Dental Clinic Management System Requirements Specification

Version 1.1

April 14, 2018

**Table of Contents**

[1. Executive Summary 4](#_Toc511689767)

[1.1 Project Overview 4](#_Toc511689768)

[1.2 Purpose and Scope of this Specification 4](#_Toc511689769)

[2. Product/Service Description 5](#_Toc511689770)

[2.1 Product Context 5](#_Toc511689771)

[2.2 User Characteristics 5](#_Toc511689772)

[2.3 Assumptions 5](#_Toc511689773)

[2.4 Constraints 5](#_Toc511689774)

[2.5 Dependencies 6](#_Toc511689775)

[3. Requirements 6](#_Toc511689776)

[3.1 Functional Requirements 6](#_Toc511689777)

[3.2 Non-Functional Requirements 8](#_Toc511689778)

[3.2.1 User Interface Requirements 8](#_Toc511689779)

[3.2.2 Usability 8](#_Toc511689780)

[3.2.3 Performance 9](#_Toc511689781)

[Capacity 9](#_Toc511689782)

[Availability 9](#_Toc511689783)

[Latency 9](#_Toc511689784)

[3.2.4 Manageability/Maintainability 10](#_Toc511689785)

[3.2.4.1 Monitoring 10](#_Toc511689786)

[3.2.4.2 Maintenance 10](#_Toc511689787)

[3.2.4.3 Operations 10](#_Toc511689788)

[3.2.5 System Interface/Integration 10](#_Toc511689789)

[Network and Hardware Interfaces 10](#_Toc511689790)

[Systems Interfaces 10](#_Toc511689791)

[3.2.6 Security 11](#_Toc511689792)

[Protection 11](#_Toc511689793)

[Authorization and Authentication 11](#_Toc511689794)

[3.2.7 Data Management 11](#_Toc511689795)

[3.2.8 Standards Compliance 11](#_Toc511689796)

[3.2.9 Portability 12](#_Toc511689797)

[3.3 Domain Requirements 12](#_Toc511689798)

[4. User Scenarios and Use cases 13](#_Toc511689799)

[4.1. User Scenarios 13](#_Toc511689800)

[4.2. Use cases 21](#_Toc511689801)

[4.3. Use case diagrams 32](#_Toc511689802)

[4.3.1. Client Use cases 32](#_Toc511689803)

[4.3.2. Doctor Use cases 33](#_Toc511689804)

[4.3.3. Admin Use Cases 34](#_Toc511689805)

[4.3.4. General Use Cases 35](#_Toc511689806)

# Executive Summary

## Project Overview

Considering the heavy burden of multiple tasks dentists cope with daily, it seems of relevance to create a management system encompassing the handling of on-site tasks (in the clinic) and off-site tasks (outside the clinic) alike. The dentistry management system aims to create an easy-to-use, all-in-one tool for practitioners at every stage of their work.

The environment will be designed in such a way that it meets several important requirements presented by the practitioners and clients altogether, providing the latter with a fast and reliable way of managing their visits, payments, medical history, and treatment plans in a comprehensive and well-organized set of tools while aiding the management of the agenda and logistics.

In order to substantiate our goals, we find it very important to make a strong connection between the user’s needs and our implementation. This mindset is important to our approach, since we intend to continuously improve aspects of the product accordingly.

Furthermore, another important aspect of our work is providing different views of the system in a flexible way, making it easier for all types of users to access their information with ease.

To facilitate the aforementioned aspects in the design, the product shall incorporate different technologies such as PHP, Android, Firebase BaaS.

The implementation of these technologies will guarantee a high degree of optimization in terms of load balancing and refactoring. Using these technologies would also make it possible for users to access their data remotely, at all times.

Additionally, the system shall be organized in hierarchical cloud-based manner, ensuring some degree of abstraction and modularity, in order to make possible the management of different departments in the clinics independently, but with centralized control satisfying safety concerns.

## Purpose and Scope of this Specification

The purpose of this specification is to assess the current state of the product design and to document the entire process based on design issues and the audience.

This specification encompasses several aspects of the process being discussed in an as broad scope as possible. Thus in this scope we address the following:

* In depth documentation of the features of the product
* Technical overview of the system processes and views
  + This is discussed in Part 2.1 and throughout the document
* User and System Requirements
* Components & Functional/non-functional requirements
  + These are discussed in Part 3 in some detail
* Definition of users’ means of using and accessing the product
  + Use cases/scenarios discussed in Part 4
* Dependencies and Constraints
  + These are discussed in Part 2.4/5 of the Document

Aspects not included in the scope are as follows:

* Legislative requirements for the product
* Auditing and financial considerations of the product

# Product/Service Description

## Product Context

DCMS is an essential work-tool which serves our purpose and mission, in that it provides all necessary functionalities and benefits of a management system. Given the lack of a present system which fulfills the broad scope of requirements in the current market, DCMS is built with the user in mind, thriving on a high level of user-friendliness, thus making it an asset to patients and practitioners alike.

This product gives a clean, flexible and efficient solution to the daunting task of managing patient’s data and medical history on paper, with an electronic, cloud solution.

The flexibility is offered in several ways:

* Multiple platform accessibility: web and android
* Multiple ways to sign in: email and password, Gmail, Facebook, twitter, phone number
* Real time interaction and notifications between doctor and client

Many industries and businesses are now adopting this new philosophy, given the relatively cheap upfront cost of setting up and managing a cloud based platform of apps.

It is related to the existing system of placing orders with the many depots related to the clinic.

## User Characteristics

Our customers include the entire staff of a dental clinic: dentists, clients and administrative staff.

