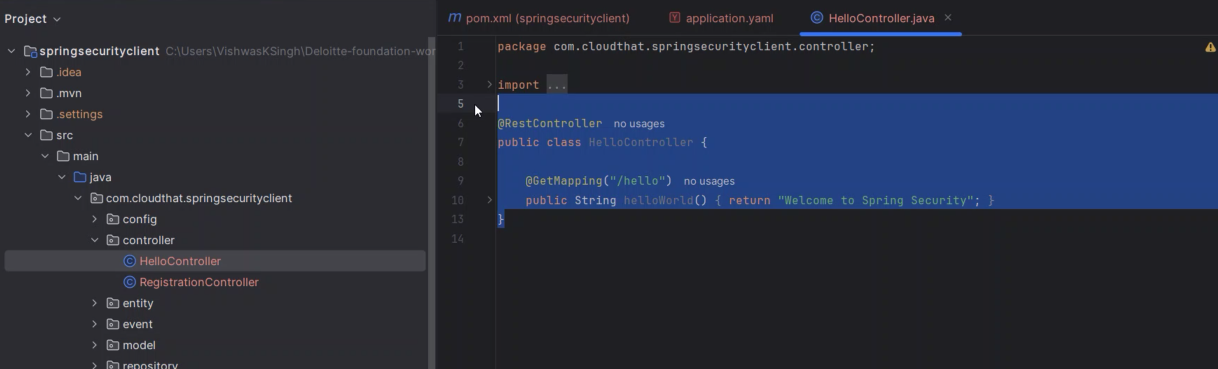




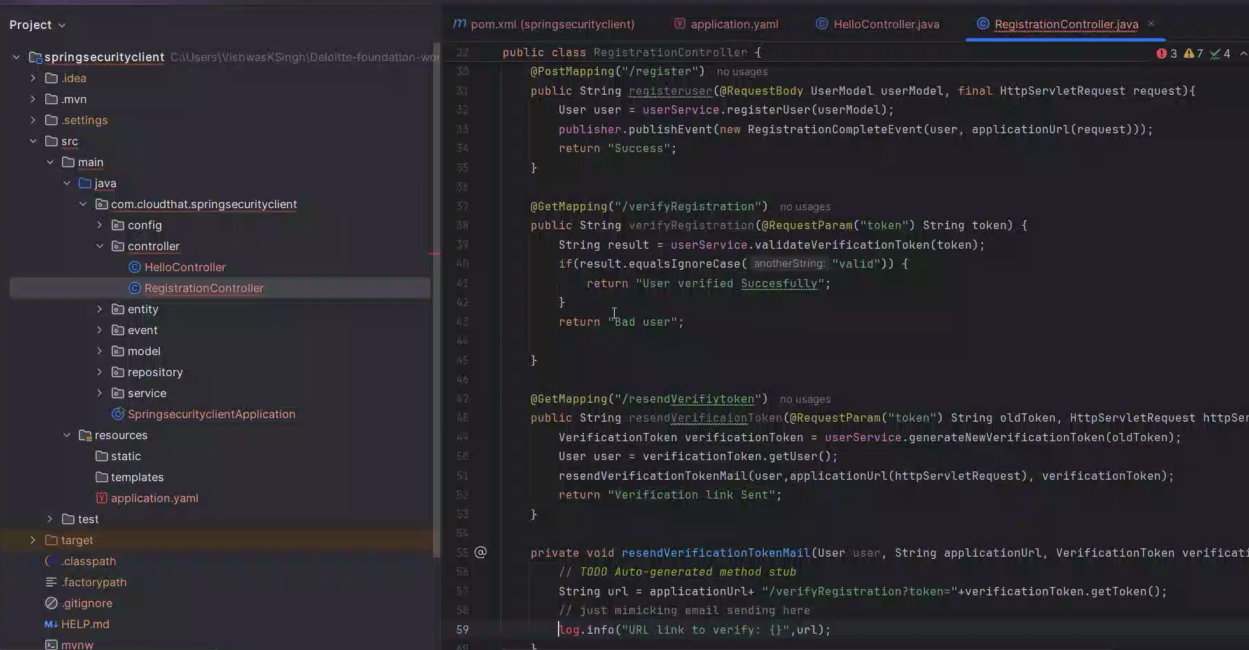
In database create a new database

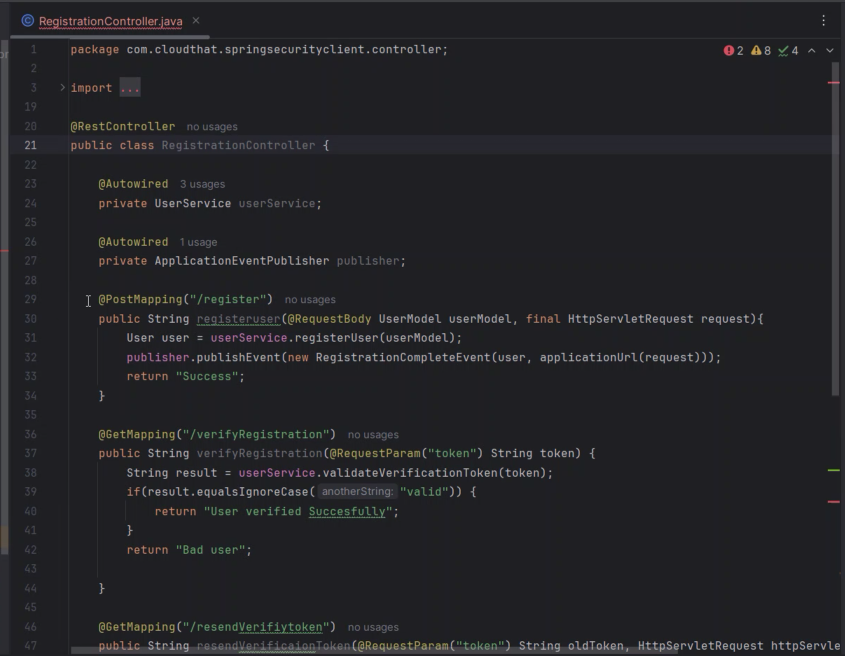
