Clinic Management System

Functional Specification

INSTITUTE *of* TECHNOLOGY CARLOW

18th April 2018
BSc (Hons) Software Development

Name: Ryan Donoghue
Year: 4th year
Student ID: C00194829
Supervisor: Paul Barry

Table of Contents

Abstract

The purpose of this document is to outline the functional and non-functional requirements for this project. It will state who is the target market of this application and why they might use it. This document also states the work that is scheduled for each of the iterations.

Introduction

This project is a clinic management system that will be used by a user to handle the day to day running of the clinic. The user in this case can be a nurse, a doctor or some other member of medical faculty. The clinic will primarily be a substance misuse clinic, but can also be a day clinic or mental health clinic.The system can also be used by and admin to manage users and manage individual clinics.

The system will allow a user to log in to the system, and access the medical records for the clinic that they reside in. They will be able to show, edit and add patients, general practitioners (GPs), clinic doctors, pharmacies, phlebotomy records, vaccination records and reports. Users will not be able to view details of other clinics, and will not be able to view details of other users.

The user will have the ability to view reports on demand, which will allow them to view a summary of all patients currently in the system, with information shown such as county of origin, who their GP is, who their clinic doctor is and who their pharmacy is.

The admin will be able to log in to the system, and be able to view all clinics and users in the system. The admin can add a new clinic, and add users to that clinic, but will not be able to view any data relating to that clinic such as patients, phlebotomy records and vaccination records.

The system will be designed with simplicity and attractiveness in mind, with helpful errors arising when a problem occurs. The UI will be fluid and inviting, leading the user to a positive UX.

The system will be designed with direct feedback from a potential user of the system (the client), and will reflect their needs and desires.

Target Market

This system is created for the users of clinics, for the day to day running of the clinic. It is intended that the system replaces paper based systems as the primary method of entering

information. The system is aimed at small to large companies, as well as new startups in the medical industry.

The system is aimed primarily toward an older generation of users, and as such will be designed and simply and intuitively as possible. However the system will remain powerful to allow more advanced users to not feel handicapped by the simplicity.

From a security perspective, the system is aimed towards those using the system in a public healthcare industry. The system must comply to strict industry regulations, and therefore qualify as a candidate for use in such an industry.

