*SHIFTSAPP – X12388761*

Product Design Specification

Version 1.0

*27/11/2015*

VERSION HISTORY

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version #** | **Implemented**  **By** | **Revision**  **Date** | **Approved**  **By** | **Approval**  **Date** | **Reason** |
| 1.0 | *Kellie Hughes* | *03/12/2015* | *TBC* | *TBC* | Initial Design Definition draft |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**UP Template Version:** 12/31/07

Table of Contents

1 Introduction 4

1.1 Purpose of The Product Design Specification Document 4

2 General Overview and Design Guidelines/Approach 4

2.1 Assumptions / Constraints / Standards 4

3 Architecture Design 4

3.1 Class Diagram 5

3.2 Cloud Architecture 6

3.3 Logical View 6

3.4 Hardware Architecture 6

3.5 Software Architecture 7

3.6 Security Architecture 7

3.7 Communication Architecture 7

3.8 Performance 7

4 System Design 7

4.1 Use-Cases 7

4.2 Database Design 8

4.3 Data Conversions 9

4.4 Application Program Interfaces 9

4.5 User Interface Design 9

4.6 Section 508 Compliance 12

5 Appendix A: References 13

6 Appendix B: Key Terms 14

# Introduction

## Purpose of The Product Design Specification Document

The Product Design Specification document is used so that the design team involved can keep up to date with the developments and changes that needed to be made throughout the process of the application. It shows all of the designs and architectures that need to be implemented to make the application a success and also keeps the team on track. There is different versions of this document throughout the application process so that it can be changed overtime if the design team decide to go in a different route or if they come into some problems. Its intended audience is the project manager, project team, and development team. Some portions of this document such as the user interface (UI) may on occasion be shared with the client/user, and other stakeholder whose input/approval into the UI is needed.

# General Overview and Design Guidelines/Approach

The design is to make a simple easy to use UI developed in PHP which can take input from the user and register them onto the system which will hold their information in a database. The system must have log in and register buttons as well as a chat system integration. Once logged in the application will have a rostering system which will be unique to the employee and will show the employee their schedule for the next week. The main focus is for the application to be easy to use and easy to understand.

## Assumptions / Constraints / Standards

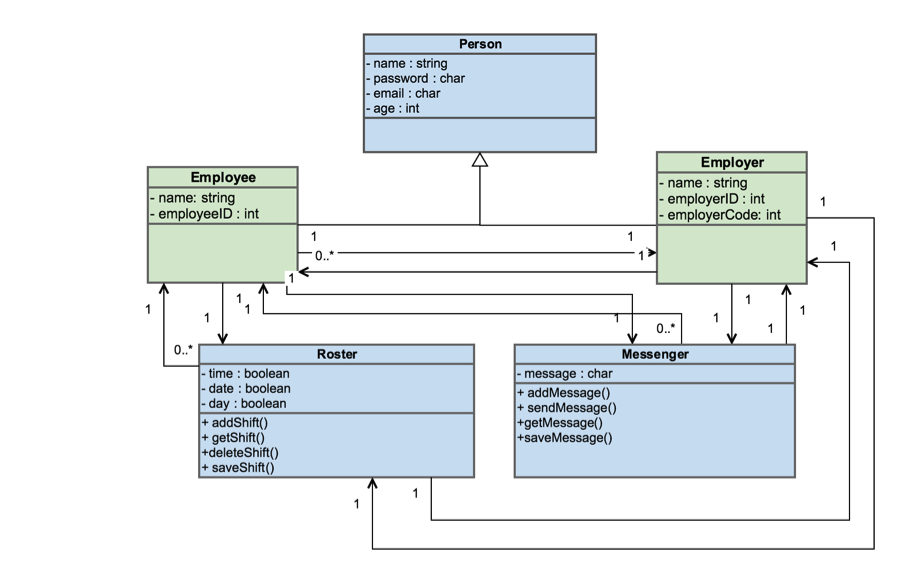
The constraints that could be faced when doing this application could be outdated coding examples online making it difficult to find the right thing that you are looking for. Another constraint could be not having the knowledge that you need to be able to perform a certain task and then having to learn the method from scratch in order to complete the task.

