Smart Hand Sanitising Dispenser System

Software Functional Specification

Group 8

Alison Kennedy (CASE)

Liviu Nastase (ECE)

Shane Cawley (CAM)

Ryan Lacey (CAM)

0. Table of contents

1. Introduction

To note: in the following document the words *dispenser*, *dispensing device*, *device*, *hand sanitising device*, *sanitising device*, and *hand sanitiser dispensing device* all refer to the physical product that will be dispatched to the user and is shown on the left hand side of the system architecture diagram, as shown in section 4.

1.1 Overview

The product to be developed is a smart hand sanitiser dispenser system, for use in places of education, healthcare and business.

The system has multidisciplinary components, including those of mechanical, electrical and software engineering, and as such will be constructed by engineers from said disciplines. The aim of the project is to add new and innovative technology to a common, everyday object to increase its use and efficiency, at a time it is needed most.

The system will allow for hand sanitising devices to be monitored and controlled remotely. More importantly, to quickly, easily, and automatically perform data analysis and gather unique insight to the public's habits and attitudes towards hand hygiene.

The device will also allow for a degree of manipulation, through the inclusion of an LED screen. The public may be influenced to use the device by persuasive imagery and messages, which can also be measured through data analytics and artificial intelligence.

The ultimate goal is to increase hand hygiene in the public domain, thereby decreasing the spread of viruses and disease, using technology.

1.2 Business Context

The smart hand sanitiser dispensing device in question is overwhelmingly in demand during today's global coronavirus pandemic crisis. Public places and businesses without a hand sanitiser dispenser at its entrance and/or placed around the facility are denounced. A business feature that was known as a convenient addition before has now become a necessity.

The addition of smart features and networking brings the device into the 21st century, allowing users to remotely control the amount of fluid dispensed and gather usage statistics. Increased efficiency, accuracy and effectiveness of a hand sanitising system will limit the spread of disease and illness in the

community. The statistics gathered can also be used in future academic research, with the aims of making public health and social breakthroughs.

1.3 Glossary

LED - Light Emitting Diode. An LED bulb is visible on the outside of the device to attract attention to itself.
Photoresistor - Also known as an LDR (Light Decreasing Resistor). Detects the amount of light present in the area it is placed. Used in the dispensing device to determine how much sanitising fluid is left inside the dispenser.
API - Application Programming Interface. A set of reusable functions and procedures used for building larger applications.
IP - The principal communications protocol of the internet.
TCP - Transmission Control Protocol. Works with Internet Protocol (IP). A communication suite used to transfer data on the internet.
Socket - A socket is essentially a node with an IP address and a port number.
Static IP Address - An IP address that does not (or very rarely) changes.
Servo Motor - Also known as a servo. A self-contained electric device that rotates or pushes parts of a machine with great precision
Internet of Things (IoT) - The interconnection of computing devices found in everyday objects through the internet.
Raspberry Pi - often referred to as ‘pi’ in this document. A small, credit-card sized computer. Often used to prototype IoT projects.
Artificial Intelligence (AI) - A smart algorithm that deduces its own findings and alters the operation of a system accordingly.
Operator - A different type of user than that who uses the dispensing device for sanitation purposes. Uses the web application to control and monitor the system.
Lurking variable - a variable that influences both the dependent variable and independent variable, making it appear that the two have a stronger relationship than that in reality.

2. General Description

2.1 Product / System Functions

User Functionality
The user will approach the device in a typical fashion to that of other automatic dispensers already on the market. The only noticeable difference for the user is that the device will flash at them when they are standing within two meters of it, and display persuasive images and messages on its LED display. Should the user place their hand underneath the device, it will automatically dispense sanitising fluid without the need to touch the physical device.

Operator Functionality
As described in the document glossary (1.3 Glossary) the operator is a special type of user of the system. The operator is able to log in to the web app and view the devices on the networks and their usage data. They will also be able to view the results of data analytics on this usage data and remotely control the device. For example, changing the amount of fluid dispensed or changing the image/message displayed on the devices' LED display.

The operator may use the information gathered and displayed to them to make better decisions. Such as, when to order new stock, to move devices that aren't being used etc.

Device - App Functionality
In order for the operator to achieve their objectives, the devices on the network will need to communicate successfully with a web app. The devices will use a network API to communicate with a central server. The devices will send their usage data periodically, which will be processed by the server and stored in a database. The information in the database will be displayed to the operator via a web application hosted on the server. An algorithm on the server will perform data analytics on the database and display its findings to the operator, also through the web application.

