The design of a web application to act as a hub of services for the transgender community in Liverpool

Lucy Alexandria Morris – G4272527

TM470 – TMA03 – 4th July 2023

A person holding a flag

Description automatically generated with medium confidence

Figure A transgender flag being waved at LGBT gay pride march by ‘ink drop’ used under Standard License from Adobe

Contents

[1. Draft Project Report 3](#_Toc135142369)

[1.1 Problem description 3](#_Toc135142370)

[1.1.1 Title 3](#_Toc135142371)

[1.1.2 Description 3](#_Toc135142372)

[1.1.3 Resources 4](#_Toc135142373)

[1.1.4 Future plan 5](#_Toc135142374)

[1.2 Account of related literature 5](#_Toc135142375)

[1.3 Account of Project Work 5](#_Toc135142376)

[2. Review 6](#_Toc135142377)

[2.1 Review of project work 6](#_Toc135142378)

[2.2 Review of project management 6](#_Toc135142379)

[2.3 Risks to project completion 6](#_Toc135142380)

[2.4 Review of personal development 6](#_Toc135142381)

[3. References 6](#_Toc135142382)

[4. Appendix 6](#_Toc135142383)

# 1. Draft Project Report

## 1.1 Problem description

### 1.1.1 Title

The design of a web application to act as a hub of services for the transgender community in Liverpool

### 1.1.2 Description

There are many vital services available to the transgender community in Liverpool, but it can be a challenge to find them. This is particularly the case for new members of the community who may be in a vulnerable situation, as the point of coming out is a huge life change that may be accompanied by mental health issues and the loss of relationships. This is compounded by transgender healthcare facing profound ignorance (and in some cases outright bigotry) from many GPs, who may attempt to block or delay attempts at medical transition, alongside extreme waiting lists for appointments at Gender Identity Clinics.

The trans community attempts to solve these issues in numerous ways, mostly informal (e.g. by word of mouth in group chats), but two more formal ways are the Liverpool Trans Wiki (TransLiverpool Wiki, 2023) which catalogues and comments on many services that are available; and the Spirit Level peer support group which invites in guests from services to explain what they offer to the community. The proposed web app will seek to build on these solutions and incorporate this specialist knowledge into it.

The web app will be usable on web browsers and on mobile devices using HTML, CSS, and JavaScript and will utilize the OpenStack Trove DBaaS, consisting of several pages. Firstly, a page that contains a map which displays services on it, which will be filterable by type (e.g. mental health or peer support) and when a service is selected more information and contact details will be provided. The map will also provide directions from the user’s location to the selected service. Another page will contain a search function for a database which will return services based on user queries. There will also be a page that will contain an events calendar. The app will be designed such that information will be one way, to protect the user’s privacy and limit the amount of sensitive data stored by the app. Ethical considerations are vital for the project, since some users may not wish to be open about their trans identity and therefore would need to conceal their use of the app. This must be at the forefront of the final product to reassure users that their identity is safe.

The requirements elicitation will also explore another feature, a map which displays the location of gender-neutral toilets at businesses such as cafes, restaurants, and pubs. However, this will be considered for further work beyond this project and will be considered out of scope to keep the amount of development to an achievable level. The scope of the project is for services in Liverpool & Merseyside due to the ability to engage with services and their users directly and due to already existing knowledge. However, if successful the app could be expanded to cover the whole of the UK, it would be a matter of gathering the data rather than any technical challenges. Also out of scope would be online only services that have no physical footprint in Liverpool & Merseyside, some of these may be included in an ‘important links’ page but not as part of the core services covered by the app. The scope of the services included is that they must offer something specific to the trans community, rather than broader services (e.g. the CMAGIC & TSS counselling services would be included, but not the general NHS counselling service offered by Talk Liverpool.)

See if this need any additions in light of work done

Analysis of impact?

Review questions asked in TMA question

### 1.1.3 Resources

The resources focused on previously involved gathering feedback from services and members of the community, and while this is very important for a full release, for the purposes of this project it has been scaled back. While feedback is important, too much time focused on it would not leave enough time for other aspects of development. This aspect of the resources list has been condensed and other essential resources added.

