

Software Architecture Design Document

Individual Track Project

Ismail Chbiki

Version 5.0

June 2022

Revision History

| Sprint | Date | Version | Description | Author |
|--------|------------|---------|--|---------------|
| 1 | 25/02/2022 | 1.0 | Initial version of SAD | Ismail Chbiki |
| 2 | 25/03/2022 | 2.0 | Basic frontend API calls | Ismail Chbiki |
| 3 | 14/04/2022 | 3.0 | Unit and Integration testing added | Ismail Chbiki |
| 4 | 13/05/2022 | 4.0 | Authentication & Authorization implemented | Ismail Chbiki |
| 5 | 03/06/2022 | 5.0 | Chatroom service implemented | Ismail Chbiki |

Table of Contents

| | |
|---|-----------|
| 1. Introduction..... | 1 |
| 1.1. Purpose | 1 |
| 1.2. Scope | 2 |
| 1.3. Definitions, Acronyms, and Abbreviations | 2 |
| 1.4. References..... | 3 |
| 1.5. Overview | 3 |
| 2. Architectural Representation | 3 |
| 3. Architectural Goals and Constraints | 4 |
| 3.1. Security | 5 |
| 3.2. Persistence | 5 |
| 3.3. Performance..... | 5 |
| 4. Use-Case View | 5 |
| 4.1. Actors..... | 6 |
| 4.2. Use-Case Realizations | 7 |
| 4.3. User Stories | 8 |
| 4.4. Acceptance Criteria | 8 |
| 5. Logical View | 10 |
| 5.1. Overview | 10 |
| 6. Process View..... | 10 |
| 7. Module Decomposition View..... | 11 |
| 8. Deployment View | 14 |
| 9. Issues and concerns | 14 |

Software Architecture Design Document

1. Introduction

This document provides a high-level overview and explains the whole architecture of Software System of the Web App. It explains how an online user will be able to use the software system to:

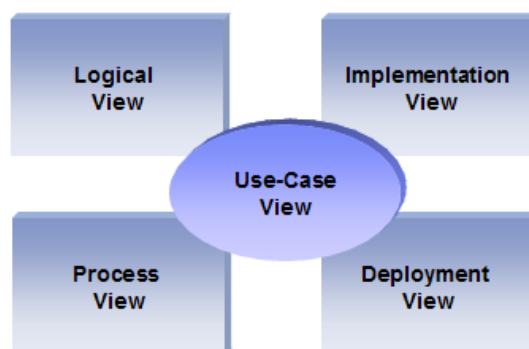
- Create a user account and login,
- Check the content of the software system (web app),
- Book a kite lesson,
- Use the chatroom service.
- Manage the Web App users & Admins

The document provides a high-level description of the goals of the architecture, the use cases support by the system and architectural styles and components that have been selected to best achieve the use cases. This framework then allows for the development of the design criteria and documents that define the technical and domain standards in detail.

1.1. Purpose

The Software Architecture Document (SAD) provides a comprehensive architectural overview of the Web App. It presents a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

In order to depict the software as accurately as possible, the structure of this document is based on the C4 and “4+1” models view of architecture [KRU41].



The C4 and “4+1” View Models allows various stakeholders to find what they need in the software architecture.

1.2. Scope

The scope of this SAD is to depict the architecture of the Web online application created by me, Ismail Chbiki – 2022.

This document describes the aspects of the Web App Software System design that are considered to be architecturally significant; that is, those elements and behaviors that are most fundamental for guiding the construction of the Web App and for understanding this project as a whole. Stakeholders who require a technical understanding of the Web App are encouraged to start by reading this document, then reviewing the Web App C4 model, and then by reviewing the source code.

1.3. Definitions, Acronyms, and Abbreviations

- **App** – Application
- **Java** – Programming language and computing platform
- **Spring Boot** – Open-source Java-based framework for Micro Services
- **SQL** – Structured Querying Language
- **RDBMS** – Relational database management system
- **ORM** – Object-Relational Mapping
- **MongoDB** – Source-available cross-platform document-oriented database program
- **SAD** - Software Architecture Document
- **UML** – Unified Modeling Language
- **RUP** – Rational Unified Process
- **HTTP** – Hypertext Transfer Protocol
- **WWW / WEB** – World Wide Web
- **JavaScript** – Scripting Programming Language for complex Web pages features
- **React** – JavaScript-based UI development library
- **OWASP** – Open Web Application Security Project
- **User** – This is any user who is a visitor on the web App
- **Admin (Web App Administrator)** - This is a user who can create / modify the Web App content
- **Super Admin (Web App owner)** – this user can read, modify and delete any Web App content (including admins) and administer other user rights / roles. Administrator can delegate or share administrative rights to other users in the system.

