ChocAn System

Software Engineering

Final Project

KATO JR., Alberto Sussumu

OLIVEIRA, Rafael Silvestre de

RIBEIRO, Jorge Luís Melo

Waukesha, December 10, 2015

Contents

**Communications Plan**

**How will you communicate with each other?**

Personal meetings on Friday at 2PM.

Daily emails on demand.

**Where are the meetings going to happen?**

In the learning commons, Todd Wehr Library.

**How will you assign tasks?**

The tasks will be divided by the amount of use-cases, that is, each member of the project is responsible for a third part of the project, covering every single artifact of the designated part (requirements, analysis, design, implementation and testing).

In order to keep everyone synchronized, the regular meetings will have as goal to keep everyone in track and integrate the modules that each other is responsible for.

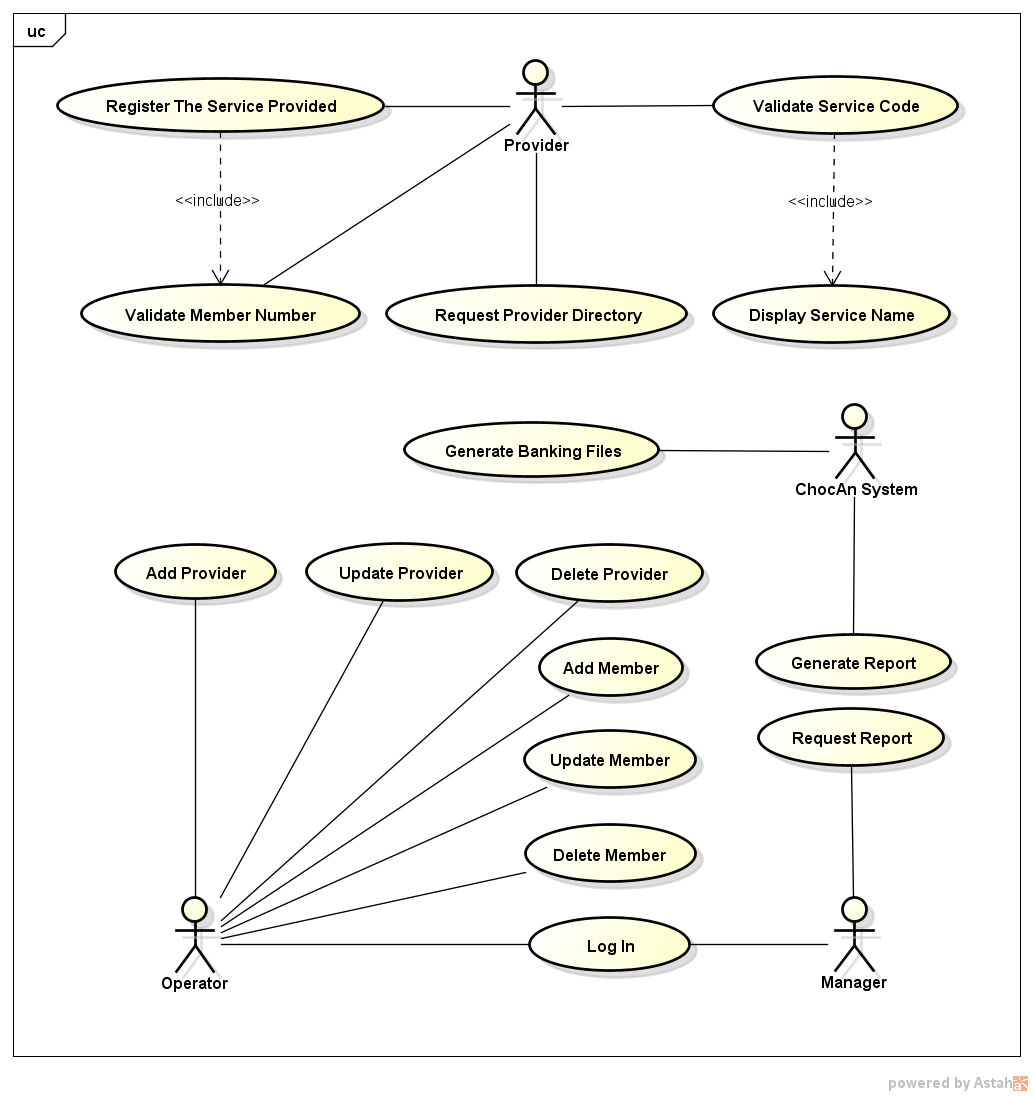
**Where will the work be stored?**

All the work will be stored in a distributed version control system (GitHub) to allow decentralized management of the software.

**How to track the work done?**

Utilizing Gantt Chart.

**Use Case Diagram**



**Use Case Specifications**

1. **Validate Member Number**

**Brief Description:** *the provider swipes the member card through the card reader on the terminal and the process of member identification is started.*

**Main Actor**: Provider

**Pre-condition:**

1. None.

**Basic flow:**

1. Provider swipes member’s card;
2. The system verifies the member number;
3. The system displays “Validated” message;
4. The use case ends successfully.

**Alternative flows:**

1. **Alternative #1 to Step 3.** Invalid number: The number is not valid anymore. A message is displayed “Invalid number”;
2. The use case is finished unsuccessfully.
3. **Alternative #2 to Step 3.** Member suspended: The member is suspended due to late fees. A message is displayed “Member suspended”;
4. The use case is finished unsuccessfully.

**Post-condition:**

1. In case of success, the login is validated, otherwise login fails.
2. **Register The Service Provided**

**Brief Description:** *the provider enters the information of the service provided in order to track the payment afterwards.*

**Main Actor**: Provider

**Pre-Conditions:**

1. Provider must be logged in the system, member must be validated, and the service code must be validated.

**Basic flow:**

1. The system display a form with text boxes waiting for the information;
2. The process starts in the field date. The provider inserts the date of the service;
3. The provider clicks on “Next” button;
4. The process continues with the fields time, date the service was provided, member name and number, service code, and fee to be paid;
5. The provider clicks on “End” button after inserts the fee;
6. The use case ends successfully.

**Alternative flows:**

1. **Alternative #1 to step 2 and 4.** The provider inserts an invalid data for the field and clicks on “Next” button: the system shows an alert informing what happened;
2. The provider clicks “OK” acknowledging the occurrence;
3. The system redirects to the field insertion screen;
4. The use case is finished unsuccessfully.

**Post-condition:**

1. The provider is redirected to the system’s menu.
2. **Validate Service Code**

**Brief Description:** *the service code is entered by the provider, then a validation process is started by the ChocAn System Server.*

**Main Actor**: Provider

**Pre-Conditions:**

1. Provider must be logged in the system.

**Basic flow:**

1. The system display a text box waiting for the service code;
2. The provider inserts the service code to be validated;
3. The provider clicks on “Validate” button;
4. The system looks for the inserted code in the designated table;
5. The system finds the information regarded to the data informed;
6. The system starts use case “4. Display Service Name”;
7. The use case ends successfully.

**Alternative flows:**

1. **Alternative #1 to step 5**. The system does not find any information in the designated table: the system shows an alert informing what happened;
2. The provider clicks “OK” acknowledging the occurrence;
3. The system redirects to the code insertion screen.
4. The use case is finished unsuccessfully.

**Post-condition:**

1. The code is passed to the use case:”4. Display Service Name”.
2. **Display Service Name**

**Brief Description:** *the system recognizes the service code provided and shows the service name for confirmation.*

**Main Actor**: Provider

**Pre-condition:**

1. Requires success response from the use case “3. Validate Service Code” to be initialized.

**Basic flow:**

1. The system displays service name;
2. The system displays a confirm box;
3. The provider clicks “yes”;
4. The provider can type comments;
5. The use case ends successfully.

**Alternative flows:**

1. **Alternative #1 to step 3**. The providers clicks “No”: The system redirects to the type code screen again.
2. The use case is finished unsuccessfully.

**Post-condition:**

1. The service name is displayed correctly and confirmed.
2. **Request Provider Directory**

**Brief Description:** *the provider requests the ChocAn System Server his/her list of services along with service names, codes, and fees.*

**Main Actor**: Provider

**Pre-Conditions:**

1. Provider must be logged in the system.

**Basic flow:**

1. The provider selects the option “Request Provider Directory” on system’s menu;
2. The system displays a confirm box;
3. The provider clicks “Yes”;
4. The system generates the Provider Directory;
5. The system displays an alert confirming the send;
6. The use case ends successfully.

**Alternative flows:**

1. **Alternative #1 to step 3**. The provider clicks “No”: The system redirects to the menu;
2. The use case is finished successfully.

**Post-condition:**

1. The provider is redirected to the system’s menu.
2. **Generate Report**

**Brief Description:** *generates the following reports: services provided by each provider, services utilized by each member, and summaries for payable accounts.*

**Main Actor:** ChocAn System

**Pre-Conditions:**

1. Use Case is run every midnight on Fridays.

**Basic flow:**

1. The following reports must be generated:
   1. Services Utilized by Each Member

The system gathers all information regarded to the services utilized. For each member, these attributes are retrieved or calculated:

Member name;

Member number;

Member street address;

Member city;

Member state;

Member ZIP code.

For each service utilized:

Date of service;

Provider name;

Service name.

* 1. Summaries for Payable Accounts

The system gathers all information about all the providers to be paid that week. This report includes the following attributes:

Provider name;

Provider number;

Provider street address;

Provider city;

Provider state;

Provider ZIP code;

Number of Consultations;

Total Fee.

Along with each individual information, the system also prints:

Total Number of Active Providers;

Total Number of Consultations;

Overall Total Fee.

* 1. Services Provided by Each Provider

The system gathers all information regarding the services provided. For each provider, these attributes are retrieved or calculated:

Provider name;

Provider number;

Provider street address;

Provider city;

Provider state;

Provider ZIP code.

For each service provided:

Date of service;

Date and time the data were received;

Member name;

Member number;

Service code;

Fee to be paid.

General attributes:

Total number of consultations;

Total fee for week.

1. The use case ends successfully.

**Alternative flows:**

1. None. Everything is automatic.

**Post-condition:**

1. All the reports are send individually to each provider/member/manager that it is intended to.
2. **Generate Banking File**

**Brief Description*:*** *an EFT data is written to disk, so that banking computers will ensure the provider’s payment.*

**Main Actor:** ChocAn System

**Pre-condition:**

1. None.

**Basic flow:**

1. The system generates EFT;
2. The system validate all the files;
3. The system write everything in the disk;
4. The system permits that bank computer access EFT data;
5. The use case ends successfully.