The design standard problems that I could come into would be making sure that it is adhereing to the standards of all usual web applications and that it doesn’t break any of the coding practices in the process. As a web application there is a high and usual standard held in which the general public expect it to be held up to.

# Architecture Design

This section outlines the system and hardware architecture design of the system that is being built.

## Class Diagram



the class diagrams shows how the different part of the application works with each other and how the users have different restrictions in the application based on their status in the company.

E.g the employer can have many employees however the employee can only have one employer. And the roster can have many employees but the employees can only have one roster.

It also shows with information is taken about the employees and the employers and shows that they are separated which means that they will have different authorisations within the application

E.g the employees will not be able to edit the roster however the employer will be able to edit.

## Cloud Architecture

## 

As the application is a cloud based application, above is the architecture of the cloud in which the system will run on. The application will be deployed to the cloud.

The architecture starts on the users web browser from this they will use the internet to get the to the application. The application will be hosted on the web host and stored on the web server with the information about the users and other parts stored in the database. The server, host and database will all be cloud based.

## Logical View

The logical view for my application would be split into 3 parts:

The cloud based user interface, the cloud based back end and also the cloud based database to hold all of the information received from the interface. The user interface should be able to support log in and to communicate with the back end of the system to choose where to take the user after all of the steps and should also communicate with the database to hold all of the information.

## Hardware Architecture

I do not think that hardware architecture applies to my project as it is a cloud based web application.

## Software Architecture

The software architecture of the system will be that it is all based on a web browser which is connected to the internet and all of the information for the application will be stored in the cloud.

## Security Architecture

The main security architecture will be to make sure that unwanted people will not be able to get into the system. This will mean the coding of the log in and registration system will need to very good and also maybe implement a capita to prevent hackers. The security on the cloud will be very good which will keep the database and all the code needed for the system safe.

## Communication Architecture

The communication architecture will be similar of that to the cloud architecture and also the use case below. The web browser will need to connect to the internet and from the internet the internet will connect to the web application.

This will lead to the log in page where a user must enter their credentials and the system will verify that information with the database and once they are in the information is passed back and forth from the database to the rostering page.

With the chat messaging system there will be back and forth connectivity to the database to save all the messages that were sent but also messages sent to and from the server so that the messages can be passed between users.

## Performance

The performance of the application is based on timing and space.

The application should be able to support having up to 200 people on the site at any one time. And should be able to retrieve information from the database for these 200 people.

