

Elisabeth Olson
Kenneth Kuan
Andy Hou
Rachel Friend
Team: **Don't Forget!**
CSE 490 F: HCI
Med-Fi Fidelity Prototype

1. Problem and solution overview

The problem was and continues to be that people forget the things they need to have with them when they're leaving for the day. We think the solution to this problem is to have a program that keeps track of a person's schedule and the items they want to be reminded of for the activities on their schedule, combined with an interface near the door that will scan the person for those items, compare against what's in the schedule, and let them know of the discrepancies. RFID tags seem to be the best solution for identifying the items to the computer and the scanner, as other options are extremely unreliable and/or unrealistic to implement in a home.

2. Tasks

Easy: The user would use the by-the-door interface to find out if they're missing anything for the day. When they approach the interface, they're relatively certain they have everything. The interface requires them to identify themselves and then type in a password, after which it shows a screen with the items they've forgotten. The screen they're shown indicates they have many items missing, so the user will click the "Scan Again" button, which will then refresh the page to say they have no items missing. If they would like to check their schedule at this point, they can also navigate to a page displaying their schedule.

Medium: The user has to add a new item to the program, so they will navigate to the Add New Item page, fill in the information, and click OK. This will show them a confirmation page before taking them back to the main screen.

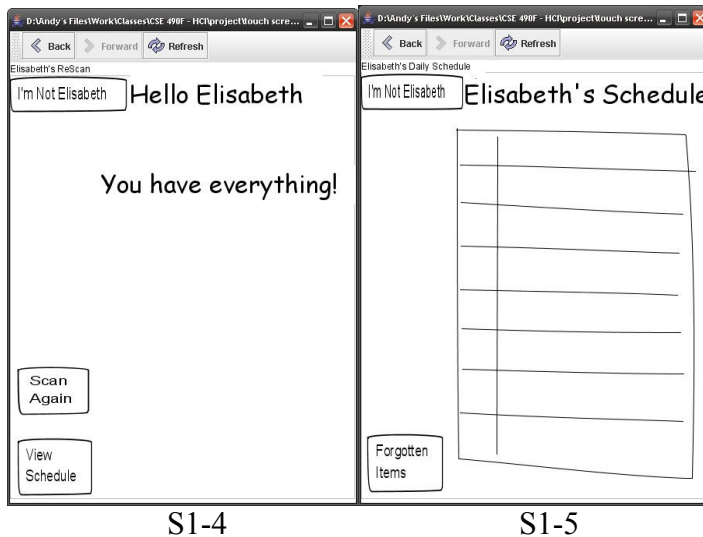
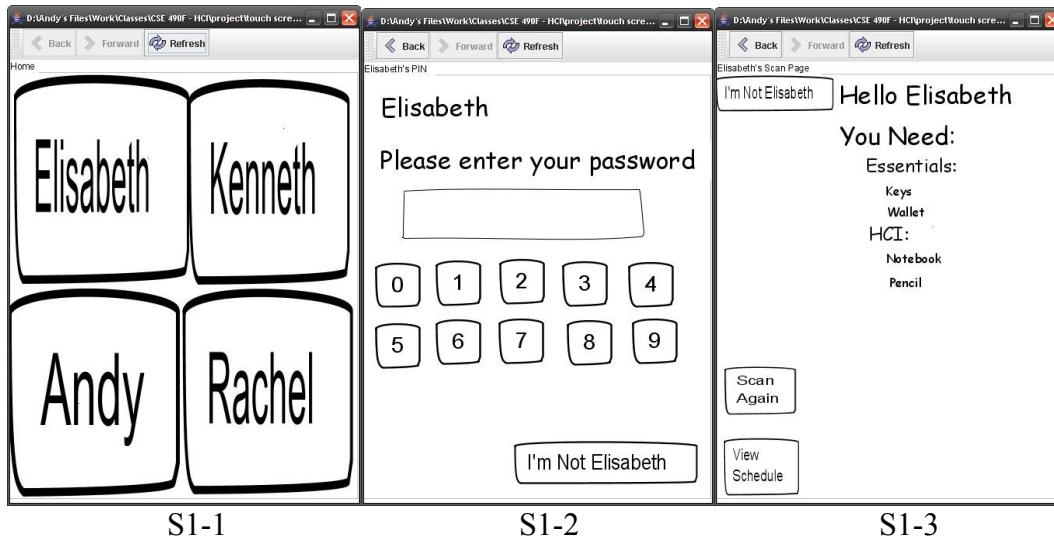
Hard: The user has to add a new activity and associate items with that activity. They will navigate to the Add New Activity page, fill in the appropriate information, click on the items they'd like to associate with the activity, and then click OK. This will show them a confirmation page before they are returned to the main screen.