**Alternatives flows:**

1. None. Everything is automatic.

**Post-condition:**

1. The system will provide proper data to the bank computer.
2. **Request Report**

**Brief Description:** *requests the ChocAn System Server a file containing the services provided by each provider.*

**Main Actor:** Manager

**Pre-Conditions:**

1. The manager must be logged in to the system.

**Basic flow:**

1. The manager selects the option “Request Services Provider Report” on system’s menu;
2. The system displays a confirm box;
3. The manager clicks “Yes”;
4. The system generates the report;
5. The system displays the report with an option to print;
6. The use case ends successfully.

**Alternative flows:**

1. **Alternative #1 to step 3.** The manager clicks “No”;
2. The system redirects to the menu.
3. **Alternative #2 to step 5.** The manager clicks “Print”: The system prints the report;
4. The use case is finished successfully.

**Post-condition:**

1. The manager is redirected to the system’s menu.
2. **Add Member**

**Brief Description:** *The operator adds a member to the ChocAn Server System.*

**Main Actor:** Operator

**Pre-condition:**

1. The operator must be logged in to the system.

**Basic flow:**

1. The system requires that operator fill a form with the new member info;
2. Operator fills the fields (Name, State, City, Street, and ZIP Code);
3. Operator click on save button;
4. The use case ends successfully.

**Alternative flows:**

1. **Alternative #1 to Step 2.** Invalid zip code: Error message is displayed on the screen;
2. The system requires the operator to type the zip code until a valid one is inserted;
3. User caser Resumes on the form screen.

**Post-condition:**

1. The system will save the new member in the database successfully.

1. **Update Member**

**Brief Description:** *the operator can edit the information of an existing member.*

**Main Actor:** Operator

**Pre-Conditions:**

1. The operator must be logged in to the system.

**Basic flow:**

1. The system shows a list containing the members;
2. The operator selects a member of the list;
3. The operator clicks the “Update” button;
4. A confirmation screen says “Are You Sure?”;
5. The operator clicks “Yes”;
6. The system loads the form with the info of that member;
7. The operator update the needed fields;
8. The operator clicks on “Update” button;
9. A confirmation screen says “Are You Sure?”;
10. The operator clicks “Yes”;
11. The system updates the selected member;
12. The use case ends successfully.

**Alternative flows:**

1. **Alternative #1 to step 4.** The operator clicks “No”;
2. The system redirects to the members list.
3. **Alternative #2 to step 7.** The operator inserts an invalid content for the field and clicks on “Next” button;
4. The system shows an alert informing what happened;
5. The provider clicks “OK” acknowledging the occurrence;
6. The system redirects to the field insertion screen.
7. **Alternative #3 to step 9.** The operator clicks “No”: the system redirects to the field insertion screen;
8. The use case is finished unsuccessfully.

**Post-condition:**

1. The system will save the updated member in the database successfully.
2. **Delete Member**

**Brief Description:** *the operator deletes an existing member from the system.*

**Main Actor:** Operator

**Pre-Conditions:**

1. The operator must be logged in to the system.

**Basic flow:**

1. The system shows a list containing the members;
2. The operator selects a member of the list;
3. The operator clicks the “Delete” button;
4. A confirmation screen says “Are You Sure?”;
5. User clicks “Yes”;
6. The system deletes the selected member;
7. This Use Case ends successfully.

**Alternative flows:**

1. **Alternative #1 to step 5**. User clicks ”No”;
2. The system redirects to the members list.

**Post-condition:**

1. The system has the selected member deleted.

1. **Add Provider**

**Brief Description:** *the operator adds a new provider record to the system by inserting the data correctly.*

**Main Actor:** Operator

**Pre-Conditions:**

1. The operator must be logged in to the system.

**Basic flow:**

1. The system displays the “Insert New Provider” screen;
2. The operator fills in the following fields:

Provider Name;

Provider Number;

Provider street address;

Provider city;

Provider state;

Provider ZIP code.

1. The operator clicks the button “Submit”;
2. The system stores the informed data in the database;
3. The use case ends successfully.

**Alternative flows:**

1. **Alternative #1 to step 3**. The operator clicks the button “Clear”;
2. All the inserted information will be cleared;
3. The cursor focus is changed to the first field;
4. The use case ends unsuccessfully.

**Post-condition:**

1. The new provider record is inserted in the database.

1. **Update Provider**

**Brief Description:** *the operator edits the information of a provider record.*

**Main Actor:** Operator

**Pre-Conditions:**

1. The operator must be logged in to the system.

**Basic flow:**

1. The system shows a list containing the providers;
2. The operator selects a provider of the list;
3. The operator clicks the “Update” button;
4. A confirmation screen asks “Are You Sure?”;
5. The operator clicks “Yes”;
6. The system loads the form with the info of that provider;
7. The operator update the needed fields;
8. The operator clicks on “Update” button;
9. A confirmation screen says “Are You Sure?”;
10. The operator clicks “Yes”;
11. The system updates the selected provider;
12. The use case ends successfully.

**Alternative flows:**

1. **Alternative #1 to step 4.** The operator clicks “No”;
2. The system redirects to the providers list.
3. **Alternative #2 to step 7.** The operator inserts an invalid content for the field and clicks on “Next” button;
4. The system shows an alert informing what happened;
5. The provider clicks “OK” acknowledging the occurrence;
6. The system redirects to the field insertion screen.
7. **Alternative #3 to step 9.** The operator clicks “No”;
8. The system redirects to the field insertion screen;
9. The use case is finished unsuccessfully.

**Post-condition:**

1. The system will save the updated provider in the database successfully.

1. **Delete Provider**

**Brief Description:** *The operator deletes a provider from the ChocAn Server System.*

**Main Actor:** Operator

**Pre-Conditions:**

1. The operator must be logged in to the system.

**Basic flow:**

1. The system displays a list with all providers;
2. The operator select a specific provider;
3. The operator clicks on delete button;
4. The system displays a message (“Are you sure?”);
5. The operator clicks “yes”;
6. The use case ends successfully.

**Alternative flows:**

1. **Alternative #1 to Step 5.** Operator clicks “NO”;
2. Use case resumes on the list screen until a next action.

**Post-condition:**

1. The system will delete a specific provider from ChocAn database.

1. **Log In**

**Brief description:** *The login provides access to the system.*

**Main Actor:** Operator

**Pre-Conditions:**

1. The operator must be logged in to the system.

**Basic flow**

1. The system displays the login form;
2. The user fills the fields (Login and password);
3. The user clicks in the login button;
4. The system validates user login;
5. The use case ends successfully.

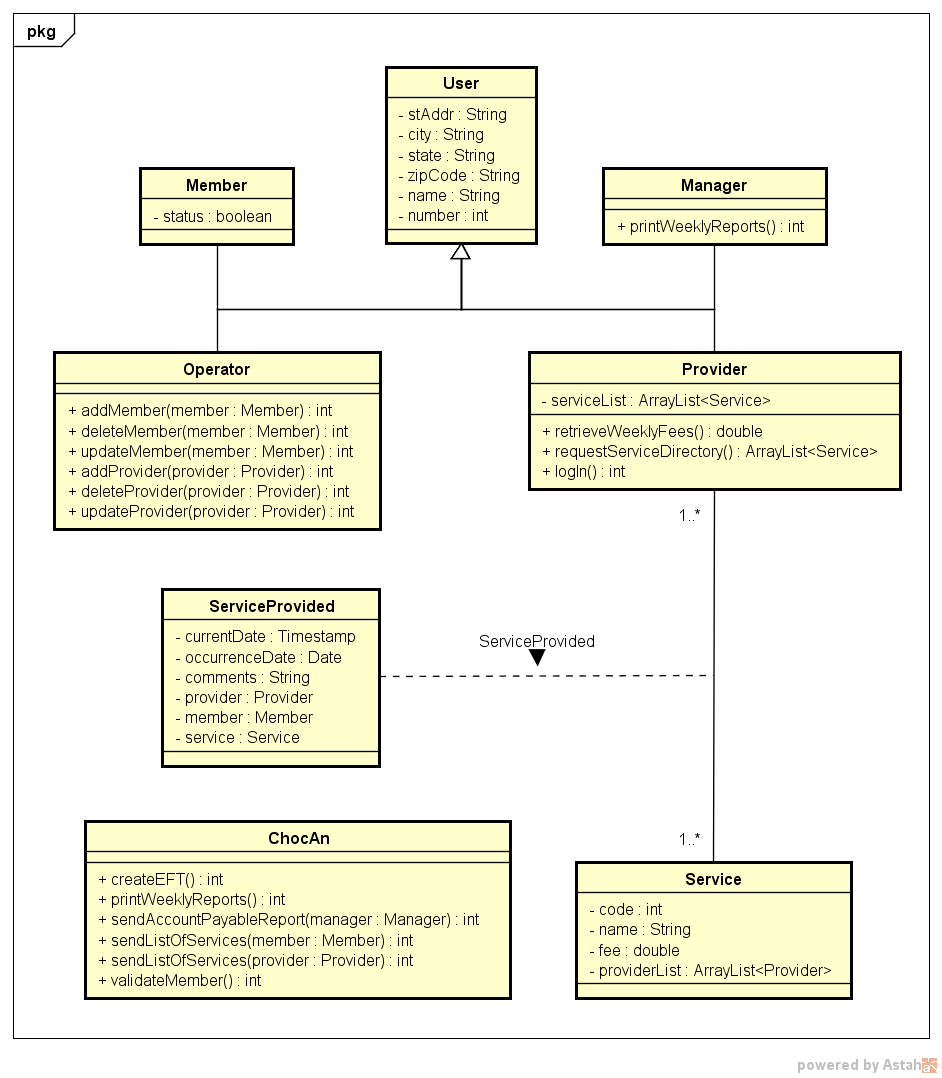
**Alternative flows:**

1. **Alternative #1 to the step 4.** Username and/or password not recognized;
2. An error message is displayed;
3. The use case ends unsuccessfully.

