Proximity Sensing in the Commuting Service

Group 1-1

Jesper Blomqvist
Lekhaz Adapa
Robert Englund

Table of Content

Abstract

The report explores the Internet of Things (IoT) project which deals with proximity sensing in the commuting sector. The goal of the project was to explore and demonstrate how bluetooth and sensors can be used to achieve proximity in different contexts. The selected scenario enabled us to design and develop a system for exploration and demonstration. Four components were developed. Firstly, a carriage system for detecting crowdedness. Secondly, an edge controller to aggregate and distribute carriage crowdedness on a train level. Thirdly, an android application for the user to correctly position on the platform and see crowdedness information for a train. Lastly, a cloud system for storing data about carriages, trains, sensors and platforms. The result shows how bluetooth and sensors can be used in consensus for proximity sensing to deliver contextual value to an end user.

Introduction

The Internet of Things (IoT) is an emerging hot topic that focuses on things and their ability to process and transfer data between each other without requiring human interaction (Gillis, 2022). The things in focus are seemingly ordinary physical objects, sensors and other items normally not considered to have computing abilities. Connecting these things to the internet enables them to communicate and share data with each other. This has the possibility to create smart systems that intertwine objects, their environment and people together. (Karen, Scott and Lyman, 2015)

Proximity is an area of IoT necessary to intertwine objects with their environment. Making interactions more practical by using data that is physically relevant. Such data can be collected using sensors that monitor the environment and react to it (Kocakulak and Butun, 2017). This brings a notion of context awareness, which helps make devices and applications more smart as they can react to the environment accordingly (Want, Schilit and Jenson, 2015).

Bluetooth is a technology that recently is interesting from an IoT perspective. It allows for a communication technology known as bluetooth low energy (LE). Which is appealing to use (Raza et al., 2015) due to its low energy consumption and ability to communicate with most end user devices, such as smartphones and bands (Hortelano et al., 2017).

Public transport is a growing sector (ReportLinker, 2022) and a key component for solving the ongoing climate crisis (American Public Transportation Association, 2008). With its rapid growth and necessity in society, a lot more people will be choosing trains and buses over other means of transportation. This is problematic because crowdedness is one of the biggest annoyances in public transport (Haywood, Koning and Monchambert, 2017). Furthermore, Haywood, Koning and Monchambert (2017) found that not being seated and closeness to other commuters was the biggest pain points for commuters. Thus, it is important to help commuters avoid crowdedness and find available seats.

Scenario

Our scenario is within the commuting services, and about helping commuters get a better experience, reducing pain points described by Haywood, Koning and Monchambert (2017). When waiting for a train on a platform, it is not possible to know anything about the carriage that a commuter would enter. It might be very crowded and no seats will be available. Space on the train can be an important factor for commuters having strollers, wheelchairs, walkers or are senior citizens. If they had entered from another door, or even a different carriage, the experience could be different for both them and their fellow commuters. Utilizing IoT and proximity can improve upon this situation and grant commuters a better experience.

Purpose and Goals

The purpose of this project is to explore and demonstrate how bluetooth technologies and sensors can be used in consensus for proximity sensing with regards to our scenario. Proximity is a highly relevant area as the user's application would adapt to its environment, making it possible for users to make well informed timely decisions.

The goal is to deliver a scalable prototype that demonstrates how this can be achieved. An application will be developed that helps the user find a suitable carriage for their commute. This will involve gathering proximity information about the train station, train and the crowdedness in its carriages. The user application can then interact with the bluetooth beacons on the platform to find a suitable location on the platform to wait for the train so that he or she can enter an optimal carriage for their commute. This scenario demonstrates how bluetooth technologies and proximity sensors can be used to share context for proximity as well as how proximity can be used in a real life situation that is scalable for larger implementations.

Method

To finish the project within the timeframe, a project plan was created that included a time schedule and working structure. Due to the project's size, no strict development methodology was followed, but work was conducted with more of an agile approach. This would allow the group to quickly respond to changes as more knowledge is required.

The work was divided into smaller parts. Dividing responsibility was thus possible, making it suitable to not only work collaboratively in group sessions, but also individually. To facilitate this, a github project (Adapa, Blomqvist and Englund, 2022b) was set up to enable each member of the group to work on their part of their project. Using GitHub helps with code collaboration as it controls the version of each file and aids with code conflicts.

