# Mobile-based Virtual traffic lights

## Research Topic, Scope, and Role

With the population constantly rising, not only just in New Zealand, but worldwide, which in turn generates more traffic congestion, henceforth the terminology of “Rush Hour” or “Peak Hour” can be searched upon as well as being commonly socially used. Traffic congestion is a never-ending issue that is still being investigated even as of today. “Traffic congestion has become a common and hard-to-solve social illness of many cities in the world today. It is one of the ultimate objectives of transport researchers, administrators and practitioners to counter the occurrence of traffic congestion and provide the cities with relief from recurring congestion.” (Feng, Saito, & Liu, 2015).

Ma, Xiao, & Ma, (2020) believe that the problem can be approached from two perspective which consists of increasing its network capacity by expanding or adding more roads or enhancing the effectiveness of the existing traffic signal control systems. The first approach is not always a feasible solution as it still encourages vehicle use, however the second perspective has been seen a good measure that improves the network performance and congestion, leading to many cities having installed advanced traffic signal control systems (TSCSs) (Ma, Xiao, & Ma, 2020).

Thus, this research topic will be focused on Virtual Traffic Lights (VTL) in the base form of mobile, where its primary goal is to mitigate traffic congestion. Necessary and relevant research and development will be addressed.

### Scope

The technology of VTL will be mobile based as it will benefit them mostly. Ideally the functionality will be focused on urban intersection, where it will safely prioritize the most populated section to let traffic flow through, whereas the least populated will have to wait slightly longer according to the congestion information given, upon the results of each green or red path that crosses the intersection. Phones or even cars itself should send out a radio wave frequency to the traffic lights itself, to calculate which section has the most vehicles. Although the primary focus will have to be on mobile phones, specifically iOS and Android as cars that are considerably old will be neglected due to its age and technology.

#### Development Scope

The idea behind this application is to try and keep it simple looking whilst being very easy to use as majority of the time this app will be very similar to a navigation system where it is handsfree most of the time.

* Appealing UI (User Interface)
  + Enticing user friendly experience that is targeted for all age groups, such as bigger appealing pictures and text to help users navigate easier.
  + Familiar experience which will result in easy to understand and usage
  + Minimal effort for interaction
* Usage of Google Maps API
  + Google maps is widely well known and trusted, giving it the familiar feel will enable users to feel more trusted towards the application.
  + Google Maps API uses Javascript API which enables customization of online maps ready to be used by external sites (Wang & Xu, 2011).
* Progress bars for wait time of lights. Traffic light status
  + A progress bar will be implemented in the navigation page to give the user an idea on when the lights will turn green.
  + A traffic light status indicator
* Users within the vicinity
  + User count within the vicinity will be displayed.
  + No user data will be displayed.
  + A user avatar will be optional to give it more of an appeal
* Traffic lights
  + Respond to congested lanes and treat that as priority.
  + Be in sync with the progress bar for users who have it displayed on their device, as well as being accurate with timing expectations
  + Prioritizing pedestrians if certain amount of people is waiting during rush hour. E.g., If more than 10 people waiting to cross, then they will also be given priority.
  + Sensors to detect if accident occurred to re-assess the priority order.
  + Sensors to detect cars and signals from either mobile device.

### Roles

#### User Interface / User Experience Designer (UI / UX Designer)

Designers who create the visuals of how the application should be as well as how it should navigate and behave. They mostly collaborate with the product managers and software developers to gather design ideas and requirements, then proceed with design concepts. (Resources for employers, n.d.)

#### Developer

Also known as a coder, programmer, or a software engineer, who develops (write), test, debugs, and maintains code or applications. (Samuels, 2021)

#### Product Manager

A person who identifies business objectives as well as customer needs and requirements. Visualizes what the success outcome will look, then gathers a team and turns that vision into existence (Mansour, n.d.)

#### Selected role

I will be taking upon the role of UI (User Interface) UX (User experience) designer for this project due to the complexity nature of this application and the limited time frame. “UX is as much about the user’s journey as their destination, with designers carefully thinking through user needs and use case scenarios to ensure a productive and satisfying experience as users work towards their goals” as said by Loeffler, Roth, Goring, & Myrbo, (2021).

## Development Process

### Scrum

Scrum is a type of Agile software development methodology, that has various small teams cooperating in an intensive manner (Dada & Sanusi, 2021). The team usually consists of Product Owner, Scrum Master, and Team Members. The product owner liaison between the customer and the team, whilst being responsible that the product asked for is being delivered in the expected agreement as shown in figure XX. The Scrum Master ensures that the practices of scrum are being as followed, therefore they must have good leadership skills, collaboration, able to resolve conflicts and skilled at process improvements (Lutkevich, 2021). The team members are responsible for creating and testing the product, which generally consists of software engineers, systems admins, UI UX designers and so on (Sergeev, 2020).

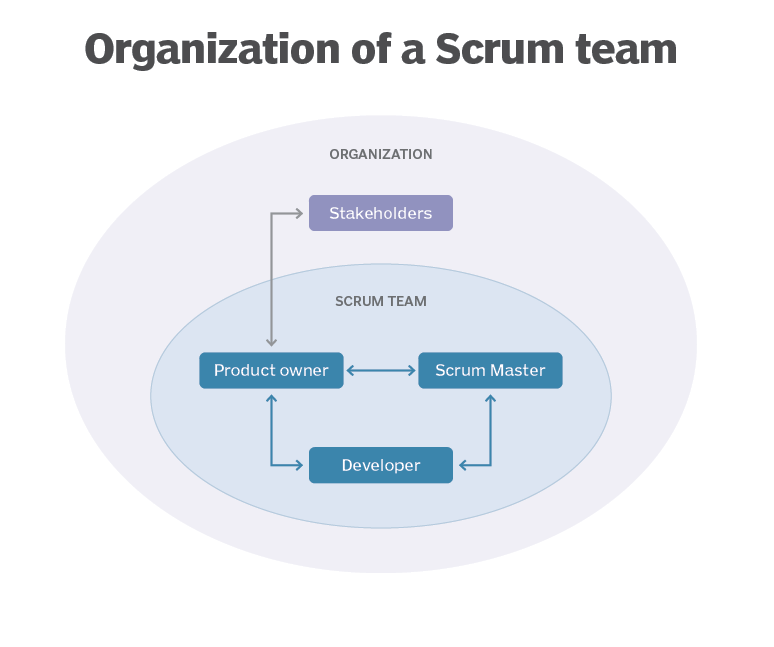


Figure Organization of Scrum Team (Lutkevich, 2021)

