National College of Ireland

Technical Report

Smart HotDesking App

15/05/22

BSc (Honours) in Computing

BSHCSD4

2021/2022

Lee Brady

X18434514

X18434514@student.ncirl.ie

Contents

Executive Summary

This report lays out the purpose and delivered functionality of my Final Year Project, which is a Smart HotDesking web application. This project differs greatly in technology from my mid-point submission, which I will explain in the Technology subheading. While what has been delivered differs from my initial goal, the project is overall a lot more sound and robust than what was envisioned in semester 1. The scope of the project has been reigned in to deliver a well-functioning ‘proof of concept’ project.

This report follows the same structure as the mid-point technical document, but the content has been completely overhauled to reflect the project’s technology and design that was decided on after approval from my project supervisor.

1.0 Introduction

1.1. Background

This project made sense for me to pursue as I had seen how it is affecting businesses in Dublin during my internship in 3rd year. A problem my employer is currently facing, is that an increase of hiring that took place during the ‘work from home phase’ of the pandemic, is now becoming a logistical issue as employees return to the office. If the company has more employees in a location than seats in their office, how can everyone return to that office?

To purchase more office space in Dublin, right now, is prohibitively expensive, with the country seeing its highest annual property price rise in 6 years in 2021. (Weston, 2021)

Hot desking is a term that refers to the practise of multiple people sharing the same physical workspace. Currently, some businesses are reporting their use of hot desking as an interim solution to this problem. Dave Egan, head of recruitment at PM Group said in December of this year:

“*We are also moving from that traditional idea of everyone having their own desk to a more flexible model. Some may have a designated desk but most will share.”* (Murray, 2021)

1.2. Aims

For my project to work, I’m going to carry through it the use case that the User works for a company that employs a hybrid work model i.e. work from home some days, work at the office on others. In some cases, hot desking is the goal of an organisation from the get-go, for businesses where being on the same schedule doesn’t impact their bottom line. (Alara, 2016)

The goal of this project is to provide a web application that facilitates smart hot desking. This means that at its core, the project will be a scheduling application. The User will be able to book time at a desk in the office.

The app will contain concepts that encourages employees (users) to book a desk slot at the office, or to work from home. This includes seeing which desks are already being used by other colleagues, and if someone already has the auto scheduler set up on a certain day to take up a desk.

1.3. Technology

The project will take the form of a web application. It will be developed in Visual Studio Code using HTML5, jQuery, JavaScript, some light CSS and PHP to deploy the webapp on Heroku. The main external library I will be using to develop the webapp is DevExtreme. DevExtreme is a self-described "HTML5 JavaScript Component Suite for Responsive Web Development" (DevExtreme, 2022). I came across DevExtreme while trying to figure out the UI layout for my webapp. I sorted through its documentation and saw one of their UI devkits, which is DxScheduler, and thought with some tweaking it could make a great scheduling system. The main configuration changes I made to their provided platform was setting up the Auto Scheduler feature I had desired for my project. Through their main Integration documentation (DevExtreme, 2022) I found that using jQuery to develop on their stack would be the best approach to create my project.

My initial Mid-Point submission was based around creating a bespoke Java Android application. One of the main things that drew me to using a web-based approach was the Adaptability aspect of jQuery + DevExtreme. This page showed me that there was a way to create a touch friendly version of the webapp, which led to me ultimately making the switch from Java + Android to a responsive webapp. The mobile version of this application does everything I wanted it to do in a touch friendly manner.

1.4. Structure

The introduction section of this report gives a summary of my work on the project at the time of completion.

The System section will describe the requirements for the project, its design and system architecture, the GUI and what testing will be conducted.

Any references used in the research for the project will be listed in a bibliography section at this report's conclusion.

The Appendix includes relevant supporting documents, like the Project Proposal and any monthly journals completed at this point.

2.0 System

2.1. Requirements

2.1.1. Functional Requirements

ID	Functional Requirement	Description
FR_1	Create Desk Booking	The User interacts with the calendar element and is presented with multiple options, allowing them to create a desk booking.
FR_2	Create Recurring Desk Booking	The User continues from FR_1 but selects Repeat, initiating the Auto Scheduler and allowing them to create a series of desk bookings.
FR_3	Edit Desk Booking	The User selects an already existing desk booking and can alter any of the options related to the booking.
FR_4	Edit Recurring Desk Booking	The User selects an already existing **recurring** desk booking and is prompted to either edit only that particular booking, or to edit the series of bookings that the Auto Scheduler has set up.
FR_5	Cancel Desk Booking	The User selects an already existing desk booking and can select to delete the booking.
FR_6	Cancel Recurring Desk Booking	The User selects an already existing **recurring** desk booking and is prompted to either delete only that particular booking, or to delete the series of bookings that the Auto Scheduler has set up.
FR_7	Group Existing Desk Bookings	The User can toggle between grouping the desk bookings on the calendar by Employee, Desk and Importance. These options are set in FR_2 and can be altered in FR_3/FR_4. This allows the user to easily identify different possibilities about the existing desk bookings, i.e., is their preferred desk booked, will there be other employees in the office etc.
FR_8	Change Date Range	The User can use the toolbar in the top left of the app to change the dates being displayed to them. The calendar will default to displaying the current Work Week.
FR_9	Change Calendar View	The User can use the toolbar in the top right of the app to change the calendar view being displayed. By default, the calendar being displayed is the Work Week (Monday through Friday).

2.1.1.1. Use Case Diagrams

2.1.1.2. FR_2: Create Recurring Desk Booking

2.1.1.3. Use Case

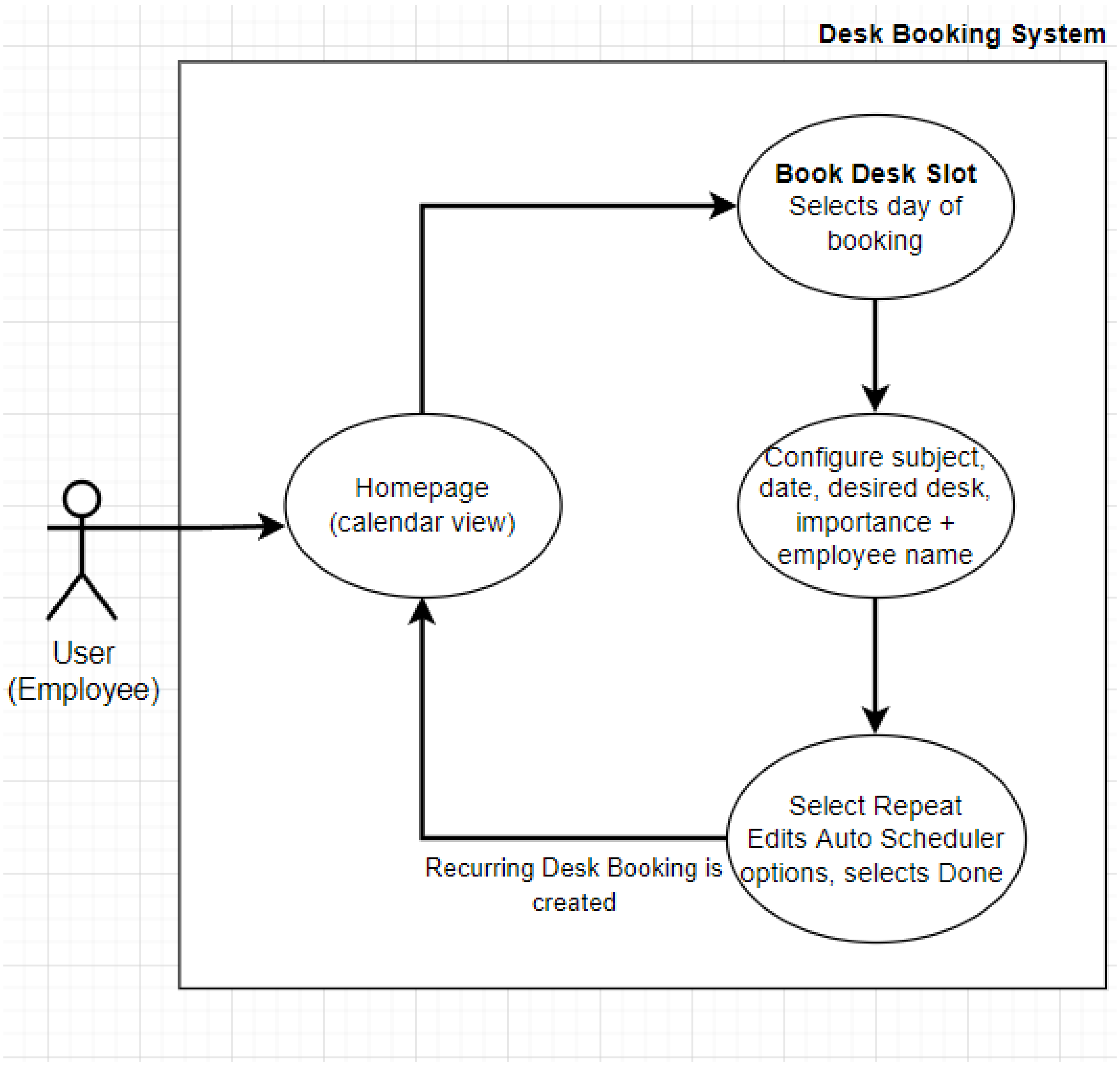
Scope

The scope of this use case is to showcase the registration process of the Smart HotDesking Web App.

