Online Survey System
JUnit Testing

Code :

```
package com.survey;
import static org.junit.jupiter.api.Assertions.assertEquals;
import java.util.ArrayList;
import java.util.List;
import org.junit.jupiter.api.Order;
import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import com.survey.bean.Login;
import com.survey.bean.SurveyQuestion;
import com.survey.bean.Topics;
import com.survey.bean.Users;
import com.survey.service.ReportService;
import com.survey.service.SurveyService;
import com.survey.service.TopicService;
import com.survey.service.UserService;
@SpringBootTest
class OnlineSurveyApplicationTests {
@Autowired
        UserService userService;

        @Autowired
        TopicService topicService;

        @Autowired
        SurveyService surveyService;

        @Autowired
        ReportService reportService;


        boolean actualResult;
        boolean expectedResult;

        List<Users> userList1 = new ArrayList<>();
        List<Users> userList2 = new ArrayList<>();

        List<Topics> topicList1 = new ArrayList<>();
        List<Topics> topicList2 = new ArrayList<>();
```

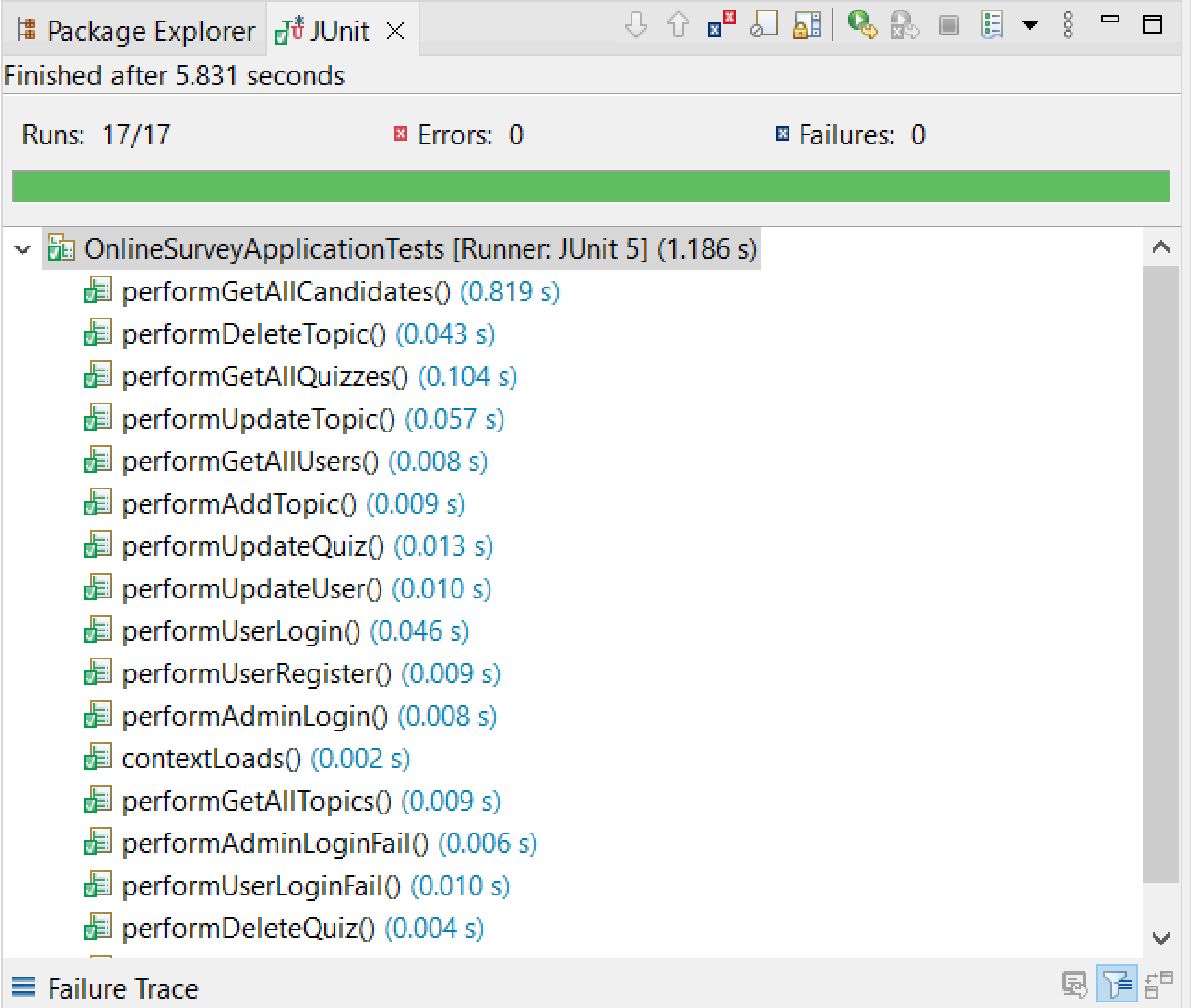
```
    List<SurveyQuestion> questionList1 = new ArrayList<>();
    List<SurveyQuestion> questionList2 = new ArrayList<>();

    Users user1 = new Users(211, "siva", "balan", "siva@gmail.com", "90877583637", "siv", "User",
"11955");
    Users user2 = new Users(212, "jeban", "ignesh", "jeban@gmail.com", "7845762349", "JI", "User",
"19151");
    Users admin1 = new Users(21, "hari", "prasath", "hari@gmail.com", "8945689034", "Vj", "Admin",
"160");
    Users admin2 = new Users(22, "vijay", "kumar", "vk@gmail.com", "9088746754", "vk", "Admin",
"18");

    Login response1 = new Login("User Login Successfully");
    Login response2 = new Login("Admin Login Successfully");
    Login response3 = new Login("Login Fail");

    Topics topic1 = new Topics(4,"ANGULAR");
    Topics topic2 = new Topics(4,"JS");

    SurveyQuestion survey2 = new SurveyQuestion(5,1,"Angular JS is perfect for?","SPA","
MPA","DPA","CPA","SPA");
    SurveyQuestion  survey3 = new SurveyQuestion(5,1,"Angular JS is perfect for?","SPA","
MPA","DPA","CPA","SPA");
    @Test
    @Order(1)
    void contextLoads() {
    }
    @Test
    @Order(2)
    void performUserLogin() {
        Login login1 = userService.login(user1);
        response1.equals(login1);
    }

    @Test
    @Order(3)
    void performUserLoginFail() {
        Login login2 = userService.login(user2);
        response3.equals(login2);
    }
    @Test
    @Order(4)
    void performAdminLogin() {
        Login login3 = userService.login(admin1);
        response2.equals(login3);
    }
    @Test
    @Order(5)
    void performAdminLoginFail() {
```

```
		Login login4 = userService.login(admin2);
		response3.equals(login4);
	}
	@Test
	@Order(6)
	void performUserRegister() {
		Users user3 = new
Users(121,"KL","Rahul","klr@gmail.com","9089786756","KLR","User","11968");
		expectedResult = true;
		actualResult = userService.registerUser(user3);
		assertEquals(expectedResult, actualResult);

	}

	@Test
	@Order(7)
	void performUpdateUser() {
		Users user4 = new
