CareAway Treatment Planner  
Design Document

**Date**: 12/7/2017

**Team Name**: Team Tangent

Eugene Mesina (014455816) Meeting Manager (Team Leader)

Felix Huang (013427386) Documentation Manager

Jimmy Chao (012677182) Accountability Manager

Taylor Tobin (011871260) Technology/Information Manager

Crystal Chun (012680952) Quality Manager

Tyler Dao (013343302) HIPAA Manager

**Table of Contents**

# 

[**Introduction**](#_7k9gpjf1wpxa) **3**

[Abstract](#_32ocmt29l9l8) 3

[Document Overview](#_eebi1egcxx2z) 3

[Scope](#_846w8g1kvh1) 3

[Architecture](#_tbn5s6od165q) 5

[Team Philosophy](#_r4atl2s1194) 7

[**Abstraction**](#_n2gtkkf78gde) **8**

[**High Level Design**](#_nidr60x4qfw9) **10**

[Swimlane Diagrams](#_r52t4ng8weme) 10

[Registration](#_pu4soaubr3kk) 10

[Authentication](#_4hwth1zh2y25) 12

[Reset Credentials](#_5htc7lguhq9k) 13

[Session Termination](#_maijv0ybwag8) 14

[Data Access](#_yca2qvmjrcc6) 15

[Error Handling](#_pzdv5m7bgm5b) 17

[User Management](#_kdtr05k2ug07) 18

[User Access Control](#_8ex8jy87cugw) 19

[Network Connectivity](#_2f82urujiwwk) 20

[Security](#_e5pcohhyalc2) 21

[Treatment Plan Creation and Revision](#_s9t7wk6tvpt3) 23

[Treatment Plan Interaction](#_5zuh4s7eew7w) 25

[Request Appointment](#_d5yk62jpkhip) 27

[Data Analysis Reports](#_9ap0415rx85c) 29

[Site Map](#_66tfrl3hfi4u) 30

[Network Diagram](#_xa6nxhbps1xd) 31

[**Low Level Design**](#_nu6v8awto4pc) **32**

[Responsibility Matrix](#_6pfak329r46v) 32

[Entity Relation Diagram](#_pcffwo8vfrux) 33

[Sequence Diagrams](#_9wvoz1e6f4hm) 34

[Registration](#_p8ygsvx7qw0n) 34

[Authentication](#_eli1jfctf75e) 35

[Resetting Credentials](#_7cwwnlkgnrbw) 36

[Session Termination](#_sq3iuqblbs89) 37

[Data Access](#_efnlg9a636fr) 39

[Error Handling](#_r6518rgj7vqy) 40

[User Management](#_7fqfju442hfu) 42

[User Access Control](#_kn10g1qs3f3g) 45

[Network Connectivity](#_g0pkwl659irm) 46

[Security](#_oncx988kr7ff) 47

[Breach Messages](#_l7qpkjgg3v7g) 48

[Treatment Plan Creation](#_vx9poz89ofbl) 49

[Treatment Plan Revision](#_77vh52ro0n5u) 51

[Treatment Plan Interaction](#_m36q9pdsbsuw) 53

[Request Appointment](#_xgwmjyjh76y4) 54

[Data Analysis Reports](#_qtouc2965xfm) 56

[Wire Frame](#_sgrtahmkk5wm) 57

## 

# 

# 

# 

# 

# **Introduction**

## Abstract

CareAway Treatment Planner is a single page application that addresses communication issues between the patient and medical professional outside of the medical environment. As a solution that exists in the healthcare domain, it is important for CareAway to adhere to Health Insurance Portability and Accountability Act (HIPAA) standards in terms of privacy and data security. The target population for the CareAway system is patients who have a diagnosis and the medical professionals who will be overseeing their care. One goal of this system is to ensure convenience and accessibility for the user. The system can be accessed from anywhere and at anytime, as long as the user has valid login credentials. The difference between CareAway and other existing solutions is that the former will offer better means for communication between patients and medical professionals through interactive treatment plans.

## Document Overview

The purpose of this document is to provide both a high and low-level description of the CareAway Treatment Planner. The Design Document will provide information in order to give an understanding of what components are needed to be built and how it will be built. Throughout the document, all business requirements for the CareAway Treatment planner will be addressed through high and low-level designs. In addition, the document will explain the architecture our team will be utilizing when we construct the components, while going in depth about the coding standard our team will be adhering to.

## Scope

This Design Document will be focusing on high and low level design for the CareAway Treatment Planner. The CareAway Treatment Planner will be a single page application, and we intend to design the system to be used on the following desktop web browsers for the minimum viable product as defined in the Business Requirements Document:

Google Chrome - Version: 62.0.3202.94 and newer

Mozilla Firefox - Version: 57.0 and newer

Apple Safari - Version 11.0.1 and newer

The Design Document will hold models for the following CORE and Content Business Requirements:

**CORE Requirements**

* + Registration
  + Security
  + Authentication / Session Termination
  + User Access Control
  + Data Access
  + Error Handling
  + User Management
  + Network Communication

**Content Requirements**

* + Treatment Plan Creation and Revision
  + Interacting with Treatment Plan
  + Appointment Scheduling
  + Data Analysis Reports

The document will contain the following High Level Designs:

* **Swim Lane Diagrams** - This modeling technique will display a breakdown of each step of a business requirement and show the actor responsibility at each point in time. This diagram will provide a high level explanation of the need of the component and the role it fulfills in the system.
* **Sitemap** - This modeling technique will display the layout of the CareAway Treatment Planner application and the pages it will lead to. This will provide a visual graphic of how the system will be presented after deployment.
* **Network Diagram** - The modeling technique will display what networks the system interacts with, what ports we will accessing the Internet from, and the firewall interaction. We will illustrate the interactions using picture representation of the connection between the client and the system.

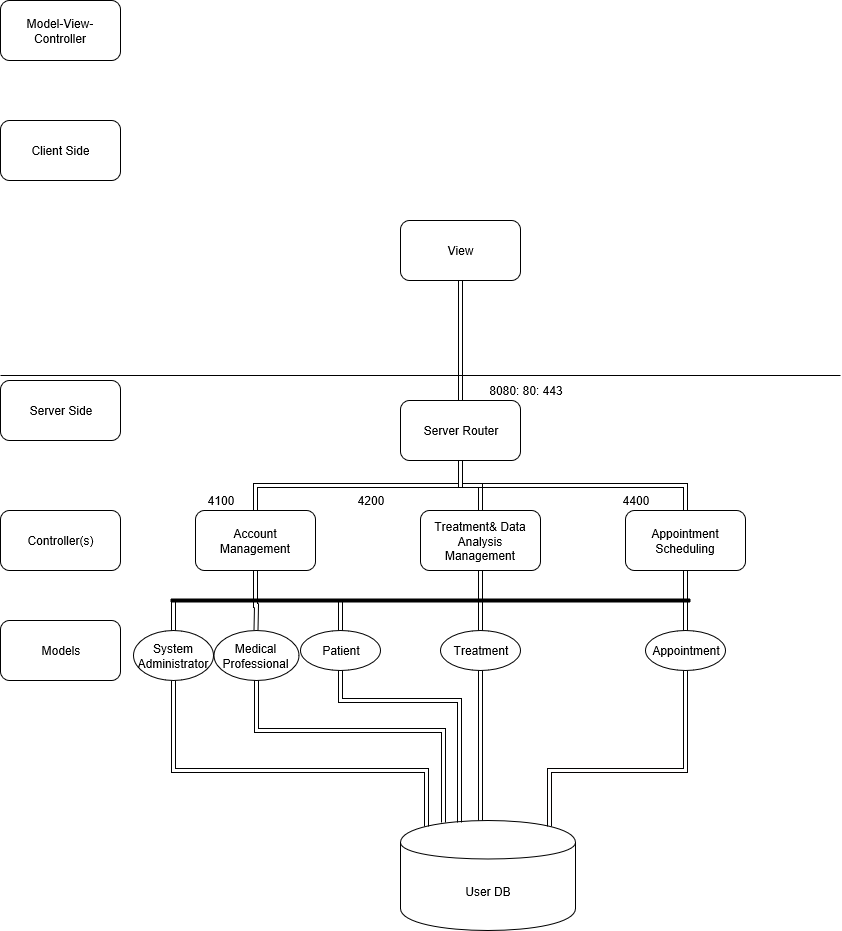
The document will contain the following Low Level Designs

* **Responsibility Matrix**- This modeling technique will be used to display what data access each actor has, and what functionality each user can do with in the system. This will provide a quick overview of the functionalities and the users that will use them throughout the system.
* **Entity Relationship Diagram**- This modeling technique will be used to model all entities within the context of the system, and the relationship and attributes associated with each entity. This will provide a visual representation of the different attributes that are associated with each entity along with the relationship they share throughout the system.
* **Sequence Diagrams** - This modeling technique will display the behavior of the system and how the system will move from one state to another state. This diagram will provide the low level functionality for each service the system will provide.
* **Wire Frame** - This modeling technique will display the low level graphical design of the web application. It will show the actual design of the webpage and the interconnections of each webpage for the application.

## **Architecture**

The team has decided to design the CareAway Treatment Planner using the Model View Controller (MVC) as the overall architecture. For MVC, although it adds complexity to the CareAway Treatment Planner, the separation of business models from the user interface will allow the team to work on different components simultaneously. In addition, due to the separation of concerns in MVC, various changes can be made in different components without affecting other models within the application. Although MVC creates inefficiency with data access for the view, it provides asynchronous calls and lowers the amount of code duplication within the application.

In the MVC architecture, the view is the representation of the web page and its content functionalities. It serves the purpose of presenting the data in a consumable form to the user. The view connects to the server router and the hosted web server through the internet. This web server contains the server router module which connects to 3 controllers. The server router handles the events happening from the client and then proxies the request made from those events to their own respective controller. Each controller has access to each model allowing for fluid data aggregation from each model, but are separated in duty for different user tasks. Most of this aggregation will come from the account module, then from the account’s different associated attributes. Lastly, all data is aggregated from one user database and processed through the controller to the view.



## 

## 

## **Team Philosophy**

The team has agreed upon creating clear readable code by using a concise and consistent coding standard. The coding convention the team will be following is the Mozilla Coding Style Guide that covers the the following:

* **Indentation**- provide 2 space indentation for each logic level
* **Comments**- JSDoc to annotate JS files, comments one line above the code line it’s referring to
* **Declaration**- uses verbs for functions, nouns for variables, camel-casing and lowercase first letter of the name
* **White Space**- No tabs or Spaces at the end of each line, use Unix Style line-breaks (‘\n’)
* **Line Length**- no more than 80 characters per line

The team has also agreed upon using the Vue Framework and abiding by its style guide. This coding standard will be following Vue’s own essential standards that include:

* Component names should always be multi-word, except for root App components.
* Component data must be a function.
* Prop definitions should be as detailed as possible.
* Always use key with v-for.
* For applications, styles in a top-level App component and in layout components may be global, but all other components should always be scoped.
* Always use the $\_ prefix for custom private properties in a plugin, mixin, etc. Then to avoid conflicts with code by other authors, also include a named scope.

The team will provide code following these specific guidelines in order to provide readable source code and in turn make future software maintenance and upgrades easier to incorporate.

Reference:

<https://vuejs.org/v2/style-guide/>

<https://developer.mozilla.org/en-US/docs/Mozilla/Developer_guide/Coding_Style>

<https://en.wikipedia.org/wiki/JSDoc>

# **Abstraction**



The CORE volatility analysis describes the system requirements needed to start running. This begins with it first running as a Single Page Application. The system will then need security to protect the web application from future and current threats due to its sensitive state. After the system has established this secure connection, then it will provide the user with data access and error handling which takes into account the business logic and data validity that the system needs.

The content of the volatility analysis diagram was created based on the value identification, actors, and functionalities of our system.

|  |
| --- |
| **Aspects** |
| Security |
| Data access |

|  |
| --- |
| Error Handling |

|  |  |  |  |
| --- | --- | --- | --- |
| UI Layer | Validation Layer | Business Layer | DataStore Layer |
| Webpage | Account Validation | Roles | Capture Typing |
| Views | Input Validation | Data Types | Capture Clicking |
| Widgets | Action Validation | Data Aggregation |  |

This abstraction provides a higher level of understanding of the system. The abstraction begins with a volatility analysis showing which aspects, requirements involved with multiple layers and parts of the system, are most important. The aspects are derived from the CORE functionalities of the system. These are prevalent throughout the system and are not limited to a single layer.

In order to derive the abstraction layer,. a volatility analysis from the Content functionalities was created to provide insight to determine the layers of the system. These four layers show the flow of the web application. The first layer is the user interface layer where the user is able to see the web page, views, and widgets of the system. The user interface layer will have inputs that will need to be captured that include clicking and typing. Next, these inputs will go through the validation layer, where inputs are checked against the business requirements that system enforces. After that, the system will then check where to store these inputs based off the role, type, and aggregation of data. This layer is the business layer and are specific to this system’s actors. Finally, the inputs are then stored into the datastore layer, this includes the database and location in which those inputs are being stored.

# 

# 

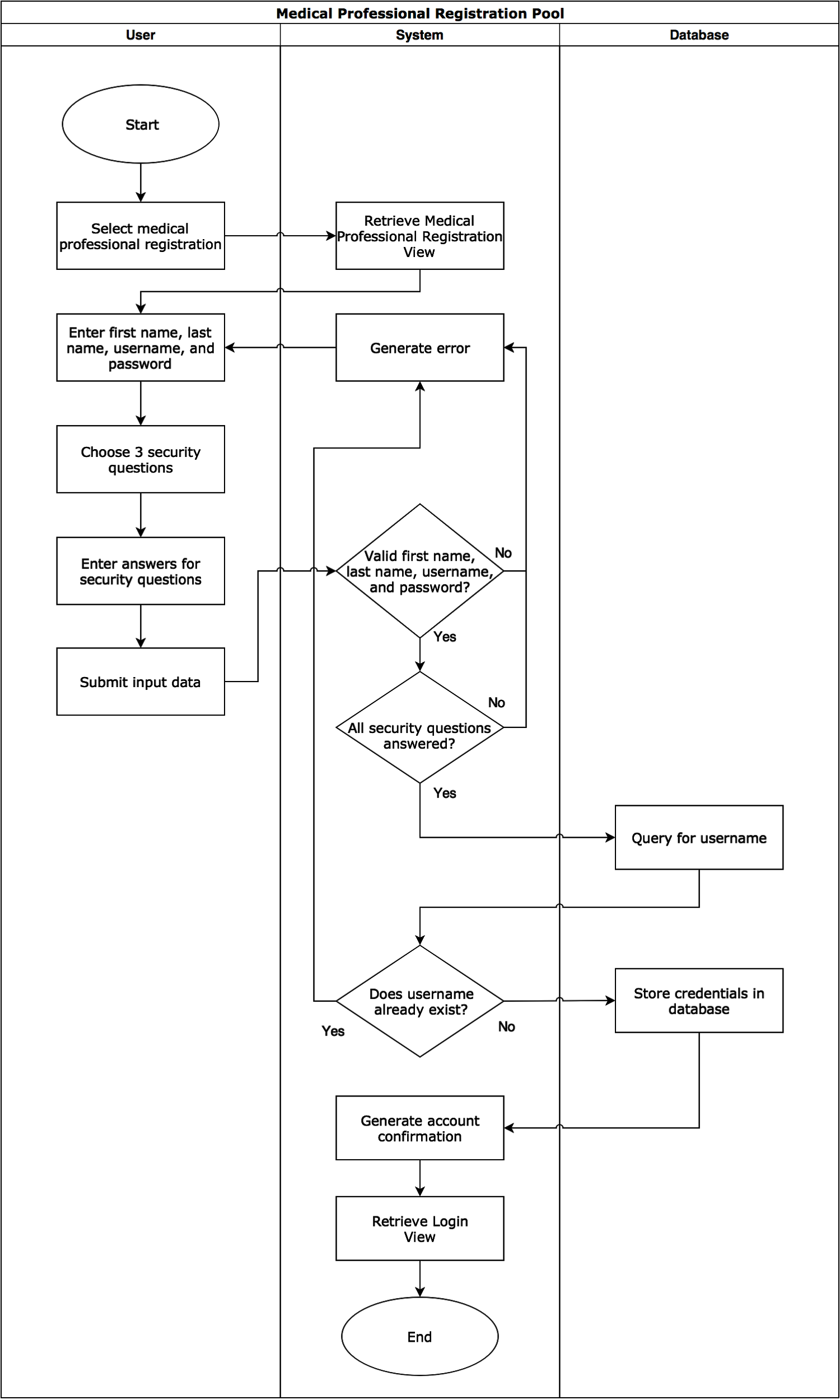
# **High Level Design**

## **Swimlane Diagrams**

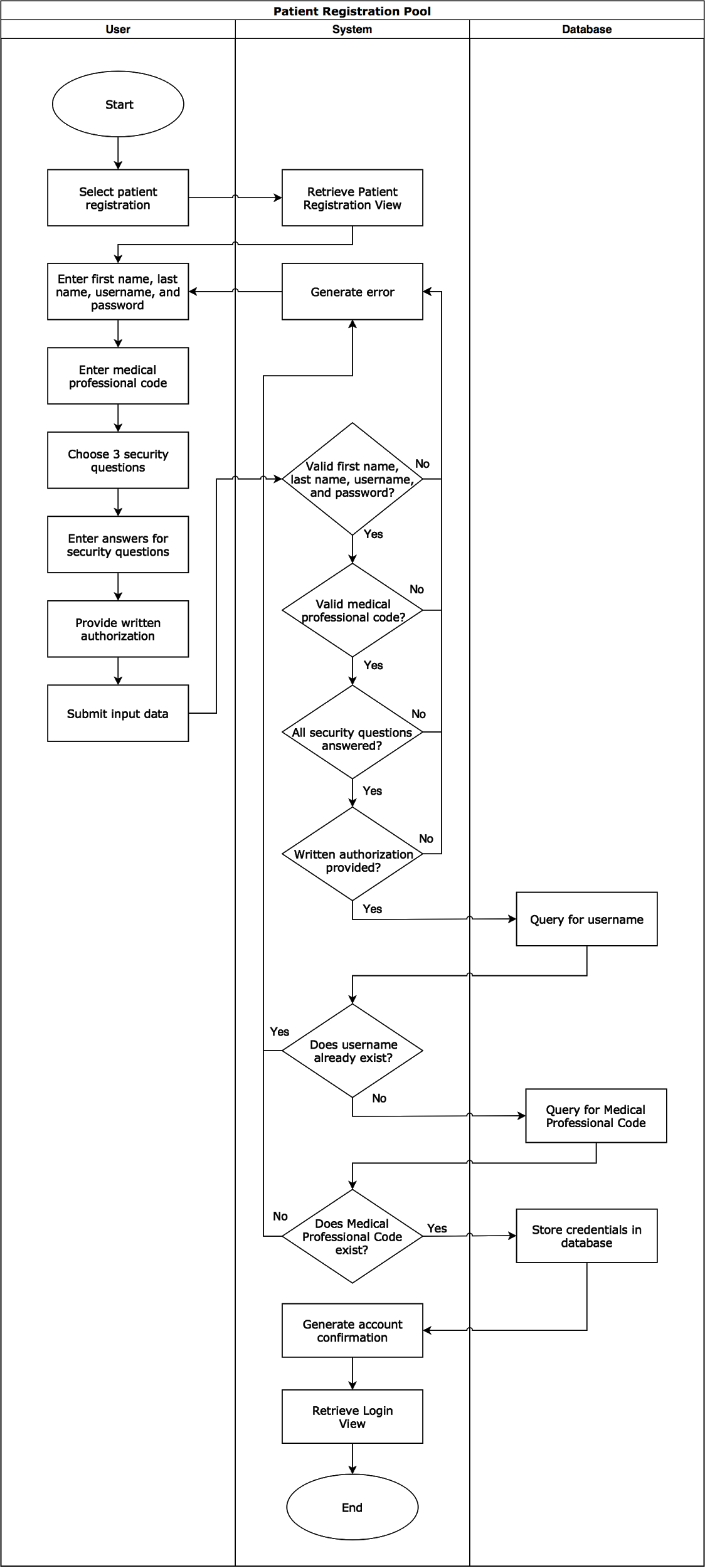
### Registration

The following diagrams show the actions that occur from the user, the system, and the database during a medical professional registration or a patient registration. The precondition for both of these diagrams is that the current user registering is not authenticated into the system, and they start from CareAway Treatment Planner’s Homepage.

*Medical professional registration*

[**](https://www.draw.io/?scale=2#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

*Patient registration*

[**](https://www.draw.io/?page=1&scale=2#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

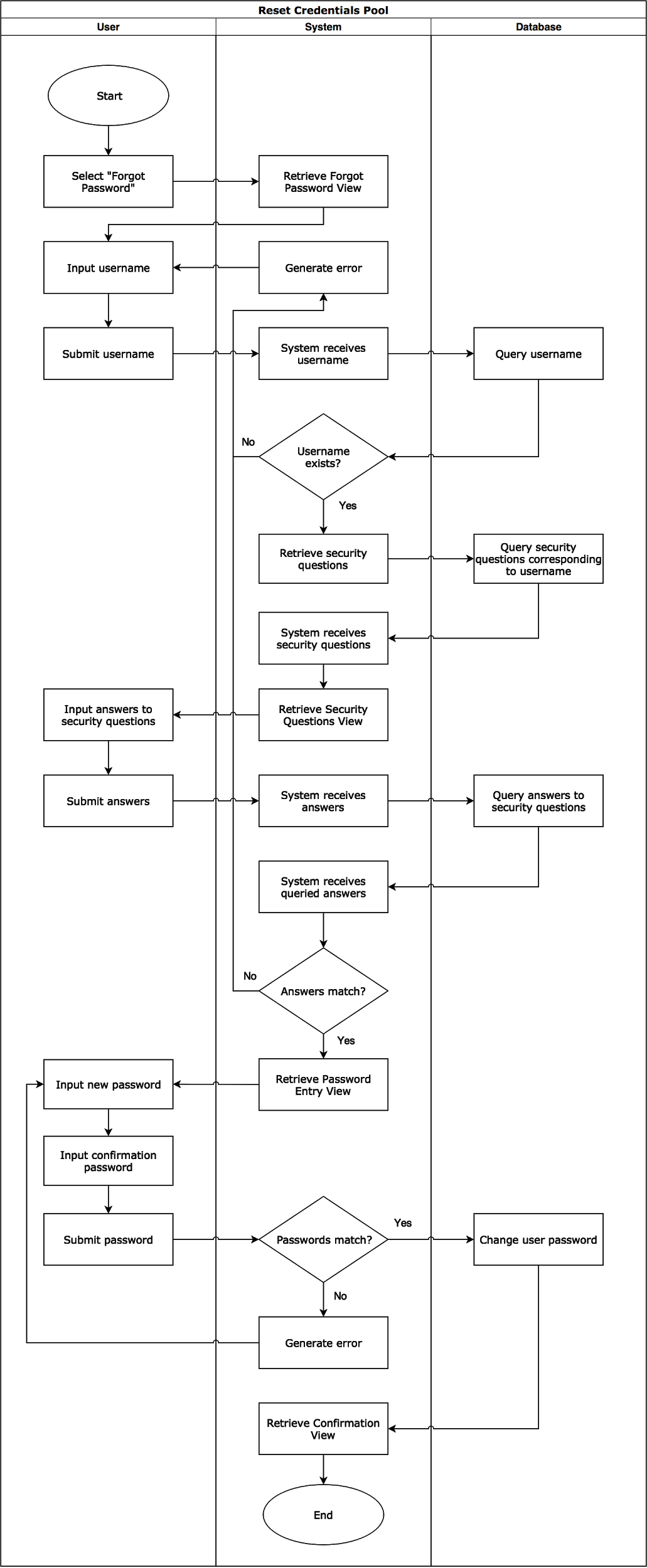
### Authentication

The following diagram portrays the actions taken by the user, the system, and the database when the authentication process is occuring. The precondition for this diagram is that the user who is logging in has not been authenticated into the system yet, and the user starts from CareAway Treatment Planner’s Homepage.

[](https://www.draw.io/?page=2#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

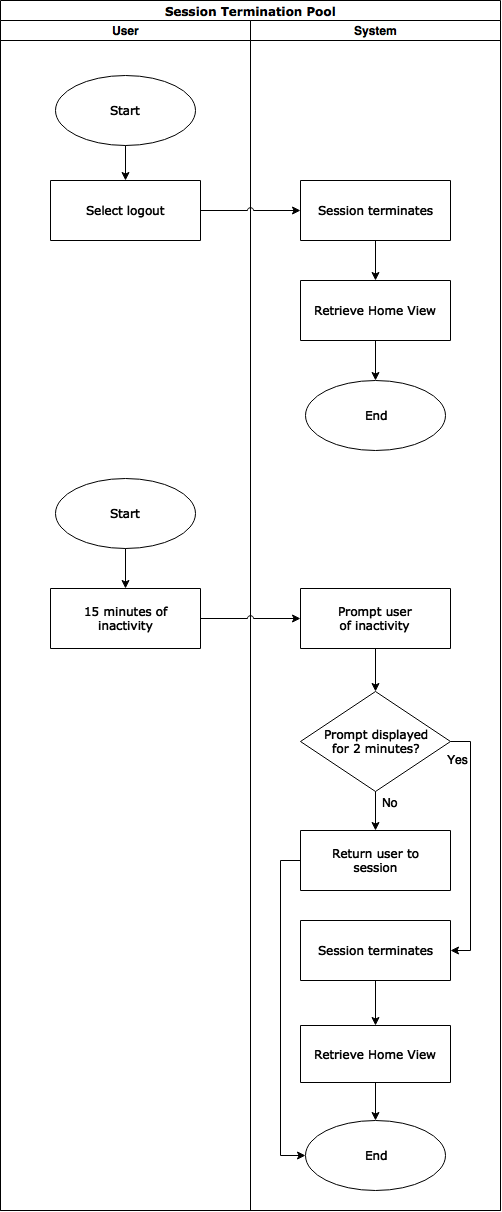
### Reset Credentials

The diagram below details the way the user, the system, and the database act when a user is resetting their credentials. The precondition for this diagram is that the user is not yet authenticated into the application and that the user starts off on the Login View.

[](https://www.draw.io/?page=3&scale=2#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

### Session Termination

The diagrams below show the way the user and the system interact during the session termination scenario. Both diagrams’ precondition is the user is authenticated in the application. The first diagram (first start) indicates the user wanting to terminate their own session. The second diagram (second start) indicates the user being logged out by the system due to inactivity.

[](https://www.draw.io/?page=4#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

### Data Access

The following diagrams show the access and updating of data by the user. The user, system, and database act accordingly when a user wants to access or update (edit) their data.