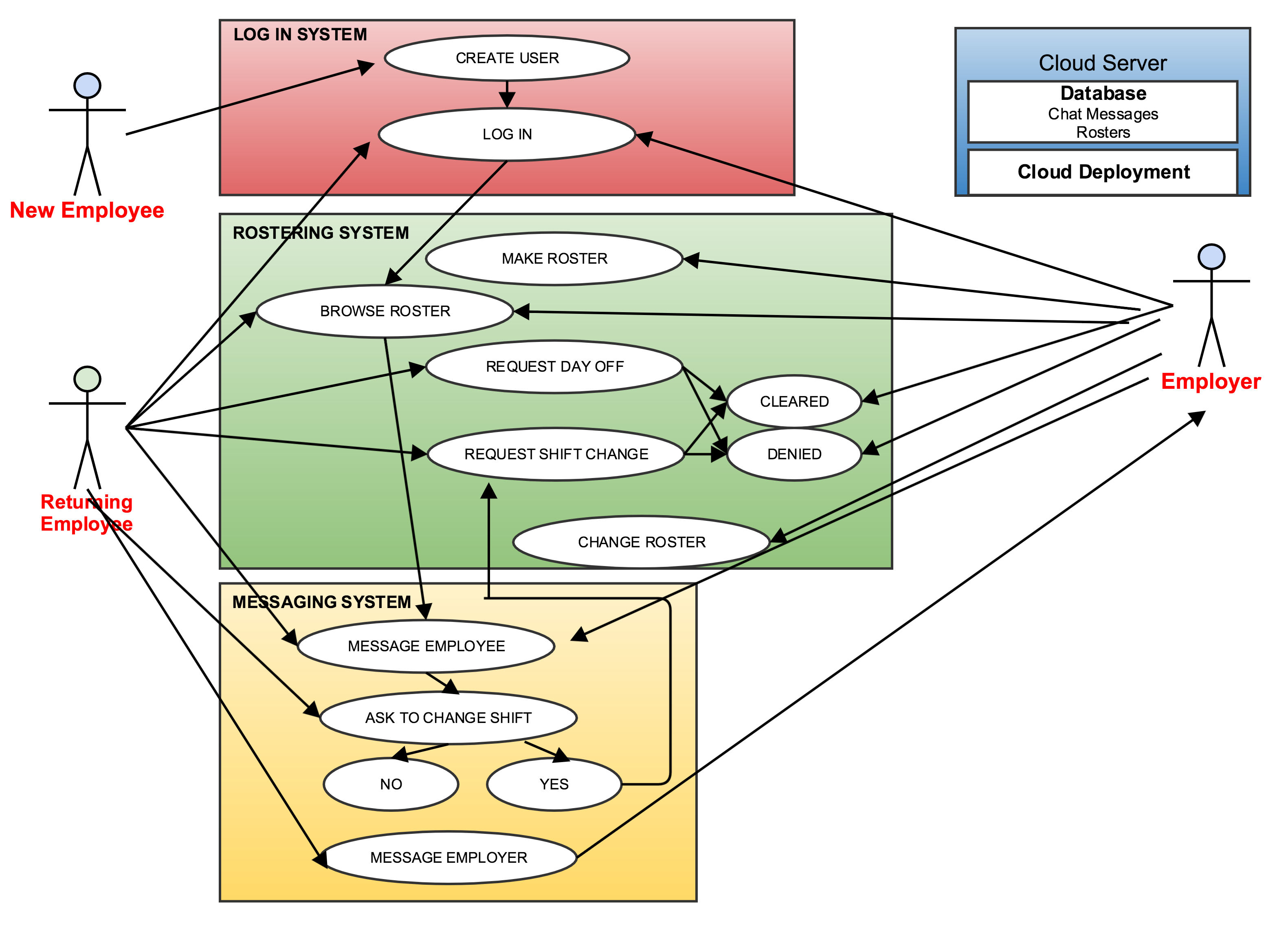
The database should have enough allocated space to be able to hold all of the log in details as well as newly registered members and should be able to hold all of the chat messages that have been going on between members. The system should also let a warning out to tell the developers when it is running out of space but this is something that the cloud can allow you to do.

The system must be able to complete tasks in a timely manner and should take no longer than 20 seconds to complete most of the tasks (with allowing an extra few seconds for peak times of the day). This means that it should be able to log in members, register members and also send and receive chat messages at a quick pace.

# System Design

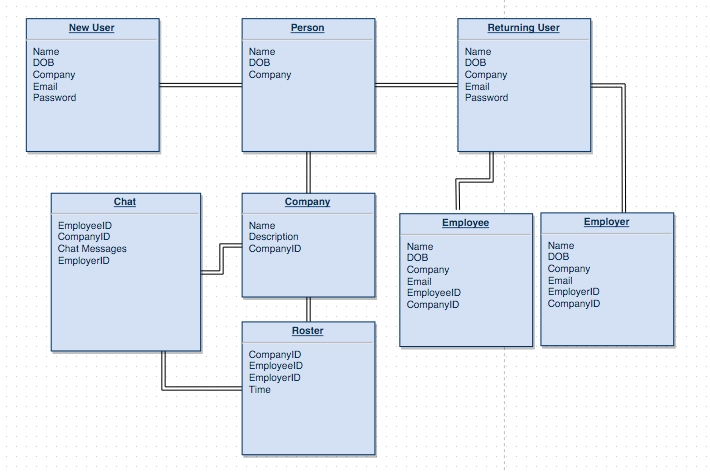
## Use-Cases

This is the use case for the application. It shows the interactions between the users (the employee and the employer) and the system



## Database Design

Below is all the things that will feature in the database and how they are connected.



