2021

Nadko Dentals Design Document

TSANKO NEDELCHEV

FONTYS | Hogeschool ICT

Version	Date	Author	Changes
1.0	22 Mar 2021	Tsanko Nedelchev	Initial Release
2.0	31 May 2021	Tsanko Nedelchev	Added C4 Architecture diagram to the Design Doc

Contents

Introduction

Nadko Dentals is an organizational application for dental technicians and dental practitioners as well as their assistants. It is a solution to the chaos of working with many people all day.

Problem Overview

The purpose of this application is to handle the day to day tasks of a dental practitioner's office. It takes into account the needs of the dental practitioner and the dental assistant. It provides full data manipulation in regards to setting, editing removing appointments adding, deleting and editing patient information and adding personalized notes to certain patients' details pages. It also notifies patients on the day of their appointments.

Solution Overview

Scope

Inside Scope	Outside Scope
"Nadko Dentals" will allow dental practitioners to view their patients.	"Nadko Dentals" is not aiming to replace proper medical documentation that is already in place.
"Nadko Dentals" will allow dental practitioners to add notes/remarks related to a patient	"Nadko Dentals" will not provide any medical insight.
"Nadko Dentals" will allow dental practitioners to add edit and remove appointments	
"Nadko Dentals" will allow dental practitioners to edit their patient's personal data records.	
"Nadko Dentals" will send notifications to the patient a day before their appointment.	

User Stories

1. Create an account

   As a patient

   I can create an account

   So that I can view/edit my personal information and make appointments.

   Acceptance criteria examples:

   An account is created

   I can login with the created account's credentials.

   Estimation: 10%

2. Request an appointment

As a patient

I can request an appointment with the dentist

So that I can plan my visit.

Acceptance criteria examples:

The available timeslots are shown.

I can request an appointment which then can be reviewed and approved by the dentist.

Estimation: 20%

3. View daily appointments

   As a dentist

   I can view my daily appointments

   So that I am informed on the tasks for the day.

   Acceptance criteria examples:

   Each task/patient appointment is displayed with its starting and ending times.

   Estimation: 10%

4. Receive notification email before an appointment

   As a patient

   I want to receive appointment notifications via email

   So that I can be reminded in case I forget.

   Acceptance criteria examples:

   I receive an email notification early in the morning on the day of the appointment.

   Estimation: 20%

5. View patient information

   As a dentist

   I can view a patient's information

   So that I may better follow that particular patient's procedures.

   Acceptance criteria examples:

   The patient's personal information is displayed.

   The patient's contact information is displayed.

   The patient's previous appointments are displayed.

   Estimation: 5%

6. Change personal information

   As a patient

   I want to be able to change my personal information from my profile page

   So that I can keep it relevant in case some of it changes.

   Acceptance criteria examples:

   I can view my profile page

   I can view my personal information

   I can change my personal information

   Estimation: 5%

7. Search for a specific patient using filters.

   As a dentist

   I can search through the list of patients

   So that I can more easily find the patient that I need.

   Acceptance criteria examples:

   I can search for a patient by their name.

   I can toggle filters that narrow the search even further.

   Estimation: 20%

8. Add/edit/delete notes for a patient

   As a dentist

   I need to be able to add different notes/remarks about a patient

   So that I can better keep track of the special circumstances of each patient.

   Acceptance criteria examples:

   I can add notes to a patient's account.

   I can view the remarks I've made in the past.

   I can change/delete non relevant remarks I have made in the past.

   Estimation: 10%

Constraints

Constraint	Description
Time	The period of time that is set for completing this project is 18 weeks divided into 6 sprints of 3 weeks each.
Experience	Some research will be needed to complete this project.
Quality	The end product needs to have a certain level of reliability as it will be used in the field.

Application Architecture

Front-end

"Nadko Dentals" front end will be handled by ReactJs

ReactJs is a JavaScript library for building user interfaces. It was developed by Facebook and is an open source framework that is used as a base in the development of single page or mobile applications. React is easy to use and easy to learn as it requires basic JS and HTML knowledge and provides countless features developed by the open source community. React uses the JSX syntax extension to define its components. It uses HTML quoting rendering and makes subcomponent rendering easier. One of the most useful react features is its ability to reuse components. Each component handles its own logic and its own rendering through the use of React state. One positively unique part about React is its virtual DOM. This means that the view rendered has already "considered" possible changes to the DOM that the user can make after which an algorithm determines the changes made to the virtual DOM and the virtual DOM sends only the changes that need to be implemented to the real DOM. This guarantees the shortest update time providing higher performance and a better user experience.
Another great thing about React is the abundance of developer tools available such as the React Developer Tools for the Google Chrome and Firefox browsers. It allows the developer to inspect the initialization of the React component hierarchies in the virtual DOM from the browser itself.