1.4. References

[C4]: Canvas Presentation – [Architecture C4.pptx \(fhict.nl\)](#)

[SRS]: Software Requirements Specification

[KRU41]: The “4+1” view model of software architecture, Philippe Kruchten, November 1995,
<http://www3.software.ibm.com/ibmdl/pub/software/rational/web/whitepapers/2003/Pbk4p1.pdf>

1.5. Overview

In order to fully document all the aspects of the architecture, the Software Architecture Document contains the following subsections.

Section 2: describes the use of each view

Section 3: describes the architectural constraints of the system

Section 4: describes the functional requirements with a significant impact on the architecture

Section 5: describes the most important use-case realization.

Section 6: describes design’s concurrency aspects

Section 7: describes how the system is composed (C4)

Section 8: describes the Web App deployment

Section 9: describes any performance issues and constraints

2. Architectural Representation

This document details the architecture using the views defined in the C4 and “4+1” model [KRU41], but using the RUP naming convention. The views used to document the Web App are:

Use Case view

Audience: all the stakeholders of the system, including the end-users.

Area: describes the set of scenarios and/or use cases that represent some significant, central functionality of the system. Describes the actors and use cases for the system, this view presents the needs of the user and is elaborated further at the design level to describe discrete flows and constraints in more detail.

Related Artifacts: Use-Case Model, Use-Case documents

Logical view

Audience: Designers.

Area: Functional Requirements: describes the design's object model. Also describes the most important use-case realizations and business requirements of the system.

Related Artifacts: Design model

Process view

Audience: Integrators.

Area: Non-functional requirements: describes the design's concurrency and synchronization aspects.

Related Artifacts: (no specific artifact).

Module Decomposition view

Audience: Programmers.

Area: Software components: describes the modules and subsystems of the application.

Related Artifacts: Implementation model, components

Data view

Audience: Data specialists, Database administrators

Area: Persistence: describes the architecturally significant persistent elements in the data model

Related Artifacts: Data model.

Deployment view

Audience: Deployment managers.

Area: Topology: describes the mapping of the software onto the hardware and shows the system's distributed aspects. Describes potential deployment structures, by including known and anticipated deployment scenarios in the architecture we allow the implementers to make certain assumptions on network performance, system interaction and so forth.

Related Artifacts: Deployment model.

3. Architectural Goals and Constraints

Server side

The Web App will not be hosted in any web server, and it will be running locally, and connecting to a local Database server running on Docker. All communication with client has to comply with public HTTPS, TCP/IP communication protocol standards.

Client Side

Users will be able to access the Web App only online (after it is hosted). Clients/users are requiring using a modern web browser such as Mozilla Firefox 10, Internet Explorer 9, Microsoft Edge, latest versions of Google Chrome or Safari.

3.1. Security

Reader rights will be granted to any user accessing the application-landing page. Security for the Web App will be integrated according to the top 10 list from OWASP.

Admin user rights will be assigned through the integrated security. Only Super Admin user can add or remove other Admins and perform other administrative tasks.

Super Admin user rights will be assigned through the integrated security. Super Admin can access and perform any given task.

3.2. Persistence

Data persistence will be addressed using a relational database (SQL) and Non-relational database (MongoDB).









