Software Design Specification

For

NZPRU Data Registration system

Prepared by Kelvin David

University of Waikato

11/04/2020

Contents

1.Introduction

1.1 Purpose

The purpose of this document is to describe the design implementations of the NZRPU's Software Requirement Specifications document. The NZRPU's Software-intensive-project is used to track people's visits to chosen venues using a Data Registration System (DRS) using tablets

1.2 Scope

This document will describe the software implementation details of the NZRPU' DRS describing it's functions, designs and possible solutions for identified issues that may affect it during the development of this project. The components involved:

Venue Tablets: This component's functions will include how visitors will be able to login to the venue tablets and enter the required data of their visit to the venue.

Process System: This component's functions will include the how tablets will be monitored, how the data from the Venue Tablets are being processed into the log and how the relational database is updated by the log's information.

Relational Database: This component's function is to store all the information like currently registered tablets, venues and visitors and their bubbles. It will be kept up to date by periodic updates from the log

2.Extra/Specialized Requirements' Specification

2.1 Extra Requirements

Social Bubble:

Only one person of each Social Bubble (assuming there is two or more within one bubble) will have a registered account, everyone within that bubble will use the same registered account when signing into the tablets.

When signing into the tablets if you belong to a Social Bubble a companion screen will appear after login asking to select the name of the current visitor (selecting a name from the pre-set names that were associated to the account by the person who registered the bubble)

Social Bubble Notifications:

If a member of a Social Bubble is marked as a suspected infected visitor only the main member (the one whom registered the bubble) will be notified, instead of notifying the entire bubble. This considers the possibilities that not all persons within each bubble has a way of communication i.e. cell phone number or email.

The notification will include the bubble's name and the name of the person who logged into the venue that was affected. But it would be assumed that the main member should be able to notify the rest of their bubble

QR Bracelets:

Each venue will have a supply of QR bracelets that can be registered to each user (in this case one QR bracelet per bubble). This will act as an alternative/optional method of logging into the tablets.

Setup: Visitors will still need to register/sign in for the first time (or when they need to assign a new bracelet) and use the 'QR register' option to use the tablet camera to scan their new bracelet. This will associate that QR bracelet with their login.

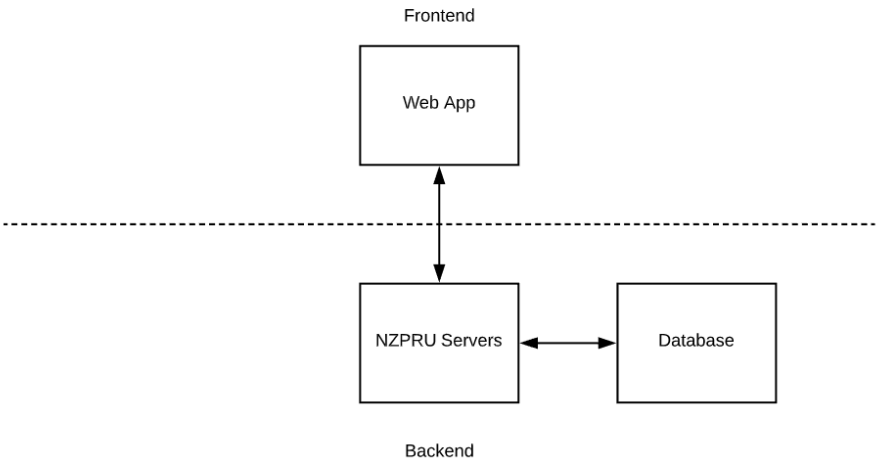
Use: Once their bracelet is associated with their account all they will need to do is select the 'QR login' option on the login page and scan their bracelet and the system will recognize their bracelet and log them in automatically. Though if the visitor still belongs to a bubble, they will need to still select their name afterwards.

The current known pricing for this method would be approx.:

- $0.18NZD (paper bracelets)
- $0.60NZD (silicon bracelets)

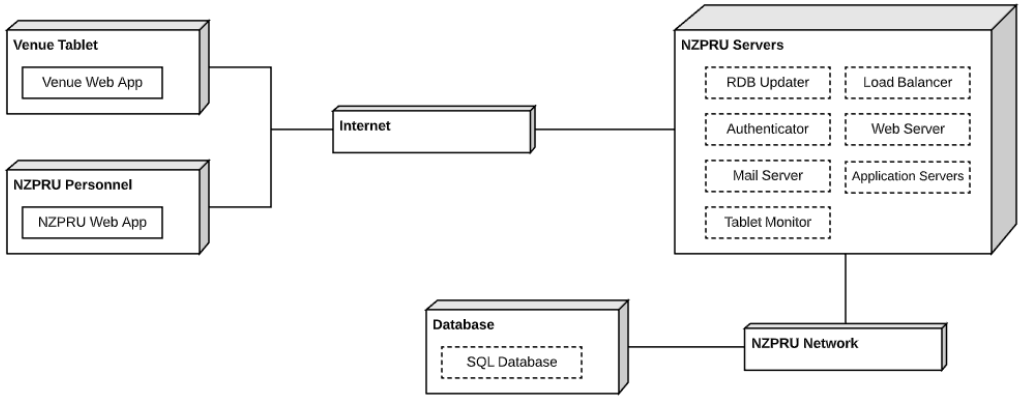
The QR bracelets can be kept by the visitors and used at any venue (decreasing in/out queues).

3.Software Architecture



The component diagram above shows a high-level summary of how the system is going to communicate. The front end holds the Web App component, this is responsible for providing an interface for users to communicate with the backend and its purpose to collection data at a high level and send it to the backend. The backend holds the NZPRU servers and Relational Database. The NZPRU server will be the only component in the backend that can communicate to the frontend. The database can only communicate with the server.

3.2 Process and Deployment



Deployment View

The deployment diagram above illustrates how the system is to be built. The relational database is only accessible through the NZPRU Network, this is to fulfil the issues the public has of storing data in the cloud. The NZPRU server is the only gateway out of the network. This also allows the database to be only accessed by authorized users as they must pass through the Authenticator to be able to query the database.

Component Processes

RDB Updater: This system is called the Relational Database Updater, it's responsible for scanning the record log database, extract the information, make corrections and update the structured record log stored in the Relational Database.

Load Balancer: This system is implemented to allow the servers to process more than one record log at a time instead of a queue. This should fulfill handling 10k log-record requests/min.

Authenticator: This system is to handle the permissions of different user roles to restrict roles from executing processes they aren't supposed to. This also will be used to handle visitor/NZPRU login requests.

Relational Database: This is a relational database that stores all the information and the relationship between them i.e. users, venues and tablets. It also holds a log of record logs.

Web Server: This will be responsible for establishing a connection between the frontend and the backend, delivering files/page content between them. For example, when a user requests a log-record the web server will handle the handle the request and store the record into the Record log then send back the page content to confirm it was completed.

Mail Server: This will be responsible for sending out automatically sending out mass alert emails to all the infected users. This is implemented so this can be done using the system rather than a standard email service like Gmail.