3. Revised interface design

The low-fi testing caused our interface to change in two notable ways. First of all, we realized after the testing that privacy was a concern for many people, so we added a password feature (see S1-2, below) to the touch screen. Ideally, this feature would be one that the user could turn on or off depending on preference, but we decided that every user in the med-fi prototype would have a password for now. The second change was the addition of confirmation pages (C4, 5, 7, 8 in Section 5) in the scheduler to let the user know that the items or activities they'd added had been successfully put into the program.

Easy Task Scenario:

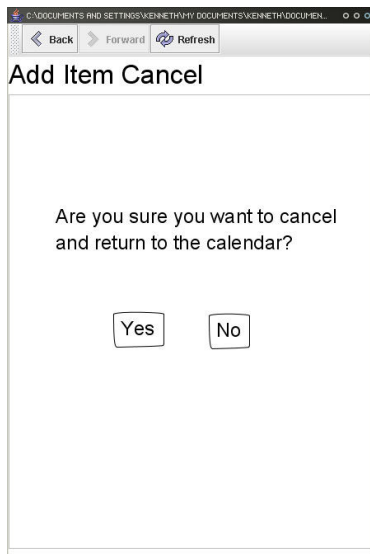
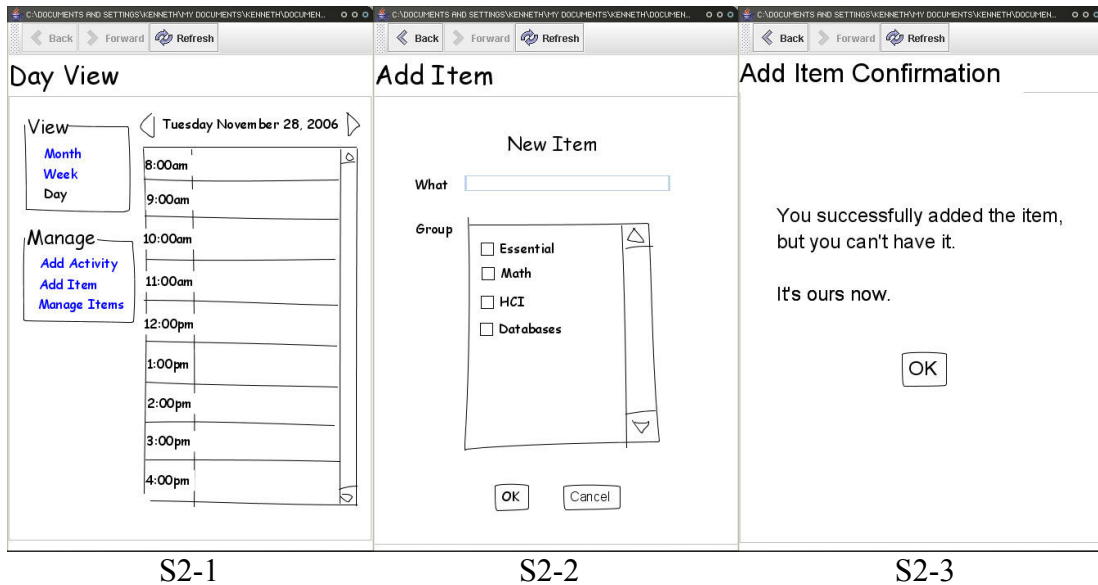
The user approaches the by-the-door interface on her way out for the day and sees the first screen (S1-1). The user then touches the button with her name, in this case “Elisabeth”, which brings her to S1-2. On this screen, the user enters her password. If the password is correct, S1-3 will come up. On this screen the system displays the items that the user has forgotten, i.e. the items that she needs for the day, but does not have with her. However, we have simulated a scanning malfunction on the first scan. The user should press the “Scan Again” button, which will update the page to show S1-4. Optionally, the user can also view her schedule for the day, by pressing the “View Schedule” button, which will bring up S1-5.



