Texas A&M University
Fall 2020 CSCE 606 Software Engineering
Medical Persistent Project

Team member:
Zihao Zheng 625003943
Minreng Wu 529005234
Mingyang Wan 128003788
Siyang Yang 431000499
Yue Zhuo 230002710

Team Members:

Customer: Dr. Michael J. Miller
Product Owner: Mingyang Wan
Scrum Master: Siyang Yang
Programming Members: Yue Zhuo, Zihao Zheng, Minreng Wu

Logistics

Github: https://github.com/Siyangyang/MedicalPersistanceSystem2020
Pivotal Tracker: https://www.pivotaltracker.com/n/projects/2467945
Heroku: https://mps-2020-fall.herokuapp.com/
Poster Video: https://www.youtube.com/watch?v=7x_gyX6fJmw
Demo Video: https://youtu.be/sr10ny8BBqk
QR Code:

Summary

Our team focused on the implementation of the patients providers Communication of Medical Persistence System. This project aims to raise osteoporosis patient's awareness of this disease and its treatments, and helps providers/medical professionals to easier communicate with their patients. Osteoporosis brings many inconveniences through its development cycle. Yet, due to the no-serious nature of the disease at its early stage, it's common for patients at an early stage of their osteoporosis to dismiss their condition and need for medication.

The previous iterations of this project developed the mainframe of this project, supporting functionality such as login, creating account, change password. With our new implementation, medical providers can now provide patient instant feedback to their patients survey. We also increased the quality of life for user experience, made the system more accessible for both provider and user by changing the display. We also fixed many potential system breaking bugs that can potentially hinder the usability of this system.

Issue Legacy

For the legacy project, the previous teams had already completed a web system that can help providers to collect surveys from patients. The legacy system has two different types of user roles: provider and admin. The provider can log in and access the patients' survey, while the admin has the ability to manage providers.

For the process of understanding the legacy project, the first step we did is to log in to the system with these two roles of users and understand the workflow of the whole system. Then, we forked and cloned the project to our local Cloud 9 VM and managed to make it run locally. Next, we tried to map the buttons and links on the frontend side to the action functions in each controller so that we can understand what feature each action function implements and how they link to our UI piece. After we did the research we mentioned above, we gained a rough understanding of the whole project and knew how to add new features and modify the existing features.

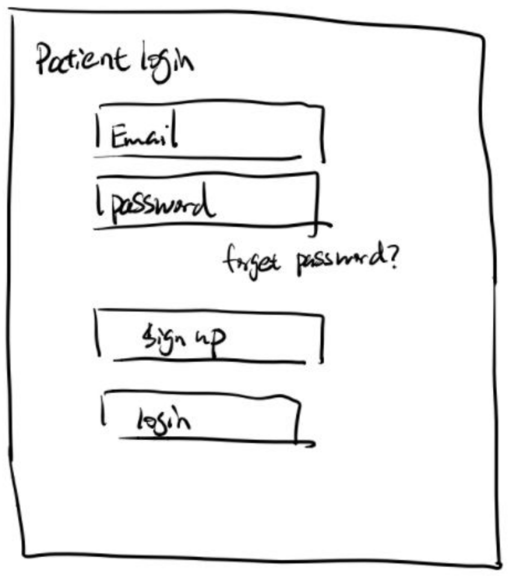
The legacy project uses PostgreSQL instead of SQLite as the database and also uses some 3rd party tools for rails, such as Devise and RailsAdmin. So we also spent some time investigating these components. And we wrote a detailed README document to introduce how to set up the project and use the external components, which helps our team members and the future developers to get started with this project more easily.

And one of the modifications we made on the legacy project is that we change the schema of the 'Users' table. The previous 'User' table uses two boolean variables 'admin' and 'provider' to represent whether a user is an admin or whether a user is a provider. But this semester, we are required to add one more role 'patient' for the users. So this schema would be kind of redundant. Therefore, we removed these two columns 'admin' and 'provider', and added one more column 'role' which has three options 'patient', 'provider', and 'admin'.

Lo-Fi UI Mockup

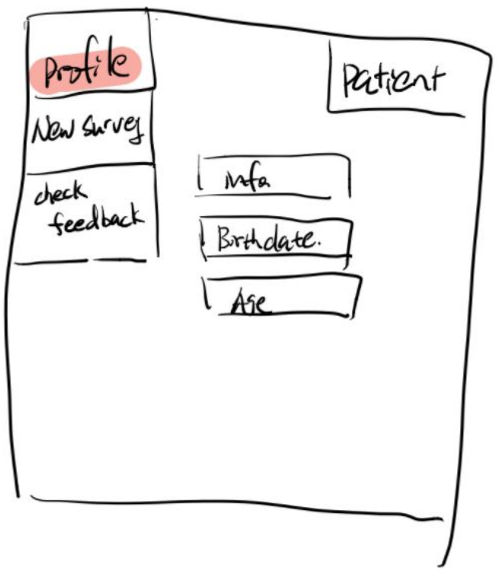
1. Patient Login Page

   Patients can log in or create a new account on a page that looks similar to this.

