Heuristics Evaluation of Team #3342 Date 10/16/2023

By Team # 3345: Matthew Brown, Tony Shi, Jacob Mixon, Malick Abban, Anish Arora, Wilson Gao

General Heuristic Evaluation of Prototype Interface

Heuristic	***Evaluation*** In the space below, enter your observation and evaluation of the degree to which the heuristic has been satisfied. Use as much space as you see fit.
1. Visibility of system status • Always keep users informed about what is going on. • Provide appropriate feedback within reasonable time.	Each page is distinct and has a text prompt letting users know what to do/read
2. Match between system and the real world • Speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. • Follow real-world conventions, making information appear in a natural and logical order.	Users are able to understand what the system requires every step of the way. The order of the heart survey being done first was a little weird before going to the other functionality of the app. It would be convenient if there was a screen stating that we will now conduct a survey regarding the recent visit.
3. User control and freedom • Users often choose system functions by mistake. • Provide a clearly marked "out" to leave an unwanted state without having to go through an extended dialogue. • Support undo and redo.	Users only have a few options (weight, blood pressure) and can move between them easily However, on the home screen, if a user hits weight on accident there's no way to get out, and they have to re-enter weight, the same issue exists for blood pressure.
4. Consistency and standards • Users should not have to wonder whether different words, situations, or actions mean the same thing. • Follow platform conventions.	There is design consistency between frames, black and white coloring along with large buttons make it easier to see what to do.

5. Error prevention • Even better than good error messages is a careful design which prevents a problem from occurring in the first place.	There isn't a way to check this on a maze, but blood pressures and weights would need to be >0 and less than some upper bound as well. Same with the login information.
6. Recognition rather than recall • Make objects, actions, and options visible. • User should not have to remember information from one part of the dialogue to another. • Instructions for use of the system should be visible or easily retrievable whenever appropriate. • Buildings are all visible and clickable • Experienced users can directly check on the buildings they are interested in by clicking on them • Recent searches feature is useful for users that use the same buildings regularly	The app design is very simple and keeps actions limited to 1 per screen (even the home screen where the action is picking your next action), so users do not need to remember anything from the previous screen The faqs section is also very convenient to easily understand how the entire app functions. I assume that there is also a way for the users to be remembered once they've already logged in instead of having to login every time.
7. Flexibility and efficiency of use • Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user so that the system can cater to both inexperienced and experienced users. • Allow users to tailor frequent actions.	This app would be very easy to use even for those with weak vision or inexperience with technology due to the large buttons and text. The buttons are all very straightforward in what action they represent.
8. Aesthetic and minimalist design • Dialogues should not contain information which is irrelevant or rarely needed. • Every extra unit of information in a dialogue	Does a great job of using screen real estate and keeping each step simple during initial log in, and on the home screen. Very simple interface, there was no irrelevant information, and easy navigation throughout the application.

competes with the relevant units of information and diminishes their relative visibility.	
9. Help users recognize, diagnose, and recover from errors • Expressed in plain language (no codes) • Precisely indicate the problem • Constructively suggest a solution.	There didn't seem to be any way to go forward in the maze by entering the wrong data. Since there is only one pretty obvious way to go forward or backwards this isn't really a problem.
10. Help and documentation • Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. • Help information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.	Learn more section is easy to find, it could be expanded to include more sub-pages explaining how data is used, etc.