Build a controller

For testing purpose HelloController

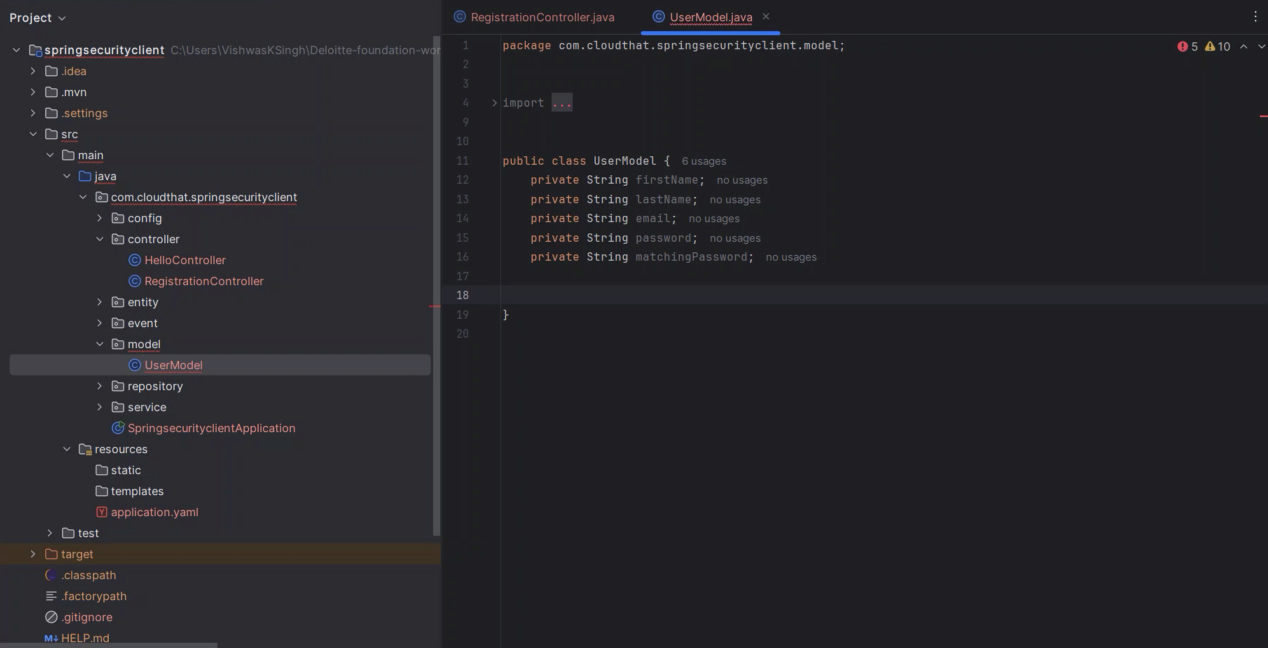


RegistrationController.java



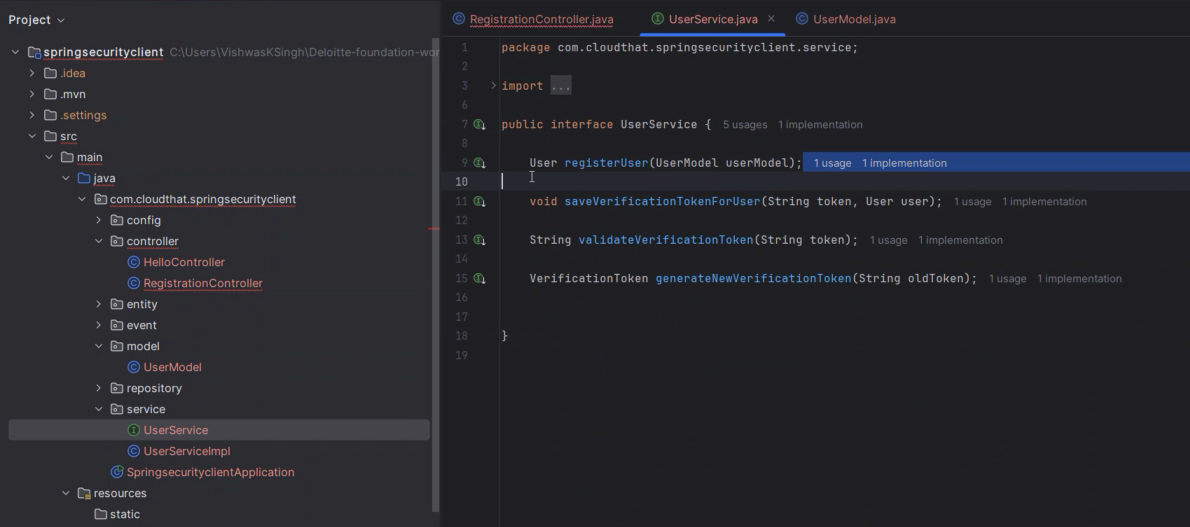