Functional Requirements

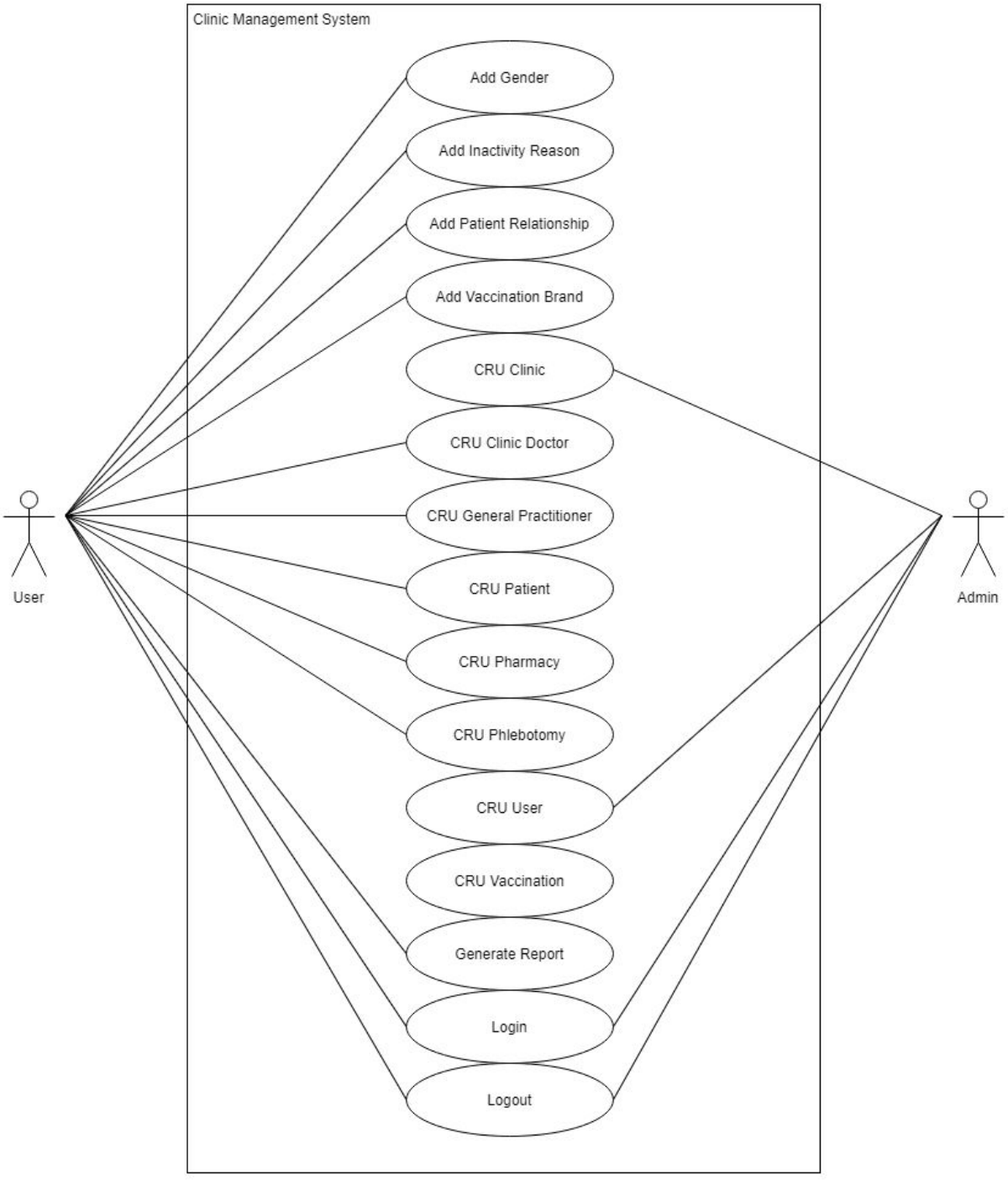
Hardware Requirements

- Linux
    - The system requires a Linux based server in order to host the system.
- Web Capable PC
    - The system requires a web capable PC in order to access the system.
- SSH Ready PC
    - The system may require an SSH ready PC in order to access the server the system is located on

Software Requirements

- Erlang, OTP & Elixir
    - The system logic will primarily be written using Elixir. Elixir is built upon an Erlang base, and when installing Elixir, Erlang is included. Elixir is available on all desktop OSs including MacOS, Windows and Linux.
- Phoenix
    - Phoenix is the web framework used by Elixir to handle and display webpages. It is also used for creating data for changesets, used by Ecto.
- Ecto
    - Ecto is used as an interface between Elixir/Phoenix and the underlying PostgreSQL, working similar to version control.
- PostgreSQL
    - PostgreSQL is used as the database, and is plugged into by Ecto to handle data coming in and out of the application
- NGINX
    - NGINX is used to serve web pages from a server to any browser
- Screen (Linux)
    - Screen is a Linux module that allows for separate “screens” within the Linux CLI, allowing for

Use Case Diagram



Brief Use Cases

Login

Name: Login
Actors: User
Precondition: User is not logged in
Description: This use case beings when the system user wants to log in to their account on the system. The user enters their username and password to log in to the system. The use case ends when the user successfully logs into the system.
Consequence: The user is now logged in

Logout

Name: Logout
Actors: User
Precondition: A valid standard user session exists.
Description: This use case begins when the user wishes to logout of the system. The user clicks the logout button. This use case ends when the user has successfully logged out and the login screen is displayed.
Consequence: The user is now logged out

CRU patient

Create Patient

Name: Create Patient
Actors: User
Preconditions: A valid standard user session exists, a valid general practitioner exists, a valid clinic doctor exists, a valid pharmacy exists
Description: The use case begins when the user successfully navigates to the add patient screen. The user fills out the form of patient details and submits the information. The use case ends when the patient is successfully added to the system.
Consequence: A new patient is added to the system.

Read Patient

Name: Read Patient
Actors: User
Precondition: A valid standard user session exists, a valid general practitioner exists, a valid clinic doctor exists, a valid pharmacy exists, a valid patient exists.
Description: The use case begins when the user successfully navigates to the show patient screen. The use case ends when the system displays the patient information to the user.
Consequence: The patients details are shown on screen

Update Patient

Name: Update Patient
Actors: User
Precondition: A valid standard user session exists, a valid general practitioner exists, a valid clinic doctor exists, a valid pharmacy exists, a valid patient exists.
Description: The use case begins when the user successfully navigates to the edit patient screen. The user fills out the appropriate patient information that requires updating on the form, and submits the form. The use case end when the patient information has successfully been updated.
Consequence: The patients details have been updated.

