|  |  |
| --- | --- |
|  |  |



IoT Solution Design

Team 3 - Improve Value Chain of Public Infrastructure

COMP6324: IoT Services Engineering

Adam Golding (z3341873)

Jie Shang (z5153884)

Xinran Li (z5139069)

Michael Liyantama (z5034658)

Qian Cheng (z5149155)

Version: 1.0

03/05/2019

Contents

[1 Introduction 3](#_Toc7817702)

[1.1 Project Description 3](#_Toc7817703)

[1.2 This document 3](#_Toc7817704)

[1.3 GitHub Repository 3](#_Toc7817705)

[2 Business Problem 4](#_Toc7817706)

[2.1 Problem Definition 4](#_Toc7817707)

[2.2 Problem Solution 4](#_Toc7817708)

[2.3 Business Model Canvas 5](#_Toc7817709)

[2.4 Business Model 8](#_Toc7817710)

[2.4.1 Phase 1 Solution Cost 8](#_Toc7817711)

[2.4.2 Expected Revenue 9](#_Toc7817712)

[2.4.3 Return on investment 9](#_Toc7817713)

[3 Solution Requirements 11](#_Toc7817714)

[3.1 Functional Requirements 11](#_Toc7817715)

[3.1.1 Position Fixing 11](#_Toc7817716)

[3.1.2 Mobile Vending with Personal Delivery 11](#_Toc7817717)

[3.1.3 Reporting and Data Analysis 12](#_Toc7817718)

[3.2 Non-Functional Requirements 13](#_Toc7817719)

[4 Actors 15](#_Toc7817720)

[5 Use Cases 16](#_Toc7817721)

[6 Solution Architecture 19](#_Toc7817722)

[6.1 Software Design 19](#_Toc7817723)

[6.1.1 Customer (Link) 21](#_Toc7817724)

[6.1.2 QR Code (Link) 21](#_Toc7817725)

[6.1.3 Mobile Vendor (Link) 21](#_Toc7817726)

[6.1.4 Bluetooth Beacon (Link) 21](#_Toc7817727)

[6.1.5 Azure App Service (Link) 21](#_Toc7817728)

[6.1.6 SQL Database 21](#_Toc7817729)

[6.1.7 Reporting / Analytics (Link) 21](#_Toc7817730)

[6.1.8 Microsoft Azure Platform 22](#_Toc7817731)

[6.2 Deployment Design 23](#_Toc7817732)

[6.2.1 Beacons 25](#_Toc7817733)

[6.2.2 Mobile Devices 25](#_Toc7817734)

[6.2.3 Desktop Devices 25](#_Toc7817735)

[6.2.4 Cell Towers 25](#_Toc7817736)

[6.2.5 Internet 25](#_Toc7817737)

[6.2.6 Microsoft Azure Platform 25](#_Toc7817738)

[6.3 Hardware Design 26](#_Toc7817739)

[6.3.1 Customer centred 26](#_Toc7817740)

[6.3.2 Vendor centred 26](#_Toc7817741)

[7 Software Requirements 27](#_Toc7817742)

[8 Hardware Requirements 27](#_Toc7817743)

[9 Type of data that will be used in the solution 28](#_Toc7817744)

[10 Management of solution 28](#_Toc7817745)

[11 Solution Roadmap 29](#_Toc7817746)

[12 User Acceptance Testing 30](#_Toc7817747)

[13 Working Solution 33](#_Toc7817748)

[13.1.1 Backend APIs 33](#_Toc7817749)

[13.1.2 User Interface 34](#_Toc7817750)

# Introduction

## Project Description

The purpose of this document is to provide information about the design of the ‘adding value to public infrastructure’ project. This phase of the project consists of the design and build of a prototype of indoor position fixing and mobile vendor with delivery to your seat for a stadium (e.g. ANZ Stadium) using the Internet of Things (IoT) as a platform.

The solution was developed to enhance the patron’s experience while attending an event at a stadium. Utilising the Mobile Vendor, a patron can continue to enjoy the event they are attending while ordering food or drinks to be delivered to your seat. or to assist with the emergency evacuation of event attendees.

GPS is unreliable indoors and therefore cannot be used to provide accurate positioning. One of the core components is to establish a matrix of beacons that will enable indoor position fixing and navigation. This would be a once off investment that will form the foundation of other modules of the solution.

An outline of the complete solution is also provided in the Solution Roadmap. The roadmap describes the modules that will be built in the future phases of the project. This includes a complete indoor navigation module which will leverage off the infrastructure already implemented in the initial phase. The indoor navigation will allow patrons to better navigation to facilities within the stadium (e.g. the best way to find your seat or to find a bathroom that is not crowded). This feature can also provide a means to assist with the emergency evacuation of event patrons and staff.

## This document

This document is split into two sections:

1. Business Related
2. Technical Solution

## GitHub Repository

All source code, solution design document and pitch slide deck are located on the following GitHub repository. (Repository is set to private. Contact [michael.liyantama@student.unsw.edu.au](mailto:michael.liyantama@student.unsw.edu.au) with your GitHub ID for access)

<https://github.com/MikeLiyantama/COMP6324-Project.git>

# Business Problem

## Problem Definition

How to improve the facilities at the stadium to create a better user experience to attract more events to the facility/stadium.

|  |  |
| --- | --- |
| Pains | Gains |
| Reduced number of people attending events due to lack of premium facilities  Reduced events being held at stadium  Low customer satisfaction  How to improve patron and staff safety | Increased customer satisfaction  Increased revenue  Improved safety at events  Short queues, less waiting time  Good choice of food, better food service  Clean environment  Overall better experience |
| Long queues at event (entry, toilet, security, exit, cafes, bars)  Poor choices for food and drinks |