Scrum generally begins with a vision of the system or product that is to be developed, followed by a list of user requirements that is currently known, which is also known as product backlog (PB). The product backlog is a list of features and requirements that are necessary to complete certain tasks of the project, or the project itself. The list can contain various tasks such as resolving defects or bugs, implementing new features or other project relating work, thus the order of the list is also based on its priorities (Sergeev, 2020). All tasks are usually completed within each sprints, that are iterations of one to four weeks. “Every Sprint starts with a Sprint planning meeting, where the Product Owner (user representative) and the team meet to discuss what must be done for the next Sprint. This translates into the next Sprint Backlog (a list of tasks that must be performed to deliver a completed increment of potentially shippable product functionality by the end of the Sprint).” As explained by Dada & Sanusi, (2021). The scrum meetings are usually daily which can last up to 15 minutes, in order to stay synchronized among everyone within the team. The sprint review is where everyone gets together and revises what has been completed within that sprint period, and demonstrates it to product owner who also advises the stakeholder. After the conclusion of the sprint review, the sprint retrospective is called upon, where the scrum master helps the team reflect on their development process within that sprint, and where applicable adapt these reflections to become more efficient and effective for the next upcoming sprint (Dada & Sanusi, 2021).

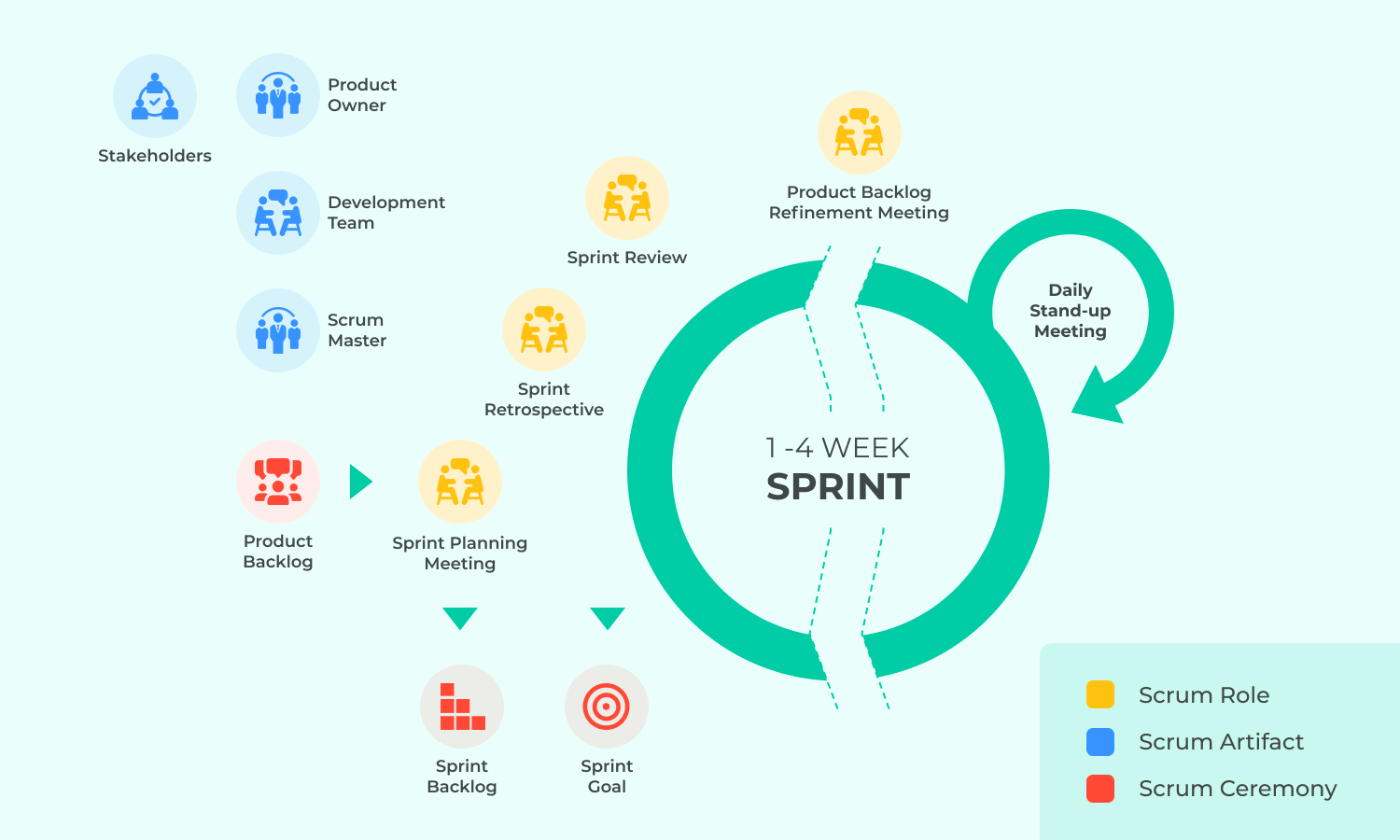


Figure Scrum Cycle (Sergeev, 2020)

### Waterfall

This methodology tackles the project in a linear progression, where each stage within the workflow must be completed before moving onto the next, as shown in figure X. This methodology is only ideal if everything carefully and well outlined from the beginning (Hoory & Bottorff, 2022). As it follows a chronological process, team members are not required to be in continuous communication, enabling them to work more independently and are not expected to provide a status report as often (Workfront, n.d.)