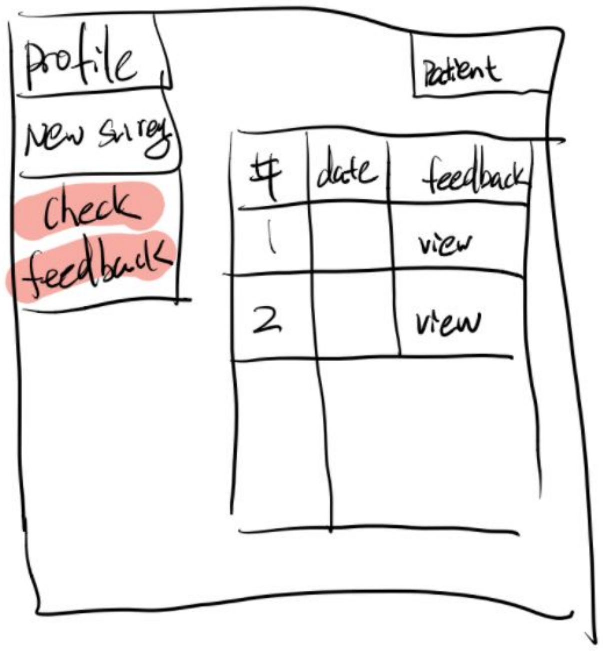


2. Patient Portal Page

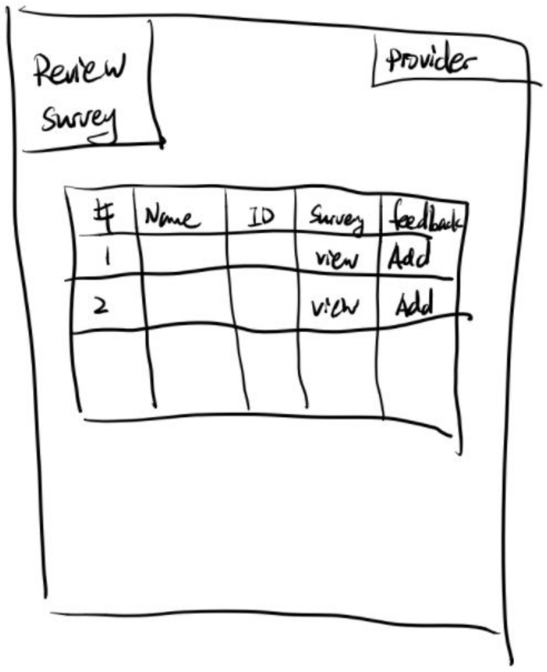
   After the patient login the system, it will redirect to this portal page. The default portal page will show the profile of the current patient.



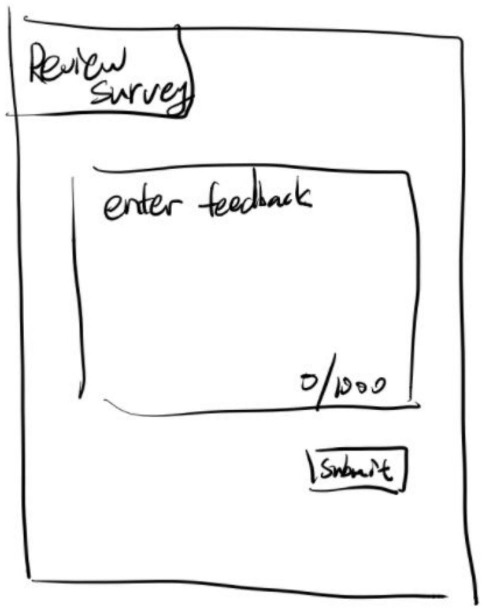
3. Patient Feedback Page
   The patient can view his/her historical survey record and the corresponding feedback for each record.



4. Provider Feedback Review Page
   The provider can review his/her patients' survey records in a table. The provider can also add feedback to a record by clicking the "add feedback" button.



5. Provider Feedback Add Page
   The provider can enter his/her feedback to a record on this page.



Iterations

Iteration 0: During the first iteration, we brainstormed the big scope of the project and divided team responsibilities to everyone. We set up the GitHub,pivotal tracker and Heroku accounts, then started to dive into the legacy project codes. First of all, we set up the working environment on the cloud9 platform in AWS, and built the Postgres database from both development and testing.After we were able to deploy the website locally on cloud 9, we digged into legacy code to decide how to refactor and build the code. Then, we started to prioritize the users' stories and followed the policy of “If it ain’t broken, don’t rebuild it”.