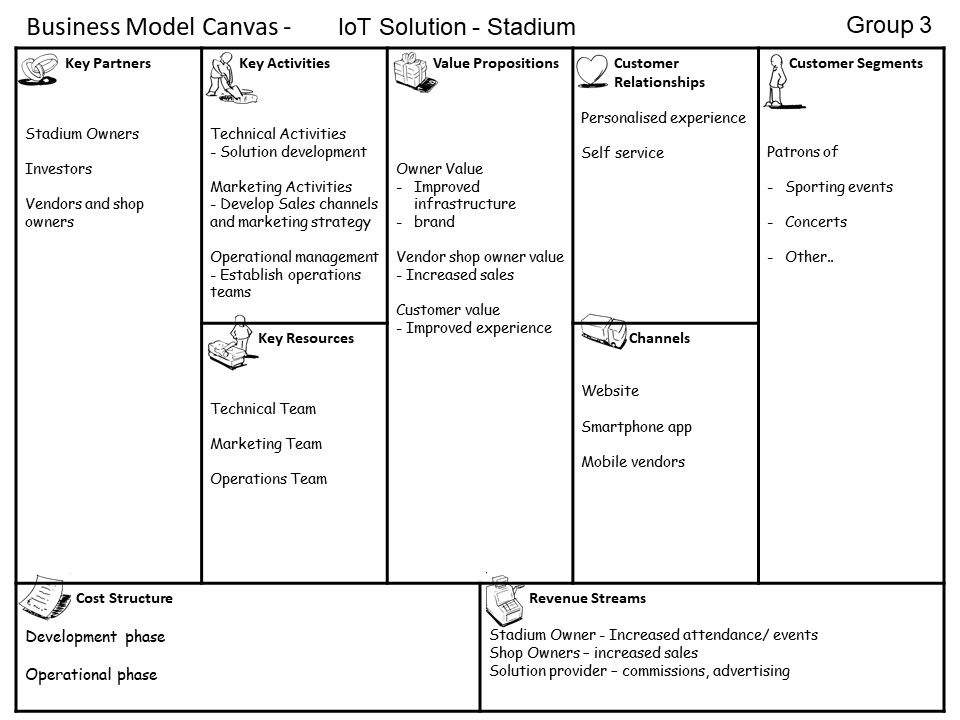
## Problem Solution

The following table outlines what will be implemented to address pains and convert into gains. The project will be broken into three phases. The Roadmap describes what issues will be addresses in each phase.

|  |  |
| --- | --- |
| Issue | Solution |
| Food delivery | Online ordering, payment and delivery to your seat.  Operational and management reporting. |
| Navigation | Route orders to appropriate vendors  The fastest way to find my seat/toilet/exit  Navigate to any part of the facility |
| Merchandise | Provide more channels to purchase merchandise with more convenient ways to receive products. |
| Facility Usage | Identify crowded facilities/areas suggest alternatives and route appropriately. |
| Safety/Emergency | Provide appropriate emergency instruction and procedures.  Alternative ways to notify people of incident and manoeuvre them to safety |

## Business Model Canvas

The Business Model Canvas for the project is shown below:



|  |  |
| --- | --- |
| Classification | Explanation |
| Key Partners | The stadium owner is a critical partner in the solution. They need to be involved in every step of the project.  Investors must be consulted and agree on how the funds will be spent.  Vendors and shop owners can provide input into the operationalising of the solution. |
| Key Activities | Key activities will be broken into 3 phases according to the Roadmap described later in the document. These key activities are required to develop, establish and operate the solution.  Technical activities include:   * Website and app development * Hardware installation * Solution maintenance   Marketing activities:   * Develop sales channels * Develop marketing strategy and campaigns   Operational Management   * Establish the operations team * Source call centre services |
| Key Resources | The key resources are required the develop and operate the solution.  Technical Team   * Backend developers * Frontend developers * Engineers for Beacon Networks * Project managers   Marketing Team   * Channel teams   Operations Team   * Administrators * Managers * Helpdesk |
| Value Propositions | Value can be addressed from different perspectives.  Owner Value   * Improved infrastructure/ assets * Improved brand awareness * Increase in number of events held at the stadium * Increase in revenue   Vendor/Shop Owner Value   * Increase in sales * Improved performance monitoring and analytics   Customer Value   * Better Service * Convenience * Improved experience |
| Customer Relationship | * More personalised experience |
| Channels | There are a number of marketing channels that we intend to develop.   * IoT-Stadium.com * Smartphone app * Physical stores located at the Stadium * Mobile vendors |
| Customer Segments | The customer segment will depend on the event that is being hosted at the stadium. Types of events will include:  Sporting events such as rugby league, soccer, cricket.  Concerts  Other |
| Cost Structure | The costs described relate to the setup and operations of the solution. The costs can be broken down into the initial development and start-up phase and the operational phase.  Development Phase   * Organisational setup * Hardware Installation * Software Development * Establish platform   Operational Phase   * Management and operations team * Helpdesk/Call centre |
| Revenue Streams | The revenue streams relate to the owners of the stadium, vendors and shop owners, the solution supplier  Stadium owner   * Increased revenue from attendance and stadium bookings * Improved business value   Shop Owner/ Vendor   * Improved profits through increase in sales   Solution provider   * Commissions from sales routed through the platform * Investment * Advertisement |

## Business Model

Initial selling price to stadium would cover the infrastructure setup as they would benefit from being a Smart Stadium by utilising the indoor navigation features and enhanced safety, they can recover the cost through rents to vendors and shops

The solution will be sold as Infrastructure-as-a-Service (IaaS) and/or Platform-as-a-Service (PaaS). The company will manage all the platform and installations of the hardware at customers’ location. Customers will be required to pay for installation fees at the front. The company will take a percentage of total sales made using the product.