3.3. Performance

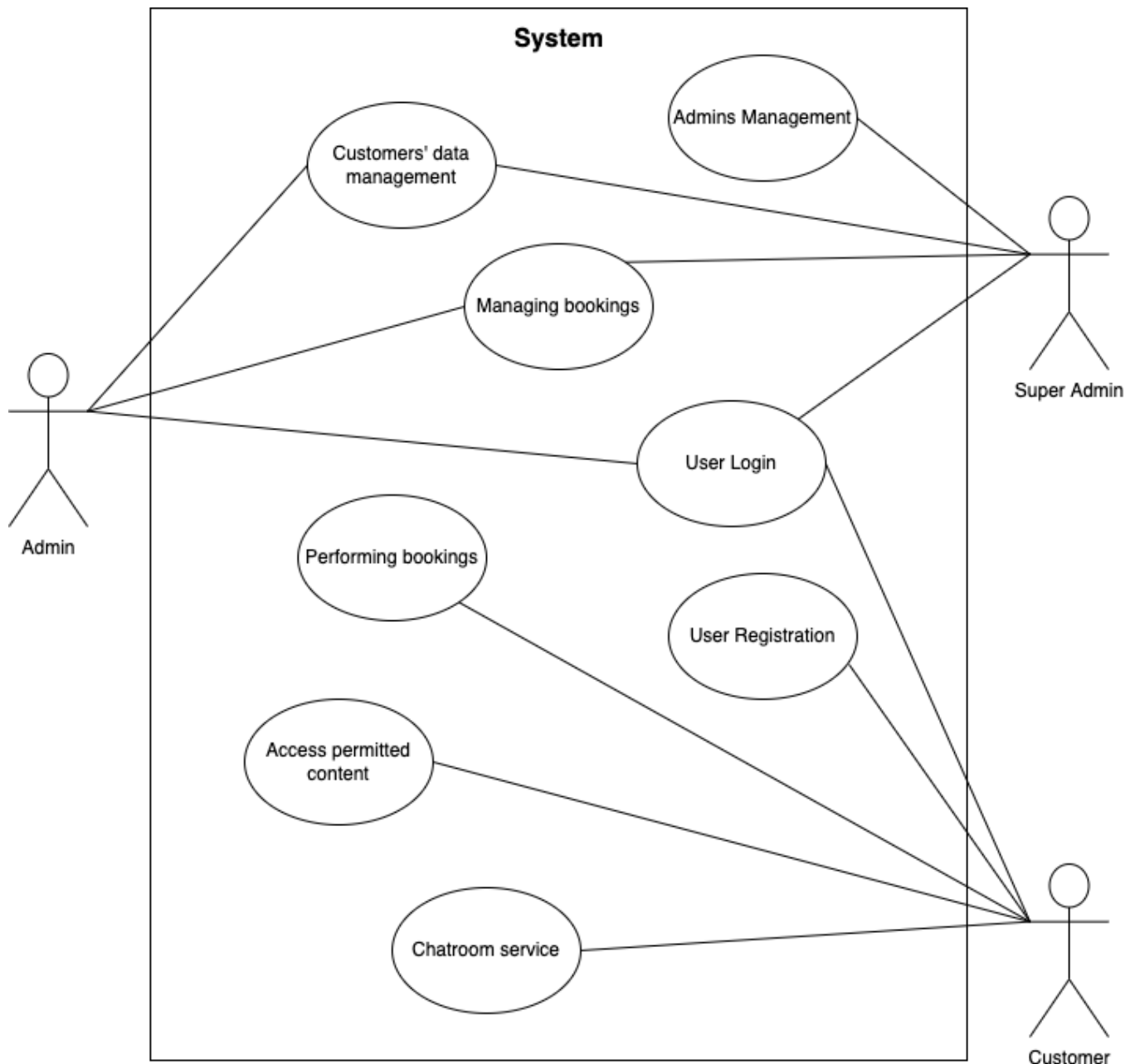
There is no particular constraints related to system performance.

It is anticipated that the system should respond to any request well under standard database and web server script timeouts, also system performance can depend on available hardware. In addition, upload / download times can depend on data size which in turn depends on user input. Therefore, actual performance can be determined only after system deployment and testing.

4. Use-Case View

This is a list of use-cases that represent major functionality of the final system [SRS]:

-  Access the Web App permitted content
-  Performing bookings
-  Managing bookings
-  User Registration
-  User Login
-  Admins creation and management
-  Managing all data for specified user / admin
-  Chatroom service