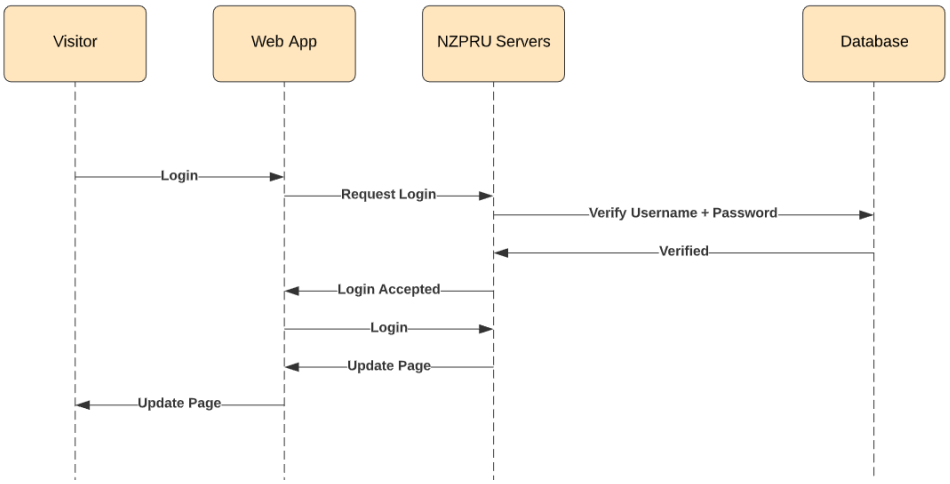
Application Servers: This will handle processes like services like handling the SQL queries and there will be multiple servers to be able to use the Load balancer to disperse the log-record request loads.

Tablet Monitor: This will periodically if all the registered tablets are alive. It send a automated message to all installers if not.

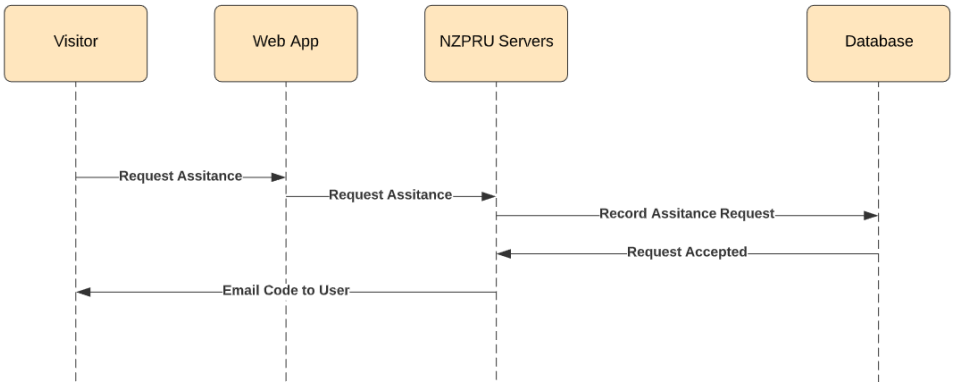
Common Interactions between components

Login

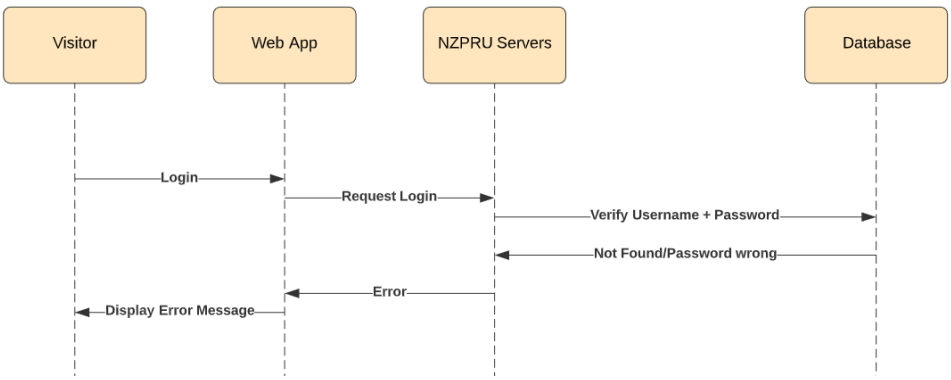
The user inputs a username + password to login. The system will check if its valid by matching the input values with an existing value in the database then redirects them into the home page of the user. If the input is invalid it will prompt the user to enter it again. If they forget their password, they can 'request assistance' which sends them a code to their registered email address which they can enter in the 'request assistance interface' to change their password. If they do not have an email registered in the system, they cannot continue.



Case: Failed Login



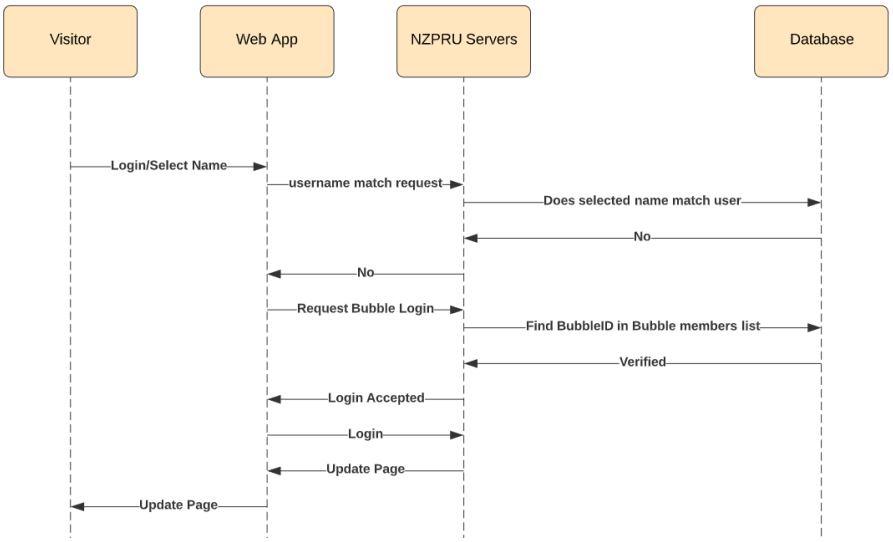
Case: Requesting Password Assistance



Login (Bubble Member)

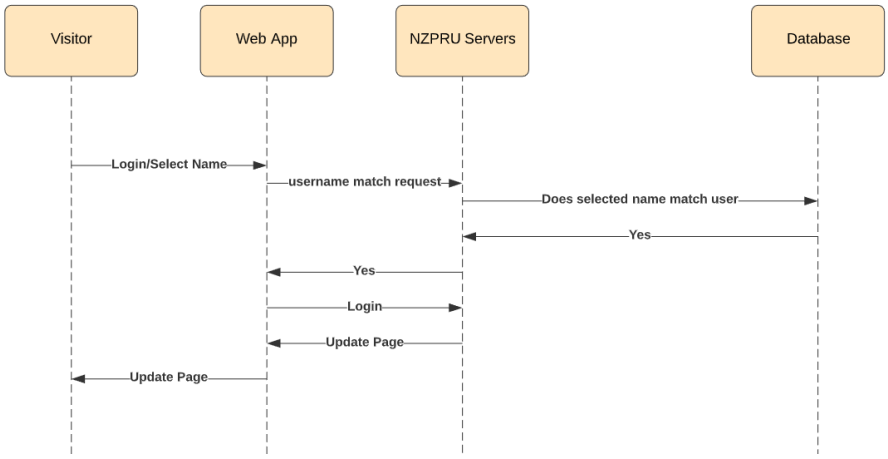
When a registered visitor has bubble of 2 or more, we have to consider who is logging in. This is done with a secondary interface asking to select the person's name. if the first and last name match what's recorded in the current username's record than the username is left unchanged. If not a BubbleID will be created to go in place of the username. The BubbleID consists of the first name of the current member + the username. So, when the system tries to verify the username + password, if cannot find the name on the Visitor list then it will check the bubble member list to find if the username matches a BubbleID. In record log BubbleID will be how it gets recorded.