2.1.1 Hardware Interfaces

The device will use a Raspberry Pi Zero W which will collect and send data from sensors to the server. Four photoresistor sensors will send information to the Pi in order to determine the level of the liquid. The levels will also be shown using an RGB LED, and when the levels are too low, a buzzer will be

activated upon attempted use. One motion sensor will count the number of people who are passing the device. An ultrasonic distance sensor will be used in order to trigger the release mechanism of the dispenser. The releasing mechanism will be actuated by a servo motor. On the front of the dispenser will be placed a display which will receive information from the pi and display persuasive messages or images.

2.1.2 Communication Interfaces

The dispensing device will be programmed to have a five character unique key. This key will be used to differentiate the different devices on the network, and passed to the server upon start up. When creating an account and logging in to the web application, the operator will have the option to ‘register’ their devices, by entering their unique keys into a form. The server will use a relational database to keep record of the relationships between operators and their devices.

Communication between the server, the devices, and the operator using a personal computer, will all take place using the internet connections. Specifically, the devices will be using WiFi. The operator may view the web application using their choice of cellular data or WiFi on their personal computer.

Components of the device (eg. sensors and motors) will communicate with the Raspberry Pi using electrical signals.

2.2 User Characteristics and Objectives

The device’s users would typically be ordinary members of the public going about their daily lives. The product has been designed with the intention to use in public buildings such as schools and hospitals, as well as private businesses. There is no particular age group or category the product is aimed at.

It is expected that the device will be more heavily used in health care settings, such as hospitals and nursing homes. With this in mind, the device will be used at a higher frequency by the vulnerable and immunocompromised in our society, as well as their doctors, nurses and carers.

The device will also be heavily used in educational settings such as schools, colleges and universities. These are heavily populated areas with a large capacity for the virus to spread. Especially in younger settings, such as

primary schools, where the inhabitants may be much less aware of the risk in not maintaining proper hand hygiene.

The objective of the typical user, would be to obtain a sanitising solution from the device, killing any viruses or bacteria present on their hands, and preventing them from spreading to others.

The objective of the operator, would be to remotely control and monitor the devices on the network. To alter the amount of fluid dispensed and to change the image/message displayed on the devices LED screens remotely. Also to obtain system data and statistics and use this information to make better informed decisions.

2.3 Operational Scenarios

2.3.1 Initial Set Up

Once the dispenser is plugged in to a wall outlet, power will be supplied to the Raspberry Pi and its connected sensors and components. The device will connect to the server using the network API, and announce its 5 character unique-key. The server software will keep a record of each device's unique-key for later use. The dispenser will then be ready to use by members of the public.

2.3.2 User Attempts to Use The Dispenser

The motion sensor will detect when a potential user is within two meters of the dispenser and feed this information to the raspberry pi. The pi will then send a command to the LED to flash for a short period of time, attracting attention. A persuasive message or image will also be shown on the LED screen, attracting attention.

The user will notice the device and place their hand below it, in the typical fashion of the vast majority of automatic dispensers already on the market. The ultrasonic sensor will detect the hand's presence and the pi will instruct the servo motor to put pressure on the bag of fluid's nozzle for a certain amount of time, dispensing liquid. This length of time can be altered by the operator remotely, to dispense more or less liquid.

Will instruct the servo to rotate a certain amount in a clockwise direction when the liquid is released and anty-clockwhise when the servo returns to the neutra state. This angle of rotation can be altered by the operator remotely, to dispense more or less liquid.

The device will register that one (additional) alert has been issued and fluid has been dispensed one (additional) time. This information, along with a timestamp and other details, will be sent to the server's database periodically.

2.3.3 Dispenser Is Empty

The device's photoresistors will detect when the sanitising fluid level is below ten percent. The device will then issue a notification to the operator through the web app, communicating through the central server.

If the dispenser is completely empty and someone attempts to use the device, a buzzer will sound, emphasising that the device is empty. The pi will also register that one (additional) person has attempted to use the device when it was empty, later sending this information to the server's database.

2.3.4 Dispenser Is Refilled

When the dispenser is refilled, the device's photoresistors will detect the fluid level of the bag, which will be monitored constantly by the Raspberry Pi. The buzzer will no longer sound when the device is used and a notification will be sent to the operator through the web app that the dispenser has been refilled.