4.1. Actors

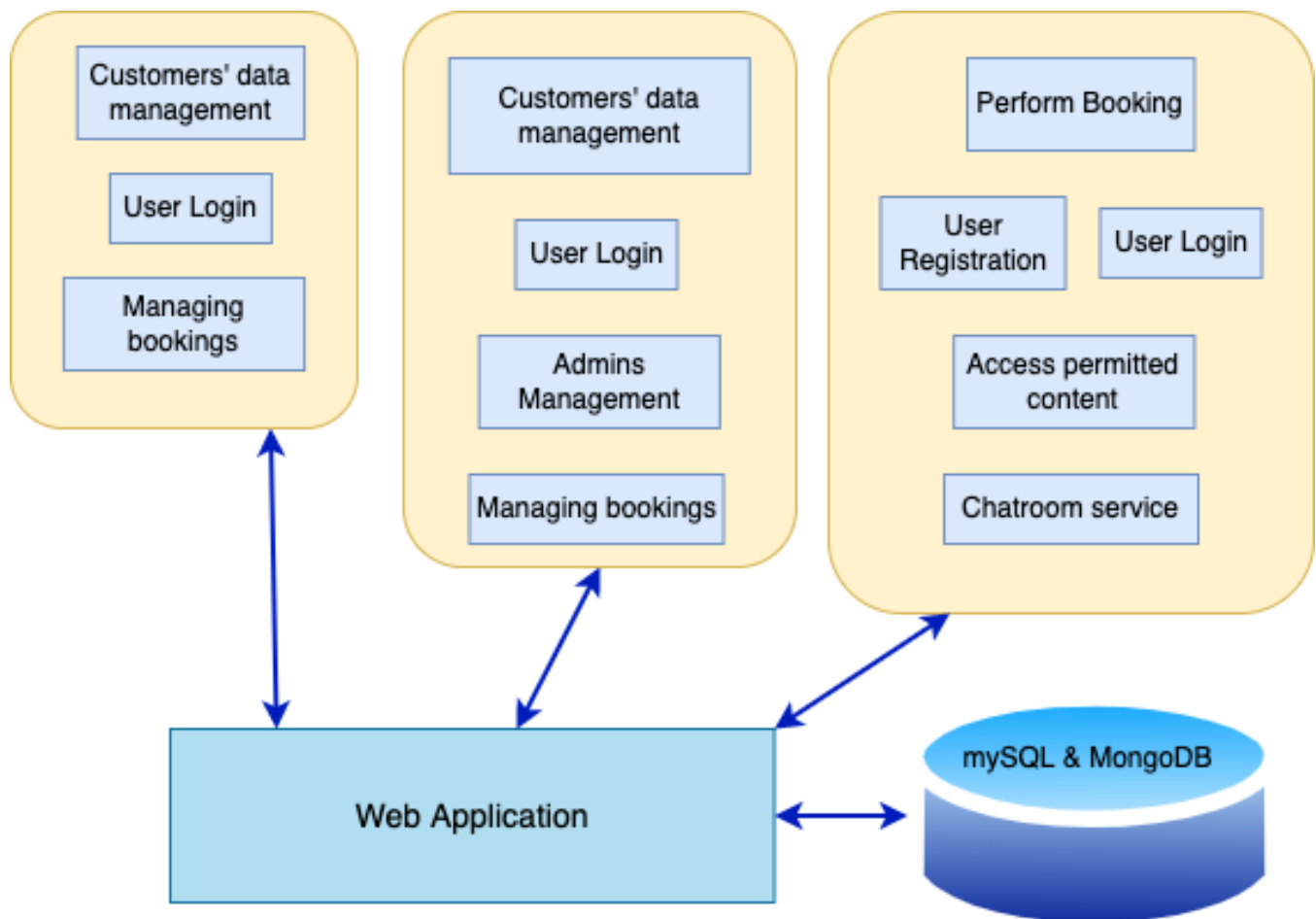
As described in the actors' correspondence diagram below, web user could be one of three types:

1. **Super Admin** – Has all the access privileges and can perform all kind of tasks operate on admins.
2. **Admin** – Has less privileges than the super admin and can only manage customers visitors and made bookings.
3. **Customers** – can login and make use of the chatroom service
4. **Normal visitors** – can only access and view the public Web App content.

5. **System – Web Server** is the fifth type of actor and is the system itself. It handles all the physical and logical process of the software.

4.2. Use-Case Realizations

Use case functionality diagram below describes how design elements provide the functionalities identified in the significant use-cases. Use cases are displayed as functionalities for the system. Functionality may enclose more than one use-case.



4.3. User Stories

Administrator stories:

As an administrator, I want to be able to login to my account, so that I can manage the web app services.

As an administrator, I want to be able to create kite lessons, so that customers can view and book from those kite lessons.

As an administrator, I want to be able to update kite lessons.

As an administrator, I want to be able to delete kite lessons.

Customer Stories:

As a customer, I want to be able to view the available kite lessons and learn more about them

As a customer, I want to be able to place a reservation for a kite lesson so that I can learn how to kite

As a customer, I want to be able to create an account so that I can login to the app

As a customer, I want to be able to login so that I can view my past bookings

4.4. Acceptance Criteria

As an administrator,

I want to be able to login to my account,
so that I can manage the web app services.

- Acceptance Criteria:
 - username, password fields must be filled with the expected correct data
 - Dashboard is displayed
 - Dashboard is not displayed in case of incorrect credentials

As an administrator,

I want to be able to create kite lessons,
so that customers can view and book from those kite lessons.

- Acceptance Criteria:
 - All fields must be filled with the correct data to create kite lesson
 - Kite lesson doesn't get created in case of empty fields or a missing field

As an administrator,

I want to be able to update kite lessons.

- Acceptance Criteria:
 - An existing kite lesson is displayed to be updated
 - All fields must be filled with the correct data to update the kite lesson
 - Kite lesson doesn't get created in case of empty fields or a missing field

As an administrator,

I want to be able to delete kite lessons.