Case: if name selected doesn't match username's first/last name (Find BubbleID)



Note: This only occurs if visitor has to select a name after login, which that interface is determined to be shown if the user has a bubble association.

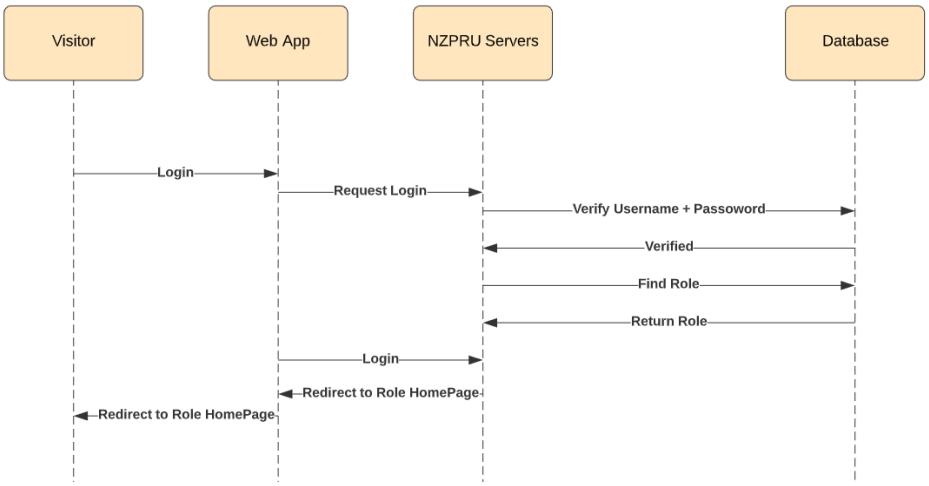
Case: if name selected matches the username's first/last name (Regular login)



These login sequences work after the original login sequences, so password/username authentication has already been processed.

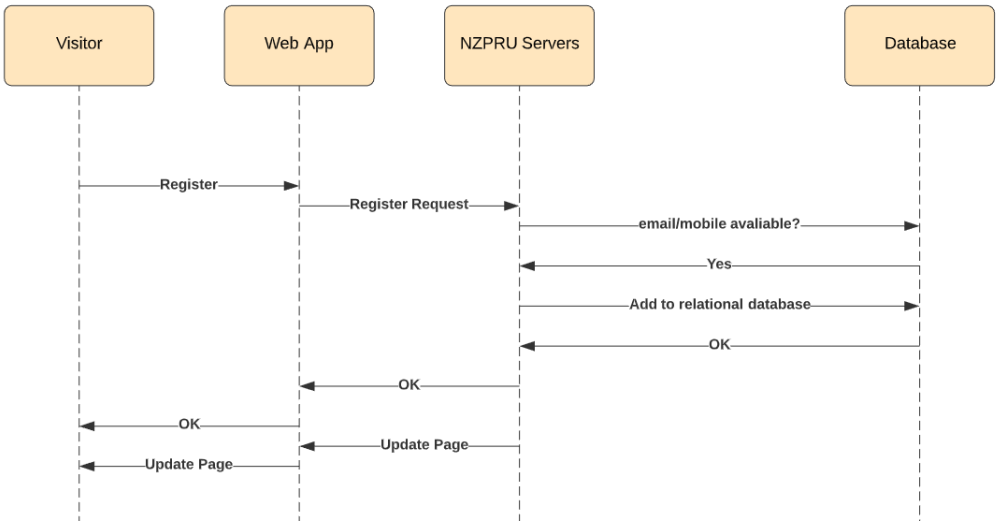
NZPRU Login

NZPRU personnel login differs slightly to the default. There are different types of NZPRU personnel, installer and admin. Each role has different permissions, so they need to be presented different home pages when they login. When a NZPRU logs in, the system will check their role and redirect them to a different port address where each role has its own homepage and access.

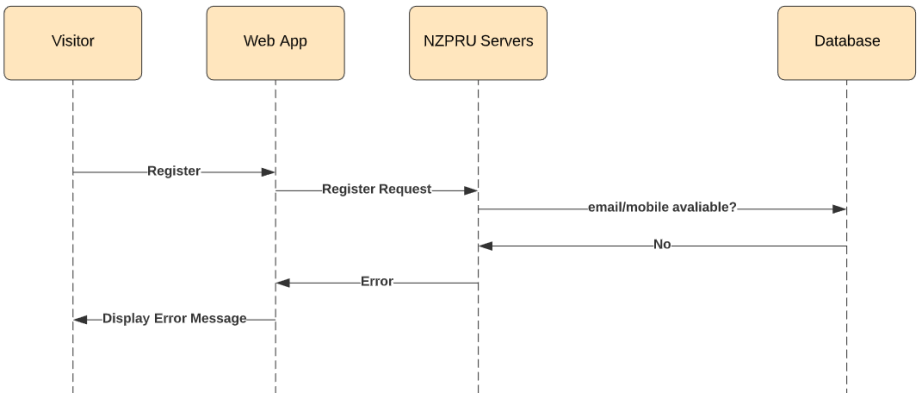


Visitor Registration

When Visitor first register their account, they will need to enter a whole host of information. When they send their register request the system will check if their mobile number or email hasn't already been registered. If it has it will display an error message respectively i.e. "That <email/mobile> is already registered". Otherwise it will add the newly registered user to the relational database as a visitor role. Then it will update the user's page and display a username by their first 2 letters of their first and last name and next available number.



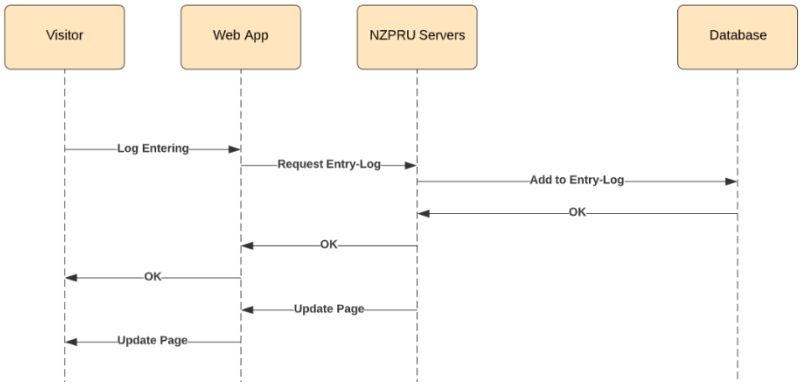
Case: Email/Mobile already registered?