2.3.5 Operator Views Usage Data

The Operator will visit the web app through their web browser on their personal computer. If the operator has not set up an account yet, they will opt to 'sign up'. The operator's details will be stored in the user database. They will then have the option to register their owned device's unique-key, which is displayed on the outer casing of the device. If the unique-key is valid, and the device in question is online, its usage information will be displayed on the web app. Many devices can be registered to the operator in the same way, and their details displayed in the web app.

2.3.6 Operator Adjusts Fluid Dispensed

The web app will provide the operator the option to alter the amount of fluid dispensed by devices on the network. The operator may select a singular device to alter or to alter every device. This will be achieved by communicating with devices on the network through the central server. The operator will have the choice to dispense 5, 10 or 15ml.

2.4 Constraints

- ***Time***
  The design and development of the prototype is under a strict timeline, concluding on the 26th of June.
  The development of a manufactured and distributed version must also follow a short and strict timeline, in order to have the maximum effect possible in slowing the spread of the coronavirus.

- ***Hardware***
  For prototyping the device, we are limited to using a raspberry pi. This means that any sensors or other electrical components needed must be compatible with the pi. The pi and its components must also be of a reasonable size to fit in the electrical housing unit of the mechanical design of the dispenser.

- ***Cost***
  Keeping costs to a minimum is a core value of the multidisciplinary team developing this product. For this reason, every effort must be taken to keep costs to a minimum, including the cost of manufacturing and hardware.

3. Functional Requirements

The requirements' criticality has been ranked on the ISTQB's scale referenced at the bottom of the document.

3.1 Discovery of the Hand Sanitising Device

Description

Upon turning on the dispensing device, communication must be established with a central server. A networking API will be preinstalled on the device, which forms a connection with the central server using TCP sockets. The TCP socket will use an predetermined arbitrary port number and the static IP address of the server.

The device will also come with a unique-key/identifier, which it will announce to the server upon establishment of a connection. This unique-key will also be found written on the casing of the device, for easy identification by the user. The server will store the unique-keys of the devices on the network to keep a record of which devices are online and offline.

Criticality

Critical

Technical Issues

The sanitising device must be connected to wifi and have a stable internet connection.

Dependencies On Other Requirements

N/A

3.2 Ability to Temporarily Store and Send Usage Data

Description

The dispensing device's software must be able to not only to control its components (screen, proximity sensors etc), but also to store this data on the device and send this information to the server periodically. This will be possible through the preinstalled API.

Criticality

Critical

Technical Issues

If the server should go offline, the device will continue to gather data and attempt to send it until the server comes back online. If the server is offline for a long period of time there is a small chance that the device's memory may fill up. In this unlikely event the device will need to be rebooted. The data gathered in the time that the device was offline may be lost.

Dependencies On Other Requirements

3.1 Discovery of the Hand Sanitising Device

3.3 Sense When a Potential User is in Proximity and Flash a Beacon

3.4 Dispense Sanitising Fluid

3.3 Sense When a Potential User is in Proximity and Flash a Beacon

Description

The device will use electronic proximity sensing to determine when a potential user is within two meters of the dispenser. An LED will then momentarily flash to alert the potential user of the device's presence.

Criticality

Minor

Technical Issues

The sensor must be programmed and adjusted precisely to get an accurate two meter range.

Dependencies On Other Requirements

N/A

3.4 Dispense Sanitising Fluid

Description

An Ultrasonic Ranging Module HC - SR04 will be used to determine when a user's hand is directly below the dispenser. The software should interpret this signal and dispense a fixed amount of sanitising fluid by actuating the releasing mechanism with the help of a servo motor.

Criticality

Critical

Technical Issues

The nozzle of the bag containing the sanitising fluid must be lined up carefully during installation in order for this mechanism to work.

Dependencies On Other Requirements

N/A

3.5 Display Remotely Controlled Images and/or Messages and Measure Their Effectiveness

Description

Selected images and messages will be displayed on the device's LED screen. This is with the aim of persuading passers by in using the dispenser. The effectiveness of each image/message will be assessed by an artificial intelligent algorithm on the central server.

Criticality

Minor

Technical Issues

Other factors, known mathematically as lurking variables, may have an effect increasing the use of the device(s). This may skew data and impact accuracy of the measure of effectiveness of each image/message.