Description

This use case shows how the User interacts with the calendar element and is presented with multiple options, allowing them to create a desk booking. The User toggles the Repeat option, which allows them to configure the Auto Scheduler. High priority, the main differentiator between this and a simple booking calendar is the Auto Scheduler function.

Use Case Diagram


Desk Booking System
User (Employee)
Homepage (calendar view)
Book Desk Slot
Selects day of booking
Configure subject, date, desired desk, importance + employee name
Select Repeat
Edits Auto Scheduler options, selects Done
Recurring Desk Booking is created

Flow Description

Precondition

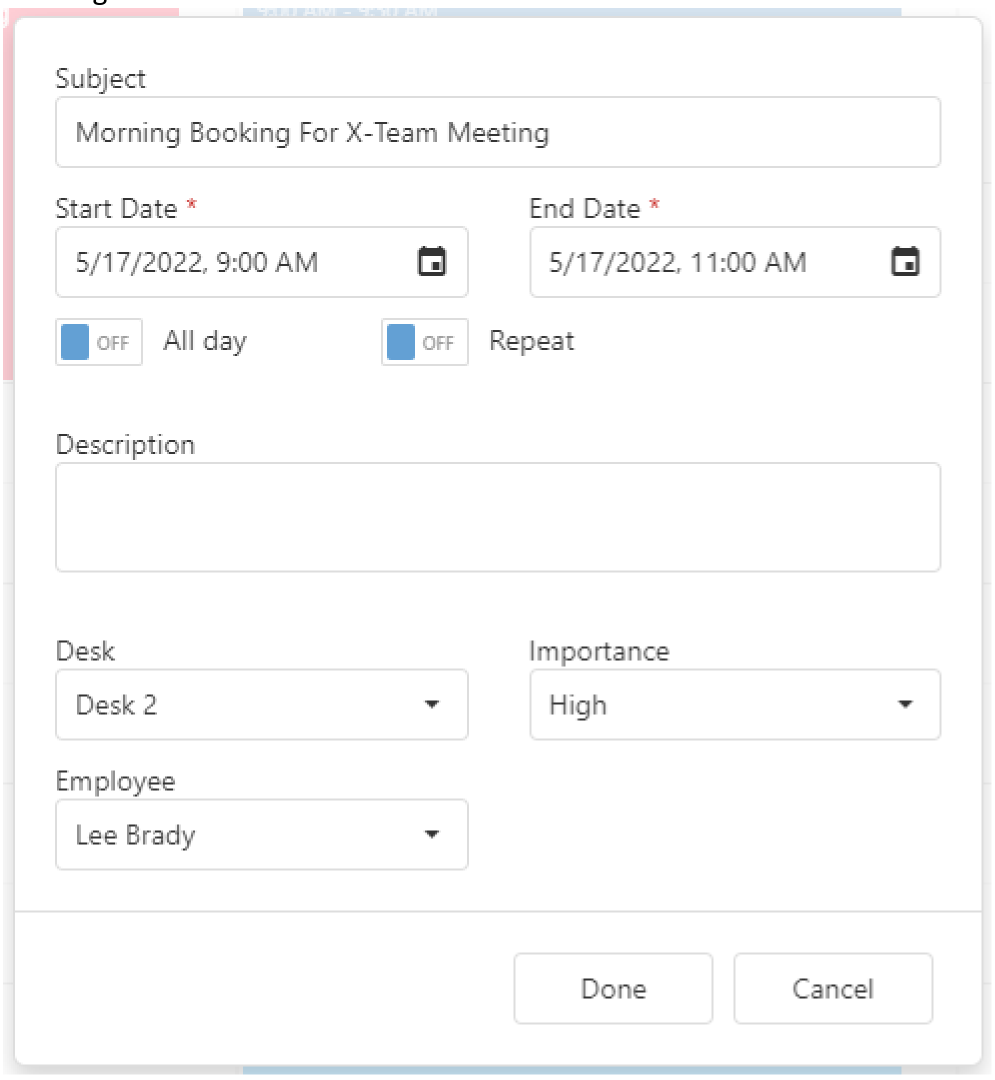
The User has accessed the web app.

Activation

The booking system starts when a User selects any empty time slot on the homepage.

Main flow

1. The web app displays the calendar view (homepage).
2. The User double clicks/taps an empty timeslot on the calendar.
3. The booking configuration screen appears and the web app focuses on this screen, allowing the user to configure multiple options about their booking



Subject: Morning Booking For X-Team Meeting

Start Date *: 5/17/2022, 9:00 AM

End Date *: 5/17/2022, 11:00 AM

OFF All day

OFF Repeat

Description:

Desk: Desk 2

Importance: High

Employee: Lee Brady

4. The User desires to set up a recurring series of desk bookings, so they don't have to manually create multiple. The User selects Repeat. The booking configuration popup expands, displaying the Auto Scheduler aspect of the system.
5. The User configures how often they want their booking to occur, when it should repeat, and if it should ever stop repeating.
6. The User selects Done, and is returned to the calendar view with their booking now included. The User can validate if the Auto Scheduler has successfully created their recurring desk booking by navigating to dates later in the calendar and seeing that their desk booking exists in the future as well.

Termination

The User is now on the home screen, after successfully creating a recurring desk booking.

Post condition

The system goes into a wait state.

