A decorative graphic on the left side of the slide consisting of two overlapping parallelograms. The front one is blue and the back one is a light green. They are positioned diagonally, with the blue one partially covering the green one.

# My Car Park Booking System

By Ryan Glennerster



# Introduction

My name is Ryan Glennerster and i'm a QA Academy Trainee. I'm here to present my Car Park Booking System.

About myself.

Learning Software Development

A small amount of self taught programming knowledge.



# Approaching the project

Learning from my first project I came into this project with a better knowledge of what to expect from myself and how to tackle the project efficiently.

I started off drawing the front end, planning how it would work and how it would look.

I completed the back-end before I fully completed the front end as I tried to prioritise what I thought was more important which was the actual functionality of the site, an amazing front end would have been no good without a back end that allows you to use the functions.

As a guide I tried to dedicate a day to each part (one day for each of the CRUD functionalities) and a day for the front end.



# Sprints

As I was working on this on my own, I allowed myself some flexibility with the sprints. I didn't go to specific with the user stories, I made then for each CRUD Functionality but knowing how I work I know I would jump between different aspects so I gave myself a goal of completing each after one day and that gave me something to push for. No matter what I got sidetracked doing I made sure my goal was hit.

The sprints went well as a whole an I managed to get everything done on time and nothing was left behind.



# Consultant Journey

Coming into the project I had a knowledge of Java from my previous project but I had a lot to learn in regards to front-end and spring boot.

I had some previous experience with HTML but I have never used it in a practical project so it was good to expand my knowledge and then apply what I was learning to something I can actually use. Making something that you can use gives you a sense of achievement just learning the theory doesn't give you.







I feel like I adapted to the new technologies pretty quickly. With the help of the trainers whenever I needed it, I really feel like I learnt a lot and I am happy with how everything came out.

# Testing

I came into this testing with a knowledge of JUnit testing from my previous project but with a fresh understanding of selenium.

Testing was something I really struggled with in the first project. Knowing this I gave myself more time to complete this and asked for help whenever I needed it, I found this helped me not only complete the testing but actually understand what I was doing.

I managed to complete the >80% coverage on this project.

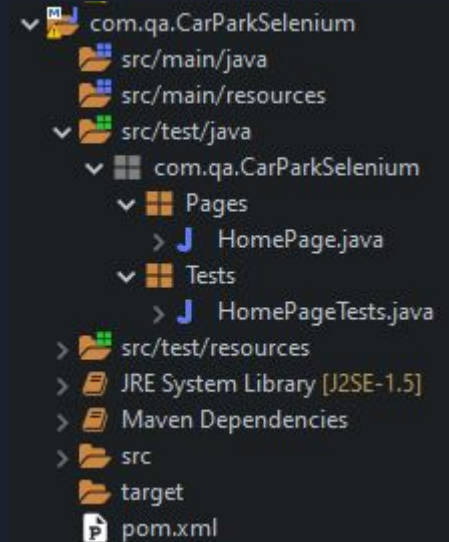
Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
✓  CarPark	 93.0 %	502	38	540
>  src/main/java	 81.6 %	168	38	206
>  src/test/java	 100.0 %	334	0	334

# Selenium Testing

I enjoyed the selenium testing as it was something I have never done before and I really felt like I caught onto it quickly.

I encountered a small problem with selenium testing, my project wouldn't actually run the tests so to overcome this I made a new project specifically for the selenium testing.

As you can see from the screen shot, the selenium testing was well organised and I utilised different folders for the different sections. Pages holds the web elements which i am interacting with and the Tests package holds the actual tests.



