Feature: Services

Feature Developer: Frank Curry
Date Submitted: 4/11/2023

Peer Reviewer: Joshua Gherman
Date Review Completed: 4/16/2023

Major Positives:

1. Validation in the Front End and Back End that meets Business Requirements. Such as valid Company Name, Description, and Phone Number.
2. A response object is created on every layer of the backend. This is a great and realistic idea since we always create response objects for each layer and rely on each function to return a response.
3. The inclusion of different names for the Serv object. It shows attention to the fact that the Serv object will have information about the Creator of the service, the ServInfo being passed, and later the Company information that would be returned to a Regular/Reputable User.
4. Validation for the Response returned by the SqlDao.
5. Limit of characters for a Service Description is limited to 1000. This was not addressed in the BRD, but the decision to add this restriction shows thoughtfulness to implementing the feature.
6. When Modifying a Service, the Service table updates a column of LastMod which shows consideration for the extensibility of the feature.
7. The UI Figma designs are very easy to understand and show the intent of each button click that overlays a pop-up confirmation notification.
8. You made sure to specify having a trycatch on the controller layer. The controller layer is the final line of defense for the backend.
9. Despite BRD stating that the system must reload upon error, it is more practical and beneficial for an error message to be displayed in place of automatically refreshing. This is an improvement to BRD specifications.

Major Negatives:

1. Design does not show validation for Phone Number having a Country Code of +1
    a. Technically this issue is a weak point to bring up, but as something a part of the BRD it should be mentioned that the final implementation should reflect these standards.
2. Design does not show access to the Log Layer
    a. You have shown where you want the logging process to happen for services, but it is good to plan out what you want the logs to contain when logging the Services.
3. Constructor for ServProvService() is no specified.
    a. Understanding what would go into the constructor would be beneficial when implementing as the framework for your Services would already be planned out.
4. There is no Serv Layer
    a. Your code references the Serv object as the parameter that would be inside your functions but there is no lifeline/constuctor for it. It would be good to know what the Serv object would contain.
5. Select (*) should not be implemented
    a. Not everything from the backend needs to be sent back to the user. The Serv object would have personifiable information that not every user should have access to.
6. Split between the Service Creation and Service Request figma designs, there is an inconsistency with how you want a user to Request a feature. Do you want the user to click on a pin to request a service or click on the User Services button to request?
    a. The issue with requesting a service through the User Services button is that there is no “job” or pin associated so you’d have to code a way for the User to link a pin through some other means. I recommend just doing requests by clicking on the pin.
7. Your figma designs are unclear on where a user can input originally what pin types they would accept and where they would be able to update them.

    a. This is important as the BRD requirements need a Service User to have access to these abilities. Your LLDs do show that you intend to meet these requirements, but the inclusion of it in your frontend should be an option.
8. When deleting a Service, your design shows that a result from the database of 1 or 0 will result in an error.Looking at ServiceCreationDeletion and ServiceCreationDelFail.
    a. Your backend has the intent to return a response value of 1 or 0 to show that the operation was a success or not. It is important to be consistent in what you want a success scenario to be. I do like your design decision of having a response that gives a boolean Fail or pass return which then uses the return value from the datastore to double check if the contents were stored correctly.
9. When a Service User is created, the Active column in the database isn't adjusted to show that a user can be tied to one and only one service.
    a. This requirement appears multiple times in the BRD so it is necessary to have check in place. To be fair, this could be included in the Check Security section of the manager, but this is never explicitly put as part of the design.
10. Service Request View does not have a "Back" button
    a. This is a nitpick and I know that you would easily find this error through implementation, but I felt it was necessary to mention. One of the examples of things overlooked in design brought up by Vong was how some groups would forget the inclusion of a Back button feature. Make sure to not forget it!
11. ServiceRequestListView LLD shows that list of services are returned by distance only. BRD states that the system must accept valid zip codes from California.
    a. To be fair, dao.getnearbyservice() may be the method that has validation for the 20 mile radius and how to checks valid zip codes, but I could not find a reference to it within your manager. These Business Rules are needed and need to be tested.

12. Cancelling and Completing a request isn't shown to remove the request from the list of 25 accepted requests.
    a. Your figma designs show that your design has the ability to display a list of 25 accepted requests, but the validation for having 25 and also removing them isn't shown. These are BRD requirements.