2.1.1.4. FR_7: Group Existing Desk Bookings

2.1.1.5. Description & Priority

2.1.1.6. Use Case

Scope

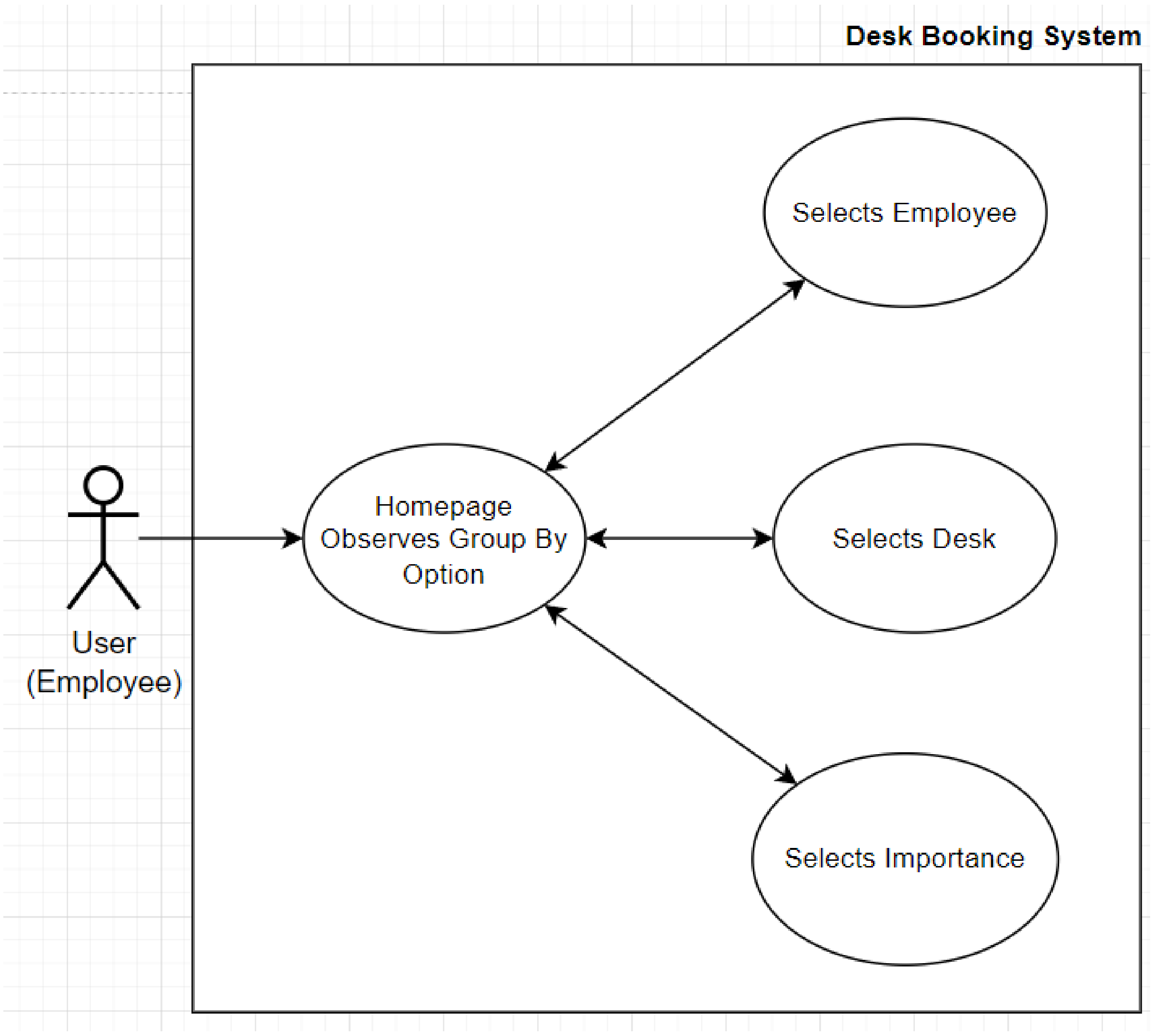
The scope of this use case is to showcase the book desk operation of the Smart HotDesking Web App.

Description

The User can toggle between grouping the desk bookings on the calendar by Employee, Desk and Importance. These options are set in FR_2 and can be altered in FR_3/FR_4.

This allows the user to easily identify different possibilities about the existing desk bookings, i.e., is their preferred desk booked, will there be other employees in the office etc.

Use Case Diagram



Flow Description

Precondition

The User has accessed the web app.

Activation

The resource filtering system is activated by default, and defaults to Grouping by Employee. The user activates change in the system by selecting either Desk or Importance.

Main flow

1. The web app displays the calendar view (homepage).
2. The User is by default viewing the existing bookings as being grouped by the Employee value that is selected for each booking. This is shown by the fact that each Employee, Desk and Importance have their own unique colour value. Eg. If Lee Brady is the Employee assigned to a desk booking, the UI reflects this by displaying the booking as red, if the group by option is toggled to Employee.
3. The User can change the Group by option underneath the calendar view by tapping/clicking on the various options.

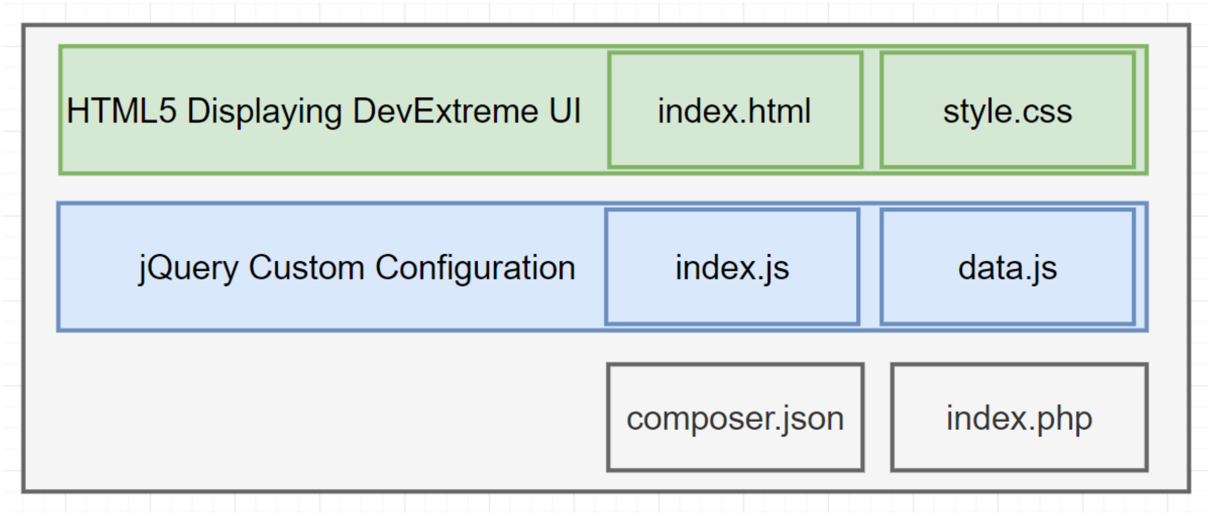
Termination

The system remains on whatever resource grouping option the User has selected.

Post condition

The system goes into a wait state.

2.2. Design, Architecture and Implementation



The basic architecture of this project comes from customising and developing around the open source dxScheduler UI suite.

The HTML5 index.html file is displaying the UI components from dxScheduler on the home page of the website. This html file includes the jQuery needed to display the needed content from the DevExpress jQuery library.

Style.css is a very simple stylesheet, as most design styling is being done when the index.html file is calling the DevExpress cdn at line:

```
    <link rel="stylesheet" type="text/css"
href="https://cdn3.devexpress.com/jslib/21.2.7/css/dx.common.css" />
```

Index.js is using JSON to configure the custom options I've written to access what we want from DevExpress Scheduler. This service has a huge library with many different options, so I followed documentation from their hub to create the desk booking system.

Data.js is more JSON being sent through JavaScript, this time to populate the calendar with example desk bookings.

Composer.json and index.php exist just to make this jQuery-based web app friendly to Heroku, for deployment. As these are one-line files that only need to be in the local remote, they have not been included in the source code on GitHub.

The core of this web app relies on the DevExtreme Ui suite. This decision was made to improve on the UI that had been created at the mid-point, which was rudimentary. Overall, I wanted to spec into using something lightweight and malleable, and I found the DevExtreme library to be both.

The library was huge and had lots of room for custom code injection. The library is accessed in theindex.html file and the index.js file. The code I have written in both, as well as the data.js file is really where the development I carried out lies.

Because the code I've written to include what I wanted from this library is very straightforward, I will use this implementation section to discuss the research I conducted that led to my completed project. Below will be various decisions I made based on the documentation available.

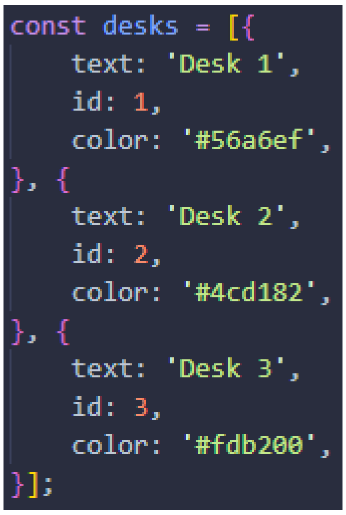
The Move from an Android to a Web Based application

Earlier in semester 2 I discussed changing the technology of the project with my supervisor, which was met with positive feedback. We agreed that a Java based Android app was a weak technology stack for this project.

From there I wanted to create a web app based on Python and Django, and I explored this for some time. I was trying to create a system that linked with Outlook calendar, and followed a tutorial (Microsoft, 2022) to make a system that linked with the MSGraph API and allowed the User to make calendar appointments that would sync with their Outlook calendar.

I really wanted to integrate the project with an outside calendar API, but ultimately the configuration of Outlook and Google Calendar did not allow for custom configurations, like specifying that the desk booking or 'appointment' was to be recurring, or that it was to take place at a certain desk etc. DevExtreme even has some Google Calendar integration (DevExtreme, 2022) which excited me, but I found that it wouldn't send the information needed to deliver on the project's goal.