### Phase 1 Solution Cost

Initial Cost (Investment):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Item | Quantity | $ Cost / Item | $ Cost (Subtotal) |
| 1. | Labour Cost |  |  |  |
|  | 1. Software Engineer | 3 | $55,000 | $165,000 |
|  | 1. Infrastructure Engineer | 1 | $55,000 | $55,000 |
|  | 1. Marketing & Sales | 1 | $50,000 | $50,000 |
| 2. | Office & Supplies |  |  |  |
|  | 1. Rent (per Year) | 5 | $9,600 | $48,000 |
|  | 1. Equipment |  |  | $5,000 |
| 3. | Server & Software (per Year) |  |  | $1,000 |
|  |  |  | Total (First Year) | $324,000 |

Cost to provide service:

|  |  |  |
| --- | --- | --- |
| No. | Item | Price / Item |
| 1. | Beacon (Raspberry Pi Zero W) | $12 - $15 |
| 2. | Server | Starts from $10 |
| 3. | Power BI License (per Month) | $13.70 |
| 4. | Extra Labour (Casual) | $25.00 per hour |

### Expected Revenue

**Product offering for stadium owner**

Indoor navigation package to provide a smarter stadium. This includes the installation of the beacons and integration into platform. Provides access to smart facility and safety and emergency features when it is released.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Item | Quantity | $ Cost / Item | $ Cost |
| 1. | Beacons + Installation | 250 | $120 | $30,000 |
| 2. | Configuration of platform and facility data for stadium |  |  | $150,000 |
| 3 | Integration with Safety and Emergency features |  |  | $125,000 |
| Total one-off fee | | | | $305, 000 |
| 4 | Maintenance of infrastructure (per year) |  |  | $50,000 |
| 5 | Platform maintenance (per year) |  |  | $50,000 |
| Total recurrent fee | | | | $100,000 |

**Product offering for vendors**

* Simple percentage-based commission of 14% of total sales

### Return on investment

Assuming that we have secured one customer with a stadium the size of ANZ Stadium that host a similar number of events.

**ROI = around 16 - 18 months (Includes first year of development)**

Projected income for the first year from stadium owner:

* Installation and configuration = $305,000
* Maintenance = $100,000

Total = $405,000

Projected revenue for vendors per year is calculated by:

= total attendance \* % who make purchase \* average spend \* commission

= 888,355 \* 0.25 \* $25 \* 0.14 = $777, 310

**Total Revenue Owner + Vendor** (per year) = $877, 310

Notes and assumptions

1. ANZ Stadium hosted 45 sporting events during 2018 with a total attendance of 888,355 (average of 19,741 per event)

<http://www.anzstadium.com.au/footer/about-us/attendance-figures/2018-attendance-figures/>

1. 25% of patrons purchase food/drinks
2. Average $25 spend per sale
3. Uber Eats commission is 30%

<https://www.uber.com/en-AU/drive/sydney/resources/pricing/>

# Solution Requirements

The information in this section is split into the following 3 columns.

Requirement No: This is a unique number for each requirement, e.g. Req-1

Description: This provides a description of the requirements.

How it is achieved: This describes at a high level how this requirement will be achieved.

## Functional Requirements

### Position Fixing

|  |  |  |
| --- | --- | --- |
|  | The solution must be designed to enable the sensing of a person’s position within the stadium. | This will be achieved by building a matrix of beacons in the stadium. A mobile device will be utilised to sense the beacons to determine which cell it is in. |
|  | The solution must be able to calculate distance between mobile devices. For example, between a vendor and a customer. | This will be achieved by utilising algorithms to calculate distances between two devices within the matrix. |
|  |  |  |

### Mobile Vending with Personal Delivery

|  |  |  |
| --- | --- | --- |
|  | The system must ensure that only users who access the website via scanning the QR code can open the store interface | QR codes will be written in specific format to distinguish between users who are accessing the store directly and who are accessing the store via scanning the QR Code at their seat |
|  | The solution must provide a user login screen in order for the customer to authenticate their identity. | The User Interface will provide a login screen. |
|  | A website will open after scanning the QR code on the ticket or the seat. | QR Codes will be placed on each seat which describe the location (e.g., Bay, Row, Seat number). A mobile device will utilise its camera to scan the QR code. |
|  | The QR code should be unique to each location. It will contain Bay, Row number and seat number at the location. | QR codes will be generated and placed at each relevant location and on printed and electronic tickets. |
|  | The solution must show the menu for the food and the customer is able to order the food online. | A user interface will be developed to enable ordering on the mobile device. The user interface will interact through APIs to complete the order process. Third party API’s will be used for certain functions e.g. Payment Gateway. |
|  | The solution should be able to show the estimated waiting time for the food delivery and can track the order. The order details, including order ID, items ordered, customer’s location and order status should also be shown on the user interface. | Algorithms will be developed to determine the expected waiting time for the order to be delivered. This algorithm will determine the most appropriate vendor to send the order to. The User Interface will provide feedback to the customer with all relevant information. |
|  | Once the order is submitted it must be routed to the vendor who has enough stock and is in the closest proximity to the customer. | An algorithm will be used to perform the routing of order to appropriate vendor. |
|  | The customer can leave feedback regarding their experience with their order | The User Interface will provide a mechanism both qualitative and quantitative feedback. |
|  | The solution must be able to monitor the stock levels in each basket. | The order basket handling module will maintain stock levels of each basket. |