Design

During design, a focus was to design a scalable and expandable prototype. With this in mind, we settled on a design in which the system was divided into four components and two message protocols. The components were: user application, carriage sensors, edge

controller and cloud. The two message protocols were MQTT and HTTP. The following sections describe each part of the design in more detail.

Message protocols

Two message protocols were chosen, HTTP and MQTT. Both of these protocols are commonly used in IoT and offer different advantages depending on usage situation. HTTP is better suited for message exchange where only one message is sent, in an ask and respond manner. Not only is HTTP designed for that communication style, it is also faster than MQTT for just sending and receiving one message (Craggs, 2022). In our discussion, we refer to this as infrequent communication.

MQTT is better suited for frequent communication, where there are lots of messages being sent (Craggs, 2022). A great benefit with MQTT is its subscriber and publisher structure. Information that is of interest can just be published when it is ready, and entities that have an interest in this information can subscribe and receive that information. For our project, this is great as crowdedness information can be regularly published and subscribed to when necessary.

For all communication we decided on using the JSON format. Because each component will be programmed with different languages, using a shared message format is important. JSON is lightweight and is easily parsed, making it a great choice for context sharing.

Cloud

For the prototype to be more realistic, we decided to include a cloud that would store information. The cloud stores information in a database about what trains exist, what carriages are connected to a train and in what order. What platforms exist and what bluetooth beacons are on the platform. A schema for a database was designed to hold this information, see figure 1.

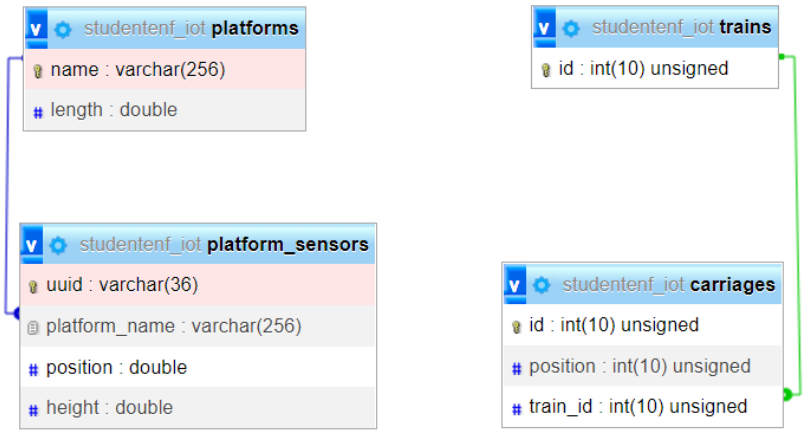

Figure 1

According to the database design, carriages are connected to trains, and platform sensors are connected to a platform. This information is crucial and will be further discussed under edge controller and user app respectively.

The cloud would be programmed as a RESTful web API to make communication with the underlying database possible for the other components. Doing it this way makes our prototype not use hardcoded values. This makes the solution more realistic, in which the prototype's applicability can better be determined for a bigger scale solution.

It is with the cloud that the HTTP protocol is being used. Communication would be infrequent and follows the structure of ask and respond, in which HTTP works great.

Carriage Sensors

In order to make a scalable solution, we decided to make each carriage its own gateway. To detect the crowdedness of a carriage, we settled on using proximity sensors. Where each seat would have a proximity sensor. Reading the sensor's value would indicate if a seat is occupied or not.

As a carriage has many seats, we needed a solution that could be scaled to many sensors. Often, sensors of the same type use the same low-level address with limited possibilities for secondary or alternative addresses. Hence, conflicts with reading sensor values can arise as no specific sensor can be targeted due to the shared address space. To solve both sensor addressability and scalability, we decided on using a multiplexor. A multiplexor can be used to connect many sensors that publish data to the same address. For our testing, we were only able to get two sensors per carriage, but, to make a solution that as closely as possible resembles reality, we still went with a multiplexor to explore a solution that can grow in sensors. Figure 2 shows a multiplexor connected to two sensors.