I took one of the demos available on DevExtreme and used it to implement the reccurenceRule (DevExtreme, 2022) I found in the documentation. This allowed for the Auto Scheduler to be included in the project. Another strong aspect I'm glad I could include was the resources[] array (DevExtreme, 2022) which is found in the index.html file. This, after being populated by data.js allows the user to easily 'filter' the displayed desk bookings by 3 categories, allowing for an easier decision-making process on whether to book a desk at a certain time.



```
const desks = [{
    text: 'Desk 1',
    id: 1,
    color: '#56a6ef',
}, {
    text: 'Desk 2',
    id: 2,
    color: '#4cd182',
}, {
    text: 'Desk 3',
    id: 3,
    color: '#fdb200',
}];
```

1 Example of the resource attribute a desk booking can have

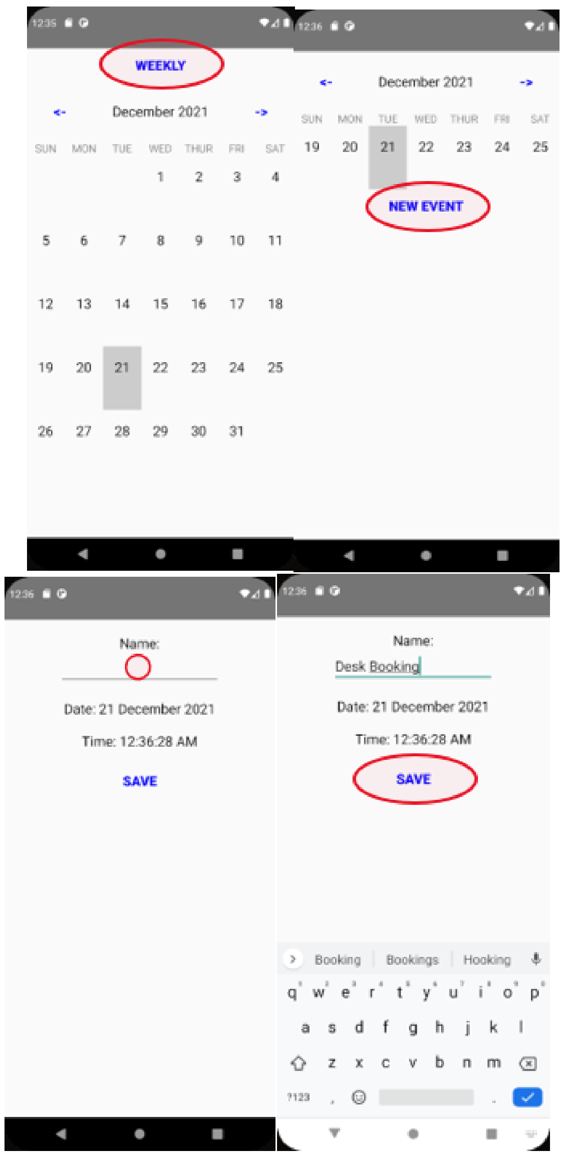
Another decision I discussed with my supervisor was the idea of how long the desk bookings should be. The suggestion was to have there be half day bookings which I agreed with. In this final implementation, there are 2 options when it comes to how long the desk booking will be. The User can select a custom time range, or they can select an All Day booking (DevExtreme, 2022). I decided on this in the end as the definition of a half day was tough to decide on, so I found it better to let the user decide whether they want to have the desk for the whole business day, or to have it for their own personal preferred amount of time.

The adaptability of the project is one of the strong points, in my opinion. I learned how to make the web app scaleable and mobile friendly using this section (DevExtreme, 2022) of the documentation. This confirmed to me that moving to this platform was the right choice, as now the website gives a better impression on mobile devices than the Android app had.

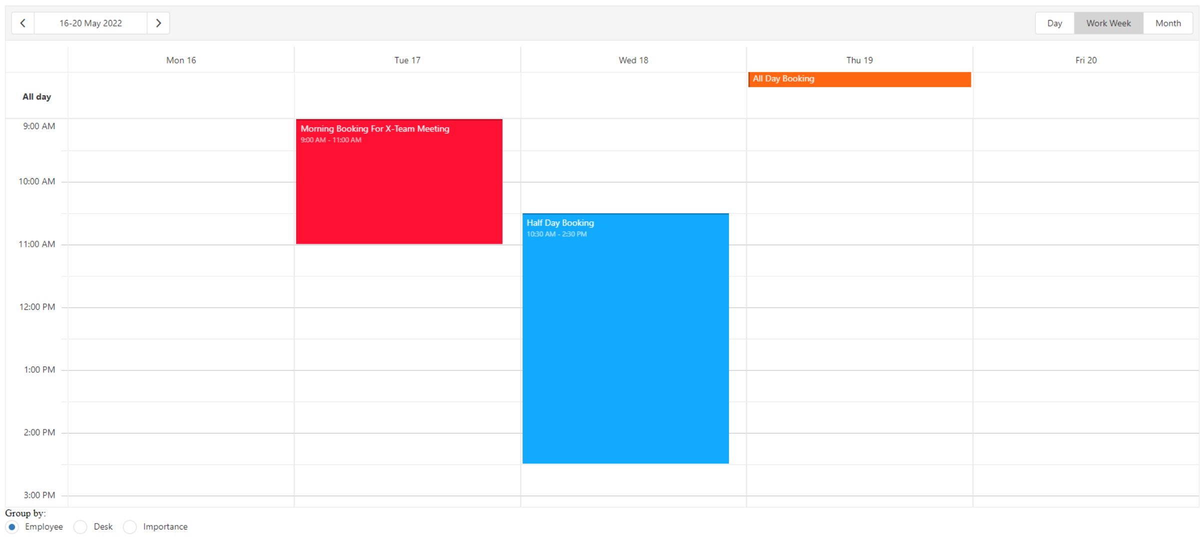
2.3. Graphical User Interface (GUI)

I'm including the GUI examples from the midpoint submission to display the, in my opinion, improved User Interface that exists at the final submission.

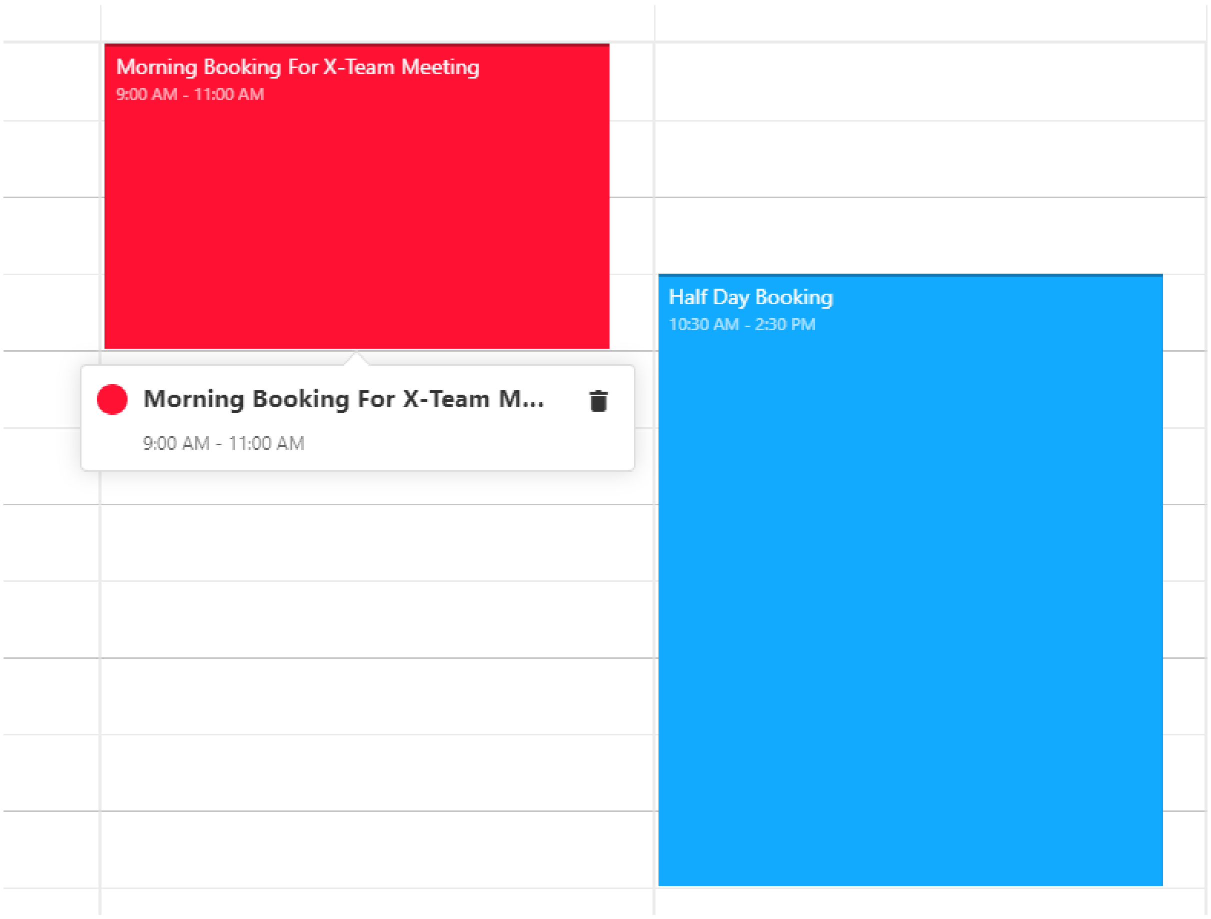
Previous GUI:



