Dublin Business School
August 2024

Assignment Cover Sheet

Please fill out and insert at the top of your README or Google Doc as preferred.

Student Name and Number as per student card: Gourav Sankar Ray (20070234)
Programme: MSc. Information Systems with Computing
Lecturer Name: Paul Laird
Module/Subject Title: Programming for Information Systems
Assignment Title: CA2 2025 - B9IS123 Programming for Information Systems

By submitting this assignment, I am confirming that:
● This assignment is all my own work;
● Any sources used have been referenced;
● I have followed the Generative AI instructions/ scale set out in the Assignment Brief;
● I have read the College rules regarding academic integrity in the QAH Part B Section 3, and the Generative AI Guidelines, and understand that penalties will be applied accordingly if work is found not to be my/our own.
● I understand that all work submitted may be code-matched report to show any similarities with other work.

Note: Technical support is available to students between 0830- 2000 hrs (Mon-Fri), 0930-1630 (Sat) only. There is no technical support after 2000 hrs. It is your responsibility to ensure that you allow time to troubleshoot any technical difficulties by uploading early on the due date.

2024 Version 1.0

Desk & Member Utilisation Tracker – CRUD Web App for CoCreate North

Programming for Information Systems (B9IS123)
Gourav Sankar Ray | 20070234 | Dublin Business School
Lecturer: Paul Laird

Github Repository for CA2 submission:
https://docs.google.com/document/d/1c1o9KqsrZfVuDRD6Pch3GU-iB2A9X6-vvMPCYIhlH_E/edit?tab=t.0

Company Overview - CoCreate North

Coworking space offering hot desk, dedicated desk, private office and corporate workspace services.

Location: O'Connell Street Upper, Dublin 1

The objective is to replace manual booking and tracking systems with an information system that will enable more effective use of the co-working space.

Project Objective:

Develop a CRUD web-app for a Hot-desking management and utilisation tracker system, for a co-working space using Javascript frameworks for FE, BE and SQLite for database. Primarily, there are two stakeholders in this system -

a) Admin: Manager of the co-working space who manages the resources, like creation/updation/deletion of desks, meetings rooms, zones, member/user profiles, and also has admin access to the system to view utilization reports, metrics, trigger system reset etc.

b) Member/User: The customer of the co-working space, who maintains an active membership with the company, for booking of desk and private office spaces. They have the rights to manage their bookings. Also, they have the option to check-in for a booking to avoid a no-show status update in the system.

Core Modules:

Desk management
User/Member management
Zone Management
Booking management
Booking status page / Check-in Screen
Reports & Analytics Screen
Settings/Configuration Screen

Features:

Create - create desk & meeting room assets, member profiles in the system, consisting of data like membership type, joining date etc.

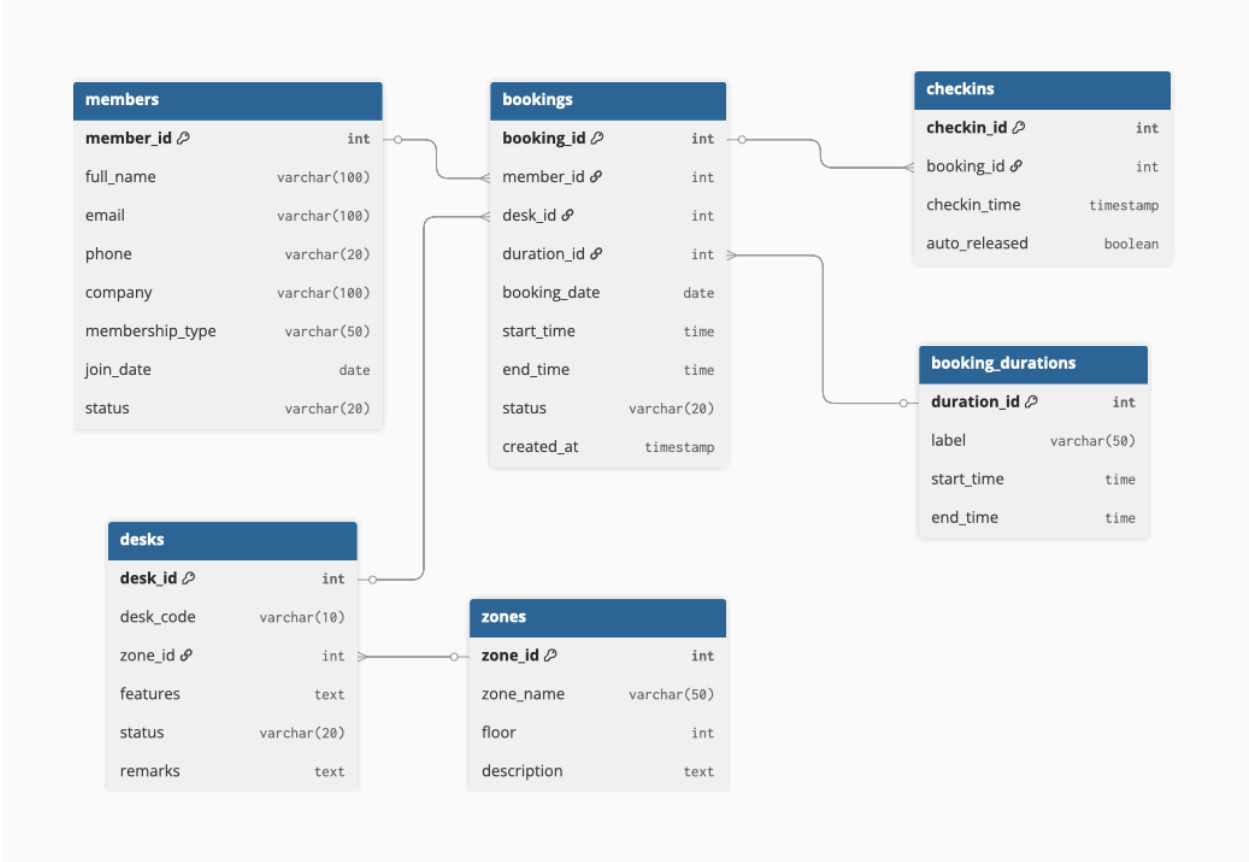
Read - read the updated booking and desk availability status from the remote database, and filter out un-available desks to show only the unoccupied desks in the create new booking flow.

