Group 15

Sprint1

UWA Carpark Bay Booking System

# 1.General Goals

A lightweight system for UWA staff without a parking permit to request a parking space reservation. The proposed system should provide the ability for both parties to update/cancel the bookings based on a set of rules. The system should allow Unipark staff to confirm booking requests and the system should also notify the customers about whether the request is approved or not. According to the booking information, an A3 Reservation PDF will be printed out at the start of the day for Unipark staff to display at the parking space.

# 2.Current System

The current booking system simply uses Excel spreadsheets to record the booking information, and there is no viable online system for customers to view all available parking spaces. Customers have to call the UniPark department to make a reservation, which is very inefficient and labor-intensive. However, it does have some features that our client still want to maintain: the reserved bays should be color coded as ‘yellow’, and there should be a 15-minute gap marked with ‘green’ between each booking.

# 3. Proposed System

## 3.1 Overview

The proposed system is a platform presenting the availability of UWA parking spaces for authorized customers. It allows convenient and efficient reservations, what else?

## 3.2 Functional Requirements

(1). Users are able to view available bays and when it is available.

(2). Users are able to request a reservation online by submitting their email and booking times.

(3). Unipark staff are able to immediately confirm bookings made by UWA staff via email link.

(4). Once the request is processed, users will get notified.

(5). A3 Reservation PDF can be generated for each successful reservation so that Unipark staff can then better manage the parking spaces.

(6). Both Users and Unipark staff are able to edit or delete bookings or requests.

* Unipark staff are notified of any booking changes made by users
* Unipark staff have their own admin interface

(7). Unipark staff are able to make a reservation on behalf of users.

(8). Google map integration with the user view of the application.

## 3.3 Nonfunctional Requirements

(1). A simple user interface. (User friendly)

(2). Suitable for UWA use i.e. conforms to UWA’s website design requirements

(3). Usage of CheckBox for user declaration. (confirm the booking is not for a current permit holder) (need to confirm if UniPark wants this - James was initially unsure)

(4). (what can be inherited from the current booking system in the sense of nonfunctional requirement?)

* That it works
* Safe and secure for users to use

### 3.3.1 User Interface and Human Factors

FOs are quite old and require a simple user interface

* Minimise the amount of buttons/clicks required for a user to book
* Minimise number of pages
* Possible page layout (2 pages): List of car parks > calender containing all the bays in that car park
* Make printing as simple as possible e.g. 1 button print all saved data

Must conform to UWA design standard as it is representing UWA

### 3.3.2 Documentation

(1). The code for building the system will be fully documented for future maintenance.

(2). The documentation should be easy-to-understand so that it can transfer the “Why” behind the code to the maintenance programmer for future reference.

(2). Maintenance programmers should be able to comprehend the code and modify where it is necessary for updates.

(3). Other organizations might also be addressed by the documentation if they want to find out whether the system is also suitable for them to use.

### 3.3.3 Hardware Consideration

(1). Processor

(2). Memory specs?

(3). Hard Drive

(Not a big deal for UWA I guess, should we add this section?)

If UWA carpark is expanded and a lot more parking spaces available, or the Unipark department starts to keep historical information for security consideration, the server should be upgraded vertically, replacing old hardwares listed above with more powerful ones. (trivial thing, mention it or not?)

### 3.3.4 Performance characteristics

Server, data management, interaction between front and backend, network connection,

data size. (probably no need for this section )

### 3.3.5 Error Handling and Extreme Conditions

Hardware related errors? Not likely

Probably database related, like logical errors, overflow, infinite loop, out of bounds?

Data back up allows for extreme conditions, storage failure, power outage

* Sever update delay/failure
  + E.g. if a user makes a booking and it does not successfully go through
* Email does not send successfully
* Two users book the same space simultaneously
* Printing error - need to be able to reprint A3 sheet

### 3.3.6 Quality Issues

### 

### 3.3.7 System Modification

The potential candidates for later modification and what sorts of modification it can be:

(1). Increase database performance by migrating to a more efficient platform (i.e MongoDB).

(2). Make user interface more appealing (i.e different looking during different festivals, UWA celebration, Christmas).

(3).

### 3.3.8 Security Issues

* Personal information leak: emails and location of user’s vehicles
* Users lying about not having a permit

### 3.3.9 Resources Issues

* Don’t have aerial view images of parking spaces (might be able find somewhere)

## 3.4 Constraints

* No previous system to work off of
* Limited time
* Team members have limited availability
* Not everyone knows flask (learning constraint)
* Need to work with the current UWA system