Iteration 1: We started to work on user stories on the login page. First of all, we made some modifications on the login page such as removing the background image and increasing the contrast. Creating an individual login page for the patients so that all patients are required to own an account before taking any survey.

Iteration 2: Our team started to implement the feedback functionality for patients. The feedback text box appeared on the ‘view feedback’ page, however, there was no backend implemented yet. The providers could not write any feedback showing to patients. At the same time, we also updated user stories on adding the guideline for creating new accounts, hiding the past surveys response from patients. Importantly, we created new scenario testing on the login page on the logic level. We used cucumber BDD testing to ensure the login page and historical survey records are robust.

Iteration 3: We have demonstrated the first time in this iteration. We had updated the 'Questions' database so that both patient and provider have access to read the feedback. And the provider was able to create and update the feedback. We also made few modifications on the front end to improve the color schema and pattern. We also added more testing cases in the cucumber to validate the feedback functionality.

Iteration 4: All the user stories were implemented and tested by our members. We had added a new story on adding timestamps on the list of surveys. We spent the most of the time on creating rspec unit testing. All the refactored and built controllers were tested by Rspec. Both happen and sad paths were covered.

Customer Meetings

Iteration 0 meeting: 09/28/2020

We demonstrated the current state of the project, what function is completed, what function is lacking, what the UI looks like, ect. Customers briefly describe the general purpose of the project, as well as the expectation for development goals for this semester. Based on the expectation of the customer, we came up with several lofi-demonstration graphs to confirm our understanding of the user story.

Iteration 1 meeting: 10/12/2020

We demonstrated modifications we did to the mainpage based on customer feedback on previous iteration meetings. New login page is made easier to access for the target stakeholders of this project. The team also gained more insight into how this project should be designed and implemented.

Iteration 2 meeting: 10/26/2020

We showed the updated stories from Iteration 1, which includes many changes to the user interface based on customer suggestions. Furthermore, we described to the user the exact process where we modified the database to accommodate the user story, and the current stage of implementation for the patient panel and provider panel, as well as our future plan including feature such as Add password guideline (requirement of password) and Separate the provider and patient login portal.

Iteration 3 meeting: 11/09/2020

During this meeting, we demonstrated the first demo to our customer. We have shown our progress and all the user stories we have accomplished. Users further pointed out a few minor suggestions to improve the user experience. The feature we demonstrated includes follow:

Iteration 4 meeting: 11/23/2020

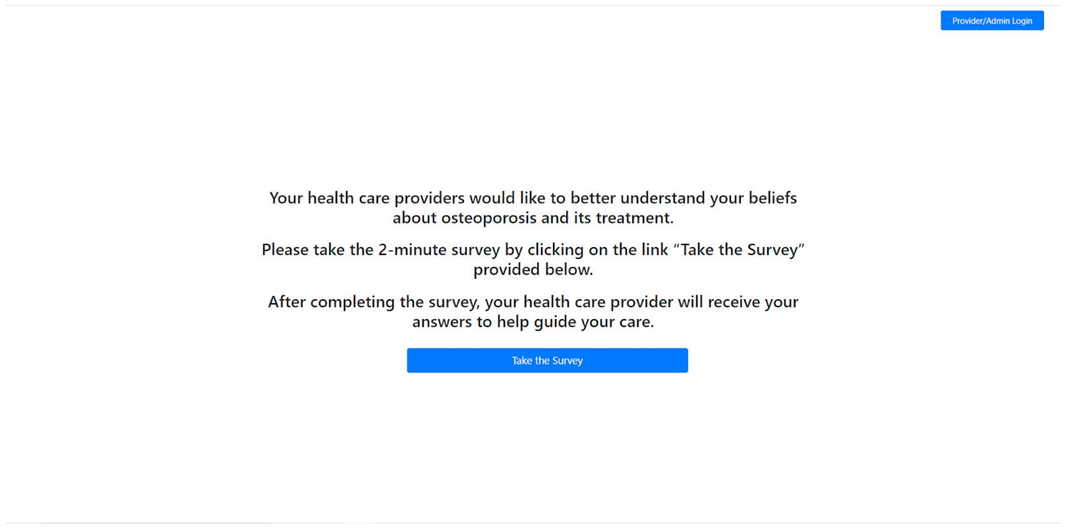
Our team had demonstrated the final version of the medical persistence system project to Professor. During the final presentation, we demonstrated the happy path for both provider user and patient use. After demonstration, we briefly discussed the future development plan and perspective Dr. Miller had for this project, as well as how he planned to utilize this project.