### Reporting and Data Analysis

|  |  |  |
| --- | --- | --- |
|  | The solution must provide a dashboard and reporting interface. | The solution will utilise Power BI as the reporting and dashboard tool. |
|  | The dashboard should show high level information for sales and delivery performance. | Power BI dashboard with sales and performance information will be developed. |
|  | Detailed reports should show information for sales, performance and orders. | Detailed reports will contain information presented in tables and charts, e.g. time series. |
|  | All reports and dashboards should provide filters for dates/times, and where relevant for products, customers etc. | Filters will be built into the reporting templates. |

## Non-Functional Requirements

|  |  |  |
| --- | --- | --- |
|  | Security  The solution must be secure. It should be protected from unauthorised access. It should provide different levels of access depending on the type of user. The solution must also be protected against virus and malware attacks. | Encrypted data over network using certificates (HTTPS)  Store data in a secure environment – data will be stored in SQL database on Azure platform |
|  | Confidentiality  The data should be transmitted and stored in a secure and safe way. Data should only be accessed by authorised users. Data should only be created, edited, deleted and viewed by appropriate users. | Implement appropriate security groups and roles for users of the solution, this includes user types who provide administration services.  Implement audit solution to track changes to and access of the data. |
|  | Accessibility  The solution should be designed to allow use by all types of users. This includes people with disabilities and special needs. The solution should use appropriate assistive technology where required. | Design for use with screen readers  Use appropriate colour schemes for people with vision impairment  Ensure fonts can be scaled |
|  | Availability  Specific modules and functions classified as critical must always be available. Users must be notified if the system (or modules of the system) are unavailable at any time. SLAs must be in place with external providers. | Critical functions need to be identified  Ensure failover is operating for critical modules  Define Service Level Agreement with external providers |
|  | Reliability  The solution must work without failure for as long as possible. The solution development methodology and process should deal with coding problems and bugs. If an error is encountered it must be handled in an appropriate way. Any transactions involving external systems, e.g. payment gateway, must work appropriately. | Ensure appropriate testing methodology has been used  Provide a mechanism to track errors and bugs |
|  | Usability  The solution should be easy to learn and use. The UI component should be easy to navigate. Users should be able to complete tasks without help in the minimum possible time and steps. The solution should handle errors gracefully. | Utilise user experience methodologies to design the user interface  Provide informative error messages to the user |
|  | Maintainability  The solution should be easy to maintain. Clear roles and responsibilities should be adopted to ensure issues are dealt with quickly. | Establish levels of support (Level 1, Level 2 and Level 3)  Define roles and responsibilities for all parts of the solution and levels of support.  Hardware (sensors) must be able to be replaced quickly and with minimum effort and skill. |
|  | Scalability and Performance  The solution must be able to handle 1,000s of users and respond in real time.  The response times must be as close to real time as possible.  The solution must be designed to minimise resource utilisation (CPU, memory, disk). | The solution will be developed on the Microsoft Azure platform. This platform allows for rapid scaling of a solution. Load balancer may be used to improve performance and scalability. |

# Actors

The types of users of the solution are described below.

|  |  |
| --- | --- |
| Actor (User Type) | Description |
| Customers | This user type will utilise the ordering module of the application. |
| Vendors | This user type will interact with the Bluetooth positioning system. They will also utilise the order delivery module of the application. |
| Managers | This group of users will utilise the reporting, analytics and dashboards. |
| System users | This group will support the application on a day to day basis. They can monitor system usage etc. |
| Administrators | The administrators of the solution can turn on and off different modules, manage security etc. |

# Use Cases

|  |  |
| --- | --- |
| Use Case - 1 |  |
| Title | Create Order |
| Description | This use case describes the process taken by a customer when they create an order. |
| Primary Actor | Customer |
| Pre-Conditions | The customer has logged into the application. |
| Post-Conditions | The order was successfully submitted and created. |
| Main Success Scenario | 1. The customer identifies their location (Bay, Row number and Seat number) by scanning the QR-code on their seat. 2. The menu is displayed. 3. The customer will scroll through the menu and select products and quantity of products to order. 4. Repeat 3 until finished. 5. Press the “Submit Order” button. 6. The order will be sent along with the location to be processed. 7. A message will be returned confirming the receipt of the order. |
| Extensions | The customer adds special instructions for the order. |

|  |  |
| --- | --- |
| Use Case – 2 |  |
| Title | Route Order |
| Description | Once an order has been received it needs to be routed to the relevant vendors for picking. |
| Primary Actor | System |
| Pre-Conditions | The vendors are logged into the application and regularly transmitting their current location. |
| Post-Conditions | The order is routed to the relevant vendors for picking. |
| Main Success Scenario | 1. The vendor’s mobile device determines their current location (i.e. Bay). 2. The location information is sent back to the application multiple times each minute which updates their current location in the application. 3. When an order is submitted the routing algorithm will determine a short list of vendors to assign the order to.  * Does the vendor have the correct stock to supply the order? * Is the vendor nearby?  1. The order is assigned to a vendor on the short list. |
| Extensions | None |