* Representatives of services e.g. manager of a laser hair removal clinic
  + Can offer feedback about the events system and the ways in which users can contact their service.
  + May offer vital perspectives of how the app effects the community
* Members of the Liverpool trans community
  + Different members of the community may have different needs regarding accessing services.
  + Could offer feedback on prototypes of the app.
  + May be a source of information about services that should be included.
  + The Liverpool Trans Wiki may an essential source of community information about the services, as it documents many of them.
* Programming languages
  + JavaScript – proficient enough to tackle much of the proposed features, but some learning may be required as it will likely extend beyond current knowledge.
  + HTML – reasonably proficient, significant problems are not anticipated.
  + CSS – only minimal experience, may need some time to learn and to use trial and error to achieve goals.
  + SQL – a small amount of experience that should be sufficient for a simple database, but for anything more complex some learning may be required.
* Cloud storage and version control
  + Microsoft OneDrive and GitHub can be used to backup files in the cloud and continue work between different devices. GitHub also provides version control with branching and reverting.
* DBaaS options
  + OpenStack Trove – A database solution, which is free and open source and will be used to store all the data for the services.
  + AWS DynamoDB – A serverless database solution from Amazon that has a limited free access that is accessed through an online GUI.
* Hardware
  + PC – used for coding, testing, writing up reports and conducting research.
  + Laptop – as above, but with portability. More focus on writing up reports, since other tasks are easier with a larger screen.
  + Phone – used for support with research, but also to test the app on a mobile device.
* Software
  + Chrome – Browser used for testing with developer tools, for investigating issues with code and to see how the app responds to different resolutions.
  + Visual Studio Code - A commonly used code editor that will be used throughout the project for all coding purposes.
  + Postman – for testing and using API’s.
  + CURL (Client URL) – for testing and using API’s using the command line.

### 1.1.4 Risk management

The goal of risk management is to identify, mitigate and minimise risks prior to them becoming a threat to the successful completion of the project (Hughes, 2012). The first part of managing risk is to identify and assess potential risks, considering how likely they are to occur and the severity of their impact. Table x (Open University, 2023) considers risks for the project and labels them so they may more easily be referred to for the risk management process.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Label | Project Activity | Risk Description | Likelihood (low, medium, high) | Impact (low, medium, high) |
| R1 | Feedback from service providers for requirements elicitation and prototyping | Providers may be busy and not wish to engage with giving feedback or may only engage in a limited way. | Medium | Medium |
| R2 | Feedback from service users for requirements elicitation and prototyping. | The feedback given by users may be not useful or relevant if questions don’t properly direct users or may miss vital feedback if questions are too closed. | High | Medium |
| R3 | Feedback from service users for requirements elicitation and prototyping. | Users may not wish to engage with giving feedback or may only engage in a limited way. | Low | High |
| R4 | Coding the application. | Attempting to code non-routine tasks may cause significant delay. | High | Medium |
| R5 | Using Visual Studio Code | Since this is the first time using this software, there may be some adjustment time to using it causing delays. | Medium | Low |
| R6 | Storing data in the cloud | Data loss in the cloud, which could be due to several reasons including accidental deletion and server failure. | Low | High |
| R7 | Installing and setting up a database | The task may take longer than anticipated, since previous work with databases was after it had been set up. | Medium | Medium |
| R8 | Unexpected Illness | Time could be lost to illness, including potentially at a critical time such as in the run up to a TMA deadline. | Low | Medium |
| R9 | Hardware failure | A sudden mechanical failure requiring the repair or replacement of a piece of hardware | Low | Medium |
|  |  |  |  |  |

The next stage of managing risk is to identify which risks are most important to address, so the likelihood and impact of each risk have been combined in a probability impact grid, as shown by Hughes, but instead of a ‘Line of Tolerance’, the grid (shown in table x) has been colour-coded to indicate the most crucial risks to address with red and orange being most critical and to be addressed with urgency and in detail; yellow important to be addressed as much as feasible; green being least important and the risk may be accepted or only convenient mitigations undertaken. Risks 2 and 4 have been identified as the most crucial and mitigating them will form notable parts of the project; whereas risks 5, 8 and 9 have been identified as the least crucial and will be accepted without mitigation.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Impact | High | 3, 6 |  |  |
| Medium | 8, 9 | 1, 7 | 2, 4 |
| Low |  | 5 |  |
|  | Low | Medium | High |
| Probability | | | |