* Administrator / Manager

Might as well be one of the other staff members

* Dentist
* Dental Assistant
* Treatment Coordinator
* Receptionist
* Other employee
* Client

## Assumptions

* The core part of our software is implemented in the android platform.
* This comes with the small overhead of having to be equipped with android devices, in case part of the personnel uses iOS devices. The clients are expected to have a basic knowledge of smartphones.
* The administrative and managerial part will be in a web interface. This assumes an active internet connection and a device which has access to the internet (laptop, PC, mobile device)
* We assume the staff is familiar with the English language.
* Also, we assume all client Dental Clinics follow a similar workflow.
* Based on the technologies we intend to use, we assume a consistent Cloud-Client connection.

The advantages of using android devices are:

* High variety of form factors (tablet, mobile phones)
* High variety of devices in different price ranges

## Constraints

Design options are constrained by:

* real time communication between doctors and clients
* real time notifications on the client side
* multiple platform integration
* security concerns on data accessibility
* high resolution photo evidences in the medical record
* offline usability

Chosen design options give us the following technical constraints:

* Using Firebase Firestore as a database
* NoSQL database design limits the execution of complex queries

## Dependencies

Dependencies that affect the requirements:

* This product is dependent on a successful communication with the related dental depots. This is achieved through the weekly generated reports of the used products quantities.
* The product is also dependent on the Cloud performance.

# Requirements

**Priority Definitions**

* Priority 1 – The requirement is a “must have” as outlined by policy/law
* Priority 2 – The requirement is needed for improved processing, and the fulfillment of the requirement will create immediate benefits
* Priority 3 – The requirement is a “nice to have” which may include new functionality

It may be helpful to phrase the requirement in terms of its priority, e.g., "The value of the employee status sent to DIS **must be** either A or I" or "It **would be nice** if the application warned the user that the expiration date was 3 business days away". Another approach would be to group requirements by priority category.

## Functional Requirements

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req#** | **Requirement** | **Comments** | **Priority** | **Date Rvwd** | **SME Reviewed / Approved** |
| R\_S\_1 | Handling multiple account types | Based on the status of the user, each will have their own view of the system | 1 |  |  |
| R\_S\_2 | Handling unregistered users | The system should be able to handle unregistered users, providing restricted information | 3 |  |  |
| R\_S\_3 | Account login restriction | The admin should be able to login only on the web, clients and doctors only on the app. | 3 |  |  |
| R\_D\_1 | Keeping track of medical records and past treatments history | Doctors should be able to view and update patients’ medical records. | 1 |  |  |
| R\_D\_2 | Providing availability agenda | Doctors provide their availability information to the system. | 3 |  |  |
| R\_D\_3 | Submitting performance reports | Doctors or other staff members can compile reports on a weekly or treatment basis. | 2 |  |  |
| R\_D\_4 | Managing appointments | Doctors should be able to manage their appointments. | 2 |  |  |
| R\_D\_5 | Viewing work related info | Doctors should be able to view their specific work information such as wage and working hours. |  |  |  |
| R\_A\_1 | Managing new staff entries | The administrator should be able to approve the creation of doctors’ accounts and other employees’ data. | 2 |  |  |
| R\_A\_2 | Adding treatments offered by the clinic. | The administrator can create new treatment plans based on patients’ requests. | 2 |  |  |
| R\_A\_3 | Managing treatments based on type and date | The administrator can edit and update treatments to better arrange them. | 1 |  |  |
| R\_A\_4 | Staff payroll management | The administrator manage payrolls based on base salary and percentage according to treatments’ pricing. | 1 |  |  |
| R\_A\_5 | Pricing and services | The administrator should be able to add and edit services offered by the clinic, especially their pricing info. . | 1 |  |  |
| R\_A\_6 | Summarizing and generating financial reports. | The admin should be able to dynamically create financial reports based on current financial data. | 2 |  |  |
| R\_A\_7 | Weekly inventory and logistics management. | The admin should be able to add weekly inventory reports regarding materials and their availability. | 2 |  |  |
| R\_C\_1 | Sign up and first login | The patients should be able to create their accounts providing credentials such as email or phone no. | 1 |  |  |
| R\_C\_2 | View list of services | The patients should be able to view the list of services offered by the clinic when signed in or not. | 2 |  |  |
| R\_C\_3 | Service details | Each service should be displayed with a description, list of doctors and list of photos. | 2 |  |  |
| R\_C\_4 | Extra profile info | After every successful new Gmail account login, the patient should provide birthday and phone no. credentials. | 1 |  |  |
| R\_C\_5 | Clinic info | The patient should be able to view basic information about the clinic including location, open hours, phone no. etc.. | 3 |  |  |
| R\_C\_6 | Scheduling new appointments | The patients should be able to create an issue for a new appointment, providing information about their problem. | 1 |  |  |
| R\_C\_7 | Receive notifications on new appointment | The patients should be notified whenever their appointment has been set with the correct time and date info. | 2 |  |  |
| R\_C\_8 | Access personal medical files and records | Patients should be able to access their medical file at all times, in order to have knowledge of their treatment history. | 1 |  |  |
| R\_C\_9 | Client is presented with the paperless consent agreement. | Only applicable to select treatments, like surgeries and orthodontics. | 2 |  |  |

## Non-Functional Requirements

### User Interface Requirements

* Different screen resolutions based on devices
* Receive native push notifications in real time
* Sliding navigation drawer for the app
* Static navigation drawer for the web page
* Simplistic and responsive design

### Usability

* Accessibility
  + The software shall be easy to access remotely and at all times, since both patients and doctors will use the application on their devices.