Unmet Requirements:

1. Service User is able to limit area where their services can be requested.
    a. There is no way from your front end views for a service user to specify a range in which they will accept job requests and appear as an available service to other users outside of their range.
    b. An extension of this is that the limit is 20 miles based upon location or zip code. Zip code is necessary to be shown due to other requirements but there is no direct requirement stating it is what you will be using during implementation. Limiting the range by location or IP address may be easier.
2. Design does not show validation for Phone Number having a Country Code of +1.
3. Does not validate that one service is tied to one user.
    a. This is a testing scenario for the backend.
4. Description cannot have a maximum of 200 words.
    a. Requirement is impossible to meet due to BRD conflictions. Limit of a service Description is 150 words, so it is impossible to meet the Non-Functional requirement of 200 words maximum.
5. Validating that the system will take no longer than 5 seconds to validate the Service Creation.
6. Missing LLD for scenario: The Service User's account is not updated correctly, resulting in the User still being viewed as Regular by the System.
7. Service Provider View does not show what happens about clicking the "View" button
    a. I understand your intent was probably to display the extra information about the pin like its description, but the way you choose to display that information could vary. Maybe you decide that clicking the view button takes you to another view rather than a pop-up.
8. Service Request View does not show what happens when clicking the "Cancel" button

    a. Same issue with the view button, but you have a LLD describing its success scenario. The frontend design should be more explicit.
9. System must allow a service to be recommended or not by a User who has requested the service
    a. This is a MAJOR component that I feel was overlooked in your design. Other issues are honestly nitpicks, but I didn't see where a user would be allowed to recommend services in LLDs or Figma design. Admins do see the recommendation # of services. This needs to be taken into consideration when implementing.
10. System must send a notification to the user no later than .4 seconds after the request was received by the server
    a. The uniqueness of the scenario implies that the timer needs to start once a response is returned from the database and ends when the User is allowed to see the response.
11. System should allow a user to cancel a service up to 3 hours before the scheduled cleaning.
    a. This requirement is a bigger deal than it originally seems. Each Service Request needs a DateTime column for the appointment.
12. System shall have Service Users able to accept multiple requests that have conflicting times with other requests

Design Reccomendations:

1. The First Design decision I want to address is with how the timer on the Services work.
    a. Due to the requirement of “System should allow a user to cancel a service up to 3 hours before the scheduled cleaning”, this means that a User must be able to set a time or date for their service cleaning.
        i. Code allowing for a User to set a date(specific to the hour) for when they want the cleaning
        ii. Code allowing for a Service User to see the list of requests and their appointments.
            1. This could be information shown when they click the “View” button
2. The use of Select * is not good implementation.
    a. You want to plan out how your constructors work so that you don’t give away too much information when it is not needed
        i. You have planned out for your Serv object to have Server, ServInfo, and Company.
            1. When passing in the ServInfo, it will contain the all the information since it is the creation of the Service.
            2. You’ll have to return the Company to a User, but they don’t need all the information. Information not needed might be the Service User’s ID.
3. During Service Creation, it is necessary to have an option to choose accepted pin types.
    a. You already planned out the LLD for how you want to approach the creation of this functionality, but there needs to be a way for a user to interact this way in the Frontend.
        i. This could be an added field in the Service Creation views. Looking at the implementation of pin types, pin types are chosen by writing a number 1-4 when creating a pin. Similarly, you could take a number as a respected pin type
        ii. You also want to adjust what happens when an Admin User clicks on to edit a Service.

4. Recommendation functionality needs to be implemented.
    a. A User needs to be able to recommend a Service, but the BRD doesn't specify when.
        i. I recommend for easy implentation, for the recommend button to be available a User is viewing a list of Services.
        ii. The harder route would have a User being able to access a list of prior requested services and then be able to click some sort of Reccomendation button.
            1. This would probably have to be its own view.
5. When trying to get a user's location, you may make use of the Google maps api with its Geocoder function. https://developers.google.com/maps/documentation/javascript/reference/geocoder
    a. The Geocoder function allows you to get the lat and longitude, which will allow you to find the zip code