## Data Conversions

I have yet to map out the way in which the information will be converting.

## Application Program Interfaces

At this time I have not decided to add any APIs but as this is only version 1.0 this may change.

## User Interface Design

Below are some screen shots of what the system should look like. These are from the early stages of development and are subject to change. The application should have 3 interfaces. The log in page, the roster page and the chat system page.

The Log In Page:

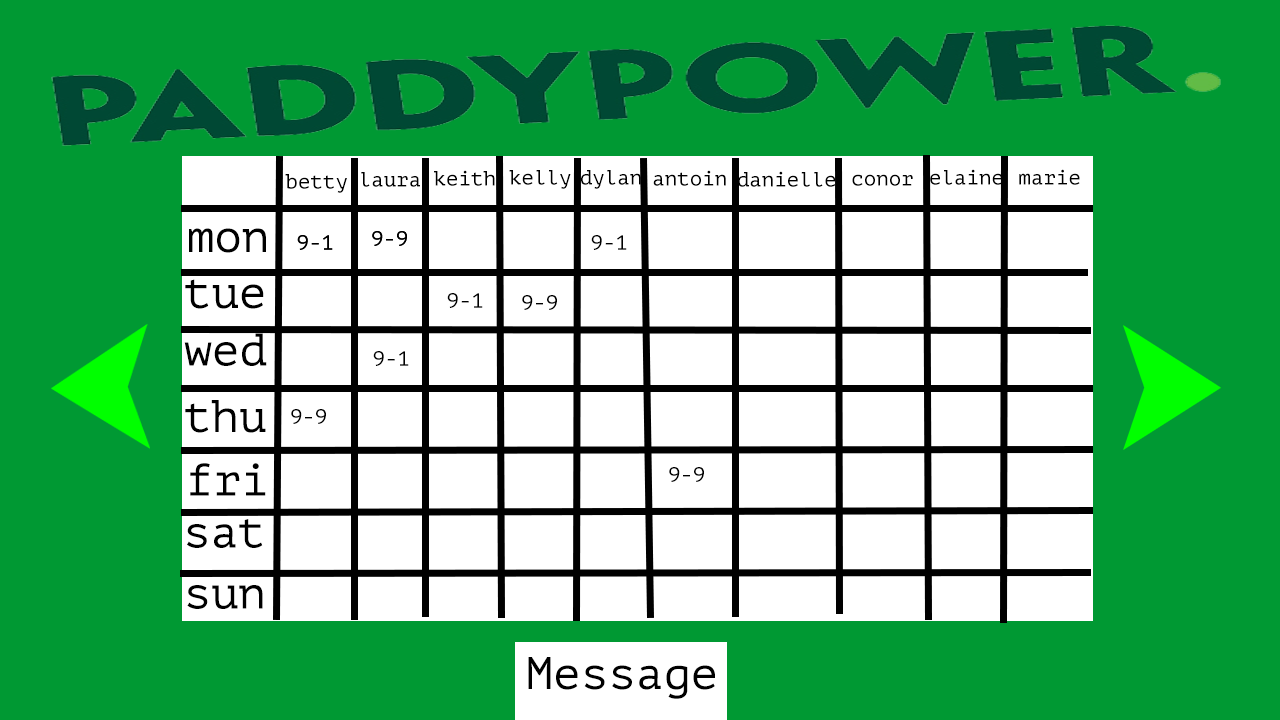


The log in page with have two buttons. The sign in button and the register button. For pre-existing users they enter their username and password and press the sign in button. For users that are new to the site and have yet to have a username and password they should press the registration page and they will be taken to the registration page.