**Post-condition:**

1. The Operator, or the manager will be redirected to their respective page.

**Class Diagram**



**Sequence Diagram**

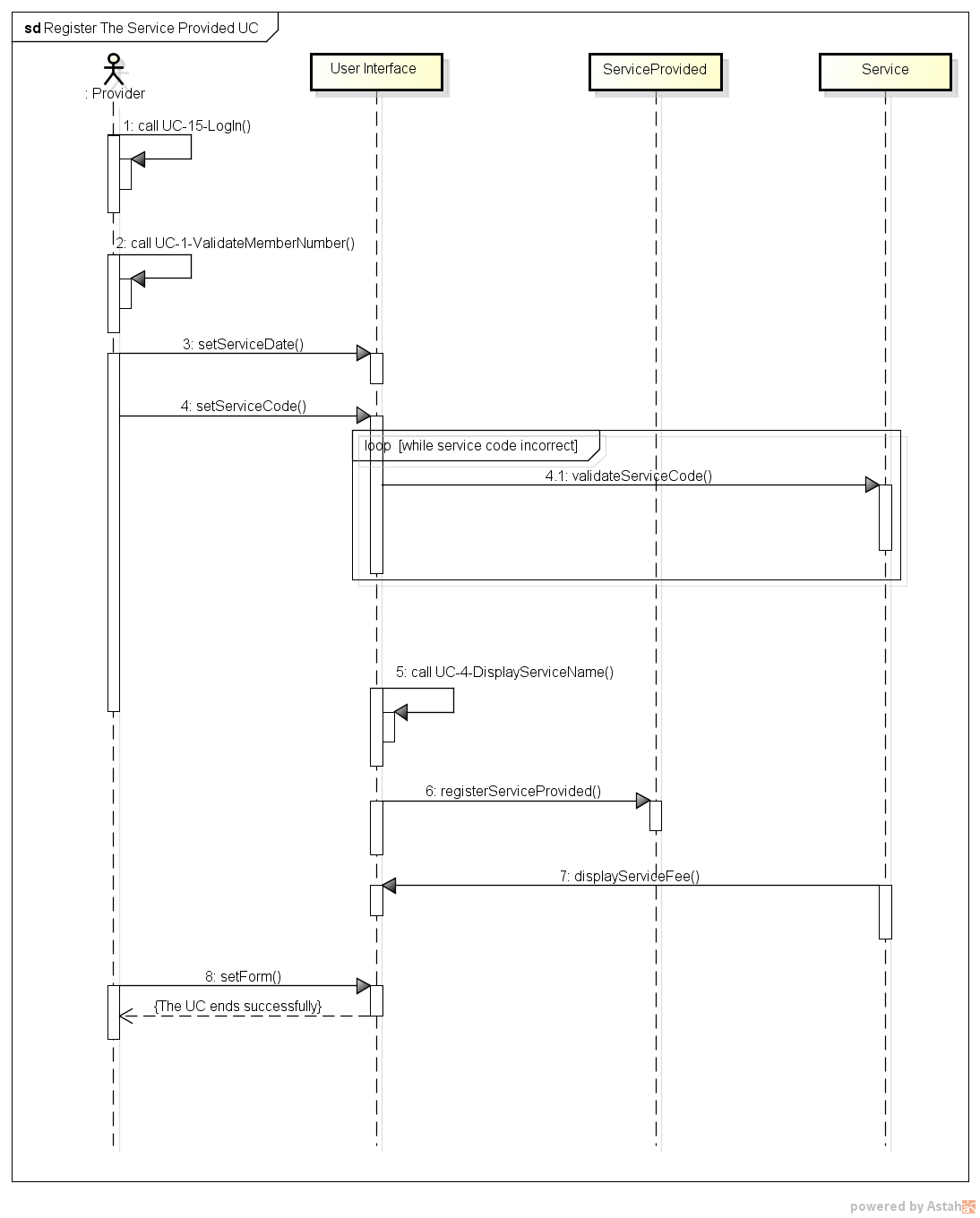
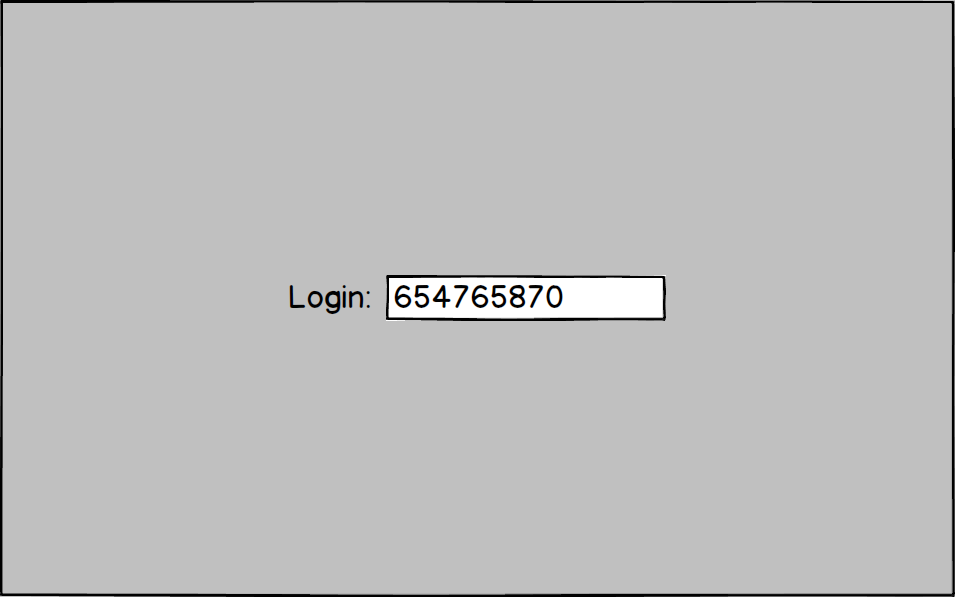


Table of Assigned Tests for each team member:

|  |  |
| --- | --- |
| **Assigned Tests** | |
| **Tester** | **Responsibility** |
| Alberto | He will be responsible for the following range of tests: 01 - 08 |
| Jorge | He will be responsible for the following range of tests: 09 - 15 |
| Rafael | He will be responsible for the following range of tests: 16 - 23 |

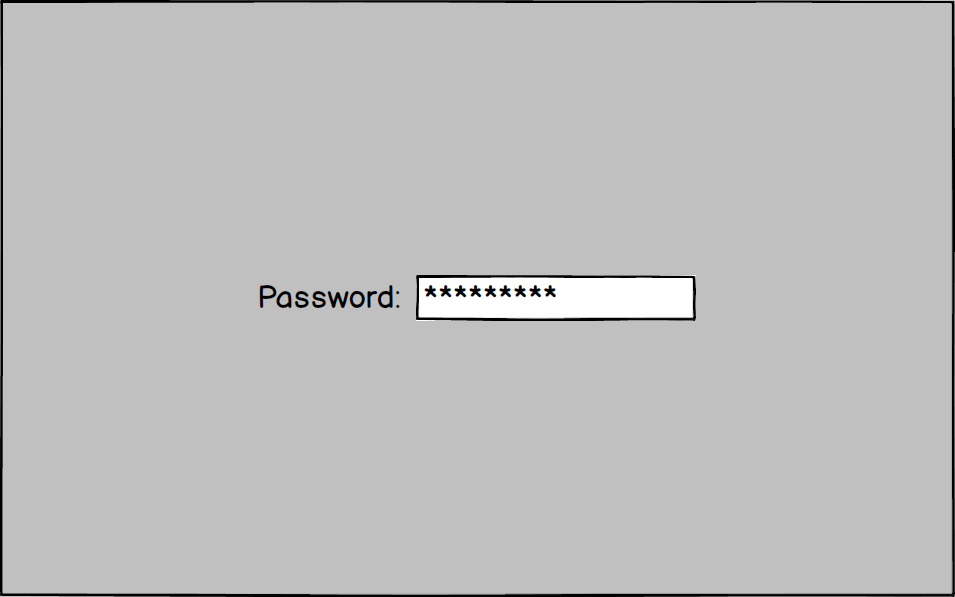
1. **Login**
   1. User type the login number.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| User types wrong login number. | 123456 | Shows message: Invalid login. Asks for login number again. |  |
| User types correct login number. | 654765870 | Redirects password screen |  |
| Provider types blank. | “” | Shows message:  Type a valid login. |  |

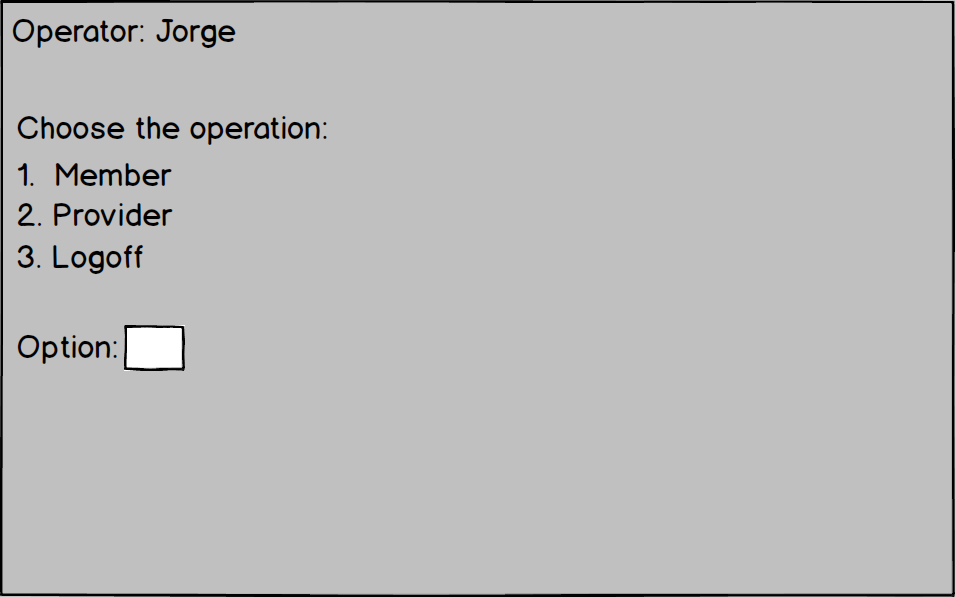
1. **Password**
   1. User types password.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Wrong password. | dragon12345 | Shows message: Invalid password. Asks for password again. |  |
| User type: Operator  &&  Valid password. | operator12 | Redirects to operator main screen |  |
| User type: Provider &&  Valid password. | provider12 | Redirects to provider main screen. |  |
| User type: Manager  &&  Valid password. | manager12 | Redirects to manager main screen. |  |