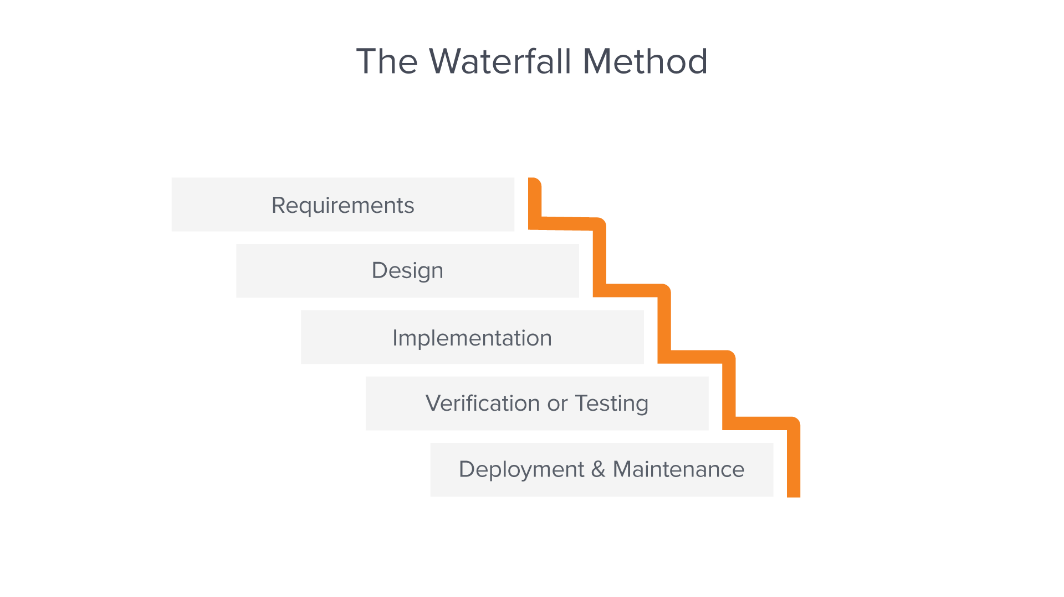


Figure Waterfall Method (Workfront, n.d.)

Requirements is the most important aspect of the entire waterfall methodology hence it being on the very top, as shown in Figure X. This is where the initial planning and project requirements gathering is done, and where it should be understood upfront among all team members before embarking onto the next step.

Design is where a technical solution is presented from the given details in the requirements stage, which can consist of layouts, scenarios, and data models (Workfront, n.d.).

Implementation is where the selected design is applied. Generally, with the chosen technology the software developers write the code and implement the system during this stage of the waterfall process (Dada & Sanusi, 2021).

Verification or testing is to ensure that the product is ready for the user or public, by testing the product and guaranteeing that there are no bugs as well as meeting the requirements, generating a good user experience with the software (Workfront, n.d.).

Deployment and maintenance are where the product is successfully deployed to the market, however the developers must vary unforeseen errors or so, and resolve them as soon as possible as well as upgrading the software if need be.

### Kanban

Kanban methodology is also a type of agile methodology, where it aims to generate improvement, flexibility within task management, and enhanced workflow (Kissflow, 2022). Kanban is more of a tool than it is a framework, where it has the transparency of the workflow on a board, hence the name Kanban, which means “Sign board” which originated in Japan by Taiichi Ohno during the 1940s in Toyota (Aurisch, Ahmed, & Barkat, 2019). This board is useful if a new member were to join the project and can easily understand what is happening, or see what is currently happening, as shown in figure x.

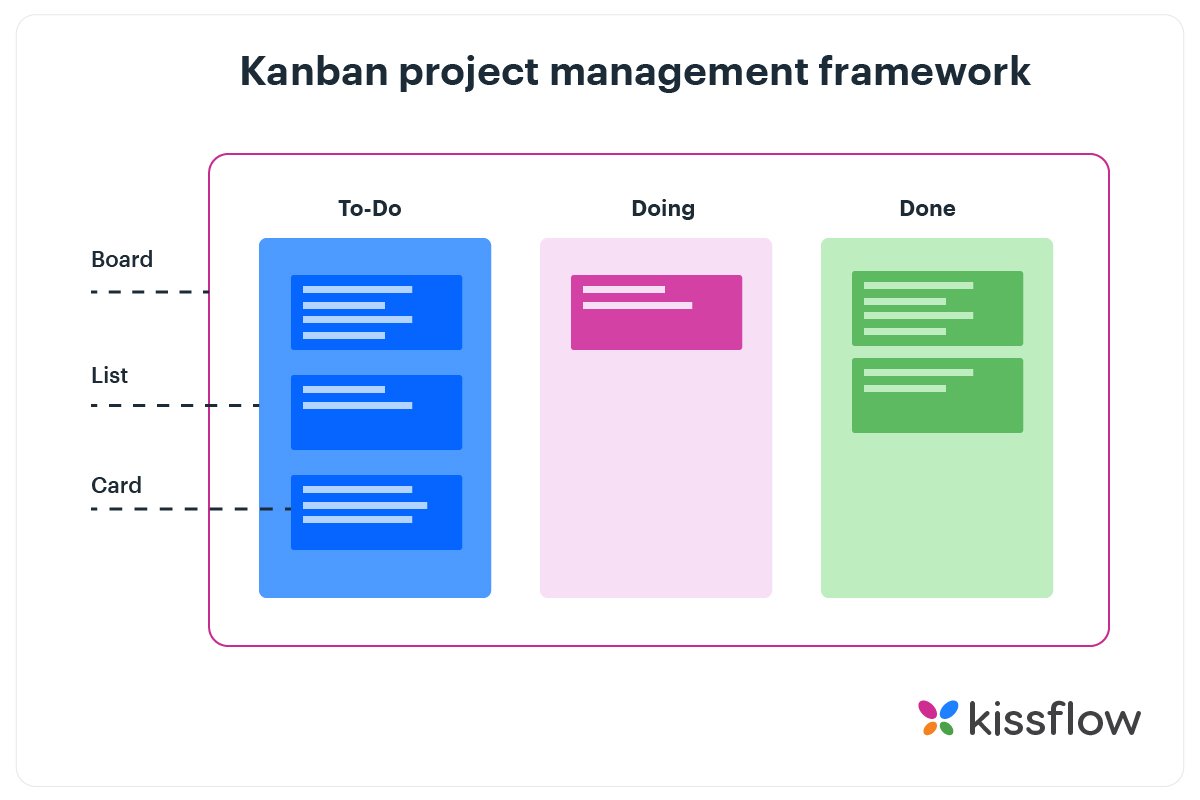


Figure Kanban Example (Kissflow, 2022)

### Chosen development process

Hanssen, Haugset, Stålhane, Myklebust, & Kulbrandstad, (2016) states “We also see that Scrum is being used for development of safety-critical systems, which have to comply with strict quality and safety standards.” There will always be a backlog of functional requirements that are likely to change and grow overtime, whilst safety requirements are reusable and normally static between products and projects (Hanssen, Haugset, Stålhane, Myklebust, & Kulbrandstad, 2016). Interaction between the PO, stakeholders, and developers will be welcome to the project as there could be unforeseen changes that are to occur, and safety should always be prioritized, although majority of the time it will be static.