In Model package UserModel class

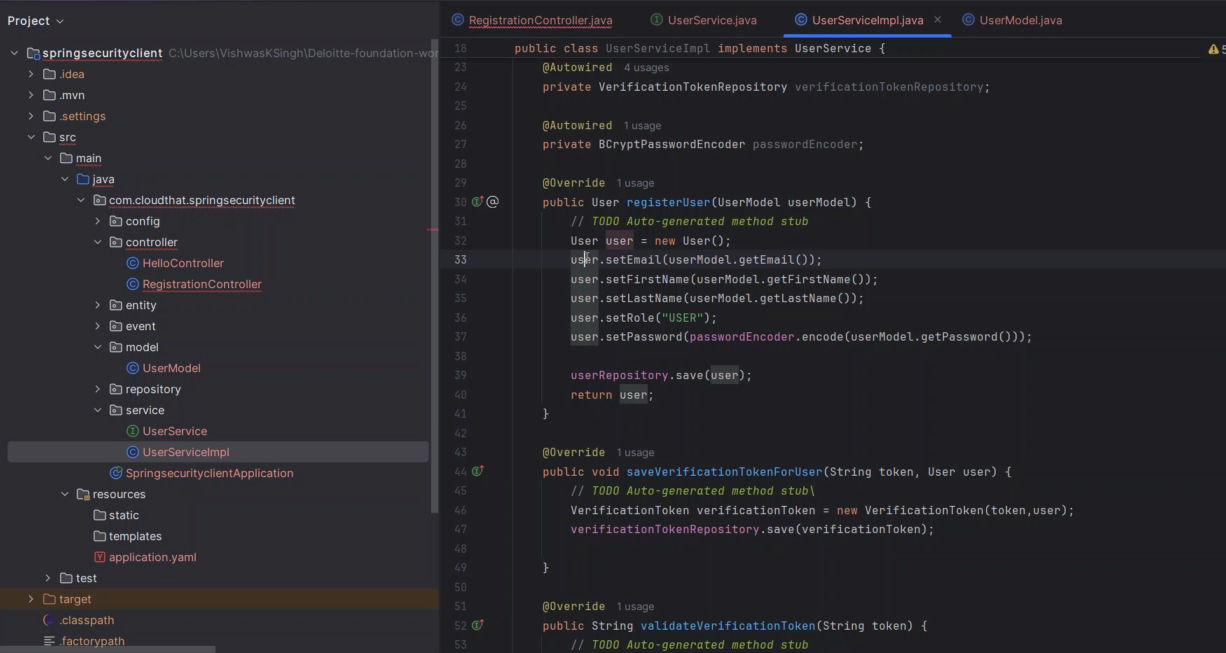


Generte getters and setters and no/all arguments constructor