User Stories

1. Feature: update the main page

   As a new patient
   So that I don't get confused about where to login or take a survey
   I want to replace the login button and the background image

This user story is simple and straightforward on the expectation and solution: the client would like the main page to be cleaner and easier to understand for patients, especially for those who might suffer from presbyopia due to age. Thus, we removed the gray background picture from the main page and adjusted the font of the main page message so that it is easier for patients to read. We also replaced the login button on the top right with a more specific "provider/admin login" button so that patients wouldn't not get confused by the main page function. The main purpose of the main page still remains the same as a guidance for different users: patients, providers, and admin. Patients will be informed of the purpose of the survey and redirected to the patient's login page to continue the survey, while providers and admin will go straight to their login page to the management portal.



Your health care providers would like to better understand your beliefs about osteoporosis and its treatment.

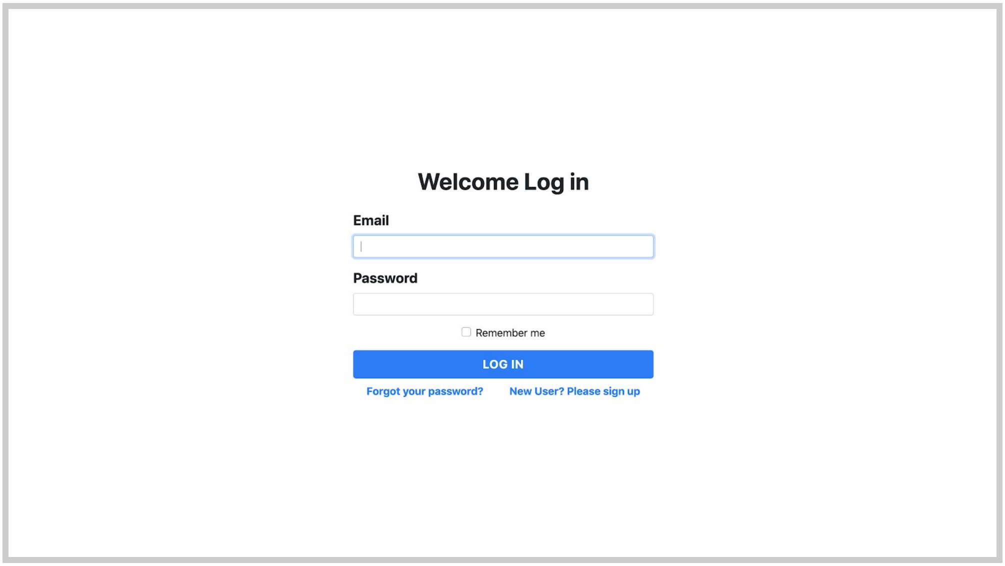
Please take the 2-minute survey by clicking on the link "Take the Survey" provided below.

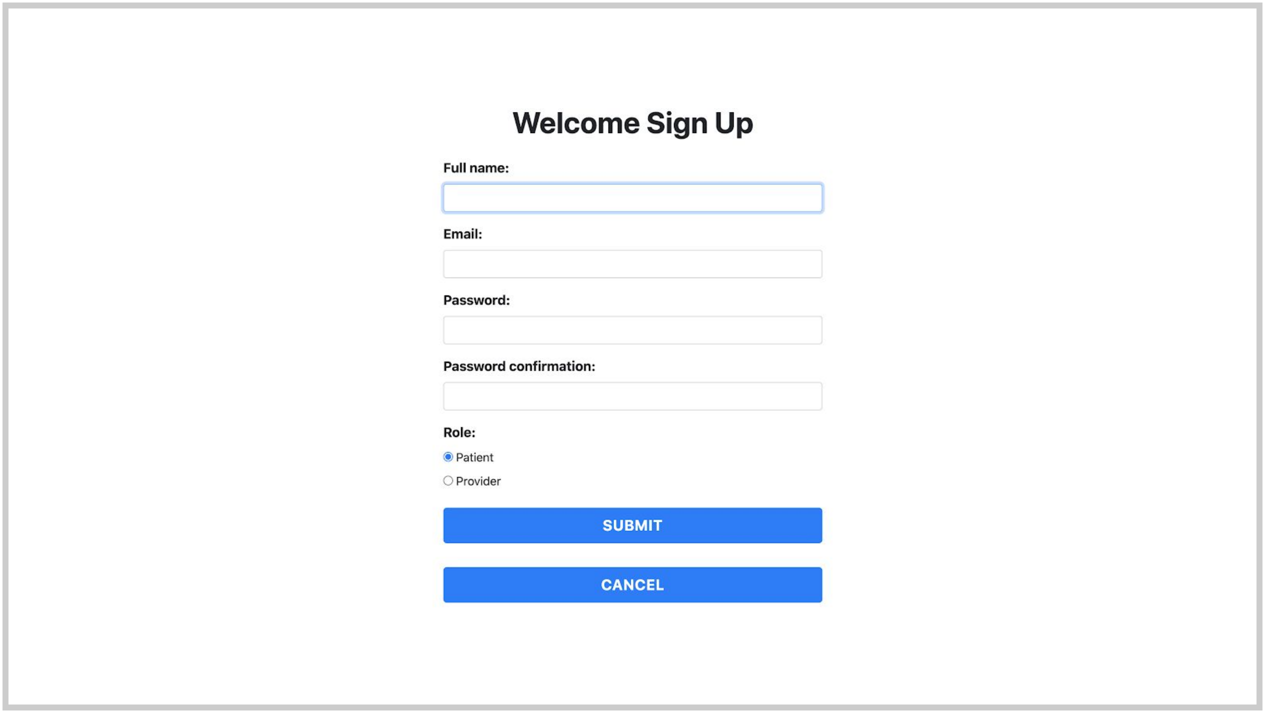
After completing the survey, your health care provider will receive your answers to help guide your care.

