CS 320 Project One Guidelines and Rubric

Competency

In this project, you will demonstrate your mastery of the following competency:

- Create unit tests using code to uncover errors

Scenario

You are a software engineer for Grand Strand Systems, a software engineering company that focuses on developing and testing back-end services. You've been given an assignment to develop a mobile application for a customer. The customer will provide you with the requirements. Your job is to code up the application and provide unit tests to verify that it meets the customer's requirements. You will be delivering the contact, task, and appointment services. The purpose of these services is to add, update, and delete contact, task, and appointment objects within the application.

Directions

ContactService, TaskService, and AppointmentService Files

For this assignment, you will incorporate the code and unit tests that you have developed for the mobile application. First, you developed the contact service and contact object, which you completed in the Module Three milestone. Second, you developed the task service and task object, which you completed in the Module Four milestone. Last, you developed the appointment service and appointment object, which you completed in Module Five milestone. **Any feedback received on these assignments should be incorporated prior to submitting the files for this final project**.

You have been asked to code up a mobile application for a client and provide unit tests to verify that it meets the customer's requirements. In order to do so, you must complete the following:

1. **Contact Service**: In the Module Three milestone, you developed the contact service. The contact service used in-memory data structures to support storing contacts (no database required). In addition, there was no UI for this assignment. You verified the contact service through JUnit tests. The contact service contained a contact object along with the contact service. The requirements were as follows:
   a. Contact Class Requirements
      i. The contact object shall have a required unique contact ID String that cannot be longer than 10 characters. The contact ID shall not be null and shall not be updatable.
      ii. The contact object shall have a required firstName String field that cannot be longer than 10 characters. The firstName field shall not be null.
      iii. The contact object shall have a required lastName String field that cannot be longer than 10 characters. The lastName field shall not be null.
      iv. The contact object shall have a required phone String field that must be exactly 10 digits. The phone field shall not be null.

v. The contact object shall have a required address field that must be no longer than 30 characters. The address field shall not be null.

b. Contact Service Requirements

i. The contact service shall be able to add contacts with unique ID.

ii. The contact service shall be able to delete contacts per contactId.

iii. The contact service shall be able to update contact fields per contactId. The following fields are updatable:

1. firstName
2. lastName
3. PhoneNumber
4. Address

2. **Task Service**: In the Module Four milestone, you developed the task service. The task service used in-memory data structures to support storing tasks (no database required). In addition, there was no UI for this assignment. You verified the task service through JUnit tests. The task service contained a task object along with the task service. The requirements were as follows:

a. Task Requirements

i. The task object shall have a required unique task ID String that cannot be longer than 10 characters. The task ID shall not be null and shall not be updatable.

ii. The task object shall have a required name String field that cannot be longer than 20 characters. The name field shall not be null.

iii. The task object shall have a required description String field that cannot be longer than 50 characters. The description field shall not be null.

b. Task Service Requirements

i. The task service shall be able to add tasks with a unique ID.

ii. The task service shall be able to delete tasks per taskId.

iii. The task service shall be able to update task fields per taskId. The following fields are updatable:

1. name
2. description

3. **Appointment Service**: In the Module Five milestone, you developed the appointment service. The appointment service used in-memory data structures to support storing appointments (no database required). In addition, there was no UI for this assignment. You verified the appointment service through JUnit tests. The appointment service contained an appointment object along with the appointment service. The requirements were as follows:

a. Appointment Requirements

        i. The appointment object shall have a required unique appointment ID String that cannot be longer than 10 characters. The appointment ID shall not be null and shall not be updatable.
        ii. The appointment object shall have a required appointment Date field. The appointmentDate field cannot be in the past. The appointmentDate field shall not be null. *Note: Use java.util.Date for the appointmentDate field and use before(new Date()) to check if the date is in the past.*
        iii. The appointment object shall have a required description String field that cannot be longer than 50 characters. The description field shall not be null.

    b. Appointment Service Requirements
        i. The appointment service shall be able to add appointments with a unique appointmentId.
        ii. The appointment service shall be able to delete appointments per appointmentId.

Specifically, the following rubric criteria must be addressed:

- Verify the **Contact class** meets the requirements through JUnit tests.
- Verify the **ContactService class** meets the requirements through JUnit tests.
- Verify the **Task class** meets the requirements through JUnit tests.
- Verify the **TaskService class** meets the requirements through JUnit tests.
- Verify the **Appointment class** meets the requirements through JUnit tests.
- Verify the **AppointmentService class** meets the requirements through JUnit tests.
- Ensure the **test coverage** for the java files has 80% coverage or higher.

When you are ready to begin your work, access Codio via the Codio module.

Note: You will need to upload and download files to the Codio virtual lab using Microsoft OneDrive, following the instructions in the Codio Student Guide located in the Codio module. It is also recommended that you periodically save your work to Microsoft OneDrive, following the instructions in the Codio Student Guide.

What to Submit

To complete this project, you must submit a **ZIP folder** containing the following deliverables:

ContactService, TaskService, and AppointmentService Files

- ContactService
    - Contact.java
    - ContactService.java
    - ContactTest.java

- ContactServiceTest.java

- TaskService
    - Task.java
    - TaskService.java
    - TaskTest.java
    - TaskServiceTest.java

- AppointmentService
    - Appointment.java
    - AppointmentService.java
    - AppointmentTest.java
    - AppointmentServiceTest.java

Project One Rubric

Criteria	Exceeds Expectations	Meets Expectations	Partially Meets Expectations	Does Not Meet Expectations	Value
Contact Class	Verifies the Contact class meets all five requirements (100%)	Verifies the Contact class meets three or four requirements (85%)	Verifies the Contact class meets one or two requirements (55%)	Does not verify the Contact class meets any requirements, or does not attempt criterion (0%)	20
Contact Service Class	Verifies the Contact Service class meets all three requirements (100%)	Verifies the Contact Service class meets two requirements (85%)	Verifies the Contact Service class meets at least one requirement (55%)	Does not verify the Contact Service class meets any requirements, or does not attempt criterion (0%)	15
Task Class	Verifies the Task class meets all three requirements (100%)	Verifies the Task class meets two requirements (85%)	Verifies the Task class meets at least one requirement (55%)	Does not verify the Task class meets any requirements, or does not attempt criterion (0%)	15
Task Service Class	Verifies the Task Service class meets all three requirements (100%)	Verifies the Task Service class meets two requirements (85%)	Verifies the Task Service class meets at least one requirement (55%)	Does not verify the Task Service class meets any requirements, or does not attempt criterion (0%)	15

Criteria	Exceeds Expectations	Meets Expectations	Partially Meets Expectations	Does Not Meet Expectations	Value
Appointment Class	Verifies the Appointment class meets all three requirements (100%)	Verifies the Appointment class meets two requirements (85%)	Verifies the Appointment class meets at least one requirement (55%)	Does not verify the Appointment class meets any requirements, or does not attempt criterion (0%)	15
Appointment Service Class	Verifies the Appointment Service class meets two requirements (100%)	Verifies the Appointment Service class meets at least one requirement (85%)	N/A	Does not verify the Appointment Service class meets any requirements, or does not attempt criterion (0%)	10
Test Coverage	Ensures that the test coverage for the JAVA files has 80% coverage or higher (100%)	Ensures that the test coverage for the JAVA files has 65–79% coverage (85%)	Ensures that the test coverage for the java files has 50–64% coverage (55%)	Ensures that the test coverage is below 50%, or does not attempt criterion (0%)	10
				Total:	100%