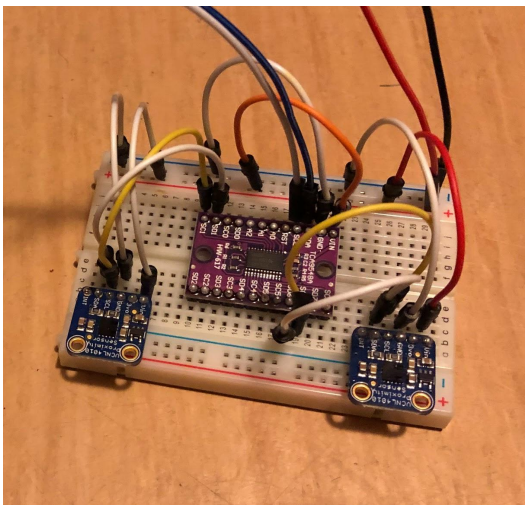

Figure 2

Since each carriage is its own gateway, gathering data about each seat is handled in smaller atomic units. This means that each carriage will process sensory data locally. This implies a better distribution of processing. Each carriage handles its own context and no central location is required for processing. A single point of failure is thus avoided, and the solution can scale freely with the number of carriages needed.

The data is then published using MQTT. Because information about crowdedness is frequent, MQTT works great. The carriage publishes the data under a MQTT topic `carriage/{id}` where the `{id}` part is replaced with the carriage id. Having each carriage publish its data under its own topic makes handling trains easier. One carriage can be moved from train to train, but the carriage id remains the same. No configuration changes are therefore required other than changing relations in the database.

Edge Controller

The edge controller works as an aggregator of data. It communicates with the cloud to determine what carriages are connected to what train. That information can be used to know what topics to subscribe to for carriage crowdedness information. In our prototype, we only have few trains and carriages, so having one edge controller aggregating all information is fine. But, the solution can easily be modified so that different edge controllers handle aggregation of different trains. Scaling up is thus easy, as multiple edge controllers can be introduced to distribute processing.

Carriage data is aggregated to train data. Much like the figure 3 shows.

Figure 3

This makes subscribing easier for the users, as it will only need to subscribe to one topic instead of multiple ones. This design also fosters future expandability for the application. If we would like to expand the application where train data is saved for later analysis, such as training machine learning models or planning for the future. This persistent storage of data does not need to be added to the edge controller. Another component can be introduced that subscribes to the same train topics users subscribe to. Then, when train data gets published the new component can save that information. Which is inline with scalable design, dividing responsibility to smaller units.

User Application

The user application consists of two screens. The first where the app is looking for a station. The second where the users location is shown on the platform with where each carriage will be and what its crowdedness will be. A prototype of the app was constructed to get an understanding of how it would look, pictured in figure 4.

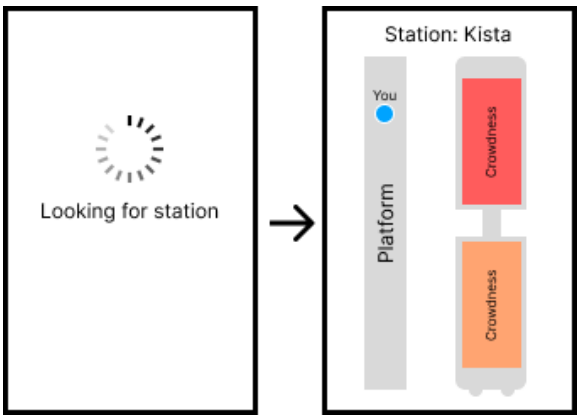

Figure 4

In this situation, the use of bluetooth becomes critical to interact with the platform. Often, train platforms are underground. Relying on GPS is thus not possible for positioning. Using bluetooth beacons is thus preferable for proximity positioning. Bluetooth beacons do also have the possibility to give more fine grained precision than GPS.

To design a system that uses bluetooth technologies, first we needed to investigate how bluetooth beacons work and how they could be used. A bluetooth beacon broadcasts a signal with a given interval. This signal can contain different information, one of which is a Universally Unique Identifier (UUID). Using UUID, it is possible to identify a beacon. Identifying a beacon allows the application to know its proximity position, as each beacon's platform position is stored in the database. Looking at the database schema from figure 1, each station has a number of sensors connected to it. It is thus possible to read what UUID can be found on a platform as well as their position on that platform.

Another value that bluetooth beacons broadcast is a Received Signal Strength Indicator (RSSI) value. Each beacon is configured with a RSSI value that is expected when the user is one meter away from it. Using that value and the received signal strength of a broadcast, an approximate distance can be calculated. Multiple beacons can thus be used to get an approximate platform position using triangulation, as the picture below depicts.