Medium Task Scenario:

The user wants to add her wallet to the item database. She places the RFID-tagged wallet on the scanner and turns her attention to the main calendar interface (S2-1).

She clicks the “Add Item” button, which opens the Add Item dialog (S2-2). She can then name the item and optionally add it to an existing group. After clicking “OK”, she is presented with a confirmation screen (S2-3) and returns to the main calendar (S2-1). However, since we couldn’t find a reasonable way for DENIM to reflect state changes, the added item doesn’t appear anywhere. If the user wants to cancel out of the New Item screen, she can click cancel and confirm her cancellation via the popup dialog (S2-4).

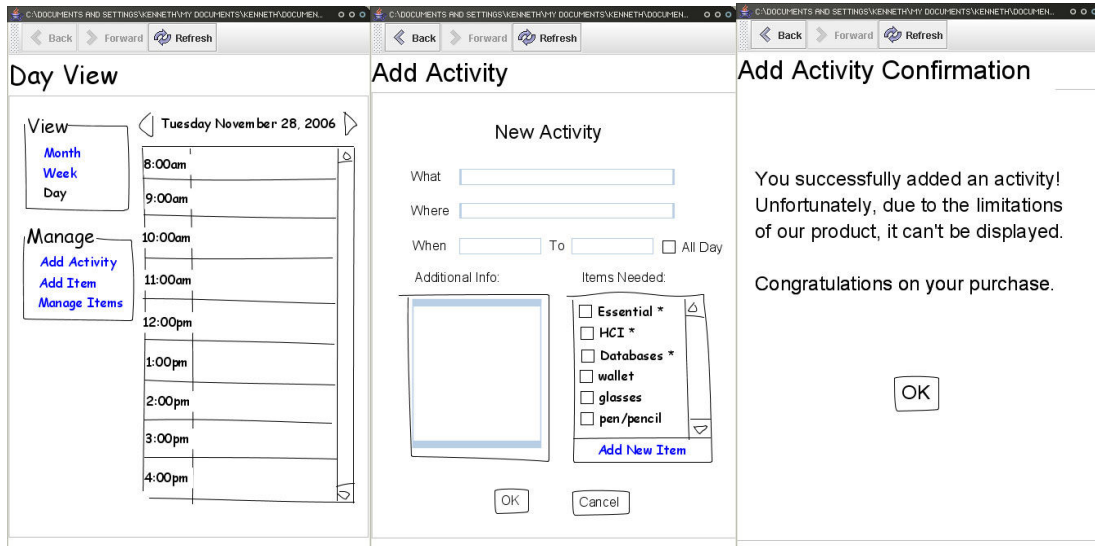


S2-4

Hard Task Scenario:

The user wants to add an activity where she’s meeting with her friend Rachel from 2:00 pm to 3:00 pm. She will need her glasses, wallet, notebook and pen, and she wants the program to check for these items. Once again, she begins at the main calendar interface (S3-1). She then clicks on the “Add Activity” button, bringing her

