CS310 Assignment #7 - Sprint3

**Deadline:** 11/29, Wednesday, 11:59:59pm (HW7)
11/30, Thursday, 11:59:59pm (Individual Project Assessment)

**Late policy:** Late submissions lose 1% of the total available points for each hour the assignment is late, for up to 48 hours after the original deadline.

**Submission:** All submissions will be via D2L. Each team only needs to submit one .zip file under “Assignment #7” (see below for details).

**Assignment Description**

In this assignment, you will continue following the Scrum workflow to complete your final system. You need to plan your Sprint3 based on the requirements in the Requirement List that were missing from your current system, your Product Backlog and the requirements we newly added (see below). Note that the newly added non-functional requirements are not required, but you will be given extra credit if you can demonstrate that you have met those. Newly added functional requirements are suggestions. You can choose to implement your own new functional requirements, but it may not lead to extra points. Your team needs to 1) produce a sprint plan; 2) follow the Scrum workflow and document the process; 3) produce the final product based on your sprint plan. This is further clarified below.

**Instructions**

As an option, you can use some Scrum tools to help you manage your project and keep track of the process. Examples are, [scrumwise](https://www.scrumwise.com), [confluence](https://confluence.atlassian.com), [trello](https://trello.com) and many more. This is a suggestion, to help with the task, but is not required.

**Requirement**

As many of you asked how to impress your customers, here is a further list of additional features that you can include in your project.

**Project 1 (Customer: Yixue Zhao)**

**Functional:**

1. Verify the user **+1**
2. Map view of search results **+5**
3. History of transactions **+3**
4. The poster and requester can also rate each other in the history of transactions **+2**
5. Integrate Facebook messenger in the app **+2**
6. The app can suggest if the requester is on the provider’s way or not, and optimize the path if there are multiple requesters **+5**
7. The two scenarios in Assignment 3 Section 6 “Requirement Change” +5 for each
8. Spam report +1
9. Posts recommendations for users (no subscription required, learn from user’s past
   behaviors) +5
10. Hide posts from specific posters +2

Non-functional:
11. Improve the performance (speed) of your system
   a. Identify at least two common user behaviors that will potentially cause long
      response time, and provide a way for each case to reduce the response time in
      your implementation. E.g., you identify that a requester always goes to the
      poster’s profile page after looking at the poster’s post. In this case, you can
      improve the performance by requesting poster’s profile information when the
      requester is looking at the poster’s post, and store the poster’s profile information
      in the cache. Then when the requester goes to the poster’s profile page, you can
      provide the poster’s profile information from the cache rather than request it from
      your server.
   b. You need to show the response time is actually reduced. E.g., you should record
      the response time before and after your improvement. Please feel free to use
      external tools.
   c. Extra points will be considered if 1) you can simulate your test cases (before and
      after improvement) under different network conditions; 2) you can replay your test
      cases automatically (e.g., using RERAN for Android)

Optional:
12. Other cool features of your choice! (Particularly nice, non-trivial features will be
    considered for extra points.)

Project 2 (Customer: Shushan Arakelyan)

Functional:
1. Sorting the profiles according to how often they are used, so that more frequent ones are
   at the top. (Extra: +1)
2. Automatically detect schedules that can be disabled on public holidays (e.g. the ones
   that are happening during working hours from Monday to Friday or some other, better
   intuition you might have). (Extra: +4)
3. Add synchronization with Google Calendar. You can add either one or all of the below
   functionalities in full or partially:
   a. Sync schedules from Focus! to Google Calendar. Everything that is part of a
      schedule should be added as an infinitely recurring event in Google Calendar.
      Once the schedule is deleted or modified, the Google Calendar should be
modified accordingly. The user should get notification about modifications in Google Calendar. **(Extra: +3)**
b. Sync events from Google Calendar with Focus! In the application have a “Google Calendar” schedule. It will serve as a “container” for the events from the Google Calendar for the week (or duration of the recurring unit of the schedule) starting from today. Let the user choose which profile should be used for each event. **(Extra: +3)**
c. As an extension to the previous subtask, you can add “smart” matching by keeping track of what profiles were used for what kind of events, at what locations and suggest those to users. **(Extra: +3)**

4.
   a. Share Focus! to Facebook or Twitter **(Extra: +1)**.
   b. When sharing suggest to take a photo to share what the user have been doing in the time they focused during the past focusing period, day or week. **(Extra: +1)**

5. The two scenarios in Assignment 3 Section 6 “Requirement Change”

6. All the other “core” functional requirements listed in the Requirement List

Optional:

