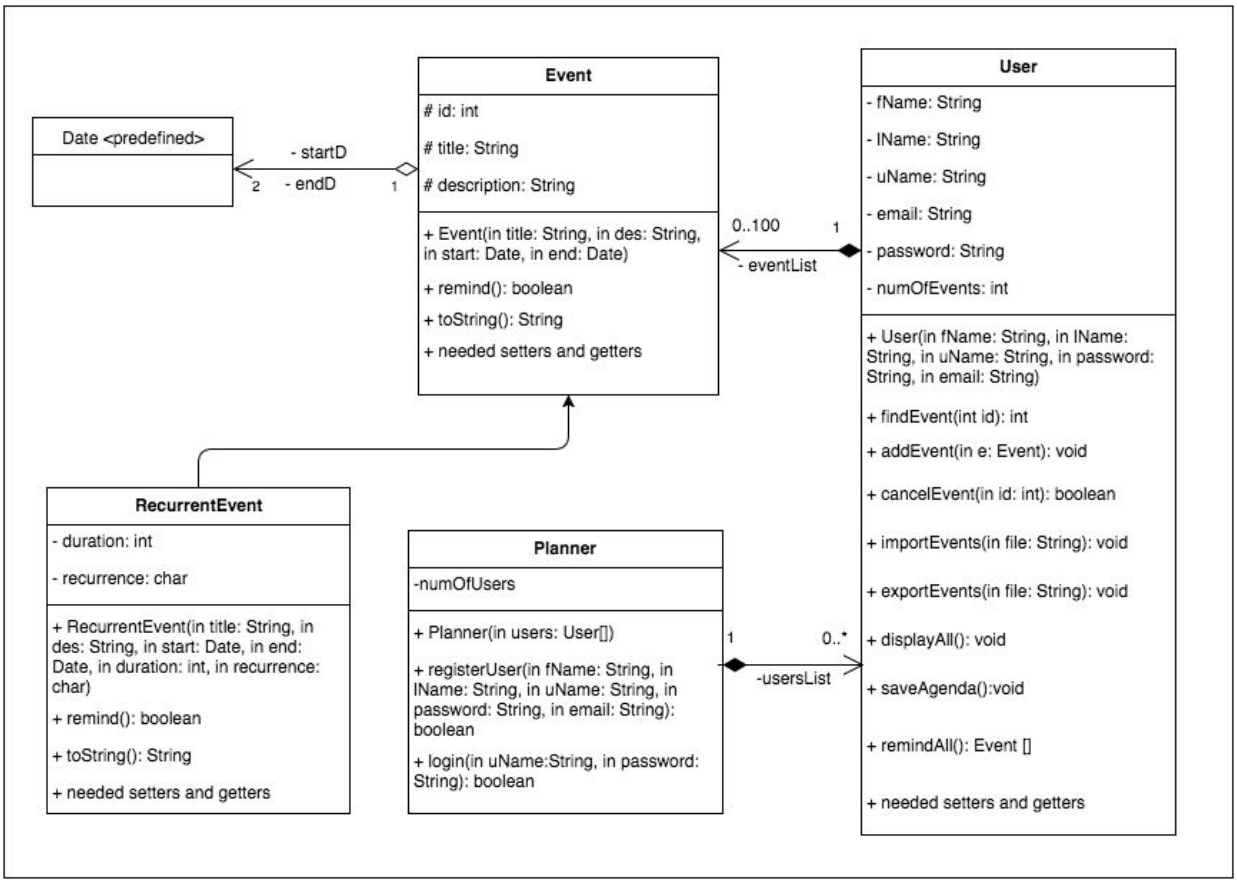
King Saud University
College of Computer and Information Sciences
Computer Science Department

CSC 113 - Project **Second Semester 1436-1437**

Phase #1
Due 7 April

You are designing a simple planning application to keep track of events and manage them. It can be used by more than one user, each with their separate data that can be accessed after a user logs in. The data for this application persists beyond the execution of each session.



What to Implement:

In the first phase of your project you should solve the problem by designing a full program **(without GUI)** with the previous description. In addition to the following:
i. Your program should be able to create a Planner object and load the users and all their information from file(s) that are stored from the previous session (an empty array will be created if it doesn't exist). It will then present two menus.

ii. First menu:

1. Register user, a new user will be added to the Planner
2. Log in: If the user has already registered before, you can retrieve their information by letting them log in with their username (uName) and password.
3. Exit: all user information (userList) will be stored in file(s) for the next session.

iii . Once a current user is selected (after logging in), display a repeated menu that:

1. Add event: adds an event to the user, each event is automatically assigned an incrementing ID, other event information are read from the user.

   There is a special kind of event that is recurrent. See example in the next page.
2. Find event: retrieves the index of an event in the eventList based on its ID
3. Cancel event: removes an event from the user's account (eventList) based on its ID
4. Reminders: Displays all upcoming events (will occur in the future)
5. Display all events for that user
6. Save Agenda: Saves all the information for user's events in a text file, it also includes all the information for that user in the beginning. The file should be named by username in a ".txt" extension.
7. Import Events: adds the events saved in some file to the user's list of events
8. Export Events: saves the user's events in a specified file (each event is written individually, not saved as an array)
9. Exit, returns to the previous (register/log in menu)

Note the following:

- Class Date is a predefined class in java. You don't need to implement it, but you need to search and find out how to use it. You can use any of the Java Date-Time classes as you see suitable.
- You can add any other attributes or methods you think are necessary.
- An event cannot have an endDate earlier than its startDate, if this occurs, your program should throw an unchecked exception of type **IllegalArgumentException** when trying to create it.
- You should have exception handling routines that make your application behave reasonably.

- **Recurrent Event:**
  A recurrent event is a special case of event that occurs repeatedly at fixed intervals, where the startDate will refer to the start date of the first occurrence of the event, and the endDate should be the date after which the event should never occur. The duration attribute refers to the number of days each occurrence of the event stays. The recurrence is a character that refers to how often it re-occurs, it could be 'D' which means daily, occurring every day; 'W' for weekly, occurring every week; or 'M' for monthly, occurring every month .

  remind() will return true if there is at least one occurrence that still did not start

toString() will return all the information for the event in addition to the start and end dates of every occurence, no need to print the duration or recurrence attributes as they are included in calculating the individual start and end dates.

Event #*id*: *Title*
description

StartDate 1 - EndDate1
StartDate2 - EndDate2
.
.
StartDateN - EndDateN

Example:
Take Friday Prayer as an example, if you set the startDate to 25 March, 2016, and the endDate to 2 May, 2016, the duration should be 1, since it does not last for more than one day, and recurrence should be 'w', since it occurs every week.

Remind:
If today was 26th of March, it will return true since the event will occur again on 1 April, 8 April, 15 April, 22 April, and 29 April.

If today was 28th of April, it will also return true since the event will occur one last time on the 29th.

If today was 30 April, it will return false, because there is no occurrence yet to happen.

toString:

Event #*1*: *Friday Prayer*
Weekly prayer in holy Jumaa

25 /03/2016 - 25 /03/2016
01 /04/2016 - 01 /04/2016
08 /04/2016 - 08 /04/2016
15 /04/2016 - 15 /04/2016
22 /04/2016 - 22 /04/2016
29 /04/2016 - 29 /04/2016

What to Submit:

i. The source code of your program.

ii. A report that includes the following (please download the report sample from the LMS):

- Introduction
- Full UML of your project

- Implementation and design details describing your work, include only the most important parts of the code **(Not the whole code)**.
- Screenshot of your sample run.