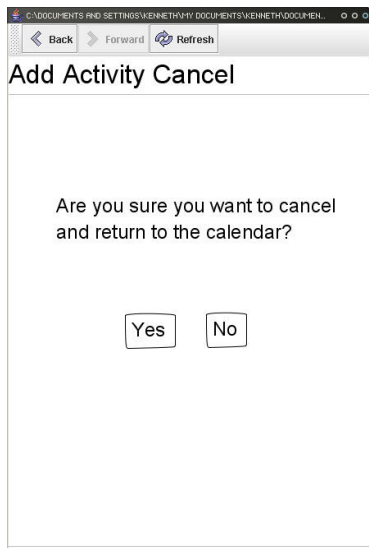
to the Add Activity Dialog (S3-2). She fills in the appropriate information, associates items with the activity, and clicks “OK”, which brings her to the activity confirmation screen (S3-3). Again, the new state isn’t reflected in the day view. The user can also cancel out of this screen, in which case she is presented with a cancellation confirmation (S3-4).



S3-1

S3-2

S3-3



S3-4

Unsupported Features:

Because of the nature of DENIM, we implemented only those screens that allow the user to complete the three representative tasks. Other features are clickable, but will lead to an “Unsupported Feature” screen. From “Day View”, clicking on “Month”, “Week”, or “Manage Items” displays the Unsupported Feature screen, and clicking on “Add New Item” from “Add Activity” does the same.

4. Prototype overview

Using Denim helped in that it took some of the problems of having a human computer out of the equation. For example, clicking on a button brings up a new page seemingly instantaneously, where before the delay was quite noticeable for the user. Furthermore, buttons on a page always go to the same place rather than sometimes having a mistake by the computer. Also, using Denim allowed the prototype to look more like the finished product without requiring us to actually build the finished product.

However, we had many problems with Denim. First of all, there were many bugs in the beta versions, some of which would cause the program to crash so we had to start over. Also, Denim is designed for a tablet PC, whereas we were all using desktops without the stylus feature that Denim relies on.

On another level, our project requires a lot of knowledge of the state, which Denim doesn't provide for. For example, we wanted to add a password feature, but as we couldn't put in a state machine to read in the password, we instead made the passwords somewhat ridiculous: one of the buttons will navigate through to the next page, and the other buttons can be clicked in whatever order for as many times as the user might want. Obviously, when we get around to actually coding in the password feature, we'll be handling that case differently.

Another problem we had was with the calendar aspect of our program. We made a design decision not to implement the calendar as more than a picture in this prototype, as it would be impossible for us to allow users to add events to the calendar in any meaningful way, and certainly not in the way we were envisioning. Rather than fitting the design to the tool, we decided to hold off on that part of design until we were using a tool better suited to it.

We also decided to continue to ignore the "Manage Items" option, as that would again require too many pages and too much effort for very little benefit at this stage. We think this feature might be best explored when we have a tool that can remember state, or possibly even after we start coding the rest of the project.