Dependencies On Other Requirements

3.1 Discovery of the Hand Sanitising Device

3.2 Ability to Temporarily Store and Send Usage Data

3.3 Sense When a Potential User is in Proximity and Flash a Beacon

3.4 Dispense Sanitising Fluid

3.6 Remotely Control the Amount of Sanitising Fluid Dispensed

Description

The device will dispense a finite amount of sanitising fluid when used. The amount should be adjustable and controlled by another user/operator from a central hub (web app).

Criticality

Trivial

Technical Issues

All sections of the software i.e the web app, server software, network API and dispenser software, must communicate perfectly in order to achieve this.

Dependencies On Other Requirements

3.1 Discovery of the Hand Sanitising Device

3.4 Dispense Sanitising Fluid

3.7 Notify the Operator When Sanitising Fluid is Low

Description

The central user/operator should be notified when sanitising fluid in the device is below ten percent.

Criticality

Minor

Technical Issues

Due to the location of the photoresistors there is a possibility of them to get contact with different liquids. This may in time cause rost or make the connections to fail. A solution is to use a protective rubber enclosure over the sensors which would mitigate this issue.

Dependencies On Other Requirements

3.1 Discovery of the Hand Sanitising Device

3.2 Ability to Temporarily Store and Send Usage Data

3.8 Perform Data Analysis Using Usage Data From All Dispensers

Description

An algorithm should be run on the central server that analyses the usage data in the database, and reports its findings to the user via a web app.

Criticality

Major

Technical Issues

Analysis of the database can be done independently on the server. However, the database is dependent on the device(s) being online in order to collect data.