- Acceptance Criteria:
 - Kite lesson is deleted
 - Kite lesson is not deleted if it has some bookings

As a customer,

I want to be able to view the available kite lessons and learn more about them

- Acceptance Criteria:
 - Kite lessons are displayed as brief items
 - Kite lesson with more details is displayed if clicked

As a customer,

I want to be able to place a reservation for a kite lesson
so that I can learn how to kite

- Acceptance Criteria:
 - All fields must be filled with the correct data
 - Reservation is made after all fields are filled
 - Booking number is communicated to the customer after the booking is made

As a customer,

I want to be able to create an account
so that I can login to the app

- Acceptance Criteria:
 - Account is created after all fields are filled with the correct data
 - Account cannot be created with the same credentials

As a customer,

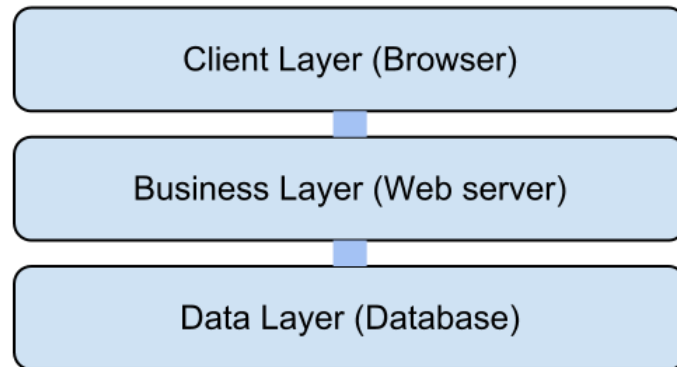
I want to be able to login
so that I can view my past bookings

- Acceptance Criteria:
 - Account is already created and active
 - Fields must be filled with correct data
 - Correct credentials are provided

5. Logical View

5.1. Overview

The Web App Software System is divided into layers based on the N-tier architecture [KRU41].



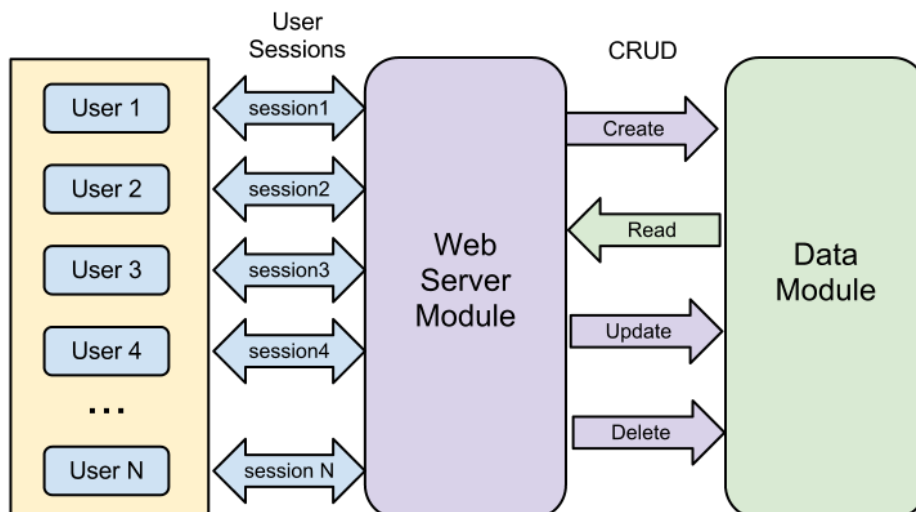
The layering model of the Web Online App is based on a responsibility layering strategy that associates each layer with a particular responsibility.

This strategy has been chosen because it isolates various system responsibilities from one another, so that it improves both system development and maintenance.

6. Process View

Due to disconnected nature of HTTP request / response and ability of relational database management system (RDMS), The Web App will handle multiple users simultaneously. Therefore, concurrency issues such as synchronous versus asynchronous mechanisms will be not considered in this document.

Process View Diagram



- ✚ User – Super Admin, Admin or Customer
- ✚ Session – HTTP session assigned by web server automatically
- ✚ CRUD – Create-Read-Update-Delete