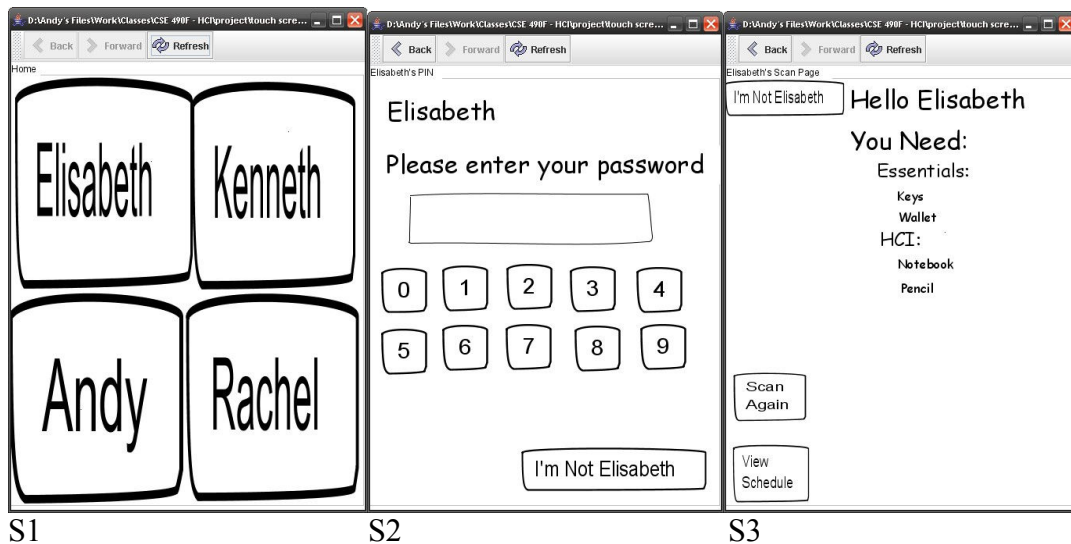
We also used a "Wizard of Oz" technique to deal with the actual RFID scanning aspect of our system. For the touch screen part, we used an extra page to simulate a scanning malfunction to see how the user would respond. For the scheduling part, we made the assumption that when the user adds a new item, the item to be added is always placed on the scanner. We made this assumption because there was no way to simulate the physical scanner in Denim.

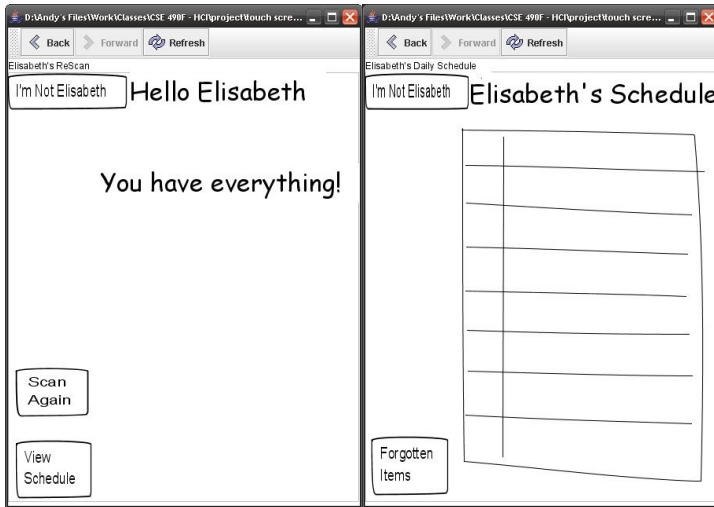
Despite these problems, however, we did implement almost all of the user interface aspect of the program. All of the by-the-door interface screens are present in our prototype, and all of the scheduling interface screens are present with the exception of the week and month views.

The scanner interface consists of S1, the main screen where the user selects his or her name. Then for each of our four example users the following screens are implemented: S2 is the where the user enters her password, S3 and S4 both show the screen where the user views the items she is missing for the day (the difference is that S3 shows the screen as it would appear if the user has forgotten all her items while S4 shows it as if she has remembered all her items), S5 shows the user her schedule for the day. We also have screens similar to S2-S5 for the other three example users, but since they are very similar, screenshots are omitted.

As mentioned in Section 3, the scheduler interface implements only the screens necessary for completion of the two scheduler-oriented tasks. Users begin at C1, the Day View screen. From there, clicking on “Month”, “Week”, or “Manage Items” brings them to C2, the “UF (Unsupported Feature): Day” screen. We decided that implementing these screens within DENIM would require an excess of similar but slightly differing screens, and that the functionality they add doesn’t represent the tasks we want them to accomplish. Clicking on “Add Activity” brings them to C6, from where they can access C7, C8, or C9 by clicking on “OK”, “Cancel”, or “Add New Item”, respectively. Returning to C1, clicking on “Add Item” brings up C3, the “Add Item” screen. From there, they can access C4 and C5 from “OK” and “Cancel”, respectively.

