International Institute of Information Technology, Hyderabad

(Deemed to be University)

CS6.401 Software Engineering – Spring 2024

Mid Semester Examination

Max. Time: 1.5 Hr **Max. Marks: 35**

Instructions to the students

1. You are **allowed** to bring up to **2 sheets (A4 size, total of 4 pages including back-to-back)** of **handwritten notes**. You are not required to submit these notes along with your answer sheets.
2. Any form of printed/scanned materials, digital notes and photocopies are not allowed.
3. Borrowing notes from other students in the examination hall is prohibited.
4. Answers written (or figures made) using pencils won't be considered for evaluation.
5. Please read the descriptions of the questions (scenarios) carefully. While answering please ensure that any assumptions made are clearly stated.
6. There are a total of five questions. Please note that first two questions are MCQs (requires 2-3 lines explanation) and carries 2.5 marks each. Rest three questions carry 10 marks each.

Good Luck

Congrats on being selected as a consultant of the WellDine Software Design Team

"WellDine" (Wellness and Dining) is a national chain of imaginary health and nutrition establishments. The company wants to increase access to healthy meals in the comfort of people's homes and raise awareness of healthy behaviors among a variety of demographics. The organization began in a tiny community and is currently preparing to spread across the nation. WellDine has two primary goals as part of its future expansion objectives:

- Firstly, WellDine seeks to launch a WellDine-Daily platform (or daily platform) that will offer a range of in-app health functionalities that cater to the diverse needs of users looking to maintain or improve their health and wellness. These will include: i) the ability to register as a user (mentor/coach or seeker); ii) allow users to add their wellness goals, dietary restrictions, food preferences, pre-existing conditions, etc.; iii) ability for users to add their daily calorie intakes, hydration intakes, logging meals along with their ingredients; iv) Allow users to monitor health levels by providing support for integrating with different wearables for adding information related to number of steps, heart rate, stress levels and other vital parameters; v) provide support for users to subscribe to training/wellness plans offered by different mentors/coaches by paying a fee (payment can be made through different mechanism like credit card, debit card, UPI, etc.); and vi) allow users to receive reductions on subscription fee based on their wellness progress which is gamified through rankings in a common and diet specific leader boards.

- Secondly, WellDine plans to integrate end-to-end food delivery system support to provide a one-stop location for the users to locate, order and receive healthy food delivered to their doorstep. This food delivery system, WellDine-Delivery (or Delivery), will cater to different sets of stakeholders: i) it will allow seekers (regular

users) to search for health food outlets based on their diet preferences, add food items to their cart and place orders by making payments and providing a delivery address; ii) allows restaurant outlets to register to the platform by specifying the broad diet category they cater to along with the food items and their pricing; and iii) provides functionality which allows delivery agents to register to the platform, get notified on orders and receive payments on successful completion of their deliveries.

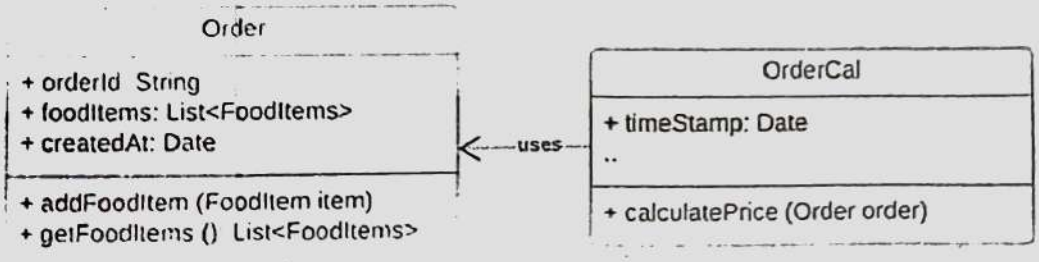
WellDine has engaged a team of approximately 15 software engineers to develop the two systems, which have been split into two teams, namely the e-commerce and in-store teams. These teams have already commenced development of some components of the software. Recognizing that poor design practices are a common issue among companies struggling with maintenance (now that AI tools also may add up to this), WellDine's board has hired you as an expert consultant to review their existing design and provide guidance to the teams regarding the new design of certain subsystems.