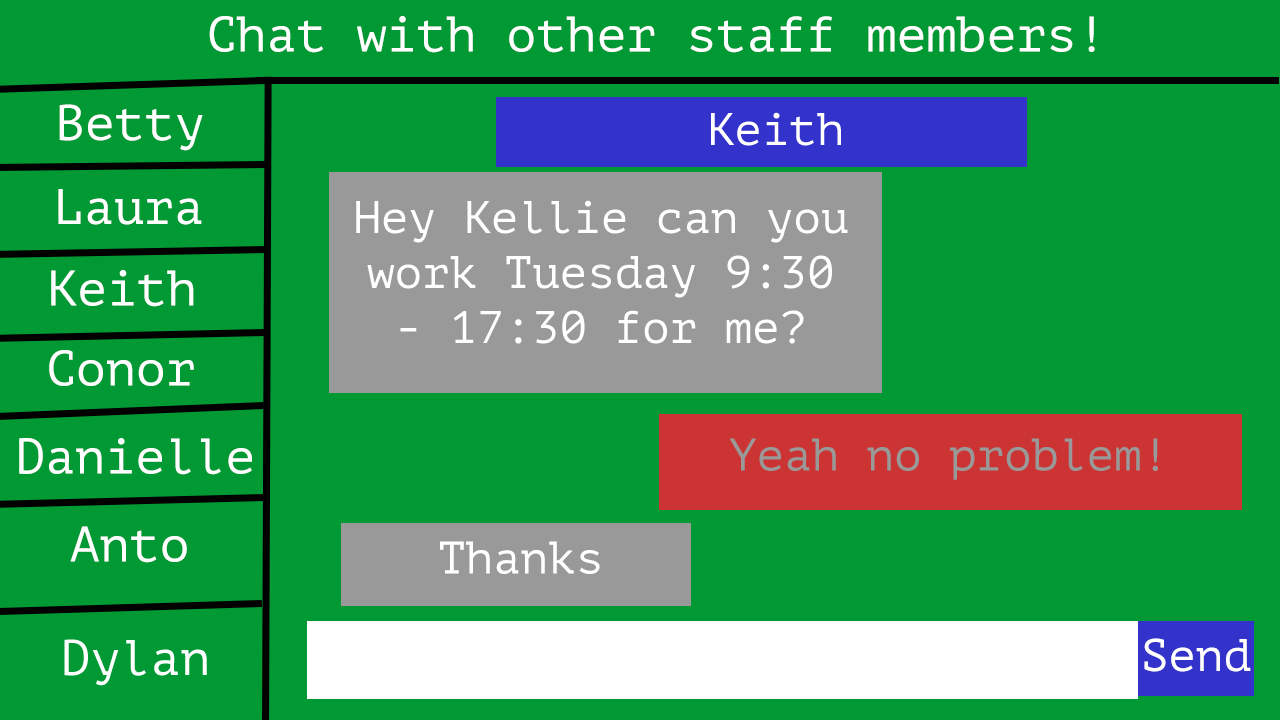
once the users have entered all their details they are ready to be a user of the system and will be taken to the roster page

The Roster Page:



the rostering page will be unique to your employment and will show the roster for the current week of your work. Here you will be able to see your shifts in work and if you have a problem you may be able to change your shifts with the help of your employer. Only the employer can change the shifts. At the bottom of this page there will be a link to the chat system

The Chat System Page:



In the chat system you will be able to chat to the other employees and your employer. Here you can message other members of staff and maybe request for a shift change or day off.

## Section 508 Compliance

As the application is in the early stages of the implementation we have not yet looked at section 508 compliance but I do believe that it is very important to make it easier for the disabled and should work our hardest to try and make it easier for them:

These ideas for making it easier include:

* Bigger buttons. (blind)
* Speech activation (blind)
* Browsers come with inverted screens if needed. (colour blind)
* Colours to suit the colour blind (colour blind)

# 

# Appendix A: References

The following table summarizes the documents referenced in this document.

|  |  |  |
| --- | --- | --- |
| **Document Name and Version** | **Description** | **Location** |
| Requirement Specification Document  Version 1.0 | Parts of this document have been better explained in the requirement specification document | Available on Moodle |

# Appendix B: Key Terms

The following table provides definitions for terms relevant to this document.

|  |  |
| --- | --- |
| **Term** | **Definition** |
| UI | User Interface, what the application will look like |
| API | Application Programming Interface, eg. Twitter API |
| GUI | Graphical User Interface, the official name for a UI |