For both diagrams:

The following data is accessible by the Medical Professional after they register:

* Password (Edit access)
* Medical Professional First Name (Read-only)
* Medical Professional Last Name (Read-only)
* Their Patients’ First Name (Read-only)
* Their Patients’ Last Name (Read-only)
* Their Patients’ Treatment Data (Read and edit access)
* Their Patients’ Medical Condition (Read-only)
* Security Questions (Read-only)
* Appointments (Read and edit access)

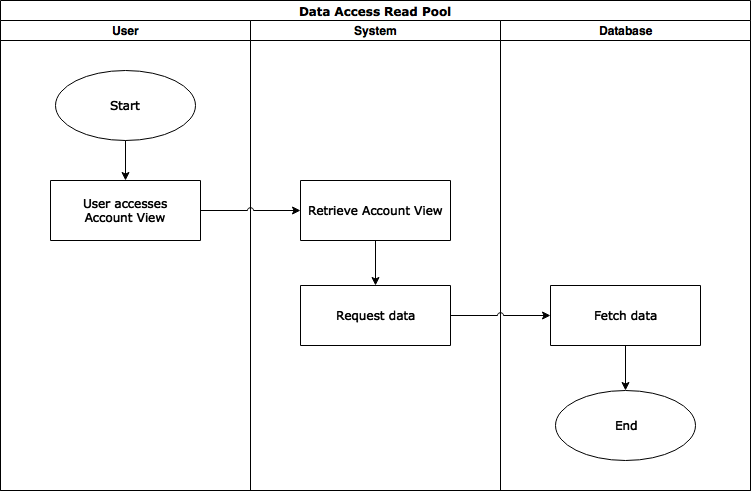
The following data is accessible by the patient after they register:

* Password (Edit access)
* First Name (Read-only)
* Last Name (Read-only)
* Their Medical Condition (Read-only)
* Their Treatment Plan (Read-only)
* Security Questions (Read-only)
* Appointments (Read and edit access)

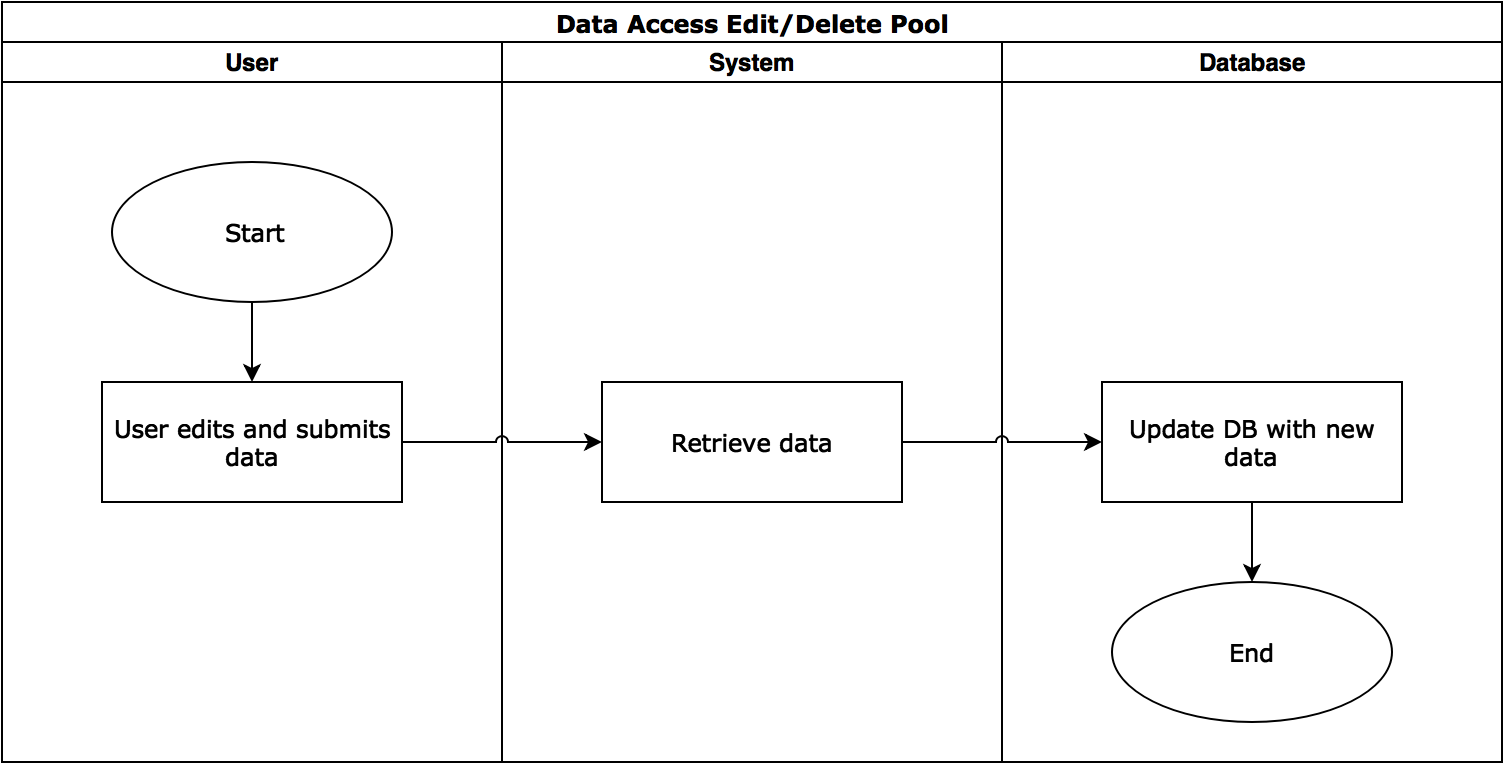
The precondition for the first diagram is that the user is verified to have access to that account view they requested.

The precondition for the second diagram is the user is authenticated into the system, and they have access to data that they are then allowed to edit.

*Reading data*

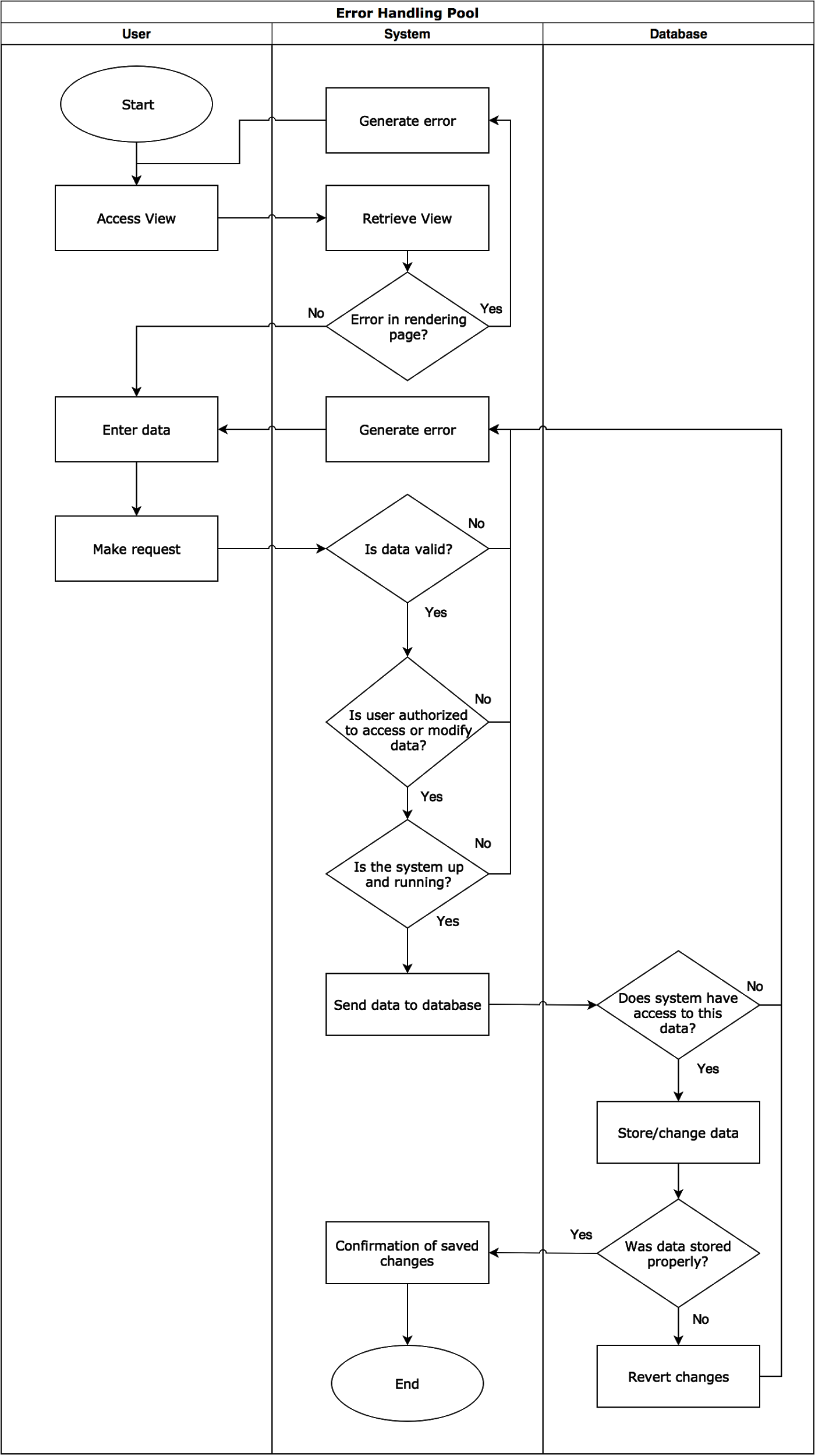
[](https://www.draw.io/?page=5#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

*Editing Data*

[**](https://www.draw.io/?page=6&scale=2#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

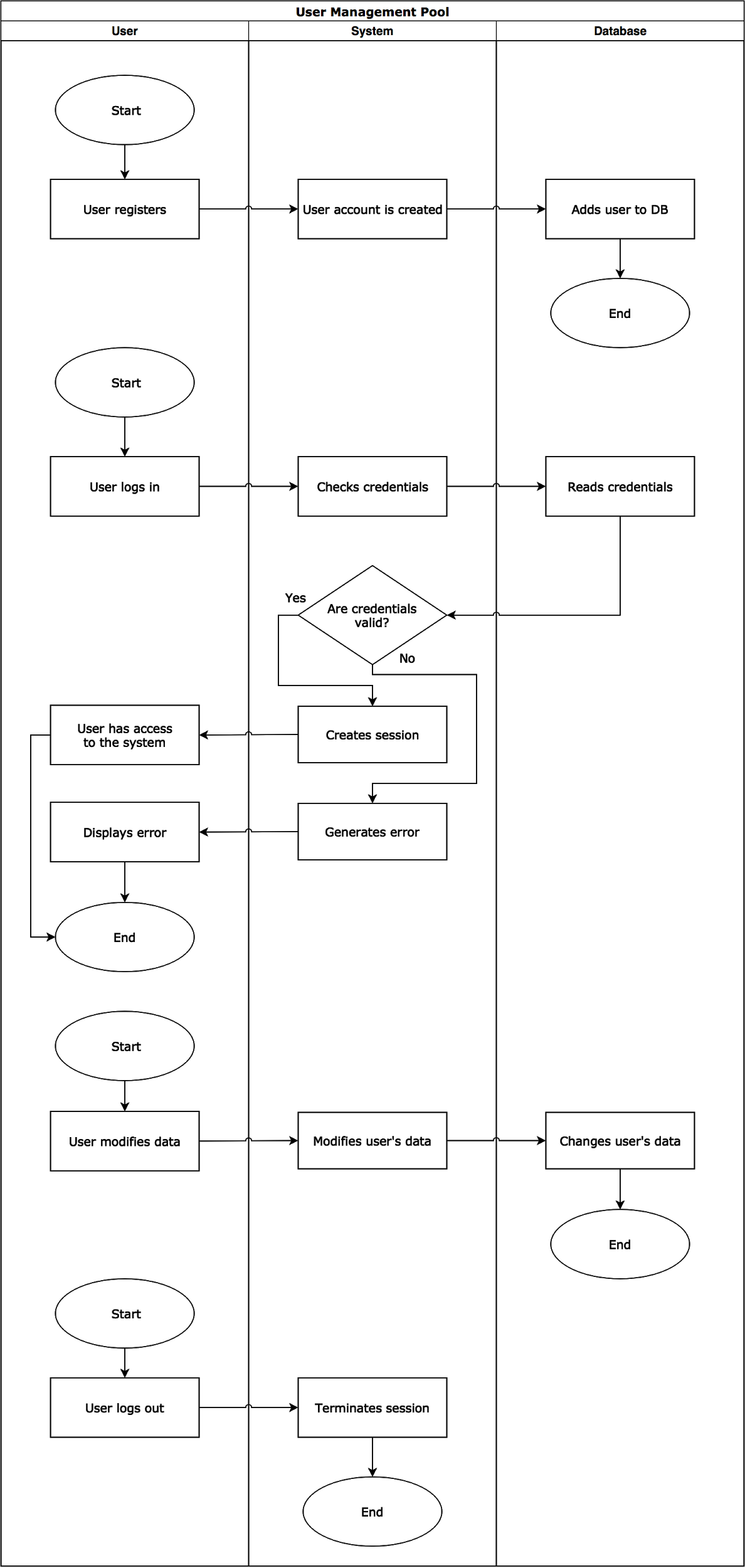
### Error Handling

The following diagram portrays the actions from the user, the system, and the database when an error occurs. The precondition is that the system is deployed. The first error checks if the specified user has access to a certain view. The next error checks for data validity, which includes if the data entered was incorrect, if the data field was left blank when it was required, or if the data was not allowed to be changed.

[](https://www.draw.io/?page=7&scale=2#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

### User Management

User management deals with creating, reading, and updating users. The diagrams below depict the actions taken by the user, the system, and the database when user management occurs. The precondition for the first diagram is that the user is not authenticated onto the system and that they are a new user (they have not registered onto our system before). The first diagram represents the creation of a user. The precondition for the second diagram is that the is not authenticated into the system. The second diagram represents reading a user. The precondition for the third diagram is that the user is allowed to access and edit the data they are sending to the system. The third diagram represents updating users. The precondition for the fourth (last) diagram is that the user is authenticated into the system. The third diagram shows the system being able to terminate user sessions, changing the state of a user.



### 

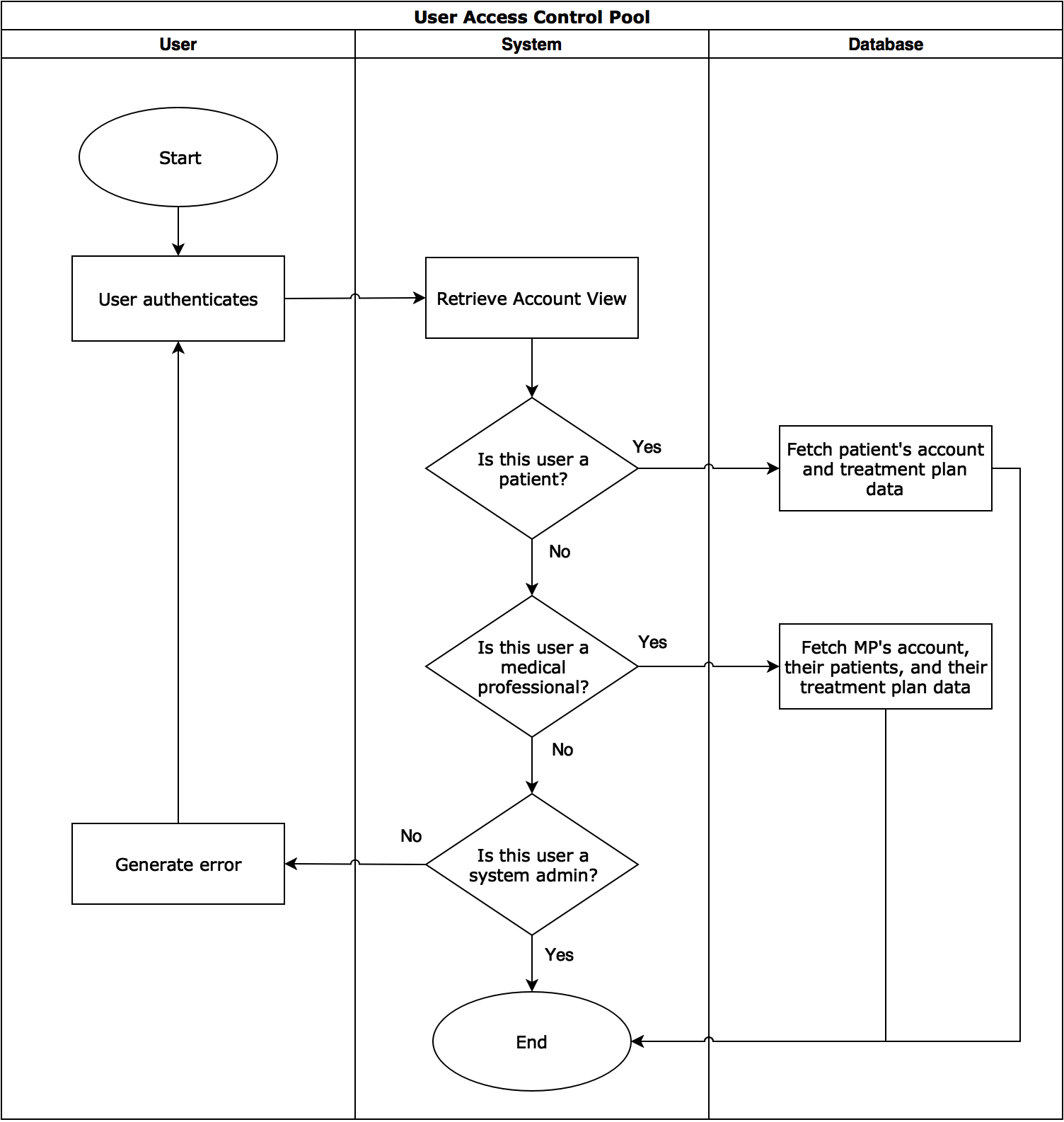
### 

### 

### 

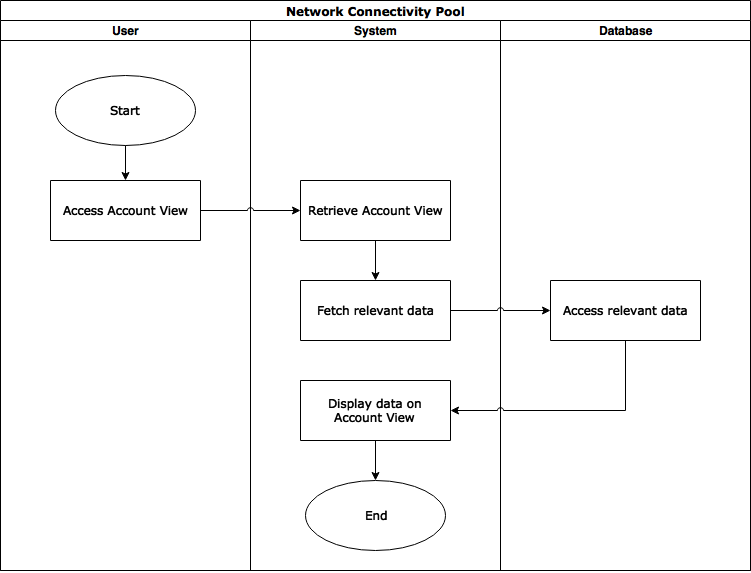
### User Access Control

The following diagrams portrays user access control and the actions that occur from the user, the system, and the database during this event. User access control shows what a specific user is allowed to view and/or edit on the web application. The precondition for the diagram is the user must have valid credentials to authenticate with.

[](https://www.draw.io/?page=9&scale=2#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

### Network Connectivity

These diagrams show actions taken by the system when the user wants to connect to the server. The precondition for the diagrams are that the system is deployed and active.

[](https://www.draw.io/?page=10#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

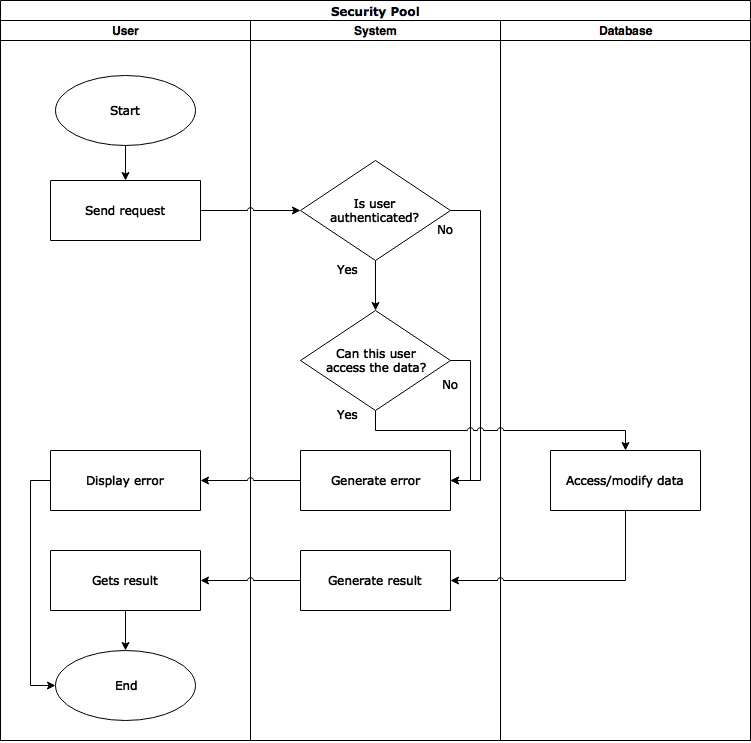
### 

### 

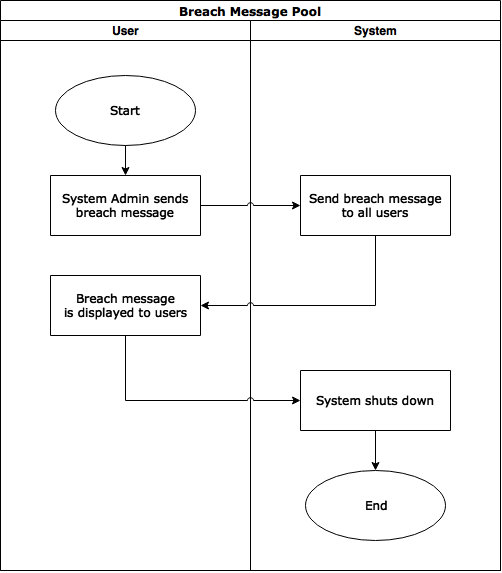
### Security

The diagram below shows how the system ensures all requests and connections to the system are secure by testing each request. The precondition for both the first and second start is that the system is active and deployed. If there is a data breach, the second start shows actions both the system administrator and the system do to send a breach message to all users. The precondition for the second start is that a system administrator exists and the system has been subject to a security breach.

*Secure connection*

[**](https://www.draw.io/?page=11#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

*Breach message*

[**](https://www.draw.io/?page=12#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

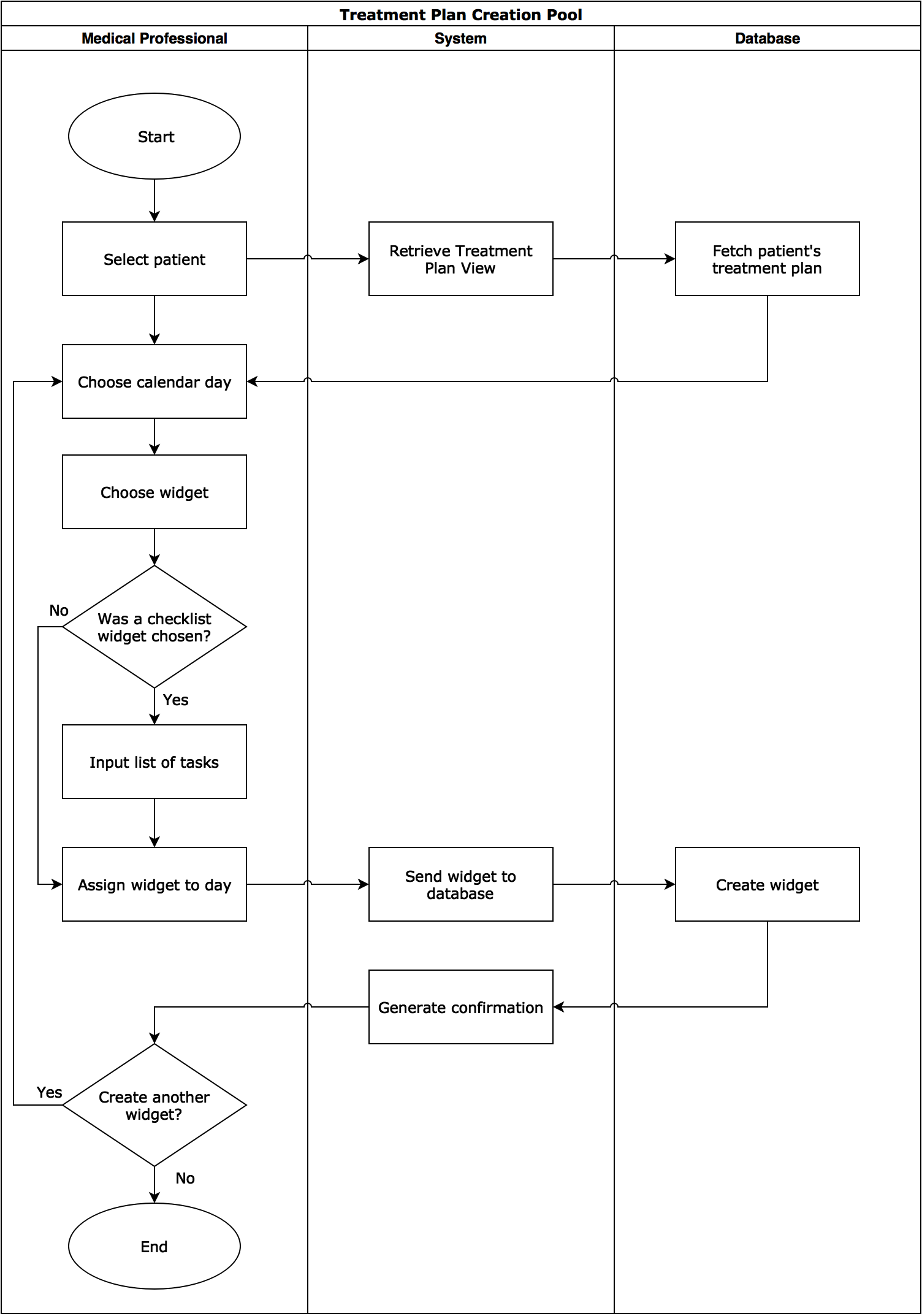
### 

### 

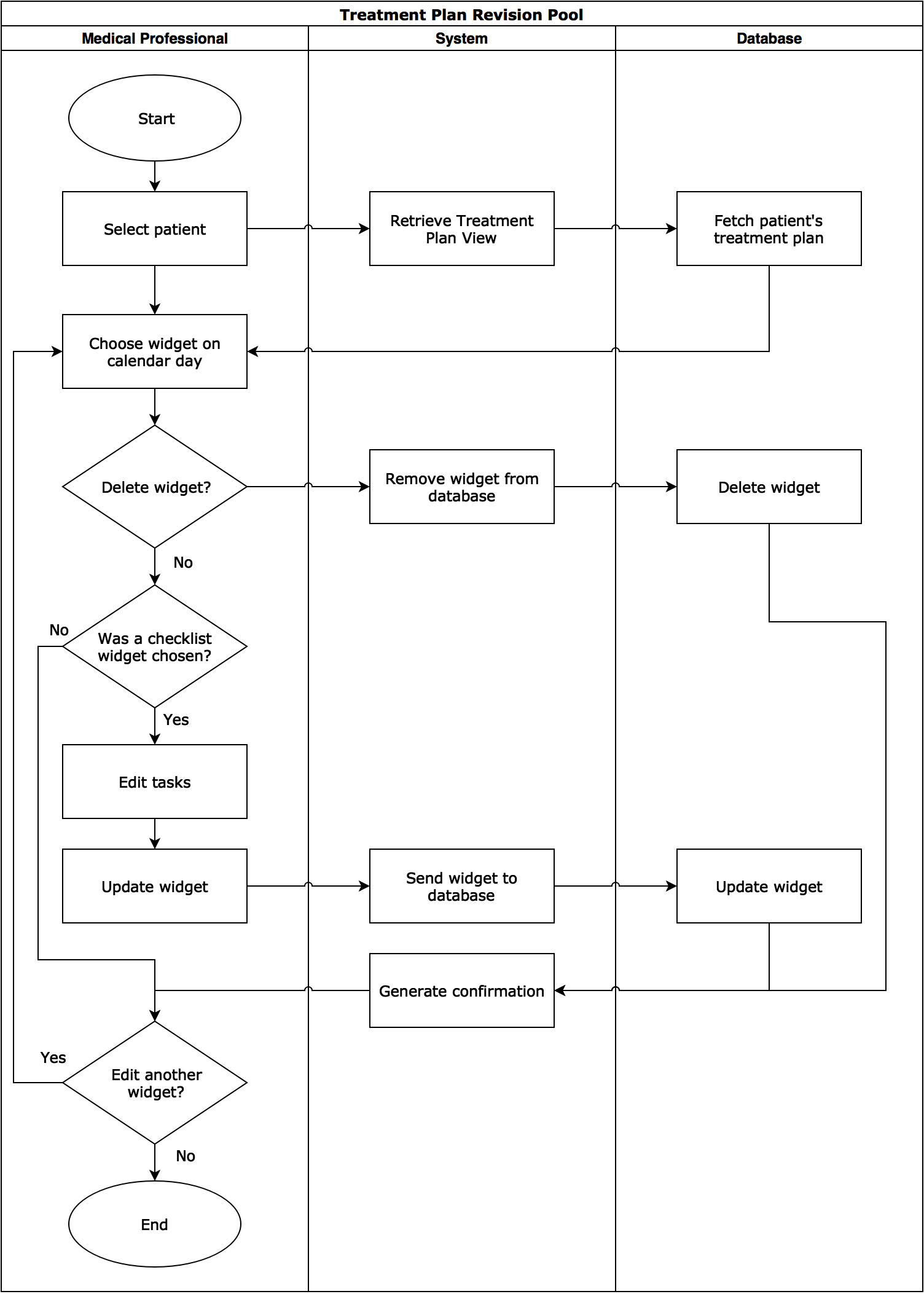
### Treatment Plan Creation and Revision

This diagram shows the actions taken by the medical professional, the system, and the database during the creation of a treatment plan. The precondition is that the medical professional is authenticated into the system and they have at least one patient registered under them.