Virtual traffic lights has to be considered a safety application because it is being used as a guidance, help speed up traffic lights, as well as making it prioritize properly, while safely letting pedestrians and drivers alike through, in a safe manner with absolute zero damage if possible. Therefore, all safety procedures, design concepts, and coding must be taken priority when developing and designing.

Scrum will be the ideal development process because of the nature of the application where it will involve safety. Whilst Waterfall has proven to be effective on other projects, it will not be effective here as it is difficult to respond to changes if they were to happen (Dada & Sanusi, 2021).Kanban will be more ideal if the project was more then halfway or so, however this project will be from scratch thus scrum will be more ideal, due to it having multiple sprints, interaction between stakeholders, and a reflection process to improve development methods within the project. “Kanban has a lack of framework which ultimately means that it does not provide a planning routine when it comes to estimations.” (Aurisch, Ahmed, & Barkat, 2019). Although majority of Scrum methodology will be used, the role PO will be merged with the team member role. This is so the designer or developer can work closely alongside the client to understand what actual users want to see and how it be used, as it will be majority of the time hands-free (driving scenario). The sprints will remain the same, however the sprint retrospective will be done at the end of each week, to improve team collaboration. As mentioned, all safety features will be thoroughly inspected, and thus this mythology will be known as “Intense Scrum”.

### Development timeline plan

Timeline

Description automatically generated

Figure Gantt Chart

As shown in figure 5, the little time frame to design this application is very limited, hence the sprint retrospective at the end of each week. The final deadline is on October 28th 2022, however it must be completed before then therefore it is decided to have everything finalized by the 27th of October 2022. First it will begin with upskilling in terms of using Adobe XD, and creating the git repository. User stories will then be collected, and using that feedback, the design will then be looked into. Once the design is complete, it will then be exported and coded and check the functionality of the application. It will have thorough testing and revision, developing multiple versions. Alongside the testing will also be the report so some form of documentation exist.

## Technology

There are numerous technologies out there that is currently being used and can differ from each industry. The three technologies are cross platformed based, where it can save on cost when developing the app, as well as code maintenance, but ultimately, it being functional on popular mobiles such as Android and iOS. Native app development could have been considered for its ideal user experience, but it can only focus on one OS at a time, losing a lot of development time, thus a cross platform will be more ideal for the VTL application.

### React Native

React Native is an open-source framework developed by Facebook, focusing on cross platform mobile development. It enables the developer to be build the application simultaneously for Android and iOS operating systems, as it only uses one codebase. “React Native is a perfect technology to save on development costs, yet still have truly native performance and user-experience flexibility.” As said by Marcak, (2022). Facebook is known for being committed in delivering a better mobile experience, and always looking for a solution, therefore React Native is the proposed solution to assist other developers to build native mobile apps (Brewster, n.d.)

The learning curve for this framework is not too high as it universally uses the language JavaScript, as well as Java, C, C++, or Swift (Sakovich, n.d.)

React Native has been used in successful businesses or apps such as Discord, Facebook, Uber Eats, Pinterest and many more. There are mutiple reasons as to why they utilized React Native, such as it being robust, reliable, development time, and friendly community. “React Native is community-driven. Like many open-source frameworks that have gained popularity, React Native has many developers flocking to forum boards to seek knowledge when necessary or offer advice when needed.” As said by Brewster, (n.d.). React Native is maturing, and progressing faster than ever (Peal, 2018)

### Flutter

Flutter was initially launched in May 2017 (Alpha) then the stable release in December 2018, making it younger then React Native. However, “it didn’t take Flutter too long to gain incredible popularity, judging by the number of stars on GitHub (143k vs React Native’s 104k). And it keeps attracting more interest than rival technologies.” as stated by Altexsoft, (2022). Flutter began as a start up, which was then later accquired by Google. It is an open-source project which heavily relies on the language Dart. Flutter is an SDK (Software Development Kit) not a framework, and provides excellent components to make the app feel more native, which in turn results in a better user experience (Berka, 2020). However, Dart has a low adaption rate as it is not a popular language, as well of a lack of third-party libraries. “However, as of today, there are 25k+ packages on Flutter’s official resource and the number keeps growing. Of course, it’s less than its main competitor, React Native, has but it’s more than sufficient.” As stated by (Altexsoft, 2022).

With its growing popularity, it has been adapted to big famous clients such as BMW, Toyota, eBay, Alibaba and many more as they are focused on providing a great customer experience.

Other than cross-platform mobile apps, the SDK has found its way inn other aspects in terms of usage. The portability and fast development cycle, enables the technology itself to beccome suitable in building a minium viable product (MVP) as well as prototying, thus able to promptly experiment with business ideas on different platforms (Altexsoft, 2022).

### Xamarin

Created by Microsoft as an open-source app platform, to assist .NET developers building modern cross platform applications for Android and iOS. With Xamarin it is possible to share up to 90% of the codebase on other major platforms (Kamienski, 2022).

Although Xamarin is an extension of .NET, it’s focus is for mobile development. Xamarin heavily uses the language C# which was also developed by Microsoft. Using the .NET framework handles a lot of the common tasks such as garbage collection, memory allocation, and interoperability with underlying platforms (Almeida, 2022).

Xamarin is due for a reform in the near future, where community plugins will need to be updated. “In an effort to consolidate the .NET platform, Microsoft is developing the evolution of Xamarin.Forms, named .NET Multi-platform App UI (MAUI). Xamarin.Essentials is also being integrated into this library, becoming MAUI.Essentials. As of writing, this new library is scheduled to come out to the general public in Q2 of this year.” as stated by Almeida, (2022)

Xamarin has also seen a decline in popularity and usage by developers as shown in figure x. Manchester Digital (n.d.) also states “Unsurprisingly, most Xamarin developers seem to hate using it.”.

Graphical user interface

Description automatically generated

Figure Technology comparison 2020 (Manchester Digital, n.d.)