CRU pharmacy

Create Pharmacy

Name: Create Pharmacy
Actors: User
Preconditions: A valid standard user session exists
Description: The use case begins when the user successfully navigates to the add pharmacy screen. The user fills out the form of pharmacy details and submits the information. The use case ends when the pharmacy is successfully added to the system.
Consequence: A new pharmacy is added to the system

Read pharmacy

Name: Read pharmacy
Actors: User
Precondition: A valid standard user session exists, a valid pharmacy exists
Description: The use case begins when the user successfully navigates to the show pharmacy screen. The use case ends when the system displays the pharmacy information to the user.
Consequence: The pharmacy is shown on screen

Update pharmacy

Name: Update pharmacy
Actors: User
Precondition: A valid standard user session exists, a valid pharmacy exists
Description: The use case begins when the user successfully navigates to the edit pharmacy screen. The user fills out the appropriate pharmacy information that requires updating on the form, and submits the form. The use case end when the pharmacy information has successfully been updated.
Consequence: The pharmacy details have been updated.

CRU doctor

Create doctor

Name: Create doctor
Actors: User

Preconditions: A valid standard user session exists
Description: The use case begins when the user successfully navigates to the add doctor screen. The user fills out the form of doctor details and submits the information. The use case ends when the doctor is successfully added to the system.
Consequence: A doctor has been added to the system

Read doctor

Name: Read doctor
Actors: User
Precondition: A valid standard user session exists,a valid clinic doctor exists
Description: The use case begins when the user successfully navigates to the show doctor screen. The use case ends when the system displays the doctor information to the user.
Consequence: The doctor is shown on screen.

Update doctor

Name: Update doctor

Actors: User

Precondition: A valid standard user session exists, a valid doctor exists.

Description: The use case begins when the user successfully navigates to the edit doctor screen**.** The user fills out the appropriate doctor information that requires updating on the form, and submits the form. The use case end when the doctor information has successfully been updated.
Consequence: The doctors details have been updated in the system.

CRU gp

Create gp

Name: Create gp
Actors: User
Preconditions: A valid standard user session exists
Description: The use case begins when the user successfully navigates to the add gp screen. The user fills out the form of gp details and submits the information. The use case ends when the gp is successfully added to the system.
Consequence: A new GP is added to the system.

Read gp

Name: Read gp
Actors: User
Precondition: A valid standard user session exists, a valid gp exists.
Description: The use case begins when the user successfully navigates to the show gp screen. The use case ends when the system displays the gp information to the user.
Consequence: The GP is shown on screen.

Update gp

Name: Update gp
Actors: User
Precondition: A valid standard user session exists, a valid gp exists.
Description: The use case begins when the user successfully navigates to the edit gp screen. The user fills out the appropriate gp information that requires updating on the form, and submits the form. The use case end when the gp information has successfully been updated.
Consequence: The GPs details are updated in the system.

CRU Clinic

Create clinic

Name: Create clinic
Actors: User
Preconditions: A valid admin user session exists
Description: The use case begins when the user successfully navigates to the add clinic screen. The user fills out the form of clinic details and submits the information. The use case ends when the clinic is successfully added to the system.
Consequence: A clinic is added to the system.

Read clinic

Name: Read clinic
Actors: User
Precondition: A valid admin user session exists, a valid clinic exists.
Description: The use case begins when the user successfully navigates to the show clinic screen. The use case ends when the system displays the clinic information to the user.
Consequence: The clinic is shown on screen.

Update clinic

Name: Update clinic
Actors: User
Precondition: A valid admin session exists, a valid clinic exists.
Description: The use case begins when the user successfully navigates to the edit clinic screen. The user fills out the appropriate clinic information that requires updating on the form, and submits the form. The use case end when the clinic information has successfully been updated.
Consequence: The clinic is updated in the system.

CRU vaccination record

Create vaccination record

Name: Create vaccination record
Actors: User
Preconditions: A valid standard user session exists
Description: The use case begins when the user successfully navigates to the add vaccination record screen**.** The user fills out the form of vaccination record details and submits the information. The use case ends when the vaccination record is successfully added to the system.
Consequence: A new vaccination record is added to the system.