Dependencies On Other Requirements

3.1 Discovery of the Hand Sanitising Device

3.2 Ability to Temporarily Store and Send Usage Data

4. System Architecture

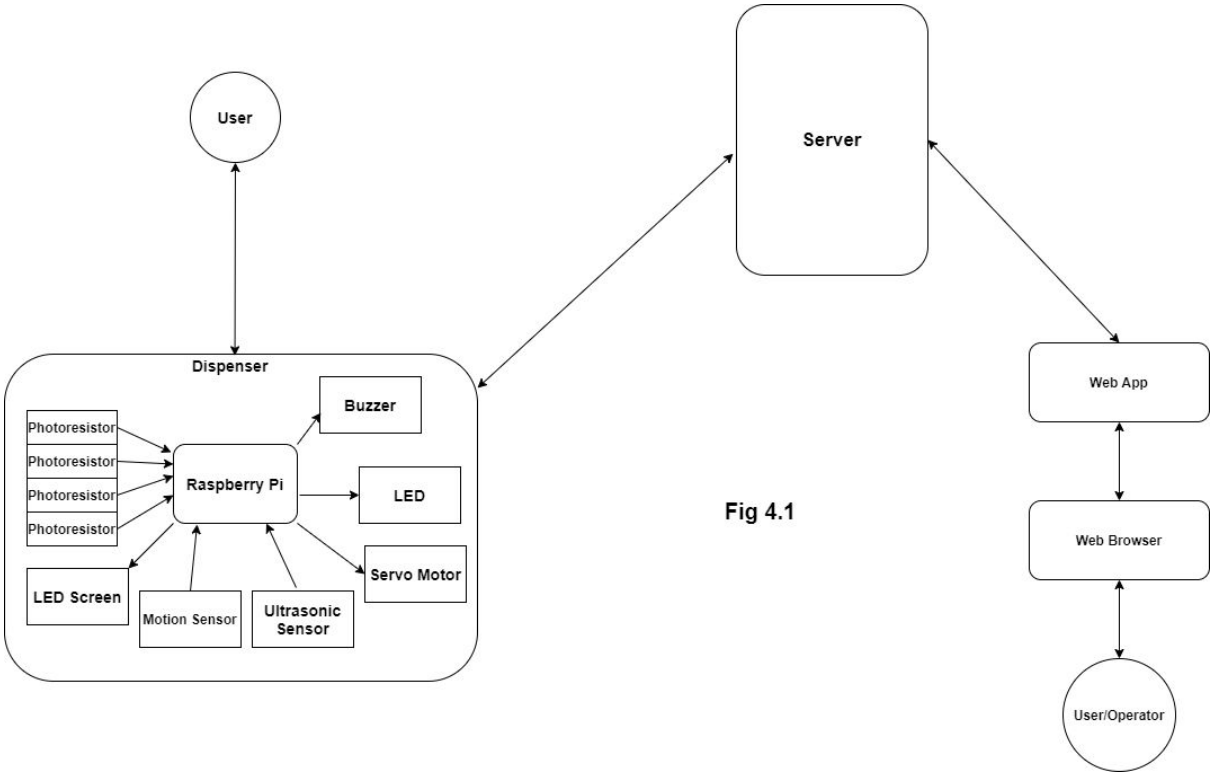


Fig 4.1

Fig 4.1 above illustrates the basic architecture of the system. The Dispenser is installed on a wall of the user's choice using adhesive strips and two screws. The device will be plugged in to a wall outlet. Once the dispenser is powered on, it will form a channel of communication with a central server using TCP sockets. The dispenser's motion sensor will alert the onboard computer, the Raspberry Pi, when a potential user is within two meters of the dispenser. The pi will then instruct the LED to flash for 2 seconds. The ultrasonic sensor will detect if the user has placed their hand below the device, and inform the pi. The pi will then activate the servo motor, which will squeeze sanitising liquid onto the user's hand. Photoresistors will inform the pi of the dispenser's current fluid level. If the fluid level is below 10%, an alert will be issued to the operator, through a web app.If the user attempts to use the dispenser when it is empty, the buzzer will sound, drawing attention to the fact it needs refilling. The pi will instruct the LED screen to display persuasive messages and images. This can be controlled by the operator by communicating with the pi through the central server.

The pi will gather and formulate usage data from information given by the sensors. This data will be sent to the central server and stored in a database. An algorithm on the server will run periodically, performing data analysis on the usage data in the database. The results as well as each dispensers' most recent usage data will be displayed to the operator via the web application.

The system will have a degree of artificial intelligence. The device's LED screen will display a variety of persuasive images and messages. An AI algorithm on the server will assess the usage data of each dispenser and determine which images/messages have the highest success in persuading people to use the individual dispensers. This information will be communicated to the operator through the web app.