### Technology Chosen – React Native

With the decline of Xamarin, it is very unlikely that this will be selected as the chosen option. With a lack of community support, it will be difficult to develop the application. Although Xamarin is free and open source, the price can be costly if it was an enterprise setting (Almeida, 2022).

Flutter is also not considered due to the lack of community support and additional third-party libraries as well as the adaption rate with the language of Dart.

For VTL, a cross platform development will be preferred, as the application itself is ideally in the form of a mobile application, and unlikely to be used as a web app or so.

With React Native, always striving to improve mobile experience, it is very likely that this is the chosen technology. With a strong supporting community and a non-steep learning curve of the technology it is also another supporting factor. Another strong driving factor is a statement said by Samsukha, (2022) “React Native is remarkably compatible with third-party plugins like Google Maps.” As this application will heavily be using Google Map’s API, and having that reassurance that it is compatible will make things less difficult.

## Design of deliverables

### Logical Decomposition

Once a software has been deployed and delivered to the customer, it will undoubtedly still require updates and fixes of bugs that are to occur. Updating and rectifying bugs is basically called maintainability, and the task of predicting this maintainability is dubbed as Software Maintainability Prediction (SMP) (Gupta & Chug, 2020). Therefore, it is vital to predict maintainability as a significant portion of cost goes into software maintenance itself. “Software maintenance is required to fix bugs, to add new features, to improve performance, and/or to adapt to a changed environment.” (Midha, Singh, Palvia, & Kshetri, 2015).

MVC (Model-View-Controller) was originally for Small Talk programming language and was widely adapted in desktop applications during the 1980s to late 1990s. Initially it was known as Model-View-Editor, which was then later changed to MVC as we know today (Kanjilal, 2021). MVC consists of 3 components which are:

* **Model**: Responsible for the business logic of the application, such as writing or reading data.
* **View:** Presenting data to the user as well as handling the user input
* **Controller:** “The Controller is responsible to process incoming requests. It processes the user’s data through the Model and passing back the results to View. It normally acts as a mediator between the View and the Model.” (Sinhal, 2017)

MVP (Model-View-Presenter) is similar to MVC as it is derived from its designed pattern, however the controller is replaced by the Presenter. The **Presenter** retrieves data from the model and utilizes the UI logic to decide to what to display. The state of view is determined by the user’s input from the view (GeeksforGeeks, 2020). “The reason why MVP is widely accepted is that it provides modularity, testability, and a more clean and maintainable codebase” (GeeksforGeeks, 2020).

MVVM (Model-View-ViewModel) originated from Microsoft for the use with Windows Presentation (WPF) and Silverlight (Dang, 2020). The **ViewModel** serves as a link between the Model and the View as it exposes data streams which are relevant to the View. (GeeksforGeeks, 2020). “In MVVM, it is supposed to be more designer-friendly that could be easily implemented by designer instead of the code developer” as stated by Lou, (2016)

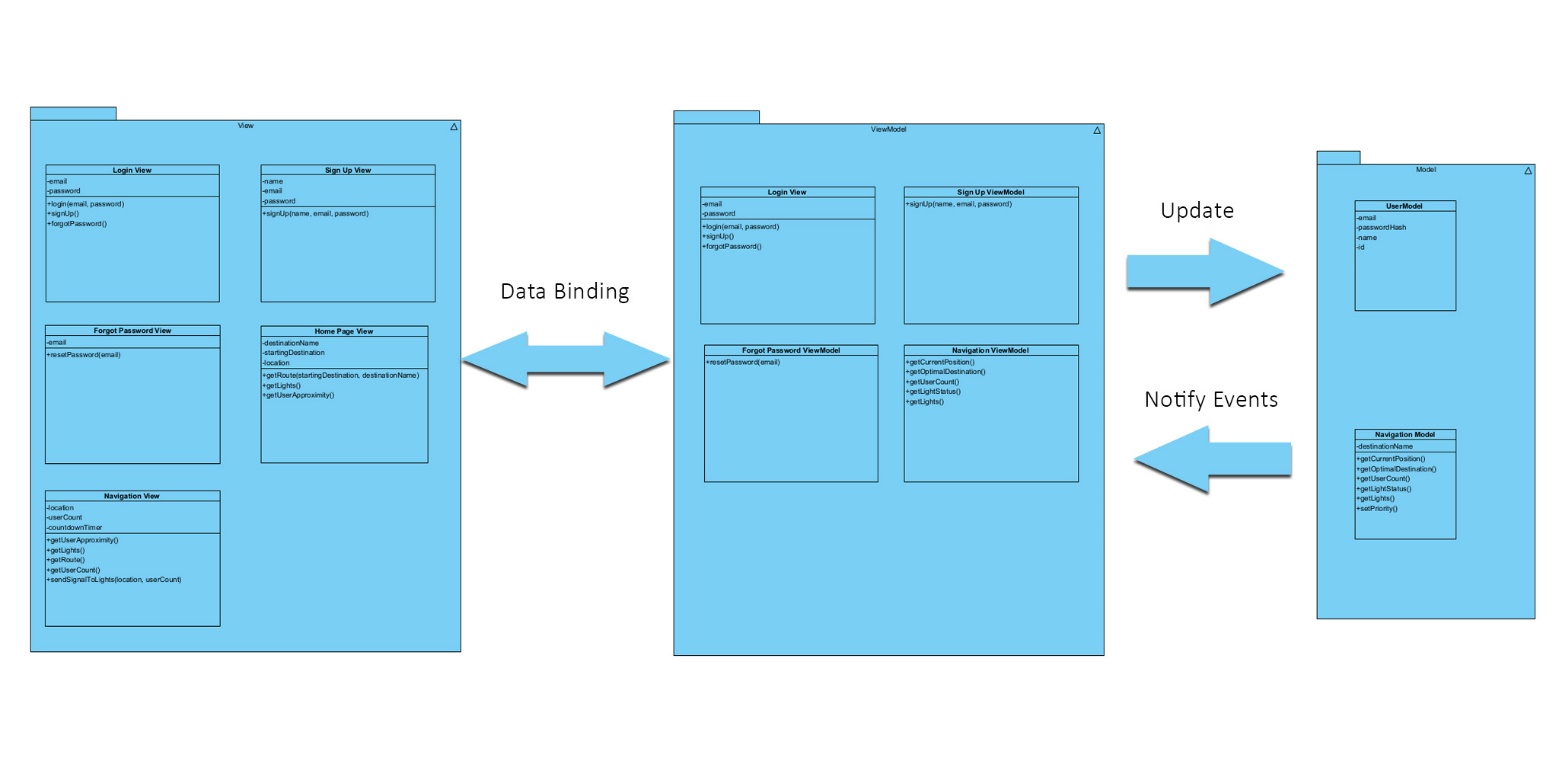
MVVM will be the chosen design pattern as it is an architecture tailored to modern UI development where the View is responsible of the designer instead of a developer (Lou, 2016). It also provides a new mechanism called **Databinding** where it allows the view to change automatically as it is directly bounded to the properties and operations of the View-Model, thus offering better testability (Lou, 2016). MVVM also provides a cleaner code structure (maintability) as it seprates codes into different areas within the application. This will be ideal for the VTL application as the app itself could be an extensible application itself, where MVVM also supports as it acts as a guide for structuring maintable, and testable code. 