* Responsiveness
  + The software shall be responsive both in design and data transactions, especially because of the reliance on the Cloud services.

* Flexibility
  + The software shall be easy to update in order to accommodate new requirements
  + The software shall be designed in such a way that the isolation and management of errors is possible

* Effectiveness
  + The software shall provide both staff and clients with practical tools of managing their data and with a convenient way of communicating their needs across the platform.

* Efficiency
  + The software will provide users and administrators with a fast and reliable way of accomplishing their goals such as creating appointments or updating medical information in little time at their own convenience.

### Performance

#### Capacity

The backend is built on top of Google’s infrastructure and thus scales very well horizontally.

* Database writes are limited to 2500 per second, which will be more than enough for our use case.
* Maximum concurrent connections for mobile/web clients are limited to 100000 per database.
* Maximum API request size is 10 MB
* Maximum number of documents that can be passed to a Commit operation in a transaction is 500
* Maximum number of composite indexes for a database is 200
* Maximum function call depth is 20

#### Availability

* The app will be live 24/7
* Is has a very low probability of downtime, around 0.05%
* It will be region independent, but available in english only
* Impact of downtime will be very minimal, considering the high reliability of the Google infrastructure.

#### Latency

Database operations will have a latency of approximately 100 ms in Cloud Firestore, and 10ms in Realtime Database.

### Manageability/Maintainability

#### Monitoring

The system will be subject to periodic evaluation. This evaluation will be performed by assessing the data integrity and also by monitoring error logs generated automatically.

Few corner cases shall be predicted and handled within the design in order to suppress non substantial errors and to detect and handle substantial errors appropriately.

In order to correct the errors, the administrator shall be able to follow specific procedures with many prompts and validations.

#### Maintenance

In order to isolate and manage issues easily, the system shall be designed in an atomic and modular manner. This will be evident in the separation of views for different user types and the avoidance of rigid relational constraints in terms of database. Also the administrator shall be provided with a proper interface to perform maintenance operations.

#### Operations

Specify any normal and special operations required by the user, including:

* Approval of a major transaction such as Sign-up or Medical record access
  + Data integrity is not possible without the administrator’s approval
* Handling of idle and unattended periods in the app
  + The user shall operate under some constraints while some of his operations are not approved
* Backup operations
  + These operations shall be handled by the cloud storage

### System Interface/Integration

Specify the use of other required products (e.g., a database or operating system), and interfaces with other systems (e.g., UWHires package interfaces with PubCookie and ODS, HEPPS system interfaces with Budget system). For each interface, define the interface in terms of message format and content. For well-documented interfaces, simply provide a reference to the documentation.

Outline each interface between the product and the hardware or network components of the system. This includes configuration characteristics (e.g., number of ports, instruction sets), what devices are to be supported, and protocols (e.g., signal handshake protocols).

#### Network and Hardware Interfaces

The app will use either Wi-Fi, or mobile data to connect to the internet. Other network related issues are automatically handled by Firebase Infrastructure, including connection monitoring, operation queueing during offline periods, etc.

#### Systems Interfaces

The users and doctors will be able to authenticate using the following methods, but the app uses a unique identifier which is not affected by the possibly different sign in method.

* classic username and password
* email and password
* Gmail account
* Facebook
* Twitter
* Phone number

The signing of the consent will be done electronically, complying with all legislative regulations, according the specified template by the clinic.

### Security

#### Protection

* Firebase Realtime Database Rules determine who has read and write access to your database, how your data is structured, and what indexes exist.
* These rules live on the Firebase servers and are enforced automatically at all times.
* Every read and write request will only be completed if your rules allow it.
* By default, your rules are set to allow only authenticated users full read and write access to your database.
* This is to protect your database from abuse until you have time to customize your rules or set up authentication.

#### Authorization and Authentication

Firebase Authentication provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to our app.

Firebase Authentication integrates tightly with other Firebase services, and it leverages industry standards like OAuth 2.0 and OpenID Connect, so it can be easily integrated with our custom backend.

To sign a user into the app, you first get authentication credentials from the user. These credentials can be the user's email address and password, or an OAuth token from a federated identity provider. Then, you pass these credentials to the Firebase Authentication SDK.

The backend services will then verify those credentials and return a response to the client.

### Data Management

* Cloud Firestore and Realtime Database, which we will primarily use are NoSQL databases.

NoSQL stands for “non SQL”, but also “not only SQL”, and it is a non relational database, which stores data in a large file usually. It offers more flexibility in database design, which does not restrict us in the relations between entities. It also scales better horizontally, meaning that it is better distributed in a more efficient network of nodes.

* Data is saved in a JSON tree. This offers easy encoding and decoding directly from Java objects in Android.
* In Cloud Firestore, differently from the Realtime Database, data is stored in Documents and Collections, thus offering better query support, and a offline first approach.
* Data access rules are clearly defined in section 3.1.6.1

### Standards Compliance

Dental Agreement Contract which includes the “Consent to Proceed” is a legal documentacion signed by the user before each specific treatment. In this document is explained in details the procedure of the treatment, the materials used, the drug prescription, the treatment fee and the clients responsibilities in case there is any problem. This is a documentation that preserves the rights of clients and doctors according to law and ethics.

### Portability

* Use of Firebase products, which offer flexibility in different platforms, such as: Android, iOS, Node.js, Java, Python and GO
* Real time updates
* Offline-first approach
* The same backend will power both the android client and web client, with possibility of adding an iOS client in the future, with not many interventions in the underlying infrastructure.

## Domain Requirements

The system manages everything related to a dental clinic, including specific features like Perio charting, anamnesis, and appointment scheduling in the style of a dental clinic.

