```java
package workshops.exercises;

import com.jayway.restassured.response.Response;
import com.jayway.restassured.specification.RequestSpecification;
import org.junit.Test;

import static com.jayway.restassured.RestAssured.given;

public class Authentication {

   String url = "https://api.trello.com/1";
   String apiKey = "";//https://trello.com/app-key
   String token =
"";//https://trello.com/1/authorize?expiration=1day&name=SinglePurposeToken&key=REP
LACE_WITH_YOUR_APIKEY
   String username = "";//Populate your trello username here
   String password = "";//Populate your trello password here

   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");
   }


   @Test
   public void loginWithCredentialsTest() {

      //Check status code
      loginWithCredentials(username, password).then().statusCode(200);
      //Check status code negative scenario(invalid password)
      loginWithCredentials(username, "Wrong password").then().statusCode(403);
      //Print response
      loginWithCredentials(username, password).prettyPrint();
      //Print code field from response
      System.out.println(loginWithCredentials(username,
password).jsonPath().getString("code"));

   }


   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() {
      //Check status code
      authenticateWithApiKeyAndToken(apiKey, token).then().statusCode(200);
      //Print response
      authenticateWithApiKeyAndToken(apiKey, token).prettyPrint();
   }
```

```
    public RequestSpecification requestSpecification(){
        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);
    }

    @Test
    public void displayDetailsAboutUser() {
        given().spec(requestSpecification()).when().get("/members/me").prettyPrint();
    }
}
```

```java
package workshops.exercises;

import com.jayway.restassured.response.Response;

import static com.jayway.restassured.RestAssured.given;

public class GetRequests {

   Authentication authentication = new Authentication();

   public Response displayBoards() {
             return given()
                   .specification(authentication.requestSpecification())
                  .when()
                  .get("/members/me/boards");

   }

   public String getBoardId(String boardName) {
      Response response = displayBoards();
      return getItemIdByName(response, boardName);
   }


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

   public String getCardId(String cardName, String boardName) {
      return getItemIdByName(getCards(boardName), cardName);
   }

   public Response getList(String boardName) {
      return given().spec(authentication.requestSpecification())
            .when().get("/boards/"+getBoardId(boardName)+"/lists");
   }

   public String getListId(String listName, String boardName) {
      return getItemIdByName(getList(boardName), listName);
   }


   public static String getItemIdByName(Response response, String itemName) {
      int i = 0;
      while (!response.jsonPath().getString("name[" + String.valueOf(i) +
"]").equals(itemName)) {
         i++;
      }
      return response.jsonPath().getString("id[" + String.valueOf(i) + "]");
   }
}
```

```java
package workshops.exercises;

import com.jayway.restassured.response.Response;
import static com.jayway.restassured.RestAssured.given;

public class PostRequests {

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

   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/");
   }

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

```java
package workshops.exercises;

import com.jayway.restassured.response.Response;
import static com.jayway.restassured.RestAssured.given;

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.requestSpecification())
            .queryParam("name", newCardName)
            .when()
            .put("/cards/"+getRequests.getCardId(oldCardName, boardName));
   }


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

```java
package workshops.exercises;

import org.testng.annotations.Test;

public class End2EndTests {

   /**
    * Exercise 1:
    * 1)Create new board with unique name
    * 2)Create 4 cards on new board
    * 3)Create new list on new board
    * 4)Rename one of the cards
    * 5)Move one of the cards to list Doing
    * 6)Move other card to list Done
    * 7)Move other card to new list
    *
    *
    * Additional tasks
    * 1)Create addCheckList method in proper class
    * 2)Add a checklist to one of the cards
    * 3)Create addAttachment method to proper class
    * 4)Add an attachment to one of the cards
    */

   GetRequests getRequests = new GetRequests();
   PostRequests postRequests = new PostRequests();
   PutRequests putRequests = new PutRequests();

   @Test
   public void end2endTest() {
      //  * 1)Create new board with unique name
      postRequests.createBoard("New board with unique name");
      // * 2)Create 4 cards on new board
      postRequests.createCard("card1", "New board with unique name");
      postRequests.createCard("card2", "New board with unique name");
      postRequests.createCard("card3", "New board with unique name");
      postRequests.createCard("card4", "New board with unique name");
      // * 3)Create new list on new board
      postRequests.createList("New list", "New board with unique name");
      // * 4)Rename one of the cards
      putRequests.changeCardName("card1", "card1Changed", "New board with unique
name");
      // * 5)Move one of the cards to list Doing
      putRequests.moveCardToOtherList("card2", getRequests.getListId("Doing", "New
board with unique name"), "New board with unique name");
      //  * 6)Move other card to list Done
      putRequests.moveCardToOtherList("card3", getRequests.getListId("Done", "New
board with unique name"), "New board with unique name");
      //  * 7)Move other card to new list
      putRequests.moveCardToOtherList("card4", getRequests.getListId("New list",
"New board with unique name"), "New board with unique name");
   }
}
```

```java
package workshops.exercises;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.annotations.AfterClass;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.DataProvider;
import org.testng.annotations.Test;

import java.util.concurrent.TimeUnit;

import static java.util.concurrent.TimeUnit.SECONDS;

public class SeleniumTest {

   private WebDriver driver;

   private String usernameLocator = ".//*[@id='user']";
   private String passwordLocator = ".//*[@id='password']";
   private String loginButtonLocator = ".//*[@id='login']";
   private String login = "html/body/div[1]/div[2]/a[1]";
   private String boardLocator =
".//*[@id='content']/div/div[2]/div/div/div[1]/ul/li[*]/a/span[2]/span[contains(.,'
%s')]";
   private String cardLocator =
".//*[@id='board']/div[1]/div/div[2]/a[*]/div[*]/span[contains(.,'%s')]";
   private String username= "";//Populate trello username here
   private String password = "";//Populate trello password here
   private String cardName = "selenium card";
   private String boardName = "selenium board";
   PostRequests postRequests = new PostRequests();


   @BeforeClass
   private void setup() {
      driver = new FirefoxDriver();
      postRequests.createBoard(boardName);
      postRequests.createCard(cardName, boardName);

      //ToDo add restAssured setup here
   }

   @AfterClass
   private void tearDown() {
      driver.quit();
   }

   @DataProvider(name = "dataProvider")
   public Object[][] dataProvider(){
      return new Object[][] {
            {boardName, cardName}

      };
   }

   @Test(dataProvider = "dataProvider")
   public void uiTestWithRestAssuredSetup(String boardName, String cardName)  {
      //Open login page
      driver.get("https://trello.com/login");
      //Populate username
      driver.findElement(new By.ByXPath(usernameLocator)).sendKeys(username);
      //Populate password
      driver.findElement(new By.ByXPath(passwordLocator)).sendKeys(password);
      //Click login button
      driver.findElement(new By.ByXPath(loginButtonLocator)).click();
```

```
      //Wait for page to be loaded
      driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
      //Verify if new board was added by rest assured setup.
      assert (driver.findElement(new By.ByXPath(String.format(boardLocator,
boardName))).getText().equals(boardName));
      //Go to new board
      driver.findElement(new By.ByXPath(String.format(boardLocator,
boardName))).click();
      //Check if card was added by rest assured setup.
      assert (driver.findElement(new By.ByXPath(String.format(cardLocator,
cardName))).getText().equals(cardName));
   }
}
```