I have decided to use React for my personal project mainly due to its ecosystem. There are a lot of sources of information about react available and with its open-source nature it provides countless tools that make the job easier for new developers. Furthermore I have already had some experience with React as my group for the group project decided we are going to use Reactjs for the front-end implementation of the project.

Back-end

"Nadko Dentals" back-end API application will be handled by the Spring framework.

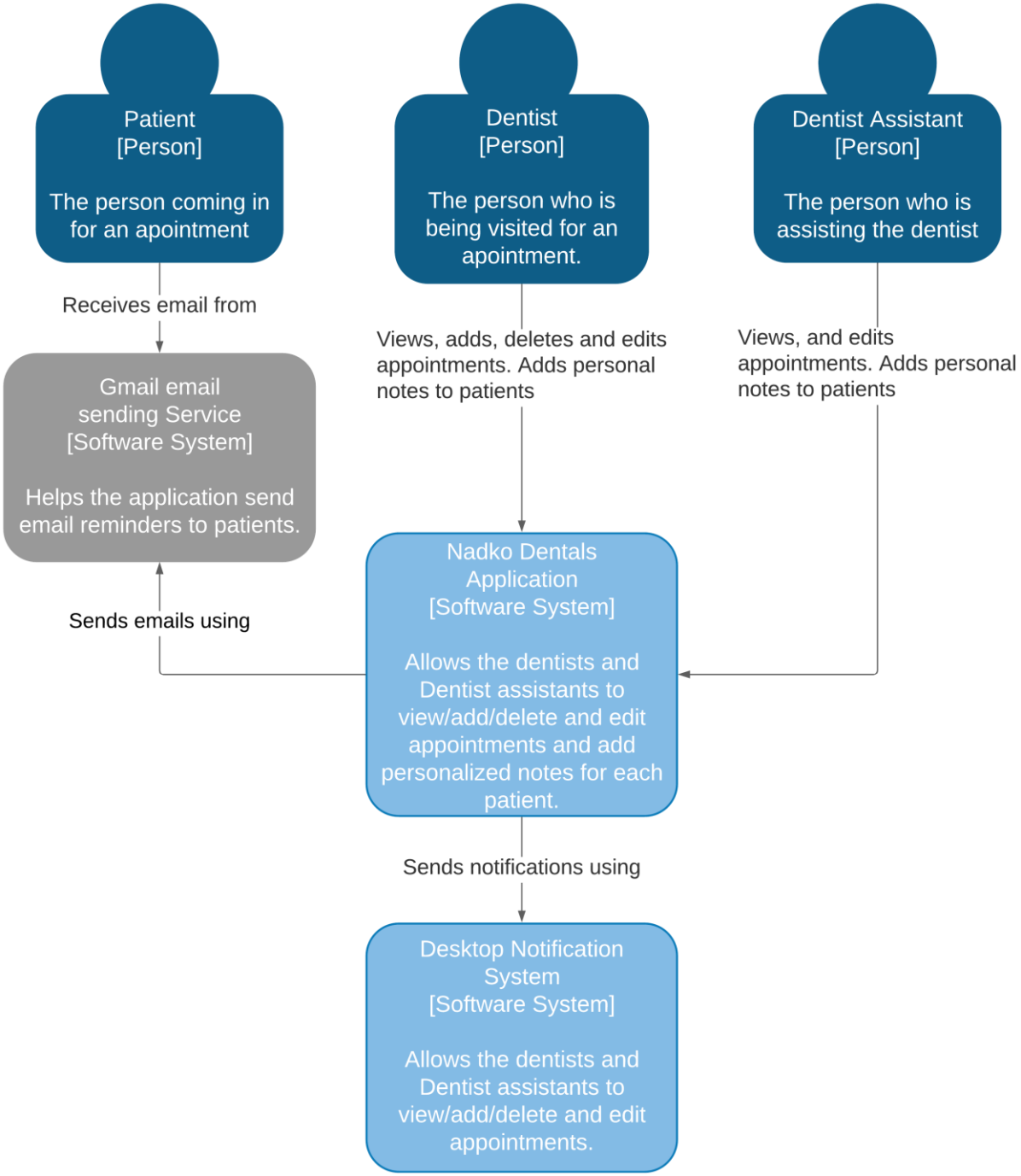
Spring framework is the most popular framework for developing enterprise Java applications. It is a modular framework which allows for selective functionality based only on the needs of the application. Spring makes testing simple because environment dependent code is moved to this framework. It also comes with a convenient API to translate technology-specific exceptions into consistent unchecked exceptions. The most important aspect of the Spring framework however is its dependency injection adaptation using Beans. Beans are the objects that form the backbone of a spring application that are managed by the Spring IoC(inversion of control) container. The Spring container is the heart of the Spring framework. It creates the objects, changes them, connects them and destroys them. It manages the different components that represent the application using Dependency injection.