2. Feature: update the login page

As a new patient
So that I want to understand the content clearly
I want to change the color scheme, the font size on the login page and use intuitive design

This story is to adjust the login page for patients so that it would be less confusing when patients are trying to login. The whole page's color scheme is redesigned to be clearer for patients who might find the previous version difficult to read. We also adjust the font size of the login page for a similar reason. The design of the login page became more intuitive so that it would be easier for patients to understand every step without any assistance.

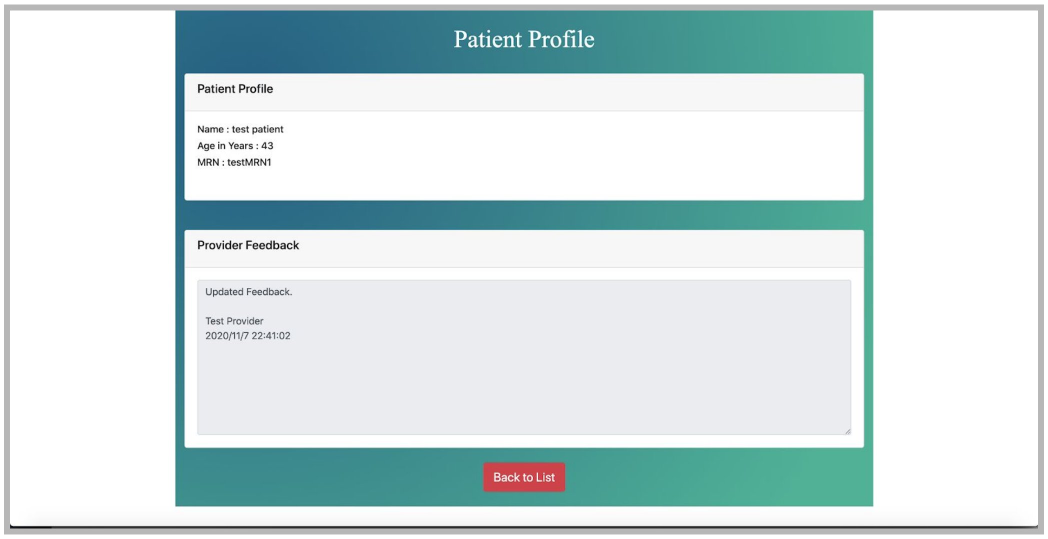




3. Feature: Feedback page for patient

As a patient,
After taking the survey
I want to view the feedback from the provider but nothing more displayed

This story is one of the main functions of the system that is to make it possible for patients to get a reminder and feedback from their provider to help them better understand the importance of medical treatment. As the main purpose of this project is to increase patients' awareness of osteoporosis, this feature is key to such purpose. It is also important that patients should only be able to see their personal information and the feedback from their provider. Their answer to questions should not be available for them after they submit the survey as it might interfere with their future attitude towards certain survey questions and diminish credibility of future surveys.