As the consultant, you have been allotted 90 minutes to complete several tasks, of which five are particularly critical. You must complete these tasks within the allocated timeframe, as your services have been billed accordingly. Each task is assigned a specific number of points, and the cumulative score of all the tasks will determine your final payment (based on total points), with a maximum of 35 points.

1. In the WellDine-Delivery system class diagram, you notice that there is an *Order* class and an *OrderCal* class as represented as below. What is your view on this design? (2.5 points)

   a. This violates the creator design principle
   b. Information expert principle is violated
   c. This is a classic instance of abstraction smell
   d. None of the above

   Describe the reason in 2-3 lines.



2. The daily team has implemented one module, *HealthManager,* to manage everything related to the health parameters of the users. You wanted to check the codebase, and you start noticing that a class, *HealthMonitor*, with attributes like user, heartRate, stressLevel, totalSleepTime, REMSleepTime (parameter related to sleep), numberOfSteps, etc. and operations – createAlert(), setheartRate(double value), etc. contains a significant amount of code. As a preliminary step, you decide to gather some metrics. You notice that the class contains around 400 lines of code and the createAlert() contains switch case for generating alerts for different parameter types (based on heartrate, stressLevel, etc).

Further, each switch condition consists of 10 lines of code for generating the alert. What are your quick observations? Select all that apply. (2.5 points)

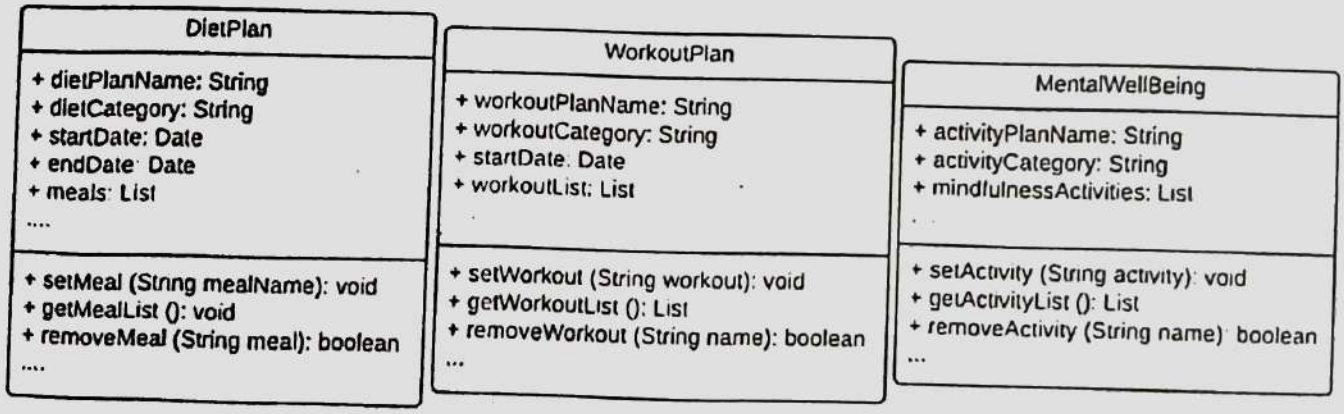
a. Cyclometric complexity will indicate that this code is very high in volume
b. Halstead metrics will potentially indicate an upward trend in program length.
c. The number of conditional switches indicate higher cyclometric complexity
d. The lines of code also indicate a need for refactoring

Describe the reason in 2-3 lines

3. The WellDine-Delivery system consists of multiple subsystems, such as i) user management, ii) restaurant/caterer management, iii) order subsystem, iv) food delivery subsystem, and v) payment subsystem, among the many other subsystems. As a first step model, the delivery management subsystem. *You must capture the structural and behavioural* aspects using the necessary diagram(s). Identify at least 4-5 classes and 1 behaviour flow. Feel free to make assumptions, but please do mention them. All the key relationships should be captured in the diagram (simple dependency relationships may be omitted). Also, provide a 2-line description of each identified class and their relationships. (10 points)

Note: You only have to focus on 4-5 important classes that are most important to develop the food delivery subsystem. Clearly state any assumptions made. These classes may or may not include classes from other subsystems.