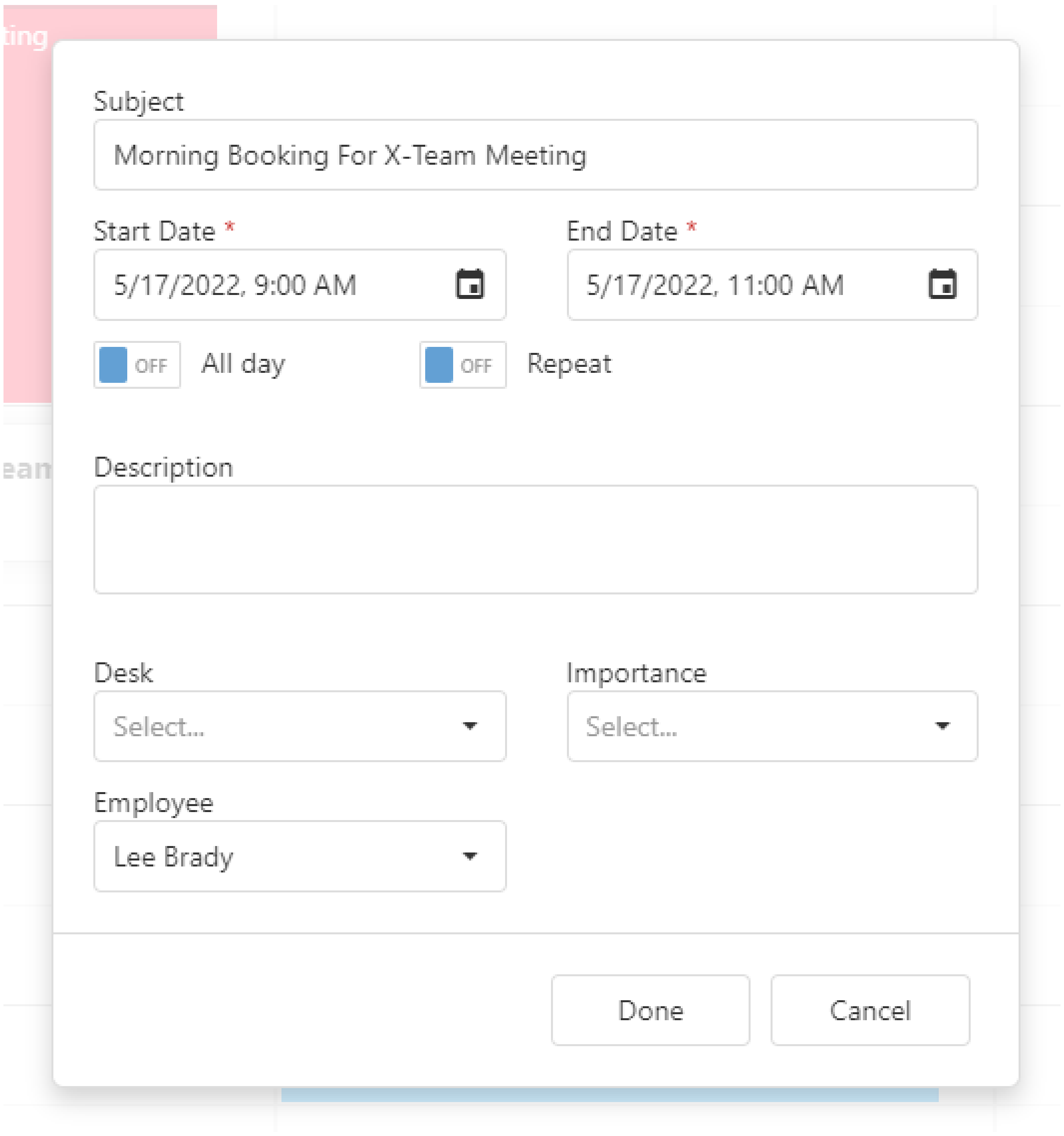
Final GUI:



2 Initial calendar view



3 Selecting an existing booking and seeing more info



Subject

Morning Booking For X-Team Meeting

Start Date *

5/17/2022, 9:00 AM

End Date *

5/17/2022, 11:00 AM

OFF All day

OFF Repeat

Description

Desk

Select...

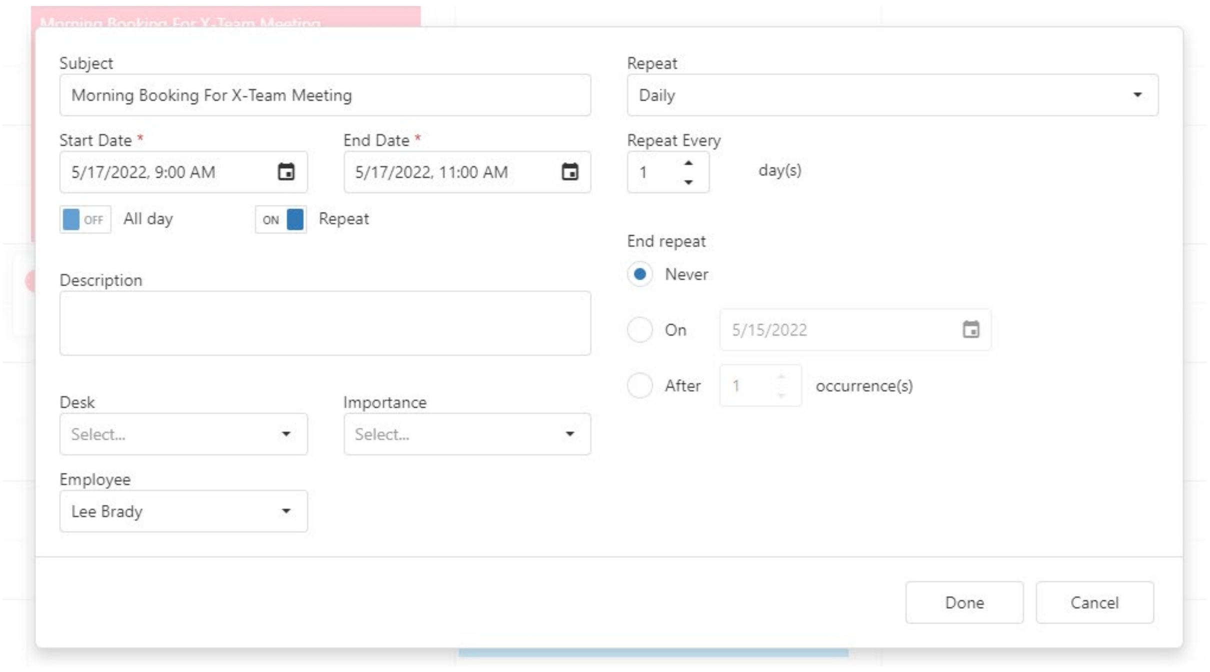
Importance

Select...

Employee

Lee Brady

4 *Creating a desk booking and setting different options*


Subject: Morning Booking For X-Team Meeting

Start Date *: 5/17/2022, 9:00 AM

End Date *: 5/17/2022, 11:00 AM

OFF All day

ON Repeat

Description:

Desk: Select...