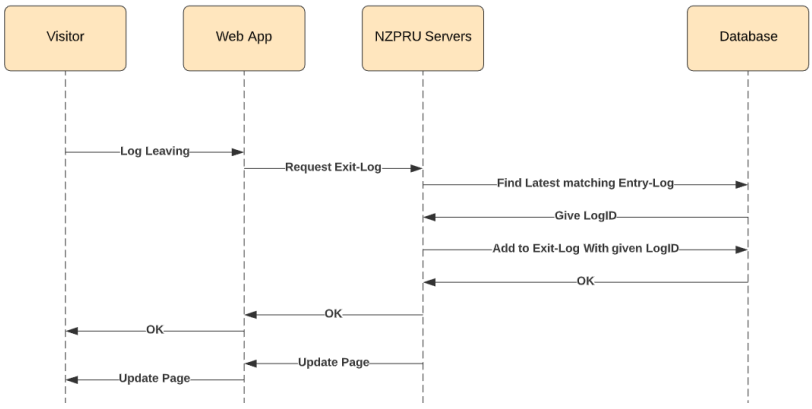
Requesting Record-Log

There are two log-record request types that can be sent to the NZPRU servers. Entering-log and Leaving-log. These logs will be stored in the database but not connected to the relational database. Whenever a visitor records an Entering-log it will assign it a key and add it the Entry log table. When they record Leaving-log it will check the Entry log table to find the latest entry where it matches the venue/user/date of the current Leaving-log being entered. Once found it will copy the same logID and add it to the exit log. If no matching Entry log can be found it will display an error message “No Entry Time Found”. This should prevent users from adding an Exit time before an Entry Time.

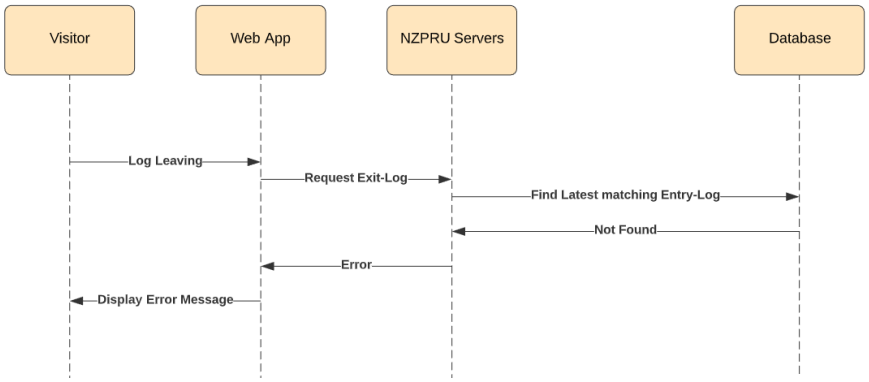
Case: Entering



Case: Leaving



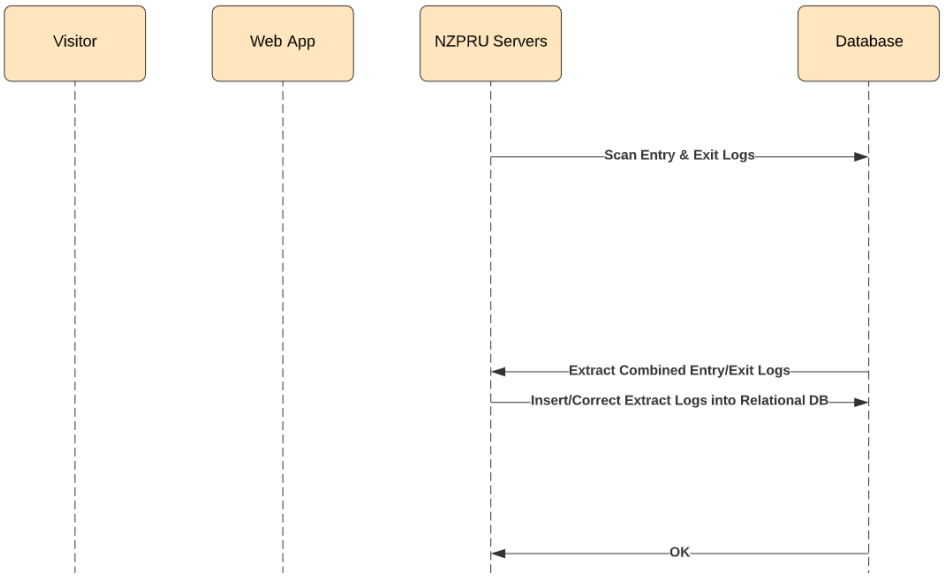
Case: Not Found?



Relational Database Updating

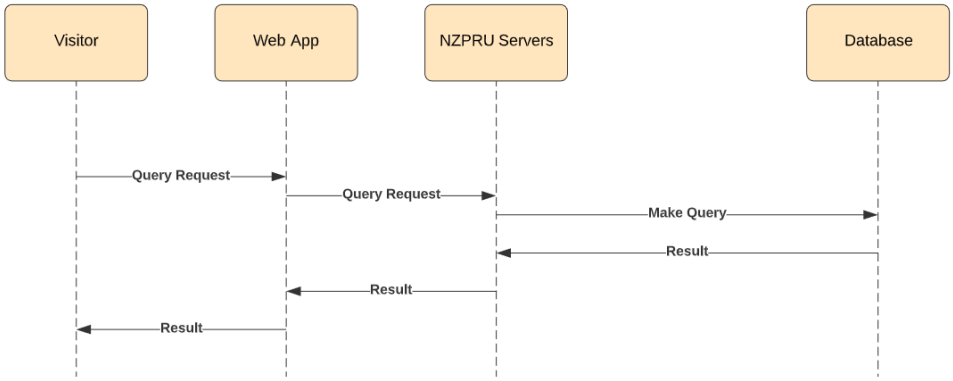
In a configurable period of time this system will scan the Entry and Exit log combining the entry and exit logs with the same logID. Once it has extracted all the matching entry and exit logs (combined into one record log) it will insert the data into the relational data base Record log table. It will be making automatic corrections whilst this happens. It will ignore any duplicates (Record logs with the same logID).