Figure MVVM Class Diagram

### Runtime Process Characteristics

The user interface of the VTL application must be simple as possible, whilst making sure it is usable. Usability refers to how well the application can be used by the users to reach their desired goals, with satisfaction, efficiency, and effectiveness (Huang & Benyoucef, 2022). Inadequate usability app design could cause users to reject the application, meaning they will not put effort into learning how to use the application further, or even delete it from their device entirely (Huang & Benyoucef, 2022). Visually alluring UI delivers an encouraging experience to app users, which in turn increases their engagement. (Fang, Zhao, Wen, & Wang, 2017). “UX, however, extends usability’s remit by considering the broader range factors that enable people to have functional and satisfying experiences with technology.” (Branch, Parker, & Evans, 2021). Screen sizes will also have to be considered, so the main test phone will be a Samsung S21 Ultra as it has a resolution of 2400x1080 (FHD+), and screen size of 158.4mm (6.2”) (Samsung, n.d.)

The mobile application will include the following screens:

* Splash Screen
* Landing Screen (Login section
* Forgot Password
* Sign up
* Welcome screen
* Password sent
* Home page
* Navigation

#### Splash Screen

The splash screen where it displays the app name as well as the desired logo. Since this screen will serves minimum purpose, it should contain a maximum response time of one second. This is only applicable during cold start, and to give the user the feeling that they started the app. This screen will be skipped if it is on warm start.

#### Landing Screen (Login Screen)

The login screen will have the app name on top as well as the login attributes, where the user must enter to login. Other necessary options will be there as well such as signing in with Google or Facebook, sign up, and forget password.

#### Forgot Password

The only requirement here is the email so that it can contact the database to find the email and then send a password. If it fails, it will stay on the same screen advising the user that there is no such email.

#### Password Sent

This screen is only applicable if the email was found within the Forgot Password section. It also contains a shortcut link right back to the login screen, so the user can login right away after successfully resetting their password or so, instead of restarting the entire app in order to get back to the login screen.

#### Sign up

The sign-up screen will only ask for name, email, and password. Since this is aiming to be a simple application, further information will be unnecessary.

#### User created

Similar to password sent screen, however it is only applicable once the correct details have been confirmed in the sign up page.

#### Home Page

This application will be using Google Map’s API, therefore the maps will be almost identical to Google Maps itself. This is to create familiarity of other navigation applications, instead of over complicating approaches on how to use the app. It will display the current location on the map. It will also give a text field at the top asking where the desired destination will be.

#### Navigation Start

This will be considered the final and main screen of the entire application where everything runs. It will contain the desired destination text, traffic light status followed by a loading bar, main navigation map system, and there will also be a display of users within the vicinity with optional settings such as text, random user avatars, or both. The user avatar will not be editable by users, instead it will be an API calling on random generated cartoon characters as the users. This is for appeal, but mainly for security reasons, as there is no intention of providing users images of what other people look like or so.

