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
```

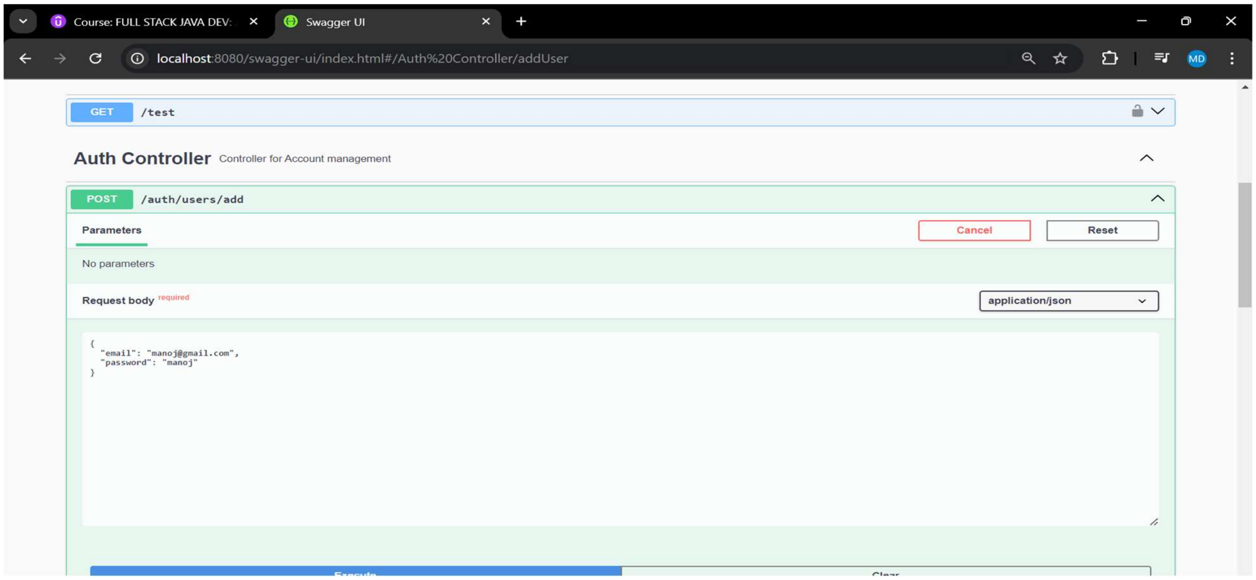
```
public class UserLoginDTO {
    private String email;
    private String password;
}
```

Updated AuthController.java

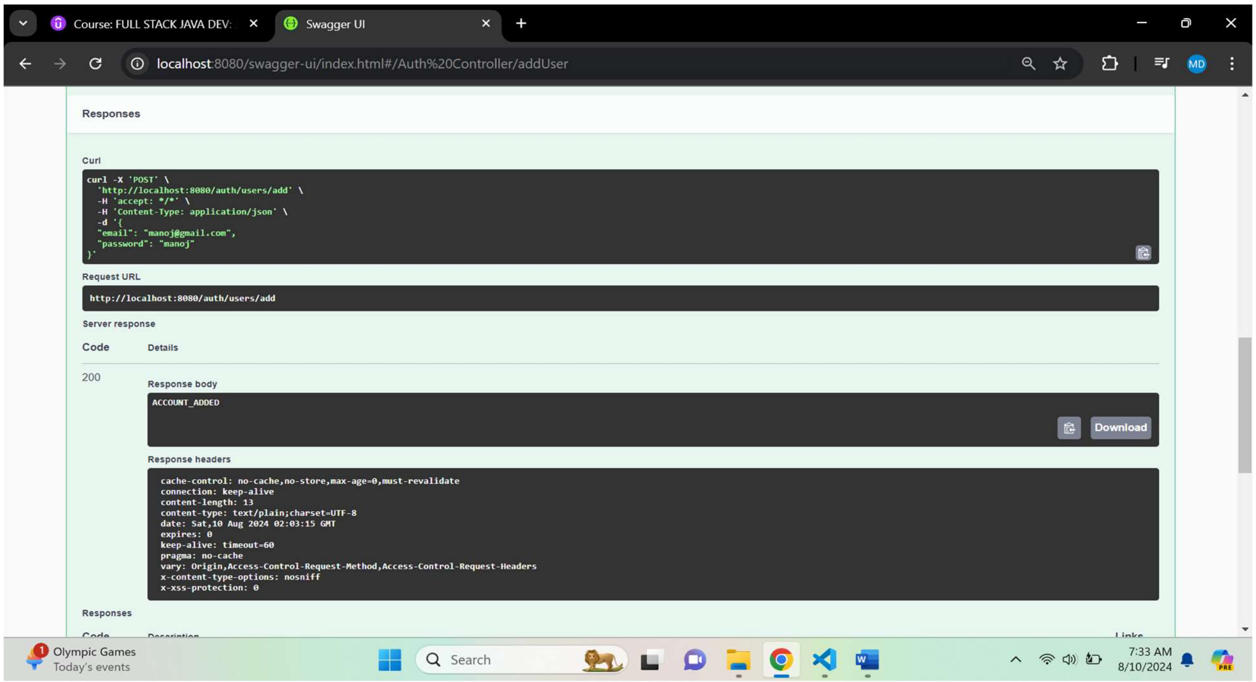
```
package org.studyeasy.SpringRestdemo.controller;