Read vaccination record

Name: Read vaccination record
Actors: User
Precondition: A valid standard user session exists,a valid clinic vaccination record exists
Description: The use case begins when the user successfully navigates to the show vaccination record screen. The use case ends when the system displays the vaccination record information to the user.
Consequence: The vaccination record is shown on screen.

Update vaccination record

Name: Update vaccination record
Actors: User
Precondition: A valid standard user session exists, a valid vaccination record exists.

Description: The use case begins when the user successfully navigates to the edit vaccination record screen. The user fills out the appropriate vaccination record information that requires updating on the form, and submits the form. The use case end when the vaccination record information has successfully been updated.
Consequence: The vaccination record is updated in the system.

CRU phlebotomy record

Create phlebotomy record

Name: Create phlebotomy record
Actors: User
Preconditions: A valid standard user session exists
Description: The use case begins when the user successfully navigates to the add phlebotomy record screen. The user fills out the form of phlebotomy record details and submits the information. The use case ends when the phlebotomy record is successfully added to the system.
Consequence: A new phlebotomy record is added to the system.

Read phlebotomy record

Name: Read phlebotomy record
Actors: User
Precondition: A valid standard user session exists,a valid clinic phlebotomy record exists
Description: The use case begins when the user successfully navigates to the show phlebotomy record screen. The use case ends when the system displays the phlebotomy record information to the user.
Consequence: The phlebotomy record is shown on screen.

Update phlebotomy record

Name: Update phlebotomy record
Actors: User
Precondition: A valid standard user session exists, a valid phlebotomy record exists.
Description: The use case begins when the user successfully navigates to the edit phlebotomy record screen. The user fills out the appropriate phlebotomy record information that requires updating on the form, and submits the form. The use case end when the phlebotomy record information has successfully been updated.
Consequence: The phlebotomy record has been updated in the system.

Add an inactivity reason

Name: Add an Inactivity Reason
Actors: User
Precondition: A valid standard user session exists
Description: This use case begins when the user successfully navigates to the add inactivity reason screen. The user fills out the inactivity reason form and submits. The use case ends when the inactivity reason has successfully been added.
Consequence: An inactivity reason has been added to the system.

Add a patient relationship

Name: Add a patient relationship
Actors: User
Precondition: A valid standard user session exists
Description: This use case begins when the user successfully navigates to the add patient relationship screen. The user fills out the patient relationship form and submits. The use case ends when the patient relationship has successfully been added.
Consequence: A patient relationship has been added to the system.

Add a vaccination brand

Name: Add a vaccination brand
Actors: User
Precondition: A valid standard user session exists
Description: This use case begins when the user successfully navigates to the add vaccination brand screen. The user fills out the vaccination brand form and submits. The use case ends when the vaccination brand has successfully been added.
Consequence: A vaccination brand has been added to the system.

Add a gender

Name: Add a gender
Actors: User
Precondition: A valid standard user session exists
Description: This use case begins when the user successfully navigates to the add gender screen. The user fills out the gender form and submits. The use case ends when the gender has successfully been added.
Consequence: A gender has been added to the system.