4. The Daily team has developed the Wellness management subsystem for the Daily system, and they want you to quickly review their design as well as their code. The goal of the Wellness management system is to provide users (seekers) with the ability to generate wellness plans based on their details, such as different health parameters, diet restrictions, goals, etc. Each wellness plan is a combination of a diet plan, workout plan and activities for mental well-being. The figure below represents the classes: 1) *WorkoutPlan, 2) DietPlan*, and 3) *MentalWellBeing*. Further, the snippet gives an overview of the *WellnessPlan* class.



Code Snippet of WellnessPlan Class:

```
import java.util.List;

public class WellnessPlan {
   String planId;
   String userName;
   private DietPlan diet;
   private WorkOutPlan workout;
   private MentalWellBeing activities;


  public WellnessPlan(String dietPlanName, String dietCategory, List mealList,
              String workOutPlanName, String workOutPlanCateogry,
                        List workOuts,String activityPlanName,
                        String activityPlanCategory, List activs,
                    String planId, String userName)
                  {

                   diet = new DietPlan(dietPlanName, dietCategory, mealList); // create   dietplan object
                   workout = new WorkOutPlan(workOutPlanName, workOutPlanCateogry, workOuts)
                   ; // create workoutplan object
                   activities = new MentalWellBeing(activityPlanName, activityPlanCategory, activs); //create
                   activity object
                   this.planId = planId;
                   this.userName = userName;
       }

       public void customizeWellnessPlan (String cateogry)
       {
            if (cateogry.equals("keto"))
              this.diet.meals.add("low Carb");
              //any other logic for customizing the workout and activities

            else if (cateogry.equals("vegan"))
              this.diet.meals.add("tofu rice");
              // any other logic for customizing the workout and activities around this diet

            else if (cateogry.equals("Whole30"))
            {
```

```
            this.diet.meals.add("apples, oranges");
            //any other logic for customizing the workound and activites around this diet
        }
    //other diet categories to be added
      }
    }
```

Are there any code smells associated with the above snippet? Does it translate to any design smell? If yes, list the identified code and design smells (3 points). Mention what qualities are affected by these smells (2 points) and provide the pseudocode (when necessary, eg: code smells) and UML diagrams (for design smells) (5 points)
(**Total:** 10 points)

Note: you are free to make assumptions (but do mention them) that may drive you towards an appropriate solution. You don't have to write the pseudocode of the entire refactored solution; instead, use pseudocode just to highlight code-level changes (in the event of code smells).

5. The WellDine team has one final query before you leave: The WellDine-Delivery system has different categories of restaurants that cater to different diet categories like keto, vegan, low-carb, etc. When a restaurant wants to sign up, they *have to sign up under one main diet category*. Any number of restaurants can be added under a category. Fitness enthusiasts can subscribe to specific restaurants as well as categories for notifications. Moreover, When a restaurant is added to the system, fitness enthusiasts are notified about the new restaurant presenting some highlighted menu items. Can you help the team with the implementation? (**Total:** 10 points)
    a. Describe your solution using UML diagrams or pseudocode (as and when necessary) (5 points)
    b. Mention any assumptions taken and provide brief description of all the involved classes and interfaces (if any) (3 points)
    c. How does your solution satisfy some qualities such as extensibility, changeability, and understandability? (2 points)

Now that you have completed the task, it's time to wait to know your final points!!!