Importance: Select...

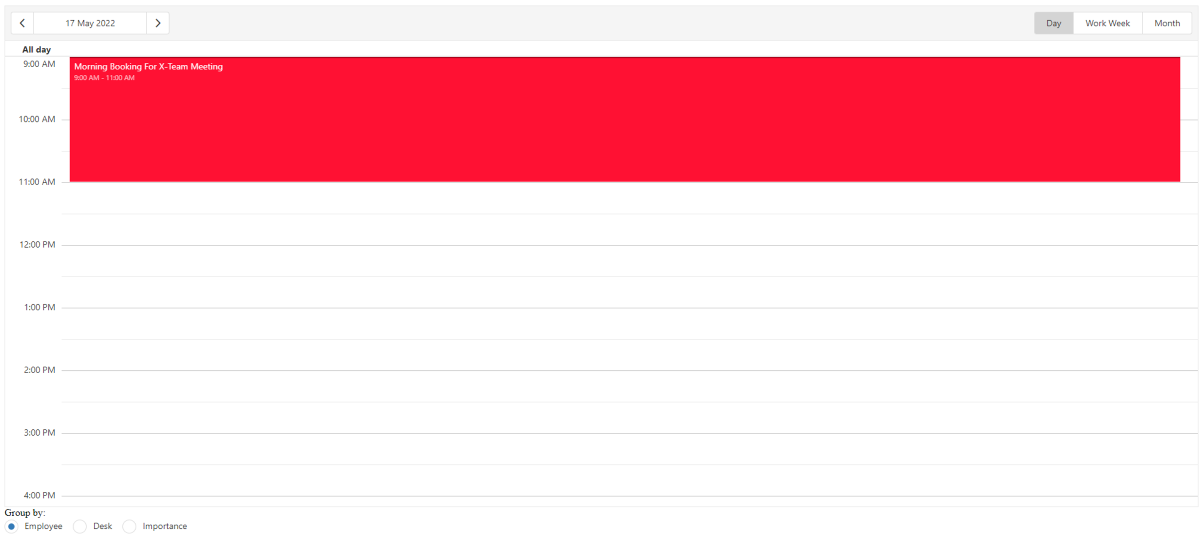
Employee: Lee Brady

Repeat: Daily

Repeat Every: 1 day(s)

End repeat: [✓ Never] On 5/15/2022 After 1 occurrence(s)

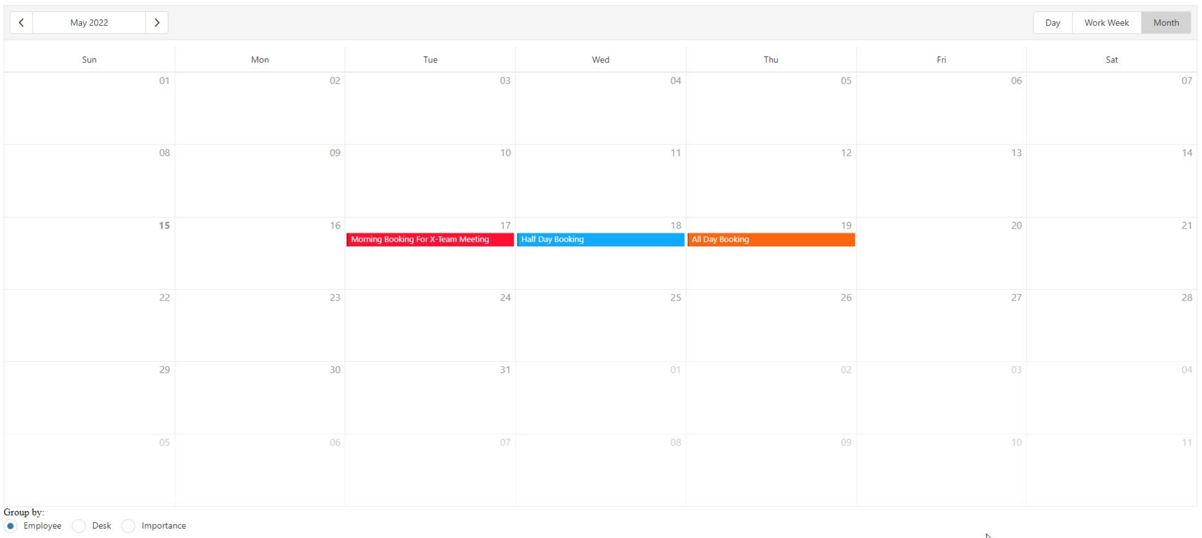
5 Setting up the Auto Scheduler



Morning Booking For X-Team Meeting
9:00 AM - 11:00 AM

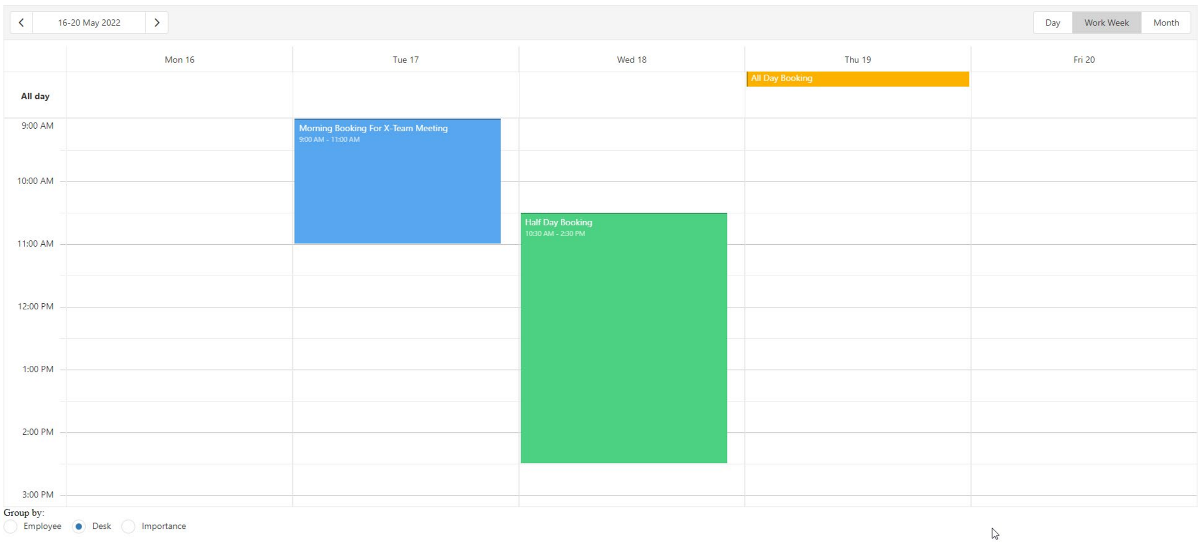
Group by: [✓ Employee] Desk Importance

6 'Day' view of calendar



Group by:
[✓ Employee] Desk Importance

7 'Month' view of calendar



Group by:
Employee [✓ Desk] Importance

8 Grouping desk bookings by 'Desk' option

2.4. Testing

The testing of this final submission's code was a lot different from when the application was written in Java. After that mid-point phase I wrote unit tests using this guide (Jenkov, 2014).

When the technology was changed to a web app using an outside framework and library (DevExtreme) I decided that due to the nature of the code I had written, simeple debugging took care of the need for unit testing. I instead turned to Mocking, which was a concept I had become familiar with in my 3rd year internship when I was writing in GoLang.

Mocking is defined as "*creating a fake version of an external or internal service that can stand in for the real one, helping your tests run more quickly and more reliably. When your implementation interacts with an object's properties, rather than its function or behavior, a mock can be used.*" (Jung, 2019)

I used Jest, which is a JavaScript testing framework, as an extension in VS Code. This allowed me to mock the jQuery dependent aspects of the project, without using real jQuery, or the DevExtreme service. The purpose of this, in theory, is that you can test the function to see if it's returning the right response, without creating any load on the service. If the library I was calling cost money based on it's use, this would save us money in testing.

The main example of mocking I carried out using Jest was with the AJAX jQuery call being made in the index.html file at

```
<scriptsrc="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
</script>
```

Example of mocking:

```
 $.ajax = jest.fn().mockImplementation(() => {
        const fakeResponse = {
            id: 1,
            name: "All",
            value: "Dummy Data"
        };
        return Promise.resolve(fakeResponse);
    });
```

Jest made the mocking process easy. With the jQuery being used in the index.js being dependent on this AJAX script import, testing whether something *was* being called in the script was important.