Spring provides consistent and familiar model for data access through JDBC, JPA etc. while retaining the special traits of the underlying data store. It provides object mapping and dynamic query derivation

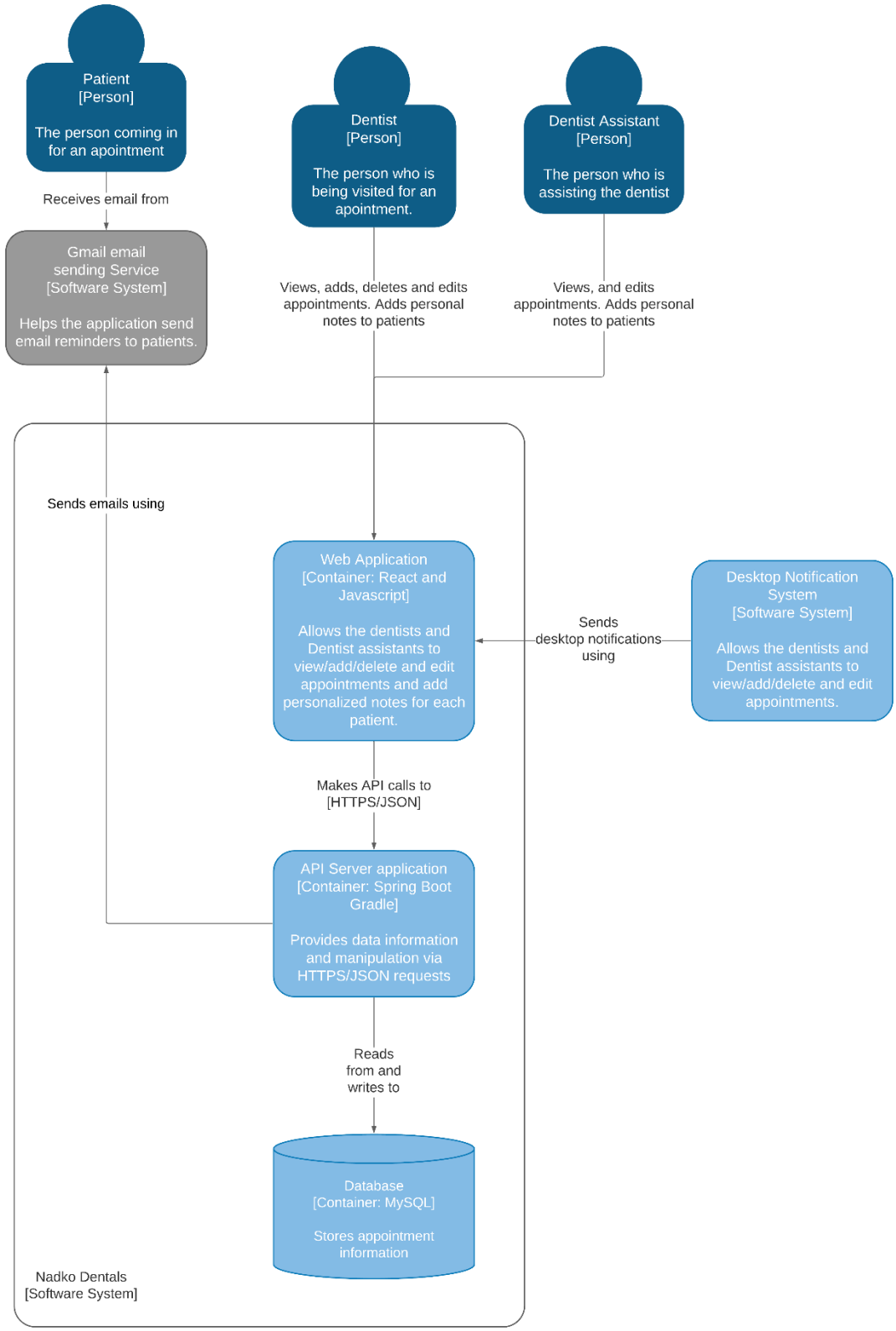
from repository method names with the possibility to integrate custom repository code. This allows the back-end API application to fetch data from the database.