Users(143,"elon","Musk","Musk@gmail.com","8906754328","Emusk","User","11945");
		expectedResult = true;
		actualResult = userService.UpdateUser(user4);
		assertEquals(expectedResult, actualResult);
	}

	@Test
	@Order(8)
	void performDeleteUser(){
		expectedResult = true;
		try {
			actualResult = userService.deleteUser((long) 121);
		} catch (Exception e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		assertEquals(expectedResult, actualResult);
	}
	@Test
	@Order(9)
	void performGetAllUsers(){
		userList1=userService.getAllUsers();
		userList2.add(user1);
		userList2.add(admin1);
		userList2.equals(userList1);
	}
	@Test
	@Order(10)
	void performGetAllCandidates(){
		userList1=userService.getAllCandidates();
		userList2.add(user1);
```

```
              userList2.add(admin1);
              userList2.equals(userList1);
         }
     @Test
     @Order(11)
     void performAddTopic() {
         expectedResult=true;
         actualResult=topicService.addTopic(topic1);
          assertEquals(expectedResult,actualResult);
     }
     @Test
     @Order(12)
     void performUpdateTopic() {
         expectedResult=true;
         actualResult=topicService.updateTopic(topic2);
          assertEquals(expectedResult,actualResult);
     }

     @Test
     @Order(13)
     void performDeleteTopic() throws Exception {
         expectedResult=true;
         actualResult=topicService.deleteTopics((long) 4);
          assertEquals(expectedResult,actualResult);
     }

         @Test
         @Order(14)
         void performGetAllTopics(){
              topicList1=topicService.getAllTopics();
              topicList2.add(topic1);
              topicList2.add(topic2);
              topicList2.equals(topicList1);
         }



         @Test
    @Order(15)
    void performUpdateQuiz() throws Exception {
         expectedResult=true;
         actualResult=surveyService.updateSurvey(survey2);
          assertEquals(expectedResult,actualResult);
    }

    @Test
    @Order(16)
    void performDeleteQuiz() throws Exception {
         expectedResult=true;
```

```
        actualResult=surveyService.deleteSurvey((long) 5);
         assertEquals(expectedResult,actualResult);
 }
        @Test
        @Order(17)
        void performGetAllQuizzes(){
                questionList1=surveyService.getAllQuizzes();
                questionList2.add(survey2);
                questionList2.add(survey3);
                questionList2.equals(questionList1);
        }
}
```

Output :



```
Finished after 5.831 seconds

Runs:  17/17          Errors:  0          Failures:  0

OnlineSurveyApplicationTests [Runner: JUnit 5] (1.186 s)
    performGetAllCandidates() (0.819 s)
    performDeleteTopic() (0.043 s)
    performGetAllQuizzes() (0.104 s)
    performUpdateTopic() (0.057 s)
    performGetAllUsers() (0.008 s)
    performAddTopic() (0.009 s)
    performUpdateQuiz() (0.013 s)
    performUpdateUser() (0.010 s)
    performUserLogin() (0.046 s)
    performUserRegister() (0.009 s)
    performAdminLogin() (0.008 s)
    contextLoads() (0.002 s)
    performGetAllTopics() (0.009 s)
    performAdminLoginFail() (0.006 s)
    performUserLoginFail() (0.010 s)
    performDeleteQuiz() (0.004 s)
```