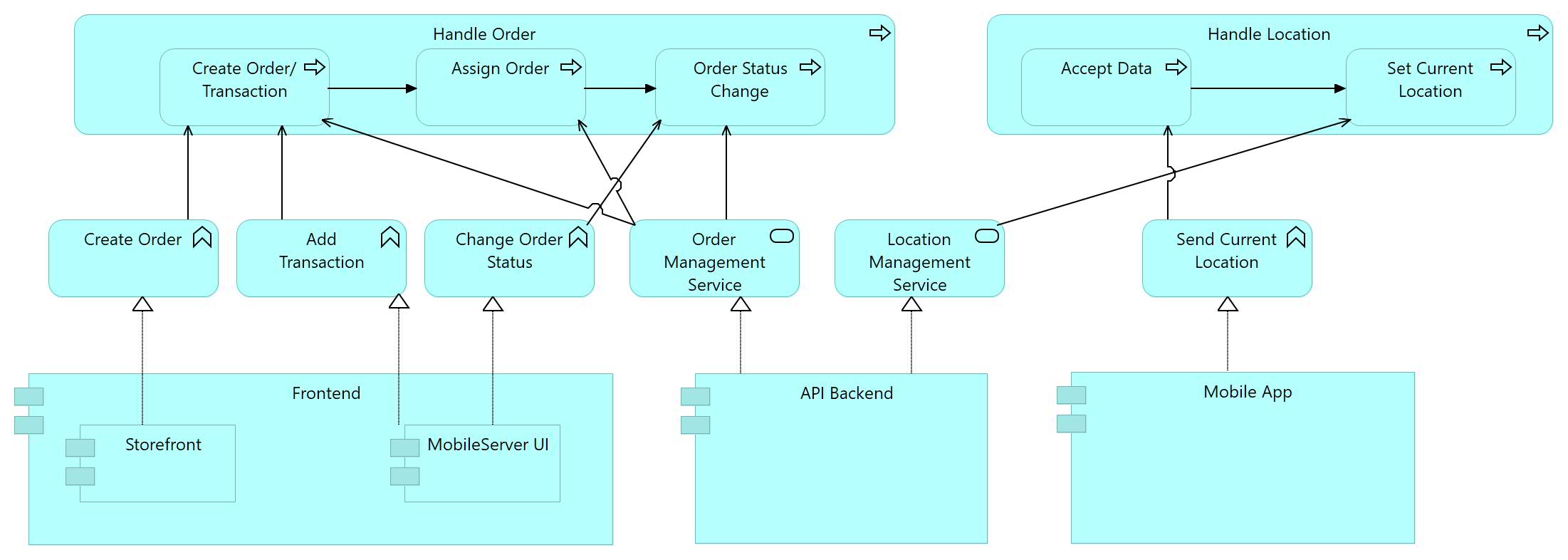
|  |  |
| --- | --- |
| Use Case – 3 |  |
| Title | Order delivered to the customer |
| Description | A vendor is assigned an order and delivers the order to the customer. |
| Primary Actor | Vendor |
| Pre-Conditions | The vendor is logged into the application and is on the short list. |
| Post-Conditions | The vendor has delivered the order to the customer. |
| Main Success Scenario | 1. The new order is routed to a vendor based on their stock level and distance from the customer. 2. The order is timestamped with the time the vendor was assigned the order. 3. The estimated delivery time is calculated, updated and sent to the customer. 4. The order is added to the vendor’s delivery list. 5. The vendor makes their way to the customer location. 6. The vendor delivers the products to the customer and timestamps the delivery time. 7. The system updates the actual delivery time against the order. 8. The order is cleared from the delivery list. |
| Extensions | The vendor makes a note regarding the order. |

|  |  |
| --- | --- |
| Use Case – 4 |  |
| Title | Customer provides feedback |
| Description | Once an order has been delivered the customer can provide feedback for the experience against the order. |
| Primary Actor | Customer |
| Pre-Conditions | The customer is logged into the application and has received their order. |
| Post-Conditions | The customer has provided feedback regarding the order. |
| Main Success Scenario | 1. The customer receives the message requesting optional feedback. 2. The customer can provide feedback in the form of stars from 0 to 5. 3. The customer can provide further descriptive feedback. |
| Extensions |  |

|  |  |
| --- | --- |
| Use Case – 5 |  |
| Title | Management reporting |
| Description | Provide reporting and analytics on the order data to assist with informed decision making. |
| Primary Actor | Management |
| Pre-Conditions | The manager is logged into the reporting solution. |
| Post-Conditions | The Manager has reviewed reports and dashboards. |
| Main Success Scenario | 1. The main dashboard is opened with the profit and performance information displayed. 2. The manager can drill down into the specific area of interest. 3. Each drill through report provides a more detailed view of the selected area. 4. The user can filter on date/time period, products, product category, delivery performance. |
| Extensions | None |

# Solution Architecture

## Software Design





### Customer [(Link)](https://github.com/MikeLiyantama/COMP6324-Project/tree/master/frontend)

The customer can access the application on their smart phone. The smart phone interacts with the Azure App Services on Azure platform for handling the order functionality. The application is an Order module that allows the customer to make an order, track the status of the order and to provide feedback. The camera on the smart phone is used to capture the QR code.

### QR Code [(Link)](https://github.com/MikeLiyantama/COMP6324-Project/blob/master/Misc/QR_Bay12Row2Seat3.jpeg)

The QR code contains encrypted data. In the case of the stadium, the QR code represents the Bay, Row Number and Seat Number, embedded into a unique URL for each seat that opens the web app once the QR code is scanned. Each seat in the stadium will have its own QR Code attached to it. The Bay, Row Number and Seat Number are included in the order creation process.