Inside service package userService



userServiceImpl

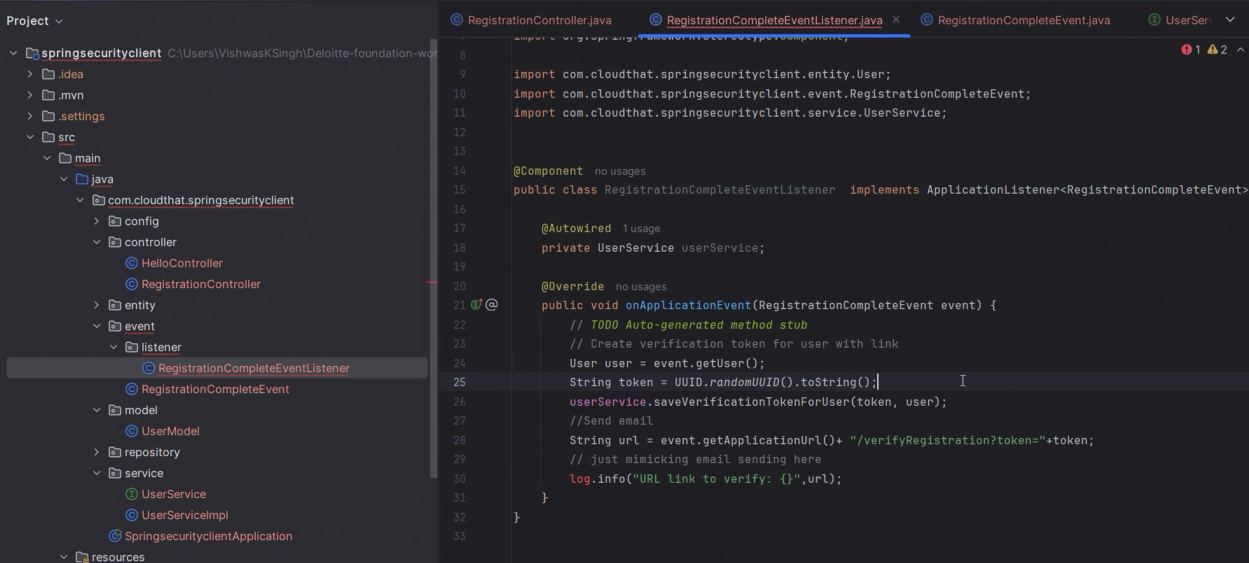


Inside event registrationCompleteEvent



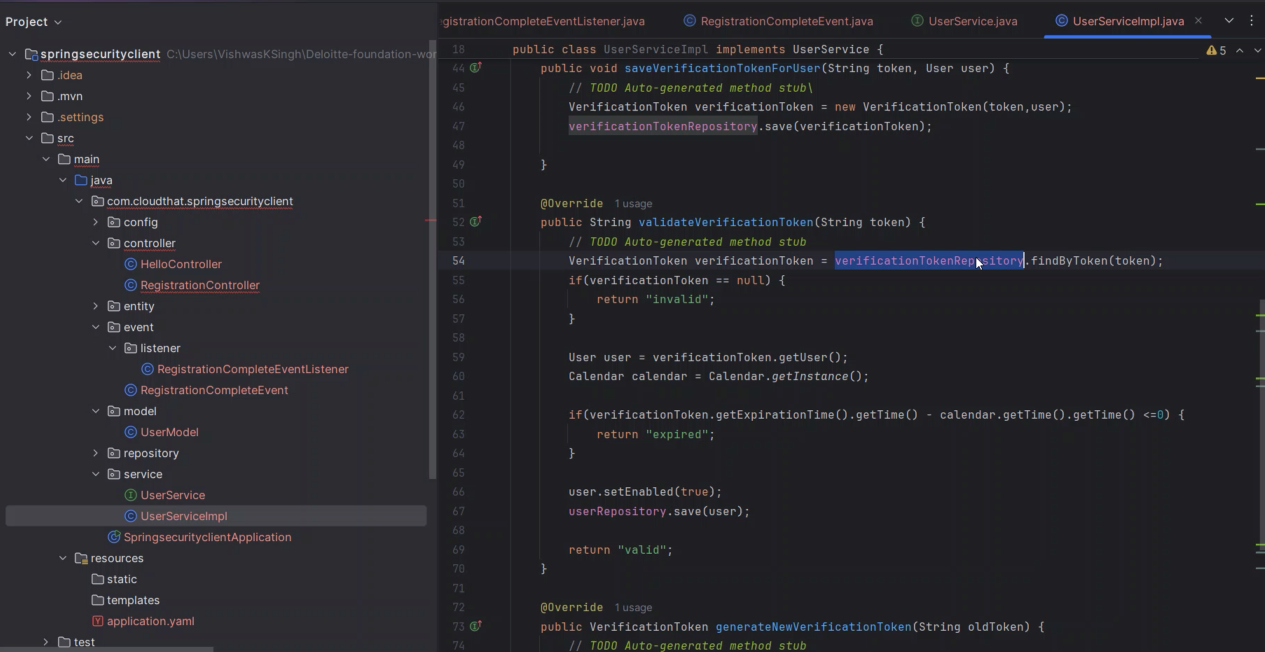
Generate getters and setters and constructor all

Inside event-listener in side listener listening the event of regiztrationCompleteEvenet