Patient Profile

Patient Profile

Name : test patient
Age in Years : 43
MRN : testMRN1

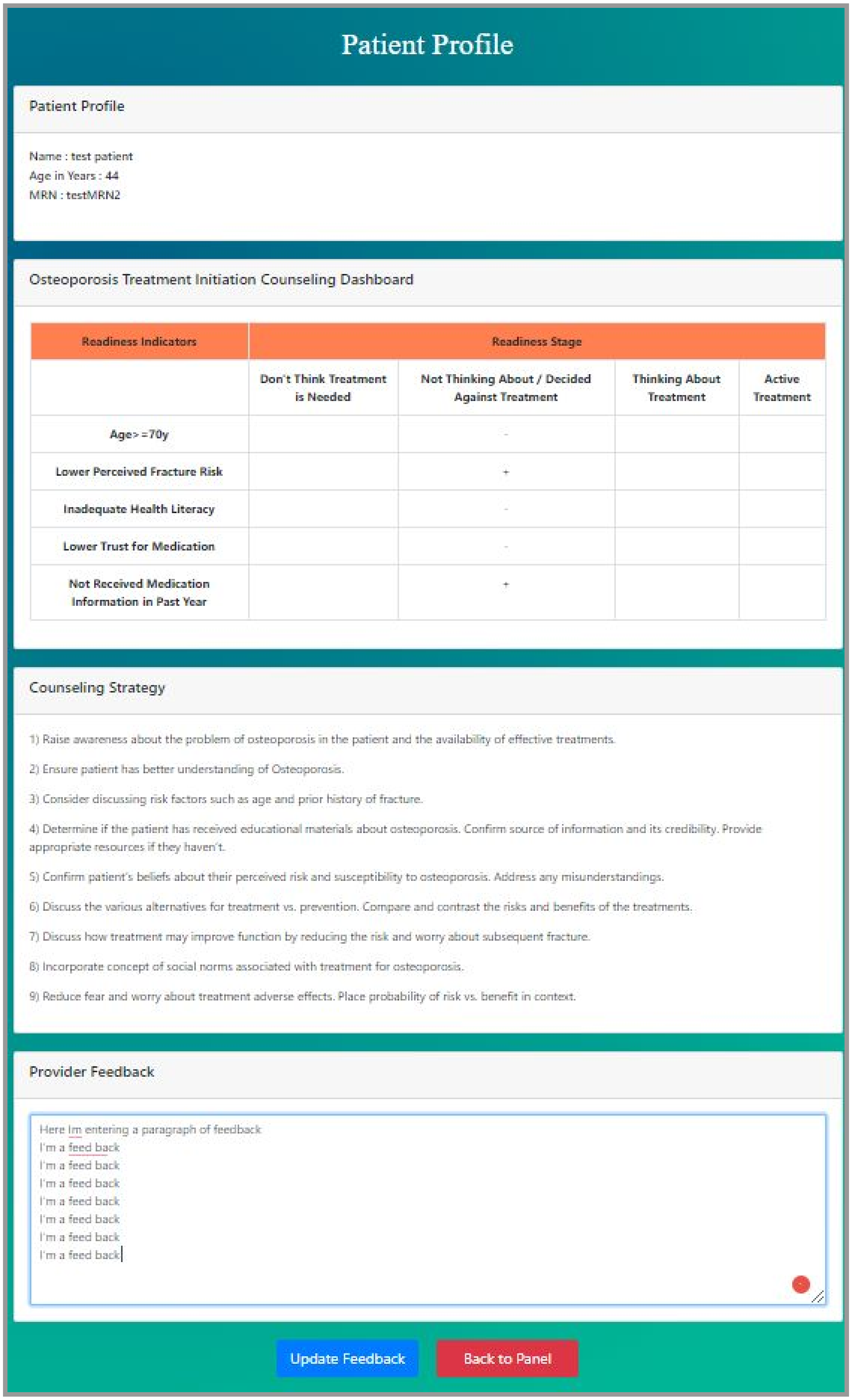
Provider Feedback

Updated Feedback.

Test Provider
2020/11/7 22:41:02

4. Feature: provider can write feedback to patient
   As a provider
   When reviewing each survey
   I want to give feedback based on the survey questions and system generated Strategy

This feature follows the previous feature and is part of the whole feedback function of the system. The provider should be able to review every patient visit he/she has in the provider portal. Also, for each patient visit, the provider should be able to see every detail information of the patient's visit especially every answer to the survey and the auto generated suggestion in the feedback page. These two features should not be visible on the patient's side. The feedback page for providers is similar to the patients side but combined with auto-generated suggestions based on the patient's answer to all the questions. Providers will be able to write feedback based on such suggestions in the feedback page.



Patient Profile

Patient Profile

Name : test patient
Age in Years : 44
MRN : testMRN2

Osteoporosis Treatment Initiation Counseling Dashboard

Readiness Indicators	Readiness Stage			
	Don't Think Treatment is Needed	Not Thinking About / Decided Against Treatment	Thinking About Treatment	Active Treatment
Age>=70y		-		
Lower Perceived Fracture Risk		+		
Inadequate Health Literacy		-		
Lower Trust for Medication		-		
Not Received Medication Information in Past Year		+		

Counseling Strategy

1) Raise awareness about the problem of osteoporosis in the patient and the availability of effective treatments.

2) Ensure patient has better understanding of Osteoporosis.

3) Consider discussing risk factors such as age and prior history of fracture.

4) Determine if the patient has received educational materials about osteoporosis. Confirm source of information and its credibility. Provide appropriate resources if they haven't.

5) Confirm patient's beliefs about their perceived risk and susceptibility to osteoporosis. Address any misunderstandings.

6) Discuss the various alternatives for treatment vs. prevention. Compare and contrast the risks and benefits of the treatments.

7) Discuss how treatment may improve function by reducing the risk and worry about subsequent fracture.