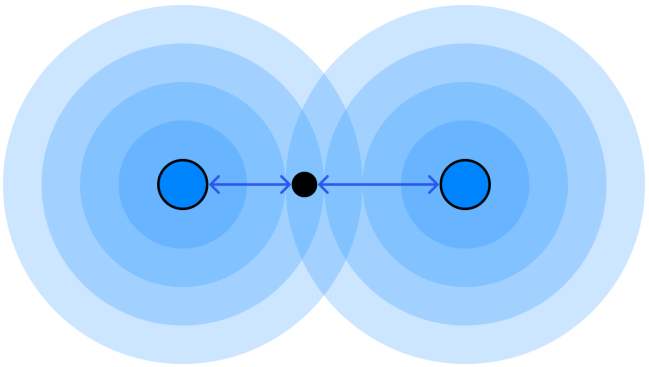

Figure 5

Received signal strength from a beacon can vary a lot depending on the environment. If an object's obscure line of sight between beacon and receiver, the calculated distance may be off. Thus, it is important to base distance on multiple readings of a broadcasted signal. We

configured the end users application to the last three seconds of RSSI values from each detected beacon.

The user application communicates both with the cloud and edge controller. When it detects a bluetooth beacon, it consults the cloud with the beacon's UUID. In return, the application receives what station they are on and what other beacons UUID that are expected to be found. It also receives information about that station's next train. Which makes it possible for the application to subscribe to that train's topic for crowdedness information. The crowdedness information is drawn on the screen using colors. Ranging from green to red depending on the crowdedness. Where green is lesser and red is greater crowdedness.

Development

Each component was developed with different languages. The application was developed in Java with Android Studio. The edge controller and carriage sensors used Python, as both components ran on the raspberry pies that the group had acquired to complete the project. The cloud was developed using MySQL and PHP as there were group members familiar with those technologies.

Two third-party libraries were used to develop the project. For MQTT an open source project Paho (Eclipse, 2016) was used for the edge controller, android app and carriage sensor. The android app also used another library known as AltBeacon (AltBeacon Contributors, 2014). The library helped with beacon detection and distance calculation.

Result

The project resulted in an application that functionally met our goals. It successfully demonstrated how sensors and bluetooth can be used to deliver a scalable proximity application. A demonstration of the application can be found as video format on (Adapa, Blomqvist and Englund, 2022a) and the code is publicly available on GitHub (Adapa, Blomqvist and Englund, 2022b).

Upon starting the application, the user is sent to a screen that scans its surroundings for a station, figure 6.

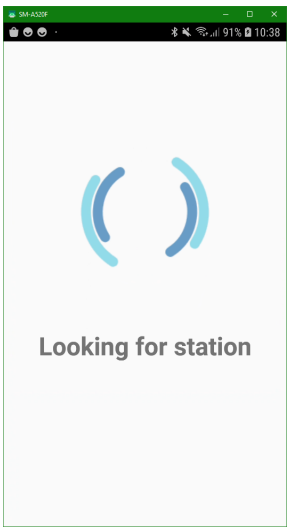

Figure 6

When it detects a station, the application switches to the station view. Where it uses the station's bluetooth beacons to determine location on the platform, figure 7.

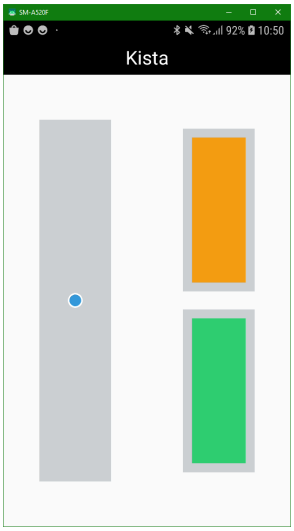

Figure 7

Each carriage publishes its crowdedness information which is aggregated on an edge controller. The edge controller can then publish an entire train's crowdedness information which the application subscribes to and displays. When the carriage's sensors sense more seat occupations, the colors will update, figure 8.

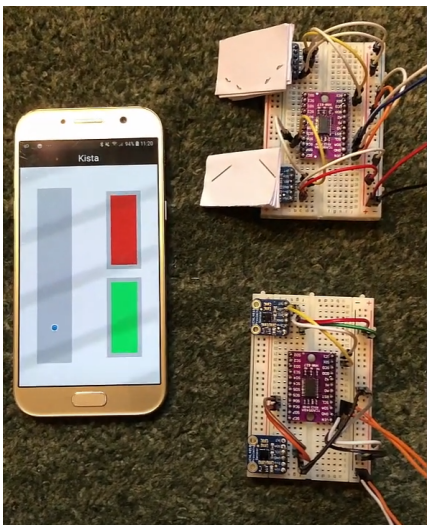

Figure 8

With the proximity based location and carriage crowdedness, the user can now decide where on the platform he or she should be standing to get a carriage with a suitable crowdedness. Figure 8 above suggests that the users should move down the platform, which is also shown with the blue dot.