5. High-Level Design

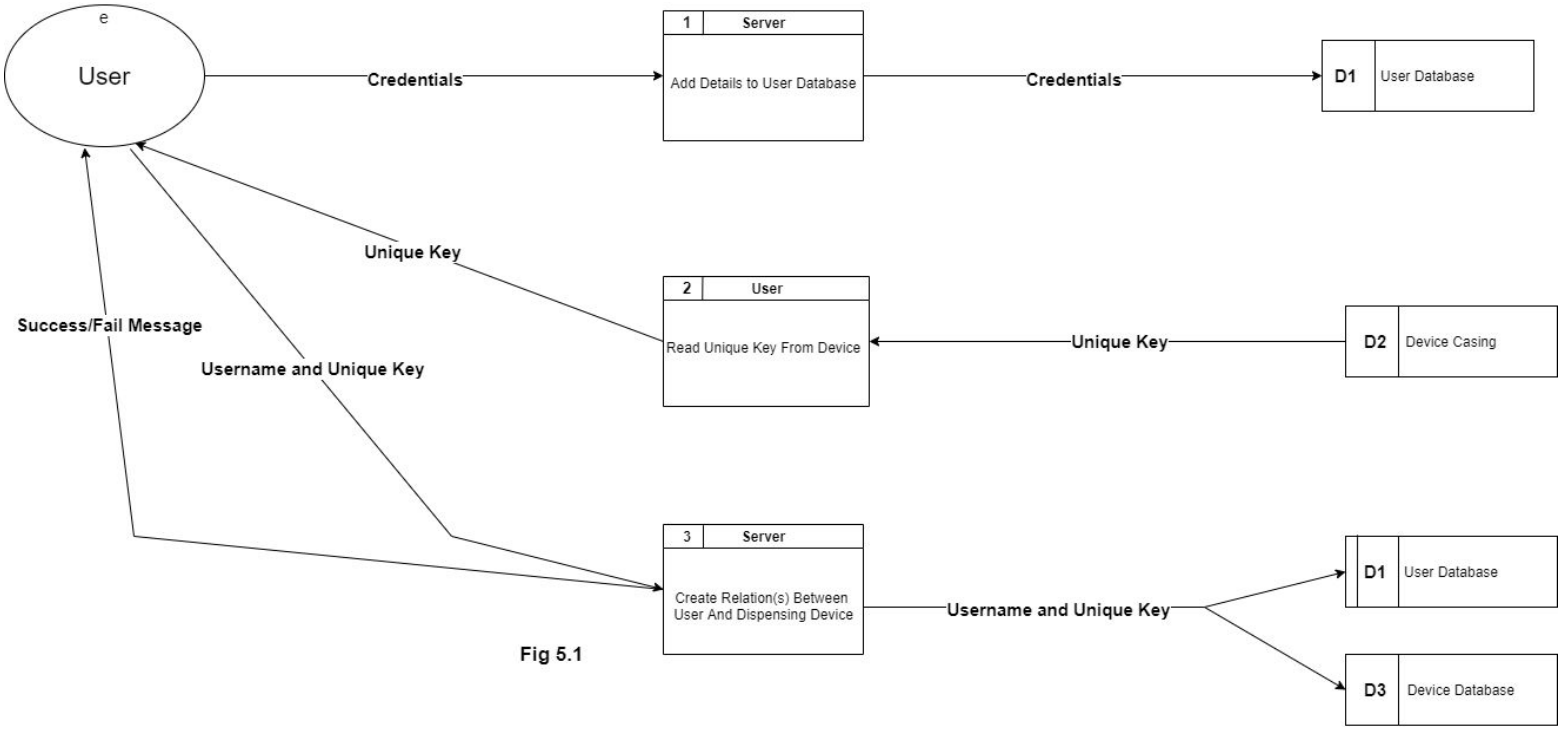


Fig 5.1

Fig 5.1 above is Data Flow Diagram, showing the data flow when a user first creates an account on the web app. The user's credentials are passed to the server, which stores this information in the user database, D1. The user then reads their device's unique key from the device's physical casing, and passes this information to the server along with their username. The server creates a relationship between these two pieces of information using a relational database system. The details/relationship is stored both in the user table and the device table of the database.