Spring Framework is a complete package framework that provides countless useful features that will help create a fast and stable implementation of the back-end API application for the Nadko-dentals project.

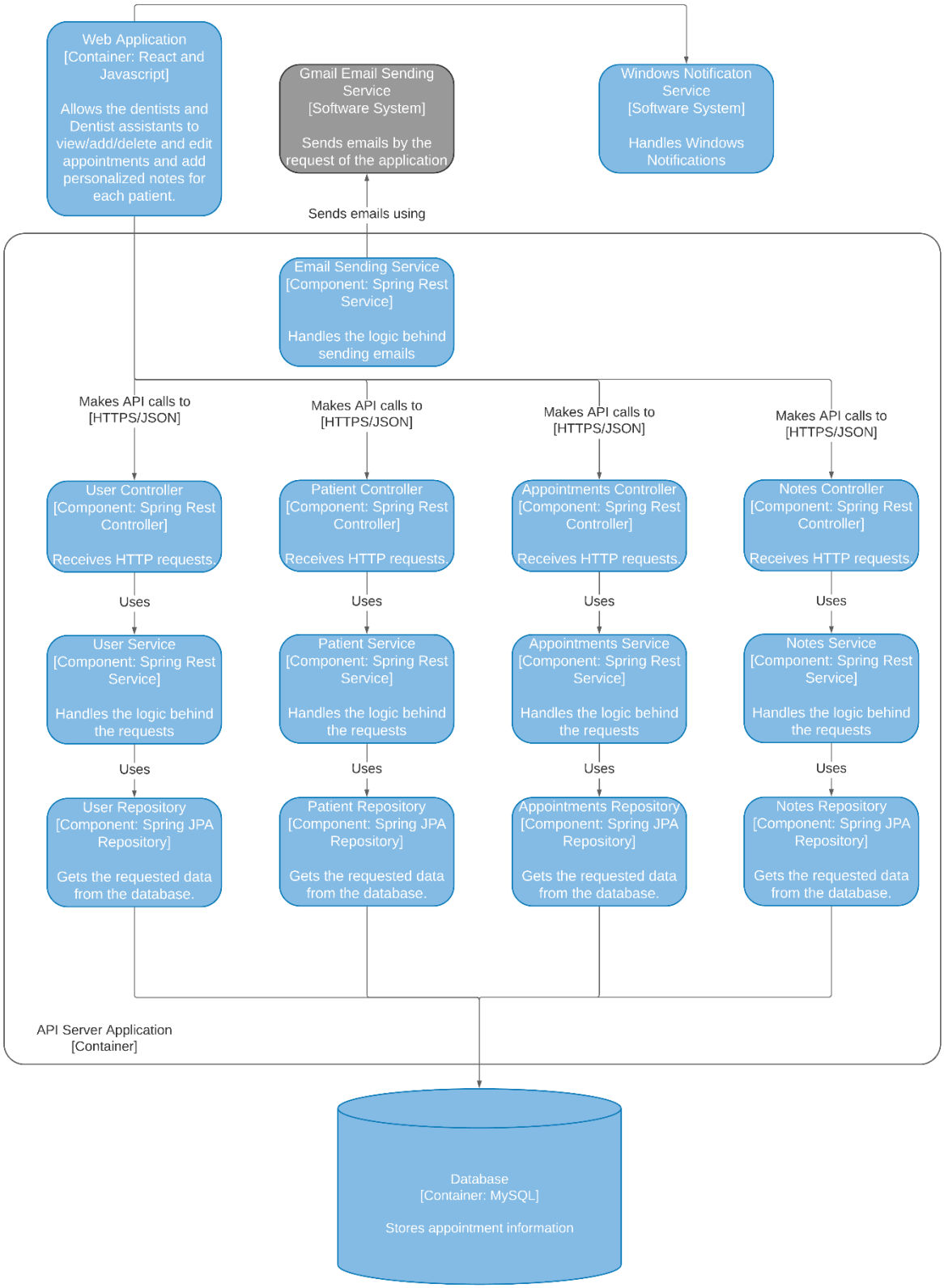
C1



C2


Patient
[Person]
The person coming in for an apointment
Dentist
[Person]
The person who is being visited for an apointment.
Dentist Assistant
[Person]
The person who is assisting the dentist
Receives email from
Gmail email sending Service
[Software System]
Helps the application send email reminders to patients.
Views, adds, deletes and edits appointments. Adds personal notes to patients
Views, and edits appointments. Adds personal notes to patients
Sends emails using
Web Application
[Container: React and Javascript]
Allows the dentists and Dentist assistants to view/add/delete and edit appointments and add personalized notes for each patient.
Sends desktop notifications using
Desktop Notification System
[Software System]
Allows the dentists and Dentist assistants to view/add/delete and edit appointments.
Makes API calls to [HTTPS/JSON]
API Server application
[Container: Spring Boot Gradle]
Provides data information and manipulation via HTTPS/JSON requests
Reads from and writes to
Database
[Container: MySQL]
Stores appointment information
Nadko Dentals
[Software System]

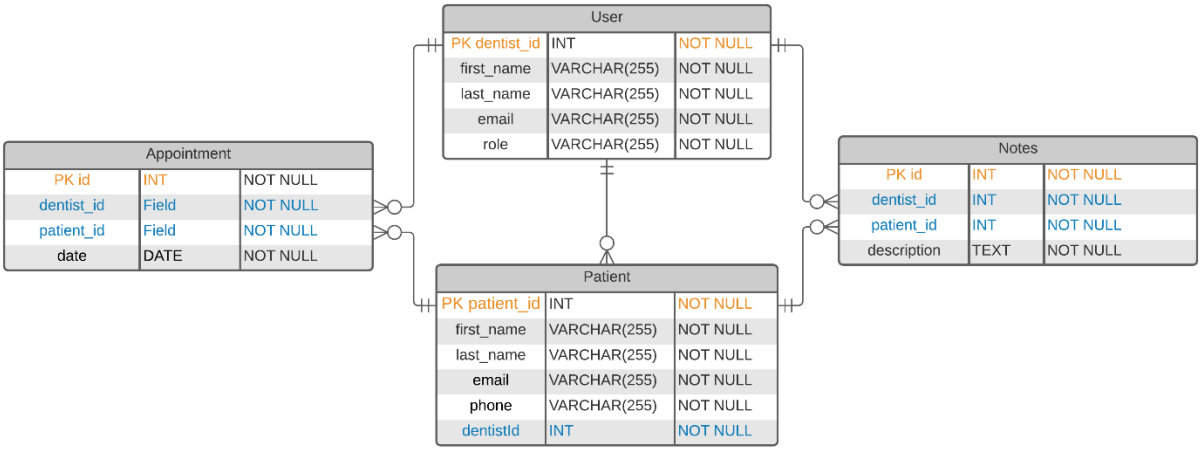
C3


Web Application
[Container: React and Javascript]
Allows the dentists and Dentist assistants to view/add/delete and edit appointments and add personalized notes for each patient.
Gmail Email Sending Service