RDB updating sequence



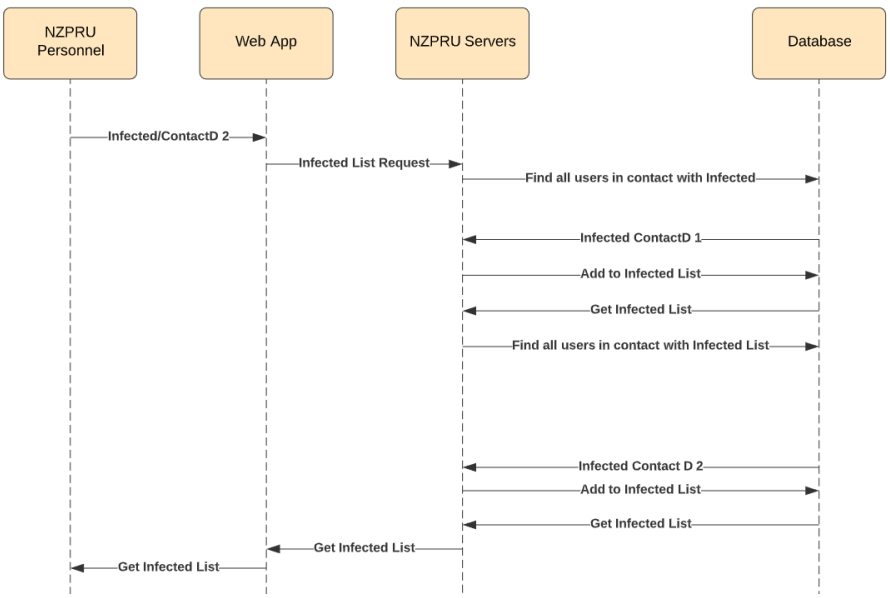
Making Queries

NZPRU personnel can make queries to the relational database to add/delete/edit/retrieve data. This will be done whenever they add/remove venues/tablets, manage user info and their roles and read any of the data. Whenever they execute these features, they will be sending a Query request via POST which will construct predefine queries. When the server receives it, the system will make the query and send the results back to the NZPRU Personnel.



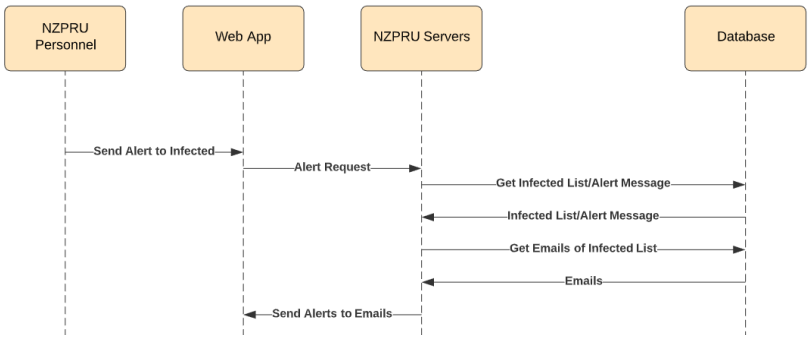
Constructing Infected List

When the NZPRU personnel construct an infected list they provide a username, contact depth. The system then scans the record log to find all the users that visited venues at the same time this user has visited. Then from there it will add said users to an infected list. Depending on the contact depth it will repeat this for the list of infected users. Example diagram contact depth 2. All duplicates (ID by username) are ignored. If a Bubble member of a user has been infected it will ID the bubble member by Bubble ID, though the in the record log the Bubble ID should still be classified as a username.



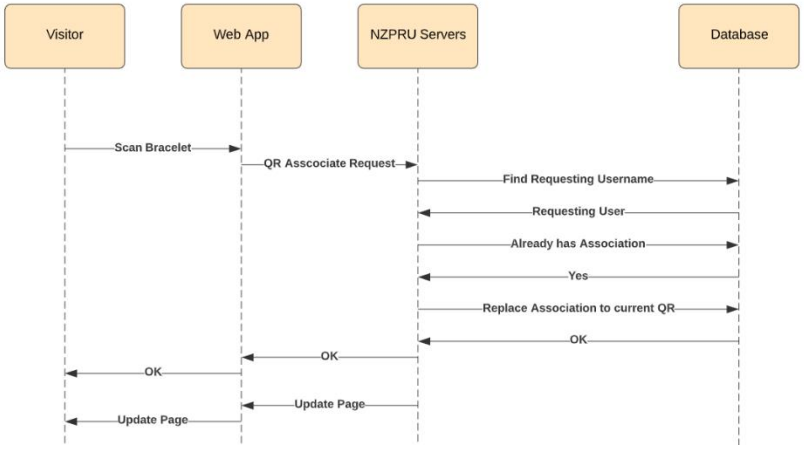
Sending Alert

When NZPRU personnel chooses to send an Alert with the generated list they will send Alert request (predefined query command). When the request is received the server will retrieve the infected list and the current alert message. It will then send the alert as an email to all infected (infected bubble members will be contacted through the contact information of their bubble owner.) with the NZPRU personnel's email as the sender. This will unused. SMTP to send the emails.

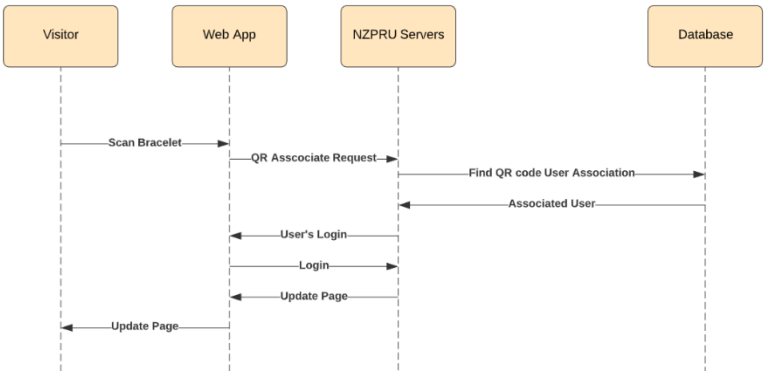


QR Bracelets

Users can use QR bracelets as a method of login after a bracelet is associated with their account. A QR bracelet becomes associated with the account. Whenever a visitor scans a bracelet the system will check if there is an association in the visitor's 'QR barcode' if there is none it will immediately associate the current bracelet. If there is it will remove the current association (making the past bracelet, clear to be associated with another account) and replace it with the current association. This is in consideration of visitor possibly losing their bracelets.



Case: Login



4.Software Architecture Design

4.1 Component Designs

Web App

The website will run on standard chrome on the tablets and will be static interfaces. The recommended languages to use are:

(HTML, CSS, JQuery/Javascript, PHP)

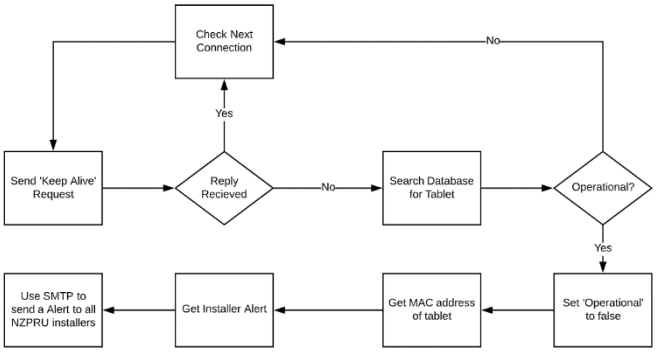
The method of updating the webpage will use the AJAX web development technique to update parts of a page rather than needing to resend/reload page by page.

The NZRPU web app and the Visitor web app will be hosted on the same domain but on different port, this is so we don't have to register a different domain name for the NZPRU web app.