### Mobile Vendor [(Link)](https://github.com/MikeLiyantama/COMP6324-Project/tree/master/frontend)

The mobile vendor will also access the application on their smart phone. The smart phone interacts with the Azure App Services on the Azure platform. The Vendor module allows them to accept and deliver orders. The vendor’s current position will be updated periodically by a mobile app via the Azure App Services application updated using the Bluetooth beacons. Each vendor will have a basket of products. The order will be routed to vendors within the vicinity of the customer.

### Bluetooth Beacon [(Link)](https://github.com/MikeLiyantama/COMP6324-Project/tree/master/Scripts/beacon)

The stadium is composed of bays. Each bay has rows and seats. Each bay is fitted with a unique Bluetooth beacon driven by a Raspberry Pi device. The beacon emits a signal which is read by the smart phone and based on a positioning algorithm is translated to a location (i.e. bay) for the vendor. This position is used in the order routing algorithm.

### Azure App Service [(Link)](https://github.com/MikeLiyantama/COMP6324-Project/tree/master/Backend)

The Azure App Service hosts the IOT Stadium application. APIs expose methods to the smart phone for both the customer and vendor modules. The application modules interact with the Azure SQL database to store and retrieve data. The App Services also contains methods for order routing.

### SQL Database

The Azure SQL database contains the data structures that hold all of the data used in the application. This includes customer, vendor, location and order related data. The data is also used in the reporting and analytics module.

### Reporting / Analytics [(Link)](https://github.com/MikeLiyantama/COMP6324-Project/blob/master/Misc/DashBoard2.pbix)

The Reporting and Analytics is enabled through Microsoft Power BI. This module will provide insights to the data collected through the application. This module is aimed at management of the facilities. This module hosts the reporting and analytics templates which are refreshed periodically to provide up-to-date data. Financial and performance data is provided to enable informed decision making.

### Microsoft Azure Platform

The Microsoft Azure Platform is used to host the solution. This platform provides the current requirements of the solution as well as future requirements. Azure is also scalable as needed.

## Deployment Design





### Beacons

Each bay of the stadium will have one beacon with a unique identifier installed. The beacons are utilising the Raspberry Pi Bluetooth module. The beacon will emit a signal that contains the unique id of the beacon. The Vendors mobile device will interpret the signal strength to determine the current position and update the application.

### Mobile Devices

The customer and vendor will utilise mobile devices to interact with the application. The customer will make orders while the vendor will update their current position, have orders routed to them and then deliver the order. The mobile device will utilise their mobile network (3G/4G/LTE) cell towers to connect to the internet. The data will be securely transmitted using HTTPS.

### Desktop Devices

The desktop devices will be used by management to view reports, dashboards and analytics. They will connect through the internet to the Power BI server. The data will be securely transmitted using HTTPS.

### Cell Towers

Mobile devices utilise their network provider to connect to the internet through the nearby cell tower. The data will be securely transmitted using HTTPS.

### Internet

The internet is used to connect all devices to the Microsoft Azure platform. The data will be securely transmitted using HTTPS.

### Microsoft Azure Platform

The Microsoft Azure platform will host the various components of the application solution. The application currently utilises the Azure Apps Service, Azure SQL Database and Power BI. The Azure platform can scale as the user base of the solution increases.

## Hardware Design

|  |  |
| --- | --- |
| Customer centred Each seat in the stadium will have a QR Code that identifies its unique location. The location is made up of: Bay, Row Number and Seat Number. The customer will use the camera on their smart phone to capture the location for the order. |  |
| Vendor centred The stadium will have a Bluetooth beacon located at the centre of each bay. The current beacon is utilising the Bluetooth module of a Raspberry Pi. The mobile device of the vendor will determine which beacon they are closest to and transmit their location to application multiple times per minute. This information is used in the order routing module. |  |

# Software Requirements

|  |  |
| --- | --- |
| Component | Description |
| Platform | Microsoft Azure (Cloud) |
| Azure Resources | SQL Database  Azure App Services  Power BI |
| Tech Stack | API Server – C# (ASP.NET Core)  Web Application – HTML, JavaScript, CSS |
| Database | Azure SQL Database |
| Mobile Browser Support | Firefox  Chrome  Safari on IOS |

# Hardware Requirements

The solution will be deployed on the Azure platform which will scale according to the demand.

|  |  |
| --- | --- |
| Resource | Description |
| Disk | 5GB |
| Beacon | Raspberry Pi 3 B+ (Bluetooth 4.1), or  Raspberry Pi Zero W |
| Camera | From mobile device |
| Bluetooth enabled | Mobile devices & Raspberry Pi |
| Mobile Device | iPhone, Android |
| Desktop Device | Windows, Linux, macOS |
| Seat | Plastic Stadium Seat |

# Type of data that will be used in the solution

The solution will store data consisting of the following:

1. Beacon unique identifier/ mobile device identifier will be used for the positioning.
2. Assets (i.e. seats, bathrooms, kiosks, vending machines, exits etc) will be mapped into the solution database.
3. Products and services such as food items, drinks, merchandise etc will be stored in the database and used for the online ordering system.
4. Financial transactions for online purchases.
5. User account information (optional).