Table x shows the mitigating actions that were taken for each risk and explains what happened. In most cases the mitigating actions were effective …

|  |  |  |
| --- | --- | --- |
| Label | Mitigating Actions | Result |
| R1 | Identify people most likely to provide feedback and be respectful of their time to get as much out of any engagements as possible | The most likely service provider to provide feedback was identified and engaged with, but personal reasons meant they had to drop out of the requirements elicitation study. Since mitigation failed adjustment to the analysis was necessary, so the requirements elicitation still went ahead without this stakeholder group and future work suggested to engage with other stakeholders. |
| R2 | Questionnaire to be carefully designed to facilitate getting high quality feedback. | The feedback was good quality and useful information was gained, so mitigation of carefully considering the questionnaire was successful. |
| R3 | Undertake feedback gathering as soon as possible, attempt to identify most likely candidates to participate and consider changes to project lifecycle if necessary. | This partially occurred, with less users participating than anticipated; however, this only had a minimal effect on the analysis and changes to the project lifecycle were made to reduce reliance on gathering feedback. |
| R4 | A skills development plan will be implemented, particularly for CSS & databases. | Skills development was undertaken for CSS and using the AWS services such as DynamoDB. |
| R5 | No mitigation to be undertaken. The primary purpose of the software is very similar to other code editors, and any shortcuts learned will only speed up tasks. |  |
| R6 | Data will be stored locally on two machines (PC & laptop) as well as in the cloud, so there is no single point of failure. |  |
| R7 | Begin working on this early in the project to ensure there is time to work through any issues. | This occurred due to the time scale involved of setting up Trove, so there was a switch to AWS. This was mitigated by starting the databases relatively early in the project, so disruption was minimised. |
| R8 | No mitigation to be undertaken but contact with tutor to be maintained so that an extension can be requested if necessary. | Unexpected illness caused a week’s delay around the time of a TMA deadline, but an extension was requested and granted, so disruption was minimal. |
| R9 | No mitigation to be undertaken, if either the PC or laptop were to suddenly fail then the other could be used as backup. |  |

### 1.1.5 Future plan

Discuss changes to plans, doing CSS 1st, changes to feedback

Skills development plan, CSS, AWS

Final part of plan TMA02->EMA?

Maybe appendix this? Or move to review?

## 1.2 Account of related literature

A previous literature review concluded that a NoSQL database would be a suitable for the project, and that OpenStack Trove would be a good choice due to previous familiarity gained from TM352, as well as it being a free open-source option. Another possible choice considered was AWS (Amazon Web Services) which has a NoSQL database DynamoDB, but concerns were around cost as AWS charges for some of its services and there is no budget for this project. The familiarity with OpenStack Trove was with the use of the database, not its installation and setup, so some skills development was required first, and the literature review will cover sources used for this purpose.

Skills development began with research into how to install and setup OpenStack Trove and in addition to official documentation for Openstack (2019), as well as a video guide from Tesora (2015) a DBaaS provider.

OpenInfra + freeCodeCamp

One thing that became immediately apparent is that OpenStack is not available for Windows and must be installed on Linux, so would require either an installation of a dual boot or virtual machine for Ubuntu 16.04. The installation would therefore require the following steps:

1. Create a VM (or duel boot install) for Ubuntu 16.04
2. Install OpenStack using the Linux CLI (command line interface)
3. Install relevant packages including Horizon, the OpenStack GUI
4. Install Trove

Each of these steps requires different guides to follow and while certainly achievable, may require further skills development. While I have installed an Ubuntu VM previously, I am aware that many other students quickly ran into issues during installation and struggled for a long time to get it to work. Further to this, while I have a little experience using the Linux CLI I may require some refreshing on this as I usually work with Windows. Therefore, given the time frame of this project alternatives were considered that may have a more speedy set up time.

The other option considered previously was AWS DynamoDB, which in comparison to OpenStack is very straightforward to begin using, being a matter of creating an account and going through some confirmation of credentials. DynamoDB can then be interacted with through the AWS GUI on a browser. The first challenge of using DynamoDB is to ensure that the ‘free tier’ is stuck to, so that inadvertent charges are not incurred. The AWS free tier provides 25GB of storage as well as 25 Write Capacity Units and Read Capacity Units (<https://aws.amazon.com/dynamodb/pricing/> ). Given the relatively small amount of data that will be required for this project, this will be sufficient. However, a full release must keep this limit in mind, and either consider monetisation to recoup the costs, or switch to a free alternative.