# User Scenarios and Use cases

* 1. **User Scenarios**

*Notice:* *We are using the terms “User” and “Client” interchangeably, depending on context.*

Scenario C1: User is not logged in

1. User is presented with a list of services.
2. User is presented with detailed info about the clinic.
3. User is presented with the option to Sign in.

Scenario C2: User opens a service from the list

1. User is presented with a detailed view of the service, which includes detailed info, list of doctors and supporting photos.
2. User can share or bookmark the specific service.

Scenario C3: User opens a supporting photo

1. On the detailed view of the service, user taps one of the photos in the list.
2. The photo is opened in fullscreen.
3. User is able to share the photo in other apps like Google Keep or Messenger.

Scenario C4: Users signs in for the first time

1. User chooses to sign in, by pressing a button in the navigation drawer.
2. It is redirected to Sign in activity.
3. User is presented with 2 options: Sign in using Google or using their phone number directly.

Scenario C5: User chooses to sign in using their phone number

1. User presses “Sign in with phone number” button.
2. User selects the country, with the corresponding prefix.
3. User enters the phone number, without the 0 prefix.
4. User presses “Verify phone number”.
5. An SMS arrives within seconds, and automatic verification happens, if Google Play Services version installed is > 22
6. Else, the user has to input the received code manually.
7. On Success, User has successfully signed in using their phone number.

Scenario C6: User chooses to sign in using Google account

1. User presses “Sign in with Google” button.
2. User selects an account from the list of signed in accounts, or adds a new one manually.
3. User is successfully signed in using their Google account.

Scenario C7: First time user sign in, add other personal data

1. If User is signed in for the first time with a Google account
2. It is presented with a form where he can enter his birthday and phone number.
3. If User is signed in for the first time with a phone number, it should also enter his name, email and birthday.
4. User clicks Ok
5. Registration is successful and complete.

Scenario C8: User signs in (not for the first time)

1. User signs in with one of the methods.
2. In the database, an entry with the details exists, indicating that it is not a new user.
3. User is greeted with “Welcome back”
4. Name, email and profile picture are loaded from the database entry.
5. Sign out button is added.

Scenario C9: User signs out

1. User is already signed in.
2. User clicks Sign out.
3. User is signed out, and the next time he visits the app, will not be automatically signed in.
4. Profile picture and Name fields are replaced with the default values.
5. Sign out button is not shown.

Scenario C10: User opens “My Profile” section.

1. User clicks “My Profile”
2. User is presented with 3 tabs: “Info”, “Treatments” and “Anamnesis”.
3. In “Info”, User sees his account details and photo
4. User can update his phone number.

Scenario C11: User opens “Treatments” section of ‘My Profile”

1. User opens “My Profile”.
2. User scrolls to “Treatments” tab.
3. User is presented with a list of all treatments, till date.
4. Each treatment information card includes the name, date started and status.

Scenario C12: User opens a specific Treatment

1. User opens “My Profile”.
2. User opens “Treatments” tab.
3. User taps on one of the treatments.
4. User is presented with the list of Appointments in that specific treatment.
5. User is presented with the status of the Treatment, “undergoing”, “finished”.
6. User can see the payment status of the Treatment, “paid”, “unpaid”, “partially paid”.

Scenario C13: User opens a specific Appointment

1. User taps on one of the Appointments in the Treatment.
2. User is presented with a detailed description of the condition, and has the dentists recommendations regarding medications and future care.
3. User can see the accompanying photos with his appointment.

Scenario C14: User opens Anamnesis section

1. User opens “My Profile”
2. User scrolls to “Anamnesis” tab on the right.
3. User is presented with their personal Anamnesis, which gets edited by the specific doctor.
4. User can share or print a pdf version of their Anamnesis.

Scenario C15: User manages appointments

1. On the home screen, User scrolls to the right, on “Appointments” section.
2. User is presented with all his past and future appointments.
3. Each appointment information card has a title, Doctor and service required, date and status which is indicated by a color (Red, Green on Yellow)

Scenario C16: User sets a new appointment

1. On the home screen, User scrolls to the right, on “Appointments” section.
2. User taps on the floating action bar on the bottom right to schedule a new appointment.
3. User specifies the day he wishes to schedule the appointment, along with the service required and an optional descriptive message.

Scenario C17: User is assigned a specific appointment

1. Administrator assigns a timeslot in the specified day to the user’s appointment.
2. User receives a push notification, even if the app is closed.
3. User confirms or rejects the appointment.
4. User receives a reminder SMS 2 hours before the appointment.

Scenario D1: Doctor creates an account

1. Doctor signs in the app as a regular user, using their preferred method.
2. The administrator assigns the doctor’s profile as an employee, by adding the required extra fields.
3. The next time the doctor signs in, he is presented with the Doctor’s view.

Scenario D2: Doctor is not an employee of the clinic anymore

1. The administrator removes the doctor’s account from the list of employees.
2. The doctor can access the app as a regular user.

Scenario D3: Doctor opens the app as an already signed in user

1. The user is checked if exists in the list of employees.
2. If positive, the specific View for the Doctor is shown.
3. Doctor is presented with a dashboard consisting of “My appointments”, “My patients”, “Services”, “My availability”.

Scenario D4: Doctor adds his available hours in the system

1. On every Saturday, the doctor should enter his available hours for the next week.
2. Doctor opens the Dashboard on the Home screen.
3. Doctor taps “My availability”
4. Doctor clicks the floating action bar to add different time-spans for each day of the upcoming week and finally clicks submit.
5. This information is presented to the administrator, who adjusts the timetable according to these specifications.