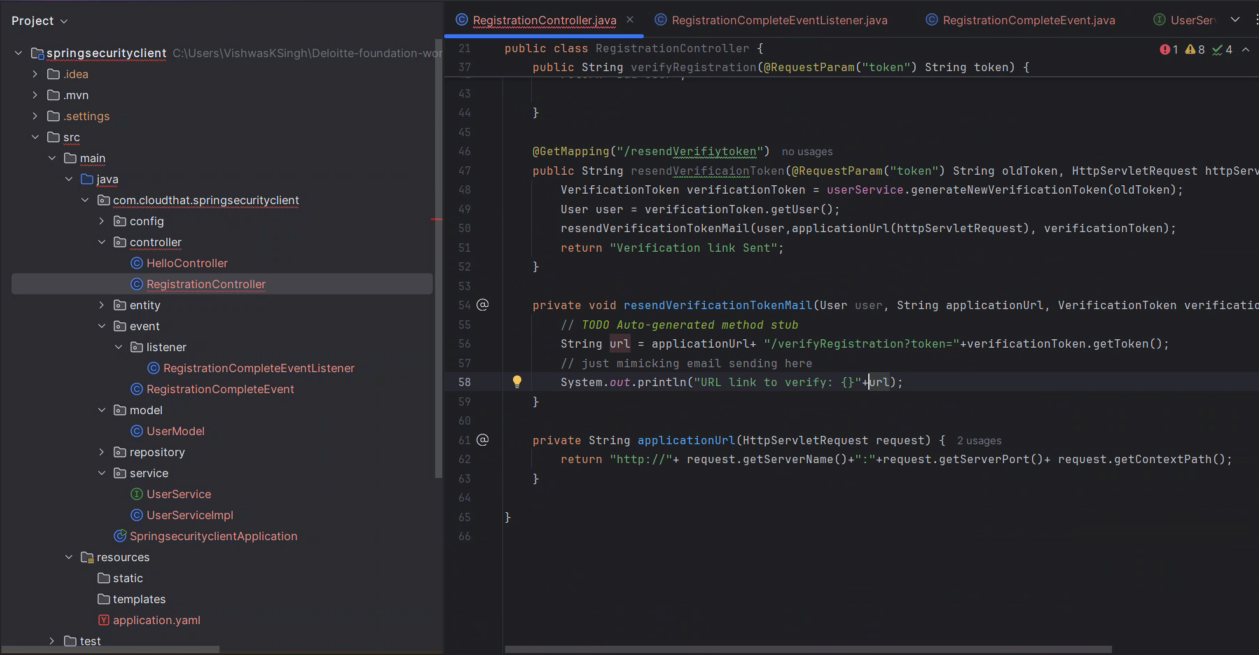


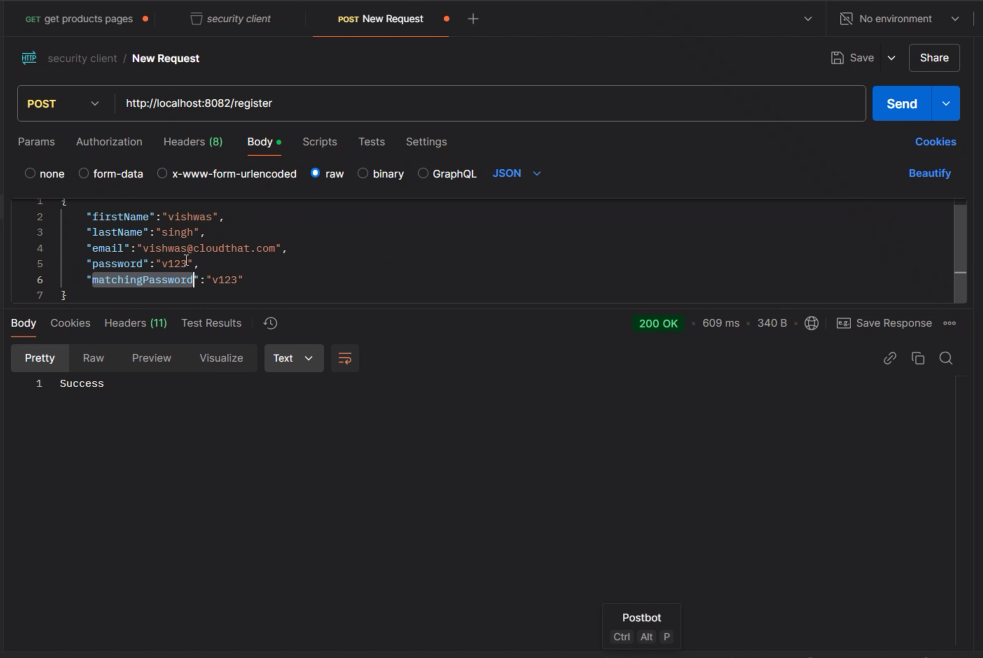
Then an token - uniqueId has to be generated. And it has expiration time also. And then we have to send an email but here instead of sending an email we are generating an url and opening it. And I the controller code we are returning “success”.

