CSCI4448: Project Part 3 Spring 2014

Alexia Newgord, Mark Lohaus

Summary

Here is a summary of the work we have performed over the last two weeks.

- Set up our development environment. This is a Linux server running Apache as our web server and MySQL as our database system,
- Studied documentation on the Yii framework,
- Installed the Yii framework,
- Developed our Responder Controller, view and model,
- Created Git repository for version control and issue tracking.

Mark got the development environment set up (with the help of a system admin at his work) and installed Yii. He also designed the database and inserted the tables in MySQL. Mark also helped get the Responder controller started. Alexia dove deep into the Yii documentation and further developed the Responder MVC—specifically by adding CRUD operations for patient records. She also has been maintaining a git repository of our progress and created a set of milestones/issues based on the system requirements.
We have only completed about one third of our project. We have two more controllers to implement. Now that we have learned Yii basis our development should go faster.

Design Patterns Used so Far

The main design pattern we are working on currently is the visitor pattern. We are also building a tree structure of questions and there is a possibility that that composite pattern may help as well. There are three places we need to leverage the question tree: When admins build the question tree, when responders answer questions about a patient and when emergency room staff view the answered questions. The html that these questions are wrapped in changes slightly for each of these rolls, so we are using the visitor pattern so that each of these three views passes themselves to the question model. The question model then calls that view's method which defines the HTML that is to surround the question.

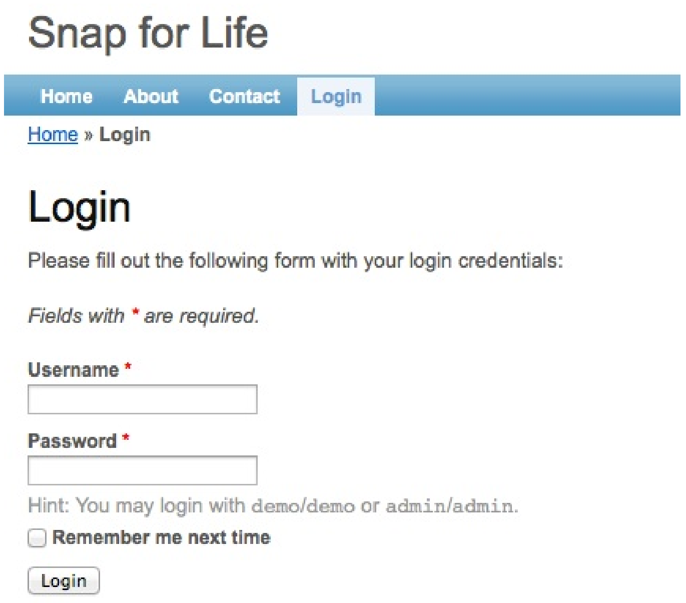
Design changes

Our original design called for question and question sets. The rule to make this work was that you could add questions to question sets and question sets to questions but not questions to questions or question sets to questions sets. This was a database restriction because the foreign key of a question needs to be predictable (a set or a question and not an either or situation). However, one of our real word questions we need to ask required adding a question set to a question set, and we were stuck. Therefore we decided on a simpler design there are only questions and no question sets. Now we add questions to other questions. A question can have more than one question under it. Now the foreign key of a question database record is just a parent question. Because all or our nodes are of the same type, the composite pattern may come into play as well to help us build our tree structure by accumulating the nested HTML. The class diagram, below, shows this design change. The impact of this design change on the user experience is that when a question is answered that has child question, the answer will be recorded and then the child questions will show to the user. The parent question might not be that meaningful in its own, but because the ER will see the tree structure it will make sense. For instance a tree structure might look like this

```
Trauma?
        Low BP?
        Bleeding?
                Life threatening?
                Type?
                        Arterial?
                        Venous?
```

“Type” is not really a question. It’s really a category (what we used to call a question set). However if “type” shows as “Yes” with, say “Arterial” as “yes” below it, it is unlikely to cause confusion because these questions will always be displayed as a visual tree to ER personnel and the association between “Type” and “Arterial” and “venous” should be clear.

Screenshots:



Login

Please fill out the following form with your login credentials:

*Fields with * are required.*

Username *

Password *

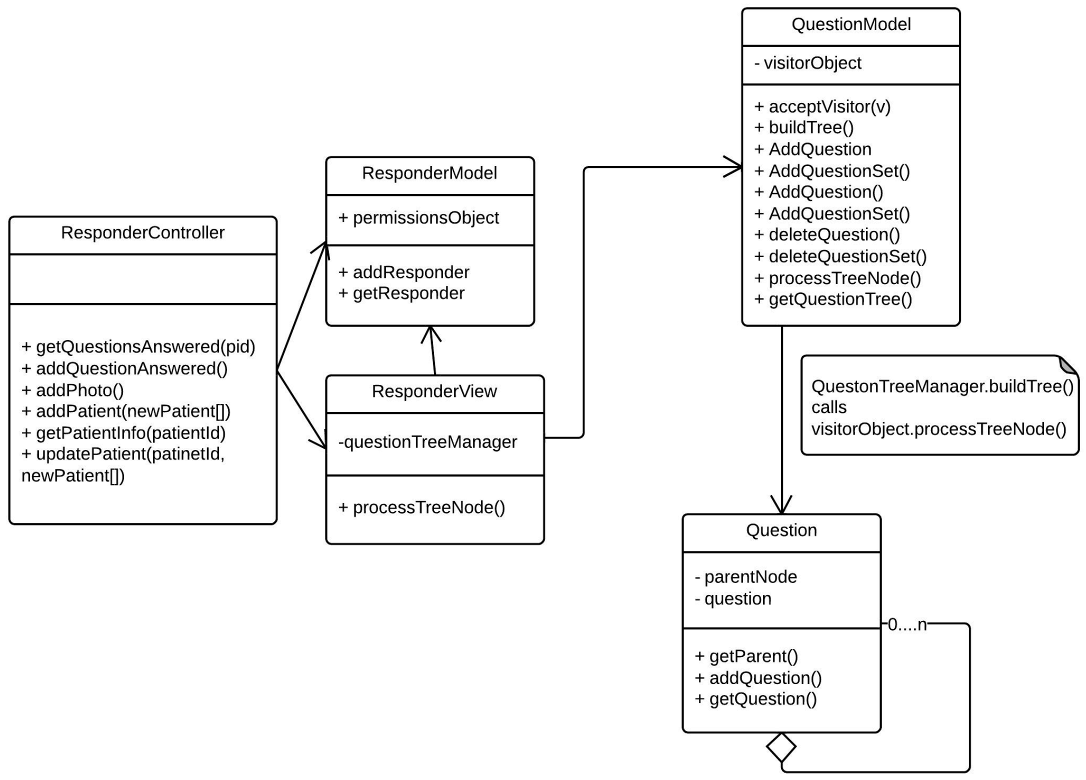
Hint: You may login with demo/demo or admin/admin.

Remember me next time

Login

Class Diagram

Classes implemented or in the process of being implemented



Plans for next Iteration

We need to develop our other two controllers: MedicalAdminController and EmergencyRoomController. However, first we must finish developing our QuestionModel class so that these controllers can delegate question tree question to it. Another key piece will be creating are the AJAX calls that will be defined in our views. Question tree management and the answering of patient questions will be handled with AJAX calls.

By April 25 we must our application must at least

- support creating and extending question trees by admins,

- allow emergency responders to create new patient records, upload photos and answer questions about these patients,
- provide a view for emergency room personnel so they can see patients photos and answered questions in near real time.

If we finish these core functionalities we “may” be able to work on other aspects of the application, such as having a user based login and more complete system and patient management.