7. Other cool features of your choice! (Particularly nice, non-trivial features will be considered for extra points.)

**Project 3 (Customer: Sarah Cooney)**

**Functional:**

1. Create automatic transactions which occur on a given date. For instance, an automatic deduction from the entertainment budget for Netflix on the 15th of the month.
2. Receipt Scanning for entering transactions.
3. Rollover of leftover money in categories from period to period.
4. A savings category without limit that extra money is rolled into at the end of the period.
5. Support for adding multiple items to different categories on one transaction entry form.
6. Shared budgets to which multiple users can add transactions.
7. All the other “core” functional requirements listed in the Requirement List

**Non-functional:**

8. Improve the performance (speed) of your system
   a. Identify at least two common user behaviors that will potentially cause long response time, and provide a way for each case to reduce the response time in your implementation. You need to show the response time is actually reduced. E.g., you should record the response time before and after your improvement. Please feel free to use external tools.
b. Extra points will be considered if 1) you can simulate your test cases (before and after improvement) under different network conditions; 2) you can replay your test cases automatically (e.g., using RERAN)

Optional:
9. Other cool features of your choice! (Particularly nice, non-trivial features will be considered for extra points.)

Project 4 (Customer: Umang Gupta)

Functional:
1. Show frequent tags at the bottom of search bar (ex: yelp)
2. Allow uploading images in comments/ ratings / poll
3. Create moderator / admin page to moderate content
   a. Allow users to report inappropriate/malicious content
   b. Show flagged content to the moderators
4. User verification via sending email to user upon signing up
5. Notification
   a. Email notifications of activities.
      i. Notification can be configured --- Hourly & immediate
   b. Show notification on the app itself
6. Extra functionalities mentioned in the end of Assignment #3
7. All the other “core” functional requirements listed in the Requirement List

Non-functional:
8. Improve the UI & UI experience for users. No matter how you many features you put in the app, a good and intuitive UI always almost takes the spotlight.

Optional:
9. Other cool features of your choice! (Particularly nice, non-trivial features will be considered for extra points.)

Deliverable

You should submit a zip file containing the following items.
1. Your “improved” project.
2. A README file to explain how to run your app if there are any extra steps.
3. Your sprint 3 document, which should include the information specified below.

Your sprint 3 document must have the following sections.
1. Project Title and Authors
a. Your team name as appeared on D2L
b. A list of all team members (names and USC ID numbers)

2. Preface
3. Daily Scrums (you should have at least 6 of these)
   During the Daily Scrum, each team member answers three questions:
   - What did I do yesterday that helped the team meet the Sprint goal?
   - What will I do today to help the team meet the Sprint goal?
   - Do I see any impediment that prevents me or the team from meeting the Sprint goal?
   You should document your daily scrum, including the following information.
   a. Meeting start time and end time (~15 min is recommended)
   b. Meeting location
   c. Summary of every team member’s performance (short summary of the above three questions)

4. Sprint Review
   Your entire team should hold a sprint review meeting at the end of sprint 3. In the meeting, you should review the work that was completed and the planned work that was not completed. Also, you should reflect on the past Sprint, identify and agree on continuous process improvement actions. You should document your sprint review including the following information.
   a. Meeting start time and end time
   b. Meeting location
   c. Meeting summary (e.g., What has been done? What was planned but wasn’t completed? Why? What went well during the Sprint? What could be improved in the next Sprint?)

5. Individual Project Assessment - due by each student within 24 hours of the Assignment 7 deadline
   You are expected to work on this part of the assignment on your own. Any collusion is inappropriate, and if spotted, it will be treated as academic dishonesty and sanctioned as such. Furthermore, failing to submit an individual project assessment will result in a project grade of 0.
   a. Briefly (1-2 sentences) describe each team member’s, including your own, role on the team
   b. Briefly (1-2 sentences) describe each team member’s, including your own, contribution to the team.
   c. Describe any strengths or weaknesses in each member’s performance. Provide as much detail as necessary (e.g., Github link). Clearly bulletize and mark each strength with a “+” and each weakness with a “-“.
   d. For each team member, including yourself, provide a numerical score. Your baseline should be 100, meaning that an “average” team member’s contribution to your team is 100% of their expected effort. A team member who went above and beyond would contribute over 100% as compared to an average member. For example, you may think that someone worked twice as hard as they should
reasonably be expected to, so you would give them a score of 200. Analogously, a team member may have contributed less than they were expected to, in which case their score would be under 100. For example, a team member may have done half of what was expected of them, in which case their score would be 50. Your total scores must average to 100. For example, the individual scores for a five-person team should total exactly 500 (e.g., 100, 75, 95, 110, 120).