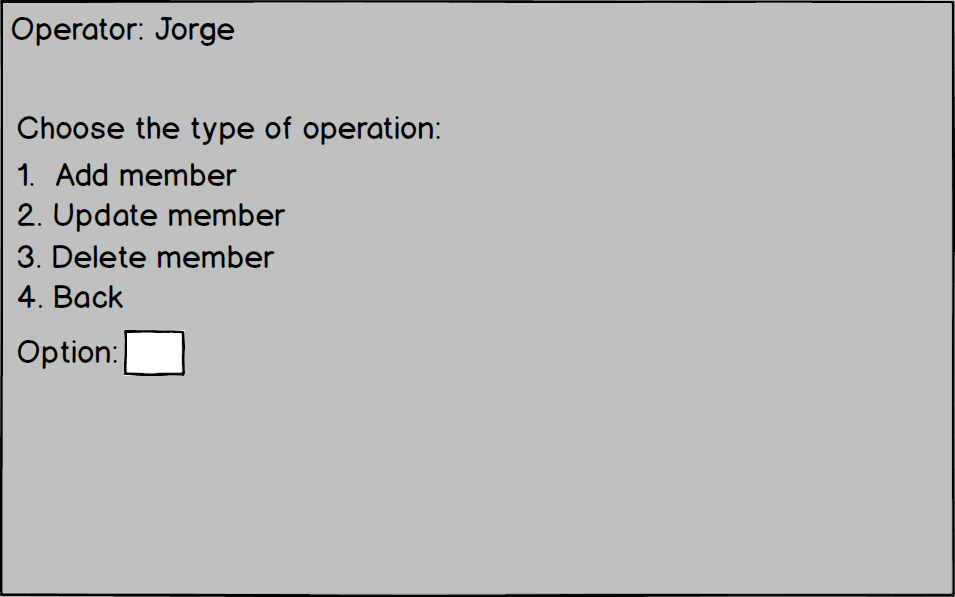
1. **Operator Main Screen**
   1. Operator types operation number.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Operator types inexistent option. | 98 | Shows message:  This operation is not valid. |  |
| Operator types blank. | “” | Shows message:  Type a valid operation |  |
| Operator selects member operations. | 1 | Redirects to screen: Member - operations |  |
| Operator selects provider operation. | 2 | Redirects to screen:  Provider - operations |  |
| Operator selects logoff operation. | 3 | Redirects to login screen |  |

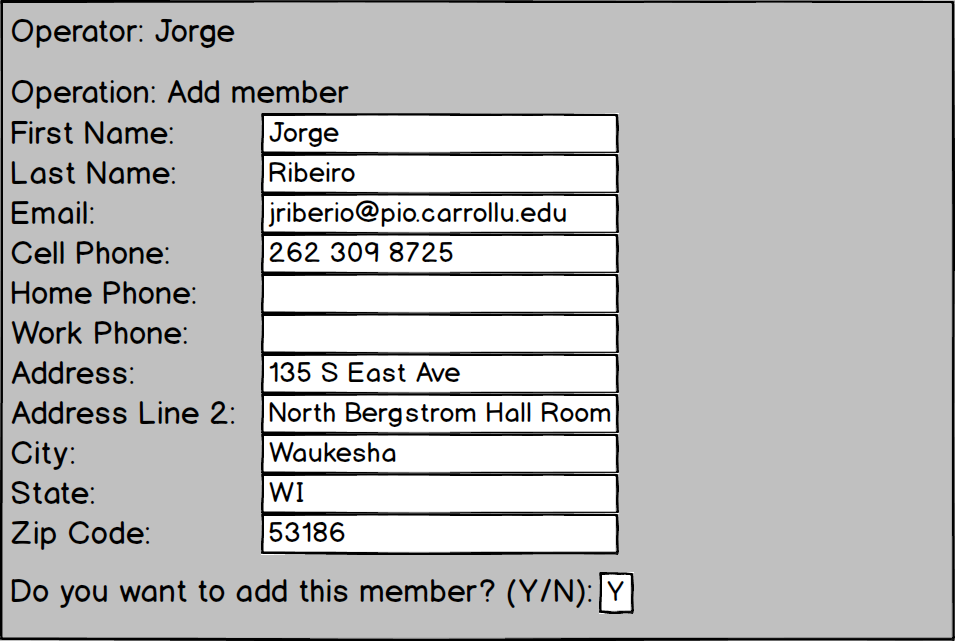
1. **Member Operations Screen**
   1. Operator types one member-operation number.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Operator wants to Add a member. | 1 | Redirects to screen Add a member. |  |
| Operator wants to update a member. | 2 | Redirects to screen Update a member. |  |
| Operator wants to delete a member. | 3 | Redirects to screen  Delete a member. |  |
| Operator wants to return to previous screen. | 4 | Redirects to screen  Operator main screen. |  |
| Operators types an invalid number. | 77 | Shows message: Invalid operation. |  |
| Operator inputs a not valid number. | a | Shows message:  Input a valid number. |  |
| Operator keys ‘enter’ (blank value) | “” | Shows message:  Type something valid. |  |

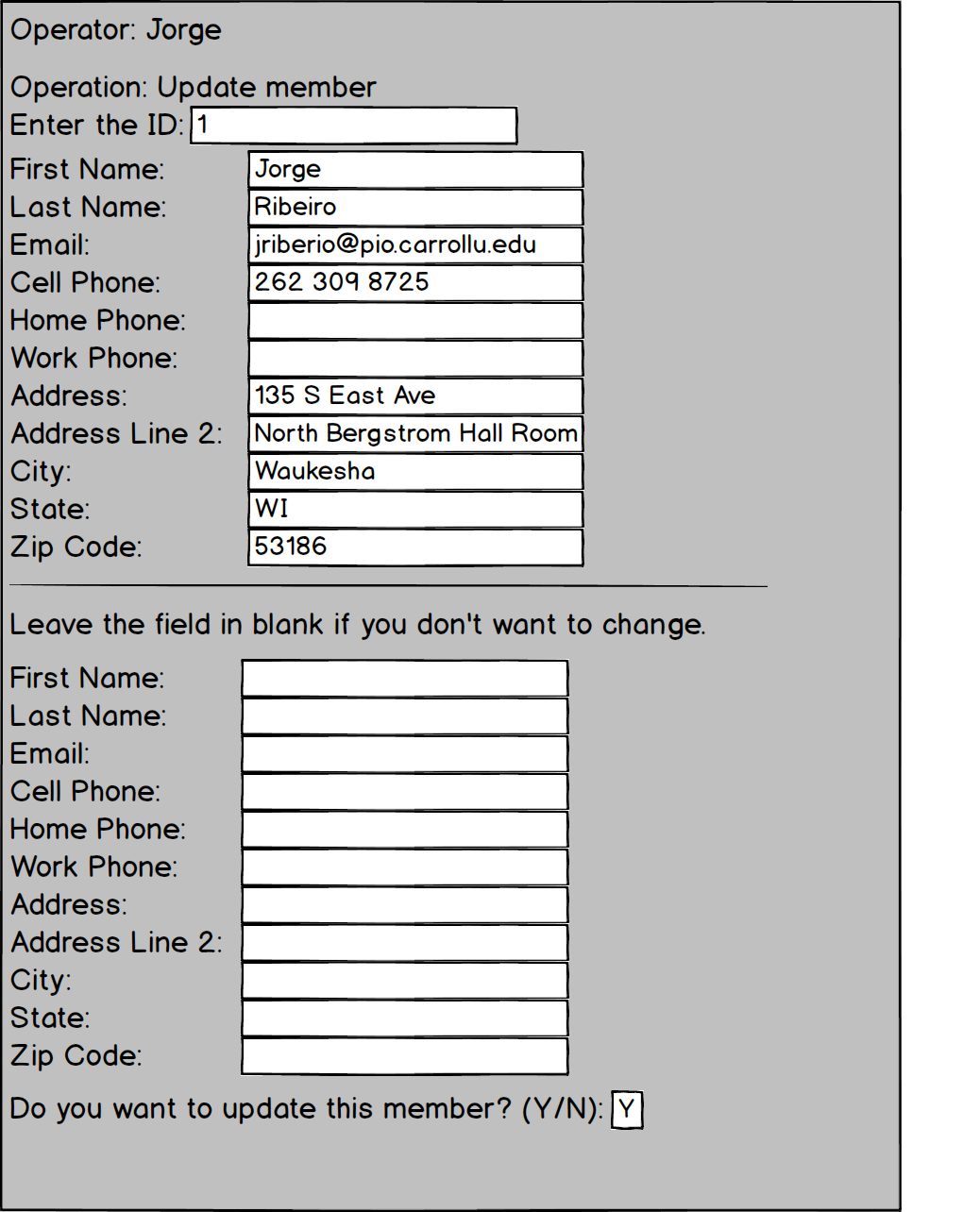
1. **Add a Member**
   1. Operator types First Name (Required)
   2. Operator types Last Name (Required)
   3. Operator types Email (Required)
   4. Operator types cell/Home/Work (at least one Required)
   5. Operator types Address (Required)
   6. Operator types Address Line 2 (Optional)
   7. Operator types City (Required)
   8. Operator types State (Required)
   9. Operator types Zip Code (Required)



* 1. Test Table

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Operator left a blank field in any of the required fields. | “” | System asks for the operator fill the field again. |  |
| All the required fields are filled properly and operatory types Y-y. | All member information and Y | Saves a new member record in the database. |  |
| Operator types an invalid Zip Code. | 12 | Requests for a new Zip Code. |  |
| Operator types a wrong state abbreviation. | ZZ | Requests for new US state abbreviation. |  |
| Operator does not want to add the member anymore. | N-n | Returns to member operation screen without saving anything in the database. |  |
| Operator types a different letter than (Y-N) in the question. | i | Shows message:  Invalid answer. Request for other letter input. |  |

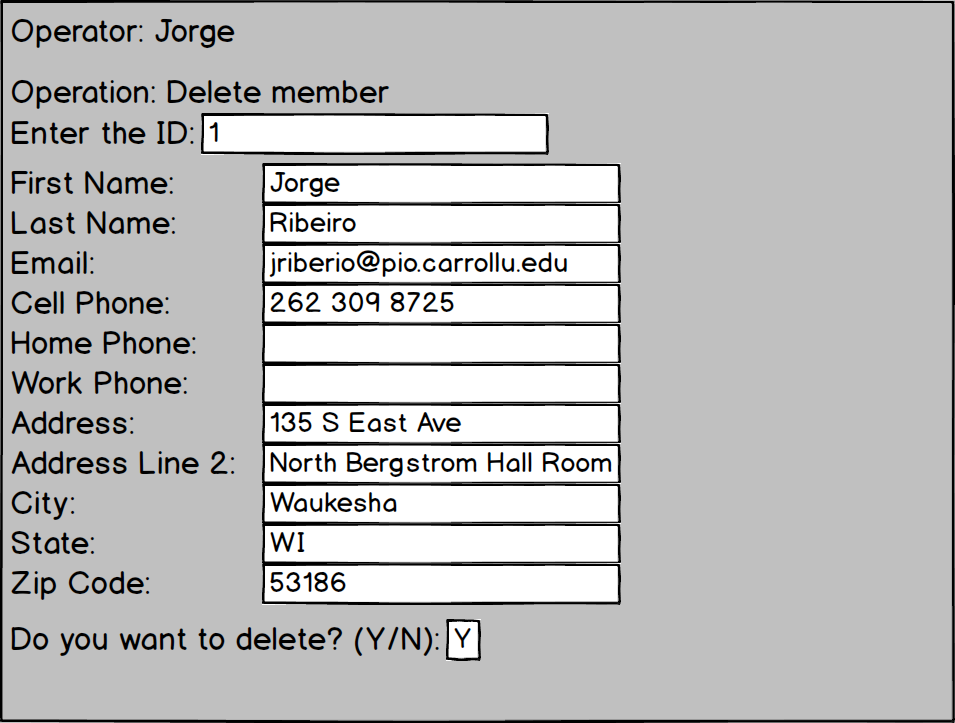
1. **Update a Member**
   1. Operator types the member ID.
   2. System displays all member information.
   3. Operator leave the field in blank for no changers, otherwise fill with new data.
   4. Operator types Y