Update - update the assets, bookings and members details in the system.

Delete - delete assets, bookings, members data from the remote database.

- Desk booking and management
- Sorting of available desks grouping them by desk type
- Searching for available seats, before creating a new booking
- Searching for free slots for a seat, in the upcoming week, before re-scheduling the booking
- Handle validations on the server-side like valid input for booking date, ensuring booking a desk on a past date is not possible
- Utilisation dashboard consisting of charts like pie chart/ bar chart for utilisation data visualisation
- Additional Metrics like occupancy %, member activity
- Manage master data and system setup
    - Add/Edit Booking Durations (first half, second half, full day)
    - Manage Zones (names, floors, descriptions)

ER Diagram


members
member_id int
full_name varchar(100)
email varchar(100)
phone varchar(20)
company varchar(100)
membership_type varchar(50)
join_date date
status varchar(20)
bookings
booking_id int
member_id int
desk_id int
duration_id int
booking_date date
start_time time
end_time time
status varchar(20)
created_at timestamp
checkins
checkin_id int
booking_id int
checkin_time timestamp
auto_released boolean
booking_durations
duration_id int
label varchar(50)
start_time time
end_time time
desks
desk_id int
desk_code varchar(10)
zone_id int
features text
status varchar(20)
remarks text
zones
zone_id int
zone_name varchar(50)
floor int
description text

Tools & Technologies:

Frontend: React
Backend: Node.js + Express
Database: SQLite
Version Control: GitHub
Documentation: Google Docs (Editor access)

Test Credentials:

Admin -
Username - admin
Password - admin

Member 1 -
Username - john@abc.com
Password - user

Member 2 -
Username - bob@abc.com
Password - user

Backend Architecture:

API endpoints:

A. Admin Portal - Zones management

Create Zone(s) -
POST http://localhost:3000/api/zones/create
Req body -


```
{
"zones": [
{
"zone_id": 1,
"zone_name": "Zone GA",
"floor": 0,
"description": "Near Entrance"
},
{
"zone_id": 2,
"zone_name": "Zone GB",
"floor": 0,
"description": "Near Exit"
},
{
"zone_id": 3,
"zone_name": "Zone 1A",
"floor": 1,
"description": "Near Entrance"
},
{
"zone_id": 4,
"zone_name": "Zone 1B",
"floor": 1,
"description": "Near Cafeteria"
}
]
}
```


Get all zones -
GET http://localhost:3000/api/zones/all
Update zone -
PUT http://localhost:3000/api/zones/update/4

Req body -

```
{
"description": "Near Conference Room"
}
```

Delete zone -

DELETE **http://localhost:3000/api/zones/delete/2**

B. Admin Portal - Desks Management

Get Desk By ID -

GET **http://localhost:3000/api/desks/view/1**

Delete Desk -

DELETE **http://localhost:3000/api/desks/delete/1**

Update Desk -

PUT **http://localhost:3000/api/desks/update/6**

Req body -

```
{
"status": "available"
}
```

Get All Desks -

GET **http://localhost:3000/api/desks/all**

Create Desk -

POST **http://localhost:3000/api/desks/create**

Req body -

```
{
"desk": {
"desk_code": "CUBICLE",
"zone_id": "1",
"features": "Cubicle Desk",
"status": "available"
}
}
```

C. Admin Portal - Members Management

Get Member By ID -

GET **http://localhost:3000/api/members/view/1**

Get Member By Email -
GET http://localhost:3000/api/members/viewByEmail/john@abc.com
Delete Member -
DELETE http://localhost:3000/api/members/delete/2
Update Member -
PUT http://localhost:3000/api/members/update/1
Req body -

```
{
  "status": "Active"
}
```