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


Course: FULL STACK JAVA DEV:
Swagger UI
localhost:8080/swagger-ui/index.html#/Auth%20Controller/addUser
GET
/test
Auth Controller
Controller for Account management
POST
/auth/users/add
Parameters
Cancel
Reset
No parameters
Request body
required
application/json

```
{
  "email": "manoj@gmail.com",
  "password": "manoj"
}
```

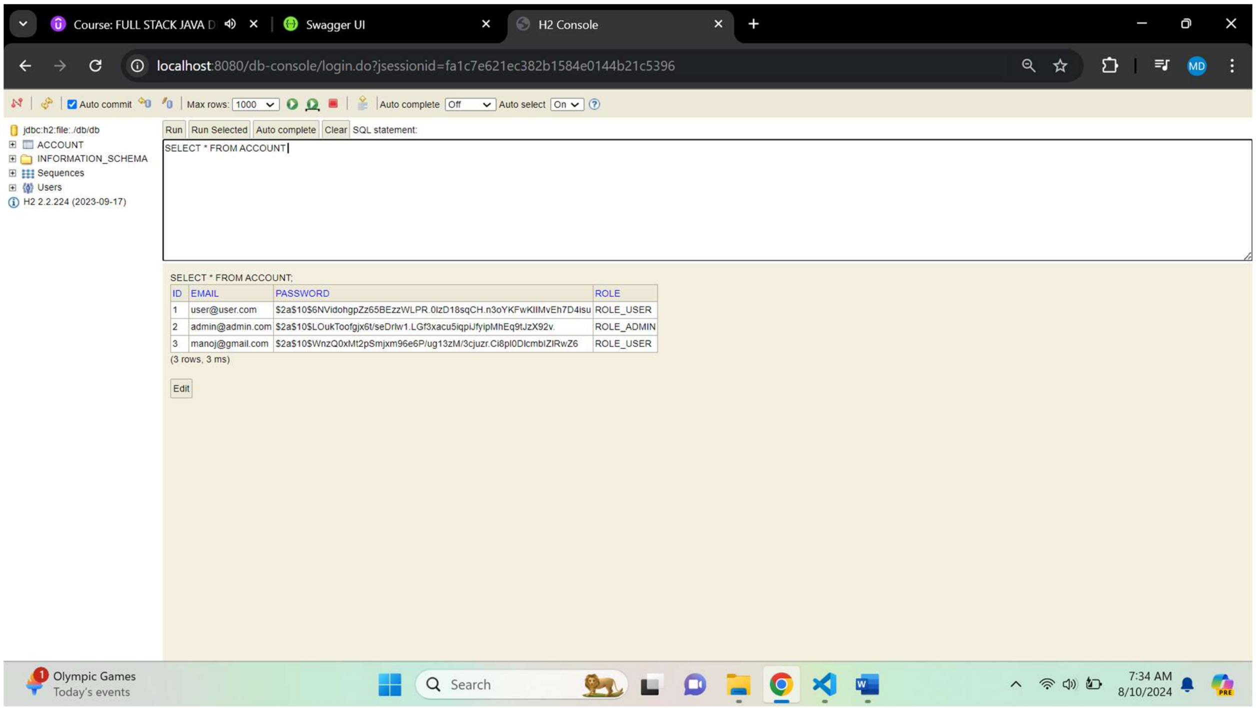

Course: FULL STACK JAVA DEV:
Swagger UI
localhost:8080/swagger-ui/index.html#/Auth%20Controller/addUser
Responses
Curl

```
curl -X 'POST' \
  'http://localhost:8080/auth/users/add' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
  "email": "manoj@gmail.com",
  "password": "manoj"
}'
```

```
http://localhost:8080/auth/users/add
```

```
ACCOUNT_ADDED
```

```
 cache-control: no-cache,no-store,max-age=0,must-revalidate
 connection: keep-alive
 content-length: 13
 content-type: text/plain;charset=UTF-8
 date: Sat,10 Aug 2024 02:03:15 GMT
 expires: 0
 keep-alive: timeout=60
 pragma: no-cache
 vary: Origin,Access-Control-Request-Method,Access-Control-Request-Headers
 x-content-type-options: nosniff
 x-xss-protection: 0
```



localhost:8080/db-console/login.do?jsessionid=fa1c7e621ec382b1584e0144b21c5396

☑ Auto commit | Max rows: 1000 | Auto complete: Off | Auto select: On

SQL statement:

```
SELECT * FROM ACCOUNT
```

SELECT * FROM ACCOUNT;

ID	EMAIL	PASSWORD	ROLE
1	user@user.com	$2a$10$6NVidohgpZz65BEzzWLPR.0IzD18sqCH.n3oYKFwKIIMvEh7D4isu	ROLE_USER
2	admin@admin.com	$2a$10$LOukToofgjx6t/seDrlw1.LGf3xacu5iqpiJfyipMhEq9tJzX92v.	ROLE_ADMIN
3	manoj@gmail.com	$2a$10$WnzQ0xMt2pSmjxm96e6P/ug13zM/3cjuzr.Ci8pl0DlcmbIZIRwZ6	ROLE_USER

(3 rows, 3 ms)