VERIFYREGISTRATION

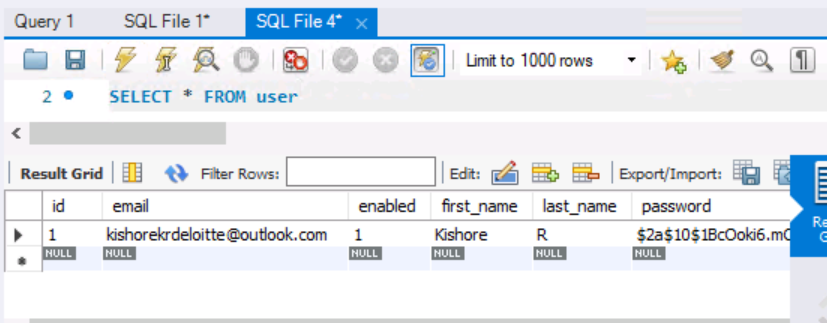


validateToken in UserServiceImpl





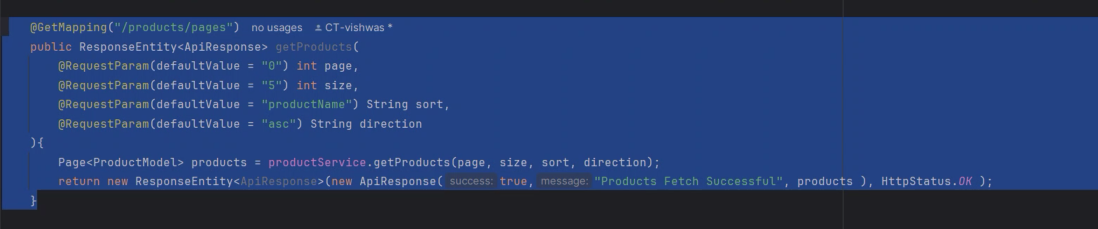
Refer sir code for above pogramming



Controller>method>service>repo

In productappV2

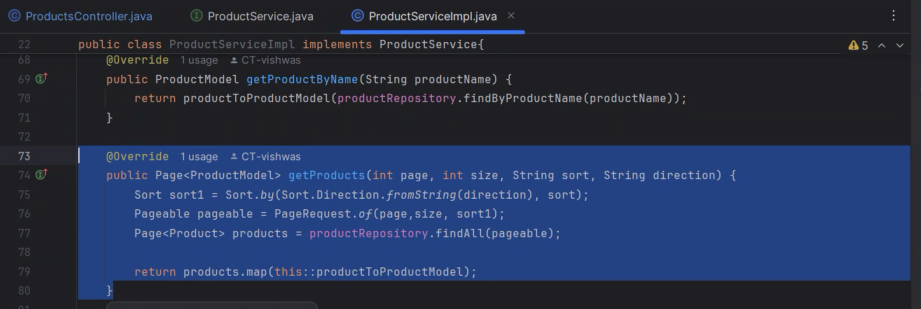
In ProductsController.java



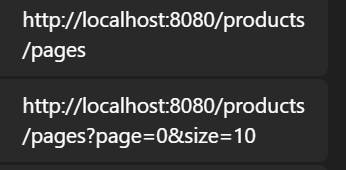
In Service package > productService



In Service package > productServiceImpl



That return line is java 8 expression.



@GetMapping("/products/pages")public ResponseEntity<ApiResponse> getProducts( @RequestParam(defaultValue = "0") int page, @RequestParam(defaultValue = "5") int size, @RequestParam(defaultValue = "productName") String sort, @RequestParam(defaultValue = "asc") String direction){ Page<ProductModel> products = productService.getProducts(page, size, sort, direction); return new ResponseEntity<ApiResponse>(new ApiResponse(true,"Products Fetch Successful", products ), HttpStatus.OK );}

Page<ProductModel> getProducts(int page, int size, String sort, String direction);

@Overridepublic Page<ProductModel> getProducts(int page, int size, String sort, String direction) { Sort sort1 = Sort.by(Sort.Direction.fromString(direction), sort); Pageable pageable = PageRequest.of(page,size, sort1); Page<Product> products = productRepository.findAll(pageable); return products.map(this::productToProductModel);}