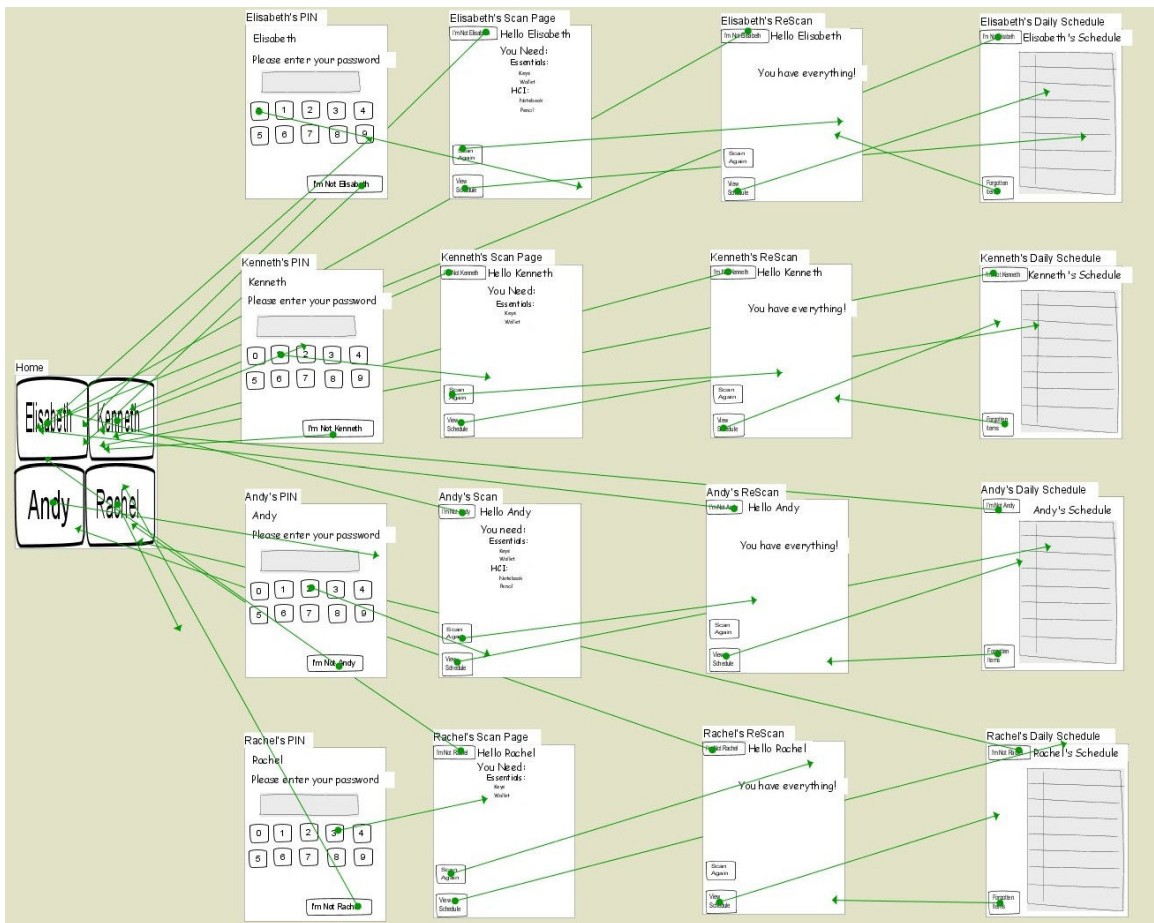
5. Prototype screenshots or scripts