NZPRU Server

Tablet Monitor

The system will the HTTP 'Keep Alive' method to check if the tablets are still functioning. In configurable transmissions intervals it will send a keyword "Keep-Alive" in the connection header to signal the connection to check if it is still alive, if a response is received do nothing. If there is no response the system will find the tablet in the relational database and mark, it's ''Operational' Boolean to false and send an Automated email to all installers to notify that a tablet is not functioning. This means that the connection between the tablets and the web server will be consistent. Though if the MAC address is no longer in the relational database the system will ignore the disconnection as this would be assumed it was unregistered.



Web Server:

The recommended web server to implement in the NZPRU servers is NGINX, an open source web server. According to documentation it can handle 10K+ simulations connections with a low memory footprint allowing to put less stress on the servers, it also handles heavy traffic better than it counterparts, Apache, which would do well for scalability depending the how many venues/tablets are added.

Secure Communication

SRS states that when log-records or queries are sent to the NZPRU servers they must be sent on secure channels. The standard practice for this is to use the HTTPS protocol which is the HTTP protocol implemented with TLS encryption.

This means the data being transferred from NZPRU web app or the Tablet Venue web app will be encrypted in a hash function that makes the information unreadable to third parties. It also requires Authentication using a public and private key. All senders will be given a public key (registered tablets and NZPRU personnel) and they will encrypt their requests and only to be decrypted by the private key on site.

Authenticator:

The main structure of the Authenticator will be using Identity Based Access Control (IBAC) access control model by Authenticating user logins by their username and password. Role Based Access Control (RBAC) will be used for NZPRU personnel to restrict access to features that certain personnel aren't authorized to use.

RBAC:

- Personnel
    - Can query the database to create an Infected Suspect List
    - Can query the database to alert Infected Suspect List
- Installer
    - Can add/remove venues
    - Can add/remove tablets
- Admin
    - Can manage users and user roles
    - Can configure log scan frequency
    - Can query the database
    - Can respond to assistance requests

IBAC: Users will be provided a username generated by the first two letters of their first/last name and will be asked to provide a password. If they provide an unregistered username or wrong password it will deny them access to the web app. All registered users will be stored in an Access Control List (ACL).

Mail Server:

SendPulse server is another good choice for Mail Server and can be implemented into the NZPRU server using PHP or java. There is no charge to use this implementation and can send 12,000 messages per month for free. It is also using Simple Mail Transfer Protocol (STMP) which is standard in mail servers.

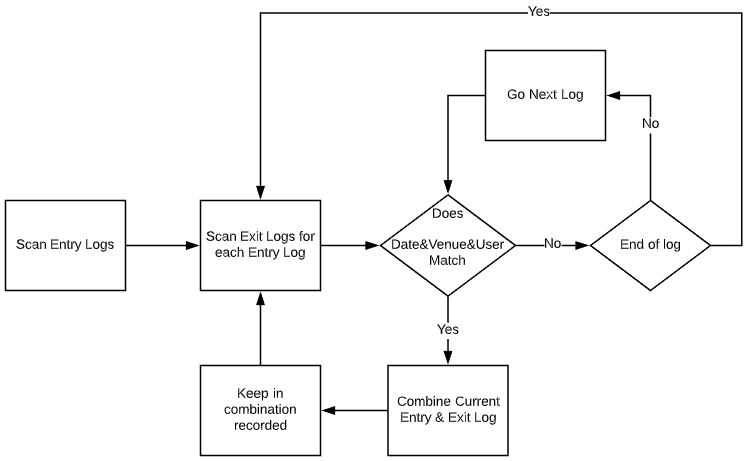
Load Balancer:

There are two Load Balancers that will used for the NZPRU system. Both will use Round Robbin.

Round Robbin: This will be the main algorithm used for the visitors, a set number of application servers will be available to this load balancer. This system will rely on a rotation sequence that delegates traffic to the first available server and bumps servers that have finished to the bottom.

The Visitor load balancer will only accept traffic from port 80 (main web app) and the NZPRU load balancer will only accept traffic from a different port (NZPRU web app).

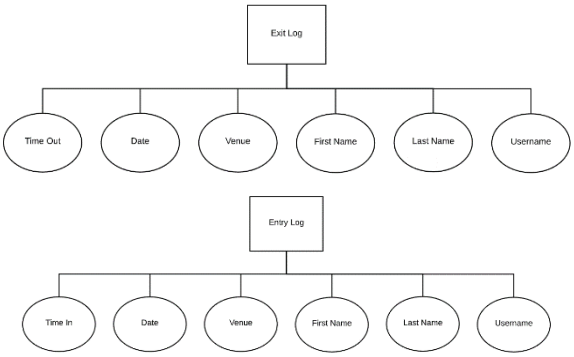
Relational Database Updater



This process above shows the basic algorithm of how the RDB Updater will function when it scans the logs. It will first retrieve the entry logs, match them with their exit logs by comparing the date&venue&user. It will keep record of all the combined record logs and input them into the Relational Database's Record Log.

Database

Log Database



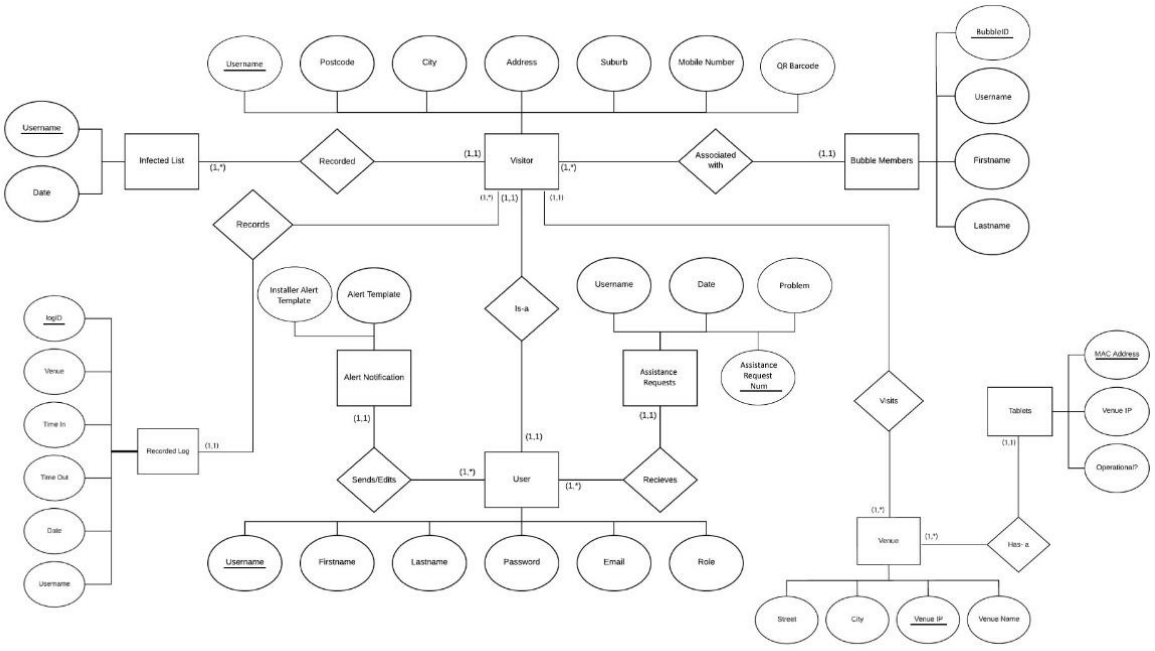
Log Database ER diagram

This ER diagram above represent the databases that are not apart of the relational database. These databases hold the log-record inputs of the visitors, separating the Entry log and the Exit Log. These logs are kept separate from the main Record Logs in the relational database as they are always being constantly added, So these logs are periodically added to the relational database by adding the Entry/Exit logs with same ID together and storing it into the relational database's Record logs.

