to: file

from: Tom

subj: Adding support for clinic administrators registering the clinicians associated with their clinics

date: 04/05/2012



**Functionality**

add an 'AdministerClinic' button to the header. ( or 'AdministerAccount' or just 'Adiminister' if 'clinic is doing to seem odd to someone from, e.g. a hospital.)

Button is visible only when the current, signed-in user is a clinic administrator (or owner? same?)

when click button it goes to a clinic#administer page

which shows information on the clinic: address, shipping address, billing address, ... and a table of the clinicians that are members of / affiliated with this clinic.

this is an 'edit' screen, in the sense that the administrator can change any of the information on the clinic, and can add/delete/edit entries for individual clinicians. Maybe they do this by clicking the name of a clinician in a table, and it goes to a seperate edit page for that clinician. If the latter, when they submit the changes, the site returns to the clinic#admin page, showing the updated clinician information in the table (if any of the changed fields are visible)

same deal if the administrator adds/deletes a clinician; the responce is to repaint the clinic screen showing the clinician gone/added to the clinicians table

How do we deal with clinician passwords?

One option is that they all use the same password — the password for the clinic (or at least for clinicians at the clinic; we may want the administrator and the clinic's owner to see information available to individual clinilcians; they, at least, would need individual passwords. So: one password for the clinic owner, one for (each of the) administrator(s), and one to be used by the clnicians associated with this clinic.

**Implementation**

[ ... ]

Sounds like:

1. a controller for Clinic with #show and #edit actions for individual clinics [ plus info on the clinic as a whole, PLUS a link to the Netsuite backend Account for this clinic. ] Note there is no #index operation — the list of all clinics is available only to Tru Medical ops staff, not the administrators of individual clinics.

1.1. view/clinics/show.html.erb with in-place editng or an 'Edit' link

; and/or edit/html.erb [DONE].

1.2 schema migration to create the attributes for clinics; run rake db:migrate

1.3 run rake db:migrate

1.4 create clinics\_v0\_0430.xls from the customer file that Kim emailed on <date>

1.5 load the clinics table in the database from this file

1.6 create routes for clinics in routes.rb

1.7 once have completed 2.1-2.5 below, go back to #edit and #show views for Clinic and add a table listing the clinicians associated with this clinic. Allow the clinic adminstrator to:

1. view a clinician's record

2. edit a clinician's record

3. delete a clinician

4. add a new clinician

[ See if one of the jQuery table libraries would be the most succinct way of implementing this.]

1.8 Add a clickable link in the #show page that will take the clinic adminsitrator to the Netsuite 'account' for this 'customer'. Is there one? NOt if we define each clinician associated with the clinic as a 'customer' to Netsuite. Hide the clinicians from Netsuite? Essentially logging any of the clinicians associated with a particular clinic in as that customer.

2. a controller for Clinician with #show, #edit, #delete actions

2.1 Clinicians (plural) controller

2.2 views/clinicians directory with show.html.erb edit.html.erb destroy.html.erb

2.3 define the schema for clinics table in a migration ( add a column for type of clinic: clinic, hospital, ... ); migrate the database the include this table

2.4 Create a (largely empty) clinicians\_v0\_0430.xls table placing in it at least an administrator entry for each clinic (broadly defined; set the administration? field to true; update the clinic\_id field in this row to reference the clinic each of these people administers. Put in a dummy physical therapist for one of the clinics. He/she should see the price schedule for his/her clinic, but should not be able to see the AdministerClinic button.

2.5 load the clinicians table in the database from this file.

Some way of identifying someone who logs in as an administrator.

3. boolean column in users table? Problem is that I haven't figured out how to load this table from a spreadsheet since the passwords are encrypted and the encryp[ted value is created when the user is created; it is not in the spreadsheet.

Possible answer(s) to the problem with (3);

1. for testing the first couple, use Rails console to set the administrator? field in a one user record for each of the three test clinics we have.

2. longer term (production release):

1. build a update-from-csv-file action that just updates the existing tuples in the table, rather than rewriting them

2. dump the users file (including the encrypted passwords) and then load that back in with the additional adminstrator? information.

4. Administer\_Clinic button in the header that is only visible if the current\_user is a clinic administrator. I the signed in user clicks this button, she is taken to the Clinic#show screen for the clinic that she administers. Means User class has to have a clinic\_id field (which will only be filled in for users that are clinicians (or clinic adminstrators). Use STI implementation of subtypes of User for this, or just add a couple of boolean fields: clinic\_administrator? clinician? owner? [ What about dealers and reps? — two additional fields? ] Use migrations to add these columns to the User table to avoid the problem of wiping out their password? And then set them by signining in and hitting MyAccount button? Or is there some way to turn off pasword encryption now, and then turn it on later? [ DONE by adding a master password to the system; then have to go into each user account using the master password, and reset the user's password to "apple#{tm\_customer\_id}".

How determine that? if user\_logged\_in then if current\_user.administrator? or current\_user.owner?

5. Careful about

— implicit scoping problems: we are calling out from controller#actions within the devise/Worden Rails engine to controllers#actions in the containing app.

— naming/scoping conflicts with an active-admin based administration subsystem that I would like to insert for Kim's use in R2.

**clinic/show/edit screens**

show screen:

list of attributes/attribute values for clinic

table listing name, email, password for clinicians; edit/delete buttons on each row, a button below table for adding a new clinician.

edit screen:

a semantic\_form\_for with editable fields for the clinic

same table for clinicians

an 'edit' button that brings up a clinician#edit screen allowing administrator to change (name fields), email, and password; when user hits 'update' button on that screen, it goes back to the clinic#view screen, now showing the new email and/or password for the clinician in question. Alternatively, in-place editing for the email and password fields for a clinician, or If in-place editing then have to use AJAX to actually update the clinicians record in the database when user clicks 'update' button on row.

'delete' button on row of a clinician, calls Clinician#destroy and returns to repaint the current Clinic#edit screen, with the table of clinicians no longer showing the just-deleted clinician.

If the clinic administrator hits the 'Add Clinician' button it

goes to the Clinician#create action which paints an empty form for the clinician, but with the clinic name filled in.

when administrator hits the 'Create' button on that screen it inserts the clinicians into the database, and returns the administrator to the Clinic#view page for the clinic she has been working with. The table of clinicians will now show the newly added clinician. ( Perhaps the row for the newly added clinician should be highlighted. )

**Question(s):**

1. how to keep track of the current Clinic so can return to its view screen after updating/adding/deleting a clinician.

2. how to get encrypted passwords for new clinicians into the database: invoke register operation supplied by devise?

3. how to change a clinician's password: delete the clinician and add him to the system again, calling devise's 'register' method as in 2 above?