**Getting Started with Auth Controller -Add User Api**

Create a AccountDTO In payload/auth folder and Update all 3 dtos

AccountDTO.java

package org.studyeasy.SpringRestdemo.payload.auth;

import lombok.Getter;

import lombok.Setter;

@Setter

@Getter

public class AccountDTO {

    private String email;

    private String password;

}

TokenDTO.java

package org.studyeasy.SpringRestdemo.payload.auth;

import lombok.Getter;

import lombok.Setter;

@Setter

@Getter

@AllArgsConstructor

public class TokenDTO {

    private String token;

}

UserLoginDTO.java

package org.studyeasy.SpringRestdemo.payload.auth;

import lombok.Getter;

import lombok.Setter;

@Getter

@Setter

public class UserLoginDTO {

    private String email;

    private String password;

}

Updated AuthController.java

package org.studyeasy.SpringRestdemo.controller;

import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;

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.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.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.tags.Tag;

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(@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("/users/add")

    @ResponseStatus(HttpStatus.OK)

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

        try {

            Account account = new Account();

            account.setEmail(accountDTO.getEmail());

            account.setPassword(accountDTO.getPassword());

            account.setRole("ROLE\_USER");

            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);

        }

    }

}

Output