# Creating a booking

```
<div class="row container">
  <section class="col3">
    <div class="align-items-center" id="createBookingForm" style="display: none;">
      <label for="carMake">Your Car Make</label>
      <input name="make" id="carMake" class="form-control"/>
      <label for="carModel" class="form-label">Your Car Model</label>
      <input type="text" name="carModel" id="carModel" class="form-control"/>
      <label for="arrivalDate" class="form-label">Your Arrival Date</label>
      <input type="date" name="arrivalDate" id="arrivalDate" class="form-control"/>
      <label for="LeavingDate" class="form-label">Your Departure Date</label>
      <input type="date" name="LeavingDate" id="departureDate" class="form-control"/>
      <button type="reset" class="btn btn-primary">Reset Fields</button>
      <button type="submit" class="btn btn-primary" value="submit" onclick="createUser();">Submit</button>
    </div>
  </section>
</div>
```

```
@PostMapping("/create")
public ResponseEntity<CarPark> createBooking(@RequestBody CarPark car) {
    return new ResponseEntity<CarPark>(this.service.create(car), HttpStatus.CREATED);
}
```

```
public CarPark create(CarPark car) {
    return this.repo.save(car);
}
```

```
const createUser = () => {

    const carMakeValue = carMake.value;
    const carModelValue = carModel.value;
    const arrivalDateValue = arrivalDate.value.toString();
    console.log(typeof(arrivalDateValue));
    const leavingDateValue = leavingDate.value.toString();

    let obj = {
        make: carMakeValue,
        model: carModelValue,
        arrivalDate: arrivalDateValue,
        leavingDate: leavingDateValue
    };

    console.log(obj);

    axios.post("http://localhost:8080/create", obj, {
        "headers":{
            "Access-Control-Allow-Origin": "*"
        }
    }).then((resp) => {
        console.log(resp);
        document.getElementById('createBookingForm').style.display = 'none';
        output.innerHTML = "Booking Successfully created!";
        setTimeout(() => {
            output.innerHTML = "";
            location.reload();
        }, 3000);
    }).catch((err) => console.error(err));
}
```



# Reading the bookings

```
const printToScreen = (information) => {  
  
  const newColumn = document.createElement("div");  
  newColumn.className = "col";  
  
  viewAllBookings.appendChild(newColumn);  
  
  const newBooking = document.createElement("div");  
  newBooking.className = "card";  
  newColumn.appendChild(newBooking);  
  
  const bookingBody = document.createElement("div");  
  bookingBody.className = "card-body";  
  newBooking.appendChild(bookingBody);  
  
  const bookingMake = document.createElement("h5");  
  bookingMake.className = "card-title";  
  bookingMake.innerText = information.make;  
  bookingBody.appendChild(bookingMake);  
  
  const bookingText = document.createElement("p");  
  bookingText.className = "card-text";  
  bookingText.innerHTML = "Model: " + information.model;  
  bookingText.innerHTML += "<br>";  
  bookingText.innerHTML += "Arrival Date: " + information.arrivalDate;  
  bookingText.innerHTML += "<br>";  
  bookingText.innerHTML += "Departure Date: " + information.leavingDate;  
  bookingBody.appendChild(bookingText);  
  
  const bookingFooter = document.createElement("div");  
  bookingFooter.className = "card-footer";  
  newBooking.appendChild(bookingFooter);  
  
  const deleteBookingButton = document.createElement("button");  
  deleteBookingButton.className = "btn btn-primary";  
  deleteBookingButton.innerText = "Delete";  
  deleteBookingButton.addEventListener("click", () => deleteBooking(information.id));  
  bookingFooter.appendChild(deleteBookingButton);  
  
  const editBookingButton = document.createElement("button");  
  editBookingButton.className = "btn btn-primary";  
  editBookingButton.innerText = "Edit Booking";  
  editBookingButton.addEventListener("click", () => showUpdateForm(information.id));  
  
  bookingFooter.appendChild(editBookingButton);  
  
  return newColumn;  
}
```

```
const getBookings = () => {  
  axios.get("http://localhost:8080/getAll")  
    .then((response) => {  
      console.log(response.data);  
  
      for (let data of response.data){  
  
        printToScreen(data)  
  
        for (let i in data) {  
          console.log(data[i]);  
        }  
      }  
    }).catch((err) => {  
      console.error(err);  
    });  
}
```

```
public List<CarPark> getAll() {  
  return this.repo.findAll();  
}
```

```
@GetMapping("/getAll")  
public ResponseEntity<List<CarPark>> getCars() {  
  return ResponseEntity.ok(this.service.getAll());  
}
```



# Reflection

As a whole I feel like the project went well and I feel positive going into the next one.

I feel like I have made noticeable improvement from my first project whether that be the way I work, the way I learn and the way I plan out my time, I feel like I have made improvements in all aspects.

I think focusing on the planning, improvement of sprints, user stories will be something I look to do in my next project especially when I am working alongside other people I feel like that side of it becomes a lot more important.

I think a key takeaway from my time doing this project is really taking the time to understand what i'm doing, not just mindlessly using things from google or youtube, researching but also taking the time to understand what it means and how it's getting me to my end goal.

Overall I feel like I planned my time and used my time a lot better this project and I noticed it had a positive impact on my work but I think in the future I can still work on this and really take my work to the next level.



# Thank you for listening!

I'll give a quick demo of my project in action!