Discussion

The results demonstrate a proof of concept that can be further developed and fulfills the project's goals. With the development of the application, it can be concluded that it is possible to use bluetooth and sensors to provide proximity sensing. Bluetooth can be used to interact with the platform, automatically showing the correct station in the application and your relative position to it. Sensors are used in each carriage to detect passengers.

Furthermore, the project also explores how to interconnect these different sensors to create a holistic application that adds value to the end user. Context gets shared and connected so that decisions can be made. Which is demonstrated by working with different IoT components and communication protocols. The platforms and trains are autonomous units that are connected.

Another goal was also to develop the application in a scalable manner. When designing the application, we focused on a system that can be broken down into atomic units. This helps with scalability as it is possible to introduce more atomic units to spread the load. Our project does not indicate, or make us believe, that there would be any scalability issues. Each carriage is its own gateway and more edge controllers can be introduced to handle more trains.

The usage of a publisher and subscriber model also helps with decoupling and scalability. In the case of building out the application, it will be easy as more components can be connected using the publisher subscriber model. If for instance more information needs to be done with the train crowdedness information, a new component can be constructed that

subscribes to the `train/{id}` topic. This addition will not disrupt anything with the currently implemented system. This makes the solution highly expandable for future development.

However, there are limitations with our solution that need to be addressed to further improve it. Relying solely on proximity sensors on seats means the sensing mechanism is one dimensional and can be fooled into deciding on false positives, making a carriage appear more crowded than it in actuality is. It is common for passengers to use available seats to store backpacks and other belongings. If a backpack were to be placed on a seat, that seat would make the system think the seat is occupied. This can be remedied by using arrays of sensors with different, complementing, sensing capabilities, such as heat sensors on the seats. Given the small budget for the project, this solution was not possible to explore.

It should also be noted that this project focused primarily on detecting seated passengers, but modern metro trains' total passenger capacity includes a large number of standing passengers. In fact, according to Railvolution (2020), the modern train cars used in Stockholm's metro system have a fixed seating of 140 seats per train car, and the capacity to transport an additional 490 standing passengers at a 5 passengers per square meter density. It is thus important to further investigate methods for detecting standing passengers, as they make up the majority of the capacity during full load.

Future work

The design of the project and idea allows for multiple extensions and further development of the project. In this chapter, we outline some idéas for further development of the project.

Firstly, the carriages now only detect passengers using proximity sensors. This limitation was discussed in the discussion and can be addressed by implementing more sensors. Heat sensors could be combined to detect if the occupation is a person or not.

Moreover, the heatmap can be more precise. There are two ways of improving the heatmap. First, each carriages can be divided into more sectors. For example, one sector per carriage door. This would give a more precise picture of exactly what door to stand in front of. Second, each seat's status could also be shown together with the heatmap. This would allow users to plan where to stand if they have company and want to find a number of seats that would allow them to sit together.

In addition, carriage crowdedness now only takes into account seated passengers. Even though this might be enough, as a passengers goal might only be to find available seats, adding detection for standing passengers could be great. If a user is only supposed to travel a few stops, they might want to stand instead. Detecting two different heatmaps, seated and standing, would solve this. However, it becomes trickier to design a system that can reliably detect standing passengers. One way is to reintroduce bluetooth into the carriages. Each carriage could scan for bluetooth signals. Subtracting the known seated passengers with the bluetooth devices found could give an indicator to how many passengers are standing. Of course this is limited to passengers having bluetooth activated devices. It could also be problematic as one user might have multiple devices that are detected, making the system believe the carriage is more crowded.

The app could also be extended to include a travel planner with guidance. The project shows that bluetooth is a reliable way for proximity sensing. The app could thus also include guiding the user to the right platform, and even exiting at the right station.

Furthermore, the application could be extended to enable the user to see other trains crowdedness, not only the next train for the platform. The user might want to compare the next three trains for a given platform, or maybe check the trains that are arriving on other platforms. This could help the user more with planning his or her trip. This addition would also be easy to implement. The train view component can easily be reused and to fill it with information only requires the app to start subscribing to another topic, the topic for the wanted train.