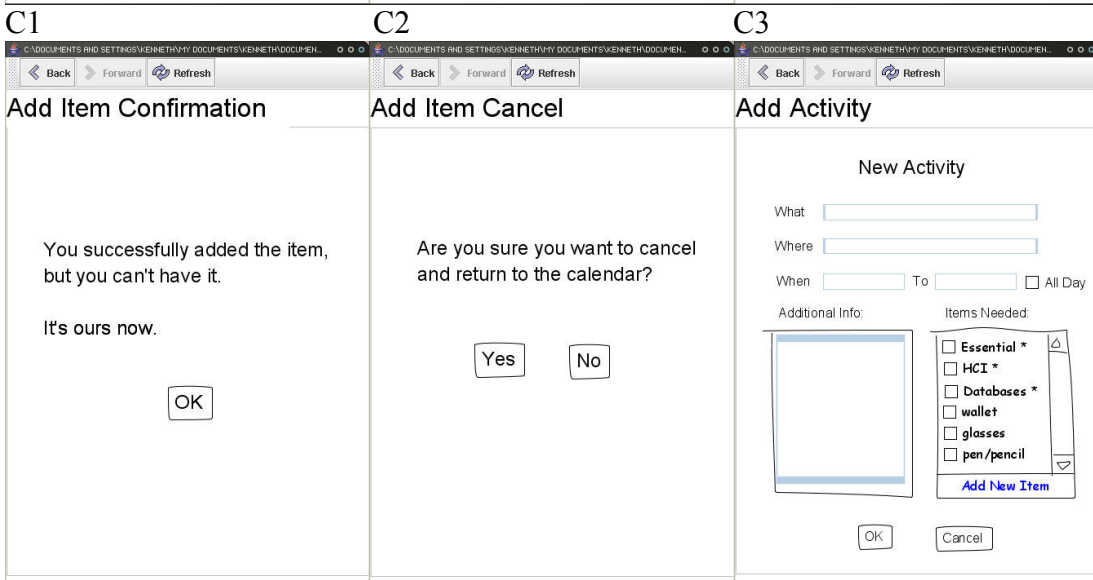
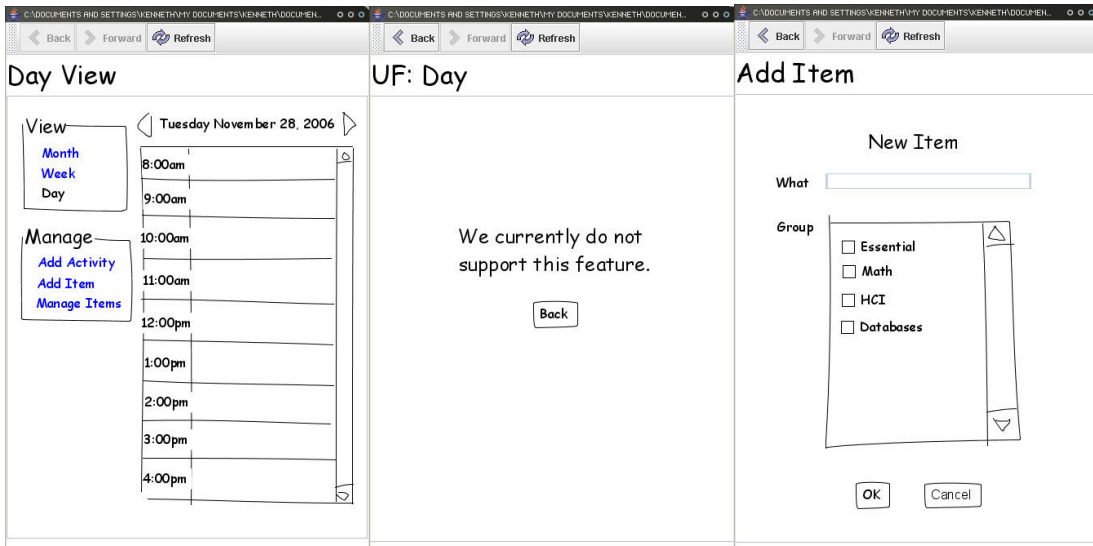