8) Incorporate concept of social norms associated with treatment for osteoporosis.

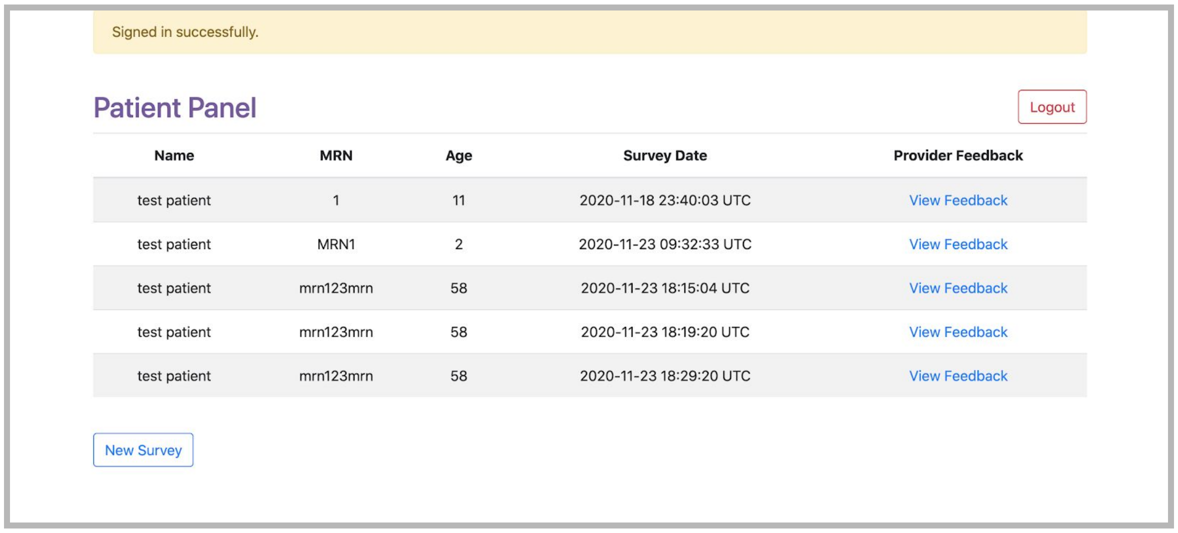
9) Reduce fear and worry about treatment adverse effects. Place probability of risk vs. benefit in context.

Provider Feedback

Here Im entering a paragraph of feedback
I'm a feed back
I'm a feed back
I'm a feed back
I'm a feed back
I'm a feed back
I'm a feed back
I'm a feed back

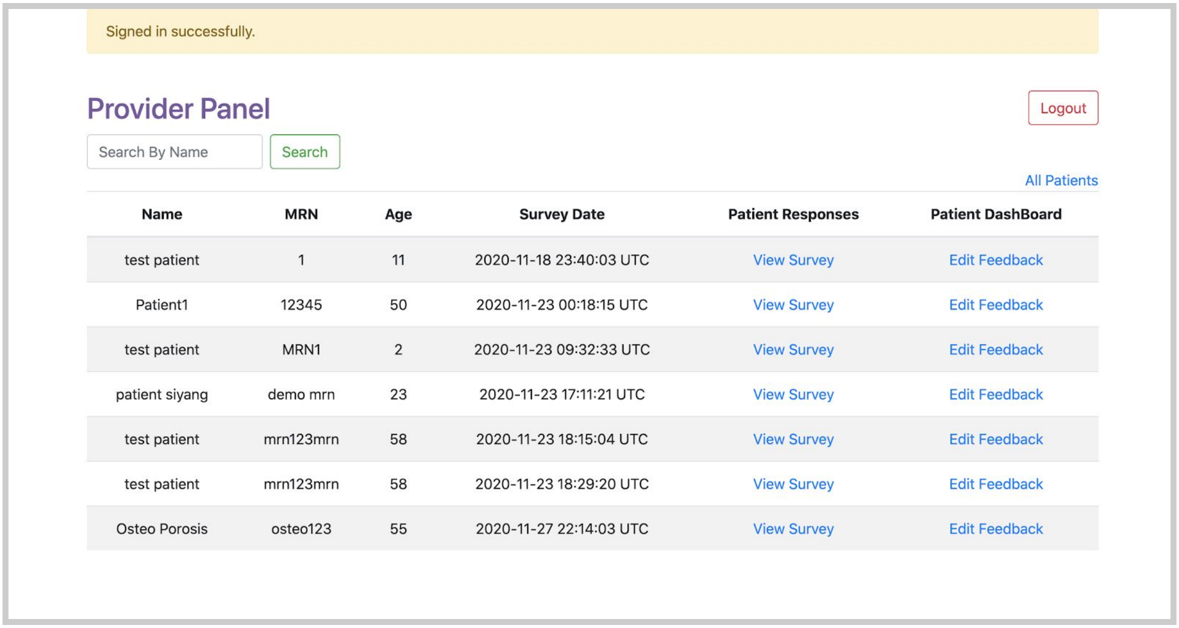
5. Feature: separate the login portal for patient and provider
   As a patient or provider
   When I try to login my account
   I need to know which is the correct portal for me to login

This feature is to further clarify the different functions on the main page. The patient and provider should not be sharing the same login button. The patient should be directed to the “Take the survey” button while providers and admin should use the top right button that is specifically assigned to them. After logging in, patient users will be redirected to the "Patient Panel", while provider users will be redirected to the "Provider Panel".