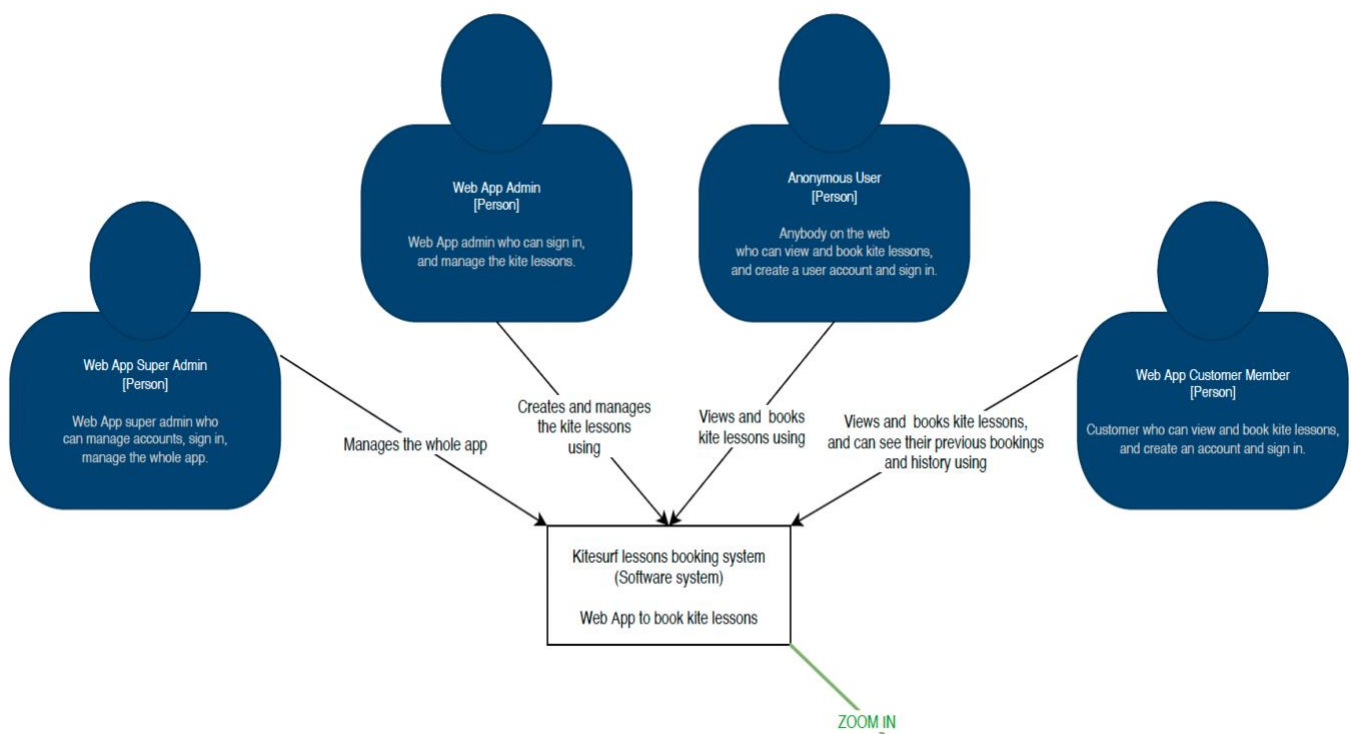
7. Module Decomposition View

Module decomposition view based on principles separation of concerns and abstraction and supports goals of modifiability and usability.

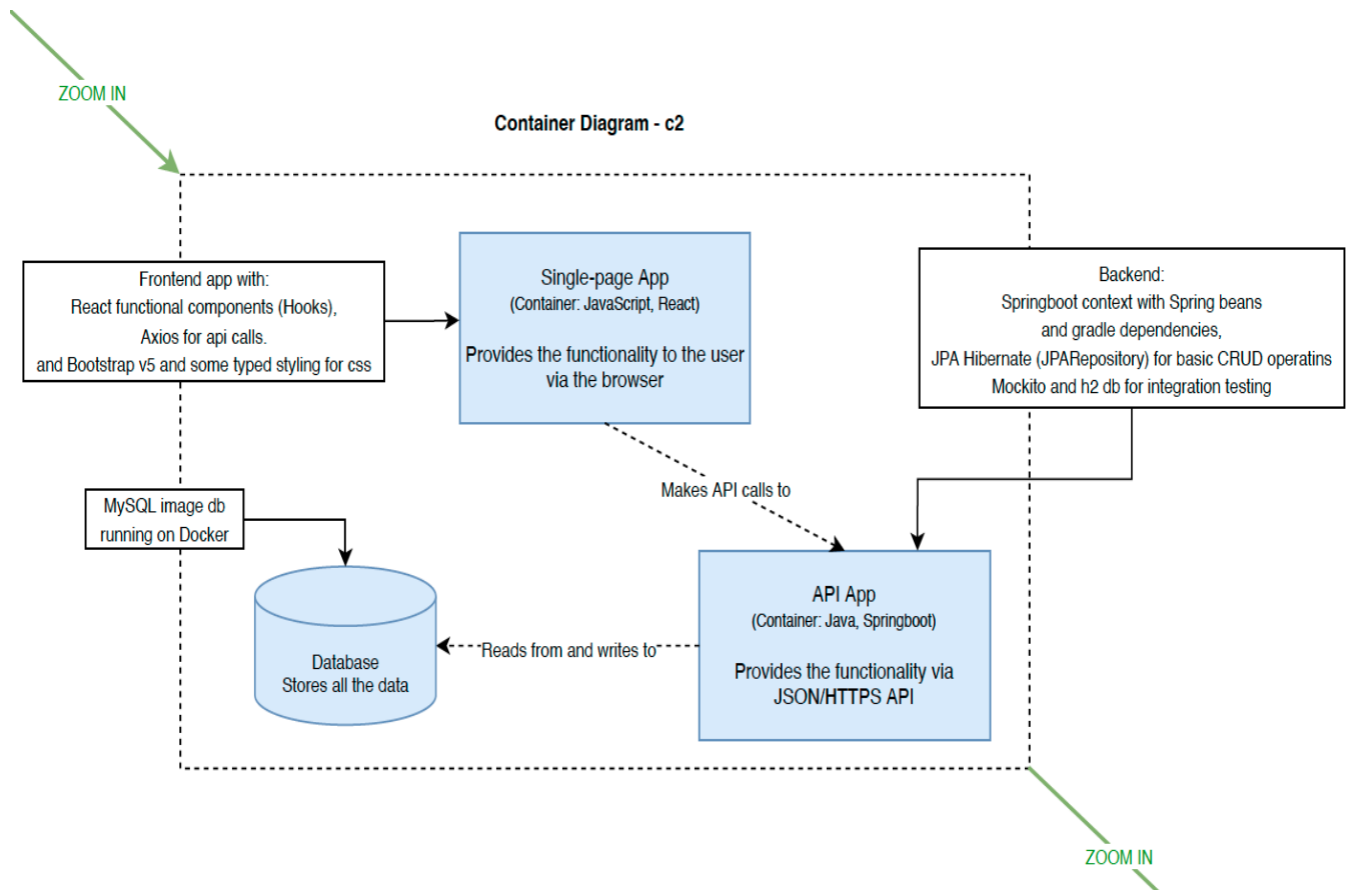
C4 diagram model is used to bring the reader closer to the Software System architecture.