*Create treatment plan*

[**](https://www.draw.io/?page=13&scale=2#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

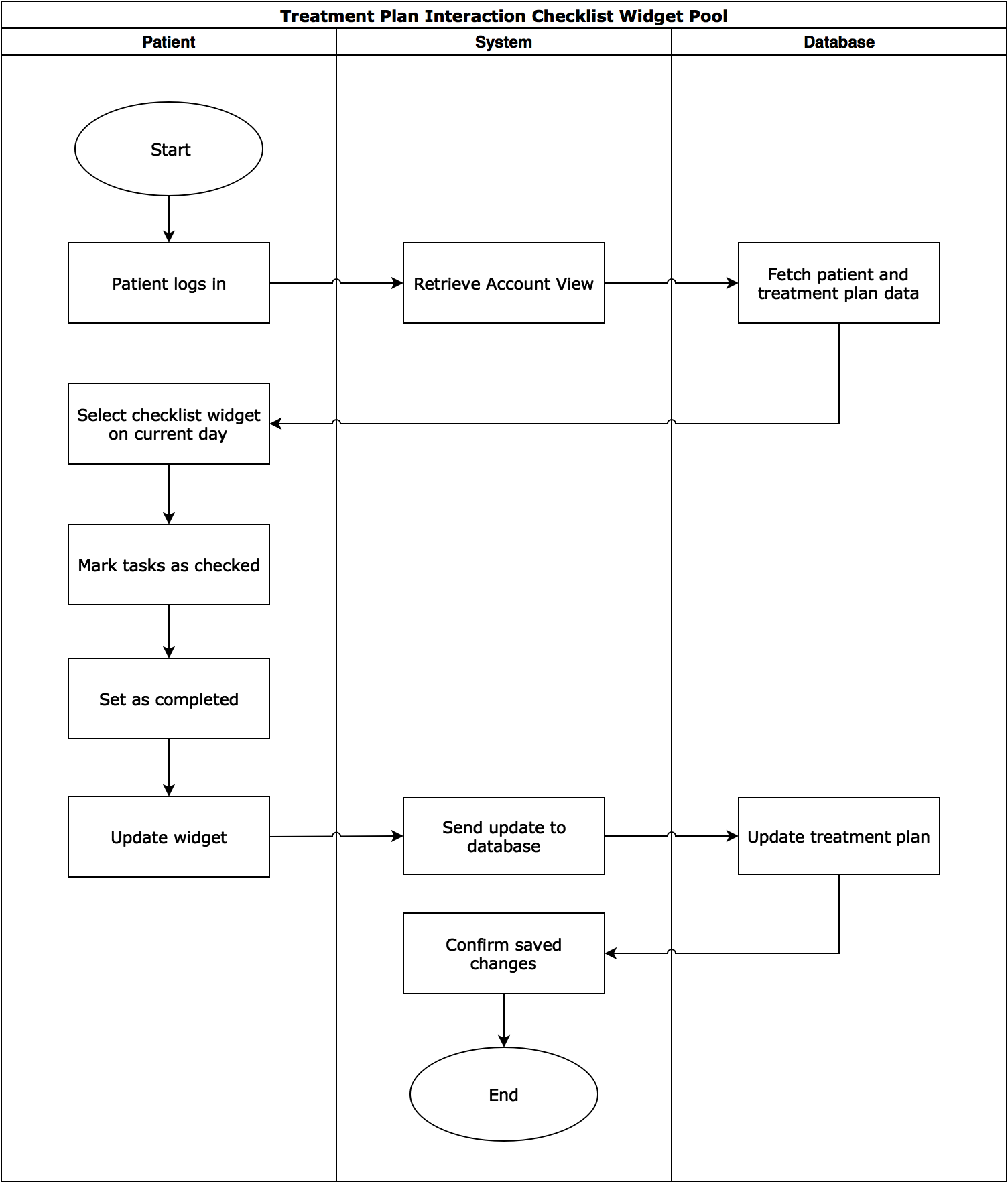
*Revise treatment plan*

[**](https://www.draw.io/?page=14&scale=2#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

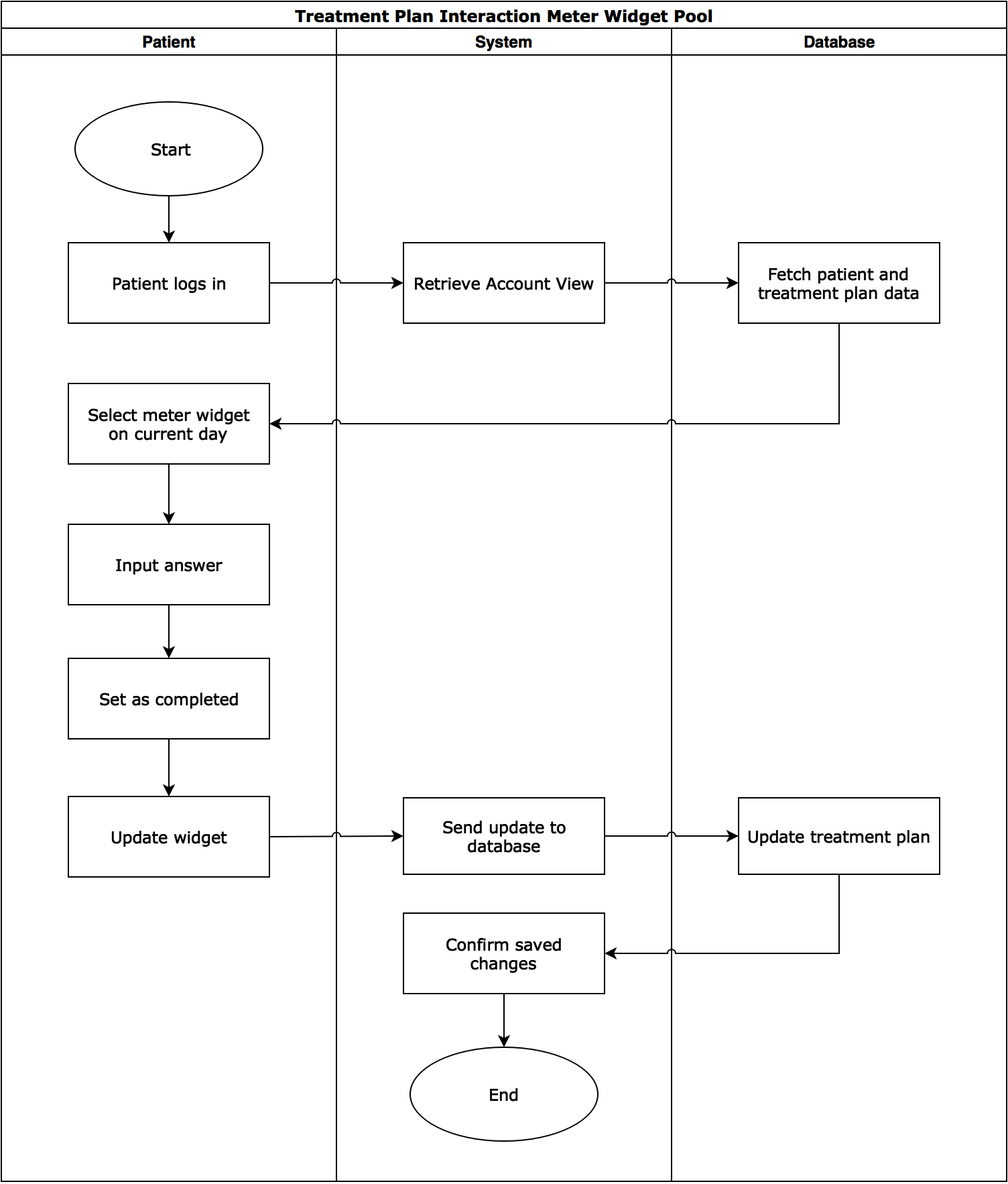
### Treatment Plan Interaction

The following diagram shows the actions taken by the patient, the system, and the database acting during a patient’s interaction with their treatment plan. The precondition for this diagram is that the patient is authenticated into the system and that they already have a preexisting treatment plan.

*Interaction with checklist widget*

[**](https://www.draw.io/?page=15&scale=2#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

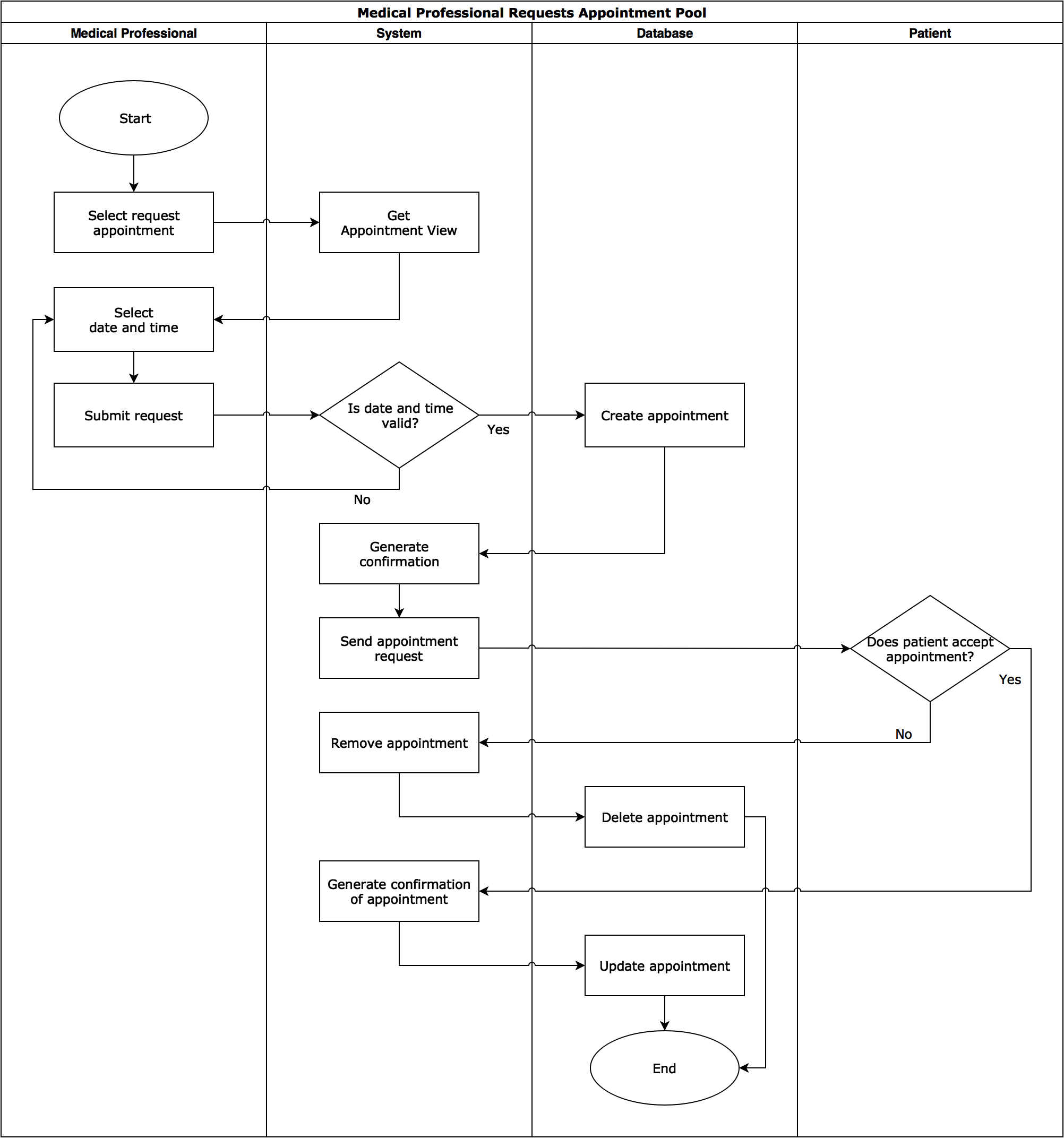
*Interaction with meter widget*

[**](https://www.draw.io/?page=16&scale=2#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

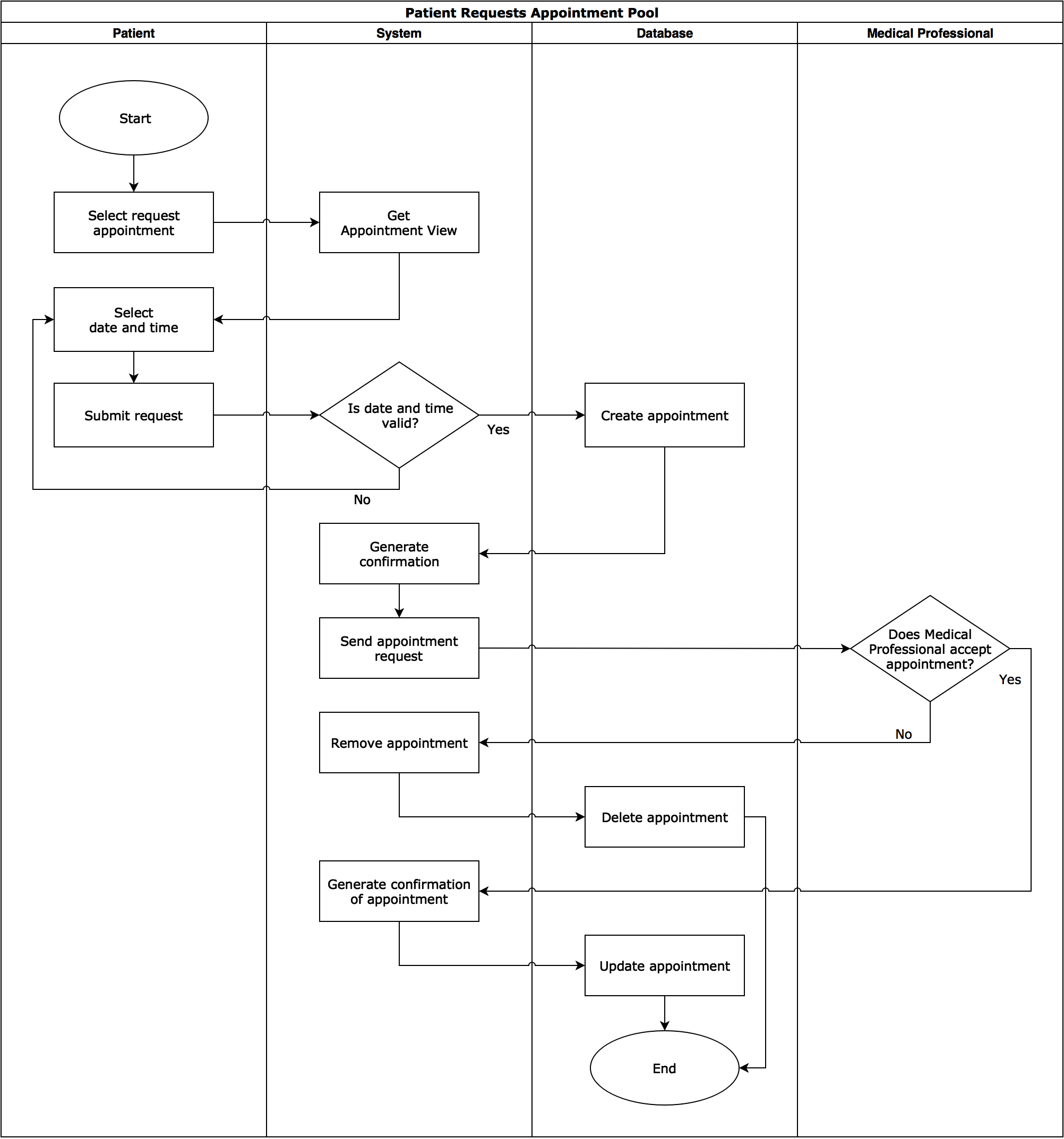
### Request Appointment

Outlined below are the swimlane diagrams for requesting appointments from both the patient’s point of view (first diagram) and the medical professional’s point of view (second diagram) and the actions taken by the patient, the system, the database, and the medical professional when an appointment is requested. The precondition of the first diagram is that the patient is authenticated into our system. The precondition of the second diagram is that the medical professional is authenticated into our system and has at least one patient to request an appointment with.

*Medical professional requests appointment, patient responds to request*

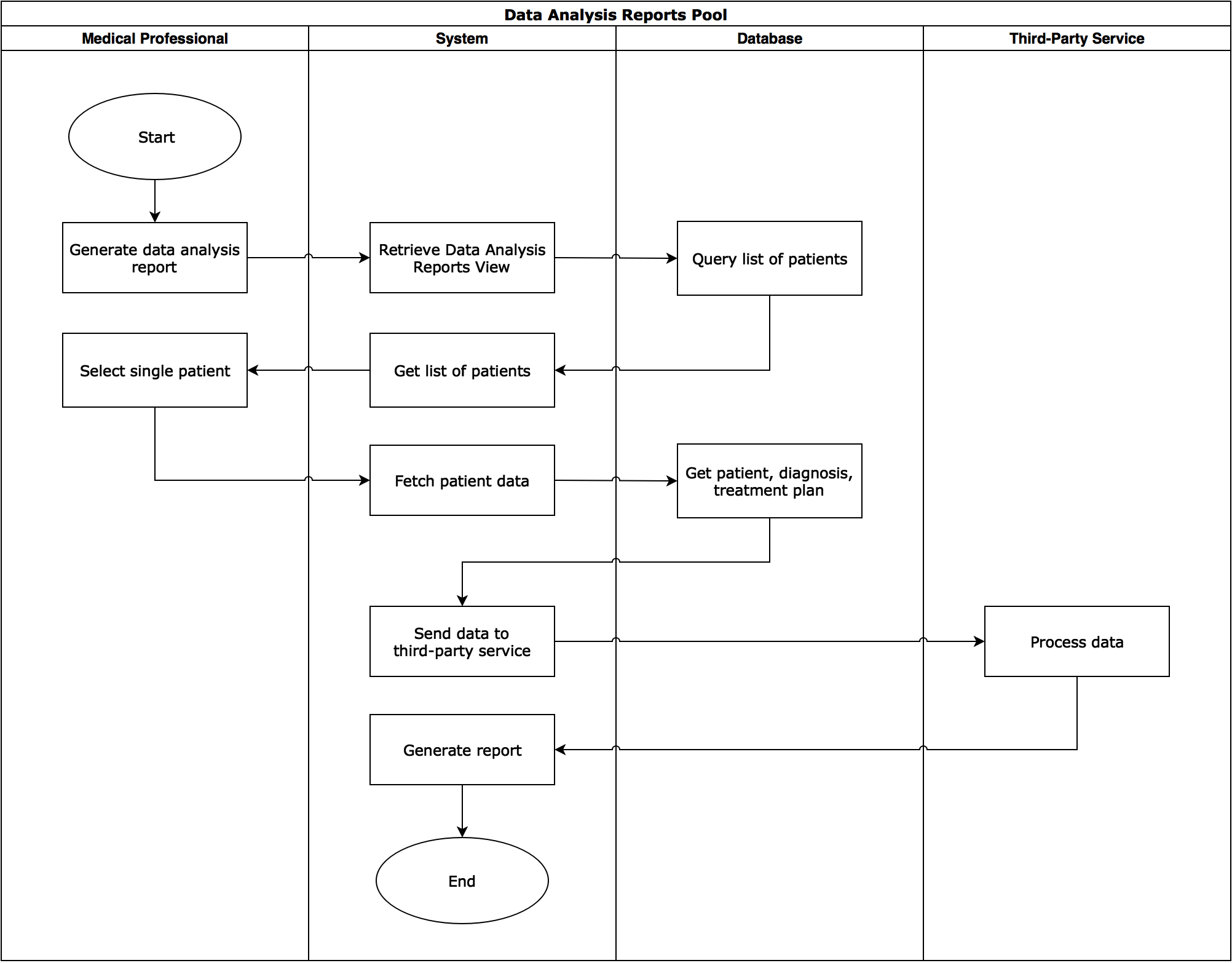
[**](https://www.draw.io/?page=17&scale=2#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

*Patient requests appointment, medical professional responds to request*

[**](https://www.draw.io/?page=18&scale=2#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

### Data Analysis Reports

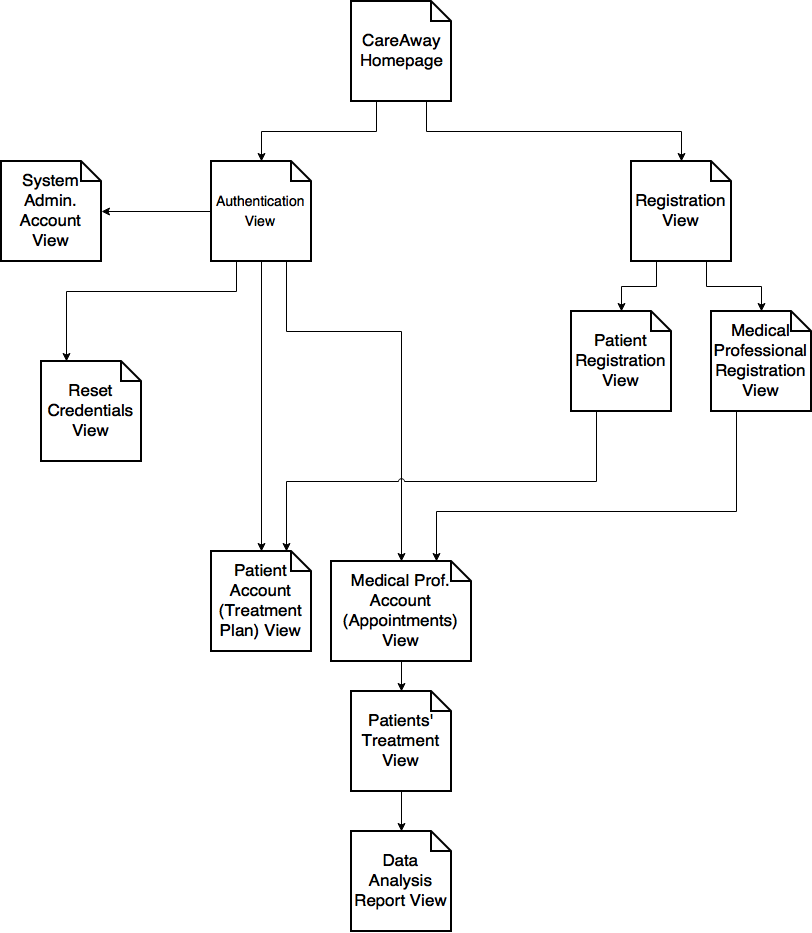
Data Analysis Reports deals with generating reports for a medical professional for a given patient. The following diagram outlines the actions taken by the medical professional, the system, the database, and the third-party service during this functionality. The precondition for this diagram is that the medical professional exists and is authenticated into the system. The medical professional needs to have at least one patient and the patient must have preexisting treatment data prior to report generation.

[](https://www.draw.io/?page=19&scale=2#G1dV15oeiTY21oDchTFBTqpEqx_CtMgA91)

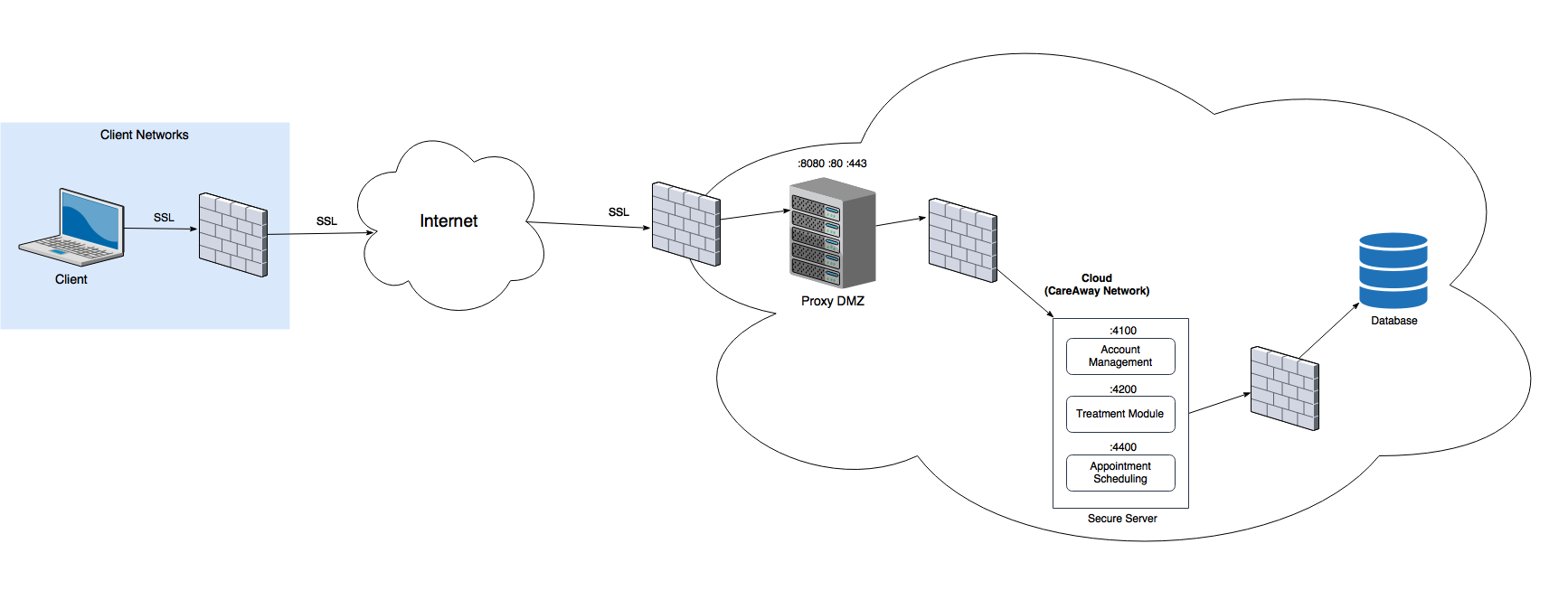
## 

## 

## **Site Map**



This is a high level sitemap for the CareAway Treatment Planner. On this sitemap, each page shows the subsequent pages that it could possibly lead to.