* 1. Test Table

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Fields are filled properly and operatory types Y-y. | All member requested information and Y | Updates member record in the database. |  |
| Operator types an invalid Zip Code. | 12 | Requests for a new Zip Code. |  |
| Operator types a wrong state abbreviation. | ZZ | Requests for new US state abbreviation. |  |
| Operator does not want to add the member anymore. | N-n | Returns to member operation screen without updating anything in the database. |  |
| Operator types a different letter than (Y-N) in the question. | i | Shows message:  Invalid answer. Request for other letter input. |  |

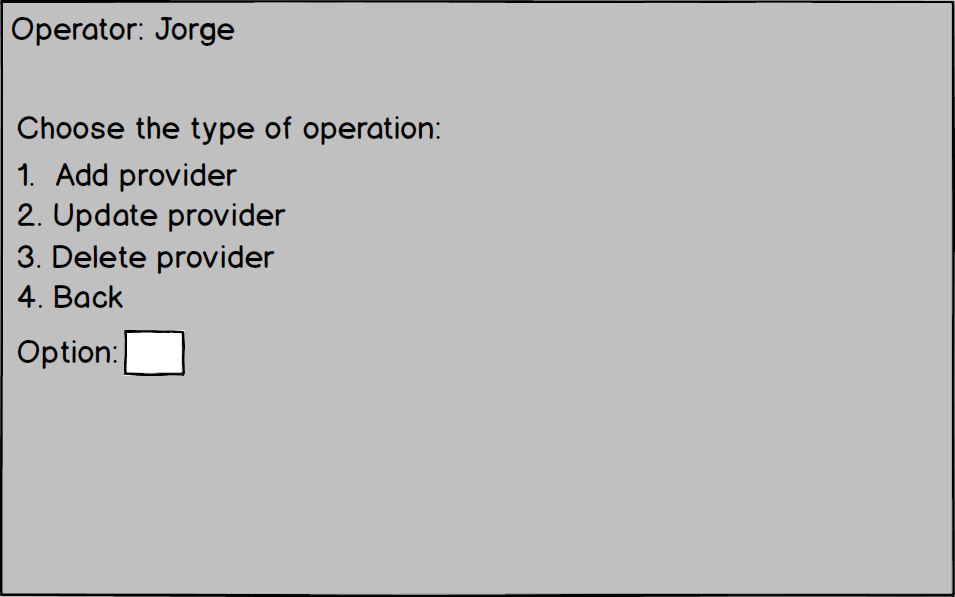
1. **Delete Member**
   1. Operatory types the member ID
   2. System displays all the member information.
   3. Operators types Y and deletes the member from the system.



* 1. Test Table

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Operator types invalid member ID | 83729 | Show message: invalid member ID. |  |
| Operator types valid member ID | 123456 | Display all member information |  |
| Operator types Y-y to the delete question. | Y-y | System deletes the member record from the database |  |
| Operator types N-n to the delete question | N-n | System redirects operator to member operation screen |  |
| Operator types invalid letter for the delete question. | 3 | System display message: invalid input and asks for the answer again. |  |

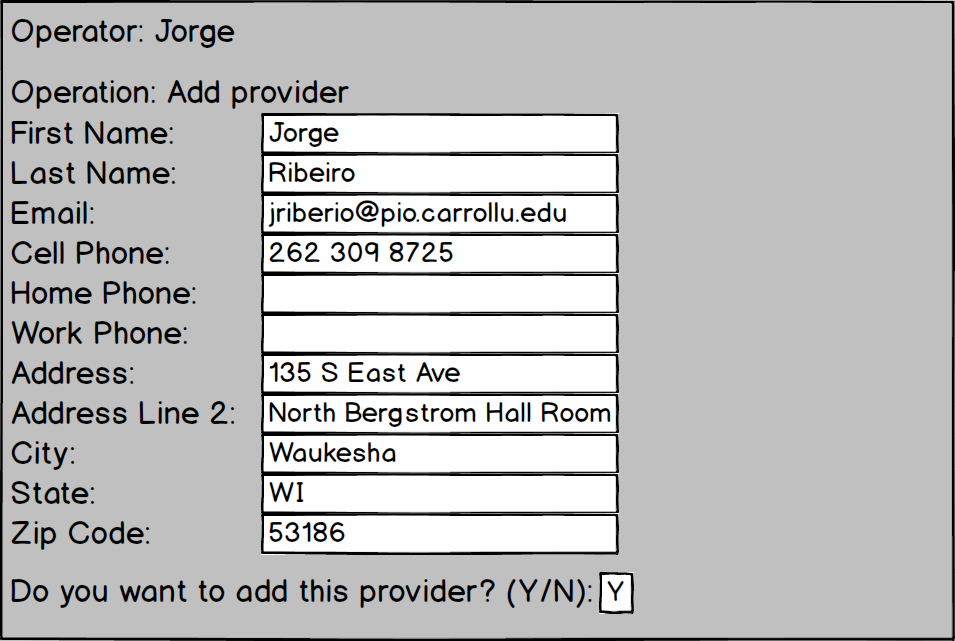
1. **Provider Operations Screen**
   1. Operator types one provider-operation number.



* 1. Test Table

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Operator wants to Add a provider. | 1 | Redirects to screen Add a provider. |  |
| Operator wants to update a provider. | 2 | Redirects to screen Update a provider. |  |
| Operator wants to delete a provider. | 3 | Redirects to screen  Delete a provider. |  |
| Operator wants to return to previous screen. | 4 | Redirects to screen  Provider main screen. |  |
| Operators types an invalid number. | 77 | Shows message: Invalid operation. |  |
| Operator inputs a not valid number. | a | Shows message:  Input a valid number. |  |
| Operator keys ‘enter’ (blank value) | “” | Shows message:  Type something valid. |  |

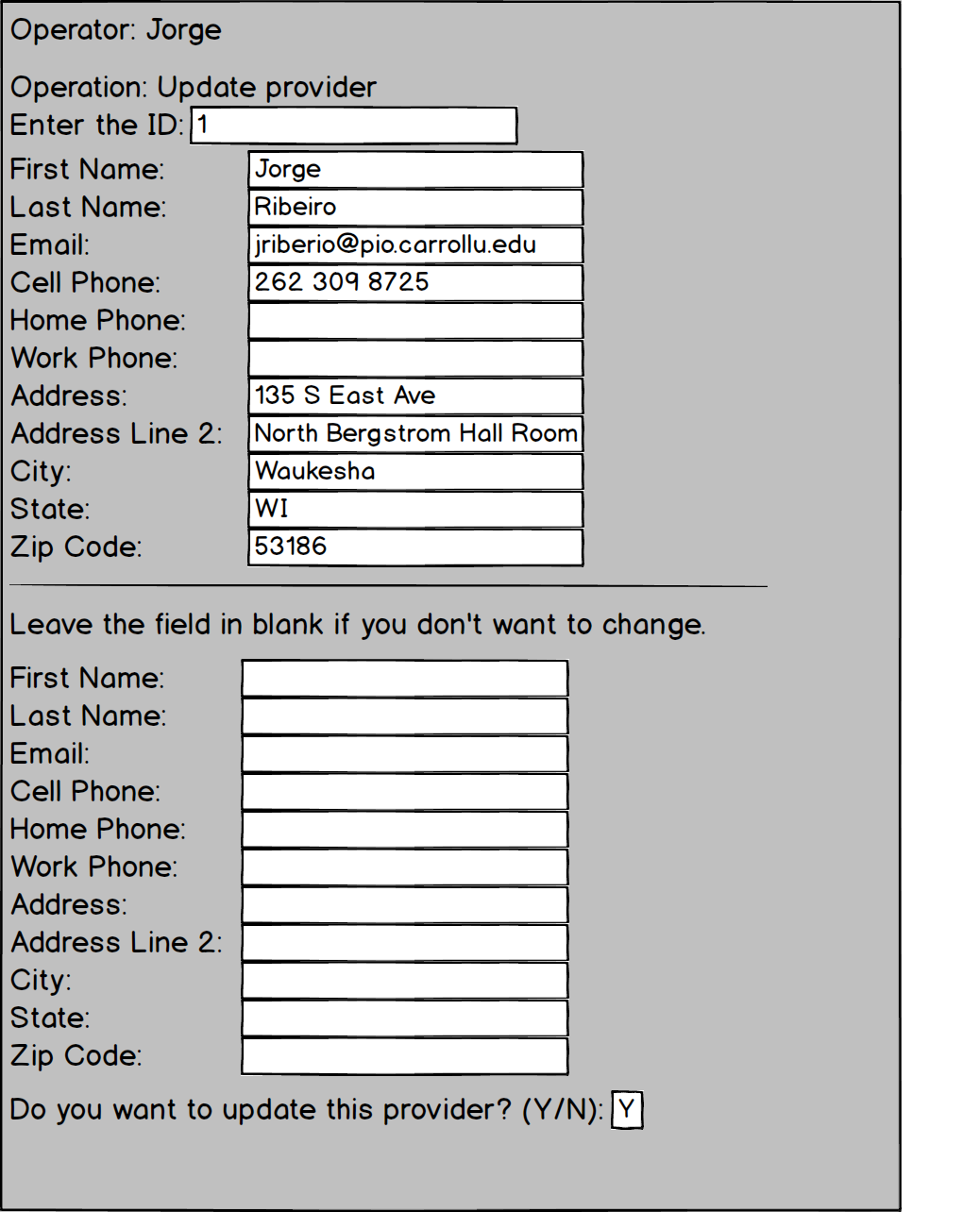
1. **Add a Provider**
   1. Operator types First Name (Required)
   2. Operator types Last Name (Required)
   3. Operator types Email (Required)
   4. Operator types cell/Home/Work (at least one Required)
   5. Operator types Address (Required)
   6. Operator types Address Line 2 (Optional)
   7. Operator types City (Required)
   8. Operator types State (Required)
   9. Operator types Zip Code (Required)



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Operator left a blank field in any of the required fields. | “” | System asks for the operator fill the field again. |  |
| All the required fields are filled properly and operatory types Y-y. | All member information and Y | Saves a new member record in the database. |  |
| Operator types an invalid Zip Code. | 12 | Requests for a new Zip Code. |  |
| Operator types a wrong state abbreviation. | ZZ | Requests for new US state abbreviation. |  |
| Operator does not want to add the member anymore. | N-n | Returns to member operation screen without saving anything in the database. |  |
| Operator types a different letter than (Y-N) in the question. | i | Shows message:  Invalid answer. Request for other letter input. |  |