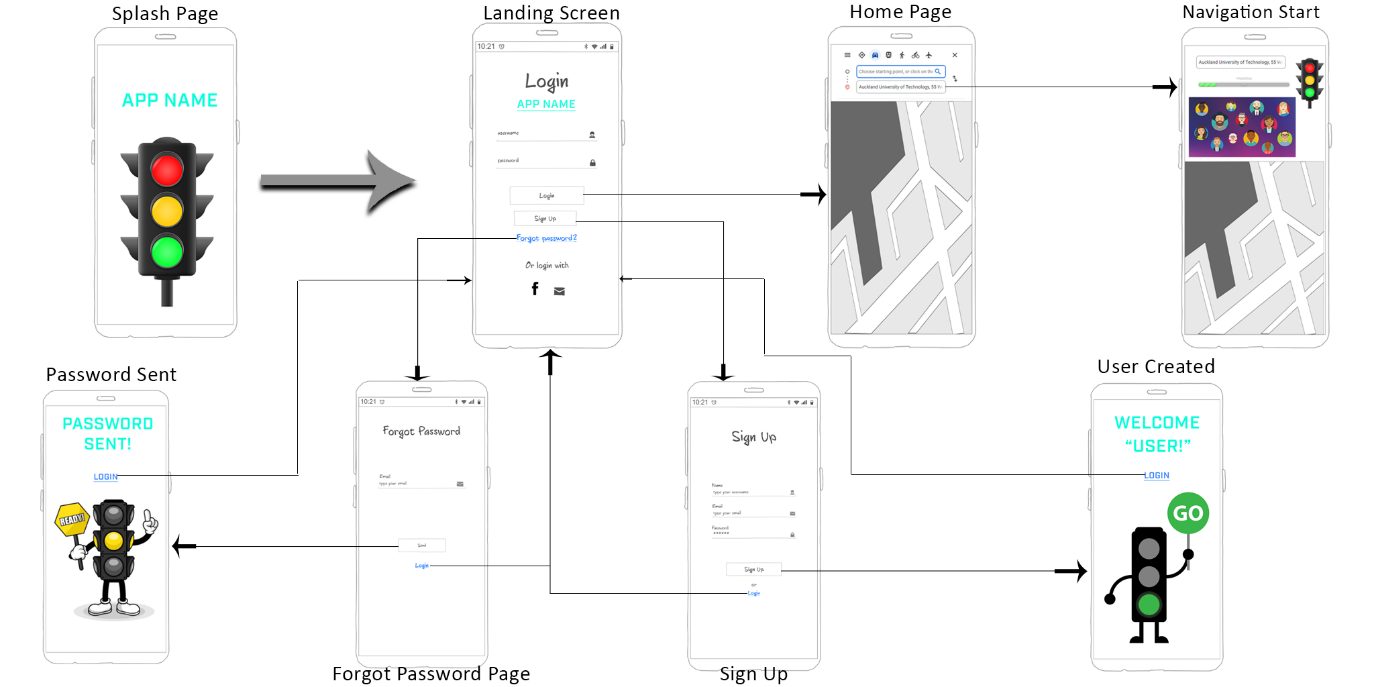


Figure UI and Navigation diagram

As shown in figure 8, the objective is to try and keep everything as simple as possible for the user, thus giving it a similar look to other applications out there will help ease familiarity on how to use the application. Also want to incorporate big friendly graphical images to give the user the feel of satisfaction that they are using the application correctly.

Diagram

Description automatically generated

Figure Sequence Diagram

This sequence diagram shows the mobile aspects of how it should ideally function. An alt case where if the user is successfully logged in or not. Once logged in, the user should be able to enter the destination, then it should communicate with the server and google map API, obtaining the optimal route and returning it directly to the user’s phone. From the mobile, it should also send signals to the traffic light where it calculates priority for which side is more congested, and then have it prioritized.

### Physical characteristics

#### From a user’s perspective (Mobile phone and car scenario)

Ideally all is required is their mobile phone with geo location services enabled and some form of geo fencing. This is to track of where the user currently is and sending out some form of signal to the traffic light where it will then calculate the number of people within that intersection and prioritize which section to go. For example, if one lane is heading north, the north traffic light will gather signal waves from users mainly from that lane, instead of lanes heading south, east, or west, finally it will all come together and calculate which lane is more congested and then have that lane prioritized. Therefore, the design of the overall application is mainly focused on the mobile aspect.

#### From a user’s perspective (Mobile phone pedestrian scenario)

Similar to the car situation, however it will just calculate the number of users waiting for their turn to cross the intersection. This will be less prioritized in comparison to a car scenario, for example if one person was waiting in the entire section, the traffic light will give them the very least priority, however, if there were more then ten users within the vicinity then it will get somewhat priority.

#### From a user’s perspective (Head unit or modern car technology scenario)

Very similar to the mobile and car scenario, however instead of phone, the application is built into the latest car technology. Though not all cars will have modern technology, therefore a smart head unit can also have the application installed, giving people with older generation cars a chance to utilize this technology. However, the UI and development will not be as concentrate on as much in comparison to the mobile phone application, meaning it will receive updates or so late.

#### Traffic Light perspective

The final physical characteristic will have to be the traffic light, where it will have some implementation of machine learning where it will do quick intense algorithms calculating and prioritizing which section will go or so. Other sensors will also be built in such as car collision detection where it can send a warning message to users within the vicinity, that section will have a slight delay or so.

To conclude the physical characteristics section, the main components here will be:

* Mobile phone (iOS or Android)
* GPS technology (On both phone and cars depending on which technology they go for)
* Traffic light and other sensors

## Evaluation Criteria

Verifying effective evaluation criteria requires identifying relevant factors to its corresponding component, and then focus on those aspects according to their significance in satisfying the client’s needs in this criterion. (AcqNotes, 2021)

### Response Time

“Response times under 100 milliseconds will feel instantaneous to the user. Response times up to one (or a few) second(s) are acceptable to the user, if they rarely occur. Delays greater than a few seconds will significantly degrade the user experience.” (Willocx, Vossaert, & Naessens, 2016).

### Splash Screen (And other pages)

Launching it from cold start should be no more than 1 second. From hot start, it should be almost instantaneous. Pages such as sign up, landing page, forgot password, and sign up should also be almost instantaneous.

Pages such as home page, sign up complete, password sent, and navigation can be up to 2-3 seconds max as it must communicate with the API or database.

### Functionality (Navigation)

Delays, lag, and latency cannot be afforded once navigation has started as it has to be near precise to prevent inaccurate directions and unnecessary accidents. Traffic light loading bar should be accurate and respond identically to the actual traffic light within the vicinity. Optional avatars within the vicinity will be deemed working if another user is spotted in the same intersection.

### Functionality (User aspects)

Able to create a new user in the sign-up screen, reset password, and able to login successfully. If user is already created, an alert message should appear to notify that the email is already in use (or user already exist).

### Usability

Does the application meet its initial goal where it is easy to use? Does it require further knowledge to use the application? Are users satisfied with using the application?

If these conditions are met then the evaluation will be considered a success, and then can be revised again to add further evaluation criteria.

# References

AcqNotes. (2021, April 07). *Proposal Development*. Retrieved from AcqNotes: https://acqnotes.com/acqnote/tasks/evaluation-criteria

Almeida, J. (2022, April 12). *Pros and Cons of Xamarin App Development*. Retrieved from Distant Job: https://distantjob.com/blog/xamarin-app-development/

Altexsoft. (2022, August). *The Good and the Bad of Flutter App Development*. Retrieved from Altexsoft: https://www.altexsoft.com/blog/engineering/pros-and-cons-of-flutter-app-development/

Aurisch, R., Ahmed, M., & Barkat, A. (2019). An outlook at Agile methodologies for the independent games developer. *International Journal of Computers and Applications*, 812-818.

Berka, M. (2020). *Flutter Pros & Cons – Should You Use It In Your Project?* Retrieved from Invotech: https://invotech.co/blog/flutter-pros-cons-should-you-use-it-in-your-project/

Branch, J., Parker, C., & Evans, M. (2021). Do User Experience (UX) Design Courses Meet Industry’s Needs? Analysing UX Degrees and Job Adverts. *The Design Journal*, 631-652.

Brewster, C. (n.d.). *15 Examples of Successful Companies Using React Native in 2022*. Retrieved from Trioblog: https://www.trio.dev/blog/companies-use-react-native

Dada, O. A., & Sanusi, I. T. (2021). The adoption of Software Engineering practices in a Scrum environment. *African Journal of Science, Technology, Innovation and Development* .