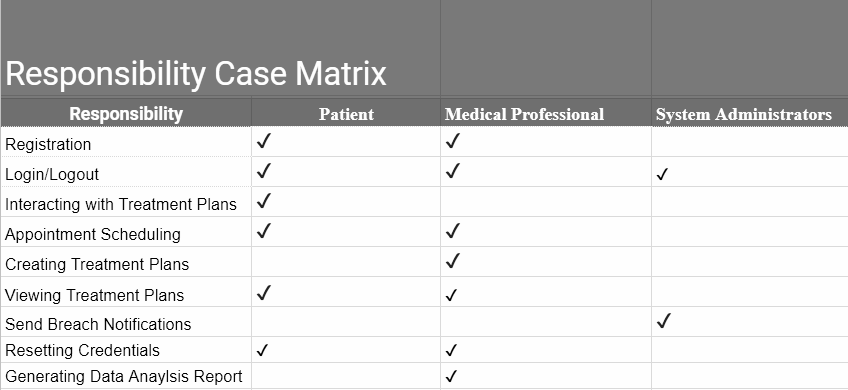
## Network Diagram



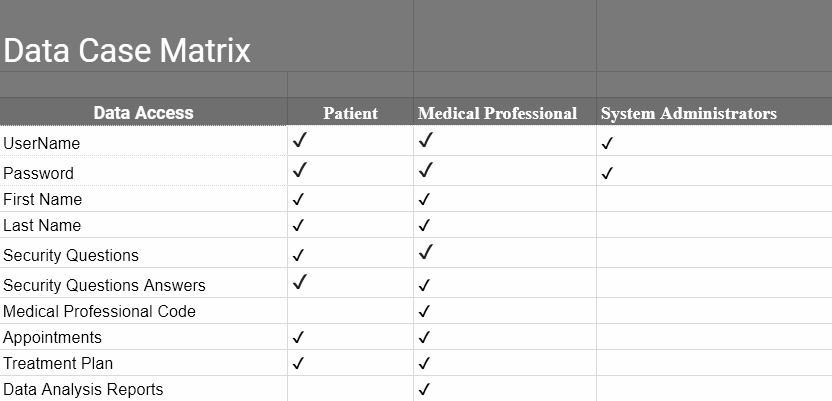
# **Low Level Design**

## **Responsibility Matrix**

The Responsibility Matrix illustrates the different functionalities each actor partakes in the system. The following below shows the actors per each columns and the responsibilities per each row as display below.



Below is the Data Case Matrix illustrates the different data that each actor makes and has access to within the CareAway Treatment Planner.The following below shows the actors per each columns and the data per each row as display below.



## Entity Relation Diagram

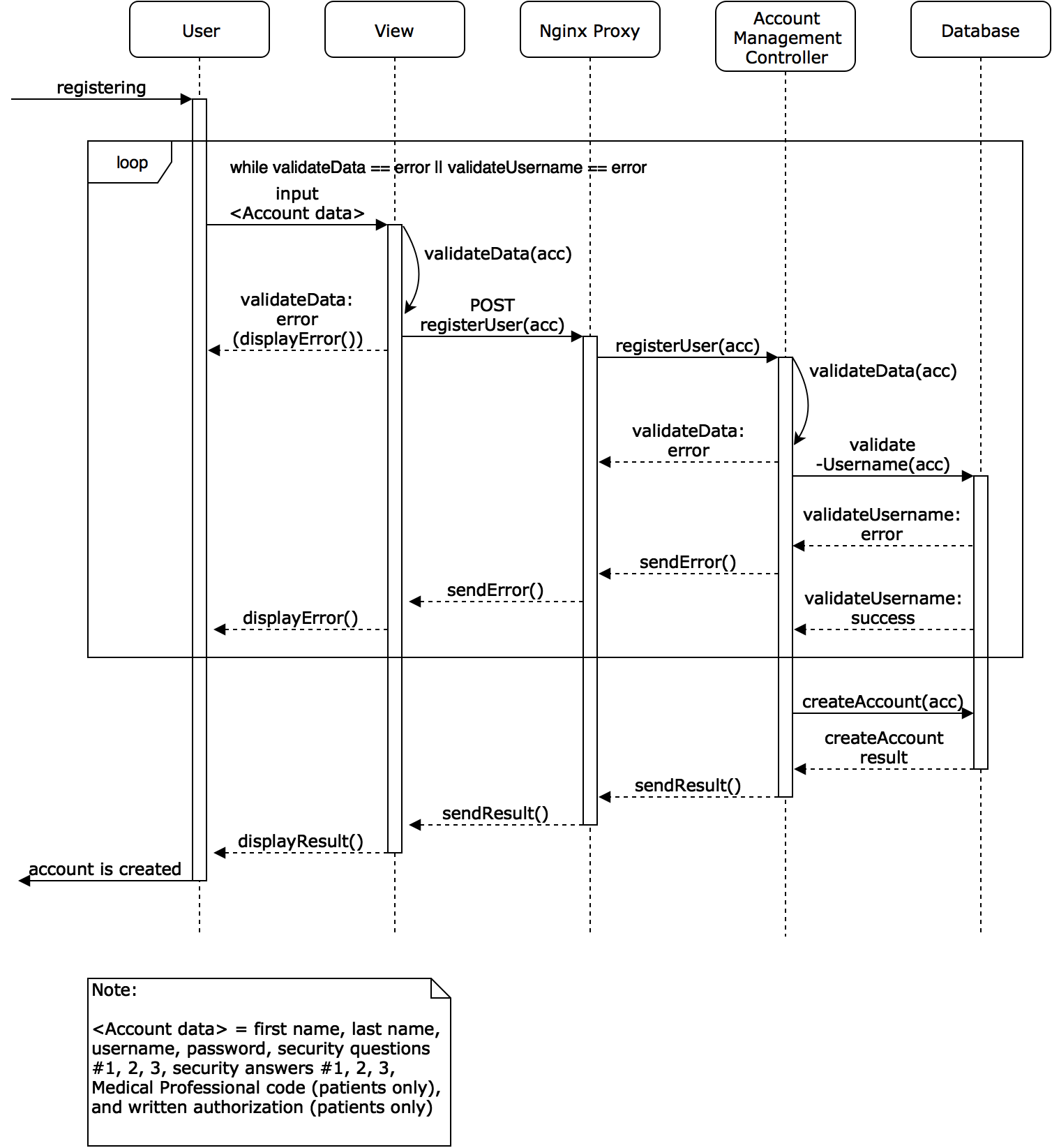
## 

## 

## **Sequence Diagrams**

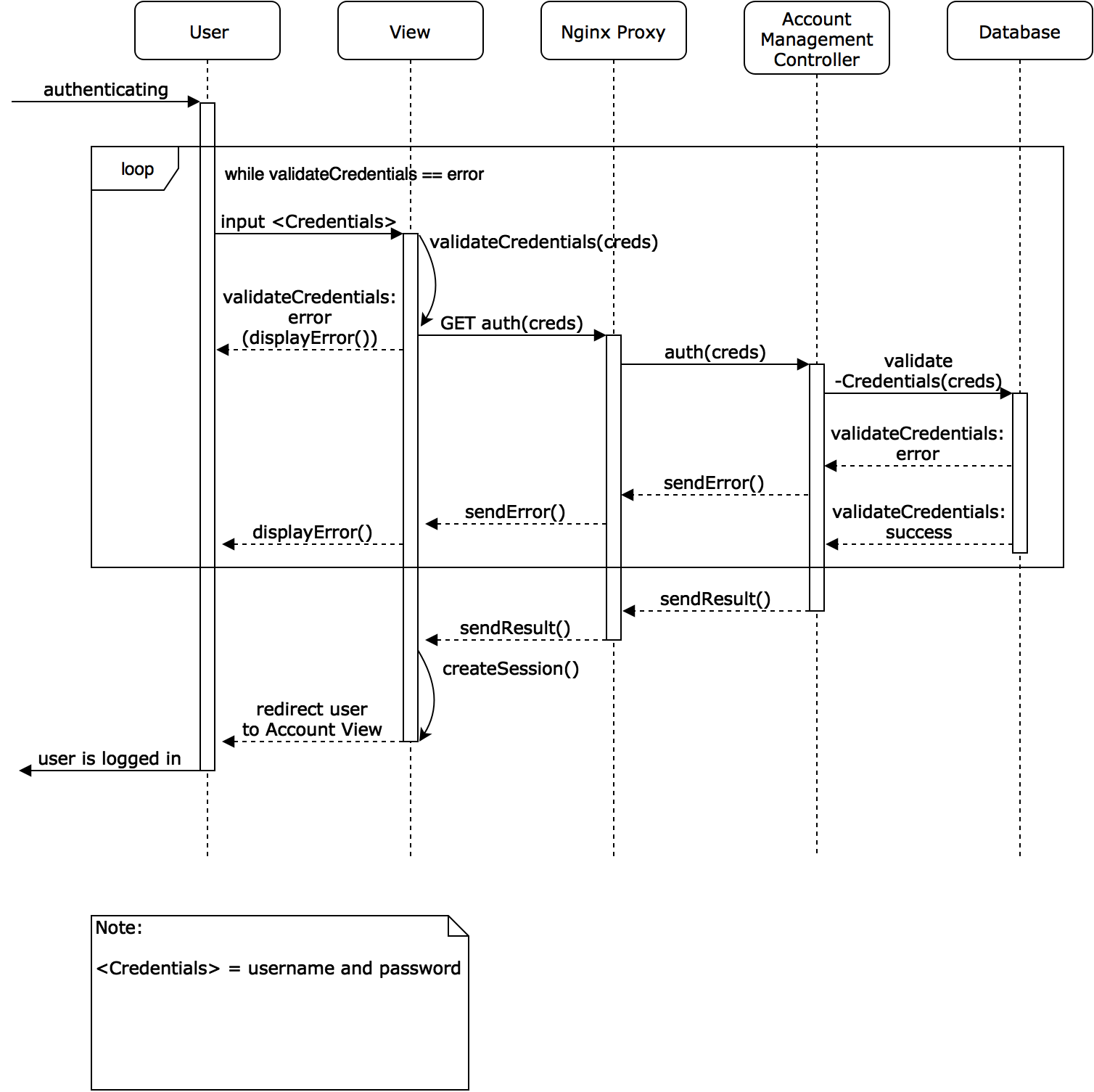
### Registration

During the registration of a user (medical professional or patient), the following details the actions that will occur by the User, View, Nginx Proxy, Account Management Controller, and Database. The precondition for this diagram is that the user is currently on either the Patient Registration View or the Medical Professional Registration View, depending on if they are either a patient or medical professional respectively. The user must also not be authenticated into the system.

[](https://www.draw.io/?scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

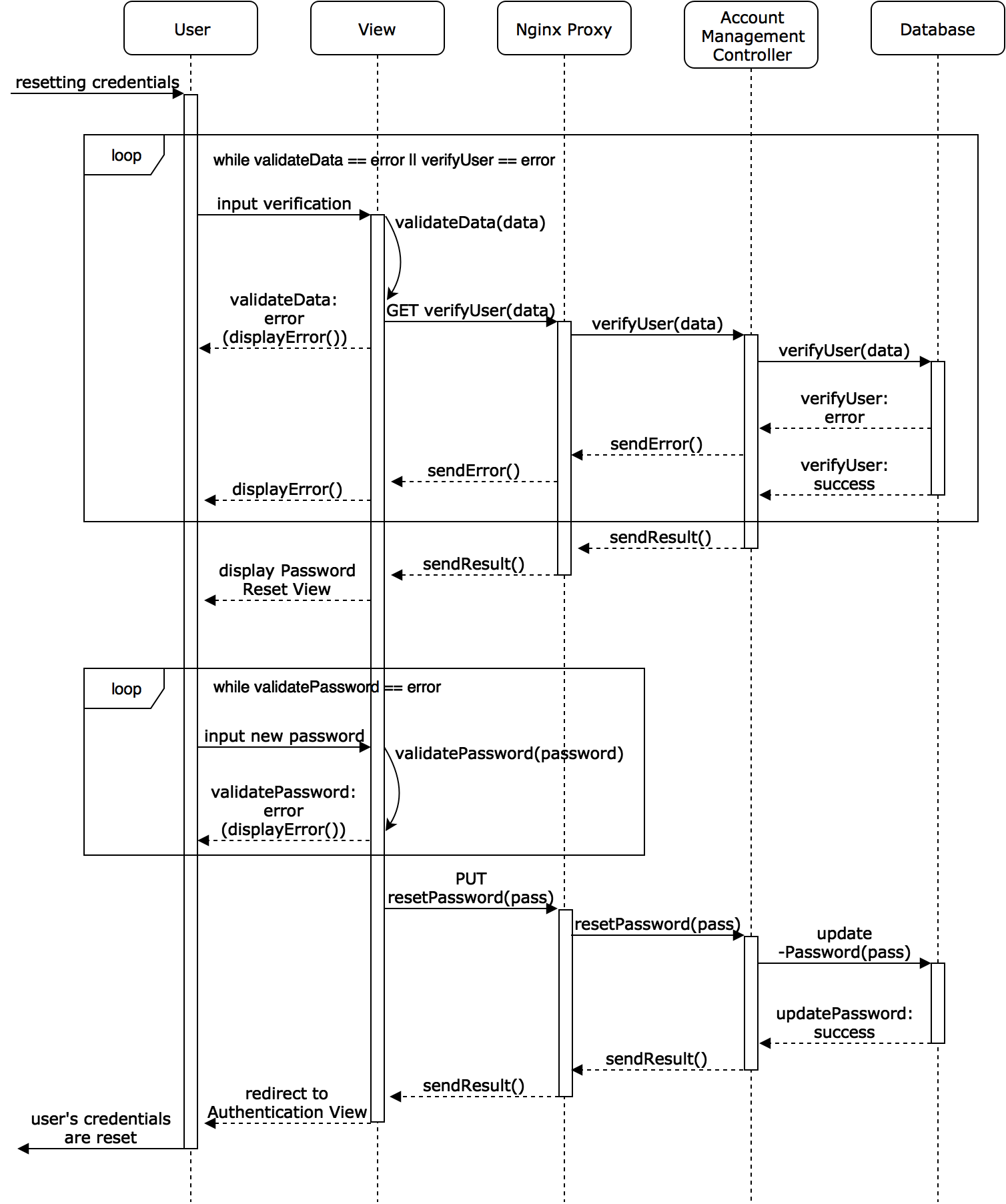
### Authentication

This diagram shows the process of a user (medical professional, patient, or system administrator) providing their credentials for authentication. The precondition is that the user is currently on the Authentication View and they are not authenticated into the system. The view handles the user’s inputted data, which (once verified as appropriate input) is sent through the proxy. The proxy hands it off to the Account Management Controller that validates whether or not the credentials match one of the credentials in the database. The results are returned to the user.

[](https://www.draw.io/?page=1&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

### Resetting Credentials

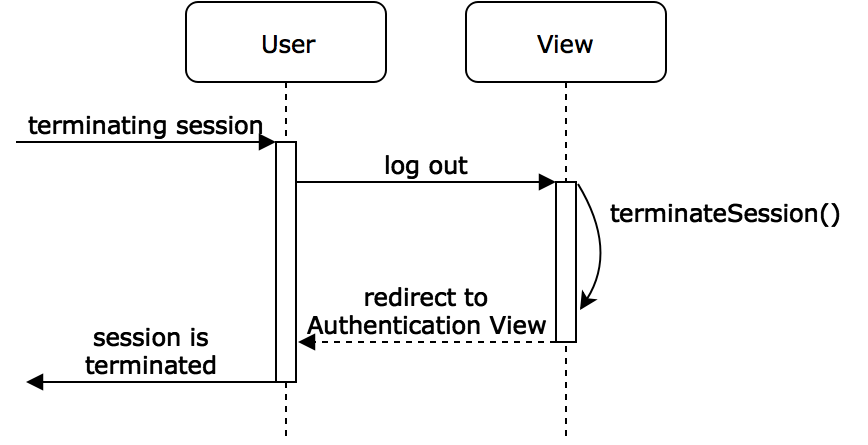
This diagram shows the process of a user (medical professional, patient, or system admin) resetting their credentials by verifying who they are and then providing a new password if they are verified. The precondition is that the user is on the Reset Credentials View and they are not authenticated into the system.

[](https://www.draw.io/?page=2&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

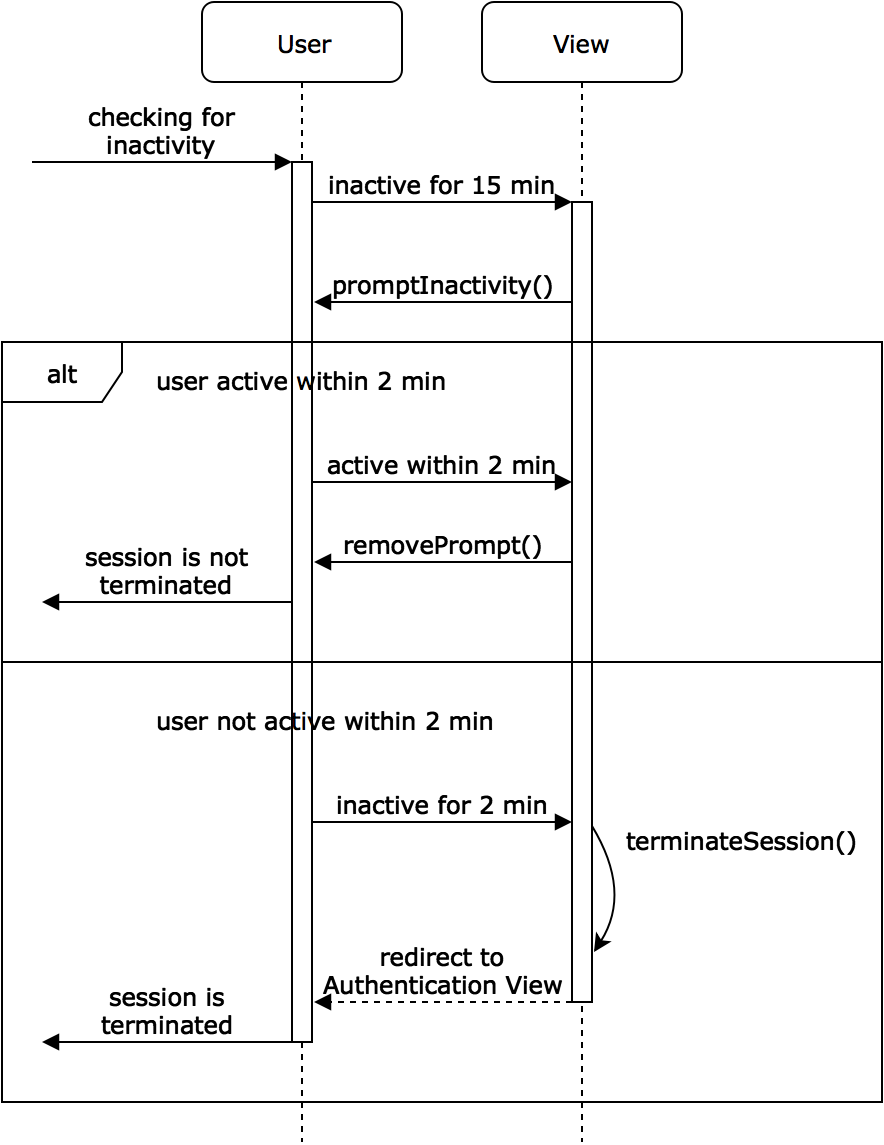
### Session Termination

The following diagrams show two ways a user can terminate a session, one by logging out and one by being inactive for a certain amount of time. The precondition for both diagrams is that the user is authenticated into the system.

*Logging out*

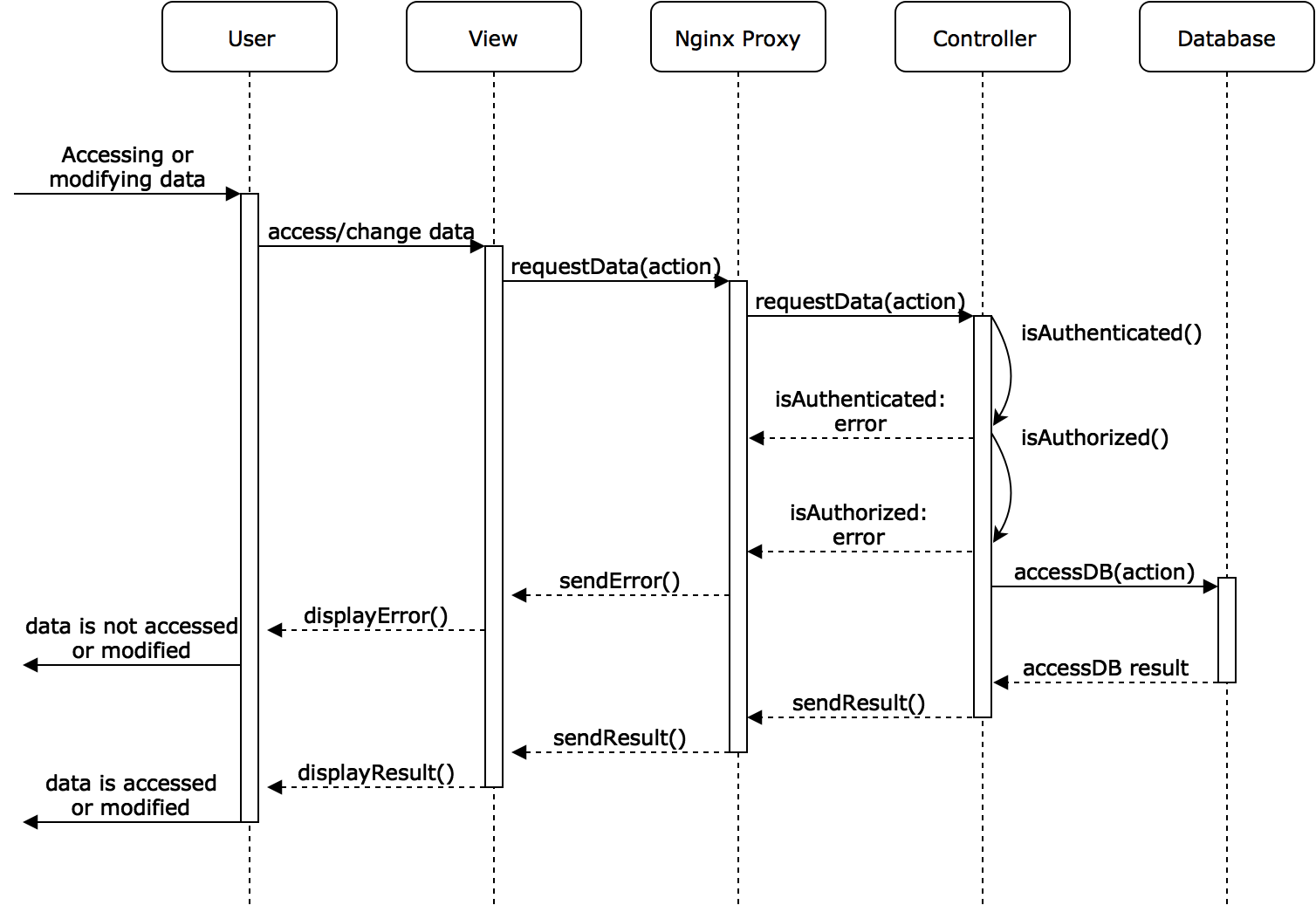
[**](https://www.draw.io/?page=3&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

*Being inactive*

[**](https://www.draw.io/?page=4&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

### Data Access

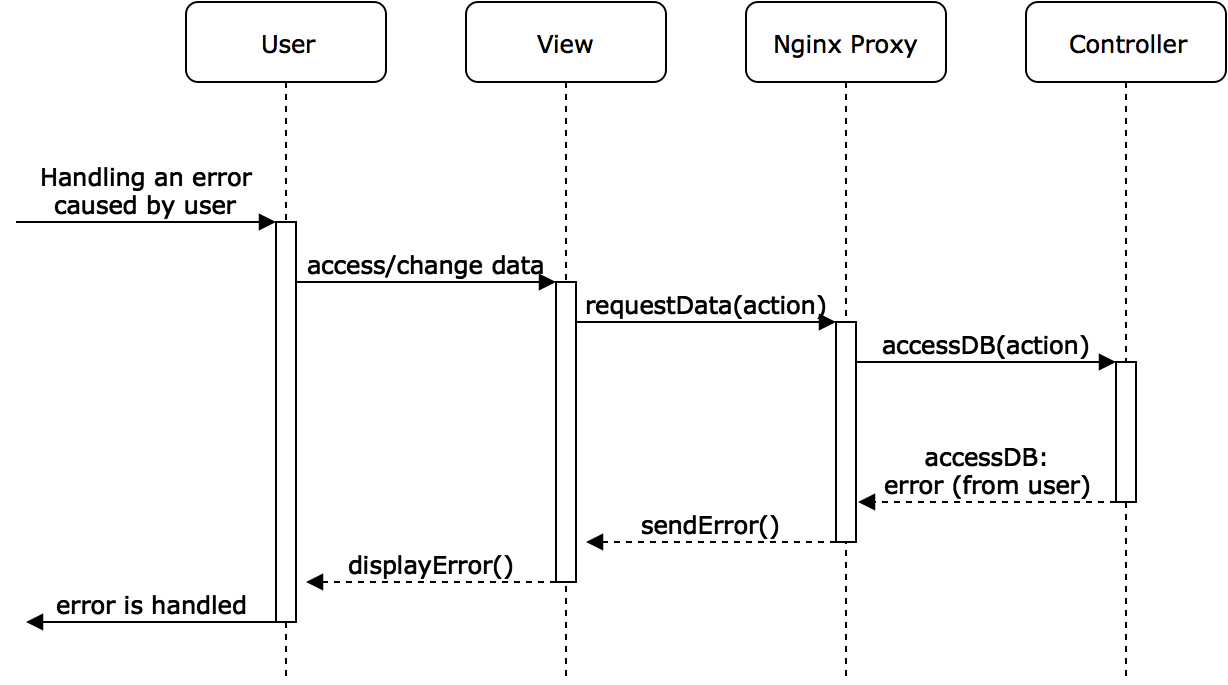
This diagram shows the process of a user requesting to access or modify data. Only users who are authenticated and authorized can access certain data. The precondition for this is that the system is active and deployed.