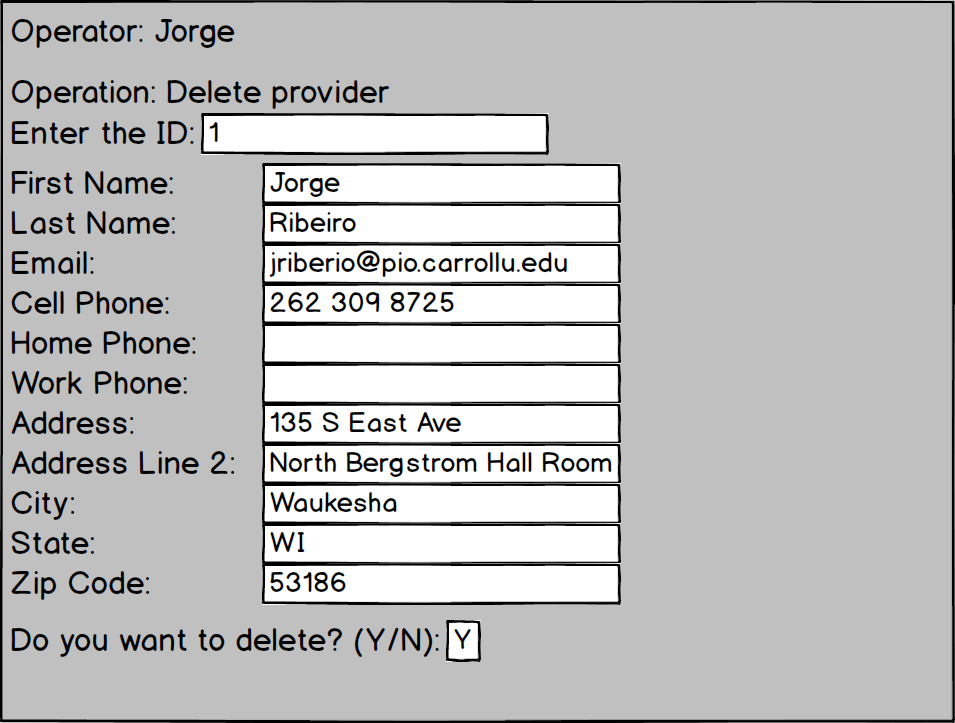
1. **Update Provider**
   1. Operator types the provider ID.
   2. System displays all provider information.
   3. Operator leave the field in blank for no changes, otherwise fill with new data.
   4. Operator types Y



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Fields are filled properly and operatory types Y-y. | All provider requested information and Y | Updates provider record in the database. |  |
| Operator types an invalid Zip Code. | 12 | Requests for a new Zip Code. |  |
| Operator types a wrong state abbreviation. | ZZ | Requests for new US state abbreviation. |  |
| Operator does not want to add the provider anymore. | N-n | Returns to provider operation screen without updating anything in the database. |  |
| Operator types a different letter than (Y-N) in the question. | i | Shows message:  Invalid answer. Request for other letter input. |  |

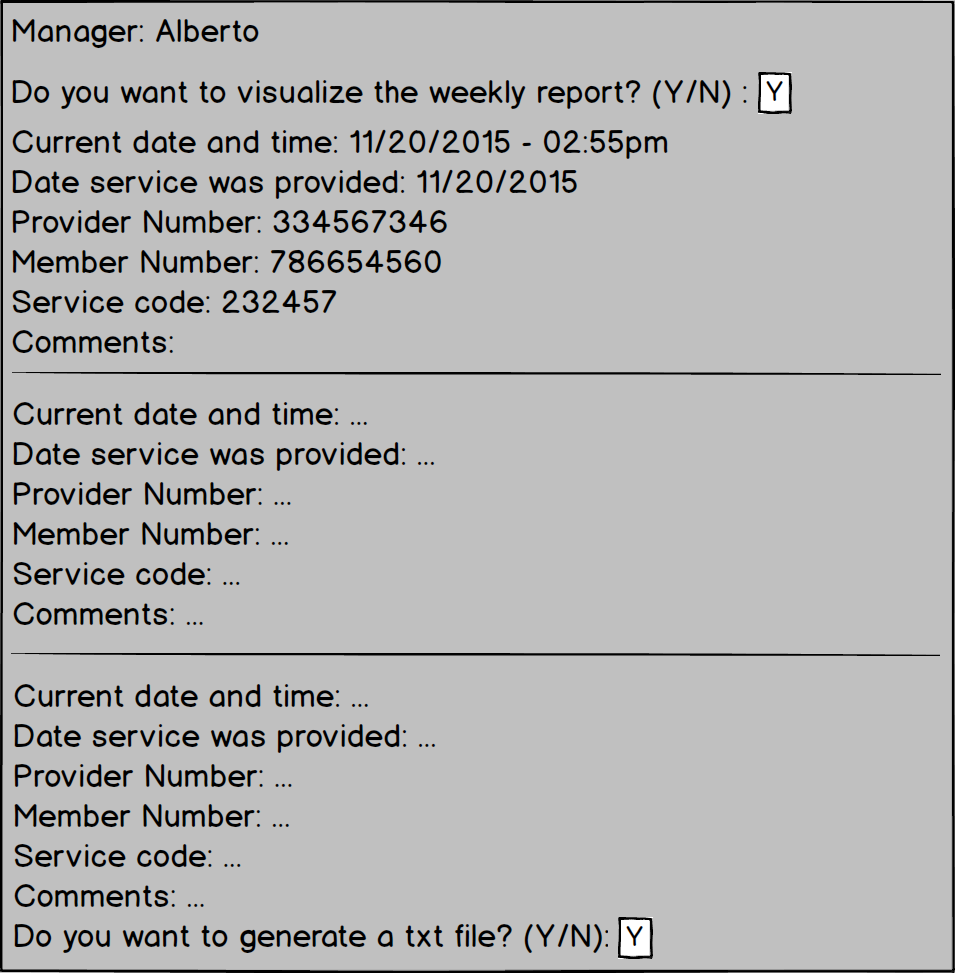
1. **Delete Provider**
   1. Operatory types the provider ID
   2. System displays all the provider information.
   3. Operators types Y and deletes the provider from the system.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Operator types invalid provider ID | 83729 | Show message: invalid provider ID. |  |
| Operator types valid provider ID | 123456 | Display all provider information |  |
| Operator types Y-y to the delete question. | Y-y | System deletes the provider record from the database |  |
| Operator types N-n to the delete question | N-n | System redirects operator to provider operation screen |  |
| Operator types invalid letter for the delete question. | 3 | System display message: invalid input and asks for the answer again. |  |

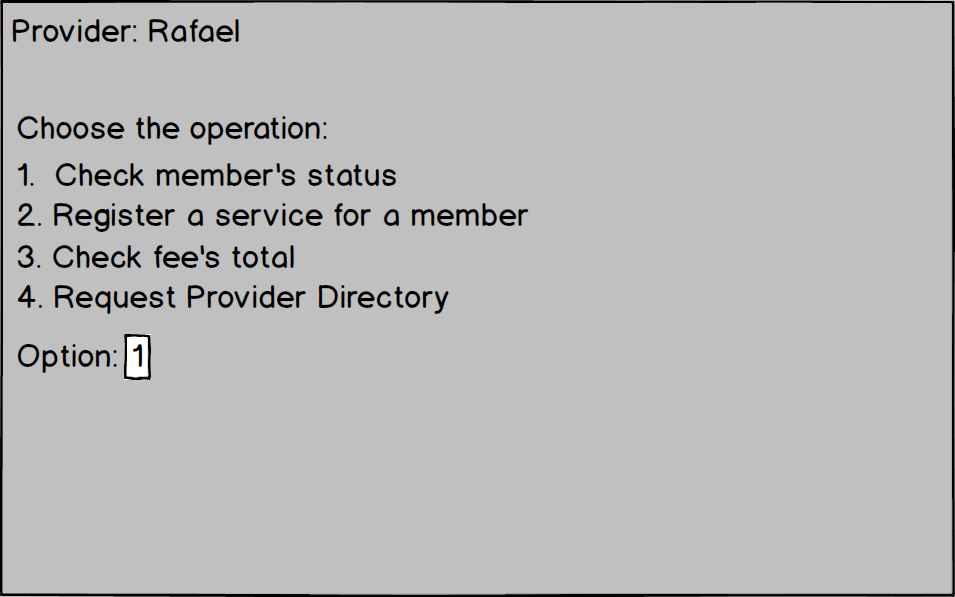
1. **Main Screen**
   1. Manager confirms that he or she wants to see the report.
   2. Manager confirms that he or she wants to generate a txt file.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Manager wants to visualize the report or generate the txt file. | Y-y | Shows the report in the first field and generates the txt file in the second field. |  |
| Manager does not want to visualize the report anymore or generate the txt file. | N-n | Returns to login screen. |  |
| Manager types a different letter than (Y-N) in the question. | i | Shows message:  Invalid answer. Request for other letter input. |  |

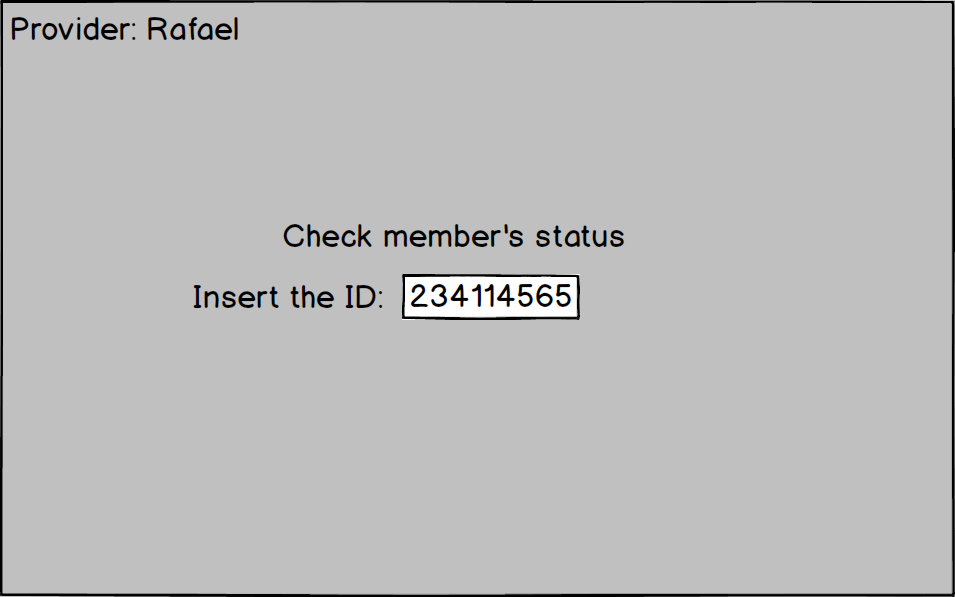
1. **Provider Main Screen**
   1. Provider types one operation.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Provider types inexistent option. | 98 | Shows message:  This operation is not valid. |  |
| Provider types blank. | “” | Shows message:  Type a valid operation. |  |
| Provider selects Check member’s status. | 1 | Redirects to screen: Check member’s status.. |  |
| Provider selects Register a service for a member. | 2 | Redirects to screen: Register a service for a member. |  |
| Provider selects Check fee’s total. | 3 | Redirects to screen: Check fee’s total. |  |
| Provider selects Request provider directory | 4 | Redirects to screen: Request provider directory. |  |