Entry Log: This is the table that holds all the log-records of when users input into 'Record Entering' feature. It holds the date and venue the request was sent the shared username of the visitor bubble the first and last name of the current member and the time they entered the venue.

Exit Log: This is the table that holds all the log-records of when users input into 'Record Leaving' feature. It holds the date and venue the request was sent the shared username of the visitor bubble the first and last name of the current member and the time they left the venue.

Relational Database ER Diagram



The diagram above represents the structure and relationships of the tables in the relational database.

User: This is the general base for all users using the system for a user to be registered they must fulfill all properties under the user table. The role property will be used to distinguish the different user permissions of NZPRU personnel, admin, installer and a visitor. The property will be regularly look at to know what page to display on the frontend interface.

Visitor: Visitor is a User type create through the venue tablets, while requiring all of properties under user it will also require extra properties to be fully registered. This data will be regularly used to get visitor info when they make a log-record.

Bubble Members: This table will consist of visitors who are associated with a registered visitor. They will only need to provide their First/Last name and will be associated to username (the visitor who's bubble they belong to). If they need to be notified, they will be notified using the information of the Associated Username.

Venue: This table consists of the registered venues. This table is associated with Visitors as when they visit these venues the visitor and venue are recorded together to the record log. This is identified by venueIP due to the possibilities that venues may have the same name.

Tablets: This table consists of all the registered tablets. Tablets are identified by their MAC address and are associated with Venues by their venueIP.

Record Log: This table consist of updated completed record logs extracted from the log database. It holds all the complete record logs to be used later to create the infected visitor list by its date, time in and time out.

Infected List: This table will hold lists created by the infected list generator system whenever a personnel request it. It's associated with visitors as the personnel will input a username and contact depth which will use the record logs associated with that user to create the infected list.

Alert Notifications: This table holds the templates of the automated messages for infected alerts and installer alerts. They can be edited any time by NZPRU admins.

Assistance Request: This table holds all the assistance requests of forgotten username or email. NZPRU admins will accept and handle these requests and remove them from the database once fulfilled.

The Databases will be running MySQL and will use SQL commands to query the database. Common commands include:

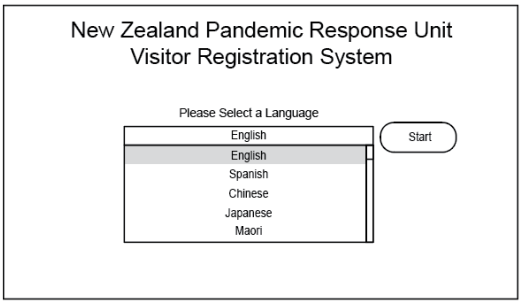
- CREATE: create a table
- SELECT: select a record
- DELETE: delete a record
- INSERT: insert a record
- UPDATE: change a record

A combination of these commands will be predefined on the application servers to carry the common tasks that query the database like adding log-records, managing venues/users.

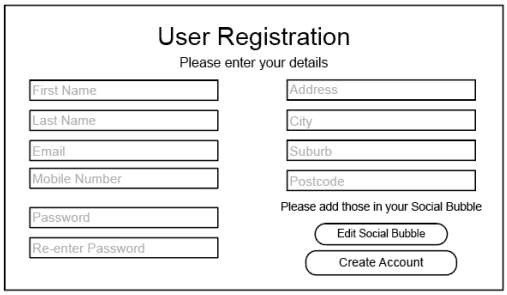
4.2 User Interface Design

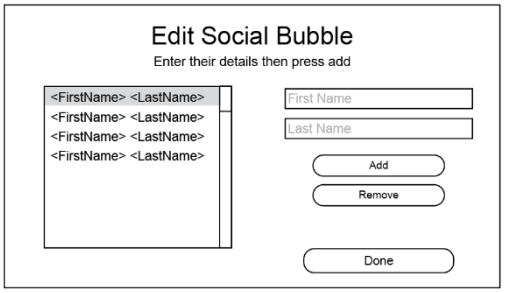
Venue Tablet Interface

In consideration of visitors with disabilities, the UI design is kept simple by utilizing large buttons. But it is assumed that QR bracelets will be used for these cases as it makes logging in for them a lot easier.

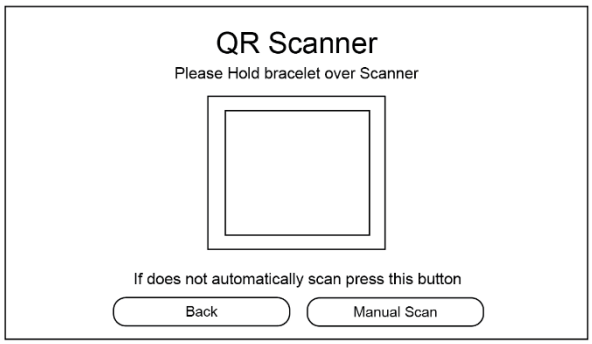


To address the possible language barriers first screen displayed on the tablet will be a language selector which will change the interface text to the chosen language. The login selection screen consists of three buttons limits visitor's ability to get side-tracked and confused as the choices are obvious and straightforward.

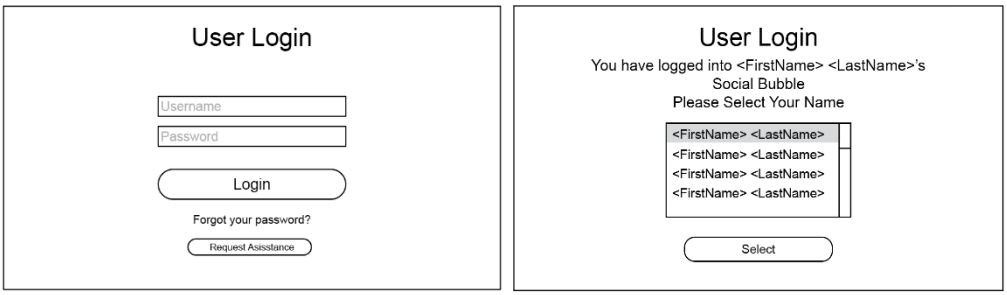




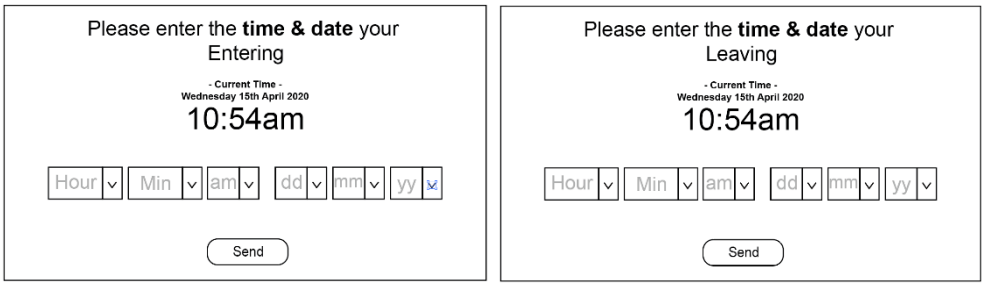
This wireframe shows the how User's Register. When creating a Social Bubble, the member registering the Bubble only needs to enter their information and adds only the First/Last name of each member in their Bubble. This is assuming that everyone within their Bubble has accessible contact with the main member when a notification is released.