# Management of solution

|  |  |
| --- | --- |
| Component | Description |
| Server Hardware/ Operating System | The Azure platform will support this infrastructure. This component will be managed through the Azure subscription. |
| Beacons | The Solution Team will be responsible for the installation and maintenance of the beacons in the stadium. |
| Application | The Solution Team will be responsible for developing, installing and maintaining the software application on the Azure platform. |
| Administration | The Solution Team will be responsible for administering the application. |
| User Support | The Solution Team will employ a third party to handle user enquiries and to escalate issues. |

# Solution Roadmap

The solution roadmap provides a description of the complete set of products that form the total solution. The complete solution will be delivered in a phased approach with the core functionality developed first.

|  |  |  |
| --- | --- | --- |
| Module | Description | Delivery |
| Indoor Position Fixing | Indoor position fixing using a matrix of Bluetooth beacons. This is used in phase one of the solution to assist with the Mobile Vendor with Personal Delivery module. It will also be used as the core for the indoor navigation. | May 2019 |
| Mobile Vendor with Personal Delivery | This feature will allow the user to order food and snacks from their seats and have it delivered to them. It includes online ordering, payment, order tracking and delivery tracking. | May 2019 |
| Indoor Navigation | Indoor Navigation will guide patrons and workers in real-time through the stadium and visualise their location on a digital map. This feature will utilise the infrastructure implemented during the Position Fixing phase. | December 2019 |
| Facility Usage | This feature will allow the user to find particular areas of the facility that have some properties, for example to find the bathroom with the smallest queue. This feature will integrate with the Navigation module. | December 2019 |
| Evacuation | This feature will provide an evacuation plan for the user that is sent to the phone. This feature will integrate with the Navigation module. | April 2020 |

# User Acceptance Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Req Id | Description | Expected Result | Notes | Reference/ Components |
| Req-1 | Search for the real-time locations of vendors and locations of customers that has made orders. | The coordinates of the location cell that a vendor and a customer is in are recorded in the database. | The Bluetooth on the mobile device will be utilised to sense the beacons to determine which cells the vendors are in.  Locations of customers should be recorded once customers make orders online. | [6.1.2](#_QR_Code) QR code  [6.2.1](#_Beacons) Beacons  [6.1.8](#_Microsoft_Azure_Platform) Microsoft Azure Platform |
| Req-2 | Calculate the distance between two users, such as a vendor and a customer. | The Manhattan distance between the two users. | The Manhattan distance will be calculated based on the coordinates  Of both users from the database. | [6.1.6](#_SQL_Database) SQL database  [6.1.8](#_Microsoft_Azure_Platform) Microsoft Azure Platform |
| Req-3 | Verify whether it is compulsory to login in before making any order. | It pops up a login screen after scanning the QR code. An error message will be displayed after invalid login. | Authenticate customers’ valid identity for user safety and privacy consideration. | [6.2.5](#_Internet) Internet  [6.1.5](#_Azure_App_Service) Azure App Service  [6.1.8](#_Microsoft_Azure_Platform) Microsoft Azure Platform |
| Req-4 | Check what will be displayed in the user interface after scanning the QR code and login. | A webpage will be provided after valid login. | QR codes will be placed on seats and tickets and be scanned by a camera on mobile device. | [6.1.2](#_QR_Code) QR code  [6.2.5](#_Internet) Internet  [6.1.6](#_SQL_Database) SQL database  [6.1.8](#_Microsoft_Azure_Platform) Microsoft Azure Platform |
| Req-5 | Validate QR code uniqueness. | Each QR code reveals different location cell. | QR codes will be generated and placed at each relevant location and on printed and electronic tickets. | [6.1.2](#_QR_Code) QR code  [6.3.1](#_Customer_centred) Customer centred |
| Req-6 | Order food online with the menu displayed in the user interface. | The order is made and completed successfully. | APIs will be interacted in the whole process. Third party API’s will be used for certain functions e.g. Payment Gateway. | [6.2.2](#_Mobile_Devices) Mobile Devices  [6.2.4](#_Cell_Towers) Cell Towers  [6.2.5](#_Internet) Internet  [6.2.6](#_Microsoft_Azure_Platform) Microsoft Azure platform |
| Req-7 | Show the display on the user interface after the order is finished (food delivered). | The order details, such as order ID, items ordered, vender ID, customer’s location and order status is shown on the user interface. Feedback can be added and recorded as well. | The estimated delivery time will be calculated using algorithm. | [6.1.3](#_Mobile_Vendor) Mobile Vendor  [6.1.4](#_Bluetooth_Beacon) Bluetooth beacon  [6.1.5](#_Azure_App_Service) Azure App Service  [6.1.8](#_Microsoft_Azure_Platform) Microsoft Azure Platform |
| Req-8 | Look into the order status during the process. | Each order status (order submitted, assigned, delivered) is showed correctly. |  | [6.1.4](#_Bluetooth_Beacon) Bluetooth beacon  [6.1.5](#_Azure_App_Service) Azure App Service  [6.1.8](#_Microsoft_Azure_Platform) Microsoft Azure Platform |
| Req-9 | Verify the correctness of given route and assigned vendor for an order. | The route is in the closest proximity to the customer and vender has enough relevant stock. | The assignment of vendors will be calculated using algorithm. | [6.1.3](#_Mobile_Vendor) Mobile Vendor  [6.1.4](#_Bluetooth_Beacon) Bluetooth beacon  [6.3.2](#_Vendor_centred) Vendor centred |
| Req-10 | Post a feedback with a score and a comment when an order is delivered. | The feedback is should in the user interface and recorded in the database. | The feedback is both quantitative and qualitative. | [6.1.1](#_Customer) Customer  [6.1.5](#_Azure_App_Service) Azure App Service  [6.2.6](#_Microsoft_Azure_Platform) Microsoft Azure platform |