Learning how to use DynamoDB:

freeCodeCamp – DynamoD Tutorial: Basic Operatios

Be A Better Dev – AWS DynamoDB Tutorial For Beginners

Tables, partition key, sort key

Whilst using AWS does not require a lengthy and complicated setup process, it does present other challenges. The first of which, is setting up the APIs, as although AWS provides low level APIs these can be cumbersome to use and it is recommended that using the AWS SDK (Software Development Kit) is a better approach (AWS reference). The involves:

1. Create a table in DynamoDB (the database)
2. Create functions for CRUD operations in AWS Lambda (another AWS service – explain?)
3. Create the APIs using the AWS API Gateway service (explain?)

Amazon (reference) provides a step-by-step tutorial for this process, which was followed as part of skills development and provided a basis on how to set up the APIs for the project.

Check what is needed in this section – is it a new lit review or collecting previous lit?

1 – databases / API

Literature for openstack and subsequent decision to change to aws

Literature for aws and skills development

## 1.3 Account of Project Work

The first task was to set up a DynamoDB table (the terminology used for the database) and the API’s so that the database can be accessed. While it is possible to access the database using ‘low level API’s’, i.e. constructing the API requests manually in the correct format with a valid digital signature, it is considered much simpler to use the AWS SDK (Software Development Kit). The AWS SDK constructs the requests and converts the responses on your behalf, and the structure of the SDK between the application and DynamoDB is shown in figure x.

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Programming.LowLevelAPI.html>

A diagram of a software application

Description automatically generated with low confidence

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Programming.SDKOverview.html> (SDK image)

Amazon provides step by step instructions on how to build a CRUD API (i.e. with operations to Create, Read, Update and Delete), which also utilizes other Amazon services; Lambda and API Gateway. Firstly a table is created in DynamoDB, then create a function in AWS Lambda which serves as the back-end of the API. Lambda runs code without provisioning or managing servers, and the code for the function is provided in the tutorial but then edited to reflect the table name and the ‘put’ operation to include the columns of the DynamoDB table. However, the ‘put’ operation is likely not to be used by the app as currently planned, and the table can be populated by using the AWS Dashboard GUI. Figure x shows the edited put operation.

A screen shot of a computer code

Description automatically generated with low confidence

<https://docs.aws.amazon.com/apigateway/latest/developerguide/http-api-dynamo-db.html>

Then the HTTP API’s are created in AWS API Gateway, providing an end point for the function that has been created in Lambda. Routes are also created in API Gateway with integrations that connect with the Lambda function. This can then be tested using the CURL command line tool, and figure x shows the use of the PUT function adding 3 items to the DynamoDB table with the result shown in the AWS Dashboard GUI in figure x and with two GET requests in the command line, one for all the table and one for a specific ID. The item with ID = 124 is deleted using the DELETE command and the result is shown again using a GET request and in the GUI in figure x.

![A picture containing text, screenshot, font, black and white

Description automatically generated]()

A picture containing text, receipt, screenshot, font

Description automatically generated

A picture containing text, receipt, screenshot, font

Description automatically generated

The next stage was then to implement the map and pins. Both Google Maps and HERE Maps would be reasonable choices for this project, HERE Maps was chosen due to previous experience working with it in TMA352. HERE provide a number of code examples which provided a good basis to start off working with the map, such as to show how to initialise the map, centre it on a specified location, restrict the movement of the map to within specified boundaries and place a marker at a specified location. (further explanation here referencing images below).

(show code snippet and image of functioning map/pin)

<https://developer.here.com/documentation/examples/maps-js>

Accessing the API for the map requires creating an account with HERE, registering an app with them and then generating an API key which can be inserted in the code as shown in figure x.

A screen shot of a computer

Description automatically generated with low confidence

The next step was then to use a GET request to access the DynamoDB table to find location data for services and then put a pin at that location. The first part of this was to

Find step by step guide for implementing cognito, user pools etc.

# 2. Review

## 2.1 Review of project work

Reflection on process and what has been achieved

Not analysis of what is covered on TMA01/02

Discuss strengths e.g. lit review

Weaknesses so far include planning / skills dev plan

## 2.2 Review of project management

Short account of project lifecycle being used – perhaps cover changes to it here

Not a general discussion of lifecyles

## 2.3 Risks to project completion

Risks initially identified, strategies to mitigate them and how effective they were

Any risks that may still occur and how they may be mitigated

## 2.4 Review of personal development

What I have learnt so far, effectives ways to work and learn

What I need to extend current knowledge

Reflect on skills acquired or improved – project management and self management

Look back at TMA01/02 here

# 3. References

# 4. Appendix