Scenario D5: Doctor views the list of appointments.

1. On the dashboard, the user taps on “My Appointments” tab.
2. The doctor is presented with a tabbed view consisting of “Today’s Appointments” and “Upcoming appointments”.
3. On each tab, there is a list with all the assigned appointments, ordered according to the start time.

Scenario D6: Doctor manages his patients and treatments.

1. Doctor taps on “My Patients” tab.
2. It is presented with a list of users.
3. Doctor presses on one of the records in the list.
4. The profile info of the patient in displayed.
5. The doctor can manage or start a new treatment.

Scenario D7: Doctor starts a new appointment in a treatment.

1. Doctor selects a user.
2. Doctor selects a treatment or starts a new one.
3. Doctor adds a new appointment.
4. Doctor enters detailed description of the appointment, which the user can later access in his account.
5. Doctor can add accompanying photos, and use a Perio chart, to better keep track of the problems.
6. Doctor assigns a price to the individual appointment, which gets added to the treatment total.

Scenario A1: Admin user login (successful):

1. Admin opens the /login page
2. Admin chooses an authentication method such as Google Sign-in or Facebook Sign-in by clicking on the respective button.
3. A popup window is displayed, prompting the user to choose an account
4. After selecting the account, the user might need to provide credentials for their account unless they are already signed in.
5. The authentication request is sent to the cloud
6. Upon successful authentication, the user is redirected to the Admin Panel page.

Scenario A2: Admin user login (failure):

1. Admin opens the /login page
2. Admin chooses an authentication method such as Google Sign-in or Facebook Sign-in by clicking on the respective button.
3. A popup window is displayed, prompting the user to choose an account
4. After selecting the account, the user might need to provide credentials for their account unless they are already signed in.
5. The authentication request is sent to the cloud
6. Upon authentication failure, the user is displayed an error message informing them about the error.

Scenario A3: Admin manages their profile

1. Admin is logged in
2. He/she navigates to the My Profile button on the Dashboard
3. After clicking the My Profile button, the personal information page is rendered within the same page, containing editable text fields/labels
4. He/she selects field to update and edits the current information
5. He/she clicks Save Changes
6. The form is submitted
   1. In case of error, an error message is displayed above the edit form
   2. In case of success, a success message is displayed above the edit form

Scenario A4: Admin view and manage existing services

1. Admin is already logged in
2. He/she navigates to the Home button unless the button is selected (highlighted)
3. When the home info is fully rendered within the current page, the admin clicks on the respective card with text : Services
4. After clicking the card, the list of services is displayed in a tabular form.
5. The admin clicks the Edit button for a specific service row.
6. A modal form is displayed
7. After changing the proper text fields, the admin clicks submit.
8. The form is submitted
   1. In case of error, an error message is displayed at the top of the modal
   2. In case of success, the modal is closed and the information is changed in the table.

Scenario A5: Admin adds new services

1. Admin is already logged in
2. He/she navigates to the Home button unless the button is selected (highlighted)
3. When the home info is fully rendered within the current page, the admin clicks on the respective card with text : Services
4. After clicking the card, the list of services is displayed in a tabular form.
5. The admin clicks the New button
6. A modal form is displayed
7. After writing on the proper text fields, the admin clicks submit.
8. The form is submitted
   1. In case of error, an error message is displayed at the top of the modal
   2. In case of success, the modal is closed and the information is displayed in the table in a new row

Scenario A6: Admin views clients’ information

1. Admin is already logged in
2. He/she navigates to the Home button unless the button is selected (highlighted)
3. When the home info is fully rendered within the current page, the admin clicks on the respective card with text : Clients
4. After clicking the card, the list of users is displayed in a tabular form.
5. The admin clicks on any client’s row.
6. A new page is rendered on top of the clients’ list, containing the specific user profile information.
7. After clicking on certain labels or profile photo, the content is made editable where applicable.
8. When applicable, the edited information is submitted and saved

Scenario A7: Admin views staffers’ information

1. Admin is already logged in
2. He/she navigates to the Home button unless the button is selected (highlighted)
3. When the home info is fully rendered within the current page, the admin clicks on the respective card with text : Staff
4. After clicking the card, the list of staffers is displayed in a tabular form.
5. The admin clicks on any staffer’s row.
6. A new page is rendered on top of the staffers’ list, containing the specific user profile information.
7. After clicking on certain labels or profile photo, the content is made editable where applicable.
8. When applicable, the edited information is submitted and saved.

Scenario A8: Admin adds new staff members

1. Admin is already logged in
2. He/she navigates to the Home button unless the button is selected (highlighted)
3. When the home info is fully rendered within the current page, the admin clicks on the respective card with text : Staff.
4. After clicking the card, the list of staffers is displayed in a tabular form.
5. The admin clicks on the bottom-right plus button
6. Upon button click, a modal form is shown
7. After clicking on certain labels or profile photo, the content is made editable where applicable.
8. After writing on the proper text fields, the admin clicks submit.
9. The form is submitted
   1. In case of error, an error message is displayed at the top of the modal
   2. In case of success, the modal is closed and the information is displayed in the table in a new row.

Scenario A9: Admin views and manages existing appointments

1. Admin is already logged in
2. He/she navigates to the Home button unless the button is selected (highlighted)
3. When the home info is fully rendered within the current page, the admin clicks on the respective card with text : Appointments
4. After clicking the card, the list of appointments is displayed in tabular form, with a text field showing the date of the current day.
5. Clicking on the date field, a calendar is shown and the admin chooses the desired date
6. After changing the date, the appointments list is refreshed
7. The admin clicks on any appointment row.
8. After clicking on a specific row, a modal is displayed containing an edit form
9. After editing the proper text fields, the user clicks Save changes on the modal
10. The form is submitted
    1. In case of error, an error message is displayed at the top of the modal
    2. In case of success, the modal is closed and the information is updated in the table.