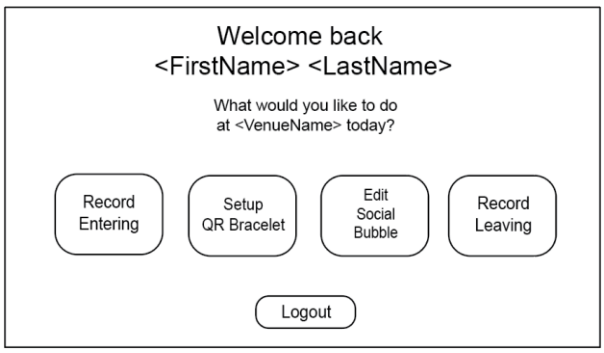
This wireframe shows how the QR scanner is going to be used by the visitors. This screen will appear in both login and when they need to register a bracelet. The scanner should automatically scan the bracelet when lined up correctly, but if the visitor has trouble lining up the QR bracelet they can choose to manual scan which will take a picture and find the QR code using that method.



This is the default user login interface. Basic login with either Email/Given Username and password. If a visitor belongs to a Bubble with two or more members, they will be presented with this companion screen which asks them to select their name from the Bubble's member list.

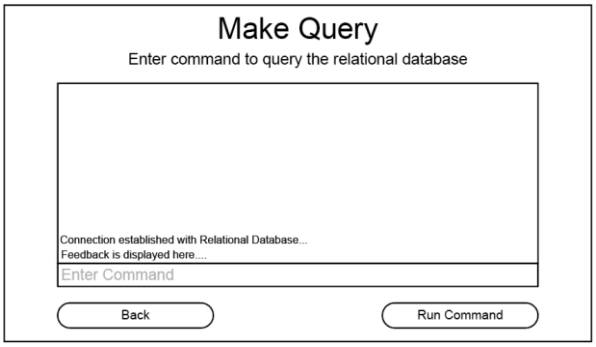


These wireframes show the interface when you select Entering/Leaving. To minimize human error of typing error the interface is designed with a drop box instead of a textbox. This interface also takes the possibility that people may not have access to the time, so it shows the current time and date on the page.

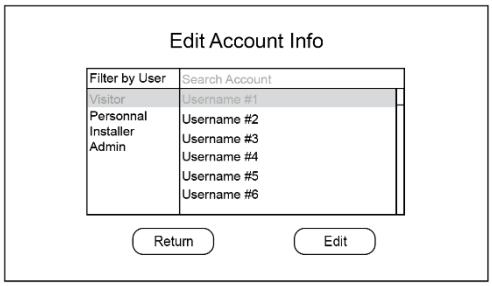


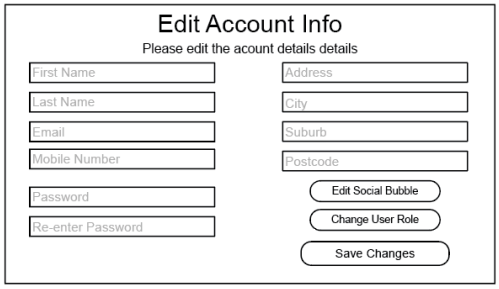
This is the home screen when the Visitor logs in. They have a choice of replacing/setting up their QR bracelet, requesting a 'add/remove member' of their social bubble and selecting weather their leaving or entering. Keeping all options to large buttons to keeping it simple and straightforward.

Interfaces that Query the Database

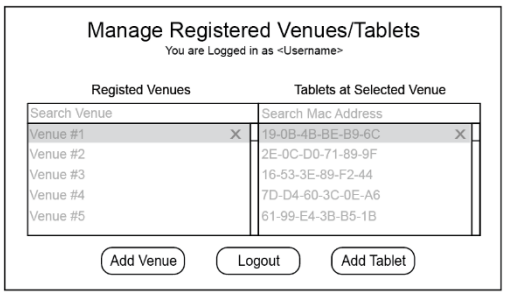


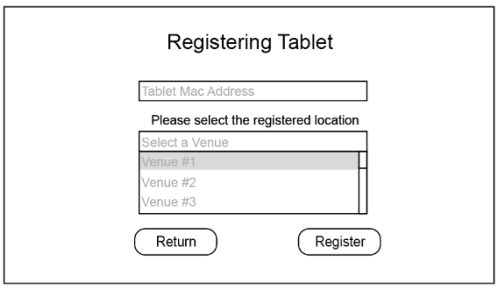
This is the wire frame for the Basic query console that is only accessible by the NZPRU admins. Here they can make any query to the relational database using any MYSQL command lines.

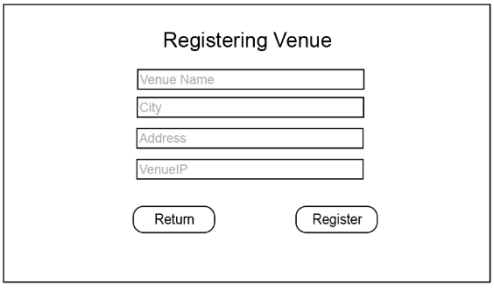




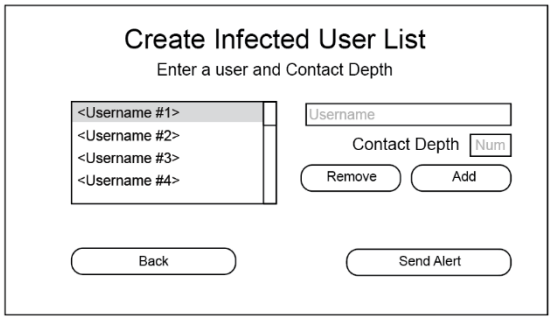
Another interface that automatically queries the relational database is the Edit Account Info. This is accessible to only NZPRU admins. It allows admins locate a registered account by username (ID) and edit their properties. Once the changes are saved a query is made for each of the changes for the relational database to handle.







This interface shows the NZPRU installer's feature of adding/removing tablets and venues. The interface queries the relational database whenever it to add or remove a tablet/venue



This interface shows the NZPRU personnel's feature of sending out an automated alert. This interface also automatically queries the relational database as there will be a NZPRU system in place to automatically create a list dependant on the added infected user and it's contact depth.

5.Conclusion

5.1 Conclusion

The NZPRU Data Registration System will be purposed to keep track of all the visitors that come in and out of a venue. This Software Design Specification base outline of the first iteration of the system, changes to the specification can be made to better fit NZPRU's needs. Further consultation will be required to provide a seamless and finished system.