[](https://www.draw.io/?page=5&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

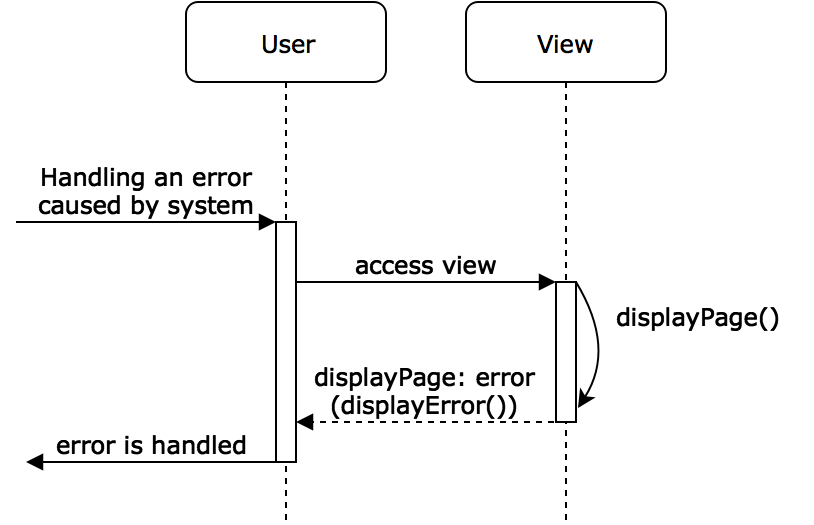
### Error Handling

These diagrams show how the system handles errors made by both the user and the system. The errors made by the system can occur in both the front end (second diagram) and the back end (third diagram). The errors caused by the front end entail rendering errors and the errors from the backend entail improper data access or storage. The precondition for all three diagrams is that the system is active and deployed.

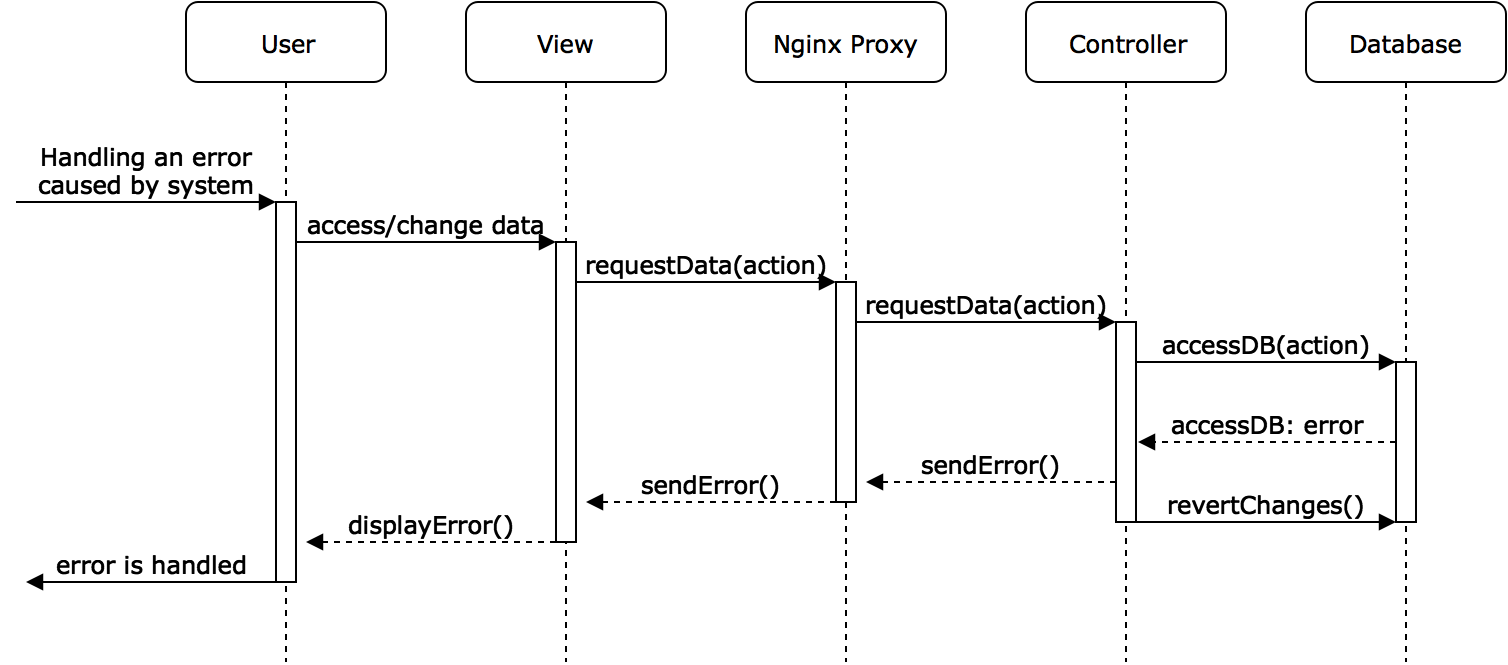
*Errors made by user*

[**](https://www.draw.io/?page=6&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

*Errors made by system (frontend)*

[**](https://www.draw.io/?page=7&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

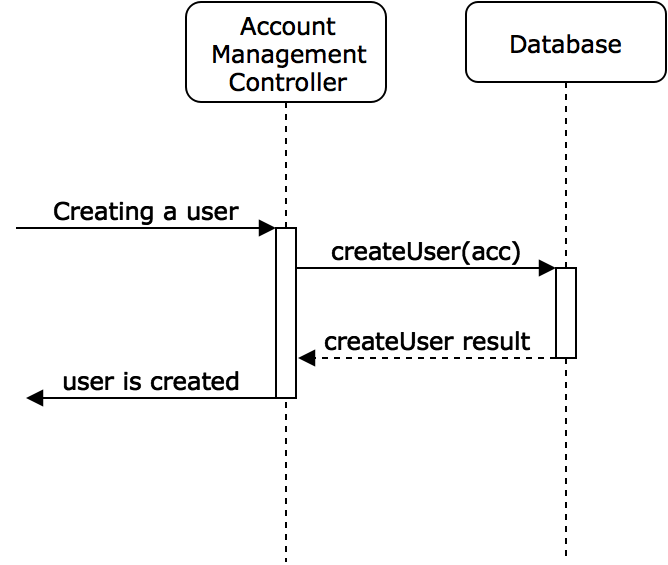
*Errors made by system (backend)*

[**](https://www.draw.io/?page=8&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

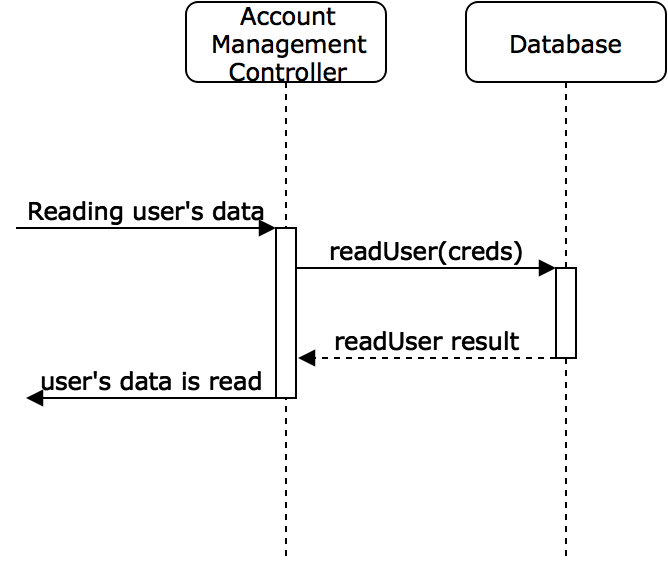
### User Management

These diagrams encapsulate the system’s actions during the creation, reading, and updating of its users. Creating a user is also covered in the Registration functionality. Reading a user is also covered in the Authentication functionality. Updating a user is also covered in the Resetting Credentials functionality.

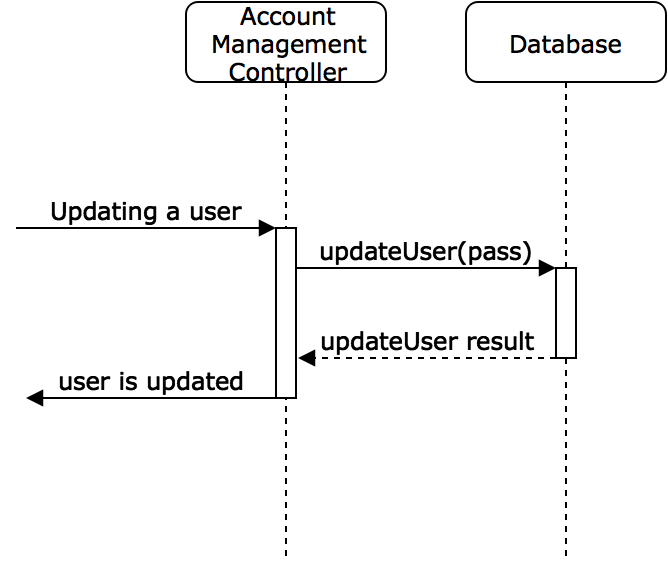
*Create user*

[**](https://www.draw.io/?page=9&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

*Read user*

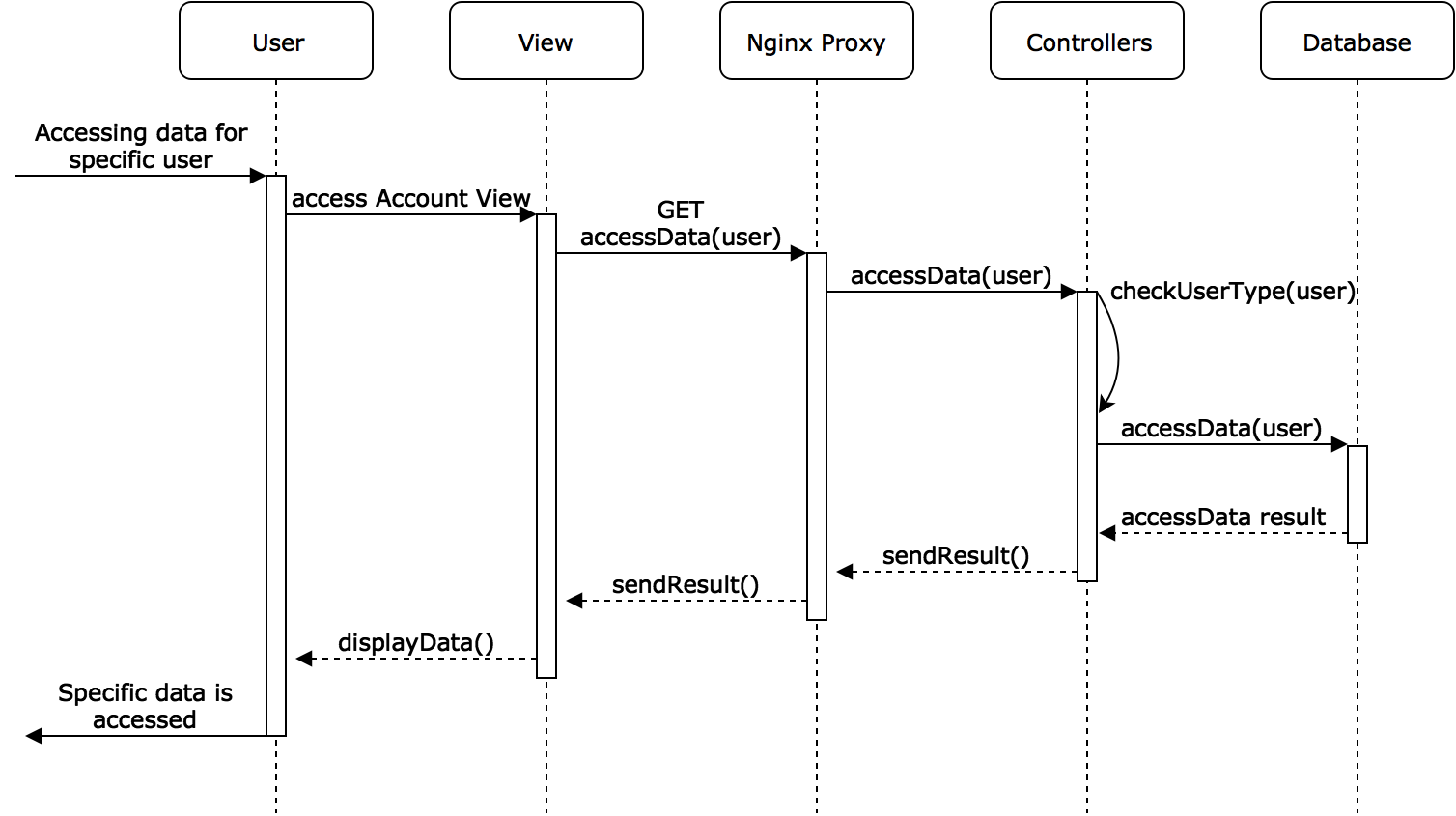
[**](https://www.draw.io/?page=10&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

*Update user*

[**](https://www.draw.io/?page=11&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

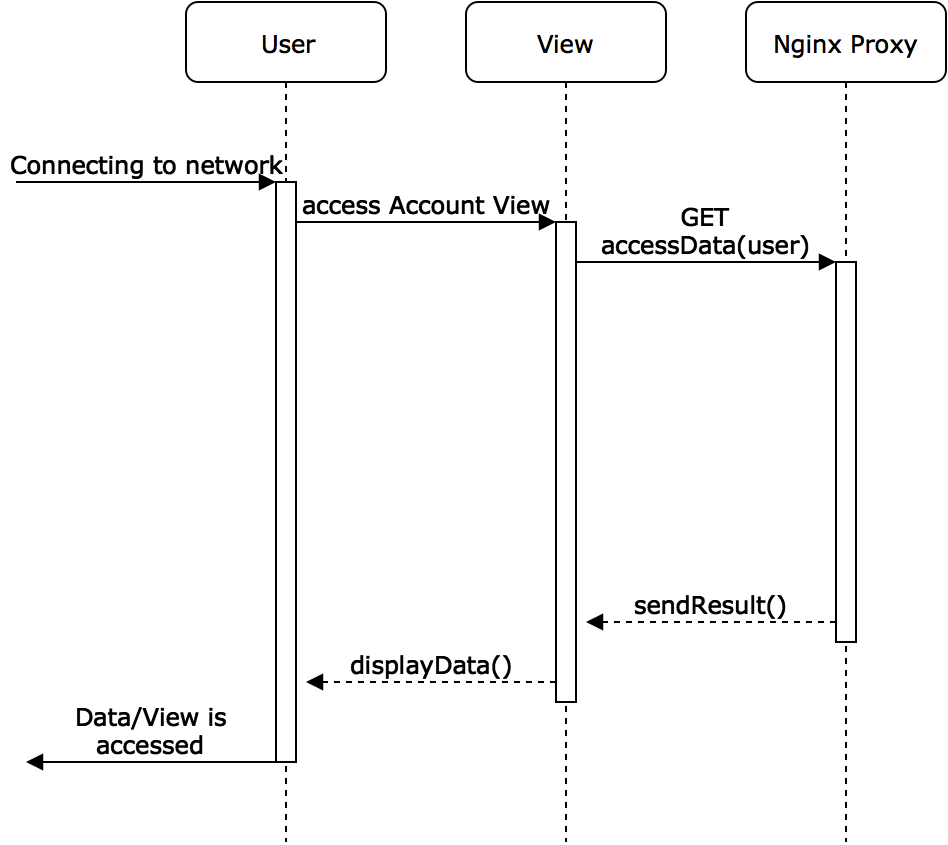
### User Access Control

This diagram shows the process of the system fetching specific data depending on what type of user is trying to access the data. The precondition is that the system is active and deployed.

[](https://www.draw.io/?page=12&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

### Network Connectivity

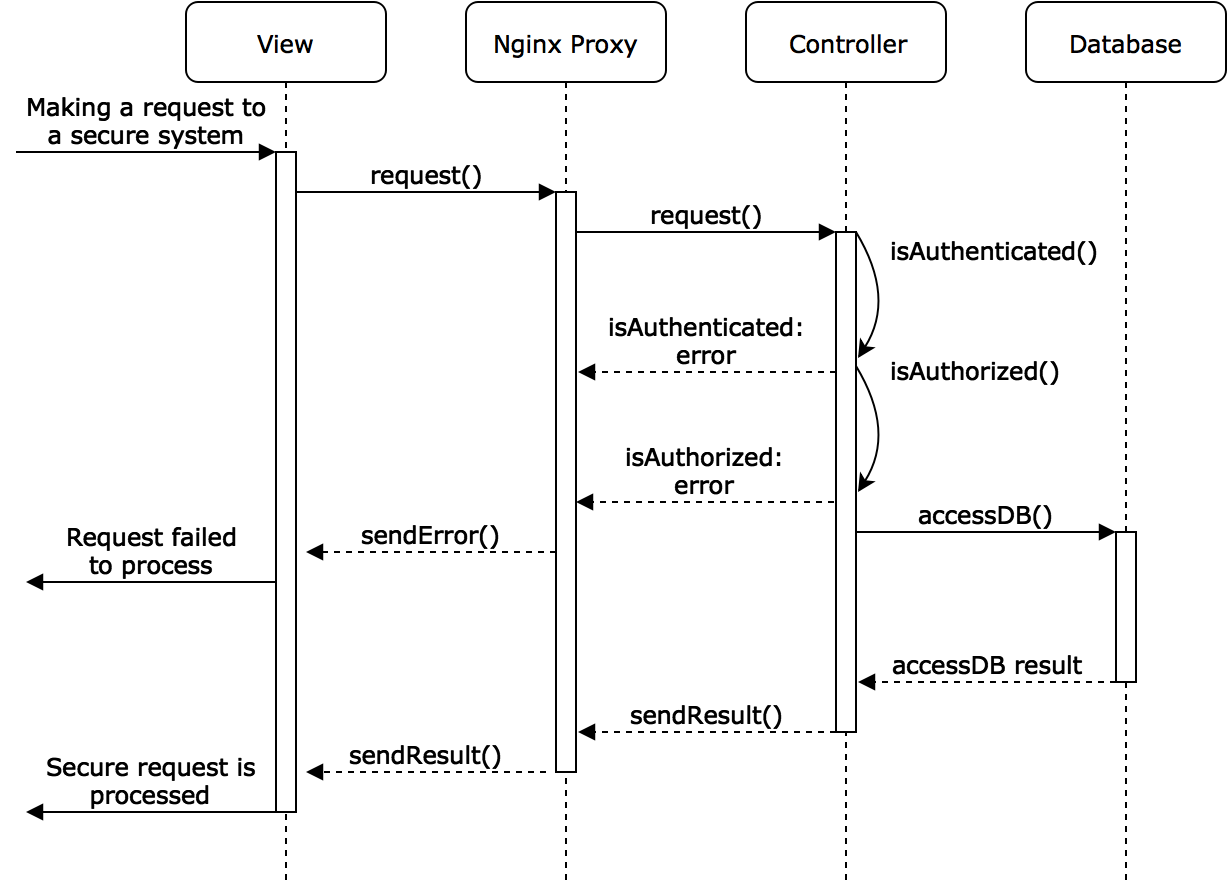
This diagram depicts how a user could connect to the system. The precondition is that the system is active and deployed.

[](https://www.draw.io/?page=13&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

### 

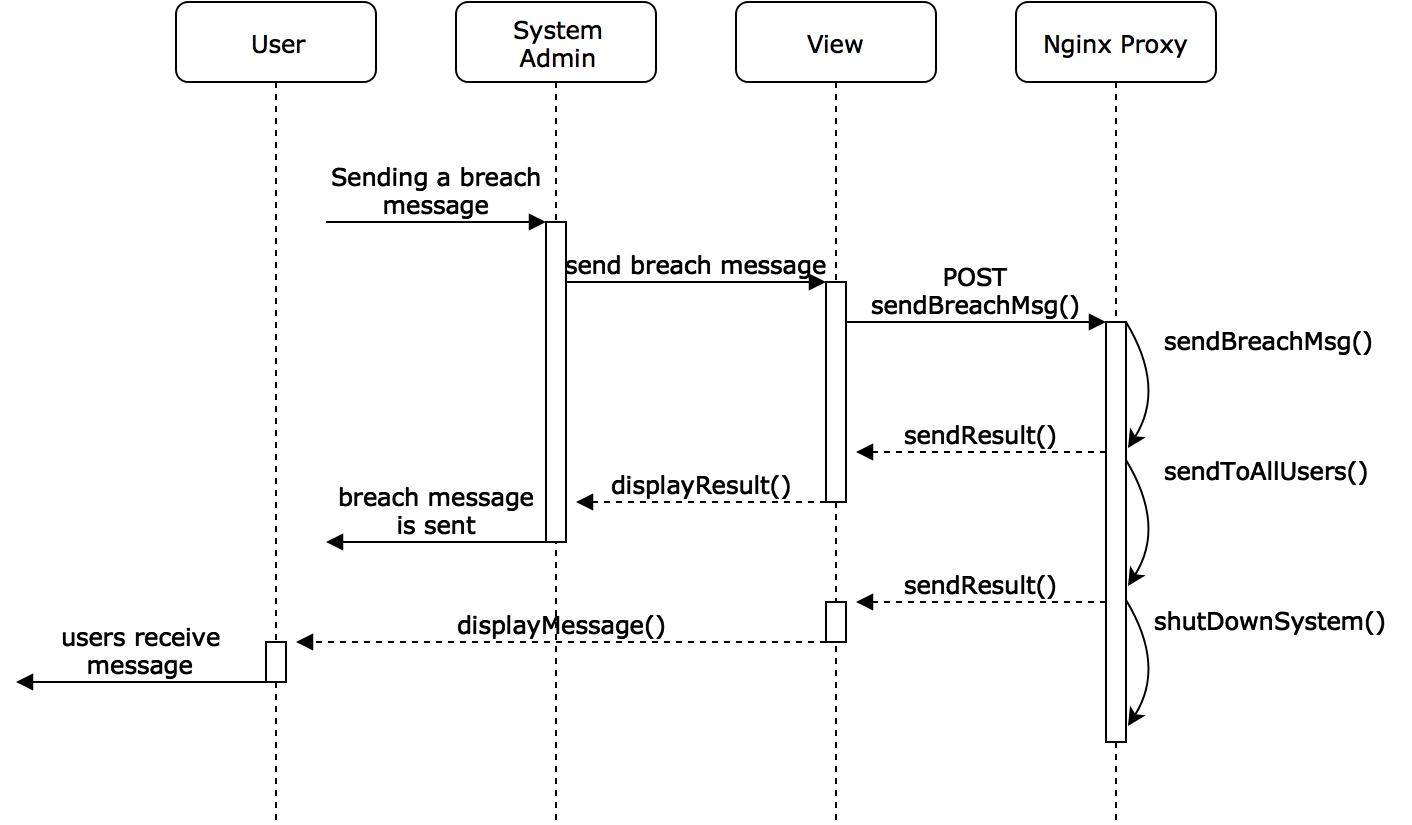
### Security

This diagram shows how the system ensures that all requests and connections to the server are secure. The precondition is that the system is active and deployed.

[](https://www.draw.io/?page=14&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

### Breach Messages

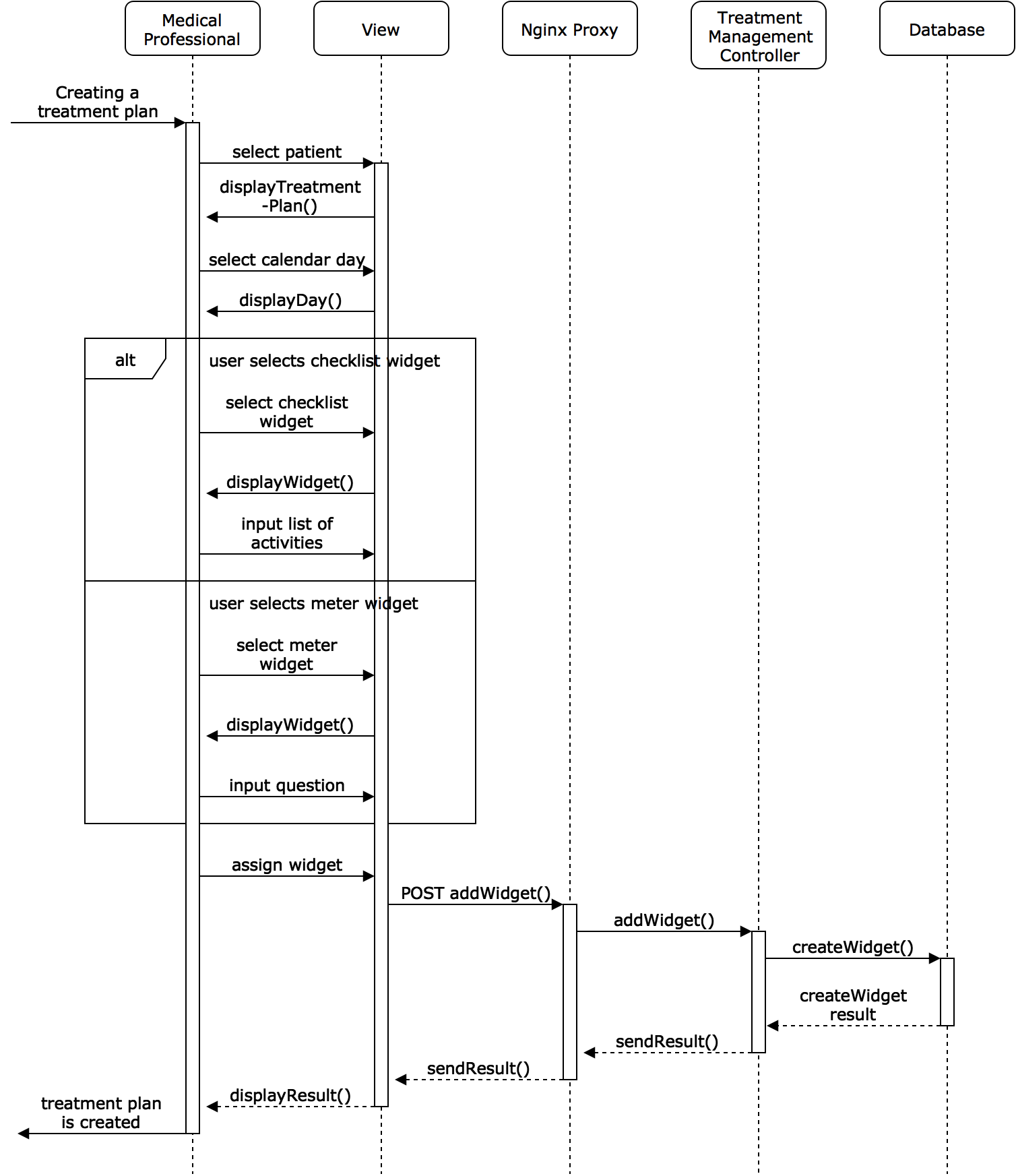
This diagram shows the process of a system administrator sending a breach message to all users. The precondition is that the system administrator is authenticated into the system and is on the System Administrator Account View. The system administrator must know of the breach beforehand.