System Context Diagram – c1

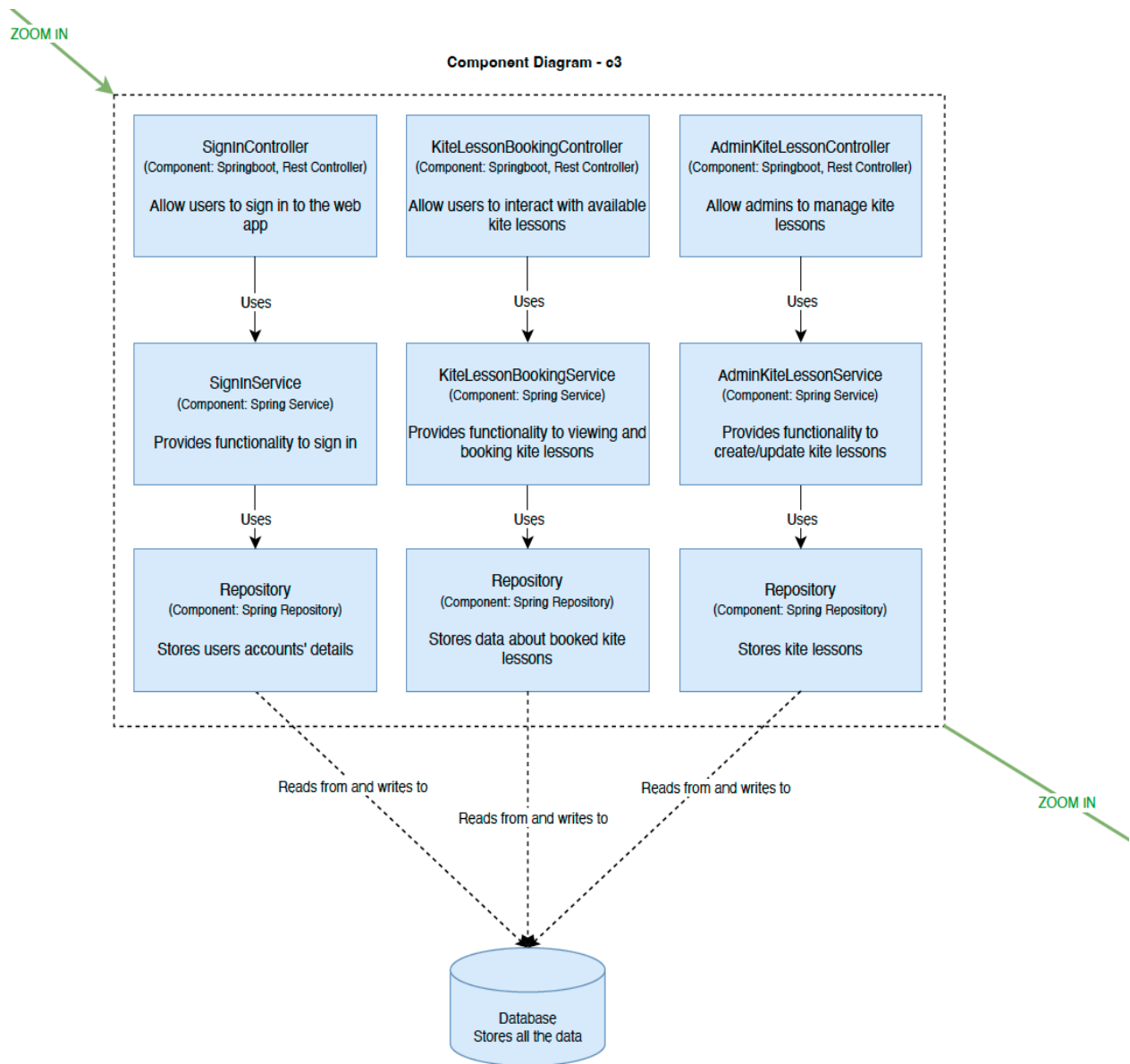
System Context Diagram - c1



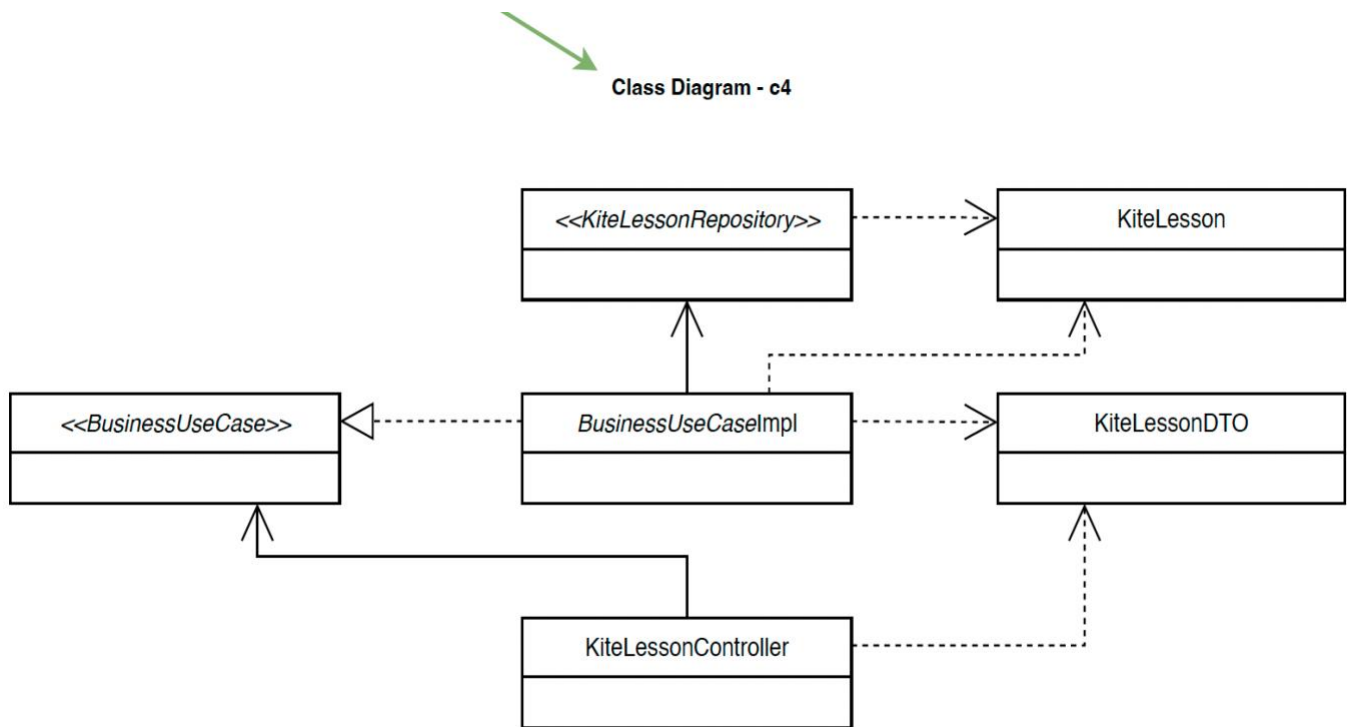
Container Diagram – c2



Component Diagram – c3



Class Diagram – c4



8. Deployment View

The Web App deployment has not been considered yet. All future implementation details will be included in this section.

9. Issues and concerns

- The Web App is not going to be deployed on a web server which means that online testing won't take place
- Due to the implementation of new fresh-learned technologies, there might be some system defects in the future as the software system becomes more complex or have more users
- Frontend react dependencies versions get updated more often which means there might be incoherent dependencies' versions implemented in different components