Get All Members -
GET http://localhost:3000/api/members/all
Create Member -
POST http://localhost:3000/api/members/create
Req body -

```
{
"member": {
"full_name": "John Doe",
"email": "john.doe@ccn.com",
"phone": "8923748",
"company": "CoCreate North",
"membership_type": "admin",
"join_date": "2025-01-01",
"status": "Active"
}
}
```

D. Members Portal - Bookings Management

Create Booking -
POST http://localhost:3000/api/bookings/create
Req body -

```
{
"booking": {
"member_id": 1,
"desk_id": 1,
"duration_id": 1,
"booking_date": "2025-11-23",
"status": "pending"
}
}
```

Get all bookings -
GET http://localhost:3000/api/bookings/all
Get Booking by ID -
GET http://localhost:3000/api/bookings/view/1
Update Booking -
PUT http://localhost:3000/api/bookings/update/1
Req body -

```
{
"start_time": "08:00"
}
```

Delete booking -
DELETE http://localhost:3000/api/bookings/delete/1
Get Free Slots Next Week (for re-scheduling/UPDATE booking) -
GET http://localhost:3000/api/desks/getFreeSlotsNextWeek/1
GET Available Seats For Selected Date (View seat availability prior to create booking) -
GET http://localhost:3000/api/desks/getAvailableSeatsForDate/2025-12-05

E. Member Portal - Check-in management

Create checkin -
POST http://localhost:3000/api/checkins/create
Req body -

```
{
"checkin": {
"booking_id": 1,
"checkin_time": "09:30:00",
"auto_released": 0
}
}
```

F. Admin Portal - Dashboard Metrics / Reports / System

Desk utilisation data by selected date -
GET http://localhost:3000/api/dashboard/deskutilisation/2025-12-06

Response -


```
{
"booked_count": 1,
"available_count": 1,
"booked_percentage": 50,
"available_percentage": 50
}
```


Member utilisation data from last week-
GET http://localhost:3000/api/dashboard/memberutilisation

Response -


```
[
{
"member_id": 1,
"slots_booked_last_week": 2,
"total_slots_last_week": 10,
"util_percentage": 20,
"anomaly": false
},
{
"member_id": 2,
"slots_booked_last_week": 1,
"total_slots_last_week": 10,
"util_percentage": 10,
"anomaly": false
}
]
```


Admin System Reset - Release resources
DELETE http://localhost:3000/api/adminsystem/delete

References:

1. **OpenAI (2025)** ChatGPT-generated code snippet for re-usable Navbar UI component [online]. Available at: https://chatgpt.com/share/69397baf-95bc-8000-867f-d2f075ee00ac
2. **ChatGPT (2025)** Mock-data setup by mocking the DB using jest framework.
3. **Jest (2025)** Expect matchers documentation [online]. Available at: https://jestjs.io/docs/expect
4. **Jest (2025)** Mock functions documentation [online]. Available at: https://jestjs.io/docs/mock-functions
5. **Recharts contributors (n.d.)** ***Recharts*** **–** A composable charting library built on React components. Available at: https://www.npmjs.com/package/recharts
6. **NPM (2025)** Axios documentation: https://www.npmjs.com/package/axios
7. **ChatGPT (2025)** SQL statement to fetch member information in adminDashboardModel.js
8. **ChatGPT (2025)** CSS styling in admindeskmanagement.css
9. **ChatGPT (2025)** Download PDF utility function using html2canvas
10. **NPM (2025)** html2canvas for converting HTML to PDF format documentation: https://www.npmjs.com/package/html2canvas?activeTab=readme
11. **NPM (2025)** better-sqlite3 NPM package for SQLite db: https://www.npmjs.com/package/better-sqlite3
12. **NPM (2025)** react minimal pie chart documentation: https://www.npmjs.com/package/react-minimal-pie-chart
13. **MDN (no date)** 'Center an element', available at: https://developer.mozilla.org/en-US/docs/Web/CSS/Layout_cookbook/Center_an_element
14. **MDN (no date)** 'height' property, available at: https://developer.mozilla.org/en-US/docs/Web/CSS/Reference/Properties/height
15. **React Router (no date)** 'useNavigate()', available at: https://reactrouter.com/en/main/hooks/use-navigate
16. **React (no date)** 'Built-in React Hooks', available at: https://react.dev/reference/react/hooks

17. **React (no date)** 'Handling Forms', available at: https://react.dev/learn/sharing-state-between-components#handling-form-submission
18. **React (no date)** 'Writing Markup with JSX', available at: https://react.dev/learn/writing-markup-with-jsx
19. **React (no date)** 'JavaScript in JSX with Curly Braces', available at: https://react.dev/learn/javascript-in-jsx-with-curly-braces
20. **React (no date)** 'JSX In Depth', available at: https://legacy.reactjs.org/docs/jsx-in-depth.html
21. **React (no date)** 'Learn React', available at: https://react.dev/learn