[](https://www.draw.io/?page=15&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

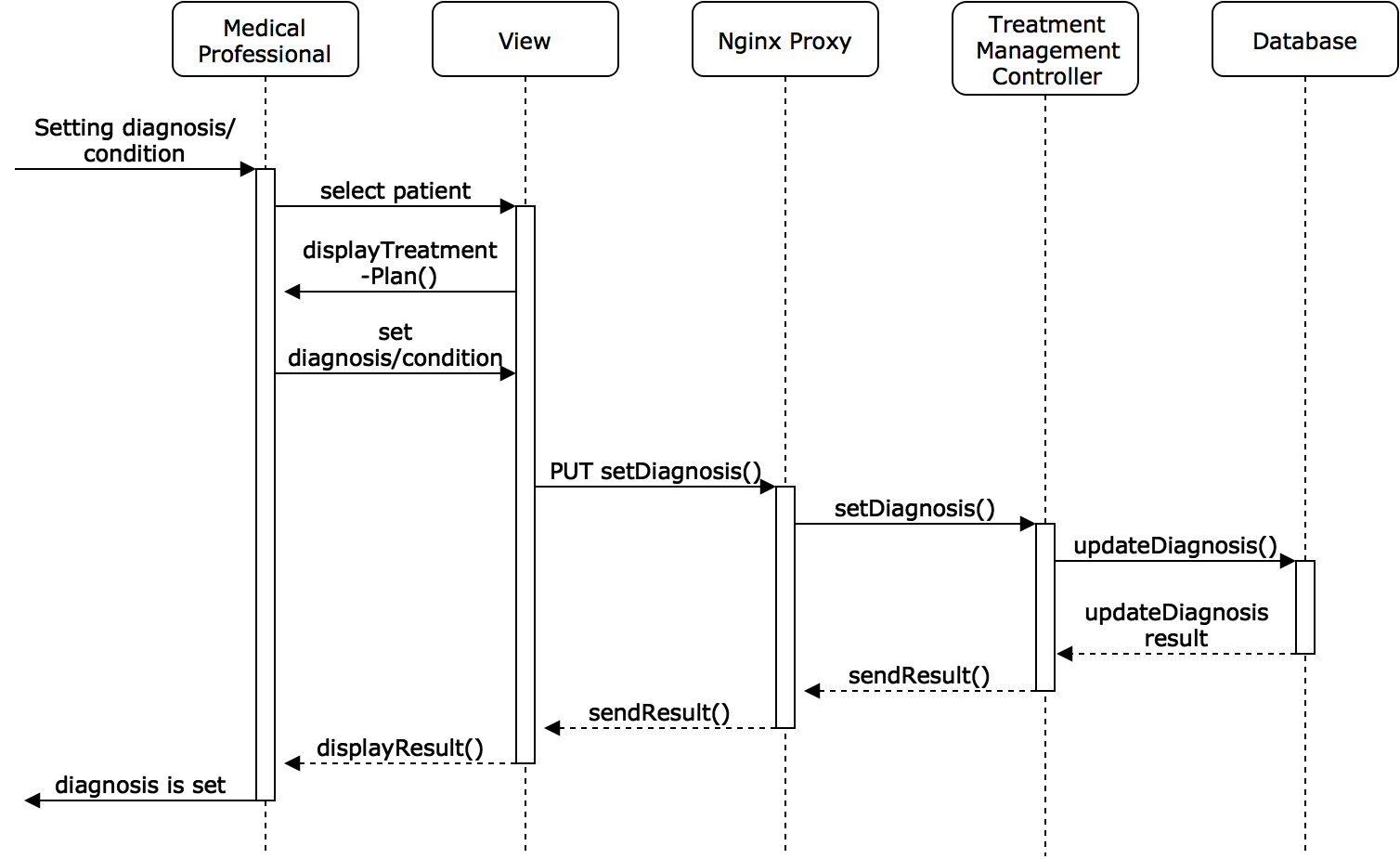
### Treatment Plan Creation

This diagram shows the process of either creating a treatment plan (first diagram) or setting the diagnosis for a patient (second diagram) by a medical professional. The precondition is that the medical professional is authenticated into the system and has at least one patient registered under him/her.

*Adding a widget for treatment plan*

[**](https://www.draw.io/?page=16&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

*Setting diagnosis for treatment plan*

[**](https://www.draw.io/?page=17&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

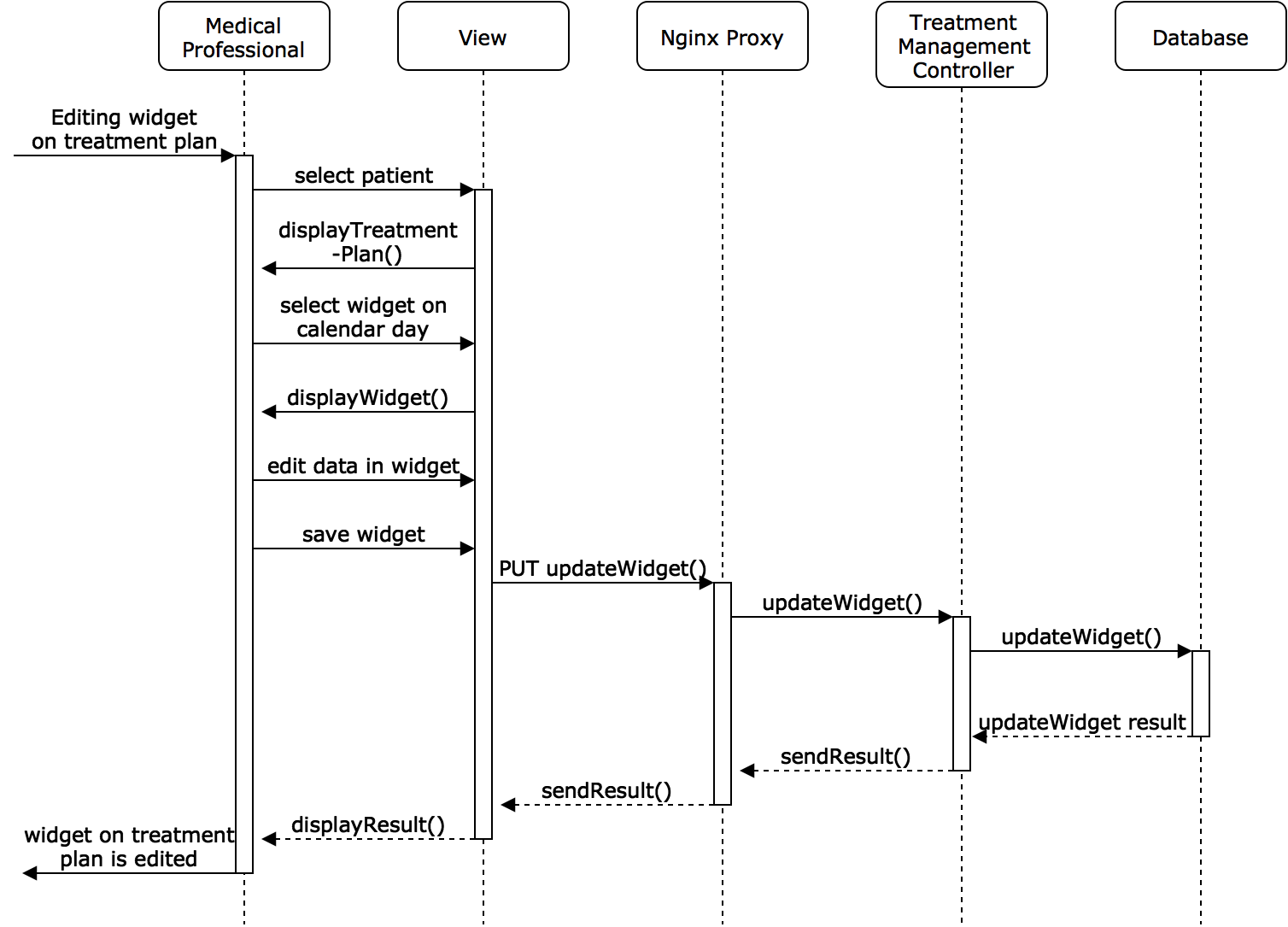
### 

### 

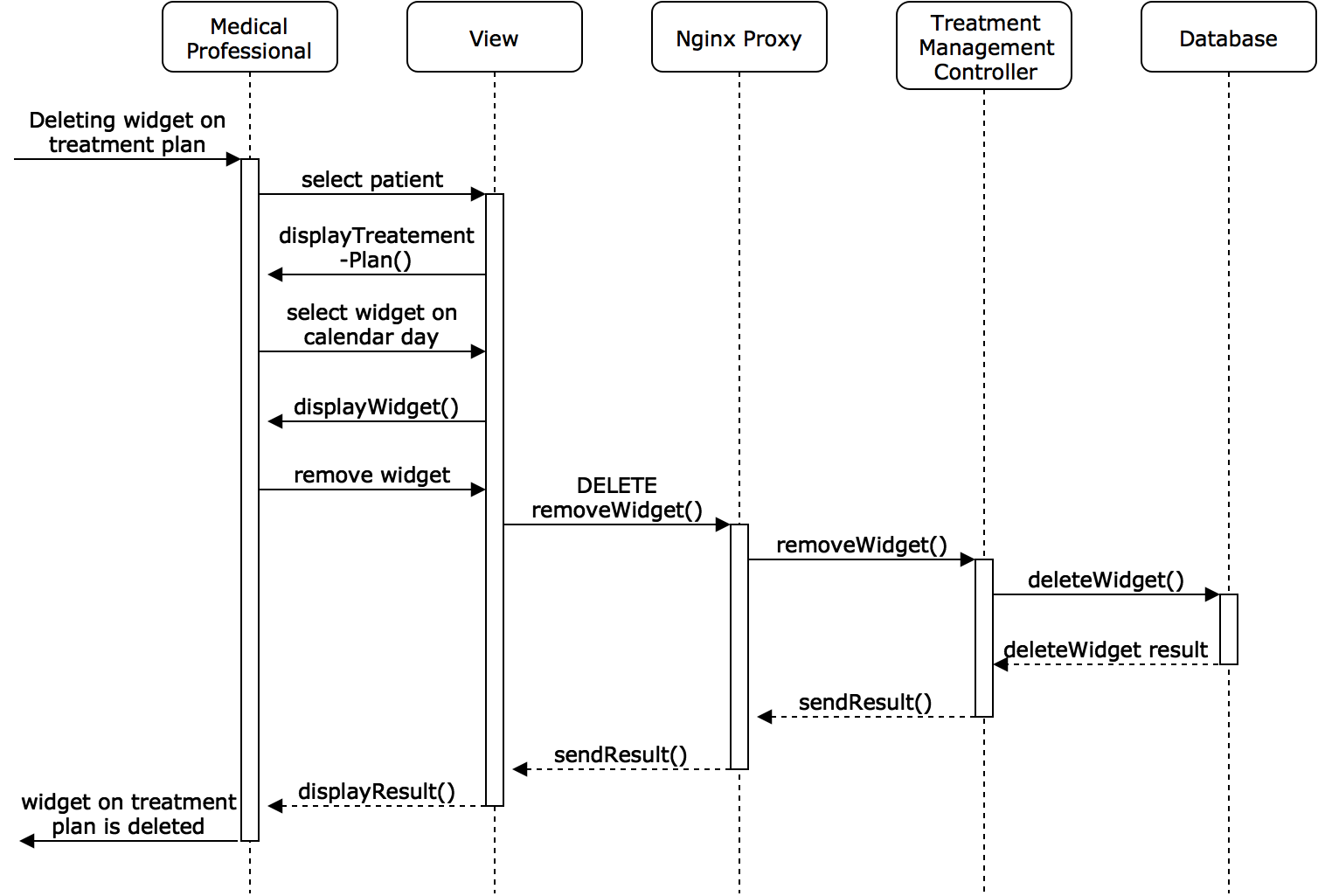
### Treatment Plan Revision

The following diagrams show the actions taken by the system when a medical professional either edits the treatment plan (first diagram) or deletes part of a treatment plan (second diagram). The precondition for both diagrams is that the medical professional is authenticated into the system and have had at least one patient registered under him/her. The patient(s) must have a treatment plan assigned to them previously.

*Editing widget*



*Deleting widget*



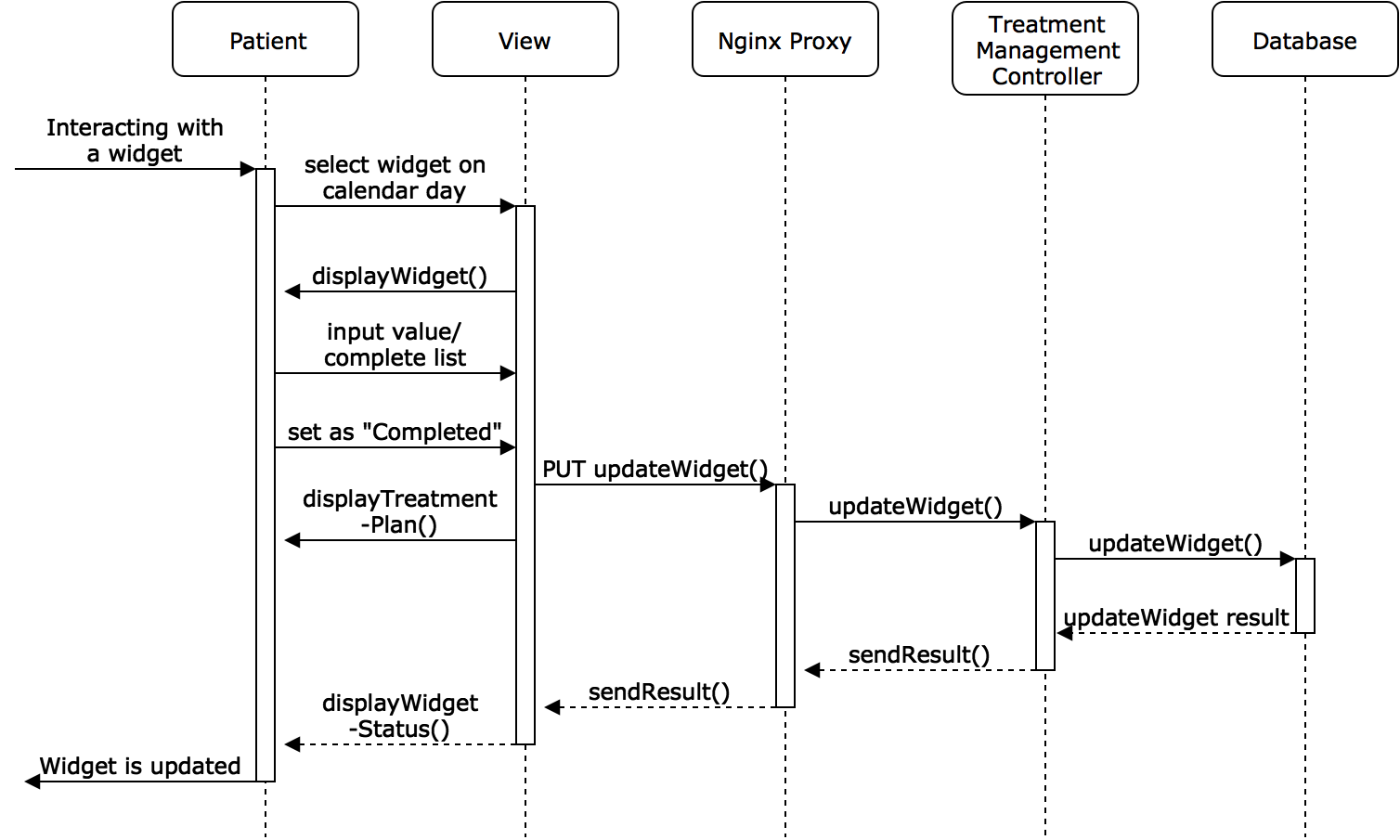
## 

### 

### 

### Treatment Plan Interaction

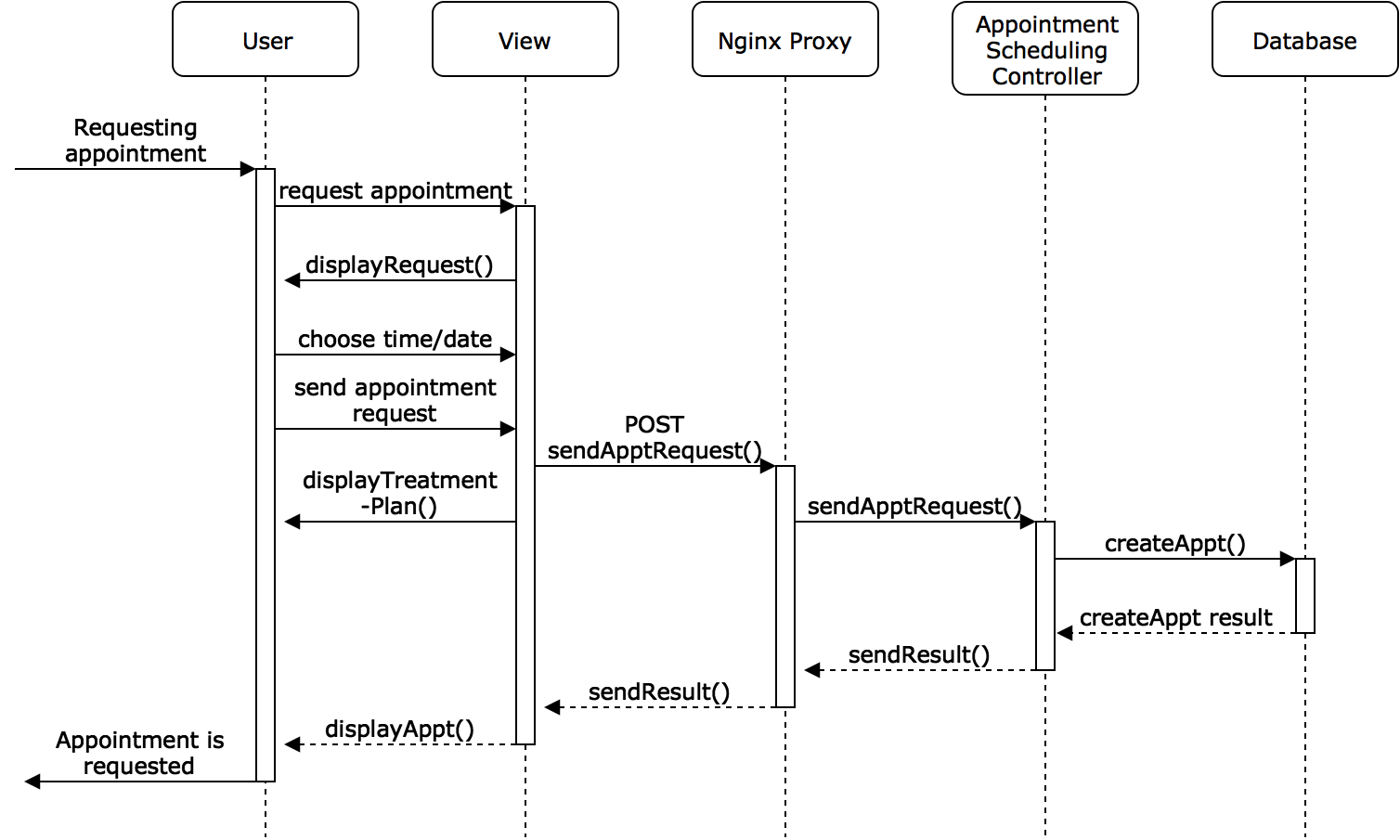
The following diagram depicts the flow of data and actions through the system when a patient interacts with their treatment plan. The precondition for this diagram is that the patient is authenticated into the system and has existing treatment plans assigned to him/her.

[](https://www.draw.io/?page=20&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

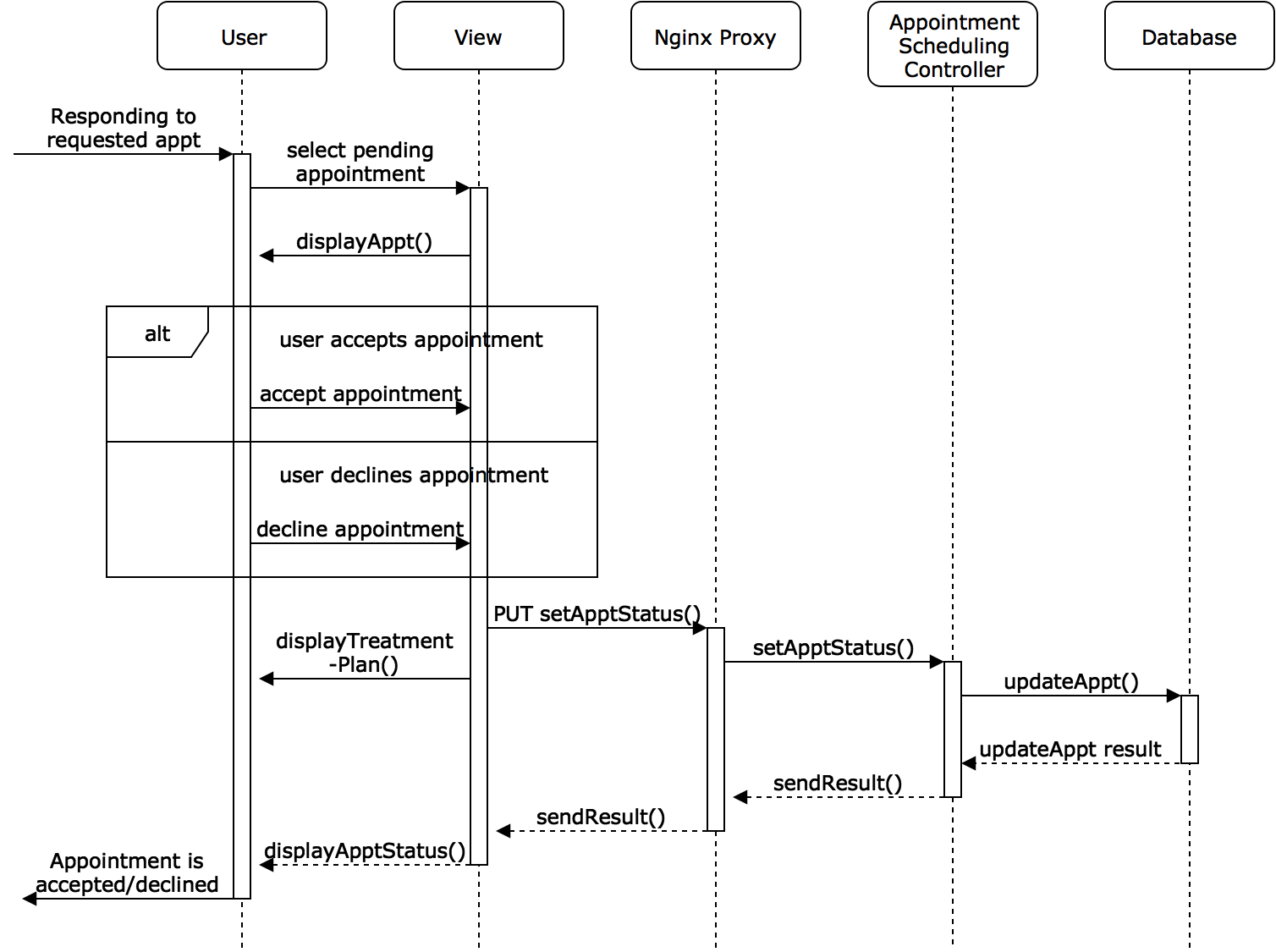
### Request Appointment

These diagrams show the flow of actions through the system when a user creates, accepts, or denies an appointment request, and the flow of actions when a user cancels an accepted appointment request. The precondition for all three diagrams is that the user is authenticated into the system and the system is active and deployed. For the second diagram, the precondition is that the user has a pending appointment request. For the third diagram, the precondition is that the user has an approved appointment existing in the system that has not passed yet.