Dang, A. T. (2020, September 27). *MVC vs MVP vs MVVM*. Retrieved from Level Up Coding: https://levelup.gitconnected.com/mvc-vs-mvp-vs-mvvm-35e0d4b933b4

Fang, J., Zhao, Z., Wen, C., & Wang, R. (2017). Design and performance attributes driving mobile travel application engagement. *International Journal of Information Management*, 269-283.

Feng, X., Saito, M., & Liu, Y. (2015). Improve urban passenger transport management by rationally forecasting traffic congestion probability. *International Journal of Production Research*, 3465-3474.

GeeksforGeeks. (2020, December 02). *Difference Between MVC, MVP and MVVM Architecture Pattern in Android*. Retrieved from GeeksforGeeks: https://www.geeksforgeeks.org/difference-between-mvc-mvp-and-mvvm-architecture-pattern-in-android/#:~:text=MVC%20(Model%20%E2%80%94%20View%20%E2%80%94%20Controller,android%20architecture%20pattern%20among%20developers.

Gupta, S., & Chug, A. (2020). Software maintainability prediction of open source datasets using least squares support vector machines. *Journal of Statistics and Management Systems*, 1011-1021.

Hanssen, G., Haugset, B., Stålhane, T., Myklebust, T., & Kulbrandstad, I. (2016). Quality Assurance in Scrum Applied to Safety Critical Software. *International Conference on Agile Software Development*, 92-103.

Hoory, L., & Bottorff, C. (2022, March 25). *What Is Waterfall Methodology? Here’s How It Can Help Your Project Management Strategy*. Retrieved from Forbes: https://www.forbes.com/advisor/business/what-is-waterfall-methodology/

Huang, Z., & Benyoucef, M. (2022). An Empirical Study of Mobile Application Usability: A Unified Hierarchical Approach. *International Journal of Human–Computer Interaction* .

Kamienski, N. (2022, May 31). *REACT NATIVE VS XAMARIN: PROS AND CONS (2022 EDITION)*. Retrieved from Pagepro: https://pagepro.co/blog/react-native-vs-xamarin-pros-and-cons/

Kanjilal, J. (2021, December 3). *Comparing the MVC MVP and MVVM Design Patterns*. Retrieved from Developer.com: https://www.developer.com/design/mvc-vs-mvp-vs-mvvm-design-patterns/

Kissflow. (2022, March 2). *Kanban Methodology: The Simplest Agile Framework*. Retrieved from Kissflow: https://kissflow.com/project/agile/kanban-methodology/

Loeffler, S., Roth, R., Goring, S., & Myrbo, A. (2021). Mobile UX design: learning from the Flyover Country mobile app. *Journal of Maps*, 39-50.

Lou, T. (2016). A comparison of Android Native App Architecture MVC , MVP and MVVM.

Lutkevich, B. (2021, October). *Scrum*. Retrieved from TechTarget: https://www.techtarget.com/searchsoftwarequality/definition/Scrum

Ma, D., Xiao, J., & Ma, X. (2020). A decentralized model predictive traffic signal control method with fixed phase sequence for urban networks. *Journal of Intelligent Trasportation Systems*, 455-468.

Manchester Digital. (n.d.). *Is Xamarin dead ?* Retrieved from Manchester Digital: https://www.manchesterdigital.com/post/foresight-mobile/is-xamarin-dead

Mansour, S. (n.d.). *Product Manager: The role and best practices for beginners*. Retrieved from Atlassian: https://www.atlassian.com/agile/product-management/product-manager#:~:text=A%20product%20manager%20is%20the,that%20vision%20into%20a%20reality

Marcak, M. (2022, May 4). *REACT NATIVE PROS AND CONS – 2022 UPDATED*. Retrieved from Pagepro: https://pagepro.co/blog/react-native-pros-and-cons/

Midha, V., Singh, R., Palvia, P., & Kshetri, N. (2015). Improving Open Source Software Maintenance. *Journal of Computer Information Systems* , 81-90.

Peal, G. (2018, June 20). *Sunsetting React Native*. Retrieved from Medium: https://medium.com/airbnb-engineering/sunsetting-react-native-1868ba28e30a

Resources for employers. (n.d.). *UI/UX Designer job description*. Retrieved from Resources for employers: https://resources.workable.com/ui-ux-designer-job-description#:~:text=UI%20UX%20designers%20create%20the,also%20process%20flows%20or%20sitemaps

Sakovich, N. (n.d.). *Cross-Platform Mobile Development: Five Best Frameworks*. Retrieved from Sam Solutions: https://www.sam-solutions.com/blog/cross-platform-mobile-development/

Samsukha, A. (2022, May 13). *Best Cross-Platform App Development Frameworks in 2022*. Retrieved from Emizentech: https://www.emizentech.com/blog/cross-platform-app-development-frameworks.html

Samsung. (n.d.). *Galaxy S21 Ultra*. Retrieved from Samsung: https://www.samsung.com/levant/smartphones/galaxy-s21-ultra-5g/specs/#:~:text=\*%20Display%20Size%3A%20Measured%20diagonally%2C,to%20the%20rounded%20corners%20and

Samuels, M. (2021, May 19). *What is a software developer? Everything you need to know about the programmer role and how it is changing*. Retrieved from Zdnet: https://www.zdnet.com/article/what-is-a-software-developer-everything-you-need-to-know-about-the-programmer-role-and-how-it-is-changing/

Sergeev, A. (2020, June 10). *What is Scrum Process Lifecycle?* Retrieved from Hygger: https://hygger.io/blog/what-is-scrum-lifecycle/

Sinhal, A. (2017, January 4). *MVC, MVP and MVVM Design Pattern*. Retrieved from Medium: https://medium.com/@ankit.sinhal/mvc-mvp-and-mvvm-design-pattern-6e169567bbad

Wang, F., & Xu, Y. (2011). Estimating O–D travel time matrix by Google Maps API: implementation, advantages, and implications. *Annals of GIS*, 199-209.

Willocx, M., Vossaert, J., & Naessens, V. (2016). Comparing performance parameters of mobile app development strategies. *MOBILESoft'16*, 38-47.

Workfront. (n.d.). *Waterfall Methodology*. Retrieved from Workfront: https://www.workfront.com/project-management/methodologies/waterfall#:~:text=The%20waterfall%20methodology%20is%20a,detailed%20documentation%2C%20and%20consecutive%20execution.