Scenario A10: Admin confirms requested appointments

1. Admin is already logged in
2. He/she navigates to the Home button unless the button is selected (highlighted)
3. When the home info is fully rendered within the current page, the admin clicks on the respective card with text : Appointments
4. The admin clicks the Requested button
5. After clicking the card, the list of appointment requests is displayed in a tabular form.
6. After clicking on a specific row, a modal form is displayed,
7. After writing on the proper text fields(where applicable) and selecting a date in the date field (calendar) , the admin clicks Confirm.
8. The form is submitted
   1. In case of error, an error message is displayed at the top of the modal
   2. In case of success, the modal is closed and the information is displayed in the table in a new row

Scenario A11: Admin adds new appointment for new client ( not requested through app )

1. Admin is already logged in
2. He/she navigates to the Home button unless the button is selected (highlighted)
3. When the home info is fully rendered within the current page, the admin clicks on the respective card with text : Appointments
4. After clicking the card, the list of appointments is displayed in a tabular form.
5. The admin clicks the New button
6. A modal form is displayed with
7. After writing on the proper text fields and selecting a date in the date field (calendar) , the admin clicks submit.
8. The form is submitted
   1. In case of error, an error message is displayed at the top of the modal
   2. In case of success, the modal is closed and the information is displayed in the table in a new row

Scenario A12: Admin views and manages payrolls for staffers

1. Admin is already logged in
2. He/she navigates to the Home button unless the button is selected (highlighted)
3. When the home info is fully rendered within the current page, the admin clicks on the respective card with text : Payrolls
4. After clicking the card, the list of staffers with their payroll information is displayed in tabular form.
5. The admin clicks on any appointment row.
6. After clicking on a specific row, a modal is displayed containing an edit form
7. After editing the proper text fields, the user clicks Save changes on the modal
8. The form is submitted
   1. In case of error, an error message is displayed at the top of the modal
   2. In case of success, the modal is closed and the information is updated in the table.

Scenario A13: Admin views Logistics categories

1. Admin is already logged in
2. He/she navigates to the Home button unless the button is selected (highlighted)
3. When the home info is fully rendered within the current page, the admin clicks on the respective card with text : Payrolls
4. After clicking the card, a new section is rendered within the same page, showing several info cards based on the type of logistics
5. The user selects any type of logistic he/she wants to manage

Scenario A14: Admin manages a specific logistics’ category

1. Admin is already in the Logistics section following the A13 scenario
2. He/she navigates to the specific logistics’ category by clicking on the proper info card
3. After clicking the card, a new section is rendered within the same page, showing the specific logistic info in a tabular form.
4. The user selects any row that he/she wants to edit
5. After clicking on a specific row, a modal is displayed containing an edit form
6. After editing the proper text fields, the user clicks Save changes on the modal
7. The form is submitted
   1. In case of error, an error message is displayed at the top of the modal
   2. In case of success, the modal is closed and the information is updated in the table.
   3. **Use cases**

|  |  |
| --- | --- |
| **Name** | Admin login (Web) |
| **Summary** | Admin chooses an authentication method to login in the system |
| **Actor** | Admin |
| **Description** | The admin opens the login page, chooses an authentication method such as Google Sign-in or Facebook Sign-in. Upon successful authentication the admin is directed to the Admin Panel |
| **Precondition** | The user who wants to login must have an Google or Facebook account |
| **Alternatives** | There are no alternative options |
| **Post Condition** | Gain access to the Admin Panel |

|  |  |
| --- | --- |
| **Name** | Admin manages their profile (Web) |
| **Summary** | Admin goes to My Profile and can edit their ata |
| **Actor** | Admin |
| **Description** | The admin opens My Profile. All text fields are editable in case of a change. After updating admin saves the new data |
| **Precondition** | Admin must be logged in |
| **Alternatives** | In case of an error, a message is displayed above the edit form |
| **Post Condition** | Admin views and edits their personal data |

|  |  |
| --- | --- |
| **Name** | Admin views information (Web) |
| **Summary** | Admin can view clients and staff personal information |
| **Actor** | Admin, Client, Staff member |
| **Description** | The admin is able to go to the list of clients and staff members. After clicking the respective client/staff member a new page with their personal information is displayed |
| **Precondition** | Admin must be logged in |
| **Alternatives** | There are no alternative options |
| **Post Condition** | Admin has general knowledge about the clinics clients and staff members |

|  |  |
| --- | --- |
| **Name** | Admin manages logistics and payroll(Web) |
| **Summary** | Admin can manage the clinics logistics and staff members payroll |
| **Actor** | Admin, Staff member |
| **Description** | The admin is able edit the clinics logistics (materials and equipments). The admin should manage the staff members payroll (working hours and wages) |
| **Precondition** | Admin must be logged in and must have information about clinics logistic and staff members working hours and static wages |
| **Alternatives** | There are no alternative options |
| **Post Condition** | Admin is aware about the clinics materials and equipments and staff members payroll |

|  |  |
| --- | --- |
| **Name** | User / Doctor signs in (mobile app) |
| **Summary** | User or doctor authenticates in the application. |
| **Actor** | The User (Client), Doctor. |
| **Description** | The User uses one of the available authentication methods. If it is the first time, it also adds the extra required information. |
| **Precondition** | The user who wants to login must have a Google Account. |
| **Alternatives** | The User can sign in using an active phone number. |
| **Post Condition** | Have an active account in the dental clinic database. Can access personal files and set appointments. |