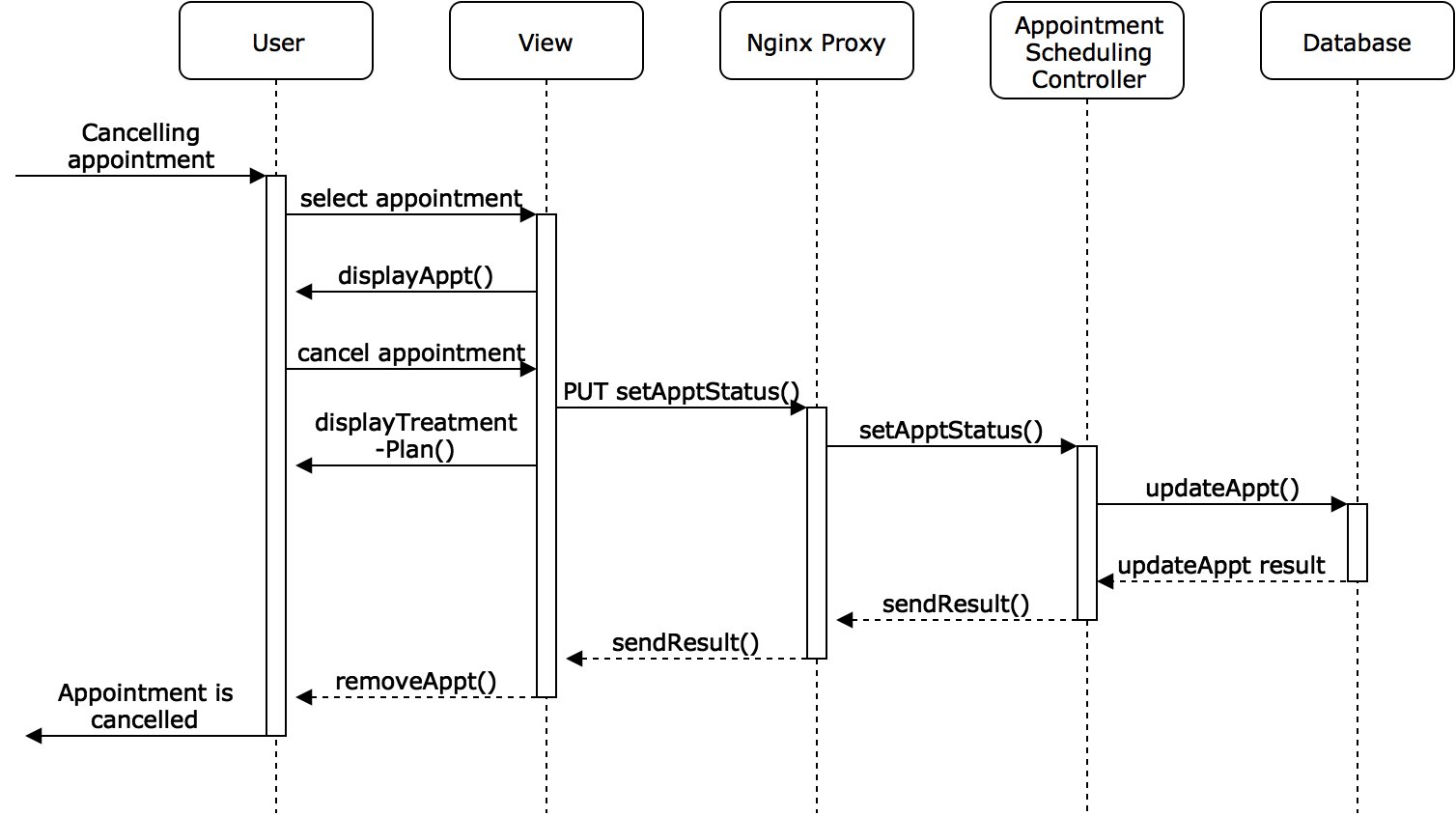
*Request appointment*

[**](https://www.draw.io/?page=21&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

*Respond to appointment request*

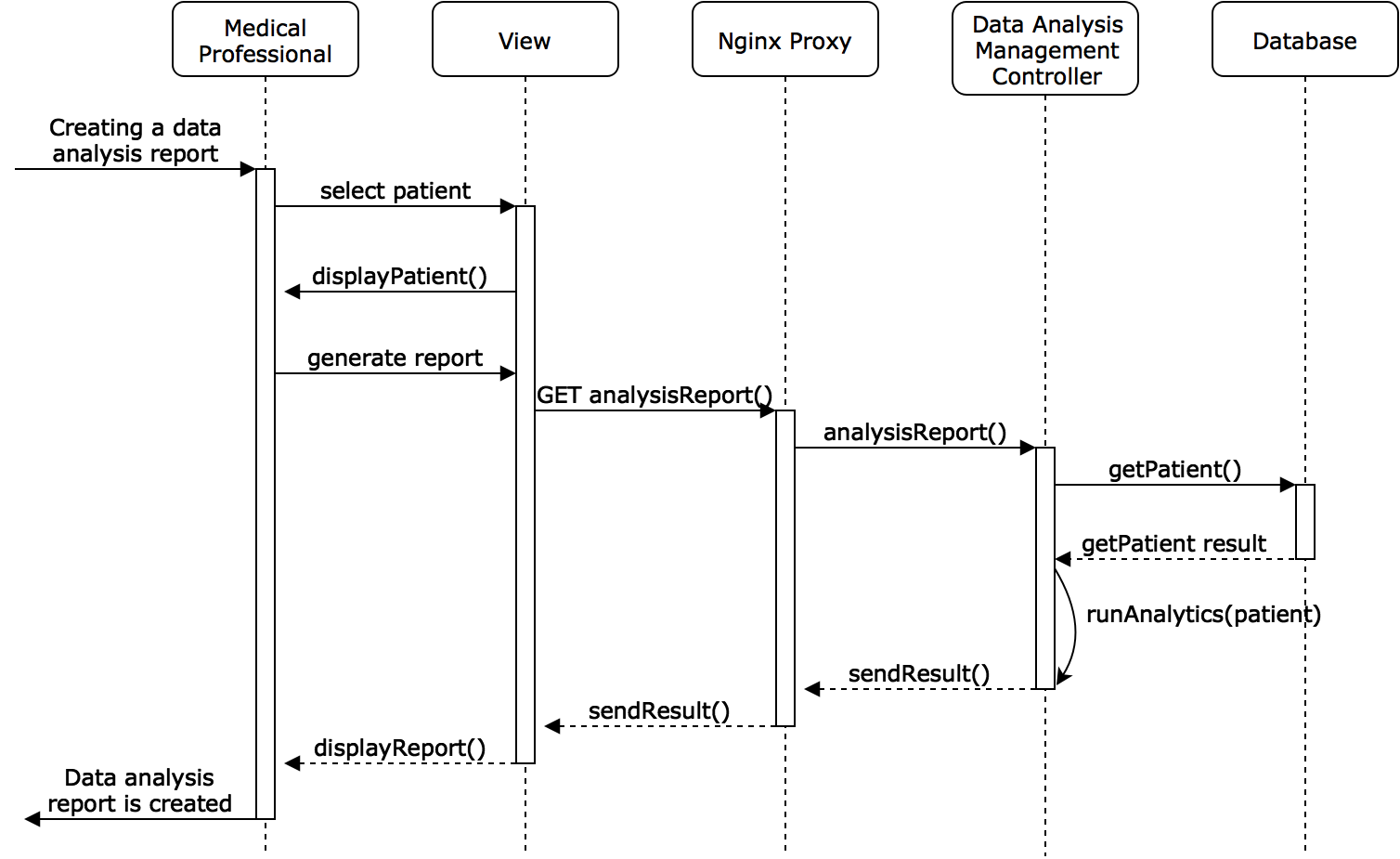
[**](https://www.draw.io/?page=22&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

*Cancel appointment*

[**](https://www.draw.io/?page=23&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

### Data Analysis Reports

This diagram shows how a medical professional can create a data analysis report of a patient. The precondition is that the medical professional is authenticated into the system and has at least one patient registered under him/her. The patient should also have previous treatment data. The medical professional first selects the patient they want to generate the report for and then the request for that patient’s data analysis report gets sent to the system’s proxy. The system’s proxy forwards the request to the Data Analysis Management Controller, which generates it based off of the patient’s aggregated data.

[](https://www.draw.io/?page=24&scale=2#G1A8exFzRgb5gGMDlpEivRA6Dlcmr0NqlL)

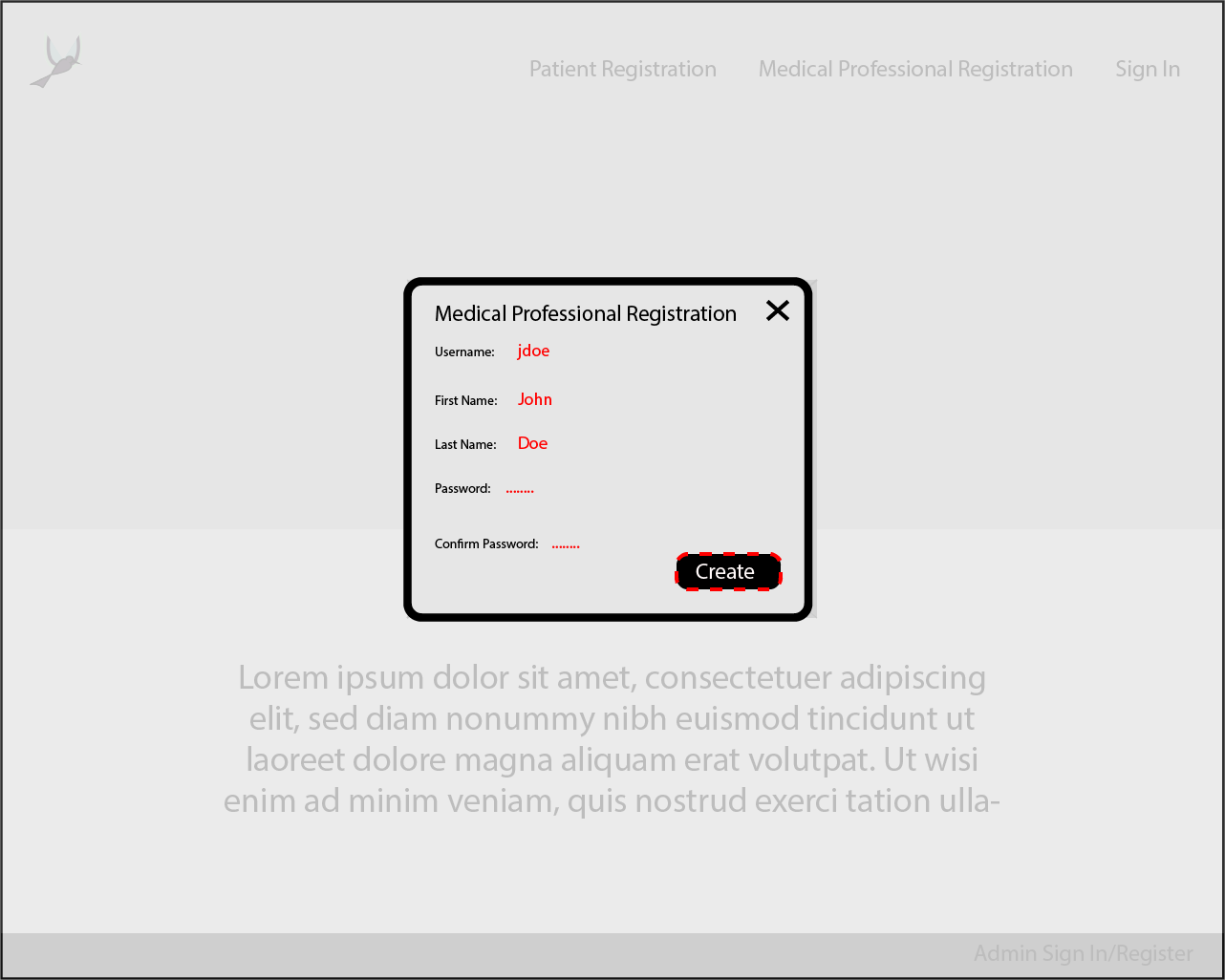
## [Wire Frame](#_f2hljk506fn8)

