```java
public class Authentication {

   String url = "";
   String apiKey = "";
   String token = "";

   /**
    * Excercise 1: login with credentials
    * @param username
    * @param password
    * @return response
    * 1) Populate parameters and post methods with proper values
    * 2) Test method with valid and invalid values
    * 3) Print response using prettyPrint method
    * 4) Print code field from response using jsonPath.get() method
    * 5) comment out header method in response and print the output using
prettyPrint method.
    */
   public Response loginWithCredentials(String username, String password) {
      return  given()
            .baseUri(url)
            .header("Content-Type", "application/x-www-form-urlencoded;
charset=UTF-8")
            .formParam("factors[user]", username)
            .formParam("factors[password]", password)
            .param("method", "password")
            .when().post("/authentication");
   }

   /** Excercise 1 test
    * 3) Print response
    * 4) Print code field frome response
    */
   @Test
   public void Exercise1Test() {
      loginWithCredentials("", "").jsonPath().get("code");
      //ToDo: positive and negative scenario, print response, print code field,
print response with header commented out.
   }

   /**Excercise 2 login with api key
    * 1) Add parameter key with proper value
    * 2) Add parameter token with proper value
    * 3) Add endpoint /Members/me
    * 4) Test method positive and negative scenario(200, 401)
    * 5) Print the response using prettyPrint method
    */

   public Response authenticateWithApiKeyAndToken(String apiKey, String token) {
      return given()
            .baseUri(url)
            .param( "key", apiKey)
            .param("token", token)
            .when().get("/Members/me");
   }

   @Test
   public void authenticateWithApiTest() {
      //ToDo: test authenticateWithApiKeyAndToken method here
   }

   /**
    * Exercise 3
    * Use specification builder to not repeat yourself (url, apiKey, token)
    *
    */
```

```
    public RequestSpecification authenticationSpecification(){
        return given().baseUri(url)
                .param( "key", apiKey)
                .param("token", token);
    }

    public RequestSpecification postRequestSpecification() {
        return given().baseUri(url)
                .contentType("application/json")
                .queryParam( "key", apiKey)
                .queryParam("token", token);
    }

}
```

```java
public class GetRequests {

   Authentication authentication = new Authentication();


   /** Exercise 1 Boards
    * 1) Create get request for /members/me/boards endpoint
    * 2) Create method that will return id of chosen board taking board name as a
parameter
    * 3) Display permision level for choosen board
    */


   public Response displayBoards() {
      Response response = given()
            .spec(authentication.authenticationSpecification())
            .when().get("/members/me/boards");
      return response;
   }

   @Test
   public void jsonTest() {
      Response response = displayBoards();
      List<Map> boards = response.jsonPath().get();
      System.out.println(boards.stream().filter(board ->
board.get("name").equals("Agile Automation app")).findFirst().get().get("id"));
   }
   @Test
   public void displayBoardDetailsTest() {
      displayBoards().prettyPrint();
   }

   @Test
   public void displayPermissionLevelTest() {

System.out.println(displayBoards().jsonPath().get("prefs.permissionLevel[0]").toStr
ing());
   }

   @Test
   public void displayBoardIdTest() {
      System.out.println(getBoardId("Agile Automation app"));
   }


   public String getBoardId(String boardName) {

      Response response = displayBoards();
      //Simple resolution
      int i = 0;
      while (!response.jsonPath().getString("name[" + String.valueOf(i) +
"]").equals(boardName)) {
         i++;
      }
      return response.jsonPath().getString("id[" + String.valueOf(i) + "]");
      //No to simple resolution
      //List<Map> boards = response.jsonPath().get();
      //return boards.stream().filter(board -> board.get("name").equals("Agile
Automation app")).findFirst().get().get("id").toString();
   }

   /**
    * Exercise 4 Cards
    * 1) Display Card
    * 2) Display id of Card using card name
    */
```

```
   public Response getCards(String boardName) {
      return given()
            .spec(authentication.authenticationSpecification())
            .when().get("/boards/"+getBoardId(boardName)+"/cards");
   }

   public String getCardId(String cardName, String boardName) {
      Response response = getCards(boardName);
      int i = 0;
      while (!response.jsonPath().getString("name[" + String.valueOf(i) +
"]").equals(cardName)) {
         i++;
      }
      return response.jsonPath().getString("id[" + String.valueOf(i) + "]");
   }

   @Test
   public void displayCards() {
      getCards("Agile Automation app").prettyPrint();
   }

   @Test
   public void displayCardId() {
      System.out.println(getCardId("Cookies", "Agile Automation app"));
   }

   /**Exercise 5 Lists
    * Display list
    * Display list id by list name
    */

   public Response getLists(String boardName) {
      return given()
            .spec(authentication.authenticationSpecification())
            .when().get("/boards/"+getBoardId(boardName)+"/lists");
   } //"Agile Automation app"

   public String getListId(String listName, String boardName) {
      Response response = getLists(boardName);
      int i = 0;
      while (!response.jsonPath().getString("name[" + String.valueOf(i) +
"]").equals(listName)) {
         i++;
      }
      return response.jsonPath().getString("id[" + String.valueOf(i) + "]");
   }

   @Test
   public void displayLists() {
      getLists("Agile Automation app").prettyPrint();
   }

   @Test
   public void displayListsId() {
      System.out.println(getListId("To Do", "Agile Automation app"));
   }
}
```

```java
public class PostRequests {

   public GetRequests getRequests = new GetRequests();

   public Authentication authentication = new Authentication();

   public Response createBoard(String boardName) {
      return given()
            .spec(authentication.postRequestSpecification())
            .queryParam("name", boardName)
               .when().post("/boards/");
   }

   public Response createCard(String cardName, String boardName) {
      return given()
            .spec(authentication.postRequestSpecification())
            .queryParam("name", cardName)
            .queryParam("idList", getRequests.getListId("To Do", boardName))
            .when().post("/cards/");
   }

   @Test
   public void createCardTest() {
      createCard("Testowa karta", "Agile Automation app");
   }
   /**
    * Create list
    *
    */

   Response createList(String listName, String boardName) {
      return given()
            .spec(authentication.postRequestSpecification())
            .queryParam("name", listName)
            .queryParam("idBoard", getRequests.getBoardId(boardName))
            .when().post("/lists/");
   }

   @Test
   public void createListTest() {
      createList("Testowa lista", "Agile Automation app");
   }

   /**
    * Exercise: json as body
    *
    */

}
```

```java
public class PutRequests {

   public GetRequests getRequests = new GetRequests();
   Authentication authentication = new Authentication();


   public Response changeCardName(String oldCardName, String newCardName, String
boardName) {
      return given()
            .spec(authentication.authenticationSpecification())
            .queryParam("name", newCardName)
            .when()
            .put("/cards/"+getRequests.getCardId(oldCardName, boardName));
   }

   @Test
   public void changeCardNameTest() {
      changeCardName("Cookies", "Cookies updated", "Agile Automation app");
   }


   public Response moveCardToOtherList(String cardName, String listId, String
boardName) {
      return given()
            .spec(authentication.authenticationSpecification())
            .queryParam("idList", listId)
            .when()
            .put("/cards/"+getRequests.getCardId(cardName, boardName));
   }

   @Test
   public void moveCardToOtherListTest() {
      moveCardToOtherList("Cookies updated", getRequests.getListId("To Do", "Agile
Automation app"), "Agile Automation app");
   }

}
```