Signed in successfully.

Patient Panel

Name	MRN	Age	Survey Date	Provider Feedback
test patient	1	11	2020-11-18 23:40:03 UTC	View Feedback
test patient	MRN1	2	2020-11-23 09:32:33 UTC	View Feedback
test patient	mrn123mrn	58	2020-11-23 18:15:04 UTC	View Feedback
test patient	mrn123mrn	58	2020-11-23 18:19:20 UTC	View Feedback
test patient	mrn123mrn	58	2020-11-23 18:29:20 UTC	View Feedback


Signed in successfully.

Provider Panel

Name	MRN	Age	Survey Date	Patient Responses	Patient DashBoard
test patient	1	11	2020-11-18 23:40:03 UTC	View Survey	Edit Feedback
Patient1	12345	50	2020-11-23 00:18:15 UTC	View Survey	Edit Feedback
test patient	MRN1	2	2020-11-23 09:32:33 UTC	View Survey	Edit Feedback
patient siyang	demo mrn	23	2020-11-23 17:11:21 UTC	View Survey	Edit Feedback
test patient	mrn123mrn	58	2020-11-23 18:15:04 UTC	View Survey	Edit Feedback
test patient	mrn123mrn	58	2020-11-23 18:29:20 UTC	View Survey	Edit Feedback
Osteo Porosis	osteo123	55	2020-11-27 22:14:03 UTC	View Survey	Edit Feedback

Issues and Discussion

BDD/TDD process

The BDD process consists of three steps, writing a test, implementing the feature to pass the test, then improving the existing codes. The entire process iterates and keeps improving the code during this process until reaching desirability. We chose the following BDD process to improve our product. By utilizing BDD process, we were able to first convert our user stories into testable features, then following that we were able to develop based on the goal of “developing the code that will pass the feature test.”- this allows us to pinpoint the most efficient way to make desired features, and caught bugs earlier during the development. We achieved high coverage with both cucumber and rspec testing for the entire project, including the part developed by previous groups.

Furthermore, since our project is based on legacy code that didn't develop based on the BDD process(un-likely), we also have to write tests that cover the features developed by previous groups’ iterations. These previous development iterations along with features developed were poorly documented, whereas we had to spend a lot of time reading the raw code in order to understand how the code functions and why it behaved in a certain way. Developing these testing files also helped us understand how previous code functions as well as how to utilize some of the existing functions to make our lives easier. We also found old/redundant code during the testing, allowing us to trim the code making it look more beautiful.

Configuration Management

For configuration management, we strictly follow the standard of rails. We put all the configuration files under the 'config/' folder, such as the API routes configurations and the database settings. For the package management, we used Gem to configure all dependencies.

For Git branch management, every member of our team was asked to create a branch for personal development. And every member worked on their own branch. And once one completes a feature, he/she can create a pull request from their branch to the master branch. And we had created 4 releases in total throughout the whole semester. Each release corresponded to the completion of each iteration.

Issues

a. GitHub

At first, we all worked on the master Git branch. This caused many conflicts and branch merging issues. Therefore, In order to minimize conflicts, we asked every team member to create and work on their own branch. And once one finishes the work, he/she can create a pull request and add other team members as reviewers. If more than two team members reviewed and approved the pull request, then this pull request could be merged. In this way, we minimized the possibility of conflicts and risks of checking in the wrong code.

b. Heroku

Initially, every member of our team created their own Heroku app for deploying the production release. We found it hard to manage all the created apps with different versions. So we decided to create a public Heroku account and all members were only allowed to push the latest merged master branch to the remote.

c. Devise & RailsAdmin

The previous teams had already integrated Devise and RailsAdmin in the legacy code. The main benefits of using these two tools are that they can provide many features for the backend data management and permission control, which would save us lots of time for reinventing the wheels. So our job is to learn how to use these two tools and how to make modifications to the legacy code to meet the customer requirement through Devise and RailsAdmin. At first, we didn't even know where the entrypoint of the system login was. Also when we used the RailsAdmin to manage the data for the admin user of our system, it would occur a page crashing issue. After we read the documents and Wikis under the GitHub repo of these two projects, we eventually solved these issues.