Finally, our project's work can be adapted to other end users. It might be interesting to provide train drivers and traffic control information about crowdedness, as well as saving it for later use. Saving the data could help with route planning or model training. In both of those scenarios, lots of data is beneficial. Other means of public transportation could also benefit from our research, such as buses. There, proximity could help with tracking what stop you are on as well as showing the coming buses and their crowdedness so that the users can decide on which bus to take.

The implementation of an IoT based data gathering and processing solution could also provide benefits to other scientific fields and contexts, such as the ability to correlate crowdedness data to propagation of infections, e.g. RS, influenza and covid viruses, could contribute to epidemiological research insight.

References

Adapa, L., Blomqvist, J. and Englund, R. (2022a). *IoT Demo*. *www.youtube.com*. Available at: https://www.youtube.com/watch?v=FS8Zr1-cvk0 [Accessed 3 Jan. 2023].

Adapa, L., Blomqvist, J. and Englund, R. (2022b). *robengrobban/IoT-Project*. [online] GitHub. Available at: https://github.com/robengrobban/IoT-Project [Accessed 3 Jan. 2023].

AltBeacon Contributors (2014). *GitHub - AltBeacon/android-beacon-library: Allows Android apps to interact with BLE beacons*. [online] GitHub. Available at: https://github.com/AltBeacon/android-beacon-library [Accessed 5 Jan. 2023].

American Public Transportation Association (2008). *Public Transportation Reduces Greenhouse Gases and Conserves Energy*. [online] Available at: https://www.apta.com/wp-content/uploads/Resources/resources/reportsandpublications/Documents/greenhouse_brochure.pdf.

Craggs, I. (2022). *MQTT Vs. HTTP: Which protocol is the best for IoT or IIoT?* [online] www.hivemq.com. Available at: https://www.hivemq.com/blog/mqtt-vs-http-protocols-in-iot-iiot/ [Accessed 28 Dec. 2022].

Eclipse (2016). *Eclipse Paho - MQTT and MQTT-SN software*. [online] Eclipse.org. Available at: https://www.eclipse.org/paho/ [Accessed 7 Jan. 2023].

Gillis, A. (2022). *What is IoT (Internet of Things) and How Does it Work? - Definition from TechTarget.com*. [online] IoT Agenda. Available at: https://www.techtarget.com/iotagenda/definition/Internet-of-Things-IoT.

Haywood, L., Koning, M. and Monchambert, G. (2017). Crowding in public transport: Who cares and why? *Transportation Research Part A: Policy and Practice*, 100, pp.215–227. doi:10.1016/j.tra.2017.04.022.

Hortelano, D., Olivares, T., Ruiz, M., Garrido-Hidalgo, C. and López, V. (2017). From Sensor Networks to Internet of Things. Bluetooth Low Energy, a Standard for This Evolution. *Sensors*, 17(2), p.372. doi:10.3390/s17020372.

Hsu, C.-L. and Lin, J.C.-C. (2016). Exploring Factors Affecting the Adoption of Internet of Things Services. *Journal of Computer Information Systems*, 58(1), pp.49–57. doi:10.1080/08874417.2016.1186524.

Karen, R., Scott, E. and Lyman, C. (2015). The Internet of things: an Overview. *The Internet Society (ISOC)*, 80, pp.1–50.

Kocakulak, M. and Butun, I. (2017). *An overview of Wireless Sensor Networks towards internet of things*. [online] IEEE Xplore. doi:10.1109/CCWC.2017.7868374.

Railvolution (2020). *Stockholm's New Metro Fleet*. [online] Railvolution. Available at: https://www.railvolution.net/news/stockholm-s-new-metro-fleet [Accessed 7 Jan. 2023].

Raza, S., Misra, P., He, Z. and Voigt, T. (2015). Bluetooth smart: An enabling technology for the Internet of Things. *2015 IEEE 11th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. doi:10.1109/wimob.2015.7347955.

ReportLinker (2022). *The Global Public Transportation Market size is expected to reach $285.7 billion by 2028, rising at a market growth of 5.4% CAGR during the forecast period*. [online] GlobeNewswire News Room. Available at: https://www.globenewswire.com/news-release/2022/11/22/2561072/0/en/The-Global-Public-Transportation-Market-size-is-expected-to-reach-285-7-billion-by-2028-rising-at-a-market-growth-of-5-4-CAGR-during-the-forecast-period.html [Accessed 7 Jan. 2023].

Want, R., Schilit, B.N. and Jenson, S. (2015). Enabling the Internet of Things. *Computer*, [online] 48(1), pp.28–35. doi:10.1109/mc.2015.12.