[Software System]
Sends emails by the request of the application
Windows Notificaton Service
[Software System]
Handles Windows Notifications
Sends emails using
Email Sending Service
[Component: Spring Rest Service]
Handles the logic behind sending emails
Makes API calls to [HTTPS/JSON]
Makes API calls to [HTTPS/JSON]
Makes API calls to [HTTPS/JSON]
Makes API calls to [HTTPS/JSON]
User Controller
[Component: Spring Rest Controller]
Receives HTTP requests.
Patient Controller
[Component: Spring Rest Controller]
Receives HTTP requests.
Appointments Controller
[Component: Spring Rest Controller]
Receives HTTP requests.
Notes Controller
[Component: Spring Rest Controller]
Receives HTTP requests.
Uses
Uses
Uses
Uses
User Service
[Component: Spring Rest Service]
Handles the logic behind the requests
Patient Service
[Component: Spring Rest Service]
Handles the logic behind the requests
Appointments Service
[Component: Spring Rest Service]
Handles the logic behind the requests
Notes Service
[Component: Spring Rest Service]
Handles the logic behind the requests
Uses
Uses
Uses
Uses
User Repository
[Component: Spring JPA Repository]
Gets the requested data from the database.
Patient Repository
[Component: Spring JPA Repository]
Gets the requested data from the database.
Appointments Repository
[Component: Spring JPA Repository]
Gets the requested data from the database.
Notes Repository
[Component: Spring JPA Repository]
Gets the requested data from the database.
API Server Application
[Container]
Database
[Container: MySQL]
Stores appointment information

C4

C4 UML.pdf

Data Architecture



CI/CD setup