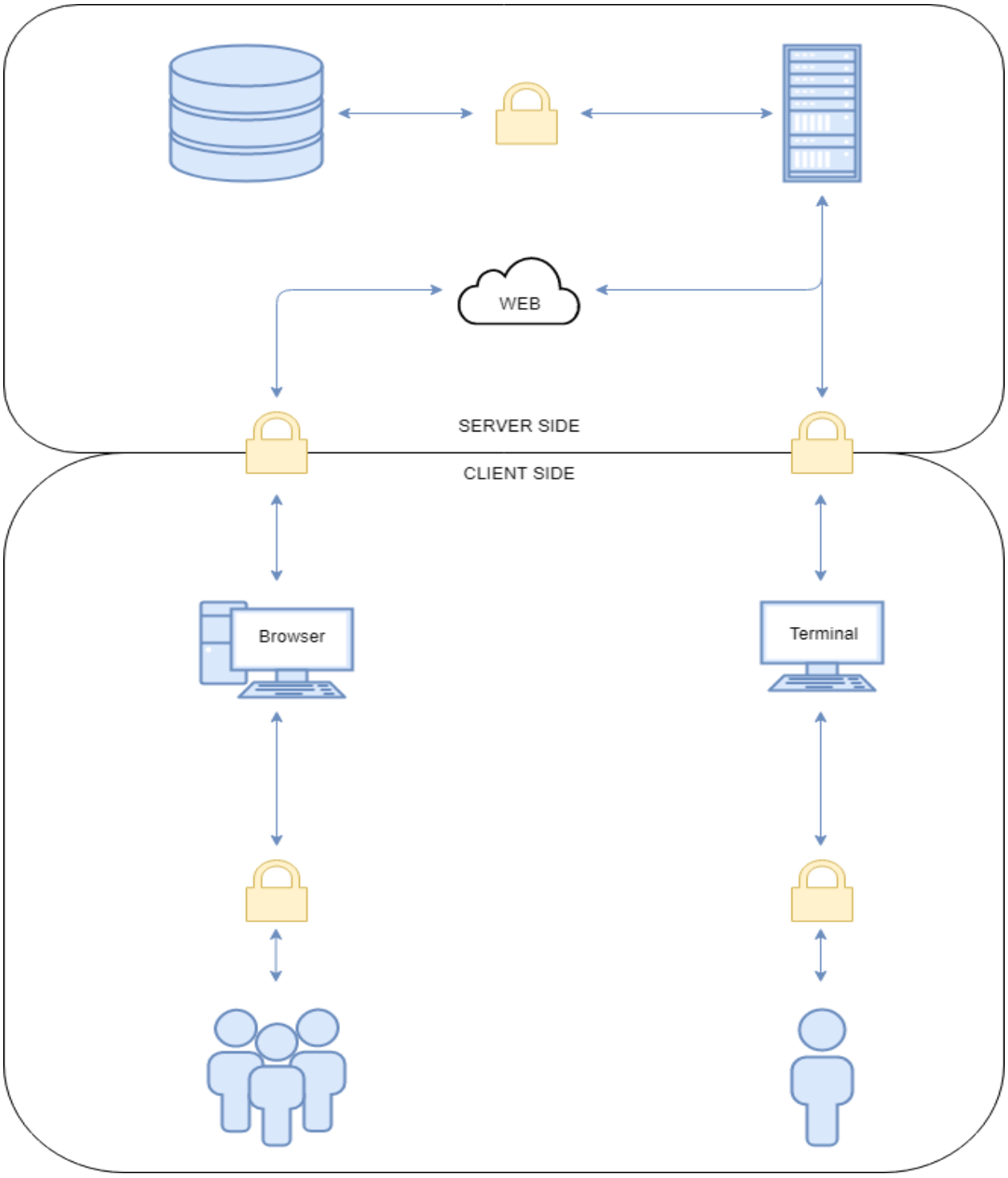
Create new user

Name: Create a user
Actors: User
Precondition: A valid admin user session exists, a valid clinic exists
Description: The use case begins when the user successfully navigates to the add new user screen. The user enters the information for the user into the displayed form, then submits. The use case end when the user is successfully added to the database.
Consequence: A new user has been added to the system.

Generate report

Name: Generate
Actors: User
Precondition: A valid standard user session exists
Description: The use case begins when the user successfully navigates to the report page. The use case ends when the report screen is shown.
Consequence: The report is shown on screen.

System Architecture



Supplementary Specification

Functionality

- The system will log all errors on the server it is installed on. This includes runtime errors, failed database inserts and incomplete forms.
- The system will allow users to CRU clinic entities and patients.

Usability

- The system will be usable by a wide range of users, ranging from basic to advanced users.
- Users should be shown appropriate error messages in the event of a problem occurring.

Reliability

- The system should have close to zero downtime.
- Should an error occur with a patient record, a backup must be obtainable

Performance

- The application should be as fast as possible, with little to no latency between operations.

Supportability

- The application should be easy to maintain
- The application should have appropriate facilities to be accessible from anywhere for maintenance
- Updating the application should be as simple as possible
- Installing the application should be as simple as possible

Security

- The system should be as secure as possible, both from a user encryption method, to the encryption of sensitive data stored by the application
- The system security should comply to relevant industry standards and comply to GDPR legislation.

Iterations

Iteration 1: October - December

- Continue learning Elixir.
- Begin development of basic functionalities of system proof of concept(POC) for showing to client.
- Further research into login, encryption functionalities.
- Bring POC to client, obtain preliminary feedback.

Iteration 2: December - February

- Continue learning Elixir.
- Begin developing system.
- Begin client feedback iterations.
- Continue developing system using feedback from client.

Iteration 3: February - April

- Begin implementing login and encryption mechanisms.
- Continue receiving feedback from client.
- Polish application to presentable standard.
- Present finished application to client.