3.0. Conclusions

As mentioned earlier, one of the drawbacks from the development of the project is that it's limited to its own system. By this I mean it doesn't communicate with an outside calendar API. I think this is a drawback as if a large company were to implement a system like this for their offices, it would almost definitely be integrated in with their organisation's stack.

In the end it was more important to create the Auto Scheduler than have it connect to Outlook or Google Calendar, as that functionality would take out the need for users to create multiple similar desk bookings, making the app much more satisfying to use. As my supervisor said in an earlier interaction, the app should be doing all of the work for you.

In a way, the app being completely self-contained is also a plus. I think this as the project that's live now serves as a good proof of concept. This is because when changes are made to the calendar live on Heroku right now, they aren't persistent across sessions. This means that if I want to show this project off to someone, be it the person evaluating it for my final grade, a potential employer, or a friend, they won't be seeing changes that other users may have made which may confuse them. When the web app is launched, it is a clean slate, with just a little bit of data populated to demonstrate it's features. In this way, it's a sandbox application, with no danger of the demo not working for the next tester.

I'm much happier with the final delivery of this project than I was with the mid-point. I think the technology is much more interesting and flexible than the earlier concept.

4.0. Further Development or Research

With additional time and resources, I would do a few things. These are mainly if the project was being adapted to a business, which would scale the app's scope up greatly:

- Create a login system the web app to verify the user is an employee of the business
- Send the created desk bookings to the user's external calendar
- Have a visualisation of where the desks in the office are
- Allow the user to make decisions on when to book time in the office based on weather forecasts, commute time estimates based on where the employee is vs the office etc.

With these changes I think this could be a very plausible project for a business that fits the use case to take into their ecosystem.

6.0. Appendices

6.1. Project Proposal

Objectives

I want to create a project that facilitates smart hot desking. I want to do this because there are examples of companies right now that have expanded their workforce during the pandemic, but don't have enough physical desks in their current offices for when employees return from the full work from home period.

I want to make a HotDesking application that allows the employee to book time at a desk in the office, but I want to incorporate some intuitive features that lets the application do the work and require little User input.

I want the app to connect to the User's outlook calendar and suggest desk bookings around meetings for example. Other potential features could be a google places activity in the app that suggests what days/times are best to book a desk based on traffic, or maybe an algorithm that recommends when to book a desk based on when members of your team are usually booking desks too.

The project is going to be a proof-of-concept app that shows the potential of an intelligent scheduling system. The theoretical User would be an employee in a company that employs a hybrid work model.

Background

I chose to take on this project as companies in Dublin at the moment face a huge challenge when it comes to office space. While I was on placement last semester, the company I interned for hired more employees over the pandemic than the number of desks available in the office. The company is now returning to office, and has adapted a hybrid approach. Employees mark their preference on whether they'd like to be never, partly, or always in the office. This accommodates workers as well as helps the strain on the office.

HotDesking is going to be an important aspect of the office environment going forward. It's relevant as it's extremely expensive to rent new property in Dublin, or to expand a current office.

I have a clear goal for the structure of the app. With a strong and detailed project plan as well as guidance from my supervisor I think I can deliver a promising application.

State of the Art

There are other HotDesking applications, such as HotDesk. Apps like HotDesk come at this issue from the approach that the User is in need of a rentable space to work from. This would partially suit the business need I have outlined, but not entirely. The interface of HotDesk also mimics AirBnB, with a map view helping the User find a space to work in. In our case, we'll be focusing on the available space in one office, not trying to find an entire new space.

In the main use case for my app, the User is an employee of a company that *has* a physical space the employee can make use of. In our case, if the User (employee) isn't going to be in the office, they'll be working from home.

I find it more interesting to try and tackle the issue of *when should an employee be at a desk?* There is a lot of discourse at the moment about working from home, and a push from workers for employers to adopt a hybrid approach if an entirely from home solution isn't possible. I think there could be a great potential in this sector to try and make a conscious decision on when an employee really needs to be in the office.

I want this app to stand out as something that can make it clear when the best time to be in the office is, and if possible, make the most of an employee's time when they are at a desk.

Technical Approach

There are many ways I think this could be approached. I think the main outline of the project should be to tackle the basic function, which is a scheduling application. The core functionality should be there before applying any kind of "smart" features. I want to have a working example of selecting a day of the week, and booking a time slot.

Another aspect that could be included in the project is a token system. To show that the office has limited space, maybe once the employee has selected the average amount of days they'll be working weekly, they'll be given a certain amount of "in day" tokens. These can be spent when booking time in the office, and are limited as each employee can only occupy a space one at a time. Just an idea, came around as I was typing another paragraph.

After this simple scheduling system is in place and the core functionality of using CRUD to make, change or delete bookings is there, then I want to play around with the intelligent aspect. There are a couple of aspects to this, one being integrated with an external calendar, and the other being a custom, algorithm based approach. Developing the base of the app will hopefully bring clarity to which path to take.

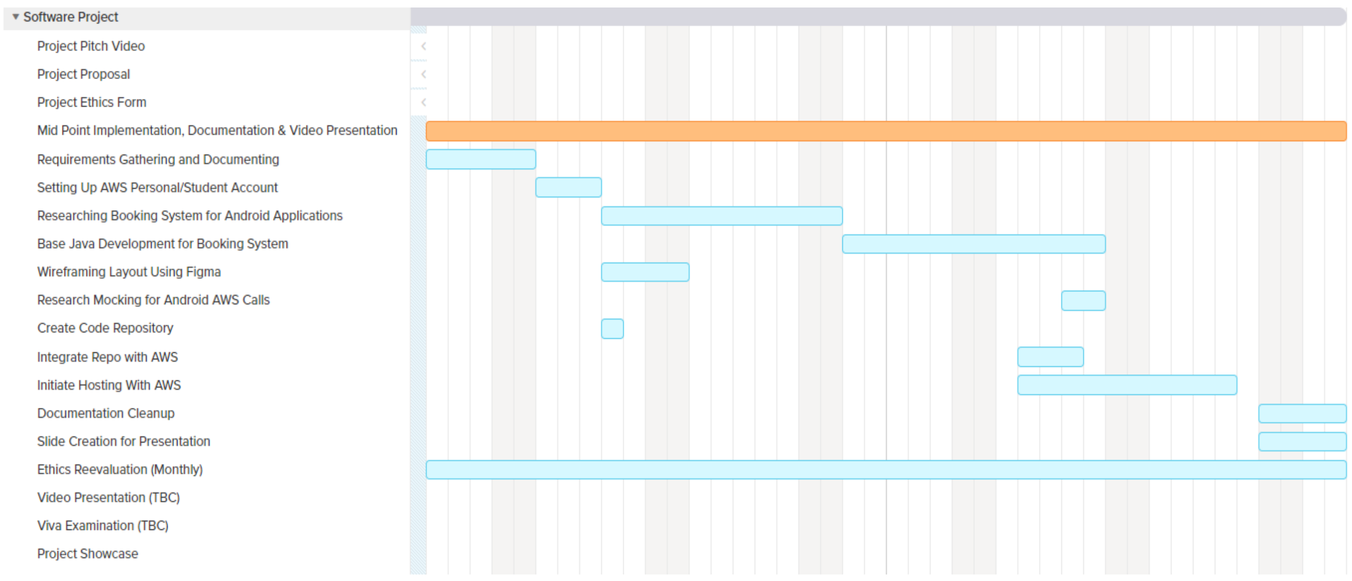
The project plan will contain information on the approach I'm taking. As discussed above, I think it will take quite a linear approach at first, so the initial (pre midpoint) development can be straightforward without lots of stops for research. I feel that the requirement identifying aspect will come after the base has been laid.

Technical Details

I intend to use:

- Java, Groovy
- Android Studio
- Algorithms like Optimized Calculation Algorithm for Business Days
- AWS Technologies (EC2 Instances, Lambda Functions, S3 Buckets, ECS Clusters, AWS Config)
- MongoDB
- Terraform (infrastructure as code, also relating to AWS)

Project Plan



This screenshot denotes my initial rough plan of what work should be completed before the midpoint presentation. I have allotted time for research, mainly on how to create the core application that should be solid and functioning before integration with AWS technologies.

