**AuthController – List Users**

In AuthController.java -> add below code

@GetMapping(value = "/users", produces = "application/json")

    @Operation(summary = "List user api")

    public List<Account> Users(){

        return accountService.findall();

    }

In AccountService.java -> add below code

public List<Account> findall(){

  return accountRepository.findAll();

}

Remove setRole line in controller

account.setRole("ROLE\_USER");

In AccountService.java -> modify save() method

 public Account save(Account account) {

        account.setPassword(passwordEncoder.encode(account.getPassword()));

        if(account.getRole() == null){

            account.setRole("ROLE\_USER");

        }

        return accountRepository.save(account);

    }

In payload -> create a new payload -> AccountViewDTO.java

package org.studyeasy.SpringRestdemo.payload.auth;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import lombok.AllArgsConstructor;

import lombok.Getter;

import lombok.Setter;

@Setter

@Getter

@AllArgsConstructor

public class AccountViewDTO {

    @Id

    @GeneratedValue(strategy = GenerationType.SEQUENCE)

    private long id;

    private String email;

    private String role;

}

Updated AuthController.java

package org.studyeasy.SpringRestdemo.controller;

import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;

import java.util.ArrayList;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.security.authentication.AuthenticationManager;

import org.springframework.security.core.Authentication;

import org.springframework.security.core.AuthenticationException;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.ResponseStatus;

import org.springframework.web.bind.annotation.RestController;

import org.studyeasy.SpringRestdemo.model.Account;

import org.studyeasy.SpringRestdemo.payload.auth.AccountDTO;

import org.studyeasy.SpringRestdemo.payload.auth.AccountViewDTO;

import org.studyeasy.SpringRestdemo.payload.auth.TokenDTO;

import org.studyeasy.SpringRestdemo.payload.auth.UserLoginDTO;

import org.studyeasy.SpringRestdemo.service.AccountService;

import org.studyeasy.SpringRestdemo.service.TokenService;

import org.studyeasy.SpringRestdemo.util.constants.AccountError;

import org.studyeasy.SpringRestdemo.util.constants.AccountSuccess;

import io.swagger.v3.oas.annotations.Operation;

import io.swagger.v3.oas.annotations.responses.ApiResponse;

import io.swagger.v3.oas.annotations.tags.Tag;

import jakarta.validation.Valid;

import lombok.extern.slf4j.Slf4j;

@RestController

@RequestMapping("/auth")

@Tag(name = "Auth Controller", description = "Controller for Account management")

@Slf4j

public class AuthController {

    @Autowired

    private AuthenticationManager authenticationManager;

    @Autowired

    private TokenService tokenService;

    @Autowired

    private AccountService accountService;

    public AuthController(TokenService tokenService, AuthenticationManager authenticationManager) {

        this.tokenService = tokenService;

        this.authenticationManager = authenticationManager;

    }

    @PostMapping("/token")

    @ResponseStatus(HttpStatus.OK)

    public ResponseEntity<TokenDTO> token(@Valid @RequestBody UserLoginDTO userLogin) throws AuthenticationException {

        try {

            Authentication authentication = authenticationManager

                    .authenticate(new UsernamePasswordAuthenticationToken(userLogin.getEmail(), userLogin.getPassword()));

            return ResponseEntity.ok(new TokenDTO(tokenService.generateToken(authentication)));

        } catch (Exception e) {

            log.debug(AccountError.TOKEN\_GENERATION\_ERROR.toString() + ": " + e.getMessage());

            return new ResponseEntity<>(new TokenDTO(null), HttpStatus.BAD\_REQUEST);

        }

    }

    @PostMapping(value = "/users/add",  produces = "application/json")

    @ResponseStatus(HttpStatus.CREATED)

    @ApiResponse(responseCode = "400", description = "Please enter a valid email and Password length between 6 to 20 characters")

    @ApiResponse(responseCode = "200", description = "Account Added")

    @Operation(summary = "Add a new user")

    public ResponseEntity<String> addUser(@Valid @RequestBody AccountDTO accountDTO){

        try {

            Account account = new Account();

            account.setEmail(accountDTO.getEmail());

            account.setPassword(accountDTO.getPassword());

            accountService.save(account);

            return ResponseEntity.ok(AccountSuccess.ACCOUNT\_ADDED.toString());

        } catch (Exception e) {

            log.debug(AccountError.ADD\_ACCOUNT\_ERROR.toString() + ": "+e.getMessage());

            return ResponseEntity.status(HttpStatus.BAD\_REQUEST).body(null);

        }

    }

    @GetMapping(value = "/users", produces = "application/json")

    @Operation(summary = "List user api")

    @ApiResponse(responseCode = "200", description = "List of users")

    public List<AccountViewDTO> Users(){

        List<AccountViewDTO> accounts = new ArrayList<>();

        for(Account account : accountService.findall()){

            accounts.add(new AccountViewDTO(account.getId(), account.getEmail(), account.getRole()));

        }

        return accounts;

    }

}