| Req-11 | Verify the stock levels for each vendor. | The stock level is recorded in the database. | The order basket handling module will maintain stock levels of each basket. | [6.1.3](#_Mobile_Vendor) Mobile Vendor  [6.1.6](#_SQL_Database) SQL database |
| Req-12 | Review dashboard and reports for a certain period. | There is a dashboard tool available for generating reports. | Power BI will be utilised. | [6.1.6](#_SQL_Database) SQL database  [6.1.7](#_Reporting_/_Analytics) Reporting and Analytics  [6.2.3](#_Desktop_Devices) Desktop Devices  [6.2.6](#_Microsoft_Azure_Platform) Microsoft Azure platform |
| Req-13 | Examine high level sales and delivery performance in a certain period. | The information, such as profits, trends, is presented in a simple charm and table. | Power BI will be utilised and the information is presented in a clear and concise style. | [6.1.6](#_SQL_Database) SQL database  [6.1.7](#_Reporting_/_Analytics) Reporting and Analytics  [6.2.6](#_Microsoft_Azure_Platform) Microsoft Azure platform |
| Req-14 | Examine detailed reports that reveals information for sales, performance and orders. | The information is presented in charms and tables. | Power BI will be utilised. | [6.1.6](#_SQL_Database) SQL database  [6.1.7](#_Reporting_/_Analytics) Reporting and Analytics  [6.2.6](#_Microsoft_Azure_Platform) Microsoft Azure platform |
| Req-15 | Verify the filter availability of dashboards and reports. | Certain filter requests are satisfied in all dashboards and reports. | The filter options include dates/times, and where relevant for products, customers, etc. | [6.1.7](#_Reporting_/_Analytics) Reporting and Analytics  [6.2.6](#_Microsoft_Azure_Platform) Microsoft Azure platform |
| Req-16 | Validate the security of the solution by attempting authorised and unauthorised accesses for different users (such as customers, venders, system administrative and managers). | Only authorised access is allowed, and the levels of access varies depending on the type of user. | Data is encrypted by utilizing HTTPS and stored on Azure platform. | [6.2.4](#_Cell_Towers) Cell Towers  [6.2.5](#_Internet) Internet  [6.2.6](#_Microsoft_Azure_Platform) Microsoft Azure platform |
| Req-17 | Access data with different authorities for different users (such as customers, venders, system administrative and managers). | The type and forms of Data varies with different users. Most users can only view data, and only users with relatively high authority can download, create, edit and delete data. | The changes and access of data will be tracked. | [6.1.6](#_SQL_Database) SQL database  [6.2.4](#_Cell_Towers) Cell Towers  [6.2.5](#_Internet) Internet  [6.2.6](#_Microsoft_Azure_Platform) Microsoft Azure platform |
| Req-18 | Verify whether the user interface is designed as accessible considering all types of users. | Fonts can be scaled and there are different colour schemes available for users to choose. | Special needs for different user groups will be considered, such as people with vision impairment or poor eyesight. | [6.1.1](#_Customer) Customer  [6.1.3](#_Mobile_Vendor) Mobile Vendor  [6.1.5](#_Azure_App_Service) Azure App Service |
| Req-19 | Verify whether the SLAs is in place and mock the food order and delivery procedure | The related functions and models run smoothly. |  | [6.1.5](#_Azure_App_Service) Azure App Service  [6.1.8](#_Microsoft_Azure_Platform) Microsoft Azure Platform |
| Req-20 | Examine the robustness of the solution. | If an error is encountered, it is handled in an appropriate way. |  | [6.1.5](#_Azure_App_Service) Azure App Service  [6.1.8](#_Microsoft_Azure_Platform) Microsoft Azure Platform  [6.2.5](#_Internet) Internet |
| Req-21 | Evaluate the readability of the user interface on the mobile device and dashboards. | The interface can be easily understood and utilised by those who have not been exposed to the solution | Informative messages will be sent to the user when some error occurs or when the operation runs successfully | [6.1.5](#_Azure_App_Service) Azure App Service  [6.1.8](#_Microsoft_Azure_Platform) Microsoft Azure Platform |
| Req-22 | Examine the maintainability of the solution | All resource deployed can be easily monitored on a dashboard. |  | [6.1.5](#_Azure_App_Service) Azure App Service  [6.1.7](#_Reporting_/_Analytics) Reporting and Analytics  [6.1.8](#_Microsoft_Azure_Platform) Microsoft Azure Platform |
| Req-23 | Examine the scalability and performance of the solution | New resources can easily be deployed on demand. | The solution will be developed on the Microsoft Azure platform, which allows for rapid scaling of a solution | [6.1.5](#_Azure_App_Service) Azure App Service  [6.1.7](#_Reporting_/_Analytics) Reporting and Analytics  [6.1.8](#_Microsoft_Azure_Platform) Microsoft Azure Platform |

# Working Solution

The working solution is hosted on the Microsoft Azure platform.

### Backend APIs

<http://iotstadium.azurewebsites.net/api/>

APIs

|  |  |
| --- | --- |
| customer | |
| Login | Login to application mechanism |
| basket | |
| SetBasketLocation | updates the current location of the basket |
| RestockBasket | resets the contents of the basket to full for each item |
| order | |
| GetOrder | Get the specific order |
| Undelivered | Get a list of undelivered orders from a particular basket |
| GetOrderStatus | Get the status of a particular order |
| CreateOrder | Create a new order |
| DeliverOrder | Vendor completes delivery of the order |
| UpdateCustomerFeedback | Updates the feedback details for a particular order |
| UpdateDeliveryTime | Update the delivery time stamp for a particular order |
| AcceptOrder | For the Basket to accept an order. Update the order with basket\_id |
| CancelOrder | Customer cancels the order, stock is replaced into the basket |
| products | |
| GetMenu | Get the list of products |
| GetProduct | Get the details of a product |

### User Interface

The frontend application can be accessed through the following URL:

<https://iotstadiumui.azurewebsites.net/>