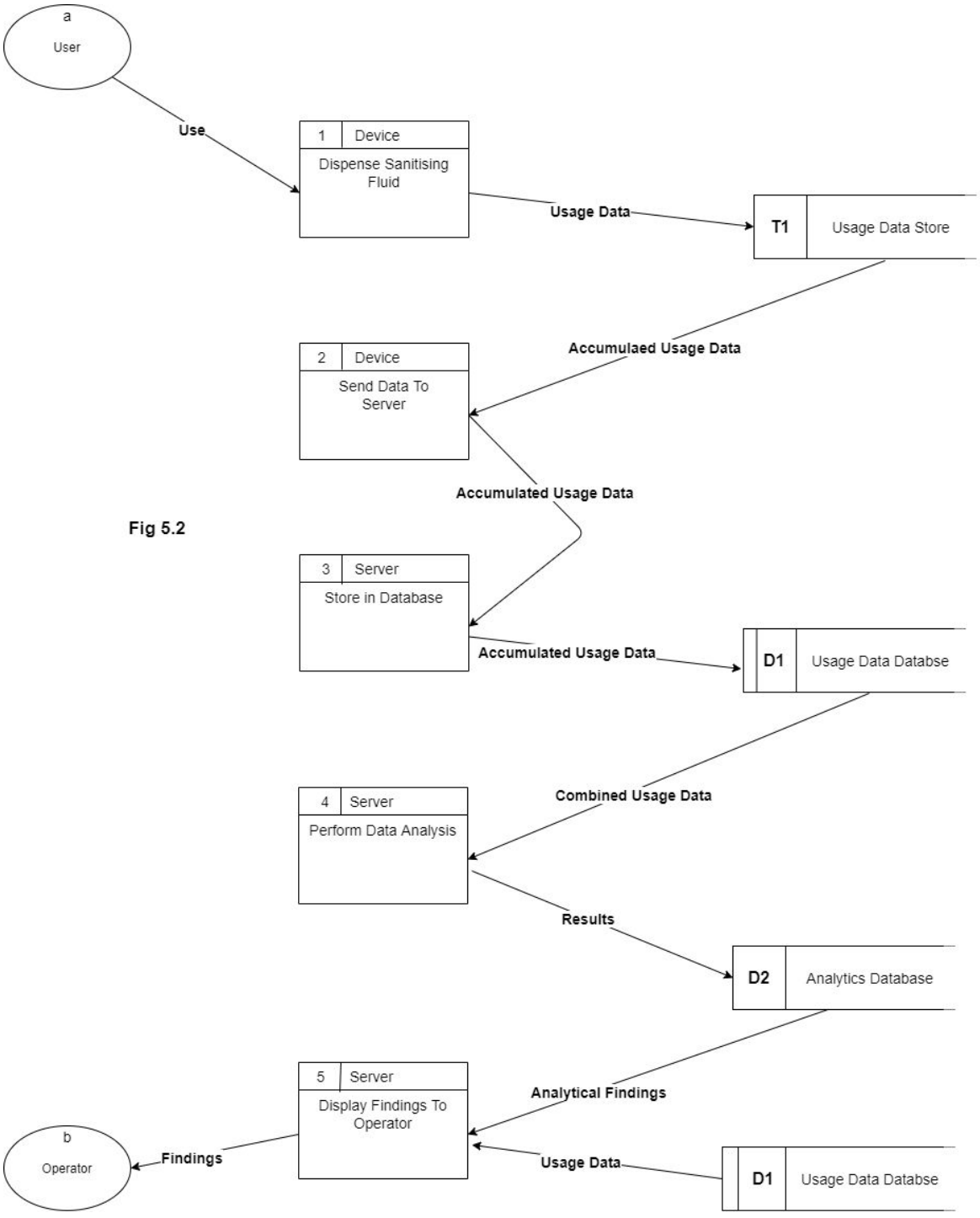
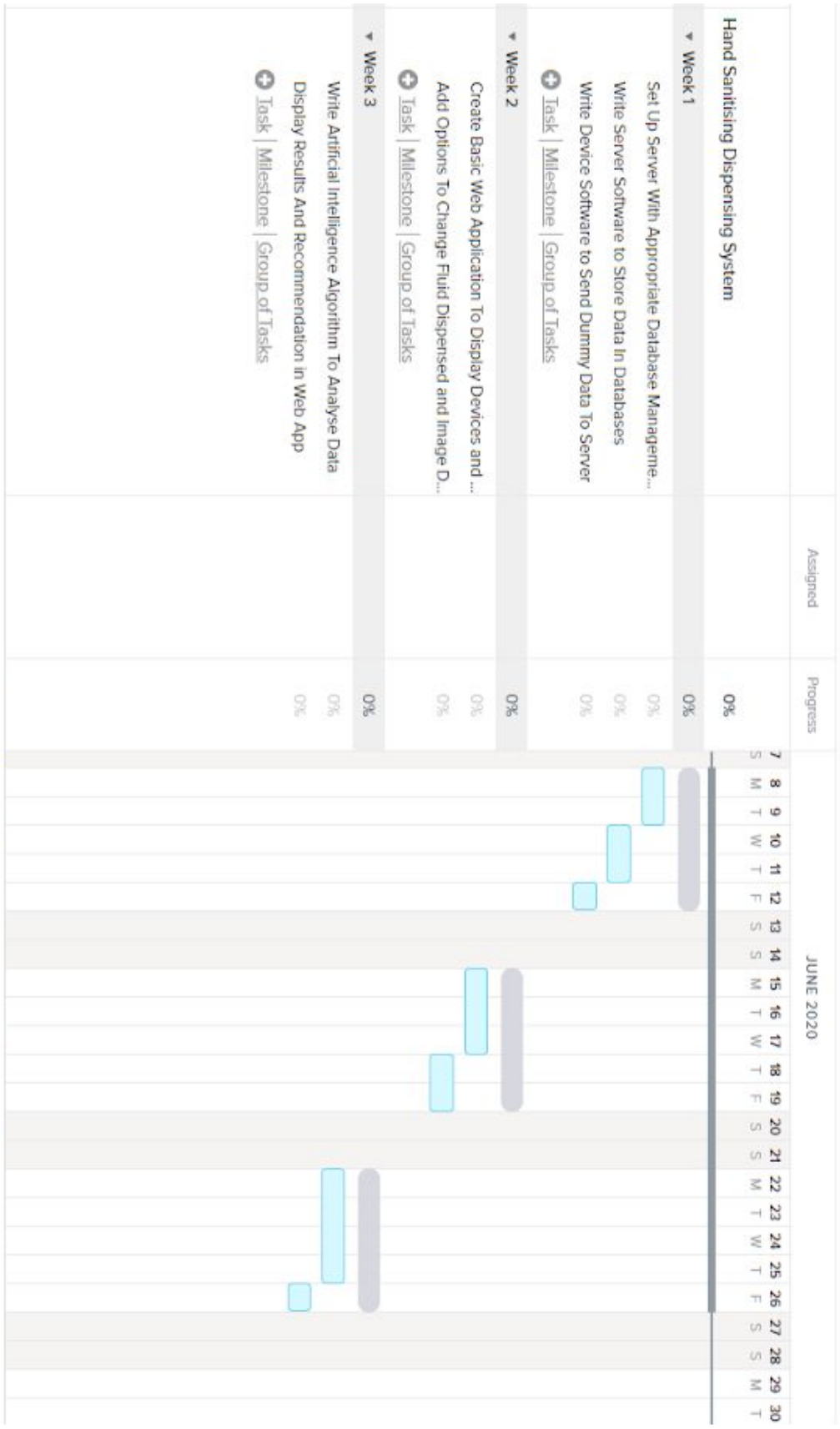