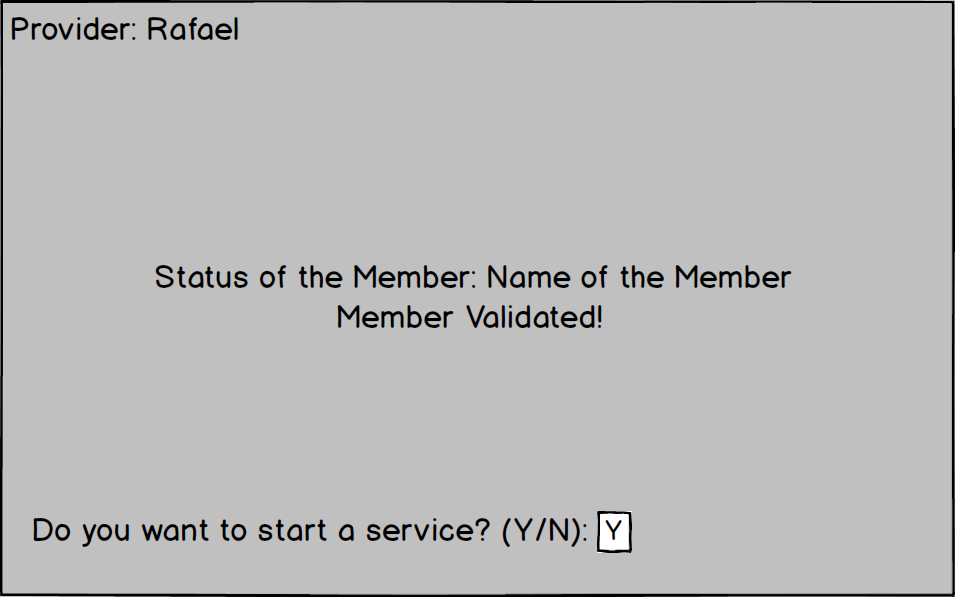
1. **Check Member’s Status**
   1. Provider types the member’s number.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Provider types an inexistent ID. | 000000000 | Shows message:  This ID is inexistent or invalid. |  |
| Provider types blank. | “” | Shows message:  Type a valid ID. |  |
| Provider types a valid ID. | 234114565 | Redirects to screen that shows the member’s status.. |  |

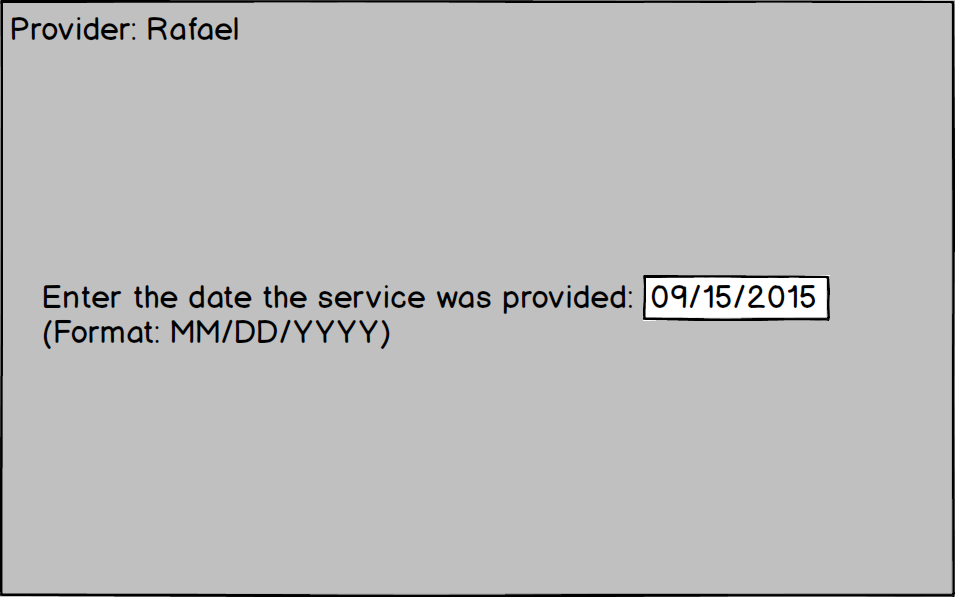
1. **Show Member’s Status**
   1. Provider confirms that he or she wants to start a service.



* 1. Test Table

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Provider wants to start a service with the member validated. | Y-y | Redirects to screen that asks for the date of service |  |
| Provider does not want to start a service with the member validated or suspended | N-n | Returns to Provider Main Screen. |  |
| Provider types a different letter than (Y-N) in the question. | i | Shows message:  Invalid input. Request for other letter input. |  |

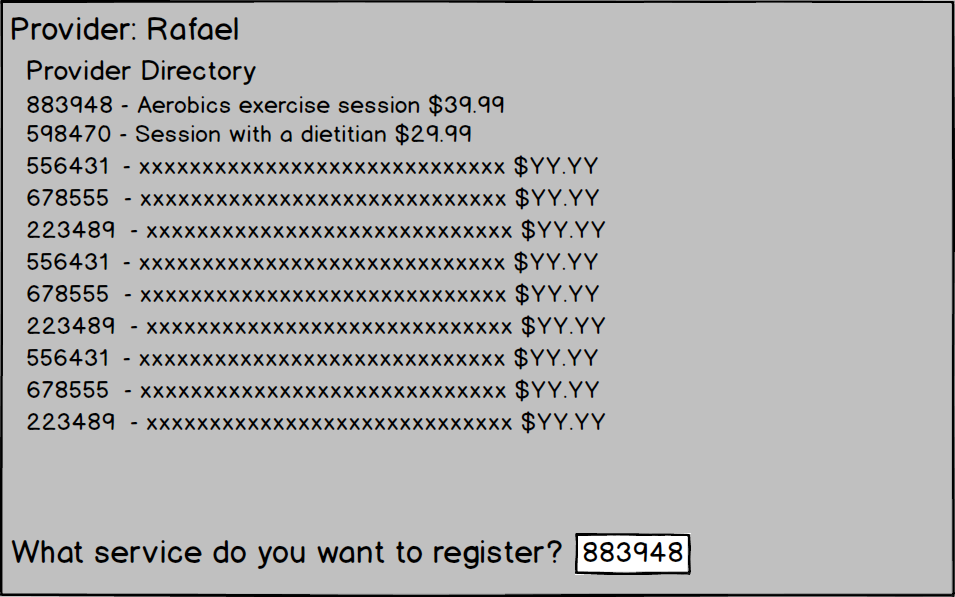
1. **Request the Date of the Service Provided**
   1. Provider inputs the date of the service.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Provider keys an invalid input. | AA-12-2312 | Shows message: Invalid input Requests date again. |  |
| Provider types blank. | “” | Shows message:  Type a valid date. |  |
| Provider types a valid date. | 09/15/2015 | Redirects to screen Provider Directory |  |

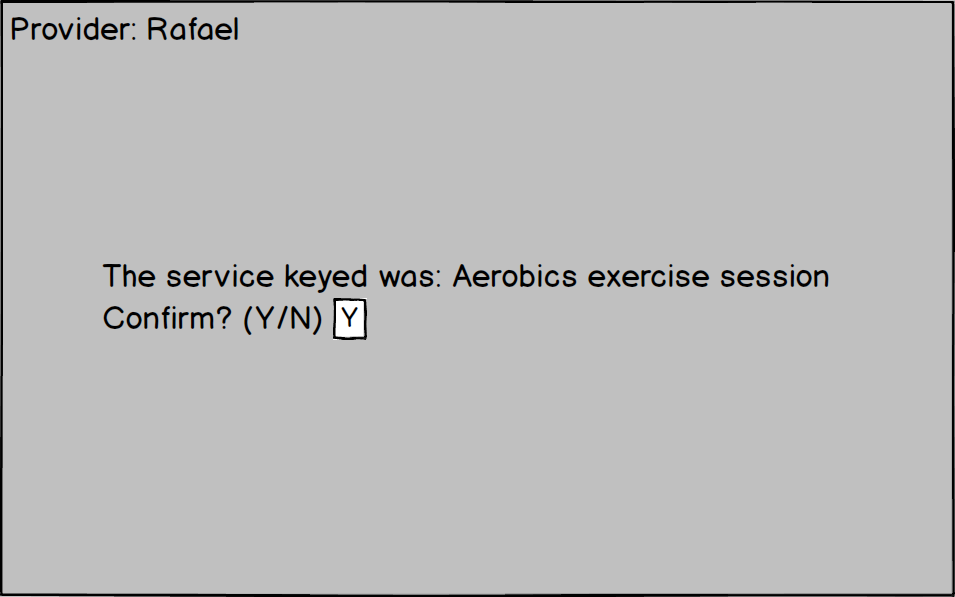
1. **Shows the Provider’s Directory**
   1. Provider inputs the service number.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Provider keys an invalid input. | BG4324K | Shows message: Invalid input. Requests service code again. |  |
| Provider types blank. | “” | Shows message:  Type a valid service code. |  |
| Provider types a valid service code. | 883948 | Redirects to screen Confirm service code |  |