## 

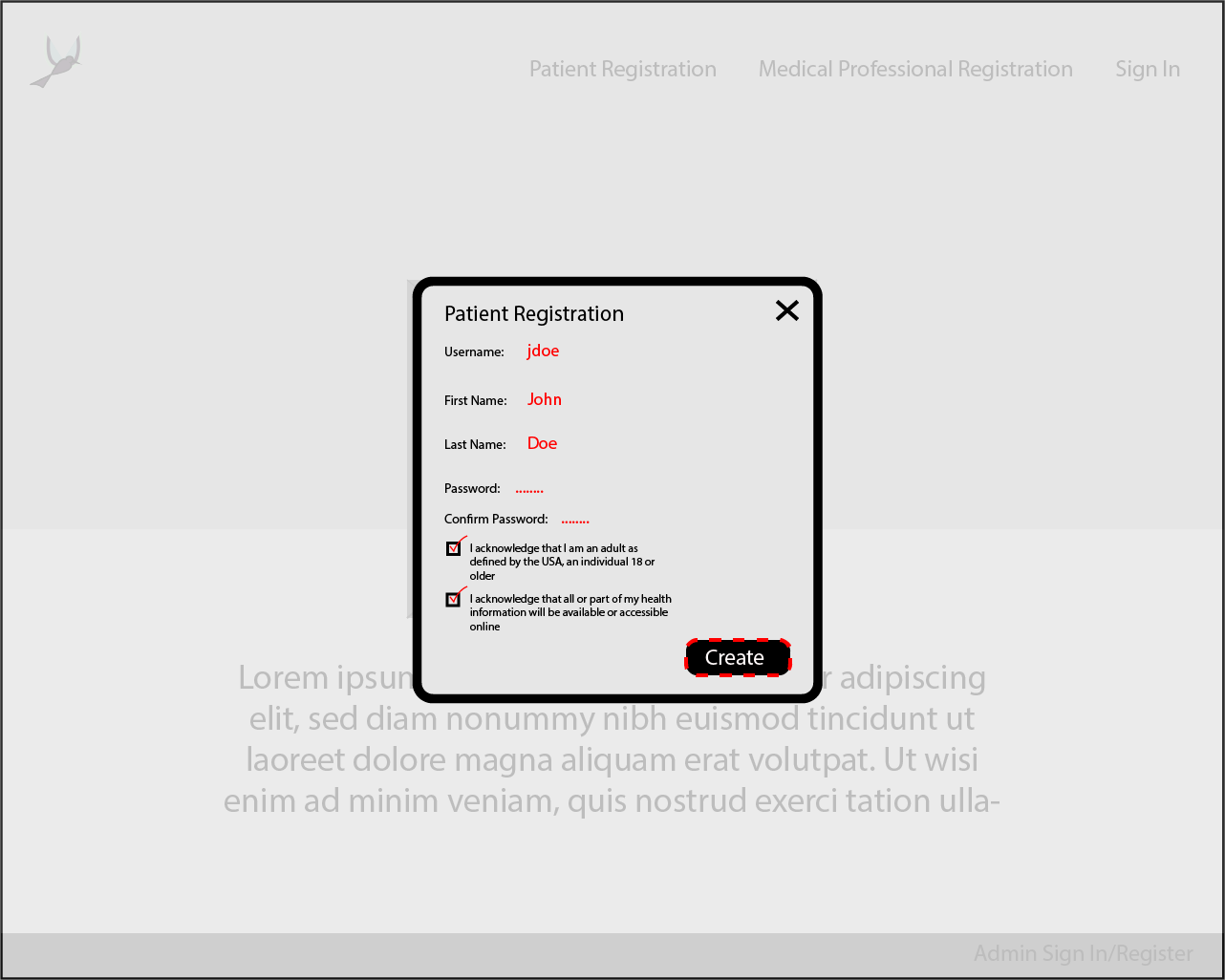
CareAway Homepage



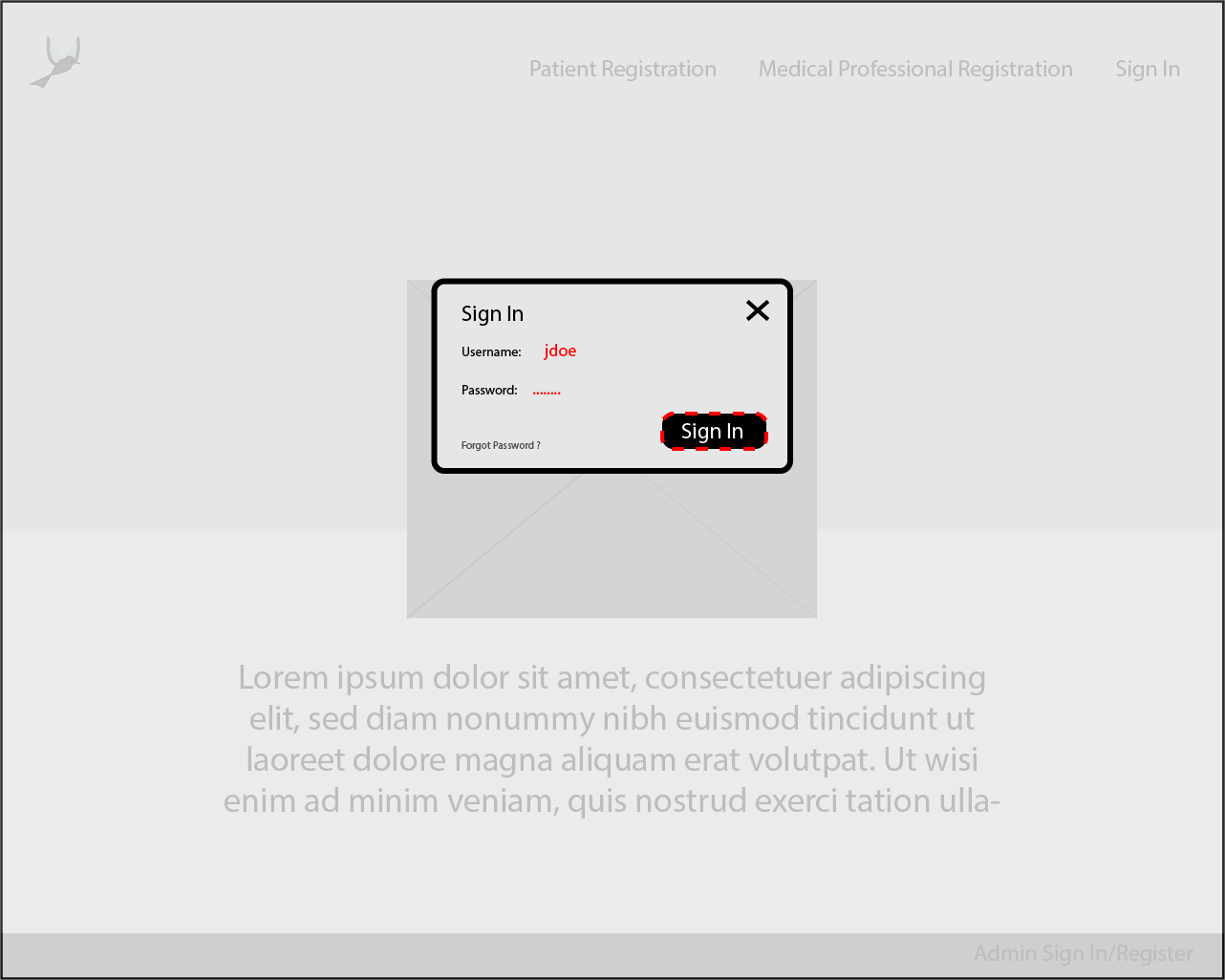
Medical Professional Registration View



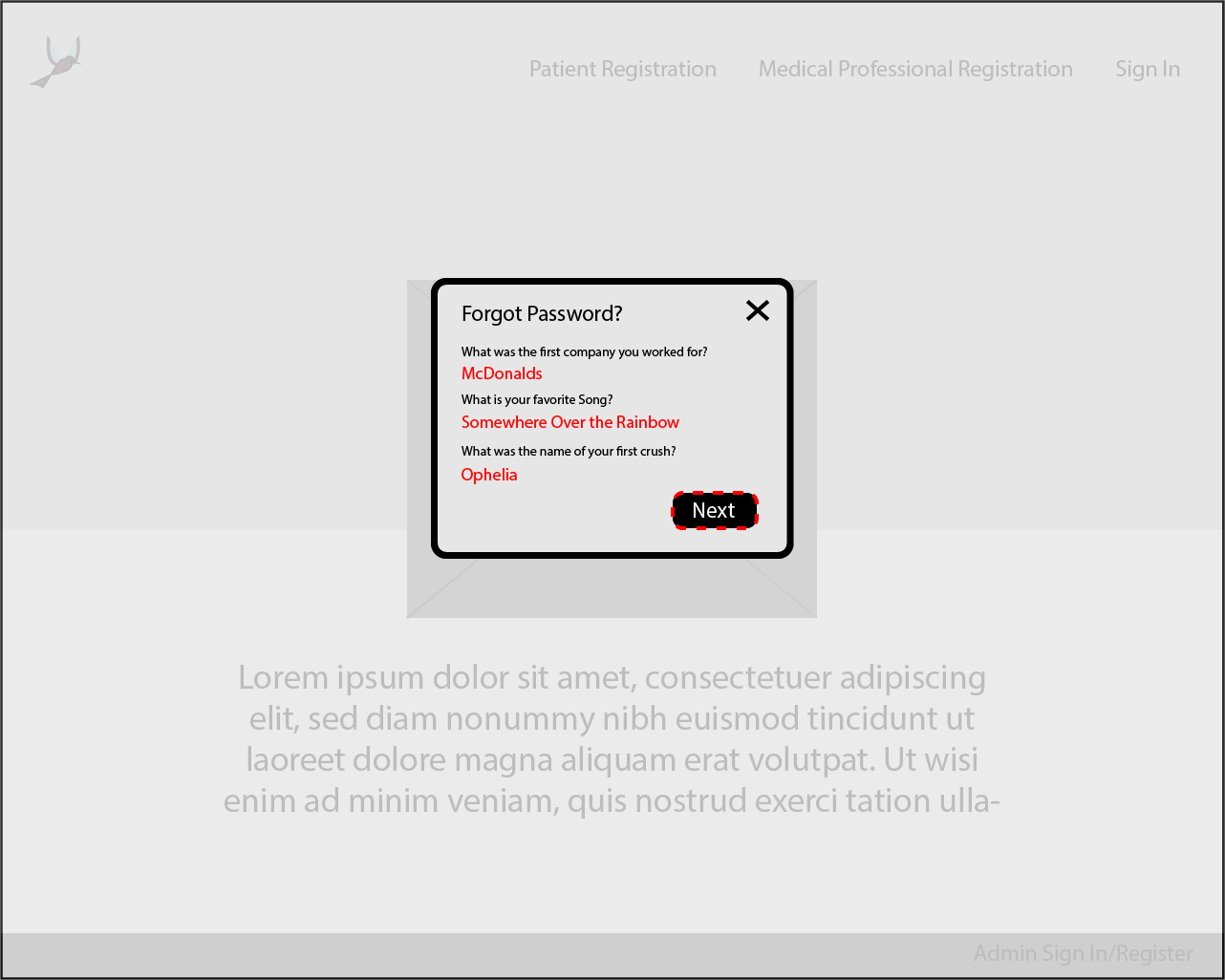
Patient Registration View

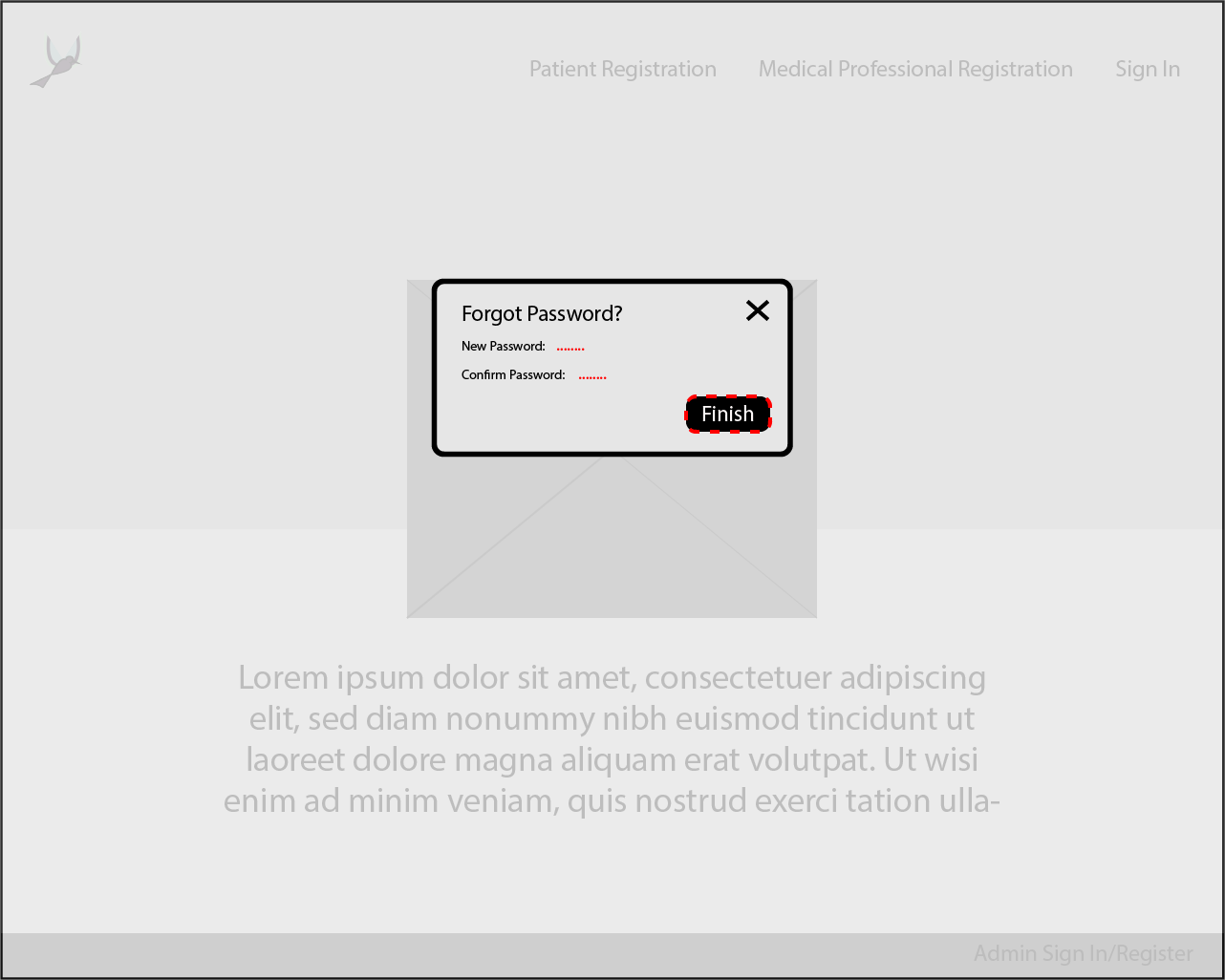


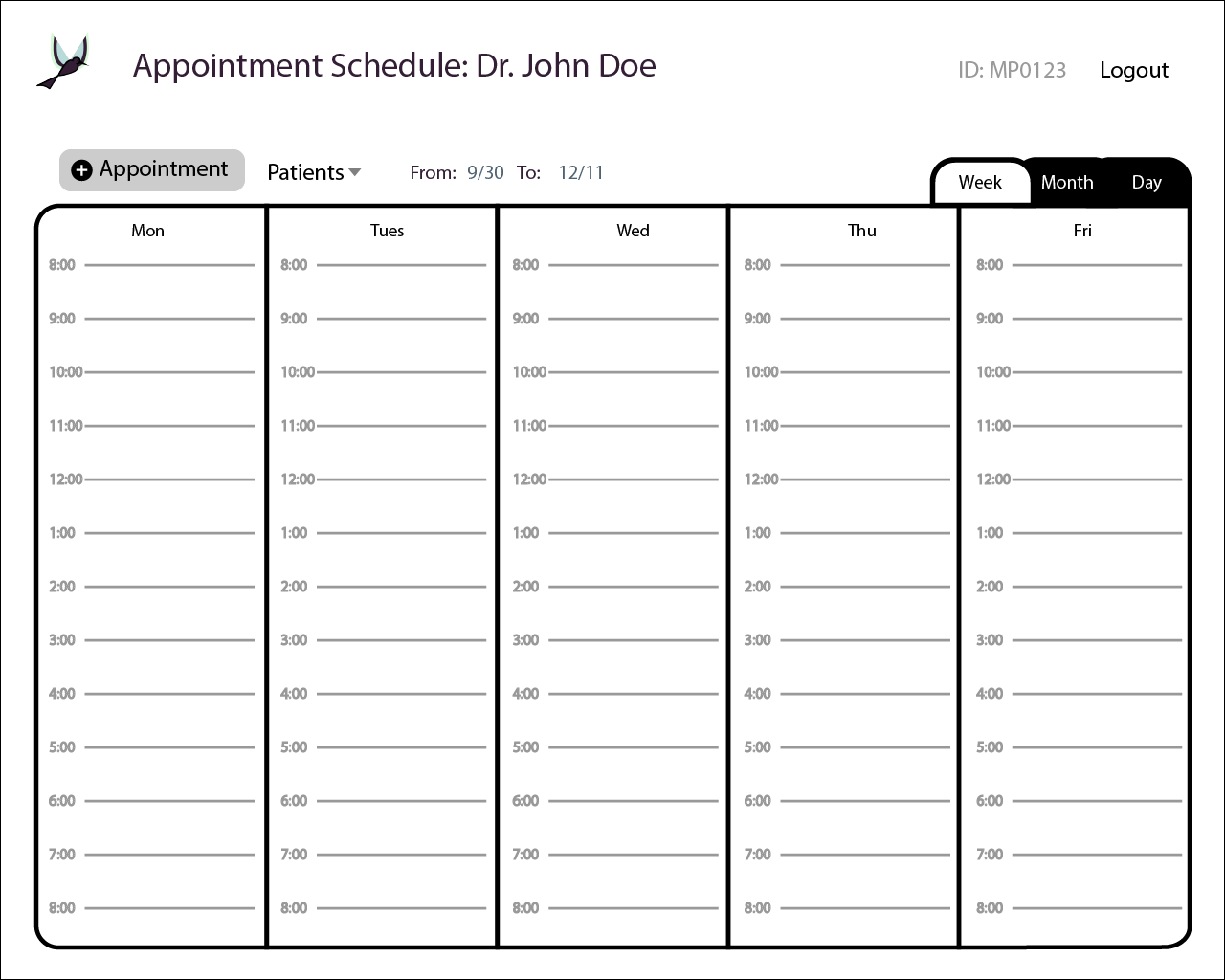
Authentication View

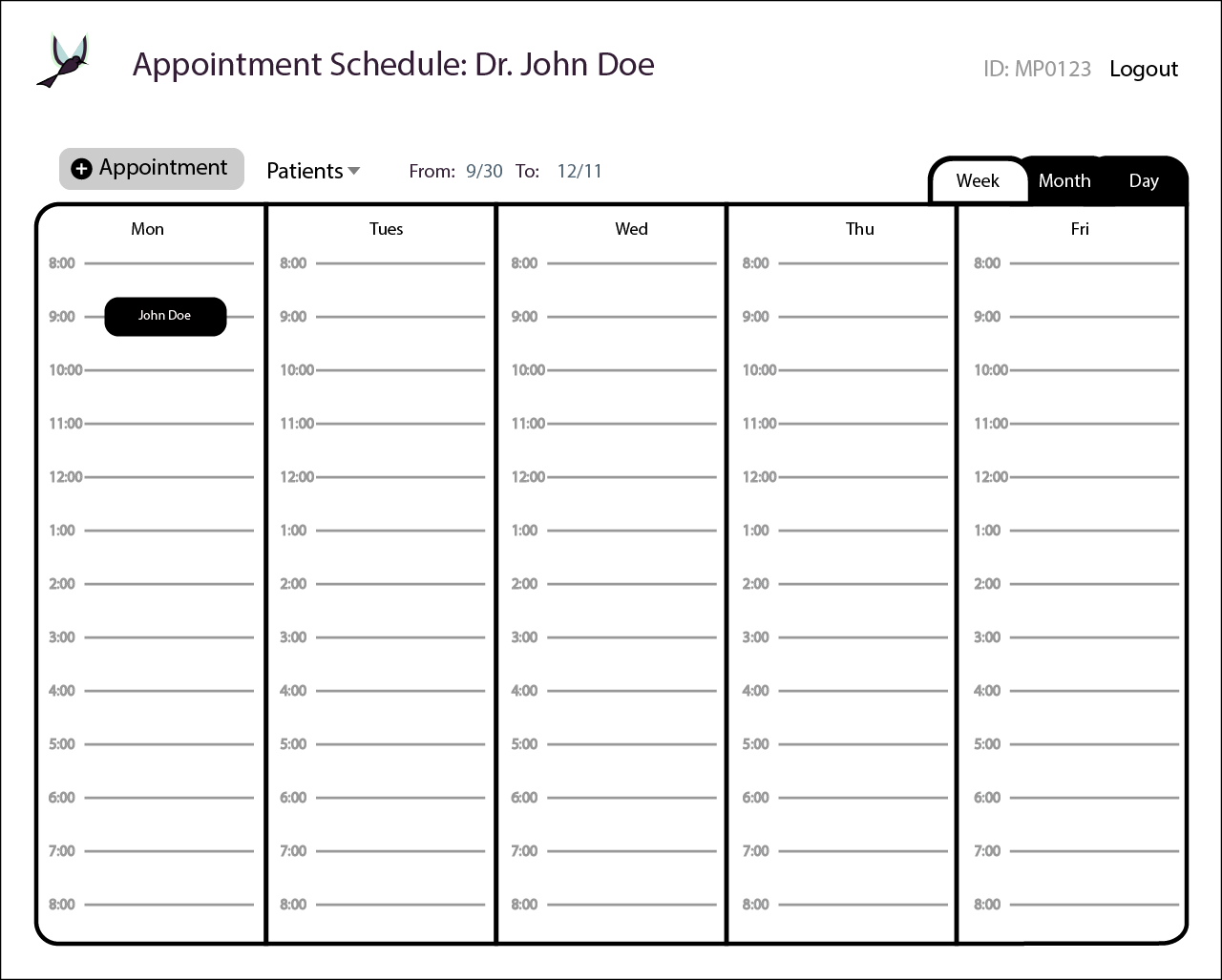


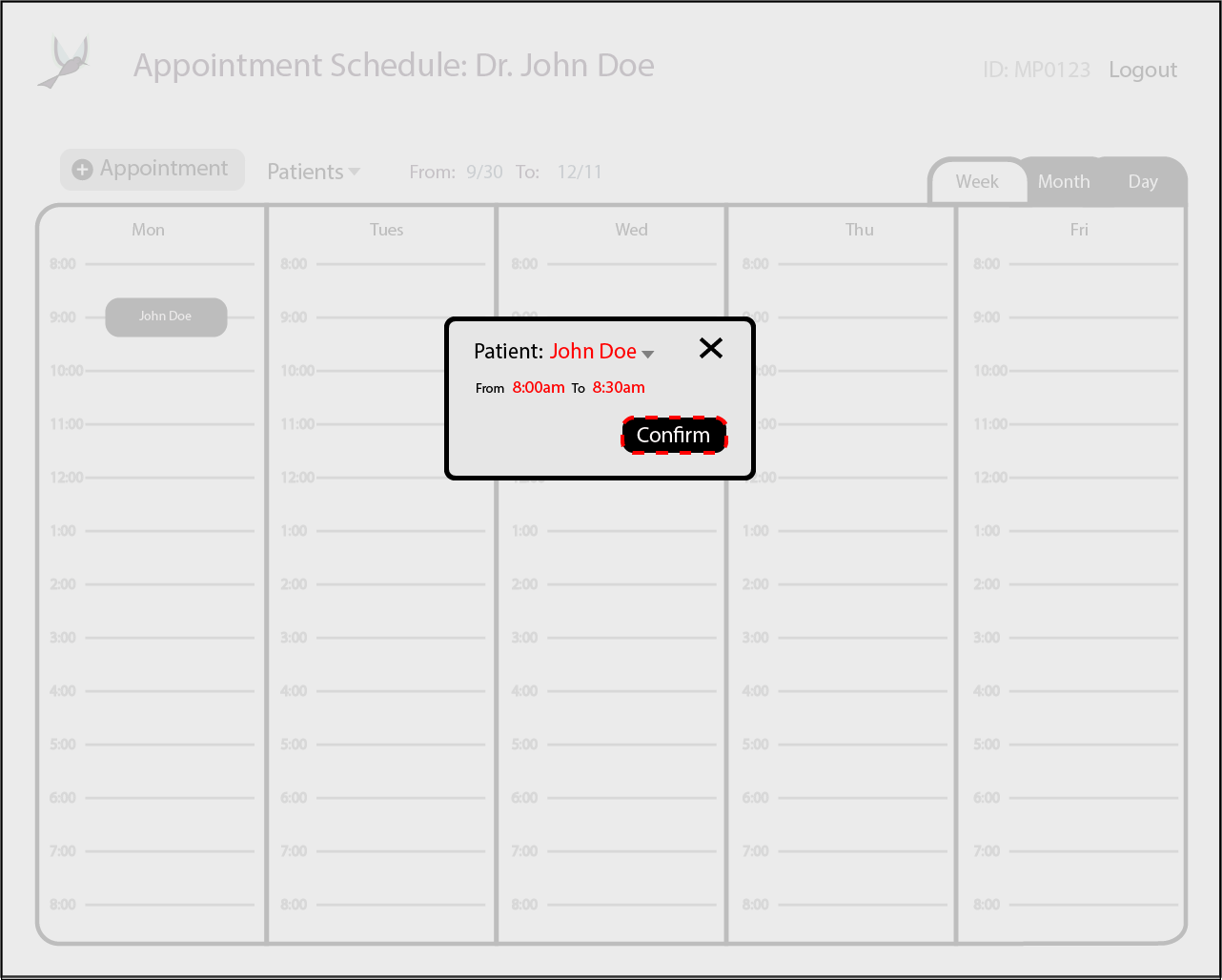
CareAway Homepage with Security Breach

Reset Credentials View - Three security questions

Reset Credentials View - Entering a new password

Medical Professional Account (Appointment) View (week)

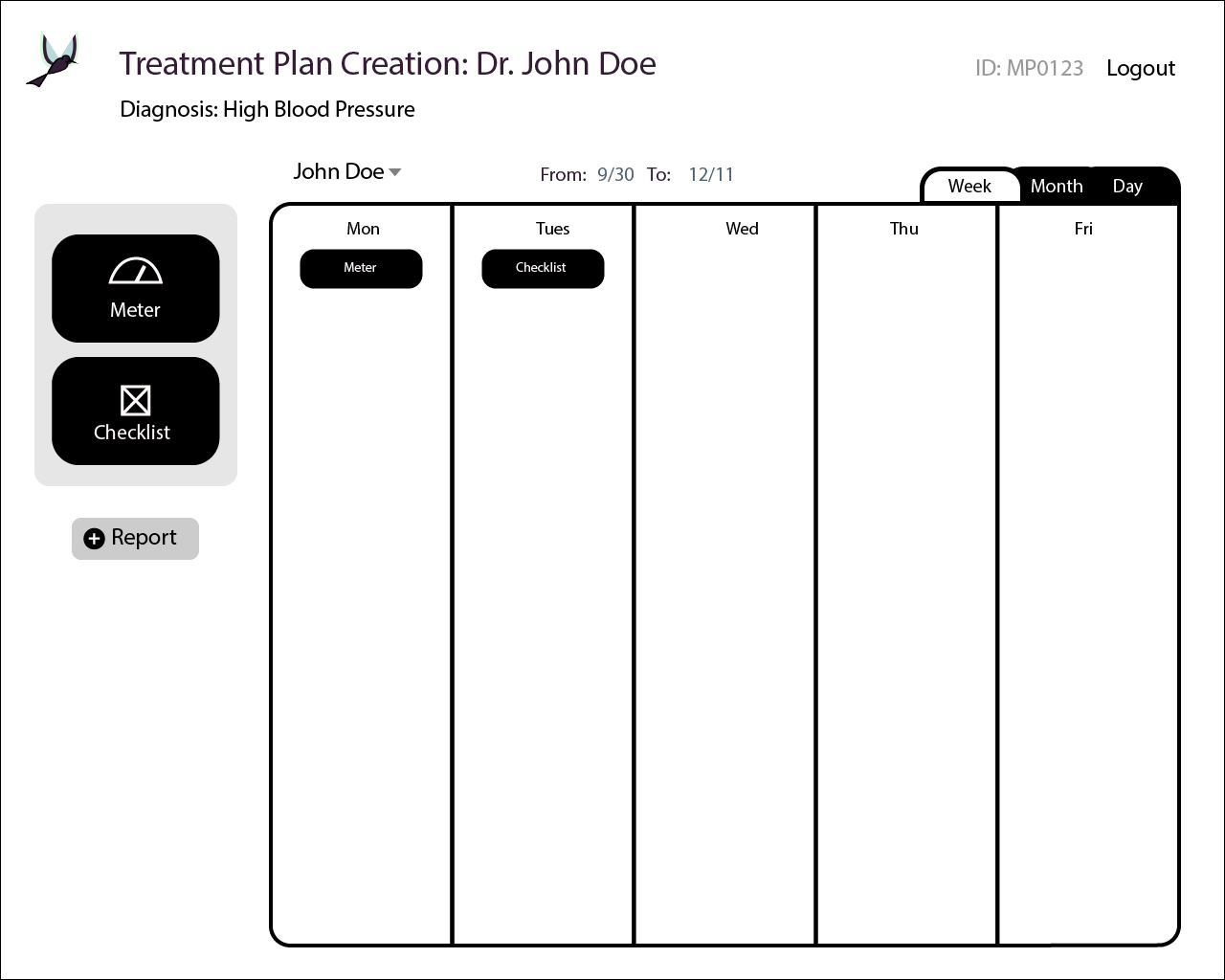
Medical Professional Account (Appointment) View - week, with one appointment request

Medical Professional Account (Appointment) View with Appointment Request confirmation page

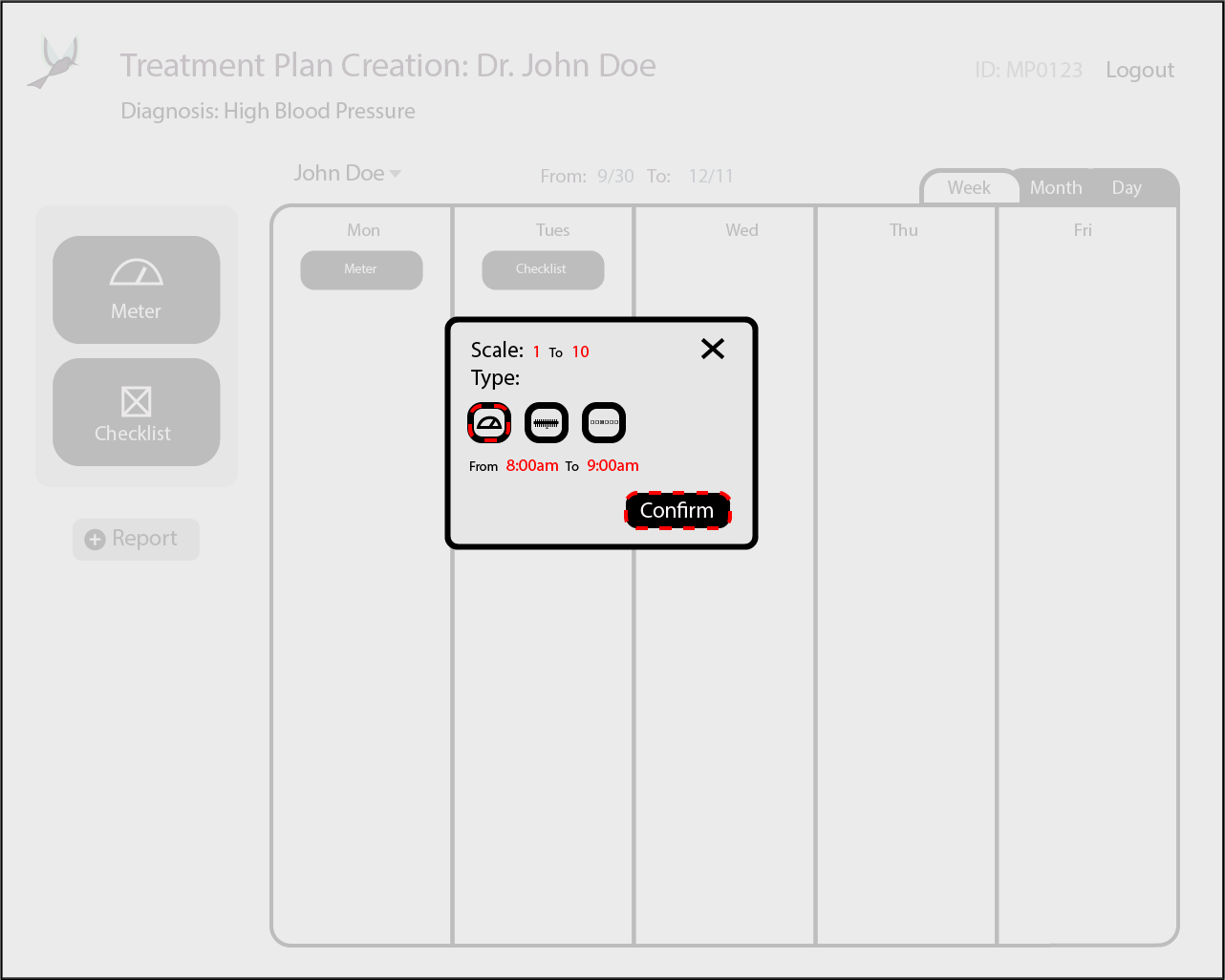
Medical Professional Account (Appointment) View (month)

Patient’s Treatment View (week) with no diagnosis set

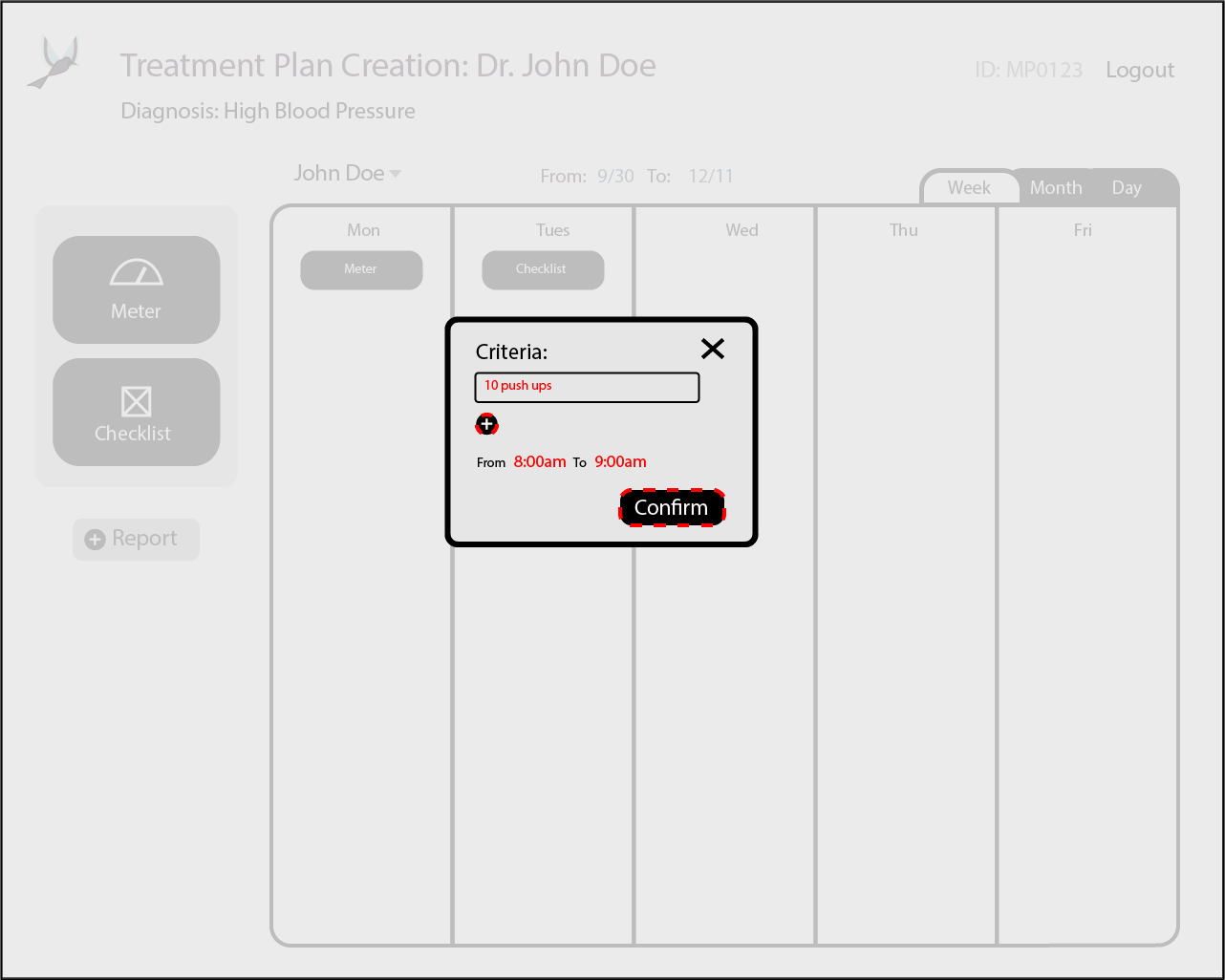
Patients’ Treatment View (week) with diagnosis set

Patients’ Treatment View (week) with two widgets set

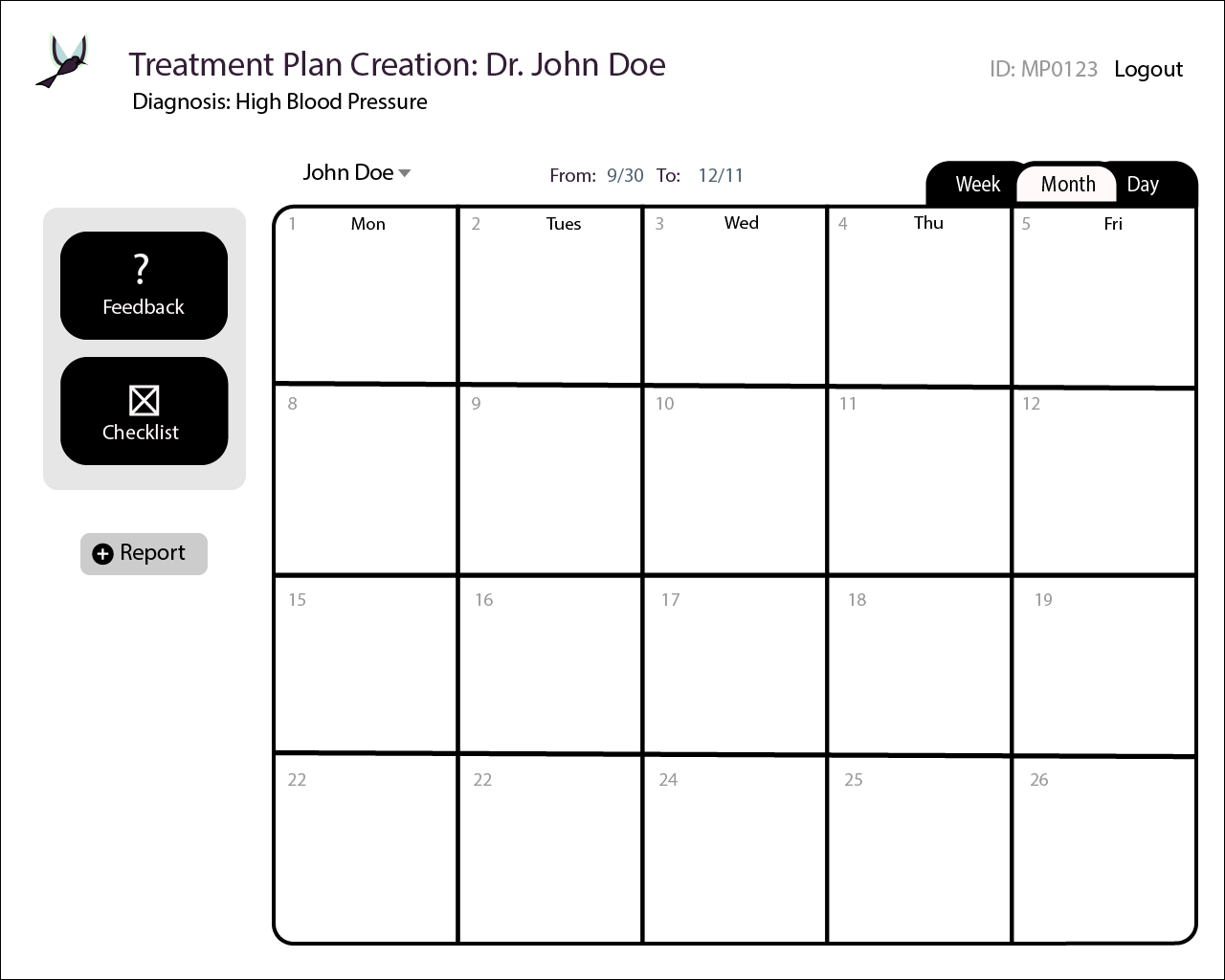
Medical Professional Meter Widget View



Medical Professional Checklist Widget View



Medical Professional Patients’ Treatment View (month)

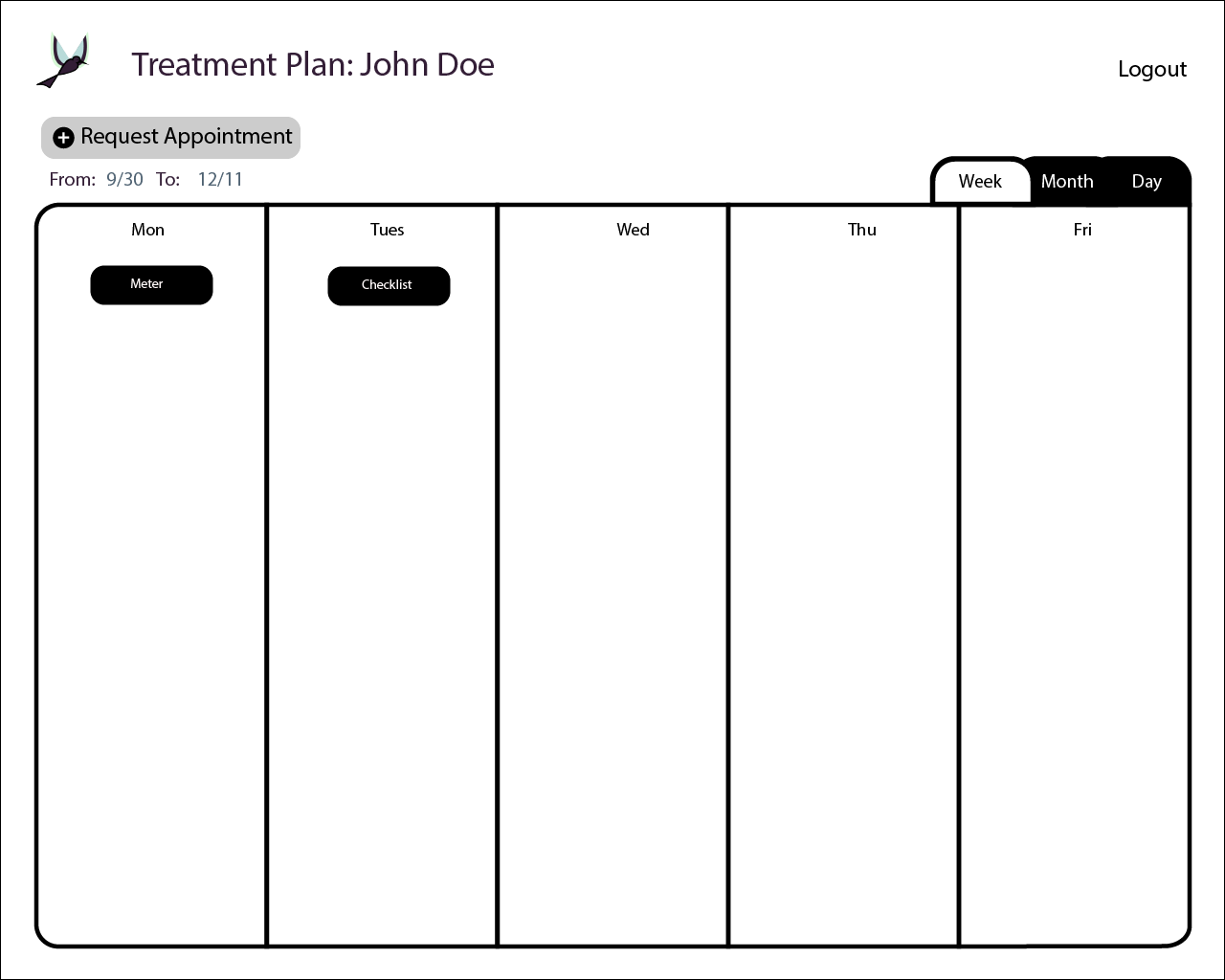


Data Analysis Report View



Patient Treatment Plan View (week)



Patient Treatment Plan With Widgets View (week)  


Patient Treatment Plan Meter Widget View



Patient Treatment Plan Checklist Widget View



Patient Treatment Plan Appointment View (Month)

Patient Treatment Plan Appointment View



System Admin Control Panel View

