Low Fidelity Prototype Review

MoeKid101

Lo-fi prototype author: hidden.

- **(Consistency)**

  The requirement for consistency is multifaceted. In lo-fi prototype it is still invisible whether colors and styles of UI components are consistent, but I notice that the author is dedicated to consistency in terminology and interaction pattern.

  First, in both task page (P2) and subtask page (P3), a dialog-like "submit"-"cancel" interaction style is adopted, yielding clear knowledge of what the user himself is doing. Second, consistent terminology "progress status" and "task dependencies" are used on all pages involving these two aspects. These two features enables the user to

  With all these advantages worth appreciation, I find a minor drawback. Striving for consistent sequences of actions for similar situations implies that different situations are encouraged to have different presenting styles. I have a small suggestion for subtask page (P3) where the text enclosed in rectangles have different meanings: "Visit attraction 1" and "submit" are both UI components to edit your task list, but "for more detail" is intended to provide information. I personally think changing the style of displaying "for more detail" button is a good choice, for example using underline and colored text "for more detail".

- **(Universal Usability)**

  The task dependencies page (P4) is intended to introduce the meanings of different task dependencies, which helps novice users get an idea of what the provided function is. Considering the application is intended almost only for working purposes, demanding a short time to learn, such function is appropriate and excellent.

  In my opinion, it would be even better if language and date format can be adjusted so that the application is usable world-wide.

- **(Informative Feedback)**

  I personally agree that there could be more feedback in the action sequences. Presenting all the tasks on home page is admirable because after creating a new task, the user returns to the home page and directly notice the newly created task. However, after editing a task on P2 or editing a subtask on P3, there is no direct feedback on the user's actions when they return to the original page. If somehow the user gets idea of the changes he have just made, he might be more satisfied with your product.

- **(Dialogs to Yield Closure)**

  In this lo-fi prototype, most action sequences are not long and are completed within only one page, *e.g.* editing tasks on P2 and editing subtasks on P3. Therefore, from my perspective, there is no strong demand on dialogs to yield closure in this project. I'd like to humbly express my viewpoint that if you aren't planning to add another type of intuitive feedback after the user takes some action, you could add a dialog saying "Task editing succeeded" or "subtask created".

- **(Easy Reversal of Actions)**

  I guess the author tried to allow for reversal of actions through the "cancel" button. Such a "cancel" function is really useful to reverse user actions at very low programming cost. I would agree that the author had tried to stick to this rule. Speaking with a more discerning perspective, it would be more considerate if you are willing to add another **UNDO** button on the home page so that certain actions such as editing the title of a task could be easily reverted.

- **(Internal Locus of Control)**

  The author followed this rule closely.

  Firstly, there is no acausality in the human-computer interaction process as all the feedbacks are initiated by some certain user actions, thus ensuring user's feeling of control over the UI interface. Secondly, every page has a return button "←" on top-left corner acting the role of a navigator. Such navigation also contributes to the user's feeling of in control.

  I would favor this design more if a description of the page returned to through "←" button is provided. For example, on task page, "← Home Page" tells the user that he will return to P1 by clicking that button, and then the user would feel the interface more controllable.

- **(Reduce Short Term Memory)**

  As is mentioned above, almost all action sequences are completed within one page, so the user don't have to remember anything temporarily, through which the author successfully sticked to this rule. Maybe the only short term memory requirement is that when a novice user want's to create task dependencies, he has to remember what the four types of dependencies mean by going through the task dependences page (P4), where there is only 4 chunks of information, allowing most people to store in their short term memory. Therefore I would agree that the author succeeded in reducing requirements on users' short term memory.

- **(Prevent Errors)**

  As far as I'm concerned, the only place where people make improper inputs is the start date and deadline entry box. In this part the author didn't explicitly gray out the date and time that would let to tasks ending before it even started. I think it helps improve the design if errors can be prevented through imposing restrictions on the date and time users can choose.

- **(Additional Comments)**

  The advantages of this design is quite obvious: simplicity. Such characteristic ensures that the application is easy to learn and retention is expected to be admirable. What's more, the entire project progress bar at the bottom of home page is an excellent extra feature that users might need.

  Sorry but I will focus on the drawbacks in this design so that it might help you improve in the high-fidelity prototype. However, after all, I'm a layman and feel free to stick to your own opinions if you don't agree with me.

  (1) A description of each task is a required feature by tutor, but I haven't seen this feature in the author's design.

  (2) There is no clear illustration about how is the progress page reached by user actions. Therefore I can't be sure about what is this page intended for.

  (3) According to my interpretation of the requirements, subtask is a concept described by a start-start dependency and an end-end dependency on another task, but the author seems to treat subtasks as a different feature from task dependencies. Meanwhile, I suppose that dependencies are not unique, *e.g.* TaskA might depend on two or more tasks to start. However, this feature doesn't seem to be presented by the author.