Fig 5.2

Fig 5.2 above is also a data flow diagram, illustrating how the device gathers statistics and displays them to the operator. Firstly, a user approaches the device and receives sanitising fluid. The usage data (eg time of use, number of uses etc) is temporarily stored on the raspberry pi device. This can happen many times with many users. After a period of time, the device takes the accumulated usage data from its temporary storage and sends it to the server. The server processes this data and stores it in the usage data table of the database. Data analysis is performed on the server, using the data found in the usage data table. The results are stored in another table of the database. The server fetches this data, along with

the accumulated usage data and displays this information to the operator via a web application.

6. Preliminary Schedule



The above gantt chart shows the preliminary schedule for completing the prototype of the software for the device, before the project deadline on Friday the 26th of June. The schedule operates under the assumption that the group will continue to have a singular representative from the School of Computing, Alison Kennedy.

7. Appendices

Design Brief as Found On Loop:

Your team is tasked with designing an intelligent automated dispenser for hand sanitising purposes. This smart dispenser is a networked device that is connected to a centralised management system. Therefore, this design challenge involves the input of different disciplines such as manufacturing specification and design, electronic/computer engineering, and network-oriented software design and development. As such you must work together as a team to identify how these different elements can be integrated to create a single platform.

The following functionality is required of the system:

- The physical device should be wall mounted or free standing, and allow for simple installation in general public utilities such as hospitals, schools, businesses etc. It should be designed to be modular and easily replaceable if parts are damaged.
- Users should be able to obtain the sanitising solution without the necessity to touch the device by using electronic proximity sensing.
- The device should be smart in the sense that it is networked and can be monitored and controlled remotely. A Raspberry Pi single board computer will be incorporated in the system for the purpose of reading sensors, controlling dispensing motors/valves, and interfacing the device to the internet.
- The device should dispense a finite volume of sanitizing fluid. This volume can be set remotely via networked communications.
- The device should report on usage statistics to a centralised networked service. For example, the device could record the time of use and volume of fluid dispensed each time it is used.
- The device should be designed so that it is easy to refill the system.
- A warning should be issued by the device to a centralised network service over the network when the level of sanitising solution falls below a

predetermined level, which is based on the usage levels for that particular device.

- The device should monitor when a person is in close proximity (e.g. 2m) and should flash a beacon to highlight the presence and purpose of the system in order to act as a reminder for people to use the dispenser.
- The device should gather data so that data analytics can be conducted offline to assess the efficacy of the system as a method of sterilisation. (for example, how often is the dispenser ignored and at what times etc.)

ISTQB Defect Severity Scale:

ISTQB Definition
severity: The degree of impact that a defect has on the development or operation of a component or system.

Classification
The actual terminologies, and their meaning, can vary depending on people, projects, organizations, or defect tracking tools, but the following is a normally accepted classification.

Critical: The defect affects critical functionality or critical data. It does not have a workaround. Example: Unsuccessful installation, complete failure of a feature.
Major: The defect affects major functionality or major data. It has a workaround but is not obvious and is difficult. Example: A feature is not functional from one module but the task is doable if 10 complicated indirect steps are followed in another module/s.
Minor: The defect affects minor functionality or non-critical data. It has an easy workaround. Example: A minor feature that is not functional in one module but the same task is easily doable from another module.
Trivial: The defect does not affect functionality or data. It does not even need a workaround. It does not impact productivity or efficiency. It is merely an inconvenience. Example: Petty layout discrepancies, spelling/grammatical errors.

Source: http://softwaretestingfundamentals.com/defect-severity/