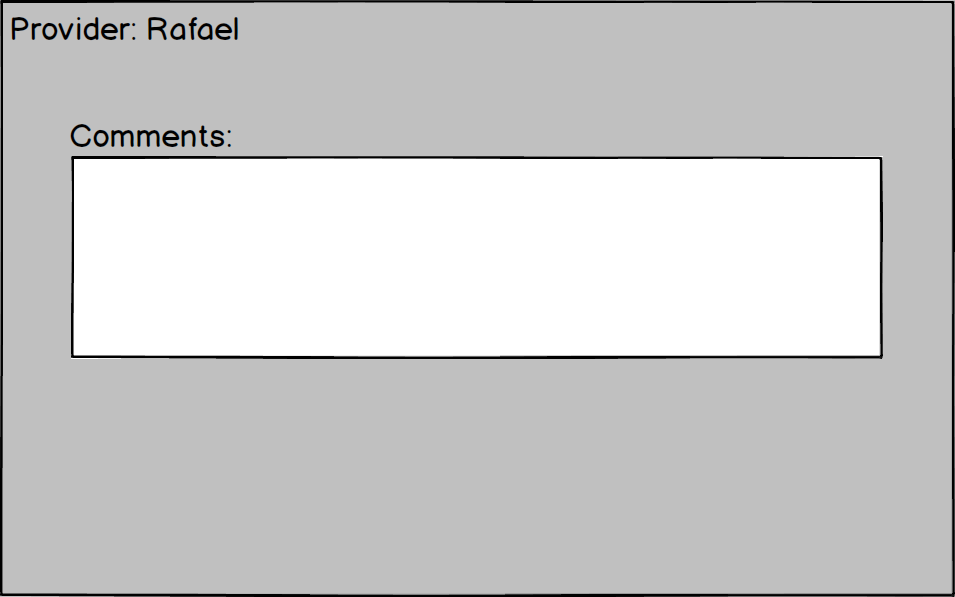
1. **Confirm Service Code**
   1. Provider confirms the service name.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Provider confirms the service name. | Y-y | Redirects to screen Comments. |  |
| Provider does not confirm the service name | N-n | Returns to Provider Directory. |  |
| Provider types a different letter than (Y-N) in the question. | i | Shows message:  Invalid input. Request for other letter input. |  |

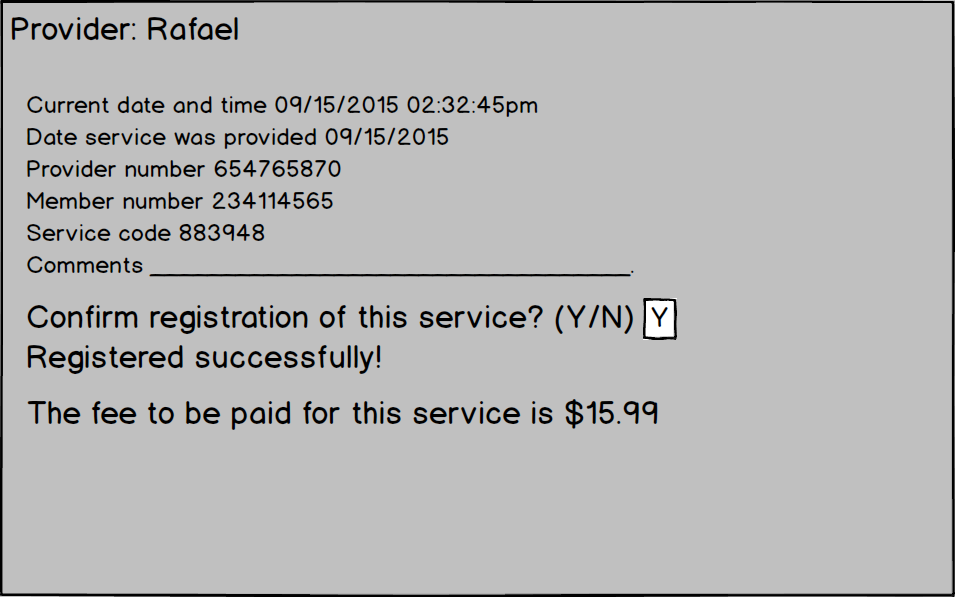
1. **Comments**
   1. Provider adds comments to the service.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Provider types blank. | “” | Provider does not want to add comments. |  |
| Provider types some information. | “Service provided in less than 30 minutes. The member requested a second session for improve the results.” | Provider adds a comment to the service and is redirected to the screen Description of Service. |  |

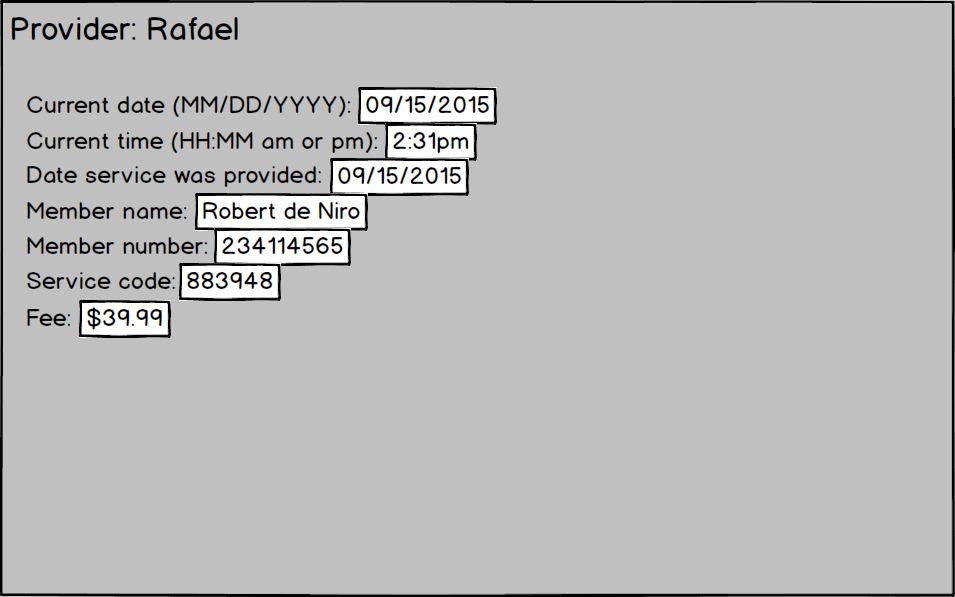
1. **Description of Service**
   1. Provider confirms the registration of the service.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Provider confirms the service name. | Y-y | Redirects to screen Verification Form. |  |
| Provider does not confirm the service name | N-n | Returns to Provider Directory. |  |
| Provider types a different letter than (Y-N) in the question. | i | Shows message:  Invalid input. Request for other letter input. |  |

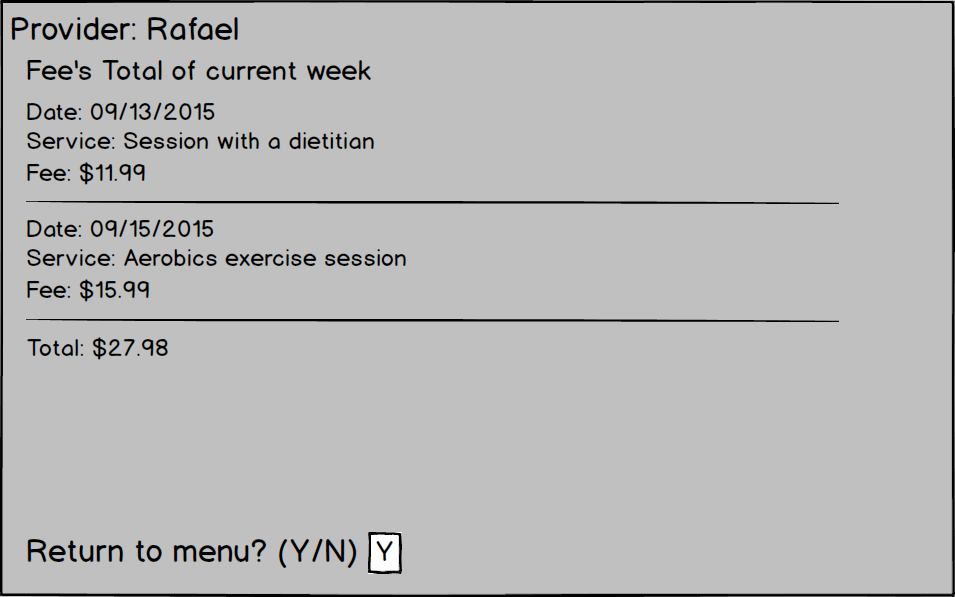
1. **Verification Form**
   1. Provider fills a verification form of the service provided.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Provider left a blank field in any of the required fields. | “” | System asks for the provider fill the field again. |  |
| Provider types an invalid date. | 12/DD/2018 | Requests for a new date. |  |
| Provider types an invalid time. | 2:ABpm | Requests for a new time. |  |
| Provider fills all the fields. | Fields filled properly | Service completed. Returns to provider main screen. |  |

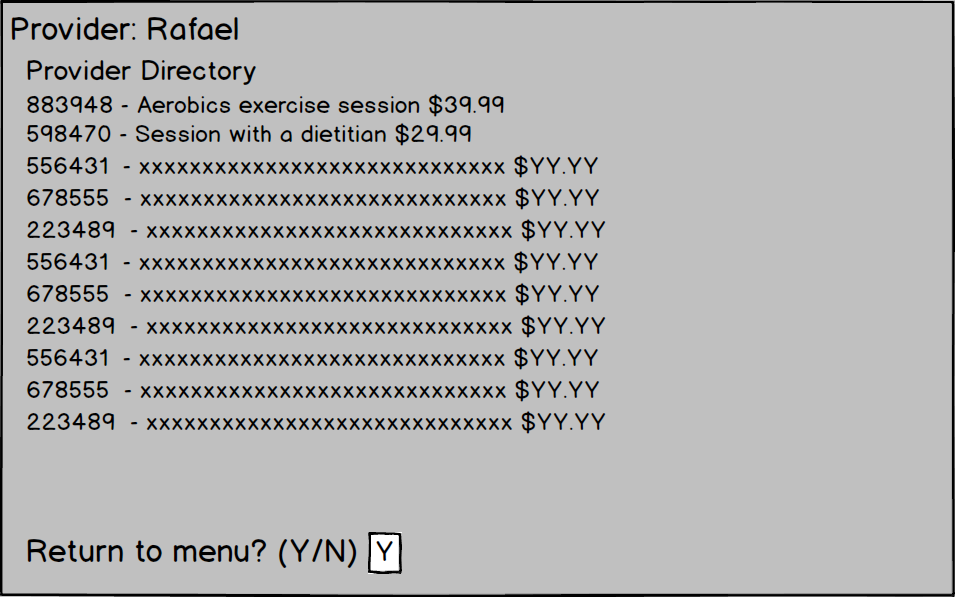
1. **Check Fee’s Total**
   1. Provider confirms the returning to menu.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Provider confirms the service name. | Y-y | Redirects to screen Verification Form. |  |
| Provider does not confirm the service name | N-n | Returns to Provider Directory. |  |
| Provider types a different letter than (Y-N) in the question. | i | Shows message:  Invalid input. Request for other letter input. |  |

1. **Request Provider Directory**
   1. Provider confirms the returning to menu.



* 1. Test Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Input data | Expected results | Passed: Y || N? |
| Provider confirms the service name. | Y-y | Redirects to screen Verification Form. |  |
| Provider does not confirm the service name | N-n | Returns to Provider Directory. |  |
| Provider types a different letter than (Y-N) in the question. | i | Shows message:  Invalid input. Request for other letter input. |  |