S4

S5



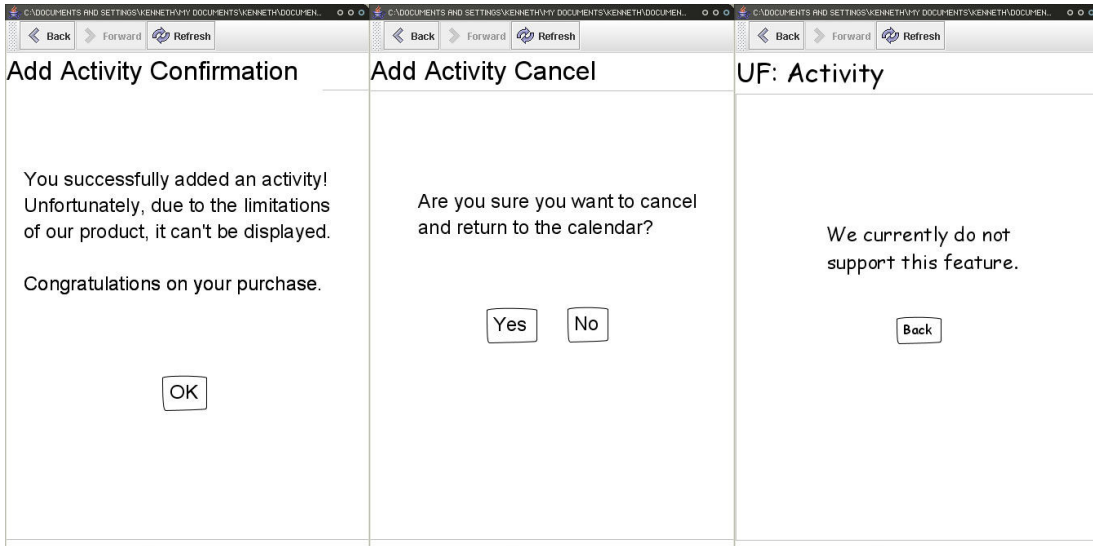
S6: Storyboard



C4

C5

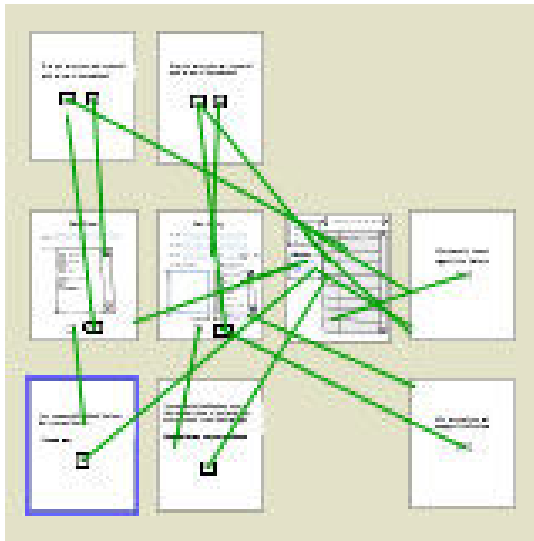
C6



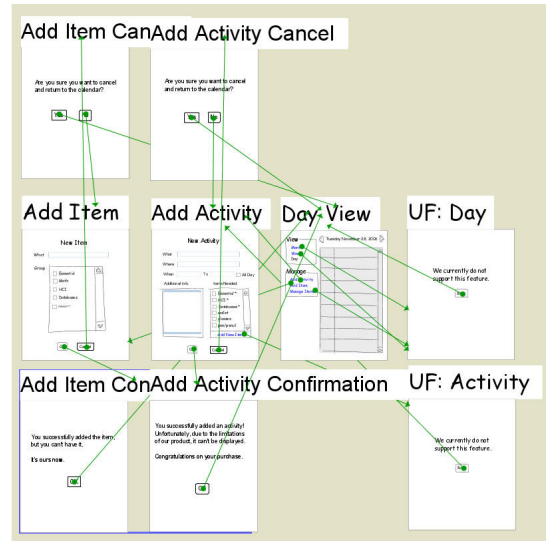
C7

C8

C9



C10: Overview



C11: Storyboard