I see the “Researching Booking System for Android Applications” section as being following a tutorial for a similar booking system, to see the difficulty level of an app like this, what parts can be hosted in AWS and to get familiar with the structure of this kind of app.

I also have segmented time to set up accounts for GitHub and AWS, as well as an integration period for the 2. In my internship I gained a lot of experience with pushing code to GitHub and reflecting changes in AWS using S3 Buckets. This experience will help me along with the integration but there is a catch; when I was interning I had access to Enterprise accounts for both services, this is why I’ve allocated myself time to look into the process, as I’m unfamiliar with it when it comes to a consumer account.

This Gantt chart leaves me room to move things around, as other CA’s may take priority on certain days. For every task, I estimated more time than I think I’ll need to complete it, so if I run into issues, I can hopefully use the extra time to debug. Some fields are intentionally vague, like “Base Java Development for Booking System”. This is because the previous task is to research the booking system, and that task will inform how much effort and what features should go into the development task.

Testing

Aside from standard unit testing, I plan to use the Mocking strategy of testing to validate my code. Mocking in this case refers to setting up a call to an AWS service and then ‘mocking’ that call to save an actual call. For example when using AWS Config, a call could be PutMetricData, which assigns a piece of information as a Metric, or a statistic. The reason mocking is important is that it saves resources and money.

Assigning a dummy result from an interface shows us whether or not we’re getting a valid response from the AWS service, without the real call to the service that will cost us more money.

While I won’t be using Golang, here is a good example of the kind of Mocking I intend to implement while testing my solution. I implemented this strategy in my internship as it was standard in my team and as a DevOps practice.

6.1. Reflective Journals

Supervision & Reflection Template

Student Name	Lee Brady
Student Number	X18434514
Course	BSHCSD4

Month: October 2021

What?

Reflect on what has happened in your project this month?

This month I started the idea generation process for my project. I decided on the idea of a smart hotdesking app, and created a project pitch video for it. I have been assigned Frances Sheridan as my project supervisor.

So What?

Consider what that meant for your project progress. What were your successes? What challenges still remain?

It was a big starting step to come up with a project idea that will carry me through to graduation. I wanted to pick something complex but manageable. I think I picked a strong idea for a project, but seeing what aspects I should include will be challenging.

Now What?

What can you do to address outstanding challenges?

Now, I am working on my formal project proposal. I intend to create a very specific project timeline to follow, as the structure will help me stay on track. I will also complete the Project Ethics form.

Student Signature	Lee Brady

Supervision & Reflection Template

Student Name	Lee Brady
Student Number	X18434514
Course	BSHCSD4

Month: November 2021

What?

Reflect on what has happened in your project this month?

I continued to meet with my project supervisor and got advice on various aspects of the project. I performed elicitation techniques, and created UML diagrams to illustrate use cases for my project.

So What?

Consider what that meant for your project progress. What were your successes? What challenges still remain?

I am happy with progress so far. I need to continue to work on development, and figure out the main methods for my classes in android studio.

Now What?

What can you do to address outstanding challenges?

I need to work on my technical report in conjunction with developing the project, as this reflection will help me with future work.

Student Signature	Lee Brady

Supervision & Reflection Template

Student Name	Lee Brady
Student Number	X18434514
Course	BSHCSD4

Month: December 2021

What?

Reflect on what has happened in your project this month?

I continued to meet with my project supervisor and got advice on various aspects of the project.

I worked on my midpoint technical report, and made sure my documentation was all in order for the submission (video presentation, monthly reports)

So What?

Consider what that meant for your project progress. What were your successes? What challenges still remain?

I developed a Java Android prototype to use for my midpoint submission, which was critical to show progress so far.

Now What?

What can you do to address outstanding challenges?

I need to get feedback on the midpoint submission and use that to continue improving my project.

Student Signature	Lee Brady

Supervision & Reflection Template

Student Name	Lee Brady
Student Number	X18434514
Course	BSHCSD4

Month: January 2022

What? *Reflect on what has happened in your project this month?* I applied for an extension last month for the midpoint presentation and submission, due to personal circumstances. I created a Gantt chart for my semester 2 project plan, and completed the midpoint presentation code needed for my project demo. I performed elicitation techniques, and created UML diagrams to illustrate use cases for my project.
So What? *Consider what that meant for your project progress. What were your successes? What challenges still remain?* I am happy with progress so far. I need to take the current progress with my project code and incorporate more technologies, to satisfy the grading rubric.
Now What? *What can you do to address outstanding challenges?* I need to follow my semester 2 project plan to try and stay on track for a good project. I will also start meeting with my supervisor again starting next Monday.

Student Signature	Lee Brady

Supervision & Reflection Template

Student Name	Lee Brady
Student Number	X18434514
Course	BSHCSD4

Month: February 2022

What? *Reflect on what has happened in your project this month?* I did not complete a lot of project work this month due to other assignments. I continued to meet with my project supervisor to keep them up to date on the project.
So What? *Consider what that meant for your project progress. What were your successes? What challenges still remain?*
Now What? *What can you do to address outstanding challenges?* I'm currently unsure on the next step with the project, as I haven't been spending as much time thinking about it as I had last semester.

Student Signature	Lee Brady

Supervision & Reflection Template

Student Name	Lee Brady
Student Number	X18434514
Course	BSHCSD4

Month: March 2022

What? *Reflect on what has happened in your project this month?* I met with my supervisor and discussed changing the technology of the project to a Python/Django web based approach this was approved.
So What? *Consider what that meant for your project progress. What were your successes? What challenges still remain?* I am very happy that this has been given the go ahead, this will hopefully make the project more interesting and achieve a higher grade.
Now What? *What can you do to address outstanding challenges?* I can now re-spec the project plan to allow some time to research the new technology I will be using.

Student Signature	Lee Brady

Supervision & Reflection Template

Student Name	Lee Brady
Student Number	X18434514
Course	BSHCSD4

Month: April 2022

What? *Reflect on what has happened in your project this month?* I have created a Python/Django based Microsoft Outlook demo, but it has limitations that have made me change the technology to draw from a library called DevExtreme.
So What? *Consider what that meant for your project progress. What were your successes? What challenges still remain?* The final project goal now seems a lot more achievable.
Now What? *What can you do to address outstanding challenges?* I will now work on fleshing out the implementation I have working with this new tech stack and work towards the final due date in May.

Student Signature	Lee Brady

References

Bibliography

Alara, 2016. *Hot Desking: The New Work Style.* [Online]
Available at: https://www.thefarmsoho.com/magazine-blog/hot-desking/
[Accessed 20 December 2021].

DevExtreme, 2022. *DevExtreme Responsive Web Development.* [Online]
Available at: https://js.devexpress.com/Documentation/
[Accessed 14 May 2022].

DevExtreme, 2022. *JavaScript Component Suite.* [Online]
Available at: https://js.devexpress.com/
[Accessed 14 May 2022].

Hill, C., 2021. *CalendarTutorialAndroidStudio.* [Online]
Available at: https://github.com/codeWithCal/CalendarTutorialAndroidStudio
[Accessed 10 December 2021].

Jenkov, J., 2014. *A Simple Unit Test.* [Online]
Available at: https://jenkov.com/tutorials/java-unit-testing/simple-test.html
[Accessed February 2022].

Jung, J., 2019. *How to test software, part I: mocking, stubbing, and contract testing.* [Online]
Available at: https://circleci.com/blog/how-to-test-software-part-i-mocking-stubbing-and-contract-testing/#:~:text=Mocking%20means%20creating%20a%20fake,a%20mock%20can%20be%20used.
[Accessed 15 May 2022].

Microsoft, 2022. *Build Python Django apps with Microsoft Graph.* [Online]
Available at: https://docs.microsoft.com/en-us/graph/tutorials/python
[Accessed 15 May 2022].

Murray, M., 2021. *All likely to change in office of the future.* [Online]
Available at: https://www.irishtimes.com/special-reports/new-year-new-career/all-likely-to-change-in-office-of-the-future-1.4751218
[Accessed 20 December 2021].

Weston, C., 2021. *Property prices now surging at a rate of 13.5pc.* [Online]
Available at: https://www.independent.ie/business/personal-finance/property-mortgages/property-prices-now-surging-at-a-rate-of-135pc-41152368.html
[Accessed 20 December 2021].