|  |  |
| --- | --- |
| **Name** | User sets new appointment from the app |
| **Summary** | The user can request a new appointments on a specific day, for a specific service. |
| **Actor** | The User (Client), The administrator |
| **Description** | The User, upon sign in, is eligible to request a new appointment electronically from the app. |
| **Precondition** | The user should be signed in. |
| **Alternatives** | Calling the clinic, and having them do it for you. |
| **Post Condition** | The user is notified when the appointment is scheduled. He has the option to accept it or decline it, if the assigned time slot is not suitable. |

|  |  |
| --- | --- |
| **Name** | User accesses details for a previous appointment |
| **Summary** | User accesses all descriptions, recommendations and photos, the doctor has uploaded from the appointment. |
| **Actor** | User (Client), Doctor |
| **Description** | In “My Profile”, “Treatments”, open a specific treatment, than a specific appointment. |
| **Precondition** | The user has undergone at least one appointment. |
| **Alternatives** | The administrator does it from his account. |
| **Post Condition** | The user can share the photos or descriptions. |

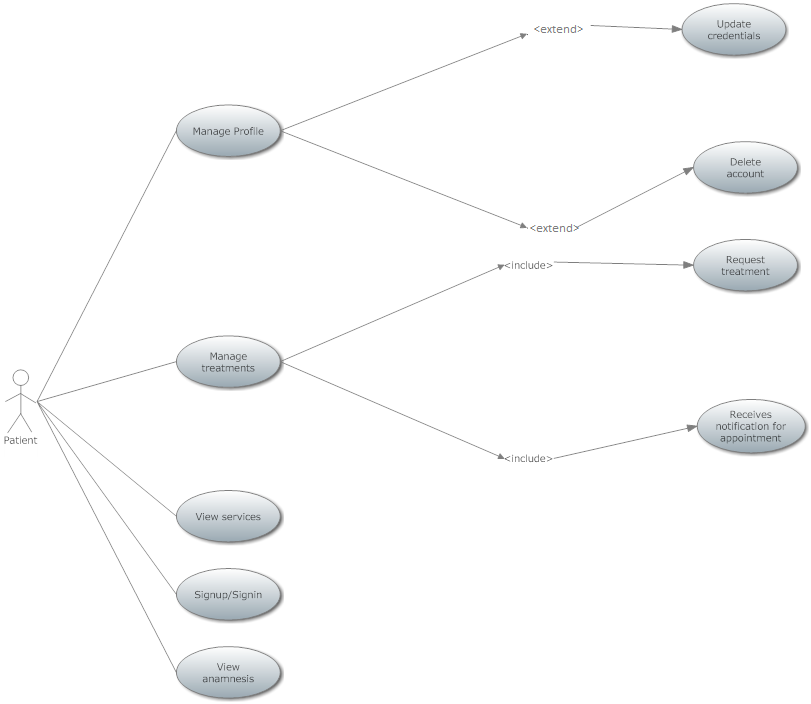
|  |  |
| --- | --- |
| **Name** | User views their anamnesis information |
| **Summary** | User accesses their medical history information and are able to download every related information as a file at any time |
| **Actor** | User (Client), Doctor |
| **Description** | In “My Profile”, “Anamnesis”, a list in the form of a timeline containing the information is presented |
| **Precondition** | The user should have provided the information beforehand. |
| **Alternatives** | The user requests this information verbally from the assigned doctor. |
| **Post Condition** | The doctor can rely on this sensitive information in their work. |

|  |  |
| --- | --- |
| **Name** | The admin manages the requested appointments |
| **Summary** | Accept, Decline, specify time-slot for a requested appointment. |
| **Actor** | User, Administrator |
| **Description** | The admin gets a list of all requested appointments from all user accounts. |
| **Precondition** | The signed in users have requested an appointment from the app. |
| **Alternatives** | The admin sets up an appointment manually for a not signed in user. |
| **Post Condition** | The appointment is added in the timetable, and presented to the doctor’s appointment list for the day. |

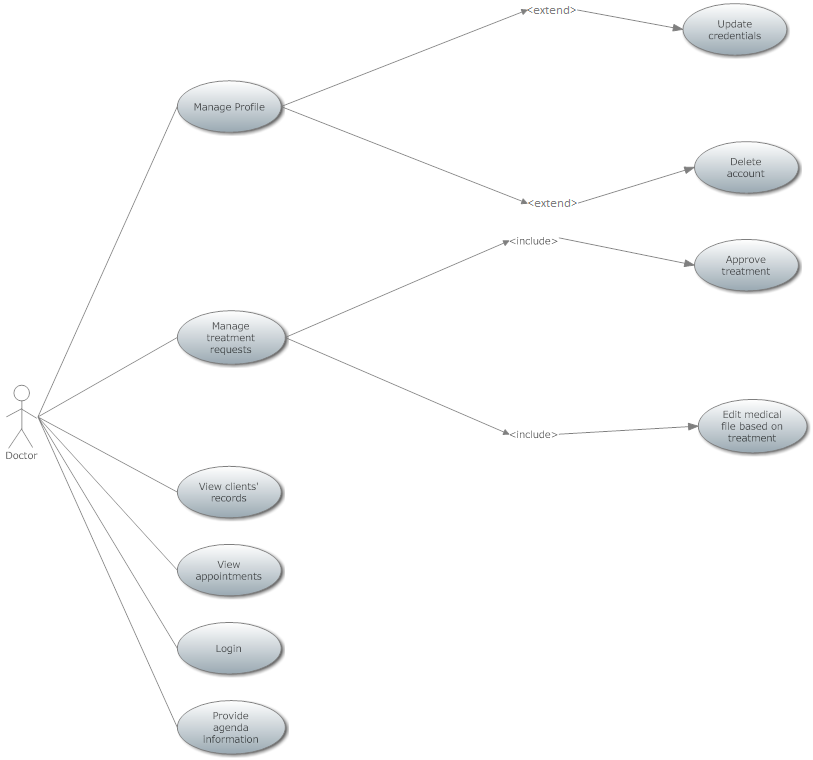
|  |  |
| --- | --- |
| **Name** | The doctor provides their agenda information |
| **Summary** | The doctor should provide their information for their current agenda to the system. |
| **Actor** | Doctor, Administrator |
| **Description** | The doctor can access the My Availability tab in the navigation bar, and then provides the info for the respective days. |
| **Precondition** | The doctor has already been registered in the system. The doctor has already logged in. |
| **Alternatives** | No alternative |
| **Post Condition** | The information is placed in the system and is used in the overall business logic. |

|  |  |
| --- | --- |
| **Name** | The doctor views his appointments |
| **Summary** | The doctor should access the appointments list for the current day and forthcoming days. |
| **Actor** | Doctor, Administrator |
| **Description** | The doctor can access the My Appointments tab in the navigation bar, and then views the proper information based on the day. |
| **Precondition** | The doctor has already been registered in the system. The doctor has already logged in. |
| **Alternatives** | No alternative |
| **Post Condition** | None |

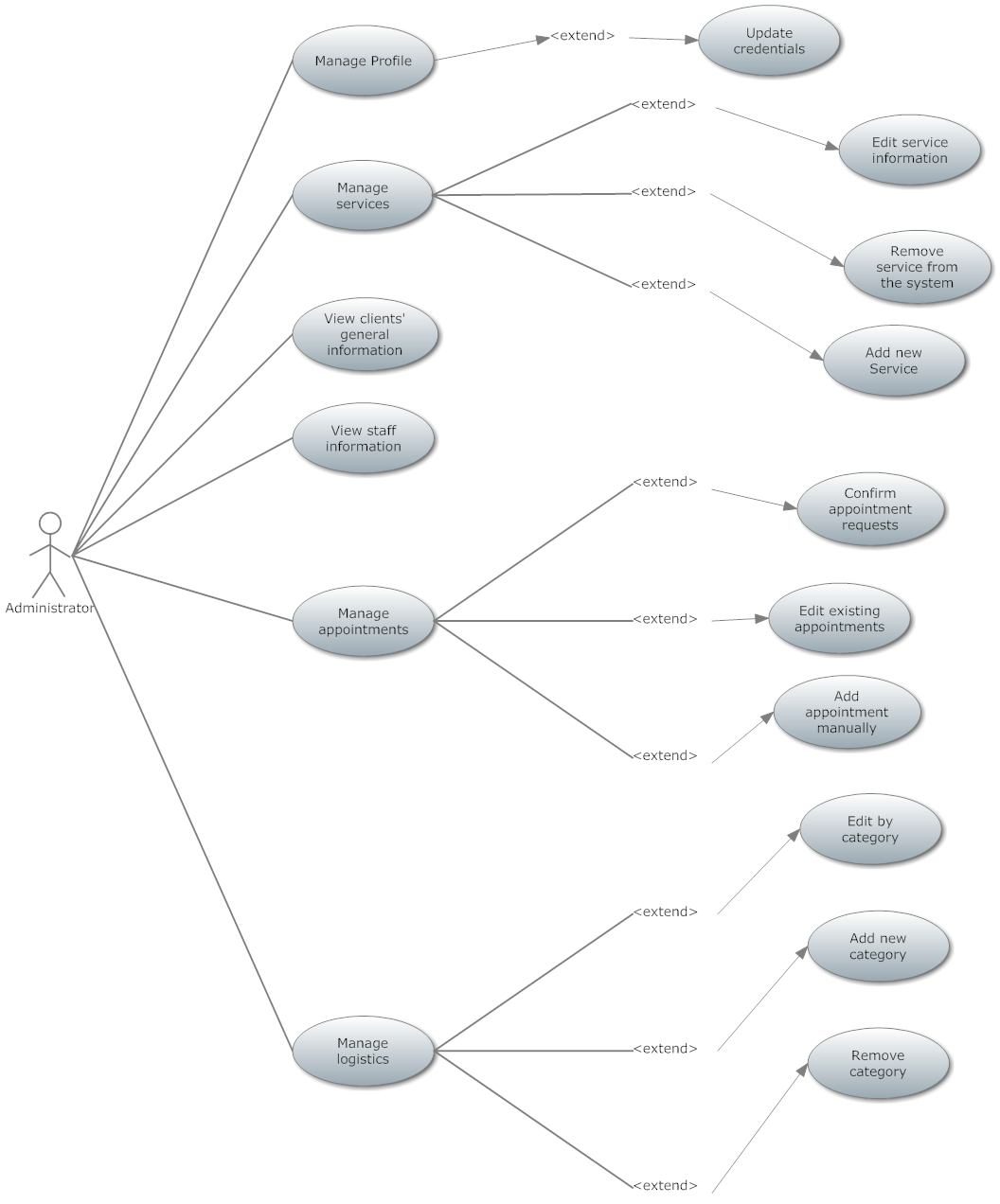
.

* 1. **Use case diagrams**
     1. **Client Use cases**

* + 1. **Doctor Use cases**

****

* + 1. **Admin Use Cases**



* + 1. **General Use Cases**

